processes

MDPI

Article

Chaos-Enhanced Archimede Algorithm for Global Optimization of Real-World Engineering Problems and Signal Feature Extraction

Ahmed Bencherqui [1], Mohamed Amine Tahiri [1], Hicham Karmouni [1], Mohammed Alfidi [1], Youssef El Afou [2], Hassan Qjidaa [3] and Mhamed Sayyouri [1,*]

[1] Laboratory of Engineering, Systems, and Applications, National School of Applied Sciences, Sidi Mohamed Ben Abdellah-Fez University, Fez 30040, Morocco; ahmed.bencherqui@usmba.ac.ma (A.B.); mohamedamine.tahiri@usmba.ac.ma (M.A.T.); hicham.karmouni@usmba.ac.ma (H.K.); alfidi_mohammed@yahoo.fr (M.A.)

[2] National School of Applied Sciences, Sidi Mohamed Ben Abdellah-Fez University, Fez 30040, Morocco; Youssef.elafou@usmba.ac.ma

[3] Laboratory of Electronic Signals and Systems of Information, Faculty of Science, Sidi Mohamed Ben Abdellah-Fez University, Fez 30040, Morocco; qjidah@yahoo.fr

* Correspondence: mhamed.sayyouri@usmba.ac.ma; Tel.: +212-662224655

Abstract: Optimization algorithms play a crucial role in a wide range of fields, from designing complex systems to solving mathematical and engineering problems. However, these algorithms frequently face major challenges, such as convergence to local optima, which limits their ability to find global, optimal solutions. To overcome these challenges, it has become imperative to explore more efficient approaches by incorporating chaotic maps within these original algorithms. Incorporating chaotic variables into the search process offers notable advantages, including the ability to avoid local minima, diversify the search, and accelerate convergence toward optimal solutions. In this study, we propose an improved Archimedean optimization algorithm called Chaotic_AO (CAO), based on the use of ten distinct chaotic maps to replace pseudorandom sequences in the three essential components of the classical Archimedean optimization algorithm: initialization, density and volume update, and position update. This improvement aims to achieve a more appropriate balance between the exploitation and exploration phases, offering a greater likelihood of discovering global solutions. CAO performance was extensively validated through the exploration of three distinct groups of problems. The first group, made up of twenty-three benchmark functions, served as an initial reference. Group 2 comprises three crucial engineering problems: the design of a welded beam, the modeling of a spring subjected to tension/compression stresses, and the planning of pressurized tanks. Finally, the third group of problems is dedicated to evaluating the efficiency of the CAO algorithm in the field of signal reconstruction, as well as 2D and 3D medical images. The results obtained from these in-depth tests revealed the efficiency and reliability of the CAO algorithm in terms of convergence speeds, and outstanding solution quality in most of the cases studied.

Keywords: Archimedean optimization algorithm; chaotic maps; optimization engineering problems; artificial intelligence

check for updates

Citation: Bencherqui, A.; Tahiri, M.A.; Karmouni, H.; Alfidi, M.; El Afou, Y.; Qjidaa, H.; Sayyouri, M. Chaos-Enhanced Archimede Algorithm for Global Optimization of Real-World Engineering Problems and Signal Feature Extraction. *Processes* **2024**, *12*, 406. https://doi.org/10.3390/pr12020406

Academic Editors: Weizhong Dai and Olympia Roeva

Received: 23 November 2023
Revised: 10 January 2024
Accepted: 14 February 2024
Published: 18 February 2024

1. Introduction

The exploration of problems characterized by high nonlinearities and multiplicities of local optima is a major concern in the fields of computer science, artificial intelligence, and machine learning. Traditional methods often face insurmountable challenges in regard to solving complex, multimodal problems. In such cases, nature-inspired approaches stand out by exploiting their ability to combine diversification and stochastic intensification to achieve optimal solutions in these complex contexts [1,2]. In recent decades, a considerable range of metaheuristic optimization algorithms has emerged as valuable alternatives for tackling these problems. These approaches have proven their effectiveness in finding solutions to demanding computational challenges, and they continue to gain in importance as the practical applications of artificial intelligence and machine learning multiply [3,4].

Two key features of metaheuristic methods are intensification and diversification, which play a crucial role in the search for optimal solutions [5]. Intensification targets the currently most promising solutions and selects the best-performing candidates, while diversification enables the optimizer to explore the search space more efficiently, largely through the use of randomization.

The ongoing evolution of the field of global optimization has seen the emergence of several new metaheuristic algorithms, each bringing its own innovations for improving computational efficiency, solving large-scale problems, and implementing robust optimization codes. Among these newcomers, ABC (Artificial Bee Colony) has adopted an approach that simulates the behavior of bees in their search for nectar [6]. Cuckoo search (CS) was inspired by the brooding behavior of certain birds to explore the search space [7]. HS (Harmony Search) introduced a notion of harmonious search inspired by music [8]. The BA (Bat Algorithm) exploits bat behavior to optimize functions [9]. Other algorithms, such as Genetic Algorithm (GA) [10–12], Chicken Swarm Optimization (CSO), Gray Wolf Optimizer (GWO), Firefly Algorithm (FA), Whale Optimization Algorithm (WOA), and Antlion Optimizer (ALO), also saw the light of day, each bringing their own perspective to solving complex problems [13–18].

This diversity of approaches testifies to the richness and adaptability of metaheuristic algorithms in the search for solutions to a variety of domains and problems. It also reflects the scientific community's ongoing commitment to exploring new avenues for pushing back the boundaries of optimization.

Recently, a new population-based metaheuristic optimization algorithm called Archimedes' Optimization Algorithm (AO) was proposed [19], inspired by Archimedes' Law from the field of physics. The operating principle of Archimedes' optimization algorithm (AO) is inspired by Archimedes' famous law governing the buoyancy of bodies immersed in a fluid. This fundamental law establishes a balance between the buoyant force exerted by the fluid on an immersed object and the weight of that object. When we apply this principle to AOA, the entities in the population are analogous to objects immersed in the fluid. Each of these entities is characterized by a series of parameters, including acceleration, density, and volume, which are key elements in determining whether an entity will float or sink in the algorithm's search space. The upward force, in the context of AO, represents the ability of an entity to emerge as an optimal solution. It is equivalent to the buoyancy of the entity, which is determined by the relationship between the weight of the object (representing the quality of the solution) and the weight of the displaced fluid (which can be interpreted as the potential of the solution). If the fluid displaced (the potential of the solution) is less than the weight of the object (the quality of the solution), the entity (the solution) will tend to 'sink', in the sense that it will not be retained as the optimal solution. On the other hand, when the weight of the object (the quality of the solution) is equal to the weight of the fluid displaced (the potential of the solution), the floating entity (solution) is in equilibrium. Herein lies the central conception of AO: finding the equilibrium point where a retained solution is in equilibrium, and where the net force of the fluid is zero. Although the AO algorithm is effective for solving complex problems, it cannot always overcome the pitfalls of the local optimum.

The phenomenon of chaos, characterized by its extreme sensitivity to initial conditions, has long intrigued researchers and become a key component in the field of optimization. Chaotic functions, often described as complex and unpredictable, vary irregularly over time, and are particularly sensitive to these initial conditions [20–23]. This sensitivity to initial values is an essential indicator of chaotic functions, as even a small, seemingly insignificant change in these conditions can lead to significant variations that cannot be neglected. It is in this context that metaheuristic algorithms have explored the potential offered by chaotic functions to improve their performance. Chaotic functions have proven particularly effective in the search for solutions to problems of local or premature convergence. Chaotic optimization methods have thus been adopted in many recent studies to break the vicious cycle of optimization. In addition, these chaotic functions have added value to global optimization algorithms by diversifying the search space. Chaotic theory has significant potential for improving the performance of metaheuristic algorithms.

The integration of chaos-based enhancement within various global optimization algorithms offers significant advantages. Consider, for example, its application within PSO, where it has the potential to deliver substantial improvements. In the context of PSO, chaos-based enhancement offers advantages in terms of increased diversification and exploration of the search space [24]. The chaotic properties of this approach play an essential role in preventing premature convergence to local optima. By promoting particle diversity, it enables a more exhaustive and dynamic exploration of the optimization landscape. Similarly, the introduction of chaos-based improvement within Genetic Algorithms (GA) opens up promising prospects [25]. It offers an opportunity to enhance genetic diversity within populations, thus promoting more efficient exploration of the search space. This diversification contributes to faster convergence towards optimal solutions by widening the field of exploration of potential solutions. The application of chaos in ACO (Ant Colony Optimization) mechanisms can significantly enhance the ability of ants to discover optimal paths [26]. By introducing chaotic exploration, this approach stimulates the search for alternative solutions, thus optimizing convergence towards quality solutions within the ant network. In the case of Artificial Bee Algorithms (ABC), the use of chaos-based enhancement can promote a more dynamic exploration of the search space [27]. This dynamic approach overcomes the limitations of traditional methods, stimulating further diversification of the solutions explored by the artificial bee colony.

This encouraged us to use chaotic maps to further optimize the Archimedean Optimization Algorithm (AO). In this study, we introduce a new chaos-enhanced Archimedean optimization algorithm (CAO). Our approach consists of using chaotic map functions to influence the generation of the random parameters of the AO algorithm. This integration allows us to scan the search space more dynamically and to refine various aspects of the algorithm. Thanks to this synergy, we have been able to achieve optimal fitness function values with increased success. To achieve this, we incorporated ten distinct chaotic maps to replace pseudorandom sequences in three AO components.

To evaluate the performance of the chaos-enhanced Archimedes optimization (CAO) algorithm, we have undertaken exhaustive testing using three distinct problem groups, each representing a set of specific challenges. The aim of this rigorous evaluation is to demonstrate the robustness and effectiveness of the CAO algorithm. The first group of problems consists of twenty-three benchmark functions (unimodal, multimodal, and multimodal with fixed dimension). Group 2 comprises three engineering problems, namely, the design of a welded beam, the design of a spring subjected to tension/compression stresses, and the design of a pressure vessel. These practical problems test the CAO algorithm's ability to solve complex engineering optimization challenges. This evaluation will determine the applicability of the CAO algorithm to real-world problems. The third group of problems is dedicated to validating the CAO algorithm in the field of 2D and 3D medical image and signal reconstruction. We use discrete orthogonal Meixner moments (MMs) for this purpose. The aim of this series of tests is to demonstrate the CAO algorithm's ability to solve image and signal processing problems, with an emphasis on accuracy and efficiency. To establish a meaningful comparison, we compare the performance of the CAO algorithm with that of other optimization algorithms in the literature.

The main contributions of this work are summarized as follows:

(a) With the aim of achieving an optimal balance between exploitation and exploration for the Archimedean optimization (AO) algorithm, we introduce a new Archimedean optimization algorithm enhanced by chaotic maps (CAO).
(b) To assess the effectiveness of the CAO method, we conducted extensive experiments using a set of twenty-three well-known numerical reference functions. In addition,

the method was successfully applied to three real-world engineering problems, confirming its relevance in real-world contexts.

(c) As an innovative application, CAO is used to optimize the selection of Meixner polynomial parameters, contributing to optimal reconstruction of medical signals and images. This application demonstrates the versatility of the CAO method for solving a variety of problems, from numerical optimization to medical image reconstruction.

The structure of the rest of this article is as follows: Section 3 looks at the presentation of the mathematical models underlying the standard Archimedes optimization algorithm. Section 4 is devoted to an examination of chaotic maps, while Section 5 explains in detail the Archimedes chaotic algorithm we have developed. Section 6 is dedicated to evaluating the performance of the proposed algorithm through a series of in-depth tests. Section 7 concludes our study.

2. Problem Formulation

In optimization problems, the task is often to find optimal solutions in a defined decision space. This formulation of the problem encompasses the effort to minimize a specific objective function over a set of admissible solutions, where the limits of each coordinate are precisely specified. Such a structured approach enables systematic exploration of the decision space, ensuring that the solutions sought respect the prescribed constraints.

- Objective function

$$\min F(x) = f(x_1, x_2, \ldots, x_n)$$

- Decision variables

$$X = (x_1, x_2, \ldots, x_n) \in R^n$$

- Bounds for each coordinate

$$ub_i \leq x_i \leq lb_i \, for \ i = 1, 2, \ldots, n$$

where ub_i and lb_i are the specified lower and upper bounds for each coordinate x_i, and n is the number of variables.

This complete formulation encapsulates the optimization problem, clearly defining the objective, decision variables, coordinate constraints, and decision space. It thus establishes a formal basis for analyzing and solving the optimization problem.

3. The Standard Archimedes Optimization Algorithm

The Archimedean Optimization Algorithm (AO) is a physics-inspired metaheuristic optimization algorithm that has emerged recently, and its operating principle is inspired by Archimedes' law of physics [28]. AO is a population-based algorithm where the individuals in the population are immersed objectives. AO starts a review process using an initial population of individuals endowed with random accelerations, densities, and volumes. The evaluation process begins with the suitability of this preliminary population and then continues through iterations until the end condition is reached. During each iteration, the density and volume of everyone are updated. Based on its interaction with other neighboring individuals, the acceleration of everyone is revised. The updated volume, density, and acceleration values determine the new position of everyone. AO has the distinct advantage of achieving a harmonious balance between exploration and exploitation, making it well suited to engineering optimization problems [29,30].

The main steps of the Archimedean Optimization (AO) algorithm are described below:

Step 1: Initialize the positions ($X(i)$), volume ($V_t(i)$), density ($D_t(i)$), and acceleration (acc_i) of all objects.

$$X(i) = lb_i + rand \times (ub_i - lb_i) \tag{1}$$

$$V_t(i) = rand \tag{2}$$

$$D_t(i) = rand \tag{3}$$

$$acc_i = lb_i + rand \times (ub_i - lb_i) \tag{4}$$

where lb_i and ub_i are the lower and upper bounds of the search-space, respectively.

Step 2: Update the densities ($D_t(i)$) and volumes ($V_t(i)$) of an object i using the following equation:

$$\begin{aligned} D_{t+1}(i) &= D_t(i) + rand_1 \times (D_{best} - D_t(i)) \\ V_{t+1}(i) &= V_t(i) + rand_2 \times (V_{best} - V_t(i)) \end{aligned} \tag{5}$$

where V_{best} and D_{best} are the best volume and best density, respectively, thus far.

Step 3: Calculation of the density factor and transfer operator.

When two objects first collide, they attempt to attain a stable state after some time has passed, and TF is used in the AO to do this. The shift from exploration to exploitation in the search process is accomplished by utilizing the following formula:

$$TF = \exp\left(\frac{t - t_{\max}}{t_{\max}}\right) \tag{6}$$

Similarly, the density factor d_{t+1} decreases over time, allowing one to concentrate in a favorable area.

$$d_{t+1} = \exp\left(\frac{t_{\max} - t}{t_{\max}}\right) - \left(\frac{t}{t_{\max}}\right) \tag{7}$$

where t represents the iteration number and $t_{\max}$ denotes the maximum iterations.

Step 4: Exploration phase

If TF ≤ 0.5 (objects are colliding), for $t + 1$, the update of the object's acceleration is performed using Equation (8).

$$acc_{t+1}(i) = \frac{D_{mr} + V_{mr} \times acc_{mr}}{D_{t+1}(i) \times V_{t+1}(i)} \tag{8}$$

where D_{mr} and V_{mr} denote the density and volume of the random material, respectively.

Step 5: Exploitation phase

If TF > 0.5 (objects are not colliding), for $t + 1$, the update of the object's acceleration is performed using Equation (9).

$$acc_{t+1}(i) = \frac{D_{best} + V_{best} \times acc_{best}}{D_{t+1}(i) \times V_{t+1}(i)} \tag{9}$$

Step 6: Normalize the acceleration.

The normalized acceleration is calculated using:

$$acc_{t+1}(i)_{norm} = u\frac{acc_{t+1}(i) - \min(acc)}{\max(acc) - \min(acc)} + l \tag{10}$$

where $l = 0.1$ and $u = 0.9$ are the normalization ranges, and $\min(acc)$ is the minimum acceleration value while $\max(acc)$ is the maximum acceleration value.

Step 7: Update the position.

The calculation of the object's position $t + 1$ is determined by the following equation:

$$X_{t+1}(i) = \begin{cases} X_t(i) + C1 \times rand_3 \times acc_{t+1}(i)_{norm} \times d \times (X_{rand} - X_t(i)), & TF \leq 0.5 \\ X_{best} + F \times C2 \times rand_4 \times acc_{t+1}(i)_{norm} \times d \times (T \times X_{best} - X_t(i)), & otherwise \end{cases} \tag{11}$$

T increases with time and is directly proportional to the transfer operator defined by:

$$T = C3 \times TF \tag{12}$$

F is the flag used to change the direction of movement using Equation (13).

$$F = \begin{cases} +1 \;\; if \;\; P \leq 0.5 \\ -1 \;\; if \;\; P > 0.5 \end{cases} \tag{13}$$

where $P = 2 \times rand - C4$.

Step 8: Evaluation

Select the object's position that exhibits the best fitness value upon evaluating each object.

The global optimization algorithms use random variables to explore the search space, commonly in the form of a uniform distribution (rand). However, this use of random variables frequently leads to local optima, thus limiting search efficiency. This is why we suggest using sequences generated by chaotic maps to determine the values of parameters, which, in the AO algorithm, were of a random nature. The integration of chaotic variables into the search process represents a significant advance over an entirely random approach. The advantages inherent in this approach are substantial, and chaotic maps, which we will detail in the next section, play a key role in the search for optimal solutions.

4. Chaotic Maps

In recent years, chaos theory has made significant advances and has been successfully exploited in a variety of scientific fields. Promising applications include image and signal encryption, feature selection, and parameter optimization. Chaotic maps, in particular, have proven highly useful, possessing three fundamental characteristics: sensitivity to initial conditions, randomness, and dynamics. These unique properties have enabled chaotic maps to be incorporated into several renowned optimization algorithms, including moth flame optimization (MFO), firefly algorithm (FA), artificial bee colony (ABC), biogeography-based optimization (BBO), particle swarm optimization (PSO), and gray wolf optimizer (GWO). This fusion of chaos theory with optimization algorithms has paved the way for more efficient solutions and better performance, both in solving complex problems and in improving engineering processes [31–33]. Chaotic maps have added an extra dimension to the search for solutions by introducing an element of dynamics and unpredictability, which has proved beneficial in escaping local minima and improving the quality of the solutions found.

In this subsection, we outline ten one-dimensional chaotic maps of particular relevance to optimization algorithms used to generate chaotic sequences, as detailed in Table 1 and illustrated in Figure 1. It is important to note that the initial point can be chosen arbitrarily, in the range 0 to 1 (or according to the specific scope of the chaotic map in question). The use of these dynamic values is of paramount importance, as it contributes significantly to improving the search capability of the AO.

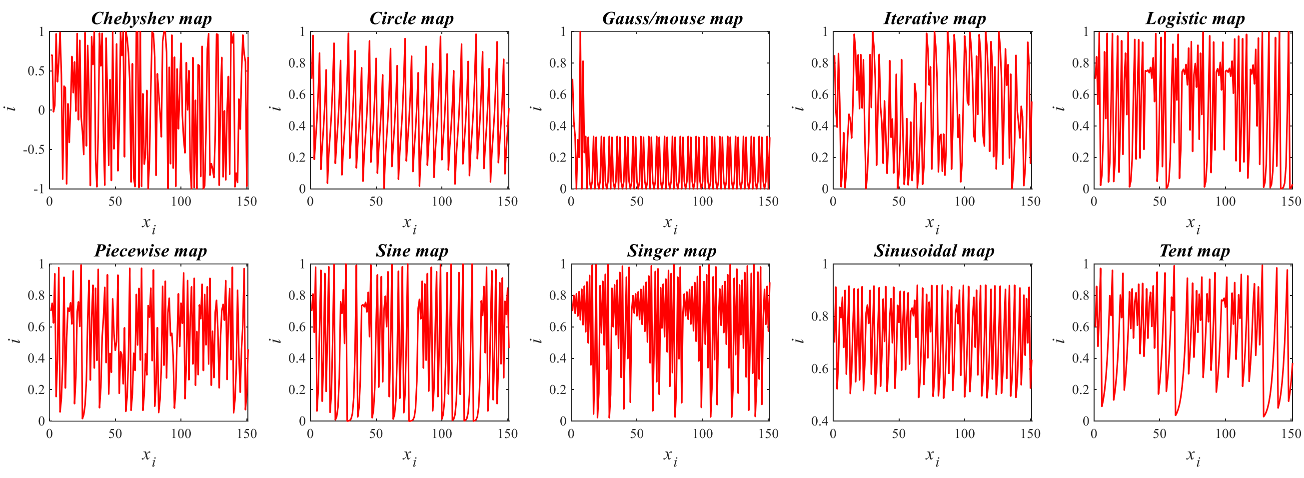


Figure 1. Chaotic map values.

