Article

An Innovative Hybrid Approach to Global Optimization

Vasileios Charilogis[1],Glykeria Kyrou[2], Ioannis G. Tsoulos[3,*], Anna Maria Gianni[4]

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

Abstract: Global optimization is critical in engineering, computer science, and various industrial applications, as it aims to find optimal solutions for complex problems. The development of efficient algorithms has emerged from the need for optimization, with each algorithm offering specific advantages and disadvantages. An effective approach to solving complex problems is the hybrid method, which combines established global optimization algorithms. This paper presents a hybrid global optimization method, which produces trial solutions of the objective problem through vector operations derived from methods, such as Genetic Algorithms, Line Search and Differential Evolution. These operations are based on samples derived either from internal line searches or genetically modified samples in specific subsets of Euclidean space. Additionally, other relevant approaches are explored to enhance the method's efficiency. The new method was applied on a wide series of benchmark problems from the recent literature and comparison was made against other established methods of Global Optimization.

Keywords: Optimization; Differential evolution; Genetic algorithm; Line search; Evolutionary techniques; Stochastic methods; Hybrid methods.

Citation: Charilogis V.; Kyrou, G.;Tsoulos I.G.; Gianni A.M; An Innovative Hybrid Approach to Global Optimization. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

The primary objective of global optimization is to locate the global minimum by thoroughly exploring the relevant range associated with the underlying objective problem. This method of global optimization is focused on identifying the global minimum within a continuous function that spans multiple dimensions. Essentially, the global optimization process is dedicated to seeking out the minimum value of a continuous, multidimensional function, ensuring that the search covers all potential ranges of the problem at hand. The objective is to find the lowest point through systematic exploration of the entire domain of the function, which is defined in a Euclidean space R^n. The optimal value of a function $f : S \to R, S \subset R^n$ is defined as follows:

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

where the set S is defined as follows:

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

Global Optimization refers to algorithms whose main objective is to find the global optimum of a problem. According to literature research there are a variety of real-world problems that can be formulated as global optimization problems, such as problems in mathematics [1–3], physics [4–6], chemistry [7–9], and medicine [10–12], biology [13,14], agriculture [15,16] and economics [17,18]. Optimization methods can be categorized into deterministic [19–21] and stochastic [22–24] based on how they approach solving the problem. The techniques used for deterministic are mainly interval methods [25,26]. Stochastic methods utilize randomness to explore the solution space, while in interval methods, the set S is divided into smaller regions that may contain the global minimum based on certain criteria. Recently, a comparison between deterministic and stochastic methods was proposed by Sergeyev et al [27].

A series of stochastic optimization methods are the so - called evolutionary methods, which attempt to mimic a series of natural processes. Such methods include the Genetic algorithms [28,29], the Differential Evolution method [31,32], Particle Swarm Optimization (PSO) methods [33–35], Ant Colony optimization methods [36,37], the Fish Swarm Algorithm [38], the Dolphin Swarm Algorithm [39], the Whale Optimization Algorithm (WOA) algorithm [40–42] etc. Also, due to the wide spread of parallel computing units, a variety of research papers related to evolutionary techniques appeared that use such processing units [43–45]. In the current work, two evolutionary methods were incorporated in the final algorithm: Genetic Algorithms and the Differential Evolution method was combined into a hybrid optimization method.

Genetic algorithms were formulated by John Holland [46] and his team and they initially generated randomly candidate solutions to an optimization problem. These solutions were modified through a series of operators that mimic natural processes, such as mutation, selection and crossover. Genetic algorithms have been used widely in areas such as networking [47], robotics [48,49], energy topics [50,51] etc. They can be combined with machine learning to solve complex problems, such as neural network training [52,53].

Furthermore, differential evolution (DE) is used in symmetric optimization problems [54,55] and in problems that are discontinuous and noisy and change over time. After studies, it was observed that differential evolution can be successfully combined with other techniques for machine learning applications, such as classification [56,57], feature selection [58,59], deep learning [60,61] etc.

Hybrid methods [62,63] in global optimization refer to techniques that combine multiple optimization strategies to solve complex problems. These methods aim to take advantage of different approaches to find the global optimum in a more efficient way, particularly when dealing with large-scale problems or strongly nonlinear optimization landscapes. A typical example of a hybrid method is the work of Shutao Li et al. who proposed a new hybrid PSO-BFGS strategy for the global optimization of multimodal functions [64]. To make the combination more efficient, they proposed an LDI to dynamically start the local search and a repositioning technique to maintain the particle diversity, which can effectively avoid the premature convergence problem. Another innovative hybrid method is the work of M. Andalib Sahnehsaraei et al. where a hybrid algorithm using GA operators and PSO formula is proposed was presented through the use of efficient operators, for example, traditional and multiple crossovers, mutation and PSO formula [65].

The current work produces through a series of steps trial solutions using the genetic operators of the Genetic Algorithm as well as solutions identified by a line search procedure. In current work, the Armijo line search is a method is used. This method is incorporated to estimate an appropriate step when updating the trial points, and it was introduced in the work of Armijo[66]. The solutions produced in the previous step are used to formulate new trial solutions using a process derived from Differential Evolution.

The remainder of this paper is divided into the following sections: in section 2, the proposed method is described, in section 3 the experimental results and statistical comparisons are presented, and finally in section 4 some conclusions and guidelines for future improvements are discussed.

2. The overall algorithm

The proposed method combines some aspects from different optimization algorithms and the main steps are subsequently:

1. **Initialization step.**
   (a) **Set** the population size $N \geq 4$.
   (b) **Set** n the dimension of the benchmark function.

(c) **Initialize** the samples $x_i, i = 1, \ldots, N$ using uniform or k-means[67] distribution.

