Article

# A novel Magnificent Frigatebird Optimization algorithm with proposed movement strategies for enhanced Global Search

**Glykeria Kyrou[1], Vasileios Charilogis[2] and Ioannis G. Tsoulos[3,*]**

[1] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece; g.kyrou@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece;itsoulos@uoi.gr
* Correspondence: itsoulos@uoi.gr

**Abstract**

Global optimization plays a critical role in solving complex real-world problems, where identifying the optimal solution within a high-dimensional and nonlinear search space is essential. Metaheuristic algorithms, inspired by natural phenomena and biological processes, have demonstrated significant effectiveness in addressing such challenges. In this context, the present study introduces an enhanced variant of the Magnificent Frigatebird Optimization (MFO) algorithm, a bio-inspired metaheuristic model that simulates the kleptoparasitic behavior of frigatebirds. The proposed variant incorporates novel movement strategies aimed at improving both convergence speed and solution quality. Specifically, local search via the BFGS algorithm for refined exploitation, and a dynamic termination criterion that detects convergence and prevents stagnation. The algorithm is validated on an extensive set of benchmark functions commonly found in the optimization literature, demonstrating superior performance compared to traditional evolutionary approaches.

**Keywords:** Global optimization; evolutionary methods; stochastic methods

Received:
Revised:
Accepted:
Published:

**Citation:** Kyrou, G.; Charilogis, V.; Tsoulos, I.G. A Novel Magnificent Frigatebird Optimization algorithm with proposed movement strategies for enhanced Global Search. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/



## 1. Introduction

The primary objective of global optimization is to locate the global minimum of a continuous and multidimensional function, in such a way as to ensure complete exploration of the search space. Global optimization aims to examine the entire problem domain in order to find the lowest possible value that is feasible. This procedure is applied to complex functions which usually include multiple local minima, making it difficult to identify the global minimum. Global optimization includes techniques that ensure that local optima are avoided while focusing on maximizing the accuracy and efficiency of the search process. The objective is to find the lowest point through systematic exploration of the entire domain of the function$f : S \to R, S \subset R^n$ and it is defined as follows:

$$x^* = \arg\min_{x \in S} f(x) \tag{1}$$

where the set $S$ is defined as follows:

$$S = [a_1, b_1] \times [a_2, b_2] \times \ldots [a_n, b_n]$$

Global optimization constitutes one of the most dynamically evolving fields of applied mathematical analysis and computer science, as it seeks the discovery of the globally optimal solution to problems characterized by high complexity, large dimensionality, and strong nonlinearity. In contrast to local methods, which identify solutions within limited regions of the search space, global optimization techniques aim to explore the entire search domain, thus providing solutions of validated quality. During the recent years, a magnitude of reviews for this problem have been published by various researchers [1–3]. Their significance is not merely theoretical; they find direct application in critical scientific disciplines such as mathematics [4,5], physics [6,7], chemistry [8,9], medicine [10,11] and economics [12,13] where the requirements for accuracy and reliability are particularly demanding.

In this research area, two main approaches have been established: deterministic [14,15] and stochastic methods [16,17]. Deterministic techniques are grounded in rigorous mathematical foundations and provide theoretical guarantees for locating the global minimum. A characteristic example is interval analysis methods, which pursue the gradual subdivision of the search space until regions of increased interest are identified. Despite their accuracy, their high computational cost restricts their applicability to relatively small-scale problems. In contrast, stochastic methods, although lacking a mathematical guarantee of finding the optimal solution, demonstrate considerable flexibility in highly complex environments. Representative examples include Controlled Random Search [18,19], Simulated Annealing [20,21], and Multistart [22,23], which, however, encounter difficulties when strong nonlinear constraints are involved.

Within the framework of stochastic approaches, particular importance is attributed to evolutionary computation, which draws inspiration from the principles of biological evolution and natural selection. Its central idea is that the mechanisms of adaptation and survival observed in nature can be modeled computationally, providing powerful tools for solving large-scale and highly complex problems. This category includes Genetic Algorithms [24,25], Evolution Strategies [26,27], Genetic Programming [28,29], and Differential Evolution (DE) [30,31]. At the same time, related stochastic methods include approaches inspired by the collective behavior of biological systems, such as Particle Swarm Optimization (PSO) [32,33] and Ant Colony Optimization (ACO)[34,35].

In recent years, research has increasingly focused on more specialized bio-inspired algorithms, which attempt to capture even more distinctive behaviors observed in the natural world. A representative example is the Magnificent Frigatebird Optimization (MFO) algorithm [36], which is based on the study of the frigatebird, a large seabird found in tropical and subtropical regions of the Americas. This species is notable for its kleptoparasitic foraging strategy: rather than hunting independently, it attacks other seabirds, forces them to drop their prey, and then swiftly dives to seize it before it sinks into the sea. This adaptive and highly effective behavior has served as the basis for the mathematical design of MFO, in which the notion of "attacking competitors to acquire resources" is translated into mechanisms of exploration and exploitation within the solution space. In this way, MFO is incorporated into the rapidly expanding family of bio-inspired algorithms, enriching the field of global optimization with innovative strategies.

The current work introduces a series of modifications to the MFO algorithm, in order to speed up the process and increase the efficiency of the algorithm. These modifications include:

- **Three movement strategies:**
  - The aggressive strategy focuses on intense movement towards better candidates, enhancing the exploration ability.
  - The conservative strategy is oriented in smaller steps around the best solution so far, promoting exploitation.

  – The mixed strategy combines the characteristics of the two previous ones, ensuring a more balanced behavior of the algorithm.
- Furthermore, **local search** was incorporated through the BFGS algorithm[37] with the aim of refining candidate solutions and improving convergence towards optimal points.
- Finally, an early **termination technique**[38] based on stochastic observations was applied. At each iteration of the algorithm, the optimal value of the objective function up to that moment is recorded. If this value remains unchanged for a predetermined number of iterations, the optimization process is stopped. In this way, unnecessary execution of further calculations is avoided when no substantial progress is observed, leading to a more efficient use of computational time.

The remains of this paper are divided as follows: in section2 the proposed MFO algorithm, a flowchart with a detailed description is presented, in section3 of the test functions used in the experiments as well as the related experiments are presented. In the 4section, there is a brief discussion of the results obtained from the experiments. In section 5some conclusions and directions for future improvements are discussed.

## 2. Materials and Methods

### *2.1. The main steps of the algorithm*

The Magnificent Frigatebird (MFO) optimization algorithm is a metaheuristic algorithm inspired by biology. It combines the cooperative foraging behavior of magnificent frigatebirds in a marine environment. Each agent represents a frigate that explores the search space looking for better food sources, i.e. the best solutions to the optimization problem. The algorithm starts by initializing a population of candidate solutions and evaluating their performance. The best individual is then recorded.

During each generation, the agents are evaluated, and a subset may undergo local search to refine their positions and improve accuracy before the population is ranked according to fitness. The search process alternates between two phases: the exploration phase, where agents perform adaptive movements to discover new and promising regions of the search space, and the exploitation phase, where they focus on refining the best solutions found so far to enhance precision and convergence. These two behaviors are expressed through **aggressive**, **conservative**, or **mixed strategies**, allowing a dynamic balance between global exploration and local intensification throughout the optimization process.

At the end of each generation, the best solution can be further improved through a local optimization step to enhance convergence accuracy. The process continues until a termination criterion, such as the maximum number of generations or population convergence, is met. The final result represents the best solution found. In conclusion, the MFO algorithm achieves a strong balance between exploration and exploitation through adaptive tuning and selective improvement, providing satisfactory performance on a range of different optimization problems. The steps of the Magnificent Frigatebird Optimization (MFO) algorithm presented in Algorithm 1.

Furthermore, the steps of the proposed method are also graphically shown in Figure 1.

### *2.2. The used termination rule*

The used termination rule is provided in the work of Charilogis and Tsoulos [38] and it has the following steps:

- At every iteration $t$, the difference between the current best value $f_{min}^{(t)}$ and the previous best value $f_{min}^{(t-1)}$ is computed:

**Algorithm 1** Magnificent Frigatebird Optimization (MFO) algorithm

**INPUT**
- $N$: population size.
- $G_{\max}$: max generations.
- $T$: inner iterations per generation.
- strategyMode$\in${aggresive, conservative,mixed} .
- $p_l \in [0, 1]$ as the local search rate.

**OUTPUT**
- $(x_{\text{best}}, f_{\text{best}})$

**INITIALIZATION**
- **Sample** initial population $P$ of size $N$ uniformly in $S$ and evaluate $f$
- **Obtain** the current best $(x_{\text{best}}, f_{\text{best}})$ from $P$
- **Set** $k = 0$, the generation counter.

