Article

Comparative Study of Modern Differential Evolution Algorithms: Perspectives on Mechanisms and Performance

Janez Brest * and Mirjam Sepesy Maučec

Posted Date: 21 March 2025

doi: 10.20944/preprints202503.1551.v1

Keywords: global optimization; differential evolution; benchmark suite; mechanisms; statistical tests; performance

Article

Comparative Study of Modern Differential Evolution Algorithms: Perspectives on Mechanisms and Performance

Janez Brest and Mirjam Sepesy Maučec

Faculty of Electrical Engineering and Computer Science, University of Maribor,
SI-2000 Maribor, Slovenia; mirjam.sepesy@um.si
* Correspondence: janez.brest@um.si

Abstract: Since the discovery of the Differential Evolution algorithm, new and improved versions have continuously emerged. In this paper, we review selected algorithms based on Differential Evolution proposed in recent years. We examine the mechanisms integrated into them and compare the performances of algorithms. To compare their performances statistical comparisons were used as they enable us to draw reliable conclusions about algorithms performances. We use the Wilcoxon signed-rank test for pairwise comparisons and the Friedman test for multiple comparisons. Subsequently, the Mann-Whitney U test was added. We conducted not only a cumulative analysis of algorithms but we also focused on their performances regarding the function family (i.e., unimodal, multimodal, hybrid, and composition functions). Experimental results of algorithms were obtained on problems defined for the CEC'24 Special Session and Competition on Single Objective Real Parameter Numerical Optimization. Problem dimensions of 10, 30, 50, and 100 were analyzed. In this paper we highlight promising mechanisms for further development and improvements, based on the performed study of the selected algorithms.

Keywords: global optimization; differential evolution; benchmark suite; mechanisms; statistical tests; performance

MSC: 68W50

1. Introduction

Differential Evolution (DE) is a simple and effective evolutionary algorithm used to solve global optimization problems in continuous space. It has also been adapted for use in discrete, most commonly binary, spaces. The original version was defined for single-objective optimization and was later extended to multi-objective optimization. This line of research, which goes in many directions, highlights the significance of this metaheuristic.

In this paper, we focus our research on differential evolution for single-objective optimization in continuous space. It was introduced in 1996 by Storn and Price. Since then, numerous modifications and improvements have been proposed. This trend continues to the present day and has been especially noticeable in recent years. In the variety of algorithm improvements, it is difficult to determine which one represents a truly significant enhancement. In the paper, we examine the improvements of the DE algorithm that have been published primarily in the past year of algorithm development.

Each year at the Conference of Evolutionary Computation (CEC), a competition for single-objective real parameter numerical optimization is held, in which DE-based algorithms hold a prominent place. In the year 2024, among the six competing algorithms, four were derived from DE. These four are of particular interest to us in this paper. One algorithm from the recent past is also included in comparison to assess whether 2024 brings significant progress in development.

In literature, algorithms are compared using different statistical techniques to perform single-problem and multiple-problem analysis. Non-parametric statistical tests are commonly used as they are less restrictive than parametric tests. In this paper, we adapted the same tests and add the Mann-Whitney U-score test, which has been used to determine the winners in the most recent CEC 2024 competition. The computation of the U-score in our research slightly differs from that used in CEC 2024, as will be described later in the paper. The paper is organized as follows. Section 2 provides the necessary background knowledge to understand the main ideas of the paper. This section outlines the basic DE algorithm and includes brief descriptions of the statistical tests used in the research. Related work is discussed in Section 3. The algorithms used for comparison are described in Section 4, where their main components are highlighted. The algorithms are complex, with several integrated mechanisms and applied settings. Complete descriptions of them are provided with references to the original papers in which they were defined. Section 5 outlines the methodology for the comparative analysis. Extensive experiments conducted to evaluate the performance of the comparative algorithms are presented in Section 6. This section also discusses the obtained results. Section 7 highlights the most promising mechanisms integrated into algorithms under research. Finally, Section 8 summarizes the findings and concludes the paper, offering suggestions for future research directions.

2. Preliminary

2.1. Differential Evolution

In this subsection, we review the DE algorithm [1,2], as it serves as the core algorithm for the further advancements studied in this paper. An optimization algorithm aims to identify a vector $\mathbf{x}$ so as to optimize $f(\mathbf{x})$; $\mathbf{x} = (x_1, x_2, \ldots, x_D)$. D is the dimensionality of the function f. The variables' domains are defined by their lower and upper bounds: $x_{j,low}, x_{j,upp}$; $j \in \{1, \ldots, D\}$.

DE is a population-based algorithm where each individual in the population is represented by a vector $\mathbf{x}_{i,g}$; $i = 1, 2, \ldots, Np$. Np denotes the population size, and g represents the generation number. During each generation, DE applies mutation, crossover, and selection operations to each vector (target vector) to generate a trial vector (offspring).

Mutation generates the mutant vector $\mathbf{v}_{i,g+1}$ according to

$$\mathbf{v}_{i,g+1} = \mathbf{x}_{r_1,g} + F \cdot (\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}). \tag{1}$$

Indexes r_1, r_2, r_3 are randomly selected within the range $\{1, Np\}$ and they are pairwise different among each other and from index i.

Crossover generates a new vector $\mathbf{u}_{i,g+1}$

$$u_{ji,g+1} = \begin{cases} v_{ji,g+1} & \text{if } rand(j) \leq Cr \text{ or } j = rn(i), \\ x_{ji,g} & \text{otherwise.} \end{cases} \tag{2}$$

$rand(j) \in [0, 1]$ is the j-th evaluation of the uniform random generator number. $rn(i) \in \{1, 2, \ \ldots, D\}$ is a randomly chosen index.

Selection performs a greedy selection scheme:

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g+1} & \text{if } f(\mathbf{u}_{i,g+1}) \leq f(\mathbf{x}_{i,g}), \\ \mathbf{x}_{i,g} & \text{otherwise.} \end{cases} \tag{3}$$

The initial population is selected uniformly at random within the specified lower and upper bounds defined for each variable x_j. The algorithm iterates until the termination criterion is satisfied.

Subsequently, in addition to (1) and (2), other mutation and crossover strategies were proposed in literature [3–5]. Although the core DE has significant potential, modifications to its original structure were recognized as necessary to achieve substantial performance improvements.

2.2. Statistical Tests

DE belongs to the stochastic algorithms that can return a different solution in each run, as they use random generators. Therefore, we run the stochastic algorithms several times and statistically compare the obtained results. Based on this, we can then say that a given algorithm is statistically better (worse) than another algorithm with a certain level of significance.

Statistical tests designed for statistical analyses are classified into two categories: parametric and non-parametric. Parametric tests are based on assumptions that are most probably violated when analyzing the performance of stochastic algorithms based on computational intelligence. For this reason, in this paper, we use several non-parametric tests for pairwise and multiple comparisons.

To infer information about two or more algorithms based on the given results, two hypotheses are formulated, the null hypothesis H_0 and the alternative hypothesis H_1. The null hypothesis is a statement of no difference, whereas the alternative hypothesis represents the presence of a difference between algorithms. When using a statistical procedure to reject a hypothesis, a significance level α is set to determine the threshold at which the hypothesis can be rejected. Instead of predefining a significance level (α), the smallest significance level that leads to the rejection of H_0 can be calculated. This value, known as the p-value, represents the probability of obtaining a result at least as extreme as the observed one, assuming H_0 is true. The p-value provides insight into the strength of the evidence against H_0. The smaller the p-value, the stronger the evidence against H_0. Importantly, it achieves this without relying on a predetermined significance level.

