MDPI

*Article*

# Next-Gen Evolutionary Algorithms: A Modified Differential Evolution Approach

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

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

**Abstract:** This paper presents an innovative optimization algorithm based on Differential Evolution (DE) that combines advanced mutation techniques with intelligent termination mechanisms. The proposed algorithm is designed to address the main limitations of classical DE, offering enhanced performance for complex optimization problems. The core scientific contribution of this research focuses on three key aspects. First, we develop a hybrid dual-strategy mutation system where the first strategy emphasizes exploration of the solution space through monitoring of the optimal solution, while the second strategy focuses on exploitation of promising regions using dynamically weighted differential terms. This dual mechanism ensures a balanced approach between discovering new solutions and improving existing ones. Second, the algorithm incorporates a novel Majority Dimension Mechanism that evaluates candidate solutions through dimension-wise comparison with elite references (best sample and worst sample). This mechanism dynamically guides the search process by determining whether to intensify local exploitation or initiate global exploration based on majority voting across all dimensions. Third, the work presents numerous new termination rules based on quantitative evaluation of metric value homogeneity. These rules extend beyond traditional convergence checks by incorporating multidimensional criteria that consider both solution distribution and evolutionary dynamics. This system enables more sophisticated and adaptive decision-making regarding the optimal stopping point of the optimization process. The methodology is validated through extensive experimental procedures covering a wide range of optimization problems. The results demonstrate significant improvements in both solution quality and computational efficiency, particularly for high-dimensional problems with numerous local optima. The algorithm's practical applications span multiple domains of modern technology and science. In engineering design, it demonstrates capability for solving complex parametric optimization problems. For artificial intelligence and machine learning, it provides a powerful tool for automated hyperparameter tuning. Additionally, the method finds applications in economic modeling and biological systems analysis. The study concludes with a critical analysis of the proposed method's advantages and limitations, along with suggestions for future research directions. Special emphasis is placed on extending the algorithm for dynamic optimization problems and integrating deep learning techniques for automated parameter configuration. The research findings highlight the proposed algorithm's potential as a high-performance tool for solving complex optimization challenges in contemporary scientific and technological contexts.

**Keywords:** Optimization; Differential Evolution; Evolutionary Algorithms; Global Optimization; Adaptive Termination; Mutation Strategies; Metaheuristics;

**Citation:** Gianni A.M.; Tsoulos, I.G.; Charilogis, V. ;Kyrou, G. Next-Gen Evolutionary Algorithms: A Modified Differential Evolution Approach. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

**Copyright:** © 2025 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Global optimization is concerned with finding the global minimum of a continuous objective function $f : S \to R$ where $S \subseteq R^n$ is a compact set. Formally, the problem is defined as:

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

where $x^*$is the global minimizer. Typically, is an n-dimensional hyperrectangle:

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

with, $a_i$ and $b_i$ representing lower and upper bounds for each variable $x_i$.

This paper focuses on the development of an innovative optimization algorithm based on DE. DE has emerged as one of the most popular tools for optimizing complex problems across various fields, such as artificial intelligence, machine learning, biological systems analysis, and engineering optimization applications[**? ? ? ?** ]. According to the literature, DE is characterized by its simplicity and flexibility, enabling it to solve high-complexity problems [**? ? ? ?** ].

This novel approach introduces three key scientific contributions. First, a hybrid dual-strategy mutation system is developed that balances exploration of the solution space with exploitation of promising regions. Exploration is achieved through linearly decreasing disturbance coefficients, while exploitation relies on dynamically weighted differential terms, as discussed in earlier research [**? ?** ]. Second, a system of five pioneering termination rules is introduced, based on multidimensional criteria that consider the distribution of solutions and evolutionary dynamics, focusing on stabilization and solution quality [**? ? ? ?** ].

Additionally, the proposed algorithm includes improvements related to parameter optimization using techniques such as modified swarm strategies and hybrid exploration methods [**? ? ?** ]. This approach has proven highly effective in addressing multimodal problems with high computational requirements [[**? ?** ]]. The algorithm's validation is carried out through extensive experimental trials, utilizing a wide range of benchmark functions to evaluate its performance and stability [[**? ? ?** ]].