**MFO main pseudocode**

**while** $k < G_{\max}$ and the termination rule described in subsection 2.2 does not hold **do**
    **Set** $k = k + 1$
    **for** $i = 1, \ldots, N$ **do**
        $f_i = f(P_i)$
        Pick a random number $r \in [0, 1]$.
        **If** $r \le p_l$ **then** $f_i = \text{LS}(P_i)$, Where $\text{LS}(x)$ is a local optimization procedure.
    **endfor**
    **Obtain** the current best $(x_{\text{best}}, f_{\text{best}})$ from $P$
    sort $P$ according to the fitness values.
    **for** $t = 1, \ldots, T$ **do**
        **for** $i = 1, \ldots, N$ **do**
            // Phase A: move using better "seabird" or best, depending on strategyMode
            **Set** $C = \{P_k : k \neq i \text{ and } f(P_k) < f(P_i)\}$
            **Pick** $SS$ a random element of $C$
            **if** strategyMode = aggresive **then**
                **for** $j = 1, \ldots, n$ **do**
                    **Pick** a random number $r \in [0, 1]$.
                    **Set** $I \in \{1, 2\}$ a random integer value.
                    **Set** $y_j = P_{i,j} + 2(r - 0.5)(SS_j - IP_{i,j})$
                **endfor**
            **else if** strategyMode = conservative **then**
                **for** $j = 1, \ldots, n$ **do**
                    **Set** $r \in [0, 1]$ a randomly selected number.
                    **Set** $y_j = P_{i,j} + 0.3(r - 0.5)(x_{best,j} - P_{i,j})$
                **endfor**
            **else** // mixed strategy
                **Select** with probability 50% the aggressive or the conservative strategy.
            **endif**
            **if** $y \in S$ and $f(y) < f(P_i)$ **then**
                **Set** $P_i = y$
            **endif**
        **endif**
        // Phase B (dive): pull towards current best solution
        **Set** $b = \arg\min_k f(P_k)$ // bestSolution in population after Phase A
        **for** $j = 1, \ldots, n$ **do**
            **Set** $r \in [0, 1]$ a randomly selected number
            Set $z_j = P_{i,j} + (1 - 2r)\left(b_j - \frac{P_{i,j}}{t}\right)$
        **if** $z \in S$ and $f(z) < f_i$ **then set** $P_i = z$
        **endfor**
    **endfor**
    **Set** $f_{\text{best}} = \text{LS}(x_{\text{best}})$
**endwhile**
**return** $(x_{\text{best}}, f_{\text{best}})$


**Figure 1.** The steps of the proposed method

$$\delta^{(t)} = \left| f_{\min}^{(k)} - f_{\min}^{(k-1)} \right| \tag{2}$$

- The algorithm terminates when $\delta^{(t)} \leq \epsilon$ for series of predefined consecutive iterations $N_k$, where $\epsilon$ is a small positive number, for example $10^{-6}$.

### *2.3. Movement Strategies*

The proposed optimization algorithm employs three distinct movement strategies that regulate the behavioral dynamics of seabirds during the search for optimal solutions. Each strategy defines a different balance between exploration of the search space and exploitation of promising regions already identified.

The aggressive strategy is characterized by strong stochastic dynamics and a tendency toward rapid discovery of new, potentially better regions in the search space. Individuals are driven toward other, more successful members of the population, performing large displacements that enhance diversity and dispersion. This approach promotes exploration and enables the algorithm to escape local optima. However, it also increases the risk of generating infeasible or inefficient solutions if appropriate boundary control mechanisms are not applied.

The conservative strategy focuses on exploitation of the best-known solution, promoting smoother and more stable movements. Individuals move gradually toward the current global best position with limited randomness, aiming to refine the accuracy of their position over time. This strategy reduces population diversity and enhances convergence.

The mixed strategy acts as an adaptive balance mechanism between the two aforementioned approaches. At each iteration, the algorithm randomly decides whether to apply the aggressive or conservative movement pattern, allowing dynamic transitions between exploration and exploitation phases. This flexibility enables the search process to adjust to the evolving characteristics of the optimization landscape, preventing premature convergence and maintaining adaptive search capability.

The integration of these three strategies ensures that the algorithm maintains both exploratory power, essential for discovering new regions, and exploitative stability, crucial for fine-tuning solutions. In conclusion, their combined action enhances the robustness and efficiency of the method.

## 3. Results

This section begins with a description of the functions that will be used in the experiments and then presents in detail the experiments that were performed, in which the parameters available in the proposed algorithm were studied, in order to study its reliability and adequacy.

### *3.1. Test Functions*

A variety of test functions were used in the conducted experiments[39–44]. These functions are used in a series of research papers. The description of each used test function is provided below. In all cases, the constant $n$ defines the dimension of the objective function.

### *3.2. Experimental results*

A series of experiments was carried out for the previously mentioned functions and these experiments were executed on an AMD RYZEN 5950X with 128GB RAM.

| NAME | FORMULA | DIM | $G_{min}$ |
|---|---|---|---|
| **ACKLEY** | $f(x) = -a \exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1)$<br>$a = 20.0, b = 0.2$ | 2 | 4.440892099e-16 |
| **BF1** | $f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$ | 2 | 0 |
| **BF2** | $f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$ | 2 | 0 |
| **BF3** | $f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1 + 4\pi x_2) + \frac{3}{10}$ | 2 | 0 |
| **BRANIN** | $f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$<br>$-5 \le x_1 \le 10,\ 0 \le x_2 \le 15$ | 2 | 0.3978873577 |
| **CAMEL** | $f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4, \quad x \in [-5,5]^2$ | 2 | -1.031628453 |
| **DIFFERENT POWERS** | $f(\mathbf{x}) = \sqrt{\sum_{i=1}^{n} \lvert x_i\rvert^{2+4\frac{i-1}{n-1}}}$ | n | 0 |
| **DIFFPOWER** | $f(x) = \sum_{i=1}^{n} \lvert x_i - y_i\rvert^p \ \ n = 2\ p = 2,5,10$ | n | 0 |
| **DISCUS** | $f(x) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$ | n | 0 |
| **EASOM** | $f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2-\pi)^2 - (x_1-\pi)^2\right)$ | 2 | -1 |
| **ELLIPSOIDAL** | $f(x) = \sum_{i=1}^{n}\left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$ | n | 0 |
| **EQUAL MAXIMA** | $f(x) = \sin^6(5\pi x)\cdot e^{-2\log(2)\cdot\left(\frac{x-0.1}{0.8}\right)^2}$ | n | 0 |
| **EXP** | $f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \le x_i \le 1$ | n | -1 |
| **GKLS** | $f(x) = \mathrm{Gkls}(x,n,w)\ \ n = 2,3\ w = 50,100$ | n | -1 |
| **GOLDSTAIN** | $f(x) = [1 + (x_1 + x_{2+1}))^2(19 - 14x_1 + 3x_1^2 14x_2 + 6x_1x_2 + 3x_2^2)]$<br>$[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$ | 2 | 3 |
| **GRIEWANK ROSENBROCK** | $f(\mathbf{x}) = \underbrace{\left(\frac{\lVert\mathbf{x}\rVert^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1\right)}_{\text{Griewank}} \cdot \underbrace{\left(\frac{1}{10}\sum_{i=1}^{n-1}\left[100(x_{i+1} - x_i^2)^2 + (1-x_i)^2\right]\right)}_{\text{Rosenbrock}}$ | n | 0 |
| **GRIEWANK** | $f(x) = 1 + \frac{1}{200}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\frac{\cos(x_i)}{\sqrt{(i)}}$ | n | 0 |
| **HANSEN** | $f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$ | 2 | -176.5417931 |
| **HARTMAN3** | $f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$ | 3 | -3.862782148 |
| **HARTAMAN6** | $f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$ | 6 | -3.22368011 |
| **KATSUURA** | $f(\mathbf{x}) = \frac{10}{n^2}\prod_{i=1}^{n}\left(1 + i\sum_{k=1}^{32}\frac{\lvert 2^k x_i - \lfloor 2^k x_i\rfloor\rvert}{2^k}\right)^{\frac{10}{n^{1.2}}} - \frac{10}{n^2} + \frac{1}{Dn}\sum_{i=1}^{n} x_i^2$ | n | 0 |
| **MICHALEWICZ** | $f(\mathbf{x}) = -\sum_{i=1}^{n}\sin(x_i)\cdot\sin^{2m}\left(\frac{i\cdot x_i^2}{\pi}\right)$ | 2,<br>5,<br>10 | -1.8013<br>-4.6877<br>-9.6602 |
| **POTENTIAL** | $V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6\right]\ \ n = 9,15,21,30$ | 5,<br>6,<br>10 | -9.103852416<br>-12.71206226<br>-28.42253189 |
| **RARSTIGIN2** | $f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$ | 2 | -2 |
| **ROSENBROCK** | $f(x) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \le x_i \le 30$ | n | 0 |
| **SHARP RIDGE** | $f(\mathbf{x}) = x_1^2 + \alpha\sum_{i=2}^{n} x_i^2,\ a > 1$ | n | 0 |
| **SHEKEL5** | $f(x) = -\sum_{i=1}^{5}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$ | 4 | -10.10774912 |
| **SHEKEL7** | $f(x) = -\sum_{i=1}^{7}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$ | 4 | -10.342377774 |
| **SHEKEL10** | $f(x) = -\sum_{i=1}^{10}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$ | 4 | -10.53640982 |
| **SINUSOIDAL** | $f(x) = -\left(2.5\prod_{i=1}^{n}\sin(x_i - z) + \prod_{i=1}^{n}\sin(5(x_i - z))\right), \quad 0 \le x_i \le \pi$ | n | -3.5 |
| **SPHERE** | $f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$ | n | 0 |
| **STEP ELLIPSOIDAL** | $f(\mathbf{x}) = \sum_{i=1}^{n}\lfloor x_i + 0.5\rfloor^2 + \alpha\sum_{i=1}^{n}\left(10^6\cdot\frac{i-1}{n-1}\right)x_i^2,\ a = 1$ | n | 0 |
| **TEST2N** | $f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i$ | 4,<br>5 | -156.6646628<br>-195.8308285 |
| **TEST30N** | $f(x) = \frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i - 1)^2\left(1 + \sin^2\left(3\pi x_{i+1}\right)\right)\right) +$<br>$+(x_n - 1)^2\left(1 + \sin^2(2\pi x_n)\right)$ | n | 0 |
| **ZAKHAROV** | $f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n}\frac{i}{2}x_i\right)^2 + \left(\sum_{i=1}^{n}\frac{i}{2}x_i\right)^4$ | n | 0 |

