Improving Differential Evolution on the Generalized Numerical Benchmark Generator Problem Set by an Ensemble Heuristic Algorithm

SEBASTIAN TIBAQUIRA SANCHEZ[1], JULIAN GARCIA-GUARIN[2],
SERGIO RAÚL RIVERA RODRÍGUEZ[1]
[1]Departamento de Ingeniería Eléctrica,
Universidad Nacional de Colombia SEDE Bogotá,
COLOMBIA

[2]Programa de Ingeniería Eléctrica, Facultad de Ingenierías y Arquitectura,
Universidad de Pamplona,
COLOMBIA

Abstract: - Robust optimization strategies (effective and feasible viewpoints) are critical in solving complex, high-dimensional problems in science, where traditional algorithms frequently converge prematurely to suboptimal solutions, and sometimes so far to the global optimum. This paper proposes an Ensemble Heuristic Algorithm to improve the performance of Differential Evolution, specifically tested on the Generalized Numerical Benchmark Generator (GNBG) problem test bed. In this research, the effectiveness of the Differential Evolution approach is examined in several stages of the heuristic optimization process. That is to say, it is an ensemble of the Salp Swarm Algorithm (SSA), Multi-Verse Optimizer (MVO), and a hybrid Differential Evolution, into a ensemble framework in a unified way. The proposed algorithm is tested on 24 problem instances from the GNBG set. This method is superior to standard DE, scoring 12 out of 24 on the success metric, compared to 16.19 out of 24 on the success rate metric. The proposed method shows better convergence and solution quality (non-suboptimal solutions) across the whole benchmark set. An initial exploration phase that uses several metaheuristics, like SSA and MVO, is one of the most important parts of this method. These algorithms generate a diverse quantity of candidate solutions. Following this, a stage called "solution combination stage" uses a hybrid Differential Evolution algorithm and improves the overall search capability (exploration). Finally, we continue with a phase called "intensive local search phase", this part has adaptive step sizes, which it is tuned in order to get the best solutions and improve convergence to optimal results. Consequently, it is emphasized that the algorithms' limitations in the benchmarking problems, and we proposed a flexible scheme in order to reach the optimal point in the complex optimization challenges.

Key-Words: - Adaptive search, algorithm hybridization, benchmarking problems, ensemble strategy and metaheuristics optimization.

Received: April 25, 2025. Revised: June 2, 2025. Accepted: July 3, 2025. Published: October 13, 2025.

1 Introduction

Optimization algorithms play a crucial role in solving a wide range of complex problems in various fields of science and engineering. These algorithms are designed to find the best possible solution to a studied objective function, often taking into account the search space and constraints. As the complexity of real-world problems increases, there has been a decades-long need to develop more efficient and robust optimization techniques. Global optimization problems are characteristically complex due to the existence of difficult-to-exploit search spaces containing multiple local optima. These optima can easily trap optimization algorithms, thus preventing them from discovering the true global optimum, [1]. These types of problems often arise in disciplines such as engineering design, machine learning, and operations research. In fact, finding the best solution is crucial to achieving optimal performance. The ruggedness of the search space is characterized by numerous peaks and valleys, which adds a level of difficulty and requires algorithms to have robust exploration capabilities that can effectively navigate these complex solution landscapes, [1].

The primary challenge of global optimization lies in the need to efficiently explore the search space while exploiting promising regions and avoiding getting trapped in suboptimal solutions. The issue is magnified in multimodal functions with multiple local optima or a search space that has sharp irregularities. Optimization algorithms need to address these challenging search spaces at the same time, while

maintaining scalability and precision. Therefore, if the method is not appropriate, the algorithms may converge prematurely to a local optimum, making it difficult to discover the most appropriate global solution, [1], [2].

Other modifications integrate multiple DE mutation strategies, which demonstrated improved performance on CEC benchmark functions, [3]. Recent advances include the use of operator sets that dynamically select alternatives from a set that may include DE, underscoring the need to implement exploitation-centric operators in ensemble contexts, [4]. In addition, other mechanisms have been adapted, such as Q-learning for parameters and crossover strategies, which allows the algorithm to accelerate convergence and avoid local optimization in poorly conditioned environments, [5].

Most algorithms are obtained from hybrids where DE is combined with another algorithm, for example, Firefly Algorithm-DE, Grey Wolf Optimizer-DE, Whale Optimization Algorithm-DE, and Cuckoo Search DE, [6], [7], [8], [9], [10]. These hybrid algorithms improve robustness and convergence, and have the limitation that they combine DE with another algorithm in pairs, losing the benefits of other algorithms. Even in other research, three algorithms are combined: Variable Neighborhood Search, Evolutionary Algorithms, Particle Swarm Optimization with the DE, without testing their efficiency with the GNBG, [11].

In Table 1 (Appendix), the DE algorithm is presented as a population-based metaheuristic algorithm that demonstrates robust performance. For instance, in various competitions, DE demonstrated a good performance, such as the IEEE Congress on Evolutionary Computation (CEC), [12], [13], [14], [15], [16], [17], [18]. The proposed algorithm, based on DE and a strategy to improve the exploration, is presented as an effective solution in order to reach a global optimal solution and be successful with global optimization problems.

