Optimizing benchmark functions using Atomic Orbital Search (AOS) with ABC for parameter selection

Julia Sánchez Esquivel

Universidad Politécnica de Madrid, Madrid 28660, Spain
julia.sanchez.esquivel@alumnos.upm.es

Abstract. This project aims to implement the Atomic Orbital Search (AOS) metaheuristic for solving mathematical optimization problems and to enhance its performance using parameter optimization through Artificial Bee Colony (ABC). The project will compare AOS's performance with Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) using metrics such as convergence speed, computational time, and solution accuracy; statistical comparisons are included. Simulations are carried out in Python, and the results are visualized in tables.

1 Introduction

Function optimization is a fundamental task in computational science, where metaheuristics like Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and the new Atomic Orbital Search (AOS) are used to solve complex problems. Each algorithm has unique features, but their performance heavily depends on parameter tuning. This work focuses on optimizing benchmark functions using AOS, PSO, and GA, with parameters tuned through the Artificial Bee Colony (ABC) algorithm. The study compares the performance of these metaheuristics under standardized conditions.

1.1 Background

Metaheuristics have proven to be effective in solving optimization problems in large and complex search spaces. GA and PSO are well-established and extensively applied, with their performance shown to depend on careful tuning of parameters like crossover rates in GA or learning coefficients in PSO. AOS, is a newer algorithm that models interactions within search spaces. However, optimizing its parameters remains underexplored. This study leverages the ABC algorithm to address this gap, applying it to optimize the parameters of AOS, PSO, and GA, and evaluating their performance on standard benchmark functions.

1.2 Motivation

The Atomic Orbital Search (AOS) is an innovative metaheuristic that remains underexplored, particularly in parameter optimization. While it introduces a novel energy-based probabilistic mechanism inspired by quantum mechanics, it can also be viewed as a hybrid or variation of existing methods. Studying AOS offers an excellent opportunity to evaluate not only its performance but also its novelty compared to well-established algorithms like GA and PSO. Function optimization serves as an effective framework for this purpose, providing both visual [5] and quantitative insights into the unique contributions and potential advantages of AOS.

2 Methodology

2.1 Atomic Orbital Search (AOS)

The Atomic Orbital Search (AOS) algorithm is a metaheuristic optimization method inspired by quantum mechanics and the behavior of electrons within atomic orbitals. In standard search terminology, AOS can be understood as a population-based search algorithm where each electron represents a candidate solution, and the nucleus corresponds to the region in the search space where high-quality solutions are concentrated. The algorithm balances exploration and exploitation by mimicking the probabilistic movement of electrons between orbital layers, with exploration achieved through random transitions to diverse regions in the search space and exploitation ensured by adjusting solutions toward better ones, guided by interactions between electrons and the nucleus (figure 1). Additionally, energy absorption and emission mechanisms dynamically regulate the behavior of solutions to prevent premature convergence and ensure thorough search space coverage. [1]

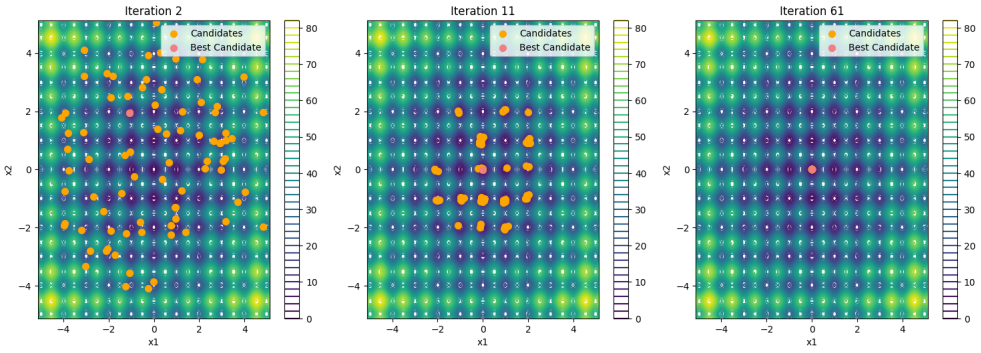

Fig. 1. AOS evolution over time in Rastrigin's optimization

2.2 Objective functions

The test functions used in this study are standard benchmarks in mathematical optimization, including Rastrigin, Sphere, Ackley, and other functions that pose typical challenges for metaheuristics. Each function is characterized by its complexity, dimensionality, and the type of optima it presents (global or local). Throughout this work, a total of 11 objective functions (figure 2) are evaluated, each specifically designed to test the ability of metaheuristics to effectively explore and exploit the search space.

