Article

# Healing Intelligence: A Bio-Inspired Metaheuristic optimization method Using Recovery Dynamics

**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**

BioHealing Optimization (BHO) is a bio-inspired metaheuristic optimization algorithm that emulates the biological process of injury and recovery. Its operation follows a cyclical mechanism comprising three main stages: an optional recombination phase, an injury phase, and a healing phase. During recombination, elements from the best-known solution are combined with differences drawn from other population members, producing candidate solutions that inherit beneficial traits while maintaining diversity. The injury phase applies stochastic perturbations to selected dimensions of solutions, using either Gaussian-like distributions or heavy-tailed variations, thereby promoting exploration of new regions in the search space. In the healing phase, the altered dimensions are guided gradually toward the current best solution, mimicking the progressive restoration of function observed in biological tissues. These core mechanisms are enhanced through adaptive strategies, including dynamic adjustment of injury intensity and probability, a "scar mapping" system that stores directional trends, focus on dimensions of higher relevance, and the introduction of high-intensity disturbance phases to overcome stagnation. The combination of these elements results in a self-regulating search process that maintains a balance between exploration and exploitation, enabling effective performance on challenging continuous optimization problems.

**Keywords:** Bio-inspired Algorithms; Metaheuristics; Regenerative Computing; Wound Healing; Evolutionary Algorithms; Global Optimization; Mutation Strategies;

Received:
Revised:
Accepted:
Published:

**Citation:** Charilogis, V.; Tsoulos, I.G. Healing Intelligence: A Bio-Inspired Metaheuristic optimization method Using Recovery Dynamicss. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

## 1. Introduction

Global optimization refers to the task of identifying the global minimum of a real-valued, continuous objective function $f(x)$ , where the variable $x$ belongs to a predefined, bounded search space $S \subset \mathbb{R}^n$. The goal is to determine the point $x^* \in S$ such that the function $f(x)$ achieves its lowest possible value over the entire domain:

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

where:

- $f(x)$: is the objective function to be minimized. This function can represent a variety of criteria depending on the problem context, such as cost, loss, error, potential energy, or any other performance metric.
- $S$: is the feasible search space, a compact subset of $\mathbb{R}^n$, meaning it is both closed and bounded. Typically, $S$ is defined as an $n$-dimensional hyperrectangle (also called a box constraint), given by:

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

This denotes that each variable $x_i$ is constrained within a finite interval: $x_i \in [a_i, b_i]$,for $i = 1, 2, ..., n$ = 1,2,...,$n$. The Cartesian product of these intervals defines the multidimensional region where the search for the global optimum takes place.

Optimization is one of the most fundamental and widely applied domains of computational intelligence, with a vast range of applications in scientific, technological, and industrial fields. Although classical optimization techniques can be effective for small- or medium-scale problems, they often fail to deliver satisfactory results when applied to complex, nonlinear, and high-dimensional environments, where issues such as non-convexity, high dimensionality, and the presence of multiple local optima dominate the search landscape. In this context, recent years have seen a continuous rise in interest toward metaheuristic methods, which offer flexible and stochastic search tools capable of addressing complex optimization problems without requiring derivative information or assumptions of continuity in the solution space.