The simple sign test for pairwise comparison assumes that if the two algorithms being compared have equivalent performance, the number of instances where each algorithm outperforms the other would be roughly equal. The Wilcoxon signed-rank test does not merely count the wins for each algorithm; rather, it ranks the differences in performance and makes the statistics based on these rankings. This test takes the mean performance from multiple runs for each benchmark function of the two algorithms. Unlike parametric tests, where the null hypothesis H_0 assumes the equivalence of means, these tests define the null hypothesis (H_0) as the equivalence of medians.

The Friedman test is a non-parametric test used to detect differences in the performance of multiple algorithms across multiple benchmark functions. It is an alternative to the repeated-measures ANOVA when the assumptions of normality are not met. In this test, the null hypothesis H_0 states that the medians of the different algorithms are equivalent, while the alternative hypothesis H_1 suggests that at least two algorithms have significantly different medians. To compute the test statistic, the performance of each algorithm is ranked for each problem, and the average rank across all problems is then calculated for each algorithm. In post-hoc analysis (e.g., the Nemenyi test) following the Friedman test, the critical distance is a threshold used to determine whether the difference in performance is statistically significant.

The Mann-Whitney U-score test, which is also called the Wilcoxon rank-sum test, is another non-parametric statistical test used to compare two algorithms and determine whether one tends to have higher values than the other. The test ranks all results from both algorithms together. It calculates the sum of ranks for each algorithm. The test statistic (U) is derived from these rank sums. A significance test determines if the observed rank differences are unlikely under the null hypothesis, which assumes the two algorithms have identical distributions.

Readers interested in a more detailed explanation of statistical tests are advised to the literature [6, 7]. Our research focuses on the results at the end of the optimization process, while others also consider the convergence of their results as it is also a desirable characteristic [8].

3. Related Works

One of the first DE surveys was published by Neri and Tirronen in 2010 [9], and another in 2011 by Das and Sugathan [10]. The first one includes some experimental results obtained with the algorithms available at that time. The second review reviews the basic steps of the DE, discusses different modifications defined at that time, and also provides an overview of the engineering applications of the DE. Both surveys stated that DE will remain an active research area in the future. The time that followed confirmed this, as noted by the same authors who published the updated survey 5 years later [3]. Several years later, Opara and Arabas in [4] provided a theoretical analysis of DE and discussed the characteristics of DE genetic operators coupled with an overview of the population diversity and dynamics models. In 2020, Pant et al. [5] published another survey of the literature on DE, which also provided a bibliometric analysis of DE. Some papers review specific components of DE. DE is very sensitive to its parameter settings, the authors in [11–14] reviewed parameter control methods. Piotrowski et al. [15] analyzed population size settings and adaptation. Parouha and Verma reviewed hybrids with DE [16]. Zhongqiang et al. [17] made an exhaustive listing of more than 500 nature-inspired meta-heuristic algorithms. They also empirically analyzed 11 newly proposed meta-heuristics with a high number of citations, comparing them against 4 state-of-the-art algorithms, 3 of which were DE-based. Their results show that 10 out of 11 algorithms are less efficient and robust than 4 state-of-the-art algorithms, they are also inferior to them in terms of convergence speed and global search ability. Cited papers show that over time DE has evolved in many different directions and remains one of the most promising optimization algorithms. The present paper analyzes the DE algorithms which were proposed very recently. The focus is on their mechanisms and performance.

The algorithms proposed in the literature are very diverse in terms of exploration and exploitation capabilities, so it is important how we evaluate and compare them. In [6], Derrac et al. give a practical tutorial on the use of statistical tests developed to perform both pairwise and multiple comparisons for multi-problem analysis. Traditionally algorithms are evaluated using non-parametric statistical tests with the sign test, the Friedman test, the Mann-Whitney test, and both the Wilcoxon Signed Rank and Rank Sum tests are among the most commonly used. They are based on ordinal results, like the final fitness values of multiple trials. Convergence speed is also an important characteristic of the optimization algorithm. A trial may also terminate upon reaching a predefined target value. In such cases, the outcome is characterized by both the time taken to achieve the target value (if achieved) and the final fitness value. The authors in [8] propose a way to make existing non-parametric tests to compare both the speed and accuracy of algorithms. They demonstrate trial-based dominance using the Mann-Whitney U test to compute U-scores with trial data from multiple algorithms. The authors demonstrate that U-scores are much more effective than traditional ways to rank solutions of algorithms. Non-parametric statistical methods are time-consuming and primarily focus on mean performance measures. To use them, raw results of algorithms in comparison are needed. Goula and Sotiropoulo [18] propose a multi-criteria decision-making technique to evaluate the performances of algorithms and rank them. Multiple criteria included best, worst, median, mean, and standard deviation of the error values. They employed equal weighting because the true weights of criteria are generally unknown in practice.

To perform a comparative analysis of algorithms, the selection of problems on which the algorithms are evaluated is crucial [19]. During the last 20 years, several problem sets were defined. CEC'17 seems to be the most difficult, as it contains a low percentage of unimodal functions (7%), and a high percentage (70%) of hybrid or composition functions.

If we compare our research with related works, we would highlight the following: We conduct an in-depth comparison of the latest DE-based algorithms proposed in 2024 with the basic DE algorithm and the jSO algorithm [20]. The analysis includes a wide range of dimensions: $D = 10$, $D = 30$, $D = 50$, and $D = 100$. To the best of our knowledge, such an in-depth analysis has not yet been performed.

4. Algorithms in the Comparative Study

In this section, we briefly describe the algorithms in comparison. We expose their main characteristics. Readers interested in detailed descriptions of the algorithms are encouraged to study given references. All algorithms are based on the original DE.

4.1. jSO

jSO was proposed in 2017 [20]. It significantly improves the performance of basic DE. Its denominator is the SHADE algorithm [21], proposed in 2013, which introduces external achieve A into DE to enrich population diversity. The archive A saves target vectors if trial vectors improve them. SHADE algorithm also introduces a history-based adaptation scheme for adjusting F_i and Cr_i. Successful F_i and Cr_i are saved into historical memory by using the weighted Lehmer mean in each generation. Using the external archive, the mutant vector is generated using a ”DE/currect-to-pbest/1” mutation strategy. In 2014, LSHADE [22] added a linear population size reduction strategy to balance exploration and exploitation. Afterward, the iL-SHADE [23] algorithm modifies its memory update mechanisms using an adjustment rule that depends on the evolution stage. Higher values for Cr and lower values for F were propagated at the early stages of evolution with the idea of enriching the population diversity. It also uses linear increase greediness factor p. jSO is based on iL-SHADE. It introduces a new parameter F_w to the scaling factor F to the mutation strategy for tuning the exploration and exploitation ability. A smaller factor F_w is applied during the early stages of the evolutionary process, while a larger factor F_w is utilized in the later stages. Using a parameter F_w, jSO adapts a new weighted version of the mutation strategy ”DE/currect-to-pbest-w/1”.