Table 1. Chaotic maps.

Maps Name	Function	Range
Map of Chebyshev	$x_{k+1} = \cos(k\,cos^{-1}(x_k))$	$(-1, 1)$
Circular map	$x_{k+1} = x_k + b - (\frac{a}{2\pi})\sin(2\pi x_k)\mathrm{mod}(1)$	$(0, 1)$
Gauss map	$x_{k+1} = \begin{cases} 0 & x_k = 0 \\ \frac{1}{x_k}\ \mathrm{mod}(1) & otherwise \end{cases}$	$(0, 1)$
Iterative map	$x_{k+1} = \sin(\frac{a\pi}{x_k}), \quad a = 0.7$	$(-1, 1)$
Logistic map	$x_{k+1} = ax_k(1 - x_k), \quad a = 4$	$(0, 1)$
Piecewise map	$x_{k+1} = \begin{cases} \frac{x_k}{p} & 0 \le x_k \le p \\ \frac{x_k - p}{0.5 - p} & P \le x_k \le 1/2 \\ \frac{1-p-x_k}{0.5-p} & 1/2 \le x_k \le 1 - p \\ \frac{1-x_k}{p} & 1 - p \le x_k \le 1 \end{cases}$	$(0, 1)$
Sine map	$x_{k+1} = \frac{a}{4}\sin(\pi x_k), \quad a = 4$	$(0, 1)$
Singer map	$x_{k+1} = \mu(7.86x_k - 23.31x_k^2 + 28.75x_k^3 - 13.302875x_k^4), \quad \mu = 1.07$	$(0, 1)$
Sinusoidal map	$x_{k+1} = ax_k^2\sin(\pi x_k), \quad a = 2.3$	$(0, 1)$
Tent map	$x_{k+1} = \begin{cases} \frac{x_k}{0.7}, & x_k < 0.7 \\ \frac{10}{3}(1 - x_k), & x_k \ge 0.7 \end{cases}$	$(0, 1)$

The idea behind this method is based on three key principles: (i) Introduction of a chaotic state into the optimization variables using a similar support approach. (ii) Extension of the range of chaotic movements to encompass the interval of optimization variables. (iii) Use of chaotic sequences to enhance the efficiency of the search process.

By combining these principles, it becomes possible to inject an element of controlled chaos into the optimization variables, explore a wider range of potential solutions, and thus improve search efficiency in a well-controlled way. This innovative approach offers promising prospects for the optimization of complex problems, in particular by widening the range of solutions explored and enabling a more diversified and efficient search.

5. Proposed Chaotic-Archimede Optimization Algorithm (CAO)

In the classic configuration of the Archimedean optimization algorithm, an initial set of random solutions originates in the extent of the search space. This set of solutions is then subjected to a series of updating equations, each contributing to the distinct exploration and exploitation of the search space. The exploration equations are responsible for guiding the solutions to various regions of the search space, seeking to exhaustively explore the different possibilities. In contrast, exploitation equations steer solutions towards the best solution identified so far, while probing the surroundings of this optimal solution. However, it is crucial to note that these equations have a random component, meaning that solutions mutate at random intervals and in random directions This random nature of the equations underlines the sensitivity to random parameters, which exert a substantial influence on the quality of the solutions obtained and, by extension, on the final results of the algorithm. A judicious modification of these random parameters could therefore play a decisive role in optimizing the overall process, potentially leading to more robust and accurate solutions.

Although not yet mathematically proven, various studies converge towards the conclusion that the integration of chaotic maps significantly improves the performance of metaheuristic optimization algorithms, as developed by many researchers [34,35]. For example, Wang et al. in [34] demonstrated the performance improvement of the Remora optimization algorithm (ROA) based solely on chaotic tent mapping. Similarly, Wang et al. in [35] introduced the Levy operator to help the crystal structure algorithm (CryStAl) to effectively free itself from the attraction of the local optimal value.

However, in the context of this study, our aim is to take advantage of ten chaotic maps, thus opening up several perspectives for improving the performance of our proposed algorithm, specifically in terms of avoiding local optimum, rather than being limited to the use of a single chaotic map or operator.

This study presents the development of a new algorithm called the Chaos-Enhanced Archimedean Optimization (CAO) algorithm. The introduction of chaotic variables into the AO search process confers significant advantages over a purely random approach. More specifically, the ergodic properties of chaotic maps are exploited to increase search efficiency by bypassing local minima.

In the CAO, chaotic sequences are generated by ten distinct types of chaotic maps, namely Chebyshev, circular, Gaussian, iterative, logistic, patchy, sinusoidal, Singer, sinusoidal, and tent, as illustrated in Table 1. These sequences replace the random sequences of the original AO in three crucial components of the optimization algorithm, namely initialization, density and volume updating, and position updating. This integration enhances the AO algorithm's ability to bypass local optima, thus increasing the probability of converging to the global optimum in a limited number of iterations.

Each chaotic map is tested independently in each of these components. The pseudocode of the CAO algorithm is shown in Algorithm 1, while Figure 2 illustrates the CAO flowchart. In this algorithm, $cc(i)$ represents a chaotic sequence generated by a specific chaotic map. The results obtained in this study show that chaotic maps increase the performance of optimization methods.

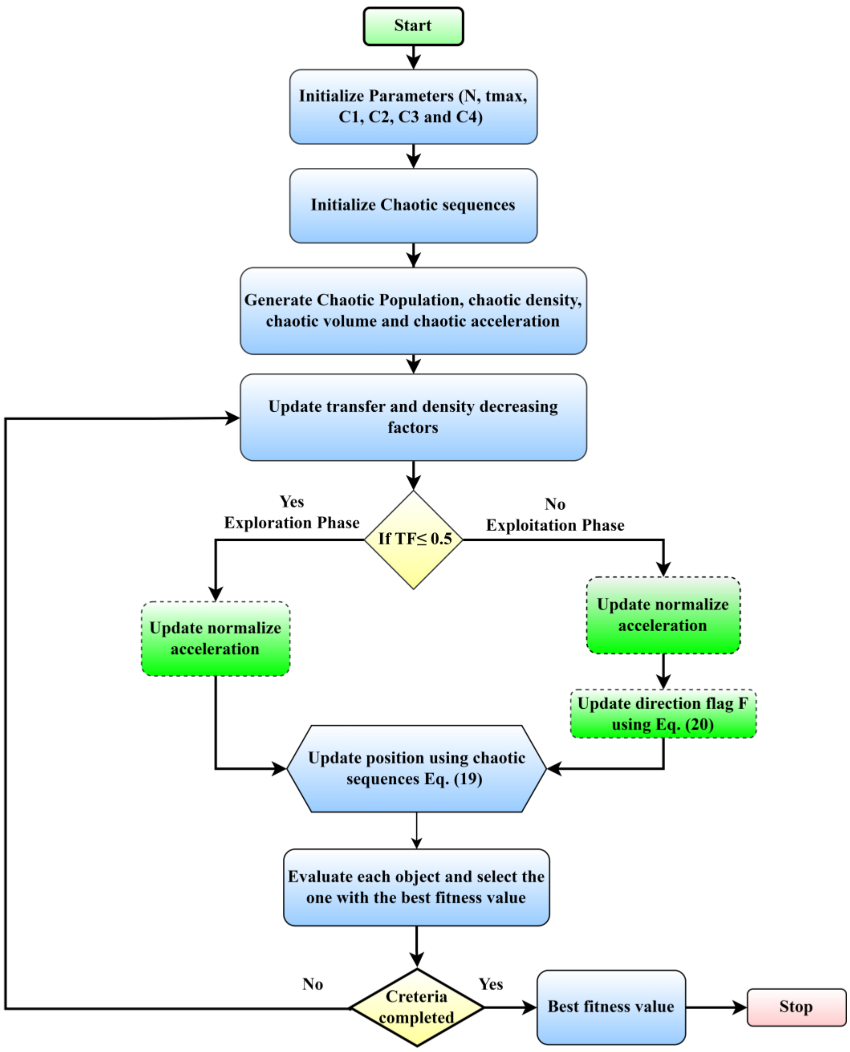


Figure 2. CAO algorithm flowchart.

Algorithm 1. pseudo code of CAO

Initialization (N, tmax, C1, C2, C3 and C4).
Initialize chaotic value $cc(i)$.
for i = 1:n
  for j = 1: n
  $cc(i) = chaotic(cc(i));$
  $X(i) = lb_i + cc(i) \times (ub_i - lb_i);$
  $V_t(i) = cc(i);$
  $D_t(i) = cc(i);$
  $acc(i) = lb_i + cc(i) \times (ub_i - lb_i);$
  end
end
Evaluate the initial population and select the one with the best fitness value.
t (iteration counter) = 1
While t < *tmax* ***do***
for *each object i* ***do***
  for j = 1: n
//Update density and volume of each object.
  $cc(i) = chaotic(cc(i));$
  $\begin{matrix} D_{t+1}(i) = D_t(i) + cc(i) \times (D_{best} - D_t(i)) \\ V_{t+1}(i) = V_t(i) + cc(i) \times (V_{best} - V_t(i)) \end{matrix} ;$
  end
//Update transfer and density decreasing factors TF and d_{t+1}.
  $TF = \exp\left(\frac{t - t_{\max}}{t_{\max}}\right); d_{t+1} = \exp\left(\frac{t_{\max} - t}{t_{\max}}\right) - \left(\frac{t}{t_{\max}}\right);$
  if $TF \leq 0.5$ *then* **---//Exploration phase**
    //Update acceleration and normalize acceleration.
    $acc_{t+1}(i)_{norm} = u\frac{acc_{t+1}(i) - \min(acc)}{\max(acc) - \min(acc)} + l;$
    for i = 1:n
    $cc(i) = chaotic(cc(i));$
    $X_{t+1}(i) = X_t(i) + C1 \times cc(i) \times acc_{t+1}(i)_{norm} \times d_{t+1} \times (X_{rand} - X_t(i));$
    end
  else *---//Exploitation phase*
    //Update acceleration and normalize acceleration.
    $acc_{t+1}(i)_{norm} = u\frac{acc_{t+1}(i) - \min(acc)}{\max(acc) - \min(acc)} + l;$
    for i = 1:n
    $cc(i) = chaotic(cc(i));$
    $P = 2 \times cc(i) - C4;$
    $X_{t+1}(i) = X_{best} + F \times C2 \times cc(i) \times acc_{t+1}(i)_{norm} \times d_{t+1} \times (T \times X_{best} - X_t(i));$
    end
  end
end while
Evaluate each object and select the one with the best fitness value.
$t = t + 1$
Return

(a) Utilization of chaotic maps in initializing the population

Since AO adopts the traditional initialization method, the randomness and diversity of the initial population cannot be guaranteed. In contrast to traditional initialization methods, the use of chaotic for initialization significantly preserves population diversity.

However, in our chaotic optimization algorithm (CAO), we make an innovative choice by using chaotic sequences $cc(i)$ instead of simple random generation *rand* (0, 1). These chaotic sequences are used to initialize the position, volume, density, and acceleration of all objects, according to the following formulas:

$$X(i) = lb_i + cc(i) \times (ub_i - lb_i) \tag{14}$$

$$V_t(i) = cc(i) \tag{15}$$

$$D_t(i) = cc(i) \tag{16}$$

$$acc(i) = lb_i + cc(i) \times (ub_i - lb_i) \tag{17}$$

In order to verify the rationality of this method, we compared the initial value generated by the chaotic map with that generated by traditional initialization, and the results are shown in Figure 3. As this figure shows, the initial value generated by the chaotic map is more extensive than that generated by the uniform random distribution, and it is not difficult to observe that the initial positions obtained on the basis of initialization by chaotic mapping are more uniformly distributed in the search space. Consequently, the former can improve the ergodicity of the initial value and accelerate the convergence speed of the algorithm.

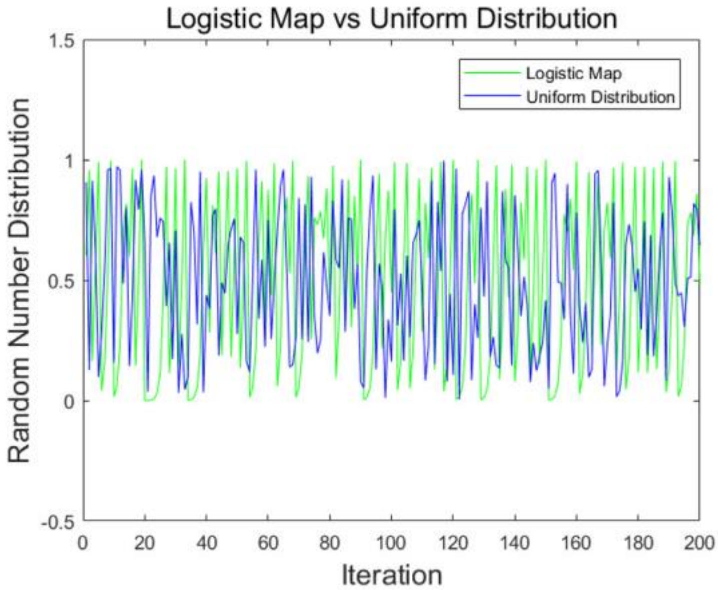


Figure 3. Initial value by utilizing both a chaotic map and a uniform random distribution.

(b) Chaotic maps applied to update density and volume

The density and volume of object *i* are updated at iteration *t + 1* according to Equation (5). In this equation, the parameters $rand_1$ and $rand_2$ are essential to the AO algorithm, and their values are usually randomly generated from the interval [0, 1]. However, a significant innovation in our approach is the use of chaotic maps to generate these parameters. Instead of using random values at each iteration to determine the optimal points, we adopt a more sophisticated method using chaotic maps to assign values to $rand_1$ and $rand_2$. This innovative approach promises substantial improvements in the search for optimal solutions. The density and volume of object *i* for iteration *t* + 1 are updated using the following equation:

$$\begin{aligned} D_{t+1}(i) &= D_t(i) + cc(i) \times (D_{best} - D_t(i)) \\ V_{t+1}(i) &= V_t(i) + cc(i) \times (V_{best} - V_t(i)) \end{aligned} \tag{18}$$

(c) Chaotic maps applied to update position

In the standard optimization algorithm (AO), the position of objects *i* is updated in accordance with Equation (11). It is important to note that the values $rand_3$ and rand_4, needed in this equation, are usually randomly generated in the interval [0, 1]. However, in our chaotic optimization algorithm (CAO), we introduce a crucial innovation: instead of relying on random values at each iteration, we use chaotic maps to determine these parameters. This innovative approach promises to open up new perspectives in improving algorithm efficiency and finding optimal solutions. The calculation of the new position of object i is determined by the following equation:

$$X_{t+1}(i) = \begin{cases} X_t(i) + C1 \times cc(i) \times acc_{t+1}(i)_{norm} \times d_{t+1} \times (X_{rand} - X_t(i)), & TF \leq 0.5 \\ X_{best} + F \times C2 \times cc(i) \times acc_{t+1}(i)_{norm} \times d_{t+1} \times (T \times X_{best} - X_t(i)), & otherwise \end{cases} \quad (19)$$

The variable X_{rand} in expression represents a random position associated with a random number generation process. This variable can take on different values at random.

F is the flag used to change the direction of movement using (Equation (20)).

$$F = \begin{cases} +1 \ \ if \ \ P \leq 0.5 \\ -1 \ \ if \ \ P > 0.5 \end{cases} \\ with\ P = 2 \times cc(i) - C4 \quad (20)$$

The following observations reinforce the demonstration of the theoretical effectiveness of the proposed chaotic algorithms:

As a first significant improvement, the chaos-enhanced Archimedean optimization algorithm determines the positions, volumes, densities, and accelerations of all objects by choosing the best solution from among those randomly generated.

Chaotic map integration supports CAO by orchestrating chaotic updates of density, volume, and position, substantially improving the exploration process.

When it comes to probing a promising area in the search space, these chaotic parameters play an essential role in fostering chaotic neighborhood exploitation.

6. Simulation Results

To evaluate the effectiveness of the proposed Chaos-Archimede Optimization Algorithm (CAO), we undertook tests on three distinct problem sets. The first group, comprising twenty-three benchmark functions (unimodal, multimodal, and multimodal with fixed dimension) identified in Table 2, served as the basis for our evaluation. The second group consists of three significant engineering problems: the design problem for a welded beam, the design problem for a tension/compression spring, and the design problem for a pressure vessel. These engineering problems enable us to test the effectiveness of our algorithms in practical contexts. Finally, the third group of tests highlights the versatility of the CAO approach. It is devoted to validating our CAO algorithm for signal reconstruction, as well as 2D and 3D medical images. In this series of tests, we used discrete orthogonal Meixner moments (MMs) as a key tool. These experiments cover a diverse range of application domains and demonstrate the adaptability and efficiency of our CAO algorithm in a variety of contexts.

Table 2. Unimodal, multimodal, and multimodal with fixed dimensions test functions.

	Functions	Descriptions	Dimensions	Range
Unimodal functions	F1	$f(x) = \sum_{i=1}^{n} x_i^2$	30, 100, 500, 1000	$[-100, 100]$
	F2	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	30, 100, 500, 1000	$[-10, 10]$
	F3	$f(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	30, 100, 500, 1000	$[-100, 100]$
	F4	$f(x) = \max_i \{ \lvert x_i \rvert, \ 1 \leq i \leq n \}$	30, 100, 500, 1000	$[-100, 100]$
	F5	$f(x) = \sum_{i=1}^{n-1} \left[100 \left(x_i^2 - x_{i+1} \right)^2 + (1 - x_i)^2 \right]$	30, 100, 500, 1000	$[-30, 30]$
	F6	$f(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30, 100, 500, 1000	$[-100, 100]$
	F7	$f(x) = \sum_{i=0}^{n} i x_i^4 + random[0, \ 1)$	30, 100, 500, 1000	$[-128, 128]$

Table 2. *Cont.*

	Functions	Descriptions	Dimensions	Range
Multimodal functions	F8	$f(x) = \sum_{i=1}^{n} (x_i \sin(\sqrt{\lvert x_i \rvert}))$	30, 100, 500, 1000	$[-500, 500]$
	F9	$f(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30, 100, 500, 1000	$[-5.12,\ 5.12]$
	F10	$f(x) = 20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30, 100, 500, 1000	$[-32, 32]$
	F11	$f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^n - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	30, 100, 500, 1000	$[-600, 600]$
	F12	$f(x) = \frac{\pi}{n}[10\sin(\pi y_1)] + \sum_{i=1}^{n}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1}) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)]$	30, 100, 500, 1000	$[-50, 50]$
	F13	$f(x) = 0.1\left(\begin{array}{c}\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + \\ (x_n - 1)^2(1 + \sin^2(2\pi x_n)) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)\end{array}\right)$	30, 100, 500, 1000	$[-50, 50]$
Multimodal functions with a fixed dimension	F14	$f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{i,j})}\right)^{-1}$	2	$[-65, 65]$
	F15	$f(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]$
	F16	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$
	F17	$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{4}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$
	F18	$f(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	$[-2, 2]$
	F19	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{i=1}^{3} a_{ij}(x_{ij} - p_{ij})^2\right)$	3	$[-1, 2]$
	F20	$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{i=1}^{6} a_{ij}(x_{ij} - p_{ij})^2\right)$	6	$[0, 1]$
	F21	$f(x) = -\sum_{i=1}^{5}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$
	F22	$f(x) = -\sum_{i=1}^{7}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$
	F23	$f(x) = -\sum_{i=1}^{10}\left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	$[0, 1]$

For the first group, made up of 23 test functions in Table 2, the aim is to determine the value of x that minimizes the function *f(x)* in each specific case. This solution search is formulated under the constraint *minf(x)*. It is important to note that the search range, the possible values for the components of x, varies according to the specific nature of each function. Each function may have distinct range requirements for the different variables, and this variability must be considered when searching for the optimal solution for each test function.

