Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 13, No. 1, 2025, 53-61.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Original/Review Paper

# Analyzing the Performance of the Red Deer Optimization Algorithm in Comparison to Other Metaheuristic Algorithms

Soheil Rezashoar* and Amir Abbas Rassafi

*Department of Transportation Planning, Faculty of Engineering, Imam Khomeini International University, Qazvin, Iran.*

**Article Info**

**Article History:**
*Received 01 August 2024*
*Revised 05 September 2024*
*Accepted 07 February 2025*

*DOI:10.22044/jadm.2025.14868.2586*

**Keywords:**
*Optimization, Metaheuristic, Nature Inspired, Red Deer Optimization Algorithm, RDOA.*

**Corresponding author: soheilshoar@edu.ikiu.ac.ir (S. Rezashoar).*

**Abstract**

This study performs a thorough comparative analysis of the Red Deer Optimization Algorithm (RDOA) in comparison to five well-established metaheuristic algorithms: Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE), Artificial Bee Colony (ABC), and Whale Optimization Algorithm (WOA). The main objective is to evaluate the performance of RDOA on a range of benchmark problems, including essential unimodal and sophisticated multimodal functions. The methodology incorporates hyperparameters optimization for each algorithm to optimize performance and assesses them on six standard benchmark problems (Sphere, Rosenbrock, Bohachevsky, Griewank, Rastrigin, and Eggholder). Convergence plots are examined to demonstrate the rate at which convergence occurs and the level of stability achieved. The results demonstrate that RDOA performs well compared to other algorithms in all benchmarks and excels in dealing with multimodal functions. However, the selection of an algorithm should be based on the specific characteristics of the problem, taking into account their distinct advantages.

## 1. Introduction

Optimization problems are common in many areas, such as engineering design, resource allocation, data analysis, and machine learning. Finding the best solutions to these problems, often defined by their complexity and lack of linearity, requires new ideas. Because they are based on actual and imagined events, metaheuristic algorithms are now useful tools for solving complex problems [1]. Metaheuristic algorithms are optimization techniques designed to address a range of optimization challenges effectively. These algorithms differentiate themselves from other optimization methods in several respects. Firstly, in contrast to gradient-based search techniques, derivative-free methods do not depend on the computation of derivatives inside the search space. Metaheuristic algorithms are significantly streamlined, augmented in adaptability, and improved in their capacity to circumvent local optima, leading to their considerable efficacy in addressing intricate optimization challenges. Metaheuristic algorithms are stochastic, commencing the optimization process by generating random results. This increases the probability that the algorithms will successfully evade premature convergence and efficiently traverse the search space. Metaheuristics attain an equilibrium between exploration and exploitation to accomplish their objectives. During the exploration phase, the algorithms extensively explore the noteworthy areas of the search space. Localized searches are conducted in these locations during the exploitation phase to determine the ideal solution. Metaheuristics have demonstrated efficacy in numerous electrical engineering applications. These encompass enhancing power generation, scheduling, and transportation in industrial environments, engineering bridges and buildings in civil construction, creating radar systems and networks in communications, and executing classification,

prediction, clustering, and system modeling in data mining. There exist various categories of metaheuristic search techniques [2], including nature-inspired and non-nature-inspired, single and multiple neighborhood structures, memory usage and memory-less approaches, single objective and multiple objective-based techniques, stochastic and deterministic methods, discrete and continuous algorithms, population-based and single point search techniques, and dynamic and static objective functions.

Recently, researchers have categorized nature-inspired algorithms into various groups, including evolutionary algorithms, swarm intelligence, physical-based algorithms, and human-based algorithms [3]. The human-based algorithms replicate human behaviors and activities. Examples of human-based methods include Teaching-Learning Based Optimization (TLBO) [4], Harmony Search (HS) [5], Tabu Search (TS) [6, 7], Group Search Optimizer (GSO) [8], and War Strategy Optimization (WSO) [9]. In evolution-based algorithms that are based on natural evolutionary principles, a random starting population is made, and each generation after that evolves. The most renowned evolution-based technique is the Genetic Algorithm (GA) [10]. In addition, the Evolution Strategy (ES) [11], Biogeography-Based Optimizer (BBO) [12], and Population-Based Incremental Learning (PBIL) [13] can also be put into this group. In swarm-based algorithms inspired by animal groups' social behavior, several entities traverse a search space to identify the optimal answer. The predominant technique swarm-based groups use is Particle Swarm Optimization (PSO) [14]. Moreover, the subsequent methods may also be categorized within this group: Ant Colony Optimization (ACO) [15], Bird Mating Optimizer (BMO) [16], Fruit-fly Optimization Algorithm (FOA) [17], Gannet Optimization Algorithm (GOA) [18], Artificial Rabbits Optimization (ARO) [19], Starling Murmuration Optimizer (SMO) [20], and Artificial Jellyfish Search (AJS) [21]. Physics-based algorithms emulate the laws of physics in the universe. The subsequent methods exemplify this category: Simulated Annealing (SA) [22], Big-Bang Big-Crunch (BB-BC) [23], Central Force Optimizer (CFO) [24], and Gravitational Search Algorithm (GSA) [25].