The findings of this study highlight the algorithm's applicability across various fields. Specifically, in engineering technologies, the algorithm offers solutions for the parameterization of complex systems [[**? ?** ]]. In machine learning, it provides exceptional accuracy in hyperparameter tuning for neural networks, while in economic modeling, it contributes to trend forecasting and resource optimization [**? ? ?** ].

The research concludes with a critical analysis of the method's advantages and limitations, focusing on opportunities for future improvements. Specifically, it suggests incorporating dynamic optimization techniques and leveraging deep learning technologies for automated parameter configuration [**? ? ?** ]. These prospects underscore the contribution of this study to the optimization literature, utilizing hybrid methodologies to solve high-complexity and multimodal problems [**? ? ?** ].

Furthermore, the study's results demonstrate the proposed algorithm's capability to solve high-dimensional problems with numerous local minima, making it valuable for modern scientific and technological challenges [**? ?** ]. This algorithm is successfully applied in domains such as engineering design, automated hyperparameter configuration in machine learning, and economic modeling [**? ? ?** ]. Its potential for extension to dynamic problems and integration with deep learning techniques for automated parameterization presents exciting prospects for the future [**? ?** ]. This work highlights the importance of hybrid methodologies in contemporary optimization research and emphasizes their contribution to solving complex problems [**? ? ?** ].

The rest of the paper is organized as follows:

- Section 1: Introduction
- Section 2: Materials and Methods2

## 2. Materials and Methods

### 2.1. *The original DE*

**Algorithm 1** Pseudocode for Standard DE Algorithm

1. Initialization:
   (a) Set parameters:
      i. Population size $NP$
      ii. Crossover probability $CR \in [0, 1]$
      iii. Differential weight $F \in [0, 2]$
      iv. Maximum iterations maxIterations
   (b) Randomly initialize population samples$[1..NP]$ within bounds
   (c) Evaluate initial fitness $fitnessArray[i]$ for each individual
   (d) Initialize global best solution $bSample$ and fitness $bF2x$
2. Main Optimization Loop (for iteration = 1 to maxIterations):
   For each individual $x_i$ in population ($i = 1\ to\ NP$):
   (a) Parent Selection:
      i. Randomly select 3 distinct indices $indexA, indexB, indexC \neq i$
   (b) Trial Vector Creation:
      i. Select random dimension $R$
      ii. For each dimension $d$ (1 to dimension):
         A. $ri$= random number $\in [0, 1]$
         B. If $ri \leq CR$ OR $d == R$:
         $trialsample[d] = samples[indexA][d] + F \times (samples[indexB][d] - samples[indexC][d])$
         A. Apply boundary correction if needed
         B. Else:
         $trialsample[d] = samples[i][d]$
   (c) Selection:
      i. Evaluate trial vector: $minValue = f(trialsample)$
      ii. If $minValue \leq fitnessArray[i]$:
         A. Update $samples[i] = trialsample$
         B. Update $fitnessArray[i] = minValue$
         C. Update global best $bSample$ if improved
3. Local Search Phase (optional):
   For each individual $i$:
   (a) If $random[0, 1] < localSearchRate$:
   $refinedFitness = localSearch(samples[i])$
   If $refinedFitness < fitnessArray[i]$:
   Update $samples[i]$ and $fitnessArray[i]$
   Update global best $bSample$ if improved
4. Termination:
   (a) When max iterations reachedor any of the termination rules is verified2.4
   (b) Return best solution bSample found