Metaheuristic techniques are typically inspired by natural, biological, social, or physical processes, aiming to simulate powerful mechanisms for balancing exploration and exploitation within complex search spaces. Classic examples include Genetic Algorithms (GA) [1], Particle Swarm Optimization (PSO) [2], and Ant Colony Optimization (ACO) [3], which have been widely used for decades. In recent years, however, a multitude of novel metaheuristics have emerged, motivated by the desire to overcome common limitations such as premature convergence and weak performance on rugged, multimodal landscapes [4]. These methods draw inspiration from a broad range of biological and ecological systems. From the animal kingdom, algorithms like Artificial Bee Colony (ABC) [6], Grey Wolf Optimizer (GWO) [7], Whale Optimization Algorithm (WOA) [8], Dragonfly Algorithm (DA) [9], Cuckoo Search (CS) [10], and Bat Algorithm [11] emulate foraging, social, or navigation behaviors. Predator–prey-based algorithms such as Harris Hawks Optimization (HHO) [12] and Snake Optimizer [13] capture hunting dynamics. Others derive from insect colonies or swarm intelligence, including Firefly Algorithm [14], Glow-worm Swarm Optimization (GSO) [15], and Butterfly Optimization [16]. Likewise, bacterial or microbial behaviors inspire algorithms such as Bacterial Foraging Optimization (BFO) [17], Virus Colony Search [18], and COVIDOA [19]. Some algorithms are motivated by botanical and plant behavior, such as the Plant Propagation Algorithm (PPA) [20], Invasive Weed Optimization (IWO) [21], and Root Growth Optimizer [22]. Other methods emerge from natural physical phenomena, including Gravitational Search Algorithm (GSA) [23], Simulated Annealing [24], and the Harmony Search algorithm [25]. More recently, complex hybrid and bio-inspired models such as Gorilla Troops Optimization (GTO) [26], Reptile Search Algorithm (RSA) [27], Sine Cosine Algorithm (SCA) [28], and Slime Mould Algorithm (SMA) [29] have been introduced. Despite the thematic diversity and creativity of modern bio-inspired metaheuristic techniques, many of them continue to face common shortcomings, such as the lack of truly adaptive dynamics, a static and rigid balance between exploration and exploitation, and the absence of documented convergence guarantees [30].These limitations underline the need for next-generation algorithms capable of self-regulating their behavior according to the state of the search, maintaining stability in convergence, and faithfully reflecting the principles and rhythms of complex biological processes.

BioHealing Optimization (BHO) is positioned within this scientific and technological context, drawing inspiration from the regenerative process of wound healing in living organisms. Wound healing is a natural function characterized by a delicate balance between disruption and restoration, aimed at re-establishing homeostasis. BHO translates this biological principle into the optimization domain, creating a multi-phase methodology in which each phase has a distinct yet interdependent role.

1. **Injury phase**: Rather than relying on static or simplistic modifications, BHO applies stochastic disturbances to selected dimensions of candidate solutions, emulating the initial, uncontrolled nature of biological injury. These disturbances may follow distributions that favor either gentler changes or rare, high-impact shifts, with their intensity dynamically adapted as the search progresses encouraging broad exploration in the early stages and gradually reducing disruption near convergence.
2. **Healing phase**: Subsequently, BHO selectively guides the modified dimensions toward the best-known solution, in a process that mirrors the progressive restoration of biological tissues. This movement is neither mechanical nor fixed; the proportion and direction of adjustments are adapted to the search conditions, maintaining diversity while also enhancing the exploitation of high-quality solutions.
3. **Recombination phase**: Optionally, this process is preceded by an information exchange mechanism inspired by Differential Evolution, where components of the best solution are combined with differences from other members of the population. This allows the inheritance of strong traits while simultaneously introducing variations that keep the search active.

Previous approaches inspired by healing processes, such as Wound Healing Based Optimization (WHO) [31] and the Synergistic Fibroblast Optimization (SFO) [32], , while interesting, implement more macroscopic models or focus primarily on biological analogies without a clear separation between exploration and exploitation. They also do not employ adaptive stochastic perturbations or integrate evolutionary recombination mechanisms.

In contrast, BHO combines stochastic disruption, guided restoration, and evolutionary recombination into a single, flexible architecture that transitions smoothly and self-regulates from exploration to exploitation. It further incorporates innovations such as dynamic adjustment of injury and healing probabilities and intensities, a "scar mapping" system that retains memory of improvement directions, focus on critical dimensions, and the introduction of high-intensity disturbance phases to overcome stagnation. Together, these elements form a methodological proposal that is fundamentally distinct from existing approaches and offers enhanced robustness and performance in demanding, high-dimensional optimization problems.

