MDPI

Article

The Echo Optimizer: A Novel Metaheuristic Inspired by Acoustic Reflection Principles

Vasileios Charilogis[1], Ioannis G. Tsoulos[2,*]

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

Abstract

The Echo Optimizer Method is an innovative optimization technique inspired by the natural behavior of sound echoes. It is based on generating modified solutions (echoes) that combine directed reflection toward the best-known solution and random noise that attenuates over time. The method introduces two groundbreaking mechanisms to enhance performance: an approximate evaluation system that avoids costly computations for unpromising solutions, and an echo memory that stores and reuses past evaluations. These mechanisms enable a significant reduction in computational resources (up to 40–60% fewer evaluations) while maintaining the method's effectiveness. The Echo Optimizer excels in balancing exploration of the solution space with exploitation of the best-known solutions, demonstrating impressive performance in problems with numerous local minima and high dimensionality. Experimental tests on standard optimization problems have shown faster convergence and reduced result variability compared to classical methods, making it a highly attractive choice for various optimization challenges, particularly in cases where evaluating the objective function is computationally expensive.

Keywords: Optimization; Echo Optimizer; Evolutionary Algorithms; Global Optimization; Adaptive Termination; Mutation Strategies; Metaheuristics;

Received:
Revised:
Accepted:
Published:

Citation: . The Echo Optimizer: A Novel Metaheuristic Inspired by Acoustic Reflection Principles. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/

1. Introduction

Global optimization deals with finding the absolute lowest point (global minimum) of a continuous objective function $f(x)$ defined over a bounded, n-dimensional search space S. Mathematically, the goal is to identify the point x^* in S where $f(x)$ attains its smallest possible value:

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

where:

- $f(x)$ is the objective function to minimize (e.g., cost, error, or energy).
- S is a compact (closed and bounded) subset of R^n, often defined as an n-dimensional hyperrectangle:

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

Here, a_i and b_i are the lower and upper bounds for each variablex_i, confining the search to a specific region.

Optimization constitutes a central field of computational mathematics with applications to multifaceted scientific and industrial problems. Optimization methods are classified into broad categories according to their underlying strategies and the properties of the problems they address. Among the most well-known techniques are classical gradient methods such as steepest descent [1] and Newton's method [2], stochastic methods like Monte Carlo [3] algorithms and simulated annealing [4,5] population-based methods including genetic algorithms [6] and differential evolution [7–10], convex optimization methods such as the ellipsoid method [11,12] and cutting-plane method [13], simplex-based methods like the Nelder-Mead method [14], response surface methods including krigin [15], trust-region methods like Bayesian optimization [16,17], conjugate gradient methods such as Fletcher-Reeves [18] and Polak-Ribière [19,20], constrained optimization methods including penalty function methods and interior-point methods, decomposition approaches like Benders decomposition [21,22] and Dantzig-Wolfe decomposition [23], space-partitioning methods such as DIRECT [24] and branch-and-bound [25,26], neural network-based methods including reinforcement learning algorithms, socially-inspired methods like particle swarm optimization [27,28] and ant colony optimization [29], physics-inspired methods including crystal structure optimization [30] and gravitational search algorithms [31], hybrid methods such as neuro-fuzzy algorithms [32], and biologically-inspired methods like photosynthesis algorithms [33] and DNA-based computing [34].

Within this context, the Echo Optimizer method (EO) introduces a novel approach based on the physical analogy of sound echoes. The core concept involves generating modified solutions that simulate sound reflection, with systematic adjustment of exploration intensity through a decay factor. This technique combines elements from population-based methods and stochastic techniques, offering a flexible mechanism for addressing diverse optimization problems. The method's ability to balance solution space exploration with exploitation of known optimal solutions makes it particularly effective for high-dimensional problems and non-convex functions.

The presentation of the method will focus on its theoretical foundations, algorithmic design, and experimental performance compared to other contemporary techniques. Furthermore, we will analyze its application potential to real-world problems, along with the challenges arising from its use in complex systems. This work aims to highlight the method's distinctive properties that make it a valuable addition to the toolkit of modern optimization techniques.

The rest of the paper is organized as follows:

The introduction provides the background and motivation for the study. The Echo Optimizer method is presented in Section 2, detailing the core algorithm and its components. Subsection 2.1 introduces the echo memory technique, followed by Subsection 2.2 which describes the approximate evaluations technique. Subsection 2.3 presents the complete algorithm. The experimental setup and benchmark results are discussed in Section 3. This includes real-world problems in Subsection 3.1, an overview of the ECHO method and its two mechanisms in Subsection 3.2, exploration and exploitation analysis in Subsection 3.3, parameter sensitivity analysis in Subsection 3.4, comparison of the ECHO method with others in Subsection 3.5, and finally the application of the Echo method with neural networks in Subsection 3.6. Conclusions and key findings are summarized in Section 4.

2. The Echo Optimizer method

The EO algorithm is an evolutionary method inspired by the natural behavior of sound echoes, combining mathematical optimization strategies with physical analogies. Its core principle lies in generating solutions (called "echoes") through a directed reflection of current solutions towards the best-known solution, coupled with the addition of random

noise that attenuates over time. The initial population of solutions is generated randomly within the bounds of the search space, and each solution is evaluated based on an objective function. During each iteration, solutions are updated using a combination of directed reflection, which pulls them closer to the best-known solution, and random noise, which allows exploration of new regions in the solution space. The noise diminishes as the algorithm progresses, enabling a transition from exploration to exploitation. This mechanism ensures the algorithm's ability to balance the discovery of new potential solutions with the refinement of promising ones, making it effective for tackling high-dimensional problems and complex landscapes with multiple local minima.

The basic EO algorithm generates new solutions, called echoes, based on a combination of directed reflection and random noise.

For each dimension d of the solution x_i, the updated value $echo_d$ is computed as:

$$echo_d = x_{i,d} + coef \cdot (x_{best,d} - x_{i,d}) + noise_d \tag{2}$$

Where:

- $coef \cdot (x_{best,d} - x_{i,d})$: A directed reflection term that moves the solution toward the best-known solution x_{best}.
- $coef$: Reflection factor
- $noise_d = random[-1, 1] \cdot (1 - currentDelay)$: A random noise term that attenuates as $currentDelay$ decreases over iterations.
- $currentDelay = initDelay - (initDelay - finalDelay) \cdot (\frac{iter}{iter_{max}})$: A decay factor that adjusts over the course of $iter_{max}$ iterations.

2.1. The echo memory technique

The memory technique introduces a unique mechanism to the EO algorithm by enabling the storage and reuse of information from previously evaluated solutions, significantly reducing computational overhead. Specifically, each evaluated solution is stored in an echo memory along with its corresponding objective function value. When a new solution is generated, the algorithm checks if a similar solution exists in memory, using a predefined tolerance level to determine similarity. If a match is found, the stored value is retrieved instead of recalculating the objective function. This feature is particularly beneficial in problems where evaluating solutions is computationally expensive. Moreover, the memory is dynamically updated with new, improved solutions, ensuring the algorithm's adaptability to evolving optimization scenarios. This mechanism enhances the algorithm's efficiency by focusing computational resources on exploring truly novel areas of the search space, rather than redundantly evaluating similar solutions.

In the memory-enhanced EO configuration, the algorithm stores previously evaluated solutions and their corresponding fitness values to avoid redundant computations. When a new solution echo is generated, the algorithm checks if a similar solution exists in the memory within a predefined tolerance (e: small number e.g. 0.4):

$$|echo_d - memory_{j,d}| < e \tag{3}$$

If a match is found:

- Retrieve $memoryFitness_j$.
- If $memoryFitness_j < f_i$, update the current solution and its fitness:
  - $x_i = echo$
  - $f_i = memoryFitness_j$

Similarity in the EO method is determined exclusively by the Euclidean distance between the coordinates of the candidate solution and those stored in memory. Similarity is eval-

uated by comparing the differences in the coordinate values for each dimension. If the absolute difference between the respective values is smaller than the predefined tolerance, the solution is considered similar to one already stored in memory. The calculation is based on the Equation 3, where the result is compared against the tolerance value e (Table 1). The objective function values are not used in the similarity determination process. The evaluation relies solely on geometric proximity, specifically the position of the solutions within the search space.

2.2. The approximate evaluations technique

The approximation technique enables the algorithm to quickly and efficiently estimate the quality of solutions, avoiding unnecessary computations for those unlikely to improve performance. In each iteration, approximate fitness values ($approxFitness$) are computed for the entire population based on the squared Euclidean distance of new solutions to the best-known solution, weighted by the current delay factor ($currentDelay$). These $approxFitness$ values are sorted, and a $cutoff$ threshold is determined corresponding to the (1−$approxThreshold$) quantile of the population. Solutions with $approxFitness$ above this cutoff are discarded without exact evaluation, thus accelerating convergence and reducing expensive objective function evaluations.

In the configuration with approximate evaluations, the algorithm uses a heuristic to estimate the quality of a solution before computing the objective function. The approximate fitness is calculated as:

$$approxFitness = \left(\sum_{d=1}^{n} (echo_d - x_{best,d})^2 \right) \cdot currentDelay \tag{4}$$

If: $approxFitness > approxThreshold \cdot f_i$:

then the solution is rejected without evaluating the objective function, and the algorithm proceeds to the next individual.