4.2. jSOa

jSOa algorithm was proposed in 2024 [24]. It proposes a more progressive update of external archive A. In the SHADE algorithm, and subsequently in the jSO algorithm, when archive A reaches a predefined size, space for new individuals is created by removing random individuals. A newly proposed jSOa introduces a more systematic approach for storing outperformed parent individuals in the archive of the jSO algorithm. When the archive reaches its full capacity, the individuals within it are sorted based on their functional values, dividing the archive into two sections: better and worse individuals. The newly outperformed parent solution is then inserted into the "worse" section, ensuring that better solutions are preserved while the less effective individuals in the archive are refreshed. Notably, the new parent solution can still be inserted into the archive, even if it performs worse than an existing solution already in the archive. Using this approach, 50% of better individuals in the archive are not replaced. Except for the archive, the other steps of the jSO algorithm remain unchanged in jSOa.

4.3. mLSHADE-LR

The mLSHADE-LR algorithm was also proposed in 2024 [25]. Its core algorithm is LSHADE, which was extended into the LSHADE-EpSin algorithm with an ensemble approach to adapt the scaling factor using an efficient sinusoidal scheme [26]. Additionally, LSHADE-EpSin uses a local search method based on Gaussian Walks, which is used in later generations to improve exploitation. In LSHADE-EpSin, a mutation strategy DE/current-to-pbest/1 with an external archive is applied. LSHADE-cnEpSin [27] is the improved version of LSHADE-EpSin, which uses a practical selection for scaling parameter F and a covariance matrix learning with Euclidean neighborhoods to optimize the crossover operator. mLSHADE-RL further enhances LSHADE-cnEpSin algorithm. It integrates three mutation strategies such as ”DE/current-topbest-weight/1” with archive, ”DE/current-to-pbest/1” without archive, and ”DE/current-to-ordpbest-weight/1”. It has been shown that multi-operator DE approaches adaptively emphasize better-performing operators at various stages. mLSHADE-RL [25] also uses a restart mechanism to overcome the local optima tendency. It consists of two parts: detecting stagnating individuals and enhancing population diversity. Additionally, mLSHADE-RL applies a local search method in the later phase of the evolutionary procedure to enhance the exploitation capability.

4.4. RDE

RDE algorithm was also proposed in 2024 [28]. Just like the previously introduced algorithms, RDE can also be understood as an improvement of the LSHADE algorithm. Like other algorithms, starting with SHADE, it uses an external archive. The research has demonstrated that incorporating multiple mutation strategies is beneficial; however, adding all-new strategies does not necessarily guarantee the best performance. RDE uses a hybridization of two mutation strategies, ”DE/current-to-pbest/1” and ”DE/current-to-order-pbest/1” strategies. The allocation of population resources between the two strategies is governed by an adaptive mechanism that considers the average fitness improvement achieved by each strategy. In RDE, parameters F and Cr are adapted similarly to how they are in jSO. In LSHADE-RSP, the use of a fitness-based rank pressure was proposed for the first time [29]. The idea is to correct the probability of different individuals being selected in DE, with rank indicators assigned to each of them. In RDE, the selection strategy from LSHADE-RSP was slightly modified, as it is based on three instead of two terms.

4.5. L-SRTDE

Like the previous three algorithms, L-SRTDE was also proposed in 2024 [30]. It focuses on one of the most important parameters of the DE, the scaling factor. L-SRTDE adapts its value based on the success rate, which is defined as the ratio of improved solutions in each generation. The success rate also influences the parameter of the mutation strategy, which determines the number of top solutions from which the randomly chosen one is selected. Another minor modification in L-SRTDE is the usage of repaired crossover rate Cr. The idea is that instead of using the sampled crossover rate Cr for crossover, the actual Cr_a value is calculated as the number of replaced components divided by the dimensionality D. All other characteristics of the L-SRTDE algorithm are taken from the L-NTADE algorithm [31], which introduces the new mutation strategy, called ”r-new-to-ptop/n/t” and significantly alters the selection step.

5. Methodology

The purpose of the analysis is to evaluate the performance of recent DE-based algorithms. To evaluate their contributions we include both two well-established baselines and promising new algorithms. To maintain depth in the analysis without overwhelming complexity, we limited the selection of algorithms to recent CEC competition. In addition to the four DE-based algorithms from CEC’24, the basic DE and jSO algorithms were used as baselines. jSO is incorporated in the comparison, as it ranked in second place in the CEC 2017 Special Session and Competition on Single Objective Real Parameter Numerical Optimization. jSO is also a highly cited algorithm, as many authors use it as a baseline algorithm for further development.

The selected algorithms were analyzed separately for each problem dimension: 10, 30, 50, and 100. They were run on all test problems a predetermined number of times. The best error value every $10 \cdot D$ evaluations was recorded for each run. Run finished after a maximum number of function evaluations was reached. The maximum number of evaluations was determined based on the problem’s dimensionality and is the same for all algorithms.

Having raw results for all algorithms, five statistical measures were calculated: best, worst, median, mean, and standard deviation of the error values. The algorithms were evaluated using the following statistical tests: Wilcoxon signed ranks test, Friedman test, and Mann-Whitney U-score test. It is important to note that the computation of the U-score in this article differs from the U-score used in the CEC 2024 competition [8,32]. In our study, we applied the classic Mann-Whitney U-test, whereas the U-score in the CEC 2024 competition incorporates the "target-to-reach" value. Convergence speed was also analyzed using convergence plots.

6. Experiments

We conducted experiments in which we empirically evaluated the progress brought by the improvements to the DE algorithm over the past year. We included the following algorithms: basic DE, jSO [20], jSOa [24], mLSHADE-LR [25], RDE [28], and L-SRTDE [30]. Although the authors of the algorithms have published some results, we conducted the runs ourselves in the study based on the publicly available source code of the algorithms. The experiments were performed following the instructions: experiments with the dimensions $D = 10$, $D = 30$, $D = 50$, and $D = 100$ were done; each algorithm was run 25 times for each test function, and the average error of the best individual in the population was computed; each run stopped either when the error obtained was less than 10^{-8} or when the maximal number of evaluations $Max_{FEs} = 10000 \cdot D$ was achieved.

6.1. CEC'24 Benchmark Suite

CEC'24 Special Session and Competition on Single Objective Real Parameter Numerical Optimization was based on the CEC'17 benchmark suite, which is a collection of 29 optimization problems which are based on shifted, rotated, non-separable, highly ill-conditioned, and complex functions. The benchmark problems aim to replicate the behavior of real-world problems, which often exhibit complex features that basic optimization algorithms may struggle to capture. The functions are:

- unimodal functions: Bent Cigar function and Zakharov function. Unimodal functions are theoretically the simplest and should present only moderate challenges for well-designed algorithms.
- simple multimodal functions: Rosenbrock's function, Rastrigin's function, expanded Scaffer's F6 function, Lunacek Bi_Rastrigin function, non-continuous Rastrigin's function, Levy function, and Schwefel's function. Simple multimodal functions, characterized by multiple optima, are rotated and shifted. However, their fitness landscape often exhibits a relatively regular structure, making them suitable for exploration by various types of algorithms.
- hybrid functions formed as the sum of different basic functions: Each function contributes a varying weighted influence to the overall problem structure across different regions of the search space. Consequently, the fitness landscape of these functions often varies in shape across different areas of the search space.
- composition functions formed as a weighted sum of basic functions plus a bias according to which component optimum is the global one: They are considered the most challenging for optimizers because they extensively combine the characteristics of sub-functions and incorporate hybrid functions as sub-components.

The search range is $[-100, 100]^D$. The functions are treated as black-box problems. Table 1 presents function groups.

Table 1. Function groups.