2. **Calculation step.**

   (a) **For** $i = 1 \ldots \mathrm{N}$ **do**

   i. **Obtain** sample x_i.

   ii. **Find nearest sample** c_i **from** x_i**:**

$$d(x_i, c_i) = \sqrt{\sum_{j=1}^{n} \left(x_{i,j} - c_{i,j}\right)^2} \quad (2)$$

   where $d(x_i, c_i)$ is the Euclidean Distance.

   iii. **Set** direction vectors: $p_1 = -\nabla f(x_i)$ end $p_2 = -\nabla f(c_i)$

   iv. **Set** initial step size for Armijo $a = a_0$

   v. **Compute** with line search Armijo the sample:

   - **Find** new points using line search minLS(x_i, c_i): $x_i^{new} = x_i + ap_1$ and $c_i^{new} = c_i + ap_2$
   - **Adjust** step size a until Armijo condition is met:

$$f(x_i^{new}, c_i{}^{new}) \leq f(x_i, c_i) + c_1 a \nabla f(x_i, c_i)^T (p_1, p_2) \quad (3)$$

   vi. **Make** sample-child with crossover with random number $g_k \in [0.0, 1.0]$:

$$\text{child}\left(x_i, x^{best}\right) = g_k x_{i,k} + (1 - g_k) x_k^{best} \quad (4)$$

   vii. **For** $j = 1, \ldots, n$ **do**

   - **Set** trial vector:

$$y_j = x_{i,j} + F \times \left(\text{minLS}(x_\iota, c_\iota)_j - \text{child}(x_i, c_i)_j\right) \quad (5)$$

     where F is the so - called Differential Weight of Differential Evolution algorithm.
   - **If** $y_j \notin \left[a_j, b_j\right]$**,** then $y_j = x_{i,j}$

   viii. **EndFor**

   - **Set** $r \in [0, 1]$ a random number. If $r \leq p_m$ then $x_i = \text{LS}(x_i)$, where LS(x) is a local search procedure like the BFGS procedure[68].
   - **If** $f(y) \leq f(x)$ then $x = y$, $x^{best} = y$.

   (b) **EndFor**

3. **Check the termination rule stated in** [69], which means that the method checks the difference between the current optimal solution $f_{min}^{(t)}$ and the previous $f_{min}^{(t-1)}$ one. The algorithm terminates when the following:

$$\left| f_{\min}^{(t)} - f_{\min}^{(t-1)} \right| \leq \epsilon \quad (6)$$

   holds for N_t iterations. The value ϵ is a small positive value. In the conducted experiments the value $\epsilon = 10^{-5}$ was used. If the termination rule of equation 6 does not hold, then the algorithm continues from Step 2.

4. **Return** the sample x^{best} in the population with the lower function value $f\left(x^{best}\right)$.

The optimization method described in section 2 combines evolutionary techniques, such as differential evolution, Armijo line search, and components of genetic algorithms, with the aim of finding the optimal solution. Initially, the population size and the dimensionality of the target function are defined, and the population samples are generated randomly using a uniform or k-means distribution. For each sample, the Euclidean distance (equation 2) to the other samples is calculated to identify the nearest one, followed by an Armijo line search (equation 3) to determine the optimal movement direction for the initial sample. Subsequently, a new offspring sample is created using a crossover process (equation 4), combining the current sample with the best discovered so far. A trial vector (equation 5) is then formed, which accounts for the adjustment of the computed samples and the differential coefficient parameter derived from the differential evolution algorithm. Periodically, local search methods, such as BFGS[68], are applied to improve the accuracy of the solution search. The method terminates when the best solution found remains nearly unchanged for a specified number of iterations. In summary, the basic steps for calculating a new sample are:

- Identification of the nearest point c_ι for each sample x_ι.
- Calculation of a sample $\mathrm{minLS}(x_i, c_i)$ through Armijo line search, between the sample x_i and the samplec_i.
- Generation of the sample using the crossover process of the Genetic Algorithm, between the sample x_ι and the best samplex^{best} .
- Computation of the trial point y_i using a process derived from Differential Evolution.

3. Experiments

Settings and benchmark functions

In the random number generator, different seeds were used to ensure the reliability of the experimental results, with the experiments being repeated 30 times. This process of repetition aims to minimize the likelihood of random errors and to enhance the validity of the results. The experiments were conducted on a system with an AMD Ryzen 5950X processor and 128 GB of RAM, operating in a Linux-Debian environment. Additionally, the open-source optimizer "GlobalOptimus" was used, which is a fully developed optimizer and is available for distribution via the link: http://www.github.com/itsoulos/GlobalOptimus (accessed on 17 September 2024). The benchmark functions used in the experimental measurements are presented in Table 1.

Table 1. The benchmark functions used in the conducted experiments.

NAME	FORMULA	DIMENSION
ACKLEY	$f(x) = -a\exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1) \quad a = 20.0$	2
BF1	$f(x) = -20\exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + 20 + \exp(1)$	2
BF1	$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$	2
BF2	$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$	2
BF3	$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$	2
DIFFPOWER	$f(x) = \sum_{i=1}^{n} \lvert x_i - y_i\rvert^p$	$n = 2\; p = 2, 5, 10$
CAMEL	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, \quad x \in [-5, 5]^2$	2
EASOM	$f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2-\pi)^2 - (x_1-\pi)^2\right)$	2
ELP	$f(x) = \sum_{i=1}^{n} (10^6)^{\frac{i-1}{n-1}} x_i^2$	$n = 10, 20, 30$
EXP	$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \le x_i \le 1$	$n = 4, 8, 16, 32$
F3	$f(x) = \left(e^{-2.0\log(2.0)\left(\frac{(x_1-0.08)}{0.854}\right)^2}\right)\left(\sin\left(5.0\pi\left(x_1^{\frac{3.0}{4.0}} - 0.05\right)\right)\right)^6 \quad x \in [0, 1]^n$	2
F5	$f(x) = \left(\left(4.0 - 2.1x_1^2 + \frac{x_1^4}{3.0}\right)x_1^2\right) + (x_1x_2) + \left((4.0x_2^2 - 4.0)x_2^2\right) \quad -5 \le x_i \le 5$	2
F9	$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad x \in [0, 1]^n$	2
GKLS[70]	$f(x) = \text{Gkls}(x, n, w)$	$n = 2, 3\; w = 50, 100$
GRIEWANK2	$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2}\frac{\cos(x_i)}{\sqrt{(i)}}$	2
GRIEWANK10	$\mathrm{f}(x) = 1 + \frac{1}{200}\sum_{i=1}^{10} x_i^2 - \prod_{i=1}^{10}\frac{\cos(x_i)}{\sqrt{(i)}}$	10
HANSEN	$f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$	2
HARTMAN3	$f(x) = -\sum_{i=1}^{4} c_i\exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3
HARTAMN6	$f(x) = -\sum_{i=1}^{4} c_i\exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6
POTENTIAL[71]	$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6\right]$	$n = 9, 15, 21, 30$
RARSTIGIN	$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2)$	2
ROSENBROCK	$f(x) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \le x_i \le 30$	$n = 4, 8, 16$
SCHWEFELH	$f(x) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	2
SCHWEFELH221	$f(x) = 418.9829n + \sum_{i=1}^{n} -x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	2
SCHWEFELH222	$f(x) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \le x_i \le 30$	2
Shekel5	$f(x) = -\sum_{i=1}^{5}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Shekel7	$f(x) = -\sum_{i=1}^{7}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Shekel10	$f(x) = -\sum_{i=1}^{10}\frac{1}{(x-a_i)(x-a_i)^T + c_i}$	4
Sinusoidal[72]	$f(x) = -\left(2.5\prod_{i=1}^{n}\sin(x_i - z) + \prod_{i=1}^{n}\sin(5(x_i - z))\right), \quad 0 \le x_i \le \pi$	$n = 4, 8$
Test2N	$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i$	$n = 4, 5, 7$
Test30N	$\frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i - 1)^2\left(1 + \sin^2(3\pi x_{i+1})\right)\right) + (x_n - 1)^2\left(1 + \sin^2(2\pi x_n)\right)$	$n = 3, 4$