Figure 1 clearly shows the trend line of the number of algorithms published per year, which has a coefficient of determination (R2) of 0.926. A trend line is considered very reliable when its R2 value is close to or equal to 1. This result indicates researchers' special interest and attention in developing and presenting new meta-heuristic algorithms, with the most significant number of these methods being published between 2020 and 2022.

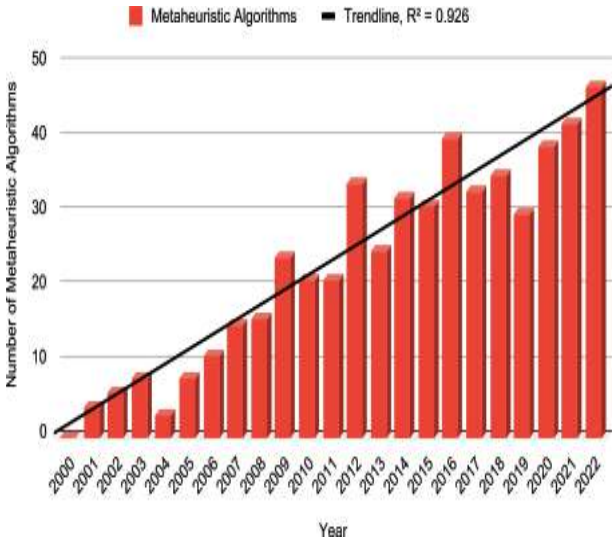


**Figure 1. Number of Metaheuristic Algorithms Developed During 2000–2022 [26].**

Figure 2 illustrates that the PSO is the most commonly utilized, with over 75,000 citations. The GA is considered the second most often used algorithm. The ACO, Differential Evolution (DE), and SA hold the positions of third, fourth, and fifth, respectively, in the ranking. The algorithms TS, GWO, Artificial Bee Colony (ABC), Cuckoo Search (CS), and HS are ranked as the fifth, sixth, seventh, eighth, ninth, and tenth most mentioned algorithms to date, respectively.

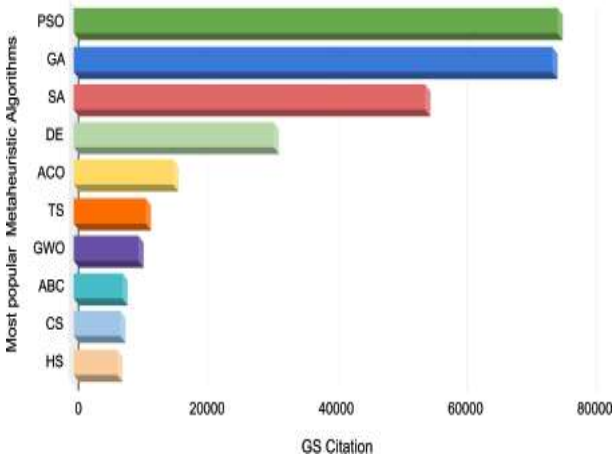


**Figure 2. Top Ten Cited MAs, Data Source—Google Scholar (GS) on December 31, 2022 [26].**

GAs, PSO, and SA are widely recognized and thoroughly researched metaheuristic algorithms [14, 27, 28]. GAs simulate natural selection by iteratively generating and improving a population of viable solutions over multiple generations. PSO is derived from social behavior and imitates the collective movement of birds or fish. SA utilizes thermodynamic annealing techniques to examine the range of possible solutions. These algorithms have been effectively utilized in various fields, including engineering, logistics, finance, and machine learning.

Red Deer Optimization Algorithm (RDOA), a newly developed metaheuristic algorithm, sets itself apart by taking inspiration from the natural world, particularly the behavior of red deer (Cervus elaphus). The method was proposed by Fathollahi Fard et al. [29] in 2020 and has subsequently attracted interest due to its distinctive optimization strategy. The algorithm emulates the characteristics of red deer, such as searching for food, being watchful, and grouping to explore and efficiently exploit the solution space.

RDOA distinguishes itself from previous algorithms by depending on these innate characteristics, presenting a fresh approach to problem-solving in optimization. The subject's originality and possible uses have generated increasing interest within the optimization community.

Although RDOA is a relatively new contribution to the discipline, researchers have already started investigating its possibilities and uses. RDOA has been utilized in diverse functions such as engineering design, robotics, and image processing, demonstrating its capacity to address intricate optimization challenges effectively. The preliminary research has demonstrated encouraging outcomes, suggesting that RDOA justifies additional examination and comparison with well-established metaheuristic algorithms.

