The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

AN ALTERNATIVE APPROACH FOR DIFFERENTIAL EVOLUTION ALGORITHM

MOHAMED SAAD [1], HEGAZY ZAHER [2], NAGLAA RAGAA [3] and HEBA SAYED [4]

[1,3,4] Department of Operations Research and Management, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza, Egypt.
[2] Department of Mathematical Statistics, Faculty of Graduate Studies for Statistical Research, Cairo University, Giza, Egypt.

Abstract

The differential evolution algorithm is an effective method for global numerical optimization that is straightforward to understand, easy to implement, reliable, and efficient. As one of the evolutionary algorithms, Differential Evolution addresses global optimization problems by iteratively refining candidate solutions through an evolutionary process. It serves as a heuristic technique for minimizing potentially nonlinear and non-differentiable continuous functions. Global minimization algorithms require efficient computational time, which motivates the use of parallel computational approaches. This new technique utilizes multiple independent parallel computing units that occasionally share the best solutions they have discovered. This study proposes an improved approach designed to minimize the number of function calls, thereby strengthening the method's efficiency in exploring the objective function's search space. The proposed parallelizing Differential Evolution algorithm has been tested on several of optimization problems, and it appears from the experimental results that the executing time of the proposed parallelizing Differential Evolution becomes faster compared with Classic Differential Evolution and the other previous modifications of Differential Evolution. Also, results indicate significant improvements in function evaluations and solution quality compared to existing methodologies, demonstrating the effectiveness of the proposed approach.

Keywords: Global Optimization, Metaheuristics, Parallel Computing, Differential Evolution Algorithm.

1. INTRODUCTION

The assignment of locating the global minimum of a continuous and differentiable function

$f: S \to R, S \subset R^n$ is expressed as:

$$x^* = \arg \min_{x \epsilon S} f(x) \qquad (1)$$

Where the set S is defined as:

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

The real-world problems can be modelled as global optimization problems, such as problems from physics [1–3], chemistry [4–6], economics [7, 8], medicine [9, 10], etc. Recently, different algorithms have been proposed to tackle the problem of Equation (1), such as simulated annealing methods [11–13], differential evolution (DE) methods [14], particle swarm optimization methods [15], ant colony optimization [16], genetic algorithms [17, 18], etc. The DE algorithm firstly generates a population of candidate solutions, which iteratively evolves through the crossover stage in order to determine the

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

global minimum of the objective function. The method has been applied across various research fields, including electromagnetics [19], energy consumption problems [20], job shop scheduling [21], and image segmentation [22]. Differential Evolution has evolved into a reliable and versatile optimization tool that is used easily. The initial publication on DE appeared as a technical report in 1995 (Storn and Price 1995). Since then, DE has demonstrated its effectiveness in competitions, such as the IEEE's International Contest on Evolutionary Optimization (ICEO) in 1996 and 1997, and in a wide range of real-world applications. DE's performance can be enhanced, and its methodology adapted to various optimization scenarios. It operates on a population of candidate solutions, referred to as individuals, and iteratively refines them toward the optimal solution. This algorithm makes minimal assumptions about the optimization problems, allowing for rapid exploration of large design spaces [23]. DE is considered one of the most versatile and robust population-based search algorithms, effectively handling multi-modal problems [24].

In this paper, a modified parallel DE presents an approach that requires fewer total evaluations to reach the optimum value, achieving convergence with minimal external iterations of the algorithm while successfully identifying the global minimum in several cases. The paper is structured as follows: the second section provides a literature review of DE and the proposed parallel DE, highlighting challenges related to parallelization and current strategies to address them. The third section describes the experimental test functions, while the fourth section discusses numerical experiments that optimize the parameters of the proposed parallel DE, along with the associated experimental results. Finally, the fifth section concludes the paper and offers suggestions for future research directions.

2. METHOD DESCRIPTION

This section provides a comprehensive overview of the base Differential Evolution algorithm, followed by an explanation of the proposed parallel Differential Evolution approach. Starts by generating a population of candidate solutions, followed by a stochastic search process through mutation which are iteratively refined through a crossover process to identify the global minimum of the objective function. The proposed method divides the processing into independent units, such as threads, allowing each to operate autonomously. Additionally, it outlines a communication framework for coordinating the various components in parallel processing and presents a termination technique specifically adapted for this parallel context.

The Base Differential Evolution Algorithm

Differential Evolution has demonstrated strong performance across a range of optimization problems in various scientific disciplines. As a stochastic population-based evolutionary method, DE operates using a population of candidate solutions and employs a stochastic search process through mutation, crossover, and selection operators, guiding the population toward improved solutions in the design space [24]. The foundational algorithm was first introduced by Storn [14] and has since undergone numerous modifications in various research studies. Examples include the compact differential evolution algorithm [25, 26], a self-adaptive DE

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

[27] that iteratively adjusts its parameters, and fuzzy logic modifications [28]. Additionally, a numerical study on various modifications of the DE method is presented by Kaelo et al. [29].