2.3 Parameter selection with ABC

The Artificial Bee Colony (ABC) algorithm is employed via NiaPy library [3] to optimize the parameters of the three metaheuristics. For AOS, ABC aims to optimize the number of particles, the number of layers, and the photon rate. Similarly, for PSO [4] and GA [2], ABC selects the key parameters influencing their performance, such as population size, crossover and mutation rates in GA, inertia weight, and learning coefficients in PSO. The parameters are selected for each objective function; therefore, it is used to evaluate the performance of each metaheuristic with the given parameters, and the ABC algorithm is responsible for finding the optimal values.

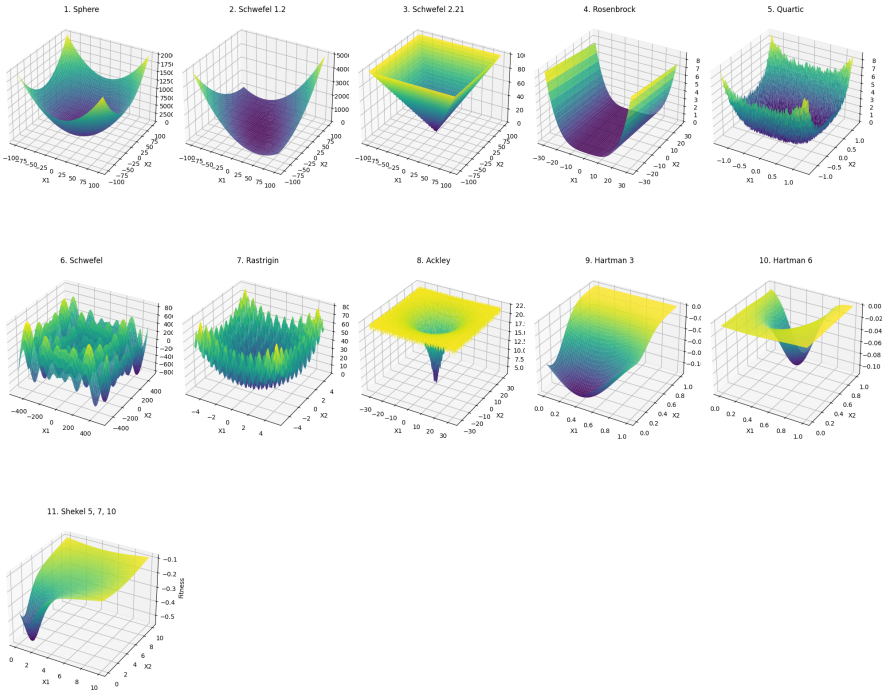

Fig. 2. All functions being tested

3 Results

3.1 Algorithms comparison

Each algorithm is evaluated on 11 objective functions, performing 5 runs per function to ensure the reliability of the results. The parameters of each metaheuristic are optimized based on the configurations defined in the implementation. The results obtained from parameter optimization are presented in tables and graphs, showcasing the ability of each metaheuristic to minimize the cost of the objective functions. AOS, PSO, and GA are compared in terms of the best solutions found, execution time, and stability of the results.

3.2 Performance Analysis

The key metrics evaluated are the **Mean Time (MT)** in seconds, **Best Fitness (BF)**, and the times the optimal solution is found **Optimal Count (OC)**. The **function (F)** and its **optimal fitness (OF)** are also displayed.

GA performs consistently across all functions, with an average mean time of 0.6261 seconds, indicating moderate efficiency. It exhibits a relatively high optimal count (2), showing its ability to find optimal solutions on average in 2 out of 11 benchmark functions. However, its best fitness (BF) varies, with several functions having suboptimal results.

PSO demonstrates faster execution with an average mean time of 0.2644 seconds, making it the quickest among the three algorithms. Despite its speed, its optimal count (3.81) is slightly better than GA, suggesting that PSO is more successful in finding optimal solutions.

		GA			PSO			AOS		
F	**OF**	**MT**	**BF**	**OC**	**MT**	**BF**	**OC**	**MT**	**BF**	**OC**
1	0.0	0.5760	0.0105	0	0.0197	0.0000	5	0.5362	0.0000	5
2	0.0	0.5540	0.0094	1	0.1892	0.0000	5	0.4725	0.0000	5
3	0.0	1.7619	0.0098	1	0.0387	0.8837	0	0.7681	0.0000	5
4	0.0	0.3323	0.0875	0	0.0243	0.0000	5	0.4488	0.0000	5
5	0.0	0.1482	0.0040	4	0.0216	0.0006	5	0.3869	0.0002	5
6	-837.9658	0.6570	-837.9639	3	0.0218	-837.9658	5	0.4258	-837.9634	5
7	0.0	0.5291	0.0004	5	0.0206	0.0000	5	0.3939	0.0000	4
8	0.0	0.6884	0.0072	2	0.0261	0.0000	5	0.5324	0.0000	5
9	-3.86278	0.6607	-3.8627	5	0.7820	-3.8628	5	0.9102	-3.8628	5
10	-3.32237	0.5854	-3.3208	1	0.6847	-3.2030	0	0.8226	-3.3215	1
11	-10.536	0.8808	-10.4954	0	1.0796	-10.5364	2	1.1907	-10.5201	0
Avg.		0.6261	-	2	**0.2644**	-	3.81	0.6703	-	**4**

Table 1. Comparison of GA, PSO, and AOS performance for various benchmark functions with optimal fitness values.