This study paper conducts a thorough comparative analysis, comparing RDOA with established metaheuristic algorithms. Our main objective is to analyze RDOA's effectiveness, convergence properties, and adaptability in addressing optimization functions across various levels of difficulty and fields.

## Research Methodology

### 2.1. Algorithm Phases

#### 2.1.1. Initialization and Population

The first random population in RDOA is called "Red Deers" (RDs) and is divided into two groups: "male RDs" and "hinds." This starting population lays the groundwork for later optimization procedures, in which male RDs vie for harem formation, a crucial component of the algorithm's architecture.

$$RD = X\,1, X\,2, X\,3, \ldots, X_{Nvar} \tag{1}$$

$$Value = f\,(RD) = f\,(X\,1, X\,2, X\,3, \ldots, X_{Nvar}) \tag{2}$$

#### 2.1.2. Roaring Phase

Male RDs mimic the roaring behavior of red deer to start the optimizing process during the roaring phase. Local search activities are used in this phase to enhance the exploitation of possible solutions. To diversify, male RDs investigate their communities while adding randomization elements.

$$male_{new} = \begin{cases} male_{old} + a_1 \times \left( \left( (UB - LB) * a_2 \right) + LB \right), \\ if\ a_3 \geq 0.5 \\ male_{old} - a_1 \times \left( \left( (UB - LB) * a_2 \right) + LB \right), \\ if\ a_3 < 0.5 \end{cases} \tag{3}$$

UB and LB constrain the search space to formulate an appropriate male neighborhood solution. They represent the upper and lower limits of the search space. Observe that the current position of male RD is $male_{old}$, and its subsequent position is $male_{new}$. Three steps, $\alpha 1$, $\alpha 2$, and $\alpha 3$, comprise the randomization process, originating from a uniform distribution between 0 and 1.

#### 2.1.3. Selection of Commanders

Not every male RD has the same skills. The algorithm's identification of a subset of male RDs as "commanders" gives users authority over the ratio of intensification to diversification. Commanders shape harems and also impact exploration and exploitation features.

$$N_C = round\,(\gamma . N_{male}) \tag{4}$$

$$N_s = N_{male} - N_C \tag{5}$$

where $N_C$ variable represents the quantity of commanders that are inherently male, the $\gamma$ variable denotes a random integer within the range of 0 to 1, and the $N_{male}$ variable signifies the total count of males. It is important to acknowledge the $\gamma$ is the initial value of the algorithm model. Its value range extends from 0 to 1.

#### 2.1.4. Fighting Phase

Random conflicts break out between stags and commanders. These conflicts simulate the battle between male RDs to get harems with more hinds. The fighting phase improves exploitation by using local search operations and choosing the best options.

$$new_1 = (C+S)/2 + b_1((UB-LB)*b_2+LB) \quad (6)$$

$$new_2 = (C+S)/2 - b_1((UB-LB)*b_2+LB) \quad (7)$$

The two novel solutions produced through the combat process are $new_1$ and $new_2$. The designations for commanders and stags are C and S, respectively. UB and LB provide the upper and lower boundaries on the viability of the new solutions. The upper and lower boundaries ($b_1$ and $b_2$) of the search space are established through the randomization of the combat process between 0 and 1, employing a uniform distribution function. Among the four options: C, S, $new_1$ and $new_2$, only the most favorable case will be selected based on Objective Fitness (OF). The OF of a male commander directly affects the magnitude of his harem, denoting the number of female hinds under his control. Commanders with elevated OF tend to amass larger harems owing to their enhanced combat and vocalization capabilities.

### 2.1.5. Formation of Harems

Commanders create harems, which represent groups of hinds. Exploration is impacted by the commanders' fitness, which is directly related to the number of hinds in each harem. The distribution of harems across commanders gives the algorithm's exploration stage an additional dimension.

$$V_n = v_n - \max v_i \quad (8)$$

where $V_n$ represents the normalized value of the $n_{th}$ commander's power (i.e., its OF), and $v_n$ denotes the power of the $n_{th}$ commander (i.e., its OF).

The normalized power of commanders can be calculated using the subsequent formula.

$$P_n = \left| \frac{V_n}{\sum_{i=1}^{a_i} V_i} \right| \quad (9)$$

The following formula can be used to determine a harem's number of hinds, where $N_{hind}$ is the total number of hinds.

$$N.harem_n = round(P_n \cdot N_{hind}) \quad (10)$$

### 2.1.6. Mating within Harems

A portion of the hinds in commanders' harems mate with them. Users can regulate diversity and exploration by adjusting the number of hinds mate with commanders using the "$\alpha$" parameter. This stage promotes population diversity and exploration.

