foundations

MDPI

Article

Parallel Balanced Grey Wolf Optimizer: A Cooperative Parallel Approach for Large-Scale Optimization Problems

Glykeria Kyrou *, Konstantinos G. Barkas, Vasileios Charilogis and Ioannis G. Tsoulos

Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece; kbarkas@uoi.gr (K.G.B.); v.charilog@uoi.gr (V.C.); itsoulos@uoi.gr (I.G.T.)
* Correspondence: g.kyrou@uoi.gr

Abstract

In recent years, large-scale optimization problems have become increasingly common in various fields, such as machine learning and data analysis, generating increased references to both computational cost and accurate solutions. The Grey Wolf Optimizer (GWO) is an efficient collective intelligence algorithm; however, its performance may be limited when high-permissive problems are allowed or in environments with strong multimodal landscapes. In this paper, we propose the Parallel Balanced Grey Wolf Optimizer (Parallel-BGWO), a parallel extension of GWO that aims to improve the balance between exploration and exploitation of the search space. The proposed algorithm divides the total population into multiple subpopulations, which cooperate through information exchange. This cooperation reduces the probability of premature convergence and enhances the global search. The parallel implementation exploits modern multi-core computing architectures, achieving a significant reduction in execution time, while at the same time maintaining or improving the quality of the final solutions. Experimental evaluations on high-dimensional benchmark functions show that ParallelBGWO exhibits faster convergence and a reduced number of objective function evaluations compared to the classical version of GWO and other prominent methods. The results highlight ParallelBGWO as an efficient approach for demanding global optimization problems.

Keywords: global optimization; GWO algorithm; evolutionary techniques; stochastic methods; large-scale problems

1. Introduction

The basic goal of global optimization is to find the global minimum of a continuous multidimensional function, which is defined as

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

with S:

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

with a_i and b_i representing lower and upper bounds for each variable x_i.

In recent years, global optimization problems have become increasingly prevalent in modern scientific fields, such as machine learning [1,2], big data analysis [3], communication systems [4], and energy management [5]. The growing complexity of these applications often leads to high-dimensional problems, making effective exploration of the solution space particularly challenging. Of particular interest are large-scale global optimization

Academic Editor: Hsien-Chung Wu

Received: 29 March 2026
Revised: 30 April 2026
Accepted: 26 May 2026
Published: 1 June 2026

(LSGO) problems [6], where computational complexity increases significantly. In such cases, classical deterministic methods often fail to provide satisfactory solutions within a reasonable time, either due to high computational cost or because they become trapped in local minima. The importance of LSGO problems has been recognized internationally through competitions organized within the framework of the IEEE Congress on Evolutionary Computation (CEC) [7–9], which significantly strengthened research in this field. To address these challenges, there has been growing interest in metaheuristic and stochastic algorithms capable of effectively approximating the global optimum, even in complex and high-dimensional environments. The Grey Wolf Optimizer (GWO) [10,11] is a widely used swarm intelligence algorithm inspired by the cooperative behavior of wolf packs. Its simplicity and relatively fast convergence make it particularly attractive for continuous optimization problems. However, when applied to large-scale problems, it exhibits limited exploration capability and an increased risk of premature convergence.

Although numerous variants of the GWO have been proposed in the literature to improve either exploration or exploitation performance, limited attention has been given to the simultaneous enhancement of scalability, computational efficiency, and adaptive convergence control in large-scale optimization environments. The novelty of the proposed approach lies in addressing these challenges through the integration of cooperative parallel population evolution, local refinement mechanisms, and adaptive termination strategies within the GWO framework. In particular, many existing approaches do not fully exploit parallel search mechanisms while simultaneously incorporating efficient stopping criteria to reduce unnecessary computational effort. Motivated by these limitations, the present work aims to improve the applicability of GWO in large-scale global optimization problems through a unified optimization scheme that enhances both search efficiency and convergence behavior. Unlike existing GWO variants, the proposed method simultaneously integrates parallel cooperation, adaptive propagation, and termination control within a unified framework. The main contributions of this work are summarized as follows:

- A parallel cooperative extension of the GWO, where the population is divided into multiple subpopulations to enhance exploration in large-scale search spaces.
- A propagation mechanism that enables controlled information exchange among subpopulations, improving convergence speed while maintaining population diversity.
- The incorporation of local refinement through the BFGS algorithm and adaptive termination strategies to improve convergence accuracy and reduce unnecessary computational effort.
- A comprehensive experimental evaluation on large-scale benchmark problems, demonstrating improved performance in terms of computational efficiency, solution quality, and robustness.

In this context, the Parallel Balanced Grey Wolf Optimizer (ParallelBGWO) is proposed, which is a parallel extension of GWO that aims to improve the balance between exploration and exploitation. The proposed algorithm divides the population into subpopulations that evolve simultaneously and exchange information, reducing the probability of being trapped at local extremes. In addition, a local optimization procedure is incorporated through the BFGS algorithm [12], while a termination rule [13] is implemented to effectively terminate the global optimization method. This modification avoids unnecessary calculations. Also, at the subpopulation level, different termination strategies (ANY, MAJORITY, and ALL) are considered, which determine when the overall process is completed based on the convergence status of the individual clusters. The use of flexible termination criteria helps to avoid unnecessary calculations and improve the overall computational efficiency of the algorithm. The proposed algorithm was tested on real-world problems. As a representative real-world application, the training of artificial neural networks (ANNs)

can be formulated as a high-dimensional, non-convex optimization problem where the objective is to determine the optimal set of model parameters by minimizing a predefined loss function. Due to the complexity of the search space and the presence of multiple local optima, conventional gradient-based methods may suffer from premature convergence. In this context, global optimization techniques provide a powerful alternative. For this reason, ANN training is employed in this work as an additional application to evaluate the effectiveness and robustness of the proposed ParallelBGWO algorithm.

The proposed ParallelBGWO algorithm is primarily intended for continuous large-scale black-box optimization problems, where the objective function is accessible only through function evaluations and analytical gradient information is unavailable. In such cases, each function evaluation may be computationally expensive, making the reduction in the total number of evaluations a critical objective. In addition, due to its parallel architecture, the method is particularly suitable for computationally demanding optimization tasks, where reducing execution time and accelerating convergence are of practical importance.

The remainder of this paper is organized as follows. Section 2 reviews the related literature and positions the proposed method within the current state of the art. Section 3 presents the BGWO algorithm, along with the proposed ParallelBGWO variant and its corresponding flowchart. Section 4 describes the benchmark test functions and reports the experimental results. Section 5 discusses the obtained results. Finally, Section 6 concludes the paper and outlines directions for future research.

2. Literature Review

2.1. Classical and Swarm-Based Metaheuristic Optimization Methods

Global optimization problems constitute a fundamental research topic across many scientific domains, as the need to determine optimal solutions arises in an increasing number of practical applications. The growing complexity of modern systems frequently leads to nonlinear, multimodal, and high-dimensional optimization problems, where classical deterministic approaches often fail to provide reliable solutions. Consequently, metaheuristic optimization algorithms have emerged as efficient alternatives due to their ability to explore large search spaces while avoiding local optima [14]. Among the earliest and most widely adopted metaheuristic approaches are evolutionary algorithms, which simulate principles of biological evolution. Genetic algorithms (GAs) constitute one of the first successful population-based optimization techniques, employing mechanisms such as selection, crossover, and mutation to iteratively evolve candidate solutions [14,15]. Differential Evolution (DE) was later introduced as an efficient optimization algorithm for continuous domains, offering simple implementation and robust convergence behavior [16,17]. Another important category of optimization methods is swarm intelligence, where search agents cooperate by simulating collective behaviors observed in natural systems. Representative algorithms include Particle Swarm Optimization (PSO) [18,19], Ant Colony Optimization (ACO) [20,21] and Artificial Bee Colony (ABC) [22,23], which have demonstrated strong performance across a variety of optimization tasks.

2.2. Recent Adaptive Metaheuristic Algorithms and Their Limitations

In recent years, significant research attention has focused on the development of advanced metaheuristic algorithms aiming to improve search efficiency and convergence behavior. Various nature-inspired and adaptive optimization techniques have been proposed to enhance the balance between exploration and exploitation during the search process, which is widely regarded as a critical factor in metaheuristic performance [24,25]. Similarly, the Bat Algorithm (BA), which simulates the echolocation behavior of bats, has shown promising results in engineering optimization and global search applications [26,27].

Additional modern swarm-based approaches include the Whale Optimization Algorithm (WOA) [28,29], Harris Hawks Optimization (HHO) [30,31], Aquila Optimizer (AO) [32,33], Arithmetic Optimization Algorithm (AOA) [34,35], Magnificent Frigatebird Optimization Algorithm (MFO) [36], and Salp Swarm Algorithm (SSA) [37,38]. These optimization algorithms seek to improve performance through adaptive movement operators, nonlinear parameter adjustment, and dynamic search strategies. However, despite their promising effectiveness, many recent metaheuristic methods continue to exhibit important limitations. In particular, they may suffer from premature convergence, insufficient population diversity preservation, and reduced robustness when applied to highly multimodal or large-scale optimization problems. Such limitations remain common challenges in modern metaheuristic optimization research and motivate the continuous development of improved and hybrid optimization strategies. To provide a clearer perspective, the limitations of existing metaheuristic methods can be categorized into several main groups:

(i) Exploration–exploitation imbalance, where algorithms fail to maintain an effective trade-off during the search process.
(ii) Premature convergence, which leads to suboptimal solutions due to loss of population diversity.
(iii) Limited scalability, particularly in large-scale global optimization (LSGO) problems.
(iv) Insufficient parallel cooperation mechanisms, as many methods rely on sequential execution or weak interaction among subpopulations. These limitations highlight the need for more robust, scalable, and cooperative optimization frameworks.

2.3. Grey Wolf Optimization Variants

Among swarm-based optimization algorithms, the GWO has attracted considerable attention due to its simplicity and effectiveness in solving continuous optimization problems. Inspired by the leadership hierarchy and cooperative hunting behavior of gray wolves, GWO models the interaction between x_a, x_b and x_c wolves to guide the search process toward promising regions of the solution space [10,11]. Despite its effectiveness, the original GWO algorithm may suffer from premature convergence and reduced exploration ability when applied to large-scale or complex optimization problems. To address these limitations, several improved GWO variants have been proposed in recent years. Chen et al. introduced the Balanced Grey Wolf Optimizer (BGWO), which aims to maintain a better balance between exploration and exploitation during the optimization process [39]. Similarly, Adhikary and Acharyya proposed the Randomized Balanced Grey Wolf Optimizer (RBGWO), incorporating stochastic perturbation mechanisms to improve search diversity [40]. Furthermore, Wang et al. developed the Adaptively Balanced Grey Wolf Optimizer (ABGWO), which dynamically adjusts the balance between exploration and exploitation throughout the search process [41].

2.4. Parallel and Cooperative Metaheuristic Frameworks

To further improve optimization efficiency, recent studies have emphasized the importance of parallel and cooperative metaheuristic strategies, where the population is divided into multiple subpopulations that explore different regions of the search space simultaneously. The exchange of information between these subpopulations improves diversity preservation, reduces premature convergence, and accelerates the optimization process. Such cooperative approaches have been successfully demonstrated in modern parallel implementations of algorithms such as Smell Agent Optimization (SAO) [42,43]. However, despite their effectiveness, most existing parallel metaheuristic approaches rely on relatively fixed communication strategies and simplistic propagation mechanisms, while

limited attention has been devoted to systematically studying the effects of structured propagation control and adaptive termination coordination.

In addition to stochastic metaheuristic approaches, deterministic global optimization methods have also been extensively studied. These methods aim to provide theoretical guarantees for convergence by systematically exploring the search space, but their computational cost often increases significantly with problem dimensionality. Recent benchmarking studies have provided a comprehensive comparison between deterministic and stochastic optimization algorithms. In particular, the large-scale study by Stripinis and Paulavičius [44] evaluates a wide range of deterministic derivative-free methods against stochastic optimizers across numerous benchmark problems, showing that deterministic methods perform well in low-dimensional settings, while stochastic approaches exhibit superior scalability and robustness in high-dimensional problems.

2.5. Research Gap and Motivation

Based on the above categorized limitations, several research gaps remain evident in the current literature. First, although numerous enhanced variants of GWO have been proposed, most existing studies focus primarily on sequential implementations and do not investigate structured cooperative parallel execution. Second, while parallel metaheuristic approaches have demonstrated computational benefits, the integration of Balanced Grey Wolf Optimization mechanisms within a parallel multi-subpopulation framework remains relatively unexplored. Third, configurable propagation strategies and adaptive termination mechanisms have not been sufficiently investigated in BGWO-based parallel optimization frameworks. Motivated by these limitations, the present work proposes a ParallelBGWO designed to improve the search efficiency of the classical GWO framework. The proposed method introduces a subpopulation-based cooperative parallel architecture, configurable information propagation strategies, adaptive termination policies, and local search refinement in order to improve convergence efficiency and optimization robustness. A summary of the related literature is shown in Table 1.

Table 1. Summary of the related literature and research gap.

Study	Main Contribution	Limitation
GWO [10,11]	Simple leadership-based optimization	Premature convergence
BGWO [39]	Improved exploration/exploitation balance	Sequential execution
RBGWO [40]	Increased stochastic diversity	No parallel cooperation
ABGWO [41]	Adaptive balancing strategy	No propagation mechanism
Parallel SAO [43]	Cooperative parallel optimization	Fixed communication scheme
Proposed Method	Parallel BGWO with adaptive propagation	Not applicable

3. Materials and Methods

3.1. The BGWO Algorithm

The BGWO is an improved version of the GWO, aiming to improve the balance between global exploration and local exploitation. In BGWO, each search agent represents a wolf, whose position corresponds to a candidate solution in the D-dimensional search space. First, a population of n wolves is randomly generated within the allowable limits of the problem. Then the objective function value is calculated for each wolf and a fitness ranking is performed. The three best solutions are defined as wolves x_a, x_b and x_c and constitute the leading trio of the population. The remaining wolves update their positions guided by the three leaders, according to the prey encirclement mechanism that characterizes

GWO. In each iteration of the algorithm, a nonlinear convergence factor a is calculated, which controls the dynamics of the search. In contrast to the classic GWO, where the factor decreases linearly, in BGWO, a nonlinear decay is used, which allows for more intense exploration in the initial iterations and a gradual increase in exploitation in the final phases of the process. Specifically, the proposed BGWO employs an exponential nonlinear decay formulation for the convergence factor a, as defined in Algorithm 1 instead of the standard linear reduction adopted in the original GWO. This formulation provides a slower reduction during the early optimization stages to reinforce exploration, followed by a smoother transition toward exploitation in later iterations. Consequently, the proposed strategy alleviates the overly rapid convergence behavior commonly associated with the linear decay mechanism of conventional GWO. For each wolf x_i, random coefficients are generated and used to calculate the control vectors and distances from the three leaders. The use of stochastic coefficients plays a key role in preserving population diversity and enhancing exploratory behavior during the search process. Randomization enables the algorithm to avoid premature convergence and escape local optima by generating diversified search trajectories. Although more advanced strategies, such as adaptive weighting schemes or attention-based mechanisms, have been proposed in recent metaheuristic approaches, these methods often introduce additional computational complexity and algorithmic overhead. In the context of large-scale optimization problems, where efficiency and scalability are critical, the use of stochastic coefficients provides a favorable trade-off between exploration capability and computational cost. Therefore, stochastic components are retained in the present formulation to maintain the simplicity, scalability, and effectiveness of the BGWO framework, while still achieving a balanced search behavior. The distances are calculated per dimension, taking into account the relative position of each wolf with respect to x_a, x_b and x_c. In the BGWO algorithm, there is a position update mechanism where independent stochastic scaling factors R_1, R_2, R_3 in the interval $[0.5, 1.5]$ are introduced. These factors randomly strengthen or weaken the influence of each leader, reducing excessive dependence on the current optimal solutions and enhancing population differentiation. For each wolf, three intermediate positions are calculated, each of which is guided by one of the leaders. The final updated position is the average of these three stochastic scaled positions.

This procedure improves the ability to escape from local minima and makes the algorithm more robust to problems with high multimodality. After all wolves have been updated, the objective function is recalculated, and the leading trio is redefined based on the new ranking. The iterative process continues until some termination criterion is met. The steps of the BGWO algorithm are presented in the following algorithm.

Algorithm 1 BGWO algorithm.

INPUT

- f: objective function
- n: population size (number of wolves)
- D: problem dimension
- T: maximum number of iterations
- k: convergence coefficient //BGWO parameter
- N_t: termination criterion

$p_l \in [0, 1]$ as the local search rate.

OUTPUT

-x_a: best solution (alpha wolf)

INITIALIZATION

- **Set** $t \leftarrow 0$ as the iteration counter
- **Initialize** wolves $x_i \in R^D$, $i \in \{1 \dots n\}$
- **Compute** fitness $F_i = f(x_i)$ for each wolf
- **Identify** best three wolves: $x_a(best), x_b, x_c$

Algorithm 1 *Cont.*

Main pseudocode

while stopping criterion is not met do

Compute nonlinear convergence factor: $a \leftarrow 2\exp^{\frac{k}{ln\frac{t-1}{T}}}$

for each wolf i, $i \in \{1 \ldots n\}$ **do**

Draw $r_1, r_2, r_3, r_4, r_5, r_6 \backsim U(0,1)$

Compute coefficient vectors: $C_a = 2 \cdot r_1, C_b = 2 \cdot r_2, C_c = 2 \cdot r_3$

Compute A-vectors: $A_a = a \cdot (2 \cdot r_4 - 1)$, $A_b = a \cdot (2 \cdot r_5 - 1)$, $A_c = a \cdot (2 \cdot r_6 - 1)$

Compute distances to leaders: $D_a = |C_a \cdot x_a - x_i|, D_b = |C_b \cdot x_b - x_i|, D_c = |C_c \cdot x_c - x_i|$

Draw stochastic scaling factors $R1$, $R2$, $R3$, $\backsim U(0.5, 1.5)$

Compute stochastic guided positions (BGWO update):

$x_1 = R_1 \cdot (x_a - A_a \cdot D_a)$,$x_2 = R_2 \cdot (x_b - A_b \cdot D_b)$,$x_3 = R_3 \cdot (x_c - A_c \cdot D_c)$

Update wolf position: $x_i \leftarrow \frac{x_1 + x_2 + x_3}{3}$

Draw $r \backsim U(0,1)$

if $r \leq p_l$**then**

Apply local search $x_i \leftarrow LS(x_i)$. In this procedure, the local optimization method of Powell [12] was used.

end if

endfor

Set $t = t + 1$

if $t \geq T$ **then** terminate

Evaluate $F_i \leftarrow f(x_i)$ for all wolves

Update x_a,x_b,x_c wolves based on best fitness values

Compute $\delta^{(t)} = \left| f_{\mathbf{min}}^{(t)} - f_{\mathbf{min}}^{(t-1)} \right|$. This termination method was introduced in [13].

if $\delta^{(t)} \leq \epsilon$ **for** $N_t$iterations **then** terminate.