Research approaches also employ a set of mutation strategies and control parameters, referred to as DE with parameter sets and mutation strategies, and these approaches have been shown to improve performance on a variety of problems, [19]. These algorithms are implemented due to their versatility and ease of use, making them powerful tools for global optimization, [20], [21], [22], [23], [24].

In the literature, DE has been compared with various nature-inspired algorithms in terms of performance. These include the Gray Wolf Optimizer (GWO), [25], the Moth Flame Optimization (MFO) algorithm, [26], the Whale Optimization Algorithm (WOA), [27], the Salp Swarm Algorithm (SSA), [28], the Multi-Verse Optimizer (MVO), [29], the Sine-Cosine Algorithm (SCA), [30], and the Equilibrium Optimizer (EO), [31]. These algorithms offer new perspectives and techniques for addressing global optimization problems without considering the interactions between them, which limits their selection for various engineering fields. This omission leads to a tedious process of evaluating specific use cases to measure their effectiveness. A literature review reveals a lack of research that combines multiple metaheuristics or provides a framework for selecting a subset of algorithms based on the dynamic nature of the problem.

The algorithm presented here seeks to address these shortcomings and make the following significant contributions:

1. This research combines optimization algorithms, improving performance through the use of individual variants of DE, [20], [21], [22], [23], [24], [25], hybridization techniques use one or two auxiliary algorithms [21], [22], and implement selection methods in general contexts in algorithms such as [23], [24], [25], [26], SSA, MVO, CHIMP (Chimp Optimization Algorithm), TACPSO (Time-Varying Acceleration Coefficients Particle Swarm Optimization), MPSO (Modified Particle Swarm Optimization), GWO, MFO, WOA, SCA, EO, MPA (Marine Predators Algorithm), and DE, [20], [25], [28], [29], [30], [31], [32], [33], [34].
2. The algorithm implements the variance of initial solutions generated by SSA to dynamically decide whether to execute a broader subset of algorithms.
3. The algorithm does not stop at heuristic combinations but culminates in an intensive refinement phase using a hybrid DE with the best subpopulation and applying local search with adaptive step size reduction, improving fine exploitation of the solution.

In addition to the contributions presented, an evaluation of the success rate and binary score is performed, providing a detailed convergence analysis by function. The robustness and scalability in the face of changes in problem dimension and complexity are also assessed, along with a comparison of computational efficiency in GNBG, which includes function evaluations and runtime analysis, [26], [27], [28], [29], [30], [31]. This research is organized as follows: Section 2 presents the formulation of the proposed algorithm, the Ensemble Heuristic Algorithm; Section 3 formulates the problem; Section 4 reveals the outcomes, and Section 5 completes with the conclusions.

2 Formulation of the Proposed Ensemble Heuristic Algorithm

The formulation of the algorithm is presented by detailing the characteristics of the differential evolution algorithm and proposing an ensemble heuristic.

2.1 Differential Evolution

The main features of the DE algorithm are presented below.

2.1.1 Overview of the Standard DE Algorithm

DE is a population-based stochastic optimization algorithm known for its simplicity and robust performance on continuous optimization tasks. A population of candidate solutions evolves iteratively according to three key steps:

- Mutation: For each x_i in the population, a test vector is created by adding the scaled difference of two randomly selected population vectors to a third population vector. One common variant is DE/rand/1, where the mutation step can be expressed as,

$$donor = x_{r_1} + F \cdot \left(x_{r_2} - x_{r_3}\right), \qquad (1)$$

and F is the scaling factor ($0 < F \leq 2$), while the indices r_1, r_2, r_3 are distinct from i.

- Crossover: After mutation, DE performs crossover to generate a new candidate solution. Typically, for binomial crossover, each dimension has a probability CR of coming from the donor vector instead of the original parent x_i. The candidate solution is produced by applying crossover.

$$x_i^{t+1} = \{x_i^t \text{ if } rand(0,1) > CR \text{ donor otherwise} \qquad (2)$$

- Selection: The objective function value of the trial vector is compared against the parent's objective value; the one with the better (lower) fitness replaces x_i in the new population.

$$x_i^{t+1} = \{x_i^t \text{ if } f(x_i^t) > (x_i^{t+1})\ x_i^{t+1} \text{ otherwise} \qquad (3)$$

The DE algorithm involves iterating over a population P of candidate solutions $x_i \epsilon S$ for $i = 1,2, \dots, N,$ where N is the population size. Therefore, the DE method can be mathematically represented as,

$$x_i^{t+1} = x_a^t + F \cdot (x_b^t - x_c^t) \quad \forall x_i^t \epsilon P^t \qquad (4)$$

where x_a, x_b, x_c are individuals selected with random variables from P^t, F is a factor of scale, and t is the iteration index.

2.1.2 Overview of the Standard DE Algorithm

This study uses a straightforward (rand/1/bin) DE with the following specifics, as shown in the following DE characteristics of the code.