$$N.harem_n^{mate} = round(\alpha \cdot N.harem_n) \quad (11)$$

The quantity of hinds in $n_{th}$ the harem that copulate with their leader is $N.harem_n^{mate}$. Regarding the solution space, we select $N.harem_n^{mate}$ of the $N.harem_k$ at random. The mating process is generally explained as follows.

$$\text{offs} = \frac{C + Hind}{2} + (UB - LB) \quad \text{x} \quad c \quad (12)$$

### 2.1.7. Mating across Harems

Commanders can increase the size of their domains by attacking other harems and mating with hinds from other harems (call it k). This process is controlled by the "$\beta$" parameter, which gives users authority over exploration and exploitation. This stage increases the algorithm's capacity for exploration.

$$N.harem_k^{mate} = round(\beta \cdot N.harem_k) \quad (13)$$

where $N.harem_k^{mate}$ is the number of hinds in the $k_{th}$ harem that mate with the commander. It is important to note that Equation (12) is used to carry out the mating process.

### 2.1.8. Mating of Stags

In order to mimic the natural behavior of red deer during the breeding season, each stag mates with the closest hind. This process balances exploration and exploitation aspects. The following formula should be used to determine the distance in $J-dimension$ space between a stag and every hind.

$$d_i = \{\sum_{j \partial J} (stag_j - hind_j^i)^2\}^{1/2} \quad (14)$$

where $d_i$ is the separation between a stag and its $i_{th}$ hind, the lowest value in this matrix represents the hind chosen. The mating process begins after a hind has been chosen. In this calculation, a stag is considered instead of a commander.

### 2.1.9. Selection of the Next Generation

The next generation is determined through two strategies: retaining elite solutions and selecting offspring based on fitness values. The selection process shapes the final population and concludes the iterative optimization cycle. The RDOA offers a unique optimization approach, allowing users to fine-

tune its behavior according to the characteristics of the problem at hand.

All the steps mentioned are presented in Figure 3 and the four main stages of RDOA are presented in Figure 4.

