Article

A novel method for large scale optimization problems, based on Differential Evolution

Glykeria Kyrou[1], Vasileios Charilogis[2] and Ioannis G. Tsoulos[3,*]

[1] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece; g.kyrou@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece;itsoulos@uoi.gr
* Correspondence: itsoulos@uoi.gr

Abstract: Global optimization is fundamental to engineering and computer science as it seeks to find better solutions to both simple and complex problems. It aims to find the most effective and efficient solution to any problem. In this paper we present a variation of the differential evolution algorithm for large-scale Global Optimization problems. Differential Evolution (DE) is a universal optimization algorithm that is applied to many practical engineering topics. The DE algorithm is a population-based algorithm like genetic algorithms and uses similar operators such as: crossover, mutation and selection.In this work, a series of modifications are proposed that aim to improve the reliability and speed of the above technique. The new method was tested on a series of large-scale problems and compared with other global optimization techniques with promising results. More specifically, the proposed algorithm has been evaluated by typical high-dimensional numerical optimization problems. The functions used are from the CEC-2010 competition for Large-Scale Global Optimization problems.

Keywords: Optimization; Differential evolution; Evolutionary techniques; Stochastic methods; Large-Scale problems

Citation: Kyrou, G.; Charilogis, V.; Tsoulos, I.G. A novel method for large scale optimization problems, based on Differential Evolution. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

The primary objective of global optimization is to locate the global minimum of a continuous and multidimensional function, in such a way as to ensure complete exploration of the search space. Global optimization aims to examine the entire problem domain in order to find the lowest possible value that is feasible. This procedure is applied to complex functions which usually include multiple local minima, making it difficult to identify the global minimum. Global optimization includes techniques that ensure that local optima are avoided while focusing on maximizing the accuracy and efficiency of the search process. The objective is to find the lowest point through systematic exploration of the entire domain of the function$f : S \to R, S \subset R^n$ and it is defined as follows:

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

where the set S is defined as follows:

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

Global optimization refers to algorithms that aim to find the global minimum optimum of a problem regardless of its complexity. Such methods find application in a wide range of scientific fields, such as mathematics [1,2], physics [3,4], chemistry [5,6], biology [9,10], medicine [7,8], agriculture [11,12] and economics [13,14].

Recently, it has been proposed [15] to separate global optimization techniques into deterministic [16,17] and stochastic ones [18,19]. Deterministic methods, such as interval techniques [20,21], are based on the analysis of the search space, dividing it into smaller regions in order to locate the region containing the global minimum. Recently, a Sergeyev et al. [22] published a comparison between stochastic and deterministic global optimization methods. A set of stochastic methods may include Controlled Random Search techniques [23–25], Simulated Annealing methods [26,27], Genetic algorithms [28,29], Differential Evolution [30,31], Particle Swarm Optimization methods [32,33], Ant Colony Optimization [34,35], etc

Differential Evolution (DE) belonging to the evolutionary methods as mentioned above, is an extremely efficient evolutionary algorithm, which gained great recognition from the late 1990s. More specifically, it was initially proposed in 1995 by Storn and Price [36,37]. It finds applications in many fields of science and engineering, in symmetric optimization problems and in problems that are discontinuous and noisy and change over time. The DE method creates randomly an initial population of solutions and, gradually it produces new solutions as a combination of the previous ones. Also, DE has been used in a variety of symmetry problems from the recent literature, such as community detection [38], structure prediction [39], motor fault diagnosis [40], clustering techniques [41] etc. It can be successfully combined with other techniques for machine learning applications such as classification methods [42,43], feature selection techniques [44,45], deep learning [46,47], etc.

The behavior of the method is controlled by a small set of parameters, such as the differential weight denoted as F, the crossover probability denoted as CR and the number of candidate solutions, called also agents, denoted as NP. In literature a variety of methods has been proposed to adapt some of these parameters, such as the Fuzzy Adaptive DE method [48], a self adapting technique for the control parameters of DE [49], the opposition - based DE method [50] etc. Also, Das et al. proposed [51] a Neighborhood - Based Mutation Operator for the Differential Evolution method. A survey of recent trends in Differential Evolution techniques is provided in the recent published work of Das et al [52].

A Differential Evolution variant and its efficiency was evaluated on a series of large - scale optimization problems from the relevant literature. More specifically, the current work introduces a number of modifications to the Differential Evolution algorithm in order to speed up the process and increase the efficiency of the algorithm, especially for large - scale problems. These modifications include: the integration of an efficient sampling method, the incorporation of a termination technique designed for the Differential Evolution method, application of different mechanisms for the differential weight parameter, as well as periodic refinement of the produced solutions using a local optimization method.

The handling of large - scale optimization problems was studied in a series of research papers from the recent literature, such as cooperative coevolution [53], Particle Swarm Optimization [54], a memetic Differential Evolution approach [55] etc.

The remain of this paper is divided as follows: in section 2 the proposed method is fully described, in section 3 the test functions used in the experiments as well as the related experiments are presented and finally in section 4 some conclusions and guidelines for future improvements are discussed.

2. Materials and Methods

The proposed algorithm incorporates a series of modifications to the original differential evolution method, which makes finding the global minimum in high - dimensional problems more efficient. The main steps of the proposed method are listed subsequently.

1. **Initialization step**.
   (a) **Set** as NP the population size of the method (number of agents).
   (b) **Create** randomly from a distribution NP agents $x_i,\ i = 1,\ldots,\mathrm{NP}$
   (c) **Compute** the fitness value f_i of each agent x_i using the objective function as $f_i = f(x_i)$.

(d) **Set** as p_l the local search rate.
(e) **Set** the integer parameter N_t as the tournament size.
(f) **Set** as N_g the maximum number of iterations allowed.
(g) **Set** as N_I the number of iterations used in the stopping rule.
(h) **Set** $k = 0$, the iteration counter.
(i) **Set** the parameter CR, which represents the crossover probability with CR≤ 1.
(j) **Select** the differential weight method, which is represented by the parameter F. In the proposed method three distinct methods were incorporated:
    i. **Number**. In this case the parameter F is chosen as a constant value.
    ii. **Random**. The random method represents the differential weight mechanism proposed by Charilogis et al. [56], where it is defined as:

$$F = -0.5 + 2r \tag{2}$$

    where r is a random number with $r \in [0,1]$.
    iii. **Migrant**. In this case the differential weight mechanism proposed in [57] was used.

2. **For** $i = 1, \ldots, \mathrm{NP}$ **do**
   (a) **Select** the agent x_i
   (b) **Select** randomly three distinct agents x_a, x_b, x_c. The selection of these agents could be performed randomly or with the application of the tournament selection procedure. During tournament selection, a subset of N_t agents are selected from the current population and the one with the lowest fitness value is selected.
   (c) **Choose** a random integer $R \in [1, n]$,where n is the dimension of the objective problem.
   (d) **Create** a trial point x_t.
   (e) **For** $j = 1, \ldots .n$ **do**
      i. **Select** a random number $r \in [0,1]$.
      ii. **If** $r \leq \mathrm{CR}$ **or** $i = R$ **then** $x_{t,j} = x_{a,j} + F \times \left(x_{b,j} - x_{c,j}\right)$ **else** $x_{t,j} = x_{i,j}$
   (f) **End For**
   (g) **Set** $y_t = f(x_t)$
   (h) **If** $y_t \leq f_i$ **then** $x_i = x_t,\ f_i = y_t$.
   (i) **Select** a random number $r \in [0,1]$. If $r \leq p_l$ then $x_i = \mathrm{LS}(x_i)$,where LS defines a local search procedure. In the proposed method the BFGS variant of Powell [58] was used.
3. **End For**
4. **Check for termination**.
   (a) **Set** $k = k + 1$
   (b) **If** $k \geq N_g$ then terminate.
   (c) **Check** the termination rule specified in the work of Charilogis et al [56]. In this work the quantity

$$\delta^{(k)} = \left| \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k)} \right| - \sum_{i=1}^{\mathrm{NP}} \left| f_i^{(k-1)} \right| \right| \tag{3}$$

   is calculated. The term $f_i^{(k)}$ stands for the fitness value of agent i at iteration k. If $\delta^{(k)} \leq \epsilon$ for a number of N_I iterations, then terminate the algorithm else goto step 2.

