MATHEMATICS AND COMPUTATIONAL SCIENCES

Vol 6(1), 2025, pp:44-66

DOI:10.30511/mcs.2024.2041661.1232

ISSN:2717-2708

http://mcs.qut.ac.ir/

Qom University of Technology

# Competition and cooperation in evading threat meta-heuristic algorithm

**Mostafa Falahaty Nejad** [1], **Majid Eshaghi Gordji** [2]

[1,2]Department of Mathematics, Semnan University, Semnan, Iran

**Abstract:** This work introduces an innovative heuristic algorithm named "Competition and Collaboration in Evading Threat (CCET)". Inspired by the escape behavior of animals such as deer, buffalo, sheep, etc., from predators like lions, leopards, tigers, etc., and also drawing parallels with soldiers evading attacks in war zones involving missiles, cannons, tanks, enemy gunfire, etc., the algorithm has been devised. In this approach, it is assumed that soldiers in war zones or domesticated animals are fleeing from threats and, despite competing in their escape, they collaborate with each other to ensure their survival. Unlike existing heuristic algorithms that rely on convergence, this proposed algorithm focuses on a novel approach based on the concept of divergence. The optimal response is determined based on the divergence of prey from the threat of the predator. The algorithm undergœs testing on 23 well-known benchmark functions, including unimodal, multimodal, and fixed-dimensional functions. The performance of the proposed algorithm is validated against recognized heuristic algorithms. Comparative results indicate that the proposed algorithm significantly demonstrates the capability to compete with well-known and powerful algorithms.

**Keywords:** Meta-heuristic algorithm; Optimization; Competition and Cooperation ; Divergence; Matlab
**2020 Mathematics Subject Classification:** 90Cxx
**Receive:** 21 September 2024, **Accepted:** 13 December 2024

## 1 Introduction

Algorithmic techniques are continually expanding, operating solely based on input and output results. Hence, to optimize nonlinear and complex problems with high dimensions across various fields such as computer science, electrical engineering, civil engineering, and other applied sciences, metaheuristic algorithms are employed. These algorithms intelligently approach solutions randomly, eliminating the need for gradients and derivatives, unlike traditional methods. Generally, they possess the capability to escape local optima and converge toward the global optimum. Given the diverse nature of problems and variations in algorithm performance, the discovery of new algorithms and improvements to existing ones are necessary for more accurate and faster problem-solving.

Metaheuristic search algorithms operate in two stages: exploration and exploitation. Exploration aims to escape local optima, while exploitation, conducted around the solution point, aims for a more precise response. Balancing exploration and exploitation is crucial in metaheuristic algorithms. Combining two or more metaheuristic algorithms or adapting them with slight modifications proves useful in both discrete and continuous problem domains.

Numerous metaheuristic algorithms exist, categorized into five groups. The first group consists of algorithms based on evolution, mimicking processes emulating the behaviors of living organisms. They leverage biological

[1]Corresponding author: mfalahatynejad@semnan.ac.ir
[2]meshaghi@semnan.ac.ir

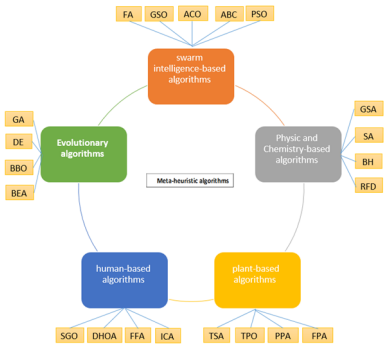


Figure 1: Types of meta-heuristic algorithms.

evolution processes such as reproduction, mutation, and combination. Genetic Algorithm (GA) [17], among evolutionary algorithms, employs techniques like inheritance, mutation, and Darwinian selection principles. Bacterial Foraging Optimization Algorithm (BEA) [8] is inspired by microbiological phenomena, involving bacterial mutation and gene transfer operations. Biogeography-Based Optimization (BBO) [48] is grounded in the geographical distribution of species. The Differential Evolution (DE) [37] algorithm draws inspiration from collaborative and competitive processes in a biological population to address the primary drawback of Genetic Algorithms.

The second category encompasses swarm intelligence-based metaheuristic algorithms, drawing inspiration from collective behavior in decentralized, self-organized systems. The PSO [20] algorithm, inspired by bird flock social behavior, was initially employed to uncover patterns in simultaneous flight and sudden trajectory changes. Another algorithm is the artificial bee colony (ABC) [18], simulating honeybee colony behavior since 2005. The GSO [16] algorithm, rooted in population-inspired search behavior, is derived from animal foraging behavior. The FA [50] algorithm mimics artificial glowworm behavior, assuming a sexual inclination among glowworms. The CSO [40] or Cuckoo Search algorithm, introduced in 2009, draws from the brood parasitism behavior of certain cuckoos. The ACO [11] algorithm, inspired by ant colonies, emphasizes the survival of the entire colony. The BA [52] algorithm, inspired by bats, balances exploration and exploitation during the search process.

The third group comprises metaheuristic algorithms based on physics and chemistry, including the Gravitational Search Algorithm (GSA) [41], inspired by gravitational attraction and Newtonian motion. Other algorithms in this category include SA [23], RFD [39], and BH [15].

The fourth group involves social human-inspired metaheuristic algorithms, considering country relationships based on colonization, culture, economy, and political relations. Examples include ICA [5], FFA [46], DHOA [6], and SGO [38].

The fifth group of algorithms is rooted in plant behavior, with noteworthy examples like FPA [53], PPA [49], TPO [14], and TSA [22]. Figure 1 illustrates various metaheuristic algorithms, and the table provides examples of different types of metaheuristic algorithms.

# 2 Competition and cooperation in Evading Threat (CCET)

## 2.1 Inspiration

The text discusses the inspiration behind the proposed algorithm initially, followed by the introduction of the corresponding mathematical model. The instinct to evade threats is inherent in all living beings. Animals such as deer, gazelle, buffalo, groundhog, and others instinctively flee when faced with predators like lions, leopards, tigers,

Table 1: Examples of meta-heuristic algorithms

| N | Algorithm | Acronym | type | Reference |
|---|---|---|---|---|
| 1 | Artifcial Ecosystem Algorithm | A EA | Evolutionary | [2] |
| 2 | Bean Optimization Algorithm | BOA | Evolutionary | [56] |
| 3 | Clonal Selection Algorithm | CSA | Evolutionary | [9] |
| 4 | Genetic Programming | GP | Evolutionary | [24] |
| 5 | Coronavirus Optimization Algorithm | COVIDOA | Evolutionary | [21] |
| 6 | Self-Organizing Migrating Algorithm | SOMA | Evolutionary | [55] |
| 7 | African Buffalo Optimizatio | ABO | swarm intelligence | [34] |
| 8 | African Vultures Optimization Algorithm | AVOA | swarm intelligence | [1] |
| 9 | Ant Lion Optimizer | ALO | swarm intelligence | [30] |
| 10 | Artifcial Cooperative Search | ACS | swarm intelligence | [7] |
| 11 | Bird Swarm Algorithm | BSA | swarm intelligence | [26] |
| 12 | Bumble Bees Mating Optimization | BBMO | swarm intelligence | [25] |
| 13 | Chaotic DragonflyAlgorithm | CDA | swarm intelligence | [42] |
| 14 | Hunting Search | HuS | swarm intelligence | [35] |
| 15 | Monkey Search | MS | swarm intelligence | [33] |
| 16 | Sail Fish Optimizer | SFO | swarm intelligence | [44] |
| 17 | Artifcial Atom | A3 | Physics/chemistry | [19] |
| 18 | Big Bang-Big Crunch | BBBC | Physics/chemistry | [12] |
| 19 | Intelligent Water Drops | IWD | Physics/chemistry | [45] |
| 20 | Water Cycle Algorithm | WCA | Physics/chemistry | [13] |
| 21 | Water Wave Optimization | WWO | Physics/chemistry | [57] |
| 22 | Brain Storm Optimization | BSO | social human | [47] |
| 23 | Competitive Learning Algorithm | CLA | social human | [3] |
| 24 | Cultural Algorithm | CA | social human | [43] |
| 25 | Coronavirus Herd Immunity Optimizer | CHIO | social human | [4] |
| 26 | Golden Ball Algorithm | GB | social human | [36] |
| 27 | Flower Pollination Algorithm | FPA | plant | [53] |
| 28 | Plant Propagation Algorithm | PPA | plant | [49] |
| 29 | Tree Physiology Optimization | TPO | plant | [14] |
| 30 | Tree Seed Algorithm | TSA | plant | [22] |

Figure 2: Avoiding economic crises.