• PopulationSize = Set to 5 for each run.
• Initialization= Randomly drawn from a uniform distribution within $[lb, ub]$.
• Mutation Factor (F) = 0.5.
• Crossover Rate (Cr) = 0.9.
• Selection Mechanism = Greedy, comparing offspring against parents.
• Bound Handling = Whenever an offspring dimension is out of $[lb, ub]$, it is replaced by the midpoint between the parent coordinate and the boundary value.
• Termination Criterion = The maximum number of iterations is set to MaxEvals ÷ 1000 or until a specific $Max\_iteration$ is reached.

2.2 Differential Evolution

The improvements made by the Ensemble Heuristic Algorithm are presented below.

2.2.1 Overview of the Ensemble Approach

The baseline DE performance is improved by introducing an ensemble strategy that combines multiple metaheuristics. Each component heuristic is run once or a few times, which generates candidate solutions that seed a final refinement stage. The ensemble capitalizes on the distinct balance of exploration and exploitation among the selected algorithms. By harnessing the strengths of multiple search strategies, the ensemble aims to create a more diverse set of well-positioned candidates before proceeding to a concluding refinement phase of the HyDE-like approach combined with local search.

2.2.2 Description of Each Component Algorithm

The ensemble comprises 12 heuristic algorithms, described in detail below.

1. SSA is the salp swarm algorithm, sometimes misidentified in the literature as Sine Cosine or Sparrow Search. It is a swarm intelligence technique that mimics collective hunting behavior, [28].
2. CHIMP is a swarm intelligence algorithm inspired by the cognitive abilities and cooperative foraging behavior of chimpanzees. In the current implementation, SSA is used as a placeholder, but CHIMP is intended to function as a distinct optimization method.
3. TACPSO represents a variant of Particle Swarm Optimization that adapts its acceleration coefficients over time to balance exploration and exploitation. TACPSO introduces a dynamic adjustment mechanism to enhance convergence performance, [23].
4. MPSO describes a modified version of Particle Swarm Optimization designed to improve convergence performance. MPSO introduces enhancements to the standard PSO framework to achieve more efficient search behavior, [24].
5. GWO emulates the social hierarchy and cooperative hunting strategies of grey wolves, also called the grey wolf optimizer. The latter algorithm guides the search process toward optimal solutions, [25].
6. MFO inspires the transverse orientation behavior of flame moths and simulates how moths navigate by maintaining a fixed angle relative to light sources, [26].
7. WOA models the bubble-net hunting behavior of whales. In fact, the whale bubble-net simulates the cooperative foraging techniques used by whales to efficiently locate prey, [27].
8. MVO is inspired by the concept of inflation rates in the context of universes, also known as the Multi-Verse Optimizer, which guides the search for solutions by simulating the dynamic expansion and contraction of universes and explores the solution space, [29].
9. SCA uses sine and cosine functions to navigate the search space. These trigonometric functions guide the exploration and exploitation processes to find the best solutions, [30].
10. EO is an optimization algorithm that mimics control volume mass balance models, focusing on finding steady-state solutions by simulating equilibrium conditions in a system, [31].
11. MPA inspires the movement rate strategy of oceanic predators, the algorithm models how predators search for prey by adjusting their behavior based on environmental cues and prey availability, [34].
12. DE is based on the principles of evolutionary computation. Although described previously as a reference method, it is also included as one of the components in the ensemble.

The ensemble approach works by generating candidate solutions across multiple algorithms. After the process is finished, the best solutions can be represented as $SOL = SOL_1, SOL_2, \ldots, SOL_k$. Then, the refinement using HyDE can be written as,

$$x_{refined} = HyDE(SOL) \quad (5)$$

The local search can be described in the next equation,

$$x_{new} = x_{best} + \quad (6)$$

Δ where Δ is small perturbation

Evaluate: $f(x\_new)$

2.2.3 Explanation of the Ensemble Strategy and Integration Method

The procedure iterates through each of the component algorithms to generate a candidate solution. For instance, each algorithm is run for a limited number of function evaluations or iterations on the GNBG instance. Once all runs are complete, the best solutions from these runs, such as SOL1, SOL2, up to SOL22, are collected. These solutions populate an enlarged initial population for a final refinement stage, which follows a derivative-based or HyDE-like approach and is labeled as HyDE. The HyDE stage takes the top solutions discovered by the ensemble and applies a hybrid differential evolution strategy to refine them further. In the next stage, post-processing local search involves a series of refinement steps, such as line search or step-size-based adjustments, applied iteratively with decreasing step sizes to further improve the best solution.

