Comparing Optimization Algorithms Through the Lens of Search Behavior Analysis

Gjorgjina Cenikj
Computer Systems Department
Jožef Stefan Institute
Jožef Stefan International
Postgraduate School
Ljubljana, Slovenia
gjorgjina.cenikj@ijs.si

Gašper Petelin
Computer Systems Department
Jožef Stefan Institute
Jožef Stefan International
Postgraduate School
Ljubljana, Slovenia
gasper.petelin@ijs.si

Tome Eftimov
Computer Systems Department
Jožef Stefan Institute
Ljubljana, Slovenia
tome.eftimov@ijs.si

Abstract

The field of numerical optimization has recently seen a surge in the development of "novel" metaheuristic algorithms, inspired by metaphors derived from natural or human-made processes, which have been widely criticized for obscuring meaningful innovations and failing to distinguish themselves from existing approaches. Aiming to address these concerns, we investigate the applicability of statistical tests for comparing algorithms based on their search behavior. We utilize the cross-match statistical test to compare multivariate distributions and assess the solutions produced by 114 algorithms from the MEALPY library. These findings are incorporated into an empirical analysis aiming to identify algorithms with similar search behaviors.

CCS Concepts

• **Computing methodologies** → **Continuous space search**; • **Mathematics of computing** → **Multivariate statistics**.

Keywords

black-box single-objective numerical optimization, optimization algorithm analysis

ACM Reference Format:
Gjorgjina Cenikj, Gašper Petelin, and Tome Eftimov. 2025. Comparing Optimization Algorithms Through the Lens of Search Behavior Analysis. In *Genetic and Evolutionary Computation Conference (GECCO '25 Companion), July 14–18, 2025, Malaga, Spain.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3712255.3726643

1 Introduction

In the past few decades, the field of numerical optimization has experienced an influx of so-called "novel" metaheuristic methods, inspired by metaphors derived from natural or human-made processes [8]. From the behaviors of various animal species to the distribution of mathematical operations, and nuclear reaction processes, seemingly any conceivable concept could be used as a foundation for introducing new metaheuristics. As argued in [9] this trend poses a risk to the scientific rigor within the field of metaheuristics, and can often be "a step backward rather than forward", and "distracts attention away from truly innovative ideas in the field of metaheuristics". The gravity of this trend has led to a call-to-action to stop the publication of such "novel" algorithms, which has been signed by almost 100 researchers in the field of optimization [1]. Yet, the evaluation of an algorithm's novelty remains a challenge. As highlighted in [9], one of the factors contributing to the difficulty in evaluating the novelty of a proposed algorithm is the fact that describing the algorithm using the terminology of the chosen metaheuristic obscures the fundamental algorithm behaviour [2]. Additionally, authors often fail to position the algorithm in the metaheuristics literature and define its relation to other algorithms.

Our contribution: We aim to explore an empirical approach to the comparison of algorithms that takes into account their search behavior by analyzing the solutions explored during the optimization process. We employ the cross-match statistical test [7] for comparing multivariate distributions of the solutions generated by 114 algorithms from the MEALPY library [10] on the Black Box Optimization Benchmarking (BBOB) [5] suite. An empirical analysis is then performed to identify algorithms that exhibit similar search behaviors.

Reproducibility: The code for conducting the experiments is publicly available at https://github.com/gjorgjinac/optimization_algorithm_statistical_comparison.

2 Background: Crossmatch Test for Comparing Multivariate Distributions

The crossmatch test [7] is a nonparametric, distribution-free statistical method for comparing two multivariate distributions based on the adjacency relationships among observations in a combined dataset. Let $X = \{x_1, x_2, \ldots, x_m\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$ represent two independent samples of size m and n, drawn from distributions F_X and F_Y, respectively. The goal of a test for comparing two multivariate distributions is to evaluate the null hypothesis: $H_0 : F_X = F_Y$ against the alternative hypothesis: $H_1 : F_X \neq F_Y$. The crossmatch test includes the following steps:

Adjacency Graph Construction: The samples of points from both distributions, X and Y, are combined into a single set of size $m + n$. The distances between the points from the combined set are used to divide the $m + n$ points into pairs, in such a way that the total distance within pairs is minimized. A crossmatch occurs if a point from X is paired with a point from Y. The total number of crossmatches is denoted by C.

