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;

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/

Received:
Revised:
Accepted:
Published:

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:

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

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:

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

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:

The remainder of the paper is organized as follows: Section 2 presents BHO. Subsection 2.1 provides the core pseudocode, and Subsection 2.2 details how the adaptive mechanisms are integrated into the main loop. Section 3 describes the experimental protocol and benchmark suite, Subsection 3.1 lists the test functions, and Subsection 3.2 reports and analyzes the outcomes. Finally, Section 4 presents the conclusions, and Section 5 outlines avenues for future work.

2. The BioHealing Optimizer algorithm

2.1. The basic body of the BioHealing Optimizer pseudocode

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 Function Evaluations
  lower, upper: bounds for each variable
Params:
  w_s0 : initial wound intensity
  w_p : probability of wounding per dimension
  h_r : probability of healing per dimension
  r_p : probability of recombination with the best solution
  F : differential weight scaling factor in recombination
  CR : crossover probability in recombination
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
    w_s = max(0.05·w_s0, w_s0·(1 - iter/iter_max))
    for i=1..NP:
        if i = elite: continue
        x_old = x_i, f_old = fit_i
        if U(0,1) < r_p:
            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) < w_p:
                ξ = stochasticStep() // N(0,1) or Lévy
                d = w_s· 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) < h_r:
                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
return x_best, f_best
```

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(h_r) 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. 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.

Below are the core equations of BHO's three components Injury, Healing, and Recombination together with the minimal auxiliary relations required for completeness:

Notation:

- $x_{i,j}^{(iter)}$: coordinate j of individual i at iteration $iter$
- $[lower_j, uppe_j]$: bounds
- $x_{best,j}^{(iter)}$: incumbent best
- clamp(z, $lower$, $upper$) = min (max(z, $lower$), $upper$)
- $U(0,1)$: uniform random in [0,1]

1. Injury (stochastic perturbation)
   With per-dimension wound probability w_p (or adaptive $w_{p,j}$):

$$x_{i,j}^{(iter)} + w_s^{(iter)}\, \xi_{i,j}^{(iter)}\, (upper_j - lower_j),\ lower_j,\ upper_j\Big)$$

where $\xi_{i,j}^{(t)}$ = stochasticStep() and the wound intensity $w_s^{(t)}$can be scheduled, e.g.,

$$w_s^{(iter)} = \max\Big(0.05\, w_{s0},\ w_{s0}\big(1 - \tfrac{tter}{iter_{\max}}\big)\Big)$$

Auxiliary (stochastic step definition):

$$\begin{cases} \mathcal{N}(0,1), & \text{(Gaussian)} \\ \text{levyScale}\, \dfrac{u}{|v|^{1/\alpha}}, \quad u \sim \mathcal{N}(0,\sigma_u^2),\ v \sim \mathcal{N}(0,1), & \text{(Lévy/Mantegna)} \end{cases}$$

with

$$\sigma_u = \left[\frac{\Gamma(1+\alpha)\ \sin\big(\pi\alpha/2\big)}{\Gamma\big(\frac{1+\alpha}{2}\big)\ \alpha\, 2^{(\alpha-1)/2}}\right]^{1/\alpha}$$

and an optional global scale levyscale.

1. Healing (guided move toward x_{best})
   With per-dimension healing probability h_r:

$$x_{i,j}^{(iter,\text{heal})} = clamp\Big(x_{i,j}^{(iter,\text{inj})} + a^{(iter)}\big(x_{best,j}^{(iter)} - x_{i,j}^{(iter,\text{inj})}\big),\ lower_j,\ upper_j\Big),$$

where $a^{(iter)} =$ `healStep`(h_r) (e.g. a simple linear rule$a^{(iter)} = 0.15 + 0.35\, h_r$, bounded in $[0,1)$)

1. Recombination (DE/best/1/bin)
   With probability r_p the DE best/1 with binomial crossover is applied:

$$v = x_{best,j}^{(iter)} + F\big(x_{r1,j}^{(iter)} - x_{r2,j}^{(iter)}\big)$$

$$C \sim \text{Bernoulli}(\text{CR}), \text{ enforce } C_{j_{\text{rand}}} = 1$$

$$x_{i,j}^{(iter,\text{rec})} = clamp\Big(C_j\, v_{i,j}^{(iter)} + (1 - C_j)\, x_{i,j}^{(iter)},\ lower_j,\ upper_j\Big)$$

1. Greedy acceptance
   After the phases (in the algorithm's prescribed order):

$$\text{accept } x_i^{\text{new}} \text{ iff } f\big(x_i^{\text{new}}\big) < f\big(x_i^{\text{old}}\big), \quad \text{else revert.}$$

2.2. Integration of adaptive mechanisms into the BHO core loop

The integration of the extensions into the BHO core loop follows the method's natural flow without altering its backbone. After initialization and before processing individuals in each iteration, the algorithm updates stagnation status and configures any exploration bursts at this point a targeted micro-restart around the incumbent best may also be triggered when lack of improvement persists. In the same pre-loop stage, the per-dimension importance scores are decayed and the current hot set is selected so that subsequent perturbations are preferentially strengthened where recent gains have been observed.

At the heart of the iteration, immediately before applying stochastic disturbances, the noise generator is determined: the random step may use a heavy-tailed Mantegna draw to allow rare long jumps or fall back to Gaussian noise, depending on settings. The injury phase then exploits the hot-dimension priorities, the multiplicative burst effects when RAGE or Hyper-RAGE windows are active, and a mild intensity boost when early stagnation is detected. The random step is gently biased toward the recent momentum direction of each coordinate through a dedicated bias coefficient, while bandage protection can be bypassed only during bursts so as not to throttle exploration. Immediately after the standard injuries, a targeted Alpha-Strike may rarely fire on a few coordinates, scaled by variable ranges and current wound strength, before control passes to healing.

The healing phase is modulated so that, during bursts, the restorative rate is temporarily reduced to avoid erasing exploration, followed by a short cooldown with amplified healing to smooth the return to stabilization. Optionally, while bursts are active, selected components may be copied from the incumbent best to inject direction without sacrificing diversity. Acceptance remains greedy, and only upon true improvement are the scar-map learning states updated: the per-dimension wounding probabilities and strengths, momentum, importance scores, and bandage. These updates feed the next injury cycle, conveying where and how stronger or more frequent disturbances are worthwhile. When no improvement occurs, the distance since last best increases and can re-trigger soft boosts, bursts, or, if needed, micro-restarts at the beginning of the next iteration. In this way, the mechanisms are woven into injury, healing, and their transitions, while preserving the clean core architecture of elite selection, recombination, disturbance, restoration, and greedy replacement.

- Scar Map: Momentum & Bandage

Algorithm 2 Scar Map: Momentum & Bandage

Input: $changed_{dims}$: Which dimensions changed, $towardBest_j \in (0,1)$, $signDir_j \in (-1,+1) towardBest_j$
Params: $scarLR$, $scar_{pmin}/scar_{pmax}$, mom_{decay}, mom_{bias}, $bandage_{len}$
State: $woundPdim_j$, $woundSdim_j$, $scarMomentum_j$, $bandage_{i,j}$, $dimScore_j$