2.3. The overall algorithm

The overall algorithm of the method follows:

Algorithm 1 Pseudocode of EO

Input:
- NP: Population size
- $initDelay \in [0,1]$: Initial delay factor
- $finalDelay \in [0,1]$: Final delay factor
- $reflectionFactor \in [0,1]$: Reflection coefficient
- $iter_{max}$: Maximum iterations
- $approxThreshold$: Approximation threshold (e.g., 0.2 - 20%)
- $memory_{size}$: Memory capacity
- $maxNoEval$: Maximum allowed consecutive non-evaluations before clearing memory
- $technique$: {ECHO or ECHO+APP or ECHO+MEM or ECHO+APP+MEM}: Selected of technique
- SR: {Function Evaluations (FEs) or $iter_{max}$}
- $searchSpace$: Feasible solution bounds

Output:
- x_{best}: Optimal solution found, f_{best}: Corresponding fitness value

Initialization:
01: Initialize population $X = \{x_i | x_i\ U(searchSpace), i = 1, ..., NP\}$
02: Evaluate initial fitness $F = \{f = f(x_i) | i = 1, ..., NP\}$
03: Set (x_{best}, f_{best}) = $argmin_{(x_i, f_i)} f_i$
04: Initialize empty memory: $M = \varnothing$, $F_M = \varnothing$
05: $iter$ = 0
06: $noEvalCounter$ = 0

Main Optimization Loop:
07: While termination criteria (SR 1) not met do
08: $currentDelay = initDelay - (initDelay - finalDelay) \cdot (\frac{iter}{iter_{max}})$
09: if $technique$ in {ECHO+APP, ECHO+APP+MEM} then
10: for i=1 to NP do
11: for d = 1 to dim do
12: $echo_{i,d} = x_{i,d} + coef \cdot (x_{best,d} - x_{i,d}) + U(-1,1) \cdot (1 - currentDelay)$
13: $approxFitness_i = \left(\sum_{d=1}^{n} (echo_{i,d} - x_{best,d})^2\right) \cdot currentDelay$
14: end for
15: end for
16: sort $approxFitness$ array (ascending)
17: $index = floor(NP * (1 - approxThreshold))$
18: $cutOff = approxFitness_{index}$
19: end if
20: for i=1 to NP do
21: If technique NOT in {ECHO+APP, ECHO+APP+MEM} then
22: $echo_{i,d} = x_{i,d} + coef \cdot (x_{best,d} - x_{i,d}) + U(-1,1) \cdot (1 - currentDelay)$
23: else if
24: $echo = echo_i$
25: if $approxFitness > cutOff$ then
26: continue to next individual
27: end if
28: end if
29: end for
30: for d = 1 to dim do
31: $echo_d = clamp(echo_d, searchSpaceLower_d, searchSpaceUpper_d)$
32: end for
33: if $technique$ in {ECHO+MEM, ECHO+APP+MEM} then
34: if $\exists j : ||echo - M_j|| < \varepsilon$ then
35: if $F_{M_j} < f_i$ then
36: $x_i \leftarrow echo$
37: $f_i \leftarrow F_{M_j}$
38: UpdateBest(x_i,f_i)
39: end if
40 $noEvalCounter \leftarrow noEvalCounter$ + 1
41: if $noEvalCounter > maxNoEval$ then
42: $M \leftarrow \varnothing$, $F_M \leftarrow \varnothing$
43: $noEvalCounter \leftarrow 0$
44: end if
45: continue to next individual
46: end if
47: end if
48: $f_{echo} = f(echo)$
49: noEvalCounter $\leftarrow 0$
50: if $f_{echo} \leq f_i$ then
51: $x_i \leftarrow echo$
52: $f_i \leftarrow f_{echo}$
53: UpdateBest(x_i,f_i)
54: if $technique$ = technique in {ECHO+MEM, ECHO+APP+MEM} and $|M| < memory_{size}$ then
55: $M \leftarrow M \cup \{echo\}$
56: $F_M \leftarrow F_M \cup \{f_{echo}\}$
57: end if
58: end if
59: end for
60: $iter$ = $iter$+1
61: end While
62: return (x_{best}, f_{best})

The EO algorithm (Algorithm 1) begins by initializing its parameters, which include the population size NP, the initial delay factor $initDelay$, the final delay factor $finalDelay$, the reflection factor $coef$, the maximum number of iterations $iter_{max}$, the approximation threshold $approxThreshold$, and the maximum memory size $maxMemorySize$. A random population of solutions x_i is generated within the predefined bounds of the search space, where i ranges from 1 to NP. Each solution in the population is evaluated using the objective function to compute its fitness f_i. The best solution, referred to as x_{best}, and its fitness f_{best} are identified. An echo memory, echoMemory, is initialized as an empty structure, along with its corresponding memory fitness values.

The optimization process enters the main loop, which runs for a maximum of $iter_{max}$ iterations. At each iteration, the delay factor $currentDecay$ is computed as a linear interpolation between $initDelay$ and $finalDelay$, depending on the current iteration $iter$ relative to $iter_{max}$. For each solution x_i in the population, a new solution (echo) is generated dimension by dimension. The new value for each dimension $echo_d$ is calculated by adding three components: the reflection term, which directs the solution toward $x_{best,d}$, the noise term, which introduces randomness and decreases as $currentDecay$ reduces over time, and the original value of the dimension. Boundary correction is applied to ensure the new solution remains within valid limits.

After generating the echo, optional techniques may be applied to improve computational efficiency. If the approximation technique is enabled, the algorithm first computes the approximate fitness values ($approxFitness$) for all echoes in the current population, based on their squared Euclidean distance to the best solution x_{best}, scaled by the current delay factor. These $approxFitness$ values are sorted in ascending order, and a $cutoff$ threshold is determined corresponding to the (1−$approxThreshold$) quantile of the population. Echoes with $approxFitness$ greater than this $cutoff$ are discarded without further evaluation, and the algorithm proceeds to the next individual.

When memory reuse is enabled, previously evaluated solutions (echoes) are stored along with their fitness values. If a new echo matches an entry in the memory (within a small tolerance), the stored fitness is reused, avoiding redundant evaluation. To prevent excessive reliance on memory, the algorithm monitors the number of consecutive individuals that avoid evaluation via memory reuse. If this number exceeds a predefined limit ($maxNoEval$), the memory is cleared automatically, ensuring continued search diversity and avoiding stagnation.

The algorithm continues until a termination condition is met, at which point the best solution x_{best} is returned as the result of the optimization process.

3. Experimental setup and benchmark results

This section first introduces the benchmark functions selected for experimental evaluation, followed by a comprehensive analysis of the conducted experiments. The study systematically examines the various parameters of the algorithm to assess its reliability and effectiveness in different optimization scenarios. The complete parameter configurations used throughout these experiments are documented in Table 1.

The experimental evaluation was performed on a high-performance computing system featuring an AMD Ryzen 5950X processor and 128GB RAM, operating under Debian Linux. To ensure statistical reliability, each benchmark function was evaluated through 30 independent runs with randomized initial conditions. The implementation was developed in optimized ANSI C++ within the OPTIMUS framework [36], an open-source optimization platform available at https://github.com/itsoulos/GLOBALOPTIMUS (last accessed June 8, 2025). The complete parameter configuration for all methods is detailed in Table 1.

The results present the mean number of objective function evaluations across all trials. Parenthetical values denote the success rate in locating the global optimum, with omitted percentages indicating perfect (100%) convergence across all repetitions. In the experimental tables, the values marked in green indicate the best performance, corresponding to the lowest number of function calls, while those marked in blue are the standard deviation from the mean of the 30 iterations for each reference function. The experiments present tables that evaluate the performance of the ECHO method, along with its individual variants, in comparison with other well-known optimization techniques. The measurements were conducted on standard benchmark functions under specific experimental settings. All parameters and initializations not explicitly stated below follow the values defined in Table 1.

Table 1. Parameters and settings

PARAMETER	VALUE	EXPLANATION
NP	$100, 4 + \lfloor 3 \cdot \log(\text{dimension}) \rfloor$	Population
$iter_{max}$	500	Maximum number of iterations for all methods
$initialDelay$	0.9	Initial echo delay factor for EO
$finalDelay$	0.1	Final echo delay factor for EO
$coef$	0.5	Reflection factor towards best solution for EO
$approxThreshold$	0.4	Approximate threshold for EO
e	0.3	Echo memory tolerance for EO
$memory_{size}$	500	Memory size
$maxNoEval$	500	Maximum iterations before clearing memory
SR	$iter_{max}$ or Function Evaluations (FEs)	Stopping rule for all methods
c_1, c_2	1.494	Cognitive and Social coefficient for CLPSO
w	0,729	Inertia for CLPSO
F	0.5	Initial scaling factor for SaDE
CR	0.5	Initial crossover rate for SaDE

3.1. Real-World Problems

In this section, experiments were conducted with a focus on real-world optimization problems, selected from the set of functions listed in Table 2, which represent approximately 85% of the entire CEC2011 benchmark set and encompass both classical and contemporary challenges in global optimization. The experimental procedure was carefully designed to ensure strict comparability among the different algorithms under study. Specifically, a fixed population size of 100 samples, generated with a uniform distribution within the feasible domain of each problem, was used for all methods except CMA-ES. CMA-ES is a model-based stochastic optimization method that operates effectively with small, logarithmically increasing population sizes because it focuses on the progressive learning of the probability distribution. Instead of relying on broad random sampling, it leverages the learning dynamics across multiple generations. This approach conserves computational resources and leads to faster convergence in many practical problems. For CMA-ES, standard practice from the relevant literature was followed, setting the population size according to the formula $Np = 4 + \lfloor 3 \cdot \log(\text{dimension}) \rfloor$ to optimally leverage the algorithm's dynamics relative to problem dimensionality. All other parameters governing the operation of the algorithms including evolution coefficients, stopping rules, and acceptance criteria remained unchanged and were set according to the detailed configurations presented in Table 1, thereby ensuring maximum objectivity in the comparative assessment of results. Through this systematic approach, the experimental outcomes faithfully reflect the behavior of the methods on demanding real-world problems, enabling reliable conclusions to be drawn regarding their practical effectiveness.

Table 2. Real world problems CEC2011.