The operating system of the running machine was Debian Linux. Each experiment was conducted 30 times, with different random numbers each time, and the averages were recorded. The software used in the experiments was coded in ANSI C++ using the freely available optimization environment of OPTIMUS, which can be downloaded from https://github.com/itsoulos/OPTIMUS. The values for the experimental parameters used in the proposed method are outlined in Table 1.

**Table 1.** The values of the parameters of the proposed method.

| PARAMETER | MEANING | VALUE |
|---|---|---|
| N | Number of agents | 200 |
| $G_{max}$ | Maximum number of allowed iterations | 200 |
| $p_l$ | Local search rate | 0.05 |
| T | Number of mfo iterations | 2 |

In the following tables that depict the experimental results, the numbers in cells stand for the average function calls, as measured on 30 independent runs. The numbers in parentheses denote the fraction of the executions where the method discovered successfully the global minimum. If this number is not present, then the method managed to locate the global minimum in every run ( 100% success).

### *3.3. The proposed method in comparison with others*

The comparison of results for nine optimization methods (PROPOSED, GENETIC[25], WOA[45], MEWOA[46], DE[30], IPSO[47], AOA[48], SAO[49] and ACO[34]) is based on a diverse set of benchmark functions. The evaluation focused on two key performance indicators: the total number of objective function evaluations and the corresponding success rates, as shown in Table 2.

The analysis of the total function evaluations reveals substantial differences among the tested algorithms. The GENETIC algorithm required 316,189 evaluations, while WOA recorded the highest computational cost with 923,113 evaluations. The MEWOA, DE, IPSO, AOA, SAO, and ACO methods required 489,691, 326,083, 314,060, 540,139, 305,505, and 260,753 evaluations, respectively. In contrast, the PROPOSED algorithm achieved the lowest total number of function evaluations, requiring only 204,488 calls in total, demonstrating exceptional computational efficiency.

This superiority was true for almost all reference functions. For example, on the Ackley function, the PROPOSED method reached convergence in only 4971 evaluations, while GENETIC and WOA required 5811 and 24,766 evaluations, respectively. Similarly, for the Branin function, the PROPOSED algorithm required 2397 evaluations, much fewer than GENETIC's 2648 and AOA's 4777 for this function.

In addition to the reduced computational cost, the PROPOSED algorithm maintained a 100% success rate for most benchmark functions, confirming its reliability. Also, the DE, IPSO and SAO algorithms also produced satisfactory results, but with higher computational cost. In contrast, the ACO algorithm presented lower consistency, with success rates below unity (e.g., 0.57 for BF1 and 0.70 for BF2).

The results show that the PROPOSED algorithm is very efficient and reliable. It manages to give equally good or even better results than other known algorithms, but with much fewer calculations.

Figure 2 summarizes the pairwise significance between the PROPOSED method and the baseline optimizers. Using the conventional star notation (ns: $p > 0.05$, *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$, ****: $p < 0.0001$), all comparisons except ACO reach ****, indicating very-extremely significant differences. Specifically, PROPOSED differs from GENETIC, WOA, MEWOA, DE, IPSO, AOA and SAO at $p < 0.0001$, whereas the comparison with