In the second group, the aim is to find optimal values for x that minimize the cost functions associated with the welded beam, tension/compression spring, and pressure vessel design problems. The search scope is adjusted accordingly to meet the specifics of each engineering problem [36].

In the third problem, we seek to optimize the polynomial Meixner parameters (β, u) by minimizing the objective function MSE (Mean Square Error), with the aim of obtaining optimal parameters that enable 1D, 2D, and 3D signals to be reconstructed with excellent quality. The dimension of this problem is equal to 2, as we are looking for optimal values of β and u. The search range for the parameters is defined as follows: The search range for β extends from 0 to N (the size of the signal), while the search range for u varies from 0 to 1. This choice of range reflects the specific conditions of the problem, and is intended to guarantee an exhaustive exploration of the possible values of β and u with the aim of obtaining optimal parameters for the reconstruction of multidimensional signals.

Chaos-Archimede Optimization Algorithms (CAO) take advantage of ten types of chaotic maps, as illustrated in Table 1. This diversity of chaotic maps has enabled us to sequentially develop ten variations of Chaos-Archimede optimization algorithms, which we have designated as follows: CAO1 (Chebyshev-AO), CAO2 (Circular-AO), CAO3 (Gauss-AO), CAO4 (Iterative-AO), CAO5 (Logistic-AO), CAO6 (Piecewise-AO), CAO7 (Sine-AO), CAO8 (Singer-AO), CAO9 (Sinusoidal-AO), and CAO10 (Tent-AO). The performance of these ten new algorithms was evaluated by comparing them with the standard AO algorithm. Notably, the comparative analysis shows that CAO10 stands out with the most remarkable performance of the set.

Continuing our exploration, we extended the comparison by including CAO10 in a set of original optimization algorithms, such as the whale optimization algorithm [36], the gray wolf optimizer [37], salp swarm algorithm [38], multiverse optimizer [39], gravitational search algorithm [40], sine cosine algorithm [41], particle swarm optimization [42], and moth-flame optimization [43]. The results obtained highlight the efficiency of CAO10, which competes with these original algorithms in a promising way, opening new perspectives for diverse optimization applications.

6.1. Reference Function Validation

(a) Extensibility test

To evaluate the performance of the Chaos-Archimede Optimization (CAO) algorithm, this section carries out an in-depth comparison between the results obtained by CAO and those of the original AO algorithm. Tests are specifically carried out on twenty-three benchmark functions (F1–F23), involving rigorous evaluations of the scalability of both algorithms.

The aim of scalability tests is to analyze the impact of dimensions on the efficiency of stochastic optimizers. They enable us to understand how problem dimensions affect the quality of the solutions generated, as well as the efficiency of the CAO when the dimension is dynamically increased. Three different dimensions are therefore examined in this study: 20, 30, and 50. To assess the performance of the algorithms, several performance measures are used. These include: (1) the mean and standard deviation of the final solutions obtained for each function. (2) Analysis of the convergence of the functions obtained to assess the speed with which the algorithms converge toward optimal solutions. (3) Ranking to identify the chaotic map that performs best among all available chaotic maps.

All optimization algorithms are subjected to the same experimental conditions, including an identical population size and a predefined maximum value set at 500. This rigorous methodological approach ensures a fair comparison of CAO versus AO performance, highlighting the potential advantages of using chaotic maps in the context of stochastic optimization.

(1) In statistics, the standard deviation is a fundamental measure used to quantify the amplitude of variation and dispersion within a data set. A standard deviation close to zero indicates that optimal solutions tend to be very close to the mean, reflecting a high concentration of results. On the other hand, a high standard deviation reflects a greater dispersion of optimal solutions over a wide range of values, suggesting greater variability. Tables 3–11 show the mean error and standard deviation of solutions obtained from experiments with the ten CAO optimization algorithms and the standard AO for dimensions of 20, 30, and 50, respectively. Analysis of these results reveals that CAO optimization algorithms outperform AO in all functions (F1-F23), regardless of the number of dimensions. Moreover, these CAO algorithms systematically display a clear superiority in the higher dimensions, underlining their ability to handle complex problems. Notably, the CAO10 algorithm stands out as offering exceptional accuracy compared to other CAO variants, reinforcing its status as the preferred choice for optimizing varied, multidimensional problems. These

results demonstrate that integrating chaotic maps into CAO optimization algorithms can significantly improve their efficiency, paving the way for more accurate solutions and greater convergence in complex optimization contexts.

(2) The convergence test represents an essential criterion for assessing the performance of algorithms in achieving the global optimum. Figures 4–6 illustrate the convergence curves of test functions using CAO optimization algorithms and standard AO for dimensions of 20, 30, and 50, respectively. As these figures show, all CAO algorithms demonstrate a remarkable ability to find optimal solutions to reference functions in all functions (F1–F23). These algorithms demonstrate reliability and stability, standing out for their ability to explore search spaces more thoroughly than the standard AO algorithm. Moreover, they converge on optimal solutions considerably faster than the standard algorithm. These observations show that the use of chaotic maps in optimization significantly improves algorithm performance, contributing to greater efficiency and a significant reduction in the time needed to reach optimal solutions.

(3) In ranking-based analysis, algorithms are evaluated and ranked according to their average performance. For this purpose, a standard ranking system is used to establish the competition between algorithms. Figure 7 show the ranking of algorithms according to their performance in 20, 30, and 50 dimensions. In this ranking system, the ranking value 1 indicates the best performance, while the ranking value 11 reflects the least favorable performance. Clearly, the ten chaotic maps outperform the original AO algorithm. In addition, the CAO10 algorithm stands out by achieving significantly higher rankings than the other CAO variants.

These results convincingly demonstrate that introducing chaotic maps into the optimization process confers a significant performance advantage over the standard Archimedean optimization algorithm. In particular, the CAO10 algorithm stands out as the preeminent choice, highlighting its exceptional efficiency and ability to obtain optimal solutions in multidimensional spaces. This ranking analysis underlines the importance of integrating chaotic variables into optimization for improved performance and a better ability to solve a variety of problems.

(b) Comparison test with other optimization algorithms

To verify the effectiveness of the proposed algorithm, CAO10 was compared with ten well-known algorithms: Whale Optimization Algorithm, Gray Wolf Optimizer, Salp Swarm Algorithm, Multiverse Optimizer, Gravitational Search Algorithm, Sine Cosine Algorithm, Particle Swarm Optimization, and Moth-Flame Optimization. The parameter values for the above algorithms have been set in accordance with their original papers. Table 12 shows the parameter values for each algorithm. In all experiments, the population size and maximum number of iterations were set to 30 and 500, respectively.

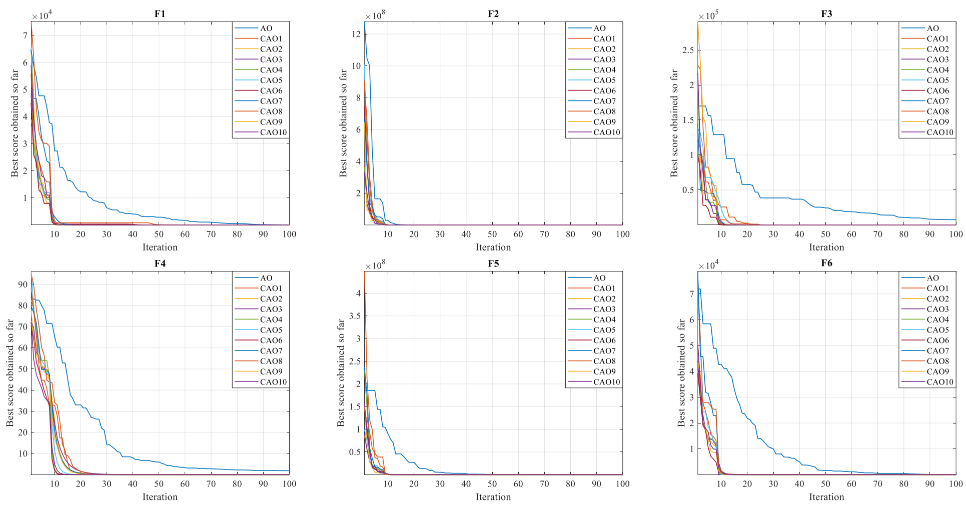


Figure 4. *Cont.*

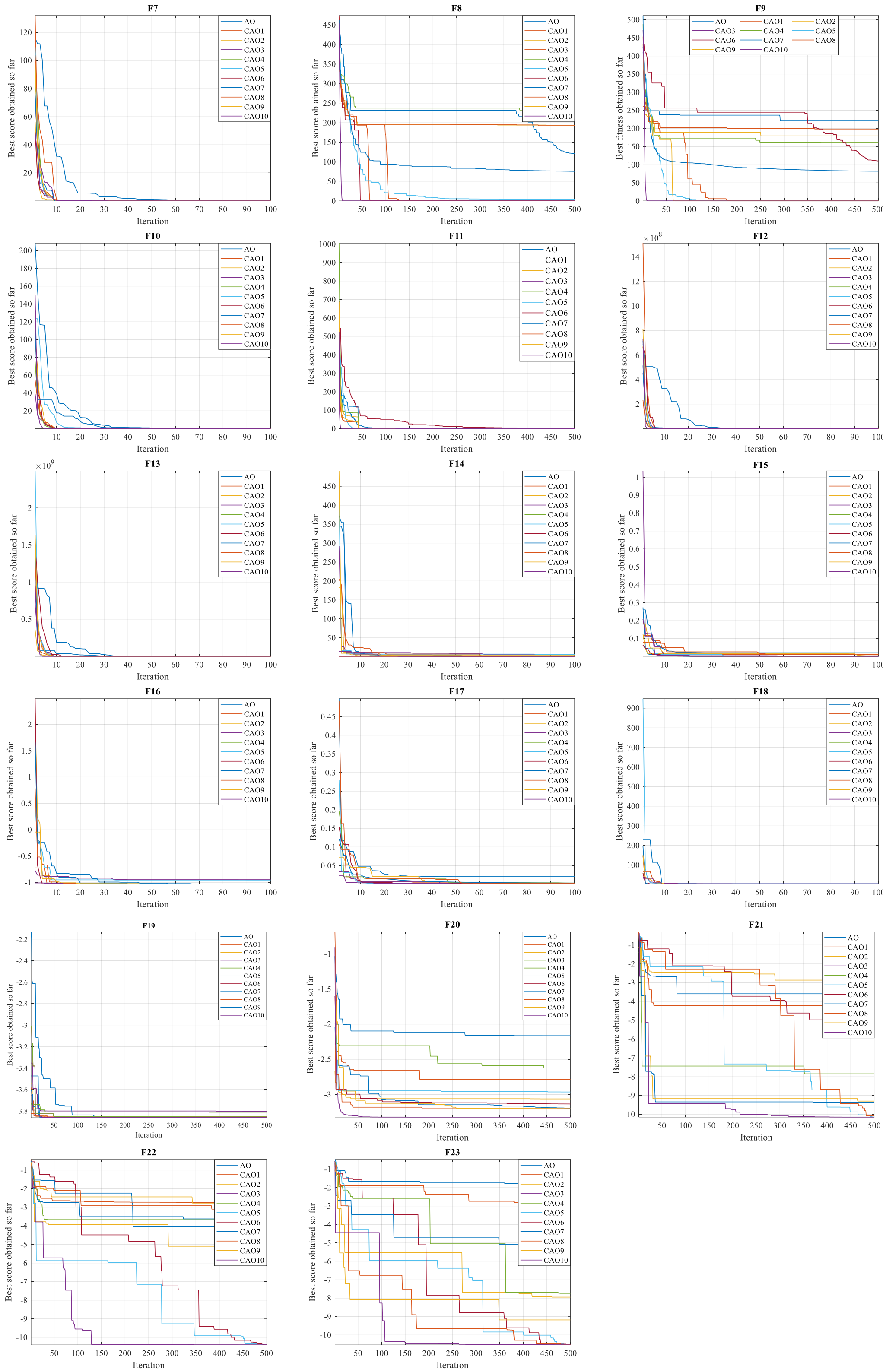


Figure 4. 20D Convergence graphs.

Table 3. Results on unimodal benchmark functions (F1 to F7) with Dim = 20.

Algorithms	Metric	F1	F2	F3	F4	F5	F6	F7
CAO1	Mean	3.1594×10^{-62}	4.2767×10^{-45}	1.4931×10^{-45}	7.3686×10^{-43}	0.0042	0.0514	9.7338×10^{-3}
	Std	1.4107×10^{-61}	9.2632×10^{-54}	5.5336×10^{-44}	3.7618×10^{-42}	0.0186	0.1182	0.0319
CAO2	Mean	1.9363×10^{-47}	1.7334×10^{-62}	5.7325×10^{-44}	1.7001×10^{-53}	0.0023	0.0812	0.0013
	Std	1.1737×10^{-46}	9.4942×10^{-62}	4.3096×10^{-42}	1.1469×10^{-51}	0.0035	0.1345	0.0280
CAO3	Mean	1.6439×10^{-46}	3.7819×10^{-48}	6.1514×10^{-40}	7.1135×10^{-43}	0.0017	0.0905	0.0034
	Std	2.3888×10^{-45}	2.5496×10^{-47}	4.3359×10^{-38}	1.7891×10^{-42}	0.0035	0.1602	0.0198
CAO4	Mean	1.8571×10^{-51}	1.5478×10^{-51}	9.7073×10^{-41}	4.7818×10^{-39}	0.0017	0.0883	1.5364×10^{-4}
	Std	1.5543×10^{-50}	9.8913×10^{-51}	2.4082×10^{-39}	5.4813×10^{-38}	0.0031	0.1391	0.0173
CAO5	Mean	1.2846×10^{-42}	5.0386×10^{-43}	1.0864×10^{-38}	3.6411×10^{-47}	0.0044	0.0580	0.0022
	Std	4.3618×10^{-41}	2.2073×10^{-42}	9.4501×10^{-37}	1.8711×10^{-46}	0.0181	0.1135	0.0167
CAO6	Mean	2.1626×10^{-41}	6.9392×10^{-45}	1.3965×10^{-38}	8.9057×10^{-40}	0.0013	0.0804	0.0044
	Std	8.1082×10^{-41}	2.3261×10^{-43}	4.0487×10^{-37}	1.1046×10^{-38}	0.0024	0.1502	0.0224
CAO7	Mean	2.5815×10^{-45}	7.3256×10^{-44}	7.4666×10^{-37}	1.0667×10^{-40}	0.0016	0.0543	0.0016
	Std	1.5931×10^{-42}	6.6050×10^{-43}	2.2757×10^{-35}	2.3155×10^{-40}	0.0032	0.0976	0.0130
CAO8	Mean	3.0509×10^{-42}	7.5109×10^{-44}	2.3251×10^{-37}	3.0214×10^{-39}	0.0012	0.0717	0.0017
	Std	2.4369×10^{-40}	9.4416×10^{-42}	2.9999×10^{-35}	5.0822×10^{-38}	0.0023	0.1229	0.0120
CAO9	Mean	1.3773×10^{-42}	4.1022×10^{-43}	5.3556×10^{-40}	2.2169×10^{-39}	0.0021	0.0562	0.0011
	Std	1.6850×10^{-41}	1.4126×10^{-42}	2.1538×10^{-38}	2.3310×10^{-38}	0.0034	0.1030	0.0205
CAO10	Mean	4.3124×10^{-68}	2.5689×10^{-70}	2.0973×10^{-54}	1.0760×10^{-62}	0.0071	0.0805	0.0012
	Std	4.7876×10^{-67}	1.2543×10^{-68}	1.5917×10^{-52}	3.1530×10^{-62}	0.0257	0.1208	0.0156
AO	Mean	3.6324×10^{-41}	2.9733×10^{-42}	7.8430×10^{-36}	9.6616×10^{-39}	0.0125	0.3160	0.0132
	Std	3.3158×10^{-40}	2.3904×10^{-41}	7.8266×10^{-34}	7.1005×10^{-38}	0.0266	0.2063	0.0242

Table 4. Results on multimodal benchmark functions (F8 to F13) with Dim = 20.

Algorithms	Metric	F8	F9	F10	F11	F12	F13
CAO1	Mean	1.128×10^{3}	0.227400	7.881×10^{-10}	1.440×10^{-6}	0.2511	0.0018
	Std	5.164×10^{3}	0.858809	7.519×10^{-10}	9.349×10^{-6}	0.4087	0.0028
CAO2	Mean	25.134100	9.274×10^{-11}	6.312×10^{-10}	1.482×10^{-9}	0.1523	0.0482
	Std	2.730×10^{2}	7.059×10^{-9}	6.498×10^{-10}	4.489×10^{-9}	0.2858	0.1142
CAO3	Mean	1.588×10^{2}	3.333×10^{-9}	2.921×10^{-5}	6.270×10^{-10}	0.1919	0.0670
	Std	5.571×10^{2}	5.292×10^{-8}	1.629×10^{-4}	4.489×10^{-9}	0.3959	0.2197
CAO4	Mean	47.835000	1.316×10^{-9}	3.727×10^{-9}	1.117×10^{-9}	0.2230	0.0337
	Std	4.622×10^{2}	4.845×10^{-9}	1.593×10^{-8}	8.992×10^{-9}	0.3903	0.1814
CAO5	Mean	5.288×10^{2}	4.422×10^{-9}	2.357×10^{-9}	3.155×10^{-8}	0.2683	0.0550
	Std	1.922×10^{3}	1.084×10^{-8}	2.638×10^{-8}	1.044×10^{-8}	0.4091	0.2154
CAO6	Mean	1.141×10^{2}	6.606×10^{-9}	8.621×10^{-9}	5.130×10^{-9}	0.1631	0.0227
	Std	4.698×10^{2}	3.252×10^{-8}	5.440×10^{-8}	1.600×10^{-8}	0.2899	0.1198
CAO7	Mean	71.133400	0.0871088	5.638×10^{-9}	3.205×10^{-7}	0.2318	0.0557
	Std	3.352×10^{2}	0.4535700	5.563×10^{-8}	1.986×10^{-6}	0.3853	0.1901
CAO8	Mean	84.312066	1.953×10^{-9}	1.111×10^{-7}	1.465×10^{-6}	0.1854	0.0345
	Std	3.613×10^{2}	1.191×10^{-8}	1.41×10^{-07}	1.170×10^{-05}	0.3095	0.1823
CAO9	Mean	29.966509	0.030900	3.551×10^{-09}	8.202×10^{-09}	0.1856	0.0650
	Std	2.804×10^{2}	0.680800	1.962×10^{-07}	3.330×10^{-16}	0.2957	0.2426
CAO10	Mean	2.3081090	3.420×10^{-12}	4.984×10^{-10}	4.897×10^{-10}	0.0837	0.1403
	Std	2.080×10^{2}	1.540×10^{-9}	1.753×10^{-08}	4.489×10^{-09}	0.2473	0.1641
AO	Mean	2.420×10^{4}	0.3440844	0.646×10^{-04}	1.465×10^{-06}	0.3082	0.1432
	Std	1.328×10^{5}	0.7684421	1.522×10^{-04}	1.170×10^{-05}	0.4069	0.3872

Table 5. Results on multimodal with fixed dimension benchmark functions (F14 to F23) with Dim = 20.