PROBLEM	FORMULA	Dim	BOUNDS
Parameter Estimation for Frequency-Modulated Sound Waves	$\min_{x\in[-6.4,6.35]6}\ f(x)=\frac{1}{N}\sum_{n=1}^{N}\left\|y(n;x)-y_{\text{target}}(n)\right\|^2$ $y(n;x)=x_0\sin\left(x_1 n+x_2\sin(x_3 n+x_4\sin(x_5 n))\right)$	6	$x_i\in[-6.4,6.35]$
Lennard-Jones Potential	$\min_{x\in\mathbb{R}^{3N-6}}\ f(x)=4\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left[\left(\frac{1}{r_{ij}}\right)^{12}-\left(\frac{1}{r_{ij}}\right)^{6}\right]$	30	$x_0\in(0,0,0)$ $x_1,x_2\in[0,4]$ $x_3\in[0,\pi]$ x_{3k-3} x_{3k-2} $x_i\in[-b_k,+b_k]$
Bifunctional Catalyst Blend Optimal Control	$\frac{dx_1}{dt}=-k_1x_1,\ \frac{dx_2}{dt}=k_1x_1-k_2x_2+k_3x_2+k_4x_3,$ $\frac{dx_3}{dt}=k_2x_2,\ \frac{dx_4}{dt}=-k_4x_4+k_5x_5,$ $\frac{dx_5}{dt}=-k_3x_2+k_6x_4-k_5x_5+k_7x_6+k_8x_7+k_9x_5+k_{10}x_7$ $\frac{dx_6}{dt}=k_8x_5-k_7x_6,\ \frac{dx_7}{dt}=k_9x_5-k_{10}x_7$ $k_i(u)=c_{i1}+c_{i2}u+c_{i3}u^2+c_{i4}u^3$	1	$u\in[0.6,0.9]$
Optimal Control of a Non-Linear Stirred Tank Reactor	$J(u)=\int_0^{0.72}\left[x_1(t)^2+x_2(t)^2+0.1u^2\right]dt$ $\frac{dx_1}{dt}=-2x_1+x_2+1.25u+0.5\exp\left(\frac{x_1}{x_1+2}\right)$ $\frac{dx_2}{dt}=-x_2+0.5\exp\left(\frac{x_1}{x_1+2}\right)$ $x_1(0)=0.9,\quad x_2(0)=0.09,\ t\in[0,0.72]$	1	$u\in[0,5]$
Tersoff Potential for model Si (B)	$\min_{\mathbf{x}\in\Omega}f(\mathbf{x})=\sum_{i=1}^{N}E(\mathbf{x}_i)$ $E(\mathbf{x}_i)=\frac{1}{2}\sum_{j\neq i}f_c(r_{ij})\left[V_R(r_{ij})-B_{ij}V_A(r_{ij})\right]$ where $r_{ij}=\|\mathbf{x}_i-\mathbf{x}_j\|, V_R(r)=A\exp(-\lambda_1 r)$ $V_A(r)=B\exp(-\lambda_2 r)$ $f_c(r)$: cutoff function with $f_c(r)$: angle parameter	30	$x_1\in[0,4]$ $x_2\in[0,4]$ $x_3\in[0,\pi]$ $x_i\in\left[\frac{4(i-3)}{4},4\right]$
Tersoff Potential for model Si (C)	$\min_{\mathbf{x}}\ V(\mathbf{x})=\sum_{i=1}^{N}\sum_{j>i}^{N}f_C(r_{ij})\left[a_{ij}f_R(r_{ij})+b_{ij}f_A(r_{ij})\right]$ $f_C(r)=\begin{cases}1, & r<R-D\\ \frac{1}{2}+\frac{1}{2}\cos\left(\frac{\pi(r-R+D)}{2D}\right), & \lvert r-R\rvert\le D\\ 0, & r>R+D\end{cases}$ $f_R(r)=A\exp(-\lambda_1 r)$ $f_A(r)=-B\exp(-\lambda_2 r)$ $b_{ij}=\left[1+(\beta^n)\zeta_{ij}^n\right]^{-1/(2n)}$ $\zeta_{ij}=\sum_{k\neq i,j}f_C(r_{ik})g(\theta_{ijk})\exp\left[\lambda_3^3(r_{ij}-r_{ik})^3\right]$	30	$x_1\in[0,4]$ $x_2\in[0,4]$ $x_3\in[0,\pi]$ $x_i\in\left[\frac{4(i-3)}{4},4\right]$
Spread Spectrum Radar Polly phase Code Design	$\min_{x\in X}\ f(x)=\max\{\lvert\varphi_1(x)\rvert,\lvert\varphi_2(x)\rvert,\ldots,\lvert\varphi_m(x)\rvert\}$ $X=\{x\in\mathbb{R}^n\mid 0\le x_j\le 2\pi,\ j=1,\ldots,n\}m=2n-1$ $\varphi_j(x)=\begin{cases}\sum_{k=1}^{n-j}\cos(x_k-x_{k+j}) & \text{for } j=1,\ldots,n-1\\ n & \text{for } j=n\\ \varphi_{2n-j}(x) & \text{for } j=n+1,\ldots,2n-1\end{cases}$ $\varphi_j(x)=\sum_{k=1}^{n-j}\cos(x_k-x_{k+j}),\quad j=1,\ldots,n-1$ $\varphi_n(x)=n,\ \varphi_{n+\ell}(x)=\varphi_{n-\ell}(x),\quad \ell=1,\ldots,n-1$	20	$x_j\in[0,2\pi]$
Transmission Network Expansion Planning	$\min\sum_{l\in\Omega}c_l n_l+W_1\sum_{l\in OL}\lvert f_l-\bar{f}_l\rvert+W_2\sum_{l\in\Omega}\max(0,n_l-\bar{n}_l)$ $Sf=g-d$ $f_l=\gamma_l n_l\Delta\theta_l,\quad \forall l\in\Omega$ $\lvert f_l\rvert\le\bar{f}_l n_l,\quad \forall l\in\Omega$ $0\le n_l\le\bar{n}_l,\quad n_l\in\mathbb{Z},\quad \forall l\in\Omega$	7	$0\le n_i\le\bar{n}_l$ $n_i\in\mathbb{Z}$
Electricity Transmission Pricing	$\min_x\ f(x)=\sum_{i=1}^{N_g}\left(\frac{C_i^{gen}}{P_i^{gen}}-R_i^{gen}\right)^2+\sum_{j=1}^{N_d}\left(\frac{C_j^{load}}{P_j^{load}}-R_j^{load}\right)^2$ $\sum_j GD_{i,j}+\sum_j BT_{i,j}=P_i^{gen},\quad \forall i$ $\sum_i GD_{i,j}+\sum_i BT_{i,j}=P_j^{load},\quad \forall j$ $GD_{i,j}^{max}=\min(P_i^{gen}-BT_{i,j},\ P_j^{load}-BT_{i,j})$	126	$GD_{i,j}\in[0,GD_{i,j}^{max}]$
Circular Antenna Array Design	$\min_{r_1,\ldots,r_6,\varphi_1,\ldots,\varphi_6}\ f(\mathbf{x})=\max_{\theta\in\Omega}AF(\mathbf{x},\theta)$ $AF(\mathbf{x},\theta)=\left\lvert\sum_{k=1}^{6}\exp\left(j\left[2\pi r_k\cos(\theta-\theta_k)+\varphi_k\frac{\pi}{180}\right]\right)\right\rvert$	12	$r_k\in[0.2,1]$ $\varphi_k\in[-180,180]$
Dynamic Economic Dispatch 1	$\min_{\mathbf{P}}\ f(\mathbf{P})=\sum_{t=1}^{24}\sum_{i=1}^{5}\left(a_iP_{i,t}^2+b_iP_{i,t}+c_i\right)$ $P_i^{\min}\le P_{i,t}\le P_i^{\max},\quad \forall i=1,\ldots,5,\ t=1,\ldots,24$ $\sum_{i=1}^{5}P_{i,t}=D_t,\quad \forall t=1,\ldots,24$ $P_{\min}=[10,20,30,40,50]$ $P_{\max}=[75,125,175,250,300]$	120	$P_i^{\min}\le P_{i,t}\le P_i^{\max}$
Dynamic Economic Dispatch 2	$\min_{\mathbf{P}}\ f(\mathbf{P})=\sum_{t=1}^{24}\sum_{i=1}^{9}\left(a_iP_{i,t}^2+b_iP_{i,t}+c_i\right)$ $P_i^{\min}\le P_{i,t}\le P_i^{\max},\quad \forall i=1,\ldots,5,\ t=1,\ldots,24$ $\sum_{i=1}^{5}P_{i,t}=D_t,\quad \forall t=1,\ldots,24$ $P_{\min}=[150,135,73,60,73,57,20,47,20]$ $P_{\max}=[470,460,340,300,243,160,130,120,80]$	216	$P_i^{\min}\le P_{i,t}\le P_i^{\max}$
Static Economic Load Dispatch (1,2,3,4,5)	$\min_{P_1,\ldots,P_{N_G}}\ F=\sum_{i=1}^{N_G}f_i(P_i)$ $f_i(P_i)=a_iP_i^2+b_iP_i+c_i,\quad i=1,2,\ldots,N_G$ $f_i(P_i)=a_iP_i^2+b_iP_i+c_i+\lvert e_i\sin(f_i(P_i^{\min}-P_i))\rvert$ $P_i^{\min}\le P_i\le P_i^{\max},\quad i=1,2,\ldots,N_G$ $\sum_{i=1}^{N_G}P_i=P_D+P_L$ $P_L=\sum_{i=1}^{N_G}\sum_{j=1}^{N_G}P_iB_{ij}P_j+\sum_{i=1}^{N_G}B_{0i}P_i+B_{00}$ $P_i-P_i^0\le UR_i\quad P_i^0-P_i\le DR_i$	6 13 15 40 140	See Technical Report of CEC2011

Table 3. Comparison of different versions of EO after 1.5e+5 FEs

1.5e+5 FEs	ECHO			ECHO+APP			ECHO+MEM			ECHO+APP+MEM		
FUNCTION	**BEST**	**MEAN**	**SD**	**BEST**	**MEAN**	**SD**	**BEST**	**MEAN**	**SD**	**BEST**	**MEAN**	**SD**
Parameter Estimation for Frequency-Modulated Sound Waves	0.242608779	0.285913022	0.022991904	0.265445773	0.296899245	0.015752369	0.2205434	0.289343164	0.020064458	0.258850589	0.297607715	0.014622033
Lennard-Jones Potential	-15.82941908	-12.20055451	1.743688768	-14.88637666	-11.5223685	1.330953508	-16.7597284	-12.14408133	2.181638831	-15.05983032	-11.64845534	1.581120548
Bifunctional Catalyst Blend Optimal Control	-0.000286591	-0.000286591	9.361236018e-11	-0.000286591	-0.00028659	5.70546361e-10	-0.000286591	-0.000286591	1.818112359e-10	-0.000286591	-0.000286565	5.283017614e-08
Optimal Control of a Non-Linear Stirred Tank Reactor	80.11262009	10805.99696	22630.65272	5.057454518	63555.61096	126770.9699	80.11262009	10635.34871	22582.30421	15.87791182	93426.02939	132248.6518
Tersoff Potential for model Si (B)	-27.33646273	-24.93278552	1.277547861	-26.83442298	-24.76088117	1.279543141	-26.31737322	-17.24953496	8.243696725	-24.79758324	-21.14424651	1.736579068
Tersoff Potential for model Si (C)	-31.48064458	-28.96566368	1.440816542	-31.2201117	-28.68971002	1.521525081	-30.13871112	-22.36198819	6.118197162	-29.61735238	-25.77718772	1.91811245
Spread Spectrum Radar Polly phase Code Design	0.446845407	0.794854574	0.206703444	0.561938657	0.79943723	0.183288986	0.612722571	1.068238166	0.284502043	0.776849133	1.200405436	0.286679334
Transmission Network Expansion Planning	250.00	250.00	0	250.00	250.00	0	250.00	250.00	0	250.00	250.00	0
Electricity Transmission Pricing	13775940.18	13776968.1	481.2205121	13776456.35	13777314.04	465.7720051	13777516.89	41308454.02	28013501.01	13777634.59	37864312.86	20803333.64
Circular Antenna Array Design	0.019998298	0.173367157	0.079463074	0.042321976	0.196041942	0.058719011	0.019998298	0.173367157	0.079463074	0.03199129	0.186723215	0.071619579
Dynamic Economic Dispatch 1	464634158.9	479810672.1	10428912.68	466831874.1	489234993.9	10947881.05	464634158.9	479810672.1	10428912.68	467796615.8	487002371.3	9846702.86
Dynamic Economic Dispatch 2	52463484.01	73536820.82	13625891.45	60130998.54	80833293.64	16987027.59	52463484.01	73536820.82	13625891.45	52432134.97	81250958.82	13916144.34
Static Economic Load Dispatch 1	6603.922422	10413.46457	7716.333599	6770.885293	17860.54451	15431.62363	6603.922422	10219.0443	7772.321644	6574.773227	20028.69498	30081.80949
Static Economic Load Dispatch 2	19338.19222	35064.4314	19829.76015	19672.98446	61011.19781	59900.02455	19338.19222	35064.4314	19829.76015	20997.28463	66986.94561	48651.49589
Static Economic Load Dispatch 3	470350624.6	470581625.1	182643.2251	470436676.5	470596641	141941.5706	470350624.6	470581625.1	182643.2251	470339617	70652010.5	171247.2501
Static Economic Load Dispatch 4	138516.9427	331608.7229	125778.3196	129514.7094	438466.1401	190314.2651	138516.9427	331608.7229	125778.3196	152983.4568	421286.3464	176238.5485
Static Economic Load Dispatch 5	8116264799	8295168707	193563112.7	8110948545	8310981439	188394364.1	8116264799	8295168707	193563112.7	8107691278	8314708744	209989303.2

3.3. Exploration and exploitation

In our work, we chose to examine this balance through a set of indicators: Initial Population Diversity (IPD), Final Population Diversity (FPD), Average Exploration Ratio (AER), Median Exploration Ratio (MER), and Average Balance Index (ABI) which, although based on population diversity, are designed to reflect both the temporal dynamics of exploration (through changes in diversity over iterations) and the tendency toward exploitation (through final population convergence). However, we acknowledge that the investigation of direct evaluation methods, such as attraction basin mapping or monitoring the concentration of solutions around local/global optima, could provide more meaningful insights into the search process and enhance the interpretation of the results. This represents an important direction for future extension of the present study.