**Table 2.** Experimental results using different optimization methods. Numbers in cells represent sum function calls.

| FUNCTION | PROPOSED | GENETIC | WOA | MEWOA | DE | IPSO | AOA | SAO | ACO |
|---|---|---|---|---|---|---|---|---|---|
| ACKLEY | 4971 | 5811 | 24766 | 15257 | 6425 | 7355 | 9590 | 5904 | 6112 |
| BF1 | 3449 | 4279 | 9924 | 8423 | 4272 | 4121 | 8356 | 4694 | 4120(0.57) |
| BF2 | 3255 | 3903 | 9597 | 7982 | 3863 | 3767 | 6987 | 4166 | 3613(0.70) |
| BF3 | 3028 | 3494 | 20117 | 7087 | 3475 | 3305 | 7029 | 3706 | 3563(0.77) |
| BRANIN | 2397 | 2648 | 5939 | 4170 | 2543 | 2522 | 4777 | 2583 | 2814 |
| CAMEL | 2757 | 3104 | 5917 | 5854 | 3054 | 2942 | 5050 | 3089 | 2421 |
| DIFFERENT POWERS10 | 6651 | 10954 | 23650 | 16588 | 10542 | 9807 | 27533 | 11234 | 10471 |
| DIFFPOWER2 | 5614 | 9053 | 13476 | 14248 | 8510 | 8121 | 14367 | 9139 | 10540 |
| DIFFPOWER5 | 13836 | 24466 | 51816 | 40313 | 25515 | 27348 | 58890 | 27918 | 32583 |
| DIFFPOWER10 | 19486 | 37664 | 73782 | 56980 | 40402 | 34923 | 74297 | 39429 | 16664 |
| DISCUS10 | 2185 | 2711 | 6476 | 4084 | 2663 | 2540 | 6302 | 2716 | 2010 |
| EASOM | 2033 | 2062 | 3153 | 3194 | 1953 | 1998 | 2691 | 2167 | 1995 |
| ELLIPSOIDAL10 | 2479 | 3309 | 8979 | 5057 | 3339 | 3540 | 6308 | 2507 | 2146 |
| EQUALMAXIMA10 | 8968 | 16399 | 63542 | 26257 | 17891 | 17231 | 20536 | 17014 | 18343 |
| EXPONENTIAL10 | 2768 | 3854 | 8408 | 6224 | 3928 | 3298 | 6216 | 3052 | 3941 |
| GKLS250 | 2701 | 2579 | 4978 | 4481 | 2412 | 2580 | 3220 | 1789 | 2238 |
| GKLS350 | 3054 | 3076 | 7226(0.96) | 4117 | 2860 | 3016 | 3314 | 1634(0.86) | 2110(0.90) |
| GOLDSTEIN | 3195 | 3802 | 9016 | 6598 | 3736 | 3856 | 6508 | 3522 | 4239 |
| GRIEWANK2 | 4289(0.80) | 4313(0.73) | 7948(0.87) | 6613(0.96) | 4689(0.80) | 4516 | 6339(0.73) | 4881(0.73) | 3624(0.43) |
| GRIEWANK10 | 5374 | 8381 | 51143 | 14380 | 8888 | 8087 | 16979 | 8788 | 5679 |
| GRIEWANK ROSENBROCK10 | 5505 | 9576 | 16592 | 13856 | 9923 | 7675 | 17936 | 9157 | 3694 |
| HANSEN | 3531 | 3345 | 10498 | 5072 | 3189 | 3546 | 4166 | 3245 | 3042(0.70) |
| HARTMAN3 | 2633 | 3142 | 8440 | 5095 | 3177 | 3103 | 4034 | 2539 | 3297 |
| HARTMAN6 | 2874 | 3941 | 22615 | 6214 | 3870 | 3688 | 5468 | 3019 | 3949(0.77) |
| KATSUURA10 | 8648(0.03) | 19539(0.03) | 61156(0.33) | 25791(0.03) | 22199(0.03) | 16452(0.03) | 24172(0.03) | 18650(0.03) | 24341(0.03) |
| MICHALEWICZ4 | 3757(0.63) | 5906(0.90) | 14045(0.87) | 7419(0.97) | 5186(0.97) | 4583(0.93) | 5746(0.93 | 4176(0.80) | 4213(0.17) |
| POTENTIAL5 | 4944 | 8166 | 39455 | 12218 | 8525 | 10128 | 10883 | 7879 | 7456 |
| POTENTIAL6 | 5887(0.37) | 12034(0.80) | 46466(0.77) | 13728(0.77) | 12073(0.73) | 13873(0.47) | 11948(0.67) | 9564(0.70) | 8212(0.13) |
| POTENTIAL10 | 8599(0.93) | 15417 | 59487 | 23210 | 16382 | 21690(0.97) | 18199(0.96) | 14463 | 9235(0.27) |
| RASTRIGIN | 3639 | 3840(0.93) | 8116 | 6127 | 4287 | 3956 | 5052 | 2813(0.93) | 2676(0.33) |
| ROSENBROCK8 | 4964 | 8085 | 14094 | 12282 | 8245 | 7843 | 16423 | 8448 | 2893 |
| ROSENBROCK16 | 6467 | 11469 | 23544 | 17851 | 11999 | 11450 | 25092 | 12016 | 3604 |
| SHARPRIDGE10 | 6006 | 10297 | 18391 | 15280 | 10924 | 10481 | 22452 | 10013 | 3594 |
| SHEKEL5 | 3230 | 4540 | 17010 | 6551 | 4361 | 3986 | 7615 | 4271 | 4492(0.63) |
| SHEKEL7 | 3212 | 4517 | 17873 | 6985 | 4310 | 4009 | 7491 | 4245 | 4719(0.67) |
| SHEKEL10 | 3168 | 4266 | 15524 | 6750 | 4217 | 4010 | 7074 | 4111 | 4442(0.47) |
| SPHERE10 | 1660 | 1706 | 6389 | 2927 | 1653 | 1636 | 5128 | 906 | 1843 |
| STEPE LLIPSOIDAL4 | 3018 | 2842(0.83) | 2129 | 2237(0.93) | 2385(0.73) | 1879 | 2701(0.43) | 1262(0.07) | 1583(0.10) |
| SINUSOIDAL8 | 3272 | 4572 | 20782 | 7032 | 4519 | 3995 | 7249 | 3901 | 4023(0.83) |
| SINUSOIDAL16 | 4454(0.93) | 6588 | 34409 | 9368 | 7349 | 4697 | 10405 | 6491 | 4669(0.63) |
| TEST2N4 | 3236 | 3878 | 12279 | 6019 | 3699 | 3611 | 5765 | 3251 | 3504(0.56) |
| TEST2N5 | 3639(0.97) | 4109(0.86) | 15351(0.83) | 6178 | 4248(0.96) | 3977(0.93) | 6647(0.93) | 3569(0.97) | 3718(0.30) |
| TEST30N10 | 3134 | 5238 | 16970 | 8809 | 5206 | 6055 | 7915 | 4915 | 6203 |
| ZAKHAROV10 | 2521 | 3314 | 11645 | 4785 | 3197 | 2864 | 5289 | 2669 | 2125 |
| **SUM** | **204488(0.95)** | **316189(0.96)** | **923113(0.97)** | **489691(0.97)** | **326083(0.96)** | **314060(0.96)** | **540139(0.95)** | **305505(0.93)** | **260753(0.77)** |

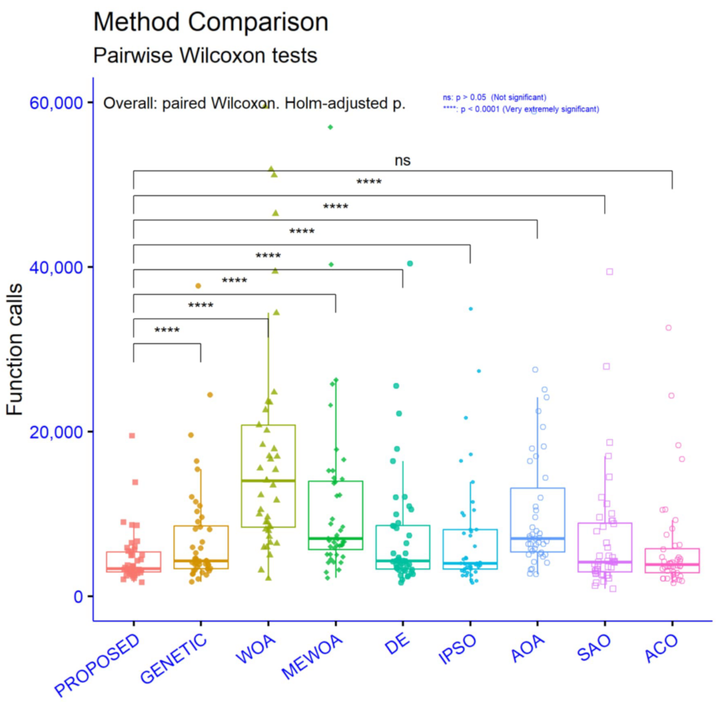


**Figure 2.** A statistical comparison of the proposed with other optimization methods

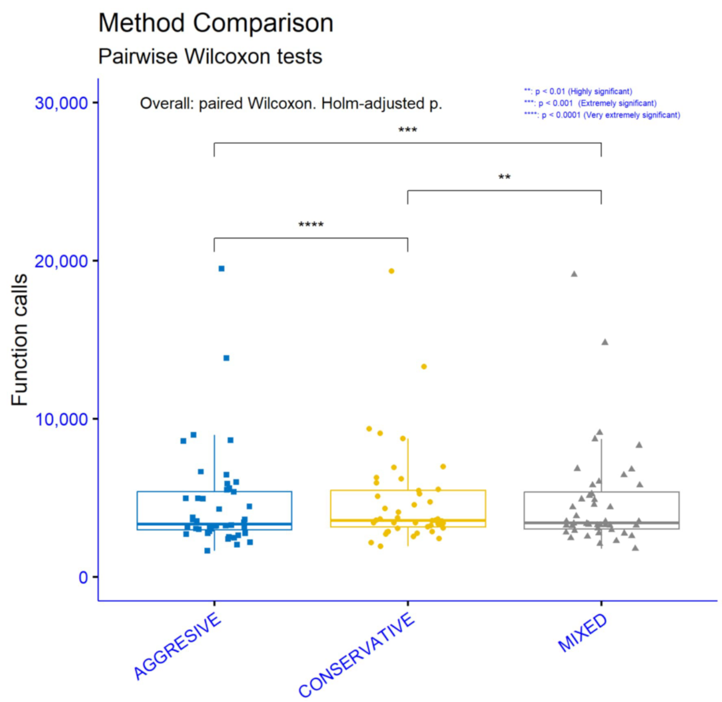


**Figure 3.** A statistical comparison of the proposed method with different movement strategies

ACO is not statistically significant ($p > 0.05$). In conjunction with the reported performance metrics, these results indicate that the advantage of PROPOSED over most baselines is highly unlikely to be due to random variation.

### 3.4. Comparative analysis of different strategic variations

To further evaluate the internal adaptability of the proposed optimization framework, three distinct strategic configurations: Aggressive, Conservative, and Mixed were examined across a comprehensive set of benchmark functions. The total number of objective function evaluations recorded for each configuration was 204,488 for the aggressive strategy, 212,680 for the conservative strategy, and 209,767 for the mixed strategy. Among these, the aggressive configuration demonstrated the lowest computational demand, indicating faster convergence behavior.

For example, the aggressive strategy in the Easom benchmark function required 2033 evaluations and Potential5 required 4944 evaluations. While in the conservative strategy Easom required 2178 evaluations and Potential5 required 5,108 evaluations. On the other hand, in the mixed strategy it required 2100 and 4914 evaluations in the corresponding functions.

In terms of reliability, all three configurations demonstrated excellent success rates, maintaining near-perfect consistency across most benchmark functions. Specifically, the success rates were 95% for the aggressive strategy, 93% for the conservative strategy, and 94% for the mixed configuration. The algorithm's stable performance highlights its robustness, as it stays reliable under different search strategies. Minor differences mainly affect speed and computational effort.

Overall, the results show that the proposed framework is flexible and adaptable, allowing its settings to adjust to different problems. The Conservative version offers a good balance between accuracy and speed, the Aggressive one works best on complex problems by avoiding early convergence, and the Mixed approach provides a stable middle ground. This adaptability makes the framework suitable for many optimization tasks.

**Table 3.** Experiments using different strategy for the proposed method.

| FUNCTION | AGGRESSIVE | CONSERVATIVE | MIXED |
|---|---|---|---|
| ACKLEY | 4971 | 4749 | 4881 |
| BF1 | 3449 | 3541 | 3520 |
| BF2 | 3255 | 3355 | 3291 |
| BF3 | 3028 | 3090 | 3076 |
| BRANIN | 2397 | 2548 | 2460 |
| CAMEL | 2757 | 2854 | 2815 |
| DIFFERENTPOWERS10 | 6651 | 6978 | 6803 |
| DIFFPOWER2 | 5614 | 5451 | 5792 |
| DIFFPOWER5 | 13836 | 13301 | 14807 |
| DIFFPOWER10 | 19486 | 19332 | 19118 |
| DISCUS10 | 2185 | 2424 | 2274 |
| EASOM | 2033 | 2178 | 2100 |
| ELLIPSOIDAL10 | 2479 | 2717 | 2566 |
| EQUALMAXIMA10 | 8968 | 9069 | 8717 |
| EXPONENTIAL10 | 2768 | 3065 | 2930 |
| GKLS250 | 2701 | 2833 | 2769 |
| GKLS350 | 3054 | 3583 | 3260 |
| GOLDSTEIN | 3195 | 3243 | 3244 |
| GRIEWANK2 | 4289(0.80) | 4319(0.67) | 4407(0.73) |
| GRIEWANK10 | 5374 | 5546 | 5249(0.97) |
| GRIEWANKROSENBROCK10 | 5505 | 5957 | 5806 |
| HANSEN | 3531 | 3557(0.80) | 3288(0.93) |
| HARTMAN3 | 2633 | 2850 | 2755 |
| HARTMAN6 | 2874 | 3182 | 2971 |
| KATSUURA10 | 8648(0.03) | 9366(0.03) | 9109(0.03) |
| MICHALEWICZ4 | 3757(0.63) | 4099(0.60) | 4404(0.60) |
| POTENTIAL5 | 4944 | 5108 | 4914 |
| POTENTIAL6 | 5887(0.37) | 6215(0.50) | 6451(0.56) |
| POTENTIAL10 | 8599(0.93) | 8745(0.93) | 8305(0.93) |
| RASTRIGIN | 3639 | 3660(0.93) | 3848 |
| ROSENBROCK8 | 4964 | 5256 | 5137 |
| ROSENBROCK16 | 6467 | 6929 | 6837 |
| SHARPRIDGE10 | 6006 | 6287 | 6024 |
| SHEKEL5 | 3230 | 3436 | 3317 |
| SHEKEL7 | 3212 | 3405 | 3288 |
| SHEKEL10 | 3168 | 3360 | 3270 |
| SPHERE10 | 1660 | 1927 | 1791 |
| STEPELLIPSOIDAL4 | 3018 | 3435(0.93) | 3245 |
| SINUSOIDAL8 | 3272 | 3665 | 3499 |
| SINUSOIDAL16 | 4454(0.93) | 4561(0.93) | 4576(0.93) |
| TEST2N4 | 3236 | 3434(0.93) | 3371(0.97) |
| TEST2N5 | 3639(0.97) | 3731(0.80) | 3511(0.77) |
| TEST30N10 | 3134 | 3590 | 3380 |
| ZAKHAROV10 | 2521 | 2749 | 2591 |
| **SUM** | **204488(0.95)** | **212680(0.93)** | **209767(0.94)** |

In Figure 3 we report pairwise significance for the exploration-exploitation control modes of the proposed optimizer. AGGRESSIVE vs CONSERVATIVE reaches **** ($p < 0.0001$), indicating a very-extremely significant difference. CONSERVATIVE vs MIXED is ** ($p < 0.01$), showing a highly significant separation. MIXED vs AGGRESSIVE is *** ($p < 0.001$), reflecting an extremely significant but comparatively smaller gap. Overall, the regimes induce materially different behaviors, with MIXED lying closer to AGGRESSIVE than to CONSERVATIVE.

### *3.5. Impact of Population Size on Algorithmic Performance*

To assess the influence of population size on convergence behavior and computational efficiency, both the proposed optimization algorithm and the Genetic Algorithm (GA) were evaluated under four different population configurations: 50, 100, 200, and 500. The experiments were conducted using a representative suite of benchmark functions.

In a population size of 50 individuals, both algorithms demonstrated rapid convergence with low computational cost. The proposed algorithm required only 1983 evaluations for the Ackley function, while BF1 required 1075 and BF2 required 983. The Genetic Algorithm converged after 2148 evaluations for Ackley, while 1258 for BF1 and 1105 for BF2. Despite their computational efficiency, both methods showed a slight decline in success rate in certain benchmark functions due to the limited population diversity. For example, the proposed algorithm achieved success rates around 0.90 for BF2 and BF3, while the GA achieved 97% for Ackley and 30% for Griewank2.

When the population size was increased to 100, both algorithms reached their most balanced performance. The proposed algorithm achieved a 100% success rate on almost all test functions with 3053 evaluations for Ackley, 1862 for BF1, 1734 evaluations for BF2 and 1211 for Branin. The Genetic Algorithm achieved comparable results, also maintaining a 100% success rate, with 3331 evaluations for Ackley, 2287 for BF1, and 1353 for Branin. This configuration provided sufficient population diversity for effective exploration without incurring significant computational overhead. As a result, a medium-sized population offered the optimal trade-off between exploration and exploitation, ensuring both robustness and computational efficiency.

Further increasing the population to 200 led to a significant increase in computational cost for both algorithms. The proposed algorithm required 4971 evaluations for Ackley while 3449 evaluations for BF1 and 3255 for BF2. The GA reached 5811 evaluations for Ackley and 3494 for BF3. Although the success rate slightly improved, the additional computational effort did not yield proportional benefits in optimization performance.

When the population size was further increased to 500, this pattern became even more apparent. The proposed algorithm required up to 10,690 evaluations for Ackley while 8346 for the BF1 function and 7825 for the BF2 function. The GA needed 13,664 evaluations for Ackley, 10,866 for BF1, and 6716 for Branin. While both algorithms maintained high success rates, the computational effort increased drastically, indicating a prolonged and redundant exploratory phase that hindered convergence speed and overall efficiency.

The comparative analysis reveals a consistent trend for both algorithms: population size has a direct and significant impact on convergence dynamics and computational efficiency. Small populations favor faster convergence but risk premature stagnation due to limited diversity. Very large populations improve reliability but impose excessive computational cost. Medium-sized populations achieve the best balance between speed, accuracy, and robustness.

Overall, the proposed algorithm consistently outperformed the Genetic Algorithm at all population levels, requiring fewer evaluations to reach convergence while maintaining equal or higher success rates. This improvement is attributed to its adaptive parameter

