Neural Computing and Applications (2024) 36:13381–13465
https://doi.org/10.1007/s00521-024-09533-0

ORIGINAL ARTICLE

MCHIAO: a modified coronavirus herd immunity-Aquila optimization algorithm based on chaotic behavior for solving engineering problems

Heba Selim[1] · Amira Y. Haikal[1] · Labib M. Labib[1] · Mahmoud M. Saafan[1]

Received: 29 May 2023 / Accepted: 22 January 2024 / Published online: 20 April 2024

Abstract

This paper proposes a hybrid Modified Coronavirus Herd Immunity Aquila Optimization Algorithm (MCHIAO) that compiles the Enhanced Coronavirus Herd Immunity Optimizer (ECHIO) algorithm and Aquila Optimizer (AO). As one of the competitive human-based optimization algorithms, the Coronavirus Herd Immunity Optimizer (CHIO) exceeds some other biological-inspired algorithms. Compared to other optimization algorithms, CHIO showed good results. However, CHIO gets confined to local optima, and the accuracy of large-scale global optimization problems is decreased. On the other hand, although AO has significant local exploitation capabilities, its global exploration capabilities are insufficient. Subsequently, a novel metaheuristic optimizer, Modified Coronavirus Herd Immunity Aquila Optimizer (MCHIAO), is presented to overcome these restrictions and adapt it to solve feature selection challenges. In this paper, MCHIAO is proposed with three main enhancements to overcome these issues and reach higher optimal results which are cases categorizing, enhancing the new genes' value equation using the chaotic system as inspired by the chaotic behavior of the coronavirus and generating a new formula to switch between expanded and narrowed exploitation. MCHIAO demonstrates it's worth contra ten well-known state-of-the-art optimization algorithms (GOA, MFO, MPA, GWO, HHO, SSA, WOA, IAO, NOA, NGO) in addition to AO and CHIO. Friedman average rank and Wilcoxon statistical analysis (p-value) are conducted on all state-of-the-art algorithms testing 23 benchmark functions. Wilcoxon test and Friedman are conducted as well on the 29 CEC2017 functions. Moreover, some statistical tests are conducted on the 10 CEC2019 benchmark functions. Six real-world problems are used to validate the proposed MCHIAO against the same twelve state-of-the-art algorithms. On classical functions, including 24 unimodal and 44 multimodal functions, respectively, the exploitative and explorative behavior of the hybrid algorithm MCHIAO is evaluated. The statistical significance of the proposed technique for all functions is demonstrated by the p-values calculated using the Wilcoxon rank-sum test, as these p-values are found to be less than 0.05.

Keywords Meta-heuristic algorithm · Optimization · Coronavirus · Herd immunity · Aquila · Chaos theory

1 Introduction

We face a large number of optimization challenges in engineering, for which optimization methods are required [1]. Optimization algorithms help us in a variety of ways in our daily lives. Optimization algorithms can be used to reduce expenses and faults in any system or to increase profits in a financial firm. Optimization algorithms can be classified into two main categories: deterministic algorithms and stochastic algorithms.

Deterministic algorithms pursue a firm procedure where for the starting point the deterministic algorithm will follow up the same path whenever we run the program. On the

✉ Mahmoud M. Saafan
saafan2007@mans.edu.eg

Heba Selim
heba_salem@mans.edu.eg

Amira Y. Haikal
amirayh@mans.edu.eg

Labib M. Labib
labib_essa@mans.edu.eg

[1] Computers and Control Systems Engineering Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt

other hand, stochastic algorithms are random in finding the optimum values. So, every time we run the program the solutions will be different [2]. Traditionally, deterministic algorithms are used to deal with optimization problems that are characterized as small dimensions and less complex problems. Despite their capability to reach an exact and specific solution to optimization problems, they experience some serious impasses, and they can easily fall into the local optimal solution [3]. We can say that stochastic algorithms can overcome deterministic algorithms' impasses. Heuristic optimization algorithms are methods that improve the efficiency of a search process by sacrificing completeness, examples of these methods are Nearest Neighbor (Greedy Algorithm) and local search algorithms. Although they can reach a "near-optimal" solution in a short time and consume a small memory space, they cannot guarantee to reach the optimal solution or reach a solution that is "good enough" [4]. As a result, meta-heuristic algorithms emerged that combine heuristic techniques in an upper-level framework to explore a search space efficiently. Metaheuristic-based algorithms provide a framework for optimization that employs stochastic features that are controlled by tunable parameters with knowledge acquisition operators to improve the current solution until the best possible solution is found [5]. Interestingly enough, nature-inspired phenomena helped in generating the most meta-heuristic algorithms, which can be divided into categories as listed in Fig. 1 [6].

Nevertheless, there is not a single optimization algorithm that can operate efficiently with all classes of optimization issues "according to the No Free Lunch (NFL) Theorem" [7]. The community-based behavior of animal flocks is often an inspiration for swarm-based algorithms. The ability to work together to survive is the main asset of such a class. Particle swarm optimization (PSO), which imitates the social behavior of flocking birds, is one of the first swarm-based algorithms [8]. The particles (solutions) search for the best position in their surroundings (search space) "global best". Throughout the flight, the first-rate places (local best) on the road to the optimum locations are noted. The grasshopper optimization algorithm (GOA) is a novel swarm intelligence algorithm inspired by grasshoppers' natural foraging and swarming behavior [9]. Saremi et al. presented the GOA algorithm in [9], which is a fascinating new swarm intelligence system that simulates grasshopper foraging and swarming behaviors. Grasshoppers are insects that are well-known as pests that wreak havoc on agricultural production and agriculture [9]. Their life cycle is divided into two stages: nymph and adulthood. Small steps and gradual movements describe the nymph phase, but long-range and rapid movements represent the maturity phase. The intensification and diversification phases of GOA are defined by nymph and adult movements. Moth-Flame Optimization (MFO) algorithm is a new nature-inspired algorithm inspired by moths' transverse orientation mechanism [10]. MFO uses a set of moths to search the decision space, reporting fitness functions at each time step and tagging the best solution with a flame. The movement of moths is based on their flames spiraling around them in a spiral direction. The Marine Predators Algorithm (MPA) is another well-known nature-inspired optimization algorithm that follows the rules that regulate optimal foraging strategy and predator–prey encounter rates in marine ecosystems [11].

The main inspiration for MPA is a widely used foraging strategy in ocean predators, specifically Lévy and Brownian movements, as well as an optimal encounter rate policy in predator–prey biological interactions. While seeking food in a prey-scarce environment, marine predators (such as sharks, tunas, and marlines) use the Lévy strategy, but when foraging in a prey-abundant environment, the pattern is frequently shifted to Brownian motion [12]. In a biological interaction between predator and prey, the optimum encounter rate policy is also determined by the sort of movement that each predator/prey makes and the velocity ratio of prey to predator [13]. Grey Wolf Optimizer (GWO) is a new type of optimization approach for swarm intelligence inspired by grey wolves (Canis lupus) [14]. The GWO algorithm is simulated after the natural hierarchy of authority and hunting mechanism of grey wolves. For mimicking the leadership hierarchy, four types of grey wolves are used: alpha, beta, delta, and omega. Furthermore, the three basic processes of hunting are implemented: seeking prey, encircling prey, and attacking prey. Heidari et al. established Harris Hawks Optimization (HHO), a novel optimization algorithm inspired by the simulation of the behavior of Harris Hawks [15]. The algorithm's base is the simultaneous attack method of Harris Hawks from numerous directions. For the time being, comparing nature-inspired human-based algorithms such as HSA to other nature-inspired algorithms gives more pleasing results. The Salp swarm algorithm (SSA) is a newly developed bio-inspired optimization algorithm based on the swarming mechanism of salps, which was first presented in 2017 [16]. SSA is an evolutionary algorithm that mimics the natural swarming mechanism of salps. In 2016, the whale optimization algorithm (WOA) was proposed by Mirjalili and Lewis [17]. It's a swarm intelligence optimization system that mimics the hunting behavior of humpback whales. Metaheuristics solve intractable optimization problems. Since the initial metaheuristic was proposed, several new algorithms have been created [18–25]. The algorithm's main goal is to solve the objective problem by mimicking the predatory behavior of a whale. In this paper, a modification in the nature-inspired and human-based optimization technique which is known

as "Coronavirus Herd Immunity Optimizer" is proposed. In the year 2019 in China, an evolution of a severe acute respiratory syndrome which is called coronavirus disease (COVID-19) caused a considerable global outbreak that gave rise to a major public health issue [26]. The coronavirus pandemic and specifically the herd immunity against COVID-19 gave the main concepts to the Coronavirus Herd Immunity Optimizer (CHIO) algorithm. Outperforming several other biologically inspired algorithms, the Coronavirus Herd Immunity Optimizer (CHIO) is a competitive human-based optimisation tool. Compared to other optimization approaches, CHIO performed efficiently. Nevertheless, CHIO can only handle local optima, which limits its accuracy in solving intricate global optimization problems. Consequently, the Coronavirus Herd Immunity Optimizer (CHIO) algorithm has a poor rate of convergence during the iterative process and is easily prone to falling into a local optimum in high-dimensional space, which is why we propose Enhanced Coronavirus Herd Immunity Optimizer (ECHIO). However, considering that AO has great local exploitation capabilities, its global exploration skills are restricted. The AO algorithm may exhibit early convergence and poor global exploration when used to optimize difficult, high-dimensional engineering problems. When optimizing complex multidimensional problems, the AO runs into challenges, such as poor exploration efficiency and poor convergence behavior. Then, in order to overcome these drawbacks, such as CHIO's poor exploitation skills and the AO algorithm's insufficient exploration capabilities, the Modified Coronavirus Herd Immunity Aquila Optimizer (MCHIAO), a novel metaheuristic optimizer, is presented. It is modified to handle feature selection issues. Three different contributions have been applied to the CHIO algorithm to increase exploration efficiency by keeping the ideal balance between it and the search's exploitation of an optimal solution. The main contributions of the current work can be summarized as:

- Cases categorizing according to the status vector.
- Enhancement of the gene's equations for new genes applying chaotic maps.
- Generating a new formula for switching between narrowed and expanded exploitation.
- Validate the proposed MCHIAO through testing against 12 state-of-the-art algorithms.
- Wilcoxon statistical analysis and Friedman average rank are conducted to validate the proposed MCHIAO.
- On 130 benchmark functions, the proposed hybrid algorithm's performance is evaluated. The benchmarks comprise 23 standard benchmark functions, 29 CEC-2017 test functions, 10 CEC-2019 test functions, 24 unimodal functions, and 44 multimodal functions.
- Test the proposed MCHIAO algorithm on six real-world engineering problems.

The sections of this paper are organized in the following order:

Section 2 presents the CHIO algorithm, the inspiration beyond it, and its main concepts. The Aquila algorithm, its origins of inspiration, and its core perspectives are all

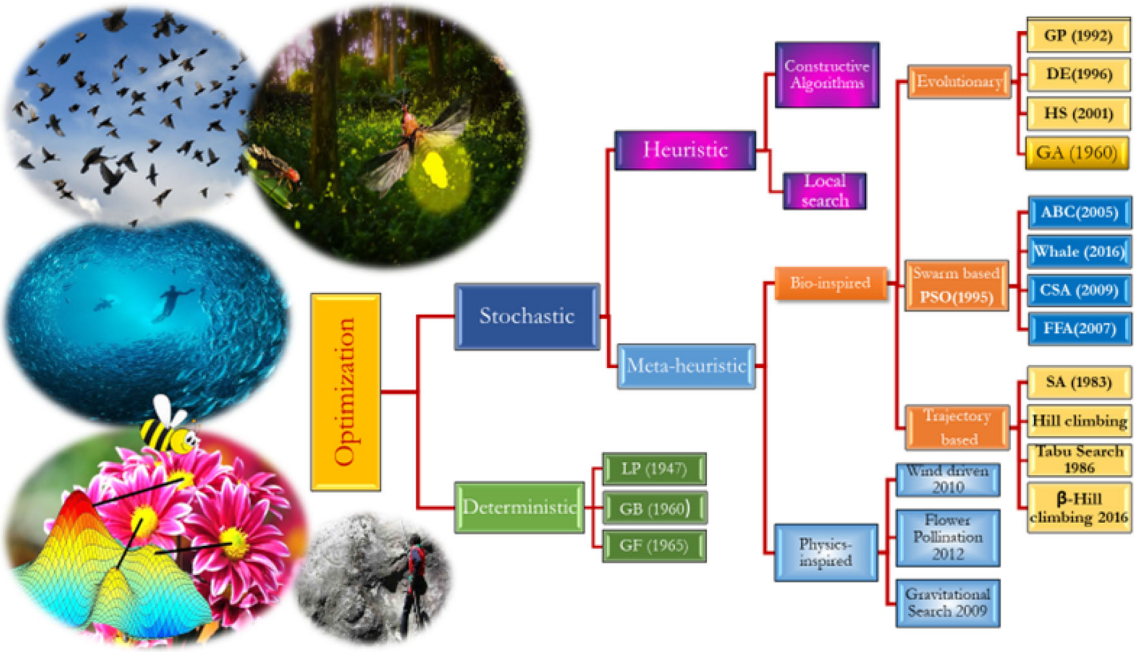


Fig. 1 Optimization methods

presented in Sect. 3. Section 4 outlines the proposed hybrid algorithm MCHIAO and the details of the contributions. Section 5 presents the results of a comparison between the MCHIAO algorithm, CHIO algorithm, Aquila algorithm, and other state-of-the-art algorithms applied to different benchmark functions with different dimensions and as well as 6 common real-world problems. Section 6 shows the conclusion and direction for future work.

2 Corona virus herd immunity optimization algorithm

2.1 Inspiration

The COVID-19 pandemic has been a global threat. Herd immunity is a highly effective method to tackle the COVID-19 pandemic [27]. The resistance to the spread of an infectious illness within a community or herd is known as "Herd Immunity." On March 13th, the British Government's chief scientific adviser, Sir Patrick Vallance pointed out that waiting for herd immunity would result in getting 60% of the population being infected with COVID-19. Herd immunity occurs when a considerable section of a community gets immune to a disease stopping its spread from one person to another. There are two ways to achieve herd immunity for COVID-19:

I. Vaccines
II. Infection

Without causing suffering or illness, vaccines create immunity and protect people against pandemics. Despite the effectiveness of the concept of vaccination, it has a serious drawback, some people may reject taking vaccines because of fear of the possible side effects and risks. Another drawback of the vaccine is that it may fade over time and requires revaccination. Natural infection is an effective path to obtaining herd immunity. It can be achieved when an appropriate portion of people in the community has recovered from a specific disease and have evolved antibodies that would work against future infection [28]. Herd immunity is suggested to be one of the mechanisms to control and slow down the COVID-19 widespread disease. Bear in mind that the CHIO algorithm put in an application "the survival of the fittest" principle from the Darwinian Theory. Herd immunity can be described as the indirect protection from contagion given to susceptible human beings when an appropriate proportion of immune persons exist in a population. The population-level effect is frequently considered in the vaccination programs which target to establish herd immunity for those who cannot be vaccinated and still be protected against contra disease. The susceptible-infectious-recovered (SIR) model has been widely utilized to survey the dynamic growth of COVID-

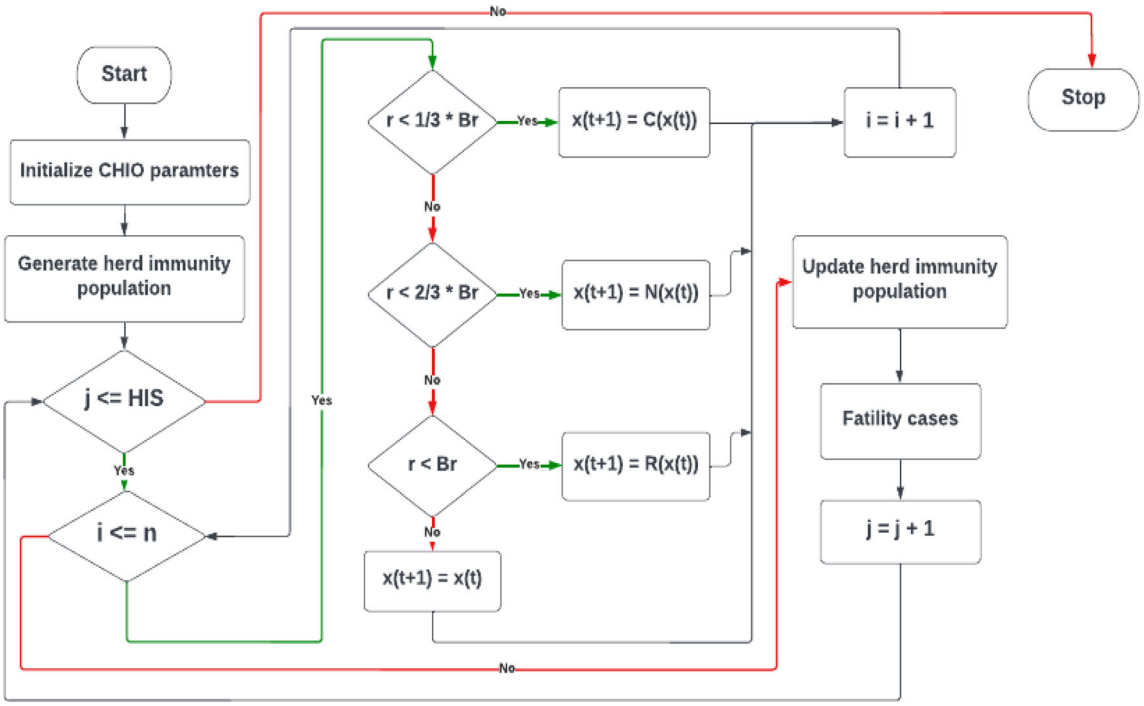


Fig. 2 CHIO flowchart [30]

19 in a large population [29]. The SIR model classifies the population into three categories:

I. Susceptible individuals: The virus has not yet infected these participants, but when a susceptible individual and an infectious individual come into infectious contact without following the recommended social distancing rules, they can be infected.
II. Infected individuals: Participants in this category have been verified to be infected, and they can spread the virus to others who are susceptible.
III. Recovered (immune) individuals: These are the individuals who are protected against the virus either because they have taken the vaccine or because they have been infected with the virus and recovered.

2.2 Mathematical model of CHIO [30]

In this section, the mathematical model of the CHIO algorithm is illustrated in steps

2.2.1 Initializing CHIO parameters

$$\min f(x) \quad x \in [lb, ub] \tag{1.1}$$

where $f(x)$ is the objective function (immunity rate) and it is calculated for each case (individual).

$$x = (x_1, x_2, x_3, \ldots, x_n) \tag{1.2}$$

where x_i represents the gene (decision variable) indexed by i, while n represents the overall amount of decision variables in every case.

$$x_i \in [lb_i, ub_i] \tag{1.3}$$

where lb_i is the lower bound of the gene x_i and ub_i is the upper bound of it.

The CHIO algorithm has two control parameters:

- Basic reproduction Rate (BR_r): By spreading the virus among individuals, it controls the CHIO operators.
- Maximum infected case age (Max_{Age}): Determine the infected cases' status when a case reaches Max_{Age}, it is either recovered or died.

The CHIO algorithm has another four parameters:

- *Max_Itr*: Represents the maximum number of iterations.
- C_0: This value represents the number of primary infected cases, which is typically one.
- *HIS* : Represents the population size.
- n : Represents the problem dimensionality.

2.2.2 Generating herd immunity population:

Randomly CHIO generates a set of individuals (cases) as many as *HIS*. The set of the generated individuals is stored as a two-dimensional matrix of size $n \times HIS$ in the herd immunity population (*HIP*) as follows:

$$HIP = \begin{bmatrix} x_1^1 & x_2^1 & \ldots & x_n^1 \\ x_1^2 & x_2^2 & \ldots & x_n^2 \\ x_1^{HIS} & x_2^{HIS} & \ldots & x_n^{HIS} \end{bmatrix} \tag{1.4}$$

where each row represents a case x^j, which is calculated as follows:

$$x_i^j = lb_i + (ub_i - lb_i) \times U(0,1) \quad \forall i = 1, 2, 3, \ldots, n.$$
$$\forall j = 1, 2, 3, \ldots, HIS. \tag{1.5}$$

For each case, the objective function is calculated using Eq. (1.1).

The fitness for each search agent is set as follows:

$$S_j = 0 \quad \forall j = 1, 2, 3, \ldots, HIS.$$
$$A_j = 0 \quad \forall j = 1, 2, 3, \ldots, HIS.$$

where S is the status vector of length *HIS* in the *HIP* for every case initiating from zero which means a case that is susceptible or rather one that represents an infected case.

2.2.3 Coronavirus herd immunity evolution

In this section, the main improvement loop of CHIO is presented. The gene (x_i^j) of the case (x^j) either remains the same or becomes classified as an infected, susceptible, or immune case according to the percentage of the basic reproduction rate (BR_r) as follows:

$$x_i^j(t+1) \leftarrow \begin{cases} x_i^j(t) & r \geq BR_r \\ C\left(x_i^j(t)\right) & r < \frac{1}{3} \times BR_r // Infected\ case \\ N\left(x_i^j(t)\right) & r < \frac{2}{3} \times BR_r // susceptible\ case \\ R\left(x_i^j(t)\right) & r < BR_r // immuned\ case \end{cases} \tag{1.6}$$

where r is a random value between 0 and 1.

- For infected cases:

$$x_i^j(t+1) = C(x_i^j(t))$$
$$C\left(x_i^j(t)\right) = x_i^j(t) + r \times (x_i^j(t) - x_i^c(t)) \tag{1.7}$$

where $x_i^j(t+1)$ is the new gene and the value $x_i^c(t)$ is chosen randomly based on the status vector (S) from any infected case x^c as $c = \{i | S_i = 1\}$ and r is the random number between 0 and 1.

- For susceptible cases:

$$x_i^j(t+1) = N(x_i^j(t))$$
$$N\left(x_i^j(t)\right) = x_i^j(t) + r \times (x_i^j(t) - x_i^m(t)) \quad (1.8)$$

where $x_i^j(t+1)$ is the new gene and the value $x_i^m(t)$ is chosen randomly based on the status vector (S) from any susceptible case x^m as $m = \{i|S_i = 0\}$ and r is the random number between 0 and 1.

- For immune cases:

$$x_i^j(t+1) = R(x_i^j(t))$$
$$R\left(x_i^j(t)\right) = x_i^j(t) + r \times (x_i^j(t) - x_i^v(t)) \quad (1.9)$$

where $x_i^j(t+1)$ is the new gene and the value $x_i^v(t)$ is chosen randomly based on the status vector (S) from any immuned case x^v as $f(x^v) = argmin_{j\{k|S_k=2\}} f(x^j)$.

2.2.4 Update the herd immunity population:

For each generated case $x^j(t+1)$, the immunity rate $f(x_i^j(t+1))$ is calculated and if the generated case $x^j(t+1)$ is better than the present case $x^j(t)$ such as$f(x^j(t+1)) < f(x^j(t))$, the current case is replaced by the generated case.

If the status vector ($S_j = 1$), the age vector (A_j) is increased by one.

Based on the herd immunity threshold, for each case $x^j(t)$ the status vector (S^j) is updated which employs the following equation:where, $\triangle f(x) = \frac{\sum_{i=1}^{HIS} f(x_i)}{HIS}$.

$$S_j \leftarrow \begin{cases} 1 & f(x^j(t+1)) < \dfrac{f(x^j(t+1))}{\triangle f(x)} \wedge S_j = 0 \wedge is\_Corona\left(x^j(t+1)\right) \\ 2 & f(x^j(t+1)) > \dfrac{f(x^j(t+1))}{f(x)} \wedge S_j = 1 \end{cases} \quad (1.10)$$

Where $is\_Corona(x^j(t+1))$ represents a binary value that equals one in case of the new case $x^j(t+1)$ gained a value from the infected case and $\Delta f(x)$ represents the mean value of the population immune rates.

2.2.5 Fatality cases

When the immunity rate $(f(x^j(t+1)))$ of the current infected case does not improve for a specific time of iterations which is specified by the parameter Max_Age such as $A_j \geq Max\_Age$ then this case is considered dead. Then, it is regenerated from scratch using the following equation:

$$x_i^j(t+1) = lb_i + (ub_i - lb_i) \times U(0,1) \quad (1.11)$$

where $\forall i = 1, 2, \ldots, n$

Over and above, A_j and S_j are set to zero.

2.2.6 Stop criterion

CHIO repeats the main loop until the maximum number of iterations is achieved. The entire number of immune cases in addition to the susceptible cases command the population and the infected individuals disappear. Figure 2 represents the flowchart of CHIO. The pseudo-code of the CHIO is described in algorithm 1.

Algorithm 1 CHIO pseudo-code [30].

{———- Step 1: Initialize the CHIO parameters ———-}
Initialize the parameters (HIS, S_r and Max_{age}).
{———- Step 2: Generate herd immunity population ———-}
$x_i^j = lb_i + (ub_i - lb_i) \times U(0,1), \quad \forall i = 1,2,\ldots,n$, and $\forall j = 1,2,\ldots,HIS$
Calculate the fitness of each search agent
Set $S_j = 0 \quad \forall j = 1,2,3,\ldots,HIS.$
Set $A_j = 0 \quad \forall j = 1,2,3,\ldots,HIS.$
{———- Step 3: Herd immunity evolution ———-}
while $(t \le Max\_itr)$ **do**
  for $j = 1\ to\ HIS$ **do**
    $is\_Corona\left(x^i(t+1)\right) = false$
    for $i = 1\ to\ N$ **do**
      if $(r < \frac{1}{3} \times \boldsymbol{BR_r})$ **then**
        $x_i^j(t+1) = C(x_i^j(t))$
        $C\left(x_i^j(t)\right) = x_i^j(t) - r \times (x_i^j(t) - x_i^c(t))$
        $is\_Corona\left(x^i(t+1)\right) = true$
      else if $(r < \frac{2}{3} \times \boldsymbol{BR_r})$ **then**
        $x_i^j(t+1) = N(x_i^j(t))$
        $N\left(x_i^j(t)\right) = x_i^j(t) - r \times (x_i^j(t) - x_i^m(t))$
      else if $(r < BR_r)$ **then**
        $x_i^j(t+1) = R(x_i^j(t))$
        $R\left(x_i^j(t)\right) = x_i^j(t) - r \times (x_i^j(t) - x_i^v(t))$
      else
        $x_i^j(t+1) = x_i^j(t)$
      end if
    end for
    {———- Step 4: Update herd immunity population ———-}
    if $(f(x^j(t+1)) \le f(x^j(t)))$ **then**
      $x_i^j(t) = x_i^j(t+1)$
    else
      $A^j = A^j + 1$
    end if
    if $f\left(x^j(t+1)\right) < \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 0 \wedge is\_Corona(x^j(t+1))$ **then**
      $S_j = 1$
      $A_j = 1$
    end if
    if $f\left(x^j(t+1)\right) > \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 1$ **then**
      $S_j = 2$
      $A_j = 0$
    end if
    {———- Step 5: Fatality condition ———-}
    if $((A_j \ge Max\_Age)\ \wedge\ (S_j == 1)$ **then**
      $x_i^j = lb_i + (ub_i - lb_i) \times U(0,1), \quad \forall i = 1,2,\ldots,N$
      $S_j = 0$
      $A_j = 0$
    end if
  end for
  $t = t + 1$
end while

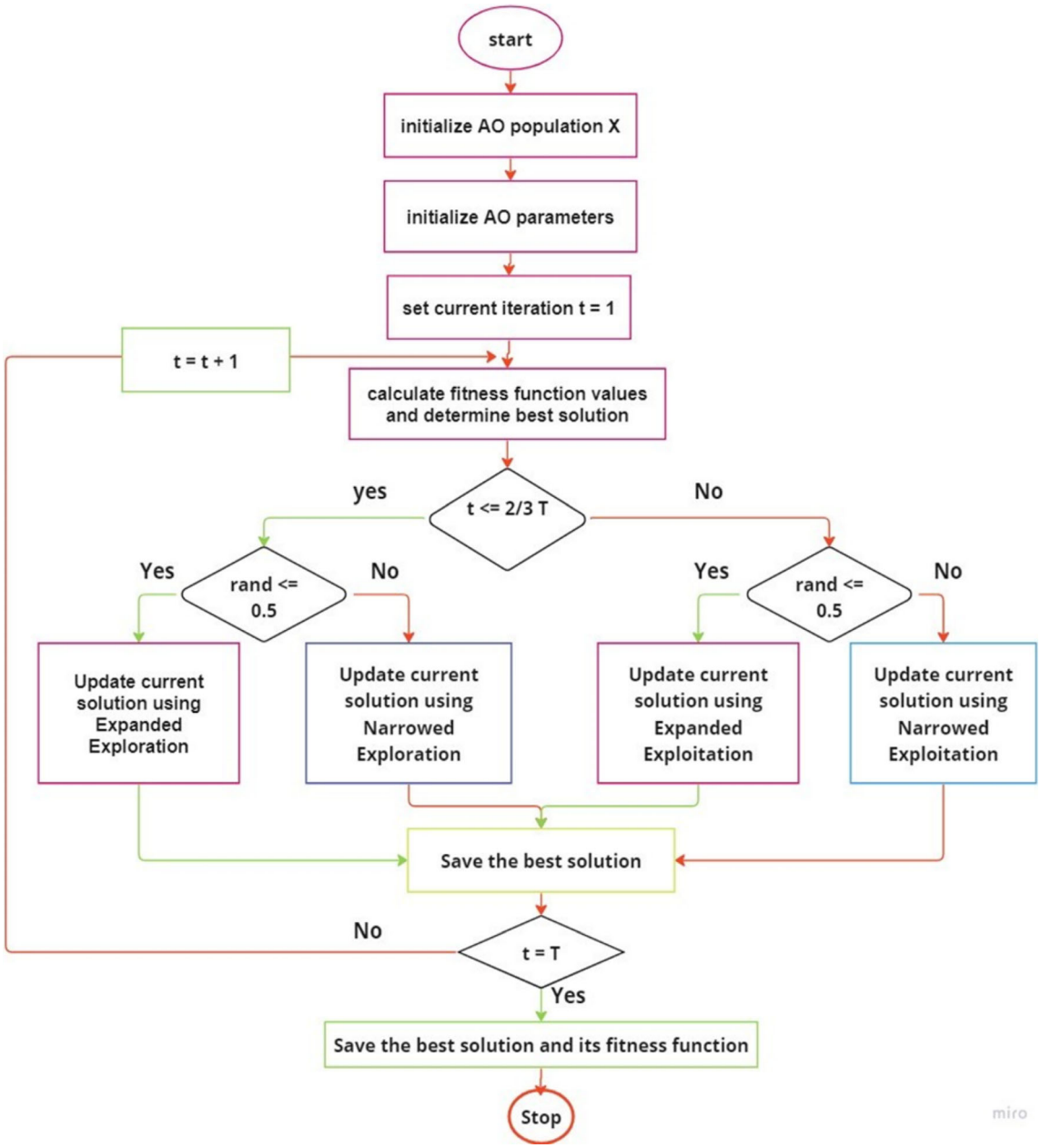


Fig. 3 Aquila flowchart [31]

3 Aquila optimization algorithm [31]

3.1 Inspiration

One of the most well-known raptors in the Northern Hemisphere is the Aquila. Aquila catches a variety of prey, primarily rabbits, hares, deeps, marmots, squirrels, and other ground animals, using its speed, agility, strong feet, and long, pointed talons. Aquila may be seen in nature, along with their peculiar behaviors.

Aquila may maintain 200 km2 or larger holdings. They build large nests on mountains and other high areas. They reproduce in the spring, and because they are monogamous, they are likely to remain together for the rest of their lives. Aquila is one of the most researched birds in the world because of its bold hunting behavior. Aquila hunt squirrels, rabbits, and many other creatures with their speed

Table 1 Chaotic maps

No	Name	Chaotic map
1	Chepyshev	$X_{i+1} = cos(icos^{-1}(X_i))$
2	Circle	$X_{i+1} = mod(X_i + b - \left(\frac{a}{2\pi}\right)\sin(2\pi X_k), 1)$
3	Gauss/mouse	$X_{i+1} = \begin{matrix} 1X_i = 0 \\ \frac{1}{mod(X_i, 1)} otherwise \end{matrix}$
4	Iterative	$X_{i+1} = sin(a\pi/X_i)$
5	Logistic	$X_{i+1} = aX_i(1 - X_i)$
6	Piecewise	$X_{i+1} = \begin{matrix} \frac{X_i}{P} 0 \le X_i < P \\ \frac{X_i - P}{0.5 - P} p \le X_i < 0.5 \\ \frac{1 - P - X_i}{0.5 - P} 0.5 \le X_i < 1 - P \\ \frac{1 - X_i}{P} 1 - P \le X_i < 1 \end{matrix}$
7	Tent	$X_{i+1} = \begin{matrix} \frac{X_i}{0.7} X_i < 0.7 \\ \frac{10}{3}(1 - X_i) X_i \ge 0.7 \end{matrix}$
8	Sine	$X_{i+1} = \frac{a}{4} sin(\pi X_i)$
9	Singer	$X_{i+1} = \pi(7.86X_i - 23.31X_i^2 + 28.75X_i^3 - 13.302875X_i^4), \pi = 1.07$
10	Sinusoidal	$X_{i+1} = aX_i^2 sin(\pi X_i)$

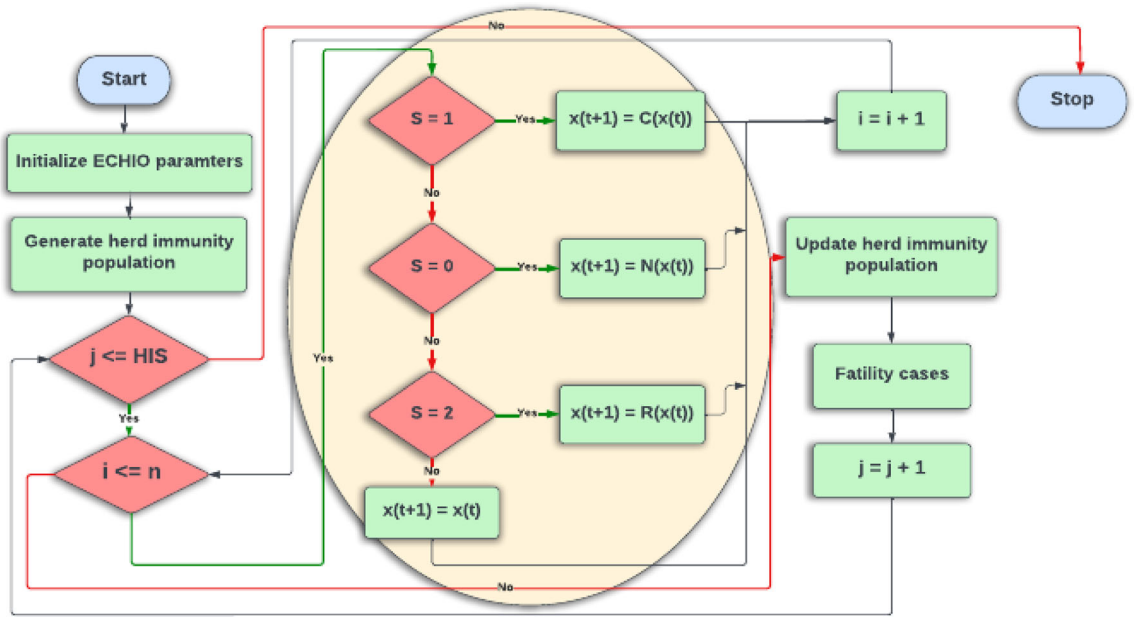


Fig. 4 ECHIO flowchart

and razor-sharp talons. Even mature deer have been known to be attacked by them.

The Aquila is known to primarily employ four different hunting techniques, each of which has several notable variations. Depending on the circumstances, most Aquila can deftly and swiftly switch between different hunting techniques. The following statements describe Aquila's hunting techniques.

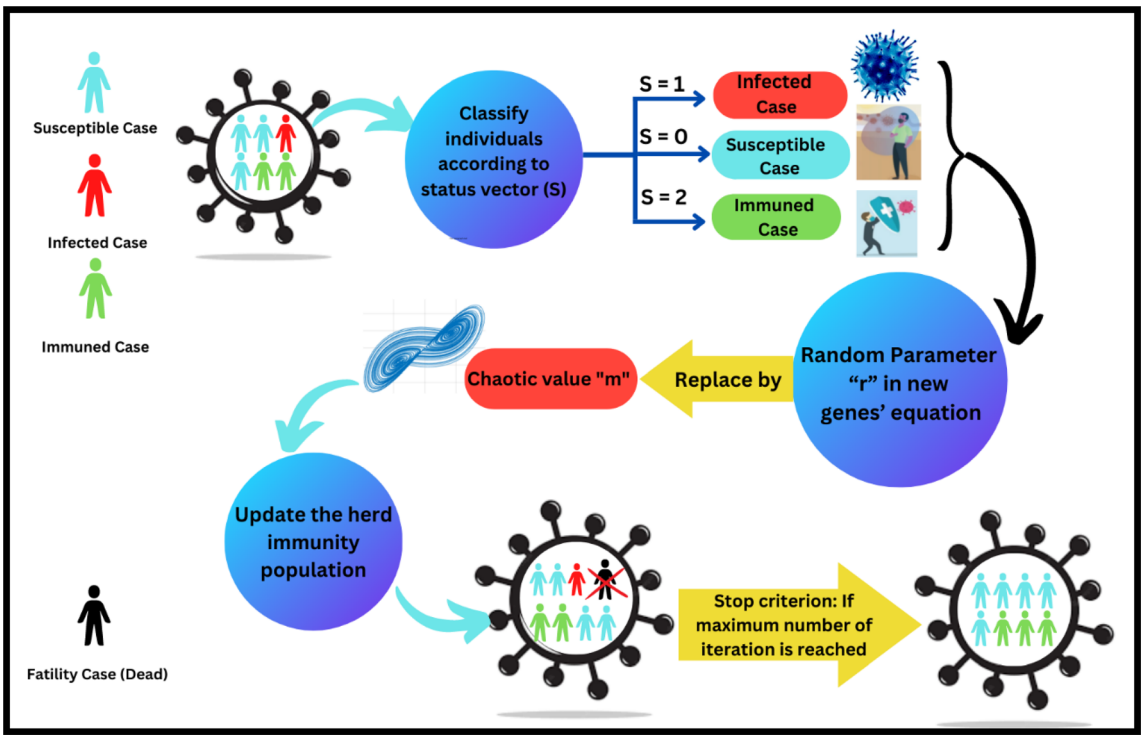


Fig. 5 Description of the proposed ECHIO

- In the first method, the Aquila hunts birds in flight using the first technique, high soar with a vertical stoop, in which it soars far above the ground.
- The Aquila undertakes a lengthy, low-angled glide after exploring its prey, increasing its speed as the wings continue to shut. The Aquila must have a height advantage over its prey for this strategy to be effective. To simulate a thunderclap just before the encounter, the wings and tail are opened, and the feet are propelled forward to seize the prey.
- Aquila is known for using the second technique, contour flying with brief glide attack, the most frequently. In this technique, the Aquila climbs at a low height above the ground. The target is then pursued relentlessly, whether it is flying or running. This strategy is advantageous for pursuing seabirds, nesting grouse, or ground squirrels.
- A low flight and a gradual descending assault are the third strategies. In this, the Aquila descends to the ground and then advances on the victim. The Aquila chooses its victim and attempts to enter by landing on the neck and back of the animal. For sluggish prey, such as rattlesnakes, hedgehogs, foxes, and tortoises, as well as any species lacking an escape reaction, this hunting technique is used.
- The Aquila walks on the ground while attempting to draw its prey in the fourth technique, known as walking and grabbing prey. It is used to remove the young of big prey, such as sheep or deer, from the coverage area.

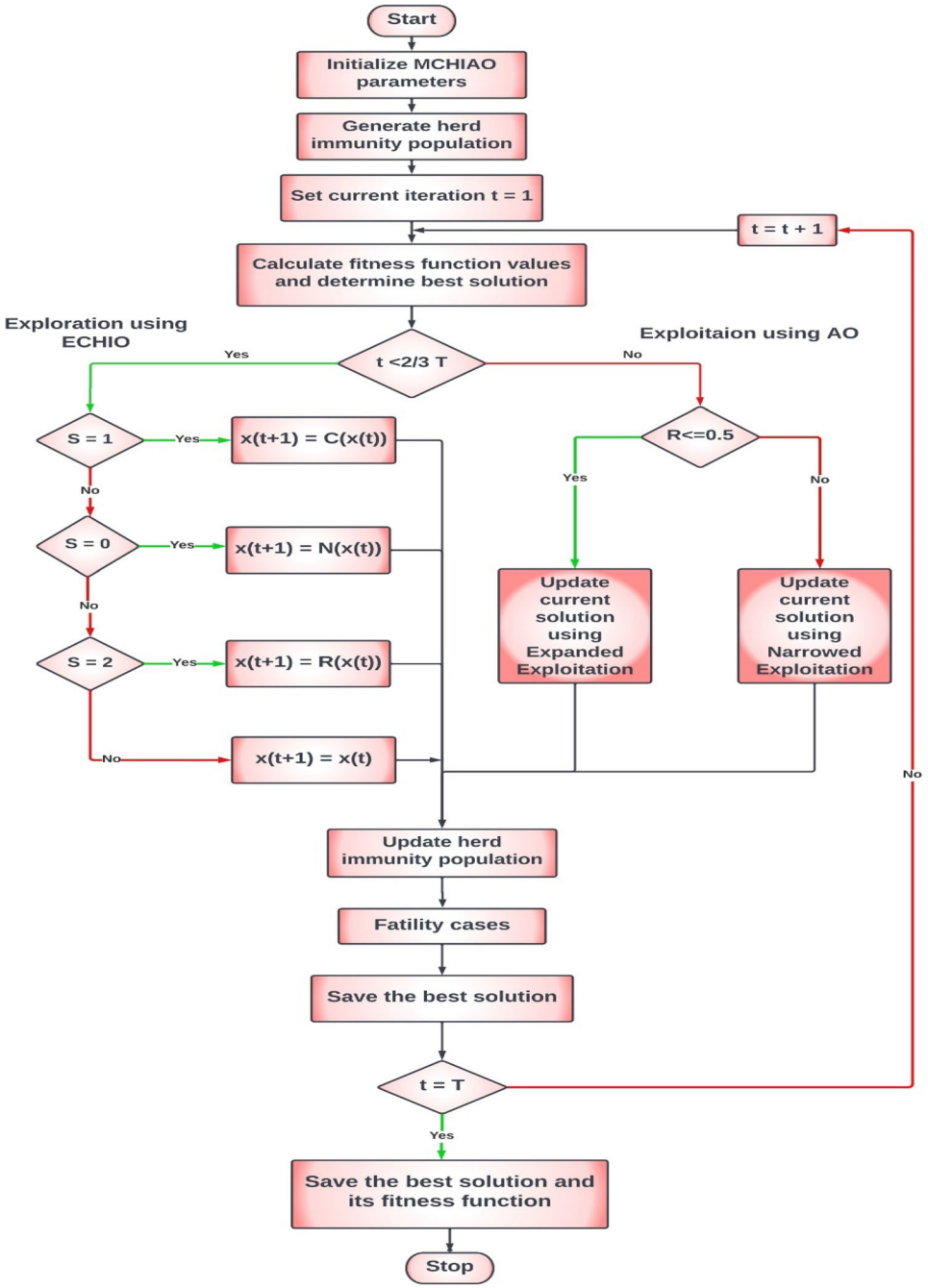
As a whole, Aquila is one of the most knowledgeable and proficient hunters—possibly second only to humans. The methods mentioned above formed the primary sources of inspiration for the suggested AO algorithm. These processes are modeled in the AO in the next subsections.