The metrics presented in Tables 4 are related to measuring and monitoring the balance between exploration and exploitation during the execution of the ECHO+APP+MEM method. Their calculation is based on changes in population diversity as well as the behavior of the algorithm across iterations.

The IPDmetric measures the diversity of the population at the beginning of the optimization process and is calculated as the average Euclidean distance between all individuals in the initial population:

$$IPD = \frac{2}{NP(NP-1)} \sum_{i=1}^{NP-1} \sum_{j=i+1}^{NP} d(x_i, x_j) \tag{5}$$

where $d(x_i, x_j)$ is the Euclidean distance between solutions x_i and x_j, and NP is the population size.

The FPD is computed in the same way but applied to the final population at the end of the execution.

The AER represents the average exploration ratio throughout all iterations. It is defined as:

$$AER = \frac{1}{G} \sum_{g=1}^{iter_{max}} \frac{IPD_g}{IPD_1} \tag{6}$$

where $iter_{max}$ is the total number of generations, IPD_g is the population diversity at iteration g, and IPD_1 and is the initial diversity.

The MER is the median of the exploration ratio values across all generations:

$$MER = \text{median}\left(\frac{IPD_g}{IPD_1}\right), \quad \text{for } g = 1, \ldots, iter_{max} \tag{7}$$

The ABI evaluates the overall balance between exploration and exploitation. It results from a weighted combination of AER and FPD (or other exploitation-related indicators), often expressed as:

$$ABI = \frac{AER}{AER + \epsilon} \cdot \left(1 - \frac{FPD}{IPD}\right) \tag{8}$$

where ϵ is a very small constant to avoid division by zero. The ABI tends to values close to 0.5 when exploration and exploitation are well balanced.

The indicator values presented by the EO method reveal important insights into its behavior across various problems. The IPD shows significant variation depending on the nature of the problem, reflecting the complexity and breadth of the search space explored by the algorithm. In contrast, the FPD is generally lower, indicating that the population has converged to more specific regions, as expected during the optimization process.

Table 4. Balance between exploration and exploitation of the ECHO+APP+MEM method in each benchmark function after 1.5e+5 FEs

FUNCTION	BEST	MEAN	SD	IPD	FDP	AER	ABI
Parameter Estimation for Frequency-Modulated Sound Waves	0.242608779	0.285913022	0.022991904	8.5901	0.85794	0.0076	0.50116
Lennard-Jones Potential	-15.82941908	-12.53895704	-12.20055451	13.91823	2.79253	0.0046	0.49556
Bifunctional Catalyst Blend Optimal Control	-0.000286591	-0.000286591	9.361236018e-11	0.0743	0	18.79141	0.49112
Optimal Control of a Non-Linear Stirred Tank Reactor	80.11262009	10805.99696	22630.65272	49184124.11	17331.72035	2.15492	0.49106
Tersoff Potential for model Si (B)	-27.33646273	-24.93278552	1.277547861	5.52126	0.85541	0.00528	0.49456
Tersoff Potential for model Si (C)	-31.48064458	-28.96566368	1.440816542	5.52126	0.76634	0.0054	0.49459
Spread Spectrum Radar Polly phase Code Design	0.446845407	0.794854574	0.206703444	8.06994	1.59855	0.00446	0.49459
Transmission Network Expansion Planning	250.00	250.00	0	0.96619	0.32347	0.00203	0.4981
Electricity Transmission Pricing	13775940.18	13776968.1	481.2205121	6.50993	0.97945	0.00637	0.50563
Circular Antenna Array Design	0.019998298	0.173367157	0.079463074	245.62332	21.0065	0.01046	0.49475
Dynamic Economic Dispatch 1	464634158.9	479810672.1	10428912.68	530.86265	72.92171	0.0076	0.50465
Dynamic Economic Dispatch 2	52463484.01	73536820.82	13625891.45	890.76948	124.84328	0.00726	0.50468
Static Economic Load Dispatch 1	6603.922422	10413.46457	7716.333599	141.01729	19.33598	0.00543	0.49608
Static Economic Load Dispatch 2	19338.19222	35064.4314	19829.76015	238.24613	38.83075	0.00855	0.50295
Static Economic Load Dispatch 3	470350624.6	470581625.1	182643.2251	218.59546	8.48096	0.0143	0.50117
Static Economic Load Dispatch 4	138516.9427	331608.7229	125778.3196	410.06721	59.70221	0.00816	0.50229
Static Economic Load Dispatch 5	8116264799	8295168707	193563112.7	750.05361	104.59389	0.00759	0.50463

The Average AER and the MER remain at relatively low levels, suggesting that the method performs gradual and controlled changes in population diversity without extreme fluctuations, which promotes search stability. Finally, the ABI hovers very close to 0.5 across all problems, indicating that the method maintains a stable and healthy balance between exploration and exploitation phases.

Overall, these data confirm EO's ability to adapt according to the problem, preserving diversity when needed and focusing on promising regions, resulting in reliable and efficient performance across a wide range of optimization problems.

3.4. Parameters Sensitivity

The sensitivity analysis of the parameters, as implemented in this work based on the methodology of Lee et al. [37] [lee], constitutes a fundamental tool for assessing the behavior and reliability of optimization algorithms. The results presented derive from a series of organized experiments, where each main parameter of the method (reflectionFactor, memoryTolerance, approximationThreshold) is varied over specific discrete values, while all other parameters are kept fixed. For each parameter combination, the method is repeatedly executed 270 times to obtain statistically valid conclusions regarding the mean, minimum, and maximum performance. The aim of this approach is threefold: (a) to identify the parameters that significantly affect the method's behavior, (b) to quantify the magnitude of this effect, and (c) to determine whether the method is robust or sensitive to parameter changes, thus documenting both its reliability and its ability to generalize to different problems.

Table 5. Sensitivity analysis of the method parameters for the **Frequency Modulated Sound Waves** problem

FrequencyModulated Sound Waves	Value	Mean	Min	Max	Main range
Reflection Factor	0.2	0.27878	0.22755	0.33379	0.01523
	0.5	0.28761	0.2097	0.33962	
	0.8	0.294	0.22323	0.3445	
Memory Tolerance	0.01	0.28852	0.22924	0.33989	0.00438
	0.2	0.28774	0.22357	0.33806	
	0.4	0.28413	0.2097	0.3445	
Approximation Threshold	0.1	0.28762	0.2097	0.33445	0.00174
	0.3	0.28588	0.21281	0.33989	
	0.6	0.28689	0.22323	0.3445	

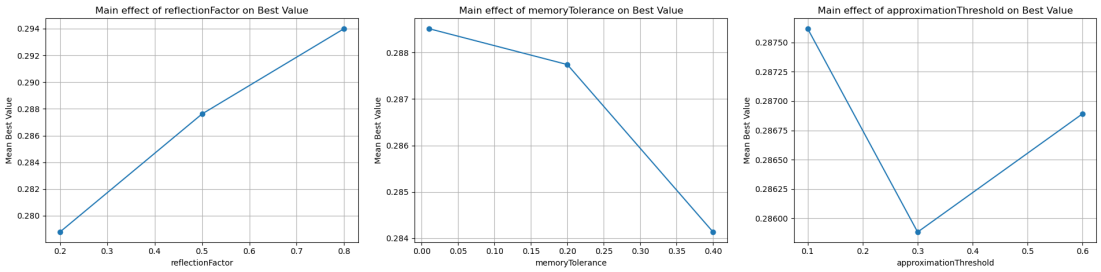

Figure 1. Graphical representation of Reflection factor, Memory Tolerance and Approximation Threshold for the **Frequency Modulated Sound Waves** problem

Table 6. Sensitivity analysis of the method parameters for the **Lennard-Jones Potential** problem

Lennard Jones Potential	Value	Mean	Min	Max	Main range
Reflection Factor	0.2	-13.5131	-19.36124	-10.25016	3.03479
	0.5	-12.83115	-19.61677	-8.00412	
	0.8	-10.47831	-17.01847	-6.01231	
Memory Tolerance	0.01	-12.21917	-19.61677	-6.97152	0.08831
	0.2	-12.29592	-19.36124	-6.21915	
	0.4	-12.30747	-18.87546	-6.01231	
Approximation Threshold	0.1	-12.04052	-18.87546	-6.01231	0.40208
	0.3	-12.33945	-17.36149	-6.64021	
	0.6	-12.44259	-19.61677	-6.21915	

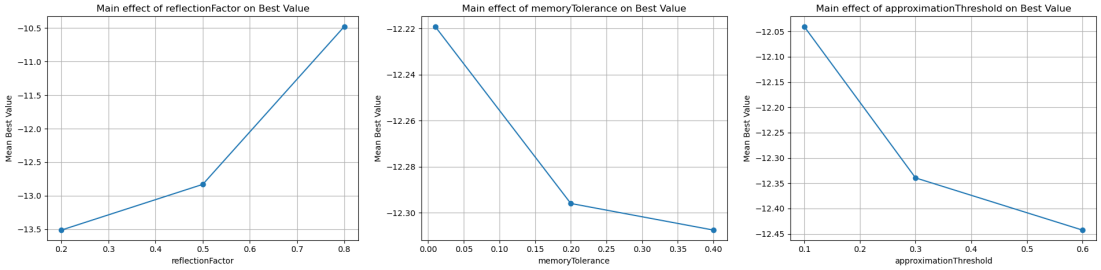

Figure 2. Graphical representation of Reflection factor, Memory Tolerance and Approximation Threshold for the **Lennard-Jones Potential** problem

Table 7. Sensitivity analysis of the method parameters for the **Dynamic Economic Dispatch 1** problem

Dynamic Economic Dispatch 1	Value	Mean	Min	Max	Main range
Reflection Factor	0.2	498709776.79	477135686.86	525130205.59	13831927.78
	0.5	487372163.96	463051832.53	525005642.63	
	0.8	484877849.01	457195992.98	523239555.02	
Memory Tolerance	0.01	490419307.37	465337023.58	525130205.59	175347.94
	0.2	490243959.43	463051832.53	525005642.63	
	0.4	490296522.97	457195992.98	523251571.66	
Approximation Threshold	0.1	490754333.32	465337023.58	523251571.66	851763.73
	0.3	489902569.59	457195992.98	523611650.24	
	0.6	490302886.86	463051832.53	525130205.59	

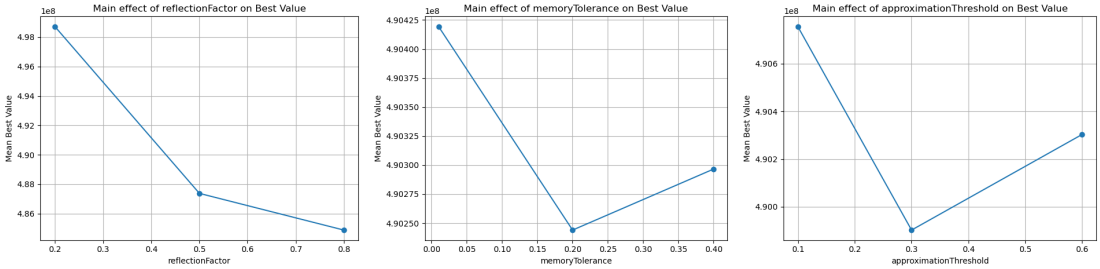

Figure 3. Graphical representation of Reflection factor, Memory Tolerance and Approximation Threshold for the **Dynamic Economic Dispatch 1** problem

Table 8. Sensitivity analysis of the method parameters for the **Static Economic Load Dispatch 1** problem

Static Economic Load Dispatch 1	Value	Mean	Min	Max	Main range
Reflection Factor	0.2	7839.71673	6487.28776	51584.75127	5235.09
	0.5	13074.8164	6557.19773	713773.3893	
	0.8	11316.61253	6565.00778	169504.9255	
Memory Tolerance	0.01	8716.80378	6538.33345	60471.06008	5531.27
	0.2	14248.08303	6557.19773	713773.3893	
	0.4	9266.25885	6487.28776	111070.1477	
Approximation Threshold	0.1	10979.50246	6487.28776	355992.0461	3084.24
	0.3	9083.6991	6569.6912	79594.33844	
	0.6	12167.9441	6557.19773	713773.3893	

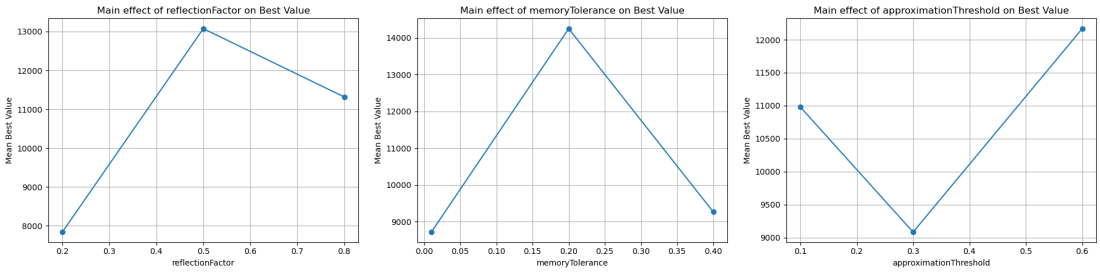

Figure 4. Graphical representation of Reflection factor, Memory Tolerance and Approximation Threshold for the **Static Economic Load Dispatch 1** problem

For each problem, a series of tables 5 - 8 and Figures 1 - 4, with detailed results for each parameter is presented. Each row of a table corresponds to a specific tested value of the parameter. The columns display the mean best value (Mean Best), the minimum value (Min), and the maximum value (Max). The "Main range" indicator summarizes the extent of the parameter's impact on the algorithm's performance, calculated as the difference between the highest and lowest mean values observed for that parameter.

The quantitative analysis of these indicators provides valuable insight into which features of the method have the greatest impact on the optimization process. For example, in

the Lennard Jones Potential problem, the parameter reflectionFactor has a main effect range of 3.03, indicating that changing this parameter results in significant performance variation. Conversely, in cases where the main effect range is very small (such as approximationThreshold in Frequency Modulated Sound Waves with a value of 0.00174), the parameter exhibits low sensitivity, which indicates that the method is stable with respect to changes in this parameter. It is also noteworthy that parameter sensitivity is not constant across problems. For instance, reflectionFactor has moderate or negligible effect on Frequency Modulated Sound Waves, but becomes a particularly important parameter for Lennard Jones Potential and Static Economic Load Dispatch 1, as reflected in the corresponding ranges. This behavior aligns with optimization theory, which states that each problem has a "landscape" that interacts differently with the exploration and exploitation mechanisms of each algorithm.

On the other hand, parameters such as memoryTolerance and approximationThreshold often have lower main effect ranges. This can be interpreted as an indication that the memory process and the evaluation approach are relatively robust for this particular set of problems, and/or that the selected values for these parameters do not differ enough to cause substantial changes in performance.

Sensitivity analysis clearly highlights the parameters that should be targeted during method optimization, while also indicating those that can be set using general rules without significant risk of performance loss.

3.5. Comparison of ECHO method with others

The analysis of the results presented in Tables 9, 10, and 11 was based on a rigorously standardized experimental protocol to ensure the reliability and comparability of the methods. Specifically, for Table 9, the termination criterion was strictly set to a fixed number of function evaluations (1,500,000), providing all methods with equal opportunities for exploration in the solution space. For each method, the best value (i.e., the lowest objective function value found) and the mean value over 30 independent runs were recorded, thus capturing both the maximum and average performance in a statistically substantiated manner. Subsequently, in Table 10, the methods were ranked for each problem according to their performance, with 1 corresponding to the best and 5 to the least effective performance among the five methods. For drawing overall conclusions, Table 11 presents the total sum of the ranks obtained by each method for both the best and mean values, and this aggregate score was divided by the total number of positions (i.e., the number of methods multiplied by two, to account for both criteria), providing a composite measure for comparing overall performance. Thus, the smallest average total score indicates the most effective method, enabling an objective evaluation of the comparative superiority of each approach across the full set of problems.

Table 9. Algorithms' Comparison Based on Best and Mean after 1.5e+5 FEs

Problem	**CLPSO** [38] **Best**	**CLPSO Mean**	CLPSO SD	**SaDE** [39] **Best**	**SaDE Mean**	SaDE SD	**jDE** [40] **Best**	**jDE Mean**	jDE SD	**-ES** [41] **Best**	**-ES Mean**	-ES SD	**EO Best**	**EO Mean**	EO ST
Parameter Estimation for Frequency Modulated Sound Waves	0.1314837477	0.2124981688	0.030223102	0.1899428536	0.2025566839	0.009271896	0.116157541	0.146008756	0.035009569	0.18160915970	0.256863966	0.044727545	0.242608779	0.285913022	0.022991904
Lennard-Jones Potential	-13.43649135	-10.25073403	1.02903617	-24.86870825	-22.6693403	1.127265561	-29.98126575	-27.49258505	1.235083397	-28.42253189	-25.78783328	2.27119571	-15.82941908	-12.20055451	1.743688768
Bifunctional CatalystBlend Optimal Control	-0.000286591	-0.000286591	1.157726295e-16	-0.000286591	-0.000286591	5.513684428e-20	-0.000286591	-0.000286591	5.513684428e-20	-0.000286591	-0.000286591	5.513684428e-20	-0.000286591	-0.000286591	9.361236018e-11
Optimal Control of a Non-Linear Stirred Tank Reactor	0.3903767228	0.3903767228	0	0.3903767228	0.3903767228	0	0.390376723	0.390376723	0	0.3903767228	0.390376723	0	0.3903767228	0.390376723	0
Tersoff Potential for model Si (B)	-28.23544117	-26.18834522	1.05654251	-3.107773136	25.4711091	16.7202543	-13.51157064	-3.983690794	6.666047747	-29.26244222	-27.5889735	1.040646284	-27.33646273	-24.93278552	1.277647861
Tersoff Potential for model Si (C)	-30.85200257	-28.87349048	0.988024149	-11.60719468	22.08963599	18.5809093	-18.76214649	-8.506037168	5.543190141	-33.19699356	-31.79270914	0.828194234	-31.48064458	-28.96566368	1.440816542
Spread Spectrum RadarPolly phaseCode Design	1.085334991	1.343956153	0.148708837	1.536501579	2.150881715	0.198607499	1.525870558	1.812042166	0.171213339	0.01	0.171988666	0.137892008	0.446845407	0.794854574	0.206703444
Transmission Network Expansion Planning	250	250	0	250	250	0	250	250	0	250	250	0	250	250	0
Electricity Transmission Pricing	13,775,010.10	13,775,395.07	222.9723613	23,481,009.86	30,034,934.81	3,264,767.4	14,020,627.84	14,820,953.78	276,142.5345	13,775,841.77	13,787,550.18	6136.744382	13,775,640.18	13,776,968.1	481.2205121
Circular Antenna Array Design	0.006933401045	0.05181551798	0.070674314	0.02142329927	0.03892428051	0.008183211	0.006820072	0.017657998	0.022383475	0.007204797576	0.008635655364	0.000917821	0.019998298	0.173367157	0.079463074
Dynamic Economic Dispatch 1	428,607,927.60	435,250,914.50	2973190.125	968,042,312.10	1,034,679,775.00	25,667,290.12	968,042,312.1	1,034,393,036	25,445,935.78	88,285.60	102,776.71	6688.08697	464,634,158.9	479,810,672.1	10,428,912.68
Dynamic Economic Dispatch 2	33,031,590.31	53,906,147.38	8,492,239.111	845,287,898.30	913,715,793.20	30,667,287.54	340,091,475.3	397,471,715.1	37,259,947.14	502,699.42	577,720.15	193,951.4891	52,463,484.01	73,536,820.82	13625891.45
Static Economic Load Dispatch 1	6554.67	7668.33	1245.137667	16,877.92	101,588.39	81,105.92078	6163.749006	6778.527028	3004.59066	6657.61	415,917.46	68,544.4983	6603.922422	10,413.46457	7716.333599
Static Economic Load Dispatch 2	19,030.36	20,699	2922.047235	2,600,565.21	9,329,466.81	4,019,053.284	1,161,578.904	3,671,587.605	1,542,286.275	763,001.22	1,425,815.44	377,126.8219	19,338.19222	35,064.4314	19,829.76015
Static Economic Load Dispatch 3	470,192,288.30	470,294,703.20	57,822.41621	478,069,615.30	541,898,763.00	20,128,777.18	471,058,115.8	471,963,142.3	529,633.389	470,023,232.30	470,023,232.30	1.848771369e-07	470,350,624.6	4,705,81625.1	182,643.2251
Static Economic Load Dispatch 4	884,980.56	1,423,887.36	285,794.4518	14,170,362.58	106,749,078.50	73,147,979.72	6,482,592.714	17,527,314.24	5,306,489.46	476,053.52	2,925,852.94	1,268,161.817	138,516.9427	331,608.7229	125,778.3196
Static Economic Load Dispatch 5	8,105,947,615.00	8,110,924,071.00	4,422,895.726	1.312720405e+10	1.354375465e+10	213,865,059.7	8,453,090,778	8,459,337,082	2,874,979.192	8,072,077,963.00	8,084,017,791.00	4,623,617.36	8,076,264,799	8,295,168,707	193,563,112.7

