Intelligent Automation & Soft Computing

Tech Science Press

DOI: 10.32604/iasc.2024.053192

ARTICLE

Chase, Pounce, and Escape Optimization Algorithm

Adel Sabry Eesa*

Computer Science Department, Faculty of Science, University of Zakho, Duhok City, 42001, Iraq

*Corresponding Author: Adel Sabry Eesa. Email: adel.eesa@uoz.edu.krd

Received: 26 April 2024 Accepted: 02 July 2024 Published: 06 September 2024

ABSTRACT

While many metaheuristic optimization algorithms strive to address optimization challenges, they often grapple with the delicate balance between exploration and exploitation, leading to issues such as premature convergence, sensitivity to parameter settings, and difficulty in maintaining population diversity. In response to these challenges, this study introduces the Chase, Pounce, and Escape (CPE) algorithm, drawing inspiration from predator-prey dynamics. Unlike traditional optimization approaches, the CPE algorithm divides the population into two groups, each independently exploring the search space to efficiently navigate complex problem domains and avoid local optima. By incorporating a unique search mechanism that integrates both the average of the best solution and the current solution, the CPE algorithm demonstrates superior convergence properties. Additionally, the inclusion of a pouncing process facilitates rapid movement towards optimal solutions. Through comprehensive evaluations across various optimization scenarios, including standard test functions, Congress on Evolutionary Computation (CEC)-2017 benchmarks, and real-world engineering challenges, the effectiveness of the CPE algorithm is demonstrated. Results consistently highlight the algorithm's performance, surpassing that of other well-known optimization techniques, and achieving remarkable outcomes in terms of mean, best, and standard deviation values across different problem domains, underscoring its robustness and versatility.

KEYWORDS

Bio-inspired optimization; metaheuristic; chase, pounce, and escape optimizer; collective behavior; engineering design problems

1 Introduction

In many fields, including physics, biology, engineering, economics, and business, optimization algorithms are the foundation of problem-solving efforts. They are primarily classified into two types: deterministic and metaheuristic. The goal of optimization algorithms is to find the best possible solution within given constraints [1]. Deterministic algorithms employ gradient-based methods and are particularly effective for solving unimodal problems. In contrast, metaheuristic algorithms leverage stochastic search strategies to navigate complex multi-dimensional landscapes in pursuit of global optima. The richness of these optimization techniques is highlighted by their application to real-world challenges, where factors like time, design specifications, and geometric complexities pose unique hurdles to finding optimal solutions. To tackle this, several metaheuristic optimization algorithms have been developed, inspired by nature, such as physics-based algorithms, Swarm Intelligence (SI), and Evolutionary Algorithms (EA). EA imitates organic evolution by using processes including recombination, mutation, and selection [2], while SI algorithms, like Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), are based on collective swarm behavior [3]. Recent years have witnessed the proliferation of metaheuristic optimization algorithms, inspired by various phenomena in nature including physics-based algorithms [4], animal behavior-based algorithms [5–9], and human behavior-based algorithms [10–12]. Each of these paradigms has unique inspirations and methodologies, developed with the goal of effectively balancing exploration and exploitation. Despite these efforts, many existing algorithms still face significant challenges in achieving this balance.

Amidst recent algorithm proposals, a critical question arises: do we need more optimization techniques? The No Free Lunch (NFL) theorem [13] indicates that no single algorithm can address all problems effectively, as each excels only in specific cases. On average, optimization techniques perform equally across all problems. The persistent challenge of balancing exploration and exploitation continues to inspire innovative adaptations. These factors drive researchers to develop or enhance algorithms tailored to specific subsets within various fields. our research introduces the Chase, Pounce, and Escape (CPE) algorithm, a novel metaheuristic approach inspired by the hunting dynamics of predators and prey. Unlike conventional optimization algorithms, the CPE algorithm partitions the population into two distinct groups, each independently exploring the search space. This unique approach facilitates efficient navigation through complex problem domains while mitigating the risk of local optima.

The contributions of this paper include:

- Introduction of the CPE algorithm: We introduce a novel nature-inspired metaheuristic as a competitive alternative to existing methods. This algorithm employs an innovative partitioning strategy, dividing the population into two groups operating in different scenarios, thereby enhancing both exploration and exploitation dynamics.
- Evaluation across various optimization scenarios: We conduct a comprehensive evaluation of the CPE algorithm's efficacy across a spectrum of optimization scenarios, encompassing 50 standard test functions and CEC-2017 benchmark functions.
- Application to real-world engineering challenges: The proposed CPE is applied and tested on four prominent engineering problems, including the design of welded beams, speed reducers, cantilever beams, and multi-plate disc clutch brakes.

To provide a comprehensive overview, we organize the remainder of the paper as follows: Section 2 reviews related work, focusing on the drawbacks identified in current literature. Section 3 delineates the approach to cooperative hunting between predators, serving as the theoretical foundation for the proposed CPE optimization algorithm. In Section 4, we elaborate on the intricacies of the CPE algorithm, outlining its methodology and design principles. Section 5 presents experimental results, showcasing the efficacy of the CPE algorithm across various optimization problems, including real-world scenarios. Finally, Section 6 concludes the paper by summarizing key findings and suggesting avenues for future research.

2 Related Work

With different motivations and approaches, many optimization algorithms have been created with the goal of efficiently striking a balance between exploration and exploitation. But in this aspect, a lot of the current algorithms face significant challenges. EAs, PSOs, and ACO are a few of the most well-known. EAs work by means of mechanisms including recombination, mutation, and selection. They are inspired by the concepts of biological evolution. To produce excellent answers for optimization issues, these algorithms mimic natural evolutionary processes. EAs are widely used in many different areas, but they frequently encounter substantial obstacles, such as premature convergence, which occurs when the algorithm becomes stuck in local optima, and maintaining an effective balance between exploration and exploitation [14].

Fish schools and flocks of birds serve as models for PSO social dynamics. Potential solutions are represented by a population of particles in PSO, which traverse the search space under the impact of their own and neighboring experiences. Despite PSO's effectiveness in resolving a variety of optimization issues, it can have trouble preserving diversity within the swarm and is prone to early convergence. In notably complex, multimodal landscapes, this may lead to inadequate investigation [15].

Ants foraging strategy served as the model for ACO. Ants leave behind pheromones on the paths they travel, which direct other ants' following motions. Finding the best routes to resources is made easier by this cooperative process. However, if the pheromone update mechanism is too aggressive, ACO algorithms may converge too quickly on paths that are not optimal. This can drastically diminish the algorithm's capacity for exploration, which will limit its ability to find global optima in intricate problem spaces [16].

Recent developments in optimization algorithms have drawn inspiration from various domains, including physics, animal behavior, and human behavior. Table 1 highlights some of the most popular metaheuristic optimization algorithms from the past decade, detailing their inspirations, advantages, and disadvantages.

Physics inspirations such as Gravitational Search Algorithm (GSA) [17] are based on the law of gravity and mass interactions. While GSA is powerful, it often converges slowly, especially in high-dimensional spaces, due to the gradual reduction in exploration capability.

Animal-inspired algorithms like the Cuckoo Search (CS) [18] and Grey Wolf Optimizer (GWO) [19] draw inspiration from animal behavior. CS, inspired by the brood parasitism of cuckoos, and GWO, mimicking the leadership hierarchy in grey wolf packs, both exhibit strong exploration abilities initially. However, they can suffer from a rapid convergence phase, limiting their exploitation capabilities in the later stages [20,21]. The Whale Optimization Algorithm (WOA) [5] and Salp Swarm algorithm (SSA) [7], inspired by the social behavior of whales and salps respectively, show similar trends where maintaining a balance between global search (exploration) and local search (exploitation) remains challenging. Harris Hawk Optimization (HHO) [9], Emperor Penguins Colony (EPC) [22], and Mayfly Optimization Algorithm (MA) [23] show promise in tackling specific problems but still face the common issue of balancing exploration and exploitation. HHO, inspired by the cooperative behavior of Harris hawks, and EPC, mimicking the huddling behavior of emperor penguins, often require careful parameter adjustment to maintain this balance.

Human behavior-based inspirations, such as Political Optimizer (PO) [10] and Heap-Based Optimizer (HBO) [11], Teaching-Learning Based Optimization (TLBO) [12] algorithms introduce unique social and computational strategies but similarly struggle with premature convergence and maintaining diversity in the solution space.

Overall, while these algorithms introduce various innovative strategies to address optimization problems, they often struggle with the trade-off between exploration and exploitation. Premature convergence, sensitivity to parameter settings, and difficulty in maintaining diversity within the population are common challenges [24]. In addition, optimization algorithms assuming function continuity struggle when faced with non-smooth or discontinuous functions. Continuous functions, which exhibit smooth behavior, enable seamless exploration. In contrast, non-smooth functions, featuring abrupt changes or non-differentiable points, challenge traditional methods relying on smoothness and differentiability. The need for adaptive, robust, and efficient optimization techniques continues to drive research in this field, motivating the development of new algorithms such as the Chase, Pounce, and Escape (CPE) algorithm proposed in this study. The CPE algorithm aims to address these challenges by leveraging novel search strategies inspired by the hunting dynamics of predators and prey, offering a competitive alternative to existing metaheuristic approaches. The CPE algorithm introduced in this study innovatively balances exploration and exploitation through two distinct predator groups. In the first group, predators employ a weighted combination of current, best, and average solutions to approach and encircle prey, facilitating targeted exploration while maintaining diversity. The second group utilizes random equation selection to update positions, mimicking the pouncing and escape process of prey, promoting adaptive exploration.

Table 1: List of optimization methods along with domain and inspiration sources

References	Year	Algorithm name	Domain	Inspiration	Advantage	Disadvantage
[17]	2009	Gravitational Search Algorithm (GSA)	Physics	Gravitation law	Simplicity, global search capability.	Slow convergence, getting stuck in local minima in last iterations.
[18]	2010	Cuckoo Search (CS)	Animals	Brood parasite of cuckoo and Levy flight	Easier to application, fewer tuning parameters.	Easily fall into local optimal, slow rate of convergence.
[12]	2011	Teaching-Learning Based Optimization (TLBO)	Human	Teacher and learner. The learning is based on two phases: acquiring knowledge from instructors and gaining insight from fellow students	Simplicity, no tuning parameters.	Weak population diversity, the tendency to fall into local optima.
[19]	2014	Grey Wolf Optimizer (GWO)	Animals	The leadership hierarchy and hunting style of grey wolves in nature	Ease of use, global search capability, and versatility.	Slow convergence speed, low solving accuracy, limited scalability for high dimensions.

(Continued)

Table 1 (continued)

References	Year	Algorithm name	Domain	Inspiration	Advantage	Disadvantage
[25]	2016	Sine Cosine Algorithm (SCA)	Physics	Mathematical model based on sine and cosine functions	Simple, user-friendly, fast due to its straightforward code, explores well due to randomness.	Good for continuous problems only, lacks memory, no global optimality guarantee, slow in complex problems.
[5]	2016	Whale Optimization Algorithm, (WOA)	Animals	Social behavior of humpback whales	Simple structure, good in various fields.	Slow convergence speed, low precision, falling into local optimal easily.
[6]	2017	Killer Whale Algorithm (KWA)	Animals	Movement killer whale in hunting and social structure	Global search capability.	Complex, limited scalability.
[7]	2017	Salp Swarm Optimization (SSA)	Animals	Swarm navigation and foraging behavior in oceans	Offers global search capability and versatility.	Low search accuracy. slow convergence speed, quickly falls into the local optimum.
[8]	2019	Butterfly Optimization Algorithm (BOA)	Animals	Butterflies mating habits and their quest for food	Global search capability.	Diminished population diversity, tendency to get trapped in local optimum, slow convergence.
[9]	2019	Harris Hawk Optimization (HHO)	Animals	Cooperative behavior and chasing styles of Harris hawks in nature	Few parameters, simplicity, quick convergence, strong local search capability.	Easily fall into a local optimum, limited exploration capabilities.

(Continued)

Table 1 (continued)

References	Year	Algorithm name	Domain	Inspiration	Advantage	Disadvantage
[22]	2019	Emperor Penguins Colony (EPC)	Animals	Regulation body temperature based on body heat radiation and spiral movement in penguin's colony	Easy to understand, straightforward implementation, adaptable.	Limited to single-objective problems, premature convergence.
[23]	2020	Mayfly Optimization Algorithm (MA)	Animals	Flight and mating behavior of mayflies	Quick convergence, effective exploration.	Weak exploration, stagnation in local optima, low convergence accuracy, lack of proper balance between exploration and exploitation.
[26]	2020	Tunicate Swarm Algorithm (TSA)	Animals	Swarm behaviors of tunicates during the navigation and foraging	Global search capability, simple structure, few parameters, fast iteration.	Easy to fall into a local optimum.
[10]	2020	Political Optimizer (PO)	Human	Human social behavior in a multi-party-political system	Convergence speed and exploitation capability.	Prematurely converges for complex problems.
[11]	2020	Heap-Based Optimizer (HBO)	Human	Interaction between the employees based on the employee hierarchy	Fewer parameters, simple configuration, ease of implementation.	Local stagnation, slow convergence speed, lack of detailed analysis of optimal solutions and a comprehensive search.
[27]	2020	Interactive Autodidactic school (IAS)	Human	Interactions between students in an autodidactic school	Few parameters, adaptable.	Complex, sensitivity to initial knowledge distribution.