GECCO '25 Companion, July 14–18, 2025, Malaga, Spain

ACM ISBN 979-8-4007-1464-1/2025/07
https://doi.org/10.1145/3712255.3726643

Test Statistic Calculation: The observed number of crossmatches, C, is used as the test statistic. Under the null hypothesis, the expected number of crossmatches is determined based on random assignment of labels to the combined data.

P-value Computation: A p-value is computed by comparing the observed number of crossmatches, C, to its null distribution. Small p-values indicate significant differences between the distributions.

3 Methodology

Our approach involves the comparison of candidate solutions explored by the algorithms during the optimization process, with the steps defined in continuation.

Algorithm Execution: All algorithms to be compared are executed on the same suite of optimization problem instances several times, with fixed random seeds in such a way that initial populations of the algorithms are shared under the same random seed.

Scaling: A min-max scaling of the populations explored by all algorithms is performed by merging the trajectories from all executions of all algorithms for a single problem instance, and scaling both the objective function values and the candidate solutions from all trajectories for the same problem instance. This ensures that the scaled populations of the same problem instance, but different algorithms, are comparable.

Statistical Testing: We use the implementation of the crossmatch test [7] in the *crossmatch* R package to test whether the populations produced by two algorithms on a single problem instance come from the same distribution. In particular, given algorithms a_1 and a_2 executed on an optimization problem instance o for I iterations and R runs (i.e., repetitions), we compare the populations generated by the two algorithms on a fixed problem instance, fixed iteration, and fixed run. More precisely, denoting by r the run number, $r \in \{1..R\}$ and by i the iteration number, $i \in \{1..I\}$, we compare the population $p_{o,a_1,i,r}$ produced by algorithm a_1 in run r and iteration i on problem instance o to the population $p_{o,a_2,i,r}$ produced by algorithm a_2 on the same problem instance, in the same iteration, and run. We execute the test on each pair of populations, with the same p-value of 0.05, applying the Bonferroni correction [12] for multiple comparisons within the same run. It is important to note that each pairwise comparison involves two independent samples, representing the populations generated by two different algorithms at the same iteration of the run. Additionally, we emphasize that we are not comparing populations from the same algorithm across different iterations of the run. The distance between individuals is captured using the euclidean distance. This yields a statistical outcome for the populations at each iteration of the search process of two algorithms executed on the same problem with the same random seed (o, a_1, a_2, i, r).

Empirical aggregation of outcomes: To measure the similarity between a pair of algorithms, we use the statistical test results to define an empirical heuristic. Specifically, we calculate the percentage (ratio) of iterations from a run on a given problem for which the statistical test fails to reject the null hypothesis, indicating that the two populations likely originate from the same distribution. For each pair of algorithms, we aggregate this percentage by calculating its mean value across all problems and runs to derive a similarity indicator.

4 Experimental Design

First, we describe the selected benchmark suite of problem instances used in our analysis, followed by the portfolio of algorithms evaluated on this benchmark suite.

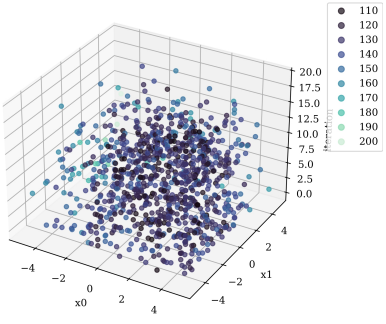
Benchmark suite: As a benchmark suite, BBOB [5] is used. The benchmark contains 24 problem classes, each with multiple instances and different dimensions. We use the first instance of each problem with problem dimension $d \in \{2, 5\}$.

Optimization algorithm portfolio: The MEALPY library [10] is an open-source Python library for metaheuristic optimization with a diverse selection of algorithms. The performance data has been utilized from a previous study [11] and has been collected using the IOHExperimenter platform [4]. Each algorithm is executed on each problem instance five times with a different random seed and a budget of $500d$ function evaluations. The population size is set to 50 for all algorithms. We compare the algorithms by running statistical analysis on the trajectories of the algorithms executed on the same problem instance with the same random seed. We remove algorithms for which the initial population does not match the initial population of the remaining algorithms under the same random seed. This results in a total of 114 algorithms to be compared with the following distribution within the eight MEALPY groups: *bio_based* (11), *evolutionary_based* (11), *human_based* (21), *math_based* (7), *music_based* (2), *physics_based* (11), *swarm_based* (44), *system_based* (7).