Update x_a,x_b,x_c wolves based on best fitness values

end while // Goto termination check

Return x_a

3.2. The Parallel Algorithm of BGWO

The parallel implementation of the BGWO is based on dividing the population into N subpopulations, which are executed independently on different processing units. First, the parameters are initialized and the termination strategy (T_S) is selected. The algorithm also employs a propagation method (N_M), a propagation rate (N_R), a number of propagated agents (N_P), and a termination criterion (N_I) to regulate communication and stopping conditions among subpopulations.

The propagation rate (N_R) determines how frequently the propagation mechanism is activated during the optimization process, controlling the number of iterations between successive information exchanges among subpopulations. Smaller values of N_R lead to more frequent communication, which may accelerate convergence but can reduce diversity, while larger values allow more independent exploration of the search space. Similarly, the number of propagated agents N_P defines how many candidate solutions are exchanged during each propagation step. Higher values of N_P increase the amount of shared information and may enhance exploitation of promising search regions; however, excessive propagation may reduce diversity and increase the risk of premature convergence. In the present work, the values of N_R and N_P were selected empirically to achieve a balance between exploration and exploitation. The specific values of N_R and N_P used in this study are reported in the Experimental Section. These parameters were selected based on preliminary experiments, where different configurations were evaluated in terms of convergence speed and solution quality. The adopted values were found to provide an effective trade-off between efficient information exchange and preservation of population diversity.

In each iteration, the number of subpopulations that have satisfied the termination criterion is calculated. This information is compared with a threshold Q,, which defines the minimum number of subpopulations that must satisfy the termination criterion according

to the selected termination strategy (T_S) as follows: ANY, where the algorithm proceeds when at least one subpopulation terminates; $MAJORITY$, where at least the majority of the subpopulations are required to have terminated; and ALL, where all subpopulations must have terminated.

Then, each subpopulation evolves in parallel by performing the basic steps of BGWO (evaluation, leader selection, and position update). At intervals determined by N_R, a propagation mechanism based on the selected propagation method N_M is applied to exchange good solutions between the subpopulations using N_P selected agents. The iteration counter is incremented and the overall termination criterion is checked. If it is not satisfied, the process is repeated. Upon termination, a local search is applied to the best solution x_{best} and the final result is returned. The steps of the parallel BGWO algorithm are presented in Algorithm 2.

Algorithm 2 The parallel algorithm of BGWO.

INPUT
- f: objective function
- N: number of subpopulations
- N_M: propagation method
- N_R: propagation rate
- N_P: number of agents for propagation
- N_I: termination criterion
- T_S: termination strategy, $T_S \in$ {ANY, MAJORITY, ALL}

OUTPUT
-X_{best}: global best solution over all subpopulations

INITIALIZATION
- **Set** $t \leftarrow 1$
- **Select** termination strategy T_S

MAIN LOOP
Compute the number of terminated subpopulations.
If $T_S \leftarrow ANY$ **then**
    $Q \leftarrow 1$
Else if $T_S \leftarrow MAJORITY$ **then**
    $Q \leftarrow N/2$
Else $T_S \leftarrow ALL$ **then**
    $Q \leftarrow N$
End if
For k$\in 1...N$ **in Parallel do**
    Execute for each subpopulation k all the instructions in the inner part of the whole of the basic algorithm.
End For
If t mod $N_R \leftarrow 0$, apply the propagation scheme N_M with N_P agents to the subpopulations.
    Set $t \leftarrow t + 1$
End if
If According to parameter N_I its termination rule does not apply, go to Main Step.
    Apply local search procedure to x_{best}.
End if

In the proposed implementation, each subpopulation is assigned to an independent processing unit, typically mapped to a separate thread or CPU core, enabling concurrent execution. A shared-memory parallelization model is adopted, where subpopulations evolve independently and synchronization is required only during the propagation phase. To reduce synchronization overhead, communication is performed at predefined intervals rather than at every iteration. Load balancing is facilitated by the adopted termination strategies (ANY, MAJORITY, ALL), which allow the algorithm to proceed without requiring all subpopulations to terminate simultaneously, thereby minimizing idle waiting. In

particular, the ANY strategy enables early continuation when at least one subpopulation satisfies the stopping criterion. The computational overhead of the propagation mechanism remains limited, as it involves only the exchange of a small number of elite solutions among subpopulations. This cost is negligible compared to the objective function evaluations and is compensated by the improved convergence behavior achieved through effective information sharing.

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

Figure 2 illustrates the creation of parallel processes and the information propagation mechanism in the parallel BGWO framework. Each thread $1, 2, \ldots, N$ corresponds to an independent processing unit (PU), which manages a subpopulation. The subpopulations evolve simultaneously, exploring different regions of the search space through the basic mechanisms of BGWO. Parallel execution allows for independent search for solutions, while at the same time supporting cooperation through propagation mechanisms. During this phase, selected high-quality solutions are transferred between subpopulations according to predefined exchange strategies. When a subpopulation finds a particularly good solution, its propagation to other subpopulations can accelerate the overall convergence. Overall, the combination of parallel exploration and controlled information exchange makes parallel BGWO more efficient, allowing a better balance between exploration and exploitation.

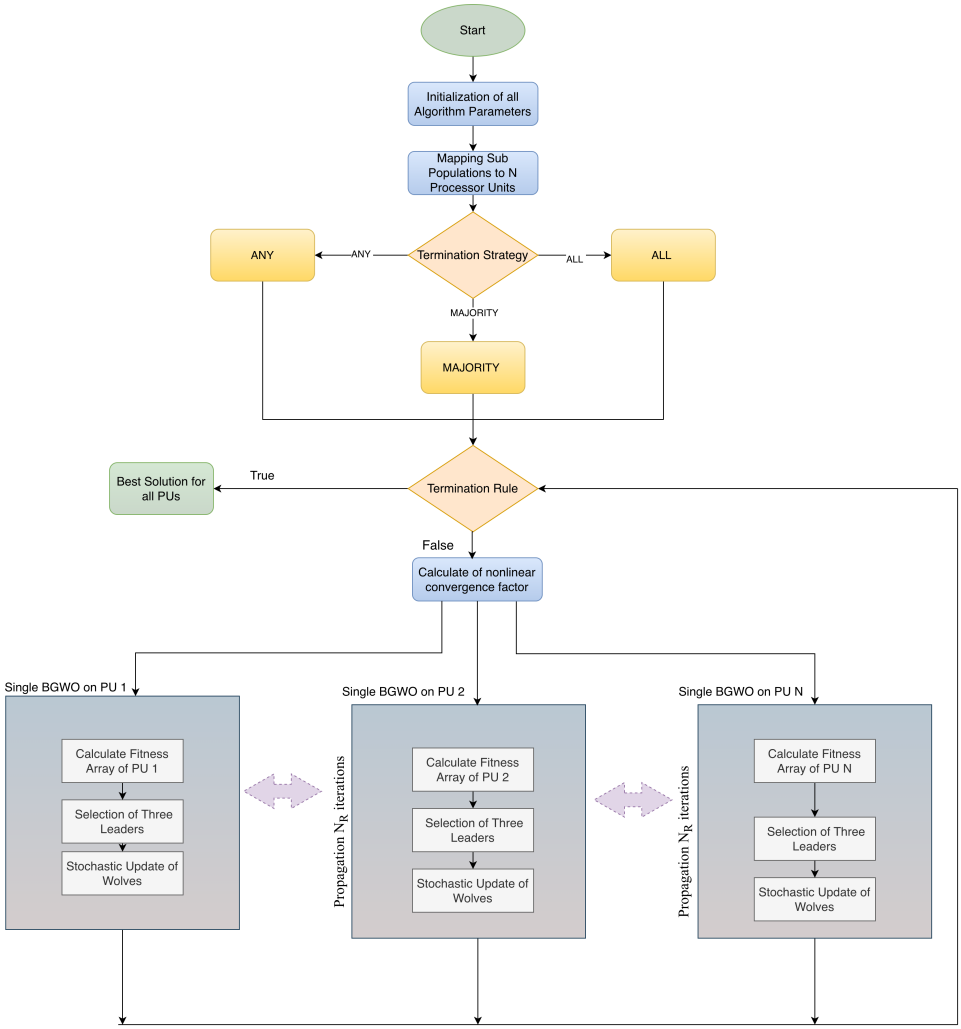

Figure 1. Flowchart of parallel BGWO.

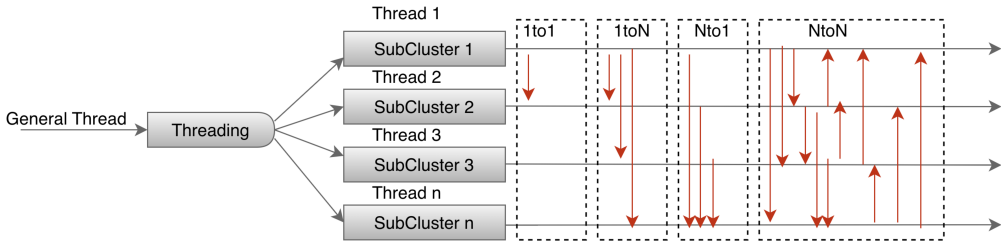

Figure 2. Thread creation diagram and propagation methods. The red arrows denote propagation between threads.

3.3. The Propagation Mechanism

During the propagation mechanism, the best values identified by each subpopulation are shared with the others, replacing their worst values. The goal of this process is to ensure that the subpopulations exchange their optimal findings, thereby enhancing the overall evolutionary process. Unlike conventional migration topologies (ring, star, or fully connected), the proposed propagation mechanism adopts a flexible and lightweight communication strategy tailored for large-scale optimization problems. These traditional topologies rely on fixed communication patterns, which may introduce increased synchronization and communication overhead as the number of subpopulations grows. In the proposed framework, simplified propagation strategies were adopted instead of conventional migration topologies, such as ring, star, or fully connected communication schemes, in order to preserve structural simplicity and reduce communication overhead within the parallel architecture. Although fixed migration topologies are commonly employed in island-model parallel metaheuristics, their use may increase synchronization and communication requirements as the number of subpopulations grows. The selected propagation mechanisms provide a lightweight and flexible communication scheme, enabling efficient information exchange while maintaining the computational efficiency of the proposed framework. The investigation of more sophisticated migration topologies constitutes an interesting direction for future research. The following four scenarios describe how propagation can take place:

1. One-to-One (1to1): In this case, a randomly selected subpopulation sends its best values to another subpopulation, also chosen at random.
2. One-to-All (1toN): Here, a random subpopulation shares its best values with all other subpopulations.
3. All-to-One (Nto1): All subpopulations send their best values to a single subpopulation, which is chosen at random.
4. All-to-All (NtoN): Each subpopulation communicates its best values to every other subpopulation.

This mechanism facilitates information exchange among subpopulations, increasing the likelihood of discovering optimal solutions through collaboration and communication.

3.4. Computational Complexity Analysis

The computational complexity of the proposed ParallelBGWO algorithm is primarily determined by the evaluation of the objective function and the propagation mechanism among subpopulations. Assuming a population size of n, problem dimensionality D, and a maximum number of iterations t, the computational complexity of the sequential BGWO algorithm can be approximated as $O(nDt)$ since each search agent updates all decision variables during every iteration. In the parallel formulation, when the population is divided into N subpopulations executed concurrently, the computational workload assigned to each processing unit is reduced to approximately $O\left(\frac{nDt}{N}\right)$, excluding communication overhead. The additional cost introduced by the propagation mechanism depends on the

selected communication strategy and the number of propagated agents N_P, resulting in an overhead of $O(N_P N)$ per propagation phase in the worst-case scenario. Consequently, the proposed parallel architecture exhibits near-linear workload reduction as the number of processing units increases, provided that communication overhead remains sufficiently limited. Therefore, the proposed framework improves computational efficiency by distributing the optimization workload across multiple processing units while maintaining acceptable communication cost. This theoretical analysis is consistent with the observed reduction in the number of objective function evaluations reported in the experimental results, which serve as a proxy for computational cost in black-box optimization problems.

4. Experimental Section

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 their reliability and adequacy.

4.1. Test Functions

A variety of benchmark test functions were used in the conducted experiments. These functions have been widely adopted in previous studies [45–48]. In the present work, the test functions are evaluated using dimensionalities ranging from 25 to 150, where the constant n denotes the dimension of the objective function. The benchmark test functions used in the experimental study are summarized in Table 2, including their mathematical formulation dimensionality and global optimum values.

Table 2. Benchmark test functions used in experimental study.

Name	Formula	Dim	Bounds	G_{min}
ATTRACTIVE SECTOR	$f(x) = \left(\sum_{i=1}^{n}(s_i x_i)^2\right)^{0.9}$	n	$[-5,5]^n$	0
BUCHE RASTRIGIN	$f(\mathbf{x}) = \sum_{i=1}^{n}[z_i \cdot (1 + 0.1 \cdot \sin(10\pi z_i))]$	n	$[-5.12,5.12]^n$	0
DIFFERENT POWERS	$f(\mathbf{x}) = \sqrt{\sum_{i=1}^{n}\lvert x_i\rvert^{2+4\frac{i-1}{n-1}}}$	n	$[-5,5]^n$	0
DISCUS	$f(x) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$	n	$[-5,5]^n$	0
ELLIPSOIDAL	$f(x) = \sum_{i=1}^{n}(10^6)^{\frac{i-1}{n-1}} x_i^2$	n	$[-5.12,5.12]^n$	0
GALLAGHER101	$f(\mathbf{x}) = \max_{i=1}^{101}\left[h_i - w_i\sqrt{\sum_{j=1}^{n}(x_j - c_{ij})^2}\right]$ $min = 101$	n	$[-5,5]^n$	0
GALLAGHER21	$f(\mathbf{x}) = \max_{i=1}^{21}\left[h_i - w_i\sqrt{\sum_{j=1}^{n}(x_j - c_{ij})^2}\right]$ $min = 21$	n	$[-5,5]^n$	0
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	$[-5,5]^n$	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	$[-100,100]^n$	0
RASTRIGIN	$f(\mathbf{x}) = An + \sum_{i=1}^{n}\left[x_i^2 - A\cos(2\pi x_i)\right]$ $A = 10$	n	$[-5.12,5.12]^n$	0
ROSENBROCK	$f(x) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right)$, $-30 \le x_i \le 30$	n	$[-30,30]^n$	0
SHARP RIDGE	$f(\mathbf{x}) = x_1^2 + \alpha\sum_{i=2}^{n} x_i^2$, $a > 1$	n	$[-5,5]^n$	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	$[-5,5]^n$	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	$[-5,10]^n$	0

4.2. Experimental Results

A series of experiments were carried out for the previously mentioned functions, and these experiments were executed on an AMD RYZEN 5950X with 128GB RAM. 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 C++ using the freely available optimization environment of OPTIMUS [49], which can be downloaded from https://github.com/itsoulos/OPTIMUS (accessed on 25 May 2026). The adoption of the publicly available OPTIMUS optimization framework improves the reproducibility of the experimental study and facilitates independent verification of the reported results by other researchers. All comparative methods were independently implemented and evaluated within the same experimental framework and were not directly adopted from the corresponding literature. All algorithms were implemented in C++ and executed under identical hardware and software conditions to ensure a fair and consistent comparison. Specifically, the number of agents (population size) was set to 200 for all algorithms, the maximum number of iterations was fixed at 200, and the local search rate was uniformly set to 5% (local search probability). These shared settings ensure that performance differences arise from algorithmic design rather than differences in experimental configuration. Algorithm-specific parameters were selected according to standard practices reported in the literature and were kept constant throughout all experiments. No problem-specific parameter tuning was applied in order to avoid bias and to maintain methodological consistency. In the conducted experiments, the propagation parameters were set to ($N_R = 1$) and ($N_P = 1$), corresponding to a minimal communication configuration. This choice was made to limit communication overhead while preserving the benefits of information exchange among subpopulations. The values used for each parameter are shown in Table 3.

Table 3. The values for the experimental parameters.

Parameter	Value	Explanation
n	200	Number of populations (wolves)
$iter_{max}$	200	Maximum number of iterations
N	1, 3, 10, 20, 30	Number of subpopulations
SR	$\delta_k^{(iter)} = \left\| f_{k,min}^{(iter)} - f_{k,min}^{(iter-1)} \right\|$	Stopping rule: Similarity
P_L	0.05 (5%)	Local search rate
N_M	1to1, 1toN, Nto1, NtoN	Propagation method
N_R	1	Number of iterations for propagation
N_P	1	Number of agents for propagation
N_T	$1, 2, 3, 4 \ldots \leq N$	Number of subpopulations required to satisfy the stopping criterion
F	0.8	Differential weight for DE
CR	0.9	Crossover probability
N_I	8	Number of iterations used in the termination rule
-	0.05 (5%)	Mutation rate for GA
-	0.05 (5%)	Selection rate for GA
-	Roulette	Selection method for GA

In the following tables, the reported values correspond to the average number of function evaluations over 30 independent runs. The values in parentheses denote the proportion of runs in which the global minimum was successfully identified. The absence of such values indicates a success rate of 100%.

4.3. Impact of the Number of Clusters

Table 4 presents a comparative analysis of the average number of objective function evaluations obtained for different numbers of subpopulations (1, 3, 10, 20, and 30) across benchmark problems of dimensions 25, 50, 100, and 150. Lower values indicate improved computational efficiency. The results show that increasing the number of clusters generally reduces the required number of function evaluations. For example, in Büche Rastrigin_100 and Different Powers_100, the use of a single cluster leads to higher computational costs of 17,769 and 23,887, respectively, while improved performance is observed in higher cluster configurations. This behavior indicates that partitioning the population into multiple subpopulations enhances diversity and promotes parallel exploration of different regions of the search space, thereby reducing premature convergence. It is also observed that configurations with 10 and 20 clusters consistently achieve the lowest or comparably low function call values, such as 6828 and 6837 for Büche Rastrigin_150 and 12,003 and 12,858 for Different Powers_150, respectively, indicating a more effective balance between exploration and exploitation. In addition to reducing the number of objective function evaluations, these intermediate configurations also improve solution quality. Specifically, configurations with 10 and 20 subpopulations achieve lower final objective values and reduced variance across independent runs, indicating more stable convergence behavior compared to both smaller and larger cluster settings. On the contrary, further increasing the number of clusters to 30 does not lead to additional improvements, and in several cases, results in increased computational cost. This performance degradation can be attributed to excessive population fragmentation, which reduces the number of agents per subpopulation and weakens the exploitation capability within each cluster. As a result, the search process becomes overly distributed and less effective in refining promising regions, leading to slower convergence and reduced solution quality. Overall, the total number of function calls decreases from 384,022 (1 cluster) to 363,819 (3 clusters) and further to approximately 173,392–174,401 for 10–20 clusters. In contrast, the cost increases again to 228,652 for 30 clusters. These results indicate that the cooperative structure of multiple subpopulations can significantly reduce computational cost provided that an appropriate balance in the number of clusters is maintained.