Description of the New Proposed Parallel DE

Implementing a parallel DE algorithm can produce weighty benefits in terms of computational efficiency and speedup, taking advantage of the power of parallel processing by executing several DE iterations concurrently, getting faster convergence, and minimizing execution time. Parallel implementations have been well beneficial for computationally intensive problems, and large-scale optimization scenarios consume the time for objective function evaluation. There are three challenges to confirming the efficient execution of parallelizing DE. The first vital challenge is the balance between exploring the search space and exploiting the better solutions, the second challenge is load balancing, which includes distributing the computational load uniformly across processing units to maximize efficiency, and the third one is communication overhead and synchronization between parallel processes. To overcome these challenges, it can be useful to use these approaches. first one, a master-slave architecture where a master process manages the overall execution and distributes subtasks to slave processes, second one, dividing the population into subgroups and processing them independently in parallel that can be combined with migration strategies, where individuals are exchanged between subgroups periodically to maintain diversity, third one, hybrid approaches that combine parallel DE with other optimization techniques [30].

Propagation Mechanism

In the proposed Parallel Differential Evolution (PDE) method, the algorithm uses a (1-to-1) propagation mechanism. Generally, a random island will send its best value to another randomly selected island. i.e., the best values of the islands are spread to the rest by replacing their worst values, this technique is shown in Figure 1.

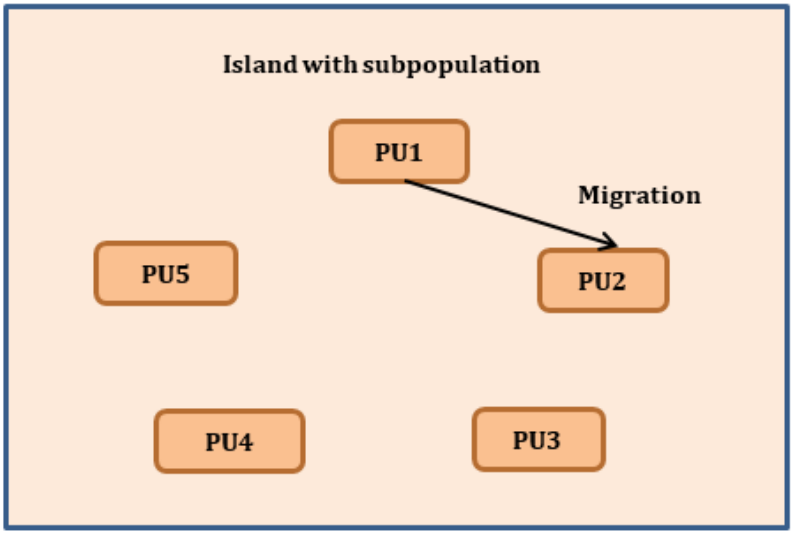


Figure 1: (1-to-1) Propagation technique

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

The Termination Rule

In the base Differential Evolution algorithm, termination occurs after reaching a predefined number of iterations, which can sometimes result in premature halting before the global minimum is identified. In contrast, the proposed PDE method employs a termination criterion that evaluates the convergence of each island separately. Specifically, the difference is calculated as follows:

$$\delta_i^{(k)} = \left| f_{i,min}^{(k)} - f_{i,min}^{(k-1)} \right|, \qquad (2)$$

In this equation, $f_{i,min}^{(k)}$ represents the best function value found for island i iteration k. If $\delta_i^{(k)} \leq \epsilon$ for a specified number of iterations, then the population evolution for the island is terminated. Furthermore, in the proposed PDE method, if this condition holds for more than one island, the entire algorithm will terminate. The proposed PDE algorithm is listed below with the proposed terminating mechanism and mechanism for communication between the islands.

The steps of the proposed PDE algorithm

1. **INPUT**:
   (a) The parameters **NP**, **CR**, **F**
   (b) The integer parameter**N**, which stands for the number of islands.
   (c) The integer parameter$\boldsymbol{N_R}$, which represents the propagation rate.
   (d) The integer parameter $\boldsymbol{N_1}$, which represents the number of islands that should Terminate in order to terminate the whole process.
2. **OUTPUT**:
   (a) The agent $\boldsymbol{x_{best}}$with the lower function value$\boldsymbol{f(x_{best})}$.
3. **Initialize** all agents in**S**.
4. Set iter = 1
5. For $\boldsymbol{i = 1, \ldots, N}$ do in Parallel, Perform for every island i as follows:
   i. **Set** $\boldsymbol{x}$ as the agent $\boldsymbol{i}$.
   ii. **Pick** randomly three agents , $\boldsymbol{b, c}$.
   iii. **Pick** a random index $\boldsymbol{R} \in \{1, \ldots, n\}$.
   iv. **Compute** the trial vector $\boldsymbol{y} = [y_1, y_2, \ldots y_n]$ with the following procedure
   v. For $\boldsymbol{j = 1, \ldots, n}$ do.
      A. **Set** $\boldsymbol{r_i} \in [0,1]$ a random number.
      B. **If** $\boldsymbol{r_j} <$ CR **or** $\boldsymbol{j = R}$ **then** $\boldsymbol{y_j = a_j + F \times (b_j - c_j)}$ **else** $\boldsymbol{y_j = x_j}$.
   vi. If $\boldsymbol{y \in S}$ AND $\boldsymbol{f(y) \leq f(x)}$ then $\boldsymbol{x = y}$.
   vii. **EndFor.**
6. **EndFor**
7. **If** iter mod $\boldsymbol{N_R = 0}$,
   Apply the propagation scheme of **Section (Propagation Mechanism)** to the islands.
8. **Set** iter = iter + 1
9. **If** the termination rule of **Section (The Termination Rule)** is not valid, go to 5.
10. **Apply** local search procedure to $\boldsymbol{x_{best}}$. The local search procedure used in the proposed method is the BFGS variant of Powell [31].