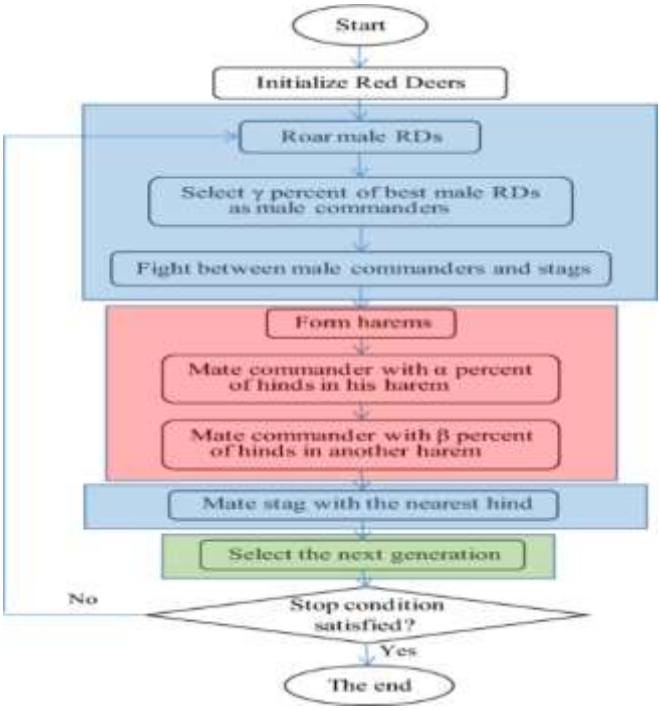


**Figure 3. Flowchart of the RDOA [29].**

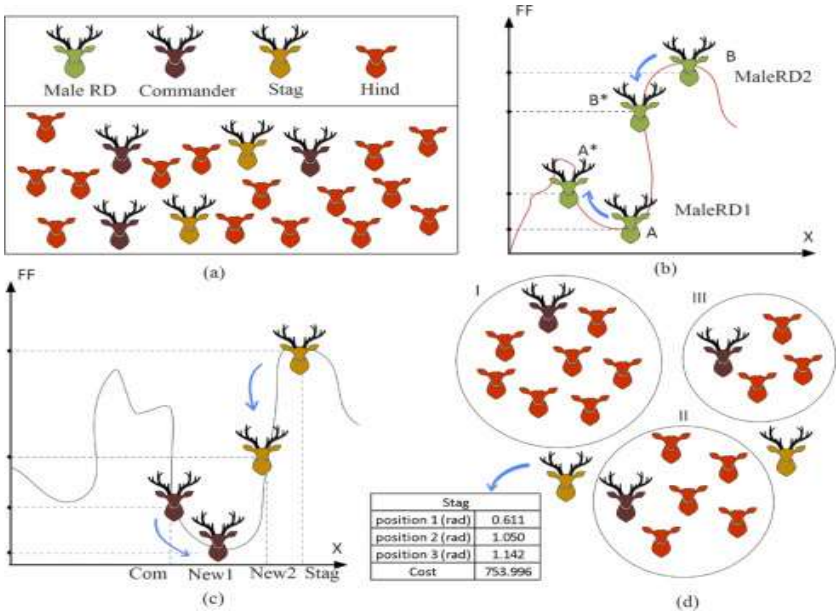


**Figure 4. Stages of the RDOA: (a) Population of RD, (b) Roaring Process, (c) Fighting Process, (d) Harems [30].**

Our research aims to comprehensively evaluate RDOA using benchmark functions to identify its strengths and limitations. To achieve this goal, we carefully planned and executed a set of tests using a variety of standard functions covering a wide range of difficulty levels. Using this framework, we could evaluate the performance of RDOA under different optimization scenarios and make relevant comparisons with other well-known metaheuristic techniques. Our methodology consists of several essential elements, including identifying and selecting benchmark functions with different difficulty levels, selecting comparison algorithms, determining parameter settings of each of the selected algorithms, designing and executing

experiments, and evaluating the performance and results obtained from each algorithm.

The algorithms selected for comparison are:

- **GA:** A population-based optimization technique inspired by the process of natural selection and genetics.
- **PSO:** A swarm intelligence-based algorithm that simulates the behavior of birds flocking or fish schooling.
- **DE:** A population-based optimization algorithm that operates on a group of candidate solutions [31].
- **ABC:** A bio-inspired optimization algorithm based on the foraging behavior of honeybees [32].
- **Whale Optimization Algorithm (WOA):** A nature-inspired algorithm inspired by the hunting behavior of humpback whales [33].

## 2. Results and Discussion

This section presents and analyzes the results obtained from our comparative analysis of the RDOA with various well-known metaheuristic algorithms. We utilized Python to develop all six metaheuristic algorithms and the RDOA. To achieve an even comparison, we optimized the hyperparameters of each algorithm to generate the most optimal outcomes. The goal was to identify the most effective hyperparameter settings that would allow for a thorough and unbiased evaluation of the algorithms. We implemented each method on the chosen benchmark functions and documented the optimal solutions achieved by each algorithm.

The following benchmark functions were chosen for this study:

- **Sphere Function:** A fundamental unimodal function, often used as a starting point for optimization algorithms due to its simplicity.
- **Rosenbrock Function:** A classic multimodal problem, characterized by a narrow, curved valley that poses challenges for optimization algorithms.
- **Bohachevsky Function:** A multimodal problem with a pair of symmetric minima, serving as a test of the algorithms' ability to locate multiple optima.
- **Griewank Function:** A multimodal problem with a flat, expansive basin around the global minimum, testing the exploration capabilities of the algorithms.
- **Rastrigin Function:** A non-convex, highly multimodal problem replete with local minima, designed to evaluate the robustness of optimization algorithms.
- **Eggholder Function:** A highly non-linear, intricate problem with a complex landscape featuring multiple peaks and valleys, serving as a rigorous test of an algorithm's exploration and exploitation prowess.

Table 1 shows the standard equations, search intervals, and global minimum of each function for optimization. The number of iterations is 500, so the algorithms have sufficient time in the search space. Also, the dimensions of each of the mentioned functions are 100. Due to implementation limitations, the algorithms were run only once. Repeated experiments are recommended in future research to obtain highly reliable results.

Table 2 summarizes the results for each pair of function and algorithm.

Figure 5 exhibits the convergence graphs for each of the benchmark functions. The charts illustrate the progression of the optimization process at each epoch, showcasing the algorithm's performance on various problem landscapes. The RDOA algorithm demonstrated impressive convergence speed in several cases, mainly when used for unimodal functions such as the Sphere Function. The fact that it can rapidly identify the global optimum in such scenarios indicates its effectiveness in handling less complex optimization problems. RDOA's agility makes it an excellent choice for applications that demand quick convergence. RDOA demonstrated robustness and outperformed other algorithms on the Griewank Function, which poses a difficult balance between local and global optima. This capacity is essential when addressing real-world problems characterized by complex solution spaces. The Rosenbrock Function demonstrated the competitive performance of the system, indicating its capacity to handle complex multimodal problems. However, further adjustments may be necessary to improve its performance on these functions. The findings suggest that specific algorithms exhibit superior performance on particular benchmark functions. For instance, when applied to the Sphere Function, the WOA demonstrates exceptional performance. However, the PSO yields remarkable outcomes when used with the Eggholder Function.