Figure 3: Escape of soldiers in war.

and similar threats. Similarly, in times of war, soldiers threatened by enemy snipers, missiles, fighter jets, landmines, etc., take evasive actions to avoid harm. Economic enterprises, confronting crises like currency fluctuations, legal instability, inflation, recession, lack of public interest, increased wages, and more, employ survival-oriented measures to navigate through the challenges. Even human-made artificial creations are designed to escape threats. For example, military drones are programmed to evade ground-to-air missiles, anti-aircraft systems, fighter jets, adverse atmospheric conditions, and other potential dangers. Existing metaheuristic algorithms typically operate based on convergence, but the proposed algorithm introduces a novel approach centered around the concept of divergence, placing a particular emphasis on escaping threats.

Figure 4: Escape the hunt from the lion.

Figure 5: Escape the drone from the threat.

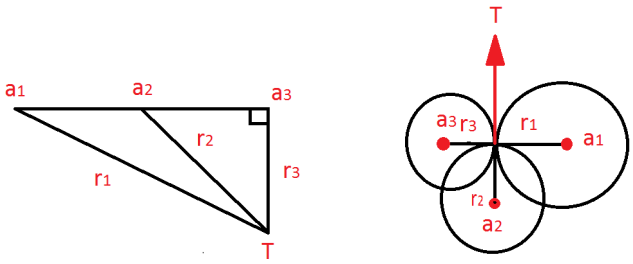


Figure 6: position of soldiers and threat.

## 2.2 Mathematical model and algorithm

In the mathematical model, soldiers, drones, animals (buffalos, deer, etc.), and economic institutions are conceptualized as soldiers facing threats, encompassing missiles, fighter jets, predatory animals, and economic crises. $N$ candidate positions are randomly generated, and the suitability for each is computed. The top three candidates are designated as the primary three soldiers, assuming that the remaining soldiers, apart from these three, have been captured under the threat. The schematic depiction of the threat location and the remaining three soldiers can take two forms, as follows:
$a_1$, $a_2$, and $a_3$ denote the optimal responses in sequence among the first n initial responses. In the context of the provided figures, T signifies the threat location, while $a_1$, $a_2$, and $a_3$ represent the positions of the soldiers. It is expected that $r_1 > r_2 > r_3$ since the best response should have the maximum escape distance from the threat. Adhering to the surprise principle, the threat location is considered unknown and dynamic, relying solely on the soldiers' distance ($r_i$). Assuming that, in the preceding stage, each soldier had the option to escape in four directions, the four presumed escape directions for each soldier are as follows:

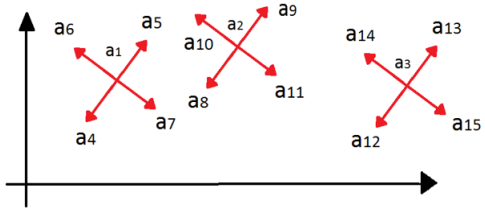


Figure 7: Quadruple escape locations for three soldiers.

- $a_4$ to $a_7$ represent escape positions for soldier $a_1$.
- $a_8$ to $a_{11}$ represent escape positions for soldier $a_2$.
- $a_{12}$ to $a_{15}$ represent escape positions for soldier $a_3$.

## 2.3 Application of internal multiplication in mathematical modeling

If vectors $a = (a_x, a_y)$ and $b = (b_x, b_y)$ are expressed in Cartesian coordinates, the dot product is written as follows:

$$a \cdot b = (a_x \times b_x) + (a_y \times b_y) \tag{2.1}$$

If $a \cdot b = 0$, then vectors $a$ and $b$ are orthogonal. For this to happen, $b_x = -a_y$ and $b_y = a_x$ must hold true.
Now, two $n$-dimensional vectors are considered as follows:

$$X = (x_1, x_2, x_3, \ldots, x_n) \text{ and } Y = (y_1, y_2, y_3, \ldots, y_n) \tag{2.2}$$

**Assuming:** $y_1 = -x_2$ and $y_2 = x_1$, then the dot product of two vectors in two dimensions becomes zero. Given this, if both vectors are extended to n dimensions, the images of one vector onto the other must be at least orthogonal in two dimensions.
Therefore, the generation of $(a_4, \ldots, a_{15})$ is as follows:

$$a_i = (v_1, v_2, \ldots, v_n), \qquad A_{a_i} = (-v_2, v_1, \ldots, v_n) \tag{2.3}$$
$$a_4 = a_1 + h_1(a_1) \tag{2.4}$$
$$a_5 = a_1 + h_2(a_1) \tag{2.5}$$
$$a_6 = a_1 + c_1\,(A_{a_1}) \tag{2.6}$$
$$a_7 = a_1 + c_2\,(A_{a_1}) \tag{2.7}$$
$$a_8 = a_2 + h_1(a_2) \tag{2.8}$$
$$a_9 = a_2 + h_2(a_2) \tag{2.9}$$
$$a_{10} = a_2 + c_1\,(A_{a_2}) \tag{2.10}$$
$$a_{11} = a_2 + c_2\,(A_{a_2}) \tag{2.11}$$
$$a_{12} = a_3 + h_1(a_3) \tag{2.12}$$
$$a_{13} = a_3 + h_2(a_3) \tag{2.13}$$
$$a_{14} = a_3 + c_1\,(A_{a_3}) \tag{2.14}$$
$$a_{15} = a_3 + c_2\,(A_{a_3}) \tag{2.15}$$

that in:

$$h_1 \in [-1, 0) \tag{2.16}$$
$$h_2 \in (0, 1] \tag{2.17}$$
$$c_1 \in [-1, 0) \tag{2.18}$$
$$c_2 \in (0, 1] \tag{2.19}$$

Each soldier, in addition to their current position, generates four new positions, leading to the following scenarios:

1. The soldier's position and the generated four positions collectively include one position superior to the existing 15 positions.

2. One soldier's position and the generated four positions include two positions superior to the existing 15 positions, while the positions of two other soldiers, along with their generated positions, collectively include one superior position.
3. One soldier's position and the generated four positions include three positions superior to the existing 15 positions, while the positions of two other soldiers, along with their generated positions, collectively include zero superior positions.

The examination of these three cases aims to assess the impact of collaboration and non-collaboration among the three soldiers as they compete with each other, enhancing the efficiency of the proposed algorithm. Collaboration is defined as follows: if a soldier has an additional suitable location besides their escape position, they share it with the other soldiers. Non-collaboration means that if a soldier has one or two additional suitable positions, they do not share them with other soldiers. In the non-collaboration scenario, the best responses are not transferred to the next iteration. Therefore, collaboration among soldiers is desirable. After generating twelve new positions, the fitness function is calculated for each, and the top three positions from the previous iteration are added. Three top candidates then advance to the next iteration from the existing 15 positions. The escape continues until the soldiers' last moment, and ultimately, the position of the top-performing soldier is considered the solution to the problem.