3.2 Mathematical model of AO

3.2.1 Generating AO population

The population of candidate solutions (X) as shown in Eq. (2.1), which is created stochastically between the upper limit (UB) and lower bound (LB) of the given problem, serves as the starting point for the optimization procedure in the population-based approach known as AO. In each

Fig. 6 MCHIAO flowchart



iteration, the best answer so far is roughly decided to be the best candidate.

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & x_{1,Dim-1} & x_{1,Dim} \\ \vdots & \cdots & x_{i,j} & \cdots & \cdots \\ x_{N,1} & \cdots & x_{N,j} & x_{N,Dim-1} & x_{1N,Dim} \end{bmatrix} \tag{2.1}$$

where N is the total number of candidate solutions (population), X_i indicates the decision values (positions) of the i^{th} solution, X is the set of current candidate solutions, which are generated randomly by applying Eq. (2.2), and Dim denotes the problem's dimension size.

$$X_{ij} = rand \times (UB_j - LB_j) + LB_j, \quad i = 1, 2, ..., N \\ j = 1, 2, ..., Dim \tag{2.2}$$

When a rand is a random number, LB_j refers to the j^{th} lower bound, and UB_j represents to the j^{th} upper limit of the given issue.

3.2.2 Aquila algorithm evolution

In this section, the Aquila optimization algorithm's main steps are presented. The AO algorithm mimics Aquila's behavior during hunting by displaying the activities taken at each stage of the hunt. Thus, the four methods used in the proposed AO algorithm's optimization processes are high soar with a vertical stoop to select the search space; contour flight with a short glide attack to explore within a diverged search space; low flight with a slow descent attack to exploit within a converge search space; and walk and grab prey to swoop.

Based on this condition, if $t \leq (\frac{2}{3}) \times T$, the exploration steps will be thrilled; otherwise, the exploitation steps will be carried out, and the AO algorithm can switch from exploration steps to exploitation steps utilizing different behaviors.

Aquila behaviors are modeled as a mathematical optimization paradigm that chooses the optimum solution while taking into account several restrictions. The following is the mathematical representation of the AO.

3.2.2.1 Step 1: Expanded exploration (X_1)

In the initial method (X_1), the Aquila determines the ideal hunting location by high-flying with a vertical stoop after identifying the prey region. Here, the AO extensively explores from a high altitude to pinpoint the location of the prey in the search space. The mathematical representation of this behavior is given in Eq. (2.3).

$$X_1(t+1) = X_{best}(t) \times \left(1 - \frac{t}{T}\right) + (X_M(t) - X_{best}(t) \times rand), \tag{2.3}$$

where $\boldsymbol{X}_1(\boldsymbol{t}+1)$ is the result of the first search technique (X_1) for the answer to the next iteration of the problem t. The best answer up until the t^{th} iteration, $\boldsymbol{X_{best}(t)}$, represents the approximate location of the prey. The enlarged search (exploration) is managed through the number of iterations using the equation ($\frac{1-t}{T}$). The location means the value of the connected current solutions at the t^{th} iteration, or $\boldsymbol{X_M(t)}$, is determined using Eq. (2.4). A random number between 0 and 1 is called rand. The current iteration and the maximum number of iterations are represented, respectively, by t and T.

$$X_M(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t), \forall j = 1, 2, ..., Dim \tag{2.4}$$

where N is the number of potential solutions (population size) and Dim is the problem's dimension size.

3.2.2.2 Step 2: Narrowed exploration (X_2)

In the second technique (X_2), the Aquila circles over the intended prey, prepares the terrain, and then strikes after being spotted from a great height. Contour flying with a short glide attack is this technique.

Here, AO prepares for the attack by closely examining the chosen location of the intended victim. This behavior is represented quantitatively in Eq. (2.5).

$$X_2(t+1) = X_{best}(t) \times Levy(D) + X_R(t) + (y - x) \times rand \tag{2.5}$$

where $X_2(t+1)$ is the result of the second search technique (X_2) and represents the answer of the subsequent iteration of the problem t. Levy(D) is the levy flight distribution function that is determined using Eq. (2.6), where D is the dimension space.

At the i^{th} iteration, $X_R(t)$ is a random solution selected from the interval [1 N].

$$Levy(D) = s \times \frac{u \times \sigma}{|\upsilon|^{\frac{1}{\beta}}} \tag{2.6}$$

where u and υ are random numbers between 0 and 1, and s is a constant value set to 0.01. σ is determined by applying Eq. (2.7).

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sine(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}}\right) \tag{2.7}$$

where β is a constant with the value 1.5.

The spiral form in the search is presented in Eq. (3.5) using the variables y and x, which are computed as follows.

$$y = r \times \cos\theta \tag{2.8}$$

$$x = r \times \sin\theta \tag{2.9}$$

where,

$$\boldsymbol{r} = \boldsymbol{r_1} + \boldsymbol{U} \times \boldsymbol{D_1} \tag{2.10}$$

$$\theta = -\omega \times D_1 + \theta_1 \tag{2.11}$$

$$\theta_1 = \frac{3 \times \pi}{2} \tag{2.12}$$

For a set number of search cycles, $\boldsymbol{r}_1$ takes a value between 1 and 20, and $\boldsymbol{U}$ is an insignificant value fixed at 0.00565. D_1 Consists of integers from 1 to the search space's length (Dim), and ω is a tiny value fixed at 0.005.

3.2.2.3 Step 3: Expanded exploitation (X_3)

In the third approach (X_3), the Aquila descends vertically with an initial attack to ascertain the prey reaction after the prey region has been precisely designated and the Aquila is

prepared for landing and attacking. Low flying with gradual descending assault is the name given to this tactic. Here, AO takes advantage of the target's chosen location to approach its victim and attack. The mathematical representation of this behavior is given in Eq. (2.13).

$$X_3(t+1) = (X_{best}(t) - X_M(t)) \times \alpha - rand + ((UB - LB) \times rand + LB) \times \delta, \tag{2.13}$$

where $X_3(t+1)$ represents the result of the third search technique (X_3) for the solution of the subsequent iteration of the constant t. The best-obtained solution,$X_{best}(t)$, reflects the approximate position of the prey up until the t^{th} iteration, and the mean value of the current solution at the t^{th} iteration, $X_M(t)$, is determined using Eq. (2.4). A random number between 0 and 1 is called *rand*. The exploitation adjustment parameters α and δ, are set in this study at a low value (0.1). The given problem's *LB* and *UB* abbreviations stand for lower and upper bounds, respectively.

3.2.2.4 Step 4: Narrowed exploitation (X_4) In the fourth technique (X_4), the Aquila approaches the target and then assaults it over the land in accordance with its stochastic motions. This strategy is known as "walk and grab prey." Finally, AO engages the prey at the last position. The mathematical representation of this behavior is given in Eq. (2.14).

$$X_4(t+1) = QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times Levy(D) + rand \times G_1, \tag{2.14}$$

where $X_4(t+1)$ is the result of the fourth search technique (X_4), which is created for the following iteration of t. The term "QF" refers to a quality function that is derived using Eq. (2.15) and is used to balance the search techniques. G_1, which is produced using Eq. (2.16), stands for multiple AO movements that are employed to follow the prey during the hunt. The flight slope of the AO used to follow the prey during the journey from the starting position (1) to the last site (t), which is created using Eq. (2.17), is represented by G_2 by decreasing values from 2 to 0. The solution as of the t^{th} iteration is X(t).

$$QF(t) = t^{\frac{2\times rand-1}{(1-T)^2}} \tag{2.15}$$

$$G_1 = 2 \times rand - 1 \tag{2.16}$$

$$G_2 = 2 \times \left(1 - \frac{t}{T}\right) \tag{2.17}$$

The quality function value at the t^{th} iteration is denoted by $QF(t)$, and the random value, rand, is a number between 0 and 1. The current iteration and the maximum number of iterations are represented, respectively, by t and T. Using Eq. (2.6), we can derive the levy flight distribution function, $Levy(D)$.

Algorithm 2 explains the pseudo-code of the AO. The flowchart of the AO is presented in Fig. 3.

Algorithm 2 Aquila Optimizer [31].

1: Initialization phase:
2: Initialize the population X of the AO.
3: Initialize the parameters of the AO (i.e., $\omega, \delta, \alpha, etc$).
4: WHILE (The end condition is not met) **do**
5: Calculate the fitness function values.
6: $X_{best}(t) =$ Determine the best-obtained solution according to the fitness values.
7: **for** ($i = 1,2,\ldots,N$) **do**
8: Update the mean value of the current solution $X_M(t)$.
9: Update the $x, y, G_1, G_2, Levy(D), etc.$
10: **If** $t \leq \left(\frac{2}{3}\right) \times T$ **then**
11: **If** $rand \leq 0.5$ **then**
12: **Step 1: Expanded exploration (X_1)**
13: Update the current solution using Eq. (3.3).
14: **If** Fitness ($X_1(\boldsymbol{t+1})$))$<$ $Fitness(X(t))$ **then**
15: $\boldsymbol{X(t) = X_1(t+1)}$
16: **If** Fitness ($X_1(\boldsymbol{t+1})$))$<$ $Fitness(X_{best}(t))$ **then**
17: $\boldsymbol{X_{best}(t) = X_1(t+1)}$
18: **end if**
19: **end if**
20: **else**
21: **Step 2: Narrowed exploration (X_2)**
22: Update the current solution using Eq. (3.5).
23: **If** Fitness ($X_2(\boldsymbol{t+1})$))$<$ $Fitness(X(t))$ **then**
24: $\boldsymbol{X(t) = X_2(t+1)}$
25: **If** Fitness ($X_2(\boldsymbol{t+1})$))$<$ $Fitness(X_{best}(t))$ **then**
26: $\boldsymbol{X_{best}(t) = X_2(t+1)}$
27: **end if**
28: **end if**
29: **end if**
30: **else**
31: *if* $rand \leq 0.5$ **then**
32: **Step 3: Expanded exploitation (X_3)**
33: Update the current solution using Eq. (3.13).
34: **If** Fitness ($X_3(\boldsymbol{t+1})$))$<$ $Fitness(X(t))$ **then**
35: $\boldsymbol{X(t) = X_3(t+1)}$
36: **If** Fitness ($X_3(\boldsymbol{t+1})$))$<$ $Fitness(X_{best}(t))$ **then**
37: $\boldsymbol{X_{best}(t) = X_3(t+1)}$
38: **end if**
39: **end if**
40: **else**
41: **Step 4: Narrowed exploitation (X_4)**
42: Update the current solution using Eq. (3.14).
43: **If** Fitness ($X_4(\boldsymbol{t+1})$))$<$ $Fitness(X(t))$ **then**
44: $\boldsymbol{X(t) = X_4(t+1)}$
45: **If** Fitness ($X_4(\boldsymbol{t+1})$))$<$ $Fitness(X_{best}(t))$ **then**
46: $\boldsymbol{X_{best}(t) = X_4(t+1)}$
47: **end if**
48: **end if**
49: **end if**
50: **end if**
51: **end for**
52: end while
53: return The best solution ($\boldsymbol{X_{best}}$).

Table 2 CEC 2005 benchmark functions (U: Unimodal, M: Multimodal, F: Fixed dimension)

Function name	Dim	Range	f_{min}	Properties
Sphere	30	[− 100, 100]	0	U
Schwefel 2.22	30	[− 10, 10]	0	U
Schwefel 1.2	30	[− 100, 100]	0	U
Schwefel 2.21	30	[− 100, 100]	0	U
Rosenbrock's Function	30	[− 30, 30]	0	U
Step Function	30	[− 100, 100]	0	U
Quartic Function	30	[− 1.28, 1.28]	0	U
Schwefel 2.26	30	[−500,500]	− 418.9829 × 5	M
Rastrigin function	30	[− 5.12, 5.12]	0	M
Ackley's function	30	[− 32, 32]	0	M
Griewank function	30	[− 600, 600]	0	M
Pendized	30	[− 50, 50]	0	M
Generalized Pendized	30	[− 50, 50]	0	M
Shekel's foxholes	2	[− 65, 65]	1	F
Kowalik	4	[− 5, 5]	0.00030	F
Camel	2	[− 5, 5]	− 1.398	F
Rastrigin	2	[− 5, 5]	0.398	F
GoldStein price	2	[− 2, 2]	3	F
Hartmann 3-D	3	[1, 3]	− 3.86	F
Hartmann 6-D	6	[0, 1]	− 3.32	F
Shekel 1	4	[0, 10]	− 10.1532	F
Shekel 2	4	[0, 10]	− 10.4028	F
Shekel 3	4	[0, 10]	− 10.5363	F

4 The proposed modified coronavirus herd immunity aquila optimization algorithm (MCHIAO)

This section outlines the hybrid algorithm that we developed by enhancing and fusing CHIO and AO, two distinct optimization algorithms. As AO is a population-based algorithm and CHIO is a human-based algorithm, they are both metaheuristic optimization algorithms. The suggested method comprises two phases: exploration and exploitation, with the exploration phase performed by ECHIO and the exploitation phase by the AO. The first of two modifications to CHIO is the categorization of cases, followed by applying chaotic maps to improve the equation values for the new genes. Based on this condition, if $t \leq (\frac{2}{3})T$, the exploration steps will be excited; otherwise, the exploitation steps will be carried out, and the MCHIAO algorithm can switch from exploration steps to exploitation steps utilizing this behavior. Finally, the two scenarios of exploitation using AO are used in the proposed algorithm in addition to an enhancement to the random value used to switch between narrowed and expanded exploitation. In the subsections that follow, the specifics of the proposed MCHIAO algorithm are covered.

4.1 Enhanced coronavirus herd immunity optimization algorithm (ECHIO)

4.1.1 Cases categorizing

For every search and optimization algorithm, exploration and exploitation exemplify influential characteristics. The inability to achieve the balance between both exploration and exploitation leads the optimization algorithms to fall into local optimum in optimization problems.

Supporting optimal "exploitation" of the existing results and promoting "exploration" for new ones is our main goal in this proposed contribution.

Instead of categorizing the cases (individuals) according to the basic reproduction rate (BR_r) parameter which leads the CHIO algorithm to be trapped in the local optima, we use the status vector (S) in classifying the cases (individuals) that caused a local optima avoidance and a faster convergence curve as follows:

$$x_i^j(t+1) \leftarrow \begin{Bmatrix} C\left(x_i^j(t)\right) & S_i = 1//Infected\ case \\ N\left(x_i^j(t)\right) & S_i = 0//susceptible\ case \\ R\left(x_i^j(t)\right) & S_i = 2//immuned\ case \end{Bmatrix} \tag{3.1}$$

Table 3 MCHIAO Vs AO and CHIO algorithms are conducted on 23 benchmark functions

Function	Algorithm indices	AO [31]	CHIO [30]	MCHIAO
F1	Mean	2.498E-110	1.897E-40	**0**
	Std	5.585E-110	4.2417E-40	0
	Rank	2	3	**1**
F2	Mean	9.6935E-55	9.8005E-28	**3.271E-205**
	Std	2.1675E-54	2.0803E-27	0
	Rank	2	3	**1**
F3	Mean	1.385E-99	2.6399E-34	**0**
	Std	3.096E-99	4.6842E-34	0
	Rank	2	3	**1**
F4	Mean	6.9898E-55	4.1704E-17	**6.621E-211**
	Std	1.563E-54	8.1929E-17	0
	Rank	2	3	**1**
F5	Mean	0.00171213	3.10247561	**6.1598E-18**
	Std	0.00091398	0.0009991	8.8012E-18
	Rank	2	3	**1**
F6	Mean	2.3877E-05	0.00302397	**6.163E-34**
	Std	1.9152E-05	0.00081059	1.3781E-33
	Rank	2	3	**1**
F7	Mean	0.00027164	0.00042438	**2.8462E-05**
	Std	0.00015596	0.00039545	1.5409E-05
	Rank	2	3	**1**
F8	Mean	−2094.1396	−1395.7482	**−2094.9144**
	Std	1.29594899	87.7140848	0
	Rank	2	3	**1**
F9	Mean	0	4.048605	**0**
	Std	0	3.07757649	0
	Rank	**1**	2	**1**
F10	Mean	4.4409E-16	7.5495E-15	**4.4409E-16**
	Std	0	0	0
	Rank	**1**	2	**1**
F11	Mean	0.00072208	0.05034689	**0**
	Std	0.00161462	0.05135439	0
	Rank	2	3	**1**
F12	Mean	0.00014737	0.00160332	**9.6307E-32**
	Std	0.00029131	0.00097396	2.7483E-33
	Rank	2	3	**1**
F13	Mean	**1.1447E-05**	0.27196928	0.00156962
	Std	1.2248E-05	0.17811665	0.00415283
	Rank	**1**	3	2
F14	Mean	6.42595173	0.99806131	**0.99800384**
	Std	5.8797439	8.1814E-05	0
	Rank	3	2	**1**
F15	Mean	0.00046154	0.00129651	**0.00030749**
	Std	5.9718E-05	3.7358E-05	2.5619E-19
	Rank	2	3	**1**
F16	Mean	−1.0304442	−1.0316039	**−1.0316285**
	Std	0.00093395	3.6921E-05	8.3925E-17
	Rank	3	2	**1**

Table 3 (continued)

Function	Algorithm indices	AO [31]	CHIO [30]	MCHIAO
F17	Mean	0.39814763	0.39834444	**0.39788736**
	Std	0.00022088	0.00051409	0
	Rank	2	3	**1**
F18	Mean	3.05734004	3.00013989	**3**
	Std	0.05647628	0.00013478	6.1672E-16
	Rank	3	2	**1**
F19	Mean	−3.8501083	−3.8559734	**−3.8627821**
	Std	0.01220757	0.00279125	1.6994E-15
	Rank	3	2	**1**
F20	Mean	−3.135271	−2.9360783	**−3.2744379**
	Std	0.07864923	0.32353334	0.06028982
	Rank	2	3	**1**
F21	Mean	−10.137405	−4.1472921	**−10.1532**
	Std	0.01166538	1.6909226	1.0616E-15
	Rank	2	3	**1**
F22	Mean	−10.377872	−4.1935533	**−10.402941**
	Std	0.04536386	1.69978168	2.1756E-15
	Rank	2	3	**1**
F23	Mean	−10.524331	−4.752378	**−10.53641**
	Std	0.01151248	1.05458509	2.8485E-15
	Rank	2	3	**1**
Percentage		2.043478261	2.826086957	1.043478261
Total Rank		2	3	**1**

4.1.2 New genes' value equation enhancement:

In the coronavirus herd immunity evolution for infected, susceptible, and immune cases instead of adding the current gene's value to the dissimilarity among the existing gene's value and a random gene adapted from infected, susceptible, and immune cases as shown in Eqs. 1.6, 1.7, and 1.8, respectively, we use subtraction in Eqs. 3.2, 3.3, and 3.4.

In addition, chaotic maps are used instead of random parameters due to their expected ability to increase the speed of convergence. Equations 3.2, 3.3, and 3.4 are modified to clarify the proposed claim. Chaotic systems or simply chaos can be described as behavior that falls between rigid regularity and randomness. One of the most essential characteristics of the chaotic system is that it exhibits extreme sensitivity to initial conditions. Many studies suggested that the spread of COVID-19 can be categorized as a chaotic system.

The chaotic behavior of the coronavirus inspired us to use the chaotic system by replacing the random parameter (r) with the parameter (m).

Local minima avoidance and rapid convergence rate are the main reasons for us to use chaos optimization. Chaos theory plays an important and effective role in overcoming these problems.

The main three important dynamic properties of chaos can be described as quasi-stochastic, ergodicity, and sensitive dependence on the initial conditions.

Quasi-stochastic can be described as the ability to replace random variables with the values of a chaotic map. Ergodicity expresses the ability of chaotic variables to search the non-recurrently of all states in a specified range.

Replacing the random parameter "r" with the chaotic value "m" allows an enormous effect on enhancing the exploration phase of the proposed ECHIO algorithm which assists in finding a new promising search region hopefully for the optimum solution.

The new enhanced equations for Eqs. 1.6, 1.7, and 1.8 respectively will be as follows:

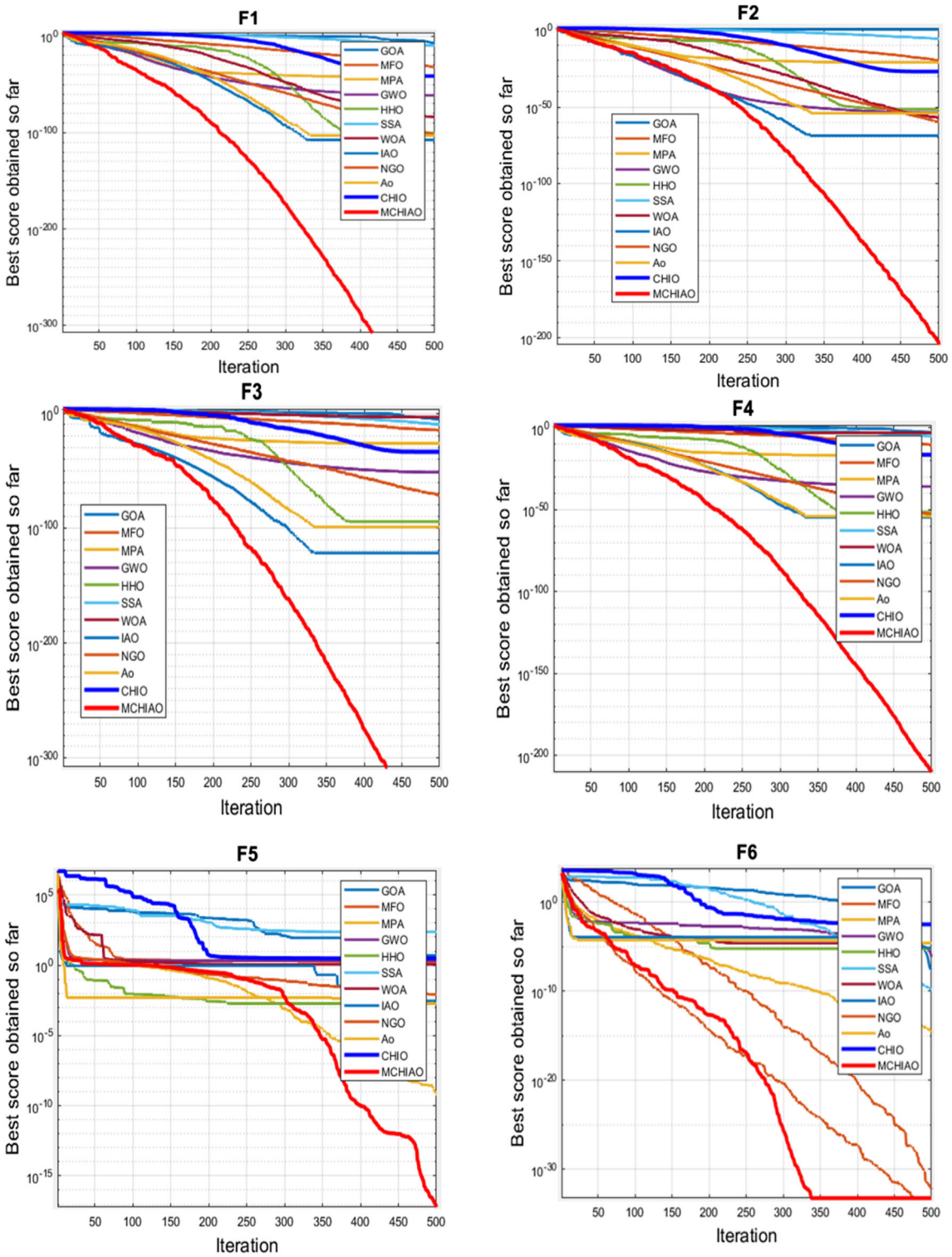


Fig. 7 The 23 CEC2005 benchmark functions and convergence plots

$$C\left(x_i^j(t)\right) = x_i^j(t) - m \times \left(x_i^j(t) - x_i^c(t)\right) \tag{3.2}$$

$$N\left(x_i^j(t)\right) = x_i^j(t) - m \times \left(x_i^j(t) - x_i^m(t)\right) \tag{3.3}$$

$$R\left(x_i^j(t)\right) = x_i^j(t) - m \times \left(x_i^j(t) - x_i^v(t)\right) \tag{3.4}$$

where the m vector can be calculated as shown in Eq. 3.5.

$$m = Chaotic - value \tag{3.5}$$

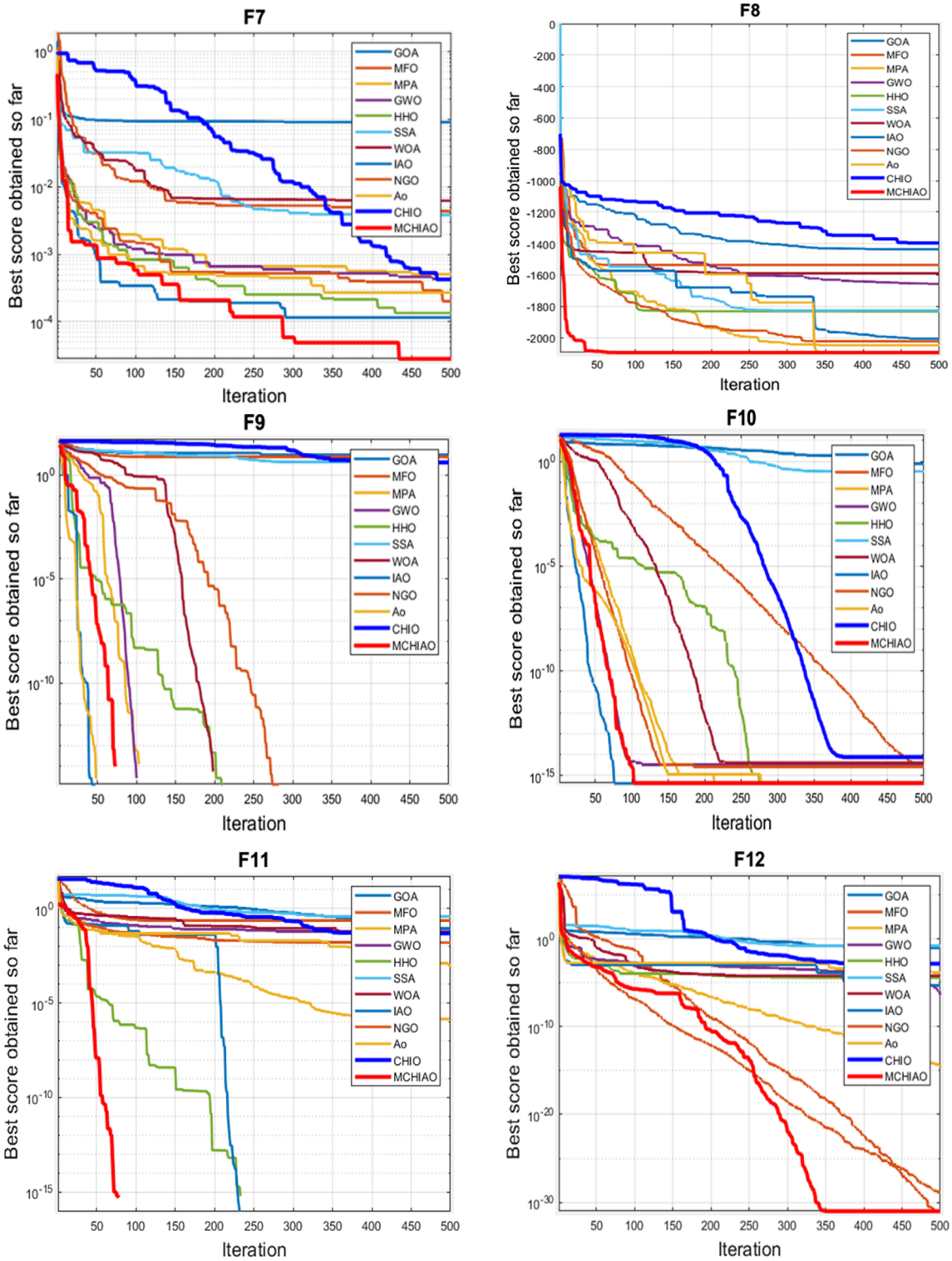


Fig. 7 continued

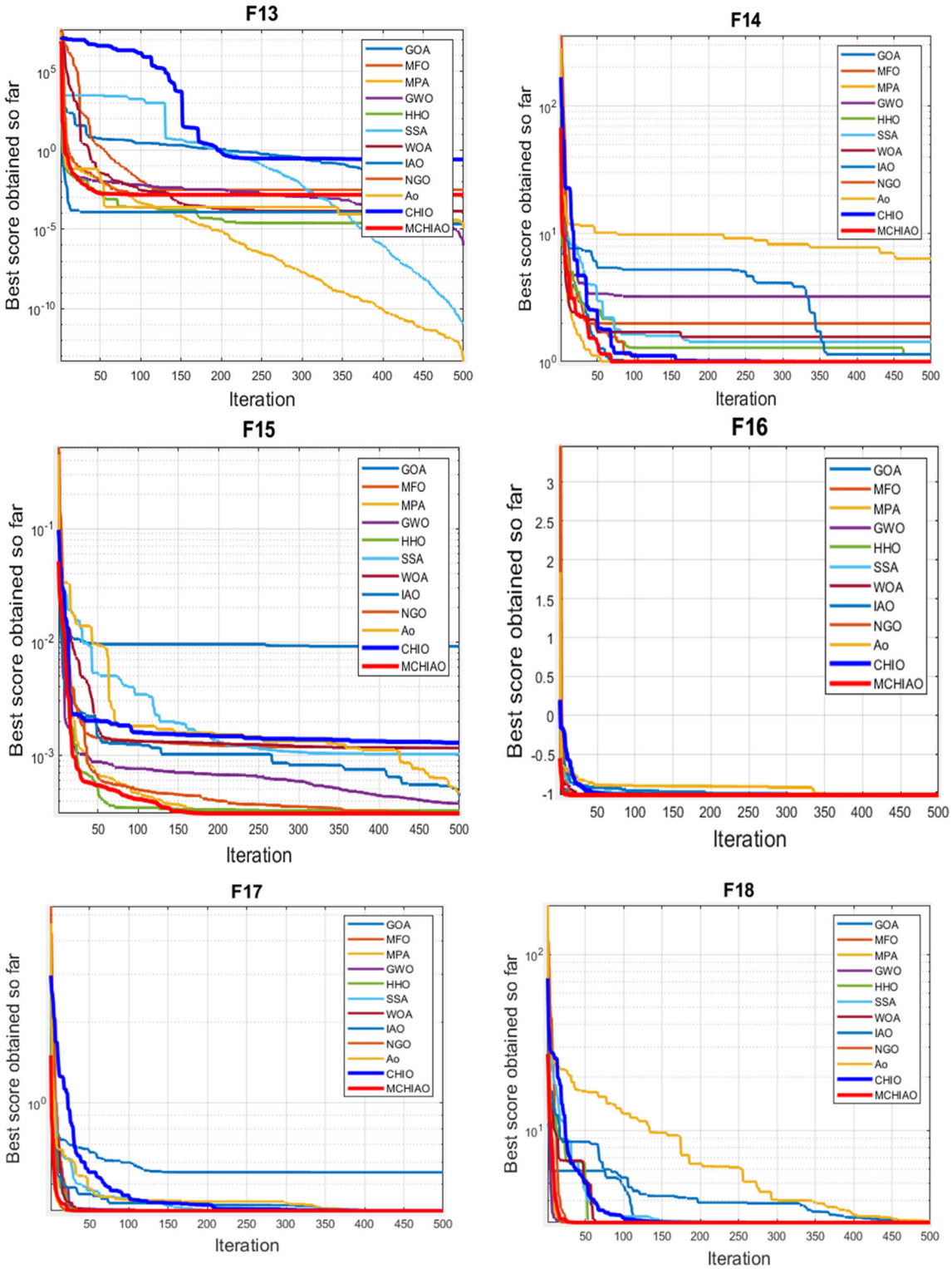


Fig. 7 continued

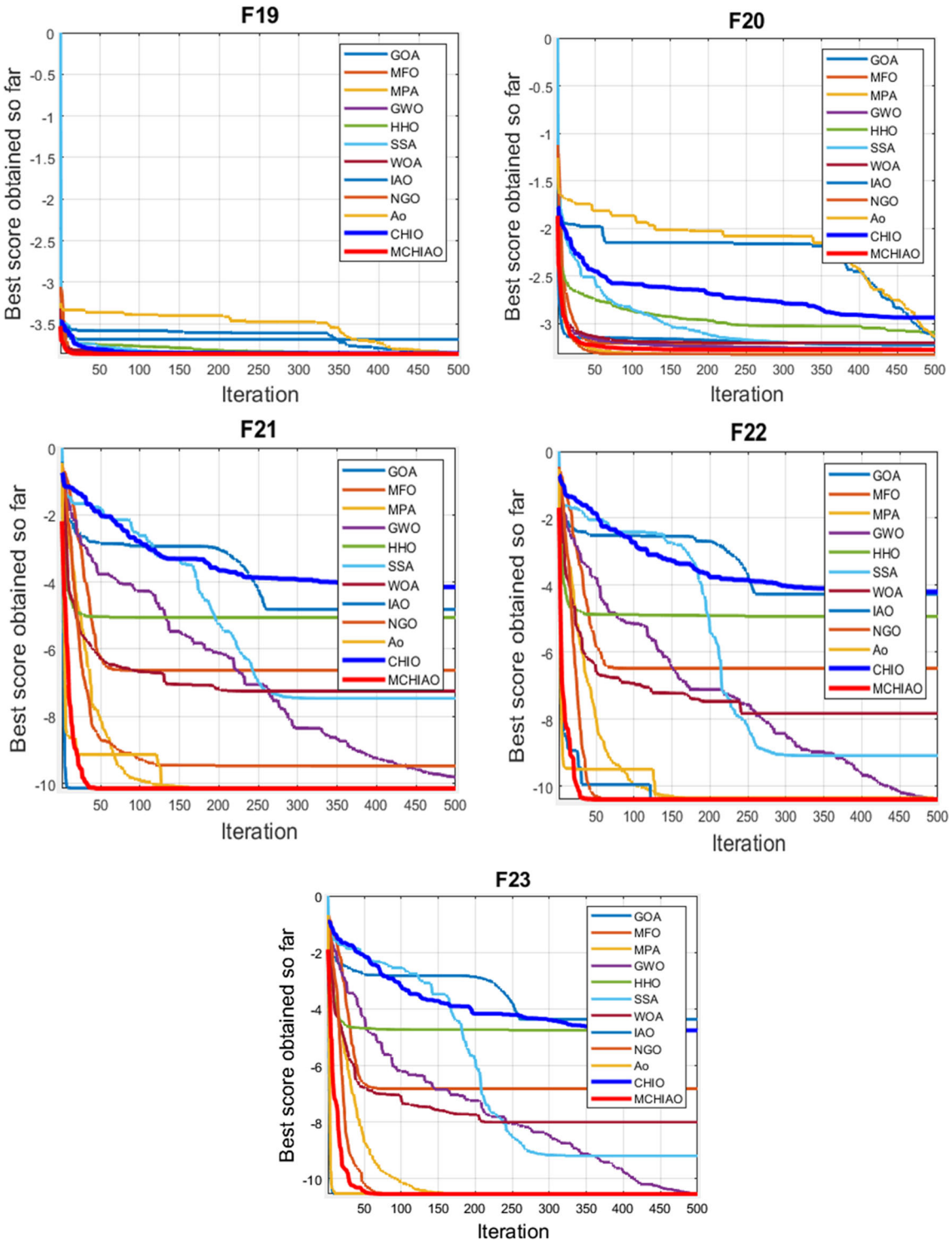


Fig. 7 continued

Table 4 MCHIAO performance against twelve different algorithms is conducted on 23 benchmark functions

Function	Algorithm	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
	Indices													
F1	Mean	1.5973E-08	4.9173E-35	6.8308E-39	7.8637E-96	1.433E-101	1.4037E-10	3.427E-84	2.498E-110	3.093E-105	0.73574718	1.307E-116	1.897E-40	**0**
	Std	8.5023E-09	9.1169E-35	1.3482E-38	1.7398E-95	2.808E-101	5.4175E-11	7.6554E-84	5.585E-110	6.916E-105	1.64518071	2.068E-116	4.2417E-40	0
	Rank	12	10	9	6	5	11	7	3	4	13	2	8	**1**
F2	Mean	2.11381135	1.4097E-20	8.6756E-22	2.4214E-54	4.4329E-52	2.3349E-06	7.7813E-58	9.6935E-55	2.3978E-69	7.4557E-25	2.3895E-60	9.8005E-28	**3.271E-205**
	Std	1.52932691	1.9221E-20	9.5054E-22	2.2222E-54	8.8648E-52	3.7522E-07	1.7384E-57	2.1675E-54	5.3432E-69	1.1903E-24	3.6745E-60	2.0803E-27	0
	Rank	13	11	10	6	7	12	4	5	2	9	3	8	**1**
F3	Mean	9.4791E-06	1.317E-15	6.6597E-27	3.9651E-52	5.9621E-95	1.6011E-10	0.00032759	1.385E-99	1.41E-122	4.4052E-15	2.1704E-72	2.6399E-34	**0**
	Std	1.3864E-05	1.8382E-15	7.4433E-27	8.8564E-52	1.0355E-94	6.6904E-11	0.00064329	3.096E-99	3.154E-122	9.8504E-15	4.542E-72	4.6842E-34	0
	Rank	12	9	8	6	4	11	13	3	2	10	5	7	**1**
F4	Mean	0.00194564	2.4508E-11	1.6817E-17	1.0294E-36	2.5847E-52	8.5423E-06	0.00084739	6.9898E-55	1.821E-55	4.5301E-08	8.2997E-54	4.1704E-17	6.621E-211
	Std	0.00416853	5.1072E-11	1.8101E-17	1.6918E-36	5.069E-52	1.9982E-06	0.001878	1.563E-54	4.0718E-55	1.013E-07	6.9701E-54	8.1929E-17	0
	Rank	13	9	7	6	5	11	12	3	2	10	4	8	**1**
F5	Mean	5.182506	1.72494152	6.7181E-10	1.99367468	0.0018464	230.156023	1.17483216	0.00171213	0.00280324	3.99879444	0.00850042	3.10247561	**6.1598E-18**
	Std	6.63212775	1.37042393	7.1244E-10	0.77706954	0.00107358	503.02212	0.2863104	0.00091398	0.00450136	0.00154793	0.00473384	0.0009991	8.8012E-18
	Rank	12	8	2	9	4	13	7	3	5	11	6	10	**1**
F6	Mean	2.4159E-08	5.5467E-33	1.9929E-15	6.9224E-07	5.8882E-06	1.4288E-10	2.209E-05	2.3877E-05	6.6967E-06	0.87486239	**0**	0.00302397	6.163E-34
	Std	1.9576E-08	6.3152E-33	8.2873E-16	3.8386E-07	6.8497E-06	7.5902E-11	2.2205E-05	1.9152E-05	6.9096E-06	0.28529918	0	0.00081059	1.3781E-33
	Rank	6	3	4	7	8	5	10	11	9	13	**1**	12	2
F7	Mean	0.09116087	0.00430673	0.00050592	0.00043497	0.00013403	0.00383037	0.00615608	0.00027164	0.0001155	0.01475049	0.00020189	0.00042438	**2.8462E-05**
	Std	0.12082312	0.00149528	0.00053054	0.00017124	0.00013774	0.0029451	0.00807216	0.00015596	0.00013216	0.01453188	0.00010891	0.00039545	1.5409E-05
	Rank	13	10	8	7	3	9	11	5	2	12	4	6	**1**
F8	Mean	−1435.5609	−1536.1287	−2047.5391	−1656.614	−1833.6913	−1826.1297	−1589.4433	−2094.1396	−2005.8999	−1398.3503	−2023.8496	−1395.7482	**−2094.9144**
	Std	158.300326	207.540052	64.8713475	194.635291	359.990807	168.660135	257.549602	1.29594899	195.288762	126.289973	64.8703279	87.7140848	0
	Rank	11	10	3	8	6	7	9	2	5	12	4	13	**1**
F9	Mean	9.12059796	7.56167572	0	0	0	4.17882502	0	0	0	0	0	4.048605	**0**
	Std	3.64109934	5.15075774	0	0	0	2.84912677	0	0	0	0	0	3.07757649	0
	Rank	**13**	**12**	**1**	**1**	**1**	11	**1**	**1**	**1**	**1**	**1**	10	**1**
F10	Mean	0.79293634	3.9968E-15	4.4409E-16	3.2863E-15	4.4409E-16	0.32925081	3.9968E-15	4.4409E-16	4.4409E-16	4.4409E-16	2.5757E-15	7.5495E-15	**4.4409E-16**
	Std	1.11064628	0	0	1.5888E-15	0	0.73621019	3.5527E-15	0	0	0	1.9459E-15	0	0
	Rank	13	9	**1**	8	**1**	12	9	**1**	**1**	**1**	7	11	**1**
F11	Mean	0.09114394	0.24050646	1.2126E-06	0.05988084	0	0.40120356	0.0616698	0.00072208	0	3.1908E-14	0.01583219	0.05034689	**0**
	Std	0.04036108	0.12791341	2.7114E-06	0.04325337	0	0.08929817	0.11260917	0.00161462	0	7.1286E-14	0.0172893	0.05135439	0
	Rank	11	12	5	9	**1**	13	10	6	**1**	4	7	8	**1**
F12	Mean	0.09072498	9.7483E-32	2.7047E-15	4.3838E-07	3.8438E-05	0.17771859	6.2435E-05	0.00014737	5.2232E-06	0.66409743	1.1564E-29	0.00160332	**9.6307E-32**
	Std	0.23437044	6.6174E-33	1.687E-15	1.099E-07	5.108E-05	0.30351227	5.0971E-05	0.00029131	6.5066E-06	0.38503571	2.103E-29	0.00097396	2.7483E-33
	Rank	11	2	4	5	7	12	8	9	6	13	3	10	**1**
F13	Mean	0.00317743	0.00313925	**4.3674E-14**	1.0192E-06	2.4685E-05	1.124E-11	0.00014477	1.1447E-05	2.2012E-05	0.35288967	0.00156962	0.27196928	0.00156962
	Std	0.00537076	0.00536129	4.6235E-14	8.7294E-07	2.8358E-05	1.0749E-11	0.00016152	1.2248E-05	3.885E-05	0.14802609	0.00415283	0.17811665	0.00415283
	Rank	11	10	**1**	3	6	2	7	4	5	13	8	12	8
F14	Mean	0.99800384	1.98809181	0.99800384	3.23239021	1.1400077	1.42401544	1.56488993	6.42595173	1.14000772	5.17505391	0.99800384	0.99806131	**0.99800384**
	Std	9.0649E-17	1.51024199	1.1992E-16	4.26562307	0.37570692	0.53132981	0.96814231	5.8797439	0.37570691	2.89009714	0	8.1814E-05	0