The rest of the paper is organized as follows:

Section 2.4 presents the biological inspiration and motivation behind the proposed method. Section 2 details the mathematical formulation and step-by-step pseudocode of the BioHealingOptimizer. Section 3 describes the experimental setup, benchmark functions, and implementation details. Section 3.2 reports and analyzes the experimental results, including comparative performance and statistical evaluations. Section 4 provides the conclusions of the study, and Section 5 outlines directions for future research and potential extensions of the method.

## 2. The BioHealing Optimizer algorithm

### *2.1. The basic body of the BioHealing Optimizer*

The overall algorithm of the method follows:

**Algorithm 1** The basic body of the BioHealing Optimizer pseudocode

```
Input:
  f: objective function to minimize
  dim: problem dimensionality
  NP: population size
  iter_max: maximum number of iterations
  FE_max: maximum number of iterations
  lower, upper: bounds for each variable
Params:
  ws_0 : initial wound intensity
  wp : probability of wounding per dimension
  hr : probability of healing per dimension
  rp : probability of recombination with the best solution
  F : differential weight scaling factor in recombination
  CR : crossover probability in recombination
  LSR : local search rate
Output:
  x_best: the best solution found
  f_best: the value of f at best solution
Initialization:
for i=1..NP:
    for j=1..dim:
        x_{i,j} U(lower_j, upper_j)
    fit_i = f(x_i)
x_best, f_best = argmin(fit_i)
iter = 0
Main loop:
while iter < iter_max or FE < FE_max:
    iter = iter + 1
    elite = argmin(fit_i)
    x_best = x_elite, fbest = fit_elite
    ws = max(0.05·ws_0, ws_0·(1 - iter/iter_max))
    for i=1..NP:
        if i == elite: continue
        x_old = x_i, f_old = fit_i
        if U(0,1) < rp:
            choose r_1 ≠ i, r_2 ≠ i, r_2 ≠ r_1
            j_r = randInt(1,dim)
            for j=1..dim:
                if U(0,1) < CR or j = j_r:
                    v = x_{best,j} + F·(x_{r1,j} - x_{r2,j})
                    x_{i,j}= clamp(v, lower_j, upper_j)
        for j=1..dim:
            if U(0,1) < wp:
                ξ = stochasticStep() // N(0,1) or Lévy
                d = ws· ξ · (upper_j-lower_j)
                x_{i,j} = clamp(x_{i,j}+ d, lower_j, upper_j)
        a =healStep(hr)
        for j=1..dim:
            if U(0,1) < hr:
                x_{i,j} = clamp(x_{i,j} + a(x_{best_j}-x_{i,j}), lower_j, upper_j)
        f_new = f(x_i)
        if f_new < f_old:
            fit_i = f_new
            if f_new < f_best:
                f_best = f_new , x_best = x_i
            else:
                x_i = x_old; fit_i = f_old
    if LSR > 0:
        for i=1..NP:
            if i = elite: continue
            if U(0,1) < LSR:
                x_old = x_i, f_old = fit_i
                x_i = localSearch(x_i)
                f_new = f(x_i)
                if f_new < f_old:
                    fit_i = f_new
                    if f_new < f_best:
                        f_best = f_new; x_best = x_i
                else:
                    x_i = x_old, fit_i = f_old
return x_best, f_best
```

**Algorithm 2** zzz

Input: $changed_{dims}$ : Which dimensions changed, $towardBest_j \in (0,1)$, $signDir_j \in (-1,+1)$
Params: scarLR, $scar_{pmin}$, $scar_{pmax}$, $scar_{pmin}$, $scar_{pmax}$, $mom_{decay}$, $bandage_{len}$
State: $woundPdim_j$, $woundSdim_j$, $scarMomentum_j$, $bandage_i$, $j$, $dimScore_j$
01 for each $j$ in $changed_{dims}$:
02 $gP$ = scarLR · (0.5 + 0.5 · $towardBest_j$)
03 $gS$ = scarLR · (0.25 + 0.75 · $towardBest_j$)
04 $woundPdim_j$ = clamp($woundPdim_j + gP$, $scar_{pmin}$, $scar_{pmax}$)
05 $woundSdim_j$ = clamp($woundSdim_j + gS$, $scar_{smin}$, $scar_{smax}$)
06 $scarMomentum_j = (1 - mom_{decay}) \cdot scarMomentum_j + mom_{decay} \cdot signDir_j$
07 $dimScore_j = dimScore_j$ + improvement_signal() // e.g. |f_old − f_new|
08 if $bandage_{len} > 0$: $bandage_{i,j} = bandage_{len}$ // "freeze" recently improved dimension

The core loop of the BHO maintains a population of candidate solutions within box constraints and repeatedly balances broad exploration with guided exploitation. It begins by sampling each vector uniformly within the per-dimension bounds, evaluating all candidates, and designating the incumbent best. At every iteration, the current elite is identified and the wound intensity follows a monotone decay schedule so that early updates encourage wide exploration while later ones stabilize around promising regions. For each non-elite individual, an optional Differential Evolution recombination (best/1, bin) may combine the incumbent best with a scaled difference of two distinct peers; all values are kept feasible through clamping to the bounds. The injury phase then applies a per-dimension stochastic disturbance with a specified probability, using either Gaussian noise or a Lévy-tailed step produced by a generic stochasticStep() procedure and scaled by the current wound intensity and the variable range, feasibility is again enforced by clamping. The healing phase gently attracts modified components toward the incumbent best with a specified probability, using a step $a$ = healStep($hr$) that increases with the healing rate and preserves bounds. The resulting trial is evaluated and accepted greedily only if it improves the previous fitness; whenever an improvement is accepted, the global best is also updated. Optionally, a lightweight local search may refine non-elite individuals with some probability, under the same greedy acceptance rule. The procedure terminates upon exhausting either the iteration budget or the cap on function evaluations, and returns the pair ($x_{best}$, $f_{best}$). This description captures the clean, modular backbone of BHO elite selection, optional recombination, injury, healing, and greedy replacement—while allowing optional extensions to be integrated without altering the fundamental methodology.

### *2.2. Mechanism A: Scar Map, Momentum & Bandag*

After each successful acceptance (when the new solution improves the previous one), Mechanism A updates, per dimension, a "scar map" that stores two quantities: the future probability of wounding and its intensity. The update follows the learning rate (scarLR) and is clamped within $scar_{pmin}/scar_{pmax}$ and $scar_{smin}/scar_{smax}$. When the accepted change moved toward the current best (towardBest_j), the adjustment is strengthened so that dimensions that contributed to progress are wounded more often and more purposefully later. In parallel, the momentum term ($scarMomentum_j$) keeps a decayed running sign of recent accepted moves ($signDir_j$) using mom_decay, allowing the next stochastic step to lean slightly toward the beneficial direction. The dimension score ($dimScore_j$) rises proportionally to the achieved improvement and later feeds the selection of "hot" dimensions. Finally, $bandage_{len}$ freezes just-improved dimensions for a few iterations, protecting the gain from immediate over-disturbance.

Integration with the core loop is straightforward: Mechanism A runs right after greedy acceptance, only when $f_{new} < f_{old}$. In subsequent cycles, the injury phase no longer uses a

**Algorithm 3** zzz

```
Params: hot_k, hot_boost_p, hot_boost_s, dim_decay
State : dim_score[j]
TopK:
idx = argsort(dim_score, descending)
hot = idx[1..min(hot_k, dim)]
Decay (per-iter):
for j=1..dim: dim_score[j] = (1 − dim_decay) * dim_score[j]
In Injury (per j):
p_base = (scar_enabled ? clamp(wound_p_dim[j], scar_pmin, scar_pmax) : wp)
scale = (scar_enabled ? clamp(wound_s_dim[j], scar_smin, scar_smax) : 1)
if j ∈ hot: p_base = boostProb(p_base, hot_boost_p)
if j ∈ hot: scale = scale * hot_boost_s
```

single wp but reads the per-dimension $woundPdim_j$ and $woundSdim_j$ and, where applicable, blends the random disturbance with momentum. The healing phase remains unchanged, while the bandage temporarily prevents new wounds on freshly improved dimensions. In this way, the core stays clean, and the auxiliary structures self-regulate the rate and targeting of exploration on a per-dimension basis.