Table 4. Comparison of different numbers of subpopulations in terms of average objective function evaluations.

Function	1 Cluster	3 Cluster	10 Cluster	20 Cluster	30 Cluster
ATTRACTIVE SECTOR_25	835	1292	1171	1170	1106
ATTRACTIVE SECTOR_50	1715	1851	1549	1534	1481
ATTRACTIVE SECTOR_100	2319	2261	1818	1809	1833
ATTRACTIVE SECTOR_150	2123	2113	1754	1754	1722
BUCHE RASTRIGIN_25	3077	4451	1606	1578	1959
BUCHE RASTRIGIN_50	10,475	9890	4514	4420 (0.97)	5220 (0.90)
BUCHE RASTRIGIN_100	17,769	15,782	7302	7006 (0.97)	10,690 (0.90)
BUCHE RASTRIGIN_150	17,275	15,319	6828	6837 (0.97)	10,623 (0.97)
DISCUS_25	914	1391	1221	1153	1116
DISCUS_50	1901	2013	1513	1549	1428
DISCUS_100	2569	2373	1827	1636	1503
DISCUS_150	2346	2250	1776	1694	1476

Table 4. *Cont.*

Function	1 Cluster	3 Cluster	10 Cluster	20 Cluster	30 Cluster
DIFFERENTPOWERS_25	3815	6022	1623	2123	2354
DIFFERENTPOWERS_50	12,072	12,440	6920	6567	7469
DIFFERENTPOWERS_100	23,887	21,327	11,816	12,344	15,667
DIFFERENTPOWERS_150	27,185	23,196	12,003	12,858	16,164
ELLIPSOIDAL_25	1918	3079	1216	1216	1354
ELLIPSOIDAL_50	7804	6986	2176	2173	2736
ELLIPSOIDAL_100	19,182	14,093	3833	3843	5923
ELLIPSOIDAL_150	23,940	17,710	4341	4518	6607
GALLAGHER21_25	1092	1649	1397	1376	1334
GALLAGHER21_50	3028	3257	3095	2740 (0.93)	2080 (0.83)
GALLAGHER21_100	4346	4051 (0.93)	3493 (0.83)	3157 (0.80)	2588 (0.67)
GALLAGHER21_150	2812 (0.80)	2300 (0.70)	2472 (0.73)	2087 (0.63)	1946 (0.63)
GALLAGHER101_25	1142 (0.93)	1768 (0.93)	1209 (0.93)	1252 (0.93)	1232 (0.93)
GALLAGHER101_50	3370 (0.83)	3524 (0.80)	2447 (0.73)	2612 (0.77)	2717 (0.87)
GALLAGHER101_100	4817	4769	4235 (0.93)	3651 (0.87)	4028 (0.90)
GALLAGHER101_150	4304	4195	3742 (0.97)	3688 (0.97)	3832 (0.93)
GRIEWANK _25	2456	3684	1399	1345	1669
GRIEWANK _50	4822	4831	2675	2612	2676
GRIEWANK _100	7646	6444	4111	4239	4096
GRIEWANK _150	7198	6136	3897	3704	3838
GRIEWANK_ROSENBROCK_25	3070	4220	1387	1434	1947
GRIEWANK_ROSENBROCK_50	6306	6245	2696	2446	4150
GRIEWANK_ROSENBROCK_100	9049	7884	3686	3641	6283
GRIEWANK_ROSENBROCK_150	8528	7502	3463	3394	6108
ROSENBROCK_25	3494	4831	1364	1311	1915
ROSENBROCK_50	9279	9174	2842	2944	4367
ROSENBROCK_100	17,258	15,410	2842	5602	9713
ROSENBROCK_150	19,511	18,071	5628	6356	10,792
RARSTIGIN_25	2486	3451 (0.97)	1535	1542	1907
RARSTIGIN_50	6278	6155 (0.97)	3888	3461 (0.93)	3876 (0.90)
RARSTIGIN_100	9052	8404	5608	5318	6810 (0.97)
RARSTIGIN_150	8088	7444	4942	4861 (0.97)	5742 (0.93)
STEP ELLIPSOIDAL_25	666	1032	1154	1160	1092
STEP ELLIPSOIDAL_50	1273	1476	1507	1500	1419
STEP ELLIPSOIDAL_100	1676	1697	1737	1745	1696
STEP ELLIPSOIDAL_150	1529	1633	1698	1691	1600
SHARP RIDGE_25	3047	4652	1372	1329	1668
SHARP RIDGE_50	6643	6420	2322	2184	2969
SHARP RIDGE_100	9566	8344	3651	3345	4752
SHARP RIDGE_150	8976	7779	3034	2880	3891
ZAKHAROV_25	1768	2040	1355	1351	1617
ZAKHAROV_50	4141	4001	1819	1830	3488
ZAKHAROV_100	5050	6268	1344	1318	5381
ZAKHAROV_150	7134	7239	1539	1513	5002
	384,022 (0.99)	363,819 (0.99)	173,392 (0.99)	174,401 (0.98)	228,652 (0.97)

The diagram of Figure 3 shows the effect of the number of clusters on the performance of the algorithm. Specifically, the cases of 1, 3, 10, 20, and 30 clusters are compared. It is observed that the use of more subpopulations reduces the number of function calls, with the cases of 10 and 20 clusters showing the smallest values. On the contrary, when only 1 cluster is used, the method shows greater dispersion and higher values, which indicates lower efficiency. The Kruskal–Wallis statistical test ($p = 0.0000013$) shows that there are statistically significant differences between different numbers of clusters. Therefore, the results show that using an appropriate number of subpopulations can improve the performance and convergence speed of the algorithm.

4.4. Impact of Propagation Mechanism

Table 5 presents the comparative evaluation of the proposed method with and without the propagation mechanism (PROP = 0 and PROP = 1) in terms of average objective function evaluations over benchmark functions of dimensions 25, 50, 100, and 150. The PROP parameter indicates the presence or absence of the information propagation mechanism between the subpopulations: the case of PROP = 0 corresponds to execution without propagation, while the case of PROP = 1 corresponds to execution with propagation enabled. The results show that the activation of the propagation mechanism leads, in almost all the examined functions and dimensions, to a reduction in the number of evaluations compared to the corresponding execution without propagation. For example, in the Büche Rastrigin_150 function, the number of evaluations is reduced from 22,453 to 6828, in the Sharp Ridge_150 from 10,708 to 3034, while in the Zakharov_150 from 9869 to 1539. The improvement is correspondingly significant in lower dimensions, which suggests that the propagation mechanism not only favors large-scale problems but also improves the overall behavior of the method. The total number of function calls is reduced from about 493,939 in the case without propagation (PROP = 0) to about 173,392 in the case with propagation (PROP = 1). This difference corresponds to a significant reduction in the overall computational cost and confirms that information propagation between subpopulations is an important mechanism for improving the efficiency of the proposed method. Overall, the results show that the integration of the propagation mechanism enhances the computational behavior of the algorithm, reducing the number of required objective function evaluations and improving efficiency.

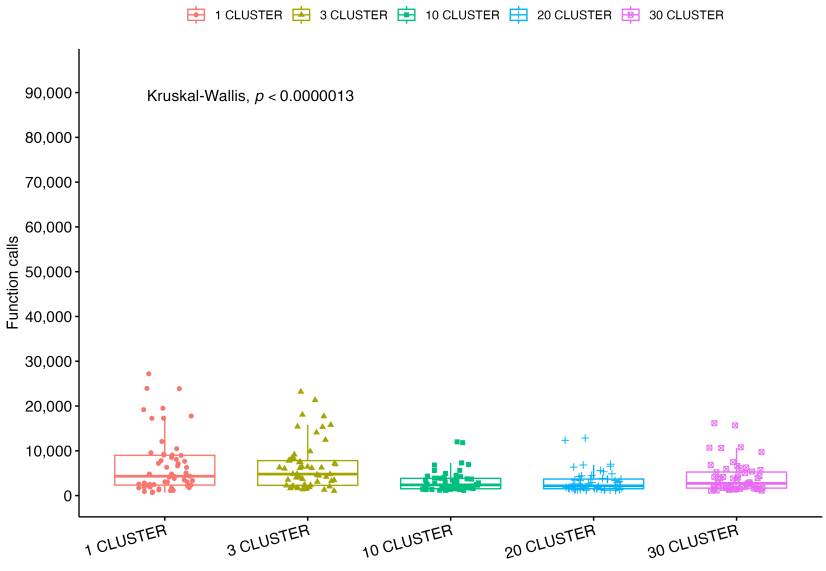

Figure 3. Impact of the number of clusters.

Table 5. Comparison of the proposed method with and without the propagation mechanism in terms of average objective function evaluations.

Function	PROP = 0	PROP = 1
ATTRACTIVE SECTOR_25	1745	1171
ATTRACTIVE SECTOR_50	2162	1549
ATTRACTIVE SECTOR_100	2414	1818
ATTRACTIVE SECTOR_150	2440	1754
BUCHE RASTRIGIN_25	8599	1606
BUCHE RASTRIGIN_50	14,282	4514
BUCHE RASTRIGIN_100	20,788	7302
BUCHE RASTRIGIN_150	22,453 (0.97)	6828
DISCUS_25	1809	1221
DISCUS_50	2022	1513
DISCUS_100	2392	1827
DISCUS_150	2349	1776
DIFFERENTPOWERS_25	9905	1623
DIFFERENTPOWERS_50	16,094	6920
DIFFERENTPOWERS_100	26,466	11,816
DIFFERENTPOWERS_150	33,123	12,003
ELLIPSOIDAL_25	5003	1216
ELLIPSOIDAL_50	9973	2176
ELLIPSOIDAL_100	20,082	3833
ELLIPSOIDAL_150	27,981	4341
GALLAGHER21_25	2765	1397
GALLAGHER21_50	3679 (0.97)	3095
GALLAGHER21_100	2044 (0.53)	3493 (0.83)
GALLAGHER21_150	667 (0.40)	2472 (0.73)
GALLAGHER101_25	2288 (0.93)	1209 (0.93)
GALLAGHER101_50	3504 (0.83)	2447 (0.73)
GALLAGHER101_100	4991 (0.90)	4235 (0.93)
GALLAGHER101_150	5032	3742 (0.97)
GRIEWANK _25	5876	1399
GRIEWANK _50	6446	2675
GRIEWANK _100	7921	4111
GRIEWANK _150	8657	3897
GRIEWANK_ROSENBROCK_25	6456	1387
GRIEWANK_ROSENBROCK_50	8186	2696
GRIEWANK_ROSENBROCK_100	9765	3686
GRIEWANK_ROSENBROCK_150	10,496	3463
ROSENBROCK_25	8468	1364
ROSENBROCK_50	12,829	2842
ROSENBROCK_100	20,141	2842
ROSENBROCK_150	25,818	5628

Table 5. *Cont.*

Function	PROP = 0	PROP = 1
RARSTIGIN_25	6901	1535
RARSTIGIN_50	8901	3888
RARSTIGIN_100	10,440	5608
RARSTIGIN_150	10,149	4942
STEP ELLIPSOIDAL_25	1351	1154
STEP ELLIPSOIDAL_50	1623	1507
STEP ELLIPSOIDAL_100	1769	1737
STEP ELLIPSOIDAL_150	1720	1698
SHARP RIDGE_25	7434	1372
SHARP RIDGE_50	9092	2322
SHARP RIDGE_100	10,380	3651
SHARP RIDGE_150	10,708	3034
ZAKHAROV_25	2631	1355
ZAKHAROV_50	4674	1819
ZAKHAROV_100	8186	1344
ZAKHAROV_150	9869	1539
	493,939 (0.97)	173,392 (0.99)

Figure 4 shows the comparison of the method when the propagation function is enabled and disabled. Specifically, PROP = 1 indicates that propagation is enabled while PROP = 0 indicates that the method is executed without propagation. It is observed that when propagation is enabled (PROP = 1), the method requires significantly fewer function calls and exhibits lower dispersion, which indicates faster convergence. The comparison is performed using the paired Wilcoxon statistical test, following the conventional star notation (ns: $p > 0.05$, *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$, ****: $p < 0.0001$), where the overall result ($p < 0.0001$) shows that the difference between the two cases is significant. Therefore, the results show that enabling propagation improves the efficiency of the proposed method.

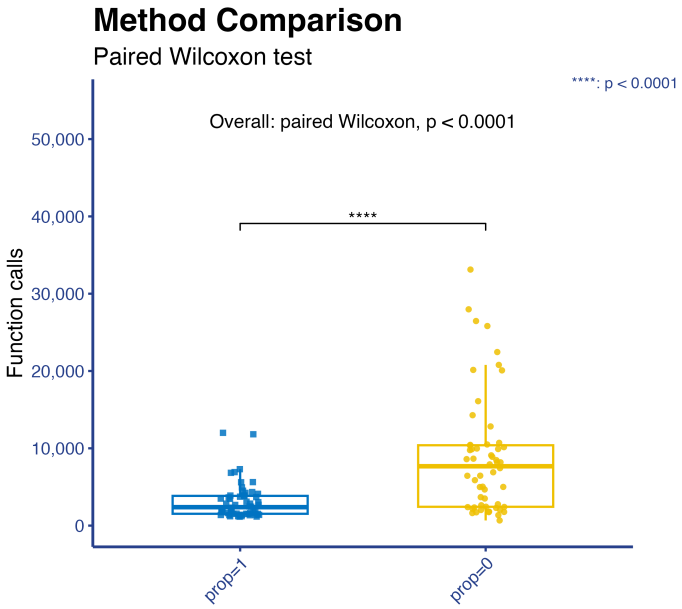

Figure 4. Impact of propagation mechanism.

It should be noted, however, that the propagation mechanism introduces additional communication overhead which is not explicitly reflected in the number of function evaluations. Nevertheless, this overhead remains relatively low, as it mainly involves the exchange of a limited number of high-quality solutions and is therefore outweighed by the significant gains in convergence efficiency observed in the experiments. This observation is consistent with the overall reduction in computational cost reported across the benchmark problems.

4.5. Communication Strategy with 10 Subpopulations

Table 6 reports the average number of objective function evaluations obtained under four different propagation strategies (1to1, 1toN, Nto1 and NtoN) when 10 subpopulations are employed. Lower values indicate better performance with the best-performing value in each row highlighted in bold for clarity. The results show that the NtoN strategy exhibits the best computational behavior overall and leads to the smallest function call values in the vast majority of the functions examined. For example, in the Büche Rastrigin_150 function, the NtoN strategy requires 6828 function calls compared to 18,787 for the 1to1 strategy, 10,340 for 1toN, and 11,452 for Nto1. Correspondingly, in Zakharov_150, the NtoN strategy requires only 1539 function calls while the 1to1, 1toN, and Nto1 strategies require 9158, 4352, and 4690, respectively. Therefore, the fully bidirectional information dissemination between subpopulations enhances cooperation and facilitates faster diffusion of useful information in the population. The 1toN and Nto1 strategies also show improved behavior compared to the basic 1to1 approach. The total number of function calls is 432,256 for the 1to1 strategy, 254,443 for 1toN, 274,853 for Nto1, and only 173,392 for NtoN. These results show that, for the case of 10 subpopulations, the NtoN strategy is the most efficient choice as it achieves the lowest overall computational requirement.

Table 6. Comparison of propagation strategies under 10 subpopulations in terms of average objective function evaluations.

Function	1to1	1toN	Nto1	NtoN
ATTRACTIVE SECTOR_25	1602	1348	1349	**1171**
ATTRACTIVE SECTOR_50	2027	1648	1698	**1549**
ATTRACTIVE SECTOR_100	2271	1912	1944	**1818**
ATTRACTIVE SECTOR_150	2282	1859	1914	**1754**
BUCHE RASTRIGIN_25	7204	3327	3521	**1606**
BUCHE RASTRIGIN_50	11,493 (0.93)	6203	6789 (0.97)	**4514**
BUCHE RASTRIGIN_100	17,882	10,472	11,567	**7302**
BUCHE RASTRIGIN_150	18,787 (0.93)	10,340	11,452 (0.97)	**6828**
DISCUS_25	1726	1359	1392	**1221**
DISCUS_50	1923	1517	1706	**1513**
DISCUS_100	2059	1911	**1691**	1827
DISCUS_150	2088	1822	**1743**	1776
DIFFERENTPOWERS_25	8481	3981	4630	**1623**
DIFFERENTPOWERS_50	14,513	9641	9649	**6920**
DIFFERENTPOWERS_100	23,558	15,731	17,144	**11,816**
DIFFERENTPOWERS_150	28,561	16,633	18,553	**12,003**

Table 6. *Cont.*

Function	1to1	1toN	Nto1	NtoN
ELLIPSOIDAL_25	4112	2151	2353	**1216**
ELLIPSOIDAL_50	8553	3614	4283	**2176**
ELLIPSOIDAL_100	16,965	6082	7615	**3833**
ELLIPSOIDAL_150	22,798	7751	9556	**4341**
GALLAGHER21_25	2319	1692	1689	**1397**
GALLAGHER21_50	3201 (0.93)	3137 (0.97)	**2914 (0.93)**	3095
GALLAGHER21_100	2419 (0.60)	3273 (0.77)	**2476 (0.63)**	3493 (0.83)
GALLAGHER21_150	**1079 (0.47)**	1507 (0.53)	1245 (0.50)	2472 (0.73)
GALLAGHER101_25	2148	1580 (0.97)	1636	**1209 (0.93)**
GALLAGHER101_50	3777 (0.97)	3572 (0.97)	3335 (0.93)	**2447 (0.73)**
GALLAGHER101_100	5272	4891	4729 (0.97)	**4235 (0.93)**
GALLAGHER101_150	4785	4282	4429	**3742 (0.97)**
GRIEWANK _25	5277	2780	3232	**1399**
GRIEWANK _50	5682	3384	3679	**2675**
GRIEWANK _100	6813	4314	4799	**4111**
GRIEWANK _150	7549	4274	4541 (0.87)	**3897**
GRIEWANK_ROSENBROCK_25	5722	2966	3467 (0.97)	**1387**
GRIEWANK_ROSENBROCK_50	7311	4285	4700 (0.97)	**2696**
GRIEWANK_ROSENBROCK_100	8723	5430	6023	**3686**
GRIEWANK_ROSENBROCK_150	9517	5367	5802	**3463**
ROSENBROCK_25	7668	3471	4288	**1364**
ROSENBROCK_50	11,170	5907	6752	**2842**
ROSENBROCK_100	17,250	10,773	11,768	**2842**
ROSENBROCK_150	23,256	12,622	14,393	**5628**
RARSTIGIN_25	5500	2935	3295	**1535**
RARSTIGIN_50	7523	5364	4541	**3888**
RARSTIGIN_100	8597 (0.97)	6728	6790	**5608**
RARSTIGIN_150	8784 (0.97)	6025 (0.97)	6138	**4942**
STEP ELLIPSOIDAL_25	1328	1205	1207	**1154**
STEP ELLIPSOIDAL_50	1593	1530	1524	**1507**
STEP ELLIPSOIDAL_100	1742	**1735**	1743	1737
STEP ELLIPSOIDAL_150	1713	1701	1723	**1698**
SHARP RIDGE_25	6833	3248	3390	**1372**
SHARP RIDGE_50	7629	3917	4348	**2322**
SHARP RIDGE_100	8889	4814	5363	**3651**
SHARP RIDGE_150	9656	4522	5010	**3034**
ZAKHAROV_25	2407	1682	1850	**1355**
ZAKHAROV_50	4004	2579	2651	**1819**
ZAKHAROV_100	7077	3297	4144	**1344**
ZAKHAROV_150	9158	4352	4690	**1539**
	432,256 (0.98)	254,443 (0.99)	274,853 (0.98)	173,392 (0.99)

Figure 5 shows the comparison of different communication modes between subpopulations when 10 clusters are used, based on the number of function calls. The configurations examined are 1to1, 1toN, Nto1, and NtoN, which express different information dissemination schemes between subpopulations. It is observed that the NtoN configuration presents the lowest values and the smallest dispersion, which indicates faster convergence of the algorithm. On the contrary, the 1to1 case presents a higher number of function calls. The overall statistical analysis was performed with the Friedman test ($p < 0.0001$), while the individual comparisons were performed with paired Wilcoxon tests following the conventional star notation (ns: $p > 0.05$, *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$, ****: $p < 0.0001$). The presence of **** ($p < 0.0001$) indicates that the differences between the methods are statistically significant. The results suggest that the exchange of information between all subpopulations is significant.