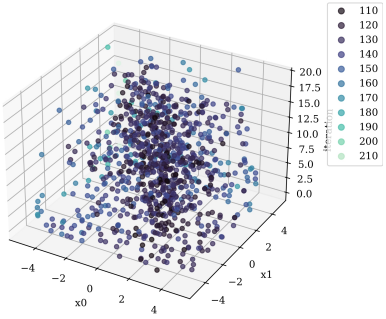
5 Results

To demonstrate the applicability of the statistical test in our use case, we start by showing how the test behaves when applied on a single pair of trajectories. As an example, we compare the trajectories of the BaseDE (Differential Evolution) algorithm to the SADE (Self-Adaptive Differential Evolution) and BaseGA (Genetic Algorithm) algorithms, executed on the first instance of the 24th $2d$ BBOB problem class. We present the example on $2d$ problems, since they are most straightforward to visualize without a loss of information which would be induced by dimensionality reduction. The algorithms are initialized with the same population and are executed for 20 iterations. Figure 1 presents the candidate solutions explored by each of the algorithms. The visualization is generated following examples in [3]. Each subplot contains the trajectory of a different algorithm. The axes on the bottom, with range [-5,5], represent the values of the candidate solutions in the two dimensions. The vertical axis, with range [0,20], represents the iteration number, while the color indicates the objective function value of the candidate solution. It can be observed that the trajectory of the BaseGA algorithm is very different from the trajectories of the BaseDE and SADE algorithms.

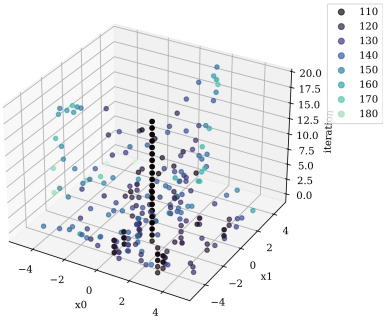
Figure 2 shows the value of the crossmatch statistic throughout the different iterations of the search process. Please note that the maximal value the statistic can obtain is 50, since this is the population size set for all algorithms, and we are comparing the algorithms at a population level. The differently colored lines depict the two different pairs of algorithms being compared, i.e., (BaseDE, SADE)



(a) BaseDE



(b) SADE



(c) BaseGA

Figure 1: Scatterplot of the candidate solutions explored by each of the three algorithms: BaseDE, SADE, BaseGA in one execution on the first instance of the 24th 2d BBOB problem class

and (BaseDE, BaseGA). Looking at the value of the crossmatch statistic in Figure 2, we can see a decreasing trend for the (BaseDE, BaseGA) algorithms (represented by the blue line). The crossmatch statistic has a value of 20 in the first iteration, indicating that the populations of both algorithms are similar, which is expected, since they start from the same initial population. As the search process continues, the value of the statistic drops, meaning that the populations explored are not as similar. On the other hand, looking at the green line representing the (BaseDE, SADE) algorithm pair, we can see that the line is somewhat oscilating around the value of 20, however, it remains mostly above the value of 15 throughout the entire search process. This means that on this problem, the

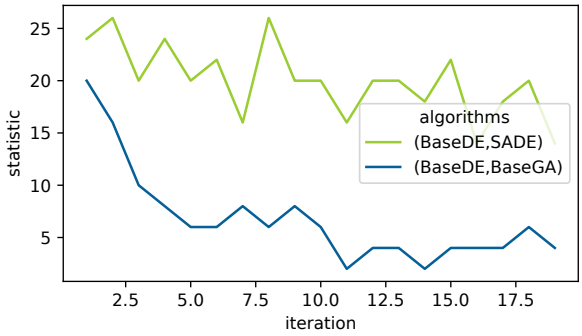


Figure 2: Values of the test statistic obtained with both tests for the algorithms (BaseDE, SADE) and (BaseDE, BaseGA) executed on the first instance of the 24th 2d BBOB problem class

SADE algorithm is much more similar to BaseDE than the BaseGA algorithm, which makes sense, since SADE is a variant of the DE algorithm.