**Algorithm 1** CCET pseudocode

Initialize Competition and Cooperation in Evading Threat population $a_i(i = 1, 2, \ldots, n)$
Calculate the fitness of each population
$a_1$=the best search agent
$a_2$=the second best search agent
$a3$=the third best search agent
**while** ($t <$ Max number of iterations) **do**
  **for** each search agent **do**
    Update the position of the current search agent by equation (2.3)-(2.19)
  **end for**
  Calculate the fitness of all search agents
  Update $a_1, a_2$ and $a_3$
  $t = t + 1$
**end whilereturn** $a_1$

Here are some points about the CCET algorithm:

- The exploration process in the proposed algorithm involves all soldiers, while the exploitation process is executed individually by each soldier.
- Since the proposed algorithm is divergence-based, it possesses an effective capability to escape from local optima.
- Different intervals for $h_1, h_2, c_1$, and $c_2$ can be adjusted to configure the settings of the exploration and exploitation processes.

# 3 Results and discussion

In this section, the CCET algorithm undergœs validation on 23 benchmark functions. These 23 functions have been extensively utilized by researchers [54, 10, 32, 51, 27, 28]. Despite the simplicity of these functions, they serve the purpose of comparing the proposed algorithm with other well-known metaheuristic algorithms such as GWO [29], PSO, GSA, DE, and FEP. The functions are detailed in Table 2, Table 3, and Table 4, where $V$-no indicates the

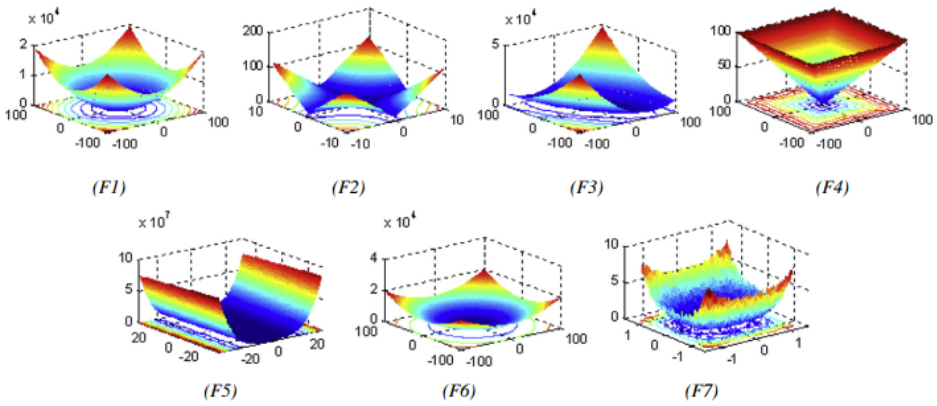


Figure 8: 2-D versions of unimodal benchmark functions [29].

number of design variables, range denotes the search space range, and fmin represents the optimum value of the function. Figures 8, 9, and 10 illustrate 2D versions of the employed benchmark functions. In general, the benchmark functions used are minimization functions and can be categorized into four groups: unimodal, multimodal, and fixed-dimension multimodal.

Table 2: Unimodal benchmark functions [31]

| Function | $V$-no | Range | $f_{\min}$ |
|---|---|---|---|
| $F_1(x) = \sum_{i=1}^{n} x_i^2$ | 30 | [-100,100] | 0 |
| $F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \Pi_{i=1}^{n} \lvert x_i \rvert$ | 30 | [-10,10] | 0 |
| $F_3(x) = \sum_{i=1}^{n} \left( \sum_{j=1}^{i} x_j \right)^2$ | 30 | [-100,100] | 0 |
| $F_4(x) = \max_i \{ \lvert x_i \rvert, \ 1 \leq i \leq n \}$ | 30 | [-100,100] | 0 |
| $F_5(x) = \sum_{i=1}^{n-1} \left[ 100 \left( x_{i+1} - x_i^2 \right)^2 + \left( x_i - 1 \right)^2 \right]$ | 30 | [-30,30] | 0 |
| $F_6(x) = \sum_{i=1}^{n} \left( [\lvert x_i \rvert + 0.5] \right)^2$ | 30 | [-100,100] | 0 |
| $F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$ | 30 | [-1.28,1.28] | 0 |

The CCET algorithm was run 30 times for each benchmark function, and the statistical results (average and standard deviation) are presented in Tables 5. To validate the outcomes, the CCET algorithm was compared with GWO [29] and PSO [48] as social intelligence-based techniques, and with GSA [41] as a physics-inspired algorithm. Furthermore, the CCET algorithm underwent comparisons with DE [37] and Fast Evolutionary Programming (FEP) [54].

Table 3: Multimodal benchmark functions [31]

| Function | $V$-no | Range | $f_{\min}$ |
|---|---|---|---|
| $F_8(x) = \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$ | 30 | [-500,500] | -418.9829×5 |
| $F_9(x) = \sum_{i=1}^{n} \left[x_i^2 + -10\cos(2\pi x_i) + 10\right]$ | 30 | [-5.12,5.12] | 0 |
| $F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$ | 30 | [-32,32] | 0 |
| $F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \Pi_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | 30 | [-600,36002] | 0 |
| $F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2[10\sin^2(\pi y_{i+1})] + (y_n-1)^2\right\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m & x_i > 0 \\ 0 & -a < x_i < a \\ k(-x_i-a)^m & x_i < -a \end{cases}$ | 30 | [-50,50] | 0 |
| $F_{13}(x) = 0.1\{\sin(3\pi x_1) + \sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]$ $+ (x_n-1)^2[1+\sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i,5,100,4)$ | 30 | [-50,50] | 0 |

Table 4: Fixed-dimenstion multimodal benchmark functions [31]

| Function | $V$-no | Range | $f_{\min}$ |
|---|---|---|---|
| $F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$ | 2 | [-65,65] | 1 |
| $F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$ | 4 | [-5,5] | 0.00030 |
| $F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$ | 2 | [-5,5] | -1.0316 |
| $F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1-\frac{1}{8\pi}\right)\cos x_1 + 10$ | 2 | [-5,5] | 0.398 |
| $F_{18}(x) = \left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]$ $\times\left[30+(2x_1-3x_2)^2\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$ | 2 | [-2,2] | 3 |
| $F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij}^2\right)$ | 3 | [1,3] | -3.86 |
| $F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij}^2\right)$ | 6 | [0,1] | -3.32 |
| $F_{21}(x) = -\sum_{i=1}^{5}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$ | 4 | [0,10] | -10.1532 |
| $F_{22}(x) = -\sum_{i=1}^{7}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$ | 4 | [0,10] | -10.4028 |
| $F_{23}(x) = -\sum_{i=1}^{10}\left[(x-a_i)(x-a_i)^T + c_i\right]^{-1}$ | 4 | [0,10] | -10.5363 |

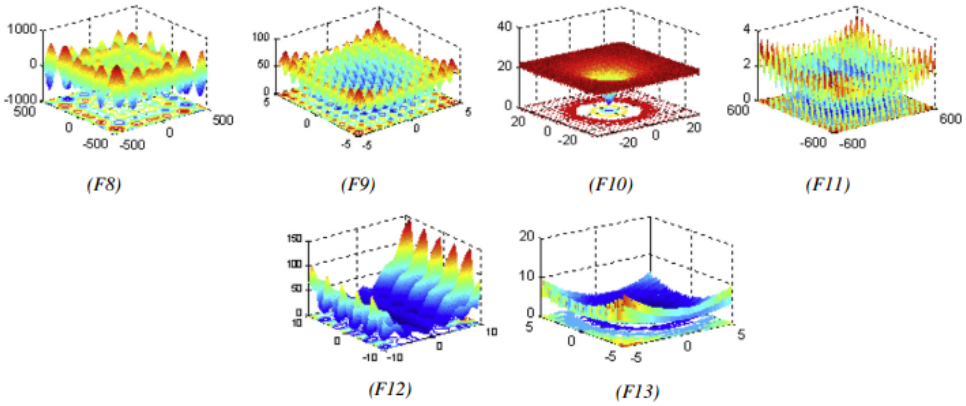


Figure 9: 2-D versions of multimodal benchmark functions [29].