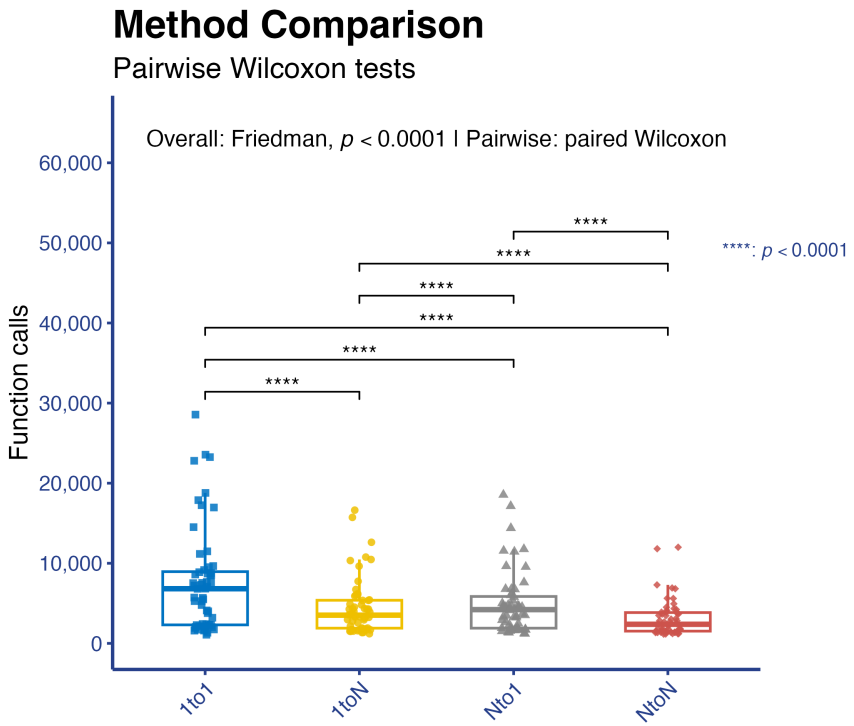

Figure 5. Communication strategy with 10 subpopulations.

4.6. Communication Strategy with 20 Subpopulations

Table 7 presents the corresponding results for the same propagation strategies when 20 subpopulations are utilized. Lower values indicate better performance, with the best-performing value in each row highlighted in bold for clarity. The results show that, also in the case of 20 subpopulations, the NtoN strategy displays the overall most efficient behavior, as it achieves the lowest or among the lowest function call values. Indicatively, in the Rosenbrock_150 function, the NtoN strategy requires 6356 function calls, compared to 23,645 for 1to1, 8431 for 1toN, and 11,263 for Nto1. Accordingly, in Rastrigin_150, the required evaluations are reduced to 4861 for NtoN compared to 8674, 5954, and 5671 for the 1to1, 1toN, and Nto1 strategies, respectively. These results suggest that the fully cooperative form of propagation significantly accelerates the diffusion of useful information between subpopulations and facilitates convergence towards better regions of the search space. The 1toN and Nto1 strategies also show improved behavior compared to the basic 1to1 approach, which confirms that even partial information propagation is beneficial for the search process. The total number of function calls is 414,978 for the 1to1 strategy, 195,980 for 1toN, 222,989 for Nto1, and only 174,401 for NtoN. These results show that, for the case of

20 subpopulations, the NtoN strategy is again the most efficient choice, as it achieves the lowest overall computational requirement.

Table 7. Comparison of propagation strategies under 20 subpopulations in terms of average objective function evaluations.

Function	1to1	1toN	Nto1	NtoN
ATTRACTIVE SECTOR_25	1663	1317	1271	**1170**
ATTRACTIVE SECTOR_50	2026	1582	1644	**1534**
ATTRACTIVE SECTOR_100	2247	1826	1893	**1809**
ATTRACTIVE SECTOR_150	2231	1797	1898	**1754**
BUCHE RASTRIGIN_25	7185	2674	2768	**1578**
BUCHE RASTRIGIN_50	11,071 (0.97)	5436 (0.97)	5108 (0.87)	**4420 (0.97)**
BUCHE RASTRIGIN_100	15,630 (0.90)	7698 (0.97)	8794 (0.83)	**7006 (0.97)**
BUCHE RASTRIGIN_150	17,743 (0.97)	7296 (0.90)	8690 (0.87)	**6837 (0.97)**
DISCUS_25	1721	1222	1330	**1153**
DISCUS_50	1833	1413	**1368**	1549
DISCUS_100	1803	1642	**1555**	1636
DISCUS_150	1927	1601	**1572**	1694
DIFFERENTPOWERS_25	9370	3382	3277	**2123**
DIFFERENTPOWERS_50	14,912	7296	7985	**6567**
DIFFERENTPOWERS_100	23,185	12,713	14,072	**12,344**
DIFFERENTPOWERS_150	27,829	13,781	15,957	**12,858**
ELLIPSOIDAL_25	4242	1600	1805	**1216**
ELLIPSOIDAL_50	7652	2879	3370	**2173**
ELLIPSOIDAL_100	14,835	4172	5690	**3843**
ELLIPSOIDAL_150	20,040	5562	6949	**4518**
GALLAGHER21_25	2347	1542	1582	**1376**
GALLAGHER21_50	2609 (0.80)	2656 (0.93)	**2031 (0.80)**	2740 (0.93)
GALLAGHER21_100	**1250 (0.40)**	2255 (0.60)	1725 (0.50)	3157 (0.80)
GALLAGHER21_150	**654 (0.40)**	1343 (0.50)	1019 (0.47)	2087 (0.63)
GALLAGHER101_25	2180	1383 (0.97)	1630	**1252 (0.93)**
GALLAGHER101_50	3554 (0.90)	2894 (0.87)	2899 (0.90)	**2612 (0.77)**
GALLAGHER101_100	4806 (0.87)	3942 (0.83)	3927 (0.83)	**3651 (0.87)**
GALLAGHER101_150	5034	4093 (0.97)	3804 (0.93)	**3688 (0.97)**
GRIEWANK _25	5255	2219	2341	**1345**
GRIEWANK _50	5545	3050	3292	**2612**
GRIEWANK _100	6280	**3760**	3823	4239
GRIEWANK _150	6933	**3596**	3887	3704
GRIEWANK_ROSENBROCK_25	5566	1851	2653	**1434**
GRIEWANK_ROSENBROCK_50	7156	3145	3754	**2446**
GRIEWANK_ROSENBROCK_100	8461	3983	4858	**3641**
GRIEWANK_ROSENBROCK_150	8923	3773	4887	**3394**

Table 7. *Cont.*

Function	1to1	1toN	Nto1	NtoN
ROSENBROCK_25	7197	1876	2995	**1311**
ROSENBROCK_50	10,930	3825	5192	**2944**
ROSENBROCK_100	17,190	6262	9295	**5602**
ROSENBROCK_150	23,645	8431	11,263	**6356**
RARSTIGIN_25	5727	1985	2346	**1542**
RARSTIGIN_50	6901 (0.93)	4141 (0.97)	4157 (0.93)	**3461 (0.93)**
RARSTIGIN_100	8313 (0.93)	5491 (0.90)	5433 (0.87)	**5318**
RARSTIGIN_150	8674 (0.97)	5954 (0.90)	5671 (0.97)	**4861 (0.97)**
STEP ELLIPSOIDAL_25	1342	1238	1198	**1160**
STEP ELLIPSOIDAL_50	1557	1516	1539	**1500**
STEP ELLIPSOIDAL_100	1772	**1733**	**1733**	1745
STEP ELLIPSOIDAL_150	1727	1705	1706	**1691**
SHARP RIDGE_25	6774	2150	2645	**1329**
SHARP RIDGE_50	7431	2899	3598	**2184**
SHARP RIDGE_100	8358	4010	4411	**3345**
SHARP RIDGE_150	8713	3680	4186	**2880**
ZAKHAROV_25	2338	1391	1557	**1351**
ZAKHAROV_50	4005	1845	2371	**1830**
ZAKHAROV_100	6815	1687	2836	**1318**
ZAKHAROV_150	9871	1787	3749	**1513**
	414,978 (0.97)	195,980 (0.97)	222,989 (0.96)	174,401 (0.98)

Figure 6 shows the comparison of different communication modes between subpopulations when 20 clusters are used, based on the number of function calls. The configurations examined are 1to1, 1toN, Nto1, and NtoN, which express different information dissemination schemes between subpopulations. It is observed that the NtoN configuration presents the lowest values and the smallest dispersion, which indicates faster convergence of the algorithm. On the contrary, the 1to1 case presents a higher number of function calls. The overall statistical analysis was performed with the Friedman test ($p < 0.0001$), while the individual comparisons were performed with paired Wilcoxon tests, following the conventional star notation (ns: $p > 0.05$, *: $p < 0.05$, **: $p < 0.01$, ***: $p < 0.001$, ****: $p < 0.0001$). The presence of **** ($p < 0.0001$) indicates that the differences between the methods are statistically significant. The results suggest that the exchange of information between all subpopulations is significant.

A combined analysis of the communication strategies for both 10 and 20 subpopulations shows that the NtoN scheme consistently achieves the best performance in terms of objective function evaluations. However, it should be noted that this conclusion is based primarily on evaluation efficiency, as runtime and communication overhead have not been explicitly measured. The improved performance of the NtoN strategy can be explained by its fully cooperative communication mechanism, which enables rapid dissemination of high-quality solutions across all subpopulations. This leads to more stable convergence behavior and increased robustness, as promising regions of the search space are more effectively exploited. In contrast, more limited communication schemes restrict information flow, resulting in slower convergence and higher variability. Although the NtoN strat-

egy introduces additional communication operations, this overhead is relatively small compared to the cost of objective function evaluations, especially in high-dimensional problems. Therefore, NtoN can be considered the most effective strategy in terms of solution quality and robustness, while its optimality should be interpreted within the context of evaluation-based performance rather than strict computational cost.

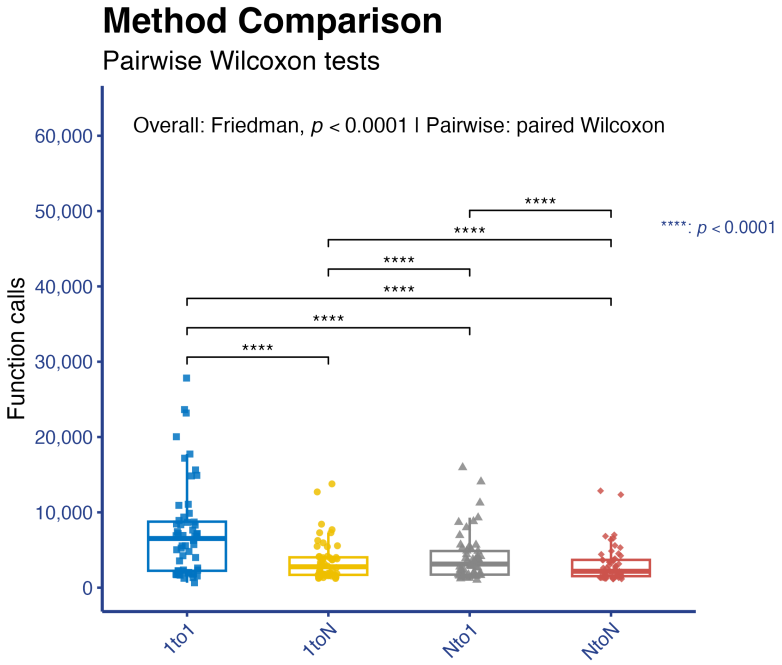

Figure 6. Communication strategy with 20 subpopulations.

4.7. The Proposed Method in Comparison with Other Methods

To provide a more rigorous and statistically sound evaluation of the proposed method, the results are reported in terms of the mean and standard deviation (mean ± std) of the final objective function values over 30 independent runs. This allows for a comprehensive assessment of both solution quality and robustness. The proposed method consistently achieves competitive or superior final objective values accompanied by low variance across independent runs. This indicates not only high accuracy but also stable and reliable convergence behavior compared to the examined algorithms.

Table 8 presents the comparative performance of the proposed method against established metaheuristic algorithms, namely GA, DE, GWO, ParallelDE, SAOP, BGWO, and WOA, in terms of average objective function evaluations over benchmark functions of dimensions 25, 50, 100, and 150. Lower values indicate superior optimization efficiency. The results show that the proposed method shows the best performance most of the time. Indicatively, in the Zakharov_150 function, the proposed method requires 1539 evaluations, compared to 38,858 for GA, 14,744 for DE, 3251 for GWO, 5613 for ParallelDE, 4341 for SAOP, 3333 for BGWO, and 11,943 for WOA. Correspondingly, on Sharp Ridge_150, the proposed method achieves 3034 function calls, compared to 12,266, 12,231, 25,399, 5614, 4055, 24,923, and 8022, respectively. Particularly important is the fact that the proposed approach maintains its efficiency not only in complex and multimodal problems but also in functions of different structures, presenting consistently low computational cost across the entire range of dimensions. The total number of function calls of the proposed method amounts to 173,392, compared to 777,899 for GA, 762,504 for DE, 772,866 for GWO, 314,144 for ParallelDE, 263,370 for SAOP, 724,617 for BGWO, and 371,962 for WOA. In conclusion,

it follows that the combination of the cooperative subpopulation structure, the information dissemination mechanism, and the overall parallelized search strategy substantially enhances the efficiency of the algorithm, making it particularly suitable for optimization and high-dimensional problems.

Table 8. Comparison of the proposed method against competing metaheuristic algorithms in terms of average objective function evaluations.

Function	GA	DE	GWO	PARALLELDE	SAOP	BGWO	WOA	PROPOSED
ATTRACTIVE SECTOR_25	2281	2234	3772	5594	1715	3239	1944	1171
ATTRACTIVE SECTOR_50	2297	2298	12556	5611	2106	17,575	4646	1549
ATTRACTIVE SECTOR_100	2344	2273	29011	5623	2691	28,735	6739	1818
ATTRACTIVE SECTOR_150	2345	2255	25348	5614	2618	24,872	6070	1754
BUCHE RASTRIGIN_25	11,233 (0.93)	11,417 (0.93)	4214	5592	2183 (0.93)	1989 (0.93)	3009 (0.97)	1606
BUCHE RASTRIGIN_50	17,589 (0.57)	21,873 (0.57)	13,509 (0.90)	5610	4657 (0.57)	4033 (0.57)	14,163 (0.70)	4514
BUCHE RASTRIGIN_100	32,881 (0.30)	39,401 (0.30)	23,867 (0.80)	5622	7806 (0.30)	7564 (0.30)	7293 (0.97)	7302
BUCHE RASTRIGIN_150	36,444 (0.40)	48,017 (0.40)	22,139 (0.87)	5611	7731 (0.40)	6472 (0.40)	5699	6828
DISCUS_25	2715	2612	4979	5596	1747	4266	2505	1221
DISCUS_50	2714	2701	19111	5614	2472	18,438	8241	1513
DISCUS_100	2752	2693	29,094	5624	2722	28,740	12,795	1827
DISCUS_150	2750	2671	25,351	5614	2483	24,876	11,185	1776
DIFFERENTPOWERS_25	13,758	14,154	3580	5595	2456	2728	1864	1623
DIFFERENTPOWERS_50	19,777	22,093	10,497	5612	8075	12,334	4211	6920
DIFFERENTPOWERS_100	28,988	29,193	25,884	5624	13,699	27,204	6439	11,816
DIFFERENTPOWERS_150	34,372	34,490	25,642	5613	15,676	25,413	5930	12,003
ELLIPSOIDAL_25	5930	5778	3598	5593	1847	2778	1843	1216
ELLIPSOIDAL_50	10,583	11,023	10,624	5611	2979	13,691	4127	2176
ELLIPSOIDAL_100	20,604	20,345	25,068	5623	5137	28,976	6014	3833
ELLIPSOIDAL_150	28,556	27,643	25,642	5612	5893	25,171	5653	4341
GALLAGHER21_25	3118 (0.93)	2867 (0.93)	3214 (0.93)	5595	2206 (0.93)	2115 (0.93)	3978 (0.93)	1397
GALLAGHER21_50	3272 (0.57)	3566 (0.57)	6234 (0.57)	5612	8557 (0.57)	6590 (0.57)	14,831 (0.57)	3095
GALLAGHER21_100	1693 (0.30)	1621 (0.30)	2901 (0.30)	5622	1691 (0.30)	8781 (0.30)	1502 (0.30)	3493 (0.83)
GALLAGHER21_150	1682 (0.40)	1617 (0.40)	2898 (0.40)	5613	1671 (0.40)	1703 (0.40)	1500 (0.40)	2472 (0.73)
GALLAGHER101_25	3083 (0.93)	3030 (0.93)	3106 (0.93)	5593	2068	1913 (0.93)	3531 (0.93)	1209 (0.93)
GALLAGHER101_50	6150 (0.57)	4617 (0.57)	5873 (0.57)	5602	6932	7429 (0.57)	13,266 (0.57)	2447 (0.73)
GALLAGHER101_100	6656 (0.30)	4340 (0.30)	8356 (0.30)	5606	18,872	12,116 (0.30)	18,537 (0.30)	4235 (0.93)
GALLAGHER101_150	6122 (0.40)	3236 (0.40)	7496 (0.40)	5601	19,066	14,717 (0.40)	15,220 (0.40)	3742 (0.97)
GRIEWANK _25	9514	9655	3695 (0.97)	5594	2237	2878 (0.97)	1893	1399
GRIEWANK _50	5511	5147	10,342 (0.97)	5610	3132	12,370 (0.83)	4184	2675
GRIEWANK _100	5078	4141	22,295	5623	4053	27,343 (0.87)	6532 (0.97)	4111
GRIEWANK _150	5284	4131	25,374	5612	3934	24,895 (0.93)	6098 (0.97)	3897
GRIEWANK_ROSENBROCK_25	16,098	17,837	3429	5595	2259	2435	1740	1387
GRIEWANK_ROSENBROCK_50	22,598	27,316	8659	5612	4413	9859	3361	2696
GRIEWANK_ROSENBROCK_100	30,869	35,225	19,410	5624	5856	21,301	4784	3686
GRIEWANK_ROSENBROCK_150	37,686	40,126	22,950	5613	6664	23,047	4450	3463
ROSENBROCK_25	15,148	15,199	3760	5595	2217	3320	1925	1364
ROSENBROCK_50	23,663	25,189	12,789	5612	4238	18057	4611	2842
ROSENBROCK_100	40,370	39,254	29,103	5623	7197	28,756	6786	2842
ROSENBROCK_150	53,927	51,902	25,367	5613	8677	24,893	6076	5628
RARSTIGIN_25	9348 (0.93)	9799 (0.93)	2959 (0.93)	5594	2067 (0.93)	2007 (0.93)	3007 (0.97)	1535
RARSTIGIN_50	11,414 (0.57)	13,738 (0.57)	13,386 (0.87)	5610	3871 (0.57)	4022 (0.57)	15,194 (0.67)	3888
RARSTIGIN_100	12,898 (0.30)	17,722 (0.30)	22,715 (0.77)	5622	4840 (0.30)	7611 (0.30)	10,638 (0.87)	5608
RARSTIGIN_150	14,392 (0.40)	19,506 (0.40)	18,708 (0.73)	5612	4693 (0.40)	6693 (0.40)	6827 (0.97)	4942