Table 4 (continued)

Function	Algorithm	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
	Rank	4	10	3	11	6	8	9	13	7	12	**1**	5	**1**
F15	Mean	0.00920522	0.00116215	0.00030749	0.00037567	0.00032803	0.00103446	0.00116228	0.00046154	0.00043937	0.00472058	0.00030885	0.00129651	**0.00030749**
	Std	0.01044301	0.00046146	1.0336E-14	7.7721E-05	1.2736E-05	0.00030129	0.00060438	5.9718E-05	4.7843E-05	0.00197314	3.0822E-06	3.7358E-05	2.5619E-19
	Rank	13	9	2	5	4	8	10	7	6	12	3	11	**1**
F16	Mean	−1.0316285	−1.0316285	−1.0316285	−1.0316284	−1.0316285	−1.0316285	−1.0316285	−1.0304442	−1.031362	−1.0180209	−1.0316285	−1.0316039	**−1.0316285**
	Std	2.4571E-13	0	2.5177E-16	2.1017E-08	3.3192E-09	5.0738E-14	1.6613E-09	0.00093395	0.00022188	0.01074748	1.4536E-16	3.6921E-05	8.3925E-17
	Rank	6	**1**	**1**	9	8	5	7	12	11	13	**1**	10	**1**
F17	Mean	0.55172094	0.39788736	0.39788736	0.39788876	0.39789141	0.39788736	0.39789082	0.39814763	0.39817291	0.50202559	0.39788736	0.39834444	**0.39788736**
	Std	0.5957949	0	1.2818E-14	1.4266E-06	5.3519E-06	5.4088E-14	5.6285E-06	0.00022088	0.00023252	0.09631877	0	0.00051409	0
	Rank	13	**1**	4	6	8	5	7	9	10	12	**1**	11	**1**
F18	Mean	3	3	3	3.00003372	3.00000012	3	3.0000229	3.05734004	3.02371878	4.00919048	3	3.00013989	**3**
	Std	2.6899E-12	1.8841E-15	1.8502E-15	4.1379E-05	2.6381E-07	3.2715E-13	2.764E-05	0.05647628	0.03201449	1.11439586	1.8729E-15	0.00013478	6.1672E-16
	Rank	6	2	4	9	7	5	8	12	11	13	3	10	**1**
F19	Mean	−3.6884514	−3.8622567	−3.8627821	−3.8616061	−3.8600216	−3.8627821	−3.8582197	−3.8501083	−3.8525069	−3.8358803	−3.8627821	−3.8559734	**−3.8627821**
	Std	0.31341614	0.002035	1.5475E-15	0.00221919	0.00261494	5.9547E-08	0.00561192	0.01220757	0.01959385	0.02003173	1.6994E-15	0.00279125	1.6994E-15
	Rank	13	5	**1**	6	7	4	8	11	10	12	**1**	9	**1**
F20	Mean	−3.2217422	−3.2080735	**−3.3219952**	−3.2661972	−3.0878425	−3.2082946	−3.2013424	−3.135271	−3.1470838	−2.8945245	−3.3219952	−2.9360783	−3.2744379
	Std	0.05210291	0.06565828	8.0614E-12	0.08834258	0.11592123	0.06280645	0.17977218	0.07864923	0.10833275	0.18102638	5.8335E-11	0.32353334	0.06028982
	Rank	5	7	**1**	4	11	6	8	10	9	13	2	12	3
F21	Mean	−4.808643	−6.6285916	−10.1532	−9.8115861	−5.0522926	−7.4769745	−7.2528565	−10.137405	−10.145363	−3.6076894	−9.4734483	−4.1472921	**−10.1532**
	Std	2.9590984	3.11617658	2.7861E-11	1.31581947	0.00316123	3.47646788	2.84040612	0.01166538	0.01126362	1.4991188	1.79380521	1.6909226	1.0616E-15
	Rank	11	9	2	5	10	7	8	4	3	13	6	12	**1**
F22	Mean	−4.2579753	−6.4884549	−10.402941	−10.40087	−4.9261201	−9.0960229	−7.8331089	−10.377872	−10.394584	−4.356908	−10.402941	−4.1935533	**−10.402941**
	Std	2.71984528	3.85239614	2.7695E-11	0.0006488	0.61096757	2.74301652	3.38904688	0.04536386	0.00923742	1.53404233	2.8577E-10	1.69978168	2.1756E-15
	Rank	12	9	2	4	10	7	8	6	5	11	3	13	**1**
F23	Mean	-4.3560121	-6.8178958	-10.53641	-10.534771	-4.7408655	-9.1881706	-8.0088575	-10.524331	-10.524856	-3.0188587	-10.53641	-4.752378	**-10.53641**
	Std	3.29588419	3.68487363	3.6471E-11	0.00091265	1.04040538	2.83763734	3.62579054	0.01151248	0.01825422	0.80314641	6.3826E-08	1.05458509	2.8485E-15
	Rank	12	9	2	4	11	7	8	6	5	13	3	10	**1**
Percentage	10.6956522	7.69565217	3.69565217	6.26086957	6.08695652	8.30434783	8.2173913	6.34782609	5.30434783	10.6956522	3.43478261	9.82608696	1.43478261	
Total Rank	12	8	3	6	5	10	9	7	4	12	2	11	**1**	

Table 5 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for 23 benchmark functions (CEC2005)

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	2.5461E-152	1.2559E-105	5.12355E-71	8.7002E-25	2.45001E-50	6.8968E-145	5.4355E-62	9.38973E-27	1.64874E-32	3.4875E-104	5.30702E-35	1.3249E-111
F2	2.0164E-161	6.8389E-102	3.81819E-61	5.67626E-13	3.56066E-45	1.3772E-148	5.47653E-40	2.42903E-20	3.23502E-08	1.65054E-84	1.25757E-25	5.14441E-96
F3	2.5454E-148	3.6492E-122	1.26053E-82	2.32745E-48	1.64178E-46	6.7566E-140	6.6864E-142	3.38225E-26	2.73632E-14	5.4857E-70	4.38819E-50	4.6279E-107
F4	1.082E-152	8.081E-123	1.61319E-81	2.03928E-40	3.63642E-51	1.2266E-145	5.0687E-139	2.79988E-28	4.26746E-28	2.3084E-106	5.13153E-41	2.7806E-126
F5	1.5166E-155	5.2051E-133	0.340978458	2.0148E-129	0.000236125	1.77E-156	2.1113E-106	3.88743E-07	4.18165E-15	1.1525E-104	1.57998E-08	9.1049E-155
F6	2.9791E-146	2.67543E-25	6.532E-48	1.2914E-114	2.57644E-92	7.4245E-123	1.1382E-99	4.59476E-98	6.6872E-101	1.89562E-96	0.008650103	1.3509E-142
F7	1.1787E-160	2.3171E-150	1.20065E-85	1.4555E-76	2.37737E-39	8.4678E-151	1.8561E-150	4.8827E-50	1.24899E-05	3.24668E-80	3.45313E-67	7.0156E-137
F8	1.2155E-163	5.0697E-165	7.2075E-142	2.128E-161	3.488E-158	4.132E-159	1.072E-161	1.0877E-136	2.6106E-156	5.2554E-167	1.311E-145	1.1618E-164
F9	1.6354E-173	2.9159E-169	0.008248341	0.006721861	1.4836E-17	3.8118E-169	5.12702E-22	0.012970667	0.009155709	1.54401E-94	5.39808E-43	5.7632E-174
F10	2.8181E-164	5.3408E-117	4.08957E-11	6.11902E-80	1.212E-28	7.7615E-161	6.69145E-94	7.26644E-20	0.020076377	2.7237E-91	8.47418E-80	2.3713E-125
F11	3.7569E-160	5.3508E-147	8.6238E-127	2.9338E-135	2.20327E-22	5.3029E-161	7.0532E-140	1.4365E-131	7.87255E-26	1.98157E-77	1.8459E-132	5.52E-152
F12	8.5862E-153	8.5311E-12	1.4739E-28	3.8857E-105	1.87118E-89	4.1205E-154	7.847E-97	2.3451E-116	2.8999E-107	5.6288E-145	0.00872443	5.2195E-142
F13	2.0828E-149	6.5091E-126	1.88415E-79	0.860200954	1.3604E-107	6.00163E-11	7.52835E-40	6.1948E-107	9.5591E-162	1.322E-150	0.512085259	3.3001E-157
F14	9.16268E-79	2.6547E-127	6.31228E-56	7.8053E-152	3.7631E-110	3.8094E-115	2.3327E-122	2.423E-161	1.693E-133	2.4063E-09	1.25256E-13	1.29388E-92
F15	3.2521E-154	2.9969E-144	5.09051E-19	5.6547E-108	2.7877E-41	1.1979E-144	1.2206E-144	2.7581E-143	5.8824E-136	2.6723E-101	1.76418E-54	8.6336E-145
F16	3.9144E-130	0.035236644	3.2916E-51	7.7826E-122	1.1167E-103	3.1806E-123	1.9125E-99	3.639E-165	5.35E-162	7.7384E-173	0.011493884	3.1987E-145
F17	1.9571E-165	0.078995395	1.90536E-67	3.5793E-131	1.9028E-121	2.2808E-108	1.275E-118	1.1336E-151	8.1139E-148	4.17015E-11	0.98397078	6.6446E-147
F18	2.1669E-108	8.73469E-08	4.12959E-46	2.523E-110	9.18335E-90	8.6726E-104	4.847E-108	1.2029E-153	2.9437E-149	3.53885E-06	1.08471E-18	1.8662E-126
F19	8.4246E-168	2.8386E-119	1.09498E-23	7.442E-129	1.1725E-153	7.0934E-111	4.2022E-147	1.2961E-166	2.1696E-165	3.9368E-169	1.22764E-05	4.7856E-151
F20	2.8656E-128	2.7535E-128	1.8027E-110	1.72517E-90	1.9334E-151	6.0639E-139	1.324E-130	5.5328E-163	1.3577E-162	1.2459E-165	9.0054E-123	2.6219E-158
F21	6.7897E-163	1.0463E-158	7.77361E-92	3.5287E-152	1.5142E-160	6.2113E-159	6.5278E-156	1.2357E-128	2.6766E-122	5.307E-167	2.4955E-141	4.0767E-164
F22	2.1661E-166	2.7836E-164	9.11118E-90	3.3361E-153	2.3329E-164	4.4595E-156	1.5134E-159	5.0774E-139	4.6635E-123	5.8373E-168	6.6651E-64	5.415E-166
F23	4.0716E-167	2.8712E-162	2.15746E-82	1.6774E-148	1.4541E-165	8.7026E-156	1.5993E-154	1.6895E-108	2.179E-106	7.2151E-171	6.32622E-74	1.6701E-166

This chaotic behavior in the final stage aids in alleviating the two issues of entrapment in local optima and slow convergence rate while solving high-dimensional problems. In this article, 10 chaotic maps have been used to improve the performance of the CHIO algorithm as illustrated in Table 1 [31].

Figure 4 illustrates the flowchart of the proposed optimization algorithm ECHIO. Figure 5 presents the graphical description of the proposed ECHIO algorithm. The pseudo-code of the proposed algorithm ECHIO is illustrated in Algorithm 3.

Algorithm 3 The proposed ECHIO pseudo-code.

1: {———- Step 1: Initialize the ECHIO parameters ———-}
2: Initialize the parameters (HIS, S_r, and Max_{age}).
3: {———- Step 2: Generate herd immunity population ———-}
4: $x_i^j = lb_i + (ub_i - lb_i) \times U(0,1)$, $\forall i = 1,2,\ldots,n$, and $\forall j = 1,2,\ldots,HIS$
5: Calculate the fitness of each search agent
6: Set $S_j = 0$ $\forall j = 1,2,3,\ldots,HIS.$
7: Set $A_j = 0$ $\forall j = 1,2,3,\ldots,HIS.$
8: {———- Step 3: Herd immunity evolution ———-}
9: **while** ($t \leq Max\_itr$) **do**
10: **for** $j = 1$ *to* HIS **do**
11: $is\_Corona\left(x^i(t+1)\right) = false$
12: **for** $i = 1$ *to* N **do**
13: **if** $(\boldsymbol{S_i = 1})$ **then**
14: $x_i^j(t+1) = C(x_i^j(t))$
15: $\boldsymbol{C\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^c(t))}$
16: $is\_Corona\left(x^i(t+1)\right) = true$
17: **else if** $(\boldsymbol{S_i = 0})$ **then**
18: $x_i^j(t+1) = N(x_i^j(t))$
19: $\boldsymbol{N\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^m(t))}$
20: **else if** $(\boldsymbol{S_i = 2})$ **then**
21: $x_i^j(t+1) = R(x_i^j(t))$
22: $\boldsymbol{R\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^v(t))}$
23: **else**
24: $x_i^j(t+1) = x_i^j(t)$
25: **end if**
26: **end for**
27: {———- Step 4: Update herd immunity population ———-}
28: **if** $(f(x^j(t+1)) \leq f(x^j(t)))$ **then**
29: $x_i^j(t) = x_i^j(t+1)$
30: **else**
31: $A^j = A^j + 1$
32: **end if**
33: **if** $f\left(x^j(t+1)\right) < \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 0 \wedge is\_Corona(x^j(t+1))$ **then**
34: $S_j = 1$
35: $A_j = 1$
36: **end if**
37: **if** $f\left(x^j(t+1)\right) > \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 1$ **then**
38: $S_j = 2$
39: $A_j = 0$
40: **end if**
41: {———- Step 5: Fatality condition ———-}
42: **if** $A_j \geq Max\_Age$ **then**
43: $x_i^j(t+1) = lb_i + (ub_i - lb_i) \times U(0,1)$, $\forall i = 1,2,\ldots,n$
44: $S_j = 0$
45: $A_j = 0$
46: **end if**
47: **end for**
48: $t = t + 1$
49: **end while**

4.2 Selection between the narrowed and expanded exploitation in the AO algorithm

In the AO algorithm, a random value *rand* is used to switch between the Narrowed and Expanded exploitation cases, as *rand* is a random number between 0 and 1.

To improve the algorithm's performance, we modified *rand* value as shown in Eq. 3.6.

$$R = rand \times \left(1 - \frac{t}{T}\right) \tag{3.6}$$

As t represents the current iteration and T is the maximum number of iterations, respectively.

4.3 Hybrid ECHIO with AO algorithms (MCHIAO)

The Coronavirus Herd Immunity Optimizer (CHIO) outperforms certain other biologically-inspired algorithms as one of the competitive human-based optimisation techniques. CHIO performed well in comparison to other optimisation techniques. However, CHIO is limited to local optima, which reduces the precision of complex global optimisation issues. Therefore, in high-dimensional space, it is simple to fall into a local optimum, and the Coronavirus Herd Immunity Optimizer (CHIO) algorithm has a low rate of convergence during the iterative process. However, AO's global exploration skills are limited, despite the fact that it possesses strong local exploitation capabilities. When AO algorithm is applied to optimize challenging, high-dimensional engineering problems, it may experience early convergence and poor global exploration. The AO encounters issues while optimizing challenging multidimensional problems, including subpar convergence behavior and subpar exploration efficiency. The Modified Coronavirus Herd Immunity Aquila Optimizer (MCHIAO), a novel metaheuristic optimizer, is then introduced to get over these limitations such as the low exploitation skills in CHIO and the insufficient exploration abilities of the AO algorithm and modify it to address feature selection problems.

Figure 6 below illustrates the flowchart of the proposed optimization algorithm MCHIAO and the pseudo-code of the proposed algorithm MCHIAO is illustrated in Algorithm 4.

The algorithm starts by initializing the MCHIAO parameters $\boldsymbol{HIS}$, $\boldsymbol{S_r}$, $\boldsymbol{\delta}$, $\boldsymbol{\alpha}$, and $\boldsymbol{Max_{age}}$. Herd immunity population is generated, and the fitness function of each search agent is calculated. Herd immunity evolution is initialized and the exploitation parameters $G_1, G_2, Levy(D)$ are updated. The selection between the exploration and the exploitation cases is based on the condition $t \leq \left(\frac{2}{3}\right) \times T$. The exploration phase is handled by the proposed ECHIO algorithm, as the cases are categorized as infected, susceptible, and immune cases based on the status vector. The exploitation phase consists of two categories, expanded exploitation and narrowed exploitation. The selection between the two exploitation cases is based on the generated formula R. The herd immunity population is updated, and the new positions of the agents are calculated. After the fatality condition is met, the best solution is obtained.

Algorithm 4 The proposed MCHIAO pseudo-code.

{——— Step 1: Initialize the MCHIAO parameters ———}
Initialize the parameters (HIS, S_r, δ, α and Max_{age}).
{——— Step 2: Generate herd immunity population ———}
$x_i^j = lb_i + (ub_i - lb_i) \times U(0,1)$, $\forall i = 1,2,\ldots,n$, and $\forall j = 1,2,\ldots,HIS$
Calculate the fitness of each search agent.
Set $S_j = 0$ $\forall j = 1,2,3,\ldots,HIS.$
Set $A_j = 0$ $\forall j = 1,2,3,\ldots,HIS.$
{———- Step 3: Herd immunity evolution ———-}
while ($t \leq Max\_itr$) **do**
  $is\_Corona\left(x^i(t+1)\right) = false$
  for $i = 1$ *to* N **do**
    Update the $G_1, G_2, Levy(D)$.
      if $t \leq \left(\frac{2}{3}\right) \times T$ **then**
        if ($S_i = 1$) **then**
          $x_i^j(t+1) = C(x_i^j(t))$
          $C\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^c(t))$
          $is\_Corona\left(x^i(t+1)\right) = true$
        else if ($S_i = 0$) **then**
          $x_i^j(t+1) = N(x_i^j(t))$
          $N\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^m(t))$
        else if ($S_i = 2$) **then**
          $x_i^j(t+1) = R(x_i^j(t))$
          $R\left(x_i^j(t)\right) = x_i^j(t) - m \times (x_i^j(t) - x_i^v(t))$
        else
          $x_i^j(t+1) = x_i^j(t)$
        end if
    else
      if $rand \leq 0.5$ **then**
        Expanded exploitation (X_3)
        Update the current solution using Eq. (3.13).
        If Fitness ($X_3(t+1)$))< $Fitness(X(t))$ **then**
        $\boldsymbol{X(t) = X_3(t+1)}$
        If Fitness ($X_3(t+1)$))< $Fitness(X_{best}(t))$ **then**
        $\boldsymbol{X_{best}(t) = X_3(t+1)}$
        end if
        end if
      else
        Narrowed exploitation (X_4)
        Update the current solution using Eq. (3.14).
        If Fitness ($X_4(t+1)$))< $Fitness(X(t))$ **then**
        $\boldsymbol{X(t) = X_4(t+1)}$
        If Fitness ($X_4(t+1)$))< $Fitness(X_{best}(t))$ **then**
        $\boldsymbol{X_{best}(t) = X_4(t+1)}$
        end if
        end if
      end if

  end for
  {———-Step 4: Update herd immunity population ———}
  if $(f(x^j(t+1)) \leq f(x^j(t)))$ **then**
    $x_i^j(t) = x_i^j(t+1)$
  else
    $A^j = A^j + 1$
  end if
  if $f\left(x^j(t+1)\right) < \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 0 \wedge is\_Corona(x^j(t+1))$ **then**
    $S_j = 1$
    $A_j = 1$
  end if
  if $f\left(x^j(t+1)\right) > \frac{f(x^j(t+1))}{\Delta f(x)} \wedge S_j = 1$ **then**
    $S_j = 2$
    $A_j = 0$
  end if
  {——— Step 5: Fatality condition ———}
  if $A_j \geq Max\_Age$ **then**
    $x_i^j(t+1) = lb_i + (ub_i - lb_i) \times U(0,1), \quad \forall i = 1,2,\ldots,n$
    $S_j = 0$
    $A_j = 0$
  end if
  $t = t + 1$
end while
return The best solution (X_{best}).

Table 6 Summary of the 29 CEC2017benchmark functions

Type	Number	Function name	$f_i(x^*)$
Unimodal	1	Shifted and rotated Bent Cigar function	100
	3	Shifted and rotated Zakharov function	300
Multimodal	4	Shifted and rotated Rosenbrock's function	400
	5	Shifted and rotated Rastrigin's function	500
	6	Shifted and rotated Expanded Schaffer's F6 function	600
	7	Shifted and rotated Lunacek Bi_Rastrigin function	700
	8	Shifted and rotated non-continuous Rastrigin's function	800
	9	Shifted and rotated Levy function	900
	10	Shifted and rotated Schwefel's function	1000
Hybrid	11	Hybrid function 1 (N—3)	1100
	12	Hybrid function 2 (N—3)	1200
	13	Hybrid function 3 (N—3)	1300
	14	Hybrid function 4 (N—4)	1400
	15	Hybrid function 5 (N—4)	1500
	16	Hybrid function 6 (N—4)	1600
	17	Hybrid function 6 (N—5)	1700
	18	Hybrid function 6 (N—5)	1800
	19	Hybrid function 6 (N—5)	1900
	20	Hybrid function 6 (N—6)	2000
Composition	21	Composition function 1 (N—3)	2100
	22	Composition function 2 (N—3)	2200
	23	Composition function 3 (N—4)	2300
	24	Composition function 4 (N—4)	2400
	25	Composition function 5 (N—5)	2500
	26	Composition function 6 (N—5)	2600
	27	Composition function 7 (N—6)	2700
	28	Composition function 8 (N—6)	2800
	29	Composition function 9 (N—3)	2900
	30	Composition function 10 (N—3)	3000

5 Experimental results & analysis

The performance of the proposed MCHIAO is evaluated for variants benchmark functions including 23 CEC 2005, 29 CEC 2017, 10 CEC 2019, 24 unimodal, 44 multimodal test functions, and six different real-world problems. The proposed MCHIAO is compared as well against twelve state-of-the-art algorithms (GOA, MFO, MPA, GWO, HHO, SSA, WOA, IAO, NOA, NGO, AO, and CHIO), conducting Wilcoxon statistical analysis to statistically validate all outcomes. MATLAB/SIMULINK® 2018b is used to accomplish all simulations. The metrics used to define the performance of the algorithm against the twelve competitive state-of-the-art algorithms are mean, standard deviation, rank, percentage, and total rank. The experimental results include the mean value (Mean), standard deviation (Std), and rank order (Rank). In the case of the minimum problem, the algorithm's discovery of the optimal solution is more likely the smaller the best value; similarly, the algorithm's comprehensive optimization capabilities are enhanced by the lower the mean value; and the algorithm's stability is improved by the smaller standard deviation. The algorithms are ranked using a ranking based on the mean value, which allows for a more intuitive comparison based on the magnitude of the mean value. The algorithm with the highest ranking among those ranked equally is shown.

The Wilcoxon signed-rank test (also called the Wilcoxon signed-rank sum test) is a nonparametric test used to compare data. For within-group comparison of algorithms

Table 7 MCHIAO performance against CHIO and AO algorithms is conducted on 29 CEC-2017 functions

Function	Algorithm Indices	AO [31]	CHIO [30]	MCHIAO
F1	Mean	74,460,200.6	1,760,094,431	**2866.08648**
	Std	123,756,337	1,467,167,371	3308.02416
	Rank	2	3	**1**
F3	Mean	4187.2174	4311.95317	**300.000076**
	Std	914.824188	1164.12591	0.00027993
	Rank	2	3	**1**
F4	Mean	436.039253	739.467171	**402.605918**
	Std	25.1789584	405.302522	1.71126457
	Rank	2	3	**1**
F5	Mean	532.319243	553.387742	**519.614328**
	Std	12.7198787	7.21201353	10.7631725
	Rank	2	3	**1**
F6	Mean	627.236527	636.244568	**603.822265**
	Std	7.84502623	5.96254295	3.41594141
	Rank	2	3	**1**
F7	Mean	761.598213	803.160009	**742.780727**
	Std	16.0291469	13.6082671	8.51299127
	Rank	2	3	**1**
F8	Mean	828.145579	839.343639	**817.148124**
	Std	4.7890986	8.26597153	6.15251227
	Rank	2	3	**1**
F9	Mean	1096.66657	1402.77982	**946.602451**
	Std	91.7687802	140.589628	79.1275347
	Rank	2	3	**1**
F10	Mean	1995.0516	3019.22908	**1931.62902**
	Std	264.267774	147.056249	328.80526
	Rank	2	3	**1**
F11	Mean	1390.7056	1386.41834	**1128.39922**
	Std	302.652315	102.101563	16.039182
	Rank	3	2	**1**
F12	Mean	5,148,551.08	10,944,616.1	**26,913.1807**
	Std	5,206,061.52	9,688,908.59	18,147.663
	Rank	2	3	**1**
F13	Mean	19,760.1968	35,853.9812	**1651.34374**
	Std	12,495.3061	16,505.8753	349.271564
	Rank	2	3	**1**
F14	Mean	4031.33268	6797.48159	**1465.48509**
	Std	2033.50287	825.477957	34.2525909
	Rank	2	3	**1**
F15	Mean	8444.98355	16,261.4922	**1590.52906**
	Std	4496.8501	8848.00604	81.6176787
	Rank	2	3	**1**
F16	Mean	1887.44346	1965.59533	**1666.42854**
	Std	88.854808	126.235428	79.8004617
	Rank	2	3	**1**
F17	Mean	1786.22388	1795.15872	**1744.94634**
	Std	13.7815982	20.7812173	10.8849663
	Rank	2	3	**1**
F18	Mean	25,885.9274	112,737.744	**2609.51094**
	Std	14,268.618	82,697.1716	1257.63002
	Rank	2	3	**1**
F19	Mean	76,665.4739	27,301.7071	**1954.85267**
	Std	128,019.754	7820.54704	37.8865398
	Rank	2	3	**1**
F20	Mean	2178.37203	2239.36288	**2040.02629**
	Std	66.5526714	80.4024286	6.17522688
	Rank	2	3	**1**
F21	Mean	2303.13739	2314.37698	**2243.56959**
	Std	47.9077996	60.8143985	62.0238696
	Rank	2	3	**1**
F22	Mean	2326.13568	3814.29715	**2302.3156**
	Std	18.6111765	580.323684	2.82049757
	Rank	2	3	**1**
F23	Mean	2653.20969	2658.01634	**2622.52767**
	Std	14.6819134	6.92611101	13.2639282
	Rank	2	3	**1**
F24	Mean	2788.92455	2793.85698	**2664.19632**
	Std	26.2276458	16.0096304	142.219745
	Rank	2	3	**1**
F25	Mean	2951.91226	3042.9232	**2920.28895**
	Std	29.4727073	56.9908093	24.5533739
	Rank	2	3	**1**
F26	Mean	3104.57715	3963.30913	**2919.55391**
	Std	190.634688	424.654747	158.402239
	Rank	2	3	**1**
F27	Mean	3108.0177	3106.14969	**3094.15989**
	Std	10.2052865	19.0242259	1.86615578
	Rank	3	2	**1**
F28	Mean	3483.35556	**3244.76013**	3290.72305
	Std	90.2595564	7.17012236	147.790981
	Rank	3	**1**	2
F29	Mean	3269.65681	3330.88552	**3172.72036**
	Std	36.3736394	133.880124	21.4615926
	Rank	2	3	**1**
F30	Mean	3,382,703.95	7,098,205.17	**1,471,154.92**
	Std	2,400,644.58	3,198,242.47	1,494,262.41
	Rank	2	3	**1**
Percentage		2.10344828	2.86206897	1.03448276
Total Rank		2	3	**1**

Table 8 MCHIAO performance against different algorithms on 29 CEC2017 benchmark function problems

Function	Algorithm Indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F1	Mean	24,234.6839	384,350,022	3812.90148	127,394,208	1,132,494.89	3437.75349	63,704,979.2	74,460,200.6	25,044,536.7	5,654,322,977	3201.26648	1,760,094,431	**2866.08648**
	Std	75,035.7168	657,485,212	3626.45268	178,056,761	604,404.663	2165.05783	101,602,674	123,756,337	35,203,099.3	2,161,689,959	2990.00475	1,467,167,371	3308.02416
	Rank	5	11	4	10	6	3	8	9	7	13	2	12	**1**
F3	Mean	1050.67866	14,118.5909	405.626063	6352.61687	724.489293	590.553327	8471.60756	4187.2174	2336.27161	15,999.298	718.626861	4311.95317	**300.000076**
	Std	1229.14703	12,339.8721	91.173181	4202.73146	312.595896	280.878195	5869.08823	914.824188	1068.68977	3714.31173	467.15951	1164.12591	0.00027993
	Rank	6	12	2	10	5	3	11	8	7	13	4	9	**1**
F4	Mean	410.044841	432.443527	410.932897	437.905009	425.636884	405.430734	463.455243	436.039253	420.303811	670.041656	404.651011	739.467171	**402.605918**
	Std	20.4666154	53.3036617	17.3247249	29.6749628	28.1483434	1.41087226	61.2555495	25.1789584	25.142443	117.261942	2.0388915	405.302522	1.71126457
	Rank	4	8	5	10	7	3	11	9	6	12	2	13	**1**
F5	Mean	574.039752	532.35278	521.618773	519.876264	562.466846	525.869128	559.689157	532.319243	531.176393	590.182709	534.499806	553.387742	**519.614328**
	Std	32.4024809	10.6908846	7.64918112	9.24151794	20.7592596	12.0574148	29.2685549	12.7198787	9.25732663	11.8897687	13.0078034	7.21201353	10.7631725
	Rank	12	7	3	2	11	4	10	6	5	13	8	9	**1**
F6	Mean	640.450441	604.209695	**601.483926**	606.339369	639.738447	616.604144	632.079576	627.236527	621.769786	650.758822	624.126103	636.244568	603.822265
	Std	23.2530346	3.38706494	2.39208791	3.96845167	11.7512831	14.8088112	13.9329682	7.84502623	6.47385999	7.36393321	13.0956761	5.96254295	3.41594141
	Rank	12	3	**1**	4	11	5	9	8	6	13	7	10	2
F7	Mean	**735.208757**	739.236207	735.758655	740.488355	793.161281	750.813586	769.973472	761.598213	772.847964	832.147656	751.866776	803.160009	742.780727
	Std	10.8629736	21.7430202	6.57606067	15.0911821	19.5446421	16.600962	6.62465757	16.0291469	23.2570247	25.4416898	10.7802371	13.6082671	8.51299127
	Rank	**1**	3	2	4	11	6	9	8	10	13	7	12	5
F8	Mean	858.919011	832.434635	817.235031	828.11774	832.197467	839.656094	838.550478	828.145579	829.092349	875.48516	820.252451	839.343639	**817.148124**
	Std	26.9750592	14.5363152	6.04180069	16.3203928	13.2972319	11.6941145	13.0369861	4.7890986	8.39066808	4.9599343	4.87152984	8.26597153	6.15251227
	Rank	12	8	2	4	7	11	9	5	6	13	3	10	**1**
F9	Mean	2616.45617	1284.65895	**921.604508**	950.733251	1525.24469	1057.79943	1526.69257	1096.66657	1103.76351	1763.43561	1104.5689	1402.77982	946.602451
	Std	1013.09777	530.438349	23.3017398	28.1701897	341.306284	235.384285	423.204561	91.7687802	165.186244	246.917907	124.161038	140.589628	79.1275347
	Rank	13	8	**1**	3	10	4	11	5	6	12	7	9	2
F10	Mean	2432.83361	2219.43165	**1690.18414**	1932.45622	2084.56878	1816.07838	2152.04689	1995.0516	1879.93045	3005.18633	2048.93097	3019.22908	1931.62902
	Std	488.357249	225.075112	341.440251	453.074132	138.360905	224.658029	219.291905	264.267774	397.080497	262.615309	242.042855	147.056249	328.80526
	Rank	11	10	**1**	5	8	2	9	6	3	12	7	13	4
F11	Mean	1193.28104	1293.69507	**1110.61967**	1450.86016	1196.43371	1195.58756	1239.6156	1390.7056	1318.96167	3162.7895	1137.48768	1386.41834	1128.39922
	Std	46.0896641	193.633385	4.04461661	1131.57096	77.2307589	50.6960342	102.25057	302.652315	180.421189	1265.68202	29.431494	102.101563	16.039182
	Rank	4	8	**1**	12	6	5	7	11	9	13	3	10	2
F12	Mean	1,941,411.13	3,230,144.04	169,817.894	1,175,587.46	4,740,745.02	4,475,326.98	3,451,914.62	5,148,551.08	6,022,207.73	121,629,505	**14,708.0355**	10,944,616.1	26,913.1807
	Std	2,385,330.63	4,306,870.23	308,501.855	1,142,062.62	4,685,270.29	4,987,771.81	3,935,688.09	5,206,061.52	6,409,990.66	60,299,803.4	9705.36079	9,688,908.59	18,147.663
	Rank	5	6	3	4	9	8	7	10	11	13	**1**	12	2

Table 8 (continued)

Function	Algorithm Indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F13	Mean	18,071.8016	16,320.2497	2068.25076	13,411.0646	17,189.255	19,955.5469	20,548.4062	19,760.1968	11,821.8392	3,906,234.78	6452.18726	35,853.9812	**1651.34374**
	Std	13,951.2531	12,167.8648	1939.40073	7434.57976	10,220.4791	10,868.4351	12,543.9061	12,495.3061	9770.91606	4,126,100.17	4534.53043	16,505.8753	349.271564
	Rank	8	6	2	5	7	10	11	9	4	13	3	12	**1**
F14	Mean	5258.39351	3693.11394	**1422.07814**	3858.21385	2105.64024	5625.78028	2874.17782	4031.33268	2675.75708	51,648.4207	1586.38785	6797.48159	1465.48509
	Std	4602.34237	2696.66428	11.3906915	2624.56672	937.207699	5139.60718	1689.35441	2033.50287	1451.85112	151,640.998	216.135938	825.477957	34.2525909
	Rank	10	7	**1**	8	4	11	6	9	5	13	3	12	2
F15	Mean	17,727.8956	16,332.2624	1677.03563	8581.88498	8825.75332	21,177.4117	11,849.3491	8444.98355	6100.7506	33,918.6672	2546.22903	16,261.4922	**1590.52906**
	Std	18,122.0129	19,785.8271	591.297039	8009.88449	3382.30023	18,339.9927	9158.67497	4496.8501	2495.90433	29,454.2704	1075.27823	8848.00604	81.6176787
	Rank	11	10	2	6	7	12	8	5	4	13	3	9	**1**
F16	Mean	2009.31196	1793.76592	1679.71559	1918.52752	1965.61551	1789.59265	1917.9241	1887.44346	1795.58049	2274.08218	1764.874	1965.59533	**1666.42854**
	Std	142.770523	110.845909	82.6569113	186.232752	122.862334	118.479216	170.774164	88.854808	153.902072	141.937109	136.418653	126.235428	79.8004617
	Rank	12	5	2	9	11	4	8	7	6	13	3	10	**1**
F17	Mean	1876.71185	1778.3551	**1738.43287**	1760.33852	1774.324	1780.70641	1817.63203	1786.22388	1774.50773	1894.38954	1767.29668	1795.15872	1744.94634
	Std	75.1000921	40.7978225	14.9240171	15.8822163	35.3816737	11.6509028	63.7164698	13.7815982	14.1090943	66.3988771	9.63607056	20.7812173	10.8849663
	Rank	12	7	**1**	3	5	8	11	9	6	13	4	10	2
F18	Mean	19,515.1453	20,965.7366	3168.99116	32,042.3228	15,350.6384	21,875.2799	16,729.3252	25,885.9274	33,881.8875	24,466,122.7	7866.29368	112,737.744	**2609.51094**
	Std	13,807.1641	13,454.9299	1937.19595	9390.77927	11,871.1252	11,617.081	12,389.5047	14,268.618	19,307.5767	38,595,045.5	5742.97786	82,697.1716	1257.63002
	Rank	6	7	2	10	4	8	5	9	11	13	3	12	**1**
F19	Mean	8649.21856	24,437.4087	1926.46562	26,542.5823	31,973.4595	9659.75991	76,013.5305	76,665.4739	14,688.2551	129,612.549	3885.29231	27,301.7071	**1954.85267**
	Std	7577.22054	37,498.1392	69.6126399	65,802.9855	43,182.2898	8609.70585	155,939.764	128,019.754	12,130.3401	99,794.7607	2982.90277	7820.54704	37.8865398
	Rank	7	8	5	4	13	3	10	9	6	12	2	11	**1**
F20	Mean	2325.72252	2206.31025	2040.98833	2143.80493	2154.99208	2117.24535	2162.31782	2178.37203	2153.21213	2292.08791	2090.35889	2239.36288	**2040.02629**
	Std	72.4073498	67.7129107	8.1207705	59.3484477	20.3016442	123.876889	63.3052936	66.5526714	45.1380312	49.8167371	72.4507961	80.4024286	6.17522688
	Rank	13	10	2	5	7	4	8	9	6	12	3	11	**1**
F21	Mean	2368.26928	2327.45291	2262.01995	2322.47578	2313.24067	2309.81803	2321.22794	2303.13739	2296.89273	2368.98018	2318.46478	2314.37698	**2243.56959**
	Std	30.8904929	37.3237114	57.3711418	10.4348315	75.5031248	44.5494251	62.7208725	47.9077996	50.0355132	33.7243	37.2745436	60.8143985	62.0238696
	Rank	12	11	2	10	6	5	9	4	3	13	8	7	**1**
F22	Mean	2957.32772	2304.24035	2302.8021	2352.82845	2312.93478	2499.63584	2372.43157	2326.13568	2317.59628	2721.20266	2310.1808	3814.29715	**2302.3156**
	Std	658.096998	21.5405264	2.4918182	117.619417	4.1449313	443.498619	185.886607	18.6111765	10.8656624	186.649468	32.8773183	580.323684	2.82049757
	Rank	12	3	2	8	5	10	9	7	6	11	4	13	**1**
F23	Mean	2690.61621	2632.8756	2626.18025	2636.05811	2696.60372	2622.76214	2647.88402	2653.20969	2640.94773	2708.81246	2636.92976	2658.01634	**2622.52767**
	Std	56.9996064	9.97922375	8.80990053	9.64959139	23.7698572	8.51577092	20.2736342	14.6819134	12.5204493	23.0868432	9.66908101	6.92611101	13.2639282
	Rank	11	4	3	5	12	2	8	9	7	13	6	10	**1**
F24	Mean	2786.51087	2788.82442	2765.09346	2756.12154	2846.30353	2749.03531	2791.9762	2788.92455	2774.08256	2829.97186	2671.16052	2793.85698	**2664.19632**
	Std	10.754943	6.20131576	8.98039609	13.7780768	41.6249992	6.2065482	39.0893125	26.2276458	4.80616326	8.50980217	149.072439	16.0096304	142.219745

Table 8 (continued)