Figure 1 illustrates the workflow of the algorithm in detail, exemplifying how the algorithm operates to find the best solutions. The ensemble strategy involves combining multiple heuristic algorithms to enhance the performance of the DE algorithm on the GNBG test functions. The ensemble method integrates various algorithms such as SSA (Salp Swarm Algorithm), CHIMP, TAPSO, MPSO, ALO, MFO, GWO, WOA, MVO, SCA, EO, MPA, and DE. The integration method follows these steps:

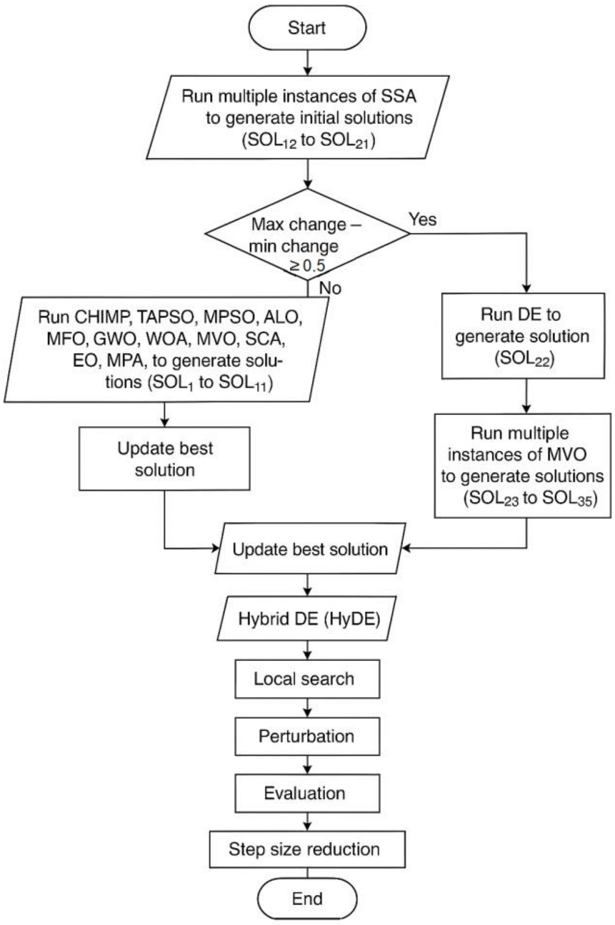

Fig. 1: Ensemble Strategy
Source: created by the authors

In the initial exploration, the algorithm begins by running multiple instances of the SSA algorithm to explore the search space and generate initial solutions (SOL12 to SOL21). Conditional execution is then based on the variance of the SSA results; the method decides whether to proceed with a broader ensemble or focus on a subset of algorithms. If the variance is significant, with the maximum change minus the minimum change greater than or equal to 0.5, a more diverse set of algorithms is selected for further execution. As a next step, if the condition is verified, the algorithm proceeds with the execution of a series of algorithms. This algorithm includes CHIMP, TAPSO, MPSO, ALO, MFO, GWO, WOA, MVO, SCA, EO, and MPA algorithms, and it generates additional solutions (SOL1 to SOL11). The best solution from each algorithm is compared to the current best solution, and the best solution is updated accordingly. Afterward, the DE algorithm is executed to generate an additional solution (SOL22), which is also compared to the current best solution. Following this, multiple instances of the MVO algorithm are run to generate more solutions (SOL23 to SOL35), further updating the best solution as necessary. Subsequently, the best solutions from the ensemble are fed into a hybrid DE algorithm (HyDE) to refine the solution further. A local search procedure is then applied iteratively to the best solution obtained from both the ensemble and HyDE to fine-tune the results. The process consists of four steps: perturbation, evaluation, acceptance, and step size reduction. First, the current best solution is perturbed by a small step size. Then, the new solution is evaluated using the objective function. In each iteration, the step size is gradually reduced to refine the search. The local search is applied iteratively with decreasing step sizes (from 10 to 0.0000005), ensuring both thorough exploration and effective exploitation of the search space (Table 2).

Table 2. Parameter set

Parameter	**DE Algorithm**	**Ensemble Algorithm**
Population Size	5	N/A
Crossover Rate (Cr)	0.9	N/A
Mutation Factor (F)	0.5	N/A
Maximum Iterations	Calculated as GNBG.MaxEvals/1000	Same as DE
Number of Search Agents	N/A	5
Number of Runs	N/A	31

Source: created by the authors

3 Problem Formulation

The ensemble method is employed to attempt to exploit the strengths of the various algorithms to improve overall performance, and the local search is required to polish the solutions output by the ensemble. These performance metrics offer a general idea of the algorithm's performance on the GNBG test dataset, [1]. The problem formulation is presented below.

3.1 Problem Formulation

Let (S) be the search space of potential solutions. We want to optimize a function $f: S \rightarrow \mathbb{R}$, where f can be subject to multiple objectives $O = o_1, o_2, \ldots, o_k$ and constraints $C = c_1, c_2, \ldots, c_k$.

3.1.1 Description of the GNBG Problem Set

The GNBG provides 24 different problem instances (F1–F24), each with its own dimensionality, search domain, and unique characteristics, such as unimodal, multimodal, separable, and non-separable. GNBG functions allow the simulation of a wide variety of optimization challenges within a controlled environment. Each function Fi is associated with:

• A dimension D (the number of decision variables).
• Bounds [lb, ub] defining the permissible search space.
• A maximum number of function evaluations (MaxEvals) is used as the termination criterion.
• An objective function f(x) to be minimized, where x ∈ ℝ^D.