Function group	**Functions**
Unimodal functions	f1, f2
Multimodal functions	f3, f4, f5, f6, f7, f8, f9
Hybrid functions	f10, f11, f12, f13, f14, f15, f16, f17, f18, f19
Composition functions	f20, f21, f22, f23, f24, f25, f26, f27, f28, f29

6.2. Results

In this subsection, we present experimental results using three metrics for a comparison of algorithms, namely the U-score, the Wilcoxon's test, and the Friedman's test. The experimental results were obtained on 29 benchmark functions for six selected algorithms (DE, jSO, jSOa, mLSHADE-LP, RDE, and L-SRTDE) on four dimensions (10, 30, 50, and 100). The main experimental results are summarized in the rest of this subsection, while additional detailed and collected results are placed in the Supplementary Materials:

- Tables S1–S24 present the best, worst, median, mean, and standard deviation values after maximal number of function evaluations for six algorithms on 29 benchmark functions.
- Convergence speed graphs are depicted in Figures S1–S16.

6.2.1. U-score

Tables 2–5 show the U-score values for six algorithms on 29 benchmark functions for dimensions $D = 10$, $D = 30$, $D = 50$, and $D = 100$, respectively. For each function, the rank of each algorithm is presented in red. At the bottom of the table, a sum of the U-score values and a sum of ranks, RS, are shown.

Table 2. U-scores of algorithms for 10D.

Func.	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
f1	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f2	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f3	1312.5/6	1612.5/3	1612.5/3	1612.5/3	1612.5/3	1612.5/3
f4	0/6	1757/3	1565/5	1796/2	1643/4	2614/1
f5	1587.5/3	1587.5/3	1587.5/3	1587.5/3	1587.5/3	1437.5/6
f6	0/6	1875/3	1517/4	2152/2	1305/5	2526/1
f7	0/6	1650/3	1576.5/4	1947.5/2	1434/5	2767/1
f8	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f9	0/6	1653.5/3	1758/2	1602.5/4	1592/5	2769/1
f10	743/6	1775/2	1775/2	1612/5	1775/2	1695/4
f11	1800/4	1998/2	2042/1	10/6	1936.5/3	1588.5/5
f12	1446/4	1934/2	1639/3	801/6	1433.5/5	2121.5/1
f13	1463.5/5	1864.5/4	1937.5/1.5	301/6	1937.5/1.5	1871/3
f14	2860/1	1270/5	1375/4	576/6	1431/3	1863/2
f15	2115/2	1074/6	2319/1	1417/3	1102.5/5	1347.5/4
f16	597/6	1824/3	1533/4	2029/2	2208/1	1184/5
f17	2772/1	1592.5/5	1704.5/2	17/6	1606/4	1683/3
f18	1760.5/4	1873/3	2399/1	115.5/6	1948/2	1279/5
f19	1239/5	398/6	2662.5/1	1446.5/4	1790.5/3	1838.5/2
f20	1034/6	1718.5/4	1738/3	1782/2	1209.5/5	1893/1
f21	565/6	1737.5/3	1737.5/3	1737.5/3	1662/5	1935.5/1
f22	90/6	2011/3	1778.5/4	1163.5/5	2248.5/1	2083.5/2
f23	1159.5/5	1526/4	1138/6	2053.5/1	1586.5/3	1911.5/2
f24	2208.5/1	1804.5/3	1955/2	1513/4	880/6	1014/5
f25	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f26	985.5/5	1033.5/4	1654/3	3068/1	2553/2	81/6
f27	1762.5/1.5	1602.5/4	1142.5/6	1485.5/5	1619.5/3	1762.5/1.5
f28	285/6	1053/5	1059/4	2358/2	2203.5/3	2416.5/1
f29	640/5	2087.5/2	2076.5/3	447/6	2632/1	1492/4
sum/RS	34676/126.5	46562.5/102	49532/89.5	40881/109	49187/97.5	51036.5/84.5

doi:10.20944/preprints202503.1551.v1

Table 3. U-scores of algorithms for 30D.

Func.	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
f1	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f2	0/6	1875/3	1875/3	1875/3	1875/3	1875/3
f3	1177/5	1350/3.5	1350/3.5	2710/1	2575/2	213/6
f4	0/6	1905/3	1430/4	699/5	2218/2	3123/1
f5	1109/6	1702.5/3	1862.5/1.5	1616/4	1862.5/1.5	1222.5/5
f6	0/6	1963/3	1519/4	717/5	2210/2	2966/1
f7	0/6	1784/3	1358/4	743/5	2382.5/2	3107.5/1
f8	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5	1562.5/3.5
f9	0/6	1501/5	1698/2	1511/4	1549/3	3116/1
f10	165/6	2045/2	1899/4	582/5	1949.5/3	2734.5/1
f11	0/6	1987/3	2090/2	653/5	1543/4	3102/1
f12	366/5	1858/3	1571/4	307/6	2397/2	2876/1
f13	573/5	1731/4	2060/2	52/6	1929/3	3030/1
f14	625/5	2038/3	1447/4	0/6	2348/2	2917/1
f15	205/6	1330/4	1255/5	1341/3	2273/2	2971/1
f16	0/6	1577/4	1803/2	1365/5	1663/3	2967/1
f17	624/5	1907/4	2427/1	1/6	2245/2	2171/3
f18	627/5	1785/3	1617/4	0/6	2328/2	3018/1
f19	1673/2	1483/4	1364/5	417/6	1666/3	2772/1
f20	0/6	1880/3	1295/4	823/5	2287/2	3090/1
f21	1575/3	1575/3	1575/3	1500/6	1575/3	1575/3
f22	0/6	1992/3	1356/4	867/5	2096/2	3064/1
f23	0/6	1680/4	1052/5	1727/3	1820/2	3096/1
f24	224.5/6	1331.5/5	1362/4	1803/2	3122/1	1532/3
f25	250/6	1646/3	1644/4	678/5	2082/2	3075/1
f26	1406/3	1123.5/4	555.5/6	869/5	2624/2	2797/1
f27	1361/6	1501/5	1565/4	1653/1	1647/3	1648/2
f28	134/6	1562/3	1558/4	839/5	2609/2	2673/1
f29	1013/5	1410/3	1712/2	1313/4	3122/1	805/6
sum/RS	16232.5/152	48647.5/100.5	45425/102	29786/129	61122.5/68.5	70661.5/57

Table 4. U-scores of algorithms for 50D.