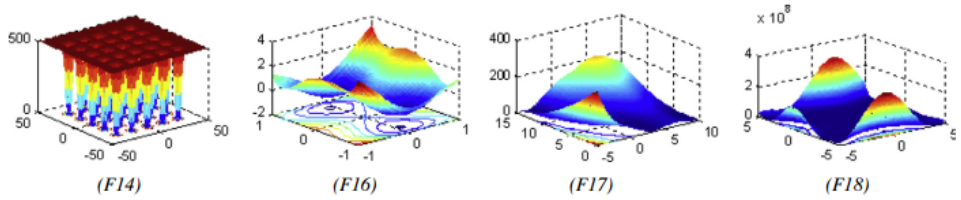


Figure 10: 2-D version of fixed-dimension multimodal benchmark functions [29].

Table 5: Results of benchmark functions

| F | CCET | | GWO | | PSO | | GSA | | DE | | FEP | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ave | std | ave | std | ave | std | ave | std | ave | std | ave | std |
| $F_1$ | 0 | 0 | 6.59E-28 | 6.34E-05 | 0.000136 | 0.000202 | 2.53E-16 | 9.67E-17 | 8.2E-14 | 5.9E-14 | 0.00057 | 0.00013 |
| $F_2$ | 0 | 0 | 7.18E-17 | 0.029014 | 0.042144 | 0.045421 | 0.055655 | 0.194074 | 1.5E-09 | 9.9E-10 | 0.0081 | 0.00077 |
| $F_3$ | 0 | 0 | 3.29E-06 | 79.14958 | 70.12562 | 22.11924 | 896.5347 | 318.9559 | 6.8E-11 | 7.4E-11 | 0.016 | 0.014 |
| $F_4$ | 0 | 0 | 5.61E-07 | 1.315088 | 1.086481 | 0.317039 | 7.35487 | 1.741452 | 0 | 0 | 0.3 | 0.5 |
| $F_5$ | 28.6681 | 0.3583 | 26.81258 | 69.90499 | 96.71832 | 60.11559 | 67.54309 | 62.22534 | 0 | 0 | 5.06 | 5.87 |
| $F_6$ | 6.8189 | 0.1366 | 0.816579 | 0.000126 | 0.000102 | 8.28E-05 | 2.5E-16 | 1.74E-16 | 0 | 0 | 0 | 0 |
| $F_7$ | 1.71E-04 | 2.2E-04 | 0.002213 | 0.100286 | 0.122854 | 0.044957 | 0.089441 | 0.04339 | 0.00463 | 0.0012 | 0.1415 | 0.3522 |
| $F_8$ | -2543.2 | 474.1163 | -6123.1 | -4087.44 | -4841.29 | 1152.814 | -2821.07 | 493.0375 | -11080.1 | 574.7 | -12554.5 | 52.6 |
| $F_9$ | 0 | 0 | 0.310521 | 47.35612 | 46.70423 | 11.62938 | 25.96841 | 7.470068 | 69.2 | 38.8 | 0.046 | 0.012 |
| $F_{10}$ | 8.88E-16 | 4.01E-31 | 1.06E-13 | 0.077835 | 0.276015 | 0.50901 | 0.062087 | 0.23628 | 9.7E-08 | 4.2E-08 | 0.018 | 0.0021 |
| $F_{11}$ | 0 | 0 | 0.004485 | 0.006659 | 0.009215 | 0.007724 | 27.70154 | 5.040343 | 0 | 0 | 0.016 | 0.022 |
| $F_{12}$ | 1.0008 | 0.1926 | 0.053438 | 0.020734 | 0.006917 | 0.026301 | 1.799617 | 0.95114 | 7.9E-15 | 8E-15 | 9.2E-06 | 3.6E-06 |
| $F_{13}$ | 2.9981 | 6.4E-04 | 0.654464 | 0.004474 | 0.006675 | 0.008907 | 8.899084 | 7.126241 | 5.1E-14 | 4.8E-14 | 0.00016 | 0.000073 |
| $F_{14}$ | 1.7215 | 2.2221 | 4.042493 | 4.252799 | 3.627168 | 2.560828 | 5.859838 | 3.831299 | 0.998004 | 3.3E-16 | 1.22 | 0.56 |
| $F_{15}$ | 0.0145 | 0.0240 | 0.000337 | 0.000625 | 0.000577 | 0.000222 | 0.003673 | 0.001647 | 4.5E-14 | 0.00033 | 0.0005 | 0.00032 |
| $F_{16}$ | -1.0316 | 6.77E-16 | -1.03163 | -1.03163 | -1.03163 | 6.25E-16 | -1.03163 | 4.88E-16 | -1.03163 | 3.1E-13 | -1.03 | 4.9E-07 |
| $F_{17}$ | 0.3982 | 3.67E-04 | 0.397889 | 0.397887 | 0.397887 | 0 | 0.397887 | 0 | 0.397887 | 9.9E-09 | 0.398 | 1.5E-07 |
| $F_{18}$ | 3 | 0 | 3.000028 | 3 | 3 | 1.33E-15 | 3 | 4.17E-15 | 3 | 2E-15 | 3.02 | 0.11 |
| $F_{19}$ | $N/A$ | $N/A$ | -3.86263 | -3.86278 | -3.86278 | 2.58E-15 | -3.86278 | 2.29E-15 | $N/A$ | $N/A$ | -3.86 | 0.000014 |
| $F_{20}$ | -2.6702 | 0.3898 | -3.28654 | -3.25056 | -3.26634 | 0.060516 | -3.31778 | 0.023081 | $N/A$ | $N/A$ | -3.27 | 0.059 |
| $F_{21}$ | -3.8044 | 1.0583 | -10.1514 | -9.14015 | -6.8651 | 3.019644 | -5.95512 | 3.737079 | -10.1532 | 0.0000025 | -5.52 | 1.59 |
| $F_{22}$ | -4.6662 | 2.5125 | -10.4015 | -8.58441 | -8.45653 | 3.087094 | -9.68447 | 2.014088 | -10.4029 | 3.9E-07 | -5.53 | 2.12 |
| $F_{23}$ | -4.7261 | 0.9022 | -10.5343 | -8.55899 | -9.95291 | 1.782786 | -10.5364 | 2.6E-15 | -10.5364 | 1.9E-07 | -6.57 | 3.14 |

## 3.1 Evaluation of Unimodal Benchmark Functions

Functions $F_1$-$F_7$ are single-objective, having only one global optimum. These functions showcase the exploitation capability in metaheuristic algorithms. The CCEE algorithm has successfully reached the global optimum in functions $F_1$-$F_4$, demonstrating superior performance compared to other algorithms. Competitive results are also observed in functions $F_5$ and $F_7$.

## 3.2 Evaluation of Multimodal Benchmark and Fixed-Dimension Multimodal Benchmark Functions

In contrast to single-objective functions, functions $F_8$-$F_{23}$ exhibit numerous local optima. These functions are employed to assess the exploration ability in metaheuristic algorithms. The CCEE algorithm has effectively reached the optimum point in functions $F_9$, $F_{11}$, and $F_{18}$, outperforming other algorithms in functions $F_8$ and $F_{10}$. Competitive results are achieved in functions $F_{14}$-$F_{18}$.

## 3.3 Convergence Speed

The convergence of the proposed algorithm is compared with GWO and PSO algorithms in Figures 13 to 35. The results illustrate the competitive performance of the CCET algorithm in terms of convergence.