### 2.2. ***Detailed formulation of the algorithm***

**Algorithm 2** Modified Differential Evolution with Dual Mutation Strategies

1. Initialization Phase:
   (a) Set population size $NP$
   (b) Initialize parameters:
      i. Crossover probability $CR \in [0, 1]$
      ii. Local search probability localSearchRate
      iii. Problem boundaries $lmargin[], rmargin[]$
   (c) Randomly initialize population $samples[i], i = 1..NP$
   (d) Evaluate initial fitness $fitnessArray[i]$ for all individuals
   (e) Identify global best solution$bSample$ and fitness $bF2x$
2. Main Optimization Loop (until max iterations reached):
   For each individual $x_i$ in population ($i = 1toNP$):
   (a) Strategy Selection:
      i. Generate random number $randStrategy \in [0, 1]$
      ii. If $randStrategy < 0.2$: Use Strategy 1
      iii. Else: Use Strategy 2
   (b) Mutation Phase:
   Strategy 1 (Exploration):
      i. For each dimension j:
      Generate random numbers $r = \in [0, 1]$
      $distance = \mid samples[i][j] - bestSample[j] \mid$
      $trialsample[j] = bSamply[j] - r \times distance$
      Apply boundary repair if needed
   Strategy 2 (Exploitation):
      i. Select $dimension + 1$ distinct random indices from population ($\neq i$)
      ii. Choose random dimension $R$
      iii. For each dimension $d$:
         A. $r_i = random[0, 1]$
         B. If $r_i > CR$ or $d == R$:
         Generate random number$R_a = \in [-0.0, 1.0]$
         $differentialWeight = \frac{1}{2} + 2 \times R_a$**[? ]**
         $trialsample[d] = samples[index_1][d] + differentialWeight \times (samples[index_2][d] - samples[otherindices_i][d])$
         Apply boundary repair
         C. Else:
         $trialsample[d] = x_i[d]$
   (c) Selection:
      i. Evaluate trial vector: $minValue = f(trialsample)$
      ii. If $minValue < fitnessArray[i]$:
      $samples[i] = trialsample$
      $fitnessArray[i] = minValue$
      If $minValue < bF2x$:
      Update global best $bSamply = trialsample$
      $bF2x = minValue$
3. Local Search Phase (optional):
   For each individual $i$:
   (a) If $random[0, 1] < localSearchRate$:
   $refinedFitness = localSearch(samples[i])$
   If $refinedFitness < fitnessArray[i]$:
   Update $samples[i]$ and $fitnessArray[i]$
   Update global best if improved
4. Termination:
   (a) When max iterations reached or any of the termination rules is verified2.4
   (b) Return global best solution $bSample$

### 2.3. ***Majority Dimension Mechanism***

The Majority Dimension Mechanism plays a significant role in optimization algorithms, where it serves as a guiding tool for the search process. It operates by comparing a current sample $s$ against two critical reference points: an optimal sample $b$ representing a known good solution or minimum, and a worse sample $w$ representing less desirable solutions. The mechanism's operation begins with a dimension-by-dimension analysis of the sample. For each vector element, it calculates the absolute differences between the sample and the corresponding elements of both reference vectors. The mechanism then determines which reference vector the current sample aligns with more closely in each individual dimension. When the function returns true, indicating the sample is closer to the optimal sample in most dimensions, this signifies the current position is near a previously discovered minimum. In this case, the optimization algorithm may make decisions such as changing direction, increasing the search step size, or applying randomized variations, aiming to avoid local minima traps and explore new regions of the solution space. Conversely, when the function returns false, indicating the sample is closer to the worse sample, this suggests the current position warrants further exploration. Here, the algorithm may focus its search on the current region, hoping for further solution improvement. In a Euclidean parameter space, this mechanism enables dynamic search adaptation. When leading to true, it encourages the algorithm to make larger jumps toward new directions or employ local minima escape methods. When leading to false, it concentrates the search around the current area for more detailed exploration. The key advantages of this mechanism include its ability to prevent stagnation at suboptimal solutions, its operational flexibility by focusing on individual dimensions rather than aggregate measures, and its broad applicability to various optimization algorithms - from gradient-based to evolutionary approaches. In summary, the Majority Dimension Mechanism functions as an intelligent director in optimal solution searches. It offers a balanced approach between exploring new regions and exploiting known good solutions, enhancing both the efficiency and reliability of the optimization process without being constrained by local optima.