3. Results

This section begins with a description of the functions that will be used in the experiments and then presents in detail the experiments that were performed, in which the parameters available in the proposed algorithm were studied, in order to study its reliability and adequacy.

3.1. Test Functions

A variety of test functions was used in the conducted experiments. These functions are used in a series of research papers [62–65]. In the present research work, these functions were used with a varying number of dimensions from 5 to 20. The description of each used test function is provided below. In all cases the constant n defines the dimension of the objective function.

- F9 function, which is defined as:

$$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad x \in [0,1]^n$$

- F12 function, having the following definition:

$$f(x) = \frac{\pi}{n}\left(10\sin(\pi y_1) + \sum_{i=1}^{n-1}\left((y_i-1)^2\left(1+10\sin^2(\pi y_{i+1})\right)\right) + (y_n-1)^2\right) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$$

- F13 function, defined as:

$$f(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

- F14 function, which is defined as follows:

$$f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{n}(x_i - a_{ij})^6}\right)^{-1}$$

- F15 function, with the following definition

$$f(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$$

- F18 function, which has the following definition

$$f(x) = -\sum_{i=1}^{4} c_1 \exp\left(-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\right)$$

  where $c_1 = 0.965$
- F19 function, defined as

$$f(x) = -\sum_{i=1}^{4} c_1 \exp\left(-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2\right)$$

  where $c_1 = 0.83$

- TEST2N function, with the following definition:

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i, \quad x_i \in [-5,5].$$

- ELP function, with the following definition:

$$f(x) = \sum_{i=1}^{n} \left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$$

- SCHWEFEL221 function, defined as

$$f(x) = 418.9829n + \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right)$$

- SINU function defined as:

$$f(x) = -\left(2.5\prod_{i=1}^{n}\sin(x_i - z) + \prod_{i=1}^{n}\sin(5(x_i - z))\right), \quad 0 \le x_i \le \pi.$$

3.2. Experimental results

A series of experiments was carried out for the previously mentioned functions and these experiments were executed on an AMD RYZEN 5950X with 128GB RAM. The operating system of the running machine was Debian Linux. Each experiment was conducted 30 times, with different random numbers each time, and the averages were recorded. The software used in the experiments was coded in ANSI C++ using the freely available optimization environment of OPTIMUS, which can be downloaded from https://github.com/itsoulos/OPTIMUS (accessed on 11 December 2024). The values for the experimental parameters used in the proposed method are outlined in Table 1.

Table 1. The values of the parameters of the proposed method.

PARAMETER	MEANING	VALUE
NP	Number of agents	200
p_l	Local search rate	0.01
F	Differential weight	0.8
CR	Crossover probability	0.9
N_g	Maximum number of allowed iterations	200
N_I	Number of iterations used in the termination rule	10
N_t	Tournament size	8

In the following tables that depict the experimental results, the numbers in cells stand for the average function calls, as measured on 30 independent runs. The numbers in parentheses denote the fraction of the executions where the method discovered successfully the global minimum. If this number is not present, then the method managed to locate the global minimum in every run (100% success).

3.3. The effect of differential weight mechanism

The table 2 presents the objective function evaluations required by the differential evolution method for different test functions and dimensions. It is noted that the sample selection in the basic differential evolution framework is random. Three approaches for calculating the differential weight are compared: constant value, random selection, and the MIGRANT approach. The values in parentheses indicate the success rate of each approach (e.g., 0.97 corresponds to 97%). In all cases, the initial sample distribution was uniform. The analysis of the results shows that the number of objective function evaluations generally increases with the dimension for all test functions. For example, in the F9 function, the evaluations increase from 69034 in dimension 5 to 75942 in dimension 20 when the differential weight is constant. Similar trends are observed in the other two approaches, with the MIGRANT approach generally requiring fewer evaluations, especially in higher dimensions. The performance of the MIGRANT approach varies depending on the function and the dimension. For instance, in the F15 function with dimension 20, MIGRANT records 6145 evaluations with a success rate of 0.97, a significantly lower number of evaluations compared to the other two approaches. Conversely, in the F13 function for dimension 20, relatively low success rates are observed (0.37 for the MIGRANT approach versus 0.63 when the differential weight is constant). The average number of evaluations for the three approaches is 931043 for a constant differential weight, 825891 for random selection, and 445122 for the MIGRANT approach. The success rates are similar across the approaches, with values of 0.82, 0.83, and 0.81, respectively. However, the MIGRANT approach demonstrates greater efficiency in several cases, achieving comparable or even higher success rates with significantly fewer evaluations. Specific functions such as TEST2N and F15 clearly show the superiority of MIGRANT, as this approach achieves high success rates (up to 0.97) with a reduced number of evaluations. Similarly, in the SCHWEFEL221 function, MIGRANT shows better efficiency for higher dimensions, with an example of 14832 evaluations in dimension 20 compared to 58552 when the differential weight is constant. In conclusion, MIGRANT emerges as an approach that offers competitive advantages over the other two, especially in higher-dimensional problems. The choice of the appropriate approach depends on the problem requirements and the complexity of the test function.

Table 2. Comparing different differential weight mechanisms.

FUNCTION	DIM	NUMBER(R)	RANDOM(R)	MIGRANT(R)
F9	5	69034	68262	9336(0.57)
F9	10	73049(0.03)	72599(0.03)	54467(0.03)
F9	15	73795(0.03)	73312(0.03)	73491(0.03)
F9	20	75942(0.03)	75189(0.03)	75711(0.03)
F12	5	9715	7986	5057
F12	10	12219	9559	4991
F12	15	14101	11027	5428
F12	20	18282	14090	5845
F13	5	5759(0.03)	5274(0.03)	3700(0.03)
F13	10	22947(0.03)	15072(0.03)	5219(0.03)
F13	15	20102(0.03)	17305(0.03)	5478(0.07)
F13	20	14189(0.63)	13606(0.50)	5482(0.37)
F14	5	7001	6346	4779
F14	10	7953	7082	5085
F14	15	15165	11991	6455(0.93)
F14	20	13338	11690	6192(0.93)
F15	5	7917	7181(0.83)	4996(0.70)
F15	10	9155	8361	5537(0.97)
F15	15	10473	9122	5909(0.97)
F15	20	11650	9729	6145(0.97)
F18	5	2443	2446	2424
F18	10	2445	2448	2422
F18	15	2447	2446	2311
F18	20	2545	2448	2402
F19	5	2403	2612	2402
F19	10	2643	2602	2532
F19	15	2596	2593	2922
F19	20	2673	2459	2611
TEST2N	5	16355	12978	5602
TEST2N	10	35647	27151	7245
TEST2N	15	56031	46514	8586(0.93)
TEST2N	20	66045	65758	10108(0.83)
ELP	5	11978	10846	5176
ELP	10	15048	14546	6555
ELP	15	17190	17170	7758
ELP	20	19063	19349	8880
SCHWEFEL221	5	8048	7210	4918
SCHWEFEL221	10	13635	10871	5626
SCHWEFEL221	15	25843(0.07)	27354(0.07)	13895(0.30)
SCHWEFEL221	20	58552(0.20)	57631(0.70)	14832(0.70)
SINU	5	12886	9938	5280
SINU	10	16103	13366	5992
SINU	15	20535	17018	7209
SINU	20	26103	20354	8601
AVERAGE		**931043(0.82)**	**825891(0.83)**	**445122(0.81)**

In figure 1, the overall analysis, which includes all pairwise comparisons, yielded a p-value of 1.1e-05. This value is significantly smaller than the conventional significance level (p=0.05), indicating that there are clear statistically significant differences among the groups. This result suggests that the compared techniques do not exhibit homogeneity and that the observed differences in outcomes are unlikely to be due to chance. The comparison between the NUMBER(R) and RANDOM(R) methods produced a p-value of 1.5e-05. This value is similarly very small, signifying that these two methods display statistically significant differences. For the comparison of NUMBER(R) with MIGRANT(R), the p-value was also 1.5e-05. This finding confirms that these two methods also exhibit significantly different performance characteristics. The statistical significance observed here underscores that the differences in results between NUMBER(R) and MIGRANT(R) cannot be ignored. The final pairwise comparison, between RANDOM(R) and MIGRANT(R), yielded a p-value of 1e-04. Although this value is larger than the previous ones, it is still well below the significance threshold of p=0.05. Therefore, in this case as well, statistically significant differences between the two approaches are evident, indicating variations in their effectiveness or the stability of their results. Overall, the very low p-values obtained from all comparisons point to clear and systematic differences among the methods. This indicates that each approach has distinct characteristics that set it apart from the others, whether in terms of reducing the number of objective function evaluations or achieving high success rates. The results of the analysis highlight the importance of selecting the appropriate method based on the specific requirements of the problem.

