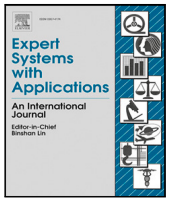
Expert Systems With Applications 238 (2024) 122070

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Partial reinforcement optimizer: An evolutionary optimization algorithm

Ahmad Taheri [a], Keyvan RahimiZadeh [b,c,*], Amin Beheshti [c], Jan Baumbach [a,d], Ravipudi Venkata Rao [e], Seyedali Mirjalili [f,h], Amir H. Gandomi [g,h]

[a] Institute of Computational Systems Biology, University of Hamburg, Notkestrasse 9, 22607 Hamburg, Germany
[b] Department of Computer Engineering, School of Engineering, Yasouj University, Yasouj 7591874934, Iran
[c] Centre for Applied Artificial Intelligence, Macquarie University, Sydney, NSW, 2109, Australia
[d] Institute of Mathematics and Computer Science, University of Southern Denkmark, Campusvej 55, 5320 Odense, Denmark
[e] Sardar Vallabhbhai National Institute of Technology (SV NIT), Surat 395 007, Gujarat State, India
[f] Centre for Artificial Intelligence Research and Optimization, Torrens University, SA, 5000, Australia
[g] Faculty of Engineering & Information Technology, University of Technology Sydney, Ultimo, NSW, 2007, Australia
[h] University Research and Innovation Center (EKIK), Óbuda University, 1034 Budapest, Hungary

ARTICLE INFO

Keywords:
Evolutionary computation
Partial reinforcement theory
Meta-heuristic optimization
Federated deep learning
ECG

ABSTRACT

In this paper, a novel evolutionary optimization algorithm, named Partial Reinforcement Optimizer (PRO), is introduced. The major idea behind the PRO comes from a psychological theory in evolutionary learning and training called the partial reinforcement effect (PRE) theory. According to the PRE theory, a learner is intermittently reinforced to learn or strengthen a specific behavior during the learning and training process. The reinforcement patterns significantly impact the response rate and strength of the learner during a reinforcement schedule, achieved by appropriately selecting a reinforcement behavior and the time of applying reinforcement process. In the PRO algorithm, the PRE theory is mathematically modeled to an evolutionary optimization algorithm for solving global optimization problems. The efficiency of the proposed PRO algorithm is compared to well-known Meta-heuristic Algorithms (MAs) using Wilcoxon and Friedman statistical tests to analyze results from 75 benchmarks of the CEC2005, CEC2014, and CEC-BC-2017 test suits, which include unimodal, multimodal, hybrid, and composition functions. Additionally, the proposed PRO algorithm is applied to optimize a Federated Deep Learning Electrocardiography (ECG) classifier, as a real case study, to investigate the robustness and applicability of the proposed PRO. The experimental results demonstrate that the PRO algorithm outperforms existing meta-heuristic optimization algorithms by providing a more accurate and robust solution.

1. Introduction

The term optimization refers to a decision-making process in which multiple solutions are available to solve a problem, and the purpose is to obtain the best solution performance in terms of performance. Therefore, the best decision is made by satisfying a specific objective function (Telikani et al., 2022). In recent years, there has been a significant growth in the complexity of optimization problems and various methods have been proposed in the literature to address this challenge. Nevertheless, due to the rapid growth in the complexity of real-world problems and the theory of "*No Free Lunch*", there is no single method or algorithm that can optimally solve all types of problems (Wolpert & Macready, 1997). Therefore, further research is necessary to overcome different types of problems with diverse features (Jiang et al., 2020).

In many cases, the traditional calculus-based algorithms are not capable of solving non-convex and non-differentiable optimization problems to find the global optimum. For instance, the objective function in gradient-based optimization algorithms must be differentiable which is not achievable for complex optimization problems (Azizi et al., 2022). Among different types of optimization methods, Meta-heuristic Algorithms (MAs) have been successfully applied to solve a wide range of various complex optimization problems such as image processing (Chen et al., 2020; Singh et al., 2022; Wang et al., 2020), neural networks training (Taheri, Ghashghaei et al., 2021, machine learning and feature

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author at: Department of Computer Engineering, School of Engineering, Yasouj University, Yasouj 7591874934, Iran.
E-mail addresses: ahmad.taheri@uni-hamburg.de (A. Taheri), rahimizadeh@yu.ac.ir, keyvan.rahimizadeh@mq.edu.au (K. RahimiZadeh), amin.beheshti@mq.edu.au (A. Beheshti), jan.baumbach@uni-hamburg.de (J. Baumbach), ravipudirao@gmail.com (R.V. Rao), ali.mirjalili@laureate.edu.au (S. Mirjalili), gandomi@uts.edu.au (A.H. Gandomi).

https://doi.org/10.1016/j.eswa.2023.122070
Received 9 December 2022; Received in revised form 6 October 2023; Accepted 6 October 2023
Available online 19 October 2023

Table 1
Comparison of Meta-heuristic Algorithms (MAs).

Algorithm	Category	Inspiration/Description
Harris Hawks Optimization (HHO) (Heidari et al., 2019)	Swarm intelligence	Inspired by the Harris' hawks chasing style and cooperative behavior in the nature, an MA is introduced.
Equilibrium Optimizer (Faramarzi, Heidarinejad, Stephens et al., 2020)	Physic-based	An optimization algorithm inspired by control volume mass balance models used to estimate both dynamic and equilibrium states.
Poor and rich optimization algorithm (Samareh Moosavi & Bardsiri, 2019)	Human-based	The efforts to improve the economic conditions and achieve wealth based on the behavior of the poor and the rich.
Ludo Game-Based Swarm intelligence (LGSI) (Singh et al., 2019)	Human-based	Simulates rules in playing the Ludo game using two or four players to improve the ability of swarm algorithms.
Nomadic People Optimizer (NPO) (Salih & Alsewari, 2020)	Human-based	Inspired by the nature of nomadic people in their continuous movements (migrations) and searches for sources of food and suitable living places in the desert, an MA is introduced.
Coronavirus Herd Immunity Optimizer (CHIO) (Al-Betar et al., 2021)	Human-based	The strategy of herd immunity and the social distancing concepts is modeled.
Slime Mould Algorithm (SMA) (Li et al., 2020)	Swarm intelligence	The positive and negative feedback of the slime mould propagation wave is simulated using adaptive weights based on bio-oscillator to find the best path.
Political Optimizer (PO) (Askari et al., 2020)	Human-based	The concepts in politics such as constituency allocation, party switching, election campaign, inter-party election, and parliamentary affairs are mathematically modeled.
Marine Predators Algorithm (MPO) (Faramarzi, Heidarinejad, Mirjalili et al., 2020)	Swarm intelligence	Inspired by the foraging strategy namely Lévy and Brownian movements in ocean predators together with optimal encounter rate policy in biological interaction between predator and prey, an MA is introduced.
Red Fox Optimization Algorithm (RFO) (Połap & Woźniak, 2021)	Swarm intelligence	The habits of red fox such as hunting, searching for food, escaping from hunters and developing population are mathematically modeled
Horse Herd Optimization Algorithm (HOA) (Sarwar et al. (2022))	Swarm intelligence	Mathematically simulates horses' social performances at different ages using six important features: defense, hierarchy, grazing, sociability, mechanism, imitation and roam.
Arithmetic Optimization Algorithm (AOA) (Abualigah et al., 2021)	Physic-based	Designed according to the main arithmetic operators' distribution behavior in mathematics such as addition, subtraction, multiplication and division.
Hunger Games Search (HGS) (Yang et al., 2021)	Swarm intelligence	Incorporates the concept of the hunger-driven activities and behavioral choices of hunger animals into the feature process on each search step.
Artificial Hummingbird Algorithm (AHA) (Zhao et al., 2022)	Swarm intelligence	Simulates the hummingbirds' flight skills in foraging strategies such as omnidirectional, diagonal, and axial flights.
Reptile Search Algorithm (RSA) (Abualigah et al., 2022)	Swarm intelligence	Inspired by encirclement and hunt mechanisms of crocodiles, an MA is introduced.
Fire Hawk Optimizer (FHO) (Azizi et al., 2022)	Swarm intelligence	Inspired by the whistling kites foraging behavior, black kites and brown falcons an MA is introduced.
Young's Double-Slit Experiment (YDSE) (Abdel-Basset et al., 2023)	Physic-based	Motivated by Young's double-slit experiment, the promising areas in the bright fringe areas, which are assumed to contain the optimum, are exploited.
Partial Reinforcement Optimizer (PRO)	Human-based	Inspired by the partial reinforcement effect (PRE) theory, an evolutionary learning/training theory in psychology, a learner is intermittently reinforced to learn or strengthen a specific behavior based on a schedule.

selection Aghdam et al., 2009), and a wide range of other engineering problems (Li et al., 2021; Luo et al., 2022; Pan et al., 2022; Taheri et al., 2022; Zeng et al., 2020).

The main MAs advantages are their flexibility, simplicity, gradient, and problem characteristic free nature (Mirjalili et al., 2014). These types of algorithms are commonly inspired by natural phenomena, such as animal life, evolutionary concepts, and physical phenomena (Faramarzi, Heidarinejad, Mirjalili et al., 2020). Genetic Algorithms (GA) (Holland, 1992), Particle Swarm Optimization (PSO) (Eberhart & Kennedy, 1995), Ant Colony Optimization (ACO) (Dorigo et al., 2006), and Simulated Annealing (SA) (Kirkpatrick et al., 1983) are the most well-known MAs. A wide range of MAs has been introduced by researchers over the last years, for example, the Fire Hawk Optimizer (FHO) (Azizi et al., 2022), Equilibrium Optimizer (EO) (Faramarzi, Heidarinejad, Stephens et al., 2020), Marine Predators Algorithm (MPA) (Faramarzi, Heidarinejad, Mirjalili et al., 2020), Whale Optimization Algorithm (WOA) (Mirjalili & Lewis, 2016), Sine Cosine Algorithm (SCA) (Mirjalili, 2016), Reptile Search Algorithm (RSA) (Abualigah et al., 2022), Harmony Search (HS) (Zong Woo Geem et al., 2001), Bat algorithm (Yang, 2010), Teaching–Learning based Optimization algorithm (TLBO) (Rao et al., 2011), or Differential Evolutionary (DE) (Storn & Price, 1997). Table 1 reviews the scope of the recent MAs optimization problems.

This paper introduces a novel evolutionary optimization algorithm based on the principles of the Partial Reinforcement Effect (PRE) theory, allowing for the solving of global optimization problems with acceptable performance. The PRE theory is based on a learning/training paradigm in psychology, where learners are intermittently reinforced rather than continuously, in order to improve their knowledge. The timing and frequency of reinforcement have a significant impact on the rate and strength of the response. A reinforcement schedule dictates which instances of behavior should be reinforced. Sometimes, a learner's behavior may be constantly reinforced every time it occurs, while other times it may not be reinforced at all. There are two types of reinforcement: continuous reinforcement and partial reinforcement. In the PRO algorithm, the PRE theory is applied to an evolutionary optimization algorithm for solving global optimization problems. The motivation behind designing this new optimization algorithm is to harness the advantages of the effective concepts and mechanisms found in the PRE theory, such as scheduling, negative reinforcement, and positive reinforcement, which have proven successful in the field of psychology, and apply them to real-parameter optimization problems. The results confirm the suitability of the PRO algorithm for incorporating these concepts into an evolutionary algorithm. Furthermore, these operations and mechanisms of the proposed PRO algorithm can be combined with other MAs to achieve higher performance in addressing

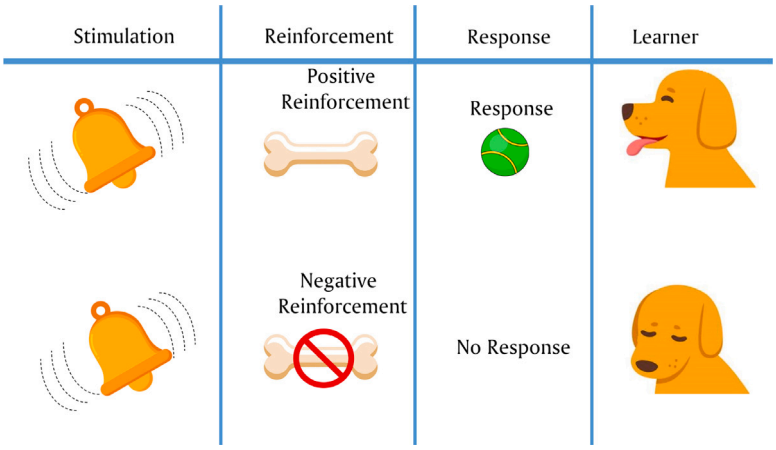


Fig. 1. An example of positive and negative reinforcement in the PRE theory.

highly complex optimization problems. To introduce the PRO, the following assumptions are considered:

- Each behavior is a decision variable and a learner is a solution.
- Each learner has a vector of behaviors.
- The stimulation is considered as making change(s) in learners.
- A response is the objective function value after each stimulation.

The process of reinforcement is modeled as an update in the priority of specific behavior (decision variable) within a schedule, based on its response. Additionally, a scheduling mechanism is employed to simulate the partial reinforcement process, with the primary objective being to achieve more responses. To thoroughly evaluate the effectiveness of the proposed PRO algorithm, a variety of benchmark suit sets are utilized for experimental analysis. In order to assess the efficiency of the PRO algorithm, two statistical analyses, namely Wilcoxon and Friedman tests, are employed on 75 different benchmarks including, unimodal, multimodal, hybrid, and composition functions. Furthermore, the PRO algorithm's performance is compared with that of recently proposed novel MAs and well-known evolutionary algorithms, such as GA, CMS-ES, and SHADE, using the CEC-BC-2017 test suite. Moreover, the PRO algorithm is applied to optimize a federated deep learning Electrocardiography (ECG) classifier, serving as a real case study to evaluate its robustness and applicability. The results affirm the efficiency of the PRO algorithm and demonstrate that it outperforms existing optimization algorithms, providing more accurate and robust solutions.

The rest of this paper is structured as follows. Section 2 introduces the PRE Theory. The detail of the proposed PRO algorithm is described in Section 3. Section 4 discusses the experimental results of the PRO algorithm in comparison with state-of-the-art optimization algorithms on various benchmark test sets. In Section 5, the federated deep learning for ECG diagnostic is optimized by the PRO algorithm. Finally, the conclusion of the paper is presented in Section 6.

2. Partial Reinforcement Effect (PRE) theory

The PRE theory was firstly introduced by Ferster and Skinner in 1957 (Ferster & Skinner, 1957). The authors explained how different reinforcement schedules could affect an organism's behavior in different ways. Their research showed that the timing and methods of reinforcement greatly influenced the strength and consistency of behaviors. The main concepts of this theory are introduced below.

2.1. Reinforcement schedules

The process of reinforcement is not simple, and many factors can affect how quickly and effectively a new skill is retained. According to Ferster and Skinner (1957), the frequency and timing of reinforcement are influenced by how quickly and easily behaviors are learned by learners. In other words, how old and new behaviors are taught depends on the time and frequency of the reinforcement process. Skinner developed several distinct reinforcement schedules that can affect the process of operant conditioning (Domjan, 1997; Staddon & Cerutti, 2003). These reinforcement schedules are described as follows:

- **Continuous reinforcement:** This type of reinforcement includes delivering reinforcement when a response occurs. Learning is usually relatively fast, but the response rate is fairly low. Extinction also occurs very quickly when enhancements are stopped.

- **Fixed-ratio schedules:**: This is type of partial reinforcement where learners' behaviors are reinforced periodically after a certain number of responses. It usually results in a fairly consistent response rate.
- **Fixed-interval schedules:** This type of reinforcement is only applied after a certain amount of time has elapsed time. The response rate remains fairly constant and begins to increase as the reinforcement time approaches, but it increases slowly as the reinforcement is delivered.
- **Variable-ratio schedules:** In this type of partial reinforcement, a behavior is reinforced after different numbers of responses. This results in a higher response rate and a slower extinction rate.
- **Variable-interval schedules:** Skinner describes the variable-interval schedules as the final form of partial reinforcement. In this type of schedule, reinforcement is delivered after a variable period. This tends to result in a faster response rate and a slower extinction rate.

2.2. Negative and positive reinforcement

Positive reinforcement: This occurs when a favorable event or result is presented after a certain behavior or response. Positive reinforcements, such as direct rewards or praise, are included to strengthen the behavior. For example, when you demonstrate excellent performance at work, you receive a bonus from your boss.

Negative reinforcement: This type of reinforcement aims to remove unfavorable consequences after a specific behavior occurs. By eliminating what is unpleasant or unfavorable, the response is strengthened in

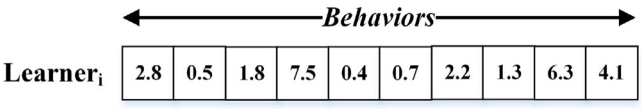


Fig. 2. Representation of a learner (solution) and behaviors (decision variables).

these situations. For instance, if your child starts screaming in a store but stops when you give them a treat, you are more likely to give the child a treat when they scream again. Consequently, your behavior eliminates unpleasant conditions (the child's crying) and negatively reinforces your behavior.

Fig. 1, illustrates an example of positive and negative reinforcements in the PRE theory. When the dog (learner) is stimulated by a bell to deliver the ball (response), the dog will be positively reinforced and deliver the ball. Alternatively, the dog will receive negative reinforcement.

3. Proposed Partial Reinforcement Optimizer (PRO) algorithm

In this section, the PRO algorithm will be described in detail. First, the preliminaries of the algorithm will be explained, including how it is inspired by and modeled based on the PRE theory. Then, the PRO algorithm will be described step by step.