(Continued)

Table 1 (continued)

References	Year	Algorithm name	Domain	Inspiration	Advantage	Disadvantage
[28]	2022	Trees Social Relations (TSR)	Plants	Hierarchical and collective life of trees in the jungle	Suitable for both discrete and continuous problems, adaptable, convergence speed.	Complex structure, sensitivity to initial knowledge.
[29]	2021	Golden Eagle Optimizer (GEO)	Animals	Golden eagles hunting process	Global search capability, adaptability, versatility.	Complex implementation, parameter sensitivity, stagnation in local optima.
[30]	2021	COOT	Animals	Animals flock of birds known as Coots	Quick convergence, global search capability, adaptability, versatility.	Stuck in local optima, complex.

3 Inspiration

In natural ecosystems, predators and prey exhibit behaviors that mirror the principles of exploration and exploitation. Predators explore vast territories in search of prey. Once prey is located, predators engage in pursuit, exploiting their position to capture and consume their quarry. Conversely, prey species employ evasive maneuvers to escape predation, contributing to the diversity and adaptability of their movements within the environment. Chase and escaping are the two primary activities involved in group hunting [31]. With the goal of trapping the flock of targets, the chasers' band together and form packs. A cooperative hunting approach is used by lions, which are an example of group chasing. They surround the herd and cover it from all sides before attacking, then they pounce. After they get closer, they pounce on the group. Typically, lions target weaker, older animals, and calves to increase their chances of capturing them [32]. On the other hand, flight initiation distance is a measure of the distance at which prey flee from predators. The distance at which prey flee from predators, on the other hand, is measured by the flight initiation distance. Predator-prey distances come in three varieties: start, alert, and flight initiation. The prey's awareness of the predator is known as the alert distance, the start distance occurs as the predator approaches, and the flight initiation distance occurs when the prey takes flight [33].

Inspired by the division of roles between predators and prey, the CPE algorithm partitions the population into two distinct groups. This approach balances exploration and exploitation. In the first group, the algorithm employs the concept of approaching and encircling prey, which involves identifying and narrowing down promising regions within the search space. In the second group, the algorithm incorporates the concept of pouncing on prey and the prey's escape process. This partitioning fosters a dynamic balance between exploration (via predator-like agents) and exploitation (through prey-like solutions).

Emulating the pursuit tactics of predators, the algorithm guides predator agents towards promising solution regions within the search space. The movement of predators is influenced by factors such as the proximity of prey and the overall distribution of the prey, facilitating comprehensive exploration of the solution landscape. When a predatory agent closes in on a prey-like solution, it executes a "pouncing" action akin to a focused local search. This mechanism enables the algorithm to exploit promising solution regions efficiently, refining candidate solutions and enhancing convergence towards optimal outcomes.

To prevent premature convergence and maintain solution diversity, prey-like solutions employ adaptive escape mechanisms. These mechanisms ensure that solutions dynamically adjust their positions in response to predator movements, thereby preventing over-exploitation of specific solution regions and encouraging continued exploration. This strategy not only enhances the diversity of the solutions but also significantly improves the algorithm's ability to find the global optimum.

Through the integration of these algorithmic strategies, the CPE algorithm achieves an effective balance and synergy between exploration and exploitation. The chase dynamics drive thorough exploration of the solution space, while the pounce mechanism facilitates efficient exploitation of promising solution regions. Additionally, the adaptive escape behavior of prey-like solutions fosters solution diversity, contributing to the algorithm's robust performance across various optimization scenarios.

4 Proposed Chase, Pounce, and Escape Optimization Algorithm

This section contains a thorough explanation of the suggested CPE method, which is based on an image of a group of lions (L) chasing after a group of impalas (I). The three different processes that make up the chase, pounce, and escape operations are approaching and encircling the prey, pouncing the prey, and escaping. Sections 4.2–4.4, respectively, provide a mathematical definition of these procedures.

4.1 Initialization

Initially, as per Eq. (1), a population P of N initial solutions is dispersed randomly across the D-dimensional problem space.

$$P_i^j = r_1 * (UL - LL) + LL, \quad i = 1, 2 \ldots N, j = 1, 2 \ldots D \tag{1}$$

Here, UL and LL denote the upper and lower bounds of the problem domain, respectively. r_1 is a random number in the range (0, 1). Each individual in P corresponds to a group of Lions represented as a D-dimensional vector, with an associated fitness value denoted by $P[I] = \{\{L_1, L_2, \ldots, L_D\}, \mathit{fitness}\}$. The best-solution is kept in I, which stand for an Impalas (group of prey). To carry out the hunting strategy, the population P is divided into two equal groups: The first group is responsible for approaching and encircling the prey, while the second group focuses on attacking the prey. As for the prey, a group of Impalas (I), they attempt to escape from danger by adjusting their positions.

4.2 Approaching and Encircling the Prey

In this procedure, each Lion (L_j) in the first part tries to encircle the prey and guess its Start-Point (SP) using Eq. (2). Where SP_j denotes the starting point of Lion j (L_j), and μ_I denotes the (mean) center of the prey's group (Impalas), which can be calculated using Eq. (3). In Eq. (3), D represents the problem dimensions and indicates the number of individuals in group I. I_j represents the position value of each individual inside the group. Each lion utilizes the SP value and a randomly generated number (r_2) from the range $(-1, 1)$ to adjust its position for encircling the prey from either the left or right. This adjustment can be achieved using Eq. (4). Where L_j^+ in Eq. (4) represents the new position of Lion j. I_j represents Impala j.

$$SP_j = \frac{(\mu_I + L_j)}{2} \tag{2}$$

$$\mu_I = \frac{1}{D} * \sum_{j=1}^{D} I_j \tag{3}$$

$$L_j^+ = (SP_j + I_j) * r_2 \tag{4}$$

to clarify the operation of this part of the population, the algorithm employs historical knowledge stored in the current point L, while also taking into account the overall performance achieved by the optimal point I. The introduction of the mean point μ acts as a balancing factor between exploration and exploitation. The weights of 0.5 assigned to L and μ in Eq. (2) signify their moderate significance, enabling controlled exploration around the current and mean points. In contrast, the weight of 1.0 designated to I in Eq. (4) imparts a more pronounced exploitation element, steering the algorithm towards promising regions within the search space. The inclusion of the random value r_2 enhances the algorithm's adaptable nature, averting premature convergence and promoting exploration of unexplored territories. This stochastic factor plays a pivotal role in breaking free from local optima. Fig. 1 provides an overview of the interplay among the current point L, the mean point μ, and the optimal point I. Meanwhile, Fig. 2 simulates and illustrates the potential movement of the current point L under various scenarios. The X-axis represents dimension 1, and the Y-axis represents dimension 2, with the problem domain ranging from −10 to 10.

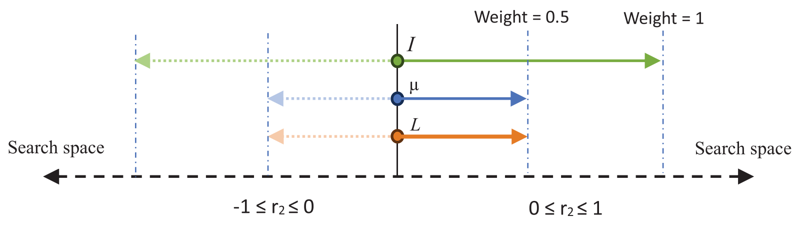

Figure 1: Interaction between the current point L, the mean point μ, and the best point I

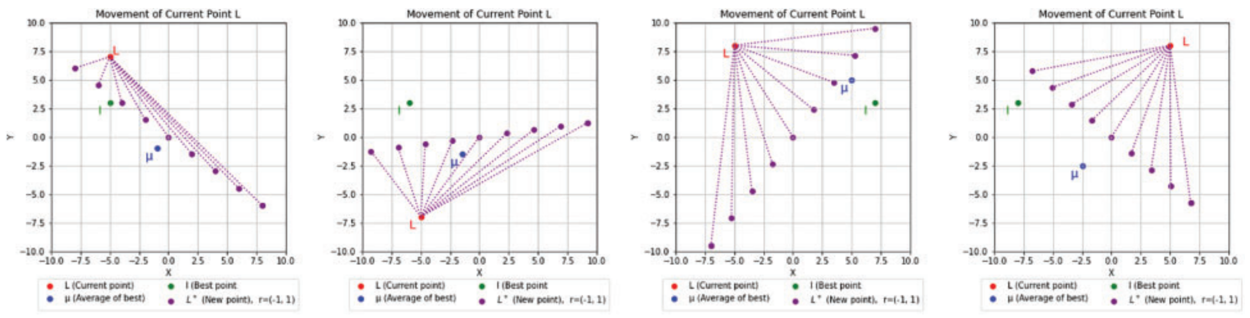

Figure 2: Potential trajectory of the current point L across various scenarios

4.3 Pouncing Process

In this process, the Lions attempt to pounce the prey. Prioritizing the weakest targets. The process for pouncing the weakest prey is defined in Eq. (5), while Eq. (6) defines the procedure for pouncing any accessible prey.

$$L_j^+ = r_3 * \left(L_j - I_{min}\right) + I_{min} \tag{5}$$

$$L_j^+ = r_4 * \left(L_j - I_j\right) + I_j \tag{6}$$

where L_j^+ represents the new position of Lion j. I_{min} represents the position (minimum value) of the weakest prey (Impala) in I. r_3 and r_4 are random numbers in the interval (0, 1). I_j represents the position of Impala j. To switch between Eqs. (5) and (6), a random number is generated. If $random\,(0,1) \leq \mathcal{K}$ then Eq. (5) is used; otherwise, Eq. (6) is used. Here, $\mathcal{K}$ is a real number in the range (0, 1) determined by the user. By offering two equations for position update, the algorithm introduces variability in the lion's movement strategy. This promotes exploration by allowing lions to choose between two different directions for updating their positions based on the current context. The motivation behind this design is to inject randomness into the movement of lions in the second group. This randomness encourages exploration by ensuring that lions do not get stuck in local optima. The algorithm takes a cautious approach Eq. (5) by emphasizing movement towards the minimum value among the best solutions, and a more adventurous approach Eq. (3) by focusing on the current best solution. The random choice of equations enables adaptive exploration based on the current context. Once the attacking process begins, Impalas try to escape using the process described in the next section.

4.4 Escaping Process

Some Impalas evaluate the degree of risk in their surroundings throughout this process. if they believe the danger to be minimal, they cautiously adjust their location within the area of the alert zone, as defined in Eq. (7). On the other hand, if the danger is high, the Impalas instinctively flee from the threat and escape randomly, as described in Eq. (8).

$$I_x^+ = r_5 * \left((I_x + \mathcal{R}) - (I_x - \mathcal{R})\right) + (I_x - \mathcal{R}) \tag{7}$$

$$I_x^+ = r_6 * (UL - LL) + LL \tag{8}$$

Here, I_x^+ denotes the new position of Impala x. I_x is random Impala selected from I. r_5 and r_6 are two random numbers in the range (0, 1). UL and LL represent the upper and the lower boundaries of the problem domain, respectively. $\mathcal{R}$ represents the radius of Impalas warning zone, and it is calculated using Eq. (9).

$$\mathcal{R} = (UL - LL)/k \tag{9}$$

where k represents a constant value specified by the user, or it can be calculated dynamically based on the value of iteration. For this work, we have set the value of k to 200. Offering two equations for updating Impala positions introduces diversity in the movement of lions, further enhancing exploration capabilities. The use of factor $\mathcal{R}$ adds controlled variability to the movement range. This strategy allows lions to explore different modes of movement when updating their positions based on Impala information. Eq. (7) involves a perturbation around the selected Impala's position, mimicking the idea of following a potential prey's movements while maintaining a connection to the current best solution. Eq. (8), on the other hand, allows for wilder exploration by randomly moving to different areas within the solution space. This combination of strategies balances the benefits of exploiting successful solutions while also exploring novel regions. The complete CPE algorithm is outlined in

Algorithm 1, and Fig. 3 presents the Flowchart. The proposed CPE algorithm is available on GitHub at: https://github.com/Adel-Sabry/CPE-optimization-algorithm.git (accessed on 6 April 2024).