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

In the proposed method, the parallel island model depicted in Figure 2, the population of agents is divided into N independent segments known as islands. This approach is a well-established variant in parallel genetic algorithms [32, 33]. For example, if there are 10 agents distributed across 2 islands, agents 1–5 will be assigned to island 1, while agents 6–10 will belong to island 2. Each island performs the differential evolution process independently.

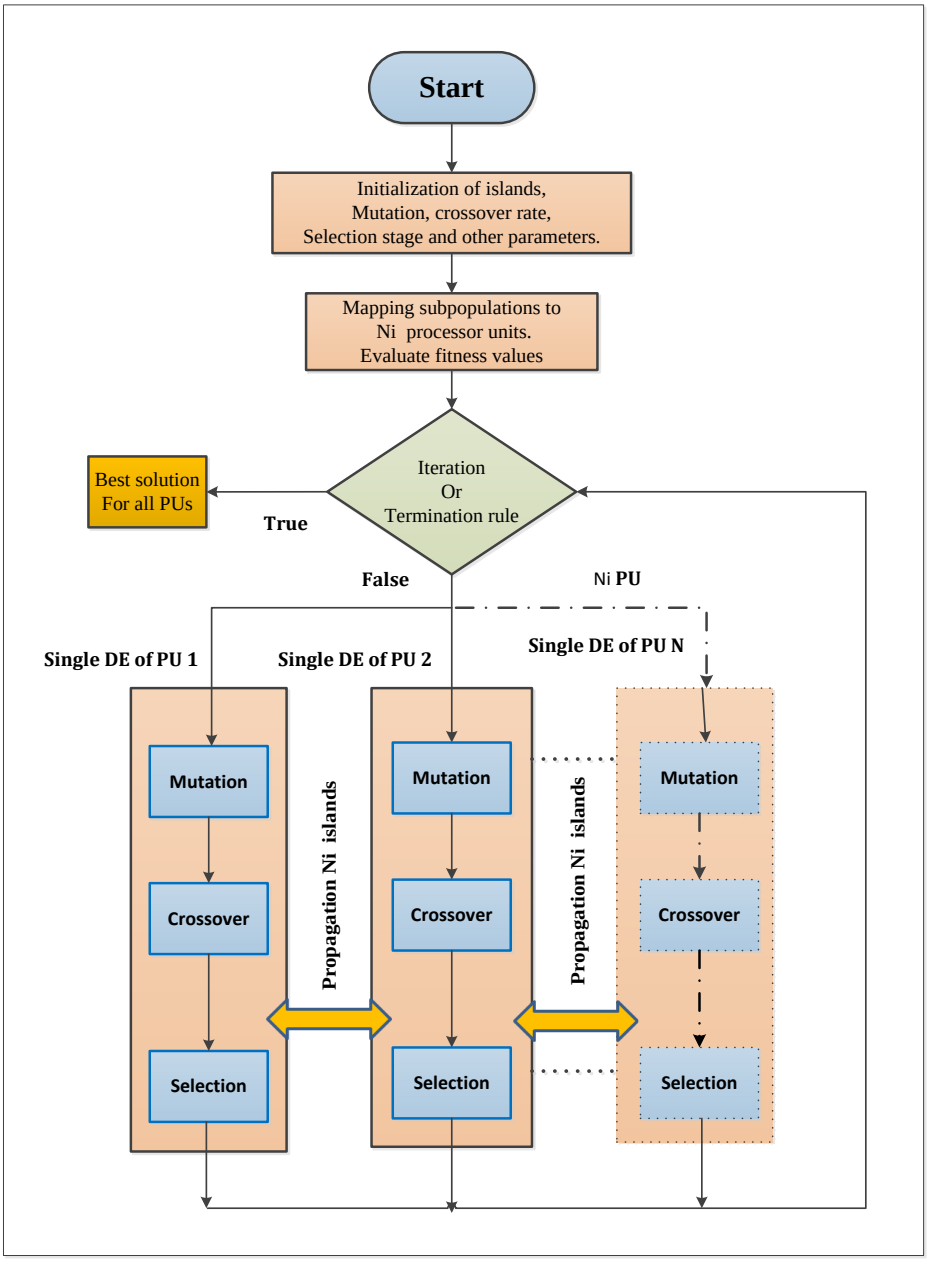


Figure 2: flowchart of proposed Parallel Differential Evolution algorithm

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

3. TEST FUNCTIONS

The benchmark functions used in the experiments have a fairly complex structure, and some of them have a large number of dimensions that make them perfect for studying, testing and comparing the results with the modified differential evolution method of Tsoulos, I.G. [34].

Benchmark functions

To evaluate the effectiveness of the suggested parallel PDE method in locating the global minimum of functions, a set of test functions that are used here is selected [35, 36]. The selected functions are as follows:

- **BF1** function (Bohachevsky1) is defined as follows:

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

With $x \in [-100, 100]^2$

- **BF2** function (Bohachevsky1) is defined as follows:

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

With $x \in [-50, 50]^2$

- **BRANIN** function is given by:

$$f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$$

At range $-5 \le x_1 \le 10$, $0 \le x_2 \le 15$.

- **CM (**Cosine Mixture**)** function

$$f(x) = 0.1 * \sum_{i=1}^{n}\cos(5\pi x_i) - \sum_{i=1}^{n} x_i^2$$

With $x \in [-1, 1]^n$

- **Camel back** function. Six Hump The function is given by:

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$

At range $-5 \le x_i \le 5$.

- **Easom** function is given by:

$$f(x) = -cos(x_1)\,cos(x_2)\,e^{(-((x_2-\pi)^2+(x_1-\pi)^2))}$$

With $x \in [-100, 100]^2$

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

- **EXPONENTIAL** function. The function is given by the following:

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

At range $-1 \leq x_i \leq 1$.

In the experiments the function used with $n = 4$. And the corresponding function are denoted by EXP16.

- **Goldstein and Price function.** The function is given by the equation:

$$f(x) = [1 + (x_1+x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] *$$
$$[30 + (2x_1-3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)].$$

With $x \in [-2, 2]^2$

- **GRIEWANK** function. The function is given by the following:

$$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2} \frac{\cos(x_i)}{\sqrt{(i)}},$$

With $x \in [-100, 100]^2$

- **Hansen** function. The function is given by the following:

$$f(x) = \left(\sum_{i=1}^{5} (i\ cos((i+1)x_1 + i)\right) * \left(\sum_{j=1}^{5} (j\ cos((j+1)x_2 + j)\right)$$

With $x \in [-10, 10]^2$

- **HARTMAN3** function. The function is given by the following:

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

Where a $=a_{ij}=\begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}$, c $=c_i=\begin{bmatrix} 1.0 \\ 1.2 \\ 3.0 \\ 3.2 \end{bmatrix}$, P $=p_{ij}=\begin{bmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4387 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}$

With $x \in [0, 1]^3$

- **HARTMAN6** function. The function is given by the following:

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

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

Where a $=a_{ij}=\begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix}$, c $=c_i=\begin{bmatrix} 1.0 \\ 1.2 \\ 3.0 \\ 3.2 \end{bmatrix}$,

P $=p_{ij}=\begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8723 & 0.5743 & 0.1091 & 0.0381 \end{bmatrix}$

With $x \in [0, 1]^6$

- **POTENTIAL** function. As a test case, the molecular conformation corresponding to the global minimum of the energy of N atoms interacting via the Lennard–Jones potential [37] is utilized. The function to be minimized is defined as follows:

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

In the current experiments, two different cases were studied: N = 3, 4.

- **SHEKEL5** function:

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

Where a $=\begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{bmatrix}$, c $=\begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{bmatrix}$,

With $x \in [0, 10]^4$

- **SHEKEL7** function:

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

Where a $=\begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{bmatrix}$, c$=\begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{bmatrix}$,

With $x \in [0, 10]^4$

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

- **SHEKEL10** function:

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

Where a = $\begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix}$, c = $\begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.6 \end{bmatrix}$,

With $x \in [0, 10]^4$

- **SINUSOIDAL** function

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

The experiments use $n = 16$, and $z = \frac{\pi}{6}$ and the corresponding function denoted by the label SINU16.

- **RASTRIGIN** function. The function is given by the following:

$$f(x) = x_1^2 + x_2^2 - cos(18x_1) - cos(18x_2),$$

With $x \in [-1, 1]^2$

- **ROSENBROCK** function. The function is given by the following:

$$f(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2),$$

At range $-30 \le x_i \le 30$, in the experiments using function with n = 16.

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

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

The function has 2^n in the specified range and in the experiments n = 4, 5, 6.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

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

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

$x_i \in [-10,10]$, The function has 30^n local minima in the search space and in the experiments n = 3, 4.

4. EXPERIMENTAL RESULTS

To compare various differential evolution methods for global optimization, certain parameters were kept constant. The parallel implementation of differential evolution was employed in these comparative experiments. The performance of the proposed PDE algorithm was assessed through a series of experiments utilizing 10 parallel units. The existing OpenMP library [38] facilitated the parallelization, and the entire technique was implemented using the C++ programming language. All experiments were conducted on a system equipped with an Intel (R) Core (TM) i7-8650U multi-core processor and 16 GB of RAM, using (the Windows 11 Pro) operating system, and the experimental settings used in the proposed PDE method are presented in Table 1. The effect of the proposed PDE is shown graphically in Figure 3, where the function calls for the functions SHEKEL5, SHEKEL7, and SHEKEL10 using the four schemes of differential weights are plotted. The population size for all consists of 10 particles. To ensure the reliability and validity of the research, experiments were conducted 30 times and concerned Tables 2. In Table 2, the columns display the average number of function calls for each problem, with the last row indicating the total number of function calls. The fraction in parentheses represents the proportion of runs in which the global optimum was found; if this number is absent, it signifies that the global minimum was identified in every independent run (100% success rate). The column labeled STATIC refers to a fixed differential weight value (F = 0.8), while the columns Ali and MDE present results from experiments of Ali, M. Montaz and Charilogis, V.; Tsoulos, I.G, respectively conducted in [36]. The column for the proposed PDE corresponds to the proposed Parallel Differential Evolution algorithm that utilizes a (1-to-1) propagation scheme with 10 threads.

Table 1: The following settings were initially used to conduct the experiments

Parameter	Value	Explanation
NP	10	Number of populations (agents)
propagation	1-to-1	scenario
N_R	5	iterations
N_I	2	islands
M	30	Number of iterations
ϵ	10^{-4}	Small positive number

The values of the global minimum for all tested benchmark functions resulting from experiments in Tables 2 and 3 are depicted in Figures 3 and 4, respectively. The plot of Figure 3 reveals the superiority of the proposed PDE method, but STATIC, Ali, and MDE

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

methods show a higher number of function calls in all problems. Note that the proposed PDE algorithm proves its proficiency by getting an identical global minimum at (1, 1) with a value of zero for the ROSENBROCK function compared with the modified MDE that gets the global minimum at (0, 0).

Table 2: Statistical Comparison of Function Calls across Different Differential Evolution Optimization Methods and PDE Using a (1-to-1) Propagation Scheme with 10 Threads

NO	PROBLEMS	Static	Ali	MDE	Proposed PDE
1	BF1 (Bohachevsky)	996	1124	889	**300**
2	BF2 (Bohachevsky)	926	1026	816	**300**
3	BRANIN	878	900	730	**300**
4	CM4 (Cosine Mixture)	1148(0.70)	1991	1103	**610**
5	Camel back	1049	904(0.93)	846	**610**
6	Easom	447	448	446	**310**
7	EXP (16)	3578	7082	3521	**300**
8	EXP (32)	7082	14125	7022	**300**
9	GOLDSTEIN and PRICE	945	993	915	**300**
10	GRIEWANK	947	921	826	**300**
11	Hansen	2104	1949	1479	**310**

Continued Table 2 Statistical Comparison of Function Calls Across Different Differential Evolution Optimization Methods and PDE Using a (1-to-1) Propagation Scheme with 10 Threads

NO	PROBLEMS	Static	Ali	MDE	Proposed PDE
12	HARTMAN3	1017	1005	952	**310**
13	HARTMAN6	4679(0.90)	3744(0.97)	3128(0.87)	**610**
14	Potential 3	21473	2284	8197	**300**
15	Potential 4	44191(0.43)	3098 (0.33)	24659(0.97)	**300**
16	RASTRIGIN	841	994	777	**300**
17	ROSENBROCK (16)	160349	160538(0.60)	38315	**300**
18	SHEKEL5	4389(0.97)	4266	2839(0.83)	**1500**
19	SHEKEL7	3905	3685	2668	**2100**
20	SHEKEL10	4049	3548	2629	**2400**
21	SINUSOIDAL (16)	6892	3628(0.97)	16905	**3000**
22	Test function 2N4	2785	2275	2221	**300**
23	Test function 2N5	4481	3170	3122	**3000**
24	Test function 2N6	6852	4286	4296	**3000**
25	Test function 30N3	1033	1098	951	**610**
26	Test function 30N4	1355	1444	1285	**610**
-	**TOTAL**	**288394**	**230529(0.8)**	**131539(0.67)**	**22580**
-	**TOTAL without ROSENBROCK (16)**	**128045**	**69991(0.2)**	**93224(0.67)**	**22280**