3.1. Proposed Partial Reinforcement Optimizer (PRO) algorithm

In order to map the rules and concepts of the PRE theory to the components of the PRO algorithm, it is necessary to model the PRE theory as an optimization algorithm. The following assumptions are considered:

Learner: A learner is a person or animal whose behaviors need to be trained/improved using the PRE theory, and it is modeled as a solution.

Behavior: A learner's behavior is considered as a decision variable solution. In other words, a solution (learner) is a vector of decision variables (behaviors), Fig. 2.

Population: In the PRO algorithm, a group of learners forms a population. Each row in Fig. 3 represents a solution (learner) and each element X_i^d represents a decision variable (behavior).

Fitness (objective-function) evaluation: The fitness of behaviors for each learner X_i is calculated using a user-defined objective function $f_{X_i} \leftarrow F(X_i)$ on the decision variables (behaviors) $\{X_i^1, X_i^2, X_i^3, \ldots, X_i^N\}$.

Time (Interval): The number of iterations between two stimulations, evaluations, or reinforcement phases, which specifies the decision variables (behaviors) of a learner during the search process, is considered as a time interval. In the proposed algorithm, a scoring mechanism is used, so that a behavior with a higher score has a higher chance of being reinforced in the next iterations.

Response: The main goal is to obtain more responses. In this study, a response is defined as a successful improvement in the objective function value. Therefore, F(X′) < F(X), where F(X') and F(X) are the current and the previous objective-function values of a specific solution after the stimulation phase, respectively.

Schedule: The concept of a schedule is how and when behaviors need to be reinforced, being modeled for a data structure during different intervals. As shown in Fig. 4, each learner has a specific schedule. Since each scaler represents the score/priority of a specific learner's behavior with a higher score/priority has a higher chance of being selected in the next iteration. Furthermore, the variable-interval scheduling scheme (Staddon & Cerutti, 2003) is modeled as a dynamic mechanism to conduct a stochastic analysis using Eqs. (1)–(3). How and when behaviors need to be reinforced is a variable and is updated during the search process. Therefore, a subset of behaviors $\mu \subseteq \{1, 2, 3, \ldots, N\}$, with the highest priorities in schedule i is selected at each iteration for $learner_i$. In this way, first, the priorities in the schedule are sorted in descending order, then, the first λ number of items are selected as the candidate behaviors using Eq. (3).

$$\tau \leftarrow \frac{FEs}{MaxFEs} \tag{1}$$

$$SR \leftarrow e^{-(1-\tau)} \tag{2}$$

$$\mu \subseteq \{1, 2, 3, \ldots, N\} \quad \big| \forall j \in \mu, Schedule^j \geq Schedule^{*,\lambda},$$
$$\lambda \leftarrow \{ \ \|\mu\| \ \big| \ \|\mu\| = \lceil U(1, \ N \times SR) \rceil \ \} \tag{3}$$

where, τ is the time factor, FEs is the number of function evaluations, and MaxFEs is the maximum number of function evaluations. Also, SR is the selection rate, μ is a subset of behaviors selected based on scheduling, λ is the size of selected subset, and N is the total number of behaviors (decision variables). Schedule* denotes the schedule with sorted priorities and $Schedule^{*,\lambda}$ is the λth item in the Schedule*.

Stimulation: Any attempt to stimulate a learner's behaviors to elicit a response is modeled by applying operations that change the decision variables of a proposed solution. It should be noted that any operation can be applied to stimulate (change) learners' behaviors (decision variables). In the PRO algorithm, the following operations are utilized to generate new solutions, as outlined in Eqs. (4)–(6).

$$SF_i \leftarrow \tau + U(0, \bar{\beta}), \text{ where } \bar{\beta} \leftarrow \sum_{j \in \mu} \left(\frac{Schedule_{i,j}}{Max\left(Schedule_i\right)} \right) \tag{4}$$

$$S_i^{\mu} \leftarrow \begin{cases} (X_{best}^{\mu} - X_i^{\mu}) & \text{If rand} < 0.5 \\ (X_i^{\mu} - X_j^{\mu}) & \text{Otherwise.} \end{cases} \tag{5}$$

$$X_{i,new}^{\mu} \leftarrow X_i^{\mu} + SF_i \times S_i^{\mu} \tag{6}$$

where, SF_i is the stimulation factor and $\bar{\beta}$ is the mean of a normalized score/priority of the selected decision variables for ith learner based on its scheduler.

Reinforcement: To conceptualize reinforcement, we use the following mechanism to update the scheduling. Positive reinforcement is then applied to increase the score of a specific behavior. The learner's objective function, Eq. (7), is used as a response after the improvement in the stimulation phase.

$$Schedule_i^{\mu} \leftarrow Schedule_i^{\mu} + \left(Schedule_i^{\mu} \times RR\right) \tag{7}$$

where, RR is the reinforcement rate, $Schedule_i^{\mu}$ represents priorities of the selected decision variables (behaviors) for the i*th* solution (learner).

On the other hand, negative reinforcement is applied when there is no response. In this situation, the objective function of a learner decreases after the stimulation phase, resulting in a decrease in the score of specific behavior (Eq. (8)). The decision variables (behaviors) with higher scores will be selected for stimulation and reinforcement in the next iteration.

$$Schedule_i^{\mu} \leftarrow Schedule_i^{\mu} - \left(Schedule_i^{\mu} \times RR\right) \tag{8}$$

Rescheduling: This concept refers to the process of applying a new schedule for a learner during training, when the learner consistently receives negative reinforcement for all behaviors. In this case, the PRO utilizes the standard deviation (Std) of the schedule as a metric to determine when it is necessary to reschedule the learner. This mechanism is applied by using Eqs. (9) and (10).

$$\text{Schedule}_i \leftarrow \begin{cases} U(0,1) & \text{if Std(Schedule}_i) = 0 \\ Do \text{ nothing} & \text{otherwise} \end{cases} \tag{9}$$

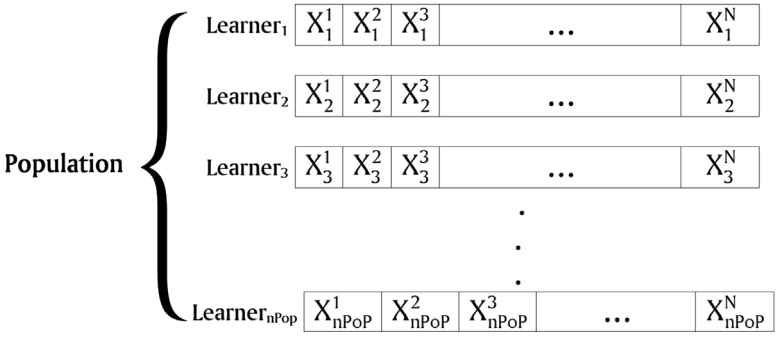


Fig. 3. Representation of the population.

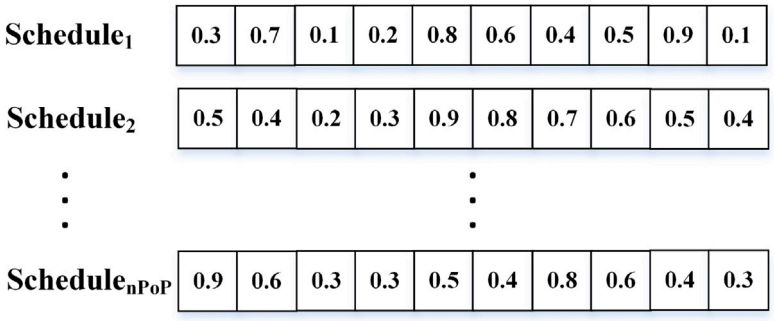


Fig. 4. Representation of a learner (solution) and behaviors (decision variables).

$$X_i \leftarrow \begin{cases} U(L_B, U_B) & \text{if Std}(\text{Schedule}_i) = 0 \\ Do\ \text{nothing} & \text{otherwise} \end{cases} \quad (10)$$

where, Std(Schedule$_i$) is the standard deviation of the schedule of ith learner, L_B and U_B are the lower bound and upper bound, respectively. Also, U(0,1) and U(L_B,U_B) refer to random values with uniform distribution between (0,1) and (L_B,U_B).

3.2. The PRO algorithm

This section describes the PRO algorithm based on the beforementioned preliminaries. Furthermore, the flowchart of the proposed PRO algorithm is represented in Fig. 5 to provide more clarification. According to Algorithm 1, the population and hyper-parameters are initialized before the searching process of the PRO. Then, an iterative process is performed to optimize the objective function. In doing so, at first, candidate decision variables (behaviors) of i*th* solution (learner) X_i is selected according to the corresponding scheduler (lines 7–9). After that, an operation is randomly selected to stimulate the selected decision variables of the i*th* solution X_i for producing a new solution $X_{i,new}$ (lines 10–11). In line 12, the new solution is clipped to keep decision variables in a feasible range. The new solution $X_{i,new}$ is evaluated in (line 13), which is based on its objective-function $f'_X \leftarrow F(X_{i,new})$Then, the comparison process is conducted to apply positive (line 16) or negative (line 19) reinforcements according to the response (lines 14–20). The best solution X_{best} is updated in line 21 and the rescheduling process is launched in line 22. Finally, the best solution will be returned as a result.

3.3. A visualization example

In this section, the process of scheduling and partial reinforcement mechanism in the PRO algorithm is illustrated by a visualization example. Each line in Table 2 represents an iteration of the PRO, and the following processes are conducted at each iteration: (a) A decision

Algorithm 1 The PRO Algorithm

1: **Initialization:**
2: *Initialize Population*
3: *Initialize Schedules*
4: *FEs, MaxFEs, SF, RR, SR*
5: **while** FEs ≤ MaxFEs) **do**
6: **for** i = 1 to nPop **do** ▷ for all learners.
▷ Determine Behaviors of the i^{th} learner based on the scheduler.
7: Calculate time parameter, Eq. (1).
8: Calculate selection rate (SR) based on Eq. (2).
9: Select λ number of behaviors with highest priority in Schedule$_i$ based on Eq. (3)
▷ Stimulate the selected behaviors of the i^{th} learner to get response, Eqs. (4)–(6)
10: Update Beta and SF according to Eqs. (4)–(5)
11: Calculate $X^{\mu}_{i,new}$according to Eq. (6)
12: Apply bound constraints ▷ Evaluate the i^{th} learner response
13: $f'_X \leftarrow F\left(X_{i,new}\right)$
▷ Apply Positive or negative reinforcement according to the response.
14: **if** (f'_X is better than f_X) **then**
15: Accept new behavior $X^{\mu}_{i,new}$
16: Apply positive reinforcement, Eq. (7)
17: **else**
18: Reject new behavior $X^{\mu}_{i,new}$
19: Apply negative reinforcement, Eq. (8)
20: **end if**
21: Update the best solution (X_{best})
22: Conduct rescheduling process according to Eqs. (9)–(10))
23: FEs = FEs + 1
24: **end for**
25: **end while**
return the best solution (X_{best})

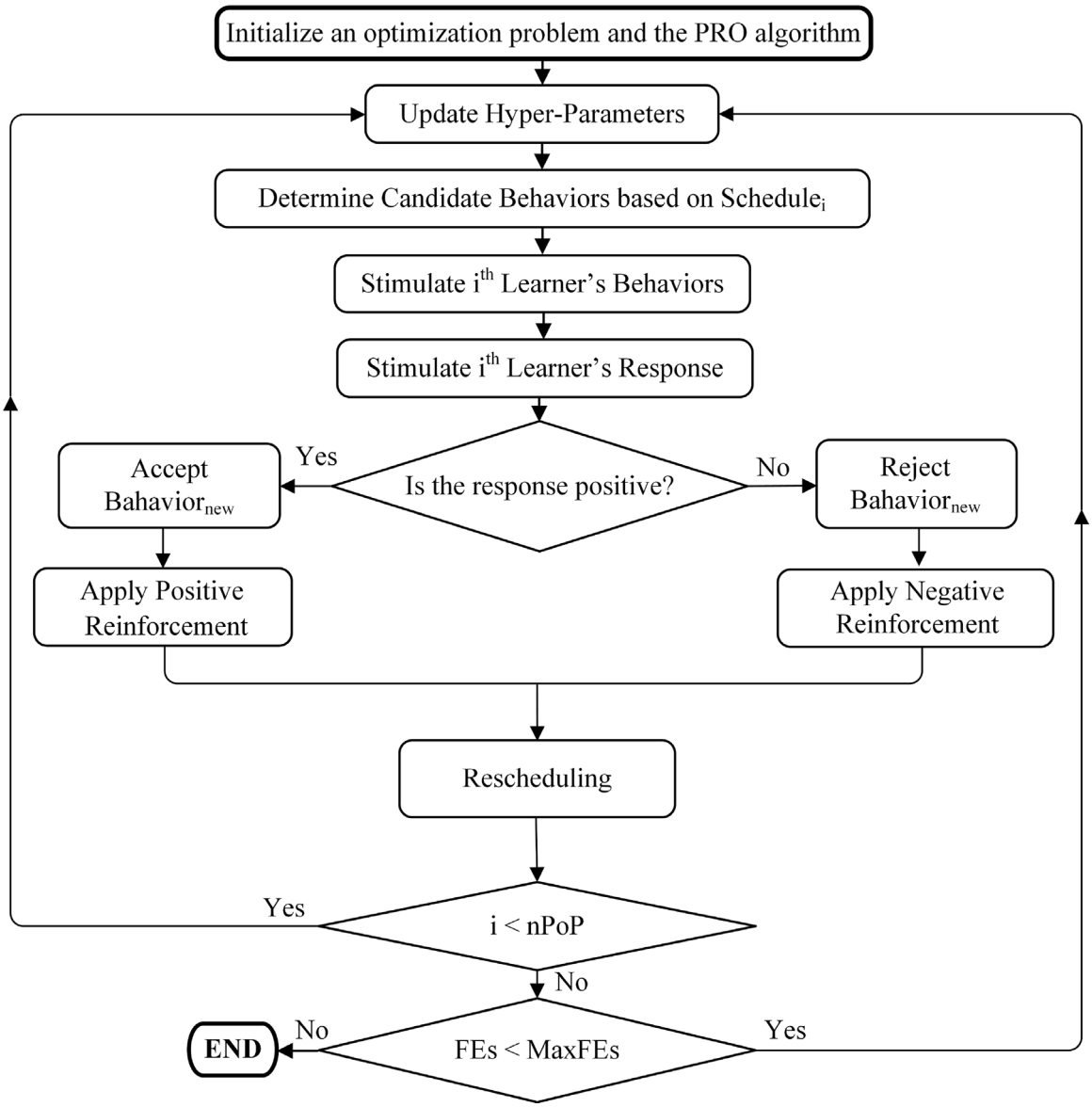


Fig. 5. The proposed PRO algorithm.

variable (behavior) is selected based on the schedule, (b) the selected decision variable (behavior) is updated/stimulated, (c) the new position is evaluated by F_x, and (d) the learner (position) is positively or negatively reinforced based on its response.

In Fig. 6, T_0 represents the initial position. T_1 and X_2 are selected based on their priority (0.6) in the schedule and updated/stimulated. The new position for $X_1 = -8$ and $X_2 = +1$. Then, as the learner could not achieve better F_x, the learner will be negatively reinforced (the priority of X_2 in the schedule is decreased to 0.3). In T_2, the schedule selects X_1 and updated. The new position for $X_1 = -6$ and $X_2 = -3$. As the new position obtains a better F_x, positive reinforcement is applied and the priority of X_1 in the schedule is increased to 0.6. The search process continues through T_3 and T_4 by selecting and updating X_1. Finally, in T_5, the optimum value is found and the search process is terminated. As shown, a dynamic scheduling technique is specially tailored for each learner as a search agent. In this way, a priority value is kept by the schedule for each behavior (decision variable) to determine which decision variables must be updated at a certain time. A scheduler acts like history to guide the algorithm towards a direction that has more potential to find the optimum solution with respect to the search space and optimization perspective.

Table 2
Interaction of the PRO Algorithm.

T	Schedule		Learner		F_X	Reinforcement
	X_1	X_2	X_1	X_2		
0	0.4	0.6	−8	−3	5	–
1	0.4	0.6[a]	−8	+1	5	Negative
2	0.4[a]	0.3	−6	−3	4	Positive
3	0.6[a]	0.3	0	−3	2	Positive
4	0.8[a]	0.3	+5	−3	3	Negative
5	0.2	0.3	0	0	0	Positive

[a] Indicates a selected behavior by the scheduler.

4. Experimental study

4.1. Benchmarks

In order to evaluate the performance of the proposed PRO algorithm, a set of 75 extremely complex mathematical benchmark functions has been selected from well-known test suites including IEEE

Table 3
Summary of the unimodal benchmark functions.

Function No.	Functions	Search range	$F_i^* = F_i(X^*)$
F01	Shifted sphere function	$[-100, 100]^D$	−450
F02	Shifted Schwefel's problem 1.2	$[-100, 100]^D$	−450
F03	Shifted rotated high conditioned elliptic function	$[-100, 100]^D$	−450
F04	Shifted Schwefel's problem 1.2 with Noise in Fitness	$[-100, 100]^D$	−450
F05	Schwefel's problem 2.6 with global optimum on bounds	$[-100, 100]^D$	−310
F06	Rotated high conditioned elliptic function	$[-100, 100]^D$	100
F07	Rotated bent cigar function	$[-100, 100]^D$	200
F08	Rotated discus function	$[-100, 100]^D$	300

Table 4
Summary of the multimodal benchmark functions.