Table 8. *Cont.*

Function	GA	DE	GWO	PARALLELDE	SAOP	BGWO	WOA	PROPOSED
STEP ELLIPSOIDAL_25	1718 (0.93)	1664 (0.93)	3078	5593	1806 (0.93)	1831	1590	1154
STEP ELLIPSOIDAL_50	2201 (0.57)	1661 (0.57)	5163	5610	2369 (0.57)	3615 (0.90)	2273 (0.93)	1507
STEP ELLIPSOIDAL_100	2028 (0.30)	1629 (0.30)	10101 (0.97)	5622	2355 (0.30)	6624 (0.57)	2708 (0.97)	1737
STEP ELLIPSOIDAL_150	1752 (0.40)	1617 (0.40)	12778 (0.97)	5612	1977 (0.40)	6730 (0.44)	2527 (0.97)	1698
SHARP RIDGE_25	11,324	11,800	4098	5596	2124	4087	2110	1372
SHARP RIDGE_50	11,454	12,866	16,581	5612	3106	18,454	5907	2322
SHARP RIDGE_100	12,105	12,682	29,123	5624	3995	28,790	8936	3651
SHARP RIDGE_150	12,266	12,231	25,399	5614	4055	24,923	8022	3034
ZAKHAROV_25	5760	4939	8426	5598	2177	9075	10,581	1355
ZAKHAROV_50	15,253	8154	23,034	5611	3083	27,447	25,061	1819
ZAKHAROV_100	36,693	12,572	3329	5623	3878	5763	9463	1344
ZAKHAROV_150	38,858	14,744	3251	5613	4341	3333	11,943	1539
	777,899 (0.84)	762,504 (0.84)	772,866 (0.91)	314,144	263,370 (0.84)	724,617 (0.85)	371,962 (0.92)	173,392 (0.99)

Figure 7 shows the comparison of the performance of various optimization algorithms in terms of the number of function calls. Specifically, the comparison of the algorithms GENETIC, DE, GWO, PARALLELDE, SAOP, BGWO, PROPOSED, and WOA. It is observed that the proposed algorithm presents the smallest values of function calls and less variance compared to most algorithms; therefore, it has faster convergence.The GENETIC, DE, and BGWO algorithms show greater dispersion and higher values, while PARALLELDE shows low values. The Kruskal–Wallis statistical test ($p < 2.2 \times 10^{-16}$) shows that there are statistically significant differences between the algorithms. Overall, the results indicate that the proposed method is superior in terms of efficiency.

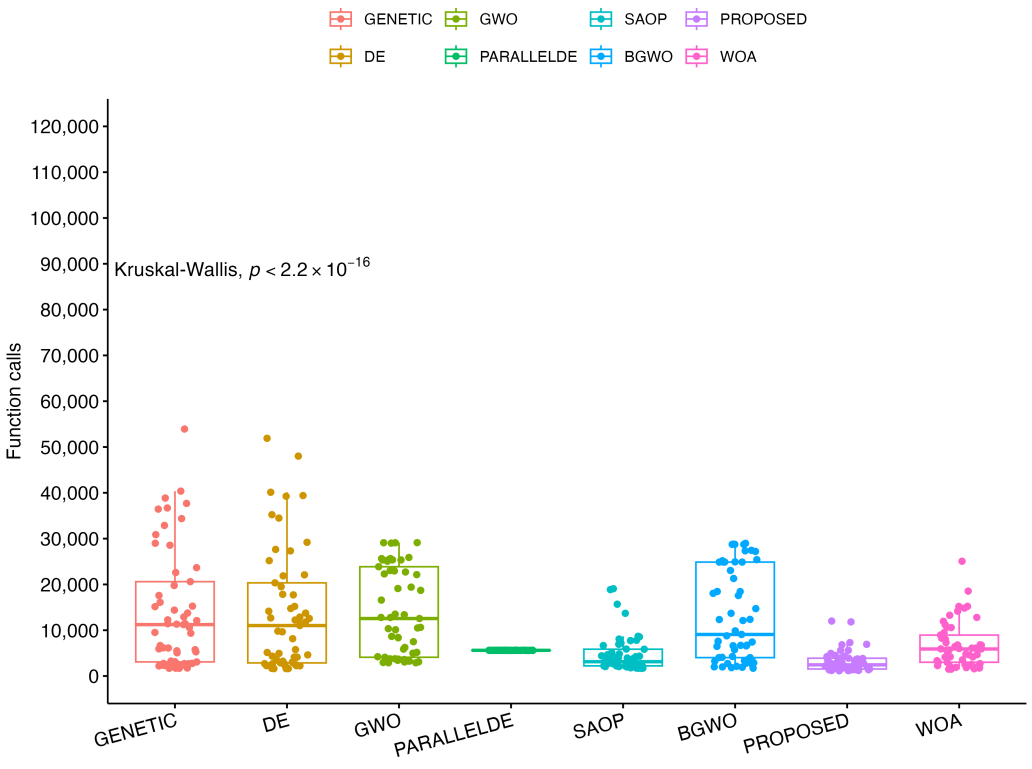

Figure 7. The proposed method in comparison with others.

4.8. Robust Statistical Evaluation of Algorithm Performance

To provide a more rigorous and statistically sound evaluation of the proposed method, the final optimization performance is assessed using multiple statistical indicators. Specifically, the results are reported in terms of the mean and standard deviation (mean ± std) of the final objective function values over 30 independent runs, as summarized in Table 9. The proposed method consistently achieves near-zero objective values with negligible variance, demonstrating both high accuracy and strong robustness compared to the competing algorithms. In contrast, the majority of the compared methods exhibit significantly higher objective values and larger variability, indicating less stable performance. Overall, these results highlight the superiority of the proposed method in terms of both solution quality and robustness across the examined benchmark functions.

Table 9. Statistical comparison of optimization algorithms on benchmark functions in terms of mean and standard deviation (mean ± std) of the final objective values over 30 independent runs.

Function	GA	DE	GWO	PARALLELDE	SAOP	BGWO	WOA	PROPOSED
RASTRIGIN_25	4.84 (18.12)	5.07 (19.08)	2.35 (8.81)	0 (0)	5.37 (20.37)	1.69 (6.87)	3.34 (18.03)	0 (0)
RASTRIGIN_50	42.61 (49.41)	69.74 (80.28)	2.58 (7.66)	0 (0)	47.32 (55.60)	25.76 (35.22)	37.57 (53.86)	1.49 (5.65)
RASTRIGIN_100	90.40 (60.40)	184.99 (123.56)	20.33 (48.36)	0 (0)	106.29 (73.03)	51.67 (35.30)	22.65 (57.87)	1.55 (8.39)
RASTRIGIN_150	111.56 (93.76)	216.70 (178.74)	34.29 (71.21)	0 (0)	126.39 (105.46)	72.69 (66.19)	6.79 (36.61)	0.63 (3.39)
STEP ELLIPSOIDAL_25	209.96 (801.86)	169.30 (634.53)	0 (0)	0 (0)	50.80 (206.66)	0 (0)	0 (0)	0 (0)
STEP ELLIPSOIDAL_50	2578.93 (3001.44)	2798.96 (3259.50)	0 (0)	0 (0)	1413.39 (1665.82)	0.29 (0.97)	0.50 (2.20)	0 (0)
STEP ELLIPSOIDAL_100	7265.83 (4901.31)	7331.79 (4950.73)	0.03 (0.19)	0 (0)	5816.37 (3915.09)	1.74 (2.79)	0.58 (3.13)	0 (0)
STEP ELLIPSOIDAL_150	7842.70 (6466.45)	7842.70 (6466.45)	0.48 (2.61)	0 (0)	7344.09 (6055.11)	2.58 (3.29)	0.22 (1.20)	0 (0)

4.9. Performance Comparison on Low-Dimensional Benchmark Functions

To further strengthen the experimental evaluation, a subset of standard benchmark functions commonly used in deterministic global optimization is considered, including both GKLS instances and classical low-dimensional test functions such as BRANIN, GOLDSTEIN, and HARTMAN [50,51]. Specifically these problems correspond to low-dimensional settings, providing a suitable testbed for assessing performance in scenarios where deterministic methods are particularly effective, as summarized in Table 10. The proposed method consistently requires significantly fewer function evaluations to reach the target accuracy across all tested instances. In particular, it achieves substantial reductions in computational effort compared to the other considered methods, often exceeding 50%. These results demonstrate the efficiency and robustness of the proposed approach in low-dimensional deterministic optimization problems.

Table 10. Performance comparison of optimization algorithms on low-dimensional benchmark functions reported in terms of the number of function evaluations required to reach the target accuracy.

Function	Dimension	GA	DE	GWO	PARALLELDE	SAOP	BGWO	WOA	PROPOSED
BRANIN	2	2017	1940	3325	6547	2123	3118	2297	570
GKLS250	2	1806	1754	3668	6478	1468	3009	2016	547
GKLS350	3	1761	1711	3691	10,454	1352 (0.97)	3178	2366 (0.87)	507
GOLDSTEIN	2	2792	2676	5376	6888	3149	3794	2470	750
HARTMAN	3	2109	2062	3222	7828	1646	2790	2640	575

4.10. Performance Comparison Between DIRECT and the Proposed Method

Based on the results presented in Table 11, where the values correspond to the number of objective function evaluations, the performance of the proposed method relative to DIRECT varies [52] across benchmark functions and problem dimensions. For the Zakharov function, the proposed method consistently requires significantly fewer evaluations across all tested dimensions (100, 150, and 200), indicating a clear efficiency advantage. In the AttractiveSector function, the performance is mixed: the proposed method is less efficient at 100 dimensions, comparable at 150 dimensions, and more efficient at 200 dimensions. Similarly, for the Step Ellipsoidal function, the proposed method requires more evaluations at lower dimensions (100 and 150) but outperforms DIRECT at 200 dimensions. Overall, while DIRECT exhibits stable performance due to its deterministic nature, the proposed method demonstrates improved efficiency in higher-dimensional cases, suggesting a more effective handling of increased problem complexity.

Table 11. Number of objective function evaluations for DIRECT and the proposed method.

Function	Direct	Proposed
ATTRACTIVE SECTOR_100	1198	1818
ATTRACTIVE SECTOR_150	1798	1754
ATTRACTIVE SECTOR_200	2398	1562
STEP_ELLIPSOIDAL_100	1000	1737
STEP ELLIPSOIDAL_150	1500	1698
STEP ELLIPSOIDAL_200	2000	1497
ZAKHAROV_100	10,182	1344
ZAKHAROV_150	12,257	1539
ZAKHAROV_200	12,278	1276

4.11. A Sensitivity Analysis

A sensitivity analysis is conducted to assess the impact of the propagation parameters N_R and N_P on the performance of the proposed algorithm using the Rosenbrock function ($D = 25$). The results expressed in terms of the number of function evaluations are presented in Figures 8 and 9. As illustrated in Figure 8, the parameter N_R has a significant influence on the computational cost. In particular, increasing N_R generally results in a higher number of function evaluations, although the relationship is not strictly monotonic. This behavior suggests that larger propagation ranges enhance exploration capabilities, at the expense of increased computational overhead due to redundant evaluations. In contrast, Figure 9 shows that the parameter N_P has a relatively limited effect on the number of function evaluations, as only minor variations are observed across different parameter settings. Overall, moderate values of both parameters appear to provide an effective trade-off between convergence efficiency and computational cost. Furthermore, the communication cost increases with higher values of N_R and N_P, since broader propagation mechanisms lead to more frequent interactions among agents. Notably, the solution quality remains largely stable across different parameter configurations, indicating that the proposed algorithm is robust with respect to N_R and N_P. This observation suggests that these parameters primarily influence computational effort rather than the final optimization accuracy.

To further investigate the effect of information exchange mechanisms, a sensitivity analysis of the four propagation strategies (1to1, 1toN, Nto1, and NtoN) is conducted under varying values of the propagation parameters N_R and N_P. The corresponding results are presented in Figures 10 and 11. As illustrated in Figure 10, the choice of propaga-

tion strategy has a pronounced impact on convergence behavior. In particular, the NtoN strategy achieves the lowest number of function evaluations for smaller values of N_R, indicating accelerated convergence due to intensive information exchange. However, as N_R increases, the performance of NtoN deteriorates, suggesting that excessive propagation may introduce redundant communication and reduce overall efficiency. In contrast, more localized strategies, such as 1to1 exhibit more stable behavior across different values of N_R, albeit with generally slower convergence. The 1toN and Nto1 strategies demonstrate intermediate performance, effectively balancing exploration and exploitation. Figure 11 further highlights the influence of the propagation strategies with respect to N_P. Specifically, the NtoN strategy consistently attains the lowest number of function evaluations demonstrating significantly faster convergence. This observation indicates that frequent and widespread information exchange can substantially accelerate the optimization process. Nevertheless, such aggressive propagation schemes may lead to premature homogenization of the population, whereby diversity is rapidly reduced and the algorithm becomes prone to convergence to suboptimal solutions. Conversely, more restrictive strategies preserve population diversity for longer periods, resulting in slower yet potentially more robust convergence. Overall, the propagation strategy introduces a trade-off between convergence speed, communication cost, and population diversity. Moderate interaction schemes (such as 1toN and Nto1) provide a balanced compromise, achieving efficient convergence while mitigating the risk of premature homogenization.

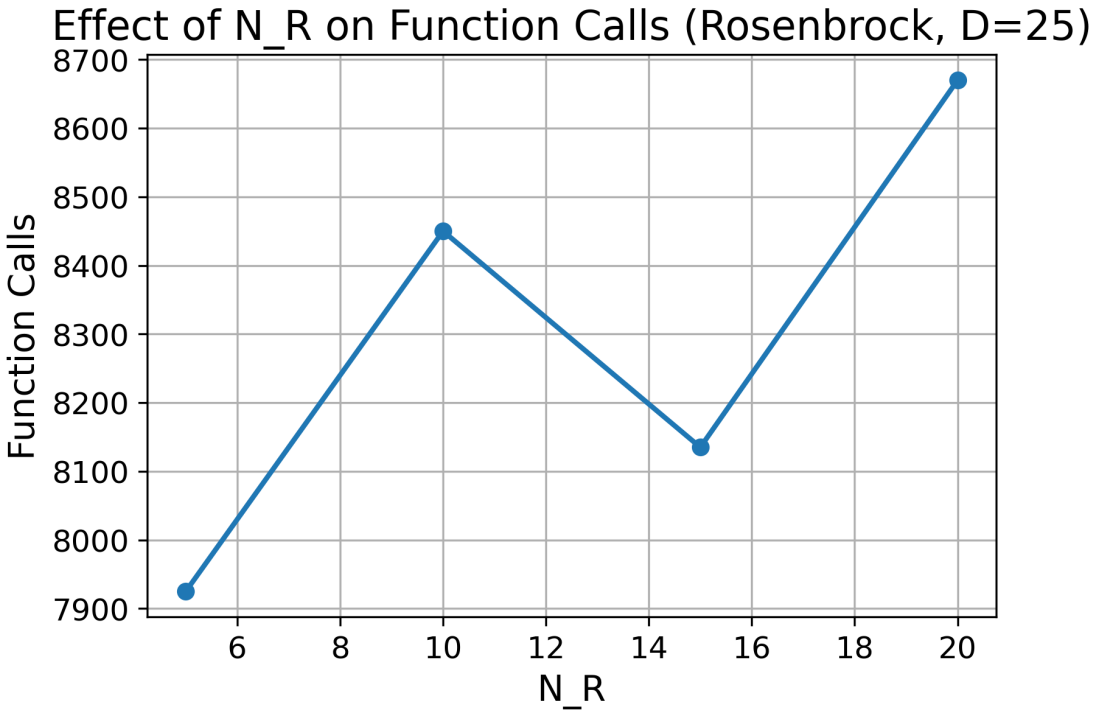

Figure 8. Sensitivity analysis of the propagation parameter N_R in terms of the number of function evaluations on the Rosenbrock function (D = 25).