| FUNCTION | N=50 | GA(N=50) | N=100 | GA(N=100) | N=200 | GA(N=200) | N=500 | GA(N=500) |
|---|---|---|---|---|---|---|---|---|
| ACKLEY | 1983 | 2148(0.97) | 3053 | 3331 | 4971 | 5811 | 10690 | 13664 |
| BF1 | 1075(0.97) | 1258 | 1862 | 2287 | 3449 | 4279 | 8346 | 10866 |
| BF2 | 983(0.90) | 1105 | 1734 | 2037 | 3255 | 3903 | 7825 | 9791 |
| BF3 | 885(0.90) | 1015 | 1612 | 1827 | 3028 | 3494 | 7280 | 8797 |
| BRANIN | 627 | 689 | 1211 | 1353 | 2397 | 2648 | 5953 | 6716 |
| CAMEL | 813 | 863 | 1445 | 1638 | 2757 | 3104 | 6820 | 7910 |
| DIFFERENT POWERS10 | 2108 | 2711 | 3534 | 5334 | 6651 | 10954 | 15520 | 25428 |
| DIFFPOWER2 | 1519 | 2427 | 2888 | 4700 | 5614 | 9053 | 13599 | 22630 |
| DIFFPOWER5 | 3679 | 6279 | 6454 | 12695 | 13836 | 24466 | 39269 | 66396 |
| DIFFPOWER10 | 4908 | 9505 | 9758 | 19182 | 19486 | 37664 | 47689 | 91211 |
| DISCUS10 | 562 | 675 | 1089 | 1364 | 2185 | 2711 | 5435 | 6632 |
| EASOM | 533 | 528 | 1029 | 1041 | 2033 | 2062 | 5084 | 5188 |
| ELLIPSOIDAL10 | 641 | 820 | 1239 | 1662 | 2479 | 3309 | 6170 | 8099 |
| EQUALMAXIMA10 | 2272 | 4025 | 4373 | 8329 | 8968 | 16399 | 22053 | 39624 |
| EXPONENTIAL10 | 768 | 958 | 1399 | 1939 | 2768 | 3854 | 6832 | 9420 |
| GKLS250 | 861 | 751 | 1410 | 1328 | 2701 | 2579 | 5999 | 5995 |
| GKLS350 | 1068(0.97) | 888(0.97) | 1836 | 1663 | 3054 | 3076 | 6478 | 5938 |
| GOLDSTEIN | 932 | 1086 | 1623 | 1979 | 3195 | 3802 | 7762 | 9688 |
| GRIEWANK2 | 1509(0.67) | 1378(0.30) | 3092(0.73) | 2557(0.57) | 4289(0.80) | 4313(0.73) | 10040(0.90) | 10882(0.93) |
| GRIEWANK10 | 1806(0.83) | 2610 | 3139(0.97) | 4729 | 5374 | 8381 | 12575 | 19950 |
| GRIEWANK ROSENBROCK10 | 1467 | 2334 | 2730 | 4745 | 5505 | 9576 | 13676 | 23160 |
| HANSEN | 1048(0.87) | 1156(0.83) | 1876(0.93) | 2007(0.90) | 3531 | 3345 | 7224 | 8015 |
| HARTMAN3 | 753 | 838 | 1327 | 1652 | 2633 | 3142 | 6598 | 8008 |
| HARTMAN6 | 790 | 1012 | 1512 | 2025 | 2874 | 3941 | 7010 | 9779 |
| KATSUURA10 | 2591(0.03) | 5212(0.03) | 4882(0.03) | 10239(0.03) | 8648(0.03) | 19539(0.03) | 26799(0.03) | 51916(0.03) |
| MICHALEWICZ4 | 1230(0.37) | 1482(0.40) | 2142(0.53) | 3154(0.80) | 3757(0.63) | 5906(0.90) | 9094(0.93) | 12347(0.90) |
| POTENTIAL5 | 1205 | 1984 | 2523 | 4283 | 4944 | 8166 | 11784 | 20239 |
| POTENTIAL6 | 1514(0.13) | 2732(0.27) | 2881(0.23) | 5617(0.50) | 5887(0.37) | 12034(0.80) | 15758(0.83) | 27537(0.97) |
| POTENTIAL10 | 2527(0.47) | 4956(0.57) | 4958(0.77) | 8409(0.93) | 8599(0.93) | 15417 | 19233 | 34576 |
| RASTRIGIN | 1390(0.73) | 1260(0.90) | 2499(0.93) | 2323(0.87) | 3639 | 3840(0.93) | 7904 | 8761(0.93) |
| ROSENBROCK8 | 1396 | 2122 | 2484 | 4164 | 4964 | 8085 | 12581 | 20176 |
| ROSENBROCK16 | 1861 | 3236 | 3316 | 6133 | 6467 | 11469 | 16465 | 29665 |
| SHARPRIDGE10 | 1601 | 2580 | 2940 | 5194 | 6006 | 10297 | 15017 | 25314 |
| SHEKEL5 | 884 | 1100 | 1611 | 2261 | 3230 | 4540 | 8010 | 10847 |
| SHEKEL7 | 902 | 1110 | 1688 | 2279 | 3212 | 4517 | 8052 | 10817 |
| SHEKEL10 | 953 | 1115 | 1607 | 2217 | 3168 | 4266 | 7895 | 10506 |
| SPHERE10 | 434 | 430 | 831 | 855 | 1660 | 1706 | 4132 | 4238 |
| STEP ELLIPSOIDAL4 | 922(0.97) | 629(0.13) | 1700 | 1392(0.50) | 3018 | 2842(0.83) | 6528 | 6095(0.80) |
| SINUSOIDAL8 | 1002(0.93) | 1300 | 1877 | 2443 | 3272 | 4572 | 7623 | 10946 |
| SINUSOIDAL16 | 1432(0.63) | 2225(0.83) | 2498(0.80) | 4038(0.97) | 4454(0.93) | 6588 | 10649 | 16327 |
| TEST2N4 | 959(0.73) | 1033(0.63) | 1557(0.87) | 1969(0.93) | 3236 | 3878 | 6856 | 8420 |
| TEST2N5 | 932(0.40) | 1105(0.53) | 1651(0.63) | 2196(0.77) | 3639(0.97) | 4109(0.86) | 7659 | 9566(0.86) |
| TEST30N10 | 935 | 1214 | 1552 | 2818 | 3134 | 5238 | 8436 | 13206 |
| ZAKHAROV10 | 645 | 811 | 1250 | 1670 | 2521 | 3314 | 6234 | 8102 |
| **SUM** | **58905(0.88)** | **84695(0.87)** | **107672(0.92)** | **165058(0.93)** | **204488(0.95)** | **316189(0.96)** | **502636(0.97)** | **773388(0.96)** |

**Table 4.** Experiment different number of population (proposed method and Genetic Algorithm)

control and dynamic balance between local and global search, which enhance convergence efficiency without compromising precision.

In summary, population size plays a crucial role in determining the optimization efficiency of both algorithms. For practical applications, a moderate population configuration ensures optimal performance, achieving rapid, accurate, and stable convergence with minimal computational overhead.

### *3.6. Performance Evaluation on Practical Problems*

To further examine the practical efficiency and scalability of the proposed optimization algorithm, two real-world engineering design problems were investigated: the GasCycle[50] and the Tandem Queueing System[51]. These problems were selected because they differ significantly in mathematical formulation and computational complexity, providing a comprehensive framework for evaluating the algorithm's performance under diverse and realistic conditions.

Each problem was tested across multiple dimensional configurations, ranging from 40 to 400 variables, in order to assess how the algorithm behaves as the search space becomes more complex. For every configuration, the execution time in seconds was recorded as the main performance indicator. This experimental setup enables a direct comparison of how computational efficiency changes with increasing dimensionality.

- **GasCycle Thermal Cycle**
  Vars: $\boldsymbol{x} = [T_1, T_3, P_1, P_3]^\top$. $\quad r = P_3/P_1,\ \gamma = 1.4.$

$$\eta(\boldsymbol{x}) = 1 - r^{-(\gamma-1)/\gamma}\,\frac{T_1}{T_3}, \qquad \min_{\boldsymbol{x}} f(\boldsymbol{x}) = -\eta(\boldsymbol{x}).$$

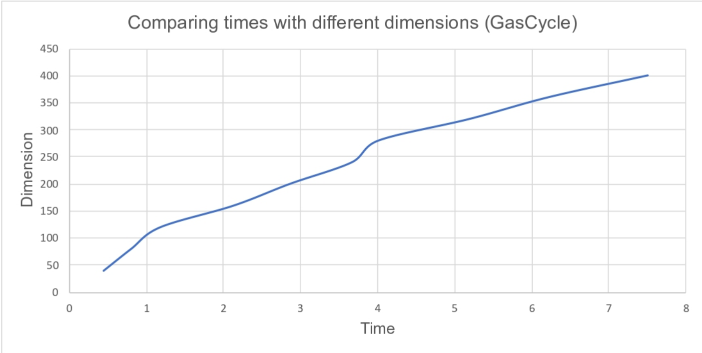


**Figure 4.** Comparing times with different dimensions( GasCycle)

Bounds: $300 \le T_1 \le 1500,\ 1200 \le T_3 \le 2000,\ 1 \le P_1, P_3 \le 20.$
Penalty: infeasible $\Rightarrow f = 10^{20}$.

The GasCycle problem represents a thermodynamic optimization task involving nonlinear relationships and interdependent design parameters. As the dimensionality of the problem increases, the algorithm exhibits a smooth and predictable increase in computational effort. Specifically, the execution time increases from 0.449 s (dimension 40) to 7.511 s (dimension 400).

The graph illustrates a clear, nearly linear increase in execution time as the problem's dimensionality grows from 40 to 400 variables. This trend indicates that the algorithm scales smoothly, maintaining computational stability while handling larger search spaces. The consistent growth in time reflects the efficiency of the adaptive mechanism, which balances the computational effort across dimensions without abrupt fluctuations.