Algorithm 1: CPE algorithm

```
Input: $P[N]$ (population $P$ of $N$ solutions), $UL$ (Upper Limit of the problem),
        $LL$ (Lower Limit of the problem), $T$ (max iteration).
Output: Best solution (group of Impalas)
Method 1: /*Initialization: Initialize $P$ with random $N$ solutions.
    -   foreach $P_i \in P$ do
            $r_1 \leftarrow random\ (0, 1)$
            $P_i = initialize\ (UL, LL, r_1)$ using Eq. (1).
        endforeach
    -   Evaluate each solution in $P$ and keep the best in $I$.
    -   $\mathcal{R} \leftarrow (UL - LL)/200$      // $\mathcal{R}$ (Radius of alert zone for each impala).
    -   $z \leftarrow N/2$.                     // Divide the population into two groups.
    -   $t \leftarrow 1$                        // initial iteration.
Method 2:
While $t \le T$ do
        /*Process 1: approach and encircle the prey*/
        $\mu_I \leftarrow CalculateMeanOf\_I(I)$ using Eq. (3)
        for $i \leftarrow 0$ to $z$ do
            foreach $L_{i,j} \in L_i$ do
                $SP_j \leftarrow Claculate\_SP\ (\mu_I, L_{i,j})$ using Eq. (2)
                /*Update position*/
                $r_2 \leftarrow random\ (-1, 1)$
                $L^+_{i,j} \leftarrow Update\_Position(I_j, SP_j, r_2)$ using Eq. (4)
            endforeach
            if($L^+_i$ is better than $L_i$)
                    then $L_i \leftarrow L^+_i$
            if($L^+_i$ is better than $I$)
                    then $I \leftarrow L^+_i$
        endfor
        /*process 2: Pounce*/
        for $i \leftarrow z$ to $N$ do
            foreach $L_{i,j} \in L_i$ do
                if ($random\ (0, 1) \ge 0.5$ /* Pounce the weakest Impala
                    then
                    $r_3 \leftarrow random\ (0, 1)$
                    $L^+_{i,j} \leftarrow Pounce\ (L_{i,j}, I_{min}, r_3)$
                else /* Pounce any Impala*/
                    r4 $\leftarrow random\ (0, 1)$
                    $L^+_{i,j} \leftarrow Pounce\ (L_{i,j}, I_j, r_4)$ using Eq. (6)
            endforeach
            if($L^+_i$ is better than $L_i$)
                    then $L_i \leftarrow L^+_i$
```

(Continued)

Algorithm 1 (continued)

```
            if($L_i^+$ is better than $I$)
                    then $I \leftarrow L_i^+$
            /*Process 3: Escape*/
            $I_x \leftarrow$ select randomly one Impala from $I$
            $danger \leftarrow random\ (0, 1)$
            if ($danger \le 0.5$) /*update the position using $\mathcal{R}$*/
                    then
                    $r_5 \leftarrow random\ (0, 1)$
                    $I^+ \leftarrow Update\ (I_x, \mathcal{R}, r_5)$ using Eq. (7)
            else
                    /*Escape randomly*/
                    $r_6 \leftarrow random\ (0, 1)$
                    $I^+ \leftarrow Run\ (I_x, UL, LL, r_6)$ using Eq. (8)
             if($I^+$ is better than $I$)
                      then $I \leftarrow I^+$
    endfor
endwhile
```

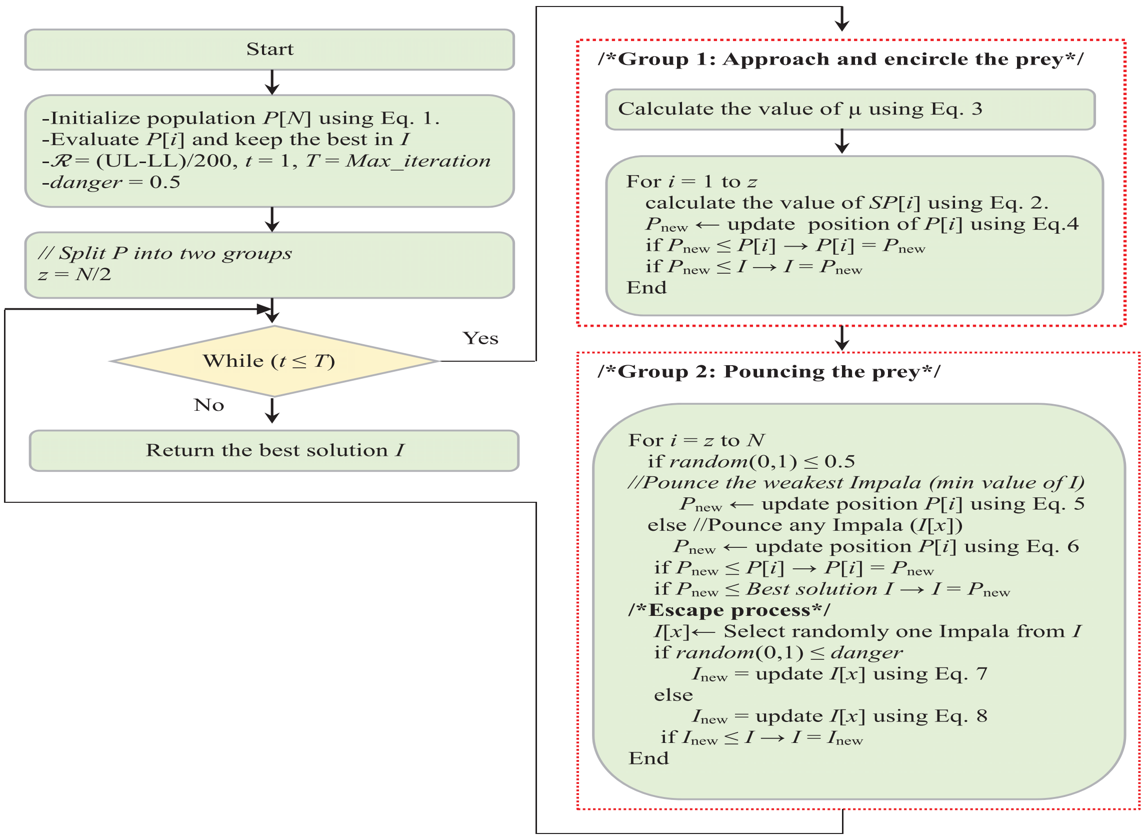
Start
-Initialize population $P[N]$ using Eq. 1.
-Evaluate $P[i]$ and keep the best in I
-$\mathcal{R}$ = (UL-LL)/200, t = 1, T = $Max\_iteration$
-$danger$ = 0.5
// Split P into two groups
$z = N/2$
While ($t \le T$)
Yes
No
Return the best solution I
/*Group 1: Approach and encircle the prey*/
Calculate the value of μ using Eq. 3
For i = 1 to z
calculate the value of $SP[i]$ using Eq. 2.
$P_{new} \leftarrow$ update position of $P[i]$ using Eq.4
if $P_{new} \le P[i] \rightarrow P[i] = P_{new}$
if $P_{new} \le I \rightarrow I = P_{new}$
End
/*Group 2: Pouncing the prey*/
For i = z to N
if $random(0,1) \le 0.5$
//Pounce the weakest Impala (min value of I)
$P_{new} \leftarrow$ update position $P[i]$ using Eq. 5
else //Pounce any Impala ($I[x]$)
$P_{new} \leftarrow$ update position $P[i]$ using Eq. 6
if $P_{new} \le P[i] \rightarrow P[i] = P_{new}$
if $P_{new} \le$ Best solution $I \rightarrow I = P_{new}$
/*Escape process*/
$I[x] \leftarrow$ Select randomly one Impala from I
if $random(0,1) \le danger$
I_{new} = update $I[x]$ using Eq. 7
else
I_{new} = update $I[x]$ using Eq. 8
if $I_{new} \le I \rightarrow I = I_{new}$
End

Figure 3: Flowchart diagram of the proposed CPE algorithm

4.5 Time Complexity Analysis

The time complexity can generally be calculated using the formula $O(NET)$, where N is the population size, E is the time complexity of the evaluation function, and T is the number of iterations. The suggested CPE algorithm divides the population size into two groups. The first group uses (4) to operate one evaluation function every iteration; as a result, the time complexity of this group is $O(\frac{NET}{2})$. In contrast, the second group uses (7) and (8) to run two evaluation functions in each iteration of the two processes (attack and escape). As a result, the time complexity of this group is determined to be $O\left(\frac{2NET}{2}\right) = O(NET)$. The suggested CPE has an $O(\frac{NET}{2} + NET)$ total time complexity.

5 Experimental Results

5.1 Experiment 1: Using 50 Test Functions

This experiment utilized 50 commonly used test functions to assess the performance of the proposed algorithm. The characteristics of these functions are extensively described in Table A1 in Appendix A. Specifically, functions 1–25 are unimodal and lack local optima, making them suitable for evaluating the algorithm's exploitation ability. Meanwhile, functions 26–50 are multimodal and have numerous local optima, enabling the evaluation of the algorithm's exploration capability. For more details about the mathematical formulas of these test functions, the reader can refer to reference [10]. The performance of the proposed algorithm is compared to 11 well-known algorithms, including WOA, SCA, GWO, Soccer League Competition (SLC), Krill Herd (KH), TLBO, GSA, CS, Biogeography Based Optimizer (BBO), Differential Evolution (DE), and PSO. These algorithms were selected because they represent a diverse range of nature-inspired metaheuristics that have been widely used and validated in the field of optimization. Each algorithm is executed 25 times for each test function, and Mean and Standard Deviation (SD) are recorded as metrics. Results with a value lower than the specified tolerance, $\delta = \mathrm{E} - 11$, are considered to be 0 during the evaluation. To ensure a fair comparison, the Number of Function Evaluations (NFEs) is set to 30,000 for all algorithms. The names of the algorithms and their parameter settings are presented in Table 2.

Table 2: Optimization algorithms' names and their parameters

Algorithm	Parameters	Algorithm	Parameters
CPE	*p-size* $= 20$, $\mathcal{R} =$ UP-LL/200, r_2 range $(-1, 1)$, *danger* condition ≤ 0.5, r_1, r_3–r_6 are random number between (0, 1)	TLBO	*p-size* $= 50$
WOA	*p-size* $= 30$, $b = 1$, (a, C, l) from corresponding equation	GSA	*p-size* $= 50$, $a = 20$, $G_0 = 100$, k (*p-size* to 1),
SCA	*p-size* $= 50$, $a = 2$, (l, r_2, r_3, r_4) from corresponding equation	CS	*nests* $= 20$, $pa = 0.25$
GWO	*p-size* $= 50$, $\{a, C\}$ from corresponding equation	BBO	*p-size* $= 50$, *hmp* $= 1$, *elit* $= 2$, *st-size* $= 1$, *mir* $= 1$, *mer* $= 1$, *mt* ≥ 0.05
SLC	*n-teams* $= 5$, *n-fixed-player* $= 11$, *n-substitute* $= 11$	DE	*p-size* $= 20$, $CR = 0.5$, $F = 0.5$
KH	*p-size* $= 50$, $v_f = 0.02$, $D_{max} = 0.005$, $N_{max} = 0.01$, $S_r = 0$	PSO	*p-size* $= 50$, $c_1 = 2$, $c_2 = 2$, w (0.2 to 0.9)

The Mean and SD results for functions 1–25 in the exploitation test and functions 26–50 in the exploration test are shown in Tables 3 and 4, respectively. The best result is highlighted in bold in the tables. Table 5 presents the performance ranks across a comprehensive set of 50 test functions. The evaluation employs Freidman mean rank values as a means to consolidate the algorithmic performances, providing a collective perspective on their effectiveness across all functions. Results indicate that the proposed CPE algorithm demonstrates superior performance on certain test functions, obtaining the top rank (ranked first) in 9 out of the 50 functions, specifically F2, F6, F13, F17, F26, F29, F33, F34, and F39. For the remaining 41 test functions, the proposed CPE algorithm exhibits either superior or comparable performance in comparison to the other algorithms, except for functions F20 and F25 where the proposed algorithm obtains ranks of 9 and 7, respectively. The strengths of the CPE algorithm lie in its innovative partitioning strategy, which effectively enhances exploration and exploitation dynamics. This allows it to perform well on both unimodal and multimodal functions. However, it faces challenges with certain functions (e.g., F20 and F25), indicating areas where parameter tuning or additional mechanisms might be needed to improve performance. Other algorithms, such as GWO and PSO, also demonstrated strong performance, particularly in specific types of functions where their specialized strategies are highly effective. For example, PSO excels in exploitation due to its velocity and position update mechanisms. However, these algorithms may have limitations in maintaining diversity and avoiding local optima, which can impact their performance on more complex multimodal functions. Furthermore, the mean rank analysis reveals that the proposed CPE algorithm achieves better mean ranks of approximately 3.72 and 3.7 for the first and second sets of test functions, respectively. These results provide valuable insights into the algorithm's overall performance across the entire range of test functions. The CPE algorithm's strengths lie in its innovative partitioning strategy, which enhances both exploration and exploitation dynamics. However, there is potential for improvement in fine-tuning the parameters or exploring additional strategies to address functions where performance was less optimal.