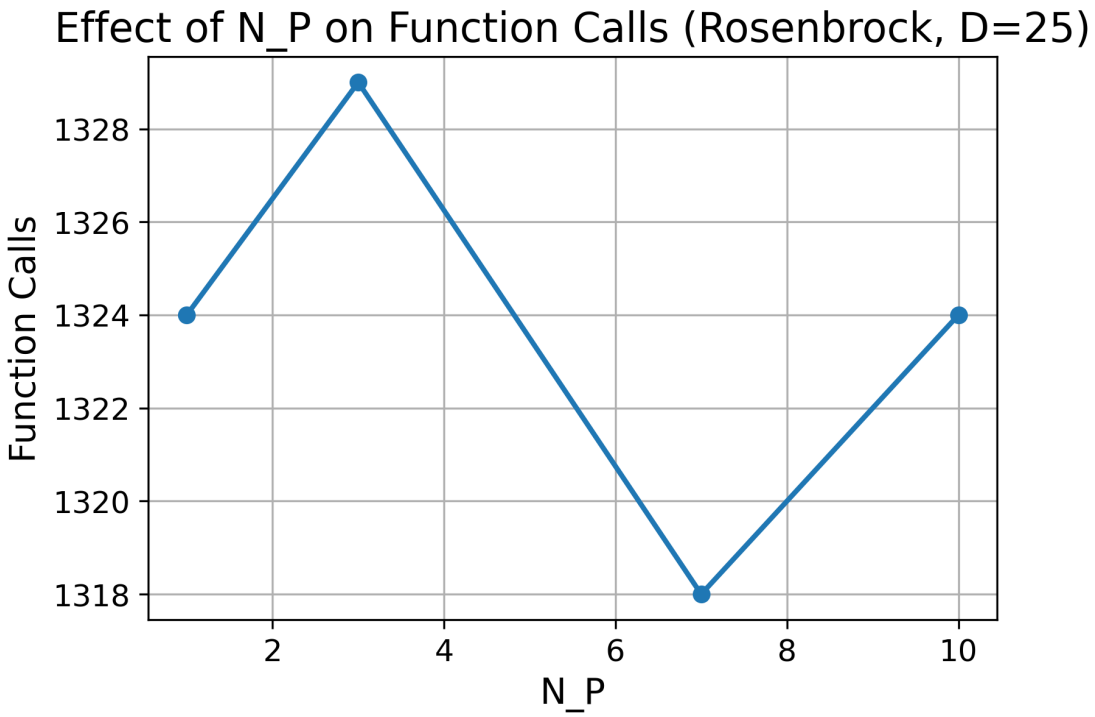

Figure 9. Sensitivity analysis of the propagation parameter N_P in terms of the number of function evaluations on the Rosenbrock function (D = 25).

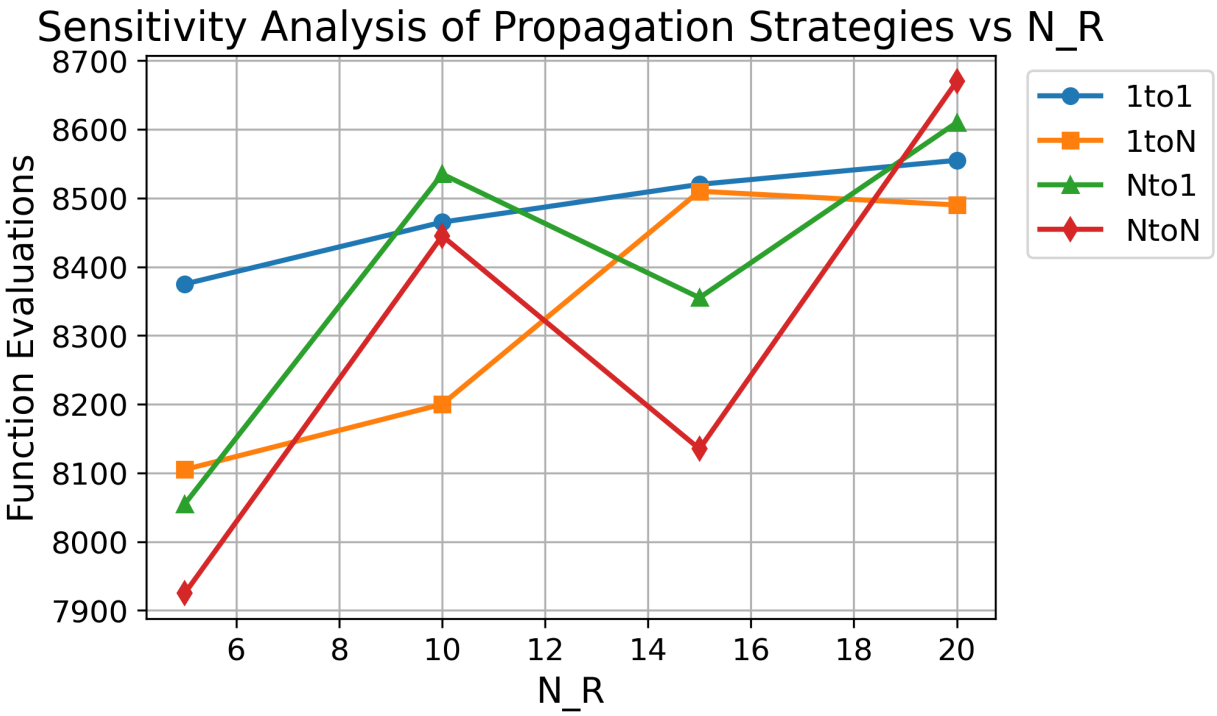

Figure 10. Effect of different propagation strategies (1to1, 1toN, Nto1, NtoN) on convergence performance under varying values of N_R.

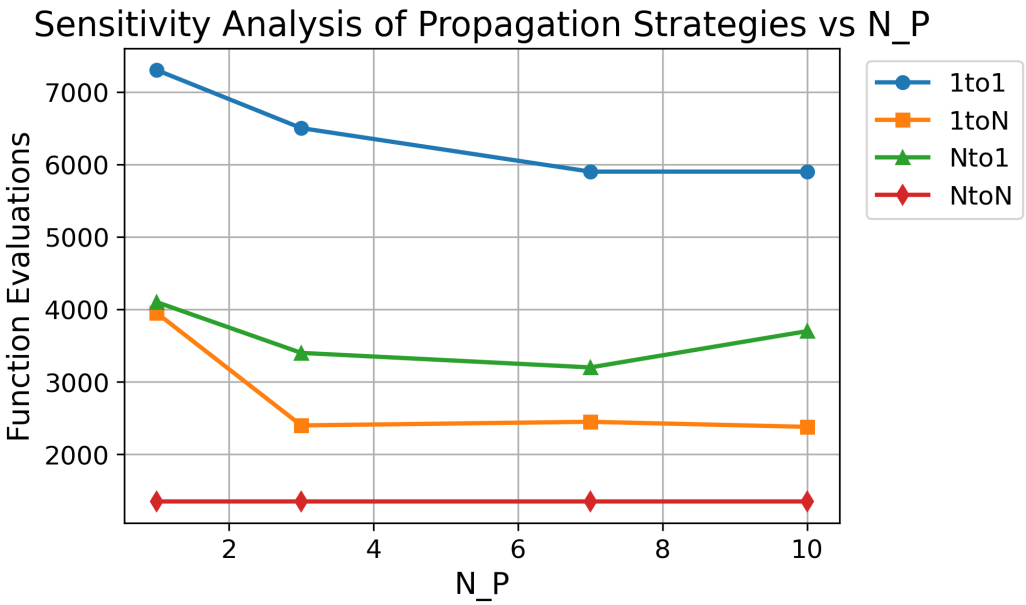

Figure 11. Effect of different propagation strategies (1to1, 1toN, Nto1, NtoN) on convergence performance under varying values of N_P.

4.12. 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 [53] and the Tandem Queueing System [54]. 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 25 to 500 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: $\boldsymbol{x} = [T_1, T_3, P_1, P_3]^\top$. $r = P_3/P_1$, $\gamma = 1.4$.

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

  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 scenario presents a more computationally demanding optimization problem, allowing a clearer assessment of algorithmic scalability under increased complexity.

Figure 12 outlines the comparison of the number of calls of the objective function with respect to the dimension of the problem for the GasCycle problem. It is observed that most methods generally show a constant or slightly decreasing trend in the number of function calls as the dimension increases. The PARALLELDE algorithm maintains a significantly higher number of calls in all dimensions. GENETIC, GWO, BGWO, and DE move in medium-value levels, presenting mild fluctuations. The proposed method (ParallelBGWO) displays one of the lowest values in the number of calls and maintains a relatively constant and controllable behavior throughout the range of dimensions.

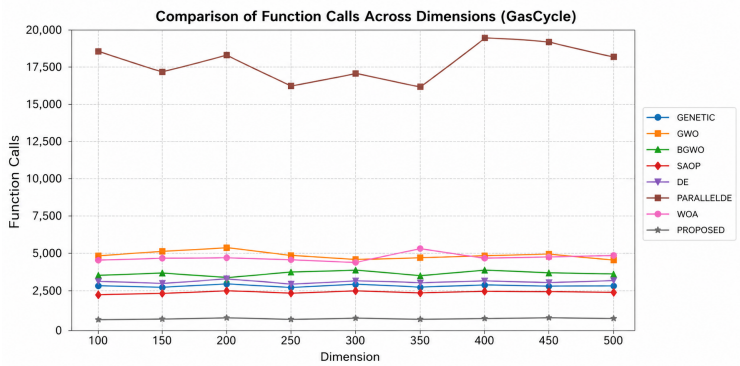

Figure 12. Comparison of function calls across dimensions (GasCycle).

Figure 13 shows the comparison of the execution time of the algorithms with respect to the dimension of the problem for the GasCycle problem. It is observed that the execution time increases progressively with increasing dimensions for all methods, which is expected due to the increased computational complexity. PARALLELDE shows significantly longer execution times compared to the other algorithms. GENETIC, GWO, BGWO, SAOP, DE, and WOA show a similar scaling with a relatively smooth and linear increase in time. The proposed method (ParallelBGWO) follows a corresponding incremental path, maintaining execution times comparable to most algorithms of the same category and clearly lower than PARALLELDE.

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

$$
\begin{aligned}
&7000 \le t_0 \le 10,000,\\
&30 \le T_1 \le 500,\ 30 \le T_2 \le 600,\ 30 \le T_3 \le 1200,\\
&30 \le T_4 \le 1600,\ 30 \le T_{5A}, T_{5B} \le 2000,\\
&0 \le s_{1..4}, s_{5A}, s_{5B}, r_p, k_{A1}, k_{A2}, k_{B1}, k_{B2} \le 1.
\end{aligned}
$$

Objective:

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

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

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

Figure 14 shows the comparison of the number of calls of the objective function with respect to the dimension of the problem for the algorithms under consideration. It is observed that the number of function calls remains relatively constant for most methods as the dimension increases, with small fluctuations per case. The BGWO algorithm displays the highest number of calls in almost all dimensions, maintaining high values with small changes. GENETIC and PARALLELDE exhibit greater fluctuations. In contrast, GWO, SAOP, DE, and WOA move at lower call levels with relatively smooth behavior. The proposed method (ParallelBGWO) exhibits among the lowest values in the number of calls and maintains stable behavior as the dimension increases.

Figure 15 shows the comparison of the execution time of various algorithms with respect to the dimension of the problem. We observe that as the dimension increases, the execution time for all algorithms also increases. The proposed method (ParallelBGWO) clearly exhibits better scaling behavior compared to the classic BGWO and other algorithms, maintaining lower execution times in all dimensions, especially in large dimensions

(400–500). In contrast, algorithms such as Genetic and ParallelDE show a sharp increase in execution time in high dimensions, which indicates lower computational efficiency on large problems. Overall, the results demonstrate that the proposed ParallelBGWO achieves a better balance between performance and computational cost, making it a suitable choice for large-scale problems.

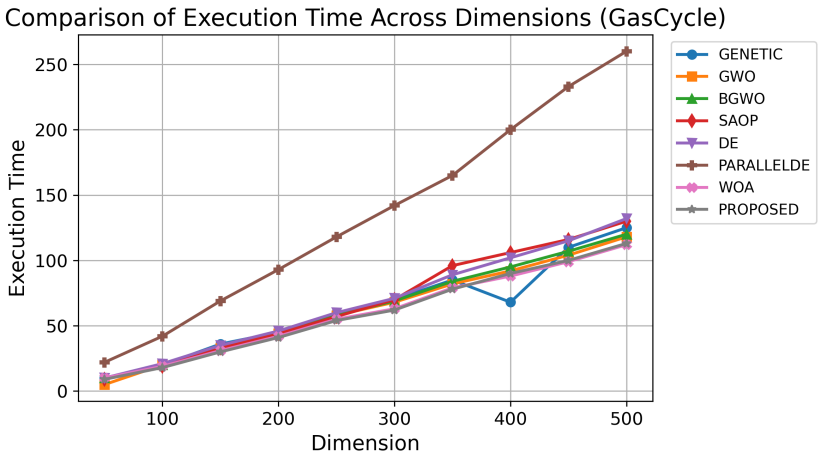

Figure 13. Comparison of execution time across dimensions (GasCycle).

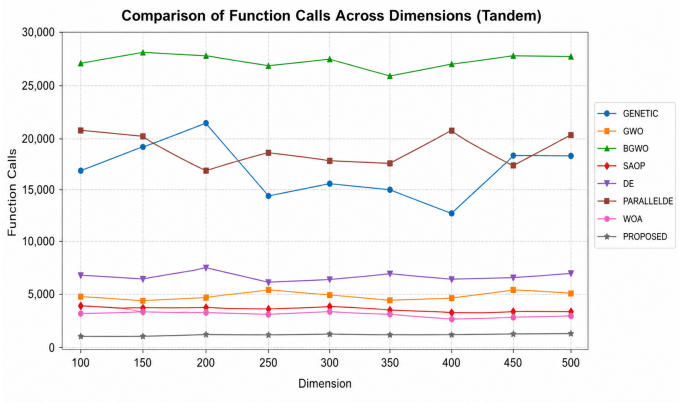

Figure 14. Comparison of function calls across dimensions (Tandem).

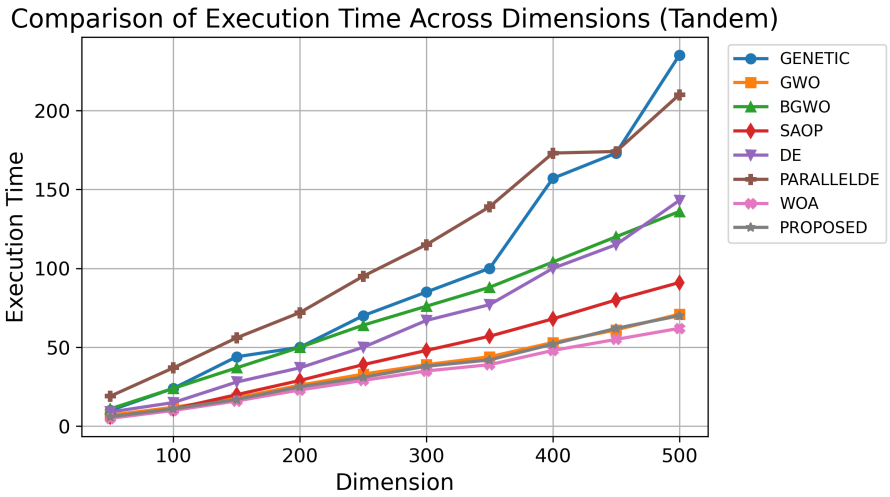

Figure 15. Comparison of execution time across dimensions (Tandem).

4.13. Comprehensive Evaluation of Solution Quality and Efficiency on Real-World Problems

Table 12 presents the final solution quality obtained by all compared algorithms on the GasCycle and Tandem case studies. The results are reported in terms of the mean and standard deviation over 30 independent runs. As observed, all algorithms achieve comparable objective values, particularly for the GasCycle problems, where the solutions converge to nearly identical values with negligible variance. This indicates that the considered problems can be reliably solved by multiple methods. Notably, the proposed method attains solution quality that is on par with the best-performing algorithms without any observable degradation in accuracy. These findings confirm that the previously demonstrated reductions in function evaluations and execution time are not achieved at the expense of solution quality. Moreover, the low standard deviation values across most cases indicate stable and consistent performance. Overall, the results demonstrate that the proposed approach effectively balances computational efficiency and solution accuracy in real-world optimization scenarios. Furthermore, it is important to note that both the GasCycle and Tandem case studies correspond to well-defined optimization problems in which all candidate solutions remain feasible by construction. Consequently, constraint violations are not applicable, and all methods consistently produce feasible solutions.

Table 12. Final solution quality obtained by all compared algorithms on the GasCycle and Tandem case studies, reported as mean and standard deviation (mean ± std) over 30 independent runs.

Function	GA	DE	GWO	PARALLELDE	SAOP	BGWO	WOA	PROPOSED
Tandem_50	27.72 (0.16)	27.83 (0.19)	28 (2.42)	28.24 (0.33)	28.69 (0.67)	27.84 (0.18)	28.96 (0.68)	28.16 (0.34)
Tandem_100	27.67 (0.10)	27.80 (0.14)	28.64 (2.00)	28.19 (0.29)	28.80 (0.76)	27.85 (0.25)	29.04 (0.66)	28.20 (0.39)
Tandem_200	27.66 (0.11)	27.87 (0.21)	28.11 (2.23)	28.25 (0.30)	28.72 (0.56)	27.81 (0.16)	28.99 (0.68)	28.26 (0.37)
GasCycle_50	-0.93 (9.75×10^{-10})	-0.93 (8.67×10^{-10})	-0.93 (0)	-0.93 (0)	-0.93 (0.17)	-0.93 (0)	-0.92 (0.01)	-0.9 (0.01)
GasCycle_100	-0.93 (8.01×10^{-10})	-0.93 (1.16×10^{-9})	-0.93 (0)	-0.93 (0)	-0.91 (0.16)	-0.93 (0)	-0.92 (0.01)	-0.9 (0.01)
GasCycle_200	-0.93 (1.30×10^{-9})	-0.93 (3.43×10^{-9})	-0.93 (0)	-0.93 (0)	-0.92 (0.012)	-0.93 (0)	-0.92 (0.01)	-0.9 (0.013)

4.14. Impact of the Subpopulation Termination Criterion on Practical Problems

Figure 16 shows the effect of different subpopulation termination criteria (ANY, MAJORITY, ALL) on the number of objective function evaluations for the GasCycle problem in different dimensions. From the results, it is observed that the ANY strategy presents the smallest number of function calls in all dimensions. This shows that a more flexible termination criterion can significantly reduce the computational cost. In contrast, the MAJORITY strategy requires a larger number of evaluations as termination is triggered only when the majority of subpopulations are completed. The ALL strategy displays the highest values since it requires the completion of all subpopulations before termination. Therefore, the results show that the more stringent the termination criterion, the higher the overall computational cost.

In the Tandem problem, a similar behavior is observed regarding the effect of the termination criteria on the number of evaluations of the objective function. The ANY strategy also presents the lowest function call values in all dimensions; thus, it is a more flexible termination criterion that can significantly reduce the computational cost. The MAJORITY strategy presents greater variability and generally requires more evaluations, while the ALL strategy leads to the largest function call values, as the search process continues until all subpopulations are completed. Therefore, the stringency of the termination criterion significantly affects the efficiency of the algorithm, with the ANY strategy offering the most computationally efficient approach. This improvement in computational efficiency is achieved without any noticeable degradation in solution quality, as confirmed by the results

reported in Table 13. The final objective values obtained using the ANY strategy remain comparable to those achieved with the MAJORITY and ALL strategies. The experimental results for this case are shown graphically in Figure 17.