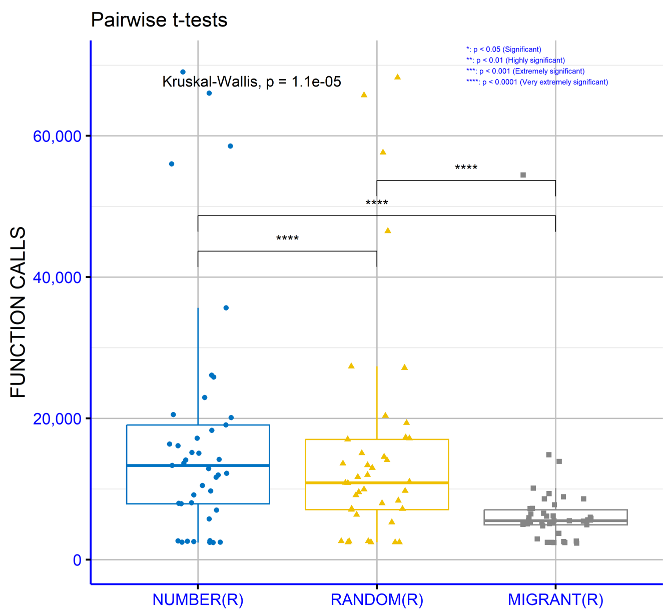

Figure 1. Statistical comparison between the different variations of the proposed method for the series of objective problems.

3.4. *The effect of selection mechanism*

Table 3 compares four approaches for computing the differential weight: random differential weight with random selection (RANDOM(R)), random differential weight with tournament selection (RANDOM(T)), MIGRANT differential weight with random selection (MIGRANT(R)), and MIGRANT differential weight with tournament selection (MIGRANT(T)). Values in parentheses indicate the success rate for each approach (e.g., 0.97 corresponds to 97%). In all cases, the initial sample distribution was uniform. The results analysis shows a systematic reduction in the number of objective function evaluations when transitioning from RANDOM(R) to MIGRANT(T). For instance, in function F9 with a dimension of 5, the evaluations decrease from 68262 in RANDOM(R) to 4337 in MIGRANT(T), but the success rate drops significantly from 0.57 to 0.23. Similar trends are observed in other cases, where MIGRANT(T) drastically reduces the evaluations, but the success rate is alarmingly low in many instances. MIGRANT(T) records the lowest average number of evaluations (186307) compared to RANDOM(R) (825891), RANDOM(T) (503875), and MIGRANT(R) (445122). However, MIGRANT(T)'s success rate is only 0.66, a significant disadvantage as it often fails to find the correct solution. This success rate is much lower than the other approaches, which maintain rates close to 0.80-0.83. Examples such as the TEST2N function highlight the challenges of MIGRANT(T). In dimension 20, MIGRANT(T) reduces the evaluations to 4595 compared to 65758 for RANDOM(R), but the success rate falls to 0.10 from RANDOM(R)'s 0.93. A similar scenario occurs in the SINU function with a dimension of 5, where MIGRANT(T) requires 3326 evaluations with a success rate of 0.77, compared to RANDOM(R)'s 9938 evaluations but with higher reliability (success rate of 1.0). It is essential to note that despite the reduction in evaluations, MIGRANT(T) is particularly ineffective when solution accuracy is a priority. Applications requiring high accuracy (i.e., high success rates) will face significant challenges with this approach, limiting its usability to specific problem domains. In conclusion, MIGRANT(T) offers remarkable reductions in the number of evaluations, but its very low success rate (66% on average) is a major drawback. This makes it less suitable for scenarios where solution reliability is critical, even though the reduction in evaluations is impressive. Its selection should be made cautiously, depending on the problem's requirements.

Table 3. Experimental results comparing random and tournament selection.

FUNCTION	DIM	RANDOM(R)	RANDOM(T)	MIGRANT(R)	MIGRANT(T)
F9	5	68262	67178	9336(0.57)	4337(0.23)
F9	10	72599(0.03)	71680(0.03)	54467(0.03)	8156(0.03)
F9	15	73312(0.03)	75608(0.03)	73491(0.03)	13156(0.03)
F9	20	75189(0.03)	75827(0.03)	75711(0.03)	19711(0.03)
F12	5	7986	4085	5057	3366(0.90)
F12	10	9559	4113	4991	3135
F12	15	11027	4367	5428	3246(0.97)
F12	20	14090	4916	5845	3317(0.90)
F13	5	5274(0.03)	4442(0.13)	3700(0.03)	3023(0.03)
F13	10	15072(0.03)	9837(0.03)	5219(0.03)	3714(0.03)
F13	15	17305(0.03)	6425(0.03)	5478(0.07)	3823(0.03)
F13	20	13606(0.50)	5324(0.07)	5482(0.37)	3882(0.10)
F14	5	6346	3685	4779	3162
F14	10	7082	3824	5085	3319
F14	15	11991	4774(0.80)	6455(0.93)	3646(0.47)
F14	20	11690	4438	6192(0.93)	3607(0.93)
F15	5	7181(0.83)	4806(0.50)	4996(0.70)	3304(0.20)
F15	10	8361	4837(0.90)	5537(0.97)	3683(0.53)
F15	15	9122	4933(0.80)	5909(0.97)	3799(0.60)
F15	20	9729	5117	6145(0.97)	3773(0.60)
F18	5	2446	2446	2424	2425
F18	10	2448	2447	2422	2325
F18	15	2446	2450	2311	2422
F18	20	2448	2445	2402	2511
F19	5	2612	2602	2402	2502
F19	10	2602	2447	2532	2524
F19	15	2593	2529	2922	2626
F19	20	2459	2612	2611	2533
TEST2N	5	12978	4448	5602	3494(0.97)
TEST2N	10	27151	6881	7245	3980(0.50)
TEST2N	15	46514	9605(0.97)	8586(0.93)	4330(0.17)
TEST2N	20	65758	12407(0.93)	10108(0.83)	4595(0.10)
ELP	5	10846	3961	5176	3324
ELP	10	14546	4795	6555	3702
ELP	15	17170	5511	7758	4026
ELP	20	19349	5891	8880	4336
SCHWEFEL221	5	7210	3767	4918	3457
SCHWEFEL221	10	10871	4223	5626	3395
SCHWEFEL221	15	27354(0.07)	16887(0.23)	13895(0.30)	5112(0.03)
SCHWEFEL221	20	57631(0.70)	15425(0.70)	14832(0.70)	5667(0.13)
SINU	5	9938	4013	5280	3326(0.77)
SINU	10	13366	4588	5992	3648(0.93)
SINU	15	17018	5240	7209	4051(0.87)
SINU	20	20354	6039	8601	4807(0.70)
AVERAGE		**825891(0.83)**	**503875(0.80)**	**445122(0.81)**	**186307(0.66)**

In figure 2, the general Kruskal-Wallis test produced a p-value of 1.1e-08. This exceptionally low p-value indicates the presence of statistically significant differences among the groups. This result suggests that the approaches under evaluation exhibit substantial variability in their performance or behavior, which cannot be attributed to random chance. The comparison between RANDOM(R) and RANDOM(T) yielded a p-value of 1.9e-05. This very low value confirms the existence of statistically significant differences between the two approaches, emphasizing their distinct performance characteristics. The analysis of RANDOM(R) compared to MIGRANT(R) produced a p-value of 1e-04. Although higher than the previous value, it is still low enough to indicate statistically significant differences, highlighting variations that cannot be overlooked. The comparison of RANDOM(R) with MIGRANT(T) yielded a p-value of 1.8e-05, once again indicating the presence of statistically significant differences between the two approaches. Conversely, the p-value for the comparison between RANDOM(T) and MIGRANT(R) was 0.51. This value exceeds the conventional significance threshold, suggesting insufficient statistical evidence to conclude meaningful differences between these two approaches. Similar results were observed for the comparison between RANDOM(T) and MIGRANT(T), where the p-value was 0.38, further reinforcing the lack of statistically significant differences. Finally, the analysis between MIGRANT(R) and MIGRANT(T) produced a p-value of 0.0034, which is significantly below the significance level (p=0.05). This highlights the existence of statistically significant differences between these two variations of the MIGRANT approach. In summary, most comparisons revealed clear differences among the evaluated approaches, as reflected in the exceptionally low p-values, while some exceptions indicated no statistically significant differences.