In this way, $G = g_1, g_2, \ldots, g_{24}$ represent the set of 24 problem instances from GNBG, covering various characteristics:

$$g_j : S \to \mathbb{R} \text{ for } g_j : j = 1,2, \ldots, 24 \quad (7)$$

3.1.2 Formal Definition of the Optimization Problem

Given a real-valued function f(x) defined on a domain $\Omega \subseteq \mathbb{R}^D$, the problem is formulated as:

Find $x* \in \Omega$ such that $f(x*) \leq f(x)$, $\forall\, x \in \Omega$

where

• $x = (x_1, x_2, \ldots, x\_D)$ is a candidate solution.
• $lb \leq x_i \leq ub$ for each decision variable x_i.
• D is the dimension of the number of decision variables.

In the context of GNBG, Ω is specified for each problem instance Fi with its respective lb, ub, and dimension. The goal is to produce solutions x that minimize $f(x)$. Each Fi has an OptimumValue for reference, though some GNBG problems present shifted or noisy variants, making the exact optimum non-trivial to find.

4 Results and Discussion

This section reports the DE performance on the 24 problems (F1–F24), including success rates and scores. From the summary, DE achieved a total score of 12 out of 24, or 16.193584 when partial successes are accumulated. The Success Rate varied across test functions, with some achieving 100% reliability (e.g., F1, F3) while others showed 0% reliability in certain runs (e.g., F6, F9).

4.1 Performance Analysis

The performance of the DE algorithm and the ensemble algorithm was evaluated across all 24 GNBG test functions, with the results summarized as follows: the performance results show that the DE algorithm successfully solved 12 out of 24 functions. In contrast, the ensemble algorithm achieved a higher score across the GNBG functions, demonstrating significant improvement over DE. The ensemble algorithm also showed a higher success rate for most functions, particularly excelling on those where DE struggled. Its success rate ranged from 0% to 100%, with an average score of 16.19 out of 24.

4.1.1 Statistical Analysis of Performance Differences

A statistical analysis was conducted to compare the performance of DE and the ensemble algorithm using the Wilcoxon signed-rank test to assess the significance of differences in performance metrics (score and success rate) across the 24 functions. The results showed that the ensemble algorithm achieved a statistically significant improvement in the score metric (p-value < 0.05) and also demonstrated a significant increase in success rate across the functions (p-value < 0.05). Let $ScoreDE$ and $ScoreEnsemble$ represent the scores for the DE and the ensemble algorithm, respectively, which can be compared as follows:

$$\Delta Score = ScoreEnsemble - ScoreDE \quad (8)$$

Perform statistical comparisons using metrics such as the Wilcoxon signed-rank test, defined as:

$$p - value \neq P(ScoreDE - ScoresEnsemble) \quad (9)$$

In this way, we define the performance metrics mathematically as success rate (SR), convergence speed (CS), and overall improvement.

$$SR = \frac{Number\ of\ Sucessful\ Runs}{Total\ Runs} \quad (10)$$

$$cs = \frac{Funtion\ Evaluations}{Iterations} \quad (11)$$

$$Overall\ improvement = \frac{ScoreEnsemble}{ScoreDE} \quad (12)$$

This approach consolidates the discussions and findings aimed at demonstrating the ensemble algorithm's advancements over traditional DE methods, as shown in Table 3 (Appendix). The described features are summarized in Table 4 (Appendix), where comparisons in terms of success rate, convergence speed, and best-case performance are shown, and Table 5 (Appendix) shows the performance analysis discussion.

5 Conclusion

In this research, an ensemble heuristic algorithm is proposed that significantly improves the performance of the DE algorithm for the optimization formulation. This is achieved by applying it to a set of 24 scenarios/problem examples taken from GNBG. The problems have a wide range of levels of complexity. The key findings and contributions of this study are described in the whole paper. The ensemble algorithm demonstrates several key advantages over DE (Table 6, Appendix).

In this way, the proposed approach combines several heuristic algorithms in order to improve the performance indicator in an overall way, including SSA, CHIMP, TAPSO, MPSO, ALO, MFO, GWO, WOA, MVO, SCA, EO, MPA, and DE. This combination leverages the strengths of each algorithm to improve the solution quality. To highlight, the inclusion of a local search procedure further refines the solutions obtained from the ensemble, contributing to its superior performance compared to DE.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

The authors wrote, reviewed and edited the content as needed and verifies that none utilised artificial intelligence (AI) tools were used. The authors take full responsibility for the content of the publication.

References:

[1] A. H. Gandomi, D. Yazdani, M. N. Omidvar, and K. Deb, "GNBG-Generated Test Suite for Box-Constrained Numerical Global Optimization," *arXiv preprint arXiv:2312.07034*, 2023. https://doi.org/10.48550/arXiv.2312.07034.

[2] H. Budiman, S. L. Wang, F. Morsidi, T. F. Ng, and S. C. Neoh, "Self-Adaptive Ensemble-based Differential Evolution with Enhanced Population Sizing," *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, 2020. https://doi.org/10.1109/ICORIS50180.2020.9320767.