Algorithms	Metric	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
CAO1	Mean	23.5815	0.7105	0.3121	0.6889	0.4504	0.6033	0.2810	3.7579	5.4131	3.9216
	Std	10.8277	0.8045	0.5615	0.6711	0.6893	0.2101	0.1678	0.1280	3.9482	1.1888
CAO2	Mean	23.8706	0.2734	0.3081	0.4911	0.4548	0.5483	0.3202	3.8651	3.7238	3.8291
	Std	11.4591	0.3355	0.5541	0.3181	0.6102	0.2939	0.1500	0.1651	1.3278	0.0517
CAO3	Mean	31.6299	1.4277	0.3079	0.6752	0.4524	0.5031	0.2954	3.7590	1.0592	3.8082
	Std	0.41229	1.6146	0.5556	0.6677	0.6841	0.3595	0.1492	0.1682	0.1421	0.3970
CAO4	Mean	8.40730	0.6766	0.3040	0.5198	0.4749	0.5548	0.2938	3.9112	3.9740	3.8283
	Std	33.6743	1.2477	0.5514	0.5011	0.6844	0.2472	0.1360	0.2399	0.1241	0.2028
CAO5	Mean	15.7013	0.9387	0.3093	0.4331	0.4683	0.5235	0.4689	3.8067	3.8458	3.9310
	Std	22.3390	0.7604	0.5639	0.4098	0.6746	0.3146	0.2938	0.1692	0.1553	1.1082
CAO6	Mean	24.0695	0.3185	0.3139	0.4919	0.4848	0.5148	0.2917	5.9160	0.8748	3.7528
	Std	10.5741	0.3032	0.5611	0.5331	0.0635	0.3276	0.1999	0.0601	0.1946	0.3190
CAO7	Mean	24.0737	0.7898	0.2891	0.7011	0.4567	0.4898	0.4005	3.9008	3.8113	3.8739
	Std	10.8516	0.7628	0.5315	0.7910	0.7009	0.3756	0.2455	0.1454	0.1721	2.2105
CAO8	Mean	23.7654	0.6129	0.3119	0.6771	0.4580	0.5075	0.3241	4.7385	3.7550	3.8911
	Std	11.4256	0.6960	0.563	0.6212	0.7014	0.3539	0.3456	0.3129	0.4660	1.0406
CAO9	Mean	31.7341	0.6033	0.2629	0.7104	0.4975	0.5429	0.3343	5.8325	4.8077	3.8313
	Std	0.80712	0.5942	0.4756	0.7408	0.7069	0.3129	0.2659	2.2863	2.1332	0.234
CAO10	Mean	7.95817	0.1204	0.2999	0.2184	0.4882	0.4955	0.2783	3.6677	0.9978	3.7450
	Std	33.9401	0.0826	0.5475	0.2718	0.2718	0.3839	0.1771	0.1095	0.0039	0.1277
AO	Mean	32.0059	7.4968	0.3206	0.8011	0.4998	0.6318	0.4710	7.7926	7.8387	3.9236
	Std	0.26778	9.0191	0.5494	0.8237	0.6997	0.1871	0.2549	0.2027	0.1019	0.2180

Table 6. Results on unimodal benchmark functions (F1 to F7) with Dim = 30.

Algorithms	Metric	F1	F2	F3	F4	F5	F6	F7
CAO1	Mean	6.9002×10^{-56}	4.3889×10^{-46}	2.8527×10^{-50}	4.2177×10^{-43}	0.0013	0.0680	0.0020
	Std	8.8970×10^{-55}	1.6921×10^{-45}	2.0478×10^{-46}	2.9221×10^{-42}	0.0021	0.1099	0.0301
CAO2	Mean	2.0179×10^{-48}	1.7606×10^{-62}	1.0364×10^{-50}	5.7924×10^{-54}	0.0012	0.0668	0.0023
	Std	1.1076×10^{-47}	1.0465×10^{-61}	5.6617×10^{-50}	3.0722×10^{-53}	0.0025	0.1158	0.0081
CAO3	Mean	4.9040×10^{-46}	5.7033×10^{-49}	2.5248×10^{-43}	1.8416×10^{-44}	0.0022	0.0439	0.0048
	Std	1.0077×10^{-44}	4.3851×10^{-47}	2.5420×10^{-41}	3.2764×10^{-43}	0.0036	0.0671	0.0174
CAO4	Mean	1.4185×10^{-49}	2.7644×10^{-44}	2.2256×10^{-41}	2.4460×10^{-40}	0.0013	0.0960	0.0024
	Std	3.0797×10^{-49}	1.2080×10^{-43}	6.0544×10^{-39}	3.0646×10^{-39}	0.0025	0.1386	0.0181
CAO5	Mean	7.9361×10^{-41}	5.5580×10^{-45}	1.0328×10^{-45}	1.4898×10^{-41}	0.0033	0.0960	0.0018
	Std	3.6388×10^{-40}	8.9315×10^{-44}	3.4739×10^{-43}	8.7590×10^{-40}	0.0109	0.0688	0.0114
CAO6	Mean	5.1744×10^{-44}	4.4873×10^{-46}	3.5064×10^{-41}	1.8246×10^{-39}	0.0017	0.0948	0.0012
	Std	5.1589×10^{-43}	4.2547×10^{-45}	1.4646×10^{-39}	1.6917×10^{-38}	0.0025	0.1491	0.0231
CAO7	Mean	2.9060×10^{-43}	2.1857×10^{-45}	5.0238×10^{-41}	8.6015×10^{-41}	0.0019	0.0670	0.0039
	Std	1.5628×10^{-42}	6.0383×10^{-45}	2.6723×10^{-39}	3.1661×10^{-40}	0.0026	0.1033	0.0190
CAO8	Mean	6.2145×10^{-41}	9.7401×10^{-45}	3.7685×10^{-39}	4.3596×10^{-39}	0.0052	0.0760	0.0038
	Std	6.3794×10^{-40}	6.2932×10^{-43}	1.8435×10^{-37}	2.9606×10^{-38}	0.0214	0.1170	0.0218
CAO9	Mean	7.5027×10^{-44}	1.0543×10^{-51}	1.5536×10^{-37}	2.3961×10^{-46}	0.0031	0.0835	6.0726×10^{-04}
	Std	7.4475×10^{-43}	2.3570×10^{-51}	8.7301×10^{-36}	3.2826×10^{-45}	0.0040	0.1346	0.0274
CAO10	Mean	1.1623×10^{-66}	4.0243×10^{-66}	4.3410×10^{-59}	4.8664×10^{-60}	0.0019	0.1576	0.0013
	Std	6.2719×10^{-66}	1.8822×10^{-65}	2.3865×10^{-57}	1.4679×10^{-59}	0.0031	0.1515	0.0234
AO	Mean	1.0825×10^{-40}	6.6106×10^{-44}	1.9768×10^{-37}	2.9982×10^{-38}	0.0069	0.3194	0.0051
	Std	2.2192×10^{-39}	1.7556×10^{-43}	7.6752×10^{-36}	2.0650×10^{-37}	0.0047	0.2003	0.0187

Table 7. Results on multimodal benchmark functions (F8 to F13) with Dim = 30.

Algorithms	Metric	F8	F9	F10	F11	F12	F13
CAO1	Mean	52.697001	$3.294 \times 10–^{08}$	1.850×10^{-09}	5.011×10^{-09}	0.1381	0.0649
	Std	$3.006 \times 10^{+02}$	2.894×10^{-07}	6.374×10^{-07}	2.336×10^{-08}	0.2774	0.1940
CAO2	Mean	23.330010	2.415×10^{-04}	4.589×10^{-08}	5.585×10^{-07}	0.1574	0.0824
	Std	$3.553 \times 10^{+02}$	0.849300	1.809×10^{-07}	4.629×10^{-06}	0.3011	0.1570
CAO3	Mean	$1.214 \times 10^{+02}$	1.329×10^{-05}	9.265×10^{-08}	1.992×10^{-08}	0.1779	0.0556
	Std	$2.600 \times 10^{+02}$	3.123×10^{-04}	3.094×10^{-07}	1.677×10^{-07}	0.3364	0.2147
CAO4	Mean	91.173000	9.416×10^{-08}	1.334×10^{-08}	6.218×10^{-08}	0.1865	0.0358
	Std	$2.387 \times 10^{+02}$	8.232×10^{-07}	3.551×10^{-08}	2.195×10^{-07}	0.3347	0.1818
CAO5	Mean	$1.819 \times 10^{+02}$	0.2393342	9.295×10^{-08}	9.967×10^{-07}	0.2566	0.0655
	Std	$2.982 \times 10^{+02}$	1.4389110	3.808×10^{-07}	4.127×10^{-06}	0.3844	0.1811
CAO6	Mean	$1.144 \times 10^{+02}$	0.063000	5.721×10^{-09}	9.769×10^{-08}	0.1862	0.0909
	Std	$3.260 \times 10^{+02}$	0.206623	3.485×10^{-08}	5.083×10^{-07}	0.3813	0.2269
CAO7	Mean	40.235900	4.242×10^{-09}	7.161×10^{-10}	2.315×10^{-07}	0.1767	0.0773
	Std	$2.829 \times 10^{+02}$	5.530×10^{-08}	2.708×10^{-08}	6.464×10^{-04}	0.3302	0.1732
CAO8	Mean	$1.761 \times 10^{+02}$	2.882×10^{-08}	1.0661×10^{-08}	3.582×10^{-05}	0.1563	0.0540
	Std	$1.606 \times 10^{+03}$	1.866×10^{-07}	5.102×10^{-08}	1.908×10^{-06}	0.3400	0.2567
CAO9	Mean	$1.354 \times 10^{+02}$	0.7106120	1.447×10^{-08}	4.202×10^{-09}	0.2829	0.0517
	Std	$2.710 \times 10^{+02}$	1.0148110	7.504×10^{-08}	4.608×10^{-08}	0.3036	0.1689
CAO10	Mean	17.014212	6.319×10^{-10}	2.465×10^{-11}	5.004×10^{-09}	0.2543	0.0524
	Std	$2.604 \times 10^{+02}$	1.191×10^{-08}	1.913×10^{-11}	1.923×10^{-08}	0.3421	0.1734
AO	Mean	$2.994 \times 10^{+02}$	0.748800	3.492×10^{-07}	3.396×10^{-05}	0.3405	0.1537
	Std	$1.582 \times 10^{+03}$	0.819100	1.674×10^{-06}	3.876×10^{-05}	0.3647	0.3426

Table 8. Results on multimodal with fixed dimension benchmark functions (F14 to F23) with Dim = 30.

Algorithms	Metric	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
CAO1	Mean	31.918	1.1166	0.3081	0.6889	0.4899	0.5171	0.4047	3.8699	5.3699	3.8539
	Std	0.0699	1.2725	0.5628	0.6711	0.6985	0.3029	0.2557	0.2449	4.8676	0.1391
CAO2	Mean	31.936	0.9725	0.3080	0.4911	0.4986	0.5312	0.4397	3.8709	0.9948	3.8559
	Std	0.1160	0.9758	0.5600	0.3181	0.7026	0.3043	0.3033	0.1236	0.0123	0.1689
CAO3	Mean	32.037	0.9263	0.3105	0.6752	0.4977	0.5368	0.3294	5.8206	3.9013	3.9448
	Std	0.0142	0.8718	0.5618	0.6677	0.1488	0.2868	0.1753	0.1237	0.0854	1.0730
CAO4	Mean	31.973	0.1353	0.2910	0.5198	0.4976	0.4685	0.3022	5.9415	3.8704	3.9107
	Std	0.0211	0.6356	0.5250	0.5011	0.7081	0.3552	0.1299	0.0813	0.0246	0.9368
CAO5	Mean	31.954	0.3539	0.3036	0.4331	0.4874	0.4880	0.3211	7.7962	3.7861	3.9607
	Std	0.0686	0.2355	0.5524	0.4098	0.6919	0.3801	0.1898	0.3882	0.2508	0.0660
CAO6	Mean	7.9194	0.8295	0.3134	0.4919	0.4779	0.3578	0.2871	7.8445	3.8344	3.8886
	Std	33.945	0.7270	0.5640	0.5331	0.6502	0.5928	0.1834	0.0885	0.1883	0.2245
CAO7	Mean	31.964	0.8388	0.2694	0.7011	0.4820	0.4690	0.4029	3.8997	4.8758	3.9378
	Std	0.1856	0.7919	0.4932	0.7910	0.6969	0.3992	0.2639	0.1499	2.1707	0.0595
CAO8	Mean	31.885	1.4273	0.2995	0.6771	0.4830	0.5411	0.3148	3.9203	3.8131	3.8889
	Std	0.0522	2.1896	0.5718	0.6212	0.6941	0.2783	0.1701	0.1310	0.2318	0.1612
CAO9	Mean	15.972	0.5239	0.3012	0.7104	0.4737	0.4795	0.3305	3.7537	3.8840	3.7793
	Std	22.601	0.3270	0.5496	0.7408	0.7095	0.3992	0.1632	0.2041	0.1461	1.1065
CAO10	Mean	7.7902	0.1977	0.3222	0.2184	0.4799	0.5024	0.2753	3.7018	0.9758	3.7749
	Std	11.730	0.6718	0.5497	0.2718	0.6983	0.3657	0.1387	0.2791	0.0453	1.0418
AO	Mean	32.998	1.4459	0.3236	0.8011	0.5868	0.5943	0.4518	7.9899	3.8359	5.1871
	Std	0.1388	7.1337	0.5439	0.8237	0.7067	0.3673	0.1722	0.4604	2.6451	1.8885

Table 9. Results on unimodal benchmark functions (F1 to F7) with Dim = 50.

Algorithms	Metric	F1	F2	F3	F4	F5	F6	F7
CAO1	Mean	1.4014×10^{-45}	8.8678×10^{-57}	1.1554×10^{-40}	5.8719×10^{-42}	0.0069	0.1093	0.0040
	Std	7.6487×10^{-45}	2.1708×10^{-56}	5.5669×10^{-39}	5.3266×10^{-41}	0.0232	0.1638	0.0291
CAO2	Mean	2.6731×10^{-51}	2.4961×10^{-59}	1.9667×10^{-43}	7.4677×10^{-54}	0.0021	0.1861	5.4088×10^{-04}
	Std	2.0123×10^{-50}	1.3672×10^{-58}	1.0771×10^{-42}	4.8434×10^{-53}	0.0071	0.1962	0.0080
CAO3	Mean	1.8612×10^{-48}	6.0549×10^{-50}	3.6566×10^{-43}	8.6094×10^{-43}	0.0029	0.1253	0.0042
	Std	3.3861×10^{-46}	4.6964×10^{-49}	2.7153×10^{-41}	2.2832×10^{-41}	0.0031	0.1751	0.0226
CAO4	Mean	2.9703×10^{-43}	2.7805×10^{-45}	3.0920×10^{-38}	5.7765×10^{-39}	0.0054	0.0967	4.6365×10^{-04}
	Std	1.6591×10^{-41}	6.2944×10^{-45}	1.6459×10^{-36}	3.3849×10^{-38}	0.0074	0.1482	0.0135
CAO5	Mean	1.4490×10^{-43}	1.5213×10^{-45}	1.6830×10^{-45}	4.8799×10^{-40}	0.0053	0.1653	0.0028
	Std	1.7810×10^{-42}	3.7847×10^{-44}	1.6462×10^{-43}	8.1681×10^{-39}	0.0121	0.1630	0.0239
CAO6	Mean	2.8062×10^{-43}	3.2209×10^{-47}	2.7807×10^{-41}	8.7902×10^{-40}	0.0050	0.1185	0.0018
	Std	1.1892×10^{-42}	7.7997×10^{-46}	1.5526×10^{-39}	9.6235×10^{-39}	0.0177	0.1697	0.0178
CAO7	Mean	1.2935×10^{-44}	6.5273×10^{-49}	3.7121×10^{-41}	1.2178×10^{-39}	0.0069	0.1665	0.0018
	Std	7.2688×10^{-44}	2.4915×10^{-46}	4.7472×10^{-39}	8.3726×10^{-38}	0.0167	0.1897	0.0201
CAO8	Mean	2.1431×10^{-50}	3.2920×10^{-53}	1.2780×10^{-38}	3.6523×10^{-44}	0.0031	0.1188	0.0014
	Std	8.4976×10^{-50}	3.8756×10^{-52}	1.4820×10^{-35}	1.6212×10^{-43}	0.0252	0.1546	0.0224
CAO9	Mean	4.8437×10^{-43}	6.7968×10^{-46}	1.4760×10^{-38}	3.6295×10^{-41}	0.0034	0.0841	8.9504×10^{-04}
	Std	1.8334×10^{-42}	9.7044×10^{-45}	3.0308×10^{-36}	6.4156×10^{-40}	0.0040	0.1349	0.0152
CAO10	Mean	3.6296×10^{-65}	3.1257×10^{-67}	9.5705×10^{-50}	9.8494×10^{-58}	0.0033	0.0790	7.4727×10^{-05}
	Std	8.2527×10^{-64}	3.2552×10^{-66}	4.1655×10^{-47}	6.5074×10^{-57}	0.0116	0.1022	0.0148
AO	Mean	1.5736×10^{-41}	9.6721×10^{-45}	7.9429×10^{-35}	1.1173×10^{-37}	0.0080	0.4122	0.1962
	Std	4.8437×10^{-43}	5.4167×10^{-44}	4.8572×10^{-33}	1.1009×10^{-36}	0.0090	0.1355	0.0278

Table 10. Results on multimodal benchmark functions (F8 to F13) with Dim = 50.

Algorithms	Metric	F8	F9	F10	F11	F12	F13
CAO1	Mean	50.576180	8.948×10^{-10}	7.102×10^{-11}	8.376×10^{-10}	0.1909	0.0332
	Std	$4.529 \times 10^{+02}$	6.276×10^{-09}	7.895×10^{-09}	4.557×10^{-09}	0.3220	0.1311
CAO2	Mean	$1.089 \times 10^{+02}$	1.145×10^{-09}	6.343×10^{-10}	4.510×10^{-10}	0.2746	0.0317
	Std	$3.043 \times 10^{+02}$	6.065×10^{-09}	7.622×10^{-09}	3.941×10^{-09}	0.4175	0.0927
CAO3	Mean	$2.454 \times 10^{+02}$	2.308×10^{-09}	1.864×10^{-06}	2.348×10^{-09}	0.1883	0.1515
	Std	$8.856 \times 10^{+02}$	1.333×10^{-08}	5.692×10^{-06}	1.071×10^{-08}	0.3099	0.2685
CAO4	Mean	19.685672	1.323×10^{-08}	8.463×10^{-09}	1.078×10^{-09}	0.3117	0.0771
	Std	$5.120 \times 10^{+02}$	2.827×10^{-08}	6.374×10^{-08}	7.045×10^{-09}	0.2585	0.2566
CAO5	Mean	$9.737 \times 10^{+03}$	3.089×10^{-08}	1.829×10^{-09}	2.185×10^{-09}	0.3554	0.1723
	Std	$5.333 \times 10^{+04}$	1.630×10^{-07}	2.002×10^{-08}	1.132×10^{-08}	0.3874	0.2590
CAO6	Mean	75.224186	6.554×10^{-09}	3.572×10^{-08}	2.693×10^{-09}	0.4354	0.0339
	Std	$3.022 \times 10^{+02}$	3.432×10^{-08}	1.473×10^{-07}	1.040×10^{-08}	0.4623	0.1805
CAO7	Mean	$1.046 \times 10^{+02}$	1.120×10^{-07}	3.017×10^{-09}	7.572×10^{-10}	0.1572	0.0428
	Std	$3.382 \times 10^{+02}$	2.570×10^{-06}	2.594×10^{-08}	3.839×10^{-09}	0.2686	0.1845
CAO8	Mean	20.061502	3.531×10^{-09}	7.303×10^{-08}	1.915×10^{-09}	0.3418	0.1002
	Std	$3.566 \times 10^{+02}$	5.911×10^{-08}	1.790×10^{-07}	9.994×10^{-09}	0.3668	0.2592
CAO9	Mean	24.954709	6.108×10^{-06}	3.673×10^{-09}	8.857×10^{-10}	0.2028	0.0707
	Std	$3.791 \times 10^{+02}$	2.172×10^{-04}	9.125×10^{-09}	4.306×10^{-09}	0.2753	0.2037
CAO10	Mean	18.734093	7.141×10^{-10}	6.571×10^{-10}	3.384×10^{-10}	0.1750	0.0199
	Std	2.7839001	1.735×10^{-09}	1.493×10^{-09}	4.752×10^{-09}	0.3443	0.1116
AO	Mean	$8.926 \times 10^{+07}$	6.732×10^{-06}	7.217×10^{-05}	2.255×10^{-04}	0.5738	0.2460
	Std	$4.315 \times 10^{+08}$	2.183×10^{-04}	2.243×10^{-04}	1.122×10^{-04}	0.4749	0.3448

Table 11. Results on multimodal with fixed dimension benchmark functions (F14 to F23) with Dim = 50.