**Table 1. Benchmark Functions for Optimization.**

| Functions | Equation | Search Range | Global Minimum |
|---|---|---|---|
| **Sphere** | $f(x)=\sum_{i=1}^{D} x_i^2$ | [-100, 100] | $fx*=0, atx*=(0,\ldots,0)$ |
| **Rosenbrock** | $f(x)=\sum_{i=1}^{D-1}\left(100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right)$ | [-100, 100] | $fx*=0, atx*=(1,\ldots,1)$ |
| **Bohachevsky** | $f(x)=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)-\frac{4}{10}\cos(4\pi x_2)+\frac{7}{10}$ | [-100, 100] | $fx*=0, atx*=(0,\ldots,0)$ |
| **Griewank** | $f(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$ | [-600, 600] | $fx*=0, atx*=(0,\ldots,0)$ |
| **Rastrigin** | $f_{10}(x)=\sum_{i=1}^{D}\left(x_i^2-10\cos(2\pi x_i)\right)+10\cdot D$ | [-100, 100] | $fx*=0, atx*=(0,\ldots,0)$ |
| **Eggholder** | $f(x)=-(x_2+47)\sin\sqrt{\left\lvert\frac{x_1}{2}+x_2+47\right\rvert}-x_1\sin\sqrt{\lvert x_1-(x_2+47)\rvert}$ | [-512, 512] | $fx*=-959.6407, atx*=(512,404.2319)$ |

**Table 2. Summary of the Results Obtained for Each Function-Algorithm Pair.**

| Functions | GA | PSO | DE | ABC | WOA | RDOA |
|---|---|---|---|---|---|---|
| **Sphere** | 9.21e+02 | 5.32e+03 | 4.45e+03 | 3.45e+00 | 5.73e-92 | 4.55e-05 |
| **Rosenbrock** | 1.27e+07 | 4.10e+07 | 6.24e+07 | 1.48e+03 | 2.50e-01 | 5.00e-02 |
| **Bohachevsky** | 0.00e+00 | 0.00e+00 | 0.00e+00 | 4.00e-02 | 0.00e+00 | 0.00e+00 |
| **Griewank** | 1.58e+00 | 1.95e+00 | 1.21e+00 | 5.00e-02 | 0.00e+00 | 1.41e-05 |
| **Rastrigin** | 2.25e+03 | 7.21e+03 | 1.82e+03 | 9.26e+01 | 0.00e+00 | 3.00e-02 |
| **Eggholder** | 1.00e-02 | -3.10e+09 | -1.16e+04 | -1.89e+04 | -1.98e+04 | -1.98e+04 |