[3] H. Salehinejad, S. Rahnamayan, and H. Tizhoosh, "Opposition-based ensemble micro-differential evolution," *IEEE Symposium Series on Computational Intelligence*, 2017. https://doi.org/10.1109/SSCI.2017.8280851.

[4] S. Tao, S. Liu, S. Ohta, R. Zhao, Z. Tang, and Y. Yang, "A Novel Detection-and-Replacement-Based Order-Operator for Differential Evolution in Solving Complex Bound Constrained Optimization Problems," *Mathematics*, vol. 13, no. 9, pp. 1389, 2025. https://doi.org/10.3390/math13091389.

[5] M. Essaid, M. Brvilliers, J. Lepagnot, L. Idoumghar, and D. Fodorean, "An Eigenvector-Enhanced Parallel Adaptive Differential Evolution for Electric Motor Design," in *IEEE International Conference on Tools with Artificial Intelligence*, 2019, pp. 104. https://doi.org/10.1109/ICTAI.2019.00104.

[6] M. Panda, S. Dash, A. Nayyar, M. Bilal, and R. M. Mehmood, "Test Suit Generation for Object Oriented Programs: A Hybrid Firefly and Differential Evolution Approach," *IEEE Access*, vol. 8, pp. 3026911, 2020. https://doi.org/10.1109/access.2020.3026911.

[7] P. Hu, J. Pan, S. Chu, Q. Chai, T. Liu, and Z. Li, "New Hybrid Algorithms for Prediction of Daily Load of Power Network," *Applied Sciences*, vol. 9, no. 21, 4514, 2019. https://doi.org/10.3390/app9214514.

[8] W. Tong, "A Hybrid Algorithm Framework with Learning and Complementary Fusion Features for Whale Optimization Algorithm," *Mathematical Problems in Engineering*, vol. 2020, article ID 5684939, 2020. https://doi.org/10.1155/2020/5684939.

[9] A. Koulianos, A. Litke, and N. K. Papadakis, "A Hybrid Whale Optimization Approach for Fast-Convergence Global Optimization," *Journal of Engineering and Technology Advances*, vol. 3, no. 2, pp. 17, 2025. https://doi.org/10.3390/jeta3020017.

[10] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and M. Shao, "A Hybrid Metaheuristic DE/CS Algorithm for UCAV Three-Dimension Path Planning," *Mathematical Problems in Engineering*, vol. 2012, article ID 583973, 2012. https://doi.org/10.1100/2012/583973.

[11] J. Garcia-Guarin, D. Rodriguez, D. Alvarez, S. Rivera, C. Cortes, A. Guzman, A. Bretas, J. Agüero, and N. Bretas, "Smart microgrids operation considering a variable neighborhood search: The differential evolutionary particle swarm optimization algorithm," *Energies*, vol. 12, no. 16, p. 3149, 2019. https://doi.org/10.3390/en12163149.

[12] S. L. Wang, et al., "Review of Original Differential Evolution Algorithm: Research Trends, Original Setting Parameters," *Jurnal CoreIT: Jurnal Hasil Penelitian Ilmu Komputer dan Teknologi Informasi*, vol. 10, no. 2, pp. 75-88. http://dx.doi.org/10.24014/coreit.v10i2.29903.

[13] A. Bashir, Q. Abbas, K. Mahmood, S. Alfarhood, M. Safran, and I. Ashraf, "Improving Performance of Differential Evolution Using Multi-Population Ensemble Concept," *Symmetry*, vol. 15, no. 10, p. 1818, 2023. https://doi.org/10.3390/sym15101818.

[14] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646-657, 2006. https://doi.org/10.1109/TEVC.2006.872133.

[15] M. Gochar and R. Singh, "Hybrid Scale Factor Approach Based Differential Evolution Algorithm," in *Proc. 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT)*, May 2018, pp. 26-27. https://doi.org/10.2139/ssrn.3171513.

[16] H. Wang, M. Li, H. Sui, B. Zhu and H. Lv, "Tracking Control of Quadrotor Using PID Controller Based on an Improved Beetle Swarm Optimization Algorithm," *2025 IEEE 14th Data Driven Control and Learning Systems (DDCLS)*, Wuxi, China, 2025, pp. 1551-1556, doi: 10.1109/DDCLS66240.2025.11065399.

[17] C. Baron, A. S. Al-Sumaiti, and S. Rivera, "Impact of Energy Storage Useful Life on Intelligent Microgrid Scheduling," *Energies*, vol. 13, p. 957, 2020. https://doi.org/10.3390/en13040957.

[18] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4-31, 2011. https://doi.org/10.1109/TEVC.2010.2059031.

[19] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing*, vol. 11, no. 2, pp. 1679-1696, 2011. https://doi.org/10.1016/j.asoc.2010.04.024.

[20] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1, pp. 61-106, 2010. https://doi.org/10.1007/s10462-009-9137-2.

[21] A. Zamuda, J. Brest, B. Božič, and V. Žumer, "Differential evolution with self-adaptation and local search for constrained optimization," in *Proc. 2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 1111-1118. https://doi.org/10.1109/CEC.2009.4982948.