Algorithms	Metric	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
CAO1	Mean	31.874	0.9322	0.3055	0.1637	0.4995	0.4806	0.3814	3.8509	3.9448	3.9409
	Std	0.0465	0.8668	0.5478	0.0409	0.7069	0.3978	0.4140	0.1732	1.1602	1.1492
CAO2	Mean	31.917	0.1638	0.3002	0.4812	0.4668	0.5282	0.4510	3.8910	3.7960	3.9572
	Std	0.0682	0.0646	0.5530	0.3717	0.6909	0.3089	0.2553	0.1070	1.3177	0.0726
CAO3	Mean	24.043	0.2942	0.3040	1.0520	0.4791	0.4969	0.3017	3.9100	3.8041	5.9418
	Std	11.334	0.1701	0.5658	0.9127	0.6936	0.3773	0.1710	0.1315	0.3023	0.0674
CAO4	Mean	31.976	1.8398	0.3109	0.8081	0.4547	0.5124	0.3173	3.8638	3.8865	3.9464
	Std	0.0042	1.6369	0.5621	0.9290	0.7076	0.3391	0.1510	0.0887	0.0943	0.1587
CAO5	Mean	31.876	1.3204	0.3147	0.7224	0.4800	0.5029	0.3687	3.8620	3.8693	3.8476
	Std	0.4820	1.5182	0.5641	0.6030	0.7008	0.3546	0.2967	0.1743	0.2381	0.1844
CAO6	Mean	31.930	2.1473	0.3134	1.1640	0.4993	0.5075	0.3338	0.9762	3.9215	5.1173
	Std	0.1420	1.8419	0.5649	1.1249	0.7075	0.3262	0.1845	0.1684	0.0909	1.9043
CAO7	Mean	32.287	0.9121	0.3093	1.3451	0.4935	0.5137	0.4635	3.8378	4.8509	4.8710
	Std	0.0970	0.8089	0.5560	1.2379	0.7043	0.3444	0.3737	0.2739	2.0968	2.3336
CAO8	Mean	31.828	1.8464	0.3057	0.2626	0.4971	0.5074	0.2967	5.9450	3.8956	3.8634
	Std	0.0010	1.7214	0.5464	0.1301	0.7052	0.3633	0.1610	0.0708	0.1400	1.1657
CAO9	Mean	32.034	0.1696	0.2998	0.9402	0.4955	0.4998	0.4130	3.8688	3.9100	3.9037
	Std	0.1250	0.0429	0.5620	0.9664	0.7046	0.3690	0.2628	0.1256	0.1563	0.9978
CAO10	Mean	16.075	1.2854	0.2083	0.6132	0.4493	0.4995	0.2803	3.9291	0.9998	3.8473
	Std	22.609	1.1661	0.5630	0.4642	0.6790	0.3534	0.1555	0.0738	0.0639	0.1782
AO	Mean	32.988	2.5993	0.3138	1.4186	0.4997	0.5326	0.4773	7.9595	5.8418	5.9869
	Std	0.0990	1.7676	0.5370	1.2984	0.1453	0.3496	0.2462	0.0707	0.1536	0.0108

Table 12. Parameter values for all algorithms.

Algorithms	Parameters
Proposed CAO	N (Population size) = 30, *tmax* = 500 C1 (Control variable 1) = 2, C2 (Control variable 2) = 6, C3 (Control variable 3) = 2 and C4 (Control variable 4) = 0.5
WOA	a1 = [0, 2]; a2 = [−2, −1]; b = 1
GWO	a = [0, 2]; r1 ∈ [0, 1]; r2 ∈ [0, 1]
MVO	Existence probability ∈ [0.2, 1]; traveling distance rate ∈ [0.6, 1]
SSA	1 ∈ [0, 1]; c2 ∈ [0, 1]
GSA	α = 20; G 0 = 100
SCA	a = 2; r 4 = [0, 1]; r 2 = [0, 2]
PSO	c1 = 2; c2 = 2; v max = 6
MFO	b = 1; t = [−1, 1]; a ∈ [−2, −1]

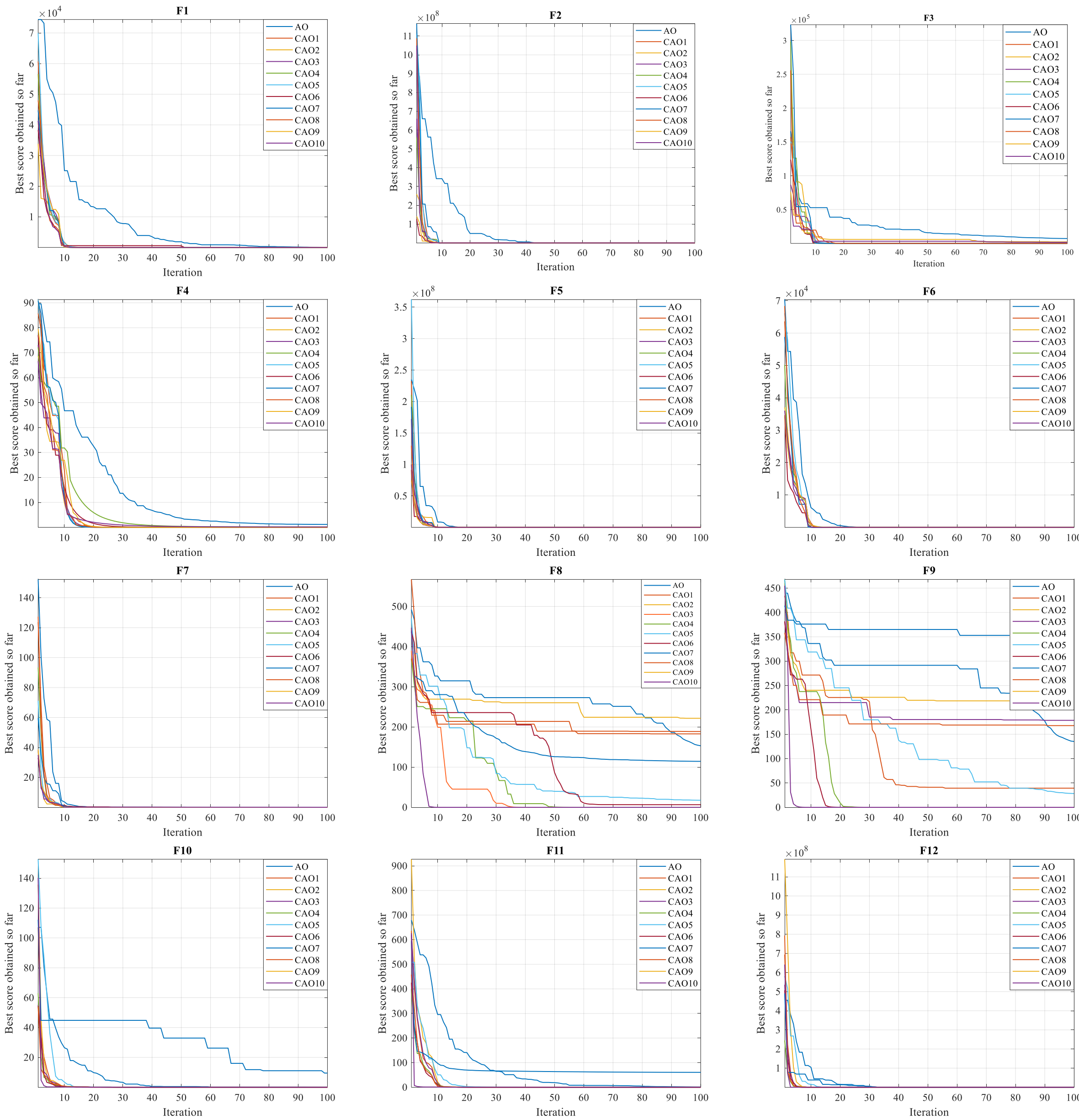


Figure 5. *Cont.*

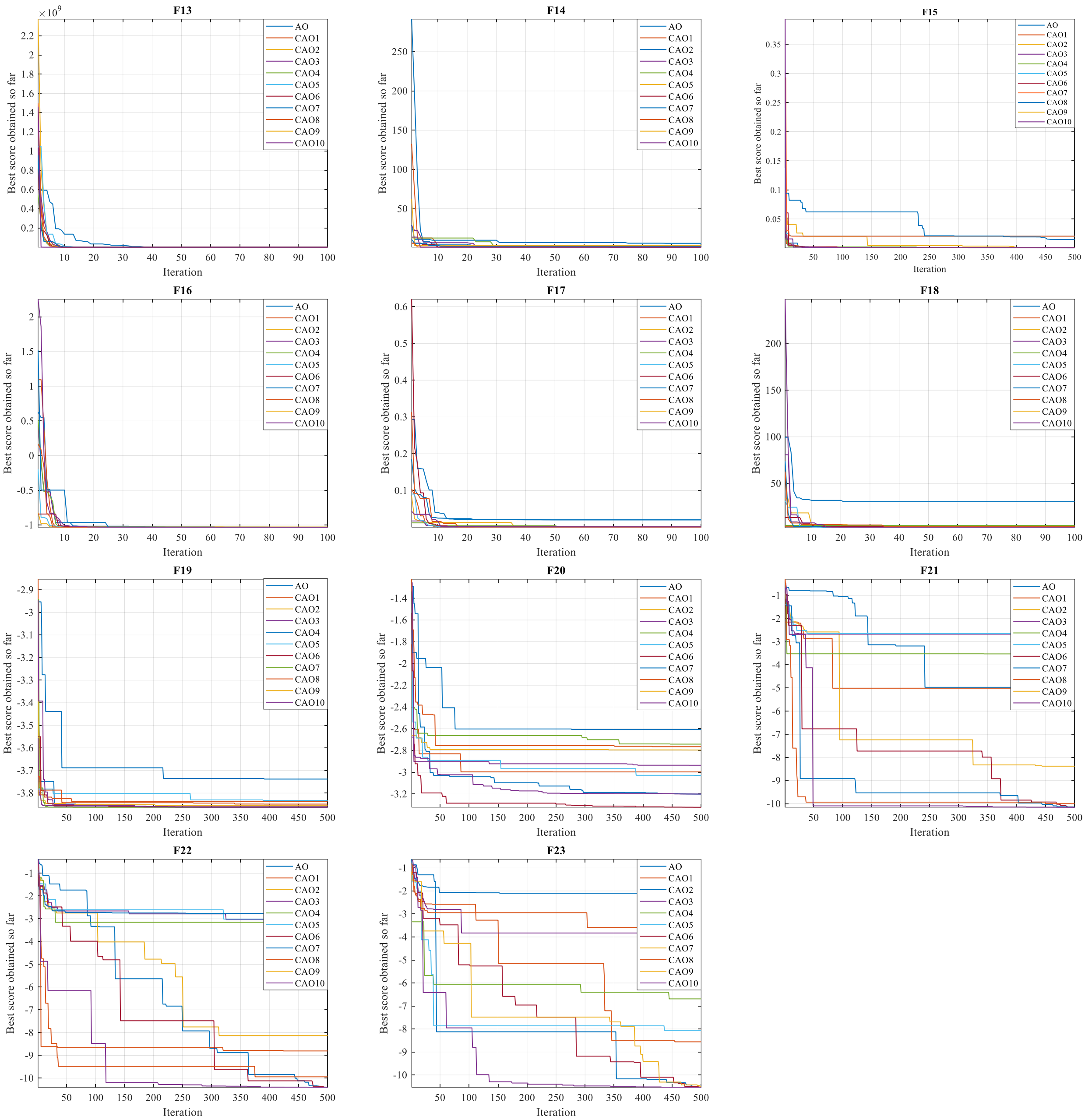


Figure 5. 30D Convergence graphs.

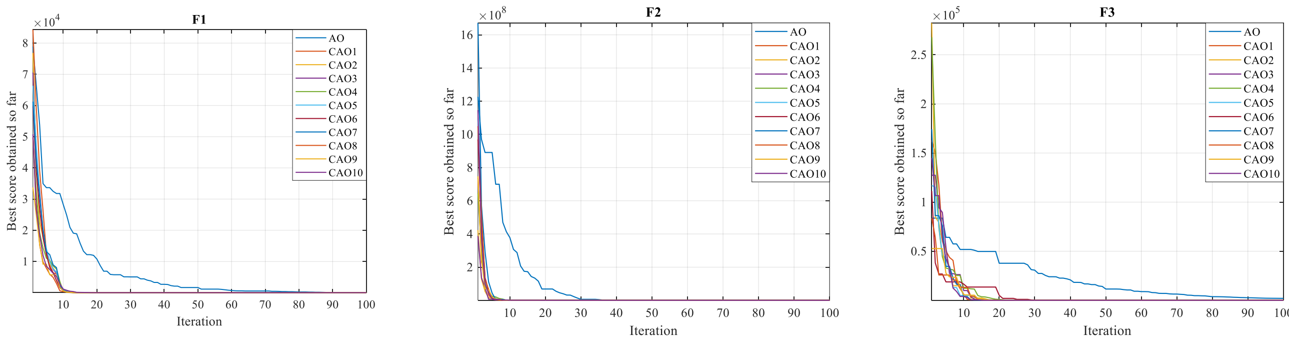


Figure 6. *Cont.*

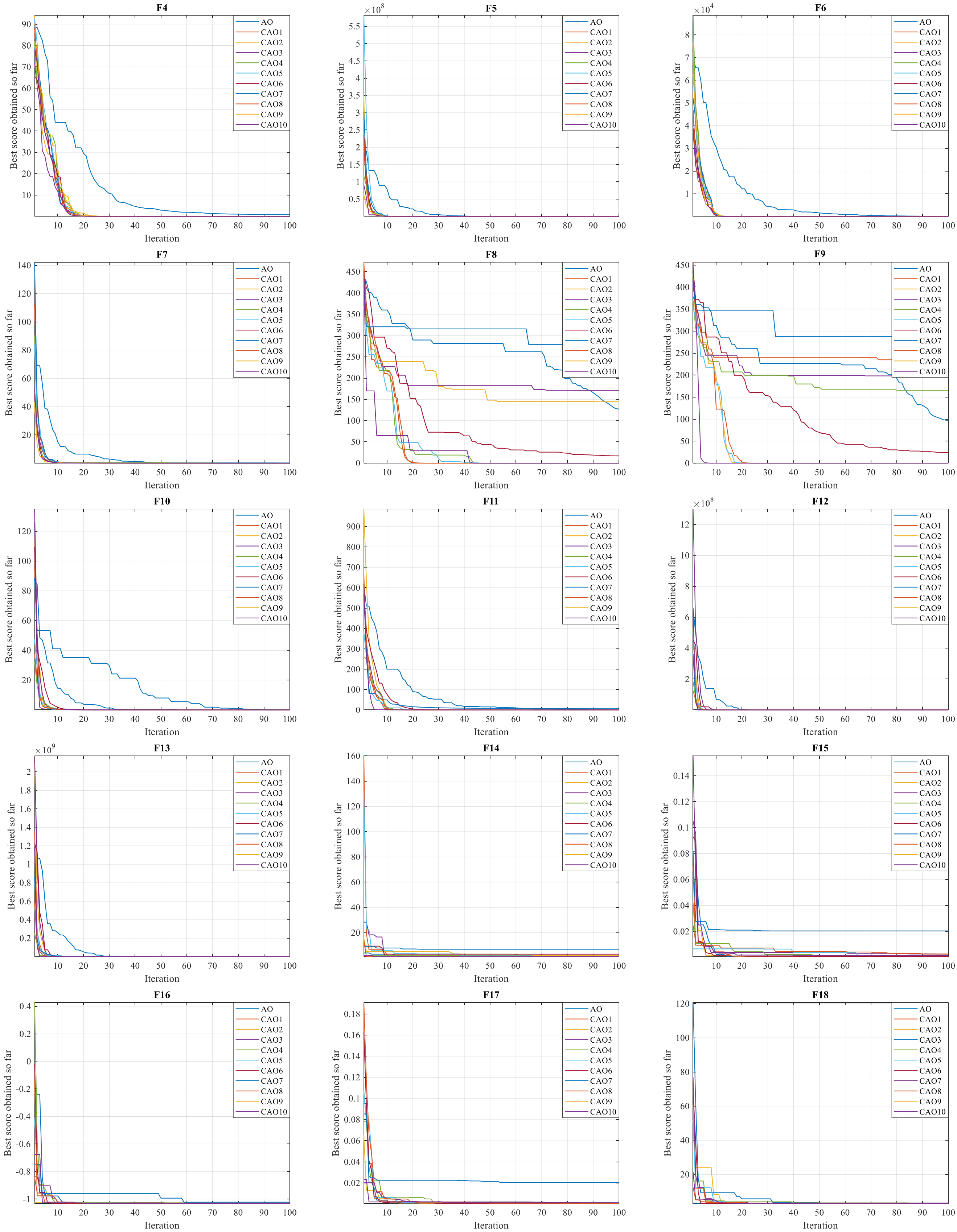


Figure 6. *Cont.*

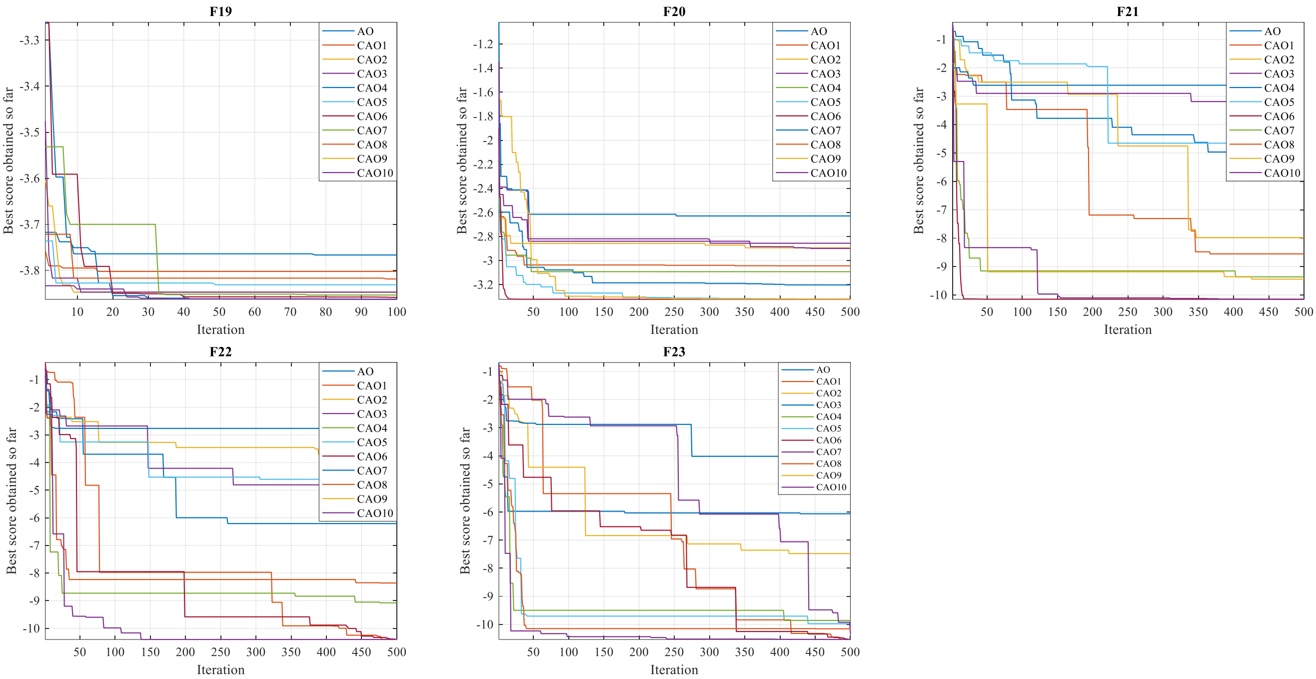


Figure 6. 50D Convergence graphs.