Function No.	Functions	Search range	$F_i^* = F_i(X^*)$
F09	Shifted Rosenbrock's function	$[-100, 100]^D$	390
F10	Shifted rotated Griewank's function without bounds	$[-0, 600]^D$	−180
F11	Shifted rotated Ackley's function with global optimum on bounds	$[-32, 32]^D$	−140
F12	Shifted Rastrigin's function	$[-5, 5]^D$	−330
F13	Shifted rotated Rastrigin's function	$[-5, 5]^D$	−330
F14	Shifted rotated Weierstrass function	$[-5, 5]^D$	90
F15	Schwefel's problem 2.13	$[-\pi, \pi]^D$	−460
F16	Shifted and rotated Rosenbrock's function	$[-100, 100]^D$	400
F17	Shifted and rotated Ackley's function	$[-100, 100]^D$	500
F18	Shifted and rotated Weierstrass function	$[-100, 100]^D$	600
F19	Shifted and rotated Griewank's function	$[-100, 100]^D$	700
F20	Shifted Rastrigin's function	$[-100, 100]^D$	800
F21	Shifted and rotated Rastrigin's function	$[-100, 100]^D$	900
F22	Shifted Schwefel's function	$[-100, 100]^D$	1000
F23	Shifted and rotated Schwefel's function	$[-100, 100]^D$	1100
F24	Shifted and rotated Katsuura function	$[-100, 100]^D$	1200
F25	Shifted and rotated HappyCat function	$[-100, 100]^D$	1300
F26	Shifted and rotated HGBat function	$[-100, 100]^D$	1400
F27	Shifted and rotated expanded Griewank's plus Rosenbrock's unction	$[-100, 100]^D$	1500
F28	Shifted and rotated expanded Scaffer's F6 function	$[-100, 100]^D$	1600

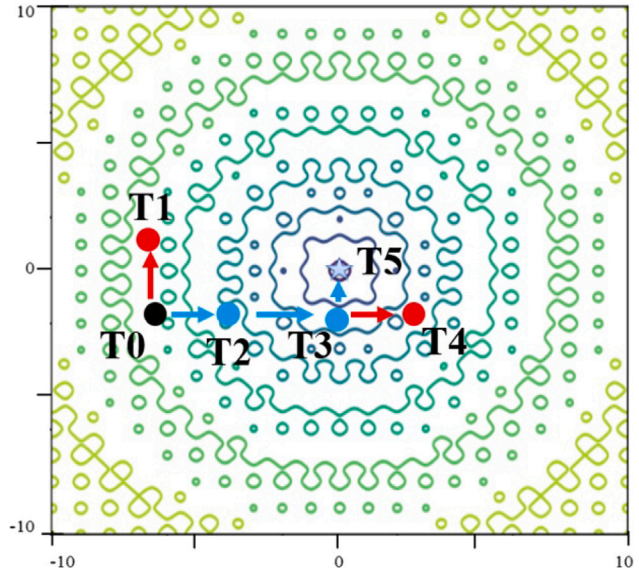


Fig. 6. Example of scheduling and partial reinforcement mechanism in the PRO.

CEC2005 (Suganthan et al., 2005), IEEE CEC2014 (Liang et al., 2014), and CEC-BC-2017 benchmarks (Awad et al., 2017), and applyied by the PRO algorithm to minimize these benchmark functions. These benchmark functions are categorized into four classes: (a) Unimodal, (b) Multimodal, (c) Hybrid, and (d) Composition functions. Furthermore, to investigate the efficiency of the proposed algorithm, the PRO is applied to solve several real-world optimization problems. The 3-D maps for 2-Dimensional benchmark functions are illustrated in Appendix.

4.2. Mathematical benchmark functions

4.2.1. Unimodal functions

Table 3 presents a summary of the unimodal benchmark, including functions F01–F08. Functions F01–F03 are selected from CEC2005 and functions F04–F08 are chosen from CEC2014. According to the characteristics of this group of benchmark functions, each function has exactly one global optimum and no local optima. These functions are suitable for testing intensification and exploitation capabilities of an optimization algorithm.

4.2.2. Multimodal functions

Multimodal functions include a large number of local optima. Functions F09–F28 of Table 4 represent multimodal functions. This type of objective function can be used to evaluate the exploration and diversification capabilities of an optimization algorithm, as well as its ability to avoid local optima.

4.2.3. Hybrid functions

Table 5 includes hybrid benchmark functions F29–F37. Each hybrid function includes several subcomponents. In real-world optimization problems, hybrid functions may include different properties of both basic unimodal and multimodal functions (Liang et al., 2014). Hence, this kind of benchmark function is suitable to simultaneously investigate the exploration and exploitation capabilities of an algorithm.

4.2.4. Composition functions

The composition functions employ the hybrid functions as the basic functions. These functions merge the different properties of the subcomponents better than the hybrid benchmark functions and are extremely complex benchmarks. This kind of benchmark functions is able to keep continuity around the global or local optima (Liang et al., 2013). In an experimental level, the complexity of these benchmark functions is increased significantly because of their shifted and rotated

Table 5
Summary of the hybrid benchmark functions.

Function No.	Functions	Search range	$F_i^* = F_i(X^*)$
F29	Expanded extended Griewank's plus Rosenbrock's function	$[-3, 1]^D$	−130
F30	Shifted rotated expanded Scaffer's F6	$[-100, 100]^D$	−300
F31	Hybrid composition function	$[-5, 5]^D$	120
F32	Hybrid function 1 (N = 3)	$[-100, 100]^D$	1700
F33	Hybrid function 2 (N = 3)	$[-100, 100]^D$	1800
F34	Hybrid function 3 (N = 4)	$[-100, 100]^D$	1900
F35	Hybrid function 4 (N = 4)	$[-100, 100]^D$	2000
F36	Hybrid function 5 (N = 5)	$[-100, 100]^D$	2100
F37	Hybrid function 6 (N = 5)	$[-100, 100]^D$	2200

Table 6
Summary of the composition benchmark functions.

Function No.	Functions	Search range	$F_i^* = F_i(X^*)$
F38	Composition function 1 (N = 5)	$[-100, 100]^D$	2300
F39	Composition function 2 (N = 3)	$[-100, 100]^D$	2400
F40	Composition function 3 (N = 3)	$[-100, 100]^D$	2500
F41	Composition function 4 (N = 5)	$[-100, 100]^D$	2600
F42	Composition function 5 (N = 5)	$[-100, 100]^D$	2700
F43	Composition function 6 (N = 5)	$[-100, 100]^D$	2800
F44	Composition function 7 (N = 3)	$[-100, 100]^D$	2900
F45	Composition function 8 (N = 3)	$[-100, 100]^D$	3000

characteristics. A summary of the composition benchmark functions is presented in Table 6.

4.3. Performance criteria

In this study, the mean and standard deviation errors (SD) of objective functions are considered to present the results. The value of $f_{min}(X)$-F(X*) is used to compute the objective function error. Here, $f_{min}(X)$ represents the minimum value of the objective function achieved by the optimization algorithm during the search process and F(X*) represents the global optimum (X*) of the objective function value in each test function (Bujok et al., 2019). The minimum function error is considered to be zero if its value is less than 1×10^{-8}. According to the recommended implementation conditions in Liang et al. (2013), such an error function value is considered a fair approximation of the global optimum solution. The overall and second-best results are highlighted in boldface and italic, respectively.

4.4. Implementation settings

The settings used in all the experimental studies are as follows: To implement and program the proposed PRO and other algorithms, MATLAB software is used and each algorithm is independently run 30 times for each benchmark function. The maximum number of function evaluations (MaxFEs) for all benchmark functions is set to D × 104, where D is the dimension of the problem (Liang et al., 2013, 2014). It is necessary to mention that some results in this study are taken from the previous work of the authors (Taheri, RahimiZadeh et al., 2021). The other parameters for ACPSO (Hasanzadeh et al., 2013), WOA (Mirjalili & Lewis, 2016), IGHS (Xiang et al., 2014), IHS (Mahdavi et al., 2007), and DOLTLBO (Xu et al., 2020) are same as those used in the corresponding literature. The parameters for WOA are set to recommended values nPop = 30, and A = [2,0]. For SCA algorithm nPop = 30. And also for the IHS, HMCR = 0.9, HMS = 5, PAR_{min} = 1.00E-2, PAR_{max} = 9.90E−1, bw_{min} = 1.00E-4 and $bw_{max} = (U_B - L_B)/20$. For IGHS algorithm, $HMCR_{min}$ = 9.00E−1, HMS = 20, PAR_{min} = 9.00E−1, PAR_{max} = 9.50E−1, $HMCR_{max}$ = 9.90E−1, β = 5.00E−2, α = 2.50E−1, and ζ = 1.00E−1. Jr = 0.3, weight = 10, and nPop = 100 are the settings for DOLTLBO. For ACPSO algorithm, nPop is set to 30, α = β = 1.00E−1 and swarm's number is 6. In addition, the parameter settings for, Bat (Yang, 2010), HFPSO, PSO (Eberhart & Kennedy, 1995), and DE algorithm (Wolpert & Macready, 1997) are set based on the recommended values in Bujok et al. (2019). The control parameters, c_1 and c_2, for HFPSO are set to 1.49445, β_0 = 2, α = 0.2, γ = 1, and nPop is set to 50. For PSO algorithm, the recommended settings are used, c_1,c_2 = 1.05, w_{max} = 1, w_{min} = 0.3, and nPop = 40. For DE the control parameters, nPop and CR are set to 50 and 0.2, respectively. And also for Bat, f_{min} = 0, f_{max} = 2, γ = 0.9, A_i = 1.2, r_i = 0.1, and nPop is set to 90.

4.5. Discussion

4.5.1. Unimodal functions

According to the unimodal benchmark functions (F01-F08) results presented in Table 7, the proposed PRO algorithm obtains the best results for functions F01, F02, F03, F06, F07, F08, and also the second-best for function F04. However, for unimodal function F05, our proposed algorithm achieves the 5th best place among 12 algorithms and the obtained result is acceptable. These results demonstrate that the PRO algorithm is capable enough to perform strong exploitation better than other competitors. In addition, the PRO algorithm finds the optimum values for functions F01, F02, F07, and F08.

4.5.2. Multimodal functions

Table 8 presents the results for multimodal benchmark functions. As reported, the proposed PRO helps to surmount multimodal optimization problems in solving benchmark functions (F09-F28). The PRO algorithm obtains the best results for 8 benchmark functions F09, F13, F19, F20, F21, F25, F26, and F27. And also, our proposed algorithm obtains the second-best rank for functions F10, F23, and F28. However, for multimodal functions F12, F15 and F24, our proposed algorithm achieves its worst results (the 5th, 7th and 7th best place among 12 competitors, respectively). It is reasonable because of the stochastic nature of the PRO and due to the theory of "No Free Lunch", which describes the fact that there is no single method or algorithm that is suitable for all kinds of problems (Wolpert & Macready, 1997). Thus, it can be proven from the results that the PRO is able enough to keep diversification at an acceptable level and avoid trapping to local optima in solving problems with a huge number of local optimum.

4.5.3. Hybrid functions

For the hybrid functions (F29–F37), which are suitable to simultaneously investigate the exploration and exploitation abilities of the optimization algorithms, the proposed PRO algorithm gets the best rank and outperforms other MAs, and shows better performance by obtaining the best results for functions F31, F32, F33, F34, F35 and F36, Table 9. It is clear to be seen from the results that the proposed PRO algorithm is able to simultaneously keep a balance between exploitation and exploration which is essential to deal with complex optimization problems.

4.5.4. Composition functions

Table 10 presents the results for composition benchmark functions (F38–F45). According to these results, the best ranks for benchmark functions F41, F44, and F45 are obtained by the PRO algorithm. Furthermore, for functions F42 and F43, the second-best results are achieved by the proposed PRO algorithm.

Table 7
Comparison results of the PRO and the state-of-the-art algorithms on unimodal test functions.

Function		TLBO	Bat	DE	HFPSO	ACPSO	PSO	IHS	IGHS	SCA	WOA	DOLTLBO	PRO
F1	Mean	**0.00E+00**	2.94E+04	**0.00E+00**	**0.00E+00**	1.69E+02	1.66E+03	*1.63E−07*	**0.00E+00**	5.40E+04	4.39E−01	**0.00E+00**	**0.00E+00**
	SD	**0.00E+00**	6.41E+03	**0.00E+00**	**0.00E+00**	3.54E+02	1.72E+03	*2.01E−08*	**0.00E+00**	9.78E+03	1.55E−01	**0.00E+00**	**0.00E+00**
F2	Mean	**0.00E+00**	3.28E+04	2.08E+03	**0.00E+00**	2.45E+03	2.21E+02	7.04E+02	*1.41E+00*	9.69E+04	5.18E+04	**0.00E+00**	**0.00E+00**
	SD	**0.00E+00**	9.09E+03	3.32E+02	**0.00E+00**	3.80E+03	2.08E+02	3.04E+02	*1.22E+00*	1.71E+04	8.27E+03	**0.00E+00**	**0.00E+00**
F3	Mean	7.00E+05	1.56E+08	5.30E+07	*6.37E+05*	6.98E+06	3.88E+06	7.43E+06	5.63E+06	7.86E+08	2.87E+07	6.51E+05	**5.70E+05**
	SD	5.80E+05	8.61E+07	1.22E+07	*3.17E+05*	4.02E+06	2.71E+06	3.09E+06	2.92E+06	1.90E+08	1.10E+07	4.27E+05	**4.32E+05**
F4	Mean	**8.50E+01**	1.21E+05	8.28E+03	3.16E+02	1.20E+04	3.83E+02	4.83E+03	4.46E+02	1.20E+05	1.58E+05	2.15E+02	*1.76E+02*
	SD	**1.91E+02**	3.21E+04	2.08E+03	7.02E+02	9.07E+03	5.11E+02	2.87E+03	2.18E+02	2.38E+04	5.10E+04	2.21E+02	*1.12E+02*
F5	Mean	3.41E+03	2.75E+04	*2.85E+03*	4.86E+03	7.49E+03	6.29E+03	5.14E+03	**2.54E+03**	2.68E+04	1.72E+04	3.32E+03	3.26E+03
	SD	8.00E+02	2.96E+03	*4.95E+02*	2.08E+03	2.56E+03	2.85E+03	1.22E+03	**3.98E+02**	2.29E+03	5.22E+03	5.30E+02	2.26E+02
F6	Mean	2.75+05	3.05E+08	2.84E+07	4.11E+05	5.40E+06	2.44E+06	7.40E+06	2.02E+06	1.30E+09	3.29E+07	*2.44E+05*	**5.46E+04**
	SD	1.80+05	1.24E+08	6.99E+06	2.56E+05	3.69E+06	3.10E+06	6.16E+06	2.44E+06	3.21E+08	1.49E+07	*1.16E+05*	**3.15E+04**
F7	Mean	*8.15-01*	3.12E+10	1.79E+01	1.16E+03	1.72E+04	4.39E+08	9.33E+03	7.82E+03	8.64E+10	3.53E+06	4.18E+03	**0.00E+00**
	SD	*1.84+00*	9.77E+09	5.78E+01	6.29E+03	1.16E+04	9.09E+08	9.50E+03	8.04E+03	1.53E+10	3.54E+06	7.24E+03	**0.00E+00**
F8	Mean	4.33+01	1.08E+05	2.21E+02	4.18E+00	1.24E+04	5.81E+02	6.37E+03	3.93E+03	2.31E+05	2.91E+04	*6.50E−01*	**0.00E+00**
	SD	4.19+01	2.29E+04	2.10E+02	1.70E+01	1.61E+04	1.82E+03	6.03E+03	3.43E+03	3.98E+04	2.14E+04	*2.08E+00*	**0.00E+00**

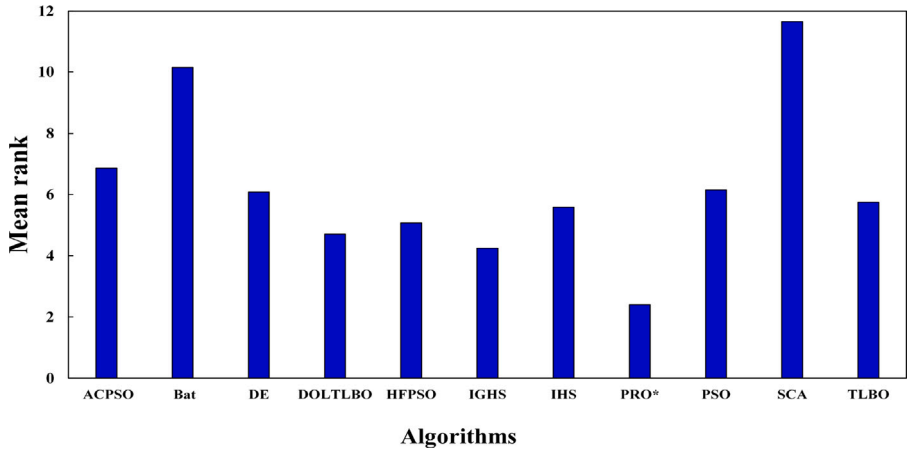


Fig. 7. Rankings of the PRO and the state-of-the-art algorithms based on the Friedman test on IEEE CEC test suites (D = 30).