AOS shows a strong performance in terms of optimal count (4), outperforming both GA and PSO. This indicates that AOS is the most reliable algorithm in finding optimal solutions across the benchmark functions. The mean time (0.6703 seconds) for AOS is higher than PSO, but still competitive compared to GA. Additionally, AOS consistently produces competitive best fitness values, indicating its robustness.

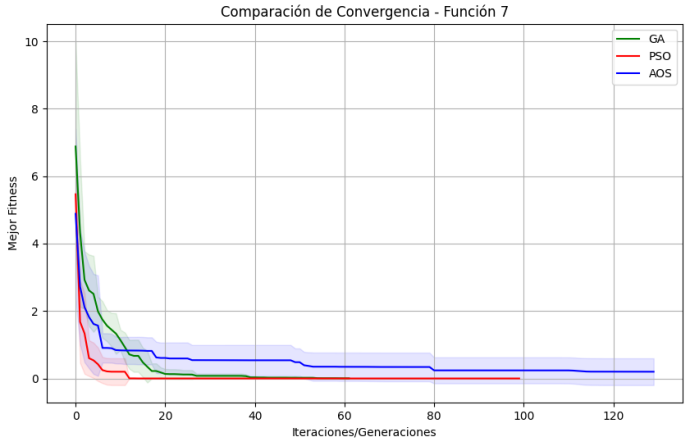

Fig. 3. Convergence comparison on Rastrigin function

Convergence analysis was also carried out for each function, where all algorithms converged quickly and similarly as shown in figure 3. Among them, PSO showed the fastest convergence, further supporting its efficiency in terms of both execution time and solution quality.

3.3 Statistical Analysis

To assess the performance differences between AOS, PSO, and GA, we applied two non-parametric tests. The Friedman test was first conducted to evaluate whether there

are significant overall differences in performance among the algorithms. Given the rejection of the null hypothesis in the Friedman test, we followed up with the Nemenyi post-hoc test to perform pairwise comparisons between the algorithms. This two-step approach ensures a rigorous evaluation of both overall and individual differences in performance. [6]

Friedman Test

The result of the Friedman test reveals a **test statistic** of approximately **30.54** and a **p-value** of around 2.33×10^{-7}. The large test statistic suggests that there are notable differences in the performance of the three algorithms: AOS, PSO, and GA. The extremely small p-value, much smaller than the commonly used significance level of 0.05, indicates that these differences are statistically significant. Therefore, we can confidently reject the null hypothesis, which posits that there is no difference in the performance of the algorithms. This suggests that at least one of the algorithms outperforms the others.

Nemenyi Test

AOS vs PSO: The p-value of **0.9950** indicates that there is no significant difference in performance between AOS and PSO. This suggests that both algorithms perform similarly in the context of the problems tested. With respect to GA, the p-value of 0.0000 indicates a highly significant difference between both AOS and GA, as well as PSO and GA. This suggests that both AOS and PSO outperform GA by a significant margin, showing a clear distinction in their performance compared to GA.

4 Conclusions

This study compared AOS, PSO, and GA on standard benchmark functions, with parameter optimization performed using the ABC algorithm. Statistical tests revealed that AOS and PSO significantly outperformed GA, with no significant difference between AOS and PSO in performance. This suggests that AOS, while inspired by quantum mechanics, may be a variation of PSO, sharing similar mechanisms and effectiveness.

AOS consistently produced competitive results, with slightly better performance in some cases, demonstrating its potential for optimization tasks. However, its unique elements did not provide significant advantages over PSO in this study. Future work should explore AOS on more complex problems and investigate hybrid approaches to further enhance its performance and to study how parameters affect algorithms performance.

References

1. Azizi, M.: Atomic orbital search: A novel metaheuristic algorithm. *Applied Mathematical Modelling* **93**, 657–683 (2021). https://doi.org/10.1016/j.apm.2020.12.021. https://www.sciencedirect.com/science/article/pii/S0307904X20307198
2. Rashed, A.: A Python Library for Genetic Algorithms (2018). https://pypi.org/project/geneticalgorithm/
3. MIT: Micro framework for building nature-inspired algorithms. https://niapy.org
4. Miranda, L. M.: A Python-based Particle Swarm Optimization Library (2018). https://pyswarms.readthedocs.io
5. Hunter, J. D.: Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* **9**(3), 90–95 (2007). https://doi.org/10.1109/MCSE.2007.55
6. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**, 1–30 (2006). https://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf