MDPI

Article

Optimization: parallel stochastic methods and mixed termination rules

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

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; itsoulos@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, Greece; am.gianni@uoi.gr
* Correspondence: itsoulos@uoi.gr;
† Current address: Department of Informatics and Telecommunications, University of Ioannina, Greece.
‡ These authors contributed equally to this work.

Abstract: Parallel optimization constitutes a powerful tool for solving optimization problems in various domains. By harnessing multiple computational resources simultaneously, optimization methods can achieve faster convergence and improved performance. This specific parallel implementation involves the concurrent execution of different methods, such as evolutionary algorithms, swarm-based optimization, and the utilization of multiple restarts. The primary objective is the efficient exploration of the search space and the attainment of optimal solutions in shorter time frames without squandering computational power. However, defined termination criteria are essential to prevent uncontrolled execution of the algorithm, aiming to conserve computational resources and time. Within the scope of this study, an innovative combination of termination rules and a mechanism for transferring optimal solutions among different methodological approaches is proposed. The proposed enhancements have been tested on a series of well-known optimization problems from relevant literature, and the results are reported here.

Keywords: Global optimization; Parallel techiniques; Termination rules; Evolutionary techniques

Citation: Charilogis V.; Tsoulos I.G.; Gianni A.M; Optimization: parallel stochastic methods and mixed termination rules. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

1. Introduction

The science of optimization refers to the process of acquiring the best solution from all available alternatives in a given problem [1]. Specifically, it pertains to seeking the optimal solution of an optimization problem throughout the entire search space (Euclidean space), aiming to minimize or maximize a cost or performance function. The search space can be complex and multidimensional, with various constraints and parameters. The main issue is that searching for the best solution across the entire space can require excessive time and computational resources, especially when the search space is large.

A typical type of global optimization problem involves minimizing a function $f(x)$, where x is a vector of parameters, and $f(x)$ represents the cost or performance function that needs to be minimized. The parameters x are constrained to a search space S, where each parameter x_i must take values within the interval $[a_i, b_i]$. The search for the optimal solution involves locating the point that minimizes the function $f(x)$ within the search space S, that is:

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

The problem of global optimization is typically complex and requires the use of various techniques such as local optimization, parallelization, or advanced optimization methods that consider the structures of the problem.

The problem of global optimization can be divided into three main parts:

- Decision Variables: These are the parameters or variables that can be adjusted or changed during the optimization process. In the context of the problem, these variables represent the potential solutions that are evaluated to find the optimum. Their quantity determines the dimensionality of the problem.
- Constraints: These are the restrictions or limiting conditions that must be taken into account when searching for the optimal solution. These constraints define the domain of feasible solutions and ensure that the solutions being considered meet the requirements of the problem. In other words, they ensure the domain of the objective function.
- Objective functions or evaluation functions or test functions: These represent the mathematical model to be optimized. The goal is to find the optimum value, minimum when minimizing and maximum when maximizing, for this function, which usually represents a performance or efficiency metric.

Every optimization problem requires the development of a suitable method or algorithm to search for the optimal solution, taking into account the aforementioned aspects of the problem. These methods can be classified into the following categories:

- Based on mathematics: The Newton-Raphson method & Delta Coefficient [2], the Gradient Descent method [3], Heuristic Methods [4], etc.
- Based on physics: Simulated Annealing (SA) [5], Henry's Gas Solubility Optimization (HGSO) [6], Gravitational Search Algorithm (GSA) [7], Small World Optimization Algorithm (SWOA) [8], etc.
- Based on natural evolution of living organisms: Genetic Algorithm (GA) [9], Differential Evolution (DE) [10,11], Evolutionary Programming (EP) [12], Genetic Programming (GP) [13], etc.
- Based on behaviors of living organisms: Particle Swarm Optimization (PSO) [14,15], Ant Colony Optimization (ACO) [16], Bat Algorithm (BA) [17], Whale Optimization Algorithm (WOA) [18], Grasshopper Optimization Algorithm (GOA) [19], etc.
- Based on human society: Teaching-Learning-Based Optimization (TLBO) [20], Group Counseling Optimizer (GCO) [21], Maternal Optimization Algorithm (MOA) [22], etc.
- Based on games: Golf Optimization Algorithm (GOA) [23], Ring Toss Game-Based Optimization (RTGBO) [24], Volleyball Premier League Algorithm (VPL) [25], etc.

However, the above optimization methods require significant amounts of computational power and time; therefore, parallelization of the methods is essential. Parallel optimization represents a significant approach in the field of optimization problem solving and is applied across a wide range of applications, such as optimization of machine learning model parameters [26–29], control system design and optimization [30–32], energy and resource management [33–36], as well as problems related to sustainable development and enhancing sustainability [37–39].

By harnessing multiple computational resources, parallel optimization allows for the simultaneous execution of multiple algorithms, leading to faster convergence and improved performance. These computational resources can communicate with each other to exchange information and synchronize processes, thereby contributing to faster convergence towards common solutions. Additionally, leveraging multiple resources enables more effective handling of exceptions and errors, while increased computational power for conducting more trials or utilizing more complex models leads to enhanced performance[40]. Of course, this process requires the development of suitable algorithms and techniques to effectively manage and exploit available resources. Each parallel optimization algorithm requires a coherent strategy for workload distribution among the available resources, as well as an efficient method for collecting and evaluating results.

In this work, the parallelization of three different stochastic optimization methods belonging to distinct categories was implemented1. These include Particle Swarm Optimization (PSO) 2.2, Multistart2.3, and Differential Evolution (DE)2.1. PSO is an optimization method inspired by the behavior of swarms in nature. In PSO, a set of particles moves in the search space seeking the optimal solution. Each particle has a current position and

velocity, and it moves based on its past performance and that of its neighboring particles. PSO continuously adjusts the movement of particles aiming for convergence to the optimal solution [41–43]. In contrast, the Multistart method follows a different approach. Instead of focusing on a single initial point, it repeats the search from various initial points in the search space. This allows for a broader exploration of the search space and increases the chances of finding the optimal solution. The Multistart method is particularly useful when optimization algorithms may get trapped in local minima [44–46]. Finally, the DE method relies on differential operators and is particularly effective in optimization problems that involve searching through a continuous search space. By employing differential operators, DE generates new solutions that are then evaluated and adjusted until the optimal solution is achieved [10,47,49].

The following sections are organized as follows: In section 2, the three main algorithms participating in the overall algorithm are described. In section 3 the algorithm for parallelizing the three methods is described, along with the proposed mechanism for propagating the optimal solution to the remaining methods. In section 4, termination rules are described, along with the proposed termination rule. In section 5, experimental models and experimental results are described. Finally, in section 6, the conclusions from the application of the current work are discussed.

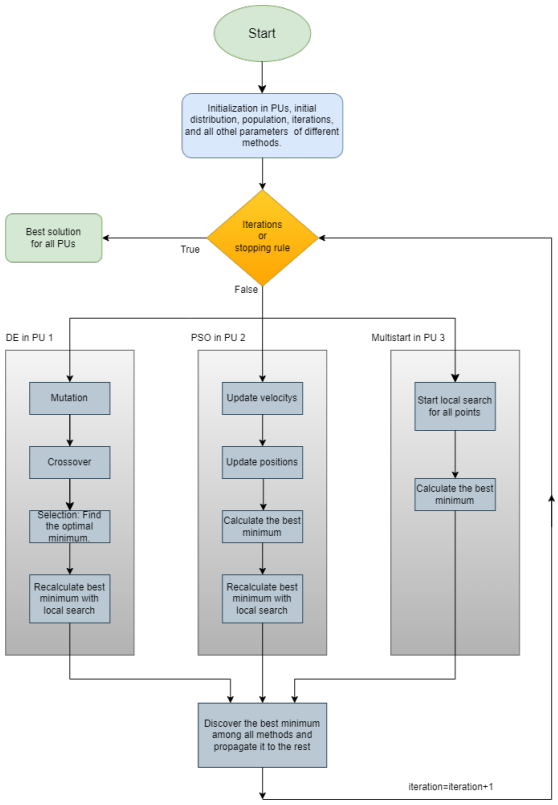

Figure 1. Flowchart of the overall process

In Figure 1, the flowchart of the general algorithm is depicted, illustrating the collaboration among the three methods. Additionally, the proposed mechanism for disseminating the optimal solution from one method to the others is also shown.

2. Methods description

2.1. The method: Differential evolution

The Differential Evolution operates through the following steps:

Initially, initialization takes place with a random population of solutions in the search space. Then, each solution is evaluated based on the objective function. Subsequently,

a process is iterated involving the generation of new solutions through modifications, evaluation of these new solutions, and selection of the best ones for the next generation. The algorithm terminates when a termination criterion is met, such as achieving sufficient improvement in performance or exhausting the number of iterations.

The main steps of the DE method are presented in the Algorithm.1

Algorithm 1 The main steps of the DE method.

1. **INPUT:**
   (a) The population size $N_d \geq 4$. The members of this population are also called agents.
   (b) The crossover probability CR $\in [0, 1]$.
   (c) The differential weight $F \in [0, 2]$.
2. **OUTPUT**:
   (a) The agent x_{best} with the lower function value $f(x_{\text{best}})$.
3. **Initialize** all agents in S.
4. **While** termination criteria are not hold **do**
   (a) **For** $i = 1 \ldots N_d$ **do**
      i. **Select** as x the agent i.
      ii. **Select** randomly three agents a, b, c with the property $a \neq b$, $b \neq c$, $c \neq a$.
      iii. **Select** a random position $R \in \{1, \ldots, n\}$
      iv. **Create** the vector $y = [y_1, y_2, \ldots, y_n]$ with the following procedure
      v. **For** $j = 1, \ldots, n$ **do**
         A. **Set** $r_i \in [0, 1]$ a random number.
         B. **If** $r_j <$ CR **or** $j = R$ **then** $y_j = a_j + F \times (b_j - c_j)$ **else** $y_j = x_j$.
      vi. If $y \in S$ AND $f(y) \leq f(x)$ then $x = y$.
      vii. **EndFor**
   (b) **EndFor**
5. **End While**

2.2. The method: Particle swarm optimization

The Particle Swarm Optimization (PSO) is an optimization method inspired by the behavior of animals in nature. In PSO, particles move within a solution search space, determining their position and velocity. Based on attraction towards the best-known positions (local search) and the best position found by any particle (global search), particles collaborate to explore and exploit the search space.

The main steps of the PSO method are presented in the Algorithm.2

Algorithm 2 The main steps of the PSO method.

1. **Initialization**.
   (a) **Set** iter $= 0$ (iteration counter).
   (b) **Set** the number of particles N_p.
   (c) **Set** the maximum number of iterations allowed $\text{iter}_{\max}$
   (d) **Set** the local search rate $p_l \in [0, 1]$.
   (e) **Initialize** randomly the positions of the m particles $x_1, x_2, ..., x_m$, with $x_i \in S \subset R^n$
   (f) **Initialize** randomly the velocities of the m particles $u_1, u_2, ..., u_m$, with $u_i \in S \subset R^n$
   (g) **For** $i = 1..N_p$ do $p_i = x_i$. The p_i vector are the best located values for every particle i.
   (h) **Set** $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$
2. **Termination Check**. Check for termination. If termination criteria are met then stop.
3. **For** $i = 1..N_p$ **Do**
   (a) **Update** the velocity:
   $$u_i = \omega u_i + r_1 c_1 (p_i - x_i) + r_2 c_2 \left(p_{\text{best}} - x_i\right)$$
   The parameters $r_1,\ r_2$ are random numbers with $r_1 \in [0, 1]$ and $r_2 \in [0, 1]$.
   The constant number $c_1,\ c_2$ are in the range $[1, 2]$.
   The variable ω is called inertia, with $\omega \in [0, 1]$.
   (b) **Update** the position
   $$x_i = x_i + u_i$$
   (c) **Set** $r \in [0, 1]$ a random number. If $r \leq p_m$ then $x_i = \text{LS}(x_i)$, where $\text{LS}(x)$ is a local search procedure.
   (d) **Evaluate** the fitness of the particle i, $f(x_i)$
   (e) **If** $f(x_i) \leq f(p_i)$ then $p_i = x_i$
4. **End For**
5. **Set** $p_{\text{best}} = \arg\min_{i \in 1..m} f(x_i)$
6. **Set** iter $=$ iter $+ 1$.
7. **Goto** Step 2

2.3. The method: Multistart

The approach of multiple starts belongs to the simplest techniques for global optimization. At the outset of the process, an initial distribution of points is made in the Euclidean space. Subsequently, local optimization begins simultaneously from these points. The discovered minima are compared, and the best one is retained as the global minimum. Local optimizations rely on the BFGS method.

The main steps of the multistart method are presented in the Algorithm.3.

Algorithm 3 The main steps of the Multistart method.

1. **Initialization** step.
   (a) **Set** N_m as the total number of samples.
   (b) **Set** (x^*, y^*) as the global minimum. Initialize y^* to a very large value.
2. **Sampling** step.
   (a) **For** $i = 1 \ldots N_m$ **Do**
      i. **Sample** a point $x_i \in S$
      ii. $y_i = \text{LS}(x_i)$. Where LS(x) is a local search procedure.
      iii. **If** $y_i \leq y^*$ then $x^* = x_i, y^* = y_i$
   (b) **EndFor**

3. Parallelization of stochastic methods

In scientific literature, parallelization typically involves distributing the population among parallel computing units to save computational power and time[53,54]. In the present study, we concurrently compute the optimal solutions of three different stochastic optimization methods originating from different classification categories.

The proposed overall algorithm follows:

Algorithm 4 The proposed overall algorithm

1. **Set** as $N_I = 3$ the total number of parallel processing units.
2. **Set** $K = 0$ the iteration number.
3. **For** $j = 1, .., N_I$ do in parallel
   (a) **Execute** an iteration of the DE algorithm described in algorithm 1 on processing unit 1 and recalculate the optimal minimum using a local optimization approach.
   (b) **Execute** an iteration of the PSO algorithm described in algorithm 2 on processing unit 2 and and recalculate the optimal minimum using a local optimization approach.
   (c) **Execute** an iteration of the Multistart algorithm described in algorithm 3 on processing unit 3.
   (d) **Find** the best element from all optimization methods j and **propagate** it to the rest of processing units.
4. **End For**
5. **Update** $K = K + 1$
6. **Check** the proposed termination rule. If the termination rule is valid, then goto step 6a else goto step 3.
   (a) **Terminate** and report the best value from all processing units.

4. Termination rules

The termination criteria that follow leverage the similarity parameter.

4.1. The termination rule: Best fitness

In this termination technique, at each iteration k, the difference between the current best value $f_{min}^{(k)}$ and the previous best value $f_{min}^{(k+1)}$ is calculated, i.e., the absolute difference:

$$\left| f_{\text{min}}^{(k)} - f_{\text{min}}^{(k-1)} \right| \tag{2}$$

If this difference is zero for a series of predefined consecutive iterations N_k, then the method terminates.

4.2. The termination rule: Mean or sum fitness arrray

In this termination rule, we compute the average function value for each generation of the population. If this value remains relatively unchanged over a certain number of consecutive iterations, it indicates that the method may not be making significant progress towards discovering a new global minimum. Therefore, termination is warranted.

Hence, in every iteration k, we compute:

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

The termination rule is defined as follows: terminate if $\delta^{(k)} \leq \epsilon$ for a predefined number N_k of iterations.