The formula (function) of Majority Dimension Mechanism :

$$\text{checkSample}(s,b,w) = \begin{cases} \text{true}, & \text{if } \sum_{d=1}^{dimension} \mathbb{I}(|s_d - b_d| < |s_d - w_d|) > \sum_{d=1}^{dimension} \mathbb{I}(|s_d - w_d| \\ \text{false}, & \text{else} \end{cases}$$

Where:

$s = [s_1, s_2, \ldots, s_d]$ : The sample being evaluated

$b = [b_1, b_2, \ldots, b_d]$ : The best sample

$w = [w_1, w_2, \ldots, w_d]$ : The worst sample

### 2.4. ***The termination rules***

The algorithm's termination mechanism evaluates similarity between specific values in the fitness array across consecutive iterations. Each rule focuses on different aspects of population evolution by monitoring changes in critical metrics from the fitness array. For the best solution, the stability of its value is checked across iterations. If the difference remains negligible for a specified number of iterations, similarity is considered achieved. Similarly, an equivalent check is applied to the population's worst solution, monitoring the stability of its value. Furthermore, the mechanism evaluates similarity among solution groups. For top solutions, it examines the sum of a percentage of the best fitness values per iteration, while for worst solutions it performs an analogous check on the sum of corresponding values. An additional rule measures similarity in solution range by comparing the difference between best and worst values across iterations. Finally, a rule checks similarity in improvement rate by comparing how much the best and worst solutions have improved between iterations. All these rules aim to identify moments when fitness array values show significant similarity for sufficient duration, indicating the population has reached stability. This logic ensures the algorithm terminates only when adequate result stabilization occurs, optimizing runtime without compromising solution quality. Using

multiple similarity rules provides a comprehensive convergence picture, covering various aspects of population evolution.

General parameters:

- $e$ controls the precision requirement (typically set to 1e-6)
- $sim$ determines the required stability duration (commonly 5-10 iterations)

#### 2.4.1. Best solution similarity

The termination mechanism is based on a simple criterion that evaluates the similarity of values in the fitness array. Specifically, at each iteration $iter$, the difference

$$\delta_{sim}^{(iter)} = \left| f_{sim,min}^{(iter)} - f_{sim,min}^{(iter-1)} \right|, \tag{2}$$

is calculated, where $f_{sim,min}^{(iter)}$ represents the best fitness value found up to iteration $iter$. If the difference $\delta_{sim}^{(iter)}$ remains less than or equal to a predefined accuracy threshold $e$ for at least $sim$ consecutive iterations, then the population is considered to have converged and the process terminates. This criterion ensures that the algorithm will stop only when minimal progress occurs in the best solution for a sufficient time period, thereby optimizing computational time.

#### 2.4.2. Worst solution similarity

The termination mechanism is based on the difference

$$\delta_{sim}^{(iter)} = \left| f_{sim,max}^{(iter)} - f_{sim,max}^{(iter-1)} \right|, \tag{3}$$

where$f_{sim,max}^{(iter)}$ represents the worst fitness value at iteration iter. If $\delta_{sim}^{(iter)} \leq e$ for $sim$ consecutive iterations, the algorithm terminates, indicating the population has stabilized. This guarantees termination only occurs when the worst solution stops improving significantly. This logic provides an efficient way to monitor population stability, optimizing computational time without compromising solution quality. The mechanism is particularly useful in applications where fitness value variance across the population is a critical factor for determining convergence.

#### 2.4.3. Top Solutions Similarity

The termination mechanism evaluates the difference

$$\delta_{best} = \left| \sum_{i=NP-K}^{NP} f_{sorted}^{(iter)}[i] - \sum_{i=NP-K}^{NP} f_{sorted}^{(iter-1)}[i] \right|, \tag{4}$$

when $K = \lceil \text{population} \times \text{sumRate} \rceil$,

where $f_{sorted}^{(t)}[i]$ represents the i-th best fitness value in the sorted population at iteration $iter$, $NP$ is the population size, and $K$ is an integer smaller than $NP$. The process terminates when the difference $\delta_{best}$ remains below a specified threshold $e$ for a predetermined number of iterations, indicating stabilization of the overall performance of the population's top solutions. This criterion focuses on monitoring the collective trend of the K best solutions, providing a robust and reliable convergence measure. It proves particularly valuable in optimization problems where the stability of the population's elite solutions serves as a critical indicator of the algorithm's final convergence, ensuring the process completes only when sufficient stability is achieved among the top-performing solutions.