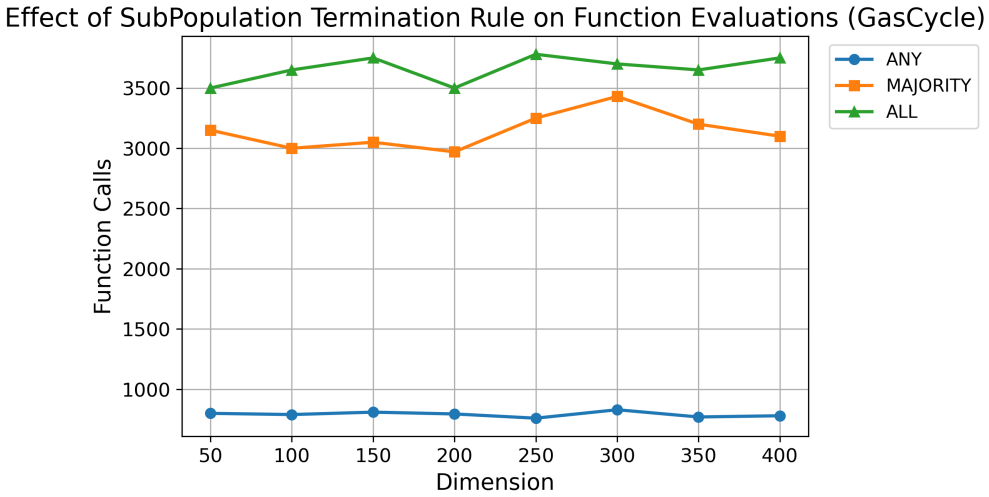

Figure 16. Effect of subpopulation termination rule on function evaluations (GasCycle).

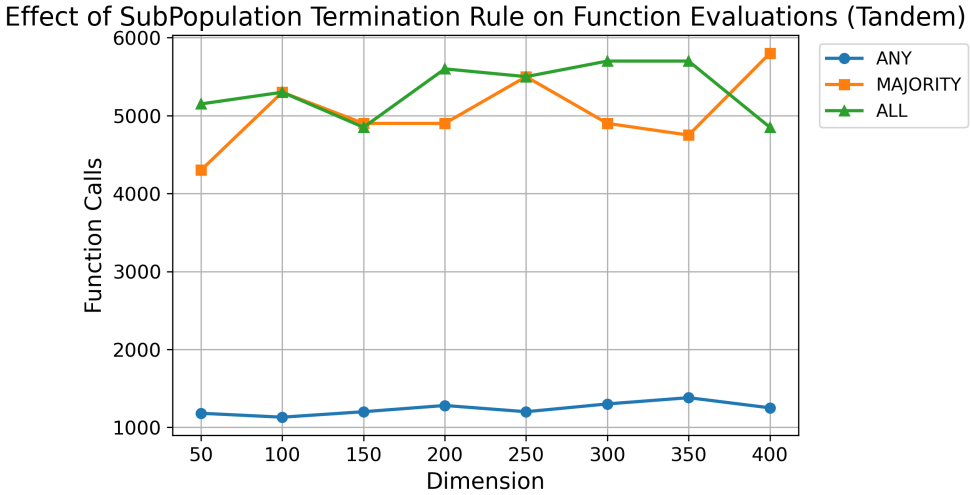

Figure 17. Effect of subpopulation termination rule on function evaluations (Tandem).

Table 13 reports the mean and standard deviation of the final objective values obtained by the ANY, MAJORITY, and ALL strategies for the Tandem and GasCycle case studies across different problem sizes. The mean values correspond to the average performance over multiple independent runs, while the standard deviation reflects the consistency and robustness of each strategy. The results indicate that all three strategies achieve comparable objective values across all cases, with only minor differences observed. In addition, the relatively low standard deviation values suggest stable performance across runs for all strategies. In conjunction with the results presented in Figures 16 and 17, which show a reduction in the number of function evaluations for the ANY strategy, these findings suggest that the improvement in computational efficiency is not accompanied by a significant change in final solution quality. Overall, the ANY, MAJORITY, and ALL strategies demonstrate similar levels of solution quality, with ANY offering improved efficiency while maintaining comparable performance.

Table 13. Comparison of final objective values (mean ± standard deviation) for the ANY, MAJORITY, and ALL strategies on Tandem and GasCycle practical problems.

Function	Any	Majority	All
Tandem_25	28.22 (0.35)	28.12 (0.45)	27.99 (0.35)
Tandem_50	28.20 (0.39)	28.05 (0.37)	28.04 (0.44)
Tandem_100	28.26 (0.37)	27.97 (0.44)	27.99 (0.32)
Tandem_150	28.21 (0.36)	28.09 (0.43)	28.10 (0.36)
GasCycle_25	−0.91 (0.011)	−0.92 (0.008)	−0.92 (0.010)
GasCycle_50	−0.91 (0.018)	−0.92 (0.013)	−0.92 (0.016)
GasCycle_100	−0.90 (0.013)	−0.92 (0.011)	−0.92 (0.008)
GasCycle_150	−0.91 (0.013)	−0.91 (0.02)	−0.92 (0.008)

4.15. Neural Network Training Experiments

To validate the effectiveness of the proposed method, it was applied to the training of artificial neural networks [55,56] for classification problems using benchmark datasets from publicly available repositories.

The neural network training experiments were conducted using a Multilayer Perceptron (MLP) architecture with a single hidden layer consisting of 10 neurons. The network weights were initialized within the range $[-10, 10]$. The evaluation was performed on classification datasets obtained from the UCI Machine Learning Repository using a ten-fold cross-validation protocol. All input features were normalized prior to training. The training objective was defined as the minimization of the Mean Squared Error (MSE). The optimization process follows a hybrid global–local strategy. In the global phase, the ParallelBGWO is employed with a population of 50 agents, distributed across subpopulations during parallel execution (1, 5, and 10 threads). The propagation mechanism N_M is set to the Nto1 communication scheme, enabling controlled information exchange among subpopulations. In the local phase, the BFGS algorithm is applied for 200 iterations to refine the best solution obtained from the global search. The optimization process is terminated using a similarity-based stopping criterion.

The classification datasets were selected from widely used benchmark repositories, including the UCI Machine Learning Repository, the KEEL dataset repository, and the StatLib database. These datasets were chosen due to their diversity in dimensionality, class distribution, and classification complexity, allowing for a comprehensive evaluation of the performance and robustness of the proposed optimization method. Results are reported as the mean classification error over multiple independent runs.

1. The UCI database, https://archive.ics.uci.edu/ (accessed on 28 March 2026) [57].
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php (accessed on 28 March 2026) [58].
3. The Statlib URL, https://lib.stat.cmu.edu/datasets/index (accessed on 28 March 2026).

From these databases, the following datasets were obtained:

1. **Circular** dataset, which contains artificially generated data.
2. **Heart** dataset [59], a medical dataset used for the prediction of heart diseases.
3. **Ionosphere** dataset, a climate dataset [60].
4. **Liverdisorder** dataset [61], a medical dataset.
5. **Lymography** dataset [62].
6. **Pima** dataset [63].
7. **Spiral** dataset, which is an artificial dataset.
8. **Statheart**, a dataset related to the detection of heart diseases.

9. **Wdbc** dataset [64].
10. **Wine** dataset, used for the detection of quality of wines [65,66].

All experiments were conducted 10 times, each with a different random seed. To validate the results, the widely used ten-fold cross-validation method was applied. For the classification datasets, the mean classification error is presented, computed according to the following formula:

$$E_C\left(N\left(\vec{x}, \vec{w}\right)\right) = 100 \times \frac{\sum_{i=1}^{N}\left(\text{class}\left(N\left(\vec{x_i}, \vec{w}\right)\right) - y_i\right)}{N} \quad (2)$$

In this equation, the set T denotes the used test set. In the experiments, the following methods were used:

1. ADAM optimizer [67] was used to train a neural network with 10 processing nodes.
2. BFGS optimizer [68] incorporated to train a neural network with 10 processing nodes.
3. GENETIC, where a neural network [14,15] was used to train a neural network with 10 processing nodes.
4. PROPOSED, where the described method is used to train a neural network with 10 processing nodes.

The selected classification datasets are widely used benchmark problems in the machine learning and neural network literature, enabling reliable evaluation of classification performance. Furthermore they provide diversity in terms of dimensionality class distribution and classification complexity, allowing for a comprehensive assessment of the effectiveness and robustness of the proposed optimization method across different types of learning tasks. In addition, these datasets correspond to optimization problems of varying difficulty in neural network training, including high-dimensional and non-convex search spaces. This makes them suitable for evaluating the performance of the proposed method in large-scale black-box optimization settings.

The results obtained by the previously mentioned methods are presented in Table 14.

Table 14 indicates that PROPOSED is the most effective method overall in terms of classification error rate, since it achieves the lowest average error, 19.84% compared with 26.26% for GENETIC, 34.78% for ADAM, and 35.17% for BFGS. Because lower values correspond to better performance, this result demonstrates a clear overall superiority of PROPOSED over both the two gradient-based baselines and the evolutionary GENETIC approach. This advantage is not marginal but substantial, suggesting that the proposed scheme attains a more favorable balance between adaptation and generalization across the examined classification datasets.

At the individual dataset level, PROPOSED ranks first in 9 out of the 10 datasets, yielding the lowest error on Circular, Heart, Ionosphere, Lymography, Pima, Spiral, Statheart, Wine, and Wdbc. The only exception is Liverdisorder, where GENETIC achieves a slightly lower error rate than PROPOSED. Nevertheless, this isolated case does not alter the overall pattern, which is characterized by consistent and widespread superiority of the proposed method across heterogeneous classification problems. It is particularly noteworthy that the advantage of PROPOSED is preserved even on demanding datasets such as Spiral, while especially pronounced improvements are observed on Heart, Statheart, Wine, and Wdbc, relative to the competing methods.

Overall, the results support the conclusion that PROPOSED does not merely provide occasional strong outcomes, but rather exhibits greater robustness and stronger generalization ability across nearly the entire experimental setting. Therefore, it can be regarded as the most reliable method among the four compared models under the present evaluation protocol. Results are reported as the mean classification error over multiple independent runs.

Table 14. Average classification error rates on some classification datasets using a series of methods.

Dataset	ADAM	BFGS	GENETIC	PROPOSED
Circular	19.95%	6.08%	5.99%	4.30%
Heart	38.53%	39.44%	28.34%	18.17%
Ionosphere	16.64%	15.29%	15.14%	15.00%
Liverdisorder	41.53%	42.59%	31.11%	32.89%
Lymography	39.79%	35.43%	28.42%	24.60%
Pima	34.85%	35.59%	32.19%	25.25%
Spiral	47.67%	47.99%	48.66%	41.80%
Statheart	44.04%	39.65%	27.25%	20.62%
Wine	29.40%	59.71%	19.20%	11.24%
Wdbc	35.35%	29.91%	8.56%	4.50%
AVERAGE	**34.78%**	**35.17%**	**26.26%**	**19.84%**

The statistical analysis based on the Friedman test with the Nemenyi post hoc procedure showed that there are overall statistically significant differences among the four models on the classification datasets of Table 14. Specifically, the overall Friedman p-value is 5.28×10^{-5}, which is well below 0.05, thus rejecting the null hypothesis of equivalent model performance in terms of classification error.

The pairwise comparisons reported in Figure 18 indicate that PROPOSED significantly outperforms both ADAM and BFGS, since in both cases, the obtained p-value is 3.95×10^{-4}, corresponding to an extremely significant difference. In contrast, the comparisons ADAM-BFGS ($p = 1.0000$), ADAM-GENETIC ($p = 0.1096$), BFGS-GENETIC ($p = 0.1096$), and GENETIC-PROPOSED ($p = 0.3069$) are not statistically significant at the $\alpha = 0.05$ level. Therefore, the results establish PROPOSED as the statistically strongest approach relative to the classical ADAM and BFGS schemes, whereas its superiority over GENETIC, although visible at the level of average error, is not statistically confirmed by the Nemenyi post hoc test.

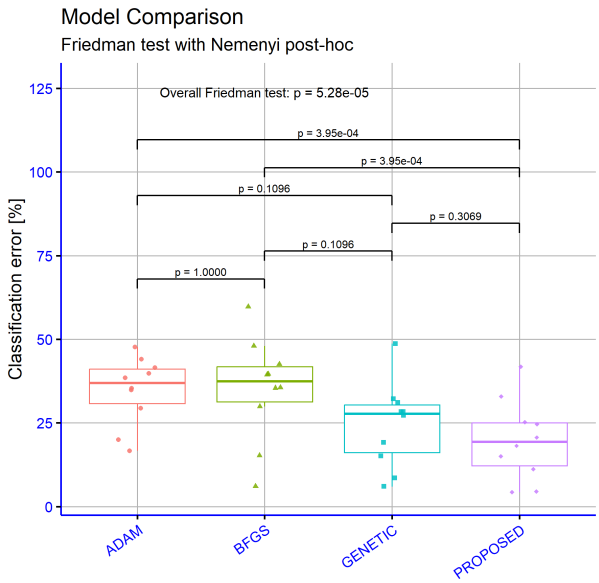

Figure 18. Statistical comparison of ANN training methods based on Friedman/Nemenyi post hoc analysis.

5. Discussion

5.1. Effect of the Parallel Architecture

The experimental results demonstrate that the effectiveness of ParallelBGWO derives from the coordinated integration of four principal structural components: population partitioning into cooperating subpopulations, inter-cluster information propagation, communication strategy selection, and flexible termination criteria. Collectively, these mechanisms improve the balance between exploration and exploitation, which is fundamental in large-scale optimization. Furthermore, the findings indicate that parallelization contributes not only to reduced execution time but also to enhanced search dynamics, since the concurrent exploration of multiple search regions mitigates premature convergence and facilitates the dissemination of beneficial search information. The analysis of the number of subpopulations further indicates that algorithmic performance is highly dependent on the degree of population partitioning. While the transition from a single-population model to multiple cooperating clusters significantly reduces computational cost, the improvement is not monotonic. Specifically, configurations employing 10 and 20 subpopulations provide the most favorable trade-off, whereas increasing the number to 30 does not yield proportional benefits. This suggests that excessive fragmentation may weaken the internal evolutionary pressure of each subpopulation and reduce exploitation efficiency. Consequently, an appropriate compromise between diversity preservation and local search pressure is necessary. Overall, these findings suggest that the effectiveness of the parallel architecture is strongly dependent on maintaining a balance between subpopulation diversity and intra-cluster selection pressure. While increasing the number of subpopulations enhances exploration through distributed search, excessive fragmentation weakens convergence pressure, leading to diminishing returns. This highlights the importance of selecting an appropriate level of parallelism depending on the problem complexity.

5.2. Impact of Communication and Propagation Mechanisms

The propagation mechanism is shown to constitute a critical component of the proposed method. The comparison between PROP = 0 and PROP = 1 demonstrates that enabling information propagation significantly decreases the number of objective function evaluations, indicating that subpopulation cooperation is essential to the efficiency of the optimization process. This finding suggests that the rapid dissemination of locally generated high-quality solutions enhances the global guidance of the search process. Similarly, the evaluation of alternative communication strategies indicates that the fully bidirectional NtoN scheme consistently achieves the lowest computational cost relative to the 1to1, 1toN, and Nto1 approaches. This superiority is further validated through Friedman and Wilcoxon statistical tests, confirming that full information exchange among subpopulations represents the most effective strategy for exploiting cooperative parallelism. These observations reveal a fundamental trade-off between communication intensity and population diversity. While frequent and global information exchange (NtoN) accelerates convergence by rapidly disseminating high-quality solutions, it also increases the risk of premature convergence due to loss of diversity. Conversely, more restrictive communication schemes preserve diversity but slow down convergence. Therefore, selecting an appropriate propagation strategy is crucial and should be adapted to the problem landscape.

5.3. Analysis of Termination Strategies

With respect to subpopulation termination criteria, the ANY strategy consistently yields the lowest number of function evaluations on the GasCycle and Tandem problems, whereas the MAJORITY and ALL strategies result in higher computational cost. This outcome is expected, as stricter termination rules prolong execution even when a

subset of subpopulations has already converged. These observations indicate that termination flexibility constitutes a significant determinant of algorithmic efficiency rather than a minor implementation detail. The proposed framework benefits when unnecessary waiting among subpopulations is minimized, thereby allowing more efficient use of computational resources.

5.4. Comparative Performance Evaluation

The comparison with established metaheuristic methods further reinforces the effectiveness of the proposed approach. Across benchmark problems with dimensionality ranging from 25 to 150, ParallelBGWO achieves the lowest number of function evaluations compared with GA, DE, GWO, ParallelDE, SAOP, BGWO, and WOA. Its superiority is particularly evident on demanding functions such as Zakharov and Sharp Ridge. Moreover, in the practical GasCycle and Tandem optimization problems, ParallelBGWO maintains low computational cost and execution time as dimensionality increases, thereby demonstrating improved scalability. This property is particularly desirable in large-scale optimization contexts, where the growth of computational burden must remain controlled as problem complexity increases. While the proposed method demonstrates clear advantages in terms of computational efficiency, a comprehensive evaluation must also consider solution quality. The experimental results show that the proposed approach consistently achieves competitive or superior final objective values, accompanied by low standard deviation across independent runs. This indicates not only high accuracy but also stable and reliable convergence behavior. Therefore, the method improves computational efficiency without compromising solution quality, achieving a balanced overall performance. This behavior can be further interpreted through the fundamental differences between stochastic and deterministic optimization approaches. Deterministic methods, particularly gradient-based techniques, are known to provide strong convergence guarantees and high solution accuracy in smooth and well-structured problems, making them preferable for problems where reliability and precision are critical. However, their performance typically deteriorates in high-dimensional and multimodal search spaces, where they are more prone to premature convergence. An important observation is that the proposed method exhibits improved relative performance as the dimensionality increases. This suggests that the cooperative parallel structure becomes increasingly effective in handling the complexity of high-dimensional search spaces, where deterministic methods tend to struggle due to the exponential growth of the search domain. In contrast, stochastic metaheuristic methods exhibit enhanced exploration capabilities and greater robustness, making them more suitable for large-scale and non-convex optimization problems. This distinction further supports the design of the proposed framework, which integrates global stochastic search with local deterministic refinement in order to achieve both exploration efficiency and solution accuracy. From a practical standpoint, these results indicate that the proposed method is particularly suitable for large-scale optimization problems, where balancing exploration and exploitation is critical. In such cases, the integration of parallel exploration with adaptive information exchange provides a robust mechanism for achieving both fast convergence and high solution quality.

5.5. Practical Applicability and Neural Network Validation

The application of ParallelBGWO to neural network training further demonstrates its practical utility. In this setting, the proposed method achieves the lowest average classification error compared with ADAM, BFGS, and GA, while attaining superior performance on the majority of the examined datasets. Statistical analysis using Friedman and Nemenyi post hoc procedures confirms that the superiority of ParallelBGWO over

ADAM and BFGS is statistically significant. Although the method also outperforms the genetic algorithm in average terms, this difference is not statistically significant at the same confidence level. Overall, the experimental analysis consistently indicates that the performance of the proposed framework is governed by the interplay between exploration capability, communication efficiency, and population diversity. The results highlight that no single configuration is universally optimal; instead, the algorithm benefits from a careful balance of its design components depending on the problem characteristics.