# 4 Application of CCET algorithm in practical problems

Two practical problems are optimized by CCET algorithm and verified with other well-known algorithms.

## 4.1 Pressure vessel design

In this problem the goal is to minimize the total cost (material, forming and welding) of the cylindrical pressure vessel shown in Fig. 11. Both ends of the vessel are capped while the head has a hemi-spherical shape. There are four optimization variables: the thickness of the shell ($Ts$), the thickness of the head ($Th$), the inner radius ($R$), the length of the cylindrical section without considering the head ($L$). The problem includes four optimization constraints and is formulated as follows [31]:

$$
\begin{aligned}
&\text{Consider } \overrightarrow{x} = [x_1 x_2 x_3 x_4] = [T_s T_h R L],\\
&\text{Minimize } f(\overrightarrow{x}) = 0.6224 x_1 x_3 x_4 + 1.7781 x_2 x_3^2 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3,\\
&\text{Subject to } g_1(\overrightarrow{x}) = -x_1 + 0.0193 x_3 \leq 0,\\
&\qquad\quad g_2(\overrightarrow{x}) = -x_3 + 0.00954 x_3 \leq 0,\\
&\qquad\quad g_3(\overrightarrow{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0,\\
&\qquad\quad g_4(\overrightarrow{x}) = x_4 - 240 \leq 0,\\
&\text{Variable range } 0 \leq x_1 \leq 99,\\
&\qquad\quad 0 \leq x_2 \leq 99,\\
&\qquad\quad 10 \leq x_3 \leq 200,\\
&\qquad\quad 10 \leq x_4 \leq 200,
\end{aligned}
\tag{4.1}
$$

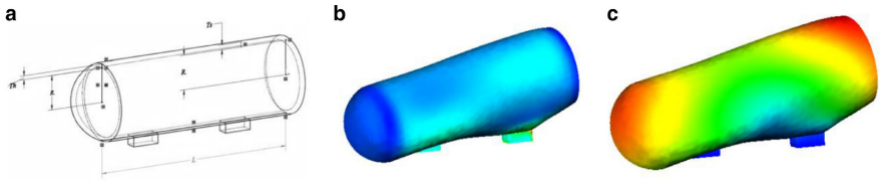


Figure 11: Pressure vessel ($a$) shematic ($b$) stress heatmap ($c$) displacement heatmap [31].

Table 6: Comparison of CCET optimization results with literature for the pressure vessel design problem.

| Algorithm | Optimum variables | | | | Optimum |
|---|---|---|---|---|---|
| | $T_s$ | $T_h$ | $R$ | $L$ | |
| CCET | 0.937500 | 0.500000 | 48.329000 | 112.679000 | 6410.3811 |
| GSA | 1.125000 | 0.625000 | 55.9886598 | 84.4542025 | 8538.8359 |
| PSO | 0.812500 | 0.437500 | 42.091266 | 176.746500 | 6061.0777 |
| GA | 0.812500 | 0.434500 | 40.323900 | 200.000000 | 6288.7445 |
| DE | 0.812500 | 0.437500 | 42.098411 | 176.637690 | 6059.7340 |
| ACO | 0.812500 | 0.437500 | 42.103624 | 176.572656 | 6059.0888 |

## 4.2 Tension/compression spring design

The objective of this test problem is to minimize the weight of the tension/compression spring shown in Fig. 12. Optimum design must satisfy constraints on shear stress, surge frequency and deflection. There are three design variables: wire diameter ($d$), mean coil diameter ($D$), and number of active coils ($N$).

Optimization problem is formulated as follows:

$$
\begin{aligned}
&\text{Consider } \overrightarrow{x} = [x_1 x_2 x_3] = [dDN],\\
&\text{Minimize } f(\overrightarrow{x}) = (x_3+2)x_2x_1^2,\\
&\text{Subject to } g_1(\overrightarrow{x}) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \leq 0,\\
&\qquad g_2(\overrightarrow{x}) = \frac{4x_2^2 - x_1x_2}{12566\,(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} \leq 0,\\
&\qquad g_3(\overrightarrow{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0,\\
&\qquad g_4(\overrightarrow{x}) = \frac{x_1+x_2}{1.5} - 1 \leq 0,\\
&\text{Variable range } 0.05 \leq x_1 \leq 2.00,\\
&\qquad 0.25 \leq x_2 \leq 1.30,\\
&\qquad 2.00 \leq x_3 \leq 15.0,
\end{aligned}
\tag{4.2}
$$

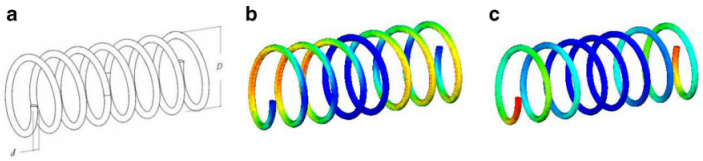


Figure 12: ($a$) Schematic of the spring; ($b$) stress distribution evaluated at the optimum design; and ($c$) displacement distribution evaluated at the optimum design [31].

Table 7: Comparison of CCET optimization results with literature for the tension/compression spring design problem.

| Algorithm | Optimum variables | | | Optimum weight |
|---|---|---|---|---|
| | $d$ | $D$ | $N$ | |
| CCET | 0.051370 | 0.349096 | 11.76279 | 0.0126788 |
| GSA | 0.050276 | 0.323680 | 13.525410 | 0.0127022 |
| PSO | 0.051728 | 0.357644 | 11.244543 | 0.0126747 |
| GA | 0.051480 | 0.351661 | 11.632201 | 0.0127048 |
| DE | 0.051609 | 0.354714 | 11.410831 | 0.0126702 |

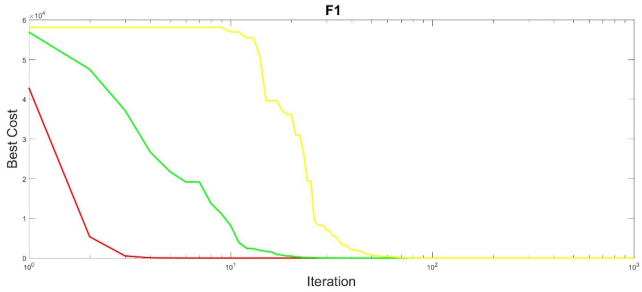


Figure 13: $F_1$ benchmark function.

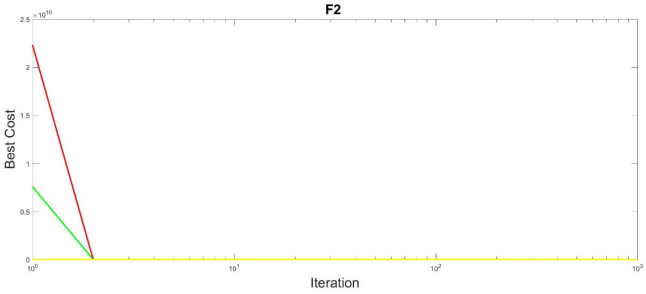


Figure 14: $F_2$ benchmark function.

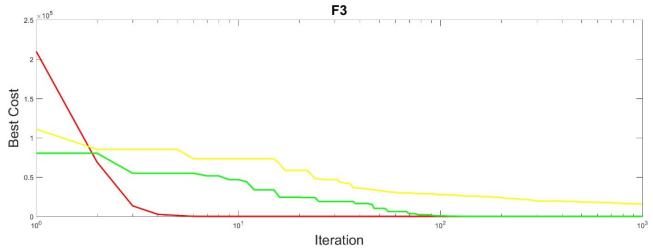


Figure 15: $F_3$ benchmark function.