- **Tandem Space Trajectory** (MGA-1DSM, EVEEJ + 2×Saturn)
  Vars **(**$D$=18**):** $\boldsymbol{x} = [t_0, T_1, T_2, T_3, T_4, T_{5A}, T_{5B}, s_1, s_2, s_3, s_4, s_{5A}, s_{5B}, r_p, k_{A1}, k_{A2}, k_{B1}, k_{B2}]^\top$.

$$\begin{aligned}
&7000 \le t_0 \le 10000,\\
&30 \le T_1 \le 500,\ 30 \le T_2 \le 600,\ 30 \le T_3 \le 1200,\\
&30 \le T_4 \le 1600,\ 30 \le T_{5A}, T_{5B} \le 2000,\\
&0 \le s_{1..4}, s_{5A}, s_{5B}, r_p, k_{A1}, k_{A2}, k_{B1}, k_{B2} \le 1.
\end{aligned}$$

Objective:

$$\min_{\boldsymbol{x}} \Delta V_{\text{tot}} = \Delta V_{\text{launch}}(T_1) + \Delta V_{\text{legs}}(T_1{:}T_4) + \Delta V_A + \Delta V_B + \Delta V_{\text{DSM}}(\boldsymbol{s}, r_p) - G_{\text{GA}} - G_J + P_{\text{hard}} +$$

$$P_{\text{soft}} = \beta \max\Big\{0,\ (T_1 + \cdots + T_4 + \tfrac{1}{2}(T_{5A} + T_{5B})) - 3500\Big\}.$$

Notes: $\Delta V_{\text{launch}}$ decreases (log-like) in $T_1$ ($\ge 6$ km/s floor), leg/branch costs decrease with TOF.

The Tandem Queueing System is a more demanding case, characterized by strong interdependencies among variables and a denser set of constraints. As the dimensionality of the problem increases, the algorithm exhibits a smooth and predictable increase in computational effort. Specifically, the execution time increases from 1.746 s (dimension 40) to 54.123 s (dimension 400).

The graph shows a steeper increase in execution time compared to the GasCycle case, reflecting the higher complexity of the Tandem problem. Despite this, the algorithm maintains stable and efficient performance.

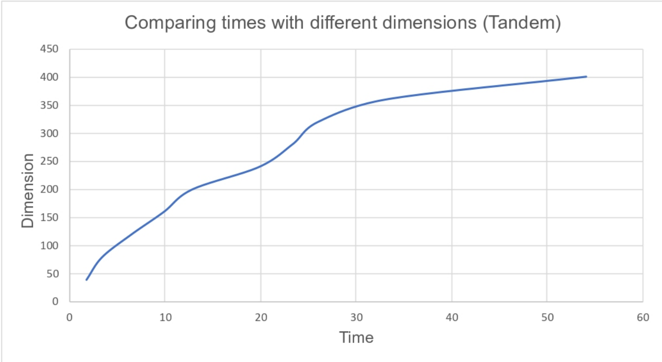


**Figure 5.** Comparing times with different dimensions( Tandem)

The results from both test problems show that the proposed algorithm maintains a good balance between accuracy and computational cost. The execution time remains within reasonable limits even as the problem size increases. Overall, the algorithm proves to be adaptable and scalable, demonstrating consistent performance across different problems.

## 4. Discussion

The experimental analysis conducted in this study provides a clear and comprehensive understanding of the proposed optimization algorithm's performance under various parameter settings and in comparison with several state-of-the-art metaheuristic methods. The evaluation, carried out on a diverse set of benchmark functions encompassing both unimodal and multimodal landscapes, offers solid insight into the algorithm's convergence behavior, reliability, and computational efficiency.

Across all benchmarks, the proposed method consistently demonstrated strong and stable convergence performance. When compared with well-known algorithms such as the Genetic Algorithm, WOA, MEWOA, Differential Evolution, iPSO, AOA, SAO, and ACO, the proposed approach achieved the lowest number of function evaluations in nearly all test cases while maintaining a 100% success rate in most functions. Moreover, the total computational cost was noticeably lower than that of the competing methods, highlighting the algorithm's efficient exploitation ability and the effectiveness of its adaptive control mechanism. These results clearly indicate that the algorithm can reach the global optimum with high reliability while requiring substantially fewer computational resources.

The statistical analysis further substantiates the superiority and reliability of the proposed optimization framework. Pairwise significance tests revealed that the differences between the proposed method and nearly all baseline algorithms were highly significant, with most comparisons achieving p-values below 0.0001. This indicates that the observed performance improvements are not attributable to random fluctuations. Moreover, the internal comparison among the exploration–exploitation control modes demonstrated statistically significant behavioral distinctions, confirming that each regime induces a distinct search dynamic. These results collectively validate the robustness of the proposed approach and provide strong statistical evidence of its effectiveness across diverse optimization scenarios.

The exploration of different internal search strategies: Aggressive, Conservative, and Mixed provided further insights into the flexibility of the proposed framework. The Conservative mode delivered a balanced compromise between accuracy and computational effort, while the Aggressive mode enhanced global exploration and proved particularly advantageous in highly multimodal landscapes. The Mixed strategy, combining the benefits of both, produced consistently reliable performance across a wide range of problems. These findings demonstrate the algorithm's capacity to adapt its search dynamics intelligently according to the complexity and structure of the optimization problem at hand.

The impact of chromosome population size was also examined for both the proposed algorithm and the standard Genetic Algorithm. In both cases, population size played a decisive role in shaping convergence behavior and overall computational cost. Smaller populations led to faster convergence but sometimes at the expense of reduced diversity, while very large populations resulted in redundant evaluations and slower progress. Optimal performance for both algorithms was observed with medium-sized populations, where the balance between exploration and exploitation was maintained.

When the two algorithms were compared directly under identical conditions, the proposed method consistently outperformed the Genetic Algorithm across all configurations. For instance, under equivalent population sizes, the proposed approach required fewer function evaluations to reach convergence while preserving a perfect success rate.

In summary, the experimental findings confirm that the proposed optimization framework achieves a well-balanced combination of computational efficiency, convergence accuracy, and robustness. It adapts effectively to different problem structures and parameter settings, consistently outperforming traditional evolutionary and swarm-based algorithms. The combination of fast convergence, stable performance across multiple environments, and consistent reliability highlights its potential as a powerful, versatile, and scalable optimization tool for real-world applications.

## 5. Conclusions

This study introduced a novel optimization framework that incorporates an adaptive mechanism to balance exploration and exploitation throughout the search process. The experimental evaluation, conducted across a diverse suite of benchmark functions, demonstrated that the proposed algorithm consistently surpasses well-established metaheuristic methods in terms of convergence accuracy, computational speed, and overall efficiency.

Statistical tests confirmed that these improvements are not the result of random fluctuations but stem from methodological advances. Moreover, the investigation of the internal search strategies Aggressive, Conservative, and Mixed highlighted the algorithm's ability to dynamically adjust its search behavior in response to the characteristics and complexity of each optimization problem. This adaptability ensures a well-calibrated interaction between global exploration and local exploitation, leading to efficient and stable convergence.

Overall, the proposed algorithm has proven to be a flexible optimization approach capable of addressing both simple and complex real-world problems. Its consistent superiority over well-known optimization techniques, together with statistically validated performance, underscores its strong potential for applications that demand precision, adaptability, and computational efficiency.

**Author Contributions:** G.K., V.C. and I.G.T. conceived of the idea and the methodology, and G.K. and V.C. implemented the corresponding software. G.K. conducted the experiments, employing objective functions as test cases, and provided the comparative experiments. I.G.T. performed the necessary statistical tests. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research has been financed by the European Union: Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan, under the call RESEARCH–CREATE–INNOVATE, project name "iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques" (project code: TAEDK-06195).