5.6. Limitations and Future Research Directions

The obtained results indicate that the effectiveness of ParallelBGWO is attributable to its ability to combine efficient global exploration with rapid exploitation of shared subpopulation knowledge. The proposed framework successfully mitigates both premature convergence and excessive search dispersion, suggesting that its architecture functions as a coherent optimization strategy. Nevertheless, several limitations warrant further investigation. The number of subpopulations and the communication strategy remain fixed design parameters and could be adaptively adjusted in future implementations. Furthermore, additional evaluation metrics, including solution-quality variance and performance on constrained optimization problems, should be considered in future studies. Finally the assessment of ParallelBGWO on deeper neural architectures and in noisier stochastic environments would further strengthen its practical validation. Overall, the findings demonstrate that ParallelBGWO constitutes a substantive extension of BGWO and provides an efficient, robust, and scalable optimization framework for large-scale global optimization problems.

6. Conclusions

The results of the present study demonstrate that the proposed ParallelBGWO constitutes a substantial and effective extension of the classical BGWO framework for solving high-dimensional and computationally demanding optimization problems. The introduced parallel architecture based on the cooperation of multiple subpopulations, coordinated information propagation, and flexible termination strategies significantly enhances the balance between exploration and exploitation. The experimental evaluation across a wide range of benchmark problems confirms that the proposed method consistently reduces computational cost, as reflected by the lower number of objective function evaluations and competitive execution times, while maintaining high solution quality and stability. In particular, configurations with 10 and 20 subpopulations provide the most favorable trade-off between convergence speed and search accuracy, highlighting the importance of appropriate population partitioning.

The analysis of termination strategies further indicates that the ANY criterion achieves the lowest computational cost, as it allows the algorithm to proceed without unnecessary synchronization delays, whereas the MAJORITY and ALL strategies impose stricter conditions that increase execution time without proportional benefits. This finding demonstrates that termination flexibility plays a crucial role in improving the efficiency of cooperative parallel optimization. Furthermore, the propagation mechanism is shown to be a key factor in accelerating convergence by enabling efficient information exchange among subpopulations. Among the examined communication strategies, the NtoN scheme exhibits the most robust overall performance, suggesting that full bidirectional information sharing enhances cooperation and improves global search guidance. The superiority of the proposed method is not limited to synthetic benchmark functions but extends to practical optimization scenarios, including the GasCycle and Tandem problems, as well as neural network training tasks. In these cases, ParallelBGWO achieves competitive or superior

performance compared to established optimization methods, while maintaining robustness and consistency across multiple runs.

Overall, the findings suggest that ParallelBGWO provides an efficient, scalable, and flexible optimization framework, particularly suitable for large-scale and complex search spaces where traditional methods may suffer from premature convergence or high computational cost. Future research directions include the development of adaptive mechanisms for dynamically tuning propagation and communication parameters, the investigation of more advanced information exchange strategies, and the extension of the proposed framework to constrained and noisy optimization environments.

Author Contributions: G.K., K.G.B., 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: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

References

1. Bertsimas, D.; Margaritis, G. Global optimization: A machine learning approach. *J. Glob. Optim.* **2025**, *91*, 1–37. [CrossRef]
2. Gauthaman, S.A.; Muniasamy, A.; Kamaldheen, R.; Kumar, S.S.M.; Murugesan, K.; Viswanathan, M.K.; Devendran, H. Introduction to machine learning and optimization in healthcare. In *Integrative Machine Learning and Optimization Algorithms for Disease Prediction*; IGI Global Scientific Publishing: Hershey, PA, USA, 2026; pp. 1–34.
3. Theodorakopoulos, L.; Karras, A.; Krimpas, G.A. Optimizing apache spark MLlib: Predictive performance of large-scale models for big data analytics. *Algorithms* **2025**, *18*, 74. [CrossRef]
4. Nassef, A.M.; Abdelkareem, M.A.; Maghrabie, H.M.; Baroutaji, A. Review of metaheuristic optimization algorithms for power systems problems. *Sustainability* **2023**, *15*, 9434. [CrossRef]
5. Recalde, A.; Cajo, R.; Velasquez, W.; Alvarez-Alvarado, M.S. Machine learning and optimization in energy management systems for plug-in hybrid electric vehicles: A comprehensive review. *Energies* **2024**, *17*, 3059. [CrossRef]
6. Kyrou, G.; Charilogis, V.; Tsoulos, I.G. A Novel Method That Is Based on Differential Evolution Suitable for Large-Scale Optimization Problems. *Foundations* **2026**, *6*, 2. [CrossRef]
7. Tang, K.; Li, X.; Suganthan, P.N.; Yang, Z.; Weise, T. *Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization*; Nature Inspired Computation and Applications Laboratory, USTC: Hefei, China, 2007; Volume 24, pp. 1–18.
8. Li, X.; Tang, K.; Omidvar, M.N.; Yang, Z.; Qin, K.; China, H. Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization. *Gene* **2013**, *7*, 8.
9. Molina, D.; Herrera, F. Iterative hybridization of DE with local search for the CEC'2015 special session on large scale global optimization. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, 25–28 May 2015; pp. 1974–1978.
10. Long, W.; Jiao, J.; Liang, X.; Tang, M. Inspired grey wolf optimizer for solving large-scale function optimization problems. *Appl. Math. Model.* **2018**, *60*, 112–126. [CrossRef]
11. Long, W.; Cai, S.; Jiao, J.; Tang, M. An efficient and robust grey wolf optimizer algorithm for large-scale numerical optimization: W. Long et al. *Soft Comput.* **2020**, *24*, 997–1026. [CrossRef]

12. Powell, M.J.D. A tolerant algorithm for linearly constrained optimization calculations. *Math. Program.* **1989**, *45*, 547–566. [CrossRef]
13. Charilogis, V.; Tsoulos, I.G. Toward an ideal particle swarm optimizer for multidimensional functions. *Information* **2022**, *13*, 217. [CrossRef]
14. Wan, H. Applying the genetic algorithm to optimization problems. *WIT Trans. Inf. Commun.* **2026**, *2*, 452–462.
15. Pereira, C.M.N.A.; Schirru, R.; Martinez, A.S. Learning an optimized classification system from a data base of time series patterns using genetic algorithms. *WIT Trans. Inf. Commun. Technol.* **2026**, *22*, 22–34.
16. Koyuncuoğlu, M.U. Sustainable configuration of production lines: Multi-objective energy-efficient optimization with adaptive differential evolution and golden ratio algorithms. *Eng. Appl. Artif. Intell.* **2026**, *165*, 113469. [CrossRef]
17. Horrillo-Quintero, P.; Garcia-Trivino, P.; Carrasco-Gonzalez, D.; Fernandez-Ramirez, L.M. Optimal control strategy based on differential evolution algorithm for seamless transition between islanded and grid-connected operation modes in microgrid clusters. *Expert. Appl.* **2026**, *297*, 129461. [CrossRef]
18. Gupta, S.; Gautam, K. Enhanced cooperative learning and local search integrated particle swarm optimization for global optimization and coverage enhancement in wireless sensor network. *Swarm Evol. Comput.* **2026**, *100*, 102262. [CrossRef]
19. Li, Z.; Wang, P.; Feng, Y.; Zhang, G.; Guo, Z.; Zhang, M.; Zhang, Y. Optimization of manipulator inverse kinematics using improved particle swarm algorithm for enhanced manufacturing efficiency. *Proc. Inst. Mech. Eng. B J. Eng. Manuf.* **2026**, *240*, 188–202. [CrossRef]
20. Chen, F.; Wu, X.; Wang, Z.; Qi, W.; Li, P. A New Ant Colony Optimization-Based Dynamic Path Planning and Energy Optimization Model in Wireless Sensor Networks for Mobile Sink by Using Mixed-Integer Linear Programming. *Biomimetics* **2026**, *11*, 44. [CrossRef] [PubMed]
21. Yang, W.; Huang, C.; Ji, P. Ant colony algorithm optimized by local search and adaptive strategy for drilling sequence path planning. *J. Mech. Sci. Technol.* **2026**, *40*, 1291–1301. [CrossRef]
22. Saini, G.; Jadon, S.S.; Chaube, S. Learning-aided Artificial Bee Colony with neural knowledge transfer for global optimization. *Sci. Rep.* **2026**, *16*, 7019. [CrossRef]
23. Saini, G.; Jadon, S.S. A neural knowledge learning-driven artificial bee colony algorithm with reinforcement adaptation for global optimization. *Appl. Soft Comput.* **2026**, *191*, 114662. [CrossRef]
24. Nouri, P.; Kamarposhti, M.A.; Nouri, T.; Radmehr, M. Optimization of fuzzy logic controller in the converter of a standalone solar power system using the firefly algorithm. *Sci. Rep.* **2026**, *16*, 10248. [CrossRef] [PubMed]
25. Sutrisno, D.; Ardhika, D.Y.; Siddiq, S.A.; Prakasa, M.A.; Fauzi, M.A.; Widiyawati, E.; Kurniawan, I.H.; Wibowo, R.S.; Budiharto Putri, V.L.; Robandi, I. Application of Firefly Algorithm for Optimizing the Available Transfer Capability in the Sulbagsel Electricity System of Indonesia. *Int. J. Intell. Eng. Syst.* **2026**, *19*, 731–744. [CrossRef]
26. Senandes, R.S.; Brante, G.; Souza, R.D.; Azarbahram, A.; López, O.L.A. Bat Algorithm-Based Energy Beamforming for Wireless Power Transfer with Dynamic Metasurface Antennas. *IEEE Trans. Commun.* **2026**, *74*, 4078–4089. [CrossRef]
27. Alsumaiei, A.A. Physics-constrained neural network for daily pan evaporation forecasting in hyper-arid climates optimized by Bat Algorithm. *J. Hydrol.* **2026**, 134936. [CrossRef]
28. Almufti, S.M.; Hassan, N.S.; Mercado, M.C.; Felix, J.W.G.; Barbosa, T.S.; Supan, F.E. Survey on whale optimization algorithm: From fundamental principles to modern adaptations and applications. *Int. J. Inf. Technol. Comput. Eng.* **2026**, *6*, 1–19. [CrossRef]
29. Su, Y.; Liu, Y. A novel marine predator whale optimization algorithm for global numerical optimization. *Eng. Optim.* **2026**, *58*, 203–239. [CrossRef]
30. Alauthman, M.; Al-Qerem, A.; Almomani, A.; Ishtaiwi, A.M.; Aldweesh, A.; Arafah, M.; Arya, V.; Gupta, B.B. Enhancing Web of Things Security Using Harris Hawks Optimization with Reinforcement Learning. *Int. J. Cogn. Comput. Eng.* **2026**, *7*, 376–388. [CrossRef]
31. Zeng, J.; Chen, M.R.; Guo, Z.; Zhang, Z.; Zhong, S. An Improved Large-Scale Multi-objective Competitive Swarm Optimizer Based on Harris Hawks Optimization. In *International Conference on Behavioural and Social Computing*; Springer Nature Singapore: Singapore, 2025; pp. 30–42.
32. Varshney, M.; Ali, M. A Modified Aquila Optimizer for Application to Plate–Fin Heat Exchangers Design Problem. *Mathematics* **2026**, *14*, 431. [CrossRef]
33. Wreford, A.I. Optimization of Deep Neural Networks for Heart Disease Diagnosis Using the Aquila Optimizer: Bridging AI and Bio-Inspired Computation. *J. Sci. Technol.* **2026**, *4*, 62–70.
34. Sheng, W.; Yang, X.; Jia, D.; Liu, K.; Han, Q.; Li, C. Fault Location Method for Distribution Networks Based on Cluster Partitioning and Arithmetic Optimization Algorithm. *Processes* **2026**, *14*, 493. [CrossRef]
35. Toğan, V.; Said Sulub, A.; Azim Eirgash, M.; Mostofi, F. Project Scheduling to Minimize the Time and Cost in Large-Scale Construction Projects with Repulsion-Based Improved Arithmetic Optimization. *J. Constr. Eng. Manag.* **2026**, *152*, 04025277. [CrossRef]

36. Kyrou, G.; Charilogis, V.; Tsoulos, I.G. A Novel Magnificent Frigatebird Optimization Algorithm with Proposed Movement Strategies for Enhanced Global Search. *Analytics* **2025**, *5*, 1. [CrossRef]
37. Lin, J.; Zhang, X.; Li, R.; Huang, Y.; Qin, Q. Multi-Strategy Enhanced Hybrid Salp Swarm Algorithm for Robust Association Rule Mining in Dynamic Environments. *Int. J. Pattern Recognit. Artif. Intell.* **2026**, *40*, 2650005. [CrossRef]
38. Nguyen, K.D.; Tran, T.T.; Vo, D.N. An efficient Salp Swarm Algorithm for optimizing wind farm layouts with neighboring farm impacts. *Int. J. Green Energy* **2026**, *23*, 420–435. [CrossRef]
39. Chen, J.; Zhu, H.; Shu, T.; Cao, C.; Deng, Y.; Cheng, Q. Balanced Grey Wolf Optimizer Algorithm for Backpropagation Neural Networks. *Mathematics* **2026**, *14*, 554. [CrossRef]
40. Adhikary, J.; Acharyya, S. Randomized Balanced Grey Wolf Optimizer (RBGWO) for solving real life optimization problems. *Appl. Soft Comput.* **2022**, *117*, 108429. [CrossRef]
41. Wang, J.; Lin, D.; Zhang, Y.; Huang, S. An adaptively balanced grey wolf optimization algorithm for feature selection on high-dimensional classification. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105088. [CrossRef]
42. Amusat, R.O.; Ocheni, U.U.; Shodiya, S.; Salau, R. Enhanced MPPT Using Hybrid Smell Agent and Particle Swarm Optimization under Partial Shading in PV Arrays. *Uniabuja J. Eng. Technol. (UJET)* **2026**, *3*, 71–81.
43. Kyrou, G.; Tsoulos, I.G.; Gianni, A.M.; Charilogis, V. Parallel smell agent optimization (SAO): Collaborative subpopulations for accelerated convergence. *Symmetry* **2025**, *17*, 592. [CrossRef]
44. Stripinis, L.; Paulavičius, R. An extensive numerical benchmark study of deterministic vs. stochastic derivative-free global optimization algorithms. *arXiv* **2022**, arXiv:2209.05759. [CrossRef]
45. Ali, M.M.; Kaelo, P. Improved particle swarm algorithms for global optimization. *Appl. Math. Comput.* **2008**, *196*, 578–593. [CrossRef]
46. Koyuncu, H.; Ceylan, R. A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems. *J. Comput. Des. Eng.* **2019**, *6*, 129–142. [CrossRef]
47. Siarry, P.; Berthiau, G.; Durdin, F.; Haussy, J. Enhanced simulated annealing for globally minimizing functions of many-continuous variables. *ACM Trans. Math. Softw. (TOMS)* **1997**, *23*, 209–228. [CrossRef]
48. LaTorre, A.; Molina, D.; Osaba, E.; Poyatos, J.; Del Ser, J.; Herrera, F. A prescription of methodological guidelines for comparing bio-inspired optimization algorithms. *Swarm Evol. Comput.* **2021**, *67*, 100973. [CrossRef]
49. Tsoulos, I.G.; Charilogis, V.; Kyrou, G.; Stavrou, V.N.; Tzallas, A. OPTIMUS: A Multidimensional Global Optimization Package. *J. Open Source Softw.* **2025**, *10*, 7584. [CrossRef]
50. Ali, M.M.; Khompatraporn, C.; Zabinsky, Z.B. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *J. Glob. Optim.* **2005**, *31*, 635–672. [CrossRef]
51. Floudas, C.A.; Pardalos, P.M.; Adjiman, C.; Esposito, W.R.; Gümüs, Z.H.; Harding, S.T.; Klepeis, J.L.; Meyer, C.A.; Schweiger, C.A. *Handbook of Test Problems in Local and Global Optimization*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 33.
52. Jones, D.R.; Martins, J.R. The DIRECT algorithm: 25 years later. *J. Glob. Optim.* **2021**, *79*, 521–566. [CrossRef]
53. Luo, B.; Su, X.; Zhang, S.; Yan, P.; Liu, J.; Li, R. Analysis of a novel gas cycle cooler with large temperature glide for space cooling. *Energy* **2025**, *326*, 136294. [CrossRef]
54. Keerthika, R.; Niranjan, S.P.; Komala Durga, B. A Survey on the tandem queueing models. *Scope* **2025**, *14*, 134–148.
55. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [CrossRef] [PubMed]
56. Suryadevara, S.; Yanamala, A.K.Y. A Comprehensive Overview of Artificial Neural Networks: Evolution, Architectures, and Applications. *Rev. Intel. Artif. Med.* **2021**, *12*, 51–76.
57. Kelly, M.; Longjohn, R.; Nottingham, K. The UCI Machine Learning Repository. Available online: https://archive.ics.uci.edu (accessed on 25 May 2026).
58. Derrac, J.; Garcia, S.; Sanchez, L.; Herrera, F. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult. Valued Log. Soft Comput.* **2015**, *17*, 255–287.
59. Kononenko, I.; Šimec, E.; Robnik-Šikonja, M. Overcoming the myopia of inductive learning algorithms with RELIEFF. *Appl. Intell.* **1997**, *7*, 39–55. [CrossRef]
60. Perantonis, S.J.; Virvilis, V. Input feature extraction for multilayered perceptrons using supervised principal component analysis. *Neural Process. Lett.* **1999**, *10*, 243–252. [CrossRef]
61. Garcke, J.; Griebel, M. Classification with sparse grids using simplicial basis functions. *Intell. Data Anal.* **2002**, *6*, 483–502. [CrossRef]
62. Cestnik, B.; Kononenko, I.; Bratko, I. ASSISTANT 86: A knowledge-elicitation tool for sophisticated users. In Proceedings of the 2nd European Conference on European Working Session on Learning, Bled, Yugoslavia, 1 May 1987; pp. 31–45.

63. Smith, J.W.; Everhart, J.E.; Dickson, W.C.; Knowler, W.C.; Johannes, R.S. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Annual Symposium on Computer Application in Medical Care, Washington, DC, USA, 6–9 November 1988; p. 261.
64. Wolberg, W.H.; Mangasarian, O.L. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proc. Natl. Acad. Sci. USA* **1990**, *87*, 9193–9196. [CrossRef]
65. Raymer, M.L.; Doom, T.E.; Kuhn, L.A.; Punch, W.F. Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2003**, *33*, 802–813. [CrossRef]
66. Zhong, P.; Fukushima, M. Regularized nonsmooth Newton method for multi-class support vector machines. *Optim. Methods Softw.* **2007**, *22*, 225–236. [CrossRef]
67. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
68. Xia, J.H.; Kumta, A.S. Feedforward neural network trained by BFGS algorithm for modeling plasma etching of silicon carbide. *IEEE Trans. Plasma Sci.* **2010**, *38*, 142–148. [CrossRef]