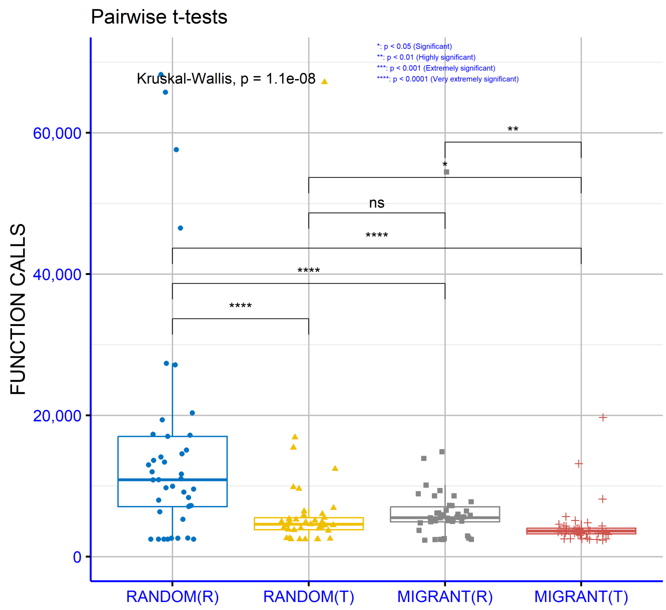

Figure 2. Statistical comparison for the used selection methods.

3.5. *The effect of sampling method*

In Table 4, the selection of samples used in the core formula of differential evolution is conducted through tournament selection. Four approaches for computing the differential weight are compared: random weight with uniform sampling (Random(U)), random weight with k-means sampling (Random(K)), MIGRANT weight with uniform sampling (Migrant(U)), and MIGRANT weight with k-means sampling (Migrant(K)). The k-means technique used to locate centers is considered also as a sampling method here. The method was introduced by James MacQueen[59] and it has been considered in a series of research papers [60,61].

The use of k-means sampling has a decisive impact on reducing the number of objective function evaluations and improving success rates. For example, in the Random method, k-means sampling (Random(K)) dramatically reduces the number of evaluations while significantly increasing success rates. In function F9 with a dimension of 10, Random(U) requires 71680 evaluations with a success rate of only 3%, whereas Random(K) reduces the evaluations to 62309 and boosts the success rate to 97%. Similar outcomes are observed in higher dimensions, such as in dimension 20 for the same function, where Random(K) achieves a success rate of 97% with fewer evaluations (74754) compared to Random(U), which has a 3% success rate with 75827 evaluations. A similar impact of k-means is evident in the MIGRANT method. For instance, in function F12 with a dimension of 5, MIGRANT(K) reduces the number of evaluations to 2310 while maintaining a notable success rate of 63%, compared to MIGRANT(U), which requires 3366 evaluations with a success rate of 90%. In more demanding functions, such as SCHWEFEL221 with a dimension of 15, MIGRANT(K) reduces the evaluations to 4134 while maintaining a success rate of 10%, whereas MIGRANT(U) requires 5112 evaluations for the same success rate. The overall effect of k-means sampling is also reflected in the average results. In the Random method, the success rate increases from 80% (Random(U)) to 94% (Random(K)), with a significant reduction in evaluations from 503875 to 432601. Similarly, in the MIGRANT method, MIGRANT(K) achieves an average success rate of 83% with only 144826 evaluations, compared to MIGRANT(U), which has a success rate of 66% and requires 186307 evaluations. These findings highlight the critical role of k-means sampling in reducing computational complexity and enhancing the method's performance. The use of k-means provides an effective strategy for boosting success rates while simultaneously conserving computational resources.

Table 4. Experiments using different sampling techniques

FUNCTION	DIM	RANDOM(U)	RANDOM(K)	MIGRANT(U)	MIGRANT(K)
F9	5	67178	42222	4337(0.23)	2979
F9	10	71680(0.03)	62309(0.97)	8156(0.03)	6017(0.97)
F9	15	75608(0.03)	71210	13156(0.03)	7990
F9	20	75827(0.03)	74754	19711(0.03)	9592
F12	5	4085	2745(0.97)	3366(0.90)	2310(0.63)
F12	10	4113	3599	3135	2727(0.93)
F12	15	4367	4060	3246(0.97)	3033
F12	20	4916	4700	3317(0.90)	3267
F13	5	4442(0.13)	2565	3023(0.03)	1995
F13	10	9837(0.03)	6744	3714(0.03)	3003
F13	15	6425(0.03)	5711	3823(0.03)	3459
F13	20	5324(0.07)	5028	3882(0.10)	3532
F14	5	3685	2351	3162	2086
F14	10	3824	3276	3319	2858
F14	15	4774(0.80)	3981(0.37)	3646(0.47)	3190(0.10)
F14	20	4438	4536(0.97)	3607(0.93)	3539(0.47)
F15	5	4806(0.50)	3015(0.53)	3304(0.20)	2133(0.20)
F15	10	4837(0.90)	4161(0.87)	3683(0.53)	3149(0.70)
F15	15	4933(0.80)	4664(0.90)	3799(0.60)	3549(0.63)
F15	20	5117	4914	3773(0.60)	3691(0.63)
F18	5	2446	1612	2425	1599
F18	10	2447	2112	2325	2093
F18	15	2450	2293	2422	2274
F18	20	2445	2376	2511	2357
F19	5	2602	1627	2502	1615
F19	10	2447	2128	2524	2107
F19	15	2529	2312	2626	2293
F19	20	2612	2394	2533	2371
TEST2N	5	4448	2972	3494(0.97)	2309
TEST2N	10	6881	5676	3980(0.50)	3374(0.80)
TEST2N	15	9605(0.97)	8999	4330(0.17)	3944(0.20)
TEST2N	20	12407(0.93)	12059(0.97)	4595(0.10)	4351(0.13)
ELP	5	3961	2600	3324	2096
ELP	10	4795	4002	3702	3079
ELP	15	5511	4884	4026	3622
ELP	20	5891	5569	4336	4041
SCHWEFEL221	5	3767	2421	3457	2316
SCHWEFEL221	10	4223	3545	3395	2870
SCHWEFEL221	15	16887(0.23)	14874(0.03)	5112(0.03)	4134(0.10)
SCHWEFEL221	20	15425(0.70)	15312(0.47)	5667(0.13)	5401(0.03)
SINU	5	4013	2626	3326(0.77)	2273(0.73)
SINU	10	4588	3849	3648(0.93)	3058
SINU	15	5240	4605	4051(0.87)	3489
SINU	20	6039	5209	4807(0.70)	3661
AVERAGE		**503875(0.80)**	**432601(0.94)**	**186307(0.66)**	**144826(0.83)**

In figure 3, the general analysis yielded a p-value of 1e-08. This value is extremely small and significantly lower than the commonly used significance level (p=0.05). The comparison between RANDOM(U) and RANDOM(K) produced a p-value of 3e-09. This exceptionally low value indicates that the two approaches differ significantly in their results. These differences reflect substantial variations in performance or behavior. The analysis of RANDOM(U) compared to MIGRANT(U) resulted in a p-value of 1e-04. Although larger than the previous values, this p-value remains sufficiently low to indicate statistically significant differences. This suggests that the two approaches exhibit distinct performance characteristics that cannot be ignored. The comparison between RANDOM(U) and MIGRANT(K) yielded a p-value of 1.6e-06, further supporting the presence of notable differences between these two approaches. Conversely, the p-value for the comparison between RANDOM(K) and MIGRANT(U) was 0.056, which exceeds the conventional significance threshold. This indicates that, in this case, there is insufficient evidence to conclude statistically significant differences between these two approaches. The comparison between RANDOM(K) and MIGRANT(K) produced a p-value of 0.016, which is lower than the significance threshold (p=0.05). This demonstrates that the two approaches exhibit some statistically significant differences in their outcomes, albeit less pronounced than in earlier comparisons. Finally, the analysis between MIGRANT(U) and MIGRANT(K) yielded two p-values, 3e-09 and 0.00041, both of which are far below the significance level. This highlights clear and strong differences between these two variations of the MIGRANT approach, suggesting that the choice of the appropriate approach can significantly impact performance. In summary, the exceptionally low p-values observed in most comparisons indicate the presence of clear differences among the approaches examined. An exception is the comparison between RANDOM(K) and MIGRANT(U), where no significant difference was recorded. These findings underscore the importance of selecting the appropriate method, taking into account the specific characteristics of each problem and the potential to optimize performance through careful parameterization.