4.3. The termination rule: Double box

The termination criterion used in this study was originally introduced in the research conducted by Tsoulos [48]. According to this criterion, the algorithm terminates when one of the following conditions is met:

- The iteration k count exceeds a predefined limit of iterations k_{max}
- The relative variance $\sigma^{(iteration)}$ falls below half of the variance $\sigma^{(k_{last})}$ of the last itaration $k_{(last)}$ where a new optimal functional value was found.

$$iteration \geq N_g \, or \, \sigma^{(iteration)} \leq \frac{\sigma^{(klast)}}{2} \tag{4}$$

4.4. The proposed termination rule: All

This termination criterion is the combination of the aforementioned simirarity types. Optimization termination is achieved when any homogeneity condition is satisfied, resulting in improved speed. Therefore, the following relationships hold 2 3 4:

$$\left| f_{\text{min}}^{(k)} - f_{\text{min}}^{(k-1)} \right| or$$

$$\delta^{(k)} = \left| \sum_{i=1}^{\text{NP}} \left| f_i^{(k)} \right| - \sum_{i=1}^{\text{NP}} \left| f_i^{(k-1)} \right| \right| or$$

$$\sigma^{(iteration)} \leq \frac{\sigma^{(klast)} o}{2} \, or$$

$$iteration \geq N_k$$

5. Experiments

All experiments conducted were repeated 30 times to ensure the reliability of the algorithm producing the results. The following table presents the critical parameters of the three global optimization methods that participated. The parallelization was achieved using the OpenMP library [58], while the implementation of the method was done in ANSI C++ within the optimization package OPTIMUS, available at https://github.com/itsoulos/OPTIMUS.

The total population N_e regardless of the combination of optimization methods. For example, if all three methods participate and the total population is $N_e = 120$, then the population of DE will be $N_d = 40$, of PSO will be $N_p = 40$, and of Multistart will be $N_m = 40$.

Table 1. The following parameters were considered for conducting the experiments

Parameter	value	Explanation
N_e	120	Total elements
N_k	200	Max iterations
N_I	3	Processing units
N_P	1	Elements for propagation
TR	Best fitness 4.1, Mean fitness 4.2, Double Box 4.3, All 4.4	Termination rule
LSR	0.1% (DE and PSO)	Local search rate
N_s	15 (DE, PSO and Multistart)	Similarity max count
F	0.8 (DE)	Differential weight
CR	0.9 (DE)	Crossover Probability
c_1	0.5 (PSO)	Parameter of PSO
c_2	0.5 (PSO)	Parameter of PSO

5.1. Test functions

The test functions [55,56] presented below exhibit varying levels of difficulty in solving them; hence, a periodic local optimization mechanism has been incorporated. Periodic local optimization plays a crucial role in increasing the success rate in locating the minima of functions. This addition appears to lead to a success rate approaching 100% for all functions, regardless of their characteristics such as dimensionality, minima, scalability, and symmetry. A study by Z.-M. Gao and colleagues [57] specifically examines the issue of symmetry and asymmetry in the test functions.

- **Bent Cigar function** The function is

$$f(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$$

with the global minimum $f(x^*) = 0$. For the conducted experiments the value $n = 10$ was used.

- **Bf1** function. The function Bohachevsky 1 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$$

with $x \in [-100, 100]^2$.

- **Bf2** function. The function Bohachevsky 2 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$$

with $x \in [-50, 50]^2$.

- **Branin** function. The function is defined by $f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$ with $-5 \leq x_1 \leq 10$, $0 \leq x_2 \leq 15$. The value of global minimum is 0.397887.with $x \in [-10, 10]^2$.

- **CM** function. The Cosine Mixture function is given by the equation

$$f(x) = \sum_{i=1}^{n} x_i^2 - \frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$$

with $x \in [-1,1]^n$. For the conducted experiments the value $n = 4$ was used.

- **Discus function** The function is defined as

$$f(x) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$$

with global minimum $f(x^*) = 0$. For the conducted experiments the value $n = 10$ was used.

- **Easom** function The function is given by the equation

$$f(x) = -\cos(x_1)\cos(x_2)\exp\Big((x_2-\pi)^2 - (x_1-\pi)^2\Big)$$

with $x \in [-100,100]^2$ and global minimum -1.0

- **Exponential** function. The function is given by

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

The global minimum is situated at $x^* = (0,0,...,0)$ with a value -1. In our experiments, we applied this function for $n = 4, 16, 64$, and referred to the respective instances as EXP4, EXP16 .

- **Griewank** function with n=2, 10 dimensions:

$$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\frac{\cos(x_i)}{\sqrt{(i)}}, \quad x \in [-100,100]^2$$

The global minimum is located at the $x^* = (0,0,...,0)$ with value 0.

- **Gkls** function. $f(x) = \text{Gkls}(x,n,w)$, is a function with w local minima, described in [50] with $x \in [-1,1]^n$ and n a positive integer between 2 and 100. The value of the global minimum is -1 and in our experiments we have used $n = 2,3$ and $w = 50,\ 100$.
- **Hansen** function. $f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j], x \in [-10,10]^2$. The global minimum of the function is -176.541793.
- **Hartman 3** function. The function is given by

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0,1]^3$ and $a = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}$$

The value of global minimum is -3.862782.

- **Hartman 6** function.

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0,1]^6$ and $a = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}$$

the value of global minimum is -3.322368.

- **High Conditioned Elliptic** function, defined as

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

Featuring a global minimum at $f(x^*) = 0$, the experiments were conducted using the value $n = 10$

- **Potential** function. As a test case, the molecular conformation **Find**corresponding to the global minimum of the energy of N atoms interacting via the Lennard-Jones potential [51] is utilized. The function to be minimized is defined as follows:
-

$$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{5}$$

In the current experiments two different cases were studied: $N = 3,\ 5,\ 7,\ 10$

- **Rastrigin** function. The function is given by

$$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2), \quad x \in [-1,1]^2$$

- **Shekel 5** function.

$$f(x) = -\sum_{i=1}^{5} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{pmatrix}$.

- **Shekel 7** function.

$$f(x) = -\sum_{i=1}^{7} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{pmatrix}$.

- **Shekel 10** function.

$$f(x) = -\sum_{i=1}^{10} \frac{1}{(x-a_i)(x-a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.6 \end{pmatrix}$. 229

- **Sinusoidal** function. The function is given by 230

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

The global minimum is situated at $x^* = (2.09435, 2.09435, ..., 2.09435)$ with a value 231
$f(x^*) = -3.5$. In the performed experiments, we examined scenarios with $n = 4, 8$ 232
and $z = \frac{\pi}{6}$. The parameter $z$is employed to offset the position of the global minimum 233
[52]. 234

- **Test2N** function. This function is given by the equation 235

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

The function has 2^n in the specified range and in our experiments we used $n = 4, 9$. 236

- **Test30N** function. This function is given by 237

$$f(x) = \frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i-1)^2\left(1+\sin^2(3\pi x_{i+1})\right)\right) + (x_n-1)^2\left(1+\sin^2(2\pi x_n)\right)$$

with $x \in [-10,10]$. The function has 30^n local minima in the specified range and we 238
used $n = 3, 4$ in the conducted experiments. 239

5.2. Experimental results

Table 2. Differential evolution: Function calls with different termination rulls

PROBLEMS	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	32561	6761	10511	10369	6114
BF2	30339	7693	9222	11524	7560
BRANIN	21650	4639	10982	4907	4289
CAMEL	27444	6116	13290	9905	5940
CIGAR10	31854	7647	4348	21116	3111
CM4	31504	7913	4175	11851	4079
DISCUS10	25477	5191	2042	13614	2034
EASOM	19591	1917	15103	1721	1721
ELP10	25870	6046	24675	12519	6031
EXP4	27134	5216	18151	6385	5040
EXP16	28161	5588	27387	7552	5339
GKLS250	25362	5227	2641	8379	2641
GKLS350	24645	5624	3298	17437	3206
GRIEWANK2	29342	8027	10458	14756	6915
GRIEWANK10	38706	9664	37839	17312	9539
POTENTIAL3	26013	5278	24823	13871	5256
PONTENTIAL5	32876	8225	32439	20828	7742
PONTENTIAL6	35041	8467	34946	19169	8197
PONTENTIAL10	41410	10330	41300	26308	10643
HANSEN	23069	5219	23050	14245	4263
HARTMAN3	24165	4613	17966	7333	4566
HARTMAN6	25963	5734	17109	9354	5550
RASTRIGIN	29501	5912	17240	8987	5999
ROSENBROCK8	34964	8503	34429	20980	8051
ROSENBROCK16	41395	10052	41335	26156	8307
SHEKEL5	25810	6064	25800	19877	5933
SHEKEL7	26177	5597	26168	20274	5677
SHEKEL10	26489	6037	26439	22960	5100
SINU4	24646	5751	24620	12511	4776
SINU8	25355	6340	25477	18741	4980
TEST2N4	25987	5848	24947	11640	5727
TEST2N9	26698	7303	26571	16165	6288
TEST30N3	25560	3169	8395	8134	2869
TEST30N4	25021	2841	10381	7853	2714
TOTAL	**965780**	**214552**	**677557**	**474733**	**186197**

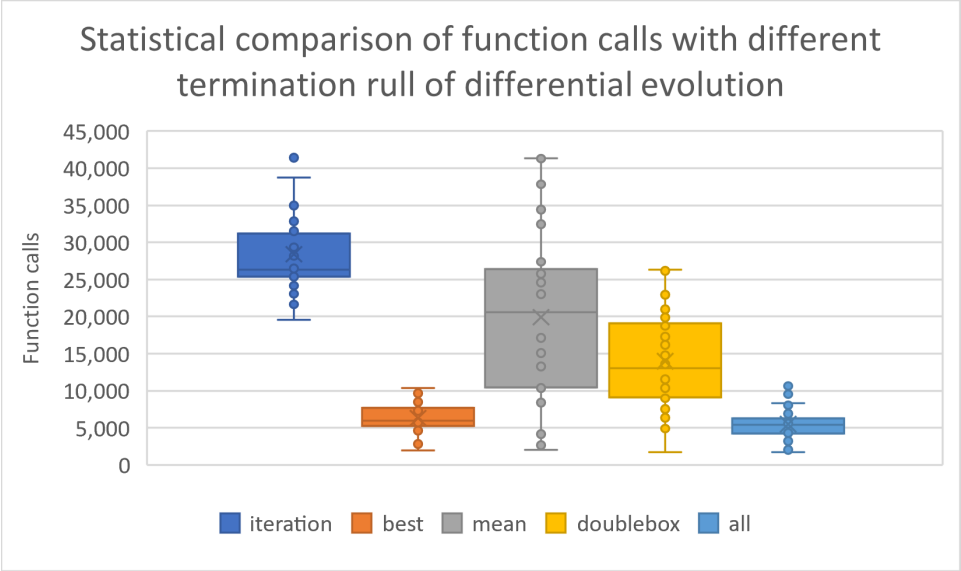

Figure 2. Statistical comparison of function calls with different termination rull of differential evolution

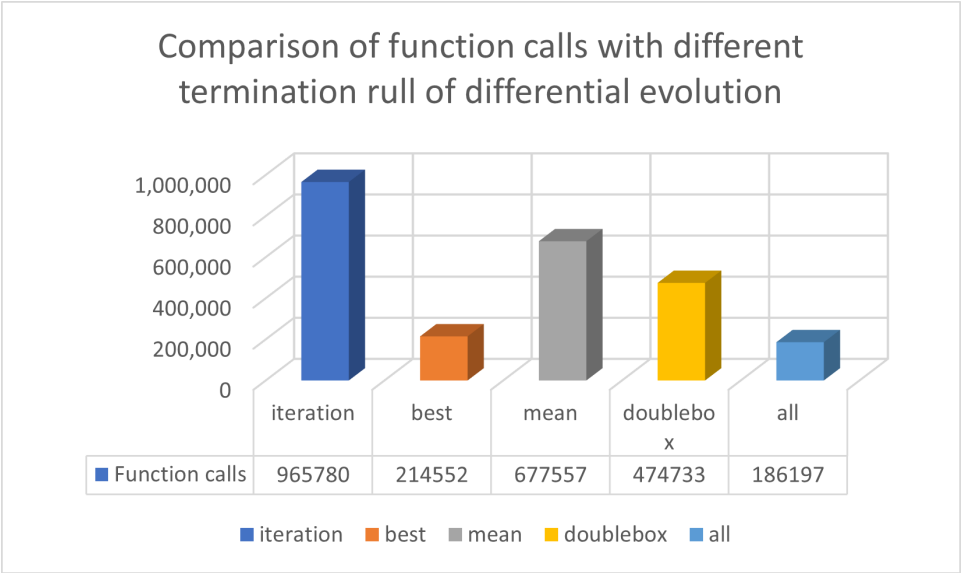

Figure 3. Comparison of function calls with different termination rull of differential evolution

Table 3. Particle swarm optimization: Function calls with different termination rulls

PROBLEMS	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	33113	3418	3127	3122	3005
BF2	32804	3265	2995	2919	2889
BRANIN	27762	2601	2549	2431	2417
CAMEL	28511	2801	2675	2547	2547
CIGAR10	39038	3987	3674	3638	3638
CM4	33051	3863	3299	3144	3144
DISCUS10	29548	2611	2409	2405	2405
EASOM	26251	2420	2232	2232	2232
ELP10	1907	1705	1741	1586	1586
EXP4	28364	2751	2558	2558	2558
EXP16	28536	2898	2676	2676	2676
GKLS250	27495	2796	2553	2422	2422
GKLS350	28350	2947	2645	3833	2558
GRIEWANK2	32596	4405	2839	4282	2820
GRIEWANK10	37070	5327	4561	4797	4248
POTENTIAL3	32890	3289	3231	3222	3170
PONTENTIAL5	50595	4926	4881	7282	4730
PONTENTIAL6	62076	6699	5537	7534	5077
PONTENTIAL10	78807	10582	6737	29865	6539
HANSEN	29527	3788	2687	3062	2587
HARTMAN3	29673	2807	2743	2550	2550
HARTMAN6	31739	3081	2915	2809	2809
RASTRIGIN	32717	3558	2990	2829	2829
ROSENBROCK8	33115	4279	4263	4036	3969
ROSENBROCK16	35519	5761	5432	5170	5170
SHEKEL5	29461	3132	2823	12476	2816
SHEKEL7	30003	3064	2855	11520	2856
SHEKEL10	30043	3128	2942	14786	2866
SINU4	27786	3047	2740	2673	2657
SINU8	28129	3177	2824	4113	2828
TEST2N4	28950	3037	2839	2681	2681
TEST2N9	30368	4188	3106	3105	3067
TEST30N3	27643	2943	3130	2762	2762
TEST30N4	27836	3202	3185	2915	2915
TOTAL	**1111273**	**125483**	**110393**	**169982**	**106023**

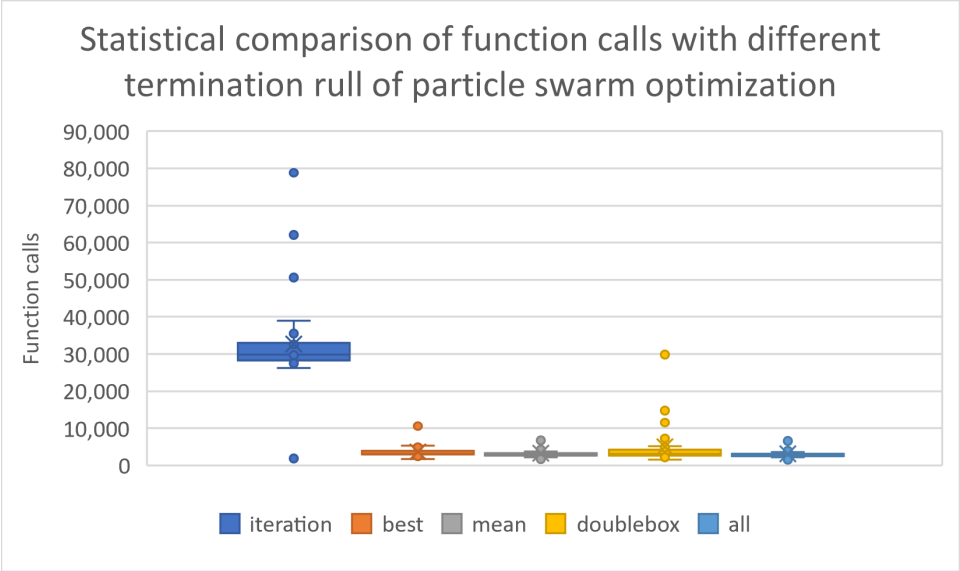

Figure 4. Statistical comparison of function calls with different termination rull of particle swarm optimization

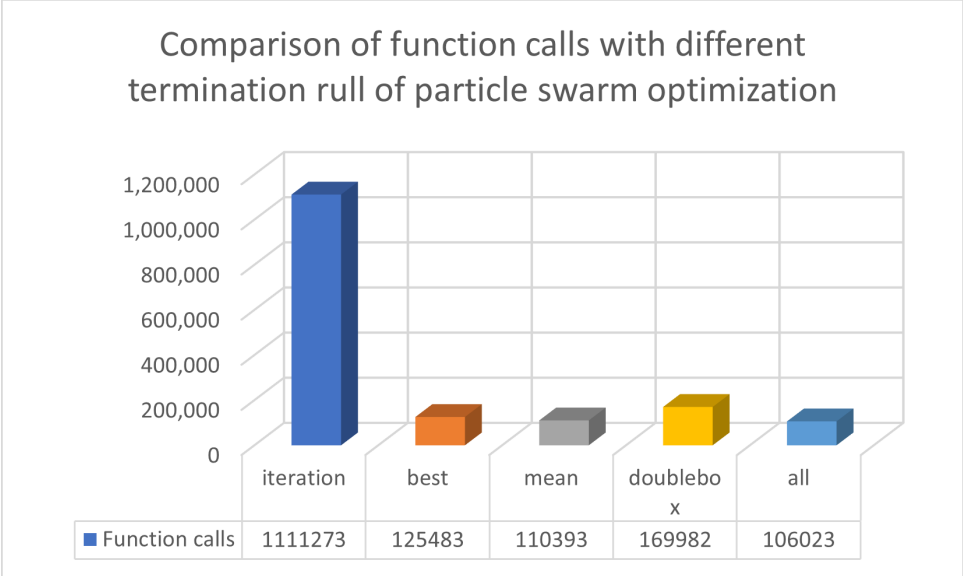

Figure 5. Comparison of function calls with different termination rull of particle swarm optimization

Table 4. Multistart: Function calls with different termination rulls

PROBLEMS	ITERATION	BEST	MEAN	DOUBLEBOX	ALL
BF1	479374	53873	479374	50762	50762
BF2	276214	37670	239220	35711	35711
BRANIN	107283	11063	11046	10521	10521
CAMEL	163508	16083	153559	15255	15255
CIGAR10	59097	14937	14937	14697	14697
CM4	726033	62236	62192	58581	58581
DISCUS10	50456	6296	6296	6056	6056
EASOM	55130	5412	55130	51132	5412
ELP10	58917	14757	14981	14517	14517
EXP4	54174	10022	54174	9774	9774
EXP16	54680	10528	54680	10280	10280
GKLS250	58547	6202	47948	5908	5908
GKLS350	81988	7560	42145	7087	7087
GRIEWANK2	188429	19179	188429	17877	17877
GRIEWANK10	620223	65206	567153	61818	61818
POTENTIAL3	126632	17773	126632	17161	17161
PONTENTIAL5	226873	31347	226873	30249	30249
PONTENTIAL6	247144	35457	247144	33883	33883
PONTENTIAL10	283230	44957	283230	43618	43618
HANSEN	201543	18568	201543	17541	17541
HARTMAN3	162934	17395	162934	16562	16562
HARTMAN6	179073	22010	179073	21015	21015
RASTRIGIN	275610	24696	275610	23015	23015
ROSENBROCK8	62010	17850	17850	17610	17610
ROSENBROCK16	69460	25300	25300	25060	25060
SHEKEL5	151648	17725	151648	16972	16972
SHEKEL7	154373	17894	154373	17127	17127
SHEKEL10	159825	17931	159825	17135	17135
SINU4	155634	16429	155634	15647	15647
SINU8	170167	19519	170167	18674	18674
TEST2N4	154092	16486	154092	15806	15806
TEST2N9	160651	22035	160651	20716	20716
TEST30N3	161628	16934	161628	16145	16145
TEST30N4	159753	16686	159753	15907	15907
TOTAL	**6296333**	**758016**	**5165224**	**769819**	**724099**

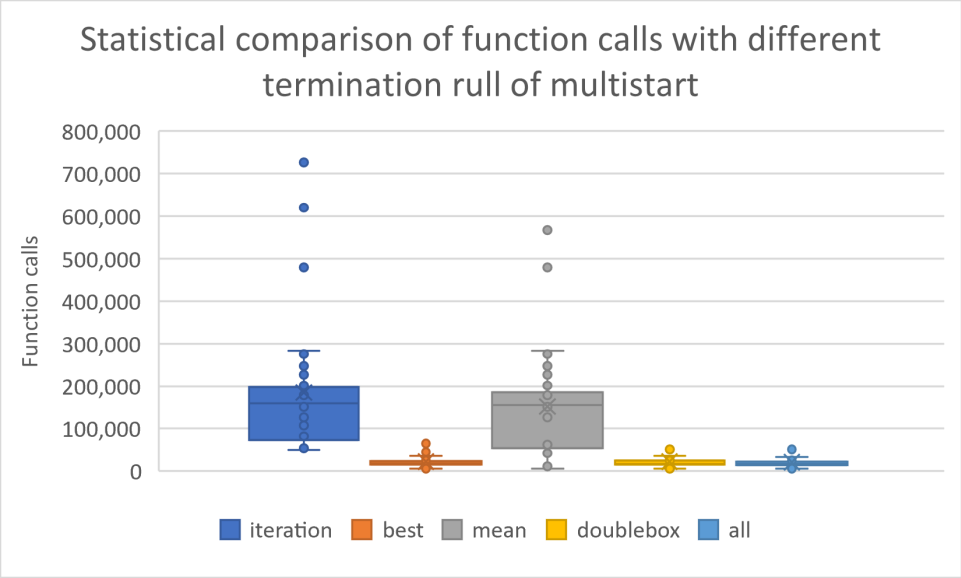

Figure 6. Statistical comparison of function calls with different termination rull of multistart

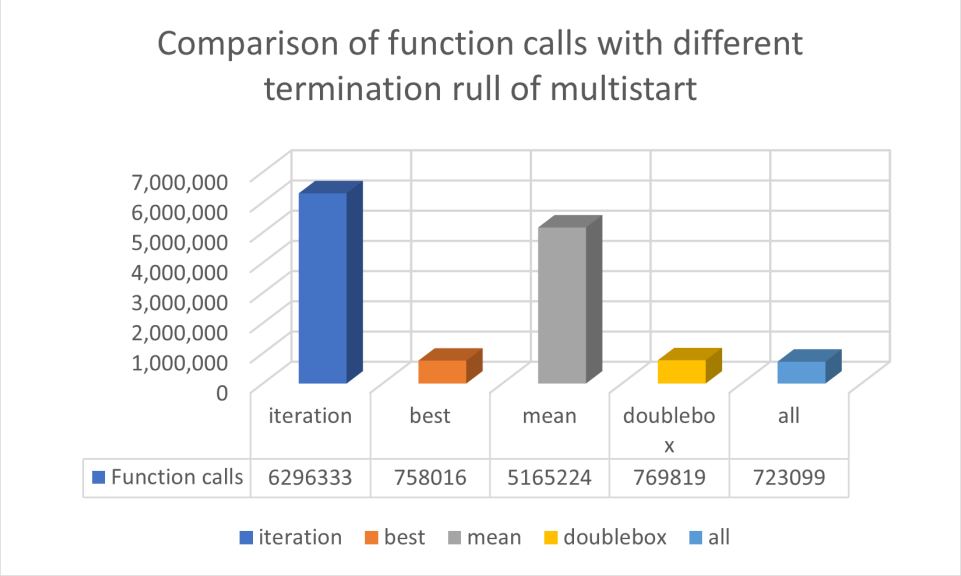

Figure 7. Comparison of function calls with different termination rull of multistartMultistart_comp

Table 5. Function calls with different stochastic methods

PROBLEMS	DE	PSO	MULT	DE + PSO	DE + MULT	PSO + MULT	DE + PSO + MULT
BF1	6114	3005	50762	3363	27918	28098	19862
BF2	7560	2889	35711	2991	20110	19985	14475
BRANIN	4289	2417	10521	2092	6399	6534	4980
CAMEL	5940	2547	15255	2384	8998	9098	6796
CIGAR10	3111	3638	14697	3326	8945	9143	7037
CM4	4079	3144	58581	2465	32360	32638	22402
DISCUS10	2034	2405	6056	2140	4062	4243	3473
EASOM	1721	2232	51132	1895	3463	3680	3004
ELP10	6031	1586	14517	2786	8617	8157	6414
EXP4	5040	2558	9774	2344	6094	6492	4837
EXP16	5339	2676	10280	2415	6319	4205	5038
GKLS250	2641	2422	5908	2246	4058	5204	3500
GKLS350	3206	2558	7087	2533	5015	12831	4100
GRIEWANK2	6915	2820	17877	3447	13003	34592	9050
GRIEWANK 10	9539	4248	61818	3984	34320	34591	23912
POTENTIAL 3	5256	3170	17161	2796	9881	10228	7535
PONTENTIAL 5	7742	4730	30249	4091	17279	17859	12907
PONTENTIAL6	8197	5077	33883	4675	19201	20166	14454
PONTENTIAL 10	10643	6539	43618	7321	24695	25680	18822
HANSEN	4263	2587	17541	3208	10295	10492	8916
HARTMAN3	4566	2550	16562	2222	9511	9745	7170
HARTMAN6	5550	2809	21015	2757	11958	12231	8816
RASTRIGIN	5999	2829	23015	2870	17398	12940	11410
ROSENBROCK 8	8051	3969	17610	3914	10474	10829	8169
ROSENBROCK16	8307	5170	25060	3133	14754	15118	11505
SHEKEL 5	5933	2816	16972	3092	9922	10149	7517
SHEKEL 7	5677	2856	17127	2995	10110	10219	7491
SHEKEL 10	5100	2866	17135	2514	10096	10219	7583
SINU4	4776	2657	15647	2514	9132	9404	6880
SINU8	4980	2828	18674	2714	10797	11097	8130
TEST2N4	5727	2681	15806	2482	9179	9343	6863
TEST2N9	6288	3067	20716	4055	12138	12084	9505
TEST30N3	2869	2762	16145	2473	9485	9340	6807
TEST30N4	2714	2915	15907	2534	9312	9758	7082
TOTAL	**186197**	**106023**	**769819**	**102771**	**425298**	**456392**	**316442**

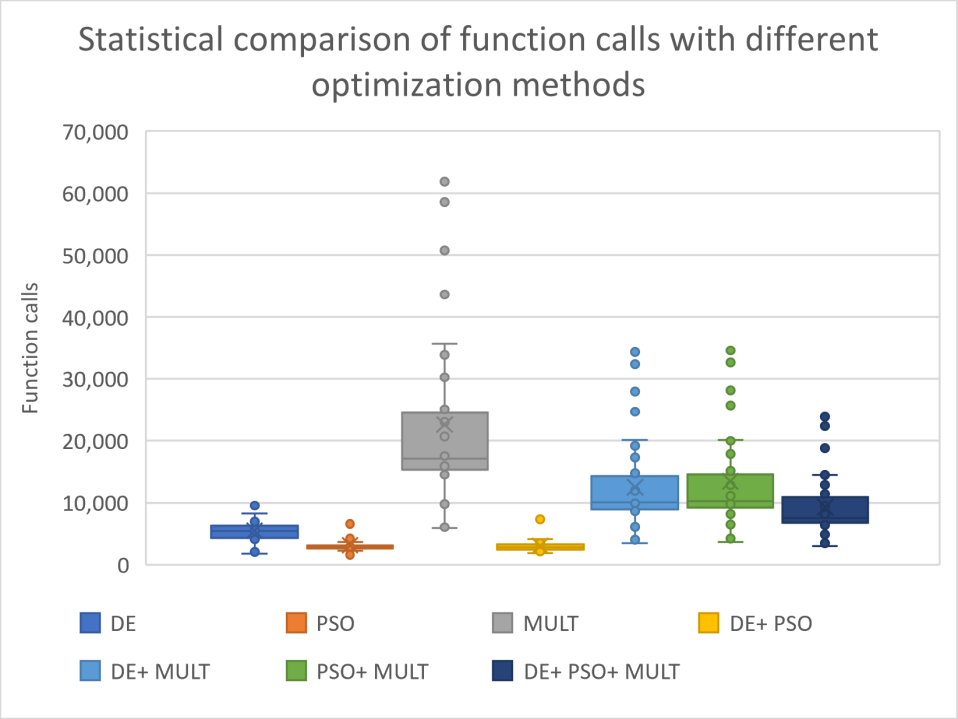

Figure 8. Statistical comparison of function calls with different optimization methods

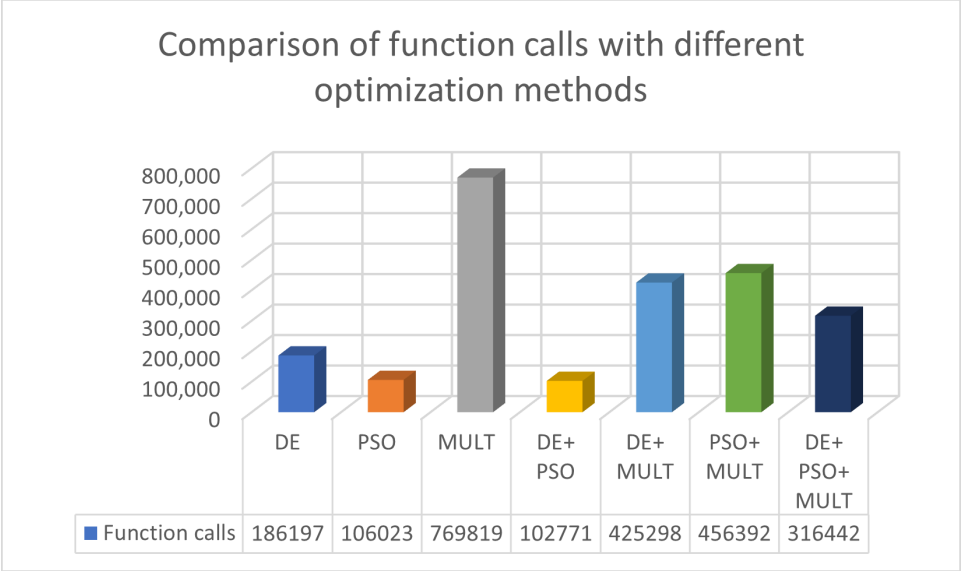

Figure 9. Comparison of function calls with different optimization methods

During the execution of the proposed algorithm 4, the calls to the objective function serve as the criterion for the computational power consumed by each PU. Decreasing the computational consumption entails a faster completion of the global optimization process. The same holds true when only one PU is active, meaning optimization is performed by a single method, as is the case in Tables 2,3 , and 4. In Table 2, we observe the performance of all termination rules for the DE method. From these, both statistical and quantitative comparisons arise, depicted in Figures 2 and 3, respectively. Table 3 analyzes the termination rules for PSO, with the corresponding statistical and quantitative comparison figures, Figure 4 and Figure 5. Following is Table 4, where the termination rules for the Multistart method are examined, along with the respective figures of statistical and quantitative comparison, Figure 6 and Figure 7. Observing the tables with the corresponding statistical

and quantitative comparisons, it is evident that the calls to the objective function for the proposed termination rule 4.4 are fewer than any other rule in any method.

In Table 5, we observe the performance of the three methods individually, as well as their combinations with parallelization. The termination rule used is the proposed 4.4 termination rule. Figures 8 and 9 are derived from Table 5 and demonstrate the statistical and quantitative comparison of objective function calls. From the above, it emerges that the collaboration-parallelization of the two methods, DE and PSO, yields better results than when they optimize individually. Furthermore, it is evident that any parallelization carried out with methods consuming significant computational power, such as Multistart, significantly reduces the objective function calls.

6. Conclusions

Our experimental observations confirm that as the number of methods involved in global optimization increases, optimal solutions are found more quickly, saving computational power. This is because the nature of each method can match the nature of the problem. Additionally, we observe the impact of the proposed termination rule, which enhances both aspects. Specifically, this combined termination rule is essential for parallelizing multiple optimization methods, as the process is terminated by any termination rule and significantly improves the overall algorithm's performance. However, further research could explore other combinations of optimization methods or termination rules to further enhance performance.

Author Contributions: V. Charilogis played a pivotal role in shaping the idea and methodology, overseeing the technical aspects related to the software, and contributing to manuscript preparation. He conducted experimental studies using various datasets, performed statistical analysis, and collaborated with all authors in manuscript preparation. It is noted his significant involvement in executing experimental studies, conducting statistical analysis, as well as collaborating with the authors during manuscript preparation. Finally, all authors reviewed and endorsed the final version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Not applicable.

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

References

1. Kawachi, M., Ando, N. (1989). Genetic Algorithms in Search, Optimization & Machine Learning. Artificial Intelligence, Vol.: 7, Issue: 1, pp.: 168. Doi: https://doi.org/10.11517/jjsai.7.1_168
2. Kincaid, D., Cheney, W. (2002). Numerical analysis: mathematics of scientific computing. The University of Texas at Austin: Publishing Pacific Grove, California. ISBN: 0-534-13014-3. Online: https://wp.kntu.ac.ir/ghoreishif/Cheney%20-%20Numerical%20Analysi.pdf
3. Kantorovich, L.V. (1960). Mathematical Methods of Organizing and Planning Production. Informs PubsOnline: Management Science . Doi: https://doi.org/10.1287/mnsc.6.4.366
4. Dantzig, G.B (1947). Maximization of a linear function of variables subject to linear inequalities, T.C. Koopmans (ed.): Activity Analysis of Production and Allocation. New York-London 1951 (Wiley & Chapman-Hall), pp.: 339-347.
5. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P. (1983). Optimization by simulated annealing. Science, Vol. 220, Issue: 4598, pp.: 671–680. DOI: 10.1126/science.220.4598.671 [CrossRef] [PubMed]
6. Hashim, F.A., Houssein, E.H., Mabrouk, M.S., Al-Atabany,W., Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics based algorithm. Future Generation Computer Systems: Vol.: 101, pp.: 646–667. Doi: https: 10.1016/j.future.2019.07.015
7. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S. (2009). GSA: A gravitational search algorithm. Information Sciences: Vol.: 179, pp.:2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
8. Du, H., Wu, X., Zhuang, J. (2006). Small-World Optimization Algorithm for Function Optimization. In Proceedings of the International Conference on Natural Computation, Xi'an, China, 24–28 September 2006, pp. 264–273. Doi: 10.1007/11881223_33
9. Goldberg, D.E., Holland, J.H. (1988). Genetic Algorithms and Machine Learning. Machine Learning (3): pp.: 95–99. Doi: 10.1023/A:1022602019183

10. Das, S. , Suganthan, P.N. (2011). Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation, Vol.: 15, Issue:1, pp.: 4–31. Doi: 10.1109/TEVC.2010.2059031
11. V. Charilogis , I.G. Tsoulos, A. Tzallas, E. Karvounis, Modifications for the Differential Evolution Algorithm, Symmetry **14**, 447, 2022.
12. Fogel, L.J., Owens, A.J., Walsh, M.J. (1996). Artificial Intelligence through Simulated Evolution. Publisher: John Wiley & Sons: Hoboken, NJ, USA. ISBN-10: 0471265160.
13. Koza, J.R. (1994). Genetic programming as a means for programming computers by natural selection. Statistics and Computing, Vol.: 4, pp.: 87–112. Doi: 10.1007/BF00175355
14. Kennedy, J., Eberhart, R. (1995). Particle Swarm Optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth,WA, Australia, 27 November–1 December 1995, Vol.: 4, pp. 1942–1948. Doi: 10.1109/ICNN.1995.488968
15. V. Charilogis, I.G. Tsoulos, Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions, Information **13**, 217, 2022.
16. Dorigo, M., Maniezzo, V., Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol.: 26, pp.: 29–41. Doi: 10.1109/3477.484436
17. Yang, X.S., Gandomi, A.H. (2012). Bat algorithm: A novel approach for global engineering optimization. Engineering Computations, Vol.: 29, pp.: 464–483. Doi: https://doi.org/10.1108/02644401211235834
18. Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Softwar, Vol.: 95, pp.: 51–67. Doi: https://doi.org/10.1016/j.advengsoft.2016.01.008
19. Saremi, S., Mirjalili, S., Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. Advances in Engineering Software, Vol.: 105, pp.: 30–47.Doi: https://doi.org/10.1016/j.advengsoft.2017.01.004.
20. Rao, R.V., Savsani, V.J., Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design, Vol.: 43, pp.: 303–315. Doi: https://doi.org/10.1016/j.cad.2010.12.015
21. Eita, M., Fahmy, M. (2014). Group counseling optimization. Applied Soft Computing , Vol.:22, pp.: 585–604. Doi: https://doi.org/10.1016/j.asoc.2014.03.043
22. Matoušová, I., Trojovský, P., Dehghani, M., Trojovská, E., Kostra, J. (2013). Mother optimization algorithm: A new human-based metaheuristic approach for solving engineering optimization. Scientific Reports, Vol.: 13, Article Number: 10312. Doi: s41598-023-37537-8
23. Montazeri, Z., Niknam, T., Aghaei, J., Malik, O.P., Dehghani, M., Dhiman, G. (2023). Golf Optimization Algorithm: A New Game-Based Metaheuristic Algorithm and Its Application to Energy Commitment Problem Considering Resilience. Biomimetics, Vol.: 5, 386. Doi: https://doi.org/10.3390/biomimetics8050386
24. Doumari, S.A., Givi, H., Dehghani, M., Malik, O.P. (2021). Ring Toss Game-Based Optimization Algorithm for Solving Various Optimization Problems. International Journal of Intelligent Engineering Systems, Vol.: 14, pp.: 545–554. Doi: 10.22266/ijies2021.0630.46
25. Moghdani, R., Salimifard, K.(2018). Volleyball premier league algorithm. Applied Soft Computing, Vol.: 64, pp.: 161–185. Doi: https://doi.org/10.1016/j.asoc.2017.11.043
26. Gu, R., Fan, S., Hu, Q., Yuan, C. & Huang, Y. (2018). Parallelizing Machine Learning Optimization Algorithms on Distributed Data-Parallel Platforms with Parameter Server. IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). DOI: 10.1109/PADSW.2018.8644533.
27. Low, Y., Gonzalez, J., Kyrola, A., Bickson,D., Guestrin, C., Hellerstein, J. (2010). GraphLab: A New Framework For Parallel Machine Learning. Source: arXiv, Computer Science.
28. Yangyang, L., Liu, G., Lu, G., Jiao, L., Marturi, N. & Shang, R. (2019). Hyper-Parameter Optimization Using MARS Surrogate for Machine-Learning Algorithms. IEEE Transactions on Emerging Topics in Computational Intelligence, pp(99):1-11. DOI: 10.1109/TETCI.2019.2918509.
29. Yamashiro, H. & Nonaka, H. (2021). Estimation of processing time using machine learning and real factory data for optimization of parallel machine scheduling problem. ScienceDirect: Operations Research Perspectives, Vol.:8, 2021, Doi: https://doi.org/10.1016/j.orp.2021.100196 .
30. Kim, H.S. & Tsai, L. (2003). Design Optimization of a Cartesian Parallel Manipulator. Journal of Mechanical Design,125(1):43-51. Doi: https://doi.org/10.1115/1.1543977 .
31. Oh, S., Jong Jang,H. & Pedrycz, W. (2009). The design of a fuzzy cascade controller for ball and beam system: A study in optimization with the use of parallel genetic algorithms. ScienceDirect: Engineering Applications of Artificial Intelligence, 22(2):261-271. Doi: https://doi.org/10.1016/j.engappai.2008.07.003 .
32. Fatehi, M., Toloei, A., Zio, E., Niaki, S.T.A. & Kesh,B. (2023). Robust optimization of the design of monopropellant propulsion control systems using an advanced teaching-learning-based optimization method. ScienceDirect: Engineering Applications of Artificial Intelligence, Vol.:126, Part: A.
33. Cai, J., Yang, H., Lai, T. & Xu, K.(2023). Parallel pump and chiller system optimization method for minimizing energy consumption based on a novel multi-objective gorilla troops optimizer. ScienceDirect: Journal Of Building Engineering, Vol.: 76, Doi: https://doi.org/10.1016/j.jobe.2023.107366
34. Yu, Y. & Shahabi, L. (2023). Engineering Application of Artificial Intelligence: Optimal infrastructure in microgrids with diverse uncertainties based on demand response, renewable energy sources and two-stage parallel optimization algorithm. ScienceDirect. Vol.:123, Part b. Doi: https://doi.org/10.1016/j.engappai.2023.106233

35. Ramirez-Gil, F.J., Pere-Madrid, C.M., Nelli Silva, E.C. & Montealerge-Rubio, W. (2021). Sustainable Computing: Informatics and Systems: Parallel computing for the topology optimization method: Performance metrics and energy consumption analysis in multiphysics problems. Vol.:30, Doi: https://doi.org/10.1016/j.suscom.2020.100481
36. Tavakolan, M., Mostafazadeh, F., Eirdmousa, S.J., Safari, A. & Mirzai, K. (2022). A parallel computing simulation-based multi-objective optimization framework for economic analysis of building energy retrofit: A case study in Iran. ScienceDirect: Journal of Building Engineering, Vol.: 45, Doi: https://doi.org/10.1016/j.jobe.2021.103485
37. Lin, G. (2020). Parallel optimization n based operational planning to enhance the resilience of large-scale power systems. Mississippi State University, Scholars Junction. Online: https://scholarsjunction.msstate.edu/cgi/viewcontent.cgi?article=4435&context=td
38. Pang, M. & Shoemaker, C.A. (2023). Comparison of parallel optimization algorithms on computationally expensive groundwater remediation designs. Sience of the Total Environment: 857(3), Doi: https://doi.org/10.1016/j.scitotenv.2022.159544
39. Ezugwu, A. (2023). A general Framework for Utilizing Metaheuristic Optimization for Sustainable Unrelated Parallel Machine Scheduling: A concise overview. Arxiv, Computer Science, Neural and Evolutionary Computing. Doi: https://doi.org/10.48550/arXiv.2311.12802
40. Censor, Y. & Zenios, S. (1997). Parallel Optimization: Theory, Algorithms and Applications. Publisher: Oxford University Press, USAISBN: ISBN-13: 978-0195100624. DOI: 10.1093/oso/9780195100624.001.0001
41. Marini, F. & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and Intelligent Laboratory Systems. Volume 149, Part B, p.:153-165. Doi: https://doi.org/10.1016/j.chemolab.2015.08.020
42. García-Gonzalo, E. & Fernández-Martínez, J. L. (2012). A Brief Historical Review of Particle Swarm Optimization (PSO). Journal of Bioinformatics and Intelligent Control, Volume 1, Number 1, June 2012, pp. 3-16(14). American Scientific Publishers. DOI:https://doi.org/10.1166/jbic.2012.1002
43. Jain, M., Saihjpal, V., Singh, N. & Singh, S.B. (2022). An Overview of Variants and Advancements of PSO Algorithm. MDPI, Applied Sciences: 12, 8392. Doi:https://doi.org/10.3390/ app12178392 .
44. Marti, R., Resende, M.G.C. & Ribeiro, C. (2013). Multi-start methods for combinatorial optimization. European Journal of Operational Research Volume 226, Issue 1, 1 April 2013, Pages 1-8. Doi: https://doi.org/10.1016/j.ejor.2012.10.012
45. Marti, R., Moreno-Vega, J. & Duarte, A. (2010). Advanced Multi-start Methods. Handbook of Metaheuristics, pp: 265–281.
46. Tu, W. & Mayne, R.W. (2002). Studies of multi-start clustering for global optimization. International Journal for Numerical Methods in Engineering. Doi: https://doi.org/10.1002/nme.400.
47. Feoktistov, V. (2006). Differential Evolution. In Search of Solutions. Optimization and Its Appl;ications, Springer. Doi: https://doi.org/10.1007/978-0-387-36896-2
48. Tsoulos, I.G. Modifications of real code genetic algorithm for global optimization. Appl. Math. Comput. 2008, 203, 598–607.
49. Bilal, M.P., 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 (2020) 103479. Doi: https://doi.org/10.1016/j.engappai.2020.103479
50. 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.
51. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
52. 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.
53. Charilogis, V.; Tsoulos, I.G. A Parallel Implementation of the Differential Evolution Method. Analytics 2023, 2, 17–30.
54. Charilogis, V.; Tsoulos, I.G.; Tzallas, A. (2023). An Improved Parallel Particle Swarm Optimization. SN Computer Science (2023) 4:766
55. 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.
56. 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.
57. Gao, Z.M., Zhao, J., Hu,Y.R., Chen, H.F. (2021).The Challenge for the Nature - Inspired Global Optimization Algorithms: Non-Symmetric Benchmark Functions. IEEE Access, July 26, 2021.
58. R. Chandra, L. Dagum, D. Kohr, D. Maydan,J. McDonald and R. Menon, Parallel Programming in OpenMP, Morgan Kaufmann Publishers Inc., 2001.