Table 10. Detailed Ranking of Algorithms Based on Best and Mean after 1.5e+5 FEs

Problem	CLPSO best	CLPSO Mean	SaDE best	SaDE Mean	jDE Best	jDE Mean	-ES best	-ES Mean	EO Best	EO Mean
Parameter Estimation for Frequency-Modulated Sound Waves	2	3	4	2	1	1	3	4	5	5
Lennard-Jones Potential	5	5	3	3	2	2	1	1	4	4
BifunctionalCatalyst Blend Optimal Control	1	1	1	1	1	1	1	1	1	1
Optimal Control of a Non-Linear Stirred Tank Reactor	1	1	1	1	1	1	1	1	1	1
Tersoff Potential for model Si (B)	2	2	5	5	4	4	1	1	3	3
Tersoff Potential for model Si (C)	3	3	5	5	4	4	1	1	2	2
Spread Spectrum Radar Polly phaseCode Design	3	3	5	5	4	4	1	1	2	2
Transmission Network Expansion Planning	1	1	1	1	1	1	1	1	1	1
Electricity Transmission Pricing	1	1	5	5	4	4	3	3	2	2
Circular Antenna Array Design	2	4	5	3	1	2	3	1	4	5
Dynamic Economic Dispatch 1	2	2	4	5	5	4	1	1	3	3
Dynamic Economic Dispatch 2	2	2	5	5	3	4	1	1	4	3
Static Economic Load Dispatch 1	2	2	5	5	1	1	4	4	3	3
Static Economic Load Dispatch 2	1	1	5	5	4	4	3	3	2	2
Static Economic Load Dispatch 3	2	2	5	5	4	4	1	1	3	3
Static Economic Load Dispatch 4	3	3	5	5	4	4	2	2	1	1
Static Economic Load Dispatch 5	3	2	5	5	4	4	1	1	2	3
TOTAL	**36**	**38**	**69**	**66**	**48**	**49**	**29**	**28**	**43**	**44**

Table 11. Comparison of Algorithms and Final Ranking

Problem	Best	Mean	Overall	Average	Rang
-ES	29	28	57	1.5833	**1**
CLPSO	36	38	74	2.0555	**2**
EO	43	44	87	2.4166	**3**
jDE	48	49	97	2.6944	**4**
SaDE	69	66	135	3.75	**5**

In the comparative assessment conducted in this subsection, the basic form of the EO was benchmarked against other modern and internationally recognized evolutionary algorithms such as SaDE, jDE, CLPSO, and CMA-ES, which serve as the state-of-the-art reference for real-world global optimization problems. The selection of these algorithms ensures a fair and realistic comparison, as all are noted for their stability and proven effectiveness in a wide range of practical applications and complex problems, as also reflected in the CEC2011 benchmark. The EO was implemented in its pure form, without the use of additional memory or surrogate evaluation mechanisms, thus highlighting the intrinsic ability of the method to address high-dimensional and complex problems transparently.

The analysis of the results indicates that EO remains highly competitive compared to the other examined methods across a substantial portion of the tested problems. It achieves comparable best and mean values, while exhibiting reduced variability between independent runs a feature that reflects its stable performance and reliability in real-world applications. Although in certain cases small deviations appear in favor of competing algorithms, particularly in highly specialized problem landscapes or in regions where intensive local exploitation is advantageous, the overall findings confirm the robustness and effectiveness of EO. The uniform use of a fixed population size (100 samples, except for CMA-ES), identical stopping criteria and parameters (as specified in Table 1), ensure absolute objectivity and fairness in the evaluation framework.

Consequently, EO proves to be a highly promising alternative among contemporary global optimization algorithms, highlighting its potential for future development and adoption in real-world applications, where the balance between speed, reliability, and implementation simplicity remains a core requirement.

3.6. The Echo method and neural networks

An additional experiment was conducted, where the ECHO+APP+MEM method was used to train artificial neural networks [43,44], by minimizing the so - called training error defined as:

$$E\left(N\left(\overrightarrow{x},\overrightarrow{w}\right)\right)=\sum_{i=1}^{M}\left(N\left(\overrightarrow{x}_i,\overrightarrow{w}\right)-y_i\right)^2 \tag{9}$$

In this equation the function $N\left(\overrightarrow{x},\overrightarrow{w}\right)$ represents the artificial neural network which is applied on a vector $\overrightarrow{x}$ and the vector $\overrightarrow{w}$ denotes the parameter vector of the neural network. The set $\left(\overrightarrow{x_i},y_i\right),\ i=1,...,M$ represents the training set of the objective problem and the values y_i are the expected outputs for each pattern $\overrightarrow{x_i}$.

To validate the ECHO+APP+MEM method, an extensive collection of classification datasets was employed, sourced from various publicly available online repositories. These datasets were obtained from:

1. The UCI database, https://archive.ics.uci.edu/(accessed on 5 July 2025)[45]
2. The Keel website, https://sci2s.ugr.es/keel/datasets.php(accessed on 5 July2025)[46].
3. The Statlib URL https://lib.stat.cmu.edu/datasets/index(accessed on 5 July 2025).

The experiments were conducted using the following datasets:

1. **Appendictis** which is a medical dataset [47].
2. **Alcohol**, which is dataset regarding alcohol consumption [48].
3. **Australian**, which is a dataset produced from various bank transactions [49].
4. **Balance** dataset [50], produced from various psychological experiments.
5. **Circular** dataset, which is an artificial dataset.
6. **Cleveland**, a medical dataset which was discussed in a series of papers [51,52].
7. **Dermatology**, a medical dataset for dermatology problems [53].
8. **Ecoli**, which is related to protein problems [54].
9. **Glass** dataset, that contains measurements from glass component analysis.
10. **Haberman**, a medical dataset related to breast cancer.
11. **Hayes-roth** dataset [55].
12. **Heart**, which is a dataset related to heart diseases [56].
13. **HeartAttack**, which is a medical dataset for the detection of heart diseases
14. **Housevotes**, a dataset which is related to the Congressional voting in USA [57].
15. **Ionosphere**, a dataset that contains measurements from the ionosphere [58,59].
16. **Liverdisorder**, a medical dataset that was studied thoroughly in a series of papers[60, 61].
17. **Lymography** [62].
18. **Mammographic**, which is a medical dataset used for the prediction of breast cancer [63].
19. **Parkinsons**, which is a medical dataset used for the detection of Parkinson's disease [64,65].
20. **Pima**, which is a medical dataset for the detection of diabetes[66].
21. **Popfailures**, a dataset related to experiments regarding climate [67].
22. **Regions2**, a medical dataset applied to liver problems [68].
23. **Saheart**, which is a medical dataset concerning heart diseases[69].
24. **Segment** dataset [70].
25. The **Sonar** dataset, related to sonar signals [71].
26. **Statheart**, a medical dataset related to heart diseases.
27. **Student**, which is a dataset regarding experiments in schools [72].
28. **Transfusion**, which is a medical dataset [73].
29. **Wdbc**, which is a medical dataset regarding breast cancer [74,75].
30. **Wine**, a dataset regarding measurements about the quality of wines [76,77].
31. **EEG**, which is dataset regardingEEG recordings [78,79]. From this dataset the following cases were used: Z_F_S, ZO_NF_S, and ZONF_S.
32. **Zoo**, which is a dataset regarding animal classification [80] .

DATASET	GENETIC	ECHO	ECHO+APP	ECHO+MEM	ECHO+APP+MEM
Appendicitis	18.10%	22.07%	22.47%	22.60%	22.77%
Alcohol	39.57%	18.91%	17.24%	17.77%	16.96%
Australian	32.10%	22.34%	22.34%	22.91%	23.16%
Balance	8.97%	7.10%	7.10%	7.22%	7.23%
Circular	5.99%	4.48%	4.48%	4.37%	4.37%
Cleveland	51.60%	45.74%	45.74%	46.60%	46.60%
Dermatology	30.58%	9.34%	9.34%	11.51%	11.51%
Ecoli	54.67%	45.89%	45.89%	48.94%	48.94%
Fert	28.50%	24.40%	24.13%	26.50%	24.57%
Glass	58.30%	50.24%	50.24%	49.43%	49.22%
Haberman	28.66%	28.71%	28.71%	28.80%	28.80%
Hayes-roth	56.18%	35.21%	35.21%	37.46%	37.46%
Heart	28.34%	18.10%	18.10%	19.00%	19.02%
HeartAttack	29.03%	20.06%	20.06%	20.15%	20.15%
Housevotes	6.62%	7.58%	7.68%	7.20%	7.10%
Ionosphere	15.14%	14.53%	14.53%	15.03%	15.08%
Liverdisorder	31.11%	31.82%	31.82%	32.48%	32.48%
LYMOGRAPHY	28.42%	24.60%	24.60%	26.02%	25.31%
Mammographic	19.88%	17.44%	17.45%	17.36%	17.36%
Parkinsons	18.05%	15.04%	14.95%	15.74%	15.21%
Pima	32.19%	26.34%	26.34%	26.78%	26.78%
Popfailures	5.94%	6.88%	6.88%	6.84%	6.84%
Regions2	29.39%	28.21%	28.21%	30.07%	30.12%
Saheart	34.86%	32.71%	32.91%	32.46%	32.46%
Segment	57.72%	14.82%	14.82%	16.91%	16.87%
Sonar	22.40%	19.98%	20.52%	20.37%	20.07%
Spiral	48.66%	41.91%	41.91%	42.24%	42.24%
STATHEART	27.25%	19.45%	19.45%	20.24%	20.24%
Student	5.61%	5.84%	5.84%	5.98%	5.98%
Transfusion	24.87%	23.70%	23.70%	24.09%	24.09%
Wdbc	8.56%	3.92%	3.92%	4.72%	4.60%
Wine	19.20%	10.45%	10.45%	12.84%	13.88%
Z_F_S	10.73%	8.48%	8.48%	8.71%	7.89%
ZO_NF_S	21.54%	5.15%	5.15%	5.61%	5.99%
ZONF_S	4.36%	2.96%	2.96%	2.91%	2.83%
ZOO	9.50%	3.60%	3.60%	3.60%	3.70%
AVERAGE	**26.46%**	**19.94%**	**19.92%**	**20.60%**	**20.50%**

Table 12. Comparative Classification Error Rates Across Datasets Using ECHO-Based and Genetic Optimization Methods

Table 12 presents the classification error rates for a wide range of datasets, evaluated using a series of techniques to train an artificial neural network with 10 processing nodes. These methods include standard Genetic Algorithm (GENETIC), the basic version of the ECHO+APP+MEM method, and three enhanced variants ECHO combined with approximate evaluations (APP), memory (MEM), and both (APP+MEM). Each value in the table corresponds to the percentage error for the respective dataset and method. The final row provides the average error rate across all datasets for each method, serving as an overall performance indicator. Lower values indicate better classification accuracy.