The functions used in the conducted experiments have been proposed by various researchers [73,74] in the relevant literature. For a more accurate comparison of the methods, efforts were made to maintain certain parameter values at equal or similar levels. The values for the parameters of the algorithm are presented in Table2, along with some explanation of each parameter.

Table 2. Parameters of optimization methods settings

PARAMETER	VALUE	EXPLANATION
N	200	Number of samples for all methods
N_k	200	Maximum number of iterations for all methods
SR	$\left\lvert f_{\min}^{(k)} - f_{\min}^{(k-1)} \right\rvert$	Best fitness: Stopping rule for all methods
N_t	12	Similarity max count for all methods
LS	0.05 (5%)	Local search rate for all methods
F	0.8	Differential weight for Differential Evolution
CR	0.9	Crossover Probability for Differential Evolution
C_1, C_2	0.5	Parameters of PSO
G_c	0.1 (10%)	Crossover rate for Genetic Algorithm
G_m	0.05 (5%)	Mutation rate for Genetic Algorithm

Experimental results

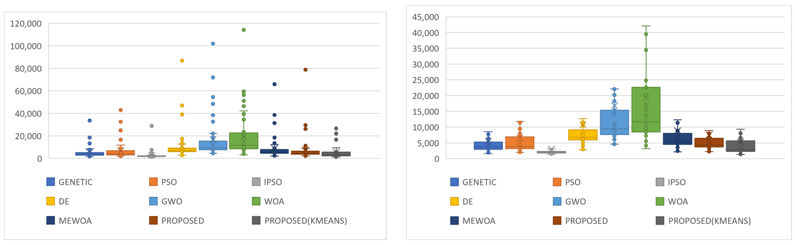
In Table 3, the average function calls for each method and every objective function is presented. Description of table captions and values:

1. The column FUNCTION represents the name of the used objective problem.
2. The column GENETIC represents the application of the Genetic Algorithm to the objective problem.
3. The PSO column represents the implementation of the classical PSO algorithm as suggested by the literature[33].
4. The column IPSO stands for the application of the Improved PSO algorithm as suggested by Charilogis and Tsoulos [75] to the objective problems.
5. The column DE represents the average function calls for the Differential Evolution optimization technique.
6. The column GWO represents the average function calls for the xxx optimization technique.
7. The column WOA represents the average function calls for the Whale Optimization technique.
8. The column MEWOA stands for the application of the Improved WOA algorithm as suggested by [75] to the objective problems.
9. The total sum of calls for each method is listed at the end of the table.
10. The success rate is indicated by the number in parentheses, which shows the executions in which the global optimum was successfully found. The absence of this number implies that the global minimum was computed with 100% success in all independent runs.

Table 3. Comparison of average function calls of proposed method against others

FUNCTION	GENETIC	PSO	IPSO	DE	GWO	WOA	ME
ACKLEY	6749	6885	3418	16183	20182	24766	13
BF1	4007	4113	1814	8268	9120	9924	7
BF2	3793	3747	1759	7913	8731	9597	6
BF3	3479	3305	1689	6327	7791	20117	5
BRANIN	2376	2522	1730	4101	6055	5939	3
CAMEL	2869	2908	1754	5609	6688	5917	4
DIFFPOWER2	5443	8657	2462	14669	22059	11436	11
DIFFPOWER5	18552	24894	4446	39018	49769	42095	31
DIFFPOWER10	18801	32534	4690	46914	54442	56381	38
EASOM	1958	1998	1753	2978	4615	3153	2
ELP10	3131	4397	1720	6288	9415	10317	5
ELP20	6160	6883	1988	10794	14173	17040	7
ELP30	9576	9438	2100	14172	18245	23532	9
EXP4	2946	3177	1677	5166	6648	5567	4
EXP16	3250	3477	1660	6498	6978	9968	4
EXP32	3561	3728	1647	7606	7419	11925	4
F3	4074	3420	1979(50)	7323	10816	11717	5
F5	1630	2055	1376	2814	48377	7393	2
F9	2494	2707	1879	4535	5502	4479	3
GKLS250	2280	2495	1623	3834	6127	4978	3
GKLS350	2612	2658	1570	3919	6919	7226(96)	3
GOLDSTEIN	3687	3856	1886	6781	9312	9016	5
GRIEWANK2	4500	3168	2299(96)	7429(96)	9253	7948(86)	7
GRIEWANK10	6409(96)	7942	2142(80)	18490	14708	51143	12
HANSEN	3209	2892	2204	4185	7253	10498	4
HARTMAN3	2751	3103	1801	5190	7293	8440	3
HARTMAN6	3219	3688	1862	5968	8814	22615	4
POTENTIAL3	4351	5154	3096	6218	11256	9232	5
POTENTIAL5	7704	10128	6353	9119	22136	39455	8
POTENTIAL6	10177(70)	11780(46)	7593	10509	32630(90)	46466(46)	113
POTENTIAL10	13357	16550(86)	10413	12721	38337	59487	18
RASTRIGIN	4106	3539	2211	6216	9043	8116	5
ROSENBROCK4	3679	5858	2025	8452	8078	10759	7
ROSENBROCK8	5269	7843	2085	11530	11167	14094	9
ROSENBROCK16	8509	11450	2432	17432	15590	23544	13
SCHWEFEL	1880	2178	1504	3437	4564	4158	3
SCHWEFEL221	2666	2529(70)	1875(70)	3909	71847	8519(96)	3
SCHWEFEL222	33724	42897	28965	86861	101960	114161	65
SHEKEL5	3325	3886	2018	6662	10043	17010	5
SHEKEL7	3360	4009	1994	6967(76)	10079	17873	5
SHEKEL10	2488	3985	1914(96)	6757(96)	10055	15524	5
SINU4	2990	3409	1961	5953	8132	10150	4
SINU8	3441	3995	1853	6973	10481	20782	5
SINU16	4320	4680	2251	6979	15356	34409	7
TEST2N4	3330	3390	1930	6396	8093	11876(96)	4
TEST2N5	4000	3604	1982(93)	6271(96)	8812	16256(86)	5
TEST2N7	4775	4020(96)	2157(73)	7074(73)	10124	25005(63)	636
TEST30N3	3210	4018	2432	6178	7647	8163	5
TEST30N4	3678	4504	3354	7006	10656	13596	7
TOTAL	261106	317168	145908	506409	802608	916996	41

At the end of every global optimization procedure, a BFGS variant of Powell [76] is utilized to improve the discovered solution and identify with certainty a local minimum. Additionally, Figure 1 presents a statistical comparison between the used optimization methods and the data provided from Table 3.



(a) With outliers appearing **(b)** Without outliers appearing

Figure 1. Comparison of function calls of proposed method against others

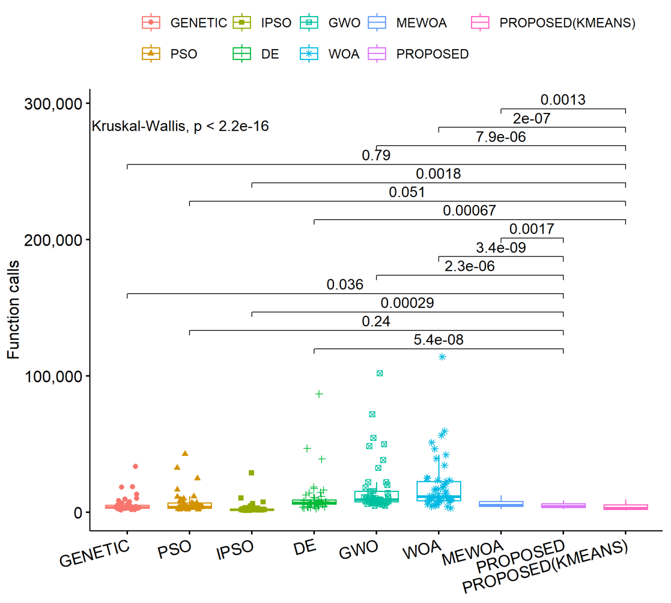


Figure 2. Statistical comparison of proposed method against others

The table 3 contains values representing objective function evaluations, with lower values indicating better performance. The goal is to minimize the number of function calls required to achieve a result. The algorithms analyzed are GENETIC, PSO, IPSO, DE, GWO, WOA, MEWOA, PROPOSED, and PROPOSED(KMEANS). The GENETIC algorithm, with a total of 261,106 evaluations, shows mixed results. In some functions, such as SHEKEL10 with 2488 evaluations, it performs well. However, in more demanding functions, like DIFFPOWER5 and DIFFPOWER10 with 18,552 and 18,801 evaluations respectively, it requires many calls, indicating reduced efficiency. The PSO algorithm, with 317,168 total evaluations, needs more calls than GENETIC and is generally considered less efficient. In difficult functions, such as DIFFPOWER10, it requires many evaluations, indicating reduced effectiveness. The IPSO algorithm is the most efficient, with only 145,908 evaluations. Its low values in several functions, such as ROSENBROCK16 with 2432 calls and TEST2N5 with 1982 calls, indicate that it requires fewer evaluations to achieve a result, making it the most efficient algorithm. The DE algorithm, with 506,409 evaluations, performs worse compared to IPSO and other algorithms. In difficult functions, such as DIFFPOWER10 with 46,914 calls and SCHWEFEL222 with 86,861 calls, it requires significantly more evaluations, indicating difficulties in optimizing efficiently. GWO has the most total evaluations (802,608), showing reduced efficiency. It encounters particular difficulties in hard problems, such as DIFFPOWER10 with 54,442 calls and SCHWEFEL222 with 101,960 calls. WOA, with 916,996 evaluations, is even less efficient than GWO. It faces serious issues in demanding functions such as DIFFPOWER10 (56,381 calls) and SCHWEFEL222 (114,161 calls), indicating poor performance. MEWOA (Modified WOA) has 419,939 total evaluations, showing some improvement over WOA but still less efficient than IPSO and PROPOSED+KMEANS. Despite the improvement, it continues to require many calls in difficult problems, such as DIFFPOWER10 (38,559 calls) and SCHWEFEL222 (65,835 calls). The PROPOSED algorithm, with 361,454 evaluations, performs better than DE but does not reach the efficiency of IPSO. It shows moderate results in some problems, such as DIFFPOWER10 with 29,549 calls, limiting its overall performance. The improved PROPOSED(KMEANS) algorithm achieves better results, with 257,033 total evaluations. In many functions, such as SCHWEFEL222 with 22,058 evaluations and DIFFPOWER10 with 26,615 evaluations, the addition of KMEANS significantly improves performance, making it competitive with IPSO. In conclusion, IPSO is the most efficient algorithm, with the lowest total number of evaluations, achieving better results with fewer calls. PROPOSED(KMEANS) also performs well, approaching the performance of IPSO. On the other hand, GWO and WOA have the worst performance, requiring the highest number of evaluations. MEWOA shows some improvements over WOA but remains less efficient than the top-performing algorithms.

In Figure [fig:StatProposedVSOthers], pairwise comparisons between the proposed methods and the other methods are presented. In all these comparisons, the critical parameter $p<0.05$ confirms the statistical significance of the results.

Overall, the proposed methods achieve a significant reduction in the required number of objective function calls and, in many cases, outperform other optimization techniques. Compared to the Differential Evolution (DE) algorithm, the proposed approaches perform better across all mathematical models, with the reduction in the number of calls exceeding 50%. This trend is confirmed by Table 3. Additionally, Figures 1a, 1b, and 2 present statistical comparisons between the global optimization methods used in the experiments, highlighting the superiority of the proposed approaches.

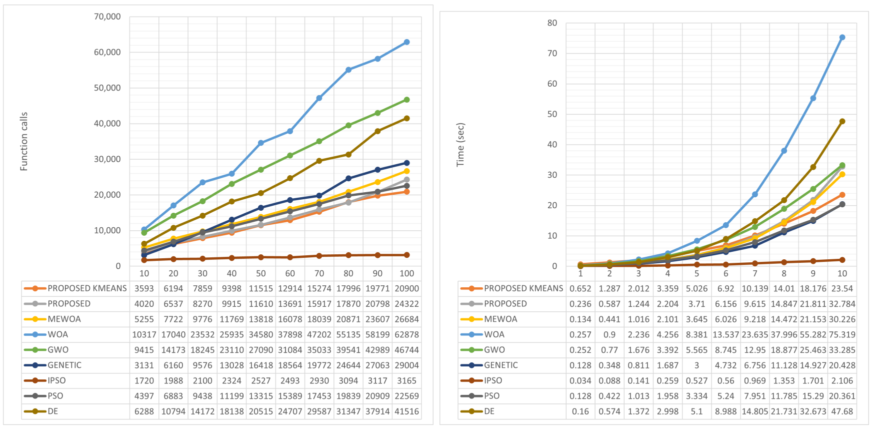
An additional experiment was executed for the High Elliptic function, which is defined as:

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

where the parameter n defines the dimension of the function. In this particular experiment, the proposed optimization method was systematically applied to a specific mathematical function as the dimension n underwent a transition from 10 to 100. Figure 3a presents the number of objective function calls required for each algorithm across different dimensions. As the problem dimension increases, the number of function calls needed to find a solution grows significantly for all algorithms, indicating the increasing complexity of the ELP function. For example, the PROPOSED(KMEANS) algorithm requires 3593 calls for 10 dimensions and 20,900 calls for 100 dimensions, showing a gradual increase. Similarly, the PROPOSED algorithm starts with 4020 calls for 10 dimensions and reaches 24,322 calls for 100 dimensions. IPSO stands out as the most efficient algorithm, with only 1720 calls at 10 dimensions and 3165 at 100, consistently maintaining low evaluations. In contrast, WOA exhibits the worst performance, with 10,317 calls at 10 dimensions and 62,878 at 100 dimensions. The rate of increase in calls varies among algorithms. For example, the PROPOSED algorithm increases from 4020 to 6537 calls between 10 and 20 dimensions (a 1.6 times increase), while the rise from 90 to 100 dimensions is more moderate, from 20,798 to 24,322 calls. MEWOA, although improved compared to WOA, requires more calls than the proposed methods, with 5255 calls at 10 dimensions and 26,684 at 100. GENETIC displays a variable pattern, with 3131 calls at 10 dimensions and 29,004 at 100. GWO and PSO follow a similar trend, with a gradual increase in calls as the dimension increases. Additionally, Figure 3b illustrates the execution time for each algorithm across different dimensions, measured in seconds. As expected, increasing the dimension leads to a significant rise in execution time for all algorithms, highlighting the increased computational load. For example, PROPOSED(KMEANS) starts with 0.652 seconds at 10 dimensions and reaches 23.54 seconds at 100. PROPOSED begins with 0.236 seconds at 10 dimensions and climbs to 32.784 seconds at 100, marking an impressive 139-fold increase. The fastest algorithm, IPSO, takes only 0.034 seconds at 10 dimensions and 2.106 seconds at 100, demonstrating excellent performance and scalability. On the other hand, WOA records the worst performance, with 75.319 seconds at 100 dimensions compared to 0.257 seconds at 10 dimensions. DE also struggles in execution time, increasing from 0.16 seconds at 10 dimensions to 47.68 seconds at 100. The increase in execution time across dimensions follows a nonlinear trend for most algorithms. For example, PROPOSED takes 0.587 seconds at 20 dimensions (more than double the 0.236 seconds at 10 dimensions) but reaches 14.847 seconds at 80 dimensions. Similarly, PSO starts at 0.128 seconds at 10 dimensions and reaches 20.361 seconds at 100 dimensions.

The data reveals significant differences in the scalability and efficiency of the algorithms. IPSO stands out as the most efficient algorithm, with the lowest number of calls and the shortest execution time. PROPOSED(KMEANS) also achieves satisfactory results, maintaining a balance between calls and execution time. On the other hand, WOA and GWO exhibit the worst performance, with significantly more calls and substantial increases in execution time as the dimension grows.

The results confirm that the complexity of optimization problems increases sharply with the dimension. Most algorithms struggle to maintain their efficiency in higher dimensions, highlighting the importance of selecting the appropriate algorithm according to the problem's requirements and the available computational resources.



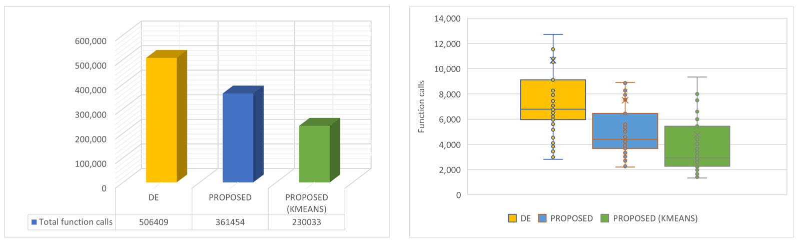
	10	20	30	40	50	60	70	80	90	100
PROPOSED KMEANS	3593	6194	7859	9398	11515	12914	15274	17996	19771	20900
PROPOSED	4020	6537	8270	9915	11610	13691	15917	17870	20798	24322
MEWOA	5255	7722	9776	11769	13818	16078	18039	20871	23607	26684
WOA	10317	17040	23532	25935	34580	37898	47202	55135	58199	62878
GWO	9415	14173	18245	23110	27090	31084	35033	39541	42989	46744
GENETIC	3131	6160	9576	13028	16418	18564	19772	24644	27063	29004
IPSO	1720	1988	2100	2324	2527	2493	2930	3094	3117	3165
PSO	4397	6883	9438	11199	13315	15389	17453	19839	20909	22569
DE	6288	10794	14172	18138	20515	24707	29587	31347	37914	41516

	1	2	3	4	5	6	7	8	9	10
PROPOSED KMEANS	0.652	1.287	2.012	3.359	5.026	6.92	10.139	14.01	18.176	23.54
PROPOSED	0.236	0.587	1.244	2.204	3.71	6.156	9.615	14.847	21.811	32.784
MEWOA	0.134	0.441	1.016	2.101	3.645	6.026	9.218	14.472	21.153	30.226
WOA	0.257	0.9	2.236	4.256	8.381	13.537	23.635	37.996	55.282	75.319
GWO	0.252	0.77	1.676	3.392	5.565	8.745	12.95	18.877	25.463	33.285
GENETIC	0.128	0.348	0.811	1.687	3	4.732	6.756	11.128	14.927	20.428
IPSO	0.034	0.088	0.141	0.259	0.527	0.56	0.969	1.353	1.701	2.106
PSO	0.128	0.422	1.013	1.958	3.334	5.24	7.951	11.785	15.29	20.361
DE	0.16	0.574	1.372	2.998	5.1	8.988	14.805	21.731	32.673	47.68

(a) Function calls **(b)** Times

Figure 3. Different variations of the ELP problem

In each iteration of the algorithm, a trial point is calculated through vector operations, similar to the process of optimization using differential evolution. The main difference between the proposed method compared to differential evolution lies in the fact that the samples for calculating the trial point are selected from nearby regions of the initial distribution, rather than being chosen randomly. However, the performance of the two methods differs in their ability to find optimal solutions, as shown in Figure4. In the case where the initial distribution is created by the k-means algorithm, the performance increases by 36.3%. The statistical analysis of the results in Figure 2 and Figure 4 indicates that the proposed methods (PROPOSED and PROPOSED+KMEANS) demonstrate statistically significantly better performance than the DE method in most cases. The use of the t-test confirms that these differences are not random but are due to substantial improvements in the efficiency of the proposed methods. The lower function call values show that the proposed methods lead to faster convergence, and thus, better optimization performance. The extraneous samples of Figure 4b have been removed.



(a) Comparison of total function calls **(b)** Statistical comparison of total function calls

Figure 4. Proposeds methods against DE

4. Conclusions

An innovative global optimization method has been proposed in this research paper, which leverages techniques derived from well-established optimization strategies. More specifically, the new method incorporates genetic operators from Genetic Algorithms alongside the Linear Search method to generate candidate solutions for the given objective functions. These candidate solutions are then combined to create new solutions utilizing approaches inspired by the Differential Evolution method. To validate the effectiveness of this new optimization approach, a comprehensive series of experiments were conducted on various problems sourced from the existing literature. Additionally, numerical comparisons were made with recognized global optimization techniques to provide a clear benchmark. The results indicate that the proposed optimization method exhibits significantly superior performance when compared to alternative methods, particularly in terms of the number of calls made to the objective function. Fewer calls to the objective function suggest better overall efficiency, highlighting the proposed method's ability to achieve optimal solutions with fewer evaluations. This efficiency is particularly critical in scenarios where each function call is computationally expensive. Statistical analyses, including both the t-test and the Kruskal-Wallis test, confirm that the observed differences in the number of calls between the proposed method and other methods are statistically significant, with a p-value of less than 0.05. This finding not only underscores the reduced resource consumption of the proposed method but also affirms that it delivers reliable results with enhanced efficiency. In summary, the proposed method stands out in terms of efficiency when compared to other optimization techniques, significantly decreasing the number of objective function calls and optimizing overall computational cost. Potential enhancements to the algorithm could involve identifying specific samples that contribute more effectively to the discovery of the optimal solution. Furthermore, since this method represents a novel approach to optimization, exploring alternative termination criteria or varying the initial sample distributions could lead to even greater performance improvements. By refining these aspects, the method could further bolster its efficiency and effectiveness in solving complex optimization problems.

Author Contributions: The conceptualization of the idea and the design of the methodology, as well as the supervision of the technical aspects related to the software, were undertaken by V.C., I.G.T., and G.K. The experiments were conducted using various datasets, and the comparative results were presented by V.C., G.K. and A.M.G. The statistical analysis was carried out by V.C. The manuscript

was prepared by G.K. and the other authors. All authors have reviewed and approved the final published version.

Funding: No external funding was received for this research.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Institutional Review Board Statement: Not applicable.

Acknowledgments: The experiments for this research were conducted using the high-performance computing system installed at the Knowledge and Intelligent Computing Laboratory, Department of Informatics and Telecommunications, University of Ioannina, which was acquired through the project "Educational Laboratory Equipment of TEI of Epirus" with MIS 5007094, funded by the Operational Programme "Epirus" 2014–2020, through ERDF and national funds

Conflicts of Interest: No conflicts of interest are declared by the authors.

References

1. Intriligator, M. D. (2002). Mathematical optimization and economic theory. Society for Industrial and Applied Mathematics.
2. Cánovas, M. J., Kruger, A., Phu, H. X., & Théra, M. (2020). Marco A. López, a Pioneer of Continuous Optimization in Spain. Vietnam Journal of Mathematics, 48, 211-219.
3. Mahmoodabadi, M. J., & Nemati, A. R. (2016). A novel adaptive genetic algorithm for global optimization of mathematical test functions and real-world problems. Engineering Science and Technology, an International Journal, 19(4), 2002-2021.
4. E. Iuliano, Global optimization of benchmark aerodynamic cases using physics-based surrogate models, Aerospace Science and Technology **67**, pp.273-286, 2017.
5. Q. Duan, S. Sorooshian, V. Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resources Research **28**, pp. 1015-1031 , 1992.
6. L. Yang, D. Robin, F. Sannibale, C. Steier, W. Wan, Global optimization of an accelerator lattice using multiobjective genetic algorithms, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **609**, pp. 50-57, 2009.
7. S. Heiles, R. L. Johnston, Global optimization of clusters using electronic structure methods, Int. J. Quantum Chem. **113**, pp. 2091– 2109, 2013.
8. W.H. Shin, J.K. Kim, D.S. Kim, C. Seok, GalaxyDock2: Protein–ligand docking using beta-complex and global optimization, J. Comput. Chem. **34**, pp. 2647– 2656, 2013.
9. A. Liwo, J. Lee, D.R. Ripoll, J. Pillardy, H. A. Scheraga, Protein structure prediction by global optimization of a potential energy function, Biophysics **96**, pp. 5482-5485, 1999.
10. Eva K. Lee, Large-Scale Optimization-Based Classification Models in Medicine and Biology, Annals of Biomedical Engineering **35**, pp 1095-1109, 2007.
11. Y. Cherruault, Global optimization in biology and medicine, Mathematical and Computer Modelling **20**, pp. 119-132, 1994.
12. 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.
13. Banga, J.R. Optimization in computational systems biology. BMC Syst. Biol. 2008, 2, 47.
14. Beites, T.; Mendes, M.V. Chassis optimization as a cornerstone for the application of synthetic biology based strategies in microbial secondary metabolism. Front. Microbiol. 2015, 6, 159095.
15. Filip, M.; Zoubek, T.; Bumbalek, R.; Cerny, P.; Batista, C.E.; Olsan, P.; Bartos, P.; Kriz, P.; Xiao, M.; Dolan, A.; et al. Advanced computational methods for agriculture machinery movement optimization with applications in sugarcane production. Agriculture 2020, 10, 434
16. Zhang, D.; Guo, P. Integrated agriculture water management optimization model for water saving potential analysis. Agric. Water Manag. 2016, 170, 5–19.
17. Intriligator, M.D. Mathematical Optimization and Economic Theory; SIAM: Philadelphia, PA, USA, 2002
18. Dixit, A.K. Optimization in Economic Theory; Oxford University Press: Oxford, MA, USA, 1990.
19. Ion, I. G., Bontinck, Z., Loukrezis, D., Römer, U., Lass, O., Ulbrich, S., ... & De Gersem, H. (2018). Robust shape optimization of electric devices based on deterministic optimization methods and

finite-element analysis with affine parametrization and design elements. Electrical Engineering, 100(4), 2635-2647.

20. Cuevas-Velásquez, V., Sordo-Ward, A., García-Palacios, J. H., Bianucci, P., & Garrote, L. (2020). Probabilistic model for real-time flood operation of a dam based on a deterministic optimization model. Water, 12(11), 3206.
21. Pereyra, M., Schniter, P., Chouzenoux, E., Pesquet, J. C., Tourneret, J. Y., Hero, A. O., & McLaughlin, S. (2015). A survey of stochastic simulation and optimization methods in signal processing. IEEE Journal of Selected Topics in Signal Processing, 10(2), 224-241.
22. Hannah, L. A. (2015). Stochastic optimization. International Encyclopedia of the Social & Behavioral Sciences, 2, 473-481.
23. Kizielewicz, B., & Sałabun, W. (2020). A new approach to identifying a multi-criteria decision model based on stochastic optimization techniques. Symmetry, 12(9), 1551.
24. Chen, T., Sun, Y., & Yin, W. (2021). Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. IEEE Transactions on Signal Processing, 69, 4937-4948.
25. M.A. Wolfe, Interval methods for global optimization, Applied Mathematics and Computation **75**, pp. 179-206, 1996.
26. T. Csendes and D. Ratz, Subdivision Direction Selection in Interval Methods for Global Optimization, SIAM J. Numer. Anal. **34**, pp. 922–938, 1997.
27. Y.D. Sergeyev, D.E. Kvasov, M.S. Mukhametzhanov, On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Sci Rep **8**, 453, 2018.
28. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
29. Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.
30. Charilogis, V., Tsoulos, I. G., & Stavrou, V. N. (2023). An Intelligent Technique for Initial Distribution of Genetic Algorithms. Axioms, 12(10), 980.
31. R. Storn, K. Price, Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization **11**, pp. 341-359, 1997.
32. J. Liu, J. Lampinen, A Fuzzy Adaptive Differential Evolution Algorithm. Soft Comput **9**, pp.448–462, 2005.
33. J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.
34. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
35. Ioan Cristian Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.
36. M. Dorigo, M. Birattari and T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine **1**, pp. 28-39, 2006.
37. K. Socha, M. Dorigo, Ant colony optimization for continuous domains, European Journal of Operational Research 185, pp. 1155-1173, 2008.
38. M. Neshat, G. Sepidnam, M. Sargolzaei, et al, Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications, Artif Intell Rev **42**, pp. 965–997, 2014.
39. Tq. Wu, M. Yao, Jh. Yang, Dolphin swarm algorithm, Frontiers Inf Technol Electronic Eng **17**, pp. 717–729, 2016.
40. Mirjalili, S., Lewis, A. (2016).: The whale optimization algorithm. Elsevier: Advances in Engineering Software 95 (2016) 51–67.
41. Nasiri, J., & Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach for clustering. Cogent Mathematics & Statistics, 5(1), 1483565.
42. Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale Optimization Algorithm and its applications. Swarm and Evolutionary Computation, 48, 1-24.
43. Y. Zhou and Y. Tan, "GPU-based parallel particle swarm optimization," 2009 IEEE Congress on Evolutionary Computation, pp. 1493-1500, 2009.
44. L. Dawson and I. Stewart, "Improving Ant Colony Optimization performance on the GPU using CUDA," 2013 IEEE Congress on Evolutionary Computation, 2013, pp. 1901-1908, doi: 10.1109/CEC.2013.6557791.
45. Barkalov, K., Gergel, V. Parallel global optimization on GPU. J Glob Optim 66, 3–20 (2016).
46. Holland, J.H. Genetic algorithms. Sci. Am. 1992, 267, 66–73.

47. Y.H. Santana, R.M. Alonso, G.G. Nieto, L. Martens, W. Joseph, D. Plets, Indoor genetic algorithm-based 5G network planning using a machine learning model for path loss estimation, Appl. Sci. **12**, 3923. 2022.
48. X. Liu, D. Jiang, B. Tao, G. Jiang, Y. Sun, J. Kong, B. Chen, Genetic algorithm-based trajectory optimization for digital twin robots, Front. Bioeng. Biotechnol **9**, 793782, 2022.
49. K. Nonoyama, Z.Liu, T. Fujiwara, M.M. Alam, T. Nishi, Energy-efficient robot configuration and motion planning using genetic algorithm and particle swarm optimization, Energies **15**, 2074, 2022.
50. K. Liu, B. Deng, Q. Shen, J. Yang, Y. Li, Optimization based on genetic algorithms on energy conservation potential of a high speed SI engine fueled with butanol–gasoline blends, Energy Rep. **8**, pp. 69–80, 2022.
51. G. Zhou, S. Zhu, S. Luo, Location optimization of electric vehicle charging stations: Based on cost model and genetic algorithm, Energy **247**, 123437, 2022.
52. J. Arifovic, R. Gençay, Using genetic algorithms to select architecture of a feedforward artificial neural network, Physica A: Statistical Mechanics and its Applications **289**, pp. 574-594, 2001.
53. F. H. F. Leung, H. K. Lam, S. H. Ling, P. K. S. Tam, Tuning of the structure and parameters of a neural network using an improved genetic algorithm, IEEE Transactions on Neural Networks **14**, pp. 79-88, 2003.
54. 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.
55. 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.
56. Maulik, U.; Saha, I. Automatic Fuzzy Clustering Using Modified Differential Evolution for Image Classification. IEEE Trans. Geosci. Remote Sens. 2010, 48, 3503–3510.
57. Zhang, Y.; Zhang, H.; Cai, J.; Yang, B. A Weighted Voting Classifier Based on Differential Evolution. Abstr. Appl. Anal. 2014, 2014, 376950.
58. Hancer, E. Differential evolution for feature selection: A fuzzy wrapper–filter approach. Soft Comput. 2019, 23, 5233–5248.
59. Vivekanandan, T.; Iyengar, N.C.S.N. Optimal feature selection using a modified differential evolution algorithm and its effectiveness for prediction of heart disease. Comput. Biol. Med. 2017, 90, 125–136.
60. Deng, W.; Liu, H.; Xu, J.; Zhao, H.; Song, Y. An Improved Quantum-Inspired Differential Evolution Algorithm for Deep Belief Network. IEEE Trans. Instrum. Meas. 2020, 69, 7319–7327.
61. Wu, T.; Li, X.; Zhou, D.; Li, N.; Shi, J. Differential Evolution Based Layer-Wise Weight Pruning for Compressing Deep Neural Networks. Sensors 2021, 21, 880.
62. H. Badem, A. Basturk, A. Caliskan, M.E. Yuksel, A new hybrid optimization method combining artificial bee colony and limited-memory BFGS algorithms for efficient numerical optimization, Applied Soft Computing **70**, pp. 826-844, 2018.
63. A.A. Nagra, F. Han, Q.H. Ling, An improved hybrid self-inertia weight adaptive particle swarm optimization algorithm with local search, Engineering Optimization **51**, pp. 1115-1132, 2018.
64. Li, S., Tan, M., Tsang, I. W., & Kwok, J. T. Y. (2011). A hybrid PSO-BFGS strategy for global optimization of multimodal functions. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 41(4), 1003-1014.
65. Andalib Sahnehsaraei, M., Mahmoodabadi, M. J., Taherkhorsandi, M., Castillo-Villar, K. K., & Mortazavi Yazdi, S. M. (2015). A hybrid global optimization algorithm: particle swarm optimization in association with a genetic algorithm. Complex System Modelling and Control Through Intelligent Soft Computations, 45-86.
66. Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. Pacific Journal of mathematics, 16(1), 1-3.
67. M. Ahmed, R. Seraj, S.M.S. Islam, The k-means algorithm: A comprehensive survey and performance evaluation, Electronics **9**, 1295.
68. , Convergence properties of the BFGS algoritm. SIAM Journal on Optimization **13**, pp. 693-701, 2002.
69. Charilogis, V.; Tsoulos, I.G. Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions. Information 2022, 13, 217.
70. M. Gaviano, D.E. Ksasov, D. Lera, Y.D. Sergeyev, Software for generation of classes of test functions with known local and global minima for global optimization, ACM Trans. Math. Softw. **29**, pp. 469-480, 2003.

71. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
72. Z.B. Zabinsky, D.L. Graesser, M.E. Tuttle, G.I. Kim, Global optimization of composite laminates using improving hit and run, In: Recent advances in global optimization, pp. 343-368, 1992.
73. M. Montaz Ali, Charoenchai Khompatraporn, Zelda B. Zabinsky, A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems, Journal of Global Optimization **31**, pp 635-672, 2005.
74. C.A. Floudas, P.M. Pardalos, C. Adjiman, W. Esposoto, Z. Gümüs, S. Harding, J. Klepeis, C. Meyer, C. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, Dordrecht, 1999.
75. V. Charilogis, I.G. Tsoulos, Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions , Information **13**, 217, 2022.
76. Powell, M.J.D. A Tolerant Algorithm for Linearly Constrained Optimization Calculations. Math. Program. 1989, 45, 547–566.