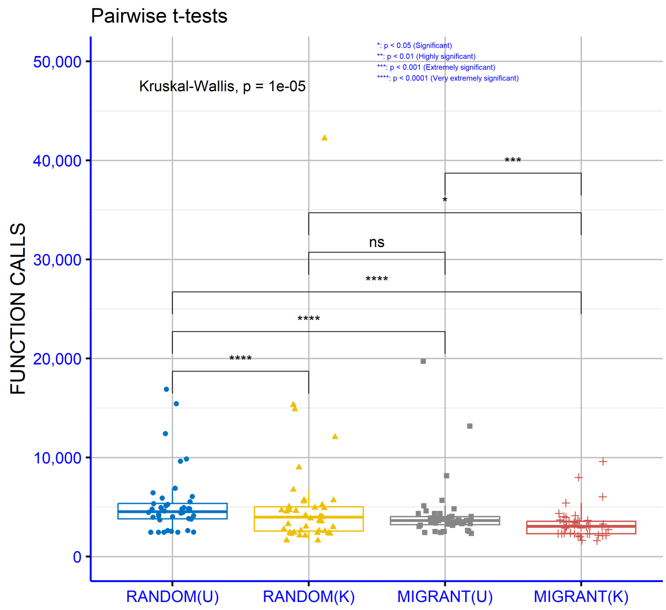

Figure 3. Statistical comparison for the different sampling techniques.

3.6. The effect of local search rate

Table 5, similarly to the previous tables, presents the number of objective function evaluations for various test functions and dimensions using the differential evolution method. In this case, the initial distribution is based on the k-means technique, sample selection is performed using tournament selection, and the differential weight is calculated stochastically. The difference between the columns lies in the frequency of applying periodic local search, which is used for selecting the differential weight. The evaluated frequencies are 0.005 (0.5%), 0.01 (1%), 0.02 (2%), and 0.04 (4%). The results indicate that higher periodic local search frequencies lead to an increased number of objective function evaluations without a commensurate improvement in success rates. For instance, in function F9 with a dimension of 10, a frequency of 0.005 requires 48872 evaluations, while a frequency of 0.04 increases the evaluations to 151795, more than threefold. However, the success rate remains at 97% for both 0.01 and 0.04 frequencies, suggesting that higher frequencies do not significantly enhance performance. Similar trends are observed in other functions. In function TEST2N with a dimension of 20, a frequency of 0.005 requires 11297 evaluations for a success rate of 73%, whereas a frequency of 0.01 increases evaluations to 12059, improving the success rate to 97%. However, a frequency of 0.04 raises the evaluations to 14491 without further increasing the success rate. In less demanding functions, such as F12, the impact of periodic local search frequency is smaller. For instance, in dimension 5, evaluations range from 2665 to 2488 without noticeable changes in results. This overall trend is reflected in the averages. With a frequency of 0.005, the average number of evaluations is 335607 with a success rate of 90%. A frequency of 0.01 increases evaluations to 432601, with the success rate reaching 94%. Higher frequencies, 0.02 and 0.04, lead to 496873 and 714296 evaluations, respectively, with success rates marginally rising to 95% and 96%. These findings emphasize the importance of selecting an optimal frequency for applying periodic local search. While higher frequencies may provide slight improvements in success rates, the computational cost increases disproportionately. Lower frequencies, such as 0.005 or 0.01, appear to offer the best balance between efficiency and accuracy.

Table 5. Experiments using different values for the local search rate.

FUNCTION	DIM	RANDOM(0.005)	RANDOM(0.01)	RANDOM(0.02)	RANDOM(0.04)
F9	5	31745	42222	50265	67467
F9	10	48872	62309(0.97)	92313	151795
F9	15	53490	71210	102089	168521
F9	20	55924	74754	109991	179798
F12	5	2665	2745(0.97)	2641	2488
F12	10	3460	3599	3749	3916
F12	15	4043	4060	4275	4630
F12	20	4581	4700	4877	5118
F13	5	2395	2565	2873	3513
F13	10	6635	6744	6975	7439
F13	15	5793	5711	5655	6251
F13	20	5246	5028	5104	6080
F14	5	2139	2351	2803	3459
F14	10	2939	3276	3817	4662
F14	15	3481(0.20)	3981(0.37)	5168(0.63)	6363(0.87)
F14	20	4010(0.90)	4536(0.97)	5297(0.93)	6597
F15	5	2571(0.27)	3015(0.53)	3639(0.73)	4574(0.73)
F15	10	3516(0.70)	4161(0.87)	4943(0.93)	6409
F15	15	3887(0.43)	4664(0.90)	5933(0.97)	7581
F15	20	4211	4914	5870	7560
F18	5	1597	1612	1643	1703
F18	10	2093	2112	2152	2228
F18	15	2273	2293	2334	2416
F18	20	2356	2376	2423	2510
F19	5	1613	1627	1660	1715
F19	10	2109	2128	2170	2243
F19	15	2290	2312	2351	2433
F19	20	2371	2394	2437	2519
TEST2N	5	2893	2972	3060	3274
TEST2N	10	5608	5676	6112	6832
TEST2N	15	8576(0.90)	8999	9279	10464
TEST2N	20	11297(0.73)	12059(0.97)	12908(0.97)	14491
ELP	5	2645	2600	2486	2486
ELP	10	4295	4002	3882	3890
ELP	15	5206	4884	4918	5004
ELP	20	5965	5569	5486	5944
SCHWEFEL221	5	2616	2421	2354	2233
SCHWEFEL221	10	3596	3545	3520	3630
SCHWEFEL221	15	14182(0.03)	14874(0.03)	14716(0.07)	15641(0.27)
SCHWEFEL221	20	15674(0.53)	15312(0.47)	16252(0.53)	17924(0.57)
SINU	5	2600	2626	2775	2917
SINU	10	3845	3849	3912	4215
SINU	15	4729	4605	4674	5094
SINU	20	5320	5209	5357	5736
AVERAGE		**335607(0.90)**	**432601(0.94)**	**496873(0.95)**	**714296(0.96)**

The general analysis for all pairwise comparisons shown in figure 4 produced a p-value of 0.49. This value is much higher than the conventional significance threshold (p=0.05), indicating that the comparisons overall do not exhibit statistically significant differences. This observation suggests that, when considered collectively, the differences among the comparisons might result from random factors. The comparison between the RANDOM(0.005) and RANDOM(0.01) approaches yielded a p-value of 0.019. This value is below the significance threshold, indicating statistically significant differences between these two approaches. These differences are likely attributable to changes in the frequency of applying periodic local search. The analysis of RANDOM(0.005) versus RANDOM(0.02) produced a p-value of 0.00046. This very low value demonstrates clear statistically significant differences between the two approaches, reflecting substantial performance changes due to the increased application frequency. The comparison between RANDOM(0.005) and RANDOM(0.04) yielded a p-value of 8.3e-06, which is extremely low. This indicates that the two approaches exhibit significantly different performance, likely due to the very high local search frequency in RANDOM(0.04). The analysis of RANDOM(0.01) versus RANDOM(0.02) yielded a p-value of 0.0001. Statistically significant differences are again evident, suggesting that the increase in frequency markedly affects the results. The comparison between RANDOM(0.01) and RANDOM(0.04) produced a p-value of 1.7e-06. This extremely low value confirms the presence of clear differences in the performance of the two approaches, due to the further frequency increase. Lastly, the comparison between RANDOM(0.02) and RANDOM(0.04) yielded a p-value of 3.1e-09. This exceptionally small value demonstrates that the increasing frequency of local search continues to significantly influence performance. Overall, the low p-values in most comparisons suggest that the frequency of periodic local search application significantly affects the performance of the approaches. However, the general analysis with p=0.49 indicates that, in combination, the approaches may not show significant differentiation. These findings underscore the importance of carefully tuning the frequency of local search application to achieve an optimal balance between performance and computational cost.