Func.	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
f1	0/6	1912.5/2.5	1912.5/2.5	1725/5	1912.5/2.5	1912.5/2.5
f2	0/6	1887.5/2.5	1887.5/2.5	1825/5	1887.5/2.5	1887.5/2.5
f3	449/6	1102.5/5	1175.5/4	2965/1	1783/3	1900/2
f4	0/6	1944/3	1349/4	730/5	2227/2	3125/1
f5	2170/3	1381.5/4	2271.5/2	0/6	2625/1	927/5
f6	0/6	2066/3	1265/4	979/5	2522/2	2543/1
f7	0/6	1821/3	1478/4	694/5	2263/2	3119/1
f8	1625.5/5	1887.5/2.5	1887.5/2.5	199.5/6	1887.5/2.5	1887.5/2.5
f9	0/6	1493/5	1558/4	1631/2	1568/3	3125/1
f10	80/6	1535.5/4	1708.5/3	546/5	2554/2	2951/1
f11	0/6	1629/4	1670/3	1115/5	1836/2	3125/1
f12	412/5	1997/2	1736/4	216/6	1906/3	3108/1
f13	623/5	1818/3	2238/2	2/6	1693/4	3001/1
f14	625/5	1567/4	2123/2	0/6	1954/3	3106/1
f15	47/6	1350/4	1055/5	1525/3	2284/2	3114/1
f16	8/6	1623/3	1571/4	799/5	2438/2	2936/1
f17	43/6	2019/2	1937/3	582/5	1721/4	3073/1
f18	560/5	1619/4	2265/2	65/6	1741/3	3125/1
f19	21/6	1043/5	1670/3	1402/4	2114/2	3125/1
f20	0/6	2015/3	1122/4	872/5	2251/2	3115/1
f21	995.5/5	1619/3	951.5/6	2026.5/2	1345/4	2437.5/1
f22	0/6	1759/3	1472/4	822/5	2203/2	3119/1
f23	0/6	1859/3	1177/4	894/5	2320/2	3125/1
f24	778/5	1445.5/4	1803.5/3	330/6	2559/1	2459/2
f25	25/6	1890/3	1504/4	603/5	2228/2	3125/1
f26	1223/4	1367/3	1190/5	0/6	2609/2	2986/1
f27	1628/4	1887.5/2.5	1887.5/2.5	197/6	3100/1	675/5
f28	49/6	1309/4	1536/3	932/5	2953/1	2596/2
f29	1019/4	803/5	555/6	2417/2	3125/1	1456/3
sum/RS	12381/158	47650/99	45956.5/102	26094/138	63609.5/65.5	76184/46.5

Table 5. U-scores of algorithms for 100D.

Func.	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
f1	44/6	2487.5/2	2487.5/2	1287.5/4	2487.5/2	581/5
f2	0/6	2037/3	3125/1	2212/2	1376/4	625/5
f3	787/6	1257/4	1260/3	3028/1	1944/2	1099/5
f4	0/6	1886/3	1552/4	631/5	2182/2	3124/1
f5	1105/5	1470/4	3059/1	0/6	2005/2	1736/3
f6	0/6	2037/3	2064/2	1441/4	989/5	2844/1
f7	0/6	1843/3	1760/4	642/5	2005.5/2	3124.5/1
f8	625.5/5	1962/4	2262.5/2	0/6	2262.5/2	2262.5/2
f9	0/6	1690/3	1678/4	1916/2	966/5	3125/1
f10	589/5	1862/3	2144/2	36/6	1619/4	3125/1
f11	0/6	1121/5	1379/3	2598/2	1282/4	2995/1
f12	296/6	1923/3	1342/4	335/5	2375/2	3104/1
f13	0/6	1901/3	2129/2	625/5	1595/4	3125/1
f14	632/6	1968/2	1711/3	703/5	1276/4	3085/1
f15	0/6	1126/4	1073/5	2463/2	1588/3	3125/1
f16	0/6	1122/5	1273/4	1743/3	2112/2	3125/1
f17	0/6	1805/3	2150/2	983/5	1312/4	3125/1
f18	229/6	1821/3	2288/2	593/5	1319/4	3125/1
f19	0/6	1149/5	1300/4	2138/2	1663/3	3125/1
f20	0/6	1829/3	1571/4	637/5	2213/2	3125/1
f21	0/6	1532/4	1727/3	2038/2	953/5	3125/1
f22	505/5	1706/4	2181/2	125/6	1833/3	3025/1
f23	152/6	1857/3	1438/4	475/5	2328/2	3125/1
f24	1100/6	1325/3	1145/5	1298/4	1481/2	3026/1
f25	523/5	1839/3	1573/4	113/6	2202/2	3125/1
f26	1013/5	1398/3	1344/4	0/6	3100/1	2520/2
f27	385/6	1529/3	1325/5	1509/4	2325/1	2302/2
f28	50/6	1533/3	1290/4	949/5	2605/2	2948/1
f29	224/6	1503/3	1091/4	983/5	3125/1	2449/2
sum/RS	8259.5/168	48518.5/97	50722/93	31501.5/123	54523.5/81	78350/47

6.2.2. Wilcoxon Signed-Rank Test

Overall U-score values for each dimension are collected in Table 6. In this table, "Total" indicates a sum of the U-scores values over all dimensions (higher value is better), while "totalRS" summarizes ranks of algorithms (lower value is better). Based on these results, we can see that L-SRTDE has the first rank, followed by RDE, jSOa, jSO, mLSHADE-LR, and DE. Although the U-score values in this paper differ slightly from those in the CEC 2024 competition due to differences in the computation procedure, as explained earlier, the ranking of the algorithms remains unchanged.

We also analyzed the performance of the algorithms across different function families (see Table 1). To achieve this, we sum the U-score values of the algorithms for each function family and identify the best-performing algorithm for each function family across different problem dimensions. The obtained results are collected in Table 7. For dimension 10, all algorithms perform equally well on unimodal functions. For multimodal functions, L-SRTDE achieves the best results, while jSOa outperforms all others on hybrid functions. For composition functions, RDE exhibits the best performance. For dimension 30, all algorithms, except for the original DE, perform equally well on unimodal functions. L-SRTDE is the top performer for both multimodal and hybrid functions, whereas RDE remains the best choice for composition functions. Similar trends are observed for dimension 50. However, for dimension 100, L-SRTDE outperforms all other algorithms across all function families except for unimodal functions, where jSOa achieves the best results.

The Wilcoxon Sign-rank test is used to compare two algorithms, and the number of all pairwise comparisons for six algorithms per one dimension is $6 \times 5/2 = 15$. Altogether we have 60 comparisons. Usually, we only compare the newly designed algorithm with other algorithms, which reduces the number of comparisons. We chose L-SRTDE to compare it with other algorithms and the summarized results are presented in Table 8. For each dimension, the comparison using the Wilcoxon sign-rank test with $\alpha = 0.05$ of the L-SRTDE algorithm with other algorithms, and the summarized results are presented as $+/\approx/-$. Sign '$+$' means that L-SRTDE performs significantly better than the compared algorithm, '$-$' means that L-SRTDE performs significantly worse than the compared algorithm, while $\approx$ means that there is no significant difference between compared algorithms. Detailed results are shown in Tables S25, S26, 27, and 28, while summarized results are collected in Table 8. The L-SRTDE algorithm shows superiority over all other algorithms in all dimensions. The latter can be seen from the lines labeled 'Total', which contain the sum of the values for each individual algorithm according to the dimension.

6.2.3. Friedman's Test

As a third test to compare the algorithms, we used the Friedman test. The obtained results for each dimension are presented in Figures 1–4 and and Tables 9–12. A critical distance indicates that there exists a statistical difference among two compared algorithms if the difference of their Friedman rank values is smaller than the critical value. We plot a dotted line of the best-obtained algorithm and the blue line to show the critical distance compared to the best-performed algorithm, which is depicted in green.

Figure 1 shows the comparison for $D = 10$, and we can see that all six algorithms are below the critical distance from L-SRTDE, which is the best algorithm. It means, there are no significant differences among all six algorithms. Table 9 tabulates Friedman ranks, shows a performance order in column Rank, and at the bottom of the table, the χ^2 and p values of the Friedman test are reported. If the p-value is lower than predefined significance level (α) it indicates that the algorithms have statistical significant performances.

The obtained results for $D = 30$, $D = 50$, and $D = 100$ are shown in Figures 2–4, respectively. One can see that L-SRTDE is statistically better than DE, jSO, jSOa, and mLSHADE-LR, while there is no statistical difference between L-SRTDE and RDE. This holds for each dimension of $D = 30$, $D = 50$, and $D = 100$.