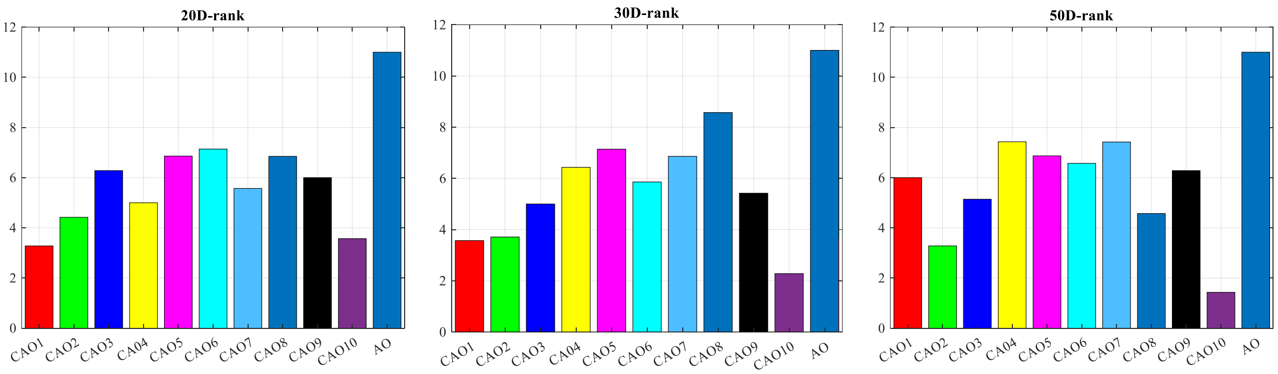


Figure 7. 20D, 30D, and 50D Rank bar graph.

The mean results (Mean) and standard deviations (Std) obtained by the different algorithms when solving problems F1–F23 are presented in Tables 13–15. The evaluation of the convergence behavior of the algorithms in solving the set of problems is also shown in Figure 8. Moreover, Tables 13–15 show conclusively that the CAO10 algorithm outperforms all other algorithms in all cases (F1–F23).

The convergence curves shown in Figure 8 highlight the exceptional convergence speed of CAO10 compared with other algorithms in all situations. Indeed, CAO10 shows remarkable convergence from the very first stages of the search, whereas other algorithms struggle to improve the quality of solutions, even after a greater number of exploratory stages.

These results highlight the strengths of the CAO10 algorithm, which manages to significantly speed up the early stages of the search due to its chaotic initialization strategy. In addition, CAO10 significantly improves its chances of avoiding local optima, thus favoring the discovery of optimal solutions.

In short, these findings attest to the undeniable effectiveness of the CAO10 algorithm, demonstrating its ability to solve a wide range of problems quickly and accurately. This combination of rapid convergence and high performance makes it a preferred choice for optimization in complex contexts.

Table 13. Comparison results on unimodal benchmark functions (F1 to F7).

Algorithms	Metric	F1	F2	F3	F4	F5	F6	F7
CAO10	Mean	0	0	0	0	0.0783	0.0033	2.563×10^{-04}
	Std	0	0	0	0	0	0	0
WOA	Mean	5.9699×10^{-73}	3.2843×10^{-26}	$4.3199 \times 10^{+04}$	48.6732	27.3683	0.2995	0.0010
	Std	0	0	0	0	0	0	0
GWO	Mean	$7.5948 \times 10^{+02}$	$6.0593 \times 10^{+10}$	$3.3671 \times 10^{+03}$	4.2158	$1.7138 \times 10^{+06}$	$5.9424 \times 10^{+02}$	0.5919
	Std	$5.7409 \times 10^{+03}$	$1.3549 \times 10^{+12}$	$1.4126 \times 10^{+04}$	15.1590	$1.6081 \times 10^{+07}$	$4.5826 \times 10^{+03}$	6.4210
MVO	Mean	0.6912	0.8424	$6.1245 \times 10^{+02}$	1.1334	44.0211	1.2652	0.0527
	Std	0	0	0	0	0	0	0
SSA	Mean	9.3343×10^{-08}	0.7420	$1.8868 \times 10^{+03}$	15.7646	$1.0708 \times 10^{+02}$	2.4597×10^{-07}	0.1490
	Std	0	0	0	0	0	0	0
GSA	Mean	$2.3130 \times 10^{+03}$	$1.9539 \times 10+04$	$5.2702 \times 10^{+03}$	7.8291	$7.9100 \times 10^{+05}$	$2.0677 \times 10^{+03}$	1.1004
	Std	$3.2912 \times 10^{+03}$	$3.7007 \times 10+05$	$1.4597 \times 10^{+04}$	11.2299	$1.1827 \times 10^{+07}$	$5.7917 \times 10^{+03}$	5.2359
SCA	Mean	$8.6473 \times 10^{+03}$	$1.7905 \times 10+03$	$4.5324 \times 10^{+04}$	71.8592	$1.0117 \times 10^{+08}$	$1.6909 \times 10^{+04}$	39.0650
	Std	$1.8116 \times 10^{+04}$	$3.9504 \times 10+04$	$4.8487 \times 10^{+04}$	21.4367	$1.2722 \times 10^{+08}$	$2.4662 \times 10^{+04}$	50.2985
PSO	Mean	-3.4910×10^{-25}	3.2843×10^{-26}	2.3859×10^{-15}	-8.0173×10^{-20}	0.5224	-0.5000	0.0018
	Std	6.0413×10^{-24}	2.6330×10^{-25}	3.0418×10^{-14}	6.4544×10^{-19}	0.1985	0.0057	0.0294
MFO	Mean	$1.0003 \times 10^{+04}$	40.0700	$2.9998 \times 10^{+04}$	69.6565	$4.2170 \times 10^{+02}$	3.9826	3.0054
	Std	0	0	0	0	0	0	0

Table 14. Comparison results on multimodal benchmark functions (F8 to F13).

Algorithms	Metric	F8	F9	F10	F11	F12	F13
CAO10	Mean	7.1672516	0	8.881×10^{-16}	0	0.025844	0.0017
	Std	$3.770 \times 10^{+02}$	0	0	0	0	0
WOA	Mean	$3.623 \times 10^{+03}$	7.212×10^{-10}	31.43220	-1.032×10^{-07}	0.177169	$1.328 \times 10^{+07}$
	Std	$8.615 \times 10^{+02}$	4.546×10^{-09}	3.109956	4.394×10^{-07}	0.344965	$1.114 \times 10^{+08}$
GWO	Mean	$1.254 \times 10^{+04}$	$1.475 \times 10^{+02}$	9.744673	$1.777 \times 10^{+02}$	$1.173 \times 10^{+07}$	0.1690040
	Std	0	$1.014 \times 10^{+02}$	3.296500	$1.165 \times 10^{+02}$	$5.037 \times 10^{+07}$	0.0029300
MVO	Mean	$3.760 \times 10^{+03}$	$1.297 \times 10^{+02}$	3.265648	33.63071	$1.394 \times 10^{+07}$	$4.145 \times 10^{+07}$
	Std	$1.170 \times 10^{+03}$	92.400120	5.764688	$1.041 \times 10^{+02}$	$8.101 \times 10^{+07}$	$1.889 \times 10^{+08}$
SSA	Mean	$7.170 \times 10^{+03}$	$2.190 \times 10^{+02}$	19.90393	76.21996	$4.983 \times 10^{+07}$	$5.291 \times 10^{+07}$
	Std	0	0	0	0	0	0
GSA	Mean	$7.084 \times 10^{+03}$	$1.748 \times 10^{+02}$	3.938290	1.364197	7.186429	3.5204640
	Std	0	0	0	0	0	0
SCA	Mean	$2.755 \times 10^{+03}$	$1.399 \times 10^{+02}$	19.41303	$2.184 \times 10^{+02}$	$3.305 \times 10^{+08}$	$5.687 \times 10^{+08}$
	Std	$3.936 \times 10^{+02}$	80.31465	2.937372	$2.278 \times 10^{+02}$	$2.700 \times 10^{+08}$	$4.470 \times 10^{+08}$
PSO	Mean	$7.317 \times 10^{+03}$	87.71319	8.928028	10.524	26.664201	$5.272 \times 10^{+02}$
	Std	0	0	0	0	0	0
MFO	Mean	88.475008	4.342×10^{-07}	1.174×10^{-06}	-8.520×10^{-05}	0.797739	0.5992800
	Std	$3.571 \times 10^{+02}$	1.643×10^{-05}	3.876×10^{-05}	0.001445	0.907648	0.4798860

Table 15. Comparison results on multimodal with fixed dimension benchmark functions (F14 to F23).

Algorithms	Metric	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23
CAO10	Mean	0.9980	6.237×10^{-04}	0.31060	4.957×10^{-04}	0.4583	0.4961	0.2838	0.9201	1.1833	0.8968
	Std	0	0	0	0	0	0	0	0	0	0
WOA	Mean	23.9687	0.81739	0.9877	0.8630	4.1213	3.84711	2.6389	2.6353	3.8773	3.9407
	Std	11.1862	0.83644	0.5634	0.7705	0.6606	0.3830	0.1299	0.1273	0.1174	0.0610
GWO	Mean	4.71250	0.00729	1.0316	0.0232	3.06361	−3.8627	3.1972	10.1484	7.9243	2.7740
	Std	27.2895	0.01319	0.1565	0.0137	6.6742	0.0945	0.2608	1.0819	2.7934	1.6997
MVO	Mean	10.7640	0.00231	0.9853	0.0226	5.6369	3.8438	3.0637	1.7879	10.3952	10.535
	Std	0.00220	0.00649	0.3523	0.0175	19.1752	0.0340	0.1322	0.6476	2.8479	2.9281
SSA	Mean	3.96820	0.00118	1.0316	0.0012	2.9999	3.8627	3.2031	5.1007	6.1271	5.5101
	Std	0	0	0	0	0	0	0	0	0	0
GSA	Mean	15.5038	0.00140	1.0316	0.0186	3.0002	3.8627	3.1326	10.147	10.4022	10.536
	Std	0	0	0	0	0	0	0	0	0	0
SCA	Mean	8.41915	0.00420	0.9728	0.0033	4.2623	3.7441	2.7264	3.8667	2.7515	10.521
	Std	30.4592	0.00801	0.2243	0.0037	4.6039	0.5463	0.4135	0.4840	0.5961	0.1332
PSO	Mean	4.95049	0.00152	1.0316	0.0192	3.0322	3.8626	3.1114	10.1534	1.1833	3.8351
	Std	0	0	0	0	0	0	0	0	0	0
MFO	Mean	31.9783	2.12677	0.3114	2.0795	0.5067	0.5076	0.3449	4.70315	4.3068	4.3554
	Std	9.963×10^{-06}	2.05953	0.5674	2.0143	0.7071	0.3713	0.1900	5.501×10^{-05}	6.311×10^{-04}	6.316×10^{-04}

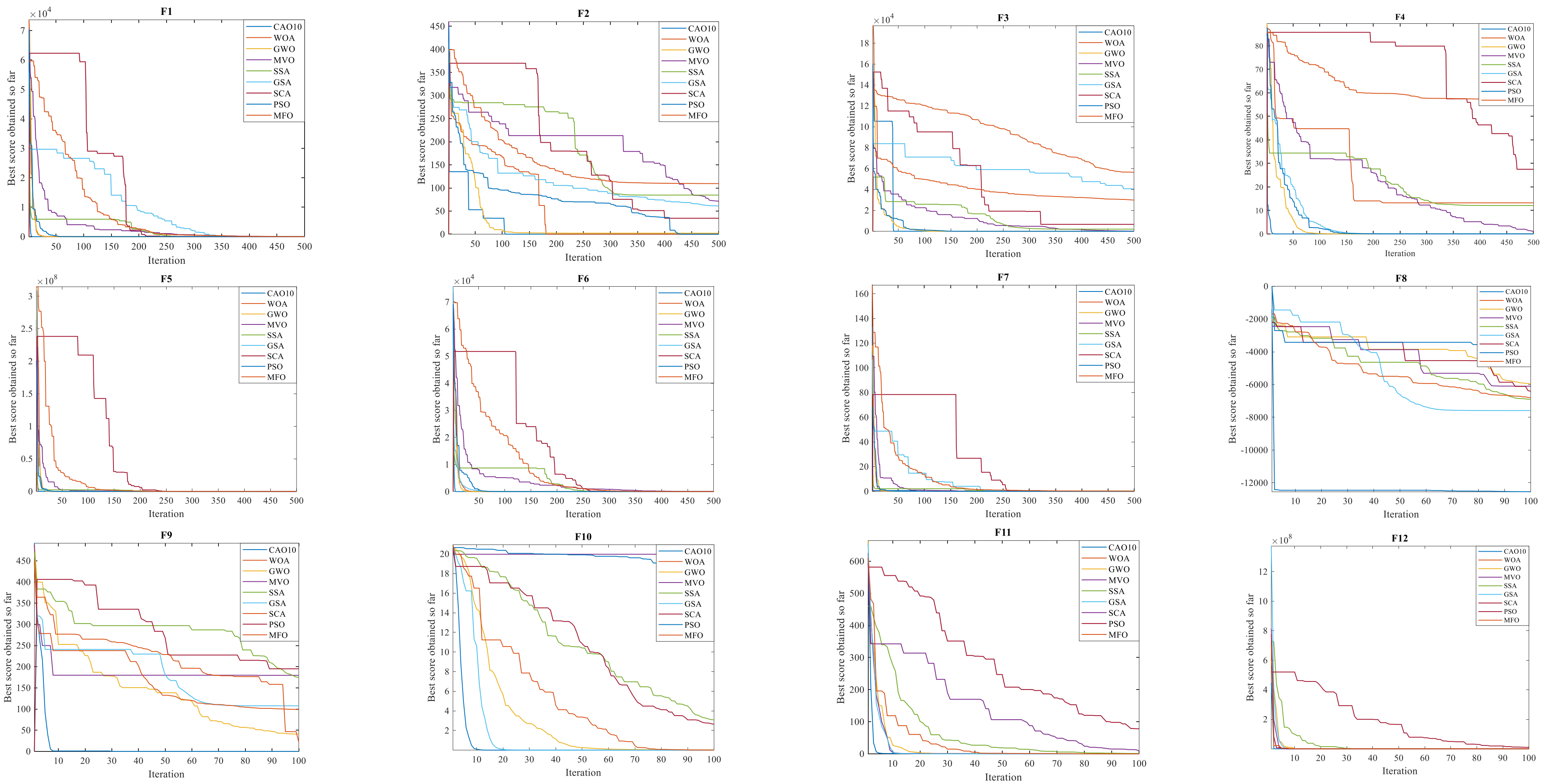


Figure 8. *Cont*.

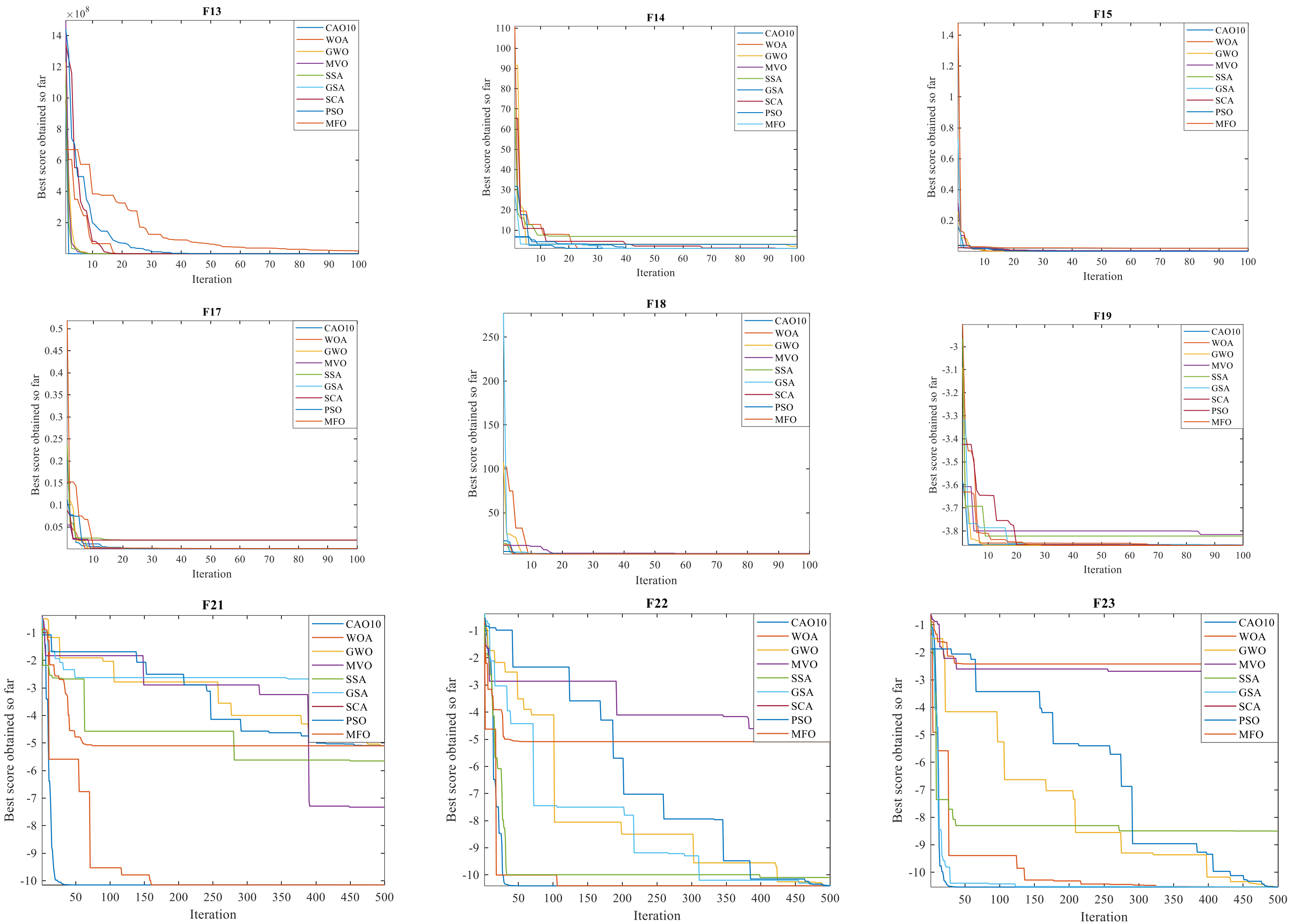


Figure 8. Convergence curves for CAO10 and other original algorithms.