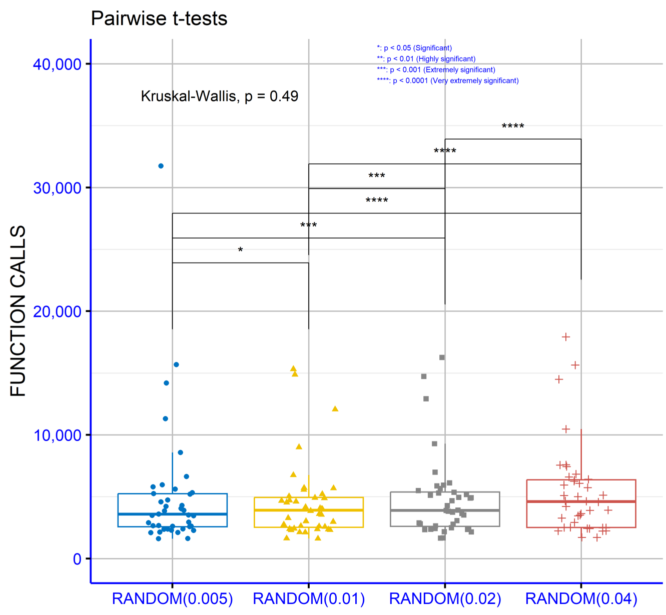

Figure 4. Statistical comparison for the proposed method and different values of parameter p_l.

3.7. Practical problems

The proposed method was tested on some problems inspired by real world, such as the molecular conformation of some atoms or the training of neural networks.

3.7.1. Lennard Jones Potential

The first case of practical function is the energy obtained by the molecular conformation of N atoms, that interacts using the Lennard-Jones potential [66]. The potential function is expressed as:

$$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

In table 6 the function calls using the proposed method for this potential are shown. In column NUMBER the results using the number variant of differential weight are shown, while in column RANDOM the average function calls obtained by the Random variance of the differential weight are shown.

Table 6. Experiments for the Lennard - Jones potential using two different differential weight mechanisms.

ATOMS	NUMBER	RANDOM
2	2982	2844
3	3856	3673
4	5385	5183
5	8012	6920
6	12866	10995
7	21288	17577
8	35714	31866
9	57342	53424
10	89383	77893
11	113164	108395
12	144190	125621
13	146164	134219
14	171496	166035
15	195007	193038
SUM	**1006849**	**937683**

Also in Figure 5 the plot of the function calls against the number of atoms for the previously used differential weight mechanisms is shown.

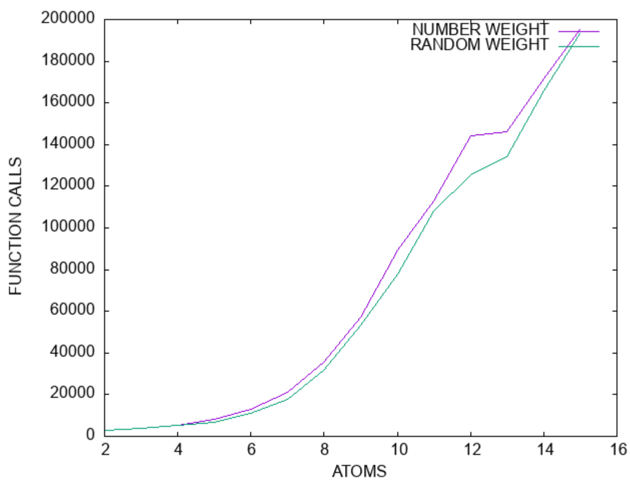

Figure 5. Plot of the function calls for the Lennard - Jones potential using two differential weight mechanisms.

As can be seen from the table of results and the relevant graph, the method that uses the random technique for the differential weight has slightly better results than the technique that uses a fixed differential weight.

3.7.2. Neural network training

Another interesting problem where the proposed method can be applied is the training of artificial neural networks for classification or regression problems. Artificial Neural networks (ANNs) [67,68] are parametric tools machine learning tools that have been used widely in a series of real - world problems, such as problems from physics [69–71], chemistry [72–74], economics [75–77], medicine [78,79] etc. Artificial neural networks are usually expressed as a function $N(\overrightarrow{x}, \overrightarrow{w})$, where the vector $\overrightarrow{x}$ defines the input pattern and the vector $\overrightarrow{w}$ stands for the weight vector of the neural network (set of parameters). The training of the neural network is performed with the adjustment of vector $\overrightarrow{w}$ in order to minimize the so - called training error which is expressed as:

$$E\left(N\left(\overrightarrow{x}, \overrightarrow{w}\right)\right) = \sum_{i=1}^{M}\left(N\left(\overrightarrow{x}_i, \overrightarrow{w}\right) - y_i\right)^2 \quad (4)$$

The set $\left(\overrightarrow{x_i}, y_i\right),\ i = 1, ..., M$ represents the train set of the neural network. The value y_i stand for the actual output for pattern $\overrightarrow{x_i}$.

The proposed method was applied to train a neural network with $H = 10$ processing nodes for the following series of classification datasets, found in the relevant literature [80,81]:

1. **Pima** dataset [82], which is used to detect the presence of the diabetes disease.
2. **Regions2** dataset, related to the detection hepatitis C [83].
3. **Wdbc** dataset [84], a medical related to breast cancer detection.
4. **Wine** dataset, used for the detection of quality of wines [85,86].
5. **Eeg** datasets, a dataset related to EEG measurements [87]. In the conducted experiments the case of Z_F_S was used.
6. **Zoo** dataset [88], used to classify animals.

The experimental results using the proposed method and a series of differential weight methods are shown in Table 7.

Table 7. Application of the proposed method with a series of differential weights on a neural network used for data classification problems. Numbers in cells represent average classification error as measured on the test set of the objective problem.

DATASET	NUMBER	RANDOM	MIGRANT
PIMA	33.91%	33.71%	26.99%
REGIONS2	28.98%	29.04%	28.54%
WDBC	8.22%	7.32%	4.51%
WINE	23.22%	23.14%	9.26%
Z_F_S	21.39%	19.17%	8.16%
ZOO	6.77%	6.70%	3.13%
AVERAGE	**20.42%**	**19.76%**	**13.43%**

As it can be deduced from this table, the method MIGRANT proved to be more effective in terms of classification error from the other methods.

4. Conclusions

This study focuses on large-scale optimization through a modified version of the Differential Evolution algorithm. The introduced modifications aim to enhance efficiency and reliability in high-dimensional problems. Key elements of the new approach include the Migrant differential weight mechanism and the use of k-means sampling to optimize sample selection. Experimental results demonstrated that the proposed method significantly reduces the number of objective function evaluations, particularly for high-dimensional problems. Variants of the differential weight mechanism, such as Migrant, offer competitive advantages with higher success rates and reduced computational cost compared to traditional approaches. Additionally, the use of k-means sampling contributes to reduced complexity and improves the algorithm's performance. In experimental tests, such as neural network training and molecular conformation optimization using the Lennard-Jones potential, the algorithm showcased exceptional capability in finding optimal solutions. However, the algorithm's performance is sensitive to parameter tuning, such as the local search rate. It was observed that higher local search frequencies can substantially increase computational costs without a corresponding improvement in success rates, highlighting the need for a careful balance between accuracy and efficiency.

Future research could focus on several directions to further enhance the method. An important aspect is the development of adaptive parameter tuning strategies that dynamically adjust to the nature of the problem. Moreover, applying the algorithm in distributed computing environments could increase its scalability and speed, making it suitable for real-time problems. Additionally, integrating the algorithm into more complex dynamic systems, such as economic forecasting problems or simulations of physical phenomena, represents a promising avenue. Finally, evaluating the algorithm's performance in large-scale machine learning applications, such as training deep neural networks, may reveal new capabilities and applications. Overall, the study highlights the potential of the proposed approach and emphasizes the importance of continued research to develop even more efficient and versatile optimization algorithms.