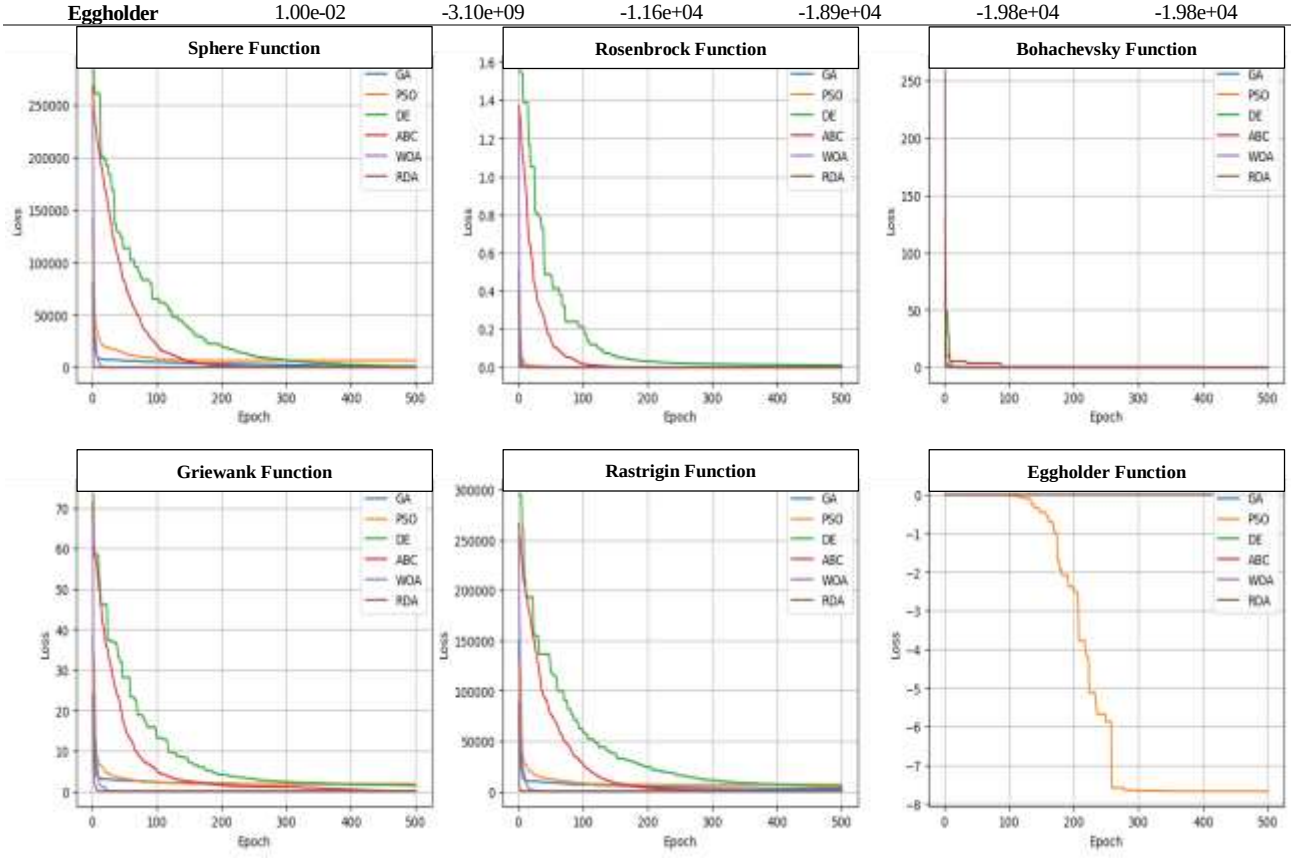


**Figure 5. Convergence Plots for the Six Benchmark Functions.**

## 3. Conclusions

In this paper, a comparison of the performance of the RDOA with five other well-known algorithms was conducted by running on six benchmark functions. These functions include:

- Sphere
- Rosenbrock
- Bohachevsky
- Griewank
- Rastrigin
- Eggholder

This research showed that most algorithms encountered difficulties escaping from local minima when using the Eggholder Function. However, the PSO approach achieved a significantly superior result. Although RDOA did not yield significant results compared to other methods, it showed a significantly faster convergence speed. The RDOA algorithm showed a significant convergence speed in several cases, mainly when used for unimodal functions such as the Sphere Function. The fact that it could quickly identify the global optimum in such scenarios indicates its effectiveness in handling less complex optimization problems. One of the striking features of RDOA was its ability to navigate effectively in complex and diverse environments. RDOA demonstrated robustness and outperformed other algorithms in the Griewank Function, which balances the difficulty between local and global optima. This capacity is essential when addressing real-world problems characterized by complex solution spaces. The Rosenbrock Function demonstrated the competitive performance of the system, indicating its capacity to handle complex, multi-faceted problems. However, further tuning may be required to improve its performance in these functions. A fundamental feature of RDOA is its ability to manage the trade-off between exploration and exploitation effectively. The process rushes through the solution space to identify prospective optimal solutions while successfully exploiting previously discovered regions. The balance of RDOA allows it to quickly adjust to different types of performance, making it highly versatile for many optimization scenarios. Finally, RDOA is still a new and rich algorithm, so future work will involve improvising variants of this algorithm and applying them to engineering problems in optimization and machine learning.

## References

[1] A. W. Mohamed, K. M. Sallam, P. Agrawal, A. A. Hadi, and A. K. Mohamed, "Evaluating the performance of meta-heuristic algorithms on CEC 2021 benchmark problems," *Neural Computing and Applications*, vol. 35, no. 2, pp. 1493-1517, 2023.

[2] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 268-308, 2003.

[3] F. Fausto, A. Reyna-Orta, E. Cuevas, Á. G. Andrade, and M. Perez-Cisneros, "From ants to whales: metaheuristics for all tastes," *Artificial Intelligence Review*, vol. 53, pp. 753-810, 2020.

[4] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303-315, 2011.

[5] K. S. Lee and Z. W. Geem, "A new structural optimization method based on the harmony search algorithm," *Computers & Structures*, vol. 82, no. 9-10, pp. 781-798, 2004.

[6] F. Glover, "Tabu search—part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190-206, 1989.

[7] F. Glover, "Tabu search—part II," *ORSA Journal on Computing*, vol. 2, no. 1, pp. 4-32, 1990.

[8] L. Abualigah, "Group search optimizer: a nature-inspired meta-heuristic optimization algorithm with its results, variants, and applications," *Neural Computing & Applications*, vol. 33, no. 7, 2021.

[9] T. S. Ayyarao, N. Ramakrishna, R. M. Elavarasan, N. Polumahanthi, M. Rambabu, G. Saini, B. Khan, and B. Alatas, "War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization," *IEEE Access*, vol. 10, pp. 25073-25105, 2022.

[10] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66-73, 1992.

[11] G. Rudolph, "Evolution strategies," *Evolutionary Computation*, vol. 1, pp. 81-88, 2000.

[12] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702-713, 2008.

[13] D. Dasgupta and Z. Michalewicz, "Evolutionary algorithms in engineering applications," *Springer Science & Business Media*, 2013.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, IEEE, 1995.

[15] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39, 2007.

[16] A. Askarzadeh and A. Rezazadeh, "A new heuristic optimization algorithm for modeling of proton exchange

membrane fuel cell: bird mating optimizer," *International Journal of Energy Research*, vol. 37, no. 10, pp. 1196-1204, 2013.