4.5.5. Statistical analysis

In this study, two well-known non-parametric statistical tests named Friedman and Wilcoxon tests (Corder & Foreman, 2009) are used to statistically analyze the experimental results and then to provide fair comparisons. To carry out these non-parametric statistics tests, prior assumptions of data samples are not required (Corder & Foreman, 2009). The Friedman test is utilized to determine algorithms rank on specific functions. Furthermore, to identify differences between the pair of the proposed PRO and other algorithms for a specific number of problem functions, a multi-problem Wilcoxon signed-rank test with two significant levels, 1% and 5%, is employed. The multiple-problem Wilcoxon signed-rank test between the PRO and the state-of-the-art population-based optimization algorithms is presented in Table 11. It can be seen from the results that the PRO algorithm achieves higher positive ranks (R^+) than WOA, DE, HFPSO, PSO, ACPSO, Bat, IHS, IGHS, SCA, TLBO. There is a considerable difference between the PRO and other metaheuristic optimization algorithms for both $\alpha = 0.05$ and $\alpha = 0.01$. Furthermore, Fig. 7 and Table 12 represent the mean rankings and descriptive statistics of the Friedman test for the PRO and state-of-the-art MAs. According to the results, the proposed PRO algorithm obtains the first rank on all types of benchmark functions. In addition, IGHS obtains the second rank, followed by DOLTLBO, HFPSO, IHS, and TLBO.

4.6. Performance of PRO on the CEC-BC-2017 benchmarks

To further investigate the performance of our proposed PRO algorithm, we used CEC-BC-2017 (Awad et al., 2017). This test suit is one of the challenging real-parameter numerical optimization benchmarks and contains 30 functions including Unimodal, Multimodal, Hybrid, and Composition functions. The mathematical formulation details of this test suite is presented in Awad et al. (2017). We compare the PRO results with the recently proposed novel MAs and other most well-known evolutionary methods such as Genetic Algorithm (GA) (Holland, 1992), Success-History Based Parameter Adaptation Differential Evolution (SHADE) (Tanabe & Fukunaga, 2013), and Covariance Matrix Adaptation Evolution Strategy (CMAES) (Hansen et al., 2003). SHADE-based algorithms use a history of the best previously found control parameters, such crossover rate and scaling factor, to conduct an adaptive parameter mechanism during each run. On the other hand, the PRO algorithm utilizes a memory-based dynamic scheduling scheme based on PRE theory to define stimulate and reinforce a specific behavior. It is important to note that the PRO does not store any parameters for applying parameter adaptation mechanism, although there are some similarities with other MAs. Please note that some of the results and implementation settings used in this section are taken from Faramarzi, Heidarinejad, Mirjalili et al. (2020). The statistical results including mean and standard deviation (SD) are presented in Table 13. According to these results, our proposed method obtains the best results for 11 functions and the second best for 7 functions, and the SHADE algorithm shows better performance on the test suite CEC-BC-2017 and achieves the best results for 12 functions and the second best results for 4 functions. The third place is taken by the Young's Double-Slit Experiment (YDSE) algorithm (Abdel-Basset et al., 2023) which achieves the best results for 8 benchmark functions.

Table 8
Comparison results of the PRO and the state-of-the-art algorithms on multimodal test functions.

Function		CHIO	HGS	RSO	SCA	SMA	WOA	YDSE	MPA	GA	CMAES	SHADE	AOA	PRO
F1	Mean	2.04E+03	7.07E+03	2.66E+09	4.60E+08	8.07E+03	6.19E+04	**1.00E+02**	**1.00E+02**	9.80E+03	**1.00E+02**	**1.00E+02**	3.05E+09	**1.00E+02**
	SD	1.68E+03	4.08E+03	1.66E+09	2.01E+08	4.26E+03	1.53E+05	**0.00E+00**	**4.79E−06**	5.94E+03	**0.00E+00**	**0.00E+00**	1.83E+09	**0.00E+00**
F3	Mean	4.22E+03	3.00E+02	4.53E+03	9.26E+02	3.00E+02	4.19E+02	**3.00E+02**	**3.00E+02**	8.72E+03	**3.00E+02**	**3.00E+02**	4.02E+03	**3.00E+02**
	SD	2.38E+03	0.00E+00	2.55E+03	4.28E+02	8.36E−05	1.09E+02	**0.00E+00**	**0.00E+00**	5.90E+03	**0.00E+00**	**0.00E+00**	1.80E+03	**0.00E+00**
F4	Mean	4.01E+02	4.05E+02	6.11E+02	4.29E+02	4.03E+02	4.16E+02	**4.00E+02**	*4.00E+02*	4.11E+02	**4.00E+02**	**4.00E+02**	5.72E+02	**4.00E+02**
	SD	1.13E+00	1.61E+01	1.41E+02	1.06E+01	8.91E−01	3.08E+01	**0.00E+00**	*5.70E−02*	1.85E+01	**0.00E+00**	**0.00E+00**	1.12E+02	**0.00E+00**
F5	Mean	5.09E+02	5.17E+02	5.67E+02	5.39E+02	5.12E+02	5.50E+02	5.06E+02	5.10E+02	5.16E+02	5.30E+02	**5.04E+02**	5.57E+02	*5.05E+02*
	SD	2.63E+00	7.34E+00	1.11E+01	7.21E+00	6.09E+00	1.85E+01	2.43E+00	3.96E+00	6.93E+00	5.83E+01	**1.01E+00**	2.66E+01	*2.22E+00*
F6	Mean	6.00E+02	6.00E+02	6.41E+02	6.15E+02	6.00E+02	6.29E+02	6.00E+02	6.00E+02	6.00E+02	6.82E+02	*6.00E+02*	6.38E+02	**6.00E+02**
	SD	4.01E−03	7.20E−01	6.42E+00	3.57E+00	1.51E−02	1.26E+01	2.62E−03	6.16E−04	6.69E−02	3.54E+01	*2.65E−07*	8.83E+00	**0.00E+00**
F7	Mean	7.19E+02	7.26E+02	7.96E+02	7.65E+02	7.21E+02	7.75E+02	7.16E+02	7.23E+02	7.28E+02	**7.13E+02**	*7.14E+02*	6.38E+02	7.15E+02
	SD	4.03E+00	1.06E+01	1.30E+01	5.71E+00	3.98E+00	1.89E+01	2.11E+00	3.91E+00	7.29E+00	**1.63E+00**	*1.24E+00*	8.83E+00	2.63E+00
F8	Mean	8.10E+02	8.17E+02	8.38E+02	8.32E+02	8.12E+02	8.36E+02	8.06E+02	8.09E+02	8.21E+02	8.29E+02	**8.04E+02**	8.33E+02	*8.05E+02*
	SD	2.54E+00	6.41E+00	9.78E+00	5.53E+00	5.67E+00	1.00E+01	2.39E+00	3.12E+00	8.96E+00	5.30E+01	**1.28E+00**	5.53E+00	*2.39E+00*
F9	Mean	9.00E+02	**9.00E+02**	1.24E+03	9.61E+02	*9.00E+02*	1.22E+03	**9.00E+02**	9.00E+02	9.10E+02	4.67E+03	**9.00E+02**	1.33E+03	**9.00E+02**
	SD	2.06E−01	**0.00E+00**	1.37E+02	2.64E+01	*8.29E−05*	2.85E+02	**0.00E+00**	1.63E−02	1.52E+01	2.06E+03	**0.00E+00**	1.38E+02	**0.00E+00**
F10	Mean	1.27E+03	1.43E+03	2.31E+03	2.06E+03	1.58E+03	2.00E+03	1.34E+03	1.44E+03	1.72E+03	2.59E+03	**1.19E+03**	2.11E+03	*1.21E+03*
	SD	9.74E+01	1.97E+02	1.89E+02	1.81E+02	2.28E+02	3.48E+02	1.27E+02	1.41E+02	2.52E+02	4.14E+02	**8.47E+01**	2.93E+02	*1.39E+02*
F11	Mean	1.11E+03	1.11E+03	1.42E+03	1.17E+03	1.11E+03	1.17E+03	**1.10E+03**	1.10E+03	1.13E+03	1.11E+03	1.10E+03	2.11E+03	*1.10E+03*
	SD	2.77E+00	1.06E+01	3.83E+02	1.66E+01	3.21E+00	5.87E+01	**2.52E−01**	1.27E+00	2.38E+01	2.54E+01	1.36E+00	2.93E+02	*1.09E+00*
F12	Mean	1.22E+05	1.72E+04	2.22E+07	5.42E+06	4.36E+04	3.57E+06	**1.20E+03**	*1.25E+03*	3.70E+06	1.63E+03	1.32E+03	7.91E+05	1.47E+03
	SD	1.55E+05	1.42E+04	6.30E+07	4.20E+06	1.75E+04	3.91E+06	**2.16E+01**	*5.43E+01*	3.40E+06	1.98E+02	1.03E+02	3.70E+05	1.33E+02
F13	Mean	3.75E+03	9.29E+03	2.88E+04	1.50E+04	1.14E+04	1.71E+04	1.30E+03	1.31E+03	1.08E+04	1.32E+03	**1.30E+03**	8.58E+03	*1.30E+03*
	SD	4.68E+03	1.06E+04	1.79E+04	9.99E+03	1.15E+04	1.22E+04	2.03E+00	2.58E+00	8.93E+03	7.83E+01	**7.10E−01**	6.96E+03	*7.74E−01*
F14	Mean	1.65E+03	1.67E+03	2.62E+03	1.52E+03	1.43E+03	1.51E+03	*1.40E+03*	1.40E+03	7.05E+03	1.45E+03	1.41E+03	7.15E+03	**1.40E+03**
	SD	2.23E+02	7.89E+02	1.54E+03	3.40E+01	1.10E+01	3.19E+01	*1.13E+00*	4.06E+00	8.16E+03	3.60E+01	9.21E+00	8.24E+03	**9.14E−01**
F15	Mean	1.94E+03	1.95E+03	7.98E+03	1.77E+03	1.51E+03	2.79E+03	*1.50E+03*	1.50E+03	9.30E+03	1.51E+03	1.50E+03	1.19E+04	**1.50E+03**
	SD	4.40E+02	4.94E+02	5.90E+03	1.39E+02	5.54E+00	1.26E+03	*1.75E−01*	5.20E−01	8.98E+03	1.64E+01	3.60E−01	5.42E+03	**1.71E−01**
F16	Mean	1.62E+03	1.74E+03	1.83E+03	1.67E+03	1.68E+03	1.80E+03	**1.60E+03**	*1.60E+03*	1.79E+03	1.82E+03	1.60E+03	2.11E+03	1.61E+03
	SD	4.03E+01	9.90E+01	1.34E+02	2.72E+01	8.20E+01	1.25E+02	**5.82E−01**	*9.90E−01*	1.29E+02	2.30E+02	2.20E+00	9.30E+01	3.77E+01
F17	Mean	*1.70E+03*	1.76E+03	1.78E+03	1.76E+03	1.74E+03	1.79E+03	1.72E+03	1.71E+03	1.75E+03	1.83E+03	1.72E+03	1.89E+03	**1.70E+03**
	SD	*2.62E+00*	5.63E+01	1.65E+01	9.56E+00	2.79E+01	3.33E+01	6.56E+00	9.44E+00	3.98E+01	1.76E+02	5.96E+00	9.05E+01	**1.34E+00**
F18	Mean	6.10E+03	1.80E+04	2.70E+04	5.13E+04	1.80E+04	2.08E+04	**1.80E+03**	*1.80E+03*	1.60E+04	1.83E+03	1.81E+03	1.33E+04	1.80E+03
	SD	3.87E+03	1.18E+04	1.47E+04	3.13E+04	1.13E+04	1.19E+04	**3.06E−01**	*5.20E−01*	1.30E+04	1.35E+01	9.47E+00	7.43E+03	3.00E+00
F19	Mean	2.27E+03	5.53E+03	2.65E+04	2.90E+03	1.91E+03	1.84E+04	1.90E+03	1.90E+03	9.69E+03	1.92E+03	**1.90E+03**	2.34E+04	*1.90E+03*
	SD	4.30E+02	6.09E+03	2.82E+04	2.82E+03	1.81E+01	2.22E+04	3.54E−01	4.50E−01	6.77E+03	2.87E+01	**2.80E−01**	2.10E+04	*3.15E−01*
F20	Mean	2.00E+03	2.01E+03	2.13E+03	2.07E+03	2.02E+03	2.14E+03	*2.01E+03*	2.02E+03	2.06E+03	2.49E+03	2.02E+03	2.10E+03	**2.00E+03**
	SD	1.00E+00	9.65E+00	4.68E+01	1.76E+01	8.53E+00	7.12E+01	*8.97E+00*	9.67E+00	6.00E+01	2.43E+02	1.00E−02	4.68E+01	**9.87E−02**
F21	Mean	2.21E+03	2.32E+03	2.23E+03	2.22E+03	2.29E+03	2.31E+03	*2.20E+03*	2.20E+03	2.30E+03	2.32E+03	2.28E+03	2.34E+04	**2.19E+03**
	SD	5.21E+00	3.26E+01	1.28E+01	4.16E+01	5.29E+01	6.98E+01	*0.00E+00*	2.04E+01	4.38E+01	6.78E+01	4.26E+01	2.10E+04	**3.18E+01**
F22	Mean	**2.27E+03**	2.39E+03	2.66E+03	2.34E+03	2.30E+03	2.34E+03	*2.27E+03*	2.28E+03	2.30E+03	3.53E+03	2.30E+03	2.66E+03	2.29E+03
	SD	**3.12E+01**	2.37E+02	1.07E+02	2.55E+01	2.38E+01	1.76E+02	*4.35E+01*	3.81E+01	2.39E+00	8.47E+02	1.62E+01	1.99E+02	2.49E+01
F23	Mean	**2.60E+03**	2.62E+03	2.67E+03	2.65E+03	2.62E+03	2.65E+03	2.61E+03	2.61E+03	2.63E+03	2.73E+03	*2.61E+03*	2.66E+03	2.61E+03
	SD	**5.70E+01**	8.16E+00	1.22E+01	6.14E+00	6.81E+00	1.98E+01	3.02E+00	3.93E+00	1.34E+01	2.43E+02	*1.71E+00*	1.99E+02	2.28E+00
F24	Mean	**2.51E+03**	2.76E+03	2.82E+03	2.75E+03	2.74E+03	2.76E+03	2.58E+03	*2.52E+03*	2.75E+03	2.70E+03	2.73E+03	2.85E+03	2.54E+03
	SD	**2.72E+01**	5.02E+01	4.80E+01	7.30E+01	4.69E+01	4.32E+01	1.13E+02	*3.84E+01*	1.49E+01	7.34E+01	3.17E+01	6.54E+01	8.02E+01
F25	Mean	**2.83E+03**	2.93E+03	3.08E+03	2.94E+03	2.93E+03	2.93E+03	*2.89E+03*	2.90E+03	2.95E+03	2.93E+03	2.92E+03	3.02E+03	2.92E+03
	SD	**1.12E+02**	2.56E+01	8.26E+01	1.64E+01	2.61E+01	6.52E+01	*5.76E+01*	4.90E−01	1.93E+01	2.09E+01	2.29E+01	8.14E+01	2.33E+01
F26	Mean	**2.76E+03**	3.31E+03	3.36E+03	3.04E+03	3.03E+03	3.31E+03	*2.78E+03*	2.85E+03	3.11E+03	3.46E+03	2.91E+03	3.73E+03	2.86E+03
	SD	**1.11E+02**	4.73E+02	2.34E+02	2.16E+01	2.99E+02	5.66E+02	*1.21E+02*	9.63E+01	3.35E+02	5.99E+02	3.49E+01	4.30E+02	9.66E+01
F27	Mean	3.10E+03	3.10E+03	3.13E+03	3.10E+03	3.09E+03	3.12E+03	3.09E+03	3.09E+03	3.12E+03	3.14E+03	**3.07E+03**	3.23E+03	*3.09E+03*
	SD	2.68E+00	1.42E+01	4.04E+01	1.55E+00	9.26E−01	2.97E+01	5.64E−01	4.60E−01	1.92E+01	2.10E+02	**7.80E−01**	5.23E+01	*2.29E−01*
F28	Mean	3.11E+03	3.22E+03	3.33E+03	3.23E+03	3.16E+03	3.41E+03	*3.09E+03*	3.10E+03	3.32E+03	3.40E+03	3.27E+03	3.53E+03	**3.09E+03**
	SD	8.53E+01	1.85E+02	1.70E+02	1.95E+01	1.18E+02	1.53E+02	*5.81E+01*	6.34E−05	1.26E+02	1.31E+02	2.23E+01	2.14E+02	**2.29E+00**
F29	Mean	3.16E+03	3.21E+03	3.32E+03	3.19E+03	3.17E+03	3.32E+03	*3.14E+03*	3.15E+03	3.25E+03	3.21E+03	**3.14E+03**	3.40E+03	3.15E+03
	SD	2.64E+01	4.99E+01	7.35E+01	2.16E+01	4.86E+01	1.05E+02	*1.87E+01*	1.28E+01	8.20E+01	1.10E+02	**1.29E+01**	1.87E+02	1.52E+01
F30	Mean	5.11E+04	2.98E+05	3.38E+06	5.07E+05	3.44E+03	3.64E+05	*3.40E+03*	3.41E+03	5.30E+05	3.00E+05	**3.20E+03**	6.19E+06	4.07E+03
	SD	6.55E+04	4.12E+05	5.60E+06	4.84E+05	7.73E+01	4.45E+05	*9.21E+00*	2.68E+01	6.30E+05	4.40E+05	**3.10E−01**	6.77E+06	7.18E+02

4.7. Parametric analyses

In this section, a parametric study is conducted to evaluate the effect of the Reinforcement Rate (RR). Different scenarios such as equal RR, higher positive reinforcement, and higher negative reinforcement for positive and negative reinforcements are used. Fig. 8 shows the effect of the hyper-parameter RR on the performance and convergence rate of the PRO algorithm for four different benchmark functions. Fig. 8(a), (b), (c) and (d) represent the effect of different values, RR = 0.1, 0.5, 0.7, and 0.9, on a unimodal function (F06), multimodal

Table 9
Comparison results of the PRO and the state-of-the-art algorithms on hybrid test functions.