Author Contributions: G.K., V.C. and I.G.T. conceived of the idea and the methodology, and G.K. and V.C. implemented the corresponding software. G.K. conducted the experiments, employing objective functions as test cases, and provided the comparative experiments. I.G.T. performed the necessary statistical tests. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: This research has been financed by the European Union: Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan, under the call RESEARCH–CREATE–INNOVATE, project name "iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques" (project code: TAEDK-06195).

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

References

1. Carrizosa, E., Molero-Río, C., & Romero Morales, D. (2021). Mathematical optimization in classification and regression trees. Top, 29(1), 5-33.
2. Legat, B., Dowson, O., Garcia, J. D., & Lubin, M. (2022). MathOptInterface: a data structure for mathematical optimization problems. INFORMS Journal on Computing, 34(2), 672-689.
3. Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. Neurocomputing, 532, 183-214.
4. Stilck França, D., & Garcia-Patron, R. (2021). Limitations of optimization algorithms on noisy quantum devices. Nature Physics, 17(11), 1221-1227.
5. Zhang, J., & Glezakou, V. A. (2021). Global optimization of chemical cluster structures: Methods, applications, and challenges. International Journal of Quantum Chemistry, 121(7), e26553.
6. Hu, Y., Zang, Z., Chen, D., Ma, X., Liang, Y., You, W., & Zhang, Z. (2022). Optimization and evaluation of SO2 emissions based on WRF-Chem and 3DVAR data assimilation. Remote Sensing, 14(1), 220.
7. Kaur, P., & Singh, R. K. (2023). A review on optimization techniques for medical image analysis. Concurrency and Computation: Practice and Experience, 35(1), e7443.
8. Houssein, E. H., Hosney, M. E., Mohamed, W. M., Ali, A. A., & Younis, E. M. (2023). Fuzzy-based hunger games search algorithm for global optimization and feature selection using medical data. Neural Computing and Applications, 35(7), 5251-5275.
9. Wang, L., Cao, Q., Zhang, Z., Mirjalili, S., & Zhao, W. (2022). Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. Engineering Applications of Artificial Intelligence, 114, 105082.
10. Hesami, M., & Jones, A. M. P. (2020). Application of artificial intelligence models and optimization algorithms in plant cell and tissue culture. Applied Microbiology and Biotechnology, 104(22), 9449-9485.
11. Filip, M., Zoubek, T., Bumbalek, R., Cerny, P., Batista, C. E., Olsan, P., ... & Findura, P. (2020). Advanced computational methods for agriculture machinery movement optimization with applications in sugarcane production. Agriculture, 10(10), 434.

12. Akintuyi, O. B. (2024). Adaptive AI in precision agriculture: a review: investigating the use of self-learning algorithms in optimizing farm operations based on real-time data. Research Journal of Multidisciplinary Studies, 7(02), 016-030.
13. Wang, Y., Ma, Y., Song, F., Ma, Y., Qi, C., Huang, F., ... & Zhang, F. (2020). Economic and efficient multi-objective operation optimization of integrated energy system considering electro-thermal demand response. Energy, 205, 118022.
14. Alirahmi, S. M., Mousavi, S. B., Razmi, A. R., & Ahmadi, P. (2021). A comprehensive techno-economic analysis and multi-criteria optimization of a compressed air energy storage (CAES) hybridized with solar and desalination units. Energy Conversion and Management, 236, 114053.
15. L. Liberti, S. Kucherenko, Comparison of deterministic and stochastic approaches to global optimization. International Transactions in Operational Research **12**, pp. 263-285, 2005.
16. Shezan, S. A., Ishraque, M. F., Shafiullah, G. M., Kamwa, I., Paul, L. C., Muyeen, S. M., ... & Kumar, P. P. (2023). Optimization and control of solar-wind islanded hybrid microgrid by using heuristic and deterministic optimization algorithms and fuzzy logic controller. Energy reports, 10, 3272-3288.
17. Xu, Z., Zhao, Z., & Liu, J. (2024). Deterministic Multi-Objective Optimization of Analog Circuits. Electronics, 13(13), 2510.
18. Hsieh, Y. P., Karimi Jaghargh, M. R., Krause, A., & Mertikopoulos, P. (2024). Riemannian stochastic optimization methods avoid strict saddle points. Advances in Neural Information Processing Systems, 36.
19. Tyurin, A., & Richtárik, P. (2024). Optimal time complexities of parallel stochastic optimization methods under a fixed computation model. Advances in Neural Information Processing Systems, 36.
20. M.A. Wolfe, Interval methods for global optimization, Applied Mathematics and Computation **75**, pp. 179-206, 1996.
21. T. Csendes and D. Ratz, Subdivision Direction Selection in Interval Methods for Global Optimization, SIAM J. Numer. Anal. **34**, pp. 922–938, 1997.
22. Sergeyev, Y. D., Kvasov, D. E., & Mukhametzhanov, M. S. (2018). On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Scientific reports, 8(1), 453.
23. W. L. Price, Global optimization by controlled random search, Journal of Optimization Theory and Applications **40**, pp. 333-348, 1983.
24. Ivan Křivý, Josef Tvrdík, The controlled random search algorithm in optimizing regression models, Computational Statistics & Data Analysis **20**, pp. 229-234, 1995.
25. M.M. Ali, A. Törn, and S. Viitanen, A Numerical Comparison of Some Modified Controlled Random Search Algorithms, Journal of Global Optimization **11**,pp. 377–385,1997.
26. L. Ingber, Very fast simulated re-annealing, Mathematical and Computer Modelling **12**, pp. 967-973, 1989.
27. R.W. Eglese, Simulated annealing: A tool for operational research, Simulated annealing: A tool for operational research **46**, pp. 271-281, 1990.
28. Sohail, A. (2023). Genetic algorithms in the fields of artificial intelligence and data sciences. Annals of Data Science, 10(4), 1007-1018.
29. Charilogis, V., Tsoulos, I. G., & Stavrou, V. N. (2023). An Intelligent Technique for Initial Distribution of Genetic Algorithms. Axioms, 12(10), 980.
30. Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., & Xu, J. (2021). An improved differential evolution algorithm and its application in optimization problem. Soft Computing, 25, 5277-5298.
31. Pant, M., Zaheer, H., Garcia-Hernandez, L., & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. Engineering Applications of Artificial Intelligence, 90, 103479.
32. Shami, T. M., El-Saleh, A. A., Alswaitti, M., Al-Tashi, Q., Summakieh, M. A., & Mirjalili, S. (2022). Particle swarm optimization: A comprehensive survey. Ieee Access, 10, 10031-10061.
33. Gad, A. G. (2022). Particle swarm optimization algorithm and its applications: a systematic review. Archives of computational methods in engineering, 29(5), 2531-2561.
34. Rokbani, N., Kumar, R., Abraham, A., Alimi, A. M., Long, H. V., Priyadarshini, I., & Son, L. H. (2021). Bi-heuristic ant colony optimization-based approaches for traveling salesman problem. Soft Computing, 25, 3775-3794.
35. Wu, L., Huang, X., Cui, J., Liu, C., & Xiao, W. (2023). Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. Expert Systems with Applications, 215, 119410.
36. Storn, R., & Price, K. (1995). Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. International computer science institute.
37. Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11, 341-359.
38. Y.H. Li, J.Q. Wang, X.J. Wang, Y.L. Zhao, X.H. Lu, D.L. Liu, Community Detection Based on Differential Evolution Using Social Spider Optimization, Symmetry **9**, 2017.
39. W. Yang, E.M. Dilanga Siriwardane, R. Dong, Y. Li, J. Hu, Crystal structure prediction of materials with high symmetry using differential evolution, J. Phys.: Condens. Matter **33** 455902, 2021.
40. C.Y. Lee, C.H. Hung, Feature Ranking and Differential Evolution for Feature Selection in Brushless DC Motor Fault Diagnosis , Symmetry **13**, 2021.
41. S. Saha, R. Das, Exploring differential evolution and particle swarm optimization to develop some symmetry-based automatic clustering techniques: application to gene clustering, Neural Comput & Applic **30**, pp. 735–757, 2018.