Table 3: Results of the proposed CPE against other 11 algorithm using (F1–F25)

Fun.	Index	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO
F1	*Mean*	**0.0E+00**	**0.0E+00**	1.6E+02	**0.0E+00**	**0.0E+00**	1.3E+00	**0.0E+00**	6.6E−02	1.0E+02	1.1E+02	3.5E−10	9.7E−03
	SD	**0.0E+00**	**0.0E+00**	1.8E+02	**0.0E+00**	**0.0E+00**	6.4E−01	**0.0E+00**	1.7E−01	2.6E+01	3.8E+01	3.1E−10	1.3E−02
F2	*Mean*	**5.7E−05**	1.8E−03	2.0E+00	1.5E−03	5.8E−04	1.3E−01	9.2E−04	8.1E−02	1.6E−01	3.9E−03	4.6E−02	2.2E+01
	SD	**7.1E−05**	1.6E−03	2.3E+00	8.3E−04	3.7E−04	5.5E−02	2.9E−04	3.7E−02	5.21E−02	4.4E−03	1.2E−02	1.9E+01
F3	*Mean*	**0.0E+00**	**0.0E+00**	2.6E−03	**0.0E+00**	**0.0E+00**	2.1E−07	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.3E−07
	SD	**0.0E+00**	**0.0E+00**	3.7E−03	**0.0E+00**	**0.0E+00**	2.8E−07	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	3.4E−07
F4	*Mean*	**0.0E+00**	**0.0E+00**	2.0E+00	**0.0E+00**	**0.0E+00**	5.3E+00	**0.0E+00**	1.5E+01	6.4E+01	4.1E+01	8.6E−06	2.5E−01
	SD	**0.0E+00**	**0.0E+00**	1.7E+00	**0.0E+00**	**0.0E+00**	7.2E+00	**0.0E+00**	1.2E+01	6.2E+00	6.6E+00	3.6E−06	1.4E−01
F5	*Mean*	**0.0E+00**	7.5E+01	6.2E+01	1.6E−06	**0.0E+00**	5.1E+00	**0.0E+00**	7.4E+00	1.6E+01	5.3E+01	1.8E+01	2.3E+00
	SD	**0.0E+00**	2.3E+01	8.8E+00	1.6E−06	**0.0E+00**	1.2E+00	**0.0E+00**	1.2E+00	1.6E+00	8.2E+00	4.4E+00	3.2E−01
F6	*Mean*	**0.0E+00**	4.2E−01	3.6E+02	1.7E+00	7.0E−02	1.4E+00	1.8E−05	8.3E−01	9.7E+01	1.0E+02	2.0E−10	9.0E−03
	SD	**0.0E+00**	1.3E−01	3.5E+02	5.5E−01	5.7E−02	6.8E−01	5.3E−05	2.4E+00	2.3E+01	3.3E+01	1.5E−10	1.1E−02
F7	*Mean*	**−275.00**	**−275.00**	−149.80	−219.12	−265.72	−79.52	−271.64	−180.40	−233.04	−257.64	−274.96	−199.76
	SD	**0.0E+00**	**0.00**	8.09	9.36	14.88	9.38	4.32	5.64	5.91	3.38	0.20	20.99
F8	*Mean*	**0.0E+00**	1.3E+05	3.9E+04	3.1E−04	**0.0E+00**	3.7E+03	**0.0E+00**	1.5E+03	1.3E+04	3.6E+04	8.6E+04	7.9E+02
	SD	**0.0E+00**	2.9E+04	1.3E+04	5.7E−04	**0.0E+00**	1.5E+03	**0.0E+00**	4.4E+02	3.2E+03	7.0E+03	8.3E+03	2.1E+02
F9	*Mean*	**0.0E+00**	**0.0E+00**	1.3E+00	**0.0E+00**	**0.0E+00**	3.5E+59	**0.0E+00**	2.7E+02	1.0E+10	3.6E+01	6.7E−05	2.6E+00
	SD	**0.0E+00**	**0.0E+00**	1.5E+00	**0.0E+00**	**0.0E+00**	1.8E+60	**0.0E+00**	3.5E+01	0.0E+00	6.6E+00	8.2E−05	2.2E+00
F10	*Mean*	**0.0E+00**	**0.0E+00**	1.7E+08	**0.0E+00**	**0.0E+00**	5.8E−08	**0.0E+00**	3.5E−03	9.3E+02	1.2E−01	7.6E+03	7.8E−01
	SD	**0.0E+00**	**0.0E+00**	3.5E+08	**0.0E+00**	**0.0E+00**	8.5E−08	**0.0E+00**	1.0E−02	1.5E+03	3.3E−01	3.8E+04	1.5E+00
F11	*Mean*	4.6E+01	4.8E+01	2.5E+06	4.7E+01	4.6E+01	2.5E+02	4.3E+01	1.3E+02	4.5E+03	4.3E+03	7.7E+01	3.1E+02
	SD	0.14E−01	5.3E−01	3.1E+06	7.3E−01	5.5E−01	9.6E+01	6.3E−01	7.8E+01	1.6E+03	2.1E+03	4.5E+01	3.1E+02
F12	*Mean*	**0.0E+00**	**0.0E+00**	9.7E−02	**0.0E+00**	**0.0E+00**	2.0E+02	**0.0E+00**	5.5E+00	1.6E+00	**0.0E+00**	**0.0E+00**	1.8E+02

(Continued)

Table 3 (continued)

Fun.	Index	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO
	SD	**0.0E+00**	**0.0E+00**	1.5E−01	**0.0E+00**	**0.0E+00**	1.1E+02	**0.0E+00**	2.4E+00	9.8E−01	**0.0E+00**	**0.0E+00**	1.7E+02
F13	*Mean*	0.50	0.67	16298.56	0.67	0.67	9.92	0.67	4.21	67.82	237.28	2.34	7617.05
	SD	2.0E−05	0.00	2.7E+04	0.00	0.0E+00	6.04	0.00	2.86	22.28	167.45	2.54	26953.37
F14	*Mean*	**0.0E+00**	**0.0E+00**	2.3E+02	1.7E−05	**0.0E+00**	2.8E+00	7.6E−08	3.1E+00	8.6E+00	4.5E+01	5.8E−01	3.4E+03
	SD	**0.0E+00**	**0.0E+00**	2.1E+02	1.8E−05	**0.0E+00**	9.2E−01	2.5E−07	2.8E+00	2.7E+00	1.3E+01	9.1E−01	1.8E+03
F15	*Mean*	**0.0E+00**	9.3E+02	9.3E+01	2.2E−06	**0.0E+00**	4.0E+02	5.4E−03	1.2E+02	6.0E+02	2.9E+02	4.9E+02	6.9E+02
	SD	**0.0E+00**	2.6E+02	3.7E+01	4.3E−06	**0.0E+00**	1.2E+02	8.2E−03	2.1E+01	7.4E+01	5.7E+01	6.6E+01	1.4E+02
F16	*Mean*	0.0E+00	−3.2E−01	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00
	SD	0.0E+00	4.8E−01	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00
F17	*Mean*	**0.02**	514.73	44.40	571.13	19.78	97.61	0.72	578.76	2.40	135.89	1.70	505.14
	SD	**0.02**	585.33	29.43	1473.63	33.11	262.51	0.74	543.73	1.60	200.14	2.02	1746.47
F18	*Mean*	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.7E−11	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	SD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.5E−11	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F19	*Mean*	**0.0E+00**	4.4E−11	1.2E−04	3.0E−02	**0.0E+00**	1.1E−10	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.8E−01	**0.0E+00**	**0.0E+00**
	SD	**0.0E+00**	1.4E−10	1.3E−04	1.5E−01	**0.0E+00**	9.6E−11	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.2E−01	**0.0E+00**	**0.0E+00**
F20	*Mean*	1.7E−10	1.7E−04	6.0E−04	1.8E−07	**0.0E+00**	9.9E−11	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	SD	3.0E−10	1.5E−04	5.9E−04	1.6E−07	**0.0E+00**	1.2E−10	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F21	*Mean*	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.3E−04	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	SD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	2.7E−04	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F22	*Mean*	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	SD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F23	*Mean*	**0.29**	**0.29**	**0.29**	**0.29**	**0.29**	**0.29**	**0.29**	0.31	**0.29**	0.30	**0.29**	**0.29**
	SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00
F24	*Mean*	**19.11**	**19.11**	19.13	**19.11**	**19.11**	**19.11**	**19.11**	**19.11**	**19.11**	20.44	**19.11**	**19.11**
	SD	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F25	*Mean*	2.4E−08	1.3E−05	2.1E−04	4.8E−07	**0.0E+00**	8.5E−03	**0.0E+00**	5.1E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	SD	4.3E−08	1.9E−05	1.8E−04	5.8E−07	**0.0E+00**	9.5E−03	**0.0E+00**	6.0E−03	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**

Table 4: Results of the proposed CPE algorithm against other 11 algorithms using function F26 to F50

Fun.	Index	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO
F26	*Mean*	4.5E−02	9.4E+01	3.2E+02	2.2E+02	2.0E+02	2.2E+02	1.9E+02	3.5E+02	1.7E+02	4.2E+00	2.0E+02	2.2E+02
	Std	1.6E−02	5.4E+01	7.4E+00	1.7E+01	4.0E+01	2.4E+01	3.8E+01	1.1E+01	5.5E+00	1.2E+00	2.1E+01	4.4E+01
F27	*Mean*	**0.0E+00**	**0.0E+00**	7.8E+01	2.0E+00	**0.0E+00**	2.5E+01	9.8E+00	3.6E+01	2.1E+02	**0.0E+00**	3.0E+02	2.5E+02
	Std	**0.0E+00**	**0.0E+00**	4.8E+01	3.1E+00	**0.0E+00**	1.2E+01	1.0E+01	8.5E+00	1.7E+01	**0.0E+00**	1.6E+01	4.6E+01
F28	*Mean*	**9.0E−01**	1.0E+00	1.1E+01	1.6E+00	1.7E+00	5.3E+00	5.6E+00	1.0E+00	2.5E+00	**9.0E−01**	9.4E+00	5.7E+00
	Std	4.0E−02	2.4E−01	2.2E+00	3.4E−01	6.0E−01	4.6E+00	3.7E+00	5.4E−16	1.2E−01	2.0E−02	5.2E−01	1.9E+00
F29	*Mean*	**0.6E+00**	5.3E+03	2.1E+09	8.4E+03	3.0E+02	2.2E+03	2.8E+00	2.7E+08	1.0E+10	9.0E+05	1.5E+03	8.1E−01
	Std	**0.3E+00**	2.3E+03	2.4E+09	2.3E+03	2.3E+02	9.3E+02	1.4E+01	1.2E+08	0.0E+00	5.0E+05	1.2E+03	1.2E+00
F30	*Mean*	**0.0E+00**	**0.0E+00**	5.4E+00	5.1E−04	**0.0E+00**	2.0E−01	**0.0E+00**	8.6E−03	2.5E+01	1.6E−02	3.3E−02	1.3E+00
	Std	**0.0E+00**	**0.0E+00**	5.4E+00	9.4E−04	**0.0E+00**	4.4E−01	**0.0E+00**	1.0E−02	2.2E+00	5.4E−02	8.6E−03	2.3E+00
F31	*Mean*	2.1E−09	1.0E+04	4.2E+07	6.6E−23	8.0E−25	3.0E+13	**0.0E+00**	1.2E−01	1.0E+10	1.4E+15	2.3E+02	5.2E+15
	Std	4.5E−09	5.2E+04	2.1E+08	3.3E−22	2.8E−24	1.2E+14	**0.0E+00**	1.8E−01	0.0E+00	6.9E+15	8.3E+02	1.7E+16
F32	*Mean*	**0.0E+00**	**0.0E+00**	1.8E+01	**0.0E+00**	**0.0E+00**	2.8E+00	**0.0E+00**	3.0E−02	1.0E+01	1.9E+00	3.8E−06	7.9E−01
	Std	**0.0E+00**	**0.0E+00**	5.2E+00	**0.0E+00**	**0.0E+00**	7.1E−01	**0.0E+00**	1.0E−01	1.9E+00	2.8E−01	1.7E−06	6.7E−01
F33	*Mean*	2.9E+00	1.3E+02	1.0E+04	5.7E+01	9.8E+01	1.4E+02	3.0E+01	7.4E+04	3.3E+03	2.8E+03	1.5E+01	5.1E+01
	Std	1.3E+00	3.4E+01	1.2E+04	1.0E+01	2.6E+01	4.1E+01	5.6E+00	1.0E+04	8.4E+02	7.4E+02	1.7E+01	1.4E+01
F34	*Mean*	9.6E−02	1.4E−01	2.5E+00	2.1E−01	1.0E−01	2.7E+00	1.6E−01	4.4E+00	4.7E+00	3.6E+00	5.2E−01	6.6E−01
	Std	2.7E−02	5.7E−02	1.4E+00	2.8E−02	1.3E−07	4.5E−01	4.9E−02	5.2E−01	4.2E−01	4.2E−01	4.6E−02	5.8E−02
F35	*Mean*	−**2.0E+03**	−1.9E+03	−8.9E+02	−1.4E+03	−1.7E+03	−1.6E+03	−1.7E+03	−1.8E+03	−1.6E+03	−**2.0E+03**	−1.9E+03	−1.7E+03
	Std	2.5E−04	1.1E+02	7.4E+01	8.7E+01	4.5E+01	5.6E+01	5.7E+01	2.9E+01	2.2E+01	2.7E−01	2.1E+01	5.6E+01
F36	*Mean*	**0.0E+00**	8.8E−03	9.4E−01	1.6E−03	**0.0E+00**	1.8E−01	**0.0E+00**	5.3E−02	9.8E−01	1.0E+00	5.9E−04	5.2E−03
	Std	**0.0E+00**	3.0E−02	2.3E−01	4.7E−03	**0.0E+00**	8.0E−02	**0.0E+00**	7.3E−02	4.4E−02	4.0E−02	2.0E−03	5.6E−03
F37	*Mean*	0.0E+00	−2.4E−01	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00
	Std	0.0E+00	4.4E−01	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00	0.0E+00
F38	*Mean*	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.6E−08	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.3E−07	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**

(Continued)

Table 4 (continued)