Function		TLBO	Bat	DE	HFPSO	ACPSO	PSO	IHS	IGHS	SCA	WOA	DOLTLBO	PRO
F29	Mean	5.89E+00	8.86E+02	5.81E+00	2.72E+00	*1.27E+00*	2.75E+00	1.85E+00	**1.11E+00**	1.43E+06	2.18E+01	3.17E+00	1.71E+00
	SD	2.38E+00	1.77E+02	4.67E−01	6.94E−01	*2.46E−01*	6.40E−01	4.25E−01	**2.16E−01**	6.59E+05	7.23E+00	1.43E+00	4.37E−01
F30	Mean	1.29E+01	1.37E+01	1.33E+01	*1.22E+01*	1.30E+01	**1.21E+01**	1.30E+01	1.33E+01	1.40E+01	1.34E+01	1.32E+01	1.24E+01
	SD	2.45E−01	2.94E−01	1.99E−01	*5.61E−01*	5.20E−01	**5.02E−01**	3.78E−01	2.23E−01	1.42E−01	3.22E−01	2.95E−01	4.12E−01
F31	Mean	4.59E+02	8.89E+02	*2.59E+02*	4.51E+02	3.46E+02	4.33E+02	2.91E+02	2.91E+02	9.49E+02	7.44E+02	2.68E+02	**0.00E+00**
	SD	5.74E+01	3.46E+02	*4.47E+01*	1.29E+02	1.70E+02	1.08E+02	1.16E+02	6.32E+01	8.88E+01	1.65E+02	7.29E+01	**0.00E+00**
F32	Mean	1.19+05	4.87E+06	1.48E+06	6.13E+04	1.32E+06	3.45E+05	3.92E+05	2.22E+05	4.54E+07	4.66E+06	*8.22E+03*	**5.11E+03**
	SD	1.09+05	3.73E+06	6.01E+05	3.84E+04	1.80E+06	5.01E+05	2.42E+05	9.09E+04	2.67E+07	3.27E+06	*5.14E+03*	**1.80E+03**
F33	Mean	2.58+03	3.10E+03	1.13E+04	3.33E+03	4.18E+04	2.96E+03	2.72E+03	1.37E+03	1.95E+09	7.02E+03	*6.76E+02*	**1.74E+02**
	SD	2.80+03	3.51E+03	7.18E+03	5.19E+03	2.14E+05	5.25E+03	4.34E+03	1.50E+03	6.24E+08	1.14E+04	*8.65E+02*	**2.74E+02**
F34	Mean	1.84+01	2.65E+02	*8.67E+00*	1.01E+01	1.34E+01	9.44E+00	1.01E+01	9.62E+00	3.85E+02	4.64E+01	1.40E+01	**4.51E+00**
	SD	2.19+01	7.84E+01	*4.98E−01*	3.79E+00	1.69E+01	1.66E+00	1.07E+01	1.48E+01	9.02E+01	4.12E+01	1.52E+01	**8.38E−01**
F35	Mean	4.36+02	5.13E+04	3.89E+03	3.44E+02	1.88E+04	2.38E+02	6.56E+03	2.41E+03	3.91E+05	2.33E+04	*2.30E+02*	**3.46E+01**
	SD	1.77+02	2.46E+04	2.20E+03	1.68E+02	1.49E+04	1.14E+02	4.59E+03	1.51E+03	2.42E+05	1.54E+04	*8.02E+01*	**1.26E+01**
F36	Mean	3.58+04	1.48E+06	2.97E+05	3.22E+04	5.35E+05	6.24E+04	2.09E+05	1.31E+05	1.35E+07	1.60E+06	*4.32E+03*	**1.09E+03**
	SD	2.47+04	2.09E+06	1.14E+05	2.33E+04	3.41E+05	4.32E+04	1.55E+05	7.00E+04	6.33E+06	1.82E+06	*3.51E+03*	**2.45E+02**
F37	Mean	2.61+02	2.13E+03	**1.52E+02**	3.77E+02	5.97E+02	3.35E+02	4.47E+02	2.26E+02	1.68E+03	8.68E+02	3.67E+02	*1.86E+02*
	SD	1.22+02	7.66E+02	**5.91E+01**	1.27E+02	2.40E+02	7.19E+01	1.62E+02	1.48E+02	1.92E+02	2.07E+02	1.32E+02	*6.34E+01*

Table 10
Comparison results of the PRO and the state-of-the-art algorithms on composition test functions.

Function		TLBO	Bat	DE	HFPSO	ACPSO	PSO	IHS	IGHS	SCA	WOA	DOLTLBO	PRO
F38	Mean	3.15E+02	4.48E+02	*3.15E+02*	*3.15E+02*	3.17E+02	3.18E+02	3.15E+02	3.15E+02	8.95E+02	3.31E+02	**2.00E+02**	3.15E+02
	SD	4.41E−12	4.85E+01	*5.78E−14*	*5.97E−13*	2.86E+00	4.30E+00	6.47E−02	4.91E−10	1.19E+02	2.61E+01	**0.00E+00**	2.51E−02
F39	Mean	*2.00E+02*	3.19E+02	2.26E+02	2.22E+02	2.26E+02	2.16E+02	2.31E+02	2.24E+02	4.80E+02	2.05E+02	**2.00e+02**	2.08E+02
	SD	*1.03E−03*	2.71E+01	1.78E+00	7.92E+00	5.17E+00	1.18E+01	6.13E+00	2.37E+00	3.26E+01	3.46E+00	**0.00E+00**	1.87E+01
F40	Mean	*2.00E+02*	2.32E+02	2.09E+02	2.07E+02	2.07E+02	2.05E+02	2.08E+02	2.06E+02	3.06E+02	2.23E+02	**2.00e+02**	2.02E+02
	SD	*1.45E−13*	8.45E+00	1.07E+00	3.23E+00	3.50E+00	2.06E+00	2.14E+00	1.44E+00	2.25E+01	1.81E+01	**0.00E+00**	4.63E+00
F41	Mean	1.17E+02	1.07E+02	*1.00E+02*	1.50E+02	1.39E+02	1.57E+0	1.25E+02	1.10E+02	1.07E+02	1.10E+02	1.30E+02	**1.00E+02**
	SD	3.77E+01	1.85E+00	*3.55E−02*	5.06E+01	6.08E+01	5.02E+01	5.37E+01	3.04E+01	7.27E−01	3.04E+01	4.64E+01	**1.82E−02**
F42	Mean	5.18E+02	1.50E+03	5.41E+02	5.65E+02	6.51E+02	5.54E+02	6.76E+02	3.42E+02	1.39E+03	9.69E+02	**2.00e+02**	*3.17E+02*
	SD	1.60E+02	4.61E+02	6.30E+01	1.78E+02	2.08E+02	1.54E+02	7.89E+01	3.78E+01	1.54E+02	4.20E+02	**0.00E+00**	*5.34E+01*
F43	Mean	1.19E+03	4.94E+03	8.45E+02	1.64E+03	1.35E+03	1.66E+03	9.86E+02	8.52E+02	2.74E+03	2.31E+03	**2.00e+02**	*6.85E+02*
	SD	2.48E+02	7.50E+02	2.39E+01	5.58E+02	4.16E+02	3.56E+02	9.34E+01	4.77E+01	4.02E+02	4.89E+02	**0.00E+00**	*7.41E+01*
F44	Mean	4.33E+05	7.60E+07	1.76E+03	7.81E+06	9.40E+05	5.40E+06	1.45E+03	*1.09E+03*	3.41E+07	5.76E+06	2.24E+04	**9.18E+02**
	SD	2.36E+06	6.98E+07	4.05E+02	1.24E+07	2.81E+06	2.07E+07	4.95E+02	*2.48E+02*	9.50E+06	4.81E+06	1.14E+05	**2.69E+02**
F45	Mean	2.62E+03	3.47E+05	3.55E+03	6.96E+03	5.82E+03	4.76E+03	3.51E+03	*1.92E+03*	7.29E+05	9.38E+04	2.52E+03	**1.46E+03**
	SD	1.12E+03	3.97E+05	1.05E+03	1.97E+04	2.98E+03	5.70E+03	1.19E+03	*6.36E+02*	2.45E+05	7.57E+04	6.45E+02	**5.86E+02**

Table 11
The Wilcoxon singed-rank test between the PRO and the state-of-the-art MAs for the IEEE CEC test suite (D = 30).

Pairwise comparison of the PRO algorithm and MAs	R^+	R^-	Decision	
			$\alpha = 0.05$	$\alpha = 0.01$
PRO-TLBO	854	49	Yes	Yes
PRO-DoLTLBO	753.5	107.5	Yes	Yes
PRO-WOA	1019	16	Yes	Yes
PRO-SCA	1035	0	Yes	Yes
PRO-IGHS	757	146	Yes	Yes
PRO-IHS	860	130	Yes	Yes
PRO-PSO	942	93	Yes	Yes
PRO-ACPSO	925	65	Yes	Yes
PRO-HFPSO	818	85	Yes	Yes
PRO-DE	830	73	Yes	Yes
PRO-Bat	999	36	Yes	Yes

function (F18), hybrid function (F32), composition function (F44), respectively. According to Fig. 8, it is clear that the PRO algorithm provides better performance with higher RR values for all types of benchmark functions. In addition, the effect of different scenarios for positive and negative reinforcements such as equal RR, higher positive reinforcement, and higher negative reinforcement is shown in Fig. 9. The results in Fig. 9(a), (b), (c), and (d) illustrate the effect of the different scenarios as a unimodal function (F06), multimodal function (F18), hybrid function (F32), and composition function (F44), respectively. It seems that the PRO algorithm performs better with unequal Positive Reinforcement Rate (PRR) and Negative Reinforcement Rate (NRR) for all cases, especially, when PRR is smaller than NRR (PRR<NRR).

4.8. Computational complexity analysis

We have analyzed the computational complexity of the PRO by utilizing the complexity analysis introduced in Suganthan et al. (2005). In our study, we compared the complexity of PRO with that of RSA (Abualigah et al., 2022) and AOA (Abualigah et al., 2021). To calculate the computational complexity, we followed the method introduced in Suganthan et al. (2005) and used the formula $(T_2 - T_1)/T_0$. This involved computing T_1, which is the time taken to run a specific benchmark function, and T_2, which is the average runtime of the optimization algorithm on the same benchmark function with specific NFEs and dimensions (D). We also calculated T_0, which is the time taken to execute the following code:

$$
\begin{aligned}
&For\ i\ =\ 1\ to\ 1000000\\
&\quad x\ =\ (double)\ 5.55\\
&\quad x = x + x;\ x = x/2;\ X = x * x;\ X = sqrt(x);\ x = ln(x);\\
&\quad x = exp(x);\ y = x/x;\\
&end
\end{aligned}
$$

Table 12
Descriptive statistics of Friedman test.

Algorithms	n	Minimum	25th percentile	Median	75th percentile	Maximum
ACPSO	45	0.1150	13.300	207.000	12 100.000	6 980 000.000
Bat	45	2.7800	205.250	2130.000	45 825.000	31 200 000 000.000
DE	45	0.0000	12.425	121.000	2272.500	53 000 000.000
DOLTLBO	45	0.0000	11.350	107.000	1137.000	651 000.000
HFPSO	45	0.0000	9.817	150.000	1280.000	7 810 000.000
IGHS	45	0.0000	1.382	131.000	1507.500	5 630 000.000
IHS	45	0.0000	8.943	208.000	2917.500	7 430 000.000
PRO	45	0.0000	0.246	20.700	315.500	570 000.000
PSO	45	0.4090	18.025	221.000	2697.500	439 000 000.000
SCA	45	3.5200	365.250	2740.000	841 750.000	86 400 000 000.000
TLBO	45	0.0000	16.050	85.000	1297.500	700 000.000
WOA	45	0.2610	31.625	331.000	24 750.000	32 900 000.000

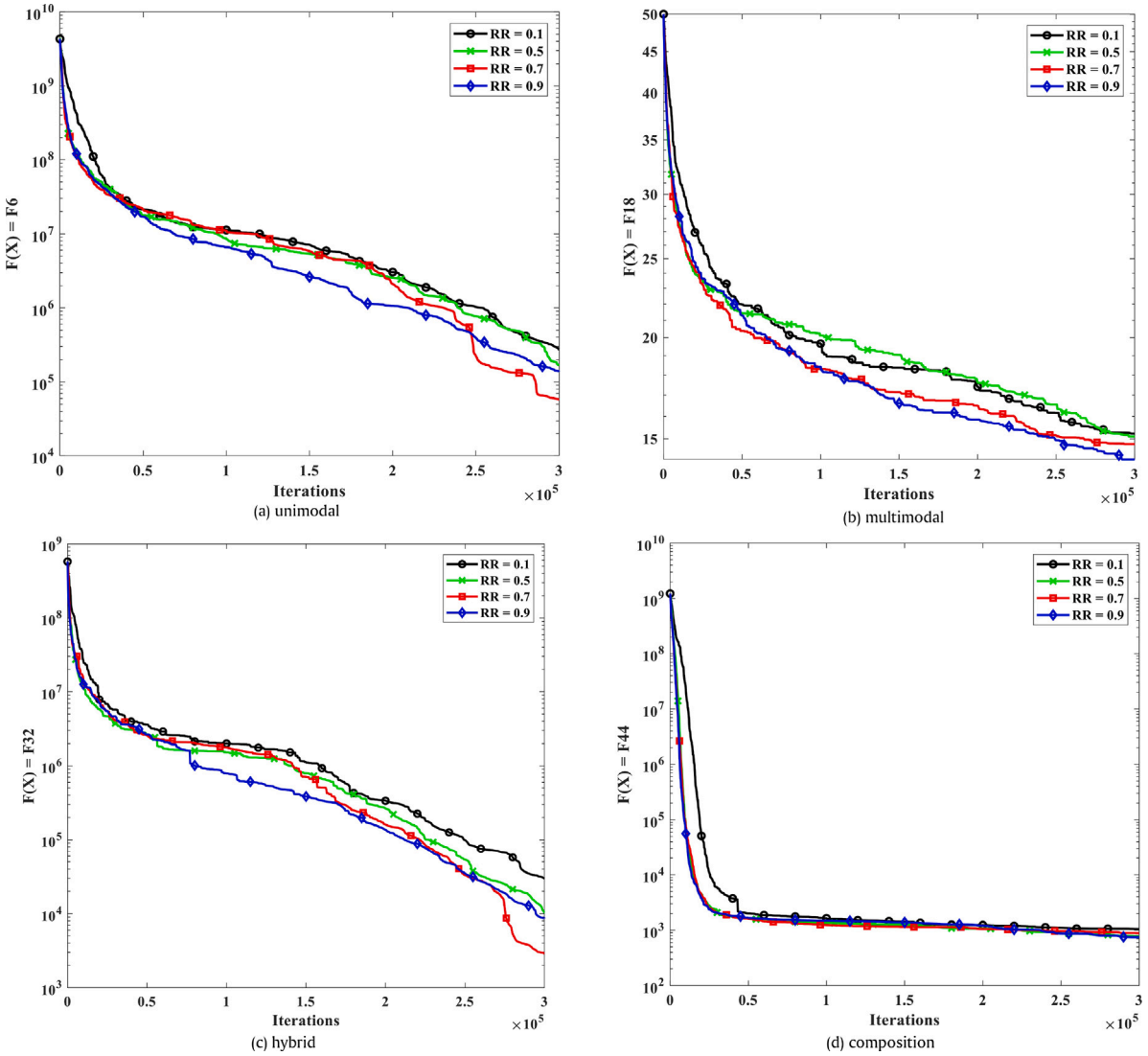


Fig. 8. The effect of the hyper-parameter RR on the performance and convergence rate of the PRO algorithm.

or our study, we conducted experiments on the F30 benchmark function from IEEE CEC2014 (Liang et al., 2014), with varying dimensions including 10, 30, 50, and 100 a maximum number of function evaluations MaxFEs = 200 000. We found that the values of T_1 for D = 10, D = 30, D = 50, and D = 100 were 2.4, 4.6, 8.5 and 24.1 s, respectively. We also determined that T_0 was 0.165 s. Based on the results presented in Table 14, it appears that the AOA algorithm has a superior performance in terms of computational complexity when confronted with problems dimensions D = 10 and D = 30. However, our proposed PRO has been found to outperform competitors in situations where the dimensions of the problem are D = 50 and D = 100. Moreover, Table 14 reveals that the complexity of PRO remains largely unaffected by an increase in problem size, whereas the complexity of both AOA and RSA algorithms is seen to increase in such situations

Table 13
Comparison results of the PRO and the state-of-the-art algorithms on CEC-BC-2017 test functions.