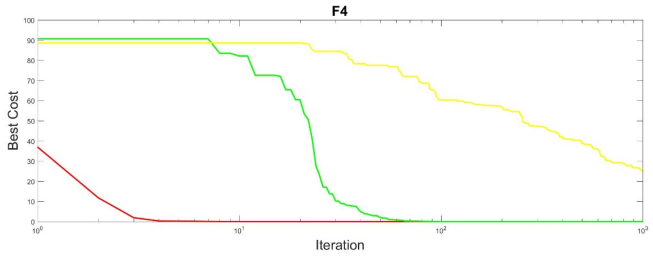


Figure 16: $F_4$ benchmark function.

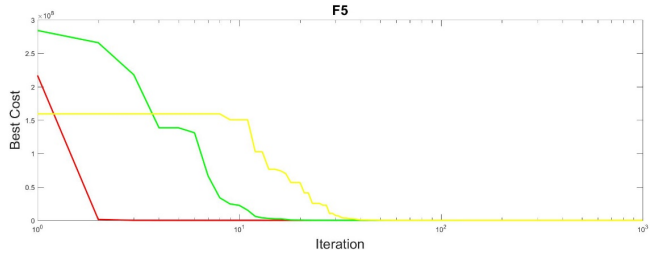


Figure 17: $F_5$ benchmark function.

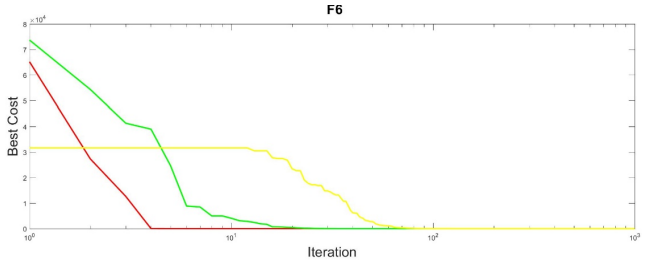


Figure 18: $F_6$ benchmark function.

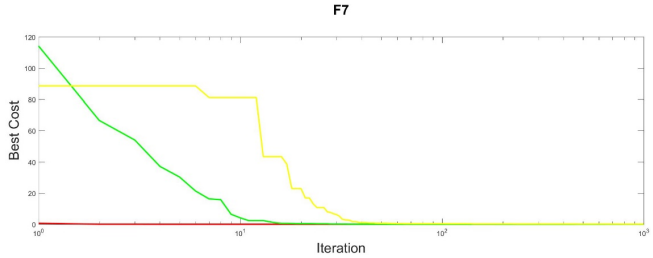


Figure 19: $F_7$ benchmark function.

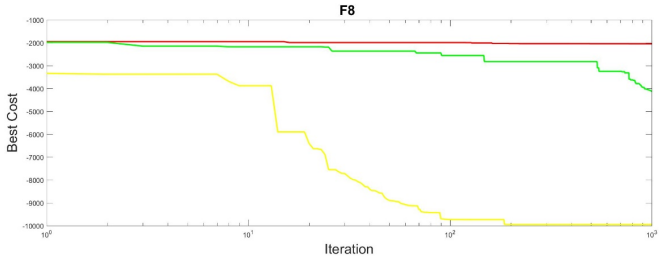


Figure 20: $F_8$ benchmark function.

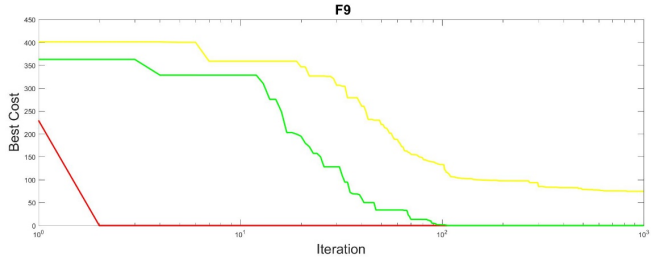


Figure 21: $F_9$ benchmark function.

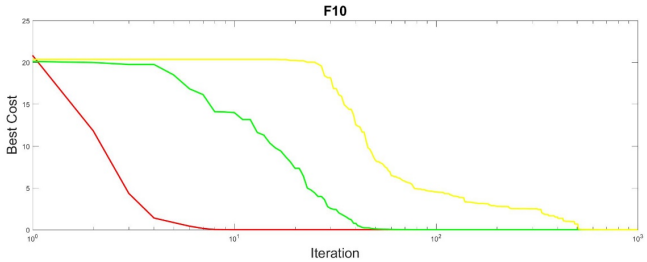


Figure 22: $F_{10}$ benchmark function.

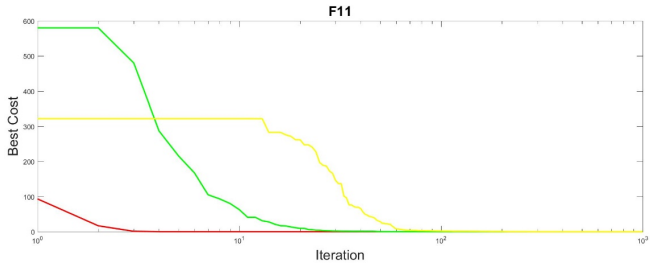


Figure 23: $F_{11}$ benchmark function.

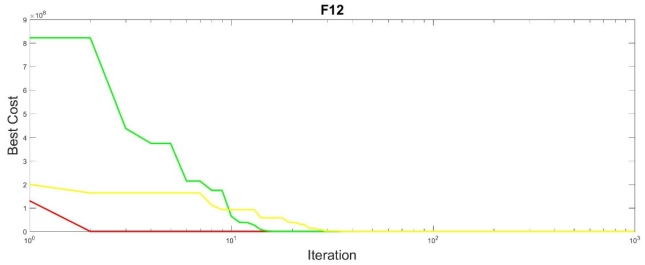


Figure 24: $F_{12}$ benchmark function.

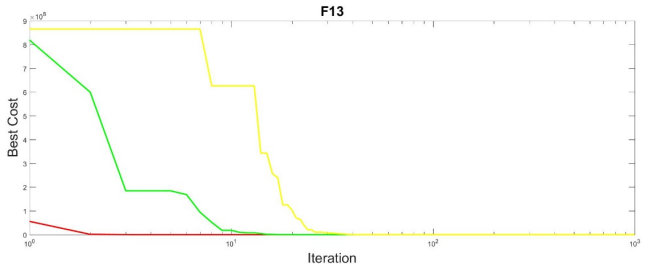


Figure 25: $F_{13}$ benchmark function.

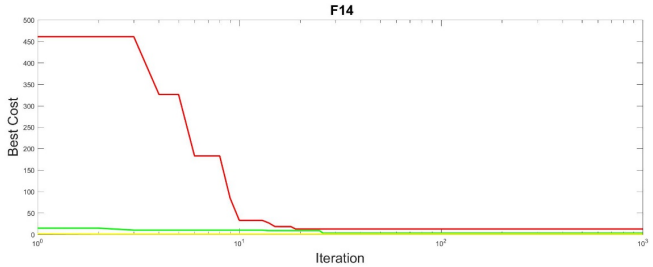


Figure 26: $F_{14}$ benchmark function.

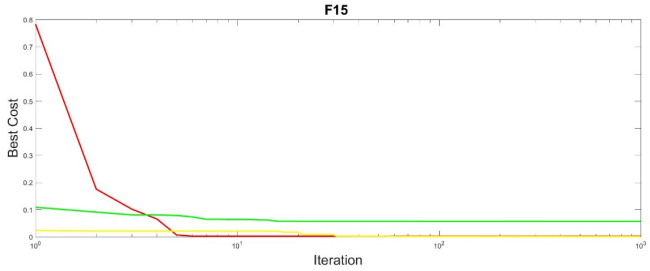


Figure 27: $F_{15}$ benchmark function.