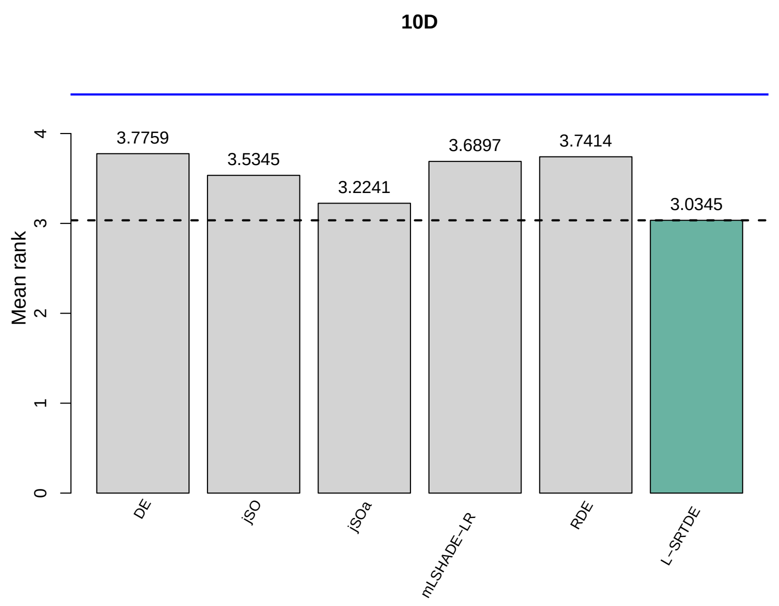

Figure 1. Friedman ranks of algorithms with a critical distance for 10D.

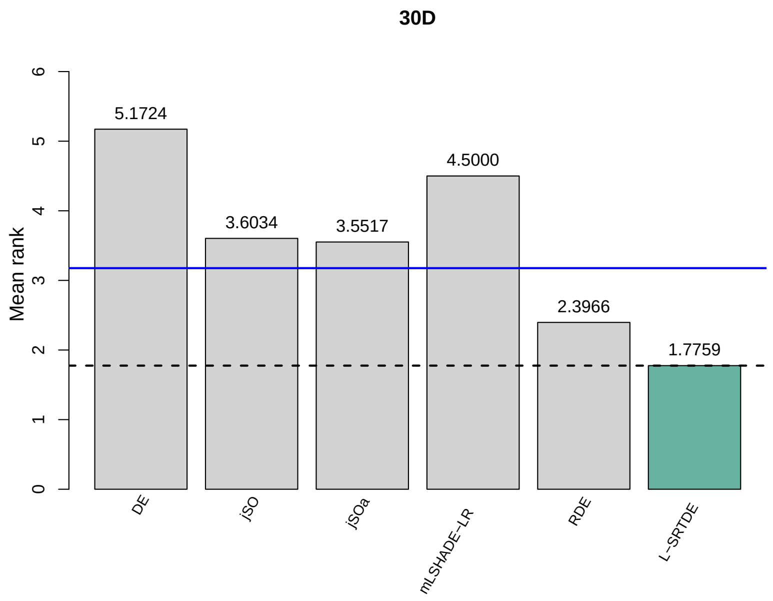

Figure 2. Friedman ranks of algorithms with a critical distance for 30D.

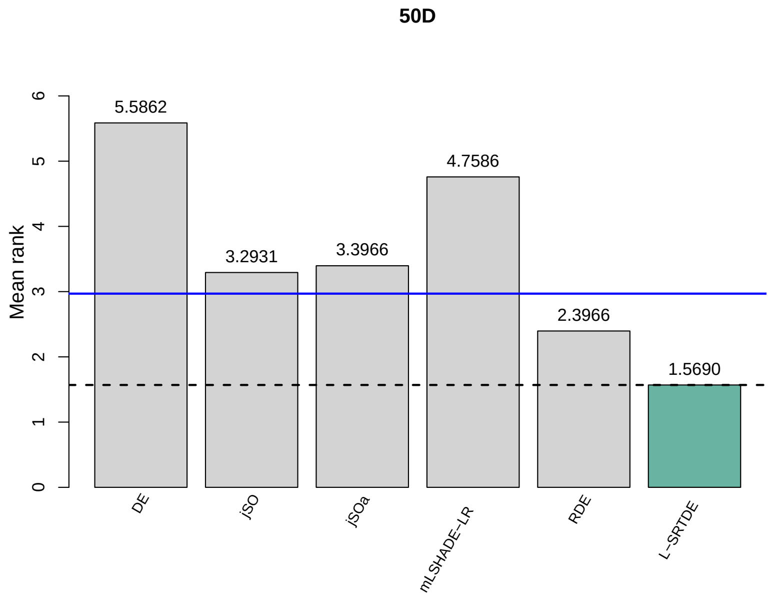

Figure 3. Friedman ranks of algorithms with a critical distance for 50D.

On $D = 100$, we can also see that algorithms RDE, jSOa, and jSO are close to the blue line (i.e., critical distance), and RDE is the only one that is slightly below the blue line.

In general, all three tests that were used in the comparison of six algorithms report very consistent comparison results for each dimension.

7. Discussion

The classic DE algorithm lacks the sophisticated mechanisms introduced in the algorithms, which were proposed later. While it is generally effective, it struggles with maintaining diversity and balancing exploration and exploitation without additional enhancements. It does not have adaptive mechanisms for F and Cr, leading to suboptimal parameter tuning. It does not have an external archive or hybrid strategies, limiting diversity control. It does not use a restart mechanism, making it prone to premature convergence.

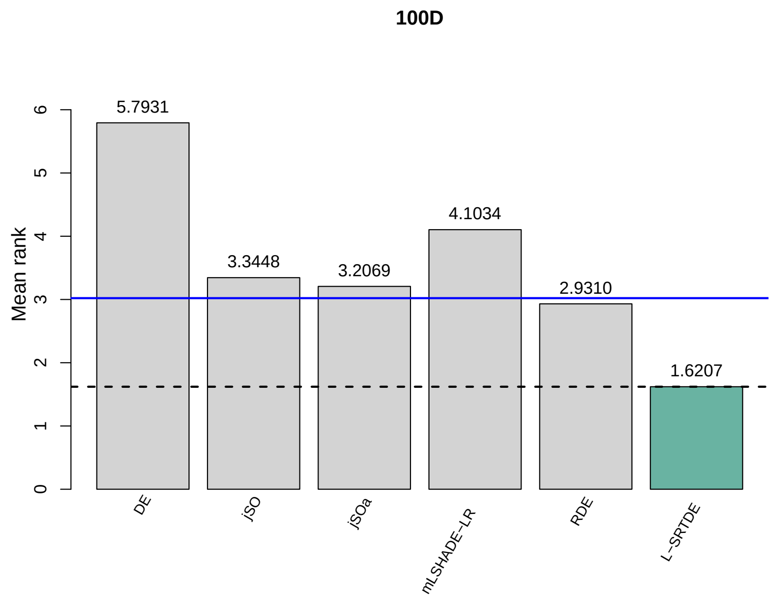

Figure 4. Friedman ranks of algorithms with a critical distance for 100D.