Function		CHIO	HGS	RSO	SCA	SMA	WOA	YDSE	MPA	GA	CMAES	SHADE	AOA	PRO
F1	Mean	2.04E+03	7.07E+03	2.66E+09	4.60E+08	8.07E+03	6.19E+04	**1.00E+02**	**1.00E+02**	9.80E+03	**1.00E+02**	**1.00E+02**	3.05E+09	**1.00E+02**
	SD	1.68E+03	4.08E+03	1.66E+09	2.01E+08	4.26E+03	1.53E+05	**0.00E+00**	**4.79E−06**	5.94E+03	**0.00E+00**	**0.00E+00**	1.83E+09	**0.00E+00**
F3	Mean	4.22E+03	3.00E+02	4.53E+03	9.26E+02	3.00E+02	4.19E+02	**3.00E+02**	**3.00E+02**	8.72E+03	**3.00E+02**	**3.00E+02**	4.02E+03	**3.00E+02**
	SD	2.38E+03	0.00E+00	2.55E+03	4.28E+02	8.36E−05	1.09E+02	**0.00E+00**	**0.00E+00**	5.90E+03	**0.00E+00**	**0.00E+00**	1.80E+03	**0.00E+00**
F4	Mean	4.01E+02	4.05E+02	6.11E+02	4.29E+02	4.03E+02	4.16E+02	**4.00E+02**	*4.00E+02*	4.11E+02	**4.00E+02**	**4.00E+02**	5.72E+02	**4.00E+02**
	SD	1.13E+00	1.61E+01	1.41E+02	1.06E+01	8.91E−01	3.08E+01	**0.00E+00**	*5.70E−02*	1.85E+01	**0.00E+00**	**0.00E+00**	1.12E+02	**0.00E+00**
F5	Mean	5.09E+02	5.17E+02	5.67E+02	5.39E+02	5.12E+02	5.50E+02	5.06E+02	5.10E+02	5.16E+02	5.30E+02	**5.04E+02**	5.57E+02	*5.05E+02*
	SD	2.63E+00	7.34E+00	1.11E+01	7.21E+00	6.09E+00	1.85E+01	2.43E+00	3.96E+00	6.93E+00	5.83E+01	**1.01E+00**	2.66E+01	*2.22E+00*
F6	Mean	6.00E+02	6.00E+02	6.41E+02	6.15E+02	6.00E+02	6.29E+02	6.00E+02	6.00E+02	6.00E+02	6.82E+02	*6.00E+02*	6.38E+02	**6.00E+02**
	SD	4.01E−03	7.20E−01	6.42E+00	3.57E+00	1.51E−02	1.26E+01	2.62E−03	6.16E−04	6.69E−02	3.54E+01	*2.65E−07*	8.83E+00	**0.00E+00**
F7	Mean	7.19E+02	7.26E+02	7.96E+02	7.65E+02	7.21E+02	7.75E+02	7.16E+02	7.23E+02	7.28E+02	**7.13E+02**	*7.14E+02*	6.38E+02	7.15E+02
	SD	4.03E+00	1.06E+01	1.30E+01	5.71E+00	3.98E+00	1.89E+01	2.11E+00	3.91E+00	7.29E+00	**1.63E+00**	*1.24E+00*	8.83E+00	2.63E+00
F8	Mean	8.10E+02	8.17E+02	8.38E+02	8.32E+02	8.12E+02	8.36E+02	8.06E+02	8.09E+02	8.21E+02	8.29E+02	**8.04E+02**	8.33E+02	*8.05E+02*
	SD	2.54E+00	6.41E+00	9.78E+00	5.53E+00	5.67E+00	1.00E+01	2.39E+00	3.12E+00	8.96E+00	5.30E+01	**1.28E+00**	5.53E+00	*2.39E+00*
F9	Mean	9.00E+02	**9.00E+02**	1.24E+03	9.61E+02	*9.00E+02*	1.22E+03	**9.00E+02**	9.00E+02	9.10E+02	4.67E+03	**9.00E+02**	1.33E+03	**9.00E+02**
	SD	2.06E−01	**0.00E+00**	1.37E+02	2.64E+01	*8.29E−05*	2.85E+02	**0.00E+00**	1.63E−02	1.52E+01	2.06E+03	**0.00E+00**	1.38E+02	**0.00E+00**
F10	Mean	1.27E+03	1.43E+03	2.31E+03	2.06E+03	1.58E+03	2.00E+03	1.34E+03	1.44E+03	1.72E+03	2.59E+03	**1.19E+03**	2.11E+03	*1.21E+03*
	SD	9.74E+01	1.97E+02	1.89E+02	1.81E+02	2.28E+02	3.48E+02	1.27E+02	1.41E+02	2.52E+02	4.14E+02	**8.47E+01**	2.93E+02	*1.39E+02*
F11	Mean	1.11E+03	1.11E+03	1.42E+03	1.17E+03	1.11E+03	1.17E+03	**1.10E+03**	1.10E+03	1.13E+03	1.11E+03	1.10E+03	2.11E+03	*1.10E+03*
	SD	2.77E+00	1.06E+01	3.83E+02	1.66E+01	3.21E+00	5.87E+01	**2.52E−01**	1.27E+00	2.38E+01	2.54E+01	1.36E+00	2.93E+02	*1.09E+00*
F12	Mean	1.22E+05	1.72E+04	2.22E+07	5.42E+06	4.36E+04	3.57E+06	**1.20E+03**	*1.25E+03*	3.70E+06	1.63E+03	1.32E+03	7.91E+05	1.47E+03
	SD	1.55E+05	1.42E+04	6.30E+07	4.20E+06	1.75E+04	3.91E+06	**2.16E+01**	*5.43E+01*	3.40E+06	1.98E+02	1.03E+02	3.70E+05	1.33E+02
F13	Mean	3.75E+03	9.29E+03	2.88E+04	1.50E+04	1.14E+04	1.71E+04	1.30E+03	1.31E+03	1.08E+04	1.32E+03	**1.30E+03**	8.58E+03	*1.30E+03*
	SD	4.68E+03	1.06E+04	1.79E+04	9.99E+03	1.15E+04	1.22E+04	2.03E+00	2.58E+00	8.93E+03	7.83E+01	**7.10E−01**	6.96E+03	*7.74E−01*
F14	Mean	1.65E+03	1.67E+03	2.62E+03	1.52E+03	1.43E+03	1.51E+03	*1.40E+03*	1.40E+03	7.05E+03	1.45E+03	1.41E+03	7.15E+03	**1.40E+03**
	SD	2.23E+02	7.89E+02	1.54E+03	3.40E+01	1.10E+01	3.19E+01	*1.13E+00*	4.06E+00	8.16E+03	3.60E+01	9.21E+00	8.24E+03	**9.14E−01**
F15	Mean	1.94E+03	1.95E+03	7.98E+03	1.77E+03	1.51E+03	2.79E+03	*1.50E+03*	1.50E+03	9.30E+03	1.51E+03	1.50E+03	1.19E+04	**1.50E+03**
	SD	4.40E+02	4.94E+02	5.90E+03	1.39E+02	5.54E+00	1.26E+03	*1.75E−01*	5.20E−01	8.98E+03	1.64E+01	3.60E−01	5.42E+03	**1.71E−01**
F16	Mean	1.62E+03	1.74E+03	1.83E+03	1.67E+03	1.68E+03	1.80E+03	**1.60E+03**	*1.60E+03*	1.79E+03	1.82E+03	1.60E+03	2.11E+03	1.61E+03
	SD	4.03E+01	9.90E+01	1.34E+02	2.72E+01	8.20E+01	1.25E+02	**5.82E−01**	*9.90E−01*	1.29E+02	2.30E+02	2.20E+00	9.30E+01	3.77E+01
F17	Mean	*1.70E+03*	1.76E+03	1.78E+03	1.76E+03	1.74E+03	1.79E+03	1.72E+03	1.71E+03	1.75E+03	1.83E+03	1.72E+03	1.89E+03	**1.70E+03**
	SD	*2.62E+00*	5.63E+01	1.65E+01	9.56E+00	2.79E+01	3.33E+01	6.56E+00	9.44E+00	3.98E+01	1.76E+02	5.96E+00	9.05E+01	**1.34E+00**
F18	Mean	6.10E+03	1.80E+04	2.70E+04	5.13E+04	1.80E+04	2.08E+04	**1.80E+03**	*1.80E+03*	1.60E+04	1.83E+03	1.81E+03	1.33E+04	1.80E+03
	SD	3.87E+03	1.18E+04	1.47E+04	3.13E+04	1.13E+04	1.19E+04	**3.06E−01**	*5.20E−01*	1.30E+04	1.35E+01	9.47E+00	7.43E+03	3.00E+00
F19	Mean	2.27E+03	5.53E+03	2.65E+04	2.90E+03	1.91E+03	1.84E+04	1.90E+03	1.90E+03	9.69E+03	1.92E+03	**1.90E+03**	2.34E+04	*1.90E+03*
	SD	4.30E+02	6.09E+03	2.82E+04	2.82E+03	1.81E+01	2.22E+04	3.54E−01	4.50E−01	6.77E+03	2.87E+01	**2.80E−01**	2.10E+04	*3.15E−01*
F20	Mean	2.00E+03	2.01E+03	2.13E+03	2.07E+03	2.02E+03	2.14E+03	*2.01E+03*	2.02E+03	2.06E+03	2.49E+03	2.02E+03	2.10E+03	**2.00E+03**
	SD	1.00E+00	9.65E+00	4.68E+01	1.76E+01	8.53E+00	7.12E+01	*8.97E+00*	9.67E+00	6.00E+01	2.43E+02	1.00E−02	4.68E+01	**9.87E−02**
F21	Mean	2.21E+03	2.32E+03	2.23E+03	2.22E+03	2.29E+03	2.31E+03	*2.20E+03*	2.20E+03	2.30E+03	2.32E+03	2.28E+03	2.34E+04	**2.19E+03**
	SD	5.21E+00	3.26E+01	1.28E+01	4.16E+01	5.29E+01	6.98E+01	*0.00E+00*	2.04E+01	4.38E+01	6.78E+01	4.26E+01	2.10E+04	**3.18E+01**
F22	Mean	**2.27E+03**	2.39E+03	2.66E+03	2.34E+03	2.30E+03	2.34E+03	*2.27E+03*	2.28E+03	2.30E+03	3.53E+03	2.30E+03	2.66E+03	2.29E+03
	SD	**3.12E+01**	2.37E+02	1.07E+02	2.55E+01	2.38E+01	1.76E+02	*4.35E+01*	3.81E+01	2.39E+00	8.47E+02	1.62E+01	1.99E+02	2.49E+01
F23	Mean	**2.60E+03**	2.62E+03	2.67E+03	2.65E+03	2.62E+03	2.65E+03	2.61E+03	2.61E+03	2.63E+03	2.73E+03	*2.61E+03*	2.66E+03	2.61E+03
	SD	**5.70E+01**	8.16E+00	1.22E+01	6.14E+00	6.81E+00	1.98E+01	3.02E+00	3.93E+00	1.34E+01	2.43E+02	*1.71E+00*	1.99E+02	2.28E+00
F24	Mean	**2.51E+03**	2.76E+03	2.82E+03	2.75E+03	2.74E+03	2.76E+03	2.58E+03	*2.52E+03*	2.75E+03	2.70E+03	2.73E+03	2.85E+03	2.54E+03
	SD	**2.72E+01**	5.02E+01	4.80E+01	7.30E+01	4.69E+01	4.32E+01	1.13E+02	*3.84E+01*	1.49E+01	7.34E+01	3.17E+01	6.54E+01	8.02E+01
F25	Mean	**2.83E+03**	2.93E+03	3.08E+03	2.94E+03	2.93E+03	2.93E+03	*2.89E+03*	2.90E+03	2.95E+03	2.93E+03	2.92E+03	3.02E+03	2.92E+03
	SD	**1.12E+02**	2.56E+01	8.26E+01	1.64E+01	2.61E+01	6.52E+01	*5.76E+01*	4.90E−01	1.93E+01	2.09E+01	2.29E+01	8.14E+01	2.33E+01
F26	Mean	**2.76E+03**	3.31E+03	3.36E+03	3.04E+03	3.03E+03	3.31E+03	*2.78E+03*	2.85E+03	3.11E+03	3.46E+03	2.91E+03	3.73E+03	2.86E+03
	SD	**1.11E+02**	4.73E+02	2.34E+02	2.16E+01	2.99E+02	5.66E+02	*1.21E+02*	9.63E+01	3.35E+02	5.99E+02	3.49E+01	4.30E+02	9.66E+01
F27	Mean	3.10E+03	3.10E+03	3.13E+03	3.10E+03	3.09E+03	3.12E+03	3.09E+03	3.09E+03	3.12E+03	3.14E+03	**3.07E+03**	3.23E+03	*3.09E+03*
	SD	2.68E+00	1.42E+01	4.04E+01	1.55E+00	9.26E−01	2.97E+01	5.64E−01	4.60E−01	1.92E+01	2.10E+02	**7.80E−01**	5.23E+01	*2.29E−01*
F28	Mean	3.11E+03	3.22E+03	3.33E+03	3.23E+03	3.16E+03	3.41E+03	*3.09E+03*	3.10E+03	3.32E+03	3.40E+03	3.27E+03	3.53E+03	**3.09E+03**
	SD	8.53E+01	1.85E+02	1.70E+02	1.95E+01	1.18E+02	1.53E+02	*5.81E+01*	6.34E−05	1.26E+02	1.31E+02	2.23E+01	2.14E+02	**2.29E+00**
F29	Mean	3.16E+03	3.21E+03	3.32E+03	3.19E+03	3.17E+03	3.32E+03	*3.14E+03*	3.15E+03	3.25E+03	3.21E+03	**3.14E+03**	3.40E+03	3.15E+03
	SD	2.64E+01	4.99E+01	7.35E+01	2.16E+01	4.86E+01	1.05E+02	*1.87E+01*	1.28E+01	8.20E+01	1.10E+02	**1.29E+01**	1.87E+02	1.52E+01
F30	Mean	5.11E+04	2.98E+05	3.38E+06	5.07E+05	3.44E+03	3.64E+05	*3.40E+03*	3.41E+03	5.30E+05	3.00E+05	**3.20E+03**	6.19E+06	4.07E+03
	SD	6.55E+04	4.12E+05	5.60E+06	4.84E+05	7.73E+01	4.45E+05	*9.21E+00*	2.68E+01	6.30E+05	4.40E+05	**3.10E−01**	6.77E+06	7.18E+02

5. Case study: Evolutionary federated deep learning for ECG diagnostic

Federated Learning (FL) is a technique used to train a global model with distributed datasets (McMahan et al., 2016). This method is particularly useful in situations where it is difficult or impossible to collect distributed datasets for creating a central dataset (Nasirigerdeh et al., 2020). For instance, in medical applications of AI, ensuring the privacy of patients' data is crucial. In many cases, individuals and medical centers hesitate to share sensitive and private data with

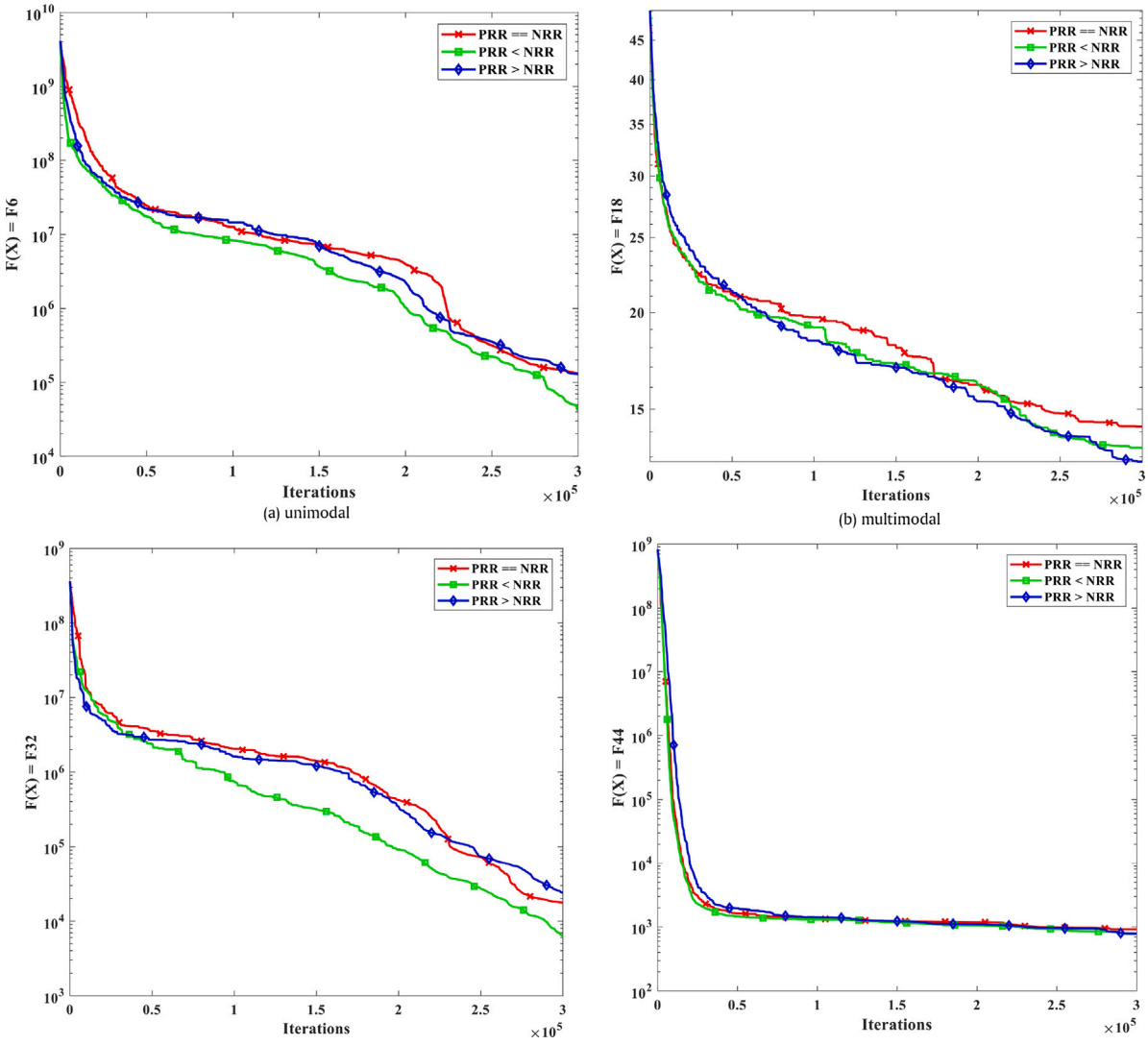


Fig. 9. The effect of different scenarios for positive and negative reinforcements values on the PRO algorithm.