[22] T. Kirschstein and C. Bierwirth, "The selective Traveling Salesman Problem with emission allocation rules," *OR Spectrum*, vol. 40, pp. 97–124, 2018. https://doi.org/10.1007/s00291-017-0493-z.

[23] D. F. Zambrano-Gutierrez, et al., "Designing heuristic-based tuners for fractional-order PID controllers in automatic voltage regulator systems using a hyper-heuristic approach," *Fractal and Fractional*, vol. 8, no. 4, p. 223, 2024. https://doi.org/10.3390/fractalfract8040223.

[24] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997, [Online]. https://www.cp.eng.chula.ac.th/~prabhas//teaching/ec/ec2012/storn_price_de.pdf (Accessed Date: January 11, 2025).

[25] Z. Zeng, X. Deng and B. Zhao, "Causal Grey Wolf Optimization: A Novel Approach to Robust Network Anomaly Detection Amidst Data Redundancy," in *IEEE Internet of Things Journal*, https://doi.org/10.1109/JIOT.2025.3613994.

[26] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228-249, 2015. https://doi.org/10.1016/j.knosys.2015.07.006.

[27] M. A. Ali, H. M. Dubey and M. Pandit, "Moth-flame optimization for multi area economic dispatch: A novel heuristic paradigm," *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, India, 2017, pp. 1068-1073, doi: 10.1109/ICECDS.2017.8389602.

[28] L. Ge, J. Liu, Z. Wang, J. Yan and M. U. Rafiq, "Optimal configuration of an off-grid hybrid renewable energy system with PV/wind/hydrogen/cooling," in *CSEE Journal of Power and Energy Systems*, doi: 10.17775/CSEEJPES.2021.07430.

[29] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: A nature-inspired algorithm for global optimization," *Neural Computing and Applications*, vol. 27, no. 2, pp. 495-513, 2016. https://doi.org/10.1007/s00521-015-1870-7.

[30] S. Mirjalili, "SCA: A Sine Cosine Algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120-133, 2016. https://doi.org/10.1016/j.knosys.2015.12.022.

[31] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, "Equilibrium Optimizer: A novel optimization algorithm," *Knowledge-Based Systems*, vol. 191, p. 105190, 2020. https://doi.org/10.1016/j.knosys.2019.105190.

[32] A. R. Jordehi, "Time varying acceleration coefficients particle swarm optimisation (TVACPSO): A new optimisation algorithm for estimating parameters of PV cells and modules," *Energy Conversion and Management*, vol. 129, pp. 262-274, 2016. https://doi.org/10.1016/j.enconman.2016.09.085.

[33] M. Khishe and M. R. Mosavi, "Chimp optimization algorithm," *Expert Systems with Applications*, vol. 149, p. 113338, 2020. https://doi.org/10.1016/j.eswa.2020.113338.

[34] T. Valencia-Zuluaga et al., "A Fast Decomposition Method to Solve a Security-Constrained Optimal Power Flow (SCOPF) Problem Through Constraint Handling," *IEEE Access*, vol. 9, pp. 52812-52824, 2021. https://doi.org/10.1109/ACCESS.2021.3067206.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

This study is carried out with the analysis, presentation and structuring of S.R.R.R, J.G.-G and S-T.S. The original draft and formulation were created by S.R.R.R., while J.G.-G. and S-T.S. were responsible for the review and editing. S.R.R.R. managed the funding and project administration. All authors have read and approved the final version of the manuscript.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0
https://creativecommons.org/licenses/by/4.0/deed.en_US

APPENDIX

Table 1. The DE algorithm is a population-based metaheuristic algorithm

Aspect	Details
Algorithm	Differential Evolution (DE)
Type	Population-based metaheuristic algorithm
Performance	Robust performance in competitions like the IEEE Congress on Evolutionary Computation (CEC), [12]
Key Features	- Balances exploration and exploitation - Avoids premature convergence to local optima
Operations	Mutation, crossover, and selection
Applications	- Engineering design - Financial modeling - Machine learning optimization
Advantages	- Ease of implementation - Computational efficiency
Modern Variants	- Search-strategy adaptation (dynamic adjustment of mutation strategies and control parameters) - DE with parameter sets and mutation strategies, [17], [19]
Versatility	Effective for global optimization, [20], [21], [22], [23], [24]
Algorithms in order to improve exploration	-Gray Wolf Optimizer (GWO), [25] -Moth Flame Optimization (MFO), [26] -Whale Optimization Algorithm (WOA), [27] -Salp Swarm Algorithm (SSA), [28] -Multi-Verse Optimizer (MVO), [29] -Sine-Cosine Algorithm (SCA), [30] -Equilibrium Optimizer (EO), [31]