### *2.3. Hot-Dims Focus (top-K & boosts for injury)*

### *2.4. the bio*

## 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 proposed algorithm to assess its reliability and effectiveness in different optimization scenarios. The complete parameter configurations used throughout these experiments are documented in Table 1.

**Table 1.** Parameters and settings

| PARAMETER | VALUE | DESCRIPTION |
|---|---|---|
| $NP$ | 100 | Population for all methods except CMA-ES |
| $NP_{CMA-ES}$ | $4 + \lfloor 3 \cdot \log(dim) \rfloor$ | Population **of** CMA-ES |
| $iter_{max}$ | 300 | Maximum number of iterations for all methods |
| $FE_{max}$ | 150,000 | Maximum number of function evaluations for all methods |
| $SR$ | $\delta_{sim}^{(iter)} = \left\| f_{sim,min}^{(iter)} - f_{sim,min}^{(iter-1)} \right\|$ [34–36] or $iter_{max}$ or $FE_{max}$ | Stopping rule for all methods |
| $LSR$ | 0 | Local search rate for all methods |
| $ws_0$ | 0.4 | Initial wound strength (injury intensity) for BHO |
| $wp$ | 0.3 | Probability of wounding per dimension for BHO |
| $HR$ | 0.2 | Probability of healing per dimension for BHO |
| $rp$ | 0.9 | Probability of recombination with the best solution for BHO |
| $F$ | 0.5 | Differential weight scaling factor (recombination) for BHO |
| $CR$ | 0.9 | Crossover probability (recombination) for BHO |
| $T_s$ | Tourament size 8 [37] | Selection of GA |
| $C_{rate}$ | double, 0.1 (10%) (classic values) | Crossover for GA |
| $M_{rate}$ | double, 0.05 (5%) (classic values) | Mutation for GA |
| $c_1, c_2$ | 1.494 | Cognitive and Social coefficient for LCPSO |
| $w$ | 0,729 | Inertia for LCPSO |
| $F$ | 0.5 | Initial scaling factor for SaDE |
| $CR$ | 0.5 | Initial crossover rate for SaDE |

### 3.1. *Test Functions*

The performance assessment of the proposed method was carried out using a comprehensive and diverse collection of well-established benchmark functions [38–40], as listed in Table 2. These test functions represent a standard suite commonly utilized in the global optimization literature for validating and comparing metaheuristic algorithms. Each function exhibits distinct characteristics in terms of modality, separability, dimensionality, and landscape complexity, thus providing a robust basis for evaluating the generalization capability of the algorithm. Notably, the functions were employed in their original, unaltered form no additional transformations such as shifting, rotation, or scaling were applied allowing for a transparent and reproducible comparison with prior studies.

**Table 2.** The benchmark functions used in the conducted experiments

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

### 3.2. ***Experimental results***

All experimental procedures were executed on a high-performance computational infrastructure equipped with an AMD Ryzen 9 5950X CPU (16 cores, 32 threads) and 128 GB of DDR4 RAM, running under a Debian Linux environment. The evaluation protocol was designed to ensure statistical rigor and reproducibility. Specifically, each benchmark function was subjected to 30 independent runs, with each trial initialized using distinct random seeds to account for stochastic variability in the algorithm's behavior.

The BioHealingOptimizer and all comparative methods were implemented in highly optimized ANSI C++ code, integrated into the GLOBALOPTIMUS optimization framework [44], which is an open-source software environment for metaheuristic experimentation. The source code is publicly available at https://github.com/itsoulos/GLOBALOPTIMUS (accessed August 1, 2025), promoting transparency and reproducibility in research.

All algorithmic parameters, including those of competing methods, are comprehensively outlined in Table 1. The primary performance metric reported is the average number of objective function evaluations (NFEs) computed over the 30 runs for each test function. Additionally, success rates defined as the percentage of runs in which the global optimum was successfully located are included in parentheses next to the corresponding mean values. In cases where all runs achieved optimal convergence, the success rate indicator is omitted for clarity. Within the result tables, best-performing entries (i.e., those requiring the fewest function evaluations) are visually highlighted in green to facilitate comparison.