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

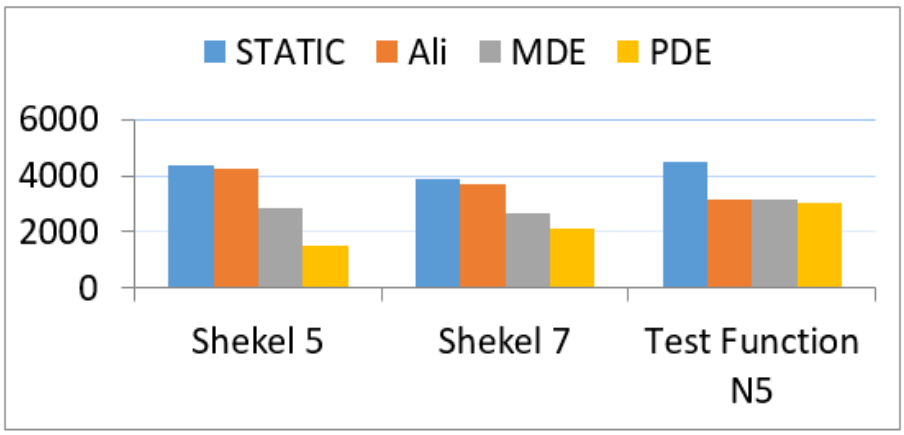


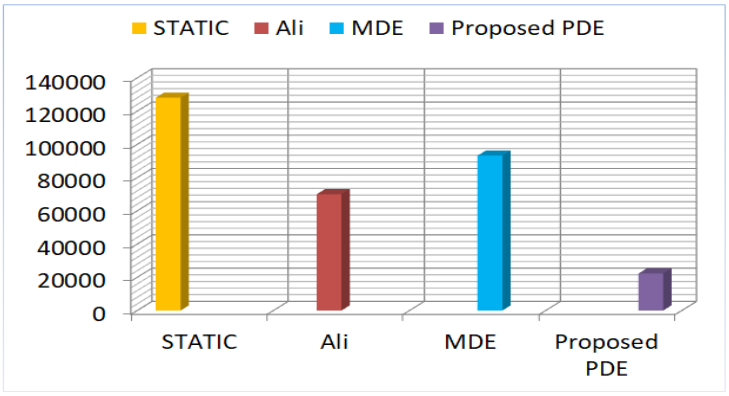
Figure 3: The effect of the usage of the proposed PDE algorithm versus different differential evolution optimization methods

Table 3: The values of the global minimum for test functions by the proposed (PDE) algorithm are approximately the same accuracy as the modified differential evolution (MDE)

NO	PROBLEMS	MDE	Proposed PDE
1	BF1(Bohachevsky)	0.0	0.0
2	BF2(Bohachevsky)	0.0	0.0
3	BRANIN	0.397887	0.397887
4	CM4(Cosine Mixture)	-	**Global max 0.4**
5	Camel	-	- 1.0316
6	Easom	- 1.0	- 1.0
7	EXP (16)	- 1.0	- 1.0
8	EXP (32)	- 1.0	- 1.0
9	GOLDSTEIN	3.0	3.0
10	GRIEWANK	0.0	0.0
11	Hansen	-176.54	-176.54
12	HARTMAN3	-3.86	-3.86
13	HARTMAN6	-3.322	-3.321
14	Potential 3	0	0
15	Potential 4	0	0
16	RASTRIGIN	- 2.0	- 2.0
17	ROSENBROCK (16)	**0.0 at (0,0)**	**0.0 at (1,1) better**
18	SHEKEL5	- 10.10	- 10.09
19	SHEKEL7	- 10.34	- 10.34
20	SHEKEL10	- 10.53	- 10.52
21	SINUSOIDAL16	- 3.5	- 3.5
22	Test function 2N4	-156.6646	-156.665
23	Test function 2N5	-195.8308	-195.831
24	Test function 2N6	-234.9969	-234.997
25	Test function 30N3	0	0
26	Test function 30N4	0	0

ISSN 1533-9211

DOI: 10.5281/zenodo.14463762



	STATIC	Ali	MDE	Proposed PDE
Total numbers of Function Calls	128045	69991.2	93224.67	22280

Figure 4: Comparison of total function calls of benchmark tested functions without ROSENBROCK function using different differential evolution optimization methods

Table 4 provides further insights, presenting computational times, additionally; the comparisons reveal a reduction in execution time, as illustrated in Figure 5, alongside a decrease in the time needed to attain the global minimum values for the functions.

Table 4: Time comparisons (seconds) when applying the proposed PDE and different differential evolution optimization methods to test functions

NO	PROBLEMS	MTDE	MDE	Proposed PDE
1	POTENTIAL 4	1.42	0.812	0.0966
2	POTENTIAL 3	0.585	0.218	0.0654
3	EXP (32)	0.425	0.412	0.320
4	EXP (16)	0.131	0.135	0.039
5	SINU (16)	0.338	0.768	0.187

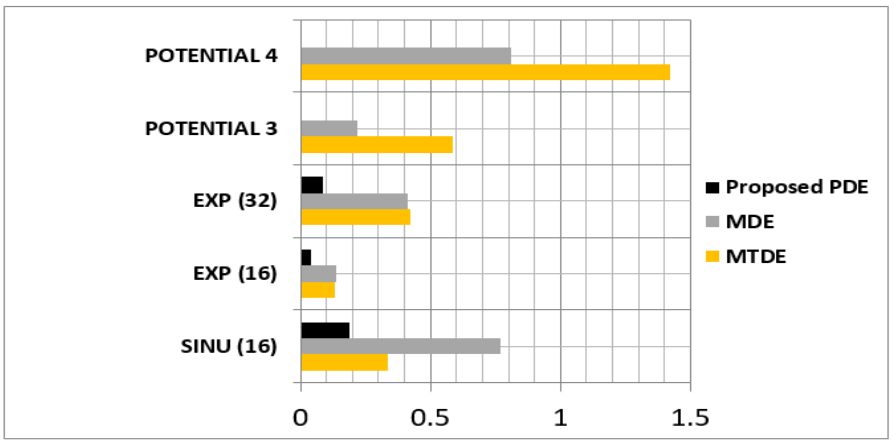


Figure 5: Comparison of executing time applying the proposed PDE and different differential evolution optimization methods to some of the test functions