42. Maulik, U., & Saha, I. (2010). Automatic fuzzy clustering using modified differential evolution for image classification. IEEE transactions on Geoscience and Remote sensing, 48(9), 3503-3510.
43. Zhang, Y., Zhang, H., Cai, J., & Yang, B. (2014). A weighted voting classifier based on differential evolution. In Abstract and applied analysis (Vol. 2014, No. 1, p. 376950). Hindawi Publishing Corporation.
44. Hancer, E. (2019). Differential evolution for feature selection: a fuzzy wrapper–filter approach. Soft Computing, 23, 5233-5248.
45. Vivekanandan, T., & Iyengar, N. C. S. N. (2017). Optimal feature selection using a modified differential evolution algorithm and its effectiveness for prediction of heart disease. Computers in biology and medicine, 90, 125-136.
46. Deng, W., Liu, H., Xu, J., Zhao, H., & Song, Y. (2020). An improved quantum-inspired differential evolution algorithm for deep belief network. IEEE Transactions on Instrumentation and Measurement, 69(10), 7319-7327.
47. Wu, T., Li, X., Zhou, D., Li, N., & Shi, J. (2021). Differential evolution based layer-wise weight pruning for compressing deep neural networks. Sensors, 21(3), 880.
48. Liu, J., Lampinen, J. A Fuzzy Adaptive Differential Evolution Algorithm. Soft Comput 9, 448–462 (2005). https://doi.org/10.1007/s00500-004-0363-x
49. 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," in IEEE Transactions on Evolutionary Computation, vol. 10, no. 6, pp. 646-657, Dec. 2006, doi: 10.1109/TEVC.2006.872133.
50. S. Rahnamayan, H. R. Tizhoosh and M. M. A. Salama, "Opposition-Based Differential Evolution," in IEEE Transactions on Evolutionary Computation, vol. 12, no. 1, pp. 64-79, Feb. 2008, doi: 10.1109/TEVC.2007.894200.
51. S. Das, A. Abraham, U. K. Chakraborty and A. Konar, "Differential Evolution Using a Neighborhood-Based Mutation Operator," in IEEE Transactions on Evolutionary Computation, vol. 13, no. 3, pp. 526-553, June 2009, doi: 10.1109/TEVC.2008.2009457.
52. S. Das, S. Subhra Mullick, P.N. Suganthan, Recent advances in differential evolution – An updated survey, Swarm and Evolutionary Computation **27**, pp. 1-30, 2016.
53. D. Sofge, K. De Jong, and A. Schultz. A Blended Population Approach to Cooperative Coevolution for Decomposition of Complex Problems. In Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), pages 413–418. IEEE, 2002.
54. F. van den Bergh and A. P. Engelbrecht. A Cooperative Approach to Particle Swarm Optimisation. IEEE Transactions on Evolutionary Computing, 3:225–239, 2004.
55. Yu Gao and Yong-Jun Wang. A Memetic Differential Evolutionary Al- gorithm for High Dimensional Functions' Optimization. International Conference on Natural Computation, 4:188–192, 2007.
56. V. Charilogis, I.G. Tsoulos, A. Tzallas, E. Karvounis, Modifications for the Differential Evolution Algorithm, Symmetry **14**, 447, 2022.
57. J. Cheng, G. Zhang, F. Neri, Enhancing distributed differential evolution with multicultural migration for global numerical optimization. Information Sciences **247**, pp. 72-93, 2013.
58. Powell, M.J.D. A Tolerant Algorithm for Linearly Constrained Optimization Calculations. Math. Program. 1989, 45, 547–566.
59. J.B. MacQueen, Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. Vol. 1. University of California Press. pp. 281–297. MR 0214227. Zbl 0214.46201, 1967.
60. Y. Li, H. Wu, A clustering method based on K-means algorithm, Physics Procedia **25**, pp. 1104-1109, 2012.
61. P. Arora, S. Varshney, Analysis of k-means and k-medoids algorithm for big data, Procedia Computer Science **78**, pp. 507-512, 2016.
62. M.M. Ali and P. Kaelo, Improved particle swarm algorithms for global optimization, Applied Mathematics and Computation **196**, pp. 578-593, 2008.
63. H. Koyuncu, R. Ceylan, A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems, Journal of Computational Design and Engineering **6**, pp. 129–142, 2019.
64. Patrick Siarry, Gérard Berthiau, François Durdin, Jacques Haussy, ACM Transactions on Mathematical Software **23**, pp 209–228, 1997.
65. A. LaTorre, D. Molina, E. Osaba, J. Poyatos, J. Del Ser, F. Herrera, A prescription of methodological guidelines for comparing bio-inspired optimization algorithms, Swarm and Evolutionary Computation **67**, 100973, 2021.
66. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
67. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
68. G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems **2**, pp. 303-314, 1989.
69. P. Baldi, K. Cranmer, T. Faucett et al, Parameterized neural networks for high-energy physics, Eur. Phys. J. C **76**, 2016.
70. J. J. Valdas and G. Bonham-Carter, Time dependent neural network models for detecting changes of state in complex processes: Applications in earth sciences and astronomy, Neural Networks **19**, pp. 196-207, 2006
71. G. Carleo,M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, pp. 602-606, 2017.
72. Lin Shen, Jingheng Wu, and Weitao Yang, Multiscale Quantum Mechanics/Molecular Mechanics Simulations with Neural Networks, Journal of Chemical Theory and Computation **12**, pp. 4934-4946, 2016.
73. Sergei Manzhos, Richard Dawes, Tucker Carrington, Neural network-based approaches for building high dimensional and quantum dynamics-friendly potential energy surfaces, Int. J. Quantum Chem. **115**, pp. 1012-1020, 2015.

74. Jennifer N. Wei, David Duvenaud, and Alán Aspuru-Guzik, Neural Networks for the Prediction of Organic Chemistry Reactions, ACS Central Science **2**, pp. 725-732, 2016.
75. Lukas Falat and Lucia Pancikova, Quantitative Modelling in Economics with Advanced Artificial Neural Networks, Procedia Economics and Finance **34**, pp. 194-201, 2015.
76. Mohammad Namazi, Ahmad Shokrolahi, Mohammad Sadeghzadeh Maharluie, Detecting and ranking cash flow risk factors via artificial neural networks technique, Journal of Business Research **69**, pp. 1801-1806, 2016.
77. G. Tkacz, Neural network forecasting of Canadian GDP growth, International Journal of Forecasting **17**, pp. 57-69, 2001.
78. Igor I. Baskin, David Winkler and Igor V. Tetko, A renaissance of neural networks in drug discovery, Expert Opinion on Drug Discovery **11**, pp. 785-795, 2016.
79. Ronadl Bartzatt, Prediction of Novel Anti-Ebola Virus Compounds Utilizing Artificial Neural Network (ANN), Chemistry Faculty Publications **49**, pp. 16-34, 2018.
80. M. Kelly, R. Longjohn, K. Nottingham, The UCI Machine Learning Repository. 2023. Available online: https://archive.ics.uci.edu (accessed on 18 February 2024).
81. J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17, pp. 255-287, 2011.
82. J.W. Smith, J.E. Everhart, W.C. Dickson, W.C. Knowler, R.S. Johannes, Using the ADAP learning algorithm to forecast the onset of diabetes mellitus, In: Proceedings of the Symposium on Computer Applications and Medical Care IEEE Computer Society Press, pp.261-265, 1988.
83. N. Giannakeas, M.G. Tsipouras, A.T. Tzallas, K. Kyriakidi, Z.E. Tsianou, P. Manousou, A. Hall, E.C. Karvounis, V. Tsianos, E. Tsianos, A clustering based method for collagen proportional area extraction in liver biopsy images (2015) Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, 2015-November, art. no. 7319047, pp. 3097-3100.
84. W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proc Natl Acad Sci U S A. **87**, pp. 9193–9196, 1990.
85. M. Raymer, T.E. Doom, L.A. Kuhn, W.F. Punch, Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, **33** , pp. 802-813, 2003.
86. P. Zhong, M. Fukushima, Regularized nonsmooth Newton method for multi-class support vector machines, Optimization Methods and Software **22**, pp. 225-236, 2007.
87. R.G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state, Phys. Rev. E **64**, pp. 1-8, 2001.
88. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.