**Institutional Review Board Statement:** Not Applicable.

**Informed Consent Statement:** Not Applicable.

**Data Availability Statement:** Not Applicable.

**Conflicts of Interest:** The authors declare no conflicts of interest.

# References

1. A. Törn, M.M. Ali, S. Viitanen, Stochastic global optimization: Problem classes and solution techniques. Journal of Global Optimization **14**, pp. 437-447, 1999.
2. Floudas, C. A., & Pardalos, P. M. (Eds.). (2013). State of the art in global optimization: computational methods and applications
3. Horst, R., & Pardalos, P. M. (Eds.). (2013). Handbook of global optimization (Vol. 2). Springer Science & Business Media.
4. ACarrizosa, E., Molero-Río, C., & Romero Morales, D. (2021). Mathematical optimization in classification and regression trees. Top, 29(1), 5-33.
5. Legat, B., Dowson, O., Garcia, J. D., & Lubin, M. (2022). MathOptInterface: a data structure for mathematical optimization problems. INFORMS Journal on Computing, 34(2), 672-689.
6. Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. Neurocomputing, 532, 183-214.
7. Stilck França, D., & Garcia-Patron, R. (2021). Limitations of optimization algorithms on noisy quantum devices. Nature Physics, 17(11), 1221-1227.
8. Zhang, J., & Glezakou, V. A. (2021). Global optimization of chemical cluster structures: Methods, applications, and challenges. International Journal of Quantum Chemistry, 121(7), e26553.
9. Hu, Y., Zang, Z., Chen, D., Ma, X., Liang, Y., You, W., & Zhang, Z. (2022). Optimization and evaluation of SO2 emissions based on WRF-Chem and 3DVAR data assimilation. Remote Sensing, 14(1), 220.
10. Kaur, P., & Singh, R. K. (2023). A review on optimization techniques for medical image analysis. Concurrency and Computation: Practice and Experience, 35(1), e7443.
11. Houssein, E. H., Hosney, M. E., Mohamed, W. M., Ali, A. A., & Younis, E. M. (2023). Fuzzy-based hunger games search algorithm for global optimization and feature selection using medical data. Neural Computing and Applications, 35(7), 5251-5275.
12. Wang, Y., Ma, Y., Song, F., Ma, Y., Qi, C., Huang, F., ... & Zhang, F. (2020). Economic and efficient multi-objective operation optimization of integrated energy system considering electro-thermal demand response. Energy, 205, 118022.
13. Alirahmi, S. M., Mousavi, S. B., Razmi, A. R., & Ahmadi, P. (2021). A comprehensive techno-economic analysis and multi-criteria optimization of a compressed air energy storage (CAES) hybridized with solar and desalination units. Energy Conversion and Management, 236, 114053.
14. Shezan, S. A., Ishraque, M. F., Shafiullah, G. M., Kamwa, I., Paul, L. C., Muyeen, S. M., ... & Kumar, P. P. (2023). Optimization and control of solar-wind islanded hybrid microgrid by using heuristic and deterministic optimization algorithms and fuzzy logic controller. Energy reports, 10, 3272-3288.
15. Xu, Z., Zhao, Z., & Liu, J. (2024). Deterministic Multi-Objective Optimization of Analog Circuits. Electronics, 13(13), 2510.
16. Hsieh, Y. P., Karimi Jaghargh, M. R., Krause, A., & Mertikopoulos, P. (2024). Riemannian stochastic optimization methods avoid strict saddle points. Advances in Neural Information Processing Systems, 36.
17. Tyurin, A., & Richtárik, P. (2024). Optimal time complexities of parallel stochastic optimization methods under a fixed computation model. Advances in Neural Information Processing Systems, 36.
18. Price, W. (1983). Global optimization by controlled random search. Journal of optimization theory and applications, 40(3), 333-348.
19. Křivý, I., & Tvrdik, J. (1995). The controlled random search algorithm in optimizing regression models. Computational statistics & data analysis, 20(2), 229-234.
20. Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. science, 220(4598), 671-680.
21. Ingber, L. (1989). Very fast simulated re-annealing. Mathematical and computer modelling, 12(8), 967-973.
22. Tsoulos, I. G., & Lagaris, I. E. (2006). MinFinder: Locating all the local minima of a function. Computer Physics Communications, 174(2), 166-179.
23. Liu, Y., & Tian, P. (2015). A multi-start central force optimization for global optimization. Applied Soft Computing, 27, 92-98.
24. Sohail, A. (2023). Genetic algorithms in the fields of artificial intelligence and data sciences. Annals of Data Science, 10(4), 1007-1018.
25. Charilogis, V., Tsoulos, I. G., & Stavrou, V. N. (2023). An Intelligent Technique for Initial Distribution of Genetic Algorithms. Axioms, 12(10), 980.
26. Yuan, F., Huang, X., Zheng, L., Wang, L., Wang, Y., Yan, X., ... & Peng, Y. (2025). The evolution and optimization strategies of a PBFT consensus algorithm for consortium blockchains. Information, 16(4), 268.
27. Shir, O. M., & Emmerich, M. (2025, July). Correlated geometric mutations for integer evolution strategies. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 1825-1832).

28. Vanneschi, L., Farinati, D., Rasteiro, D., Rosenfeld, L., Pietropolli, G., & Silva, S. (2025). Exploring non-bloating geometric semantic genetic programming. Genetic Programming Theory and Practice XXI, 237-258.
29. Maurya, P., Kushwaha, A., & Prakash, O. (2025). Medical Data Classification Using Genetic Programming: A Systematic Literature Review. Expert Systems, 42(3), e70007.
30. Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., & Xu, J. (2021). An improved differential evolution algorithm and its application in optimization problem. Soft Computing, 25, 5277-5298.
31. Pant, M., Zaheer, H., Garcia-Hernandez, L., & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. Engineering Applications of Artificial Intelligence, 90, 103479.
32. Shami, T. M., El-Saleh, A. A., Alswaitti, M., Al-Tashi, Q., Summakieh, M. A., & Mirjalili, S. (2022). Particle swarm optimization: A comprehensive survey. Ieee Access, 10, 10031-10061.
33. Gad, A. G. (2022). Particle swarm optimization algorithm and its applications: a systematic review. Archives of computational methods in engineering, 29(5), 2531-2561.
34. Rokbani, N., Kumar, R., Abraham, A., Alimi, A. M., Long, H. V., Priyadarshini, I., & Son, L. H. (2021). Bi-heuristic ant colony optimization-based approaches for traveling salesman problem. Soft Computing, 25, 3775-3794.
35. Wu, L., Huang, X., Cui, J., Liu, C., & Xiao, W. (2023). Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. Expert Systems with Applications, 215, 119410.
36. Hamadneh, T., Kaabneh, K., AbuFalahah, I., Bektemyssova, G., Shaikemelev, G., Umutkulov, D., ... & Dehghani, M. (2024). Magnificent frigatebird optimization: a new bio-inspired metaheuristic approach for solving optimization problems. Comput. Mater. Contin, 80, 2721-2741.
37. Powell, M. J. D. (1989). A tolerant algorithm for linearly constrained optimization calculations. Mathematical Programming, 45(1), 547-566.
38. Charilogis, V., & Tsoulos, I. G. (2022). Toward an ideal particle swarm optimizer for multidimensional functions. Information, 13(5), 217.
39. Ali, M. M., & Kaelo, P. (2008). Improved particle swarm algorithms for global optimization. Applied mathematics and computation, 196(2), 578-593.
40. Koyuncu, H., & Ceylan, R. (2019). A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems. Journal of Computational Design and Engineering, 6(2), 129-142.
41. Siarry, P., Berthiau, G., Durdin, F., & Haussy, J. (1997). Enhanced simulated annealing for globally minimizing functions of many-continuous variables. ACM Transactions on Mathematical Software (TOMS), 23(2), 209-228.
42. Tsoulos, I. G., & Lagaris, I. E. (2008). GenMin: An enhanced genetic algorithm for global optimization. Computer Physics Communications, 178(11), 843-851.
43. LaTorre, A., Molina, D., Osaba, E., Poyatos, J., Del Ser, J., & Herrera, F. (2021). A prescription of methodological guidelines for comparing bio-inspired optimization algorithms. Swarm and Evolutionary Computation, 67, 100973.
44. Gaviano, M., Kvasov, D. E., Lera, D., & Sergeyev, Y. D. (2003). Algorithm 829: Software for generation of classes of test functions with known local and global minima for global optimization. ACM Transactions on Mathematical Software (TOMS), 29(4), 469-480.
45. Alyasseri, Z. A. A., Ali, N. S., Al-Betar, M. A., Makhadmeh, S. N., Jamil, N., Awadallah, M. A., ... & Mirjalili, S. (2024). Recent advances of whale optimization algorithm, its versions and applications. Handbook of whale optimization algorithm, 9-31.
46. Zangmo, R., Sudabattula, S. K., Dharavat, N., Mishra, S., Basha, C. H., & Irfan, M. M. (2025). Techno-economic analysis of distribution system at various load models using MEWOA algorithm. Scientific Reports, 15(1), 8273.
47. Masoudi, M. R., Haghighi, M., & Sezavar, H. R. (2025, February). Optimization of Energy Hub Performance Using the Improved Particle Swarm Optimization (IPSO) Algorithm for Integrated Energy Systems. In 2025 12th Iranian Conference on Renewable Energies and Distributed Generation (ICREDG) (pp. 1-6). IEEE.
48. hang, R., Mao, S., Zhao, S., & Liu, C. (2024). An arithmetic optimization algorithm with balanced diversity and convergence for multimodal multiobjective optimization. Swarm and Evolutionary Computation, 91, 101724.
49. Sulaiman, A. T., Bello-Salau, H., Onumanyi, A. J., Mu'azu, M. B., Adedokun, E. A., Salawudeen, A. T., & Adekale, A. D. (2024). A particle swarm and smell agent-based hybrid algorithm for enhanced optimization. Algorithms, 17(2), 53.
50. Luo, B., Su, X., Zhang, S., Yan, P., Liu, J., & Li, R. (2025). Analysis of a novel gas cycle cooler with large temperature glide for space cooling. Energy, 326, 136294.
51. Keerthika, R., Niranjan, S. P., & Komala Durga, B. (2025). A Survey on the tandem queueing models. Scope, 14, 134-148.