## 4. Conclusions

zzz

## 5. Future Research Directions

zzz

**Author Contributions:** V.C. implemented the methodology, I.G.T. and V.C conducted the experiments, employing all optimization methods and problems and provided the comparative experiments. I.G.T. and V.C performed the statistical analysis and prepared the manuscript. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not Applicable.

**Informed Consent 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.


1. Holland, J. H. (1975). Adaptation in natural and artificial systems. University of Michigan Press.
2. Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. Proceedings of ICNN'95 - International Conference on Neural Networks (Vol. 4, pp. 1942–1948). IEEE. DOI: 10.1109/ICNN.1995.488968
3. Dorigo, M., & Di Caro, G. (1999). Ant Colony Optimization. Proceedings of the 1999 Congress on Evolutionary Computation-CEC99* (Vol. 2, pp. 1470–1477). IEEE. DOI: 10.1109/CEC.1999.782657
4. Talbi, E. G. (2009). Metaheuristics: From Design to Implementation. Wiley. DOI: 10.1002/9780470496916
5. Yang, X. S. (2010). Nature-Inspired Metaheuristic Algorithms. (2nd ed.). Luniver Press.
6. Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization: Artificial bee colony (ABC) algorithm. Journal of Global Optimization, *39*(3), 459–471. DOI: 10.1007/s10898-007-9149-x
7. Mirjalili, S., et al. (2014). Grey Wolf Optimizer. Advances in Engineering Software, *69*, 46–61. DOI: 10.1016/j.advengsoft.2013.12.007
8. Mirjalili, S., & Lewis, A. (2016). Whale Optimization Algorithm. Advances in Engineering Software, *95*, 51–67. DOI: 10.1016/j.advengsoft.2016.01.008

9. Mirjalili, S. (2016). Dragonfly Algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications, *27*(4), 1053–1073. DOI: 10.1007/s00521-015-1920-1
10. Yang, X. S., & Deb, S. (2009). Cuckoo Search via Lévy Flights. In 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC) (pp. 210–214). IEEE. DOI: 10.1109/NABIC.2009.5393690
11. Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. In Nature Inspired Cooperative Strategies for Optimization (NICSO 2010) (pp. 65–74). Springer. DOI: 10.1007/978-3-642-12538-6_6
12. Heidari, A. A., et al. (2020). Harris Hawks Optimization: Algorithm and applications. Future Generation Computer Systems, *97*, 849–872. DOI: 10.1016/j.future.2019.02.028
13. Hashim, F. A., et al. (2022). Snake Optimizer: A novel meta-heuristic optimization algorithm. Knowledge-Based Systems, *242*, 108320. DOI: 10.1016/j.knosys.2022.108320
14. Yang, X. S. (2008). Nature-inspired metaheuristic algorithms. Luniver Press.
15. Krishnanand, K. N., & Ghose, D. (2009). Glowworm Swarm Optimization for simultaneous capture of multiple local optima of multimodal functions. Swarm Intelligence, *3*(2), 87–124. DOI: 10.1007/s11721-008-0021-5
16. Arora, S., & Singh, S. (2019). Butterfly Optimization Algorithm: A novel approach for global optimization. Soft Computing, *23*(3), 715–734. DOI: 10.1007/s00500-018-3102-4
17. Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems Magazine, *22*(3), 52–67. DOI: 10.1109/MCS.2002.1004010
18. Li, M. D., Zhao, H., Weng, X. W., & Han, T. (2016). A novel nature-inspired algorithm for optimization: Virus colony search. Advances in Engineering Software, *92*, 65–88. DOI: 10.1016/j.advengsoft.2015.11.004
19. Al-Betar, M. A., Alyasseri, Z. A. A., Awadallah, M. A., & Abu Doush, I. (2021). Coronavirus herd immunity optimizer (CHIO). Neural Computing and Applications, *33*(10), 5011–5042. DOI: 10.1007/s00521-020-05296-6
20. Salhi, A., & Fraga, E. S. (2011). Nature-inspired optimisation approaches and the new plant propagation algorithm. In Proceedings of the International Conference on Numerical Analysis and Optimization (ICeMATH 2011).
21. Mehrabian, A. R., & Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics, *1*(4), 355–366. DOI: 10.1016/j.ecoinf.2006.07.003.
22. Zhou, Y., Zhang, J., & Yang, X. (2020). Root growth optimizer: A metaheuristic algorithm inspired by root growth. IEEE Access, *8*, 109376–109389.
23. Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. Information Sciences, *179*(13), 2232–2248. DOI: 10.1016/j.ins.2009.03.004
24. Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. Science, *220*(4598), 671–680. DOI: 10.1126/science.220.4598.671
25. Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. Simulation, *76*(2), 60–68. DOI: 10.1177/003754970107600201
26. Sallam, K. M., Chakraborty, S., & Elsayed, S. M. (2022). Gorilla troops optimizer for real-world engineering optimization problems. IEEE Access, *10*, 121396–121423. DOI: 10.1109/ACCESS.2022.3222872
27. Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-Qaness, M. A., & Gandomi, A. H. (2021). Reptile search algorithm (RSA): A nature-inspired meta-heuristic optimizer. Expert Systems with Applications, *191*, 116158. DOI: 10.1016/j.eswa.2021.116158
28. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. Knowledge-Based Systems, *96*, 120–133. DOI: 10.1016/j.knosys.2015.12.022
29. Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. Future Generation Computer Systems, *111*, 300–323. DOI: 10.1016/j.future.2020.03.055
30. Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. Information Sciences, *237*, 82–117. DOI: 10.1016/j.ins.2013.02.041
31. Chawla, S., Saini, J. S., & Kumar, M. (2019). Wound healing based optimization – vision and framework. International Journal of Innovative Technology and Exploring Engineering, *8*(12S2), 88–91. https://doi.org/10.35940/ijitee.L1017108125219
32. Dhivyaprabha, T. T., Subashini, P., & Krishnaveni, M. (2018). Synergistic fibroblast optimization: A novel nature-inspired computing algorithm. Frontiers of Information Technology & Electronic Engineering, *19*(7), 815–833. https://doi.org/10.1631/FITEE.1601553
33. Lam, A. (2020). BFGS in a Nutshell: An Introduction to Quasi-Newton Methods Demystifying the inner workings of BFGS optimization. Towards Data Science.
34. Charilogis, V. & Tsoulos, I.G.(2022).Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions. Information, 13, 217.Doi: https://doi.org/10.3390/info13050217
35. Lagaris, I.E. & Tsoulos, I.G. (2007). Stopping rules for box-constrained stochastic global optimization.Applied Mathematics and Computation 197 (2008) 622–632. Doi:10.1016/j.amc.2007.08.001
36. Charilogis, V.; Tsoulos, I.G. A Parallel Implementation of the Differential Evolution Method. Analytics 2023, 2, 17–30.

37. Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Chapters 3 & 5.
38. 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
39. Koyuncu, H., & Ceylan, R. (2019). A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems. Journal of Computational Design and Engineering, 6(2), 129-142.
40. LaTorre, A., Molina, D., Osaba, E., Poyatos, J., Del Ser, J., & Herrera, F. (2021). A prescription of methodological guidelines for comparing bio-inspired optimization algorithms. Swarm and Evolutionary Computation, 67, 100973.
41. Gaviano, M., Ksasov, D. E., Lera, D., & Sergeyev, Y. D. (2003). Software for generation of classes of test functions with known local and global minima for global optimization. ACM Transactions on Mathematical Software, 29(4), 469–480.
42. Lennard-Jones, J. E. (1924). On the Determination of Molecular Fields. Proceedings of the Royal Society of London. Series A, 106(738), 463–477.
43. 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).
44. 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.
45. 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
46. 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
47. 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
48. 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
49. 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