Fun.	Index	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO
F39	*Mean*	3.8E−04	7.8E−01	1.6E+07	1.5E+00	1.9E+00	2.1E+01	9.0E−02	2.2E+01	3.1E+01	3.7E+00	1.7E−01	1.8E−02
	Std	2.3E−04	3.7E−01	2.3E+07	4.0E−01	1.4E+00	1.3E+01	9.7E−02	5.8E+00	6.1E+00	9.1E−01	7.3E−01	1.6E−02
F40	*Mean*	1.3E−05	1.1E−02	4.3E+06	5.9E−02	1.6E−03	2.0E+00	1.4E−07	1.5E+00	5.0E+00	5.4E−01	1.1E−03	7.6E−03
	Std	6.6E−06	6.4E−03	7.1E+06	2.0E−02	1.4E−03	8.1E−01	2.1E−07	5.4E−01	7.6E−01	1.7E−01	5.3E−03	2.1E−02
F41	*Mean*	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**
F42	*Mean*	**−195.63**	**−195.63**	**−195.63**	**−195.63**	**−195.63**	**−195.63**	**−195.63**	**−195.63**	**−195.63**	−194.57	**195.63**	**−195.63**
	Std	**−195.63**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.7E−01	**0.0E+00**	**0.0E+00**
F43	*Mean*	**−2.02**	**−2.02**	**−2.02**	**−2.02**	**−2.02**	**−2.02**	**−2.02**	**−2.02**	**−2.02**	−2.01	**−2.02**	**−2.02**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.0E−02	**0.0E+00**	**0.0E+00**
F44	*Mean*	**−106.76**	**−106.76**	−106.71	−105.99	**−106.76**	−105.99	**−106.76**	−106.47	**−106.76**	−100.41	**106.76**	**−106.76**
	Std	**0.0E+00**	**0.0E+00**	5.0E−02	3.9E+00	**0.0E+00**	3.9E+00	**0.0E+00**	6.1E−01	**0.0E+00**	5.0E+00	**0.0E+00**	**0.0E+00**
F45	*Mean*	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	−0.90	**−1.03**	**−1.03**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.5E−01	**0.0E+00**	**0.0E+00**
F46	*Mean*	**0.40**	**0.40**	**0.40**	**0.40**	**0.40**	**0.40**	**0.40**	**0.40**	**0.40**	0.47	**0.40**	**0.40**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.0E−02	**0.0E+00**	**0.0E+00**
F47	*Mean*	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	−3.82	**−3.86**	**−3.86**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	6.0E−02	**0.0E+00**	**0.0E+00**
F48	*Mean*	−3.31	−3.27	−2.95	−3.28	−3.28	−3.28	**−3.32**	**−3.32**	**−3.32**	−3.08	−3.28	−3.25
	Std	3.9E−02	7.0E−02	2.8E−01	7.0E−02	6.0E−02	6.0E−02	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.8E−01	6.0E−02	8.0E−02
F49	*Mean*	**−2.06**	**−2.06**	**−2.06**	**−2.06**	**−2.06**	**−2.06**	**−2.06**	**−2.06**	**−2.06**	−2.05	**−2.06**	**−2.06**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	1.0E−02	**0.0E+00**	**0.0E+00**
F50	*Mean*	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	1.19	**1.00**	**1.00**
	Std	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	**0.0E+00**	4.6E−01	**0.0E+00**	**0.0E+00**

Table 5: Rank of each algorithm in each test function and Freidman mean-rank

Fun.	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO	Fun.	CPE	WOA	SCA	GWO	SLC	KH	TLBO	GSA	CS	BBO	DE	PSO
F1	3	3	12	3	3	9	3	8	10	11	6	7	F26	1	3	11	9	6.5	9	5	12	4	2	6.5	9
F2	1	5	11	4	2	9	3	8	10	6	7	12	F27	2.5	2.5	9	5	2.5	7	6	8	10	2.5	12	11
F3	5	5	12	5	5	11	5	5	5	5	5	10	F28	1.5	3.5	12	5	6	8	9	3.5	7	1.5	11	10
F4	3	3	8	3	3	9	3	10	12	11	6	7	F29	1	7	11	8	4	6	3	10	12	9	5	2
F5	2	12	11	4	2	6	2	7	8	10	9	5	F30	2.5	2.5	11	5	2.5	9	2.5	6	12	7	8	10
F6	1	6	12	9	5	8	3	7	10	11	2	4	F31	4	7	8	3	2	10	1	5	9	11	6	12
F7	1.5	1.5	11	8	5	12	4	10	7	6	3	9	F32	3	3	12	3	3	10	3	7	11	9	6	8
F8	2	12	10	4	2	7	2	6	8	9	11	5	F33	1	7	11	5	6	8	3	12	10	9	2	4
F9	3	3	7	3	3	12	3	10	11	9	6	8	F34	1	3	8	5	2	9	4	11	12	10	6	7
F10	3	3	12	3	3	6	3	7	10	8	11	9	F35	1.5	3.5	12	11	7	9.5	7	5	9.5	1.5	3.5	7
F11	2.5	5	12	4	2.5	8	1	7	11	10	6	9	F36	2	7	10	5	2	9	2	8	11	12	4	6
F12	4	4	8	4	4	12	4	10	9	4	4	11	F37	7	1	7	7	7	7	7	7	7	7	7	7
F13	1	3.5	12	3.5	3.5	8	3.5	7	9	10	6	11	F38	6	6	6	6	6	12	6	6	6	6	6	6
F14	2	2	11	5	2	7	4	8	9	10	6	12	F39	1	5	12	6	7	9	3	10	11	8	4	2
F15	1.5	12	5	3	1.5	8	4	6	10	7	9	11	F40	2	6	12	7	4	10	1	9	11	8	3	5
F16	7	1	7	7	7	7	7	7	7	7	7	7	F41	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
F17	1	10	6	11	5	7	2	12	4	8	3	9	F42	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5	11	12	5.5
F18	6	6	6	6	6	12	6	6	6	6	6	6	F43	6	6	6	6	6	6	6	6	6	12	6	6
F19	4	8	10	11	4	9	4	4	4	12	4	4	F44	3.5	3.5	7	9.5	3.5	9.5	3.5	8	3.5	11	12	3.5
F20	9	11	12	10	4	8	4	4	4	4	4	4	F45	6	6	6	6	6	6	6	6	6	12	6	6
F21	6	6	6	6	6	6	6	12	6	6	6	6	F46	6	6	6	6	6	6	6	6	6	12	6	6
F22	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	F47	6	6	6	6	6	6	6	6	6	12	6	6
F23	5.5	5.5	5.5	5.5	5.5	5.5	5.5	12	5.5	11	5.5	5.5	F48	4	9	12	6.5	6.5	6.5	2	2	2	11	6.5	10
F24	5.5	5.5	11	5.5	5.5	5.5	5.5	5.5	5.5	12	5.5	5.5	F49	6	6	6	6	6	6	6	6	6	12	6	6
F25	7	9	10	8	3.5	12	3.5	11	3.5	3.5	3.5	3.5	F50	6	6	6	6	6	6	6	6	6	12	6	6
Mean rank	3.72	5.94	9.36	5.68	3.98	8.42	3.9	7.84	7.64	8.12	5.92	7.48	Mean rank	3.7	5.1	8.76	6.16	5.02	7.86	4.64	7.1	7.84	8.6	6.52	6.7

The convergence behaviour is another important feature of an optimization method. It involves the movement of solutions along the search space towards the best answer. To observe the convergence behaviour of the CPE algorithm, Fig. 4 illustrates the search history and trajectory of 12 agents. The first column depicts the 2-D version of the function, the second column shows the search history and trajectory of 12 agents, the third column displays the trajectory of a randomly selected agent in its first dimension, and the fourth column represents the average convergence of all search agents. It is evident that the CPE algorithm's search history extensively explores promising regions of the search space, avoiding local minima traps, and effectively exploiting the best solution. The trajectory of the first dimension of the agent in column 3 shows that the variables lead to the best value in early iterations. Furthermore, Fig. 5 displays the convergence curves of the CPE algorithm compared to other algorithms, including DE, PSO, CS, GSA, GWO, and TLBO, for some unimodal and multimodal benchmark functions. The curves are plotted against the number of iterations, with a maximum of 150. For unimodal functions (1–25), the curves indicate that the CPE algorithm outperformed all other algorithms, highlighting its notable ability to rapidly explore and exploit promising areas. In the case of multimodal functions (26–50) in Fig. 5, the experimental results demonstrate that the CPE algorithm performs better than all other algorithms, except for GWO and PSO in the case of F26, and PSO in the case of F41. Nevertheless, the CPE algorithm consistently exhibits a pronounced capacity to escape local optima. These empirical observations underscore the algorithm's adaptability, which enables it to dynamically adjust its exploration strategies in accordance with the contextual requirements, manifesting a multifaceted spectrum of exploration and exploitation techniques. The first group within its methodology amalgamates lion positions, optimal solutions, and averaging techniques, thereby achieving a judicious equilibrium between intensification and diversification. In contrast, the second group introduces variance into lion movement patterns, thereby promoting exploratory behaviours while effectively mitigating stagnation. The algorithm, fundamentally, optimizes solutions through a synthesis of individual experiences, communal knowledge, and controlled stochasticity. Consequently, we assert that the CPE algorithm excels in probing the search space, showcasing rapid convergence toward optimal solutions, and possesses commendable exploration capabilities. These attributes collectively empower it to adeptly circumvent local minima.

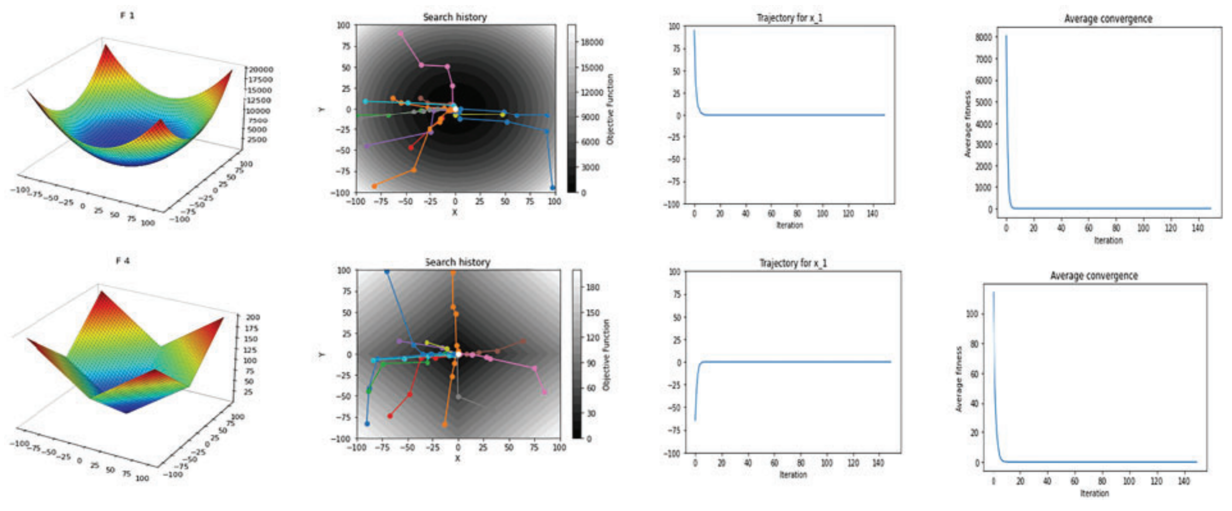

Figure 4: (Continued)

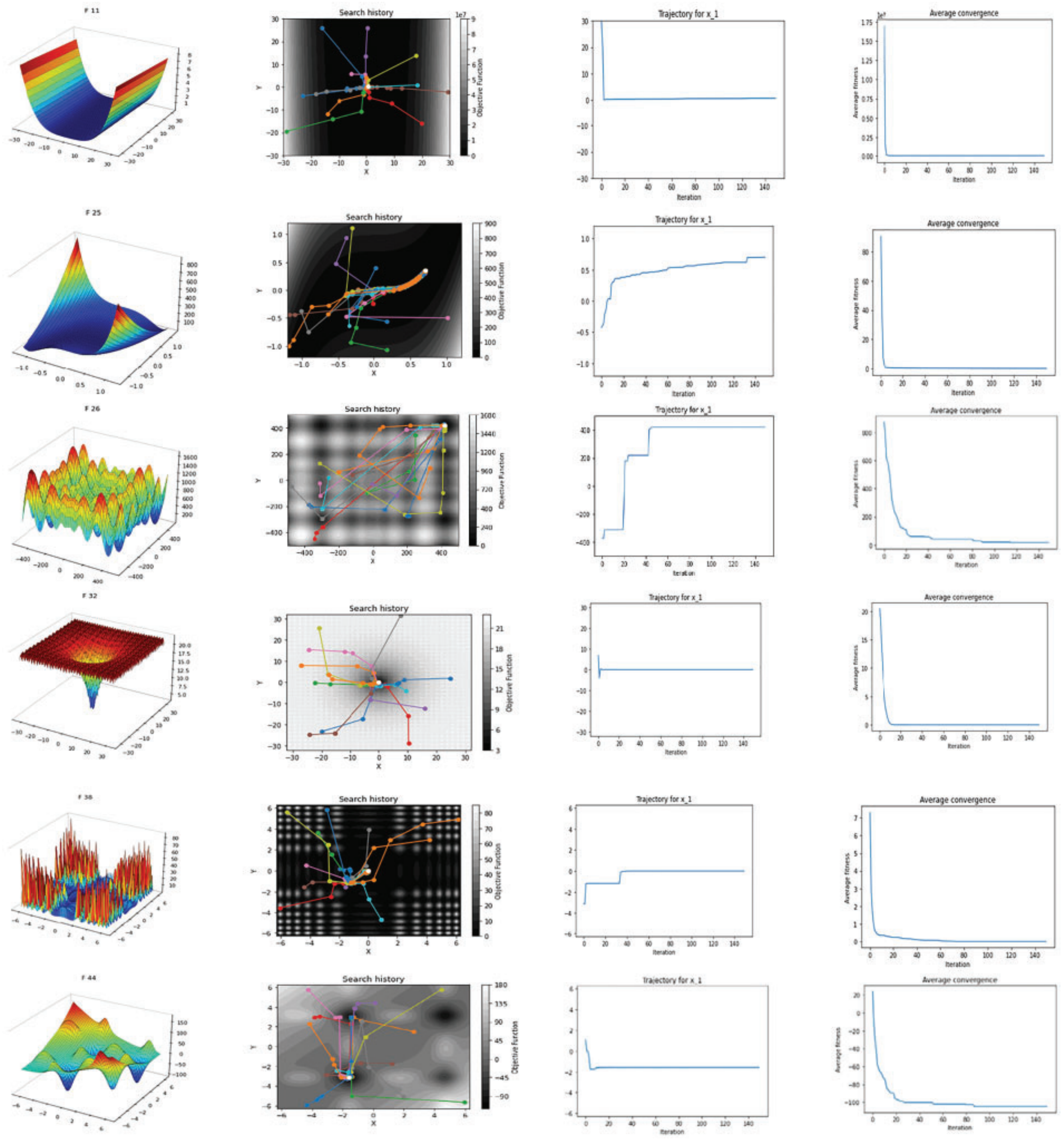

Figure 4: Search history and trajectory of 12 agents, trajectory of the first dimension of one agent, and the average convergence of all agents