Function	Algorithm Indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
	Rank	7	8	5	4	13	3	10	9	6	12	2	11	**1**
F25	Mean	**2908.27917**	2951.43433	2920.99081	2955.13106	2927.46363	2944.86707	2973.62032	2951.91226	2938.35523	3170.3599	2937.10825	3042.9232	2920.28895
	Std	89.2721981	29.1791643	23.6088574	40.7385484	55.8619009	12.6820818	33.937846	29.4727073	20.3369417	133.088119	19.4089097	56.9908093	24.5533739
	Rank	**1**	8	3	10	4	7	11	9	6	13	5	12	2
F26	Mean	3457.72895	3144.96787	**2883.62195**	3449.82039	3597.47673	3136.28732	3465.10118	3104.57715	3036.05749	3738.52856	3109.36625	3963.30913	2919.55391
	Std	710.158852	307.945597	129.683086	499.530696	577.301	517.599462	463.442104	190.634688	256.173157	241.959118	140.758955	424.654747	158.402239
	Rank	9	7	**1**	8	11	6	10	4	3	12	5	13	2
F27	Mean	3166.73034	3096.31278	3100.11481	3096.47829	3180.15634	3096.74641	3133.44264	3108.0177	3102.23132	3184.82602	3098.18074	3106.14969	**3094.15989**
	Std	57.6419448	3.75458872	13.3246995	1.70328587	52.3440543	2.27475595	23.9262984	10.2052865	5.63560494	39.9353628	6.28373752	19.0242259	1.86615578
	Rank	11	2	6	3	12	4	10	9	7	13	5	8	**1**
F28	Mean	3271.73861	3365.56304	3307.57848	3452.969	3374.7483	3341.01326	3547.78765	3483.35556	3439.45739	3629.46124	3320.45939	**3244.76013**	3290.72305
	Std	160.657307	62.7804464	132.475445	56.2142134	122.272425	108.577038	199.074846	90.2595564	108.762594	131.294559	129.384986	7.17012236	147.790981
	Rank	2	7	4	10	8	6	12	11	9	13	5	**1**	3
F29	Mean	3331.39818	3280.44162	3194.86265	3228.09404	3362.5328	3260.37233	3398.95446	3269.65681	3281.99195	3482.27562	3241.77496	3330.88552	**3172.72036**
	Std	103.2321	68.8028353	36.3615256	79.8427349	83.380995	34.5583676	142.324606	36.3736394	73.4953664	140.116864	49.2741553	133.880124	21.4615926
	Rank	10	7	2	3	11	5	12	6	8	13	4	9	**1**
F30	Mean	1,370,724.89	631,304.387	196,091.01	1,945,560.46	3,616,308.02	1,266,635.13	3,555,262.55	3,382,703.95	1,914,514.24	11,104,762.8	**171,127.971**	7,098,205.17	1,471,154.92
	Std	1,184,118.17	333,851.09	294,395.698	2,193,474.36	3,589,485.26	969,370.776	3,652,622.35	2,400,644.58	2,308,233.76	5,877,266.55	166,805.736	3,198,242.47	1,494,262.41
	Rank	5	3	2	8	11	4	10	9	7	13	**1**	12	6
Percentage	8.4137931	**7.03448276**	**2.48275862**	**6.44827586**	**8.34482759**	**5.72413793**	**9.27586207**	**7.86206897**	**6.4137931**	**12.6896552**	**4.13793103**	**10.4137931**	**1.75862069**	
Total Rank	10	7	2	6	9	4	11	8	5	13	3	12	**1**	

Table 9 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for 29 CEC2017 benchmark function problems

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44	1.63952E-44
F3	5.88901E-73	7.3172E-154	1.75602E-49	2.4609E-129	9.9195E-105	5.46488E-61	7.8283E-136	1.1555E-134	2.1782E-118	8.5692E-144	1.26088E-78	5.6982E-136
F4	1.99337E-46	7.43735E-92	5.27251E-55	9.12356E-98	9.4086E-127	2.78571E-33	1.4183E-110	7.8956E-122	3.9821E-114	0.524607699	1.79811E-28	5.0303E-153
F5	1.3119E-148	3.3936E-83	2.4266E-16	7.83042E-54	3.1795E-141	1.00686E-80	3.5115E-128	4.3695E-127	2.1107E-124	8.4841E-162	5.74512E-97	1.3436E-144
F6	1.1105E-140	1.86682E-73	0.011159909	2.7351E-40	7.4384E-143	1.7585E-101	1.6756E-124	5.6349E-132	7.5835E-119	5.4059E-165	1.9197E-109	2.0771E-150
F7	2.76342E-32	3.63571E-45	6.97319E-05	6.30437E-51	2.9409E-135	4.7319E-104	1.2501E-106	8.1965E-124	1.0043E-127	5.2045E-165	1.75974E-77	5.6381E-151
F8	1.3427E-148	1.65428E-93	2.49589E-18	4.06714E-97	1.6136E-101	2.2323E-126	4.6871E-107	1.275E-121	2.0971E-117	5.1393E-165	2.06048E-26	5.5613E-142
F9	7.6008E-160	2.3126E-131	0.389522908	1.98694E-56	2.8935E-148	1.9243E-122	1.4033E-143	1.1295E-125	3.6319E-129	4.57363E-06	4.458E-118	3.4511E-155
F10	6.0277E-134	1.50552E-93	0.100985836	7.4501E-99	7.25219E-98	0.857401702	9.96967E-89	1.9003E-129	1.03045E-57	1.509E-123	9.77932E-86	1.0943E-161
F11	6.22014E-77	1.5643E-110	3.48561E-38	3.79E-123	5.0374E-115	2.62238E-87	9.21253E-94	2.5528E-140	1.8415E-137	1.05908E-23	8.86266E-18	3.4196E-135
F12	3.26552E-90	2.55718E-98	2.2765E-36	1.94753E-91	2.4551E-110	1.4492E-113	1.33394E-96	9.83E-123	9.5622E-124	6.2827E-144	0.077955014	3.6864E-135
F13	5.3351E-113	1.78569E-76	0.005956617	1.2546E-107	3.42001E-83	1.1075E-118	1.213E-102	1.1797E-135	8.322E-122	5.0148E-129	5.08696E-33	4.5299E-135
F14	5.2677E-138	5.0409E-137	8.52122E-85	1.9381E-138	2.7243E-123	2.4474E-140	8.2099E-135	6.9154E-141	7.4844E-137	3.6967E-124	4.0046E-106	6.9462E-148
F15	2.6951E-135	4.1875E-124	8.30834E-14	5.216E-112	8.1987E-109	6.1788E-140	2.4222E-116	7.4192E-112	6.9169E-103	5.1592E-145	3.77083E-70	2.6254E-147
F16	8.9144E-143	2.21725E-90	2.96066E-33	2.8711E-123	8.5824E-142	1.4289E-104	5.015E-120	1.3675E-133	1.3662E-124	5.7264E-165	9.54509E-93	1.0274E-145
F17	1.8532E-156	5.87063E-92	0.033042588	1.934E-109	1.1846E-131	1.2061E-133	3.8844E-130	7.7184E-133	1.2271E-126	5.7298E-165	7.05522E-71	2.7339E-133
F18	3.3188E-100	2.3101E-78	4.24128E-14	2.669E-114	3.69094E-46	6.49E-104	1.09004E-45	1.9324E-132	7.67E-131	1.5568E-148	1.48411E-08	6.5066E-127
F19	4.1395E-103	9.5445E-120	9.3743E-42	6.5661E-127	1.4192E-149	4.8824E-115	1.2067E-139	2.2865E-144	4.6315E-130	1.8344E-146	5.64255E-58	1.4945E-135
F20	2.0781E-155	1.0529E-112	2.86E-17	3.00072E-83	4.6294E-99	5.9438E-82	1.1783E-80	4.7176E-137	5.6206E-121	5.7935E-138	1.04449E-62	5.0846E-148

Table 9 (continued)

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F21	1.6058E-157	3.6059E-145	6.1085E-102	3.2549E-148	8.6173E-148	8.0276E-144	3.1179E-144	4.2607E-148	2.0831E-145	5.7053E-165	6.7574E-144	8.6166E-152
F22	3.1614E-153	8.96835E-26	9.00399E-16	6.4641E-117	1.49603E-98	8.8596E-140	9.9364E-113	1.7148E-118	5.9091E-111	1.7309E-134	1.26621E-58	9.3449E-164
F23	3.7891E-148	6.7937E-75	8.54235E-75	2.8939E-111	5.6829E-151	9.3651E-36	2.1078E-109	1.7091E-139	2.2729E-131	5.7451E-165	7.60376E-96	3.9736E-134
F24	6.3364E-143	1.3127E-136	1.1979E-137	2.7418E-135	1.725E-161	1.1696E-136	5.3581E-138	1.8359E-152	9.9355E-153	7.2683E-154	2.07227E-79	9.7272E-151
F25	1.4286E-07	7.2969E-100	0.000584416	2.0221E-108	2.9007E-107	3.1453E-101	9.4491E-117	1.8982E-127	1.2639E-114	5.7488E-165	2.50179E-76	5.2873E-144
F26	7.869E-135	2.99338E-94	0.798214426	2.8034E-134	3.5179E-149	1.1621E-109	9.6134E-132	4.0897E-121	3.53E-115	5.6856E-165	4.36137E-90	1.0116E-159
F27	3.0571E-161	8.08385E-83	2.1126E-120	7.9134E-102	1.5499E-163	6.049E-100	3.918E-157	4.5573E-144	1.6962E-137	5.3572E-165	5.9375E-102	6.5326E-138
F28	8.32008E-14	3.0784E-122	7.23995E-58	1.1315E-139	2.5137E-140	1.8553E-115	4.3064E-156	5.7423E-155	8.0459E-150	5.291E-165	1.40655E-84	1.53242E-06
F29	1.5807E-145	2.3201E-133	5.78403E-90	6.3539E-127	7.0623E-153	8.4162E-133	1.3889E-151	7.083E-152	6.1028E-147	1.0906E-162	5.1711E-122	3.3209E-153
F30	6.97772E-17	4.2246E-121	1.57894E-74	1.92266E-97	4.3313E-156	0.00016145	2.3568E-137	2.3058E-149	4.5344E-124	2.8098E-132	8.89057E-76	3.9767E-159

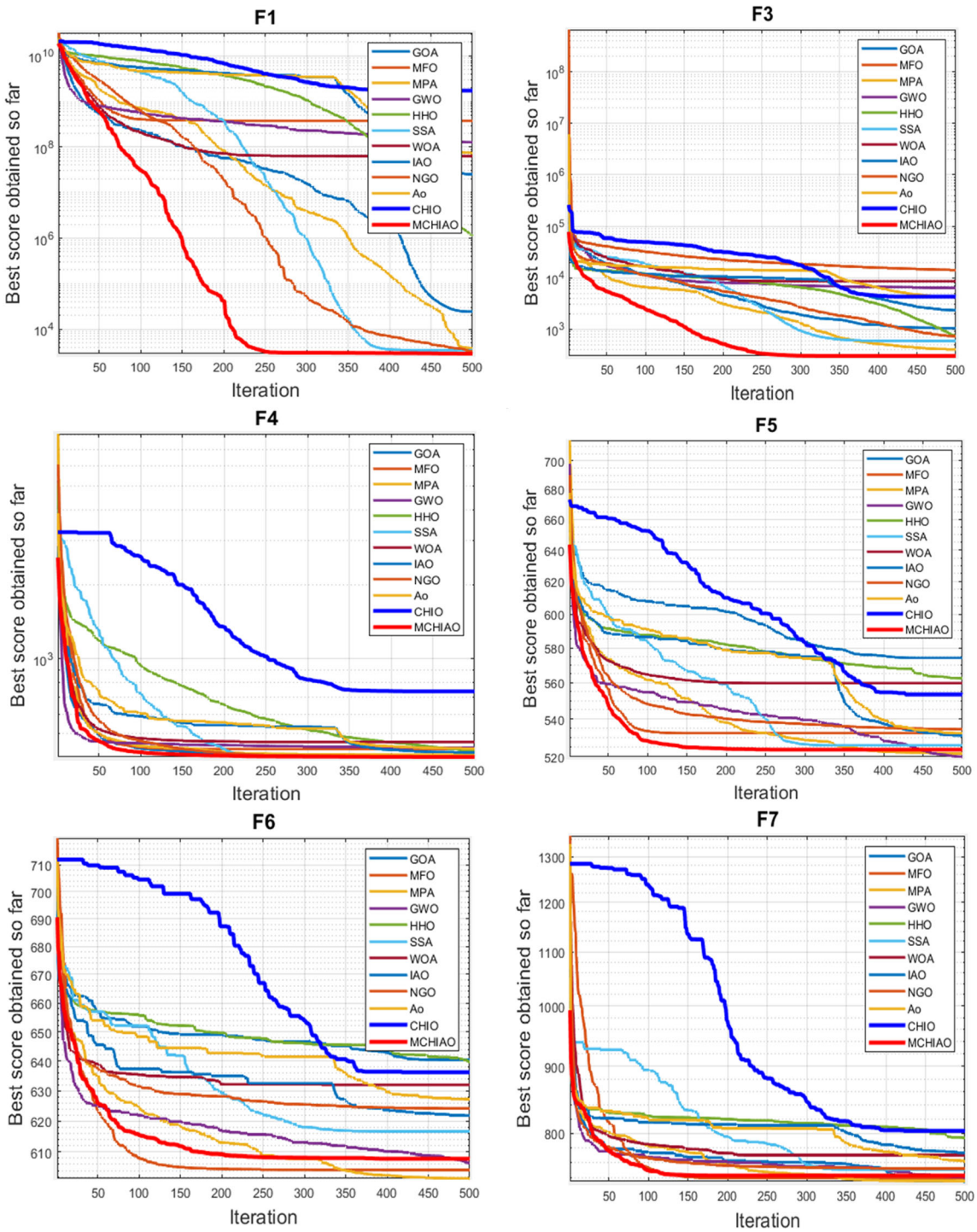


Fig. 8 The 29 CEC2017 functions and convergence plots

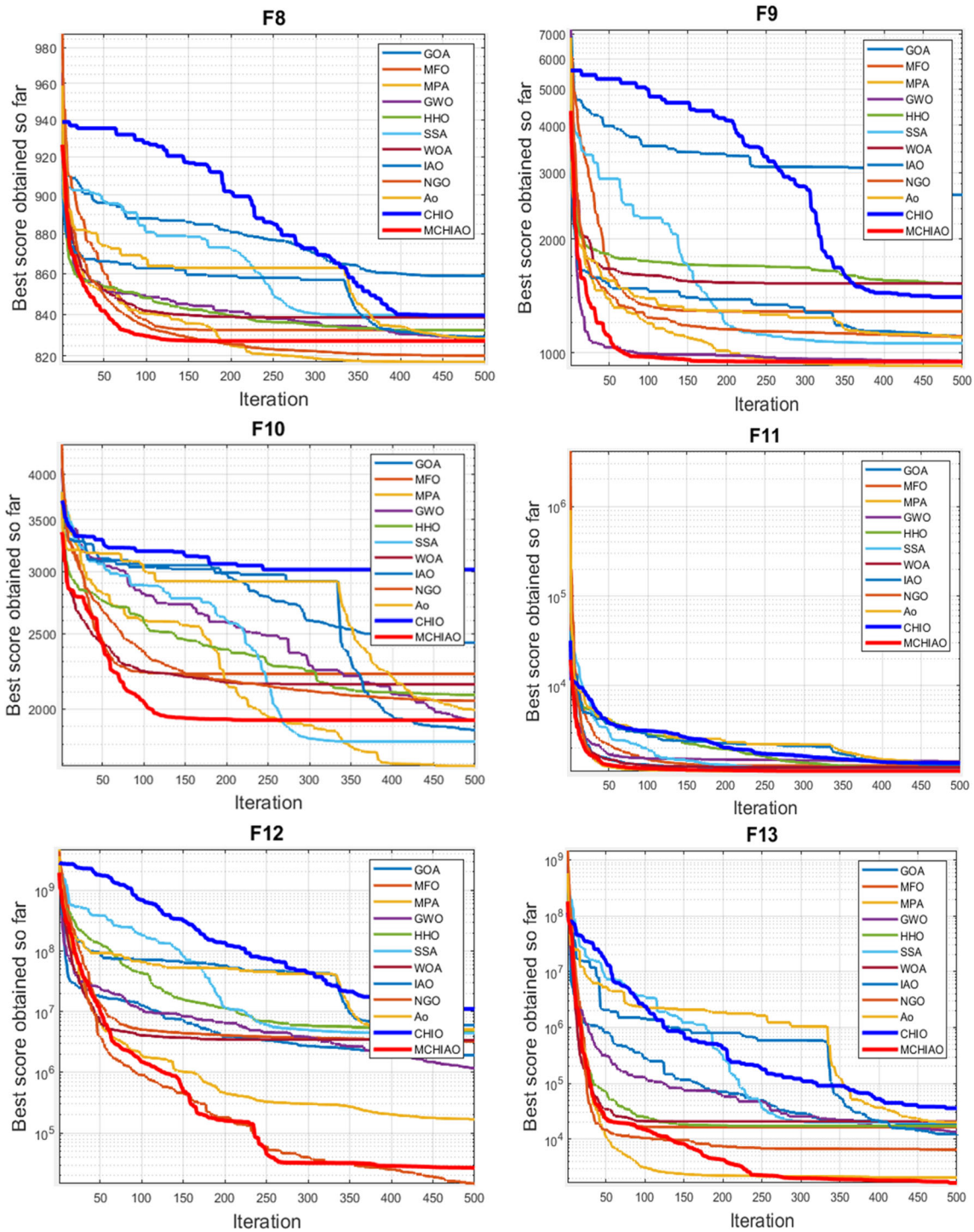


Fig. 8 continued

generally Wilcoxon signed-rank test can be applied to analyze each algorithm within-group. We have calculated the mean result for each algorithm as all of the algorithms are executed with a maximum of 500 iterations and 51 runs. The Friedman average rank test is used for evaluating the performance of the various optimization algorithms, with the best result of each measure in each function highlighted in bold in the result tables.

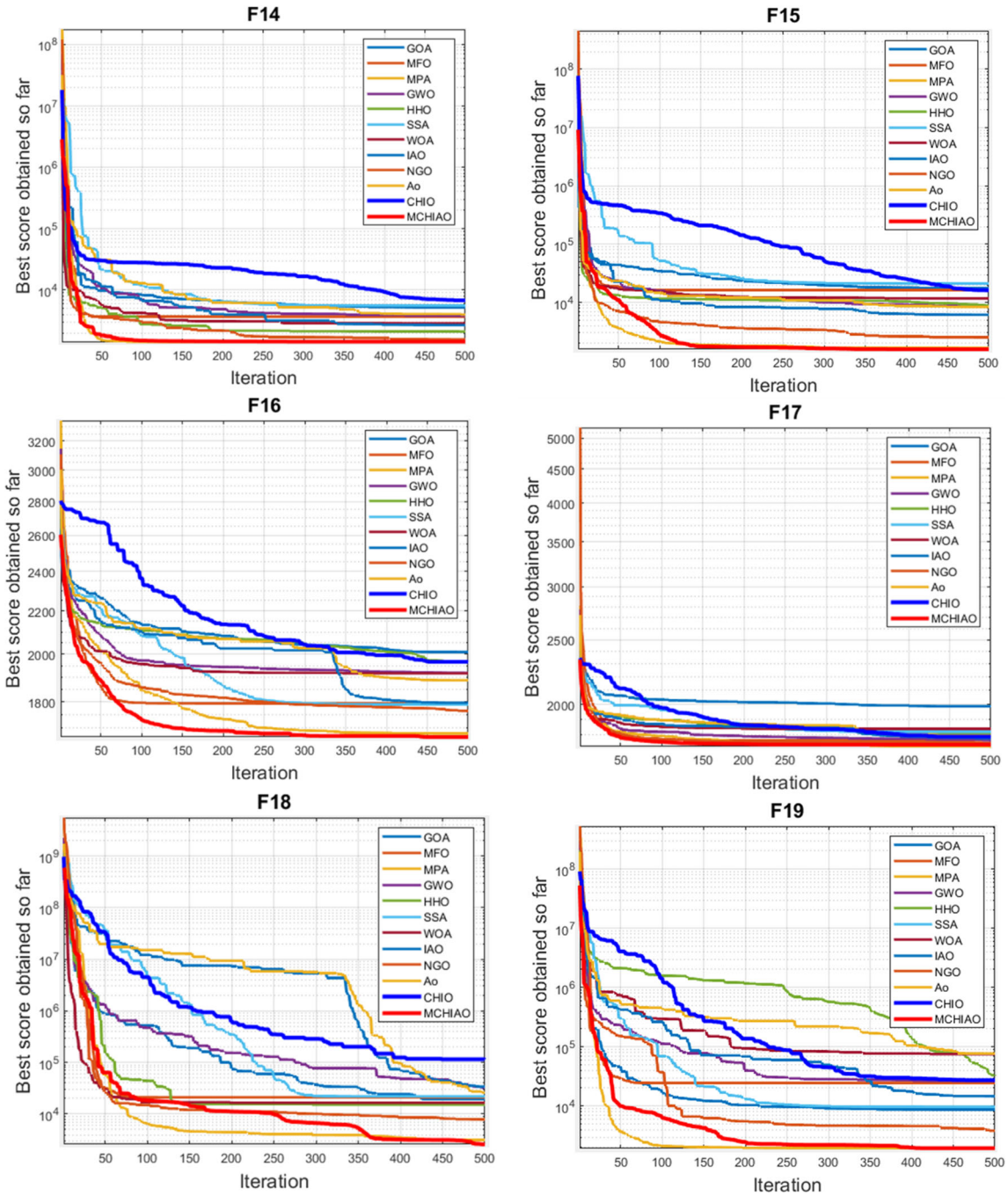


Fig. 8 continued

5.1 Case1: MCHIAO performance through 23 CEC 2005 benchmark

In this section, 23 benchmark functions are considered to evaluate the proposed MCHIAO's performance. Each of these test functions is a minimization problem of varying size and difficulty. Table 2 shows the benchmark functions, where Dim represents the function's dimension, Range represents the function's search space boundaries, and f_{min} is optimum.

5.1.1 MCHIAO Vs AO and CHIO

MCHIAO is run against the original CHIO and AO algorithms on the different functions of 23 benchmark. Numerical results in Table 3 demonstrate the mean,

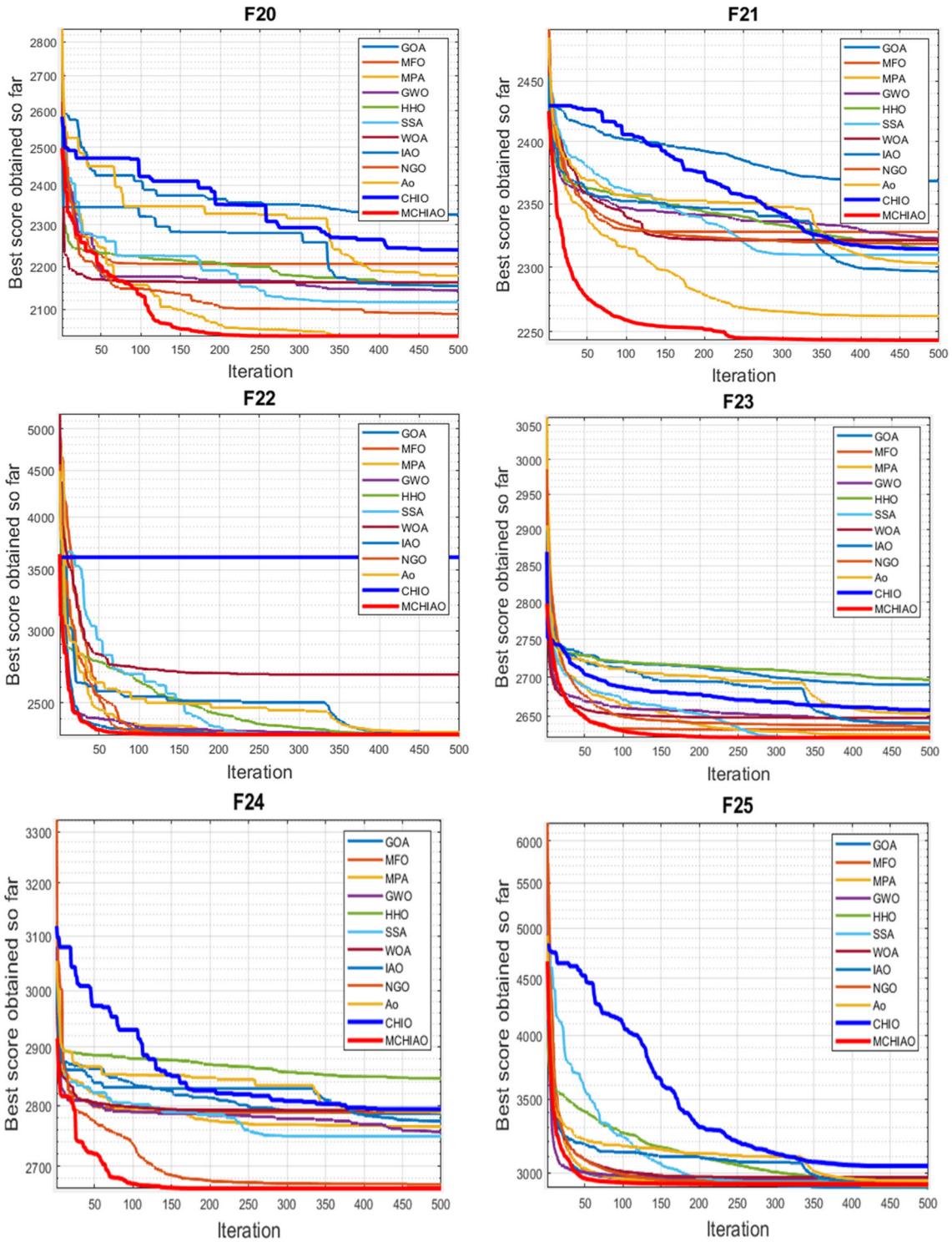


Fig. 8 continued

standard deviation, and rank of the used algorithms on every function, as it ranked first in all of the 23 benchmark functions (from Table 2) except for F_13 (x) where AO ranked the first. MCHIAO achieved total rank 1.

5.1.2 MCHIAO vs other algorithms

The proposed MCHIAO algorithm is compared against twelve different algorithms in a simulated experiment. These algorithms are GOA, MFO, MPA, GWO, HHO,

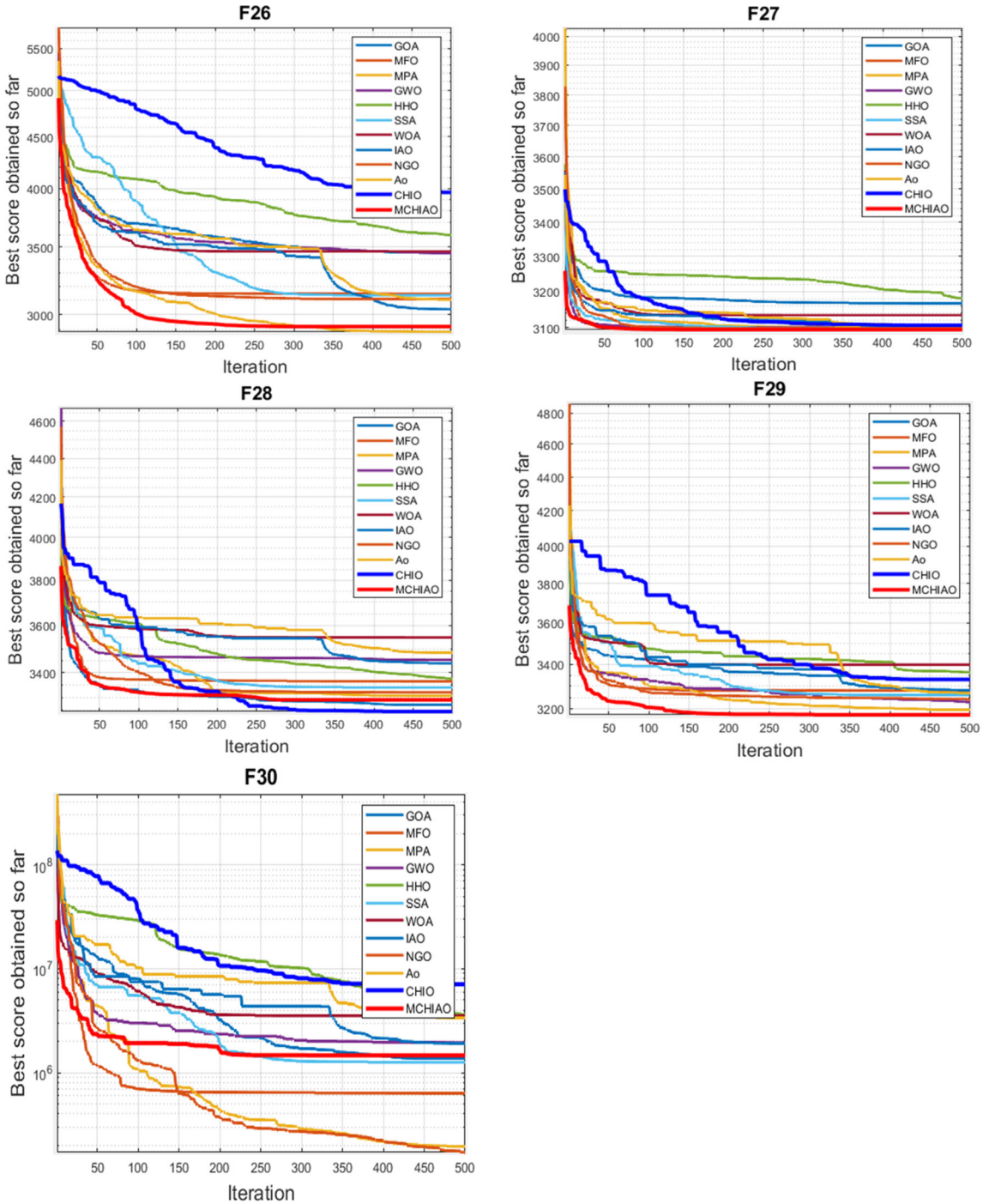


Fig. 8 continued

WOA, SSA, IAO, NOA, NGO, AO, and CHIO. All of the algorithms are executed with a maximum of 500 iterations and 51 runs. In Fig. 7 and Table 4, simulation results for all methods are presented. Table 4 presents the metrics calculated such as mean, standard deviation (Std), and Friedman average rank test for evaluating the performance of the various optimization techniques, with the best result of each measure in each function highlighted in bold.

These data show that the suggested algorithm MCHIAO performed well in both the multimodal functions from $F_8(x)$ to $F_{13}(x)$ and the unimodal functions from $F_1(x)$ to $F_7(x)$, demonstrating the first rank except for $F_6(x)$ and $F_{13}(x)$ where NGO and MPA ranked the first, respectively. However, MCHIAO showed the greatest rate and placed first in all the fixed-dimension multimodal functions from $F_{14}(x)$ to $F_{23}(x)$, except $F_{20}(x)$ where MPA ranked first.

Table 10 Review of CEC2019 benchmark function problems

No	Functions	$F_i^{\bullet} = F_i(x^*)$	Dim	Search range
1	Storn's Chebyshev Polynomial Fitting Problem	1	9	[−8192, 8192]
2	Inverse Hilbert Matrix Problem	1	16	[−16384, 16384]
3	Lennard–Jones Minimum Energy Cluster	1	18	[−4, 4]
4	Rastrigin's Function	1	10	[−100, 100]
5	Griewangk's Function	1	10	[−100, 100]
6	Weierstrass Function	1	10	[−100, 100]
7	Modified Schwefel's Function	1	10	[−100, 100]
8	Expanded Schaffer's F6 Function	1	10	[−100, 100]
9	Happy Cat Function	1	10	[−100, 100]
10	Ackley Function	1	10	[−100, 100]

Table 11 MCHIAO performance against CHIO and AO algorithms is conducted on 10 CEC-2019 functions

Function	Algorithm indices	AO [31]	CHIO [30]	MCHIAO
F1	Mean	69,363.0155	4,362,075,407	**37,453.7001**
	Std	15,915.6584	8,744,866,059	658.27772
	Rank	2	3	**1**
F2	Mean	17.3670308	17.3909001	**17.3428571**
	Std	0.00213039	0.01404747	0
	Rank	2	3	**1**
F3	Mean	12.7024252	12.7024172	**12.7024042**
	Std	2.1097E-05	3.3786E-06	0
	Rank	3	2	**1**
F4	Mean	1062.16958	2757.55276	**146.586412**
	Std	902.303042	1474.3629	102.829675
	Rank	2	3	**1**
F5	Mean	1.5350851	3.31228981	**1.2337303**
	Std	0.05752762	0.56147541	0.08531583
	Rank	2	3	**1**
F6	Mean	10.9565728	10.502067	**7.18112683**
	Std	0.67379091	1.18939858	2.18752976
	Rank	3	2	**1**
F7	Mean	453.191656	937.891665	**353.651994**
	Std	338.696778	137.105271	229.317033
	Rank	2	3	**1**
F8	Mean	5.49648782	6.78439506	**4.99222537**
	Std	0.74485931	0.20919096	0.70002441
	Rank	2	3	**1**
F9	Mean	5.23456936	422.275604	**2.91996449**
	Std	0.83734447	220.111026	0.31898007
	Rank	2	3	**1**
F10	Mean	20.4357804	20.4815469	**20.0697435**
	Std	0.07629745	0.0473569	0.15003633
	Rank	2	3	**1**
Percentage		2.2	2.8	1
Total Rank		2	3	**1**

With a total rank of one, the findings showed that the suggested algorithm MCHIAO performed the best on the Standard benchmark functions. In Fig. 7, the vertical axis shows the optimal solution to the function, while the horizontal axis shows the total number of iterations. It implies a comparison of the convergence curves of MCHIAO and

Table 12 MCHIAO performance against different algorithms on 10 CEC2019 benchmark function problems

Function	Algorithm Indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F1	Mean	8,460,886,587	2.6887E + 10	11,056,417.1	856,506,003	54,681.3046	1.5492E + 10	1.5357E + 10	69,363.0155	54,802.904	932,832.977	7,728,081.16	4,362,075,407	**37,453.7001**
	Std	8,815,647,152	2.992E + 10	14,625,619.7	2,008,268,752	6278.20288	1.5301E + 10	1.6708E + 10	15,915.6584	12,977.8569	495,449.73	15,088,062.5	8,744,866,059	658.27772
	Rank	10	13	7	8	2	12	11	4	3	5	6	9	**1**
F2	Mean	17.4814299	17.3428571	17.3428605	17.3441403	17.3588584	17.343345	17.3551804	17.3670308	17.3706946	19.8303867	17.3428571	17.3909001	**17.3428571**
	Std	0.18058401	0	4.7575E-07	0.00022961	0.0049526	5.9092E-05	0.00623899	0.00213039	0.0103316	0.01493987	2.2187E-14	0.01404747	0
	Rank	12	**1**	4	6	8	5	7	9	10	13	3	11	**1**
F3	Mean	12.7031275	12.7028597	12.7024042	12.7024043	12.7024154	12.7024042	12.7024045	12.7024252	12.7024083	12.7032846	12.7024042	12.7024172	**12.7024042**
	Std	0.00075179	0.00078895	7.8907E-12	1.3691E-07	1.0199E-05	1.2561E-15	3.4022E-07	2.1097E-05	3.3026E-07	0.00061775	8.8818E-15	3.3786E-06	0
	Rank	12	11	4	5	8	**1**	6	10	7	13	3	9	**1**
F4	Mean	46.8061406	173.858657	**25.0602983**	68.0281123	167.216592	52.7324321	275.497034	1062.16958	774.083798	5400.44382	112.065614	2757.55276	146.586412
	Std	5.16038878	190.4227	15.9668015	11.884078	52.7520855	45.3437919	76.0409004	902.303042	553.791442	482.289826	136.137921	1474.3629	102.829675
	Rank	2	8	**1**	4	7	3	9	11	10	13	5	12	6
F5	Mean	**1.16989334**	1.47260049	1.28407072	1.20909355	2.5775437	1.19271202	1.88366027	1.5350851	1.40811582	3.26961614	1.27165456	3.31228981	1.2337303
	Std	0.06879094	0.38952586	0.0983212	0.15713613	0.5709282	0.07781009	0.49203572	0.05752762	0.10494715	0.74627306	0.09605991	0.56147541	0.08531583
	Rank	**1**	8	6	3	11	2	10	9	**7**	12	5	13	4
F6	Mean	8.21988929	6.4013089	**4.31399061**	10.7241831	10.6219397	7.0084339	9.46692148	10.9565728	10.8182379	12.1381127	6.60844284	10.502067	7.18112683
	Std	2.38757717	0.56242646	1.42584448	0.44597331	0.27557772	1.60975399	2.1008186	0.67379091	0.73156578	0.78201321	1.22937832	1.18939858	2.18752976
	Rank	6	2	**1**	10	9	4	7	12	11	13	3	8	5
F7	Mean	635.404151	558.04556	186.533113	378.410831	369.171059	282.811513	607.553856	453.191656	351.871542	1236.46542	**130.133203**	937.891665	353.651994
	Std	156.259834	246.145298	101.753723	231.63836	150.618483	326.957726	284.70124	338.696778	120.049784	271.327544	163.869746	137.105271	229.317033
	Rank	11	9	2	7	6	3	10	8	4	13	**1**	12	5
F8	Mean	5.68883864	5.64890069	**4.81640382**	5.31492661	5.78376547	5.4289393	5.82896773	5.49648782	5.5888783	7.01203302	5.05130494	6.78439506	4.99222537
	Std	0.61291841	0.94412805	0.54174892	0.66054059	0.52681711	0.68927653	0.60329639	0.74485931	0.67496211	0.38712203	0.56229729	0.20919096	0.70002441
	Rank	9	8	**1**	4	10	5	11	6	7	13	3	12	2
F9	Mean	3.12312283	3.30256605	4.22237224	4.42537451	3.57550078	3.10972053	4.7340873	5.23456936	4.21679662	1070.34164	3.34993152	422.275604	**2.91996449**
	Std	0.55908185	0.4131749	0.9503355	0.41638477	0.54638664	0.47286742	0.78216302	0.83734447	0.75309536	316.316345	0.75556873	220.111026	0.31898007
	Rank	3	4	8	9	6	2	10	11	7	13	5	12	**1**
F10	Mean	20.1173897	20.1827417	20.0199781	20.5579015	20.2724701	20.0053555	20.2983862	20.4357804	**20.0053269**	20.6066986	20.0320001	20.4815469	20.0697435
	Std	0.16039596	0.12683334	0.03534671	0.06646019	0.20767643	0.00928035	0.1438089	0.07629745	0.83847721	0.26055134	0.01768933	0.0473569	0.15003633
	Rank	6	7	3	12	8	2	9	10	**1**	13	4	11	5
Percentage	7.1	7.1	3	5.2	7.5	3.9	8	9	6.7	9.5	3.5	10.9	3.1	
Total Rank	7	7	**1**	5	9	4	10	11	6	12	3	13	2	

Table 13 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for 10 CEC2019 benchmark function problems

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	7.3347E-151	7.1074E-153	2.35E-134	2.3817E-142	8.9457E-109	3.8118E-153	6.1495E-154	1.9978E-119	7.6743E-115	5.7631E-109	5.5923E-135	1.6852E-155
F2	7.4274E-146	1.1053E-15	6.3928E-75	4.2183E-100	1.8263E-109	1.0142E-132	2.6516E-97	1.4146E-114	2.8809E-113	3.40777E-19	1.5619E-20	1.402E-120
F3	2.3443E-156	7.727E-154	4.8638E-28	6.1775E-124	9.7585E-132	4.053E-56	9.29453E-86	3.3634E-158	9.9991E-156	1.7477E-127	4.69114E-05	4.5472E-135
F4	7.73582E-27	4.9769E-57	4.5839E-06	1.09159E-17	6.2181E-110	0.006118787	8.20221E-80	9.353E-123	7.5416E-118	1.36037E-23	0.00090834	4.7591E-138
F5	2.30432E-35	3.04328E-72	2.10536E-72	1.6178E-88	1.9721E-142	1.53023E-05	8.7582E-101	1.0372E-122	2.3025E-113	0.080494284	2.59658E-54	1.5217E-150
F6	1.32369E-87	1.29861E-83	6.9772E-40	1.4642E-119	5.2149E-111	1.91048E-28	5.22134E-13	5.332E-127	2.9908E-122	2.8693E-103	2.49659E-31	3.3568E-134
F7	2.5096E-111	7.78939E-40	0.733218588	5.83448E-59	1.26105E-28	0.378373097	1.41331E-58	1.38086E-93	7.74822E-68	4.87883E-10	2.59611E-96	1.6827E-134
F8	5.7935E-99	4.03734E-42	5.73795E-09	2.10425E-38	5.6543E-70	1.29514E-62	2.429E-53	5.31829E-95	4.68282E-88	2.8105E-162	2.48038E-13	2.7515E-130
F9	1.59196E-21	2.40001E-08	3.08583E-62	2.09179E-72	8.72644E-64	2.71419E-32	8.15513E-59	9.7172E-109	2.15561E-98	2.56298E-57	3.12449E-23	4.4015E-142
F10	1.87894E-82	0.036196173	1.75899E-06	2.40024E-92	1.17715E-07	1.04367E-08	0.000988155	5.20456E-41	0.013658896	1.47E-128	0.048498474	4.2613E-34

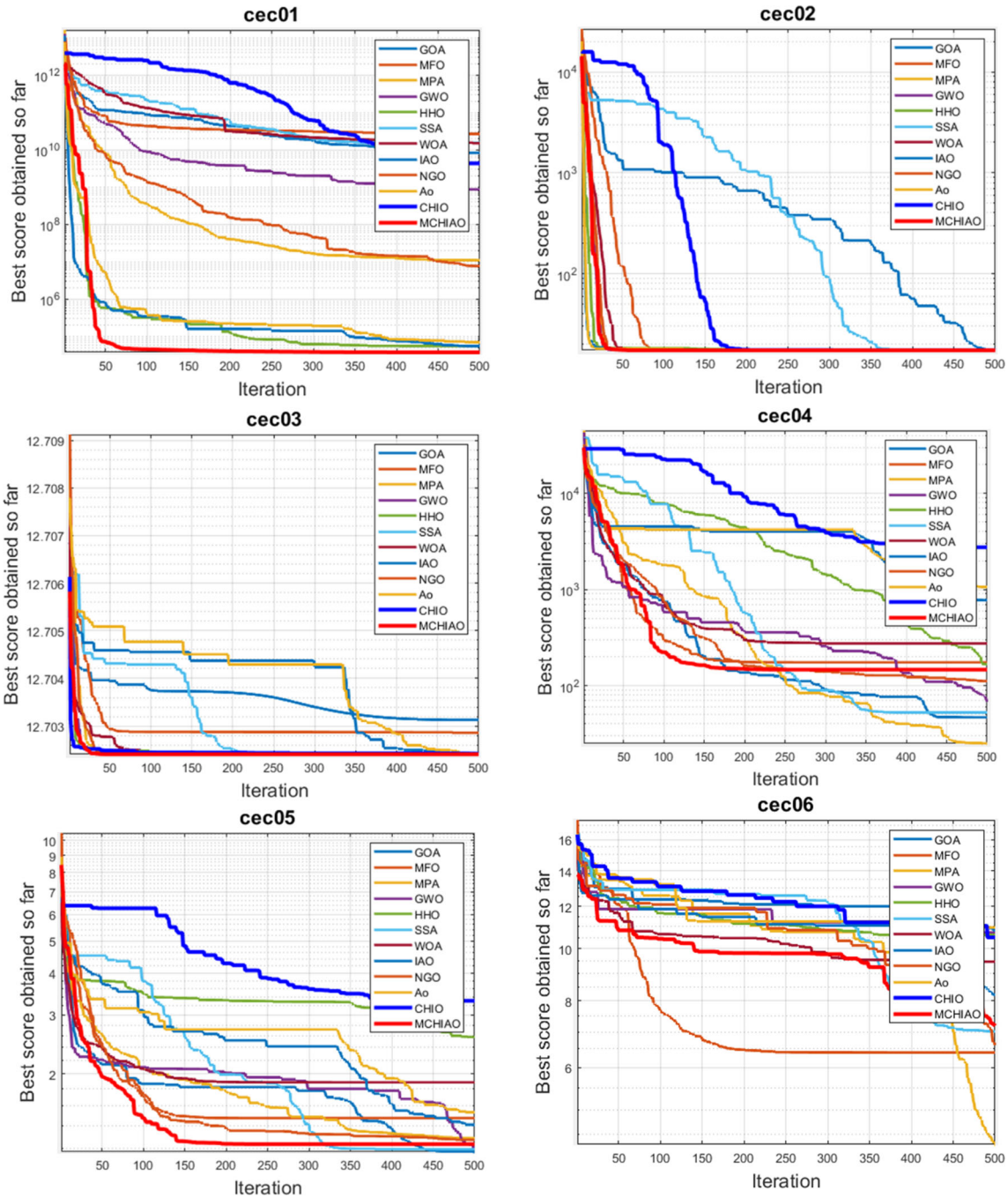


Fig. 9 Convergence curves of CEC2019 functions and convergence plots

other algorithms on common benchmark functions, demonstrating that MCHIAO has a favorable rate of convergence and the ability to find a more compact solution across all the benchmark functions, demonstrating one of its key strengths. In the majority of the twenty-three benchmark functions, it is seen that using the MCHIAO approach beats the other twelve evaluated algorithms. The Wilcoxon rank-sum test [32] is also applied in order to statically prove the efficiency of our proposed algorithm. Statistical study of the discrepancy results between two algorithms frequently makes use of the p-value of the Wilcoxon rank-sum test, which is useful in establishing if the two sets of data are significantly different. When p is less than 0.05, it indicates a substantial difference between the two methods in this test function, indicating the rejection of the null hypothesis. The comprehensive *p*-