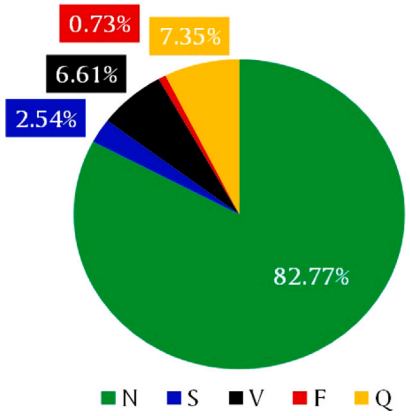


Fig. 10. Rankings of the PRO and the state-of-the-art algorithms based on the Friedman test on IEEE CEC test suites (D = 30).

Table 14
Computational complexity analysis for APA, RSA, and PRO.

Dimension (D)	Algorithm		
	AOA	RSA	PRO
10	105.63	320.5	198.2
30	185.4	467.2	201.4
50	269.1	593.3	210.1
100	414.2	941.7	227.8

research centers to generate AI-based models (Sheller et al., 2018), even though such data is essential in developing a robust model with acceptable performance (Goodfellow et al., 2017). On the other hand, due to the heterogeneity of data distribution and non-independent and identically distributed (non-iid) datasets, it is still challenging to achieve a highly accurate FL-based trained global model. In this study, the proposed PRO algorithm is applied to optimize federated deep learning for Electrocardiography (ECG) classification tasks with non-iid datasets. To do this, the MIT-BIH arrhythmia database, which is a

Fig. 11. Sample electrocardiography (ECG).

Table 15
The MIT-BIH arrhythmia dataset details.

Class	Description	Number of samples
N	Normal beat	72 471
S	Supraventricular premature	2223
V	Premature ventricular contraction	5788
F	Fusion of ventricular and normal beat	641
Q	Unclassifiable beat	6431

large public dataset for electrocardiography (Moody & Mark, 2001), is used. A distributed dataset in non-iid form is created, where each local dataset involves different number of samples. Fig. 10 and Table 15 illustrate the distribution of the non-iid datasets over 50 participants. In addition, the ResNet architecture is used He et al. (2015) as the global model in all the experiments.

5.1. The MIT-BIH arrhythmia database

MIT-BIH arrhythmia database is the first publicly standard and one of the authoritative sets for the evaluation and detection of arrhythmia. It was collected by the BIH Arrhythmia Laboratory between 1975 and 1979 and included 25 men and 22 women as subjects. The database consists of 48 half-hour excerpts from two-channel 24-hour ECG recordings. Fig. 11 represents a sample of ECG, where 22 records (the "100 series") were randomly chosen from a collection of over 4000 Holter tapes and the other 25 records (the "200 series") were randomly chosen from the same set to have a variety of rare and clinically important phenomena that would not be well-represented by a small random sample (Moody & Mark, 2001). We consider the aggregation process and a server-side procedure is launched after each round to generate the global model by applying weighted averaging as the optimization problem. The PRO algorithm attempts to find the best (tuning) coefficients of the global model.

5.2. Problem formulation

In order to apply the proposed PRO algorithm to optimize an FL process, it is necessary to formulate it as an optimization problem. Therefore, in this work, the aggregation process is considered and a server-side procedure, as shown in Algorithm (2), is launched after each round to generate the global model by applying weighted averaging as the optimization problem. The PRO algorithm tries to find the best (tuning) coefficients of the global model. Therefore, the objective function is defined as a minimization function Eq. (11). A solution is a set of coefficients $X = \{x^1, x^2, x^3, \ldots, x^{P_t}\}$, where X^j represents a decision variable j = {1,2,3,…, P_t} and P_t is the number of decision variables and the number of participants at round t.

$$\begin{aligned} &min \;\; F_X = F\left(W_{t+1}^G, D^V\right) \\ &where: \\ &\quad W_{t+1}^G = W_t^G + \sum_{i=1}^{P_t} \frac{x_i \times n_i}{N_t^{Total}} \times H_i^{L,U} \\ &\quad x \in \mathrm{R}^N, D^V \in D \end{aligned} \tag{11}$$

In this optimization function, F_X denotes the objective function and calculates the loss value of the global model W_{t+1}^G with validation data samples D^V in round t. Furthermore, n_i is the number of local samples of client *i*. In addition, a dynamic mechanism is utilized to narrow down the feasible domain of decision variables, ensuring they fall between the Lower-Bound (LB) and Upper-Bound (UB) values. Thus mechanism helps maintain the balance between diversification and intensification through the search process, as specified in Eqs. (12) and (13)

$$LP_t \leftarrow LP_{t_0} + \left(\frac{t}{T}\right)^{1/2} \tag{12}$$

$$UP_t \leftarrow UP_{t_0} - \left(\frac{t}{T}\right)^{1/2} \tag{13}$$

where, LP_{t_0} and UP_{t_0} are the initial values of the LB and UB, respectively, and LP_t and UP_t are values of the LB and UB at round t. Also, T is the maximum number of rounds (iteration).

5.3. Results and discussions

The following settings are used in these experiments. The FedSim,[1] an open-source flexible federated deep learning simulator, is used to

[1] https://github.com/AhmadTaheri2021/Federated-Deep-Learning.

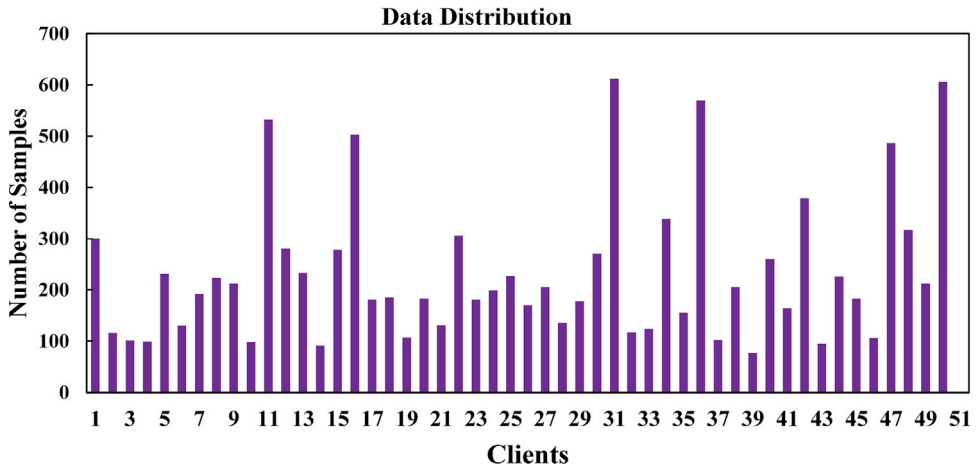


Fig. 12. Data distribution in a non-iid form.

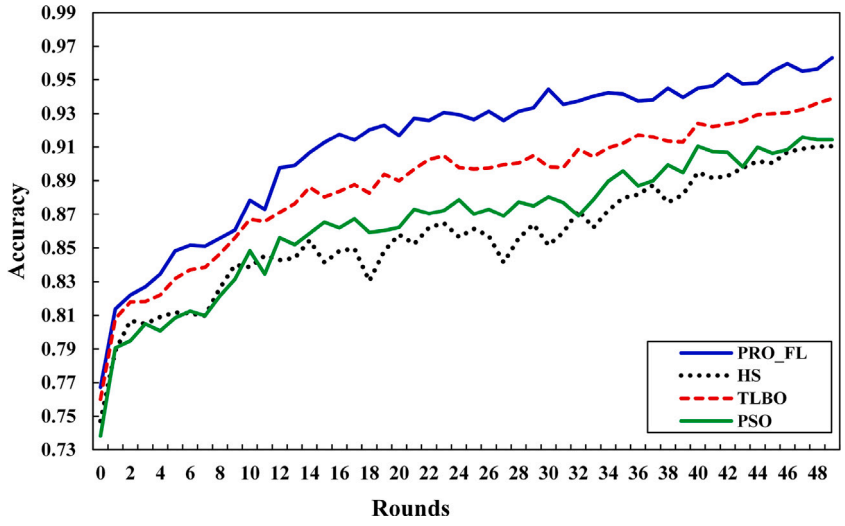


Fig. 13. Comparison the accuracy of the PRO_FL with HS and TLBO algorithms.

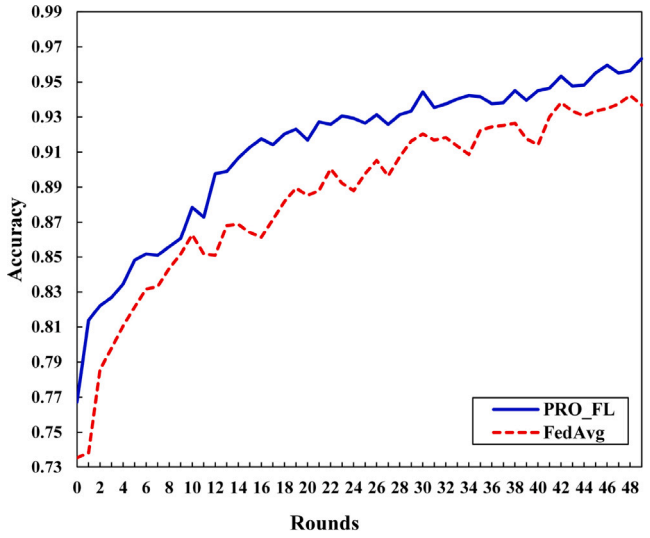


Fig. 14. Comparison the accuracy of the PRO_FL with FedAvg.

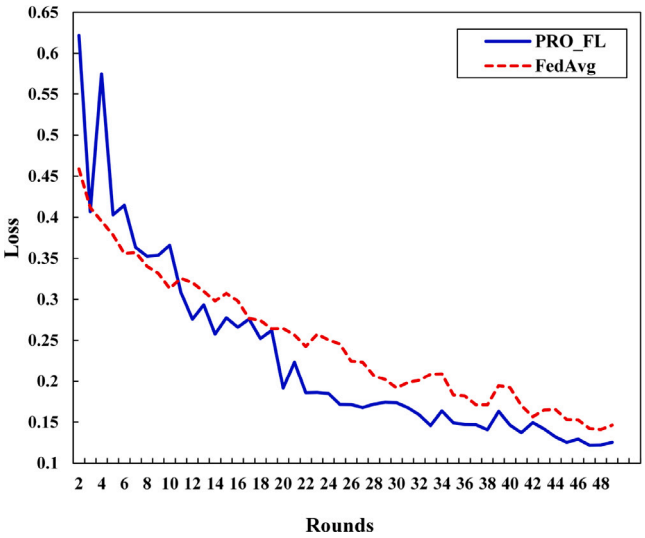


Fig. 15. Comparison the loss of the PRO_FL with FedAvg.

Algorithm 2 The PRO-FL Algorithm

Server Side Procedure:
Initialization:
  Global weights: W_0^G
for each round t = 1, 2, ...T **do**
  Select a set of participants P_t randomly from k Clients
  for each Participant $p = 1, 2, 3, ...P_t$ in parallel **do**
    Download W_t^G to Participant
    Calculate $H_{t,p}^{L,U}, n_{t,p} \leftarrow Update_p\left(W_t^G\right)$
  end for
  Aggregation:

$$W_{t+1}^G \leftarrow W_t^G + \sum_{i=1}^{P_t} \frac{x_i \times n_i}{N_t^{Total}} \times H_i^{L,U}$$

$$X \leftarrow PRO\left(W_{t+1}^G, H_{t,*}^{L,U}, D^V\right) \text{ and } N_t^{Total} \leftarrow \sum_{p=1}^{K} n_{t,p}$$

end forreturn W^G
End of Server Side Procedure

Client Side Procedure:
Initialization:
  Learning rate η, Batch size S, Epoch E
  Replace the local model of participant p $W_{t,p}^L \leftarrow W_t^G$
for local epoch e from 1 to E **do**
  for batch $\beta \in \{\beta_1, \beta_2, \beta_3, ..., \beta_S\}$ **do**
    $W_{t,p,e+1}^{L,U} \leftarrow W_{t,p,e}^{L,U} - \eta \nabla L\left(W_{t,p,e}^{L,U}, \beta\right)$
    $n_{t,p} \leftarrow n_{t,p} + \|\beta\|$
  end for
end for
Calculate and Update $H_{t,p}^{L,U} \leftarrow W_{t,p}^{L,U} - W_t^G$ **return** $(H_{t+1,p}^{L,U}, n_{t,p})$
End of Client Side Procedure

Table 16
Accuracy results.

Metric	Round	HS	PSO	TLBO	FedAvg	PRO
Accuracy (%)	10	83.99	83.14	*85.6*	83.31	**86.05**
	20	84.82	86.02	*89.37*	88.94	**92.3**
	30	86.4	87.49	*90.49*	89.62	**93.33**
	40	88.18	89.47	91.3	*92.65*	**93.95**
	50	91.04	91.43	*93.88*	93.68	**96.32**
	Best	91.04	91.58	93.88	*94.23*	**96.32**

conduct all the experiments. Python 3 and Tensorflow are used as programming languages and deep learning frameworks or tools. The maximum number of rounds, T, is set to 50, and the number of clients is set to 50 for the FL and local optimizers. Additionally, as shown in Fig. 12, the training data samples are artificially distributed in a non-iid form, where each client has a different sample size. The SGD optimizer with local learning rate of 1.0e−03 is employed, and the local epoch is set to 1 for each round.

Table 16 represents the experimental results in terms of the test set accuracy in different intervals. As evident from the results, the proposed PRO algorithm outperforms other competitors. According to results, the PRO algorithm achieves the highest accuracy (96.32%) on the test set in all intervals during the global training. The second highest accuracy is obtained by FedAvg, with an accuracy of 94.23%.

In addition, Figs. 13 to 15 illustrate the convergence rate of the implemented algorithms regarding test set accuracy for an FL-based ECG classification task. It is clear from the results in Figs. 13 to 15 that the proposed algorithm demonstrates improved performance compared to both state-of-the-art population-based optimization algorithms and the well-known FL algorithm FedAvg (Collins et al., 2022).

6. Conclusion

This study proposes a novel optimization algorithm called the Partial Reinforcement Optimizer (PRO), based on the PRE theory, to solve global optimization problems. To investigate the performance of the PRO algorithm, it was applied to minimize 75 benchmark functions commonly used in related works. Statistical analysis, including Wilcoxon and Friedman tests, as well as parametric analysis, has already conducted. The results show that the proposed PRO algorithm outperforms other MAs, demonstrating its capability to solve various types of optimization problems. In addition, to evaluate the robustness and applicability of the proposed algorithm, it was applied to optimize/improve the accuracy of a federated deep learning-based ECG classifier. The Federated Deep Learning (FDL) was formulated as an optimization problem and the PRO algorithm was employed to find the optimal coefficients for fine-tuning the server-side aggregation procedure. The results demonstrate that the proposed PRO algorithm can enhance the accuracy and accelerate the convergence rate of the FDL-based ECG classification task.

To support this endeavor, the code written to evaluate the concepts introduced in this study is accessible as an open capsule on Code Ocean.[2] As a result, the results presented in this paper can be replicated and are open to further enhancements and comparisons with future investigations.

CRediT authorship contribution statement

Ahmad Taheri: Conceptualization, Methodology, Software, Validation, Data curation, Writing – original draft, Writing – review & editing. **Keyvan RahimiZadeh:** Supervision, Conceptualization, Mythology, Investigation, Writing – original draft, Writing – review & editing, Validation. **Amin Beheshti:** Supervision, Conceptualization, Mythology, Validation, Investigation, Writing – original draft, Writing – review & editing. **Jan Baumbach:** Supervision, Conceptualization, Mythology, Validation. **Ravipudi Venkata Rao:** Supervision, Conceptualization, Mythology, Validation. **Seyedali Mirjalili:** Supervision, Conceptualization, Mythology, Validation, Investigation, Writing – review & editing. **Amir H. Gandomi:** Supervision, Conceptualization, Mythology, Validation, Investigation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Appendix

See Fig. 16.

[2] https://codeocean.com/capsule/0955184/tree/v2.