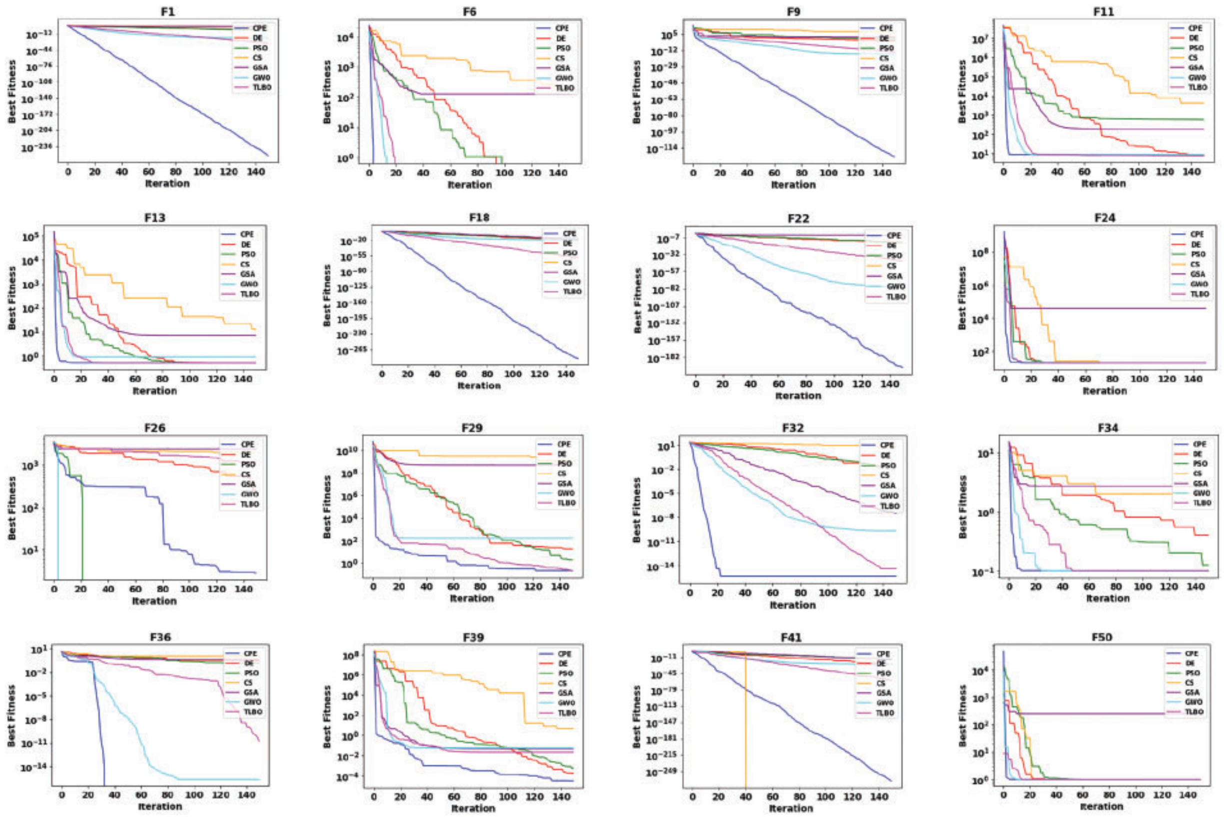

Figure 5: Comparisons of convergence curves for some benchmark functions

5.2 Experiment 2: Using CEC-2017 with 30 Dimensions

In this experiment, we utilized CEC-2017 benchmark functions with 30 dimensions, encompassing 30 test functions, which included unimodal, multimodal, hybrid, and composition functions. However, function 2 was excluded from the analysis due to its instability. The performance of the proposed CPE algorithm is compared against five well-known optimization algorithms, namely GSA, GWO, WOA, SCA, and HHO. These algorithms were chosen because they are representative of powerful techniques in the field of optimization each inspired with different domain and provide a robust benchmark for evaluating the CPE performance. For all algorithms, NOFE is set to 60,000 function evaluations. Table 6 presents the names of the algorithms and their parameter settings. Each algorithm is executed 30 times for each test function, and the Mean and SD are recorded. The best result is highlighted in bold. Table 7 presents the Mean and SD results of the proposed algorithm against the other algorithms, while Table 8 shows the rank of each algorithm in each test function. The last row in Table 8 presents the Freidman mean rank test for each algorithm across all the 29 test functions. Analysing the results from Tables 7 and 8, it is evident that the proposed CPE algorithm demonstrates superior performance on certain test functions, obtaining the first rank in 15 out of the 29 functions. For the remaining 14 test functions, the proposed CPE algorithm achieves the second rank in 10 test functions. The worst result is obtained in only two functions (13 and 14). Overall, the performance of the proposed algorithm either outperforms or is comparable to the other algorithms. The CPE strengths are again evident in its strong performance across a diverse set of test functions, particularly in achieving the top rank in over half of the functions tested. This demonstrates its robustness and adaptability to different types of optimization problems. However, its lower performance on functions 13 and 14 suggests areas where the algorithm could be refined, potentially through parameter adjustments or hybridization with other techniques. Among the alternative algorithms, GSA and GWO showed competitive performance, particularly on specific function types. GSA strength in global search capabilities makes it effective for complex landscapes, while GWO hierarchical structure aids in convergence. Nonetheless, these algorithms may struggle with maintaining a balance between exploration and exploitation, which is crucial for achieving consistent performance across varied functions. Furthermore, the mean rank analysis reveals that the proposed CPE algorithm achieves better mean ranks, approximately 1.55. These results provide valuable insights into the algorithm's overall performance across the entire range of test functions, suggesting that the proposed CPE algorithm possesses notable strengths and competitiveness. However, areas for improvement include refining the algorithm to handle specific functions where it showed relatively lower performance, potentially by adjusting parameter settings or incorporating hybrid strategies.

Table 6: Optimization algorithms' names and their parameters

Algorithm	Parameters setting	Algorithm	Parameters setting
CPE	*p-size* $= 30$, $\mathcal{R} =$ UP-LL/200, r_2 range $(-1, 1)$, *danger* condition ≤ 0.5, r_1, r_3–r_6 are random number between (0, 1)	WOA	*p-size* $= 30$, $b = 1$, (a, C, l) from corresponding equation
GSA	*p-size* $= 30$, $\alpha = 20$, $G_0 = 100$, $k =$ [*p-size* $\rightarrow 1$]	HHO	*p-size* $= 30$
GWO	*p-size* $= 30$, $[a, C]$ from corresponding equation	SCA	*p-size* $= 30$, a $= 2$, $[1, r_2, r_3, r_4]$ from corresponding equation

Table 7: Results of the proposed CPE against other 8 algorithm using CEC-2017 test functions with 30 dimensions

#	Stats	CPE	GSA	GWO	WOA	SCA	HHO	#	Stats	CPE	GSA	GWO	WOA	SCA	HHO
F1	*Mean*	**4.97E+05**	7.46E+07	1.99E+09	1.08E+09	1.84E+10	3.17E+07	F17	*Mean*	2.30E+03	2.88E+03	**2.07E+03**	2.91E+03	2.64E+03	2.50E+03
	SD	2.45E+05	1.57E+08	1.30E+09	3.15E+08	3.07E+09	7.47E+06		*SD*	2.31E+02	2.44E+02	2.00E+02	3.23E+02	2.88E+02	3.09E+02
F3	*Mean*	4.43E+04	9.62E+04	5.33E+04	2.95E+05	6.66E+04	**3.99E+04**	F18	*Mean*	7.88E+05	**5.16E+05**	2.43E+06	1.24E+07	9.49E+06	2.54E+06
	SD	6.98E+03	1.10E+04	1.32E+04	7.46E+04	1.33E+04	6.79E+03		*SD*	6.21E+05	4.03E+05	5.15E+06	6.47E+06	4.99E+06	1.02E+06
F4	*Mean*	**4.83E+02**	6.54E+02	5.96E+02	8.74E+02	2.35E+03	5.55E+02	F19	*Mean*	**8.08E+03**	1.35E+05	3.06E+06	6.09E+06	7.10E+07	8.94E+05
	SD	2.09E+01	1.25E+02	5.81E+01	4.64E+01	6.92E+02	1.46E+01		*SD*	3.27E+03	1.08E+05	8.77E+06	5.06E+06	4.12E+07	5.59E+05
F5	*Mean*	**6.09E+02**	7.39E+02	6.16E+02	8.58E+02	8.14E+02	7.40E+02	F20	*Mean*	2.50E+03	3.07E+03	**2.45E+03**	2.82E+03	2.90E+03	2.85E+03
	SD	2.43E+01	2.43E+01	2.30E+01	5.22E+01	2.64E+01	4.90E+01		*SD*	1.61E+02	2.66E+02	1.73E+02	1.98E+02	1.43E+02	1.86E+02
F6	*Mean*	6.51E+02	6.58E+02	**6.11E+02**	6.91E+02	6.60E+02	6.73E+02	F21	*Mean*	2.42E+03	2.62E+03	**2.40E+03**	2.60E+03	2.59E+03	2.61E+03
	SD	6.10E+00	3.99E+00	4.34E+00	1.52E+01	6.28E+00	9.36E+00		*SD*	4.63E+01	3.59E+01	2.46E+01	2.00E+01	2.62E+01	4.52E+01
F7	*Mean*	**8.54E+02**	9.62E+02	8.86E+02	1.29E+03	1.21E+03	1.30E+03	F22	*Mean*	**3.01E+03**	7.25E+03	4.99E+03	6.55E+03	9.01E+03	5.77E+03
	SD	2.56E+01	5.50E+01	5.02E+01	5.18E+01	4.61E+01	5.43E+01		*SD*	1.44E+03	5.08E+02	1.84E+03	2.45E+03	2.49E+03	2.99E+03
F8	*Mean*	8.97E+02	9.63E+02	**8.96E+02**	1.07E+03	1.09E+03	9.68E+02	F23	*Mean*	**2.75E+03**	3.80E+03	2.80E+03	3.07E+03	3.05E+03	3.25E+03
	SD	2.45E+01	1.64E+01	1.56E+01	4.81E+01	2.13E+01	3.46E+01		*SD*	3.35E+01	1.85E+02	6.09E+01	1.06E+02	2.96E+01	6.49E+01
F9	*Mean*	**9.06E+02**	4.17E+03	2.26E+03	1.16E+04	7.92E+03	7.55E+03	F24	*Mean*	2.95E+03	3.44E+03	**2.94E+03**	3.23E+03	3.24E+03	3.43E+03
	SD	2.25E+00	4.17E+02	6.75E+02	5.87E+03	1.54E+03	1.90E+03		*SD*	3.00E+01	1.32E+02	5.36E+01	6.98E+01	3.01E+01	1.17E+02
F10	*Mean*	**3.96E+03**	5.39E+03	4.80E+03	6.73E+03	8.66E+03	6.14E+03	F25	*Mean*	**2.90E+03**	2.99E+03	3.01E+03	3.10E+03	3.48E+03	2.94E+03
	SD	5.67E+02	4.92E+02	1.15E+03	9.07E+02	3.33E+02	3.57E+02		*SD*	2.44E+01	2.57E+01	4.51E+01	3.28E+01	1.43E+02	1.62E+01
F11	*Mean*	1.942E+03	3.97E+03	2.04E+03	6.90E+03	3.45E+03	**1.29E+03**	F26	*Mean*	**3.97E+03**	7.85E+03	4.90E+03	9.31E+03	7.60E+03	8.21E+03
	SD	9.82E+02	1.12E+03	8.34E+02	3.18E+03	1.08E+03	3.41E+01		*SD*	1.25E+03	6.30E+02	3.12E+02	1.01E+03	3.93E+02	1.18E+03
F12	*Mean*	**1.40E+07**	1.25E+08	1.34E+08	3.17E+08	2.15E+09	3.69E+07	F27	*Mean*	**3.25E+03**	5.10E+03	**3.25E+03**	3.48E+03	3.51E+03	3.65E+03
	SD	1.22E+07	1.29E+08	2.48E+08	1.29E+08	5.75E+08	2.63E+07		*SD*	1.30E+01	4.19E+02	1.78E+01	8.82E+01	4.41E+01	3.28E+02