From the analysis of the total average error rates, it is clear that the basic ECHO and ECHO+APP variants yield the lowest averages, 19.94% and 19.92%, respectively. These results are significantly better than those of the Genetic Algorithm (26.46%) and the other two ECHO variants ECHO+MEM (20.60%) and ECHO+APP+MEM (20.50%). This demonstrates that, on average, ECHO and ECHO+APP achieve superior classification performance across the datasets.

The improved performance of the basic ECHO and ECHO+APP methods can be attributed to two key aspects. First, ECHO relies on an evolutionary mechanism that balances exploration and exploitation effectively, enabling efficient search without falling into local optima. Second, the incorporation of approximate evaluations in ECHO+APP reduces computational cost by estimating fitness more quickly, without significantly compromising accuracy. On the other hand, the use of memory mechanisms in ECHO+MEM

and ECHO+APP+MEM introduces complexity, which may reduce adaptability across heterogeneous datasets.

In several benchmark cases such as Alcohol, Dermatology, Segment, and Wine the ECHO+APP method achieves the lowest error rates. Although in specific datasets like Z_F_S the ECHO+APP+MEM variant performs slightly better, overall, ECHO and ECHO+APP consistently yield the most reliable results, either matching or outperforming the more complex alternatives. This supports the claim that these two methods offer an effective and computationally efficient solution.

Moreover, the small difference in average error between ECHO (19.94%) and ECHO+APP (19.92%) suggests that the use of approximate evaluations does not degrade classification quality. Instead, it may provide practical benefits in terms of speed and computational efficiency. While memory mechanisms could potentially enhance performance on problems with many local minima, they also risk over-exploitation and reduced flexibility in search dynamics.

In conclusion, the statistical analysis clearly supports the effectiveness of the basic ECHO and its APP variant for solving classification problems of small to moderate complexity. Their ability to deliver high accuracy with minimal computational overhead makes them highly competitive and well-suited for practical machine learning applications.

4. Conclusions

The Echo Optimizer, inspired by the natural behavior of sound echoes, which utilizes original mechanisms such as echoic memory and approximate evaluation for the effective resolution of complex problems. The algorithm is designed to balance the exploration of new regions within the search space and the exploitation of the best-known solutions, offering a flexible and adaptive approach to challenges that involve significant computational complexity. Its innovative mechanisms, including echo memory and approximate evaluation, enable efficient management of computational resources without compromising accuracy or convergence speed.

Experimental evaluations highlight the competitive, performance of the Echo Optimizer across a wide range of benchmark optimization functions. Multimodal functions, characterized by numerous local minima, are a domain where the algorithm excels. The echo memory mechanism, by storing and reusing previously evaluated solutions, significantly reduces the number of required objective function evaluations. This balance allows the algorithm to effectively explore the search space while avoiding redundant computations in already explored regions.

The approximate evaluation mechanism, on the other hand, accelerates the process by identifying and rejecting unpromising solutions before fully evaluating the objective function. This makes the Echo Optimizer highly efficient for functions with broad basins of attraction, where the goal is rapid convergence to the global minimum. Furthermore, the integration of these two techniques ensures that solutions stored in memory are evaluated accurately, enhancing the reliability of the algorithm. While high-dimensional problems remain challenging, the algorithm maintains robust performance by adjusting its parameters to meet the demands of each specific problem.

The method's effectiveness is further supported by its structural flexibility. Parameters such as the decay factor and reflection coefficient can be fine-tuned to adapt to different problems, while the memory and approximation mechanisms are designed to function either independently or synergistically, optimizing the use of available resources.

Moreover, the method adapts to the requirements of complex, high-dimensional problems. While computational demands increase in larger search spaces, the algorithm remains competitive compared to leading state-of-the-art methods such as CMA-ES. Its

ability to efficiently manage the complexity of the search space without an exponential increase in computational resources underscores the adaptability of the Echo Optimizer.

In summary, the Echo Optimizer provides a comprehensive optimization framework that combines design simplicity with computational efficiency. Experimental results confirm that the method is ideal for addressing multimodal problems, achieving rapid convergence in smooth functions, and maintaining competitive performance in high-dimensional environments. These characteristics make the algorithm an excellent choice for applications requiring high accuracy, speed, and flexibility, offering significant potential for future research and development in solving complex optimization challenges.

Future Research Directions include several promising avenues:

- Dynamic Parameter Adjustment: Developing mechanisms to dynamically adapt parameters such as reflection and decay coefficients based on the algorithm's behavior during execution.
- Hybrid Approaches: Combining EO with other optimization methods like swarm algorithms or genetic algorithms could further enhance performance, particularly for highly complex problems.
- Theoretical Convergence Analysis: Further investigation of the algorithm's mathematical properties, including convergence rate and probability of escaping local optima.
- Intelligent Adaptation Systems: Incorporating artificial intelligence for automatic adaptation of EO to specific problem characteristics.
- Distributed Computing: Evaluating EO's performance in parallel and distributed systems, especially for large-scale problems.

Continued research on EO promises to further improve its flexibility and effectiveness, expanding its applicability to an even broader range of scientific and industrial problems. These research directions could lead to significant advances in both theoretical understanding and practical applications of this technique.

Author Contributions: Conceptualization: V.C; methodology: I.G.T. and V.C.; software: V.C.; validation: I.G.T. and V.C. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

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

References

1. Tapkin, A. (2023). A Comprehensive Overview of Gradient Descent and its Optimization Algorithms. International Advanced Research Journal in Science, Engineering and Technology, 10 (11), 37-45. DOI: 10.17148/IARJSET.2023.101106
2. Cawade, S., Kudtarkar, A., Sawant, S. & Wadekar, H. (2024). The Newton-Raphson Method: A Detailed Analysis. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 12(11):729-734. DOI: 10.22214/ijraset.2024.65147.
3. Bonate, P.L. (2001). A Brief Introduction to Monte Carlo Simulation. Clinical Pharmacokinetics 40(1):15-22. DOI: 10.2165/00003088-200140010-00002.
4. Eglese, R. W. (1990). Simulated annealing: a tool for operational research. European journal of operational research, 46(3), 271-281.
5. Siarry, P., Berthiau, G., Durdin, F., & Haussy, J. (1997). Enhanced simulated annealing for globally minimizing functions of many-continuous variables. ACM Transactions on Mathematical Software (TOMS), 23(2), 209-228

6. Sohail, A. (2023). Genetic algorithms in the fields of artificial intelligence and data sciences. Annals of Data Science, 10(4), 1007-1018.
7. Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., & Xu, J. (2021). An improved differential evolution algorithm and its application in optimization problem. Soft Computing, 25, 5277-5298.
8. Pant, M., Zaheer, H., Garcia-Hernandez, L., & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. Engineering Applications of Artificial Intelligence, 90, 103479.
9. Charilogis, V., Tsoulos, I.G.,Tzallas, A., Karvounis, E. (2022). Modifications for the Differential Evolution Algorithm. Symmetry, 2022,14, 447. Doi: https://doi.org/10.3390/sym14030447
10. Charilogis, V.; Tsoulos, I.G. A Parallel Implementation of the Differential Evolution Method. Analytics 2023, 2, 17–30.
11. Vishnoi, N. K. (2021). Algorithms for convex optimization. Cambridge University Press. DOI: https://doi.org/10.1017/9781108699211.016.
12. Vishnoi, N. (2018, May 3). Cutting plane and ellipsoid methods for linear programming (Lecture notes, CS 435, Lecture 9). Yale University. https://nisheethvishnoi.wordpress.com/wp-content/uploads/2018/05/lecture91.pdf
13. Pióro,M. (2004). Routing, Flow, and Capacity Design in Communication and Computer Networks. Chapter 5, 151-210.
14. Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. The Computer Journal, 7(4), 308–313. Doi: https://doi.org/10.1093/comjnl/7.4.308
15. Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. Journal of Global Optimization, 13(4), 455–492. Doi: https://doi.org/10.1023/A:1008306431147
16. Snoek, J., Larochelle, H., & Adams, R. P. (2012).Practical Bayesian optimization of machine learning algorithms.In Advances in Neural Information Processing Systems (NeurIPS 2012), 25, 2951–2959.
17. Mockus, J. (1975). Bayesian approach to global optimization: Theory and applications. Vilnius: Mokslas.
18. Fletcher, R., & Reeves, C. M. (1964).Function minimization by conjugate gradients.The Computer Journal, 7(2), 149–154. Doi: https://doi.org/10.1093/comjnl/7.2.149.
19. Polak, E., & Ribière, G. (1969).Note sur la convergence de méthodes de directions conjuguées.Revue Française d'Informatique et de Recherche Opérationnelle. Série Rouge, 3(16), 35–43.
20. Nocedal, J., & Wright, S. J. (2006). Numerical optimization (2nd ed.). Springer.ISBN: 978-0387303031
21. Benders, J. F. (1962).Partitioning procedures for solving linear programming problems.Numerische Mathematik, 4(1), 238–252.Doi: https://doi.org/10.1007/BF02192511.
22. Geoffrion, A. M., & Dempster, M. A. H. (2021).Benders decomposition in the age of big data and machine learning.Journal of Optimization Theory and Applications, 179(2), 401–425. Doi: https://doi.org/10.1007/s10957-021-01774-5.
23. Dantzig, G. B., & Wolfe, P. (1960).Decomposition principle for linear programs.Operations Research, 8(1), 101–111. Doi: https://doi.org/10.1287/opre.8.1.101.
24. Jones, D. R., & Perttunen, C. D. (1993).Lipschitzian optimization without the Lipschitz constant.Journal of Optimization Theory and Applications, 79(1), 157–181.Doi: https://doi.org/10.1007/BF00941236.
25. Land, A. H., & Doig, A. G. (1960).An automatic method of solving discrete programming problems.Econometrica, 28(3), 497–520. DOI: https://doi.org/10.2307/1907756.
26. Lemarechal, C., & Lasserre, J. (2020).Branch-and-Bound methods in Mixed Integer Nonlinear Programming.Handbook of Discrete Optimization, Elsevier, pp. 297-335. Doi: https://doi.org/10.1016/B978-0-12-801857-3.00009-3
27. Shami, T. M., El-Saleh, A. A., Alswaitti, M., Al-Tashi, Q., Summakieh, M. A., & Mirjalili, S. (2022). Particle swarm optimization: A comprehensive survey. Ieee Access, 10, 10031-10061.
28. Gad, A. G. (2022). Particle swarm optimization algorithm and its applications: a systematic review. Archives of computational methods in engineering, 29(5), 2531-2561
29. 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), 26(1), 29–41. Doi: https://doi.org/10.1109/3477.484436.
30. Rappoport, D., & Schreiber, M. (2021).Recent Advances in Crystal Structure Optimization and Prediction.Crystals, 11(6), 715. Doi: https://doi.org/10.3390/cryst11060715.
31. Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009).GSA: A gravitational search algorithm.Information Sciences, 179(13), 2232–2248. Doi: https://doi.org/10.1016/j.ins.2009.03.004
32. Jang, J. S. R., Sun, C. T., & Mizutani, E. (1997). Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence. Prentice Hall.
33. Zhang, X., & Wu, J. (2010).Photosynthesis algorithm: A new optimization algorithm inspired by natural photosynthesis process.Proceedings of the 2010 International Conference on Artificial Intelligence and Computational Intelligence, 79–83. Doi: https://doi.org/10.1109/AICI.2010.23.
34. Adleman, L. (1994).Towards a mathematical theory of DNA-based computation.DNA Computing, 1, 1–22. Doi: https://doi.org/10.1007/978-1-4615-5890-1_1.