#### 2.4.4. Bottom Solutions similarity

The termination mechanism evaluates the difference

$$\delta_{worst} = \left| \sum_{i=N-K}^{N} f_{sorted}^{(iter)}[i] - \sum_{i=N-K}^{N} f_{sorted}^{(iter-1)}[i] \right|, \tag{5}$$

when $K = \lceil \text{population} \times \text{sumRate} \rceil$, where $f_{sorted}^{(iter)}[i]$ represents the i-th worst fitness value in the sorted population at iteration$iter$, with $NP$ denoting the total population size and $K$ being an integer smaller than $NP$. The process terminates when the difference $\delta_{worst}$ remains below a predefined accuracy threshold $e$ for a specified number of consecutive iterations, indicating that the population's worst solutions have stabilized. This approach focuses on monitoring the collective behavior of the $K$ worst solutions, providing an additional convergence metric that complements classical termination criteria. It proves particularly valuable in optimization problems where the evolution of low-quality solutions serves as a critical factor for determining the algorithm's overall convergence. The mechanism ensures the optimization process completes only when stability is achieved in both the best and worst solutions of the population, offering a more comprehensive convergence assessment. The $\delta_{worst}$ metric serves as an effective early-warning system for population stagnation, particularly useful in multimodal optimization where poor solutions may indicate unexplored regions of the search space. By incorporating both solution quality extremes in termination decisions, the algorithm achieves more robust performance across diverse problem landscapes.

#### 2.4.5. Solution Range Similarity

The termination mechanism calculates the range difference

$$\delta_{range} = |(f_{worst}^{(iter)} - f_{best}^{(iter)}) - (f_{worst}^{(iter-1)} - f_{best}^{(iter-1)})|, \tag{6}$$

when $\delta_{range} \leq \epsilon$ for $sim$ iterations, where $f_{worst}^{(iter)}$ and $f_{best}^{(iter)}$ represent the worst and best fitness values respectively in the current iteration. The process terminates when $\delta_{range}$ remains below a specified threshold $e$ for a predetermined number of iterations, indicating stabilization of the solution range within the population. This criterion measures the variation in fitness value range between consecutive iterations, providing a comprehensive perspective on convergence. It proves particularly effective in problems where the contraction of solution diversity serves as a critical convergence indicator. The mechanism enhances termination reliability by simultaneously monitoring both optimal and worst solutions, ensuring the algorithm completes only when the entire population reaches equilibrium. The $\delta_{range}$ metric offers unique advantages in multimodal optimization by capturing global population dynamics rather than just elite solution behavior. When maintained below threshold $e$, it indicates the population has sufficiently explored the search space and is concentrating around promising regions. This criterion works synergistically with other termination rules to provide robust convergence detection across various problem types and population sizes.

#### 2.4.6. Improvement Rate Similarity

$$\delta_{imp} = |(f_{worst}^{(iter-1)} - f_{worst}^{(iter)}) - (f_{best}^{(iter-1)} - f_{best}^{(iter)})|, \tag{7}$$

$\delta_{imp} \leq \epsilon$ for $sim$ iterations

#### 2.4.7. Termination rule: double box

The parameter $NP$ represents the number of agents participating in the algorithm. The termination rule is defined as follows: the process terminates if the value$\delta^{(iter)}$ is less than or equal to $e$ for a predefined number of iterations $sim$. The termination criterion is the so-called DoubleBox rule, which was first proposed in the work of Tsoulos [xxx]. According to this criterion, the search process terminates when sufficient coverage of the search space has been achieved. The coverage estimation is based on the asymptotic approximation of