(Continued)

Table 7 (continued)

#	Stats	CPE	GSA	GWO	WOA	SCA	HHO	#	Stats	CPE	GSA	GWO	WOA	SCA	HHO
F13	*Mean*	6.68E+05	**3.40E+04**	3.51E+05	1.21E+06	9.15E+08	6.68E+05	F28	*Mean*	**3.23E+03**	3.56E+03	3.51E+03	3.56E+03	4.23E+03	3.35E+03
	SD	2.13E+06	9.35E+03	6.03E+05	3.84E+05	4.21E+08	4.26E+05		*SD*	2.36E+01	2.08E+02	1.98E+02	1.50E+02	2.36E+02	3.48E+01
F14	*Mean*	7.72E+05	1.03E+06	5.06E+05	3.17E+06	4.70E+05	**4.12E+05**	F29	*Mean*	**3.71E+03**	5.28E+03	3.95E+03	5.42E+03	5.05E+03	4.90E+03
	SD	5.88E+05	3.52E+05	4.37E+05	2.59E+06	3.85E+05	2.54E+05		*SD*	1.93E+02	3.33E+02	1.80E+02	3.70E+02	1.93E+02	6.31E+02
F15	*Mean*	**1.24E+04**	1.78E+04	5.95E+05	2.27E+06	4.26E+07	1.11E+05	F30	*Mean*	**2.07E+04**	3.18E+06	9.29E+06	3.98E+07	1.41E+08	7.82E+06
	SD	2.53E+02	5.93E+03	1.23E+06	3.67E+06	3.22E+07	1.13E+05		*SD*	8.88E+03	4.09E+06	8.10E+06	3.06E+07	4.86E+07	3.85E+06
F16	*Mean*	2.76E+03	3.47E+03	**2.53E+03**	4.06E+03	3.95E+03	3.23E+03								
	SD	2.53E+02	3.34E+02	2.52E+02	1.09E+03	2.38E+02	2.66E+02								

Table 8: Rank of each algorithm in each test function in CEC-2017, Freidman mean-rank

#	CPE	GSA	GWO	WOA	SCA	HHO	#	CPE	GSA	GWO	WOA	SCA	HHO
F1	1	3	5	4	6	2	F17	2	5	1	6	4	3
F3	2	5	3	6	4	1	F18	2	1	3	6	5	4
F4	1	4	3	5	6	2	F19	1	2	4	5	6	3
F5	1	3	2	6	5	4	F20	2	6	1	3	5	4
F6	2	3	1	6	4	5	F21	2	6	1	4	3	5
F7	1	3	2	5	4	6	F22	1	5	2	4	6	3
F8	2	3	1	5	6	4	F23	1	6	2	4	3	5
F9	1	3	2	6	5	4	F24	2	6	1	3	4	5
F10	1	3	2	5	6	4	F25	1	3	4	5	6	2
F11	2	5	3	6	4	1	F26	1	4	2	6	3	5
F12	1	3	4	5	6	2	F27	1.5	6	1.5	3	4	5
F13	3.5	1	2	5	6	3.5	F28	1	4.5	3	4.5	6	2
F14	4	5	3	6	2	1	F29	1	5	2	6	4	3
F15	1	2	4	5	6	3	F30	1	2	4	5	6	3
F16	2	4	1	6	5	3	Mean rank	1.55	3.84	2.40	5.02	4.83	3.36

5.3 Experiment 3: Applying CPE to Real Engineering Design Problems

In this study, we employed the proposed CPE algorithm to optimize four established engineering design problems, namely Welded Beam Design (WBD), Speed Reducer Design (SRD), Cantilever Beam Design (CBD), and Multi-plate Disc Clutch Brake Design (MDCBD) [34]. To gauge the effectiveness of the CPE algorithm, we conducted a comparative analysis with other algorithms that have also been utilized to optimize these problems, using only studies that employed the correct mathematical equations. Notably, different solutions may arise due to differences in the problem formulations [34]. Table 9 presents statistical results for the WBD problem, including Worst, Mean, Best, SD, and Number of Function Evaluations (NFEs), comparing the performance of the proposed CPE algorithm with other algorithms. The results indicate that the CPE algorithm outperforms all other algorithms in terms of Mean, Best, and SD, except for the Dynamic Stochastic Selection-Multimember Differential Evolution (*DSS-MDE*) method, which produces slightly better results but requires significantly more NFEs (18,467 NFEs *vs*. 11,807 NFEs for the proposed CPE algorithm). Similarly, Table 10 shows that the proposed CPE algorithm is superior to all other algorithms in all terms for the SRD problem, with only 5000 NFEs. In contrast, Table 11 indicates that there are no appreciable variations in the algorithms for the CBD problem in terms of Worst, Mean, and Best. However, on the positive side, the CPE algorithm requires only 173 NFEs. Lastly, Table 12 presents the results for the MDCBD problem, indicating that there were no significant differences between the algorithms' performance in terms of Best. However, the proposed algorithm performs slightly worse against other algorithms in terms of Mean and Worst, with a difference of about 0.02 and 0.1, respectively. On the positive side, the CPE algorithm requires only 173 NFEs to achieve the same best solution produced by the other algorithms, making it faster than all other algorithms.

Table 9: Statistical results of the proposed CPE *vs*. other algorithms for WBD problem

Algorithm	Worst	Mean	Best	SD	NFEs
CPE	2.7375	2.47255	2.3811	7.3E–2	**11,807**
Society and Civilization (SC) [35]	6.39968	3.25514	2.38543	9.6E–01	33,095
Simulated Annealing (SA)	3.0521	2.8327	2.7290	7.9E–02	100,000
PSO	3.0369	2.8456	2.7289	1.0E–01	100,000
GA	2.9368	2.8439	2.7620	5.0E–02	100,000
Firefly Algorithm (FFA)	3.1223	2.8691	2.7251	1.1E–01	100,000
Evolutionary Strategy (ES) [36]	3.0373	2.8443	2.7245	1.0E–01	100,000
DSS-MDE [37]	**2.38096**	**2.38096**	**2.38096**	3.2E–10	24,000

Table 10: Statistical results of the proposed CPE *vs*. other algorithms for SRD problem

Algorithm	Worst	Mean	Best	SD	NFEs
CPE	**2994.35102**	**2994.341875**	**2994.337211**	2.7E−03	**5000**
PO [10]	2994.47100	2994.471000	2994.471000	3.0−E05	5400
Grey Prediction Evolution Algorithm-Accelerated Even Grey (GPEAae) [38]	3028.22875	2995.144685	2994.468240	4.8E+00	19,980
SC [35]	3009.96474	3001.758264	2994.744241	4.0E+00	54,456
Modified Augmented Lagrange-Differential Evolution (MAL-DE) [39]	2994.47107	2994.471066	2994.471066	0.0E+00	120,000
DSS-MDE [37]	2994.47107	2994.471066	2994.471066	3.6E−12	30,000

Table 11: Statistical results of the proposed CPE *vs*. other algorithms for CBD problem

Algorithm	Worst	Mean	Best	SD	NFEs
CPE	1.3485236	1.342234	1.340006	1.64E−03	**10,000**
Ecosystem-Based Optimization (AEO) [40]	1.3400890	1.339970	1.339965	8.25E−06	15,000
Social Network Search (SNS) [41]	1.3399576	1.339958	1.339958	1.11E−15	12,000

(Continued)

Table 11 (continued)

Algorithm	Worst	Mean	Best	SD	NFEs
Improved Multi-Objective Evolutionary Algorithm (IMOEA) [42]	1.3458000	1.340907	1.339996	1.64E−03	12,000
Improved Pathfinder Algorithm (IMPFA) [43]	1.3400000	1.340000	1.340000	1.28E−05	30,000

Table 12: Statistical results of the proposed CPE *vs*. other algorithms for MDCBD

Algorithm	Worst	Mean	Best	SD	NFEs
CPE	0.442427	0.3462710	0.3136566	3.81E−02	**173**
Water Cycle Algorithm (WCA) [44]	0.313656	0.3136560	0.3136566	1.69E−16	500
AEO [40]	0.333260	0.3216842	0.3136566	8.15E−03	500
HBO2 [11]	0.313656	0.3136566	0.3136566	5.85E−17	970
Chaotic Multi-Verse Optimization (CMVO) [45]	0.337450	0.3139440	0.3136566	2.54E−07	50,000
Interval Particle Swarm Optimization ([I]PSO) [46]	**0.313656**	**0.3136560**	0.3136566	**0.00E+00**	20,000

The statistical results presented in the tables lead to the conclusion that the proposed CPE algorithm outperforms other algorithms in the majority of the tested optimization problems, achieving better Mean and Best values or similar results. Moreover, the CPE algorithm requires fewer function evaluations, making it a faster optimization method. These findings suggest that the CPE algorithm is a promising optimization method that can be applied to a wide range of problems, and it has the potential to be a useful tool for researchers and practitioners in various fields.

6 Conclusions and Future Works

This study presents a novel optimization algorithm, called the CPE algorithm, which is inspired by the hunting behavior of predators and prey. The proposed CPE algorithm utilizes a unique approach that effectively combines multiple searches. By partitioning the population into two groups, each tasked with independently exploring the search space, the algorithm can efficiently navigate through the problem domain and avoid local optima. Moreover, the algorithm employs a distinct search mechanism that incorporates the average of the best solution, and the current solution, facilitating convergence to the optimal solution. Additionally, the use of the pouncing process enables faster movement towards the best solution. The algorithm's efficacy is demonstrated across 50 well-known test functions, CEC-2017 benchmarks, and real-world engineering challenges, consistently surpassing other algorithms. However, we acknowledge that the algorithm does have certain limitations. Specifically, determining parameters such as the radius $\mathcal{R}$, r_2, and *danger* introduces challenges. The calculation of these parameters is a complex task, with factors such as problem domain influencing their values. For instance, the optimal value of $\mathcal{R}$ could be domain-dependent, potentially requiring adaptation for different problem spaces. Similarly, the optimization of ranges for r_2 and *danger* remains an open question, with potential for improvements by adjusting these parameters. Future research can refine performance through parameter tuning, enhancing adaptability across varied optimization scenarios. Expanding research prospects include fine-tuning parameters, broadening application domains to feature selection and clustering, exploring multi-objective optimization, and integrating parallel computing for enhanced performance.

Acknowledgement: Not applicable.

Funding Statement: The author received no specific funding for this study.

Availability of Data and Materials: Not applicable.

Conflicts of Interest: The author declares that he has no conflicts of interest to report regarding the present study.

References

[1] R. R. Bulatović, S. R. Đorđević, and V. S. Đorđević, "Cuckoo search algorithm: A metaheuristic approach to solving the problem of optimum synthesis of a six-bar double dwell linkage," *Mech. Mach. Theory*, vol. 61, no. 4, pp. 1–13, 2013. doi: 10.1016/j.mechmachtheory.2012.10.010.

[2] M. A. Albadr, S. Tiun, M. Ayob, and F. Al-Dhief, "Genetic algorithm based on natural selection theory for optimization problems," *Symmetry*, vol. 12, no. 11, pp. 1758, 2020. doi: 10.3390/sym12111758.

[3] J. Tang, G. Liu, and Q. Pan, "A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends," *IEEE/CAA J. Automat. Sin.*, vol. 8, no. 10, pp. 1627–1643, 2021. doi: 10.1109/JAS.2021.1004129.

[4] N. Siddique and H. Adeli, "Nature inspired computing: An overview and some future directions," *Cognit. Comput.*, vol. 7, no. 6, pp. 706–714, 2015. doi: 10.1007/s12559-015-9370-8.

[5] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, no. 12, pp. 51–67, 2016. doi: 10.1016/j.advengsoft.2016.01.008.

[6] T. R. Biyanto *et al.*, "Killer whale algorithm: An algorithm inspired by the life of killer whale," in *Proc. Comput. Sci.*, Bali, Indonesia, 2017, vol. 124, pp. 151–157. doi: 10.1016/j.procs.2017.12.141.

[7] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, 2017. doi: 10.1016/j.advengsoft.2017.07.002.

[8] S. Arora and S. Singh, "Butterfly optimization algorithm: A novel approach for global optimization," *Soft Comput.*, vol. 23, no. 3, pp. 715–734, 2019. doi: 10.1007/s00500-018-3102-4.

[9] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja and H. Chen, "Harris hawks' optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, 2019. doi: 10.1016/j.future.2019.02.028.

[10] Q. Askari, I. Younas, and M. Saeed, "Political optimizer: A novel socio-inspired meta-heuristic for global optimization," *Knowl.-Based Syst.*, vol. 195, no. 5, pp. 105709, 2020. doi: 10.1016/j.knosys.2020.105709.

[11] Q. Askari, M. Saeed, and I. Younas, "Heap-based optimizer inspired by corporate rank hierarchy for global optimization," *Expert Syst. Appl.*, vol. 161, no. 3, pp. 113702, 2020. doi: 10.1016/j.eswa.2020.113702.