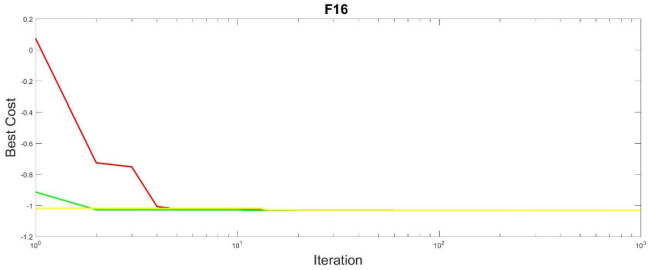


Figure 28: $F_{16}$ benchmark function.

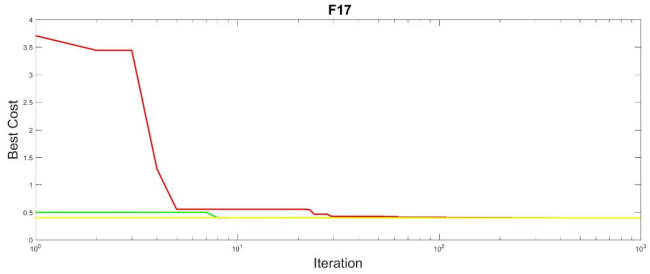


Figure 29: $F_{17}$ benchmark function.

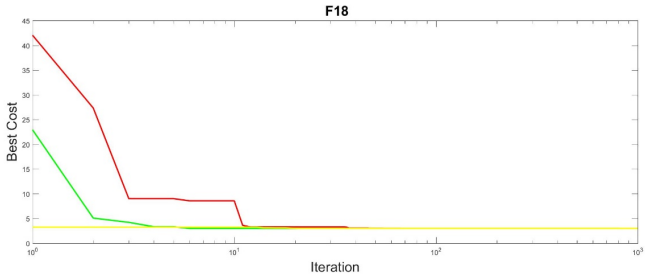


Figure 30: $F_{18}$ benchmark function.

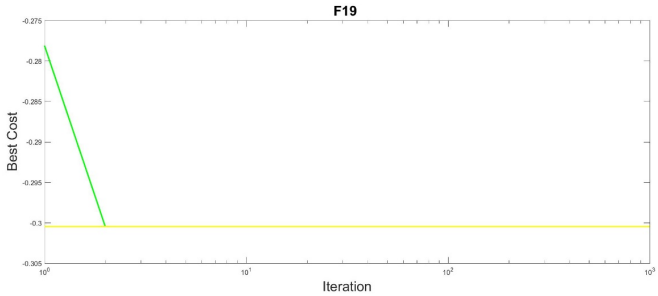


Figure 31: $F_{19}$ benchmark function.

# References

[1] Abdollahzadeh, B., Gharehchopogh, F. S., Mirjalili, S. (2021). African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. Computers Industrial Engineering, 158, 107408.

[2] Adham, M. T., Bentley, P. J. (2014, December). An artificial ecosystem algorithm applied to static and dynamic travelling salesman problems. In 2014 IEEE international conference on evolvable systems (pp. 149-156). IEEE.

[3] Afroughinia, A., Kardehi Moghaddam, R. (2018). Competitive learning: a new meta-heuristic optimization algorithm. International Journal on Artificial Intelligence Tools, 27(08), 1850035.

[4] Al-Betar, M. A., Alyasseri, Z. A. A., Awadallah, M. A., Abu Doush, I. (2021). Coronavirus herd immunity optimizer (CHIO). Neural Computing and Applications, 33, 5011-5042.

[5] Atashpaz-Gargari, E., Lucas, C. (2007, September). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In 2007 IEEE congress on evolutionary computation (pp. 4661-4667). Ieee.

[6] Brammya, G., Praveena, S., Ninu Preetha, N. S., Ramya, R., Rajakumar, B. R., Binu, D. (2019). Deer hunting optimization algorithm: a new nature-inspired meta-heuristic paradigm. The Computer Journal, bxy133.

[7] Civicioglu, P. (2013). Artificial cooperative search algorithm for numerical optimization problems. Information Sciences, 229, 58-76.

[8] Das, S., Chowdhury, A., Abraham, A. (2009, May). A bacterial evolutionary algorithm for automatic data clustering. In 2009 IEEE congress on evolutionary computation (pp. 2403-2410). IEEE.

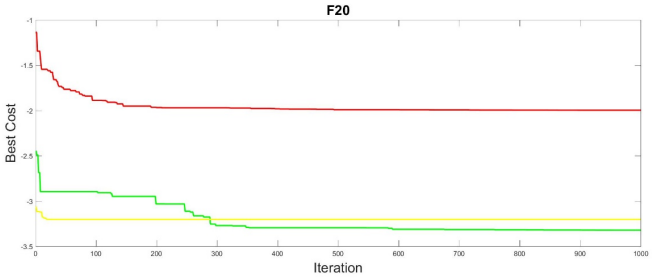


Figure 32: $F_{20}$ benchmark function.

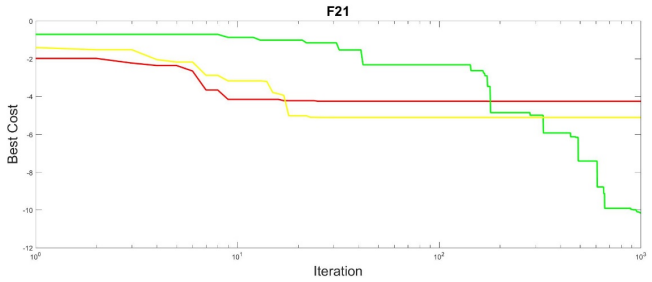


Figure 33: $F_{21}$ benchmark function.

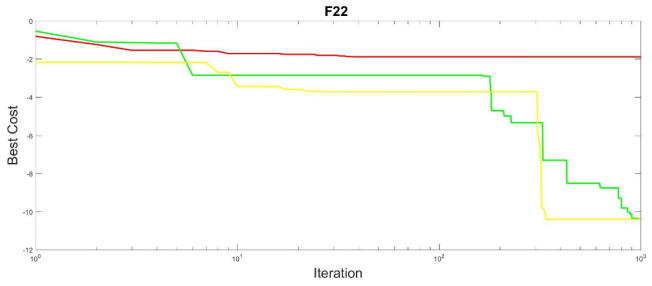


Figure 34: $F_{22}$ benchmark function.

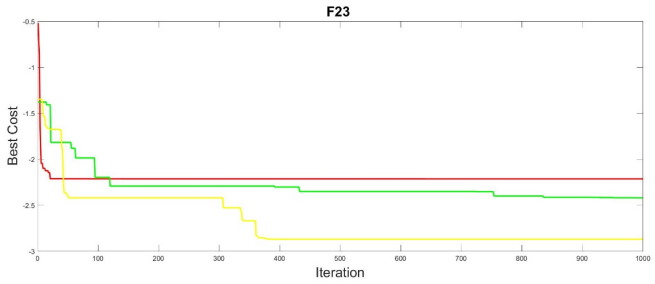


Figure 35: $F_{23}$ benchmark function.

[9] De Castro, L. N., Von Zuben, F. J. (2000, July). The clonal selection algorithm with engineering applications. In Proceedings of GECCO (Vol. 2000, pp. 36-39).

[10] Digalakis, J. G., Margaritis, K. G. (2001). On benchmarking functions for genetic algorithms. International journal of computer mathematics, 77(4), 481-506.

[11] Dorigo, M., Maniezzo, V., Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. IEEE transactions on systems, man, and cybernetics, part b (cybernetics), 26(1), 29-41.

[12] Erol, O. K., Eksin, I. (2006). A new optimization method: big bang–big crunch. Advances in engineering software, 37(2), 106-111.