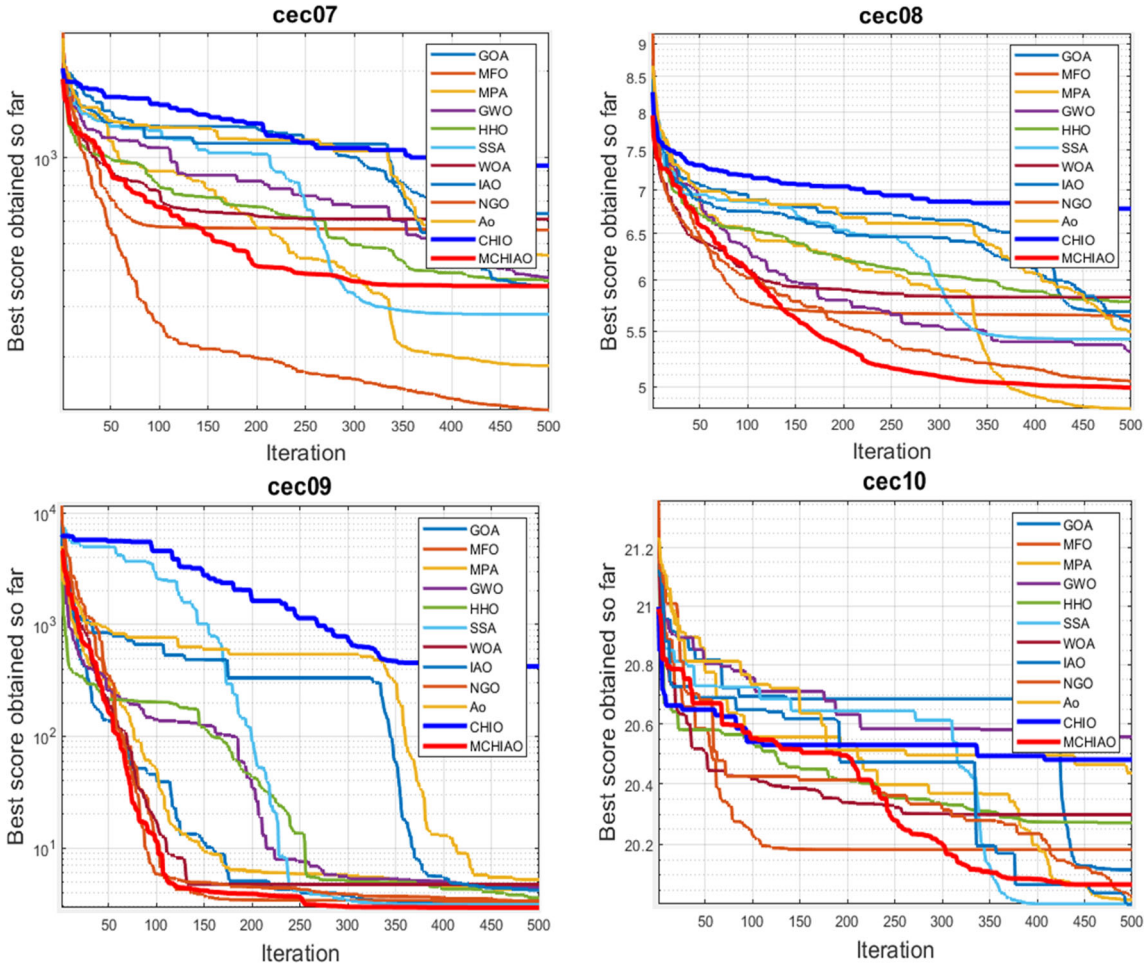


Fig. 9 continued

Table 14 Description of Unimodal fixed-dimension Problems

f.No.	Name	Vars	Range	f_{min}
F1	Beale	2	[−4.5, 4.5]	0
F2	Booth	2	[−10, 10]	0
F3	Brent	2	[−10, 10]	0
F4	Matyas	2	[−10, 10]	0
F5	Schaffer N. 4	2	[−100, 100]	0.292579
F6	Wayburn Seader 3	2	[−500, 500]	19.10588
F7	Leon	2	[−1.2, 1.2]	0
F8	Cube	2	[−10, 10]	0
F9	Zettl	2	[−5, 10]	−0.00379

values in Table 5 were acquired by applying the Wilcoxon test with each of the twelve algorithms to the MCHIAO solution findings, 51 times separately on the 23 benchmark functions test. This was necessary because the test necessitates two independent data sets. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Table 15 MCHIAO performance against CHIO and AO algorithms is conducted on 8 Unimodal fixed-dimension functions

Function	Algorithm indices	AO[31]	CHIO[30]	MCHIAO
F1	Mean	0.00041645	0.27580244	**0**
	Std	0.0004393	0.36657232	0
	Rank	2	3	**1**
F2	Mean	0.00040099	0.0001764	**0**
	Std	0.00029567	9.7635E-05	0
	Rank	3	2	**1**
F3	Mean	**1.3839E-87**	**1.3839E-87**	**1.3839E-87**
	Std	2.734E-103	2.734E-103	2.734E-103
	Rank	**1**	**1**	**1**
F4	Mean	3.737E-106	9.9749E-74	**0**
	Std	6.47E-106	9.4198E-74	0
	Rank	2	3	**1**
F5	Mean	0.2925824	0.29257936	**0.29257863**
	Std	3.6705E-06	5.5684E-07	5.5511E-17
	Rank	3	2	**1**
F6	Mean	19.1108644	19.1101276	**19.1058798**
	Std	0.0049773	0.00381332	4.3512E-15
	Rank	3	2	**1**
F7	Mean	0.00133935	0.00092498	**0**
	Std	0.00098295	0.0012904	0
	Rank	3	2	**1**
F9	Mean	−0.0037511	−0.0037912	**−0.0037912**
	Std	5.4492E-05	4.8259E-10	5.3115E-19
	Rank	3	2	**1**
Percentage		2.33333333	**2.22222222**	**1**
Total Rank		3	2	**1**

From the previous table, the proposed algorithm MCHIAO outperforms or produces similar solutions in all functions except for $F_6(x), F_{13}(x)$ and $F_{20}(x)$ where the best algorithms are NGO and MPA respectively.

The convergence curves for all thirteen algorithms on the 23 standard benchmark functions are shown in Fig. 7, where the vertical axis shows the optimal solution to the problem and the horizontal axis shows the total number of iterations. One of MCHIAO's assets is its remarkable convergence abilities, which are demonstrated by the algorithm's rate of convergence on CEC2005 benchmark functions and the ability to produce a more compact solution for all benchmark functions.

5.2 Case 2: MCHIAO performance on CEC2017 benchmark functions

The 29 CEC2017 benchmark functions are used to evaluate the proposed MCHIAO against all the same twelve state-of-the-art algorithms. The minimization problems in CEC2017 are divided into four categories: The first has three unimodal functions (F1–F3), the second has seven simple multimodal functions (F4–F10), the third has ten hybrid functions (F11–F20), and the fourth has ten composition functions (F21–F30). Table 6 shows how these functions operate. The objective function minimizes the difference between the optimal value of the i th function $f_i(x^*)$ and the best solution determined by the method $f_i(x)$; this function can be written as Eq. (4.1).

$$error = f_i(x) - f_i(x^*) \tag{4.1}$$

5.2.1 MCHIAO Vs AO and CHIO

Table 7 demonstrates the performance of the proposed MCHIAO against CHIO and AO as it ranked the first in all of the 29 functions except for $F_{28}(x)$, as it ranked second and CHIO ranked the first. MCHIAO scored a total rank 1.

5.2.2 MCHIAO vs other algorithms

Table 8 presents the experimental study findings of the suggested MCHIAO algorithm as well as the other comparator techniques in terms of mean results and standard derivation after 51 separate runs of each algorithm. The

Table 16 MCHIAO performance against twelve different algorithms is conducted on 8 Unimodal fixed-dimension functions

Function	Algorithm indices	GOA[9]	MFO[10]	MPA[11]	GWO[14]	HHO[15]	SSA[16]	WOA[17]	AO[31]	IAO[33]	NOA[34]	NGO[35]	CHIO[30]	MCHIAO
F1	Mean	0.18662965	1.059E-09	9.7185E-33	0.05862131	1.3955E-10	0.05862074	0.05862074	0.00041645	0.0001339	0.01704751	0.17586223	0.27580244	**0**
	Std	0.35517112	3.8154E-09	3.5041E-32	0.21135993	4.1648E-10	0.21136009	0.21136009	0.0004393	0.00013628	0.02111236	0.33418965	0.36657232	0
	Rank	12	4	2	10	3	8	9	6	5	7	11	13	**1**
F2	Mean	1.1682E-12	0	0	7.5984E-07	1.1651E-05	1.2767E-13	0.0044432	0.00040099	0.00027684	0.09653934	0	0.0001764	**0**
	Std	1.1255E-12	0	0	8.9867E-07	1.6328E-05	1.7118E-13	0.00373405	0.00029567	0.00010711	0.0160033	0	9.7635E-05	0
	Rank	6	**1**	**1**	7	8	5	12	11	10	13	**1**	9	**1**
F3	Mean	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	1.3839E-87	**1.3839E-87**
	Std	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103	2.734E-103
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1	**1**
F4	Mean	1.6757E-13	6.5449E-15	7.0154E-52	4.6865E-61	1.325E-145	2.8596E-15	1.988E-193	3.737E-106	5.481E-152	3.548E-20	1.516E-91	9.9749E-74	**0**
	Std	1.8321E-13	1.1336E-14	1.2151E-51	8.1172E-61	2.294E-145	3.9141E-15	0	6.47E-106	8.196E-152	6.1454E-20	2.6258E-91	9.4198E-74	0
	Rank	13	12	9	8	4	11	2	5	3	10	6	7	**1**
F5	Mean	0.29258386	0.29306237	0.29257863	0.29258016	0.29257914	0.29257864	0.29258457	0.2925824	0.29258751	0.29462715	0.29257863	0.29257936	**0.29257863**
	Std	6.7374E-06	0.0007841	0	1.1364E-06	8.6504E-07	8.3453E-09	7.5937E-06	3.6705E-06	7.4711E-06	0.00288536	2.5739E-16	5.5684E-07	5.5511E-17
	Rank	9	12	2	7	5	4	10	8	11	13	3	6	**1**
F6	Mean	19.1058798	**19.1058798**	19.1058798	19.1058983	19.106	19.1058798	19.1199195	19.1108644	19.1068955	25.3748639	19.1058798	19.1101276	19.1058798
	Std	1.6701E-09	0	2.5121E-15	9.075E-06	9.5552E-05	1.4815E-10	0.00976164	0.0049773	0.00047665	5.32750195	0	0.00381332	4.3512E-15
	Rank	6	**1**	2	7	8	5	12	11	9	13	4	10	2
F7	Mean	0.00230038	0.04875509	7.9724E-29	6.7534E-06	1.0504E-05	9.6968E-08	1.6747E-05	0.00133935	0.00068835	0.01993788	1.0559E-09	0.00092498	**0**
	Std	0.00373258	0.01859232	1.38E-28	5.5661E-06	7.2139E-06	1.6795E-07	1.1888E-05	0.00098295	0.00051262	0.01743412	8.6429E-10	0.0012904	0
	Rank	11	13	2	5	6	4	7	10	8	12	3	9	**1**
F9	Mean	−0.0037912	**-0.0037912**	**-0.0037912**	−0.0037912	−0.0037912	−0.0037912	−0.0037912	−0.0037511	−0.003787	−0.0025385	**−0.0037912**	−0.0037912	**−0.0037912**
	Std	1.6847E-13	5.3115E-19	5.3115E-19	1.8412E-09	1.9235E-10	2.0069E-14	1.0982E-09	5.4492E-05	1.1935E-06	0.00183017	4.3368E-19	4.8259E-10	5.3115E-19
	Rank	6	**1**	**1**	10	7	5	9	12	11	13	**1**	8	**1**
Percentage	8.55555556	**6.33333333**	**3.11111111**	**7.11111111**	**4.77777778**	**6**	**7.55555556**	**7.22222222**	**6.55555556**	**9.88888889**	**3.44444444**	**8.11111111**	**1.11111111**	
Total Rank	12	6	2	8	4	5	10	9	7	13	3	11	**1**	

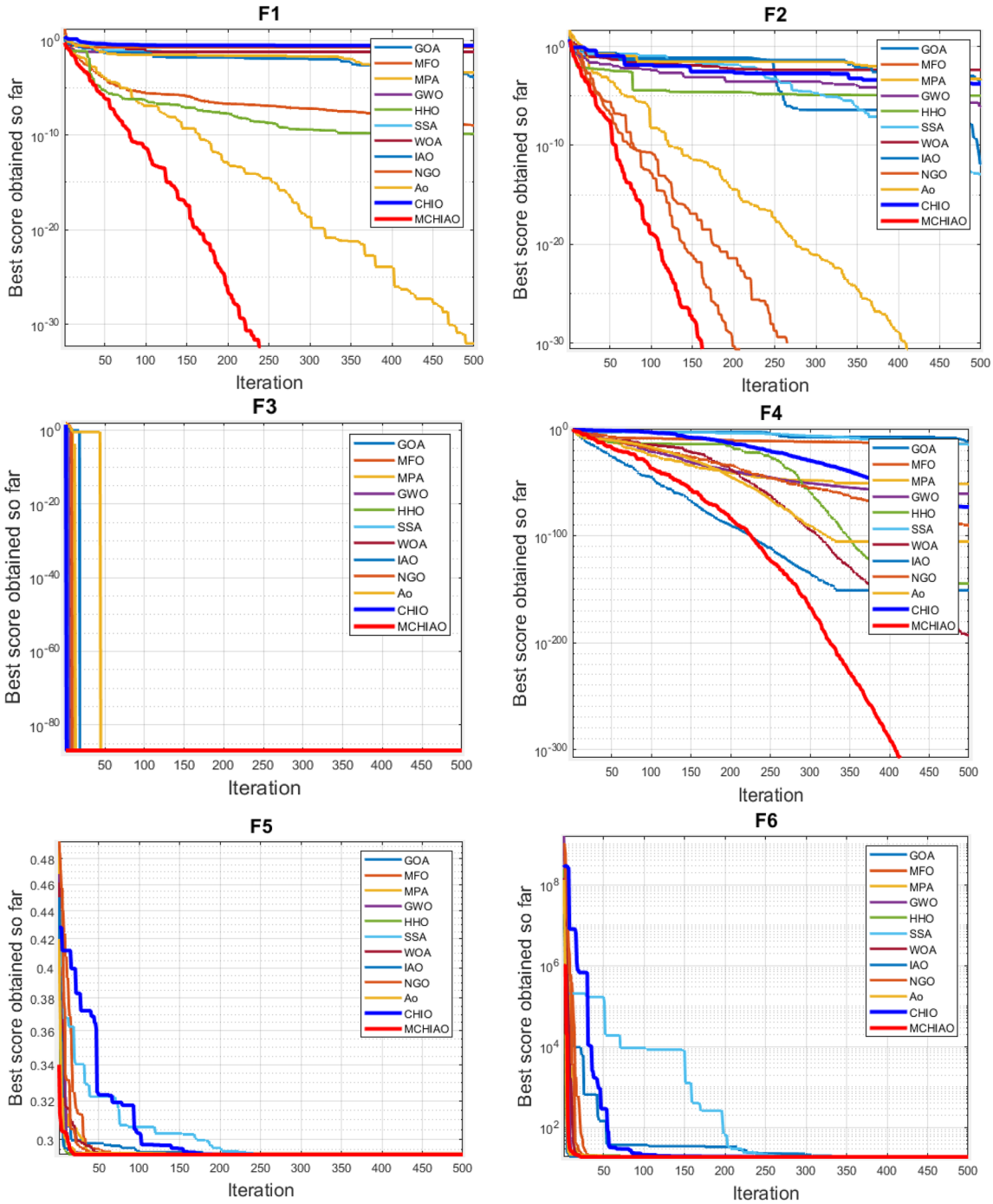


Fig. 10 The 8 Unimodal fixed-dimension and convergence plots

best outcomes are displayed in bold font in Tables 7 and 8. Optimization results presented in Table 8 indicate that the proposed MCHIAO algorithm outperforms the other twelve algorithms in 17 out of the 29 CEC2017 benchmark functions and scored a total first rank against other optimization algorithms. Our proposed algorithm MCHIAO ranked first in the unimodal functions $F_1(x)$ and $F_3(x)$. For the multimodal functions $F_4(x)$ to $F_{10}(x)$, MCHIAO ranked first in $F_4(x), F_5(x)$, *and* $F_8(x)$. MPA ranked first in $F_6(x), F_9(x)$, *and* $F_{10}(x)$, and GOA ranked first in $F_7(x)$. In the hybrid functions $F_{11}(x)$ to $F_{20}(x)$, MCHIAO ranked first in all functions except $F_{11}(x), F_{12}(x), F_{14}(x)$, *and* $F_{17}(x)$, where it ranked the second and MPA ranked first in $F_{11}(x), F_{14}(x)$, *and* $F_{17}(x)$,

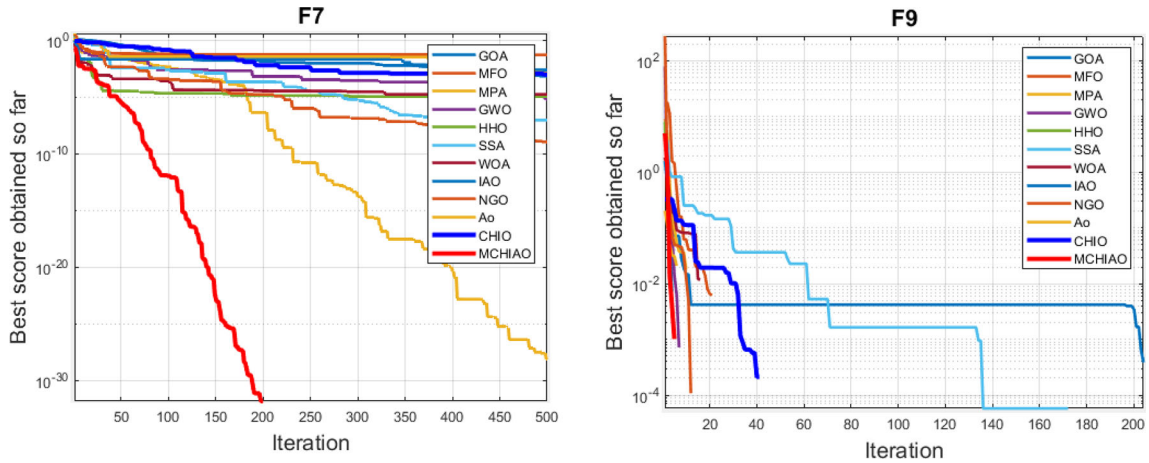


Fig. 10 continued

and NGO ranked first in $F_{12}(x)$. However, in the composition functions $F_{21}(x)$ to $F_{30}(x)$, MCHIAO ranked first in all functions except $F_{25}(x), F_{26}(x), F_{28}(x)$, *and* $F_{30}(x)$ where GOA, MPA, CHIO, and NGO ranked first, respectively. Therefore, with a total rank of 1, our suggested algorithm MCHIAO showed the greatest performance on the CEC2017 benchmark functions. Table 9 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for 29 CEC2017 benchmark functions. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 8 presents the convergence plots of the proposed MCHIAO algorithm against the same twelve algorithms for the 29 CEC2017 benchmark functions. The MCHIAO algorithm outperforms the other twelve evaluated algorithms in the majority of the 29 benchmark functions.

5.3 Case 3: MCHIAO performance on CEC2019 benchmark functions

We chose one of the most relevant and challenging test suites from the numerical optimization competitions, CEC-2019 (which has ten functions) to further demonstrate the effectiveness of the proposed technique. Table 10 illustrates the ten CEC2019 benchmark functions.

5.3.1 MCHIAO Vs AO and CHIO

Table 11 shows that MCHIAO is ranked first against CHIO and AO in all the 10 CEC-2019 benchmark functions. For the total rank, MCHIAO scored 1.

5.3.2 MCHIAO vs other algorithms

Table 12 shows the performance of the proposed MCHIAO against the same twelve algorithms conducted on the ten CEC2019 functions after 500 iterations and 51 runs of each algorithm. The data analyzed for the mean value, standard deviation, and rank produced by the twelve unique state-of-the-art optimization methods are shown in Table 12. With a high rate, our suggested algorithm MCHIAO placed first in CEC01, CEC02, CEC03, and CEC09, second in CEC08, fourth in CEC05, and fifth in CEC06, CEC07, and CEC10, and sixth in CEC04. With a total rank of 2, MCHIAO showed a promising overall performance on the CEC2019 benchmark functions. Table 13 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for ten CEC2019 benchmark functions. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 9 shows the convergence behavior of the proposed MCHIAO on CEC2019 functions.

5.4 Case 4: performance of MCHIAO on Unimodal fixed-dimension problems

Table 14 represents the details of the unimodal fixed-dimension problems. Range specifies the lower and upper boundaries of the design variables, whereas f_{min} signifies the overall minimum of the functions. Vars identifies the number of dimensions (design variables) of the functions.

Table 17 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for 8 Unimodal fixed-dimension functions

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	9.2661E-164	1.1321E-105	1.35718E-43	1.2136E-157	2.19342E-93	1.5879E-157	1.2883E-158	2.2031E-151	3.007E-149	1.3489E-156	3.7863E-165	1.6717E-165
F2	4.0042E-143	0.002538709	3.99207E-47	9.8217E-146	5.8245E-139	1.3755E-137	1.7159E-155	3.8528E-159	2.3042E-162	3.23078E-39	2.82236E-10	1.2463E-152
F3	0.033332146	0.045284328	0.000496442	0.045284382	0.318279359	1	0.318279359	1.20723E-11	1.88101E-05	3.86236E-97	0.004544709	0.318279359
F4	2.8659E-147	2.5561E-128	1.31245E-55	1.08692E-53	1.17698E-38	2.0896E-141	1.30909E-20	8.17202E-29	1.20303E-05	7.54067E-07	5.20094E-47	1.93887E-83
F5	8.8985E-145	2.0961E-150	2.1305E-78	1.8885E-116	4.6657E-124	3.4128E-129	2.6696E-118	1.1262E-144	1.2263E-140	6.96633E-94	8.22008E-95	6.0039E-133
F6	7.1708E-152	9.36832E-14	6.96201E-54	1.0711E-139	4.8607E-138	9.899E-145	1.0886E-141	3.4788E-146	2.8983E-142	0.001533511	5.62199E-55	2.1632E-146
F7	6.7122E-160	1.3321E-165	1.18145E-76	7.9194E-141	1.7603E-121	6.7003E-122	1.2704E-127	2.0221E-167	6.9669E-168	2.72648E-81	1.879E-112	3.4031E-153
F9	1.1133E-146	0.013803742	6.71229E-14	2.4132E-123	2.3232E-126	9.3458E-146	3.375E-123	9.845E-172	1.8142E-161	4.6871E-173	0.109148776	1.5378E-134

Table 18 Description of Unimodal variable-dimension problems

f.No.	Name	Vars	Range	f_{min}
F1	Sphere	30	[−100, 100]	0
F2	Powell Sum	30	[−1, 1]	0
F3	Schwefel's 2.20	30	[−100, 100]	0
F4	Schwefel's 2.21	30	[−100, 100]	0
F5	Step	30	[−100, 100]	0
F6	Stepint	30	[−5.12, 5.12]	−155
F7	Schwefel's 2.22	30	[−100, 100]	0
F8	Schwefel's 2.23	30	[−10, 10]	0
F9	Rosenbrock	30	[−30, 30]	0
F10	Brown	30	[−1, 4]	0
F11	Dixon and Price	30	[−10, 10]	0
F12	Powell Singular	30	[−4, 5]	0
F13	Xin−She Yang	30	[−20, 20]	0
F14	Perm 0,D,Beta	5	[−Var, Var]	0
F15	Sum Squares	30	[−10, 10]	0

5.4.1 MCHIAO Vs AO and CHIO

In Table 15, MCHIAO is ranked the first in all 8 functions with a total rank 1, while CHIO ranked second and AO ranked third.

5.4.2 MCHIAO vs other algorithms

Tables 15 and 16 show the mean value, standard deviation, and rank attained by the proposed MCHIAO algorithm against CHIO and AO, and the other twelve unique state-of-the-art optimization algorithms, respectively. Table 16 shows that MCHIAO demonstrated the first rank in all unimodal fixed-dimension problems except for F6 in which it ranked second. With a total rank of 1, MCHIAO had the greatest overall performance on the unimodal fixed-dimension functions. The convergence curves of the various methods are shown in Fig. 10, where the vertical axis shows the optimal solution to the function and the horizontal axis shows the total number of iterations. Table 17 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the 8 Unimodal fixed-dimension functions. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

5.5 Case 5: performance of MCHIAO on Unimodal variable-dimension problems

Table 18 represents the details of the unimodal variable-dimension functions.

5.5.1 MCHIAO Vs AO and CHIO

Table 19 demonstrates that MCHIAO obtained the first rank in all unimodal variable-dimension problems. For the total rank, MCHIAO ranked first, AO ranked second, and CHIO ranked third.

5.5.2 MCHIAO vs other algorithms

In comparison to the 12 existing meta-heuristics algorithms, Table 20 demonstrates that the novel hybrid MCHIAO algorithm offers the best attainable optimum values of the functions in terms of mean and standard values of the functions. Figure 11 addresses the convergence performance of the twelve competitive algorithms and MCHIAO variations in solving unimodal benchmark functions; the convergence solutions encountered show that the MCHIAO is better capable of finding the most optimum solution in the fewest iterations. Based on the results of variable and fixed dimension unimodal problems, we can say that MCHIAO has excellent exploitation capabilities and can produce superior results in difficult situations. Table 21 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the 15 Unimodal variable-dimension functions. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

5.6 Case 6: performance of MCHIAO on Multimodal fixed-dimension problems

The details of the multimodal fixed-dimension functions are presented in Table 22.

5.6.1 MCHIAO Vs AO and CHIO

Table 23 presents the MCHIAO against CHIO and AO where it ranked first in all functions except $F_{14}(x)$ where AO ranked first. For the total rank, MCHIAO scored rank 1, AO rank 2, and CHIO rank 3.

5.6.2 MCHIAO vs other algorithms

Table 24 illustrates the outcomes of MCHIAO and the other algorithms for the fixed-dimension multimodal benchmarks. The results show that MCHIAO generally reaches the globally optimal values. If the results are compared, MCHIAO possesses first place for all functions other than f9, f14, and f15. An important feature that should be explored is the convergence of the MCHAIO for the optimal solutions throughout the iteration numbers, hence Fig. 12 demonstrates the best solutions so far across

Table 19 MCHIAO performance against CHIO and AO algorithms is conducted on unimodal variable-dimension functions

Function	Algorithm indices	AO [31]	CHIO [30]	MCHIAO
F1	Mean	3.533E-99	9.2172E-08	**0**
	Std	7.9E-99	1.3477E-07	0
	Rank	2	3	**1**
F2	Mean	2.2334E-60	3.0776E-26	**0**
	Std	4.9941E-60	4.7155E-26	0
	Rank	2	3	**1**
F3	Mean	2.6229E-59	0.00038404	**3.414E-196**
	Std	4.5431E-59	0.00056483	0
	Rank	2	3	**1**
F4	Mean	1.6964E-60	0.0946589	**2.281E-198**
	Std	2.9382E-60	0.09033654	0
	Rank	2	3	**1**
F5	Mean	0.00020324	3.21719824	**1.4618E-07**
	Std	0.00024445	0.1541856	2.2558E-07
	Rank	2	3	**1**
F6	Mean	**−155**	**−155**	**−155**
	Std	0	0	0
	Rank	**1**	**1**	**1**
F7	Mean	1.9266E-47	1.9348E-05	**1.095E-195**
	Std	3.3368E-47	6.2951E-06	0
	Rank	2	3	**1**
F8	Mean	**0**	2.9402E-23	**0**
	Std	0	5.0925E-23	0
	Rank	**1**	2	**1**
F9	Mean	0.00380226	28.9457917	**3.1522E-05**
	Std	0.00389016	0.01378845	2.7376E-05
	Rank	2	3	**1**
F10	Mean	1.416E-95	1.0104E-09	**0**
	Std	2.4439E-95	1.1945E-09	0
	Rank	2	3	**1**
F11	Mean	0.25090981	0.77843789	**0.00106436**
	Std	0.00158201	0.19187879	0.00063133
	Rank	2	3	**1**
F12	Mean	3.5284E-97	2.8293E-05	**0**
	Std	6.1114E-97	4.8879E-05	0
	Rank	2	3	**1**
F13	Mean	−0.448377	4.34E-232	**−1**
	Std	0.11982465	0	0
	Rank	2	3	**1**
F14	Mean	229.027227	4547.1164	**0.11071688**
	Std	206.245808	112.142546	0.11643342
	Rank	2	3	**1**
F15	Mean	2.179E-122	1.6725E-08	0
	Std	3.774E-122	2.7561E-08	0
	Rank	2	3	**1**
Percentage		1.86666667	**2.86666667**	**1**
Total Rank		2	3	**1**

Table 20 MCHIAO performance against twelve different algorithms is conducted on unimodal variable-dimension functions

Function	Algorithm indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F1	Mean	403.632433	4094.56326	1.873E-21	2.1278E-19	4.4834E-98	0.00919764	2.4298E-77	3.533E-99	1.224E-132	3.5102E-08	1.2494E-82	9.2172E-08	**0**
	Std	122.707804	5588.7669	1.8678E-21	1.4081E-19	9.9275E-98	0.00788583	5.4268E-77	7.9E-99	2.736E-132	7.8491E-08	1.8211E-82	1.3477E-07	0
	Rank	12	13	7	8	4	11	6	3	2	9	5	10	**1**
F2	Mean	0.00798542	6.4942E-08	5.6365E-65	1.1175E-71	2.879E-118	1.2453E-05	3.224E-106	2.2334E-60	4.9056E-65	9.6777E-14	8.622E-166	3.0776E-26	**0**
	Std	0.0135052	1.2124E-07	1.1653E-64	2.1829E-71	6.439E-118	4.6236E-06	6.785E-106	4.9941E-60	1.0969E-64	2.164E-13	0	4.7155E-26	0
	Rank	13	11	7	5	3	12	4	8	6	10	2	9	**1**
F3	Mean	151.836984	74.6844224	1.5934E-12	4.9585E-11	1.5686E-48	43.2714219	1.7251E-49	2.6229E-59	1.4376E-72	0.00029361	2.5959E-43	0.00038404	**3.414E-196**
	Std	31.3223403	114.287696	1.4904E-12	1.2026E-11	2.6645E-48	40.202657	2.9059E-49	4.5431E-59	2.4844E-72	0.0005085	2.2791E-43	0.00056483	0
	Rank	13	12	7	8	5	11	4	3	2	9	6	10	**1**
F4	Mean	21.39239	70.2646516	7.0544E-08	3.6176E-05	4.0761E-51	17.7005538	66.0784033	1.6964E-60	2.8746E-76	4.7491E-08	8.4263E-37	0.0946589	**2.281E-198**
	Std	4.97643463	10.0470293	1.1747E-08	1.303E-05	5.0289E-51	4.46642789	22.2055949	2.9382E-60	4.4179E-76	8.2257E-08	2.8997E-37	0.09033654	0
	Rank	11	13	7	8	4	10	12	3	2	6	5	9	**1**
F5	Mean	414.772476	11.5273556	0.76345696	1.67407494	7.238E-05	0.02699684	0.23024995	0.00020324	1.8402E-05	7.47214818	0.87272358	3.21719824	**1.4618E-07**
	Std	192.92423	13.2707233	0.29401041	0.80986728	0.00011399	0.03345366	0.1344635	0.00024445	1.5698E-05	1.7264751	0.36744175	0.1541856	2.2558E-07
	Rank	13	12	7	9	3	5	6	4	2	11	8	10	**1**
F6	Mean	−59.333333	**−155**	−153	−119	**−155**	−120	**−155**	**−155**	**−155**	−110.33333	−142.33333	**−155**	**−155**
	Std	38.214308	0	3.46410162	7.54983444	0	1	0	0	0	21.0792157	9.07377173	0	0
	Rank	13	**1**	8	11	**1**	10	**1**	**1**	**1**	12	9	**1**	**1**
F7	Mean	3.5093E + 33	340.299419	8.0352E-13	4.0559E-11	9.7119E-51	9.4964E + 21	6.7112E-51	1.9266E-47	9.2408E-53	9.5236E-14	3.0698E-42	1.9348E-05	**1.095E-195**
	Std	6.0741E + 33	254.674929	1.0728E-12	1.09E-11	1.6025E-50	1.6448E + 22	5.8796E-51	3.3368E-47	1.6006E-52	1.6495E-13	2.741E-42	6.2951E-06	0
	Rank	13	11	8	9	4	12	3	5	2	7	6	10	**1**
F8	Mean	14,075.381	4660.71409	1.3098E-96	1.9379E-68	**0**	0.44717444	3.727E-192	**0**	**0**	2.033E-178	**0**	2.9402E-23	**0**
	Std	15,367.9113	7941.16881	1.9822E-96	2.1357E-68	0	0.20859832	0	0	0	0	0	5.0925E-23	0
	Rank	13	12	8	9	**1**	11	6	**1**	**1**	7	**1**	10	**1**
F9	Mean	359,208.093	43,246.3404	26.7112144	27.8834672	0.04670176	450.237218	27.9354547	0.00380226	0.01827392	28.9496898	27.7036688	28.9457917	**3.1522E-05**
	Std	261,561.065	43,991.763	0.43823296	0.79193476	0.04740405	190.326957	0.52653991	0.00389016	0.03094862	0.05986113	1.04487839	0.01378845	2.7376E-05
	Rank	13	12	5	7	4	11	8	2	3	10	6	9	**1**
F10	Mean	34.1150215	8.53686872	6.5324E-25	1.1758E-22	3.948E-100	0.00022514	6.3106E-83	1.416E-95	6.0231E-97	3.079E-18	5.142E-88	1.0104E-09	**0**
	Std	15.2100607	3.11911162	9.5258E-25	9.2619E-23	6.838E-100	0.00034067	1.0929E-82	2.4439E-95	1.0082E-96	5.2736E-18	7.1304E-88	1.1945E-09	0
	Rank	13	12	7	8	2	11	6	4	3	9	5	10	**1**
F11	Mean	2745.45931	184.689284	0.66666848	0.66669422	0.249545	59.7758329	0.66704116	0.25090981	0.25051463	0.99473482	0.66666689	0.77843789	**0.00106436**
	Std	1324.3651	129.495528	1.0475E-06	3.8516E-06	0.00065867	43.0566486	0.0002165	0.00158201	0.00107502	0.00904777	1.0355E-07	0.19187879	0.00063133
	Rank	13	12	6	7	2	11	8	4	3	10	5	9	**1**
F12	Mean	14.441236	1063.12249	6.7251E-12	9.458E-06	2.475E-103	41.1635985	5.5121E-06	3.5284E-97	1.467E-105	1.0635E-06	1.4908E-33	2.8293E-05	**0**
	Std	8.8552328	1699.43143	1.1617E-11	8.0247E-06	4.281E-103	26.734724	9.4301E-06	6.1114E-97	1.74E-105	1.842E-06	2.5821E-33	4.8879E-05	0
	Rank	11	13	6	9	3	12	8	4	2	7	5	10	**1**
F13	Mean	4.564E-186	4.34E-232	4.34E-232	3.943E-117	**−1**	2.192E-179	−0.3333333	−0.448377	−0.9967763	−0.3333333	4.34E-232	4.34E-232	**−1**
	Std	0	0	0	6.829E-117	0	0	0.57735027	0.11982465	0.00558356	0.57735027	0	0	0
	Rank	11	7	7	13	**1**	12	5	4	3	5	7	7	**1**
F14	Mean	492.525851	0.93709765	1.11904496	29.941216	55.2486101	66.2822497	628.790271	229.027227	25.434265	2734.33986	27.2083417	4547.1164	**0.11071688**
	Std	679.933928	0.82163313	0.86792398	25.3910671	76.5462353	32.6655098	634.912141	206.245808	31.3161536	840.394404	23.400685	112.142546	0.11643342
	Rank	10	2	3	6	7	8	11	9	4	12	5	13	**1**

Table 20 (continued)

Function	Algorithm indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F15	Mean	171.618778	768.702392	1.0525E-23	2.638 1E-20	1.0363E-98	2.62796247	5.4596E-79	2.179E-122	5.55E-147	6.8132E-21	1.0061E-83	1.6725E-08	**0**
	Std	66.051323	709.550807	1.4109E-23	3.4543E-20	1.7944E-98	2.28874992	9.4315E-79	3.774E-122	9.613E-147	1.1801E-20	1.4864E-83	2.7561E-08	0
	Rank	12	13	7	9	4	11	6	3	2	8	5	10	**1**
Percentage	12.2666667	**10.4**	**6.66666667**	**8.4**	**3.2**	**10.5333333**	**6.26666667**	**3.86666667**	**2.53333333**	**8.8**	**5.33333333**	**9.13333333**	**1**	
Total Rank	13	11	7	8	3	12	6	4	2	9	5	10	**1**	

the iteration numbers. The acceleration convergence curves of the suggested MCHIAO have a discernible declining rate for the multimodal fixed-dimension functions. Table 25 presents the p-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the 27 multimodal fixed-dimension functions. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

5.7 Case 7: performance of MCHIAO on Multimodal variable-dimension problems

The dimensionality of the variable-dimension multimodal benchmark functions, in addition to the local optima, increases the complexity and difficulty of optimization.

Table 26 presents the details of the multimodal variable-dimension problems. Vars specifies the number of dimensions (design variables) of the functions. Range identifies the lower and upper boundaries of the design variables, whereas f_{min} signifies the overall minimum of the functions.

5.7.1 MCHIAO Vs AO and CHIO

MCHIAO is ranked first in all functions against CHIO and AO except for $F_8(x)$ and $F_{14}(x)$ where AO ranked the first, as shown in Table 27. MCHIAO scored 1 for the total rank, while AO scored 2 and CHIO ranked rank 3.

5.7.2 MCHIAO vs other algorithms

In Table 28, MCHIAO's ability to solve multimodal functions with variable dimensions is compared with the performance of other algorithms.

The results show that MCHAIO can fully explore the search space. Except for $F_8(x)$, $F_{14}(x)$, and $F_{17}(x)$, MCHAIO consistently exceeds competitors; nonetheless, the results for these three functions are equally competitive. Table 29 presents the p-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the 17 multimodal variable-dimension functions. These results prove the significance of the proposed algorithm for all functions because the p-values are less than 0.05.

In Fig. 13, MCHIAO's convergence curves for 17 multimodal variable-dimension functions are presented and compared with the twelve algorithms to demonstrate the convergence capabilities of MCHIAO.

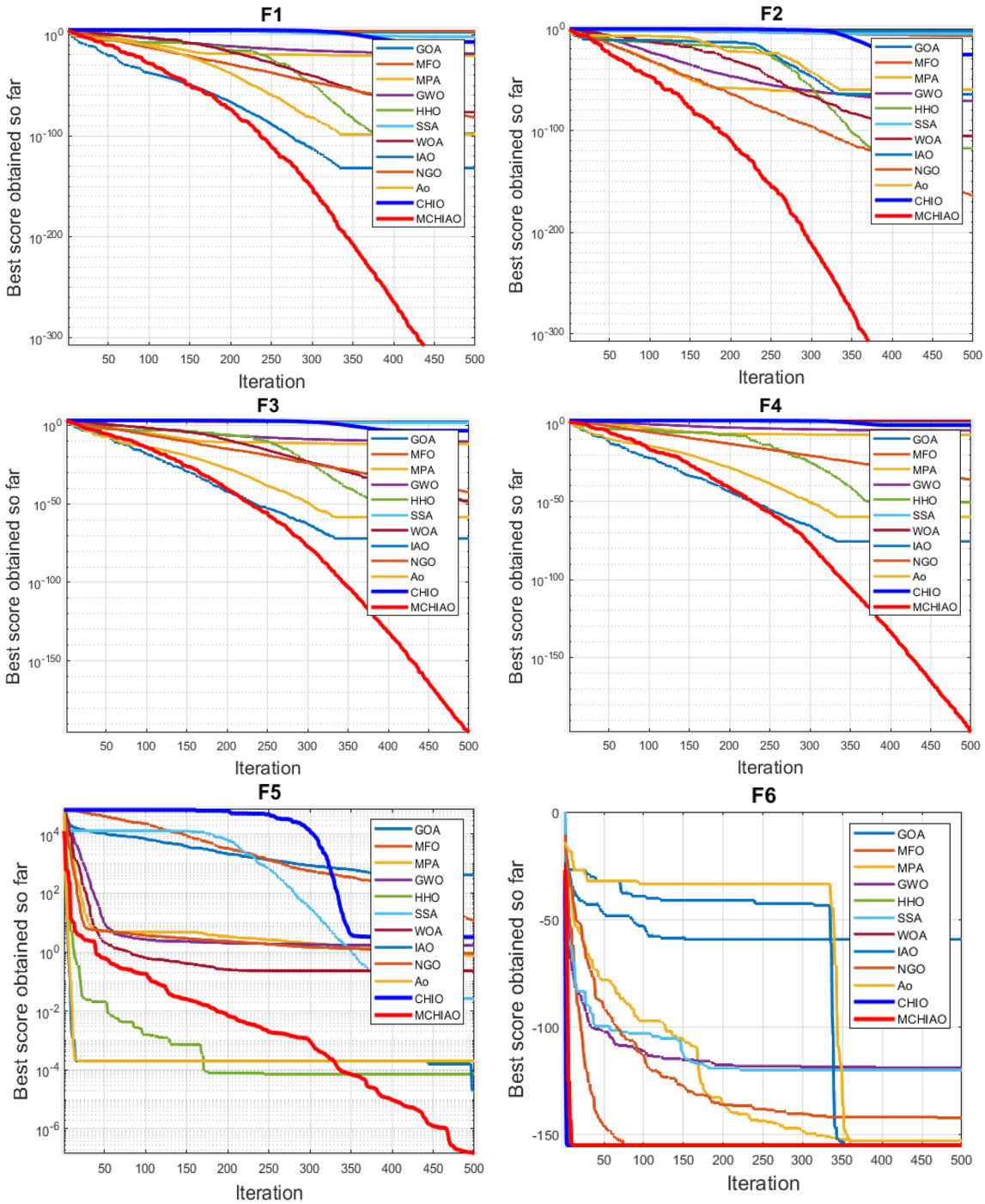


Fig. 11 The Unimodal variable-dimension functions and convergence plots

5.8 Case 8: performance of MCHIAO on real-world problems

The majority of the search spaces for real-world issues are unknown and provide several challenges. Such challenges drastically reduce the performance of optimization algorithms in the sector, which previously excelled at benchmark functions or simple case studies [36].

Scientists and practitioners must recognize the obstacles and include the appropriate adjustments, alterations, and enhancements in the algorithms to address them in order to

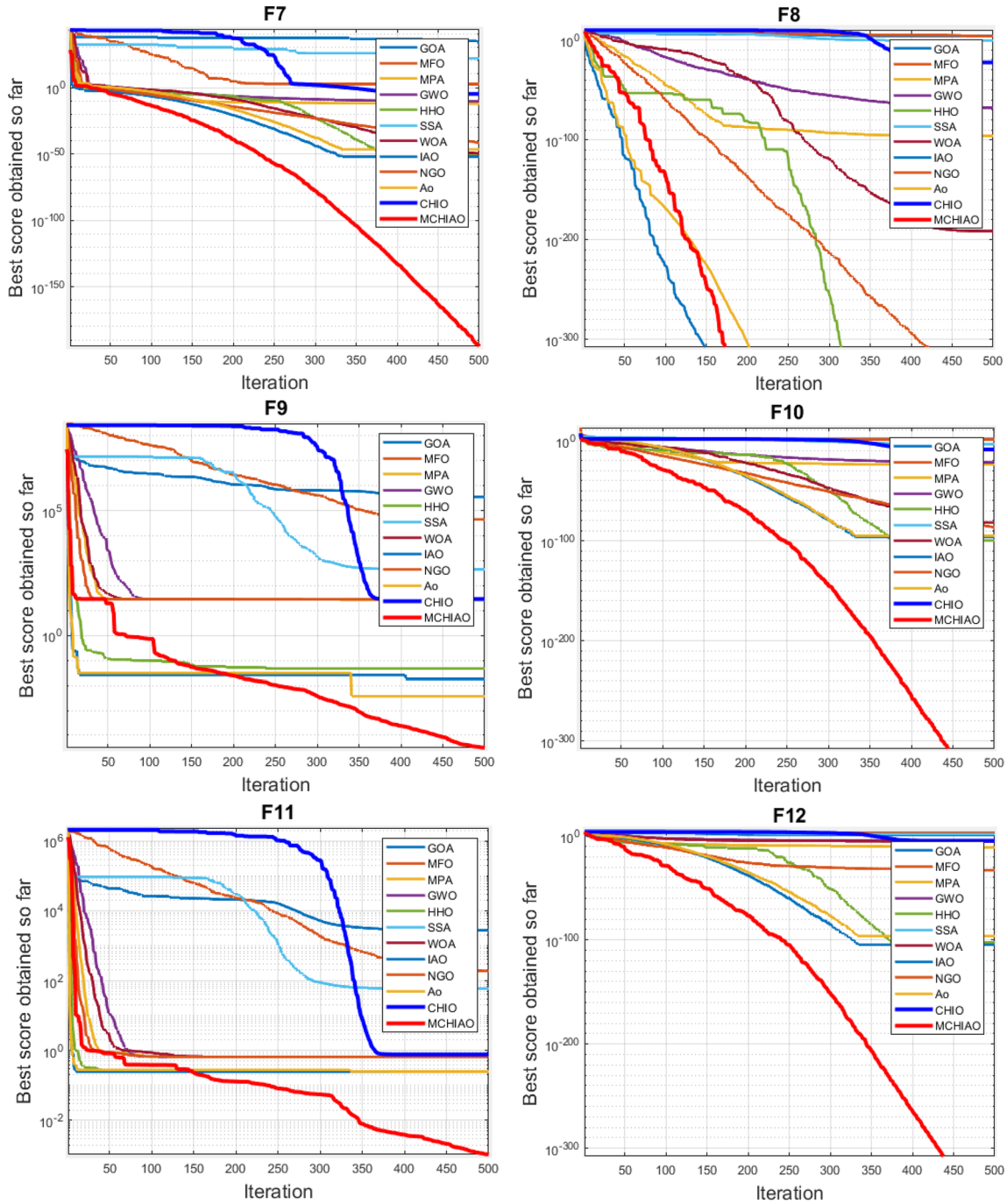


Fig. 11 continued

guarantee the quality of solutions to optimisation problems [37].

This section evaluates the performance of MCHIAO in real-world problems, this optimizer has been used to solve six engineering problems: welded beam design, pressure vessel design, tension/compression spring design, weight minimization of a speed reducer, three-bar truss design problem, and gear train design problem.

5.8.1 Tension/compression spring design

The goal is to obtain the least fabrication cost for spring. It is a weight minimization problem of a spring with three

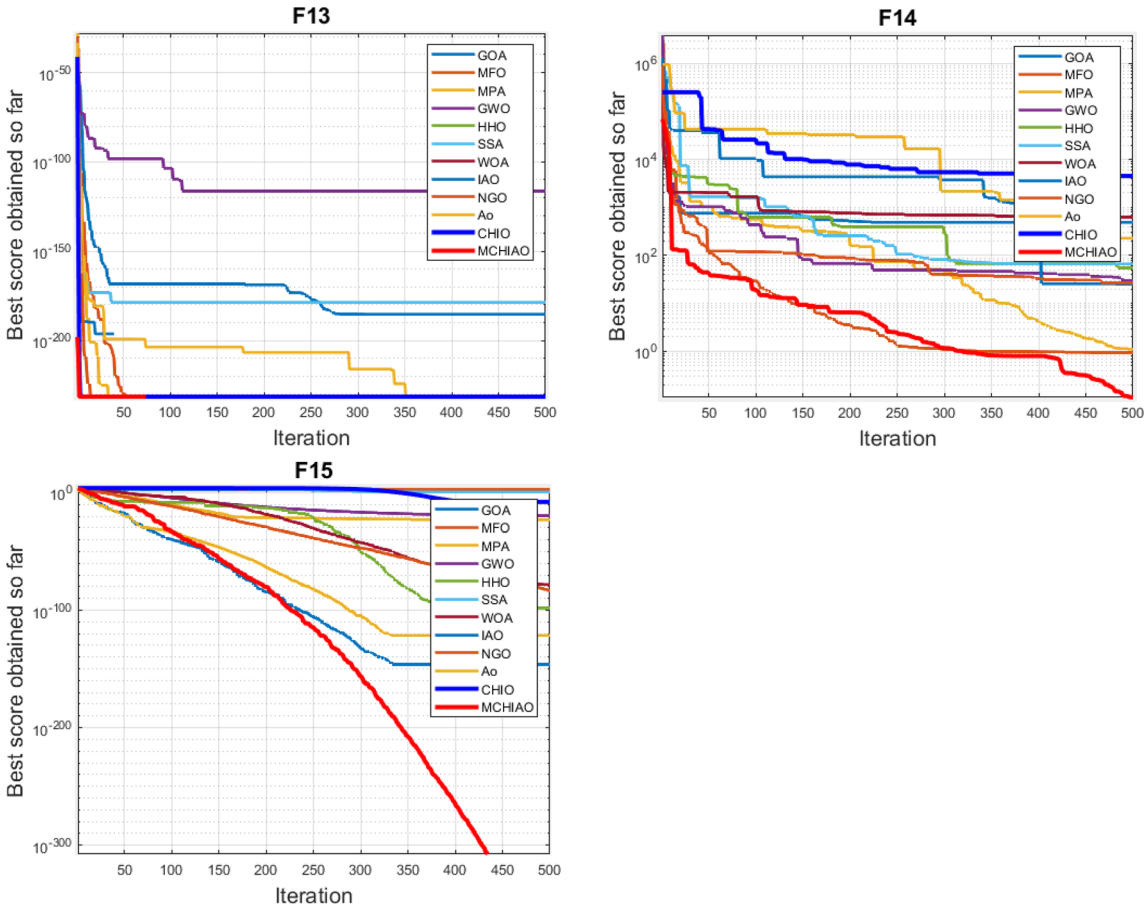


Fig. 11 continued

structural parameters: wire diameter (d or x_1), mean coil diameter (D or x_2), and the number of active coils (P or x_3). Figure 14 presents the spring and its parameters [38].

This problem has the following mathematical model:

$$\text{Consider } X = [x_1, x_2, x_3] = [d, D, P].$$

$$\text{Minimize } f(x) = (x_3 + 2)x_2x_1^2. \tag{4.2}$$

$$\text{Subject to : } g_1(x) = 1 - \frac{x_2^3 x_3}{7.1785x_1^4} \leq 0,$$

$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12.566\left(x_2x_1^3\right) - x_1^4} + \frac{1}{5.108x_1^2} - 1 \leq 0,$$

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

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

With

$$0.05 \leq x_1, \leq 2, 0.25 \leq x_2 \leq 1.3, \text{ and } 2 \leq x_3 \leq 15.$$

The performance of the proposed MCHIAO is evaluated by solving the tension/compression spring design problem and compared against the same twelve state-of-the-art algorithms as shown in Table 30. The performance of the MCHIAO algorithm is better than the twelve state-of-the-art algorithms.

For 500 iterations and 51 runs, MCHIAO outperforms the other twelve algorithms as proved in Table 31. Table 31 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the tension/compression spring design problem. These results prove the significance of the proposed algorithm for all functions because the p-values are less than 0.05.

Figure 15 presents the convergence curve of the MCHIAO optimization algorithm against the other twelve algorithms in the tension/compression spring design problem.

Table 21 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for unimodal variable-dimension functions

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	1.986E-158	5.2302E-162	3.6178E-91	8.1265E-106	4.39336E-44	1.0637E-152	7.3112E-59	1.27131E-27	7.01778E-08	3.1262E-129	3.27657E-47	1.0022E-150
F2	2.2775E-159	4.8948E-156	2.55873E-57	4.82466E-63	4.57659E-59	1.0239E-155	2.53899E-60	1.06181E-89	2.15686E-82	1.5536E-69	9.33839E-30	1.7497E-146
F3	6.6373E-159	3.4366E-159	1.59739E-85	3.2284E-100	2.4796E-50	4.8117E-157	7.85046E-55	7.82047E-16	4.45344E-06	1.61606E-07	1.73593E-48	1.5553E-150
F4	1.8028E-162	5.8536E-165	2.0538E-108	1.0656E-125	6.58467E-45	5.693E-163	5.4978E-165	2.4645E-15	0.000240266	5.3276E-10	1.81799E-58	9.6919E-157
F5	5.4503E-160	2.0294E-160	1.6244E-131	1.2823E-125	3.71441E-09	3.8912E-117	6.90237E-89	6.31459E-17	2.8402E-17	0.06721329	3.2695E-127	4.8945E-156
F6	6.5776E-193	9.80862E-16	3.1476E-178	1.957E-181	0.311016542	2.0738E-185	0.106399612	2.3289E-114	2.8176E-108	1.9018E-185	1.6806E-179	0.050866517
F7	5.2709E-164	1.4715E-158	4.50989E-82	1.19393E-95	5.6458E-45	8.0663E-163	1.67821E-50	4.67427E-34	2.03532E-24	9.70037E-65	1.0152E-46	4.842E-144
F8	7.2894E-166	8.682E-168	5.86488E-98	1.5246E-120	1.03548E-21	5.2796E-164	7.16689E-95	0.364017446	0.028264186	1.81768E-39	2.83363E-42	1.0542E-163
F9	4.166E-161	1.1437E-162	1.1897E-114	1.4708E-118	1.32034E-32	2.0016E-158	2.0925E-116	0.000258088	2.43395E-10	5.6304E-99	1.7867E-113	1.3388E-155
F10	9.8126E-164	8.0449E-163	8.84141E-86	2.16012E-96	9.49009E-40	4.3071E-153	1.00925E-48	7.8712E-31	6.73194E-30	5.47711E-53	9.30291E-40	2.4231E-147
F11	3.074E-156	6.8236E-158	1.0549E-104	3.766E-107	9.80978E-32	5.485E-155	8.1686E-108	1.36746E-32	3.73358E-31	0.013886571	4.198E-104	1.676E-144
F12	1.6743E-159	7.5487E-164	3.4977E-116	2.2805E-133	3.31314E-43	2.6267E-160	1.7657E-130	1.20914E-32	3.93392E-27	1.10828E-20	2.20282E-75	2.957E-153
F13	9.0771E-169	2.912E-145	4.717E-147	6.0106E-168	6.76891E-07	2.4451E-186	3.83578E-81	7.9533E-135	2.67436E-43	2.57695E-91	3.4413E-148	5.6134E-149
F14	2.5351E-157	2.0591E-162	1.50233E-89	3.2334E-102	5.16346E-49	1.0365E-155	1.72562E-62	1.04841E-13	4.32988E-06	7.82395E-45	6.46616E-53	6.8606E-152
F15	2.5351E-157	2.0591E-162	1.50233E-89	3.2334E-102	5.16346E-49	1.0365E-155	1.72562E-62	1.04841E-13	4.32988E-06	7.82395E-45	6.46616E-53	6.8606E-152

Table 22 Description of Multimodal fixed-dimension problems

f.No.	Name	Vars	Range	f_{min}
F1	Egg Crate	2	[−5,5]	0
F2	Ackley N.3	2	[−32,32]	−195.629
F3	Adjiman	2	[−1,2]	−2.02181
F4	Bird	2	[−2pi,2pi]	−106.765
F5	Camel 6 Hump	2	[−5,5]	−1.0316
F6	Branin RCOS	2	[−5,5]	0.397887
F7	Goldstien Price	2	[−2,2]	3
F8	Hartman 3	3	[0,1]	−3.86278
F9	Hartman 6	6	[0,1]	−3.32236
F10	Cross−in−tray	2	[−10,10]	−2.06261
F11	Bartels Conn	2	[−500,500]	1
F12	Bukin 6	2	[(−15,−5),(−5,−3)]	180.3276
F13	Carrom Table	2	[−10,10]	−24.1568
F14	Chichinadze	2	[−30,30]	−43.3159
F15	Cross function	2	[−10,10]	0
F16	Cross leg table	2	[−10,10]	−1
F17	Crowned cross	2	[−10,10]	0.0001
F18	Easom	2	[−100,100]	−1
F19	Giunta	2	[−1,1]	0.060447
F20	Helical Valley	3	[−10,10]	0
F21	Himmelblau	2	[−5,5]	0
F22	Holder	2	[−10,10]	−19.2085
F23	Pen Holder	2	[−11,11]	−0.96354
F24	Test Tube Holder	2	[−10,10]	−10.8723
F25	Shubert	2	[−10,10]	−186.731
F26	Shekel	4	[0,10]	−10.5364
F27	Three-Hump Camel	2	[−5,5]	0

5.8.2 Pressure vessel design

The objective is to obtain a pressure vessel design with the minimum fabrication cost. Figure 16 demonstrates the pressure vessel and the design parameters [38].

The four decision variables in this problem are head Thickness (T_h), shell thickness (T_s), length of the cylindrical section neglecting head(L), and Inner radius (R). This problem's mathematical formulation is as follows:

Consider $X = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L]$.

x_1 *and* x_2 are discrete while x_3 *and* x_4 are continuous.

The fitness function is nonlinear with linear and nonlinear inequality constraints.

$$\text{Minimize} f(x) = 0.6224x_1x_3x_4 + 1.778x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3. \tag{4.3}$$

$$\text{Subject to} : \text{g1}(x) = -x_1 + 0.0193x_3 \leq 0,$$

$$\text{g2(x)} = -x_2 + 0.00954x_3 \leq 0,$$

$$\text{g3(x)} = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000,$$

$$\text{g4(x)} = x_4 - 240 \leq 0,$$

With

$$0 \leq x_1, x_2 \leq 100, \text{ and } 10 \leq x_3, x_4 \leq 200.$$

Table 32 provides the results of this problem for the proposed MCHIAO against the same twelve algorithms. Table 33 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the pressure vessel design problem.

Table 23 MCHIAO performance against CHIO and AO algorithms is conducted on Multimodal fixed-dimension functions

Function	Algorithm indices	AO[31]	CHIO[30]	MCHIAO
F1	Mean	5.503E-101	7.156E-147	**0**
	Std	9.532E-101	1.239E-146	0
	Rank	3	2	**1**
F2	Mean	−195.62826	−195.62895	**−195.62903**
	Std	0.00103777	6.0025E-05	0
	Rank	3	2	**1**
F3	Mean	−2.0218067	−2.0218068	**−2.0218068**
	Std	4.4063E-08	4.256E-10	0
	Rank	3	2	**1**
F4	Mean	−106.74417	−106.7518	**−106.76454**
	Std	0.01326392	0.00639826	1.0049E-14
	Rank	3	2	**1**
F5	Mean	−1.031279	−1.0316223	**−1.0316285**
	Std	0.00031171	4.87E-06	0
	Rank	3	2	**1**
F6	Mean	0.3981097	0.39813503	**0.39788736**
	Std	0.00017101	0.00027844	0
	Rank	2	3	**1**
F7	Mean	3.05659703	3.00001478	**3**
	Std	0.01086624	6.7114E-06	7.6919E-16
	Rank	3	2	**1**
F8	Mean	−3.8502528	−3.8545256	**−3.8627821**
	Std	0.00380409	0.00016892	0
	Rank	3	2	**1**
F9	Mean	−3.1000292	−2.8642107	−3.2823641
	Std	0.13315313	0.30230971	0.06864298
	Rank	2	3	**1**
F10	Mean	−2.0626116	−2.0626084	**−2.0626119**
	Std	1.6325E-07	1.784E-06	0
	Rank	2	3	**1**
F11	Mean	1.00000001	**1**	**1**
	Std	1.9783E-08	0	0
	Rank	2	1	**1**
F13	Mean	−24.152935	−24.107559	**−24.156816**
	Std	0.00369228	0.02224092	4.3512E-15
	Rank	2	3	**1**
F14	Mean	**−42.793565**	−42.559128	−42.646245
	Std	0.2567107	0.06249043	0.25819889
	Rank	**1**	3	2
F15	Mean	4.8482E-05	**4.8482E-05**	**4.8482E-05**
	Std	1.7632E-10	7.6852E-11	6.7763E-21
	Rank	3	**1**	**1**
F16	Mean	−0.0002547	−0.00019	**−0.3618163**
	Std	7.2683E-05	2.8271E-05	0.55413702
	Rank	2	3	**1**
F17	Mean	0.29091024	0.53252908	**0.01354889**
	Std	0.25415889	0.05619264	0.02142434
	Rank	2	3	**1**

Table 23 (continued)

Function	Algorithm indices	AO[31]	CHIO[30]	MCHIAO
F18	Mean	−0.9999221	−0.9997226	**−1**
	Std	5.3918E-05	0.00012847	0
	Rank	2	3	**1**
F19	Mean	0.06447602	0.06447233	**0.06447042**
	Std	9.3665E-06	5.5926E-07	0
	Rank	3	2	**1**
F20	Mean	0.02878722	4.12675429	**1.6435E-30**
	Std	0.02300902	3.5748105	2.8466E-30
	Rank	2	3	**1**
F21	Mean	0.00702172	0.00853319	**0**
	Std	0.00716048	0.00927898	0
	Rank	2	3	**1**
F22	Mean	−19.19857	−19.187159	**−19.208503**
	Std	0.0143746	0.01012612	0
	Rank	2	3	**1**
F23	Mean	−0.9635347	−0.9635301	**−0.9635348**
	Std	1.4268E-07	3.1812E-06	0
	Rank	2	3	**1**
F24	Mean	−10.871754	−10.857572	**−10.8723**
	Std	0.0004693	0.011744	0
	Rank	2	3	**1**
F25	Mean	−186.47518	−186.02185	**−186.73091**
	Std	0.32438566	0.20930624	2.8422E-14
	Rank	2	3	**1**
F26	Mean	−10.527056	−2.3000796	**−10.53641**
	Std	0.01316684	2.3485831	0
	Rank	2	3	**1**
F27	Mean	1.47E-139	8.101E-163	**0**
	Std	2.546E-139	0	0
	Rank	3	2	**1**
Percentage		2.38461538	2.53846154	1.03846154
Total Rank		2	3	**1**

These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 17 represents the convergence curves of the MCHIAO against the other twelve optimization algorithms in the pressure vessel problem.

5.8.3 Welded beam design

Another example used to assess MCHIAO performance in the engineering domain is a well-known welded beam design, as shown in Fig. 18 [22].

In this problem, the aim is to establish the best design variables to minimize the overall manufacturing cost of a welded beam is subject to Shear stress (τ), Bending stress (θ), the bar's buckling load (Pc), the beam end deflection (δ), and other constraints. The four variables in this problem are the bar length (l), the thickness of the weld (h), height (t), and thickness (b). The mathematical formulation of this problem is as follows:

Consider $X = [x_1, x_2, x_3, x_4] = [h, l, t, b]$.

Table 24 MCHIAO performance against twelve different algorithms is conducted on Multimodal fixed-dimension functions

Function	Algorithm indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F1	Mean	3.16273245	1.721E-103	3.792E-72	4.909E-132	7.635E-123	2.5001E-13	7.168E-122	5.503E-101	9.804E-150	0.01660292	6.955E-146	7.156E-147	**0**
	Std	5.47801329	2.946E-103	6.5394E-72	8.502E-132	1.278E-122	3.9437E-13	8.775E-122	9.532E-101	1.325E-149	0.0287571	1.205E-145	1.239E-146	0
	Rank	13	8	10	5	6	11	7	9	2	12	4	3	**1**
F2	Mean	−195.62903	**−195.62903**	−195.62903	−195.62903	−195.62903	−195.62903	−195.62903	−195.62826	−195.62851	−195.26416	−195.62903	−195.62895	**−195.62903**
	Std	7.8669E-12	0	2.8422E-14	6.1621E-08	8.4314E-08	3.0479E-13	5.7471E-07	0.00103777	0.00051491	0.26468497	2.8422E-14	6.0025E-05	0
	Rank	6	**1**	3	8	7	5	9	12	11	13	3	10	**1**
F3	Mean	−2.0218068	**−2.0218068**	**−2.0218068**	−2.0218068	**−2.0218068**	−2.0218068	−2.0218068	−2.0218067	−2.0218066	−2.021272	**−2.0218068**	−2.0218068	**−2.0218068**
	Std	0	0	0	1.5332E-11	4.4409E-16	0	0	4.4063E-08	1.0867E-07	0.00054648	3.1402E-16	4.256E-10	0
	Rank	6	**1**	**1**	9	**1**	6	6	11	12	13	**1**	10	**1**
F4	Mean	−87.311699	−106.76454	−106.76454	−106.7645	−106.76453	−106.76454	−106.7645	−106.74417	−106.72799	−100.9019	−106.76454	−106.7518	**−106.76454**
	Std	33.6933034	1.4211E-14	1.4211E-14	4.0606E-05	1.9711E-05	1.21E-13	5.0426E-05	0.01326392	0.03329212	3.73530021	3.0146E-14	0.00639826	1.0049E-14
	Rank	13	**1**	4	7	6	5	8	10	11	12	3	9	**1**
F5	Mean	−0.7595736	**−1.0316285**	−1.0316285	−1.0316281	−1.0316285	−1.0316285	−1.0316285	−1.031279	−1.031486	−0.9872211	**−1.0316285**	−1.0316223	**−1.0316285**
	Std	0.47121287	0	0	1.9948E-07	2.0728E-10	4.2884E-15	4.1642E-10	0.00031171	0.00011482	0.03789106	2.2204E-16	4.87E-06	0
	Rank	13	**1**	4	8	6	5	7	11	10	12	**1**	9	**1**
F6	Mean	0.39788736	**0.39788736**	**0.39788736**	0.39788794	0.39789267	0.39788736	0.39791603	0.3981097	0.39800839	0.45842164	**0.39788736**	0.39813503	**0.39788736**
	Std	4.3379E-14	0	0	2.6612E-07	7.7256E-06	2.7135E-15	4.7246E-05	0.00017101	0.00010449	0.09451157	0	0.00027844	0
	Rank	6	**1**	**1**	7	8	5	9	11	10	13	**1**	12	**1**
F7	Mean	3	3	12	3.00010407	3.00000103	3	3.00001596	3.05659703	3.01467997	3.99350182	3	3.00001478	**3**
	Std	7.5682E-13	1.831E-15	15.5884573	7.7849E-05	1.7896E-06	1.2895E-13	2.7639E-05	0.01086624	0.01192854	0.70476246	1.1749E-15	6.7114E-06	7.6919E-16
	Rank	5	2	13	9	6	4	8	11	10	12	3	7	**1**
F8	Mean	−2.4160431	−3.860155	−3.8627821	−3.86277	−3.8614867	−3.8627732	−3.8361255	−3.8502528	−3.8525844	−3.8475736	**−3.8627821**	−3.8545256	**−3.8627821**
	Std	0.60119223	0.0045504	4.4409E-16	7.6364E-06	0.00043735	1.4743E-05	0.03359638	0.00380409	0.00784994	0.00439004	4.4409E-16	0.00016892	0
	Rank	13	7	**1**	5	6	4	12	10	9	11	**1**	8	**1**
F9	Mean	−3.2171656	−3.2427331	−3.3219952	−3.2399588	−3.1794868	**−3.3219952**	−3.2753579	−3.1000292	−3.2080637	−2.894638	−3.3219952	−2.8642107	−3.2823641
	Std	0.09107336	0.06864298	4.8393E-10	0.07105676	0.10485818	4.2882E-13	0.07287727	0.13315313	0.06138601	0.16107271	1.2274E-08	0.30230971	0.06864298
	Rank	8	6	2	7	10	**1**	5	11	9	12	3	13	4
F10	Mean	−1.9719418	**−2.0626119**	−2.0626119	−2.0626118	−2.0626119	−2.0626119	−2.0626119	−2.0626116	−2.0626112	−2.0600825	**−2.0626119**	−2.0626084	**−2.0626119**
	Std	0.15704513	0	1.1322E-15	3.2983E-08	1.5191E-08	1.4729E-15	8.8062E-10	1.6325E-07	9.6903E-07	0.00320463	0	1.784E-06	0
	Rank	13	**1**	4	8	7	5	6	9	10	12	**1**	11	**1**
F11	Mean	1.00007054	**1**	**1**	**1**	**1**	1.00006911	**1**	1.00000001	**1**	**1**	**1**	**1**	**1**
	Std	4.6602E-05	0	0	0	0	7.0822E-05	0	1.9783E-08	0	0	0	0	0
	Rank	13	**1**	**1**	**1**	**1**	12	**1**	11	**1**	**1**	**1**	**1**	**1**
F13	Mean	−9.9072251	−24.156816	−24.156816	−24.156614	−24.156812	−24.156816	−24.156671	−24.152935	−24.153385	−23.937113	−24.156816	−24.107559	**−24.156816**
	Std	12.3405074	4.3512E-15	5.8933E-11	0.0001903	5.4242E-06	1.585E-13	0.00014229	0.00369228	0.00555929	0.20366822	7.138E-13	0.02224092	4.3512E-15
	Rank	13	2	5	8	6	3	7	10	9	12	4	11	**1**
F14	Mean	−42.646245	**−42.944387**	**−42.944387**	−42.768713	−42.944369	−42.646245	−42.646244	−42.793565	−42.941963	−42.576854	**−42.944387**	−42.559128	−42.646245
	Std	0.2581987	0	0	0.23829447	3.053E-05	0.25819889	0.25819837	0.2567107	0.0012821	0.11233337	0	0.06249043	0.25819889
	Rank	10	**1**	**1**	7	4	9	11	6	5	12	**1**	13	8
F15	Mean	4.9959E-05	**4.8482E-05**	4.8482E-05	4.8482E-05	4.8482E-05	4.8482E-05	4.8482E-05	4.8482E-05	4.8482E-05	4.8495E-05	4.8482E-05	4.8482E-05	4.8482E-05
	Std	2.5574E-06	0	2.1958E-20	2.6451E-13	7.1055E-14	2.1958E-20	1.5171E-14	1.7632E-10	6.9231E-13	8.3094E-09	8.2992E-21	7.6852E-11	6.7763E-21
	Rank	13	**1**	4	8	7	5	6	11	9	12	3	10	**2**

Table 24 (continued)

Function	Algorithm indices	GOA [9]	MFO [10]	MPA [11]	GWO [14]	HHO [15]	SSA [16]	WOA [17]	AO [31]	IAO [33]	NOA [34]	NGO [35]	CHIO [30]	MCHIAO
F16	Mean	−0.0003711	−0.0300059	−0.0093304	−0.0004477	−0.3362431	−0.0006434	−0.0006463	−0.0002547	−0.0002741	−0.3334324	−0.0073453	−0.00019	**−0.3618163**
	Std	2.6956E-05	0.04743418	0.01359368	0.00041696	0.57483171	0.00015155	0.00043158	7.2683E-05	0.00010434	0.57726425	0.00702421	2.8271E-05	0.55413702
	Rank	10	4	5	9	2	8	7	12	11	3	6	13	**1**
F17	Mean	0.24633314	**0.01354889**	0.08499928	0.32851968	0.03302634	0.1375817	0.24117329	0.29091024	0.36408977	0.4763086	0.10173799	0.53252908	**0.01354889**
	Std	0.03541247	0.02142434	0.05023702	0.30222057	0.00750038	0.08156559	0.02854381	0.25415889	0.00541588	0.41383354	0.09546999	0.05619264	0.02142434
	Rank	8	**1**	4	10	3	6	7	9	11	12	5	13	**1**
F18	Mean	−0.3333333	**−1**	**−1**	−0.9999972	−0.9999974	−1	−0.9999987	−0.9999221	−0.9999593	−0.261859	**−1**	−0.9997226	**−1**
	Std	0.57735027	0	0	2.5017E-06	3.7483E-06	2.8484E-12	1.2848E-06	5.3918E-05	5.1566E-05	0.40789302	0	0.00012847	0
	Rank	**12**	**1**	**1**	**8**	**7**	**5**	**6**	**10**	**9**	**13**	**1**	**11**	**1**
F19	Mean	0.11230569	**0.06447042**	**0.06447042**	0.06447044	0.06447062	0.06447042	0.06447049	0.06447602	0.06447212	0.06494587	**0.06447042**	0.06447233	**0.06447042**
	Std	0.08285311	0	0	2.8773E-08	1.7227E-07	6.4349E-17	9.3991E-08	9.3665E-06	2.5097E-06	0.0004102	0	5.5926E-07	0
	Rank	13	**1**	**1**	6	8	5	7	11	9	12	**1**	10	**1**
F20	Mean	7.2989079	0.99830972	1.5931E-23	0.00518325	0.0233059	0.00364021	0.59569138	0.02878722	0.0113952	7.37489458	3.3374E-14	4.12675429	**1.6435E-30**
	Std	12.0805824	0.80969983	2.7518E-23	0.00404752	0.02031207	0.00284593	1.03163213	0.02300902	0.0047161	3.56072639	5.7803E-14	3.5748105	2.8466E-30
	Rank	12	10	2	5	7	4	9	8	6	13	3	11	**1**
F21	Mean	5.9948E-12	5.2591E-31	7.2786E-14	3.8613E-05	6.53E-05	1.1767E-13	6.3875E-10	0.00702172	0.00269651	0.08132504	**0**	0.00853319	**0**
	Std	5.2928E-12	4.5545E-31	1.2607E-13	5.4545E-05	0.0001131	1.0309E-13	8.1653E-10	0.00716048	0.00333618	0.04220644	0	0.00927898	0
	Rank	6	3	4	8	9	5	7	11	10	13	**1**	12	**1**
F22	Mean	−19.208503	**−19.208503**	−19.208503	−19.20827	−19.208503	−19.208503	−19.208503	−19.19857	−19.204033	−18.998729	−19.208503	−19.187159	**−19.208503**
	Std	7.2011E-13	0	2.524E-12	0.00015556	2.1464E-14	3.1594E-13	9.0797E-09	0.0143746	0.00196807	0.30160774	0	0.01012612	0
	Rank	7	**1**	6	9	4	5	8	11	10	13	3	12	**1**
F23	Mean	−0.9306742	**−0.9635348**	−0.9635348	−0.9635345	−0.9635348	−0.9635348	−0.9635343	−0.9635347	−0.9635346	−0.9608351	**−0.9635348**	−0.9635301	**−0.9635348**
	Std	0.03362996	0	3.1576E-14	4.3321E-07	2.7541E-11	1.3597E-16	9.6644E-07	1.4268E-07	2.2671E-07	0.0010039	0	3.1812E-06	0
	Rank	13	**1**	5	9	6	4	10	7	8	12	**1**	11	**1**
F24	Mean	−10.398968	−10.852493	**−10.8723**	−10.872283	−10.865697	−10.865698	−10.865682	−10.871754	−10.865509	−10.815934	**−10.8723**	−10.857572	**−10.8723**
	Std	0.5006415	0	0	1.4765E-05	0.01143629	0.01143589	0.01142276	0.0004693	0.011328	0.02485868	0	0.011744	0
	Rank	13	11	**1**	4	7	6	8	5	9	12	**1**	10	**1**
F25	Mean	−140.65621	−186.73091	−186.73091	−186.72661	−186.73091	−186.73091	−186.73088	−186.47518	−186.21309	−174.7527	−186.73091	−186.02185	**−186.73091**
	Std	79.8037167	3.4809E-14	8.9135E-11	0.0065522	4.7601E-07	6.5283E-11	3.0371E-05	0.32438566	0.44892884	12.9136218	1.8232E-09	0.20930624	2.8422E-14
	Rank	13	2	4	8	6	3	7	9	10	12	5	11	**1**
F26	Mean	−2.1752075	−5.7261558	−8.7337668	−5.1254366	−5.1213954	−6.0288749	−5.8177963	−10.527056	−10.510832	−2.4968175	−10.536409	−2.3000796	**−10.53641**
	Std	0.43185632	4.19744128	3.12226928	4.68303316	0.00604033	4.13158843	3.7829025	0.01316684	0.02642109	0.83446978	9.6276E-07	2.3485831	0
	Rank	13	8	5	9	10	6	7	3	4	11	2	12	**1**
F27	Mean	0.09954615	9.425E-99	4.4867E-68	7.81E-118	1.211E-112	1.4113E-14	3.2647E-85	1.47E-139	2.251E-152	8.114E-19	1.465E-150	8.101E-163	**0**
	Std	0.17241899	1.5599E-98	7.7575E-68	1.353E-117	2.097E-112	1.1098E-14	5.2927E-85	2.546E-139	3.785E-152	1.4054E-18	2.333E-150	0	0
	Rank	13	8	10	6	7	12	9	5	3	11	4	2	**1**
Percentage	10.6923077	3.23076923	3.92307692	6.92307692	6	5.73076923	7.46153846	9.38461538	8.38461538	11.3846154	2.42307692	9.80769231	1.42307692	
Total Rank	12	3	4	7	6	5	8	10	9	13	2	11	**1**	

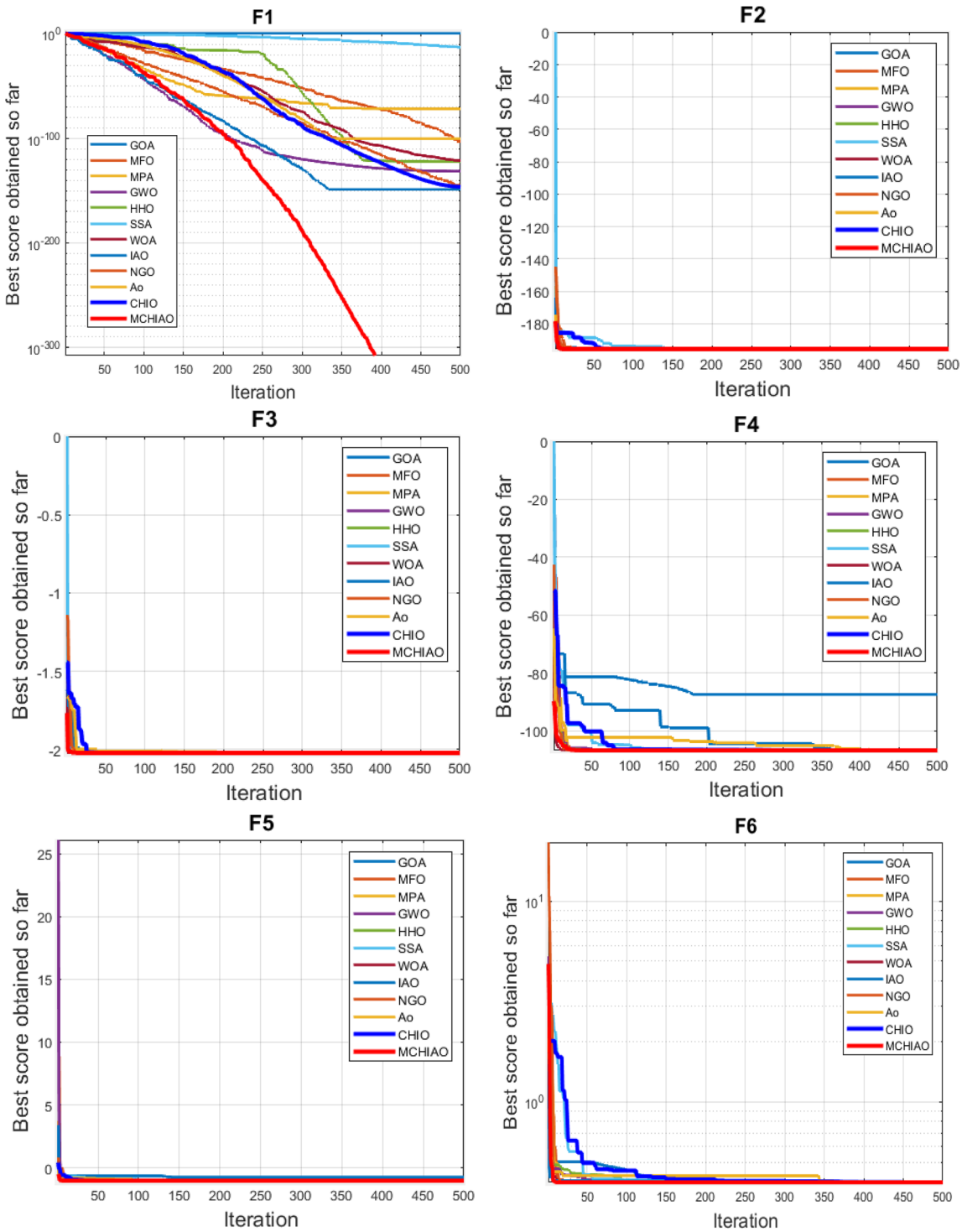


Fig. 12 The Multimodal fixed-dimension functions and convergence plots

$$\text{Minimize} f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2). \quad (4.4)$$

$$\text{Subject to g}\,1(x) = \tau(x) - 13600 \leq 0,$$
$$\text{g2}\ (\text{x}) = \sigma(\text{x}) - 30000 \leq 0,$$

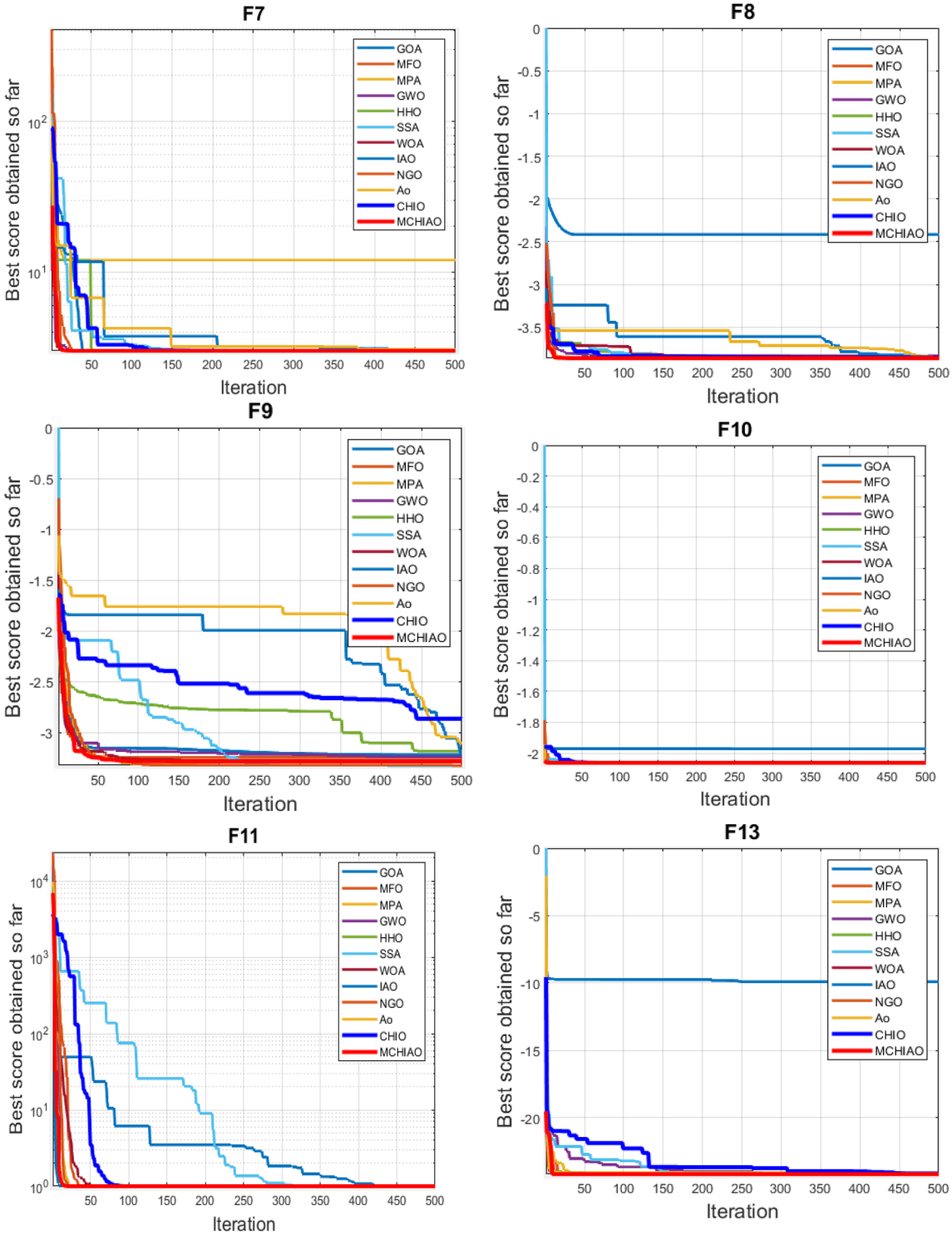


Fig. 12 continued

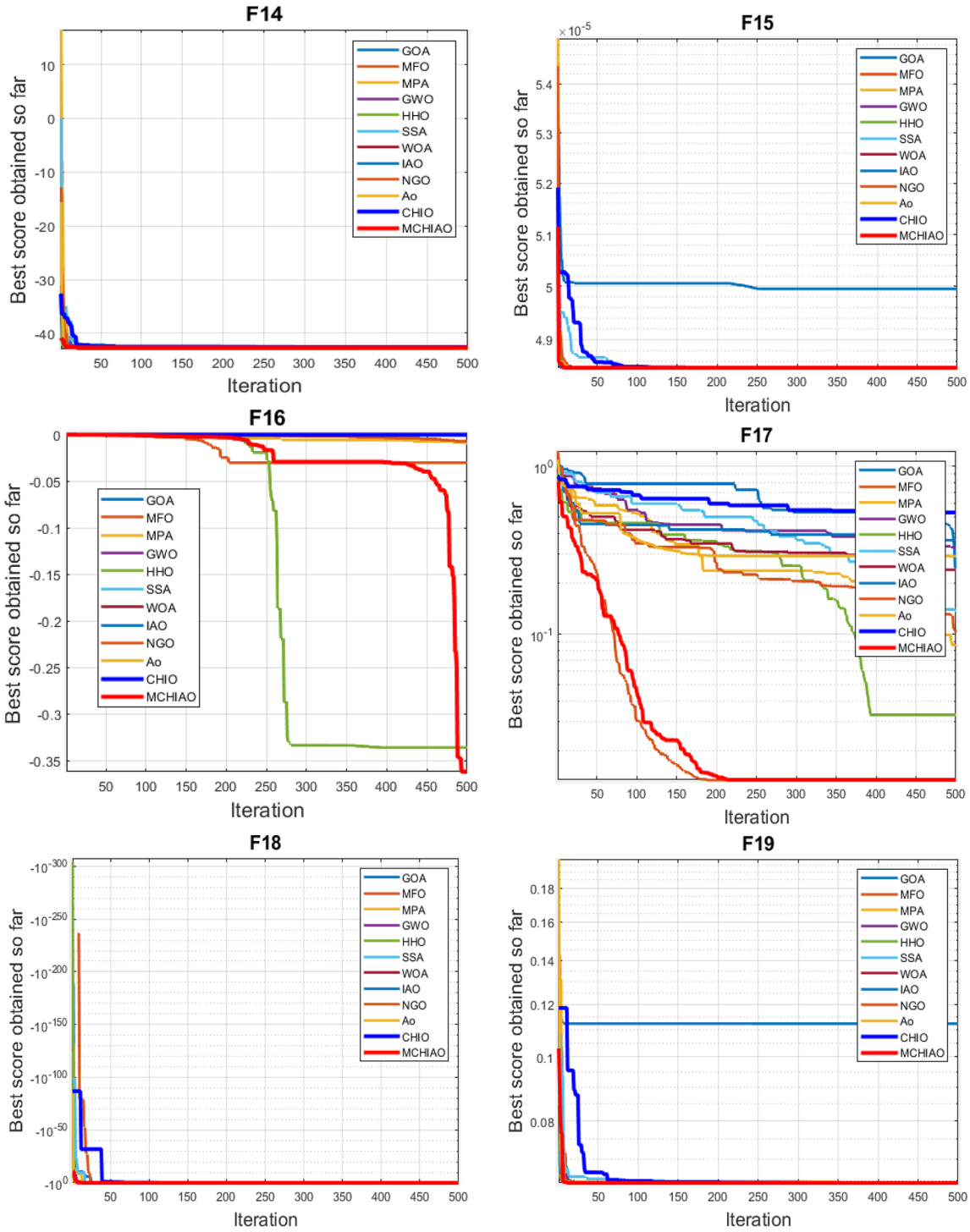


Fig. 12 continued

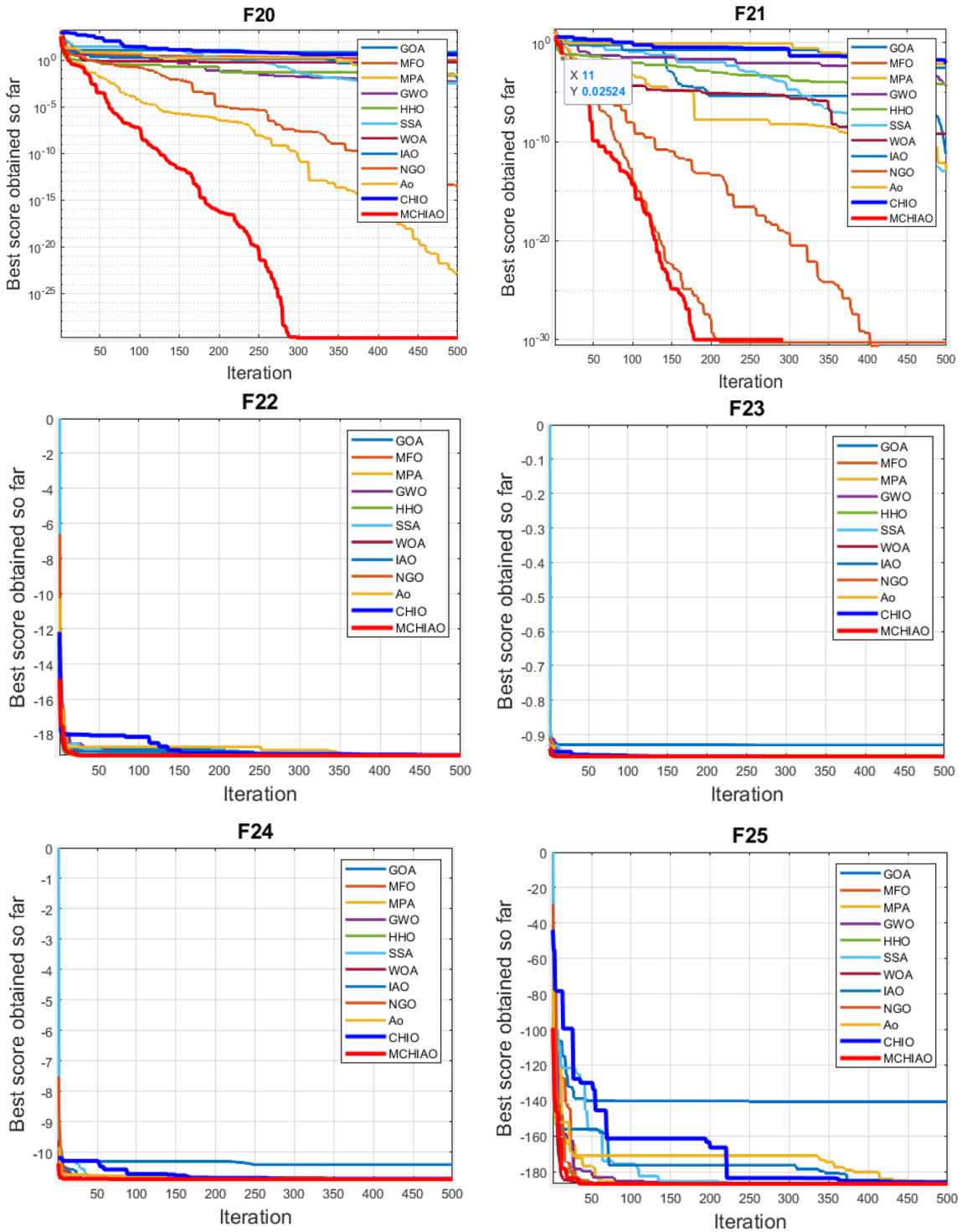


Fig. 12 continued

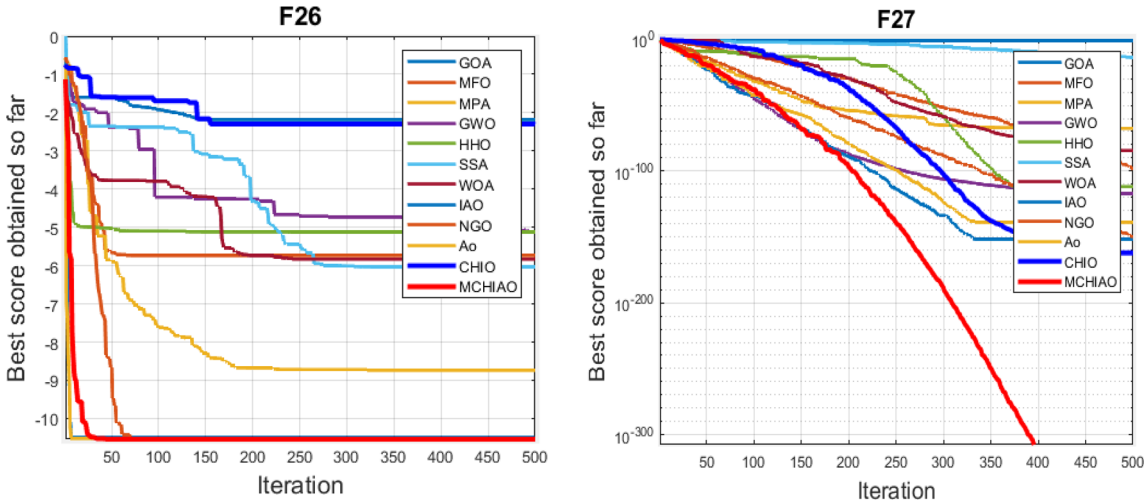


Fig. 12 continued

$$g3\ (x) = \ x1\ - \ x4 \le 0,$$

$$g4\ (x) = 0.10471x21 + 0.04811x3x4\ (14 + x2) - 5.0 \le 0,$$

$$g5(x) = 0.125 - \ x1 \le 0,$$

$$g6(x) = \delta\,(x) - 0.25 \le 0,$$

$$g7\ (x) = 6000 - \ pc\ (x) \le 0.$$

$$\text{where } \tau(x) = \sqrt{\tau' + (2\tau\tau')\frac{x_2}{2R} + (\tau'')^2},$$

$$\tau' = \frac{6000}{\sqrt{2x_1x_2}},$$

$$\tau'' = \frac{MR}{J},$$

$$M = 6000\left(14 + \frac{x_2}{2}\right),$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},$$

$$J = 2\left\{x_1x_2\sqrt{2}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},$$

$$\sigma(x) = \frac{504000}{x_4x_3^2}$$

$$\delta(x) = \frac{65856000}{(30.10^6)x_4x_3^3}$$

$$pc\ (x) = \frac{4.013\left(30.10^6\right)\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{x_3}{28}\sqrt{\frac{30.10^6}{4(12.10^6)}}\right).$$

With

$$0.1 \le x_1, x_4 \le 2 \text{ and } 0.1 \le x_2, x_3 \le 10.$$

Table 34 presents the optimization results of the welded beam design problem for the proposed MCHIAO against the same twelve algorithms. Table 35 presents the p-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the welded beam design problem. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 19 below illustrates the MCHIAO plot against the competitive optimization results in the welded beam design problem.

5.8.4 Speed Reducer problem

It is a weight minimization problem that involves the design of a speed reducer for a small aircraft's internal combustion engine. It is a challenging benchmark because it has seven design variables (x_1 *to* x_7). Figure 20 illustrates the problem's schematic [39].

This problem's mathematical model is as follows:

$$\begin{aligned}\text{Consider } X &= [x_1, x_2, x_3, x_4, x_5, x_6, x_7] \\ &= [b, m, p, l_1, l_2, d_1, d_2].\end{aligned}$$

$$\begin{aligned}\text{Minimize } f(x) &= 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) \\ &- 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right).\end{aligned} \tag{4.5}$$

$$\text{Subject to}: g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0,$$

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

Table 25 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for Multimodal fixed-dimension functions

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	1.1347E-165	4.01674E-54	1.73596E-40	9.55169E-14	6.90822E-49	4.0247E-144	3.98268E-39	1.50775E-40	7.46288E-12	0.01335283	8.89151E-27	5.11605E-36
F2	8.3225E-147	0.33989194	1.35647E-89	3.1181E-138	4.0158E-116	3.7425E-137	1.0866E-114	3.3514E-158	2.3117E-154	1.4277E-173	4.47121E-71	1.5585E-144
F3	4.6732E-152	6.2885E-06	0.030993739	2.135E-143	1.6538E-105	2.1554E-139	7.5579E-98	1.978E-164	9.6753E-159	8.4444E-176	9.00083E-13	1.0998E-145
F4	2.437E-173	4.30677E-37	6.6126E-80	8.4573E-130	1.8652E-107	2.6449E-114	4.5636E-107	1.2242E-154	2.6382E-155	1.4657E-171	2.37049E-23	7.3218E-139
F5	1.546E-168	0.480996505	1.20476E-67	4.1199E-135	1.36411E-88	1.6522E-117	4.07237E-86	7.8815E-160	2.5665E-160	4.2969E-168	0.486286377	1.6042E-139
F6	1.7013E-161	0.000320523	3.1123E-29	4.8053E-166	3.5488E-163	2.9915E-156	5.0613E-161	5.6488E-173	2.1925E-173	2.03265E-53	1.76866E-06	3.2937E-171
F7	3.5412E-129	0.001562829	1.4597E-164	3.8492E-131	6.45E-114	2.0389E-122	1.5097E-110	6.8644E-157	3.5689E-156	0.200158314	2.42587E-37	1.5008E-134
F8	5.5002E-171	8.3903E-142	1.23222E-51	1.1548E-137	3.9878E-147	3.3048E-136	5.0127E-159	1.4543E-167	5.8248E-168	1.5816E-169	7.03809E-18	1.283E-156
F9	3.6916E-132	1.319E-114	1.66507E-83	2.2948E-126	1.0242E-149	0.001439352	5.82579E-91	5.9251E-164	5.3274E-163	5.8406E-167	4.4557E-79	8.7537E-157
F10	1.0009E-179	0.902374384	3.6231E-122	4.5978E-150	6.9342E-137	9.6693E-139	1.3902E-121	7.9475E-147	2.3956E-147	5.558E-177	1.49822E-14	6.1243E-159
F11	5.96E-164	2.94611E-10	0.06705344	0.586594865	3.34343E-36	3.2713E-160	3.99538E-14	1.6877E-136	1.45269E-37	4.22579E-08	0.128827174	1.44151E-15
F13	1.4162E-167	8.87746E-31	8.183E-136	1.8463E-155	2.1593E-146	1.0283E-140	5.3622E-147	5.2544E-162	2.4593E-168	3.183E-166	7.1826E-114	5.9362E-157
F14	3.0792E-150	2.7799E-165	3.2101E-139	3.13276E-42	8.505E-130	9.7311E-146	1.2602E-136	6.27952E-76	1.02778E-21	1.8281E-177	6.3431E-165	3.2618E-167
F15	3.5771E-176	2.3106E-62	1.2109E-115	8.1232E-147	1.0388E-119	7.2113E-136	2.3997E-120	3.758E-161	6.5005E-131	4.1461E-128	5.01965E-90	7.6382E-157
F16	2.1078E-143	1.72731E-06	8.6854E-22	5.3249E-112	0.000291828	4.2477E-128	1.9214E-102	1.8545E-131	6.354E-121	1.7869E-125	9.13569E-49	1.5961E-157
F17	5.7503E-163	0.140132588	3.2316E-128	1.8146E-150	8.7034E-114	1.0984E-141	6.3447E-140	2.1443E-139	6.1962E-149	2.05723E-28	1.0879E-122	1.8061E-163
F18	1.1142E-170	0.010806149	2.15633E-23	2.1079E-135	7.1558E-130	1.651E-142	2.0276E-126	2.8589E-141	2.1527E-136	3.9845E-176	2.5327E-120	2.1746E-143
F19	5.0594E-176	0.856078639	1.89358E-11	2.8767E-152	1.777E-138	2.7595E-134	1.1121E-136	1.324E-172	1.3299E-164	8.443E-140	0.807632987	9.8912E-160
F20	1.1837E-160	4.2121E-155	3.65058E-45	1.7048E-132	2.1934E-132	4.5766E-140	4.4679E-150	1.079E-150	9.6175E-149	9.28189E-58	9.34612E-77	4.3006E-159

Table 25 (continued)

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F21	4.2585E-133	3.52178E-06	4.4874E-119	5.3569E-149	5.4518E-141	1.0315E-129	4.5628E-121	1.5179E-160	4.8851E-157	5.4828E-102	2.24663E-29	1.8552E-156
F22	9.1221E-130	0.028109244	8.5831E-113	7.9627E-145	1.2325E-104	3.3733E-122	5.2541E-105	8.5417E-159	1.5522E-152	4.2848E-173	8.13799E-74	1.1137E-151
F23	1.0581E-178	0.810757128	1.1052E-118	1.9241E-155	2.465E-126	6.4243E-132	5.1599E-138	2.7247E-157	2.4386E-138	9.3968E-178	1.05278E-32	3.3834E-162
F24	7.6098E-170	1.3508E-171	1.26997E-06	3.64429E-77	5.565E-100	3.0935E-119	2.0947E-103	1.6138E-141	4.6742E-102	2.1779E-168	7.3045E-09	1.5412E-162
F25	1.3303E-169	1.36065E-56	4.1939E-102	4.6337E-122	2.93734E-92	1.2756E-112	1.22631E-93	2.8804E-158	2.2004E-153	7.5276E-175	2.95935E-83	3.009E-150
F26	4.1787E-166	1.0177E-162	3.5371E-156	5.1059E-162	1.7703E-161	2.5552E-160	5.4149E-160	6.1204E-142	1.6576E-144	3.6268E-169	1.833E-99	2.4824E-165
F27	5.5887E-166	2.84738E-62	2.73331E-48	8.01117E-19	5.17947E-54	2.4472E-147	1.32447E-60	2.9521E-15	3.83419E-11	2.03315E-09	5.63895E-27	4.09852E-30

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

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

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

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

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

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

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

$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0,$$

$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0.$$

With

$$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5 \le x_7 \le 5.5.$$

Table 36 shows the results of this problem for the proposed MCHIAO against the same twelve algorithms. Table 37 presents the p-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the speed reducer design problem. These results prove the significance of the proposed algorithm for all functions because the p-values are less than 0.05.

Figure 21 illustrates the same 12 optimization algorithms' convergence curves against the MCHIAO on the speed reducer problem.

5.8.5 Gear train design problem

The main goal of this problem is to minimize the gear ratio for preparing the compound gear train, as shown in Fig. 22 [39].

The objective is to achieve the optimal number of teeth for four train gears to minimize the gear ratio. The gear ratio is defined as:

$$\text{Gear ratio} = \frac{n_B n_D}{n_F n_A}$$

The decision variables (n_j) are discrete. Where n_j stands for the number of gearwheel teeth j, with $j = A, B, D, F$.

Table 26 Description of Multimodal variable-dimension functions

f.No.	Name	Vars	Range	f_{min}
F52	Schwefel's 2.26	30	[−500,500]	−418.983
F53	Rastrigin	30	[−5.12,5.12]	0
F54	Periodic	30	[−10,10]	0.9
F55	Qing	30	[−500,500]	0
F56	Alpine N. 1	30	[−10,10]	0
F57	Xin−She Yang	30	[−5,5]	0
F58	Ackley	30	[−32,32]	0
F59	Trigonometric 2	30	[−500,500]	0
F60	Salomon	30	[−100,100]	0
F61	Styblinski−Tang	30	[−5,5]	−1174.98
F62	Griewank	30	[−100,100]	0
F63	Xin−She Yang N. 4	30	[−10,10]	−1
F64	Xin−She Yang N. 2	30	[−2pi,2pi]	0
F65	Gen. Penalized	30	[−50,50]	0
F66	Penalized	30	[−50,50]	0
F67	Michalewics	30	[0,pi]	−29.6309
F68	Quartic Noise	30	[−1.28,1.28]	0

Consider $[x_1, x_2, x_3, x_4] = [n_A, n_B, n_D, n_F]$.

$$\text{Minimize} f(x) = \left(\frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4}\right)^2. \tag{4.6}$$

$$\text{Subject to} : 12 \le x_i \le 60, i = 1, 2, 3, 4.$$

Table 38 illustrates the results of this problem for the proposed MCHIAO against the same twelve algorithms. Table 39 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the gear train design problem. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 23 demonstrates the diverse algorithms' convergence curves against the proposed MCHIAO algorithm.

5.8.6 Three-bar truss design problem

The primary purpose of truss design is to reduce the weight of the bar structures. As a result, three bars should be placed as shown in Fig. 24. As the goal is to reduce the weight of the bars in this position. This is an optimization problem with constraints. This problem has two design parameters (x_1, x_2) and three restrictive functions: deflection constraints of each bar, buckling, and stress [22].

The problem is mathematically expressed as follows:

$$\text{Objective function} : f(x) = \left(2\sqrt{2}x_1 + x_2\right) \times L \tag{4.7}$$

$$\text{Subject to} : g_1 = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} \times P - \sigma \le 0$$

$$g_2 = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} \times P - \sigma \le 0$$

$$g_3 = \frac{1}{\sqrt{2}x_1 + x_2} \times P - \sigma \le 0$$

where:

$$0 \le x_1, x_2 \le 1.$$

The constants are as follows: $L = 100\,cm, P = 2\frac{KN}{cm^2}$. and $\sigma = 2KN/cm^2$.

Table 40 presents the results of various algorithms on this problem for the proposed MCHIAO against the same twelve algorithms. Table 41 presents the *p*-values of MCHIAO which was acquired by applying the Wilcoxon test against all the other twelve algorithms for the three-bar truss design problem. These results prove the significance of the proposed algorithm for all functions because the *p*-values are less than 0.05.

Figure 25 below presents the various competitive optimization algorithms' convergence curves.

Table 27 MCHIAO performance against CHIO and AO algorithms is conducted on Multimodal variable-dimension functions

Function	Algorithm indices	AO [31]	CHIO [30]	MCHIAO
F1	Mean	282.517956	300.589882	**1.2785E-05**
	Std	11.9413383	8.17246486	1.1359E-08
	Rank	2	3	**1**
F2	Mean	**0**	5.14212146	**0**
	Std	0	2.58879245	0
	Rank	**1**	2	**1**
F3	Mean	0.90016154	7.89928594	**0.9**
	Std	0.00027979	0.23392407	0
	Rank	2	3	**1**
F4	Mean	313.189295	5353.14974	**0.00062537**
	Std	123.617533	270.969316	0.00074791
	Rank	2	3	**1**
F5	Mean	5.2698E-50	0.00234174	**1.772E-202**
	Std	9.1274E-50	0.00148203	0
	Rank	2	3	**1**
F6	Mean	1.8295E-13	2.8741E-06	**3.78E-180**
	Std	3.1687E-13	4.9699E-06	0
	Rank	2	3	**1**
F7	Mean	**−4.441E-16**	19.963105	**−4.441E-16**
	Std	0	0.00050852	0
	Rank	**1**	2	**1**
F8	Mean	**1.00109004**	78.9284682	12.9586629
	Std	0.00122459	10.182355	10.8528879
	Rank	**1**	3	2
F9	Mean	2.0611E-51	0.16654103	**0**
	Std	3.5699E-51	0.05773331	0
	Rank	2	3	**1**
F10	Mean	−907.70035	−634.46345	**−1174.985**
	Std	36.938001	27.9405838	2.9525E-08
	Rank	2	3	**1**
F11	Mean	**0**	0.00723185	**0**
	Std	0	0.01252592	0
	Rank	**1**	2	**1**
F12	Mean	−0.9448373	8.8232E-11	**−1**
	Std	0.09554454	3.2917E-11	0
	Rank	2	3	**1**
F13	Mean	3.5202E-12	3.1781E-11	**3.5124E-12**
	Std	9.3035E-15	6.1453E-13	3.0844E-20
	Rank	2	3	**1**
F14	Mean	**2.9718E-05**	2.71692004	0.00366247
	Std	2.1878E-05	0.03720067	0.00634355
	Rank	**1**	3	2
F15	Mean	1.4775E-05	0.44584183	**7.0298E-10**
	Std	2.4323E-05	0.0519485	4.8484E-10
	Rank	2	3	**1**
F16	Mean	−10.514488	−7.0549159	**−22.281646**
	Std	2.23441885	0.56532887	1.72818388
	Rank	2	3	**1**
F17	Mean	0.00011373	0.0018622	**3.9674E-05**
	Std	7.8379E-05	0.00125016	2.1887E-05
	Rank	2	3	**1**
Percentage		1.70588235	3	1.11764706
Total rank		2	**3**	**1**

Table 28 MCHIAO performance against twelve different algorithms is conducted on Multimodal variable-dimension functions

Function	Algorithm indices	GOA[9]	MFO[10]	MPA[11]	GWO[14]	HHO[15]	SSA[16]	WOA[17]	AO[31]	IAO[33]	NOA[34]	NGO[35]	CHIO[30]	MCHIAO
F1	Mean	195.779064	159.867747	150.7411	214.670394	0.04094356	173.351354	96.0868937	282.517956	4.15211916	241.944181	185.890226	300.589882	**1.2785E-05**
	Std	14.2396229	10.4484634	31.8677524	14.9551192	0.03761463	5.25518189	44.8828974	11.9413383	4.84463312	12.5722395	15.5963582	8.17246486	1.1359E-08
	Rank	9	6	5	10	2	7	4	12	3	11	8	13	**1**
F2	Mean	213.212412	175.315834	1.0611E-12	5.37710301	**0**	64.3452108	**0**	**0**	**0**	**0**	**0**	5.14212146	**0**
	Std	60.1561644	18.9323098	1.8378E-12	3.9547187	0	30.9141627	0	0	0	0	0	2.58879245	0
	Rank	13	12	8	10	**1**	11	**1**	**1**	**1**	**1**	**1**	9	**1**
F3	Mean	1.24126678	4.55150826	1.16539389	1.50173373	**0.9**	1.00013182	1.28425269	0.90016154	**0.9**	**0.9**	1.70045093	7.89928594	**0.9**
	Std	0.07968135	0.51261549	0.25091739	0.34861891	0	0.00017033	0.12882467	0.00027979	0	0	0.78368524	0.23392407	0
	Rank	8	12	7	10	**1**	6	9	5	**1**	**1**	11	13	**1**
F4	Mean	6,014,765.55	698,967.601	738.543958	1936.83868	599.878019	17.0622143	1484.94544	313.189295	336.902618	6783.32622	441.945988	5353.14974	**0.00062537**
	Std	1,591,210.31	331,443.778	51.0911465	128.585873	96.4914671	8.82795201	667.928629	123.617533	99.4030648	1101.80518	217.494639	270.969316	0.00074791
	Rank	13	12	7	9	6	2	8	3	4	11	5	10	**1**
F5	Mean	17.660534	4.33367443	1.4779E-14	0.00092076	7.4603E-52	9.77808828	9.6098E-51	5.2698E-50	2.4336E-42	0.04011403	1.8591E-44	0.00234174	**1.772E-202**
	Std	2.29644388	3.69684575	9.3147E-15	0.00051234	1.2916E-51	6.11006423	1.4456E-50	9.1274E-50	4.2152E-42	0.06947954	2.6367E-44	0.00148203	0
	Rank	13	11	7	8	2	12	3	4	6	10	5	9	**1**
F6	Mean	4255.55256	5,781,134.24	4.4402E-15	7.1888E-14	5.9111E-12	267,378.615	0.00212735	1.8295E-13	9.0271E-11	1.8521E-16	6.7049E-21	2.8741E-06	**3.78E-180**
	Std	6463.59716	5,116,260.36	4.1059E-15	1.245E-13	1.0238E-11	428,280.764	0.00368394	3.1687E-13	1.5635E-10	3.2078E-16	1.1613E-20	4.9699E-06	0
	Rank	11	13	4	5	7	12	10	6	8	3	2	9	**1**
F7	Mean	14.7609978	13.9951445	2.6522E-12	9.7764E-11	**-4.441E-16**	4.39930088	3.1086E-15	**-4.441E-16**	**-4.441E-16**	**-4.441E-16**	5.4771E-15	19.963105	**-4.441E-16**
	Std	4.57631062	9.76849546	9.5742E-13	2.5288E-11	0	0.52875926	3.5527E-15	0	0	0	2.0512E-15	0.00050852	0
	Rank	12	11	8	9	**1**	10	6	**1**	**1**	**1**	7	13	**1**
F8	Mean	3988.52033	83,583.9591	33.1518493	27.5944172	1.00164591	380.036462	57.7783078	1.00109004	**1.00036493**	122.558888	47.5264174	78.9284682	12.9586629
	Std	867.545935	143,751.731	8.01887663	2.44043882	0.00121835	14.8870137	14.5436214	0.00122459	0.00044793	25.4809702	16.6232993	10.182355	10.8528879
	Rank	12	13	6	5	3	11	8	2	**1**	10	7	9	4
F9	Mean	4.03320668	9.33320668	0.09987335	0.19987335	1.4011E-51	3.13320668	0.19987335	2.0611E-51	2.6498E-68	0.03414938	0.09987335	0.16654103	**0**
	Std	0.35118846	3.75810236	9.4461E-16	3.0578E-09	2.4141E-51	0.40414519	6.1157E-16	3.5699E-51	4.5896E-68	0.05914846	1.6661E-12	0.05773331	0
	Rank	12	13	6	10	3	11	9	4	2	5	7	8	**1**
F10	Mean	−923.62135	−981.43778	−1052.3759	−910.03601	−1174.9841	−977.0702	−1115.0561	−907.70035	−1169.4576	−642.18854	−965.25779	−634.46345	**−1174.985**
	Std	39.764393	35.9812521	57.2044489	32.1805448	0.00055477	24.4851118	101.289917	36.938001	4.61481437	47.7465831	46.9062952	27.9405838	2.9525E-08
	Rank	9	6	5	10	2	7	4	11	3	12	8	13	**1**
F11	Mean	1.1295	1.32786167	**0**	0.00668245	**0**	0.06153203	**0**	**0**	**0**	**0**	**0**	0.00723185	**0**
	Std	0.07717062	1.21567045	0	0.01157433	0	0.03793988	0	0	0	0	0	0.01252592	0
	Rank	12	13	**1**	9	**1**	11	**1**	**1**	**1**	**1**	**1**	10	**1**
F12	Mean	5.6978E-12	4.3541E-12	2.6115E-16	3.0403E-15	**−1**	1.4586E-15	−0.3333333	−0.9448373	−0.997219	−0.6666667	2.1322E-13	8.8232E-11	**−1**

Table 28 (continued)

Function	Algorithm indices	GOA[9]	MFO[10]	MPA[11]	GWO[14]	HHO[15]	SSA[16]	WOA[17]	AO[31]	IAO[33]	NOA[34]	NGO[35]	CHIO[30]	MCHIAO
	Std	4.66E-12	6.6125E-12	1.0204E-16	1.8077E-16	0	2.3544E-15	0.57735027	0.09554454	0.00481681	0.57735027	1.7319E-13	3.2917E-11	0
	Rank	12	11	7	9	**1**	8	6	4	3	5	10	13	**1**
F13	Mean	7.5586E-11	2.4741E-11	5.1662E-12	1.1254E-07	3.5133E-12	3.7896E-11	3.6318E-12	3.5202E-12	3.5124E-12	3.2553E-09	1.9404E-11	3.1781E-11	**3.5124E-12**
	Std	7.5454E-11	3.5745E-12	2.7388E-12	1.124E-07	9.6897E-16	1.1618E-11	1.9181E-13	9.3035E-15	1.72E-17	3.9533E-09	6.0649E-13	6.1453E-13	3.0844E-20
	Rank	11	8	6	13	3	10	5	4	2	12	7	9	**1**
F14	Mean	35,611.583	1852.6575	1.17908197	1.03078702	4.7193E-05	21.9676347	0.57985351	2.9718E-05	**2.8501E-06**	2.83970449	1.81505636	2.71692004	0.00366247
	Std	35,061.4431	3121.88539	0.71209151	0.13790932	3.8265E-05	23.7672253	0.19463221	2.1878E-05	2.0332E-06	0.26916693	0.35838144	0.03720067	0.00634355
	Rank	13	12	7	6	3	11	5	2	**1**	10	8	9	4
F15	Mean	53.8915612	23.981453	0.02470875	0.06134389	4.7791E-06	11.3200579	0.01887454	1.4775E-05	1.074E-06	1.17318891	0.02103351	0.44584183	**7.0298E-10**
	Std	21.1916908	15.8198561	0.00788769	0.03651005	3.5145E-06	8.26297111	0.00467991	2.4323E-05	6.2307E-07	0.23723524	0.00811188	0.0519485	4.8484E-10
	Rank	13	12	7	8	3	11	5	4	2	10	6	9	**1**
F16	Mean	−10.145755	−20.530085	−17.703361	−14.254971	−11.525431	−14.598895	−11.11753	−10.514488	−11.193229	−9.4756705	−17.810243	−7.0549159	**−22.281646**
	Std	1.28818886	1.96353846	2.50052887	6.06699593	0.95894524	2.97899958	0.46839078	2.23441885	0.55566821	0.56878696	0.86024933	0.56532887	1.72818388
	Rank	11	2	4	6	7	5	9	10	8	12	3	13	**1**
F17	Mean	4.49769367	3.94899182	0.00464546	0.00383578	**3.1836E-05**	0.47740739	0.00450945	0.00011373	0.00021529	0.04220439	0.00125668	0.0018622	3.9674E-05
	Std	0.58994918	6.11086318	0.00091909	0.00117831	2.9611E-05	0.21147013	0.00331151	7.8379E-05	9.4791E-05	0.02218998	0.00033578	0.00125016	2.1887E-05
	Rank	13	12	9	7	**1**	11	8	3	4	10	5	6	2
Percentage	10.8235294	10.5294118	6.11764706	8.47058824	2.76470588	9.17647059	5.94117647	4.52941176	3	7.35294118	5.94117647	10.2941176	1.41176471	
Total Rank	13	12	7	9	2	10	5	4	3	8	5	11	**1**	

Table 29 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for Multimodal variable-dimension functions

Function	MCHIAO vs. GOA	MCHIAO vs. MFO	MCHIAO vs. MPA	MCHIAO vs. GWO	MCHIAO vs. HHO	MCHIAO vs. SSA	MCHIAO vs. WOA	MCHIAO vs. AO	MCHIAO vs. IAO	MCHIAO vs. NOA	MCHIAO vs. NGO	MCHIAO vs. CHIO
F1	5.1707E-162	1.3826E-162	7.9706E-163	1.7416E-164	1.40562E-52	2.8324E-162	1.3735E-158	1.7163E-166	7.2886E-155	1.51026E-30	8.1294E-163	1.1233E-165
F2	2.1234E-179	3.7471E-179	9.6316E-129	3.0623E-173	2.98326E-24	1.2598E-176	8.77054E-33	1.14351E-35	0.002416993	2.52353E-43	4.73987E-09	9.3216E-178
F3	6.8802E-159	6.527E-168	3.057E-162	8.4999E-168	2.03641E-05	1.3635E-153	3.2903E-149	4.92092E-97	0.085775445	0.00352386	2.1247E-166	3.9595E-173
F4	3.7691E-160	8.3494E-162	1.1163E-109	1.2788E-119	1.05395E-88	2.79734E-77	1.6352E-105	5.3202E-105	1.0836E-103	5.48466E-86	1.13556E-80	1.7327E-158
F5	5.269E-162	6.282E-161	5.34687E-86	8.6765E-136	1.35288E-44	2.3421E-160	3.48187E-52	2.93591E-31	3.62268E-46	1.23611E-33	1.22207E-47	5.9367E-156
F6	2.8018E-161	6.2391E-163	1.89904E-85	2.9882E-90	2.1969E-99	5.4216E-162	4.2332E-156	9.4641E-100	1.0265E-104	5.87747E-28	1.74919E-74	4.1988E-156
F7	1.1613E-174	8.0962E-175	1.16459E-90	3.7494E-108	8.13963E-24	6.3322E-171	3.37783E-88	1.36299E-16	0.021566653	7.2973E-117	9.89429E-80	1.9398E-193
F8	1.2193E-160	2.4099E-163	1.9156E-107	4.401E-103	1.0425E-131	2.9069E-157	8.7769E-125	1.455E-158	2.0745E-159	1.41999E-63	5.3313E-116	1.1119E-155
F9	3.8525E-158	4.8347E-164	1.0942E-136	3.1651E-144	1.85196E-29	5.172E-159	9.2409E-144	3.37909E-14	0.002070763	4.3791E-101	5.2291E-134	3.4233E-158
F10	1.591E-160	4.1498E-159	1.4688E-158	5.2597E-163	0.577848726	6.8956E-160	1.4545E-151	2.4928E-169	1.8409E-162	2.0925E-165	1.9597E-160	2.4317E-165
F11	1.5789E-172	6.1392E-178	4.1485E-12	4.3705E-158	6.19144E-29	1.0115E-169	7.45289E-56	0.753727322	0.105982681	6.68804E-43	1.09901E-08	1.1587E-172
F12	1.2479E-163	1.453E-163	1.2384E-158	1.2997E-159	7.36901E-08	1.3856E-156	1.5955E-138	1.4132E-99	2.69939E-94	7.1167E-148	8.4535E-160	2.5558E-169
F13	1.4711E-160	6.0941E-160	5.5373E-147	7.0092E-166	0.104020119	4.8717E-159	8.3287E-109	5.4066E-19	0.003047007	5.6016E-76	7.4588E-160	1.9221E-173
F14	1.4122E-157	1.6403E-160	1.9314E-144	2.5269E-139	8.088E-113	7.7232E-158	1.1196E-126	4.5787E-165	3.2049E-174	2.8656E-39	3.115E-145	5.0208E-160
F15	2.4595E-157	1.1704E-160	8.3451E-142	2.7144E-141	0.139111707	1.8093E-157	6.9438E-129	0.630382047	0.00011522	1.45879E-50	7.4757E-139	2.5874E-159
F16	1.2163E-162	3.18458E-05	6.1109E-112	6.3652E-155	1.9462E-139	1.2578E-125	3.7623E-141	1.0435E-157	5.7319E-154	8.614E-166	1.40433E-77	1.4488E-165
F17	7.6153E-163	7.7969E-165	1.0203E-142	3.8872E-139	0.878155222	3.8872E-162	2.0329E-145	1.44024E-39	2.1453E-88	9.04712E-38	1.4525E-129	1.4782E-155

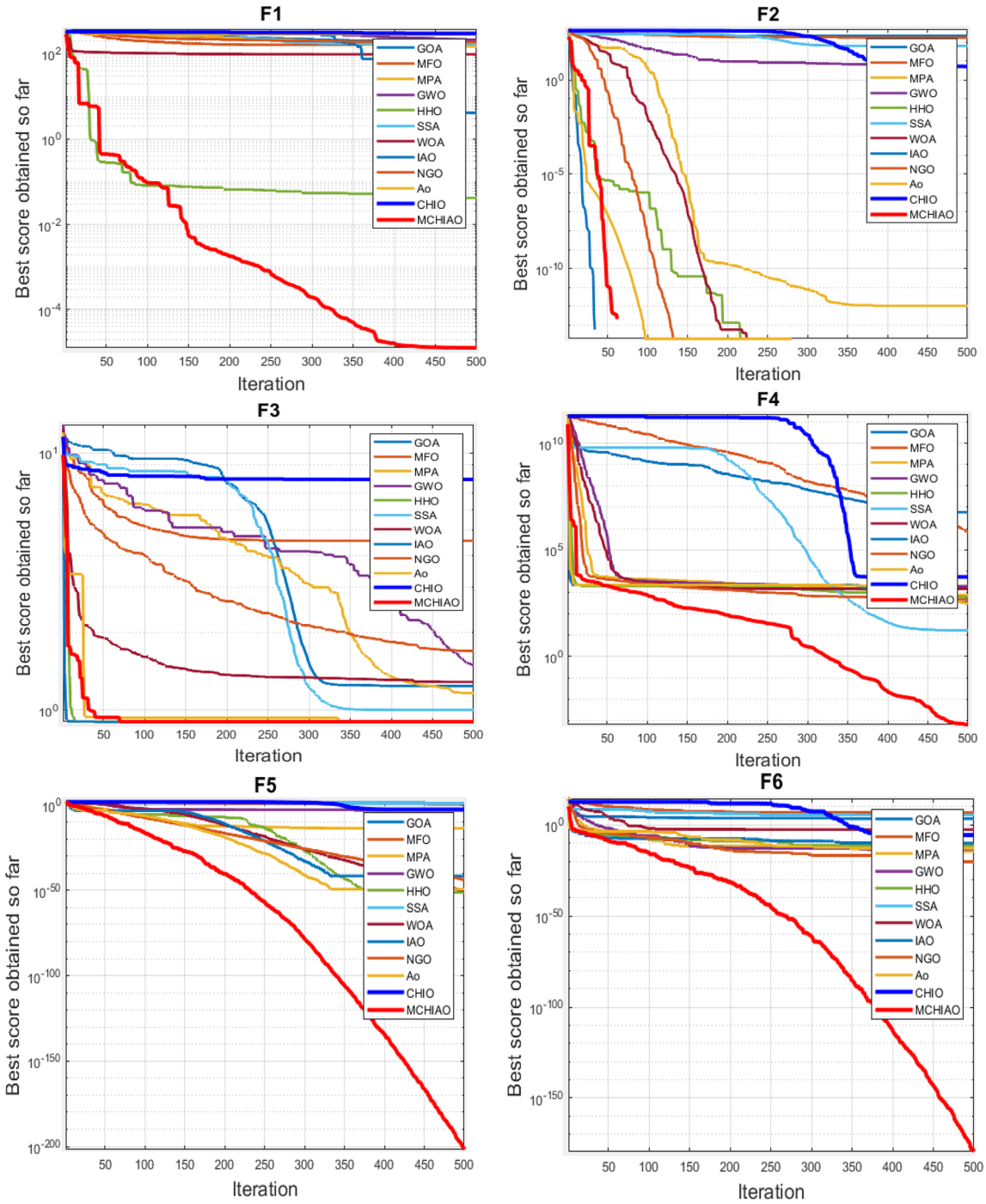


Fig. 13 The Multimodal variable-dimension functions and convergence plots

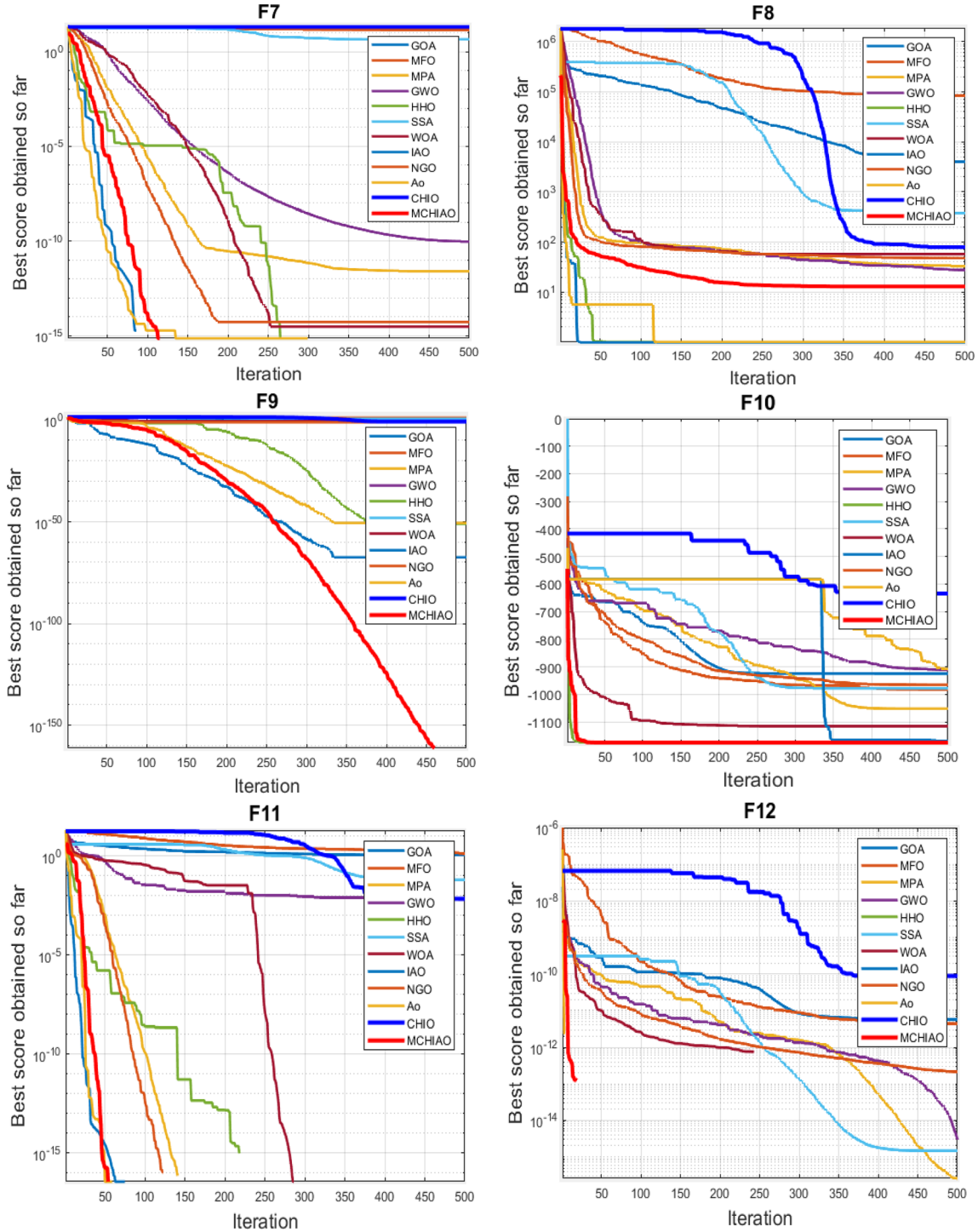


Fig. 13 continued

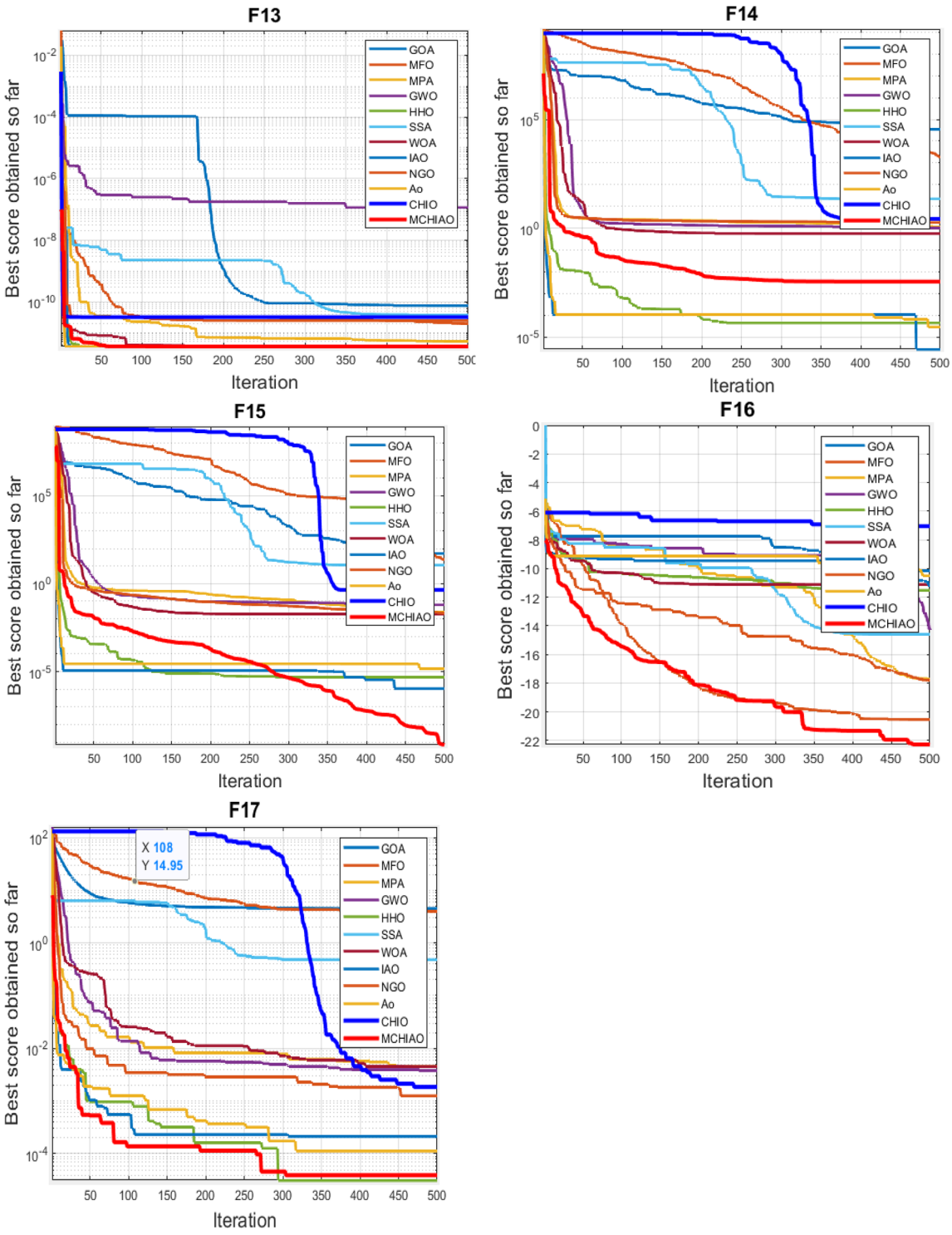


Fig. 13 continued

Fig. 14 Tension/compression spring design problem

Table 30 Tension/compression spring design results

Algorithm	Mean	Standard	Rank
GOA [9]	0.015951297	0.002906484	10
MFO [10]	0.012907016	0.000264164	4
MPA [11]	0.01281136	0.000157643	3
GWO [14]	0.012933896	0.000326614	5
HHO [15]	0.015027618	0.003059537	9
SSA [16]	0.013490251	0.001684084	7
WOA [17]	0.013942492	0.001765381	8
AO [31]	2,170,196,892	6,510,590,675	12
IAO [33]	0.019488045	0.00475156	11
NOA [34]	2.96285E + 14	3.59669E + 14	13
NGO [35]	0.012760485	0.000154712	2
CHIO [30]	0.01326582	0.000596831	6
MCHIAO	**0.012746062**	0.000166938	**1**

Table 31 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for tension/compression spring design

Algorithm	*p*-value
MCHIAO vs. GOA	1.9804E-150
MCHIAO vs. MFO	4.46507E-67
MCHIAO vs. MPA	6.09558E-35
MCHIAO vs. GWO	4.4864E-122
MCHIAO vs. HHO	1.7473E-148
MCHIAO vs. SSA	8.0538E-145
MCHIAO vs. WOA	6.4924E-144
MCHIAO vs. AO	2.2367E-159
MCHIAO vs. IAO	4.766E-158
MCHIAO vs. NOA	1.2395E-117
MCHIAO vs. NGO	3.51627E-17
MCHIAO vs. CHIO	1.3444E-135

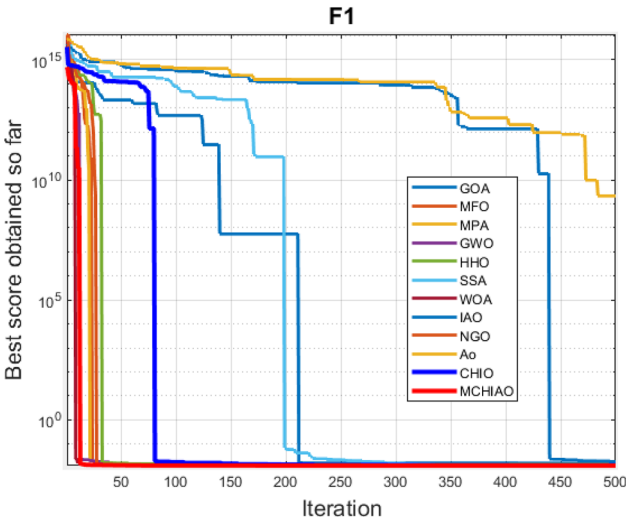


Fig. 15 Convergence curves of the 13 algorithms on the tension/compression spring design

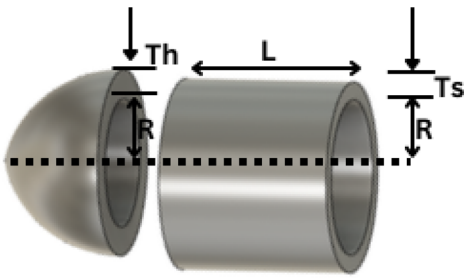


Fig. 16 Schematic of pressure vessel design

Table 32 Pressure vessel results

Algorithm	Mean	Standard	Rank
GOA [9]	35,131.10339	40,967.52571	12
MFO [10]	6497.220239	634.8879331	3
MPA [11]	7226.95381	607.0149758	6
GWO [14]	6535.891917	622.7804105	4
HHO [15]	13,847.76839	19,971.53144	10
SSA [16]	15,007.75896	11,020.65098	11
WOA [17]	12,675.26123	9480.70412	9
AO [31]	7271.549004	710.3684348	7
IAO [33]	7174.680954	711.5027682	5
NOA [34]	236,891.2726	128,237.6272	13
NGO [35]	**6096.333062**	305.6936459	**1**
CHIO [30]	8283.54537	327.2246859	8
MCHIAO	6132.290892	318.6501042	2

Table 33 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for pressure vessel problem

Algorithm	p-value
MCHIAO vs. GOA	6.326E-145
MCHIAO vs. MFO	8.10913E-79
MCHIAO vs. MPA	4.4997E-107
MCHIAO vs. GWO	3.572E-99
MCHIAO vs. HHO	6.2287E-151
MCHIAO vs. SSA	3.2334E-139
MCHIAO vs. WOA	6.0615E-132
MCHIAO vs. AO	6.0725E-133
MCHIAO vs. IAO	2.5723E-131
MCHIAO vs. NOA	2.4169E-156
MCHIAO vs. NGO	7.73373E-14
MCHIAO vs. CHIO	4.6006E-118

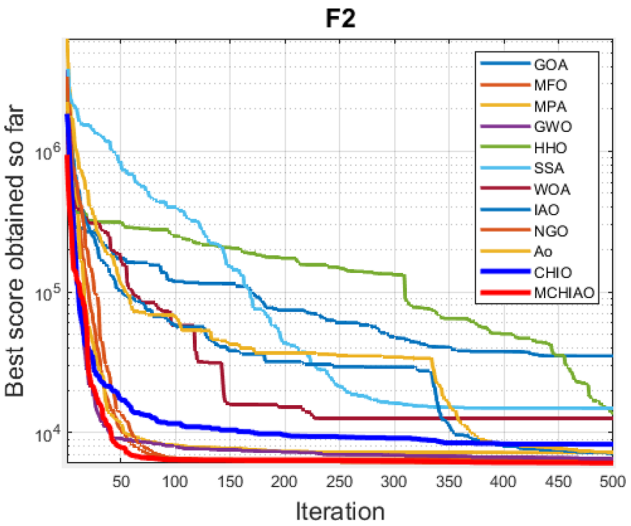


Fig. 17 Convergence curves of the 13 algorithms on the pressure vessel design problem

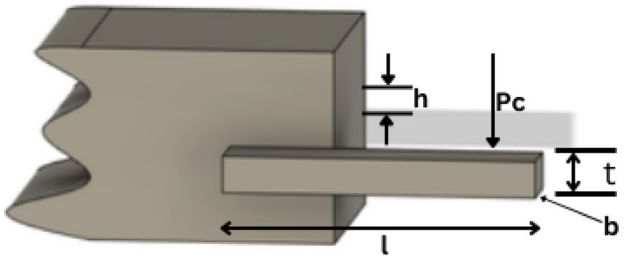


Fig. 18 Schematic of welded beam design

Table 34 Welded beam design problem results

Algorithm	Mean	Standard	Rank
GOA [9]	3.087832124	0.655528688	11
MFO [10]	2.777903465	1.013200491	10
MPA [11]	2.027376231	0.678406815	6
GWO [14]	1.734099162	0.003968062	2
HHO [15]	2.568683246	0.596323174	9
SSA [16]	2.447819513	0.516575419	8
WOA [17]	1.60077E + 14	4.80232E + 14	13
AO [31]	2.111506686	0.161034724	7
IAO [33]	2.024464177	0.115296664	5
NOA [34]	1.11692E + 13	1.90331E + 13	12
NGO [35]	1.742050269	0.034573561	3
CHIO [30]	1.843894229	0.038903465	4
MCHIAO	**1.724857129**	1.42997E-05	**1**

Table 35 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for welded beam design problem

Algorithm	p-value
MCHIAO vs. GOA	3.3461E-148
MCHIAO vs. MFO	7.9047E-144
MCHIAO vs. MPA	1.928E-131
MCHIAO vs. GWO	4.463E-84
MCHIAO vs. HHO	4.86E-149
MCHIAO vs. SSA	9.6273E-144
MCHIAO vs. WOA	9.6293E-160
MCHIAO vs. AO	1.396E-154
MCHIAO vs. IAO	6.5181E-152
MCHIAO vs. NOA	6.064E-161
MCHIAO vs. NGO	6.1375E-102
MCHIAO vs. CHIO	2.1799E-123

6 Conclusions

In this study, Modified Coronavirus Herd Immunity Aquila Optimizer (MCHIAO) is an improved hybrid algorithm proposed as a contribution to a novel nature-inspired human-based metaheuristic optimization algorithm (CHIO) with Aquila optimization algorithm for global optimization

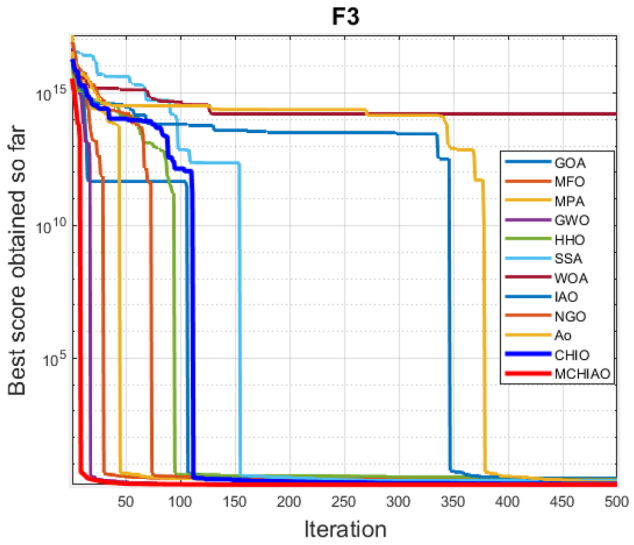


Fig. 19 Convergence curves of the 13 algorithms on the welded beam design problem

Table 36 Speed reducer problem results

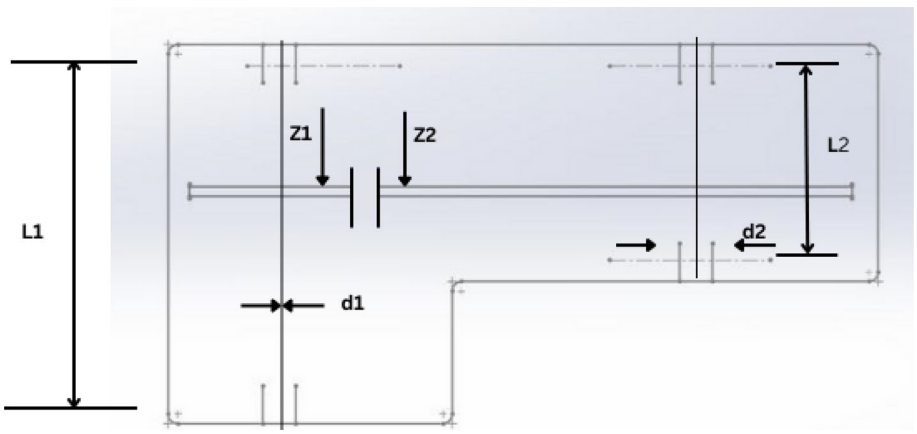
Algorithm	Mean	Standard	Rank
GOA [9]	1.07722E + 15	3.11439E + 15	13
MFO [10]	762.8288311	13.95755429	2
MPA [11]	763.9785059	15.86450865	3
GWO [14]	788.8860584	19.96261583	5
HHO [15]	881.8736378	73.09210583	8
SSA [16]	910.4944773	48.64117481	10
WOA [17]	905.7236558	138.6739242	9
AO [31]	843.163184	33.33856705	7
IAO [33]	820.8015923	38.20063308	6
NOA [34]	1.53481E + 14	1.05845E + 14	12
NGO [35]	765.4001109	18.85723823	4
CHIO [30]	954.3313325	172.5583058	11
MCHIAO	**757.5016403**	17.82148259	**1**

problems. The proposed algorithm presents three main modifications that lead to better outcomes. The first modification is applying an improved case categorizing technique that implements a balance between the exploitation and exploration stages. The second enhancement is improving the equation of the new genes' value which leads to more optimal values. Additionally, as many studies show that the COVID-19 pandemic possesses chaotic system characteristics, we applied the chaotic system in the case categorizing phase which leads to avoiding local minima and rapid convergence rate. The third enhancement is generating a new formula for the selection between the expanded exploitation and narrowed exploitation phases. The results reveal that the proposed MCHIAO can find optimal solutions and converge faster on most issues due to its strong exploration and exploitation capabilities. The proposed MCHIAO's effectiveness is evaluated using Wilcoxon statistical analyses and Friedman average rank through 23 well-known benchmark functions of various

Table 37 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for speed reducer problem

Algorithm	p-value
MCHIAO vs. GOA	7.1841E-160
MCHIAO vs. MFO	7.71414E-22
MCHIAO vs. MPA	2.7588E-101
MCHIAO vs. GWO	1.6344E-126
MCHIAO vs. HHO	1.7953E-139
MCHIAO vs. SSA	1.0121E-137
MCHIAO vs. WOA	1.4754E-129
MCHIAO vs. AO	3.8149E-148
MCHIAO vs. IAO	8.3013E-147
MCHIAO vs. NOA	1.1145E-154
MCHIAO vs. NGO	4.15843E-89
MCHIAO vs. CHIO	2.7345E-145

Fig. 20 Speed reducer design



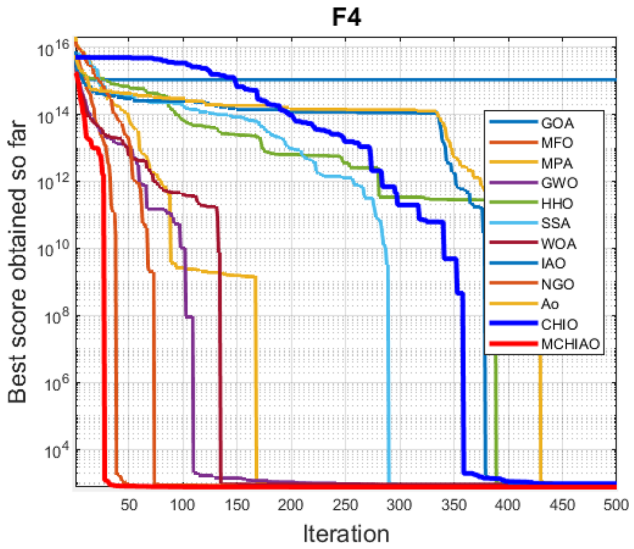


Fig. 21 Convergence curves of the 13 algorithms on the speed reducer problem

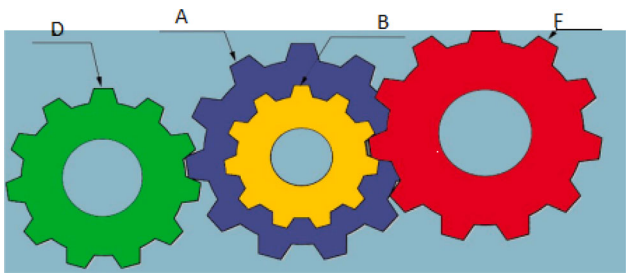


Fig. 22 Gear train design

Table 38 Gear train design results

Algorithm	Mean	Standard	Rank
GOA [9]	6.21185E-18	7.44915E-18	7
MFO [10]	**0**	0	**1**
MPA [11]	4.24881E-20	8.73172E-20	5
GWO [14]	9.86456E-12	1.37344E-11	9
HHO [15]	**0**	0	**1**
SSA [16]	3.50623E-19	5.12915E-19	6
WOA [17]	7.68122E-25	2.76945E-24	4
AO [31]	1.41497E-10	2.27371E-10	11
IAO [33]	2.38202E-10	5.74132E-10	12
NOA [34]	6.07034E-07	1.34258E-06	13
NGO [35]	1.16393E-14	2.33049E-14	8
CHIO [30]	6.37017E-11	1.05369E-10	10
MCHIAO	**0**	0	**1**

Table 39 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for gear train design problem

Algorithm	p-value
MCHIAO vs. GOA	2.155E-119
MCHIAO vs. MFO	0.005515138
MCHIAO vs. MPA	8.75194E-93
MCHIAO vs. GWO	5.4314E-139
MCHIAO vs. HHO	1.2376E-47
MCHIAO vs. SSA	1.0991E-119
MCHIAO vs. WOA	2.3808E-70
MCHIAO vs. AO	3.3244E-153
MCHIAO vs. IAO	4.7013E-150
MCHIAO vs. NOA	1.5031E-159
MCHIAO vs. NGO	2.3313E-106
MCHIAO vs. CHIO	3.8584E-142

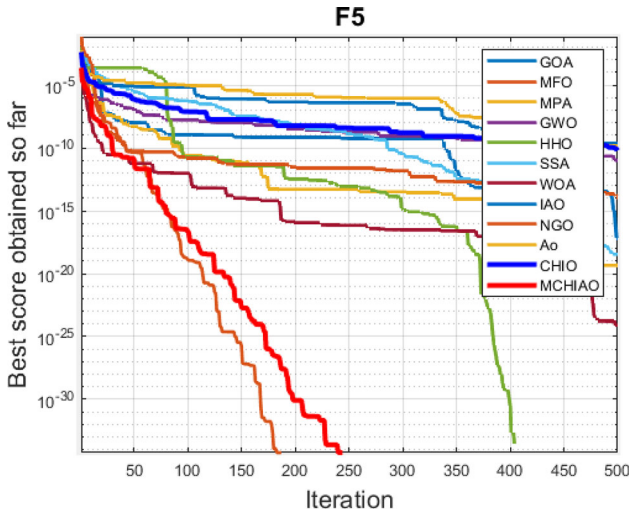


Fig. 23 Convergence curves of the 13 algorithms on the gear train design problem

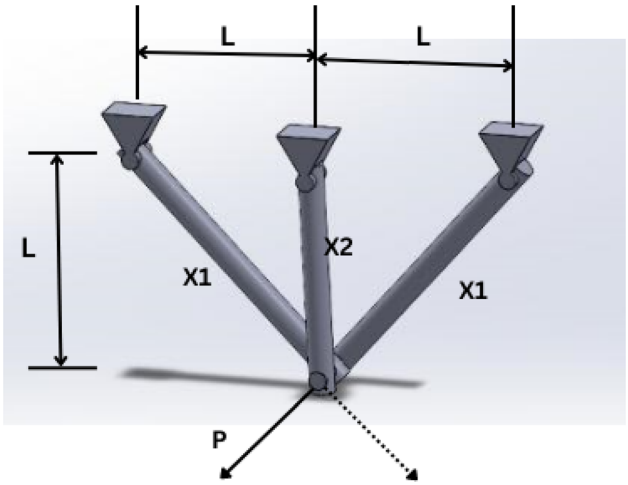


Fig. 24 Three-bar truss design

Table 40 Three-bar truss design results

Algorithm	Mean	Standard	Rank
GOA [9]	266.9145286	4.384192239	13
MFO [10]	264.0882605	0.381636882	7
MPA [11]	263.9926233	0.13055913	5
GWO [14]	263.9138724	0.010425992	4
HHO [15]	264.0204774	0.157444236	6
SSA [16]	263.9078842	0.012938126	3
WOA [17]	265.587063	2.302897999	10
AO [31]	266.4906845	2.046607548	12
IAO [33]	266.2227394	2.092995513	11
NOA [34]	265.0384931	0.630322305	8
NGO [35]	263.8977075	0.002511928	2
CHIO [30]	265.5120281	5.208503965	9
MCHIAO	**263.8958434**	1.23887E-13	**1**

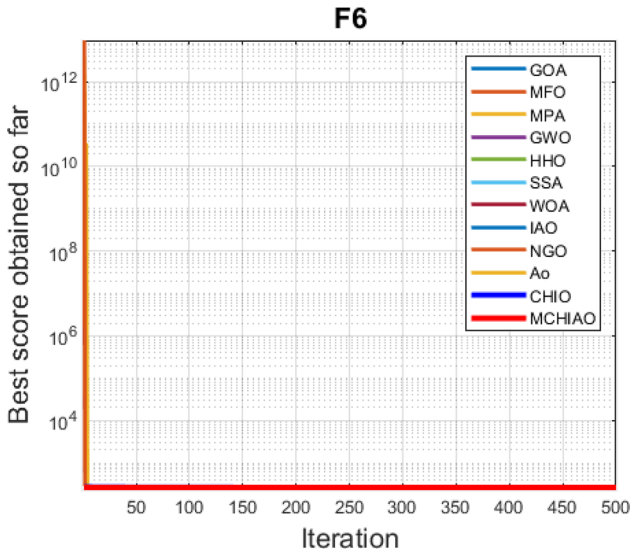


Fig. 25 Three-bar truss design plots

Table 41 The Wilcoxon rank-sum test (p-Value) for the MCHIAO algorithm against all other 12 algorithms for the Three-bar truss design problem

Algorithm	p-value
MCHIAO vs. GOA	1.7096E-162
MCHIAO vs. MFO	3.3632E-149
MCHIAO vs. MPA	3.8239E-155
MCHIAO vs. GWO	2.8163E-149
MCHIAO vs. HHO	5.411E-153
MCHIAO vs. SSA	2.4051E-140
MCHIAO vs. WOA	1.2056E-160
MCHIAO vs. AO	2.0848E-164
MCHIAO vs. IAO	1.1648E-168
MCHIAO vs. NOA	5.0247E-165
MCHIAO vs. NGO	1.8896E-117
MCHIAO vs. CHIO	2.8027E-161

dimensions and complexity: seven unimodal, six multimodal with adjustable dimensions, and 10 multimodal with fixed dimensions. These functions have been widely used to evaluate newly proposed optimization methods in the literature. The MCHIAO is also compared to the performance of other well-known and most recent optimization techniques. The algorithms used in the comparison are GWO, GOA, MFO, MPA, HHO, SSA, WOA, IAO, NOA, NGO, AO, and the original CHIO. The comparative analysis reveals that MCHIAO is conspicuously competitive since it can acquire 21 out of the 23 benchmark functions. Further, to prove the efficiency of the proposed MCHIAO algorithm, both CEC 2017 and CEC 2019 benchmark functions are tested in which MCHIAO achieves the overall best results in 17 out of 29 CEC 2017 and 4 out of 10 CEC19 benchmark functions. The exploitative and explorative behavior of the hybrid algorithm MCHIAO is assessed on a variety of classical functions, including 24 unimodal and 44 multimodal functions, respectively. The proposed MCHIAO ranked top in all 15 functions for unimodal variable-dimension issues and first in 7 out of 8 functions for unimodal fixed-dimension problems. On the other hand, MCHIAO succeeds in 24 out of 27 functions for the multimodal fixed-dimension challenges and is placed top in 14 out of 17 functions for the multimodal variable-dimension problems. As well, the proposed MCHIAO defeats the competitive optimization algorithms in five out of six real-world application problems gear train design problem, speed reducer problem, welded beam design problem, pressure vessel design problem, and welded beam design problem and ranked second in the pressure vessel problem. MCHIAO has several limitations, much like any other suggested algorithm. First, the sensitivity to parameter issue in the original CHIO and AO has been resolved by MCHIAO. Results with the mentioned issues are encouraging. This might not work, though, for other issues. Second, convergence time grows with complexity, providing a new avenue for future study to address this problem. Finally, new algorithms with improved performance may be created as this field of study continues to advance.

Appendix

A glossary of acronyms and abbreviations is listed in Table 42.

Table 42 Acronyms list

Abbreviations	Definition
A_j	Age Vector
AO	Aquila Optimizer
BR_r	Basic Reproduction rate
CHIO	Coronavirus Herd Immunity Optimizer
COVID-19	Coronavirus Disease 2019
d	wire diameter
D	mean coil diameter
Dim	Dimension size of problem
ECHIO	Enhanced Coronavirus Herd Immunity Optimizer
Eq	Equation
F	Fixed dimension
GOA	Grasshopper Optimization Algorithm
GWO	Grey Wolf Optimizer
h	the thickness of the weld
HHO	Harris Hawks Optimization
HIP	Herd Immunity Population
HIS	Herd Immunity Size
IAO	Improved Aquila Optimizer
L	length of the cylindrical section neglecting head
Lb	Lower Bound
m	Chaotic parameter
M	Multimodal function
MCHIAO	Modified Coronavirus Herd Immunity Aquila Optimizer
MFO	Moth-flame Optimization Algorithm
MPA	Marine Predators Algorithm
n_j	the number of gearwheel teeth j
N	Number of population size
NFL	No Free Lunch
NGO	Northern Goshawk Optimization
NOA	Nutcracker Optimization Algorithm
P	number of active coils
Pc	bar's buckling load
QF	Quality Function
r	Random parameter
R	Inner radius
rand	Random value
S_j	Status Vector
SIR	Susceptible Infectious Recovered
SSA	Salp Swarm Algorithm
Std	Standard deviation
t	current iteration
T	maximum number of iterations
T_h	head Thickness
T_s	shell Thickness
U	Unimodal functions
ub	Upper Bound
WOA	Whale Optimization Algorithm
δ	beam end deflection
θ	Bending stress
τ	Shear stress

Author contributions HS, AYH, LML, and MMS contributed to the design and implementation of the research and, the analysis of the results. All the authors have participated in writing the manuscript and have revised the final version. All authors read and approved the final manuscript.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). Open access funding is provided by The Science, Technology, and Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). No funding was received for this work.

Data availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Declaration

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. There are no conflicts of interest. I would like to confirm that there were no known conflicts of interest associated with this publication and that there was no significant financial support for this work that could affect its outcome.

References

1. Shabani A, Asgarian B, Gharebaghi SA, Salido MA, Giret A (2019) A new optimization algorithm based on search and rescue operations. Math Probl Eng 2019:1–23

2. Yang X (2021) Nature-inspired optimization algorithms: second edition, Mara Conner.
3. Wang FS, Chen LH (2013) Heuristic Optimization. In: Dubitzky W, Wolkenhauer O, Cho KH, Yokota H (eds) Encyclopedia of Systems Biology. Springer, New York, NY. https://doi.org/10.1007/978-1-4419-9863-7_411.
4. Osman IH, Laporte G (1996) Metaheuristics: A bibliography. Ann Ope Res 63(5):511–623
5. Fausto F, Reyna-Orta A, Cuevas E, Andrade AG, Perez-Cisneros M (2019) From ants to whales: metaheuristics for all tastes. Artif Intell Rev 1–58
6. Saafan MM, El-Gendy EM (2021) IWOSSA: An improved whale optimization salp swarm algorithm for solving optimization problems. Exp Syst Appl 176: 114901. ISSN 0957–4174, https://doi.org/10.1016/j.eswa.2021.114901.
7. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evolut Comput 1(1):67–82.
8. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks 4: 1942–1948. IEEE.
9. Saremi S, Mirjalili S, Lewis A (2017) Grasshopper optimisation algorithm: theory and application. Adv Eng Softw 105:30–47
10. Mirjalili S (2015) Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl-Based Syst 89.
11. Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH (2020) Marine predators algorithm: a nature-inspired metaheuristic. Exp Syst Appl 152: 113377, ISSN 0957–4174.
12. Humphries NE, Queiroz N, Dyer JRM, Pade NG, Musyl MK, Schaefer KM, Fuller DW, Brunnschweiler JM, Doyle TK, Houghton JDR, Hays GC, Jones CS, Noble LR, Wearmouth VJ, Southall EJ, Sims DW (2010) Environmental context explains Lévy and Brownian movement patterns of marine predators. Nature 465(7301):1066
13. Bartumeus F, Catalan J, Fulco UL, Lyra ML, Viswanathan GM (2002) Optimizing the encounter rate in biological interactions: Lévy versus Brownian strategies. Phys Rev Lett 88(9):097901
14. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. J Adv Eng Softw 69:46–61
15. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gener Comput Syst 97:849–872
16. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
17. Seyedali M, Andrew L (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
18. Balaha HM, El-Gendy EM, Saafan MM (2021) Covh2sd: A covid-19 detection approach based on harris hawks optimization and stacked deep learning. Expert Syst Appl 186:115805
19. Balaha HM, Saafan MM (2021) Automatic exam correction framework (aecf) for the mcqs, essays, and equations matching. IEEE Access 9:32368–33238
20. Balaha HM, El-Gendy EM, Saafan MM (2022) A complete framework for accurate recognition and prognosis of covid-19 patients based on deep transfer learning and feature classification approach. Artif Intell Rev 1–46.
21. Balaha HM, Shaban AO, El-Gendy EM, Saafan MM (2022) A multi-variate heart disease optimization and recognition framework. Neural Comput Appl 1–38.
22. Fahmy H, El-Gendy EM, Mohamed MA, Saafan MM (2023) ECH3OA: An Enhanced Chimp-Harris Hawks Optimization Algorithm for copyright protection in Color Images using watermarking techniques. Knowl-Based Syst 269: 110494, ISSN 0950–7051. https://doi.org/10.1016/j.knosys.2023.110494.
23. Badr AA, Saafan MM, Abdelsalam MM et al (2023) Novel variants of grasshopper optimization algorithm to solve numerical problems and demand side management in smart grids. Artif Intell Rev. https://doi.org/10.1007/s10462-023-10431-5
24. Balaha MM, El-Kady S, Balaha HM et al (2023) A vision-based deep learning approach for independent-users Arabic sign language interpretation. Multimed Tools Appl 82:6807–6826. https://doi.org/10.1007/s11042-022-13423-9
25. Balaha HM, Antar ER, Saafan MM et al (2023) A comprehensive framework towards segmenting and classifying breast cancer patients using deep learning and Aquila optimizer. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-023-04600-1
26. Kwok KO, Lai F, Wei WI, Wong SYS, Tang JW (2020) Herd immunity–estimating the level required to halt the covid-19 epidemics in affected countries. J Infect 80(6):e32–e33
27. Tzanetos A, Dounias G (2021) Nature inspired optimization algorithms or simply variations of metaheuristics? Artif Intell Rev 54:1841–1862. https://doi.org/10.1007/s10462-020-09893-8
28. Xia Y, Zhong L, Tan J, Zhang Z, Lyu J, Chen Y, Zhao A, Huang L, Long Z, Liu NN, Wang H (2020) How to understand "Herd Immunity" in COVID-19 Pandemic. Front Cell Dev Biol 8:547314
29. Zhuang L, Cressie N (2014) Bayesian hierarchical statistical SIRS models. Stat Methods Appl 23:601–646. https://doi.org/10.1007/s10260-014-0280-9
30. Al-Betar MA, Alyasseri ZAA, Awadallah MA et al (2021) Coronavirus herd immunity optimizer (CHIO). Neural Comput & Applic 33:5011–5042. https://doi.org/10.1007/s00521-020-05296-6
31. Abualigah L, Yousri D, Abd Elaziz ME, Ewees AA, Al-qaness MAA, Gandomi AH (2021) Aquila optimizer: A novel meta-heuristic optimization algorithm. Comput Ind Eng 157: 107250, ISSN 0360–8352, https://doi.org/10.1016/j.cie.2021.107250.
32. Salawudeen AT, Mu'azu MB, Sha'aban YA, Adedokun AE (2021) A novel smell agent optimization (SAO): An extensive CEC study and engineering application. Knowl-Based Syst. 232: 107486 (2021).
33. Ewees AA, Algamal ZY, Abualigah L, Al-qaness MAA, Yousri D, Ghoniem RM, AbdElaziz M (2022) Cox proportional-hazardsmodel based on improved aquilaoptimizer with whale optimizationalgorithm operators. Mathematics 10:1273. https://doi.org/10.3390/math10081273
34. Abdel-Basset M,Mohamed R, Jameel M, Abouhawwash M (2023) Nutcracker optimizer: A novel nature-inspired meta-heuristic algorithm for global optimization and engineering design problems. Knowl-Based Syst, 262: 110248, ISSN 0950–7051, https://doi.org/10.1016/j.knosys.2022.110248.
35. Dehghani M, Hubalovsky S, Trojovsky P (2021) Northern goshawk optimization: a new swarm-based algorithm for solving optimization problems. IEEE Access. 1–1. https://doi.org/10.1109/ACCESS.2021.3133286.
36. Recioui A (2018) Application of teaching learning-based optimization to the optimal placement of phasor measurement units. Book chapter in: Handbook of Research on Emergent Applications of Optimization Algorithms, IGI Global.
37. Recioui A (2021) Home load-side management in smart grids using global optimization. Book chapter in: Research Anthology on Multi-Industry Uses of Genetic Programming and Algorithms, IGI Global.
38. El-Sherbiny A, Elhosseini MA, Haikal AY (2018) A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm. Appl Soft Comput 73:24–38. https://doi.org/10.1016/j.asoc.2018.08.028
39. Guedria NB (2016) Improved accelerated PSO algorithm for mechanical engineering optimization problems. Appl Soft Comput 40: 455–467, ISSN 1568–4946, https://doi.org/10.1016/j.asoc.2015.10.048.