[12] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011. doi: 10.1016/j.cad.2010.12.015.

[13] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, 1997. doi: 10.1109/4235.585893.

[14] A. Hussain and Y. S. Muhammad, "Trade-off between exploration and exploitation with genetic algorithm using a novel selection operator," *Complex Intell. Syst.*, vol. 6, no. 1, pp. 1–14, 2020. doi: 10.1007/s40747-019-0102-7.

[15] L. Ming, D. Wenqiang, and N. Fuzhong, "An adaptive particle swarm optimization algorithm based on directed weighted complex network," *Math. Probl. Eng.*, vol. 2014, pp. 434972, 2014. doi: 10.1155/2014/434972.

[16] B. Li, Y. Zhou, Y. Qiu, and Z. Li, "An ant-based optimizer for the training of convolutional neural networks," in *Proc. 2022 China Automat. Congress (CAC)*, Xiamen, China, Nov. 25–27, 2022. doi: 10.1109/CAC57257.2022.10055228.

[17] E. Rashedi, H. Nezamabadi, and S. Saryazdi, "GSA: A gravitational search algorithm," *Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009. doi: 10.1016/j.ins.2009.03.004.

[18] X. S. Yang and S. Deb, "Engineering optimization by cuckoo search," *Int. J. Math. Modell. Numer. Optim.*, vol. 1, no. 4, pp. 330–343, 2010. doi: 10.1504/IJMMNO.2010.035430.

[19] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014. doi: 10.1016/j.advengsoft.2013.12.007.

[20] W. G. Luo and X. B. Yu, "A novel enhanced cuckoo search algorithm for global optimization," *J. Intell. Fuzzy Syst.*, vol. 43, no. 3, pp. 2945–2962, 2022. doi: 10.3233/JIFS-220179.

[21] Y. Ou, P. Yin, and L. Mo, "An improved grey wolf optimizer and its application in robot path planning," *Biomimetics*, vol. 8, no. 1, pp. 84, 2023. doi: 10.3390/biomimetics8010084.

[22] S. Harifi, M. Khalilian, J. Mohammadzadeh, and S. Ebrahimnejad, "Emperor penguins' colony: A new metaheuristic algorithm for optimization," *Evol. Intell.*, vol. 12, no. 2, pp. 211–226, 2019. doi: 10.1007/s12065-019-00212-x.

[23] K. Zervoudakis and S. Tsafarakis, "A mayfly optimization algorithm," *Comput. Industr. Eng.*, vol. 145, no. 5, pp. 106559, 2020. doi: 10.1016/j.cie.2020.106559.

[24] M. Dirik, "Comparison of recent meta-heuristic optimization algorithms using different benchmark functions," *J. Math. Sci. Model.*, vol. 5, no. 3, pp. 113–124, 2022. doi: 10.33187/jmsm.1115792.

[25] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, 2016. doi: 10.1016/j.knosys.2015.12.022.

[26] S. Kaur, L. K. Awasthi, A. L. Sangal, and G. Dhiman, "Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization," *Eng. Appl. Artif. Intell.*, vol. 90, no. 2, pp. 103541, 2020. doi: 10.1016/j.engappai.2020.103541.

[27] M. Jahangiri, M. A. Hadianfard, M. A. Najafgholipour, M. Jahangiri, and M. R. Gerami, "Interactive autodidactic school: A new metaheuristic optimization algorithm for solving mathematical and structural design optimization problems," *Comput. Struct.*, vol. 235, no. 2, pp. 106268, 2020. doi: 10.1016/j.compstruc.2020.106268.

[28] M. Alimoradi, H. Azgomi, and A. Asghari, "Trees social relations optimization algorithm: A new swarm-based metaheuristic technique to solve continuous and discrete optimization problems," *Math. Comput. Simul.*, vol. 194, no. 4, pp. 629–664, 2022. doi: 10.1016/j.matcom.2021.12.010.

[29] A. M. Balani, M. D. Nayeri, A. Azar, and M. T. Yazdi, "Golden eagle optimizer: A nature-inspired metaheuristic algorithm," *Comput. Ind. Eng.*, vol. 152, pp. 107050, 2021. doi: 10.1016/j.cie.2020.107050.

[30] I. Naruei and F. Keynia, "A new optimization method based on COOT bird natural life model," *Expert Syst. Appl.*, vol. 183, no. 2, pp. 115352, 2021. doi: 10.1016/j.eswa.2021.115352.

[31] J. R. Šćepanović, A. Karač, Z. M. Jakšić, L. B. Petković, and S. B. Vrhovac, "Group chase and escape in the presence of obstacles," *Phys. A: Stat. Mech. Appl.*, vol. 525, pp. 450–465, 2019. doi: 10.1016/j.physa.2019.03.017.

[32] G. B. Schaller, "Predator-prey relations," in *The Serengeti Lion: A Study of Predator-Prey Relations*, 1st ed. Chicago, USA: University of Chicago Press, 1972, pp. 87.

[33] F. Dumont, C. Pasquaretta, D. Réale, G. Bogliani, and A. V. Hardenberg, "Flight initiation distance and starting distance: Biological effect or mathematical artefact," *Ethology*, vol. 118, no. 11, pp. 1051–1062, 2012. doi: 10.1111/eth.12006.

[34] A. S. Eesa, M. M. Hassan, and W. K. Arabo, "Letter: Application of optimization algorithms to engineering design problems and discrepancies in mathematical formulas," *Appl. Soft Comput.*, vol. 140, pp. 110252, 2023. doi: 10.1016/j.asoc.2023.110252.

[35] T. Ray and K. M. Liew, "Society and civilization: An optimization algorithm based on the simulation of social behavior," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 386–396, 2003. doi: 10.1109/TEVC.2003.814902.
[36] M. Ebrahimi, "Design and optimization of aluminum cross-car beam assemblies considering uncertainties," M.S. dissertation, University of Toronto, Canada, 2015.
[37] M. Zhang, W. Luo, and X. Wang, "Differential evolution with dynamic stochastic selection for constrained optimization," *Inf. Sci.*, vol. 178, no. 15, pp. 3043–3074, 2008. doi: 10.1016/j.ins.2008.02.014.
[38] C. Gao, Z. Hu, Z. Xiong, and Q. Su, "Grey prediction evolution algorithm based on accelerated even grey model," *IEEE Access*, vol. 8, pp. 107941–107957, 2020. doi: 10.1109/ACCESS.2020.3001194.
[39] W. Long, X. Liang, Y. Huang, and Y. Chen, "A hybrid differential evolution augmented Lagrange method for constrained numerical and engineering optimization," *Comput.-Aided Design*, vol. 45, no. 12, pp. 1562–1574, 2013. doi: 10.1016/j.cad.2013.07.007.
[40] W. Zhao, L. Wang, and Z. Zhang, "Artificial ecosystem-based optimization: A novel nature-inspired meta-heuristic algorithm," *Neural Comput. Appl.*, vol. 32, no. 13, pp. 9383–9425, 2020. doi: 10.1007/s00521-019-04452-x.
[41] H. Bayzidi, S. Talatahari, M. Saraee, and C. P. Lamarche, "Social network search for solving engineering optimization problems," *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–32, 2021. doi: 10.1155/2021/8548639.
[42] H. G. Arab, A. M. Rayeni, and M. R. Ghasemi, "An effective improved multi-objective evolutionary algorithm (IMOEA) for solving constraint civil engineering optimization problems," *Teknik Dergi*, vol. 32, no. 2, pp. 10645–10674, 2021. doi: 10.18400/tekderg.541640.
[43] C. Tang, Y. Zhou, Q. Luo, and Z. Tang, "An enhanced pathfinder algorithm for engineering optimization problems," *Eng. Comput.*, vol. 83, no. 2, pp. 1481–1503, 2021.
[44] H. Eskandar, A. Sadollah, A. Bahreininejad, and M. Hamdi, "Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems," *Comput. Struct.*, vol. 110–111, no. 1, pp. 151–166, 2012. doi: 10.1016/j.compstruc.2012.07.010.
[45] G. I. Sayed, A. Darwish, and A. E. Hassanien, "A new chaotic multi-verse optimization algorithm for solving engineering optimization problems," *J. Exp. Theor. Artif. Intell.*, vol. 30, no. 2, pp. 293–317, 2018. doi: 10.1080/0952813X.2018.1430858.
[46] T. M. Coelho, A. M. Machado, L. Jaulin, P. Ekel, W. Pedrycz and G. L. Soares, "An interval space reducing method for constrained problems with particle swarm optimization," *Appl. Soft Comput.*, vol. 59, no. 4, pp. 405–417, 2017. doi: 10.1016/j.asoc.2017.05.022.

Appendix A

Table A1: Unimodal (F1–F25), and multimodal (F26–F50) test functions with their characteristics

Fun-name	Domain	Dim.	Fun-type	F-min	Fun-name	Domain	Dim	Fun-type	F-min
F1-Sphere	[−100, 100]	50	S, D, C, V	0	F26-Schwefels 2.26	[−500, 500]	50	S, D, C, V	0
F2-Quartic Noise	[−1.28, 1.28]	50	S, D, C, V	0	F27-Rastrigin	[−5.12, 5.12]	50	S, D, C, V	1
F3-Powell	[−1, 1]	50	S, D, C, V	0	F28-Periodic	[−10, 10]	50	S, D, C, V	0.9
F4-Schwefels 2.20	[−100, 100]	50	S, ND, C, V	0	F29-Qing	[−500, 500]	50	S, D, C, V	0
F5-Schwefels 2.21	[−100, 100]	50	S, ND, C, V	0	F30-Alpine N. 1	[−10, 10]	50	S, ND, C, V	0
F6-Step 2	[−100, 100]	50	S, ND, DC, V	0	F31-Xin-She Yang	[−5, 5]	50	S, ND, C, V	0
F7-Stepint	[−5.12, 5.12]	50	S, ND, DC, V	25-6*n*	F32-Ackley	[−32, 32]	50	IS, D, C, V	0
F8-Schwefels 1.20	[−100, 100]	50	IS, D, C, V	0	F33-Trignometric 2	[−500, 500]	50	IS, D, C, V	1
F9-Schwefels 2.22	[−100, 100]	50	IS, D, C, V	0	F34-Salomon	[−100, 100]	50	IS, D, C, V	0

(Continued)

Table A1 (continued)

Fun-name	Domain	Dim.	Fun-type	F-min	Fun-name	Domain	Dim	Fun-type	F-min
F10-Schwefels 2.23	[−10, 10]	50	IS, D, C, V	0	F35-Styblinski–Tang	[−5, 5]	50	IS, D, C, V	-39.16599*n
F11-Rosenbrock	[−30, 30]	50	IS, D, C, V	0	F36-Griewank	[−100, 100]	50	IS, D, C, V	0
F12-Brown	[−1, 4]	50	IS, D, C, V	0	F37-Xin-She Yang N.4	[−10, 10]	50	IS, ND, C, V	−1
F13-Dixon and Price	[−10, 10]	50	IS, D, C, V	0	F38-Xin-She Yang N.3	$[-2\pi, 2\pi]$	50	IS, ND, C, V	0
F14-Powell Singular	[−4, 5]	50	IS, D, C, V	0	F39-Gen. Penalized	[−50, 50]	50	IS, ND, C, V	0
F15-Zakharov	[−5, 10]	50	IS, D, C, V	0	F40-Penalized	[−50, 50]	50	IS, ND, C, V	0
F16-Xin-She Yang	[−20, 20]	2	IS, D, C, V	−1	F41-Egg crate	[−5, 5]	2	S, D, C, F	0
F17-Perm 0, D, Beta	$[-d_i, d_i]$	2	IS, D, C, F	0	F42-Ackley N.3	[−32, 32]	2	IS, D, C, F	−195.61
F18-Three-Hump Camel	[−5, 5]	2	IS, D, C, F	0	F43-Adjiman	[−1, 2]	2	IS, D, C, F	−2.02181
F19-Beale	[−4.5, 4.5]	2	IS, D, C, F	0	F44-Bird	$[-2\pi, 2\pi]$	2	IS, D, C, F	−106.7645
F20-Booth	[−10, 10]	2	IS, D, C, F	0	F45-Camel Six Hump	[−5, 5]	2	IS, D, C, F	−1.0316
F21-Brent	[−10, 10]	2	IS, D, C, F	0	F46-Branin RCOS	[−5, 10]	2	IS, D, C,F	0.3978873
F22-Matyas	[−10, 10]	2	IS, D, C, F	0	F47-Hartman 3	[0, 1]	2	IS, D, C, F	−3.862782
F23-Schaffer N. 4	[−100, 100]	2	IS, D, C, F	0.29257	F48-Hartman 6	[0, 1]	6	IS, D, C, F	−3.32237
F24-Wayburn Seader 3	[−500, 500]	2	IS, D, C, F	19.106	F49-Cross-in-tray	[−10, 10]	2	IS, ND, C, F	−2.06261218
F25-Leon	[−1.2, 1.2]	2	IS, D, C, F	0	F50-Bartels Conn	[−500, 500]	2	IS, ND, C, F	1

Note: Characteristic (S, IS, D, ND, C, DC, F, V, and F-min) means: Separable, Inseparable, Differentiable, Non-differentiable, Continuous, Discontinuous, Fixed dimensional, Variable dimensional and global optimum.