The ranking of compared algorithms reflects each algorithm's ability to balance exploration and exploitation, maintain population diversity, and adapt to different stages of evolution. L-SRTDE leads due to its success rate-driven adaptation, refining both mutation and scaling. The repaired crossover rate (Cr_a) refines the crossover mechanism, ensuring more effective variation across dimensions. RDE follows closely, leveraging hybrid strategies and adaptive resource allocation. It combine "DE/current-to-pbest/1" and "DE/current-to-order-pbest/1". The key improvement is in how resources are allocated between these strategies—an adaptive mechanism evaluates their average fitness improvement and adjusts accordingly. Additionally, RDE incorporates rank pressure-based selection, refining the probability of selecting individuals based on fitness ranks. jSOa stands out for its archive management, ensuring stability. Instead of randomly deleting older solutions, jSOa divides the archive into "better" and "worse" sections based on functional values, ensuring that stronger solutions are preserved while allowing some weaker solutions to be refreshed. This more structured archive update leads to better diversity retention without compromising convergence speed. jSO remains strong, though jSOa refines its archive. mLSHADE-LR's multi-operator approach seems to be powerful.

Table 6. Overall U-scores of algorithms.

Dimension	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
D10/RS	34676/126.5	46562.5/102	49532/89.5	40881/109	49187/97.5	51036.5/84.5
D30/RS	16232.5/152	48647.5/100.5	45425/102	29786/129	61122.5/68.5	70661.5/57
D50/RS	12381/158	47650/99	45956.5/102	26094/138	63609.5/65.5	76184/46.5
D100/RS	8259.5/168	48518.5/97	50722/93	31501.5/123	54523.5/81	78350/47
Total/totalRS	71549/604.5	191378/398.5	191636/386.5	128262/499	228442/312.5	276232/235

Each of these algorithms offers unique strengths, but L-SRTDE currently represents the most advanced evolutionary strategy in this group of algorithms.

8. Conclusions

In this study, we conducted a comprehensive review of state-of-the-art algorithms based on Differential Evolution (DE) that have been proposed in recent years, including jSO, jSOa, mLSHADE-LP, RDE, and L-SRTDE, along with classic DE. We examined the key mechanisms incorporated into

Table 7. U-scores arranged according to functions' families.

Function group	Dimension	DE	jSO	jSOa	mLSHADE-LR	RDE	L-SRTDE
Unimodal functions	D10	3125	3125	3125	3125	3125	3125
	D30	1562.5	3437.5	3437.5	3437.5	3437.5	3437.5
	D50	0	3800	3800	3550	3800	3800
	D100	44	4524.5	5612.5	3499.5	3863.5	1206
	Total	4731.5	14887	15975	13612	14226	11568.5
Multimodal functions	D10	4462.5	11698	11179	12260.5	10736.5	15288.5
	D30	3848.5	11768	10780	9558.5	14359.5	15310.5
	D50	4244.5	11695.5	10984.5	7198.5	14875.5	16626.5
	D100	2517.5	12145	13635.5	7658	12354	17315
	Total	15073	47306.5	46579	36675.5	52325.5	64540.5
Hybrid functions	D10	16053	13828	17611.5	6713	15393.5	14776
	D30	4693	15696	15634	4136	18392	25824
	D50	2339	14665	16265	5706	17687	27713
	D100	1157	13936	14645	12181	14522	27934
	Total	24242	58125	64155.5	28736	65994.5	96247
Composition functions	D10	9258.5	14418	14103.5	15388.5	16947.5	14259
	D30	5963.5	13821	12379.5	11249	20697	20265
	D50	5717.5	13939	12076.5	8221.5	22442	21978.5
	D100	3952	14222	13114	7490	19952	25645
	Total	24891.5	56400	51673.5	42349	80038.5	82147.5

these algorithms and systematically compared their performance. Specifically, we identified shared and unique mechanisms among the algorithms.

To ensure robust performance evaluation, we employed statistical comparisons, which provide reliable insights into algorithm efficacy. The Wilcoxon signed-rank test was utilized for pairwise comparisons, while the Friedman test was applied for multiple comparisons. Additionally, the Mann-Whitney U test was incorporated to enhance the statistical rigor of our analysis. We also performed a cumulative assessment of the six algorithms and evaluated their performance across different function families, including unimodal, multimodal, hybrid, and composition functions.

The experimental evaluation was conducted on benchmark problems defined for the CEC'24 Special Session and Competition on Single Objective Real Parameter Numerical Optimization. We analyzed problem dimensions of 10, 30, 50, and 100, running all algorithms on our own with parameter settings as recommended by the original authors.

The key findings from our experiments indicate that all three statistical tests consistently ranked the algorithms in the following order: L-SRTDE achieved the highest rank, followed by RDE, jSOa, jSO, mLSHADE-LP, and classic DE. Notably, a similar ranking was obtained by the method proposed in [8], which was also employed for algorithm ranking in CEC 2024 [32]. However, CEC 2024 competition was performed only at dimension $D = 30$ [32]. We extend the analysis to a lower dimension ($D = 10$) as well as higher dimensions ($D = 50$ and $D = 100$). The analysis across different function families ranks the algorithms slightly differently at dimension $D = 10$; however, at $D = 100$, L-SRTDE remains the best across all function families.

In conclusion, significant progress has been made since the first published Differential Evolution algorithm. The enhanced versions of DE now incorporate a variety of mechanisms, all of which collectively contribute to improved performance.

doi:10.20944/preprints202503.1551.v1

Table 8. The Wilcoxon signed-rank results of L-SRTDE versus other algorithms ($\alpha = 0.05$). (+ means that L-SRTDE performs significantly better than a compared algorithm.)

Dimension	Algorithm	$+/\approx/-$
10	DE	13/10/6
	jSO	7/17/5
	jSOa	8/15/6
	mLSHADE-LR	11/16/2
	RDE	6/19/4
	Total	45/77/23
30	DE	22/0/7
	jSO	19/9/1
	jSOa	19/8/2
	mLSHADE-LR	20/7/2
	RDE	17/8/4
	Total	97/32/16
50	DE	27/1/1
	jSO	23/5/1
	jSOa	23/5/1
	mLSHADE-LR	25/2/2
	RDE	20/3/6
	Total	118/16/11
100	DE	28/0/1
	jSO	25/2/2
	jSOa	24/2/3
	mLSHADE-LR	26/0/3
	RDE	21/2/6
	Total	124/6/15

Table 9. Algorithms' ranks for 10D.

Algorithm	MeanRank	Rank
DE	3.7758621	6
jSO	3.5344828	3
jSOa	3.2241379	2
mLSHADE-LR	3.6896552	4
RDE	3.7413793	5
L-SRTDE	3.0344828	1
$\chi^2 = 4.7391,\ p = 0.44855$		

Table 10. Algorithms' ranks for 30D.

Algorithm	MeanRank	Rank
DE	5.1724138	6
jSO	3.6034483	4
jSOa	3.5517241	3
mLSHADE-LR	4.5000000	5
RDE	2.3965517	2
L-SRTDE	1.7758621	1
$\chi^2 = 74.5953,\ p = 1.13\text{e-}14$		

Table 11. Algorithms' ranks for 50D.

Algorithm	MeanRank	Rank
DE	5.5862069	6
jSO	3.2931034	3
jSOa	3.3965517	4
mLSHADE-LR	4.7586207	5
RDE	2.3965517	2
L-SRTDE	1.5689655	1
$\chi^2 = 93.7564,\ p = 1.0907\text{e-}18$		

Table 12. Algorithms' ranks for 100D.