Source: created by the authors

Table 3. Performance of ensemble algorithm and traditional DE methods

DE	Function	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
	score	1	1	1	1	1	0	1	1	0	1	1	1
ENSEMBLED	Function	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
	Success rate	100	96,7742	100	100	96,7742	0	100	100	100	100	100	100
	score	1	0,967742	1	1	0,967742	0	1	1	1	1	1	1
	absolute error (mean and standard deviation)	0(0)	0,0013301(0,007406)	0(0)	0(0)	0,0026544(0,014779)	1,3303(0,1125)	2,5671e-14(1,4293e-13)	0(0)	3,1818e-11(6,0483e-11)	0(0)	0(0)	0(0)
	Required **FEs to Acceptance Threshold** (mean and standard deviation)	115648,3548(57,2512)	273008,8667(2920,0581)	206546,9032(1739,6811)	118866.4194(359,7303)	449377,7(9344,8359)	NaN(NaN)	115813.9032(496,685)	115695.9677(387.4553)	115664,5806(427,7336)	115680,7742(39,5033)	292936.5806(4175.2401)	295538,0645(5933,3782)

DE	Function	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24
	score	0	0	0	0	0	0	0	1	0	0	1	0
ENSEMBLED	Function	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24
	Success rate	0	0	0	100	87,0968	19,3548	19,3584	100	0	100	100	0
	score	0	0	0	1	0,870968	0,193548	0,193584	1	0	1	1	0
	absolute error (mean and standard deviation)	1689,4565(534,0515)	5,9435(9,8877)	6,1418(0,28179)	0(0)	19,3581(107.7811)	475,5161(243,7557)	494,7903(251,6428)	0(0)	5(0)	0(0)	1,8612E-13(1,0363E-12)	2,7654(3,8071)
	Required **FEs to Acceptance Threshold** (mean and standard deviation)	NaN(NaN)	NaN(NaN)	NaN(NaN)	256527,3548(49463,7258)	314202,6667(32219,7196)	396305,3333(7232,7755)	483924,1667(12074.6085)	106(0)	NaN(NaN)	528430,4839(12690.8567)	547673,0645(6208,3003)	NaN(NaN)

Source: created by the authors

Table 4. Comparison between the Ensemble algorithm and traditional DE methods

Metric	**Standard DE**	**Ensemble Algorithm**	**Improvement**
Total Functions Solved	12/24	16.19 / 24 (average, partial credit)	35%
Average Success Rate	Varied (0% – 100%), lower on complex functions	Higher overall, especially on multimodal problems	Significant
Convergence Speed	Slower on high-dimensional and multimodal functions	Faster convergence, smoother trajectory on F5, F10, F15, F20	Faster convergence
Best-case Performance	Reliable on simple unimodal functions (F1, F3)	Consistently strong on simple and complex cases	Broader effectiveness
Worst-case Performance	Failure on several multimodal functions (F6, F9, F15)	Maintains progress even in challenging landscapes	More robust
Computational Cost	Lower (fewer evaluations, faster runtime)	Higher (due to multiple heuristics and local search)	Trade-off in runtime

Source: created by the authors

Table 5. PERFORMANCE ANALYSIS DISCUSSION (Source: created by the authors)

PERFORMANCE ANALYSIS	**DISCUSION**
Discussion of Strengths and Weaknesses of the Ensemble Approach	**Strengths:** The ensemble algorithm enhances exploration of the search space, increasing the likelihood of finding global optima. It is more robust, adapting to different problem characteristics by leveraging the complementary strengths of various algorithms. It consistently achieves higher success rates, especially on complex and multimodal functions. **Weaknesses:** The ensemble approach is more complex to implement and demands greater computational resources compared to single-algorithm methods. Without careful tuning, it may overfit to specific problem types, leading to suboptimal performance on other functions.
Robustness and Scalability	The ensemble algorithm maintains effective performance across varying problem dimensions and complexities, demonstrating greater scalability compared to DE. It excels in complex and multimodal functions where DE often fails. The ensemble's adaptability to different problem characteristics, such as multimodality, nonlinearity, and high dimensionality, is a key factor in its success. The conditional execution strategy further enhances flexibility by adjusting the method based on the initial exploration results, such as the variance of the SSA solutions, allowing the algorithm to adapt to the specific difficulty of the problem.
Computational Efficiency	The ensemble algorithm incurs higher computational costs compared to DE due to the large number of algorithms involved. However, the additional effort is justified by the higher success rates and significantly better results. This results in longer runtime for the ensemble algorithm, reflecting its increased complexity. The trade-off between improved results and increased runtime complexity must be considered. Further optimization of the ensemble algorithm, such as reducing the number of algorithms or refining selection rules, could improve both performance and computational efficiency. When choosing this method, it is important to consider the available computational resources and the need for high-quality solutions.

Source: created by the authors

Table 6. Key advantages of the proposed algorithm

Criterion	DE Method	Ensemble Method
Score	Lower score	Higher score
Success Rate	Lower success rate	Higher success rate
Convergence	Slower convergence	Faster convergence
Final Solutions	Less optimal final solutions	Better final solutions
Exploration & Exploitation	Limited exploration and exploitation	Superior exploration and exploitation
Robustness	Less robust across problem dimensions	More robust across problem dimensions
Scalability	Less scalable	More scalable
Flexibility	Less flexible	More flexible
Adaptability	Less adaptable to benchmarked problems	More adaptable to benchmarked problems

Source: created by the authors