Next, we analyze the pairs of most similar algorithms. To this end, we calculate the mean of the similarity scores obtained in the two dimensions, and we identify the algorithm pairs with the highest mean score. Figure 3 depicts a hierarchical grouping of the algorithms based on the similarities in their search behaviour. The dendrogram is constructed by grouping algorithms using the Ward variance minimization algorithm [6] where distance is captured by the mean score obtained across both problem dimensions. We can observe several groupings of similar algorithm variants being formed: Physics-based Equilibrium Optimization algorithms (OriginalEO, AdaptiveEO); System-based Artificial Ecosystem Optimization algorithms (ImprovedAEO, EnhancedAEO, OriginalAEO, ModifiedAEO); Human-based Queuing Search Algorithm algorithms (BaseQSA, LevyQSA, OriginalQSA, OppoQSA, ImprovedQSA); Evolutionary-based Differential Evolution algorithms (SHADE, L_SHADE, JADE, SADE); Evolutionary-based Genetic Algorithm (GA) algorithms (SingleGA, MultiGA).

Other algorithm pairs which are linked together are the implementations of the same algorithm (OriginalGCO, BaseGCO), (OriginalSCA, BaseSCA), (OriginalTLO, BaseTLO), (BaseJA, OriginalJA). We can also observe many algorithms belonging to different MEALPY groups which are linked together, indicating that even though these algorithms derive inspiration from different processes, they exhibit similar search behaviour.

Nevertheless, we would like to point out that some algorithm pairs exhibiting high similarity scores (low rates of rejecting the null hypothesis) may not have completely identical trajectories. Taking this into account, one can also look into the values of the test statistic when interpreting the results of the statistical testing. The test statistic can also be used as an additional indicator of similarity.