[13] Eskandar, H., Sadollah, A., Bahreininejad, A., Hamdi, M. (2012). Water cycle algorithm–A novel metaheuristic optimization method for solving constrained engineering optimization problems. Computers Structures, 110, 151-166.

[14] Halim, A. H., Ismail, I. (2018). Tree physiology optimization in constrained optimization problem. Telkomnika (Telecommunication Computing Electronics and Control), 16(2), 876-882.

[15] Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. Information sciences, 222, 175-184.

[16] He, S., Wu, Q. H., Saunders, J. R. (2006, July). A novel group search optimizer inspired by animal behavioural ecology. In 2006 IEEE international conference on evolutionary computation (pp. 1272-1278). IEEE.

[17] Holland, J. H. (1992). Genetic algorithms. Scientific american, 267(1), 66-73.

[18] Karaboga, D., Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39, 459-471.

[19] Karci, A. (2012, September). A new meta-heuristic algorithm based on chemical process: atom algorithm. In 1st International Eurasian Conference on Mathematical Sciences and Applications (pp. 3-7).

[20] Kennedy, J., Eberhart, R. (1995, November). Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). ieee.

[21] Khalid, A. M., Hosny, K. M., Mirjalili, S. (2022). COVIDOA: a novel evolutionary optimization algorithm based on coronavirus disease replication lifecycle. Neural Computing and Applications, 34(24), 22465-22492.

[22] Kiran, M. S. (2015). TSA: Tree-seed algorithm for continuous optimization. Expert Systems with Applications, 42(19), 6686-6698.

[23] Kirkpatrick, S. (1983). Improvement of reliabilities of regulations using a hierarchical structure in a genetic network. Science, 220, 671-680.

[24] Koza, J. R. (1990, November). Genetically breeding populations of computer programs to solve problems in artificial intelligence. In [1990] Proceedings of the 2nd International IEEE Conference on Tools for Artificial Intelligence (pp. 819-827). IEEE.

[25] Marinakis, Y., Marinaki, M., Matsatsinis, N. (2010). A bumble bees mating optimization algorithm for global unconstrained optimization problems. In Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) (pp. 305-318). Berlin, Heidelberg: Springer Berlin Heidelberg.

[26] Meng, X. B., Gao, X. Z., Lu, L., Liu, Y., Zhang, H. (2016). A new bio-inspired optimisation algorithm: Bird Swarm Algorithm. Journal of Experimental Theoretical Artificial Intelligence, 28(4), 673-687.

[27] Mirjalili, S., Lewis, A. (2013). S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm and Evolutionary Computation, 9, 1-14.

[28] Mirjalili, S., Mirjalili, S. M., Yang, X. S. (2014). Binary bat algorithm. Neural Computing and Applications, 25, 663-681.

[29] Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.

[30] Mirjalili, S. (2015). The ant lion optimizer. Advances in engineering software, 83, 80-98.

[31] Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. Advances in engineering software, 95, 51-67.

[32] Molga, M., Smutnicki, C. (2005). Test functions for optimization needs. Test functions for optimization needs, 101, 48.

[33] Mucherino, A., Seref, O. (2007, November). Monkey search: a novel metaheuristic search for global optimization. In AIP conference proceedings (Vol. 953, No. 1, pp. 162-173). American Institute of Physics.

[34] Odili, J. B., Kahar, M. N. M., Anwar, S. (2015). African buffalo optimization: a swarm-intelligence technique. Procedia Computer Science, 76, 443-448.

[35] Oftadeh, R., Mahjoob, M. J. (2009, September). A new meta-heuristic optimization algorithm: Hunting Search. In 2009 fifth international conference on soft computing, computing with words and perceptions in system analysis, decision and control (pp. 1-5). IEEE.

[36] Osaba, E., Diaz, F., Onieva, E. (2014). Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts. Applied intelligence, 41, 145-166.

[37] Price, K. V., Storn, R. M., Lampinen, J. A. (2005). The differential evolution algorithm. Differential evolution: a parctical approach to global optimization, 37-134.

[38] Purnomo, H. D., Wee, H. M. (2013). Soccer game optimization: an innovative integration of evolutionary algorithm and swarm intelligence algorithm. In Meta-Heuristics optimization algorithms in engineering, business, economics, and finance (pp. 386-420). Igi Global.

[39] Rabanal, P., Rodriguez, I., Rubio, F. (2007, August). Using river formation dynamics to design heuristic algorithms. In International conference on unconventional computation (pp. 163-177). Berlin, Heidelberg: Springer Berlin Heidelberg.

[40] Rajabioun, R. (2011). Cuckoo optimization algorithm. Applied soft computing, 11(8), 5508-5518.

[41] Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S. (2009). GSA: a gravitational search algorithm. Information sciences, 179(13), 2232-2248.

[42] Sayed, G. I., Tharwat, A., Hassanien, A. E. (2019). Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection. Applied Intelligence, 49, 188-205.

[43] Sebald, A. V., Fogel, L. J. (1994). Evolutionary Programming: Proceedings of the Third Annual Conference. In Evolutionary Programming: Proceedings of the Third Annual Conference (pp. 1-386).

[44] Shadravan, S., Naji, H. R., Bardsiri, V. K. (2019). The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Engineering Applications of Artificial Intelligence, 80, 20-34.

[45] Shah-Hosseini, H. (2009). The intelligent water drops algorithm: a nature-inspired swarm-based optimization algorithm. International Journal of Bio-inspired computation, 1(1-2), 71-79.

[46] Shayanfar, H., Gharehchopogh, F. S. (2018). Farmland fertility: A new metaheuristic algorithm for solving continuous optimization problems. Applied Soft Computing, 71, 728-746.

[47] Shi, Y. (2011). Brain storm optimization algorithm. In Advances in Swarm Intelligence: Second International Conference, ICSI 2011, Chongqing, China, June 12-15, 2011, Proceedings, Part I 2 (pp. 303-309). Springer Berlin Heidelberg.

[48] Simon, D. (2008). Biogeography-based optimization. IEEE transactions on evolutionary computation, 12(6), 702-713.

[49] Sulaiman, M., Salhi, A., Selamoglu, B. I., Kirikchi, O. B. (2014). A plant propagation algorithm for constrained engineering optimisation problems. Mathematical problems in engineering, 2014(1), 627416.

[50] Yang, X. S. (2009, October). Firefly algorithms for multimodal optimization. In International symposium on stochastic algorithms (pp. 169-178). Berlin, Heidelberg: Springer Berlin Heidelberg.

[51] Yang, X. S. (2010). Test problems in optimization. arXiv preprint arXiv:1008.0549.

[52] Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization (NICSO 2010) (pp. 65-74). Berlin, Heidelberg: Springer Berlin Heidelberg.

[53] Yang, X. S. (2012, September). Flower pollination algorithm for global optimization. In International conference on unconventional computing and natural computation (pp. 240-249). Berlin, Heidelberg: Springer Berlin Heidelberg.

[54] Yao, X., Liu, Y., Lin, G. (1999). Evolutionary programming made faster. IEEE Transactions on Evolutionary computation, 3(2), 82-102.

[55] Zelinka, I. (2016). SOMA-self-organizing migrating algorithm. Self-Organizing Migrating Algorithm: Methodology and Implementation, 3-49.

[56] Zhang, X., Jiang, K., Wang, H., Li, W., Sun, B. (2012). An improved bean optimization algorithm for solving TSP. In Advances in Swarm Intelligence: Third International Conference, ICSI 2012, Shenzhen, China, June 17-20, 2012 Proceedings, Part I 3 (pp. 261-267). Springer Berlin Heidelberg.

[57] Zheng, Y. J. (2015). Water wave optimization: a new nature-inspired metaheuristic. Computers Operations Research, 55, 1-11.