Algorithm	MeanRank	Rank
DE	5.7931034	6
jSO	3.3448276	4
jSOa	3.2068966	3
mLSHADE-LR	4.1034483	5
RDE	2.9310345	2
L-SRTDE	1.6206897	1
$\chi^2 = 80.0745,\ p = 8.0961\text{e-}16$		

Supplementary Materials: The following supporting information can be downloaded at the website of this paper posted on Preprints.org.

Author Contributions: Conceptualization, M.S.M.; methodology, M.S.M. and J.B.; software, J.B.; validation, M.S.M. and J.B.; investigation, M.S.M. and J.B.; writing—original draft preparation, J.B. and M.S.M.; writing—review and editing, J.B. and M.S.M.; visualization, J.B. Both authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Slovenian Research and Innovation Agency (research core funding No. P2-0069 — Advanced Methods of Interaction in Telecommunications and P2-0041 — Computer Systems, Methodologies, and Intelligent Services).

Data Availability Statement: The data is available as supplementary material.

Acknowledgments: The authors would like to thank the developers of the jSOa, mLSHADE-LP, RDE, and L-SRTDE algorithms for making their source code publicly available.

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

References

1. Storn, R.; Price, K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization* **1997**, *11*, 341–359.
2. Price, K. *Differential Evolution: a Practical Approach to Global Optimization*; Springer Science & Business Media, 2005.
3. Das, S.; Mullick, S.S.; Suganthan, P.N. Recent advances in differential evolution–an updated survey. *Swarm and evolutionary computation* **2016**, *27*, 1–30.
4. Opara, K.R.; Arabas, J. Differential Evolution: A survey of theoretical analyses. *Swarm and evolutionary computation* **2019**, *44*, 546–558.
5. Pant, M.; Zaheer, H.; Garcia-Hernandez, L.; Abraham, A.; et al. Differential Evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence* **2020**, *90*, 103479.
6. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* **2011**, *1*, 3–18.

7. Carrasco, J.; García, S.; Rueda, M.; Das, S.; Herrera, F. Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation* **2020**, *54*, 100665.
8. Price, K.V.; Kumar, A.; Suganthan, P.N. Trial-based dominance for comparing both the speed and accuracy of stochastic optimizers with standard non-parametric tests. *Swarm and Evolutionary Computation* **2023**, *78*, 101287.
9. Neri, F.; Tirronen, V. Recent advances in differential evolution: a survey and experimental analysis. *Artificial intelligence review* **2010**, *33*, 61–106.
10. Das, S.; Suganthan, P.N. Differential evolution: A survey of the state-of-the-art. *IEEE transactions on evolutionary computation* **2011**, *15*, 4–31.
11. Dragoi, E.N.; Dafinescu, V. Parameter control and hybridization techniques in differential evolution: a survey. *Artificial Intelligence Review* **2016**, *45*, 447–470.
12. Tanabe, R.; Fukunaga, A. Reviewing and benchmarking parameter control methods in differential evolution. *IEEE Transactions on Cybernetics* **2019**, *50*, 1170–1184.
13. Al-Dabbagh, R.D.; Neri, F.; Idris, N.; Baba, M.S. Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy. *Swarm and Evolutionary Computation* **2018**, *43*, 284–311.
14. Reyes-Davila, E.; Haro, E.H.; Casas-Ordaz, A.; Oliva, D.; Avalos, O. Differential Evolution: A Survey on Their Operators and Variants. *Archives of Computational Methods in Engineering* **2024**, pp. 1–30.
15. Piotrowski, A.P. Review of differential evolution population size. *Swarm and Evolutionary Computation* **2017**, *32*, 1–24.
16. Parouha, R.P.; Verma, P. State-of-the-art reviews of meta-heuristic algorithms with their novel proposal for unconstrained optimization and applications. *Archives of Computational Methods in Engineering* **2021**, *28*, 4049–4115.
17. Ma, Z.; Wu, G.; Suganthan, P.N.; Song, A.; Luo, Q. Performance assessment and exhaustive listing of 500+ nature-inspired metaheuristic algorithms. *Swarm and Evolutionary Computation* **2023**, *77*, 101248.
18. Goula, E.M.K.; Sotiropoulos, D.G. HRA: A Multi-Criteria Framework for Ranking Metaheuristic Optimization Algorithms. *arXiv preprint arXiv:2409.11617* **2024**.
19. Piotrowski, A.P.; Napiorkowski, J.J.; Piotrowska, A.E. Choice of benchmark optimization problems does matter. *Swarm and Evolutionary Computation* **2023**, *83*, 101378.
20. Brest, J.; Maučec, M.S.; Bošković, B. Single objective real-parameter optimization: Algorithm jSO. In Proceedings of the 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 1311–1318.
21. Tanabe, R.; Fukunaga, A. Evaluating the performance of SHADE on CEC 2013 benchmark problems. In Proceedings of the 2013 IEEE Congress on evolutionary computation. IEEE, 2013, pp. 1952–1959.
22. Tanabe, R.; Fukunaga, A.S. Improving the search performance of SHADE using linear population size reduction. In Proceedings of the 2014 IEEE congress on evolutionary computation (CEC). IEEE, 2014, pp. 1658–1665.
23. Brest, J.; Maučec, M.S.; Bošković, B. iL-SHADE: Improved L-SHADE algorithm for single objective real-parameter optimization. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2016, pp. 1188–1195.
24. Bujok, P. Progressive Archive in Adaptive jSO Algorithm. *Mathematics* **2024**, *12*, 2534.
25. Chauhan, D.; Trivedi, A.; et al. A Multi-operator Ensemble LSHADE with Restart and Local Search Mechanisms for Single-objective Optimization. *arXiv preprint arXiv:2409.15994* **2024**.
26. Awad, N.H.; Ali, M.Z.; Suganthan, P.N.; Reynolds, R.G. An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems. In Proceedings of the 2016 IEEE congress on evolutionary computation (CEC). IEEE, 2016, pp. 2958–2965.
27. Awad, N.H.; Ali, M.Z.; Suganthan, P.N. Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems. In Proceedings of the 2017 IEEE congress on evolutionary computation (CEC). IEEE, 2017, pp. 372–379.
28. Tao, S.; Zhao, R.; Wang, K.; Gao, S. An Efficient Reconstructed Differential Evolution Variant by Some of the Current State-of-the-art Strategies for Solving Single Objective Bound Constrained Problems. *arXiv preprint arXiv:2404.16280* **2024**.
29. Stanovov, V.; Akhmedova, S.; Semenkin, E. LSHADE algorithm with rank-based selective pressure strategy for solving CEC 2017 benchmark problems. In Proceedings of the 2018 IEEE congress on evolutionary computation (CEC). IEEE, 2018, pp. 1–8.

30. Stanovov, V.; Semenkin, E. Success Rate-based Adaptive Differential Evolution L-SRTDE for CEC 2024 Competition. In Proceedings of the 2024 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2024, pp. 1–8.
31. Stanovov, V.; Akhmedova, S.; Semenkin, E. Dual-Population Adaptive Differential Evolution Algorithm L-NTADE. *Mathematics* **2022**, *10*, 4666.
32. Qiao, K.; Ban, X.; Chen, P.; Price, K.V.; Suganthan, P.N.; Wen, X.; Liang, J.; Wu, G.; Yue, C. Performance comparison of CEC 2024 competition entries on numerical optimization considering accuracy and speed. PPT Presentation, 2000.