35. Zabinsky, Z. B., Graesser, D. L., Tuttle, M. E., & Kim, G. I. (1992). Global optimization of composite laminates using improving hit and run. In Recent Advances in Global Optimization (pp. 343–368).
36. Tsoulos, I.G., Charilogis, V., Kyrou, G., Stavrou, V.N. & Tzallas,A. (2025). OPTIMUS: A Multidimensional Global Optimization Package. Journal of Open Source Software, 10(108), 7584. Doi: https://doi.org/10.21105/joss.07584.
37. Lee, Y., Filliben, J., Micheals,R. & Phillips ,J. (2012). Sensitivity Analysis for Biometric Systems: A Methodology Based on Orthogonal Experiment Designs. National Institute of Standards and Technology Gaithersburg (NISTIR), MD 20899. Doi: http://dx.doi.org/10.6028/NIST.IR.7855
38. Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation, 10(3), 281–295.Doi: https://doi.org/10.1109/TEVC.2005.857610
39. Qin, A. K., Huang, V. L., & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transactions on Evolutionary Computation, 13(2), 398–417. Doi: https://doi.org/10.1109/TEVC.2008.927706
40. Brest, J., Greiner, S., Boskovic, B., Mernik, M., & Zumer, V. (2006). Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE Transactions on Evolutionary Computation, 10(6), 646–657. Doi: https://doi.org/10.1109/TEVC.2006.872133
41. Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation, 9(2), 159–195. Doi: https://doi.org/10.1162/106365601750190398
42. Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the american statistical association, 32(200), 675-701. Doi: https://doi.org/10.1080/01621459.1937.105035
43. Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11).
44. Suryadevara, S., & Yanamala, A. K. Y. (2021). A Comprehensive Overview of Artificial Neural Networks: Evolution, Architectures, and Applications. Revista de Inteligencia Artificial en Medicina, 12(1), 51-76.
45. Kelly, M., Longjohn, R., & Nottingham, K. (n.d.). The UCI Machine Learning Repository. Retrieved from https://archive.ics.uci.edu
46. Alcalá-Fdez, J., Fernandez, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing, 17, 255–287.
47. Weiss, S. M., & Kulikowski, C. A. (1991). Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufmann Publishers Inc.
48. Tzimourta, K.D.; Tsoulos, I.; Bilero, I.T.; Tzallas, A.T.; Tsipouras, M.G.; Giannakeas, N. Direct Assessment of Alcohol Consumption in Mental State Using Brain Computer Interfaces and Grammatical Evolution. Inventions 2018, 3, 51.
49. Quinlan, J. R. (1987). Simplifying Decision Trees. International Journal of Man-Machine Studies, 27(3), 221–234.
50. Shultz, T., Mareschal, D., & Schmidt, W. (1994). Modeling Cognitive Development on Balance Scale Phenomena. Machine Learning, 16, 59–88.
51. Zhou, Z. H., & Jiang, Y. (2004). NeC4.5: neural ensemble based C4.5. IEEE Transactions on Knowledge and Data Engineering, 16(6), 770–773.
52. Setiono, R., & Leow, W. K. (2000). FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks. Applied Intelligence, 12(1), 15–25.
53. Demiroz, G., Govenir, H. A., & Ilter, N. (1998). Learning Differential Diagnosis of Eryhemato-Squamous Diseases using Voting Feature Intervals. Artificial Intelligence in Medicine, 13, 147–165.
54. P. Horton, K. Nakai, A Probabilistic Classification System for Predicting the Cellular Localization Sites of Proteins, In: Proceedings of International Conference on Intelligent Systems for Molecular Biology **4**, pp. 109-15, 1996.
55. Hayes-Roth, B., & Hayes-Roth, F. (1977). Concept learning and the recognition and classification of exemplars. Journal of Verbal Learning and Verbal Behavior, 16, 321–338.
56. Kononenko, I., Šimec, E., & Robnik-Šikonja, M. (1997). Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF. Applied Intelligence, 7, 39–55.
57. French, R. M., & Chater, N. (2002). Using noise to compute error surfaces in connectionist networks: a novel means of reducing catastrophic forgetting. Neural Computation, 14, 1755–1769.
58. Dy, J. G., & Brodley, C. E. (2004). Feature Selection for Unsupervised Learning. Journal of Machine Learning Research, 5, 845–889.
59. Perantonis, S. J., & Virvilis, V. (1999). Input Feature Extraction for Multilayered Perceptrons Using Supervised Principal Component Analysis. Neural Processing Letters, 10, 243–252.
60. Garcke, J., & Griebel, M. (2002). Classification with sparse grids using simplicial basis functions. Intelligent Data Analysis, 6(5), 483–502.
61. McDermott, J., & Forsyth, R. S. (2016). Diagnosing a disorder in a classification benchmark. Pattern Recognition Letters, 73, 41–43.

62. Cestnik, G., Kononenko, I., & Bratko, I. (1987). Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users. In Bratko, I. & Lavrac, N. (Eds.), Progress in Machine Learning (pp. 31–45). Wilmslow: Sigma Press.
63. Elter, M., Schulz-Wendtland, R., & Wittenberg, T. (2007). The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical Physics, 34, 4164–4172.
64. M.A. Little, P.E. McSharry, S.J Roberts et al, Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection. BioMed Eng OnLine **6**, 23, 2007.
65. Little, M. A., McSharry, P. E., Roberts, S. J., Costello, D. A., & Moroz, I. M. (2007). Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection. BioMedical Engineering OnLine, 6(23). https://doi.org/10.1186/1475-925X-6-23
66. Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., & Johannes, R. S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Symposium on Computer Applications and Medical Care (pp. 261–265). IEEE Computer Society Press.
67. Lucas, D. D., Klein, R., Tannahill, J., Ivanova, D., Brandon, S., Domyancic, D., & Zhang, Y. (2013). Failure analysis of parameter-induced simulation crashes in climate models. Geoscientific Model Development, 6(4), 1157–1171.
68. Giannakeas, N., Tsipouras, M. G., Tzallas, A. T., Kyriakidi, K., Tsianou, Z. E., Manousou, P., Hall, A., Karvounis, E. C., Tsianos, V., & Tsianos, E. (2015). A clustering based method for collagen proportional area extraction in liver biopsy images. In Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS) (pp. 3097–3100).
69. Hastie, T., & Tibshirani, R. (1987). Non-parametric logistic and proportional odds regression. Journal of the Royal Statistical Society: Series C (Applied Statistics), 36(3), 260–276.
70. Dash, M., Liu, H., Scheuermann, P., & Tan, K. L. (2003). Fast hierarchical clustering and its validation. Data & Knowledge Engineering, 44, 109–138.
71. Gorman, R.P.; Sejnowski, T.J. Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets. Neural Netw. 1988, 1, 75–89.
72. Cortez, P., & Silva, A. M. G. (2008). Using data mining to predict secondary school student performance. In Proceedings of the 5th Future Business Technology Conference (FUBUTEC 2008) (pp. 5–12). EUROSIS-ETI.
73. Yeh, I.-C., Yang, K.-J., & Ting, T.-M. (2009). Knowledge discovery on RFM model using Bernoulli sequence. Expert Systems with Applications, 36(3), 5866–5871.
74. Jeyasingh, S., & Veluchamy, M. (2017). Modified bat algorithm for feature selection with the Wisconsin diagnosis breast cancer (WDBC) dataset. Asian Pacific journal of cancer prevention: APJCP, 18(5), 1257.
75. Alshayeji, M. H., Ellethy, H., & Gupta, R. (2022). Computer-aided detection of breast cancer on the Wisconsin dataset: An artificial neural networks approach. Biomedical signal processing and control, 71, 103141.
76. Raymer, M., Doom, T. E., Kuhn, L. A., & Punch, W. F. (2003). Knowledge discovery in medical and biological datasets using a hybrid Bayes classifier/evolutionary algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 33(5), 802–813.
77. Zhong, P., & Fukushima, M. (2007). Regularized nonsmooth Newton method for multi-class support vector machines. Optimization Methods and Software, 22(2), 225–236.
78. Andrzejak, R. G., Lehnertz, K., Mormann, F., Rieke, C., David, P., & Elger, C. E. (2001). Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: dependence on recording region and brain state. Physical Review E, 64(6), 061907.
79. Tzallas, A. T., Tsipouras, M. G., & Fotiadis, D. I. (2007). Automatic Seizure Detection Based on Time-Frequency Analysis and Artificial Neural Networks. Computational Intelligence and Neuroscience, 2007, Article ID 80510. https://doi.org/10.1155/2007/80510
80. M. Koivisto, K. Sood, Exact Bayesian Structure Discovery in Bayesian Networks, The Journal of Machine Learning Research **5**, pp. 549–573, 2004.