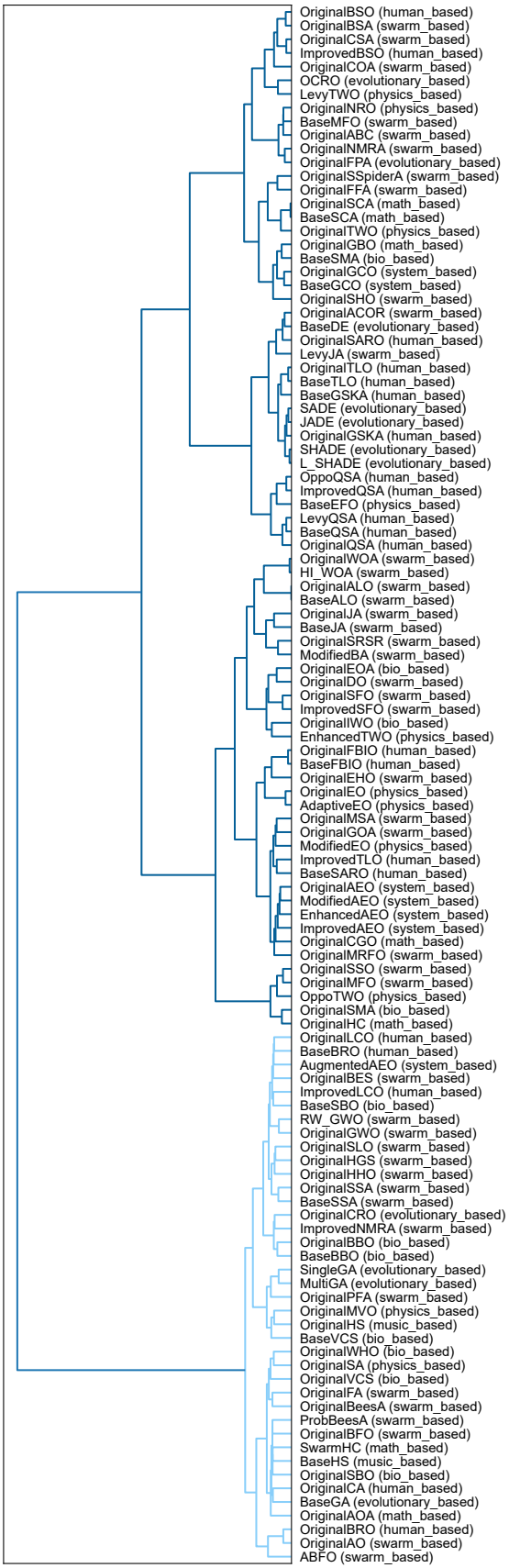


Figure 3: Algorithms grouped in a dendrogram structure

6 Conclusion

In this study, we introduced a novel empricial technique for comparing the search trajectories of optimization algorithms. Our findings revealed substantial similarities between certain algorithm pairs, particularly those where multiple variants are derived from the same base algorithm. More interestingly, our approach also revealed similarities between seemingly unrelated algorithms based on different metaphors or belonging to entirely different groups, where further investigation is needed. This approach provides a valuable metric for comparing newly proposed metaheuristic algorithms against existing ones, allowing researchers to assess the degree of similarity in their behavior and encourages the development of algorithms that bring meaningful improvements.

Acknowledgments

We acknowledge the support of the Slovenian Research and Innovation Agency through program grant No.P2-0098, young researcher grants No.PR-12393 to GC and No. PR-11263 to GP, and project grants No.J2-4460 and No. GC-0001. This work is also funded by the European Union under Grant Agreement No.101187010 (HE ERA Chair AutoLearn-SI) and the EU Horizon Europe program (grant No. 101077049, CONDUCTOR).

References

[1] Claus Aranha, Christian L. Camacho Villalón, Felipe Campelo, Marco Dorigo, Rubén Ruiz, Marc Sevaux, Kenneth Sörensen, and Thomas Stützle. 2021. Metaphor-based metaheuristics, a call for action: the elephant in the room. *Swarm Intelligence* 16, 1 (Nov. 2021), 1–6. https://doi.org/10.1007/s11721-021-00202-9

[2] Christian Leonardo Camacho-Villalón, Marco Dorigo, and Thomas Stützle. 2019. The intelligent water drops algorithm: why it cannot be considered a novel algorithm: A brief discussion on the use of metaphors in optimization. *Swarm Intelligence* 13, 3–4 (May 2019), 173–192. https://doi.org/10.1007/s11721-019-00165-y

[3] Andrea De Lorenzo, Eric Medvet, Tea Tušar, and Alberto Bartoli. 2019. An analysis of dimensionality reduction techniques for visualizing evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (Prague, Czech Republic) *(GECCO '19)*. Association for Computing Machinery, New York, NY, USA, 1864–1872. https://doi.org/10.1145/3319619.3326868

[4] Jacob de Nobel, Furong Ye, Diederick Vermetten, Hao Wang, Carola Doerr, and Thomas Bäck. 2024. IOHexperimenter: Benchmarking Platform for Iterative Optimization Heuristics. *Evolutionary Computation* 32, 3 (09 2024), 205–210. https://doi.org/10.1162/evco_a_00342

[5] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. 2009. *Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions.* Research Report RR-6829. INRIA. https://hal.inria.fr/inria-00362633

[6] Fionn Murtagh and Pierre Legendre. 2014. Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? *Journal of classification* 31 (2014), 274–295.

[7] Paul Rosenbaum. 2005. An Exact Distribution-Free Test Comparing Two Multivariate Distributions Based on Adjacency. *Journal of the Royal Statistical Society Series B* 67 (09 2005), 515–530. https://doi.org/10.1111/j.1467-9868.2005.00513.x

[8] Jörg Stork, Agoston E Eiben, and Thomas Bartz-Beielstein. 2022. A new taxonomy of global optimization algorithms. *Natural Computing* 21, 2 (2022), 219–242.

[9] Kenneth Sörensen. 2015. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research* 22, 1 (2015), 3–18. https://doi.org/10.1111/itor.12001 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/itor.12001

[10] Nguyen Van Thieu and Seyedali Mirjalili. 2023. MEALPY: An open-source library for latest meta-heuristic algorithms in Python. *Journal of Systems Architecture* 139 (2023), 102871. https://doi.org/10.1016/j.sysarc.2023.102871

[11] Diederick Vermetten, Carola Doerr, Hao Wang, Anna V. Kononova, and Thomas Bäck. 2024. Large-Scale Benchmarking of Metaphor-Based Optimization Heuristics. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Melbourne, VIC, Australia) *(GECCO '24)*. Association for Computing Machinery, New York, NY, USA, 41–49. https://doi.org/10.1145/3638529.3654122

[12] Eric W Weisstein. 2004. Bonferroni correction. *https://mathworld. wolfram. com/* (2004).