[17] W.-T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69-74, 2012.

[18] J.-S. Pan, L.-G. Zhang, R.-B. Wang, V. Snášel, and S.-C. Chu, "Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems," *Mathematics and Computers in Simulation*, vol. 202, pp. 343-373, 2022.

[19] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, "Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 114, pp. 105082, 2022.

[20] H. Zamani, M. H. Nadimi-Shahraki, and A. H. Gandomi, "Starling murmuration optimizer: A novel bio-inspired algorithm for global and engineering optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 392, pp. 114616, 2022.

[21] J.-S. Chou and D.-N. Truong, "A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean," *Applied Mathematics and Computation*, vol. 389, pp. 125535, 2021.

[22] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671-680, 1983.

[23] O. K. Erol and I. Eksin, "A new optimization method: big bang–big crunch," *Advances in Engineering Software*, vol. 37, no. 2, pp. 106-111, 2006.

[24] R. Formato, "Central force optimization: a new metaheuristic with applications in applied electromagnetics," *Progress in Electromagnetics Research*, vol. 77, pp. 425-491, 2007.

[25] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232-2248, 2009.

[26] K. Rajwar, K. Deep, and S. Das, "An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges," *Artificial Intelligence Review*, vol. 56, no. 11, pp. 13187-13257, 2023.

[27] D. E. Goldberg and J. H. Holland, "Genetic Algorithms and Machine Learning," *Machine Learning*, vol. 3, no. 2, pp. 95-99, 1988.

[28] S. Kirkpatrick, "Improvement of reliabilities of regulatins using a hierarchical structure in a genetic network" *Science*, vol. 220, pp. 671-680, 1983.

[29] A. . Fathollahi-Fard, M. Hajiaghaei-Keshteli, and R. Tavakoli-Moghaddam, "Red deer algorithm (RDA): a new nature-inspired meta-heuristic," *Soft Computing*, vol. 24, pp. 14637-14665, 2020.

[30] Y. Bektaş and H. Karaca, "Red deer algorithm based selective harmonic elimination for renewable energy application with unequal DC sources," *Energy Reports*, vol. 8, pp. 588-596, 2022.

[31] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.

[32] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *International Fuzzy Systems Association World Congress*, Springer, 2007.

[33] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51-67, 2016.

# تحلیل عملکرد الگوریتم بهینه سازی گوزن قرمز در مقایسه با سایر الگوریتم های فراابتکاری

**سهیل رضاشعار* و امیر عباس رصافی**

**گروه برنامه‌ریزی حمل‌ونقل، دانشکده فنی‌ومهندسی، دانشگاه بین‌المللی امام خمینی(ره)، قزوین، ایران.**

ارسال ۲۰۲۴/۰۸/۰۱؛ بازنگری ۲۰۲۴/۰۹/۰۵؛ پذیرش ۲۰۲۵/۰۲/۰۷

**چکیده:**

این مطالعه یک تحلیل مقایسه‌ای از الگوریتم بهینه‌سازی گوزن قرمز (RDOA) در مقایسه با پنج الگوریتم فراابتکاری به خوبی اثبات شده انجام داده است: الگوریتم ژنتیک (GA)، بهینه سازی ازدحام ذرات (PSO)، تکامل دیفرانسیلی (DE)، کلونی زنبورهای مصنوعی (ABC) و الگوریتم بهینه سازی نهنگ (WOA). هدف اصلی ارزیابی عملکرد RDOA در طیفی از مسائل، از جمله توابع چندوجهی ضروری و پیچیده است. این روش شامل بهینه‌سازی فراپارامتر برای هر الگوریتم جهت بهینه‌سازی عملکرد آن می‌شود و آن ها را بر اساس شش مسئله استاندارد (Sphere، Rosenbrock، Bohachevsky، Griewank، Rastrigin و Eggholder) ارزیابی کرده است. نمودارهای همگرایی برای نشان دادن نقطه‌ای که در آن همگرایی رخ می‌دهد و سطح ثبات به دست آمده مورد بررسی قرار گرفتند. نتایج نشان دادند که RDOA در مقایسه با سایر الگوریتم‌ها در توابع در نظر گرفته شده خوب عمل می‌کند و در برخورد با توابع چندوجهی برتری دارد. با این حال، انتخاب یک الگوریتم باید بر اساس ویژگی‌های خاص مسئله، با در نظر گرفتن مزایای متمایز آن ها باشد. این کار یک منبع عالی برای محققان بهینه‌سازی است. بینش‌هایی در مورد انتخاب الگوریتم ارائه می‌دهد و نقاط قوت و محدودیت‌های RDOA را برجسته می‌کند.

**کلمات کلیدی:** بهینه‌سازی، فراابتکاری، الهام گرفته از طبیعت، الگوریتم بهینه‌سازی گوزن قرمز، RODA.