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

5. CONCLUSION

By harnessing parallel processing capabilities, experiments can drastically reduce computational time and enhance the efficiency of the optimization process. However, parallelizing differential evolution necessitates meticulous consideration of several factors, such as load balancing, exploration-exploitation trade-offs, and communication overhead. Parallel implementations of DE present a compelling approach for accelerating optimization tasks and addressing intricate problems. The proposed technique operates within parallel computing environments, employing the differential evolution algorithm to establish independent populations across parallel computing units. These units partition the initial population of agents, periodically exchanging the best objective function values via a (1-to-1) propagation technique, where randomly selected subpopulations share information. Furthermore, even a subset of the computing units can efficiently determine when to terminate the method. The experimental results confirmed that the proposed technique effectively identified the global minimum across various problems from the related literature. Generally, while the parallel processing units increased, the number of required function calls and the time needed for the optimization process decreased. This indicates that the proposed PDE algorithm is more efficient in exploring the search space of the test functions, leading to greater accuracy and reduced time in achieving the global minimum.

References

1) Kudyshev, Z.A.; Kildishev, A.V.; Boltasseva, V.M.S.A. Machine learning–assisted global optimization of photonic devices. Nanophotonics 2021, 10, 371–383. https://doi.org/10.1515/nanoph-2020-0376

2) Ding, X.L.; Li, Z.Y.; Meng, J.H.; Zhao, Y.X.; Sheng, G.H. Density-functional global optimization of (LA2O3)n Clusters. J. Chem. Phys. 2012, 137, 214311. DOI:10.1063/1.4769282

3) Morita, S.; Naoki, N. Global optimization of tensor renormalization group using the corner transfer matrix. Phys. Rev. B 2021, 103, 045131. https://doi.org/10.1103/PhysRevB.103.045131

4) Heiles, S.; Johnston, R.L. Global optimization of clusters using electronic structure methods. Int. J. Quantum Chem. 2013, 113, 2091–2109. https://doi.org/10.1002/qua.24462

5) Yang, Y.; Pan, T.; Zhang, J. Global Optimization of Norris Derivative Filtering with Application for Near-Infrared Analysis of Serum Urea Nitrogen. Am. J. Anal. Chem. 2019, 10, 143–152. DOI: 10.4236/ajac.2019.105013

6) Grebner, C.; Becker, J.;Weber, D.; Engels, B. Tabu search based global optimization algorithms for problems in computational Chemistry. J. Cheminf. 2012, 4, 10. DOI:10.1186/1758-2946-4-S1-P10

7) Dittner, M.; Müller, J.; Aktulga, H.M.; Hartke, B.J. Efficient global optimization of reactive force-field parameters. Comput. Chem. 2015, 36, 1550–1561. DOI:10.1002/jcc.23966

8) Zhao, W.; Wang, L.; Zhang, Z. Supply-Demand-Based Optimization: A Novel Economics-Inspired Algorithm for Global Optimization. IEEE Access 2019, 7, 73182–73206. DOI:10.1109/ACCESS.2019.2918753

9) Mishra, S.K. Global Optimization of Some Difficult Benchmark Functions by Host-Parasite Co-Evolutionary Algorithm. Econ. Bull. 2013, 33, 1–18. Available at https://ssrn.com/abstract=2343448

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

10) Freisleben, B.; Merz, P. A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. In Proceedings of the IEEE International Conference on Evolutionary Computation, Nagoya, Japan, 20–22 May 1996; pp. 616–621. DOI:10.1109/ICEC.1996.542671

11) Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. Science 1983, 220, 671–680. DOI: 10.1126/science.220.4598.671

12) Van Laarhoven, P.J.M.; Aarts, E.H.L. Simulated Annealing: Theory and Applications; Riedel, D., Ed.; Springer: Dordrecht, The Netherlands,1987. https://doi.org/10.1007/978-94-015-7744-1

13) Goffe, W.L.; Ferrier, G.D.; Rogers, J. Global Optimization of Statistical Functions with Simulated Annealing. J. Econom. 1994, 60, 65–100. https://doi.org/10.1016/0304-4076(94)90038-8

14) Storn, R. On the usage of differential evolution for function optimization. In Proceedings of the North American Fuzzy Information Processing, Berkeley, CA, USA, 19–22 June 1996; pp. 519–523. DOI: 10.1109/NAFIPS.1996.534789

15) Kennedy, J.; Everhart, R.C. Particle Swarm Optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; IEEE Press: Piscataway, NJ, USA, 1995; Volume 4, pp. 1942–1948. http://dx.doi.org/10.1109/ICNN.1995.488968

16) Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. IEEE Comput. Intell. Mag. 2006, 1, 28–39. DOI: 10.1109/MCI.2006.329691