6.2. Comparative Study on Two Real-World Applications

In this subsection, we use the CAO10 algorithm to solve three crucial engineering mathematical modeling problems, namely: (a) the design of welded beams; (b) the design of tension/compression springs; and (c) the design of pressure vessels. The results obtained are compared with those of numerous algorithms, including WOA, GWO, SSA, MVO, GSA, SCA, PSO, and MFO.

(a) The welded beam design problem (WBDP)

The objective of this test is to optimize the values of the supplied variables (l, h, t, and b) to minimize the manufacturing cost for the WBD problem (Figure 9). To meet this challenge, we used the CAO10 algorithm proposed for solving the WBD problem, which we compared with various other optimization algorithms to determine its degree of superiority.

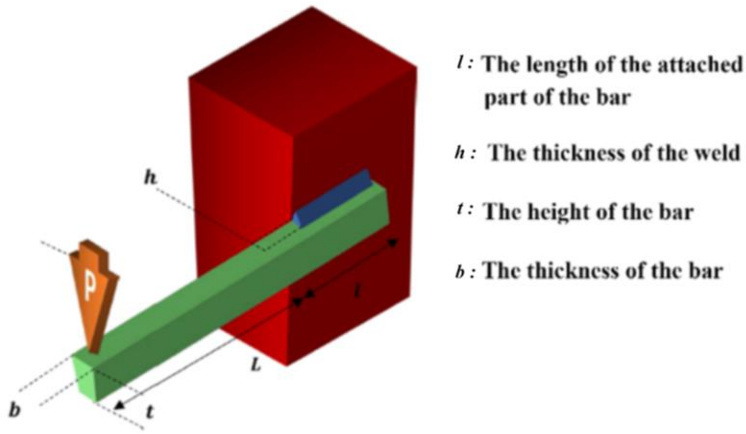


Figure 9. WBD problem [36].

The results of this comparison are shown in Table 16. They remarkably reveal that the performance of the CAO10 algorithm exceeds that of all the other algorithms considered, a significant finding. As a result, it can be concluded that the CAO10 algorithm succeeds in identifying the best solution for the WBD problem in pursuit of the best possible solution.

Table 16. Comparison results for WBDP.

Algorithms	The Optimal Values of the Variables				Optimal Cost
	h	**L**	**T**	**b**	
WOA	0.203481	3.522134	9.034608	0.205832	1.728744
GWO	0.210018	4.685682	9.612054	0.211448	2.129173
SSA	0.216840	3.332410	8.801918	0.216850	1.764686
MVO	0.205868	3.492594	9.020946	0.206450	1.730838
GSA	0.203591	3.586508	9.298896	0.209935	2.445557
SCA	0.206811	3.482429	9.734118	0.208347	1.870321
PSO	0.207641	3.590249	9.370719	0.211037	1.812654
MFO	0.211980	3.610641	9.512845	0.207155	2.236191
CAO10	**0.205403**	**3.478066**	**9.036632**	**0.205732**	**1.725388**

(b) The problem of tension/compression springs (TCSP)

In the TCSP problem, our aim is to minimize the minimum weight of this spring by optimizing three essential design variables (d, D, and the number of active coils, N), as illustrated in Figure 10. To solve this complex problem, we used the CAO10 algorithm, which was specifically designed for this task. We then compared the performance of the CAO10 algorithm with that of various competing optimization techniques.

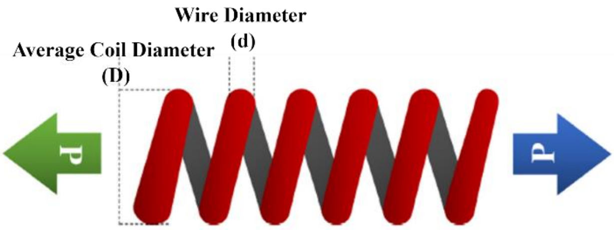


Figure 10. TCS problem [36].

The results of this comparison, summarized in Table 17, convincingly demonstrating the performance of the CAO10 algorithm. The CAO10 algorithm generates a greater number of optimal solutions than the other approaches examined.

Table 17. Comparison results for TCSP.

Algorithms	The Optimal Values of the Variables			Optimal Weight
	d	D	N	
WOA	0.054459	0.426290	8.154532	0.012838
GWO	0.086479	1.300000	2.000000	0.030506
SSA	0.071927	0.317342	14.05613	0.012738
MVO	0.050450	0.327552	13.27398	0.012734
GSA	0.053251	0.393056	9.575860	0.013263
SCA	0.053562	0.311936	14.37716	0.014633
PSO	0.067439	0.381298	9.674320	0.014429
MFO	0.057192	0.418211	14.11433	0.023833
CAO10	**0.050000**	**0.343364**	**12.11704**	**0.012671**

(c) Pressure vessel design problem (PVDP)

In the context of the PVDP, the main objective is to determine the total cost of the cylindrical pressure vessel, as shown in Figure 11. To satisfy the four constraints (Th, Ts, R, and L), optimization operations must take into account four design factors. Using the CAO10 algorithm to solve this problem, the simulation results obtained are compared with those obtained by several different optimization algorithms, as shown in Table 18. On the basis of these data, we can conclude that the suggested CAO10 algorithm has a better cost value than all other comparative algorithms, with the exception of the shortest path method.

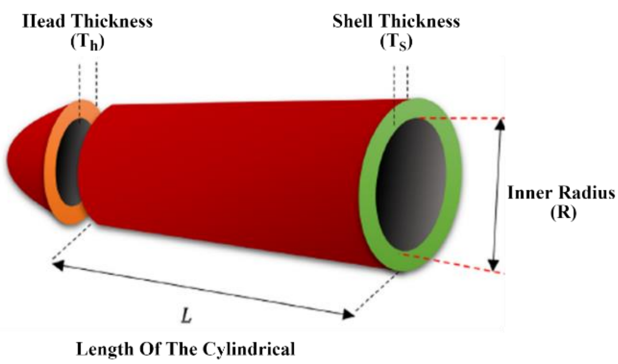


Figure 11. PVD problem [36].

Table 18. Comparison results for PVD.

Algorithms	The Optimal Values of the Variables				Optimal Cost
	Ts	Th	R	L	
WOA	0.787682	0.397850	40.79602	193.6284	$5.93255 \times 10^{+03}$
GWO	1.210365	14.23855	52.61191	200.0000	$8.04630 \times 10^{+04}$
SSA	1.252278	0.616931	64.64376	12.57938	$7.29154 \times 10^{+03}$
MVO	0.792020	0.404503	41.00121	191.1295	$5.96206 \times 10^{+03}$
GSA	0.872208	0.651819	40.57705	174.9799	$7.40800 \times 10^{+03}$
SCA	1.340987	0.593397	61.53646	35.53712	$8.21843 \times 10^{+03}$
PSO	0.912099	0.774210	42.83227	174.9999	$6.41810 \times 10^{+03}$
MFO	1.462991	1.018712	62.88539	65.99552	$8.78800 \times 10^{+03}$
CAO10	**0.781845**	**0.386473**	**40.51013**	**197.3649**	$\mathbf{5.89166 \times 10^{+03}}$

These results underline the exceptional effectiveness of the CAO10 algorithm in solving complex problems such as WBDP, TCSP, and PVDP, demonstrating its ability to achieve optimal solutions and outperform other optimization approaches. This breakthrough offers promising prospects for the application of the CAO10 algorithm in industrial optimization and engineering contexts, where the search for optimal solutions is of crucial importance.

6.3. Reconstruction of 2D and 3D Signals and Images Using Meixner Moments and the Chaos-Archimède Algorithm (CAO)

Discrete orthogonal moments [44] play a prominent role in signal and image analysis, and their usefulness extends to a multitude of varied applications. They have been successful in areas such as classification, reconstruction, watermarking, encryption, and compression [45–49]. They are a versatile tool for representing and processing complex data. In the context of signal and image reconstruction, the use of discrete orthogonal Meixner moments (MMs) is complemented by the calculation of polynomial Meixner values (MPs). These polynomials depend on local parameters, noted as (β, u). To achieve optimum reconstruction quality, it is imperative to adjust these parameters appropriately. This is where the Optimized Chaos-Archimede (CAO) algorithm comes in, proposing an innovative solution. The CAO10 algorithm is used to determine optimal values for (β, u), thus guaranteeing superior quality in image reconstruction. Figure 12 summarizes the key steps in implementing the CAO algorithm for signal and image reconstruction.

In this subsection, we validate the CAO10 algorithm's ability to reconstruct large medical signals and images using MMs. This validation process comprises three distinct tests. The first test aims to evaluate the performance of the proposed method for bio signals. The second test looks at the reconstruction of medical color images. Its aim is to demonstrate the ability of the CAO10 algorithm, in combination with MMs, to render color images accurately. Finally, the third test tackles an even more complex challenge, namely, the reconstruction of 3D images. We use the reconstruction method based on the CAO10 algorithm to achieve optimum results in the context of these three-dimensional images.

To assess the similarity between the original signal or image and those reconstructed, we use the criteria of mean square error (MSE) and peak signal-to-noise ratio (PSNR) in decibels (dB). These metrics enable us to objectively measure the quality of the reconstruction by quantifying the difference between the original signal or image and its reconstructed version, thus providing an accurate assessment of the CAO10 algorithm's performance in each test scenario.

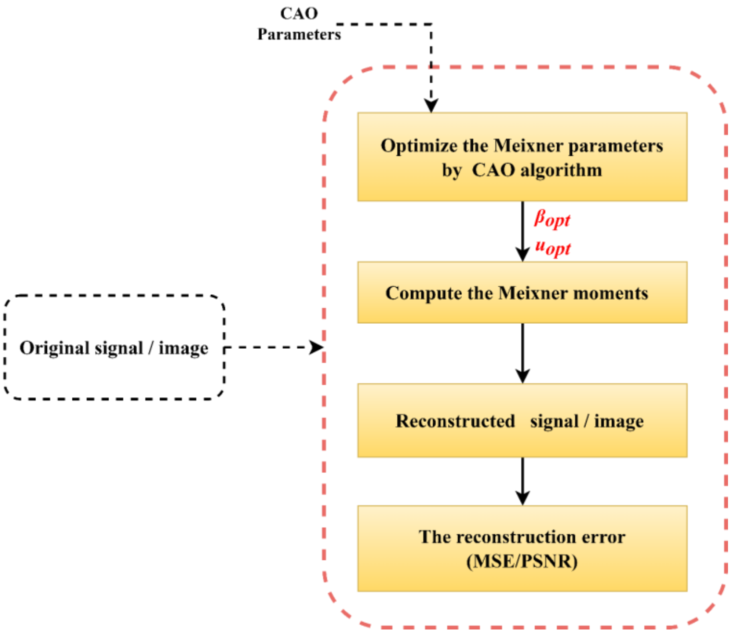


Figure 12. Reconstruction method using MMs and CAO.

$$MSE = \frac{1}{N} \times \sum_{x=0}^{N-1} \left(f(x) - \hat{f}(x) \right)^2 \tag{21}$$

$$MSE = \frac{1}{N \times M} \times \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} \left(f(x,y) - \hat{f}(x,y) \right)^2 \tag{22}$$

$$MSE = \frac{1}{N \times M \times K} \times \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} \sum_{z=0}^{K-1} \left(f(x,y,z) - \hat{f}(x,y,z) \right)^2 \tag{23}$$

$$PSNR = 10 \log_{10} \frac{k^2}{MSE} \tag{24}$$

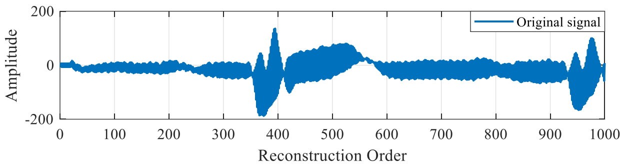
(a) Optimal bio signal reconstruction using MMs and the CAO algorithm

In this test, we verify the efficiency of the CAO10 algorithm for reconstructing large signals. To do this, we choose to reconstruct a specific ECG signal, named "MIT-BIH record 124", of size 1000 taken from the MIT-BIH database [50]. The CAO10 algorithm was employed to optimize the local parameters (β, u) of the MPs used in the reconstruction of this signal. The results obtained with the CAO10 algorithm were compared with those generated by various other optimization algorithms, such as WOA, GWO, SSA, MVO, GSA, SCA, PSO, and MFO.

Table 19 illustrates the original signal, as well as a set of signals reconstructed using the different optimization methods. This figure also highlights the reconstruction errors, measured in terms of MSE and PSNR, as well as the optimal values of the local parameters (β, u) of the MPs. In addition, Figure 13 shows the PSNR values of signals reconstructed using various algorithms.

The results of this test demonstrate that the method based on the CAO10 algorithm enables optimal determination of the parameters (β, u), leading to excellent signal reconstruction quality, characterized by low MSE and high PSNR compared with other optimization algorithms. These findings clearly underline the superiority and robustness of the CAO10 algorithm in the context of biological signal reconstruction, opening up new prospects for improving complex signal analysis in medicine and elsewhere.

Table 19. "MIT-BIH record 111" signals reconstructed by MMs using the proposed method based on the CAO algorithm method, compared with other algorithms.



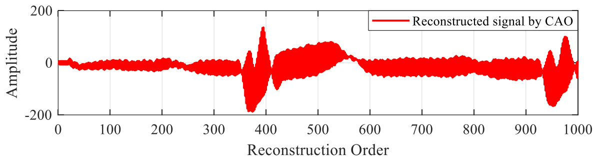
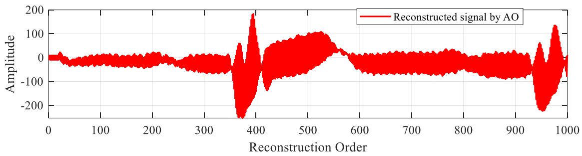
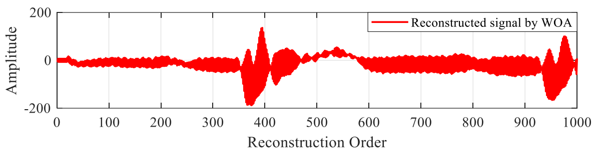
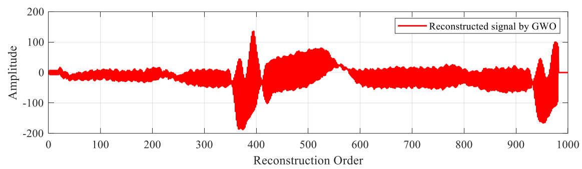
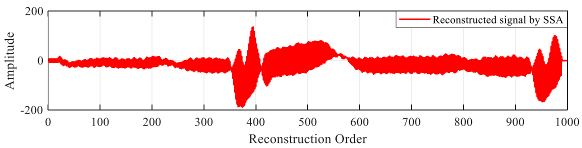
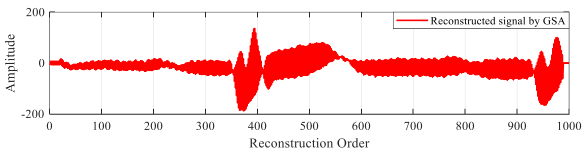
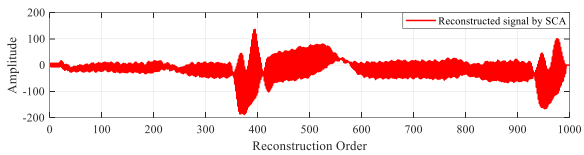
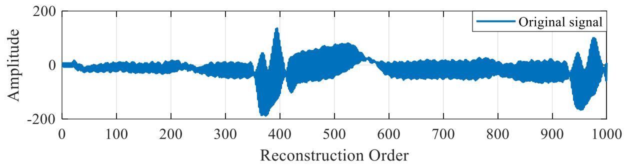
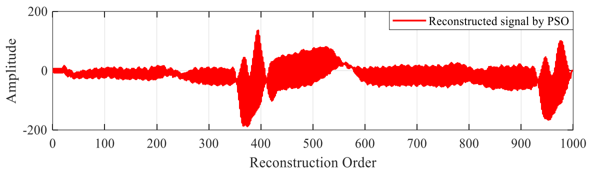
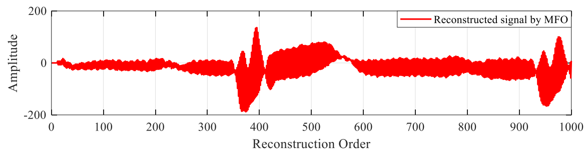
Methods	Optimal Values of β and u	Reconstructed Signal	MSE & PSNR
CAO10	$\beta = 700$ $\mu = 0.90$		$MSE = 2.341 \times 10^{-4}$ $PSNR = 78.978$
AO	$\beta = 55.781$ $\mu = 0.192$		$MSE = 0.1022$ $PSNR = 52.572$
WOA	$\beta = 270$ $\mu = 0.230$		$MSE = 0.1195$ $PSNR = 51.895$
GWO	$\beta = 120$ $\mu = 0.1$		$MSE = 0.0164$ $PSNR = 60.519$
SSA	$\beta = 11.920$ $\mu = 0.320$		$MSE = 0.0040$ $PSNR = 66.642$
MVO	$\beta = 650.874$ $\mu = 0.893$		$MSE = 0.00148$ $PSNR = 70.943$
GSA	$\beta = 20.815$ $\mu = 0.452$		$MSE = 0.00413$ $PSNR = 66.508$
SCA	$\beta = 461.533$ $\mu = 0.919$		$MSE = 0.00269$ $PSNR = 68.384$

Table 19. *Cont.*



Methods	Optimal Values of β and u	Reconstructed Signal	MSE & PSNR
PSO	$\beta = 661.22$ $\mu = 0.413$		$MSE = 0.0022$ $PSNR = 69.207$
MFO	$\beta = 600.541$ $\mu = 0.733$		$MSE = 0.00166$ $PSNR = 74.812$

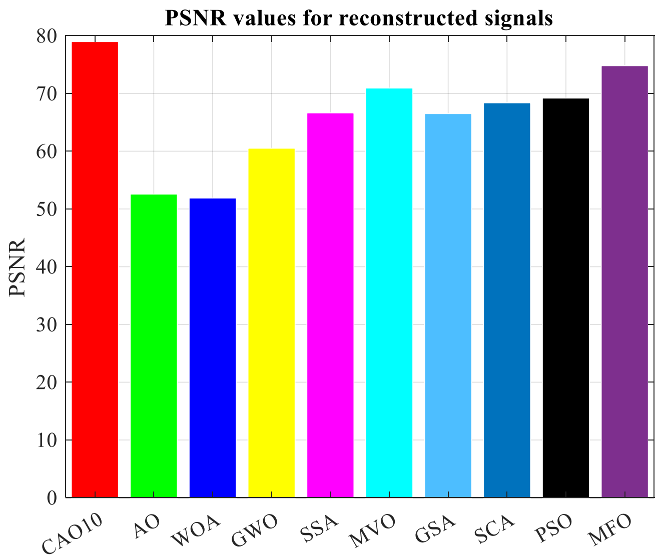


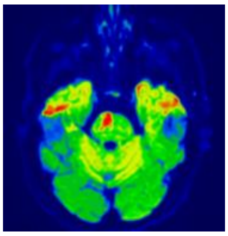
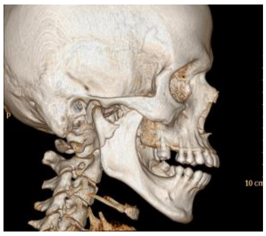
Figure 13. PSNR values for reconstructed signals.

(b) Optimal reconstruction of 2D medical images using CAO-optimized MMs

In this test, we evaluate the ability of the MM-based reconstruction method and CAO10 algorithm to reconstruct color medical images. We used 2D images of size 1024 from the database to carry out these experiments.