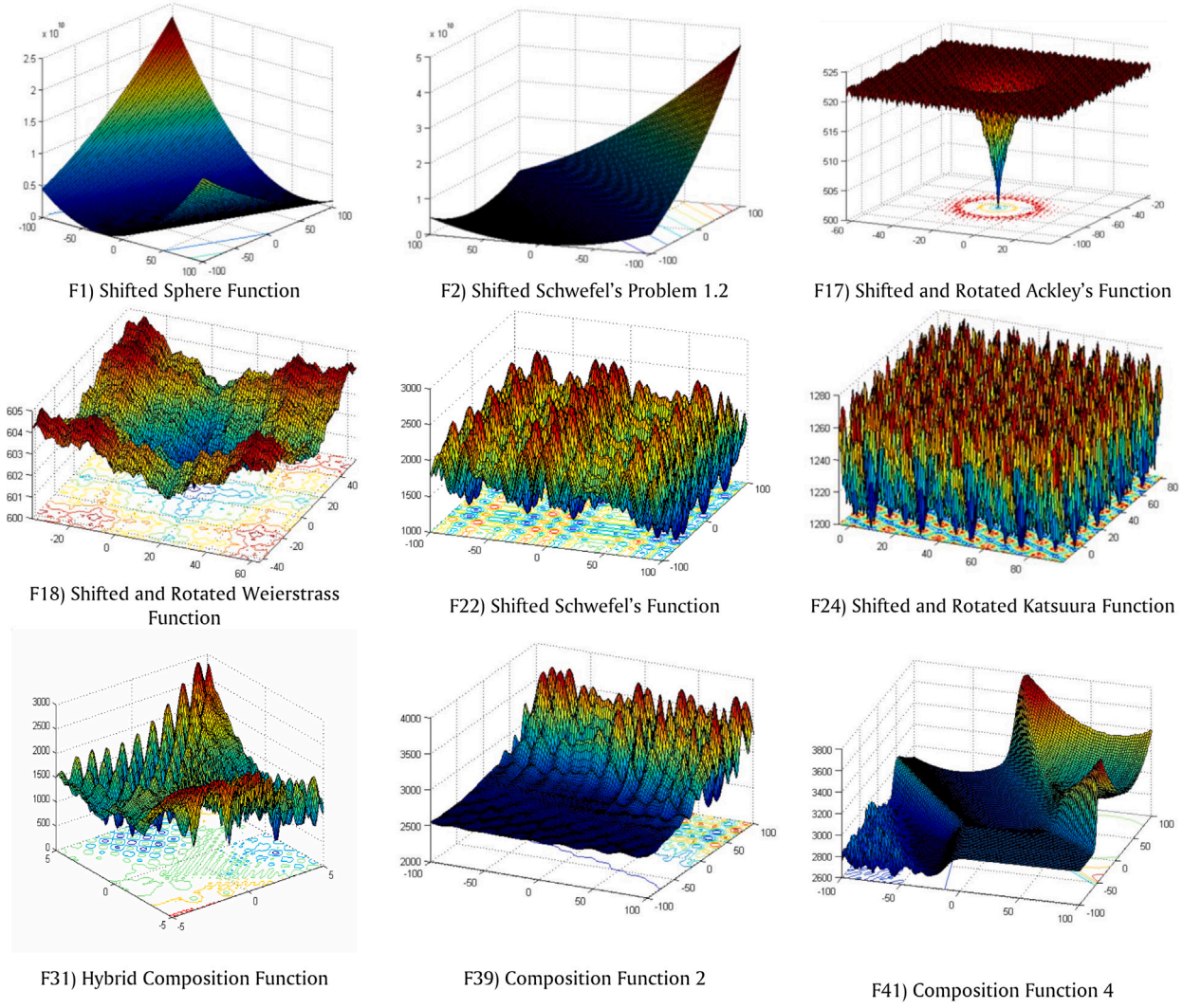


Fig. 16. 3-D map for 2-D benchmark functions.

References

Abdel-Basset, M., El-Shahat, D., Jameel, M., & Abouhawwash, M. (2023). Young's double-slit experiment optimizer : A novel metaheuristic optimization algorithm for global and constraint optimization problems. *Computer Methods in Applied Mechanics and Engineering*, *403*, Article 115652. http://dx.doi.org/10.1016/j.cma.2022.115652.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, *376*, Article 113609. http://dx.doi.org/10.1016/j.cma.2020.113609.

Abualigah, L., Elaziz, M. A., Sumari, P., Geem, Z. W., & Gandomi, A. H. (2022). Reptile search algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Systems with Applications*, *191*, Article 116158. http://dx.doi.org/10.1016/j.eswa.2021.116158, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417421014810.

Aghdam, M. H., Ghasem-Aghaee, N., & Basiri, M. E. (2009). Text feature selection using ant colony optimization. *Expert Systems with Applications*, *36*(3), 6843–6853. http://dx.doi.org/10.1016/j.eswa.2008.08.022, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417408005459.

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*(10), 5011–5042. http://dx.doi.org/10.1007/s00521-020-05296-6.

Askari, Q., Younas, I., & Saeed, M. (2020). Political optimizer: A novel socio-inspired meta-heuristic for global optimization. *Knowledge-Based Systems*, *195*, Article 105709. http://dx.doi.org/10.1016/j.knosys.2020.105709.

Awad, N. H., Ali, M. Z., & Suganthan, P. N. (2017). Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving CEC2017 benchmark problems. In *2017 IEEE congress on evolutionary computation (CEC)* (pp. 372–379). IEEE, http://dx.doi.org/10.1109/CEC.2017.7969336.

Azizi, M., Talatahari, S., & Gandomi, A. H. (2022). Fire hawk optimizer: a novel metaheuristic algorithm. *Artificial Intelligence Review*, http://dx.doi.org/10.1007/s10462-022-10173-w, URL: https://link.springer.com/10.1007/s10462-022-10173-w.

Bujok, P., Tvrdík, J., & Poláková, R. (2019). Comparison of nature-inspired population-based algorithms on continuous optimisation problems. *Swarm and Evolutionary Computation*, *50*, Article 100490. http://dx.doi.org/10.1016/j.swevo.2019.01.006, URL: https://linkinghub.elsevier.com/retrieve/pii/S2210650218301536.

Chen, Y., He, F., Li, H., Zhang, D., & Wu, Y. (2020). A full migration BBO algorithm with enhanced population quality bounds for multimodal biomedical image registration. *Applied Soft Computing*, *93*, Article 106335. http://dx.doi.org/10.1016/j.asoc.2020.106335.

Collins, L., Hassani, H., Mokhtari, A., & Shakkottai, S. (2022). FedAvg with fine tuning: Local updates lead to representation learning. arXiv:2205.13692. URL: http://arxiv.org/abs/2205.13692.

Corder, G. W., & Foreman, D. I. (2009). *Nonparametric statistics for non-statisticians*. Hoboken, NJ, USA: John Wiley & Sons, Inc., http://dx.doi.org/10.1002/9781118165881, URL: http://doi.wiley.com/10.1002/9781118165881.

Domjan, M. P. (1997). *The principles of learning and behavior* (4th ed.). (p. 512). Wadsworth Publishing.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28–39. http://dx.doi.org/10.1109/MCI.2006.329691, URL: http://ieeexplore.ieee.org/document/4129846/.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the sixth international symposium on micro machine and human science* (pp. 39–43). IEEE, http://dx.doi.org/10.1109/MHS.1995.494215, URL: http://ieeexplore.ieee.org/document/494215/.

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications*, *152*, Article 113377. http://dx.doi.org/10.1016/j.eswa.2020.113377, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417420302025.

Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems*, *191*, Article 105190. http://dx.doi.org/10.1016/j.knosys.2019.105190, URL: https://linkinghub.elsevier.com/retrieve/pii/S0950705119305295.

Ferster, C. B., & Skinner, B. F. (1957). *Schedules of reinforcement*. East Norwalk: Appleton-Century-Crofts, http://dx.doi.org/10.1037/10627-000, URL: http://content.apa.org/books/10627-000.

Goodfellow, I., Bengio, Y., & Courville, A. (2017). *Deep learning (adaptive computation and machine learning series)* (pp. 321–359). Cambridge Massachusetts.

Hansen, N., Müller, S. D., & Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, *11*(1), 1–18. http://dx.doi.org/10.1162/106365603321828970.

Hasanzadeh, M., Meybodi, M. R., & Ebadzadeh, M. M. (2013). Adaptive cooperative particle swarm optimizer. *Applied Intelligence*, *39*(2), 397–420. http://dx.doi.org/10.1007/s10489-012-0420-6, URL: http://link.springer.com/10.1007/s10489-012-0420-6.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. http://dx.doi.org/10.48550/arxiv.1512.03385, arXiv:1512.03385. URL: http://arxiv.org/abs/1512.03385.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849–872. http://dx.doi.org/10.1016/j.future.2019.02.028.

Holland, J. H. (1992). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence | MIT press ebooks | IEEE xplore. (p. 211). URL: https://ieeexplore.ieee.org/book/6267401.

Jiang, S., Li, H., Guo, J., Zhong, M., Yang, S., Kaiser, M., & Krasnogor, N. (2020). AREA: An adaptive reference-set based evolutionary algorithm for multiobjective optimisation. *Information Sciences*, *515*, 365–387. http://dx.doi.org/10.1016/j.ins.2019.12.011, URL: https://linkinghub.elsevier.com/retrieve/pii/S0020025519311193.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. http://dx.doi.org/10.1126/science.220.4598.671, URL: https://www.science.org/doi/10.1126/science.220.4598.671.

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*, *111*, 300–323. http://dx.doi.org/10.1016/j.future.2020.03.055, URL: https://linkinghub.elsevier.com/retrieve/pii/S0167739X19320941.

Li, W., Wang, G.-G., & Gandomi, A. H. (2021). A survey of learning-based intelligent optimization algorithms. *Archives of Computational Methods in Engineering*, *28*(5), 3781–3799. http://dx.doi.org/10.1007/s11831-021-09562-1, URL: https://link.springer.com/10.1007/s11831-021-09562-1.

Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). *Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization: Technical report*, URL: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014.

Liang, J.-C., Qu, B., & Suganthan, P. N. (2014). Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization.

Luo, J., He, F., & Gao, X. (2022). An enhanced grey wolf optimizer with fusion strategies for identifying the parameters of photovoltaic models. *Integrated Computer-Aided Engineering*, *30*(1), 89–104. http://dx.doi.org/10.3233/ICA-220693.

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, *188*(2), 1567–1579. http://dx.doi.org/10.1016/j.amc.2006.11.033, URL: https://linkinghub.elsevier.com/retrieve/pii/S0096300306015098.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A. y. (2016). Communication-efficient learning of deep networks from decentralized data. arXiv:1602.05629. URL: http://arxiv.org/abs/1602.05629.

Mirjalili, S. (2016). SCA: A Sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133. http://dx.doi.org/10.1016/j.knosys.2015.12.022, URL: https://linkinghub.elsevier.com/retrieve/pii/S0950705115005043.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. http://dx.doi.org/10.1016/j.advengsoft.2016.01.008, URL: https://linkinghub.elsevier.com/retrieve/pii/S0965997816300163.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007, URL: https://linkinghub.elsevier.com/retrieve/pii/S0965997813001853.

Moody, G., & Mark, R. (2001). The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, *20*(3), 45–50. http://dx.doi.org/10.1109/51.932724, URL: http://ieeexplore.ieee.org/document/932724/.

Nasirigerdeh, R., Bakhtiari, M., Torkzadehmahani, R., Bayat, A., List, M., Blumenthal, D. B., & Baumbach, J. (2020). Federated multi-mini-batch: An efficient training approach to federated learning in non-IID environments. arXiv:2011.07006. URL: http://arxiv.org/abs/2011.07006.

Pan, J.-S., Zhang, L.-G., Wang, R.-B., Snášel, V., & Chu, S.-C. (2022). Gannet optimization algorithm : A new metaheuristic algorithm for solving engineering optimization problems. *Mathematics and Computers in Simulation*, *202*, 343–373. http://dx.doi.org/10.1016/j.matcom.2022.06.007, URL: https://linkinghub.elsevier.com/retrieve/pii/S0378475422002774.

Połap, D., & Woźniak, M. (2021). Red fox optimization algorithm. *Expert Systems with Applications*, *166*, Article 114107. http://dx.doi.org/10.1016/j.eswa.2020.114107.

Rao, R., Savsani, V., & Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303–315. http://dx.doi.org/10.1016/j.cad.2010.12.015, URL: https://linkinghub.elsevier.com/retrieve/pii/S0010448510002484.

Salih, S. Q., & Alsewari, A. A. (2020). A new algorithm for normal and large-scale optimization problems: Nomadic people optimizer. *Neural Computing and Applications*, *32*(14), 10359–10386. http://dx.doi.org/10.1007/s00521-019-04575-1.

Samareh Moosavi, S. H., & Bardsiri, V. K. (2019). Poor and rich optimization algorithm: A new human-based and multi populations algorithm. *Engineering Applications of Artificial Intelligence*, *86*, 165–181. http://dx.doi.org/10.1016/j.engappai.2019.08.025.

Sarwar, S., Hafeez, M. A., Javed, M. Y., Asghar, A. B., & Ejsmont, K. (2022). A horse herd optimization algorithm (HOA)-based MPPT technique under partial and complex partial shading conditions. *Energies*, *15*(5), 1880. http://dx.doi.org/10.3390/en15051880.

Sheller, M. J., Reina, G. A., Edwards, B., Martin, J., & Bakas, S. (2018). Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation. arXiv:1810.04304. URL: http://arxiv.org/abs/1810.04304.

Singh, P. R., Elaziz, M. A., & Xiong, S. (2019). Ludo game-based metaheuristics for global and engineering optimization. *Applied Soft Computing*, *84*, Article 105723. http://dx.doi.org/10.1016/j.asoc.2019.105723.

Singh, S., Singh, H., Mittal, N., Singh, H., Hussien, A. G., & Sroubek, F. (2022). A feature level image fusion for night-vision context enhancement using arithmetic optimization algorithm based image segmentation. *Expert Systems with Applications*, *209*, Article 118272. http://dx.doi.org/10.1016/j.eswa.2022.118272, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417422014129.

Staddon, J. E. R., & Cerutti, D. T. (2003). Operant conditioning. *Annual Review of Psychology*, *54*(1), 115–144. http://dx.doi.org/10.1146/annurev.psych.54.101601.145124, URL: https://www.annualreviews.org/doi/10.1146/annurev.psych.54.101601.145124.

Storn, R., & Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359. http://dx.doi.org/10.1023/A:1008202821328.

Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y. P., Auger, A., & Tiwari, S. (2005). *Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization: Technical report*, URL: http://www.cs.colostate.edu/~genitor/functions.html.

Taheri, A., Ghashghaei, S., Beheshti, A., & RahimiZadeh, K. (2021). A novel hybrid DMHS-GMDH algorithm to predict COVID-19 pandemic time series. In *2021 11th international conference on computer engineering and knowledge (ICCKE)* (pp. 322–327). IEEE, http://dx.doi.org/10.1109/ICCKE54056.2021.9721510, URL: https://ieeexplore.ieee.org/document/9721510/.

Taheri, A., Makarian, E., Manaman, N. S., Ju, H., Kim, T.-H., Geem, Z. W., & RahimiZadeh, K. (2022). A fully-self-adaptive harmony search GMDH-type neural network algorithm to estimate shear-wave velocity in porous media. *Applied Sciences*, *12*(13), 6339. http://dx.doi.org/10.3390/app12136339, URL: https://www.mdpi.com/2076-3417/12/13/6339.

Taheri, A., RahimiZadeh, K., & Rao, R. V. (2021). An efficient balanced teaching-learning-based optimization algorithm with individual restarting strategy for solving global optimization problems. *Information Sciences*, *576*, 68–104. http://dx.doi.org/10.1016/j.ins.2021.06.064, URL: https://linkinghub.elsevier.com/retrieve/pii/S0020025521006551.

Tanabe, R., & Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. In *2013 IEEE congress on evolutionary computation* (pp. 71–78). IEEE, http://dx.doi.org/10.1109/CEC.2013.6557555.

Telikani, A., Tahmassebi, A., Banzhaf, W., & Gandomi, A. H. (2022). Evolutionary machine learning: A survey. *ACM Computing Surveys*, *54*(8), 1–35. http://dx.doi.org/10.1145/3467477, URL: https://dl.acm.org/doi/10.1145/3467477.

Wang, Z., Wang, E., & Zhu, Y. (2020). Image segmentation evaluation: a survey of methods. *Artificial Intelligence Review*, *53*(8), 5637–5674. http://dx.doi.org/10.1007/s10462-020-09830-9, URL: https://link.springer.com/10.1007/s10462-020-09830-9.

Wolpert, D., & Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82. http://dx.doi.org/10.1109/4235.585893, URL: http://ieeexplore.ieee.org/document/585893/.

Xiang, W.-l., An, M.-q., Li, Y.-z., He, R.-c., & Zhang, J.-f. (2014). An improved global-best harmony search algorithm for faster optimization. *Expert Systems with Applications*, *41*(13), 5788–5803. http://dx.doi.org/10.1016/j.eswa.2014.03.016, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417414001456.

Xu, Y., Yang, Z., Li, X., Kang, H., & Yang, X. (2020). Dynamic opposite learning enhanced teaching–learning-based optimization. *Knowledge-Based Systems*, *188*, Article 104966. http://dx.doi.org/10.1016/j.knosys.2019.104966, URL: https://linkinghub.elsevier.com/retrieve/pii/S095070511930396X.

Yang, X.-S. (2010). A new metaheuristic bat-inspired algorithm. (pp. 65–74). http://dx.doi.org/10.1007/978-3-642-12538-6_6, URL: http://link.springer.com/10.1007/978-3-642-12538-6_6.

Yang, Y., Chen, H., Heidari, A. A., & Gandomi, A. H. (2021). Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Systems with Applications*, *177*, Article 114864. http://dx.doi.org/10.1016/j.eswa.2021.114864, URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417421003055.

Zeng, X. T., Li, H., He, F., Luo, J., & Liang, Y. (2020). A novel whale optimization algorithm with filtering disturbance and non-linear step. *International Journal of Bio-Inspired Computation*, *1*(1), 1. http://dx.doi.org/10.1504/IJBIC.2020.10036562.

Zhao, W., Wang, L., & Mirjalili, S. (2022). Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications. *Computer Methods in Applied Mechanics and Engineering*, *388*, Article 114194. http://dx.doi.org/10.1016/j.cma.2021.114194.

Zong Woo Geem, Joong Hoon Kim, & Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68. http://dx.doi.org/10.1177/003754970107600201, URL: http://journals.sagepub.com/doi/10.1177/003754970107600201.