17) Goldberg, D. Genetic Algorithms in Search, Optimization and Machine Learning; Addison-Wesley Publishing Company: Reading, MA, USA, 1989. ISBN:978-0-201-15767-3

18) Michaelewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs; Springer: Berlin, Germany, 1996. ISBN: 3-540-58090-5

19) Rocca, P.; Oliveri, G.; Massa, A. Differential Evolution as Applied to Electromagnetics. IEEE Antennas Propag. Mag. 2011, 53, 38–49. DOI: 10.1109/MAP.2011.5773566

20) Lee,W.S.; Chen, Y.T.; Kao, Y. Optimal chiller loading by differential evolution algorithm for reducing energy consumption. Energy Build. 2011, 43, 599–604. DOI:10.1016/j.enbuild.2010.10.028

21) Yuan, Y.; Xu, H. Flexible job shop scheduling using hybrid differential evolution algorithms. Comput. Ind. 2013, 65, 246–260. DOI:10.1016/j.cie.2013.02.022

22) Xu, L.; Jia, H.; Lang, C.; Peng, X.; Sun, K. A Novel Method for Multilevel Color Image Segmentation Based on Dragonfly Algorithm and Differential Evolution. IEEE Access 2019, 7, 19502–19538. DOI: 10.1109/ACCESS.2019.2896673

23) Kenneth V.; Price Rainer M.; Storn. Differential Evolution: A Practical Approach to Global Optimization, Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland, Springer Berlin, Heidelberg.2005.

24) Georgioudakis, M., & Plevris, V.A comparative study of differential evolution variants in constrained structural optimization. Frontiers in Built Environment: Computational Methods in Structural Engineering, 6:102, 2020. https://doi.org/10.3389/fbuil.2020.00102

25) F. Neri, E. Mininno, Memetic Compact Differential Evolution for Cartesian Robot Control. IEEE Comput. Intell. 2010, 5, 54–65. DOI:10.1109/MCI.2010.936305

26) Mininno, E.; Neri, F.; Cupertino, F.; Naso, D. Compact Differential Evolution. IEEE Trans. Evol. 2011, 15, 32–54. https://doi.org/10.1109/TEVC.2010.2058120

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14463762

27) Qin, A.K.; Huang, V.L.; Suganthan, P.N. Differential Evolution Algorithm with Strategy Adaptation for Global Numerical Optimization. IEEE Trans. Evol. Comput. 2009, 13, 398–417. DOI:10.1109/TEVC.2008.927706

28) Hachicha, N.; Jarboui, B.; Siarry, P. A fuzzy logic control using a differential evolution algorithm aimed at modelling the financial market dynamics. Inf. Sci. 2011, 181, 79–91. https://hal.science/hal-01679171

29) Kaelo, P.; Ali, M.M. A numerical study of some modified differential evolution algorithms. Eur. J. Oper. 2006, 169, 1176–1184. DOI:10.1016/j.ejor.2004.08.047

30) Sophia Mitchell. The Parallel Implementation of Differential Evolution Method. 2023. DOI: 10.4303/JEM/101387

31) Powell, M.J.D. A Tolerant Algorithm for Linearly Constrained Optimization Calculations. Math Program. 1989, 45, 547–566. https://doi.org/10.1007/BF01589118

32) Corcoran, A.L.; Wainwright, R.L. A parallel island model genetic algorithm for the multiprocessor scheduling problem. In Proceedings of the 1994 ACM Symposium on Applied Computing, SAC '94, Phoenix, AZ, USA, 6–8 March 1994; pp. 483–487. https://doi.org/10.1145/326619.326817

33) Whitley, D.; Rana, S.; Heckendorn, R.B. Island model genetic algorithms and linearly separable problems. In Evolutionary Computing; Series Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1997; Volume 1305, pp. 109–125. https://doi.org/10.1007/BFb0027170

34) Charilogis, V.; Tsoulos, I.G.; Tzallas, A.; Karvounis, E. Modifications for the Differential Evolution Algorithm. Symmetry 2022,14. https://doi.org/10.3390/sym14030447

35) Ali, M.M. Charoenchai Khompatraporn, Zelda B. Zabinsky, A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems. J. Glob. Opt. 2005, 31, 635–672. https://doi.org/10.1007/s10898-004-9972-2

36) Floudas, C.A.; Pardalos, P.M.; Adjiman, C.; Esposoto, W.; G¨um¨us, Z.; Harding, S.; Klepeis, J.; Meyer, C.; Schweiger, C. Handbook of Test Problems in Local and Global Optimization; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1999. https://doi.org/10.1007/978-1-4757-3040-1

37) Lennard-Jones, J.E. On the Determination of Molecular Fields. Proc. R. Soc. Lond. A 1924, 106, 463–477. https://doi.org/10.1098/rspa.1924.0082

38) Chandra, R.; Dagum, L.; Kohr, D.; Maydan, D.; McDonald, J.; Menon, R. Parallel Programming in OpenMP; Morgan Kaufmann Publishers Inc.: Burlington, MA, USA, 2001. ISBN 1-55860-671-8.