The CAO10 algorithm was used to optimize MMs parameters and perform medical image reconstruction. The results obtained by the CAO10 algorithm were compared with those generated by several other commonly used optimization algorithms, including WOA, GWO, SSA, MVO, GSA, SCA, PSO, and MFO. The reconstructed images, including those with the names "Brain", "multiple-osteochondromas", "mandible-fracture", and "soft-tissue-chondroma-thumb", were evaluated in terms of reconstruction quality. Table 20 shows the results of this evaluation, including reconstructed images, optimized MMs parameter values, and reconstruction errors, while Figure 14 shows the MSE and PSNR curves of the "brain" image reconstructed using various algorithms.

Table 20. Simulation results for medical image reconstruction of size 1024.

					
CAO10	β_{opt} μ_{opt}	$\beta = 750.980$ $\mu = 0.198$	$\beta = 500.119$ $\mu = 0.0154$	$\beta = 649.772$ $\mu = 0.266$	$\beta = 420.883$ $\mu = 0.933$
	Errors	$MSE = 4.341 \times 10^{-5}$ $PSNR = 84.852$	$MSE = 3.013 \times 10^{-5}$ $PSNR = 85.093$	$MSE = 1.341 \times 10^{-5}$ $PSNR = 85.380$	$MSE = 8.026 \times 10^{-5}$ $PSNR = 84.177$

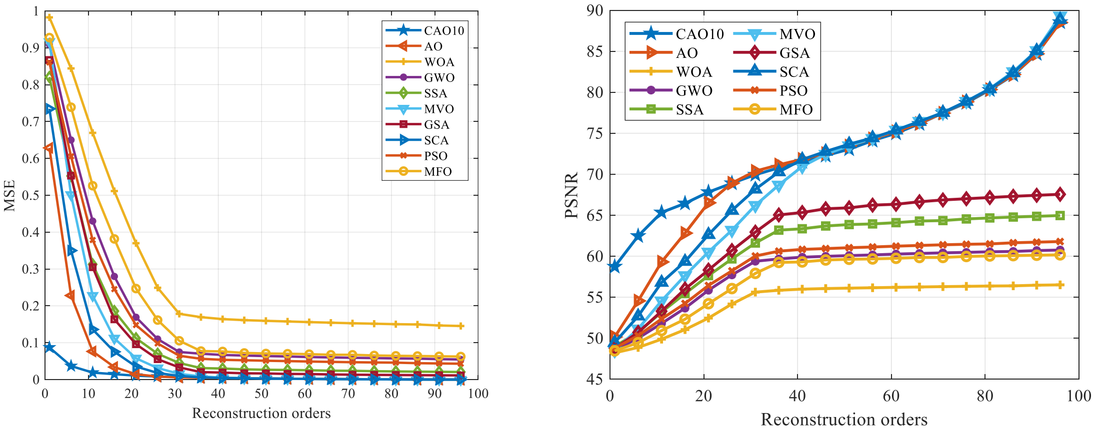


Figure 14. MSE and PSNR plot of reconstructed 'Brain' image.

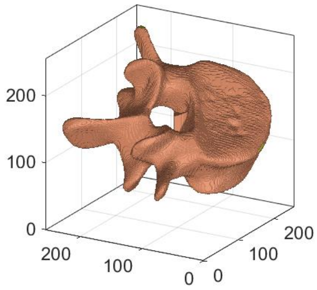
The results obtained attest to the ability of the CAO10 algorithm to determine optimal values of the parameters (β, u) enabling the reconstruction of all images with a considerably low MSE (high PSNR), thus outperforming the other algorithms evaluated. These findings eloquently demonstrate the superiority of the CAO10 algorithm in the field of medical image reconstruction.

(c) Optimal reconstruction of 3D images by MMs and the CAO algorithm

In this subsection, we evaluate the effectiveness of the proposed 3D reconstruction method based on MMs and the CAO10 algorithm. We used the "Verterba" 3D image of voxel size 256 downloaded from the [49] database. Table 21 shows the results of the reconstruction of this image using the CAO10 algorithm compared with other optimization algorithms. The values of the optimized parameters (β, u) and the error measures MSE and PSNR are also displayed.

The results of this test show that the MMs values optimized by the CAO10 algorithm lead to better reconstruction quality, characterized by very low MSE and high PSNR, compared with other methods. These results demonstrate the effectiveness of our approach to polynomial parameter selection and 3D image reconstruction.

Table 21. The comparison results of the 3D image of the "corona" with size 256 × 256 × 256.



The 3D image of the "Verterba" with size 256 × 256 × 256

Methods	**CAO10**	**AO**	**WOA**	**GWO**	**SSA**
Reconstruction errors	$MSE = 0.000784$ $PSNR = 79.1830$	$MSE = 0.0902$ $PSNR = 55.881$	$MSE = 0.00155$ $PSNR = 70.017$	$MSE = 0.0103$ $PSNR = 61.991$	$MSE = 0.0031$ $PSNR = 67.822$
Optimal values of (β, u)	$\beta = 700$ $\mu = 0.5$	$\beta = 55.781$ $\mu = 0.192$	$\beta = 530.144$ $\mu = 0.991$	$\beta = 120$ $\mu = 0.1$	$\beta = 11.920$ $\mu = 0.320$
Methods	**MVO**	**GSA**	**SCA**	**PSO**	**MFO**
Reconstruction errors	$MSE = 0.00118$ $PSNR = 72.733$	$MSE = 0.00491$ $PSNR = 66.440$	$MSE = 0.00201$ $PSNR = 69.118$	$MSE = 0.00198$ $PSNR = 69.355$	$MSE = 0.002$ $PSNR = 69.772$
Optimal values of (β, u)	$\beta = 650.874$ $\mu = 0.893$	$\beta = 20.815$ $\mu = 0.452$	$\beta = 461.533$ $\mu = 0.919$	$\beta = 661.22$ $\mu = 0.413$	$\beta = 600.541$ $\mu = 0.733$

7. Conclusions

In conclusion, this study presents a significant advance in the field of optimization thanks to the introduction of the improved Archimedes optimization algorithm, Chaotic_AO (CAO). This algorithm is based on the use of ten distinct chaotic maps to replace pseudorandom sequences in the essential components of AO, namely, initialization, density and volume updating, and position updating. This enhancement has succeeded in striking a more appropriate balance between mining and exploration, offering an increased probability of discovering global solutions.

Evaluation of CAO's performance across three distinct groups of problems highlighted its efficiency and reliability. The results of these tests clearly revealed that the CAO algorithm is not only efficient but also reliable. It demonstrated remarkable convergence speeds and exceptional solution quality in most of the cases studied. These observations confirm the real potential of the CAO algorithm for solving varied and complex problems. Overall, CAO offers exciting new prospects for improving optimization techniques, paving the way for wide-ranging applications in fields from engineering to medicine.

Following on from our current research, we plan to extend the application of the CAO algorithm to other areas of optimization, with particular emphasis on complex problems, such as multi-criteria optimization with conflicting objectives. This focus on more complex scenarios is intended to assess the robustness and versatility of the algorithm in the face of a variety of challenges.

Author Contributions: Conceptualization, Formal analysis, Writing original draft, Software: A.B. and M.A.T. Investigation, Visualization, Data curation: H.K. Validation, Methodology, Project administration, Supervision: M.S., H.Q., Y.E.A. and M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare that we have no competing interests in the submission of this manuscript.

References

1. Ayyarao, T.S.L.V.; Ramakrishna, N.S.S.; Elavarasan, R.M.; Polumahanthi, N.; Rambabu, M.; Saini, G.; Khan, B.; Alatas, B. War Strategy Optimization Algorithm: A New Effective Metaheuristic Algorithm for Global Optimization. *IEEE Access* **2022**, *10*, 25073–25105. [CrossRef]
2. Abdollahzadeh, B.; Gharehchopogh, F.S.; Khodadadi, N.; Mirjalili, S. Mountain Gazelle Optimizer: A new Nature-inspired Metaheuristic Algorithm for Global Optimization Problems. *Adv. Eng. Softw.* **2022**, *174*, 103282. [CrossRef]
3. Hajizadeh, Y.; Christie, M.; Demyanov, V. Ant colony optimization for history matching and uncertainty quantification of reservoir models. *J. Pet. Sci. Eng.* **2011**, *77*, 78–92. [CrossRef]
4. Premkumar, M.; Jangir, P.; Kumar, B.S.; Sowmya, R.; Alhelou, H.H.; Abualigah, L.; Yildiz, A.R.; Mirjalili, S. A New Arithmetic Optimization Algorithm for Solving Real-World Multiobjective CEC-2021 Constrained Optimization Problems: Diversity Analysis and Validations. *IEEE Access* **2021**, *9*, 84263–84295. [CrossRef]
5. Azizi, M.; Talatahari, S.; Gandomi, A.H. Fire Hawk Optimizer: A novel metaheuristic algorithm. *Artif. Intell. Rev.* **2023**, *56*, 287–363. [CrossRef]
6. Bencherqui, A.; Karmouni, H.; Daoui, A.; Alfidi, M.; Qjidaa, H.; Sayyouri, M. Optimization of Jacobi Moments Parameters using Artificial Bee Colony Algorithm for 3D Image Analysis. In Proceedings of the 2020 Fourth International Conference on Intelligent Computing in Data Sciences (ICDS), Fez, Morocco, 21–23 October 2020; pp. 1–7.
7. Mareli, M.; Twala, B. An adaptive Cuckoo search algorithm for optimisation. *Appl. Comput. Inform.* **2018**, *14*, 107–115. [CrossRef]
8. Alia, O.M.; Mandava, R. The variants of the harmony search algorithm: An overview. *Artif. Intell. Rev.* **2011**, *36*, 49–68. [CrossRef]
9. Alsalibi, B.; Abualigah, L.; Khader, A.T. A novel bat algorithm with dynamic membrane structure for optimization problems. *Appl. Intell.* **2021**, *51*, 1992–2017. [CrossRef]
10. Majak, J.; Pohlak, M.; Eerme, M.; Velsker, T. Design of car frontal protection system using neural network and genetic algorithm. *Mechanika* **2012**, *18*, 453–460. [CrossRef]
11. Kers, J.; Majak, J.; Goljandin, D.; Gregor, A.; Malmstein, M.; Vilsaar, K. Extremes of apparent and tap densities of recovered GFRP filler materials. *Compos. Struct.* **2010**, *92*, 2097–2101. [CrossRef]
12. Kers, J.; Majak, J. Modelling a new composite from a recycled GFRP. *Mech. Compos. Mater.* **2008**, *44*, 623–632. [CrossRef]
13. Abdelhamid, A.A.; Towfek, S.K.; Khodadadi, N.; Alhussan, A.A.; Khafaga, D.S.; Eid, M.M.; Ibrahim, A. Waterwheel Plant Algorithm: A Novel Metaheuristic Optimization Method. *Processes* **2023**, *11*, 1502. [CrossRef]
14. Agushaka, J.O.; Ezugwu, A.E. Initialisation Approaches for Population-Based Metaheuristic Algorithms: A Comprehensive Review. *Appl. Sci.* **2022**, *12*, 896. [CrossRef]
15. Louzazni, M.; Khouya, A.; Amechnoue, K.; Gandelli, A.; Mussetta, M.; Crăciunescu, A. Metaheuristic Algorithm for Photovoltaic Parameters: Comparative Study and Prediction with a Firefly Algorithm. *Appl. Sci.* **2018**, *8*, 339. [CrossRef]
16. Rahman, A.; Sokkalingam, R.; Othman, M.; Biswas, K.; Abdullah, L.; Kadir, E.A. Nature-Inspired Metaheuristic Techniques for Combinatorial Optimization Problems: Overview and Recent Advances. *Mathematics* **2021**, *9*, 2633. [CrossRef]
17. Kisi, O. Machine Learning with Metaheuristic Algorithms for Sustainable Water Resources Management. *Sustainability* **2021**, *13*, 8596. [CrossRef]
18. Ikotun, A.M.; Almutari, M.S.; Ezugwu, A.E. K-Means-Based Nature-Inspired Metaheuristic Algorithms for Automatic Data Clustering Problems: Recent Advances and Future Directions. *Appl. Sci.* **2021**, *11*, 11246. [CrossRef]
19. Hashim, F.A.; Hussain, K.; Houssein, E.H.; Mabrouk, M.S.; Al-Atabany, W. Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems. *Appl. Intell.* **2020**, *51*, 1531–1551. [CrossRef]
20. Qayyum, A.; Ahmad, J.; Boulila, W.; Rubaiee, S.; Arshad; Masood, F.; Khan, F.; Buchanan, W.J. Chaos-Based Confusion and Diffusion of Image Pixels Using Dynamic Substitution. *IEEE Access* **2020**, *8*, 140876–140895. [CrossRef]
21. Hu, T. Discrete Chaos in Fractional Henon Map. *Appl. Math.* **2014**, *5*, 2243–2248. [CrossRef]
22. Setoudeh, F.; Dezhdar, M.M.; Najafi, M. Nonlinear analysis and chaos synchronization of a memristive-based chaotic system using adaptive control technique in noisy environments. *Chaos Solitons Fractals* **2022**, *164*, 112710. [CrossRef]
23. Thounaojam, U.S. Stochastic chaos in chemical Lorenz system: Interplay of intrinsic noise and nonlinearity. *Chaos Solitons Fractals* **2022**, *165*, 112763. [CrossRef]
24. Chuang, L.-Y.; Hsiao, C.-J.; Yang, C.-H. Chaotic particle swarm optimization for data clustering. *Expert Syst. Appl.* **2011**, *38*, 14555–14563. [CrossRef]
25. Tahir, M.; Tubaishat, A.; Al-Obeidat, F.; Shah, B.; Halim, Z.; Waqas, M. A novel binary chaotic genetic algorithm for feature selection and its utility in affective computing and healthcare. *Neural Comput. Appl.* **2020**, *34*, 11453–11474. [CrossRef]
26. Yang, L.; Li, K.; Zhang, W.; Ke, Z.; Xiao, K.; Du, Z. An improved chaotic ACO clustering algorithm. In Proceedings of the 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, UK, 28–30 June 2018; pp. 1642–1649.
27. Masdari, M.; Barshande, S.; Ozdemir, S. CDABC: Chaotic discrete artificial bee colony algorithm for multi-level clustering in large-scale WSNs. *J. Supercomput.* **2019**, *75*, 7174–7208. [CrossRef]

28. Kharrich, M.; Selim, A.; Kamel, S.; Kim, J. An effective design of hybrid renewable energy system using an improved Archimedes Optimization Algorithm: A case study of Farafra, Egypt. *Energy Convers. Manag.* **2023**, *283*, 116907. [CrossRef]
29. Akdag, O. A Improved Archimedes Optimization Algorithm for multi/single-objective Optimal Power Flow. *Electr. Power Syst. Res.* **2022**, *206*, 107796. [CrossRef]
30. Nassef, A.M.; Abdelkareem, M.A.; Maghrabie, H.M.; Baroutaji, A. Metaheuristic-Based Algorithms for Optimizing Fractional-Order Controllers—A Recent, Systematic, and Comprehensive Review. *Fractal Fract.* **2023**, *7*, 553. [CrossRef]
31. Chu, H.; Yi, J.; Yang, F. Chaos Particle Swarm Optimization Enhancement Algorithm for UAV Safe Path Planning. *Appl. Sci.* **2022**, *12*, 8977. [CrossRef]
32. Valencia-Ponce, M.A.; González-Zapata, A.M.; de la Fraga, L.G.; Sanchez-Lopez, C.; Tlelo-Cuautle, E. Integrated Circuit Design of Fractional-Order Chaotic Systems Optimized by Metaheuristics. *Electronics* **2023**, *12*, 413. [CrossRef]
33. Adeyemi, V.-A.; Tlelo-Cuautle, E.; Perez-Pinal, F.-J.; Nuñez-Perez, J.-C. Optimizing the Maximum Lyapunov Exponent of Fractional Order Chaotic Spherical System by Evolutionary Algorithms. *Fractal Fract.* **2022**, *6*, 448. [CrossRef]
34. Wang, S.; Rao, H.; Wen, C.; Jia, H.; Wu, D.; Liu, Q.; Abualigah, L. Improved Remora Optimization Algorithm with Mutualistic Strategy for Solving Constrained Engineering Optimization Problems. *Processes* **2022**, *10*, 2606. [CrossRef]
35. Wang, W.; Tian, J.; Wu, D. An Improved Crystal Structure Algorithm for Engineering Optimization Problems. *Electronics* **2022**, *11*, 4109. [CrossRef]
36. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
37. Li, Y.; Lin, X.; Liu, J. An Improved Gray Wolf Optimization Algorithm to Solve Engineering Problems. *Sustainability* **2021**, *13*, 3208. [CrossRef]
38. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
39. Fu, Y.; Zhou, M.; Guo, X.; Qi, L.; Sedraoui, K. Multiverse Optimization Algorithm for Stochastic Biobjective Disassembly Sequence Planning Subject to Operation Failures. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 1041–1051. [CrossRef]
40. Duman, S.; Güvenç, U.; Sönmez, Y.; Yörükeren, N. Optimal power flow using gravitational search algorithm. *Energy Convers. Manag.* **2012**, *59*, 86–95. [CrossRef]
41. Sindhu, R.; Ngadiran, R.; Yacob, Y.M.; Zahri, N.A.H.; Hariharan, M. Sine–cosine algorithm for feature selection with elitism strategy and new updating mechanism. *Neural Comput. Appl.* **2017**, *28*, 2947–2958. [CrossRef]
42. Cui, Y.; Meng, X.; Qiao, J. A multi-objective particle swarm optimization algorithm based on two-archive mechanism. *Appl. Soft Comput.* **2022**, *119*, 108532. [CrossRef]
43. Li, Y.; Zhu, X.; Liu, J. An Improved Moth-Flame Optimization Algorithm for Engineering Problems. *Symmetry* **2020**, *12*, 1234. [CrossRef]
44. Bencherqui, A.; Daoui, A.; Karmouni, H.; Qjidaa, H.; Alfidi, M.; Sayyouri, M. Optimal reconstruction and compression of signals and images by Hahn moments and artificial bee Colony (ABC) algorithm. *Multimed. Tools Appl.* **2022**, *81*, 29753–29783. [CrossRef]
45. Tahiri, M.A.; Bencherqui, A.; Karmouni, H.; Jamil, M.O.; Sayyouri, M.; Qjidaa, H. Optimal 3D object reconstruction and classification by separable moments via the Firefly algorithm. In Proceedings of the 2022 International Conference on Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 18–20 May 2022; pp. 1–8. [CrossRef]
46. Tahiri, M.A.; Karmouni, H.; Azzayani, A.; Sayyouri, M.; Qjidaa, H. Fast 3D Image Reconstruction by Separable Moments based on Hahn and Krawtchouk Polynomials. In Proceedings of the 2020 Fourth International Conference on Intelligent Computing in Data Sciences (ICDS), Fez, Morocco, 21–23 October 2020.
47. Tahiri, M.A.; Karmouni, H.; Bencherqui, A.; Daoui, A.; Sayyouri, M.; Qjidaa, H.; Hosny, K.M. New color image encryption using hybrid optimization algorithm and Krawtchouk fractional transformations. *Vis. Comput.* **2022**, *39*, 6395–6420. [CrossRef]
48. Rabab, O.; Tahiri, M.A.; Bencherqui, A.; Amakdouf, H.; Jamil, M.O.; Qjidaa, H. Efficient Localization and Reconstruction Of 3D Objects Using The New Hybrid Squire Moment. In Proceedings of the 2022 International Conference on Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 18–20 May 2022; pp. 1–8. [CrossRef]
49. Tahiri, M.A.; Karmouni, H.; Sayyouri, M.; Qjidaa, H. 2D and 3D image localization, compression and reconstruction using new hybrid moments. *Multidimens. Syst. Signal Process.* **2022**, *33*, 769–806. [CrossRef]
50. Moody, G.B.; Mark, R.G. The impact of the MIT-BIH Arrhythmia Database. *IEEE Eng. Med. Biol. Mag.* **2001**, *20*, 45–50. [CrossRef]