```
for each j in changed_dims:
    gP = scarLR · (0.5 + 0.5 · towardBest_j)
    gS = scarLR · (0.25 + 0.75 · towardBest_j)
    woundPdim_j = clamp(woundPdim_j + gP, scar_pmin, scar_pmax)
    woundSdim_j = clamp(woundSdim_j + gS, scar_smin, scar_smax)
    scarMomentum_j = (1 − mom_decay) · scarMomentum_j + mom_decay · signDir_j
    dimScore_j = dimScore_j + improvement_signal() // e.g. | f_old − f_new |
    if bandage_len > 0 bandage_i,j = bandage_len // "freeze" recently improved dimension
    if mom_bias_j = mom_bias · sign(scarMomentum_j)
```

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 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.

- Hot-Dims Focus (top-K & boosts for injury)

Algorithm 3 Hot-Dims Focus: Boosting Probability & Intensity on Top-K Dimensions

Params: $hotk$, $hotBoost_p$, $hotBoost_s$, dim_{decay}, hot
State : $dimScore_j$

```
for j= 1..dim dimScore_j= (1 − dim_decay) · dimScore_j
hot = topK(dimScore, hotk) // steers Injury boosts
if j ∈ hot p_base = p_base ·hotBoost_p)
if j ∈ hot scale = scale ·hotBoost_s
```

The Hot-Dims Focus mechanism allocates exploration effort to the coordinates that have recently contributed to improvement. A mild decay is first applied to $dimScore_j$, ensuring that older gains gradually fade and more recent signals dominate. From the updated scores, the top $hotk$ dimensions form the set hot. During the injury phase, if a coordinate is in hot, its base wounding probability p_{base} is amplified and the disturbance intensity scale is also increased. This gently shifts the balance toward components of the search space that have proven effective, while preserving global diversity across the remaining dimensions.

- RAGE & Hyper-RAGE

Algorithm 4 RAGE / Hyper-RAGE: Explosive exploration under stagnation

State : $sinceBest$, $rageTimer$, $hyperTimer$
Params: $rageStagnThr$, $rageLen$, $ragePMult$, $rageSMult$, $rageIgnoreBandage$, $rage2StagnThr$, $rage2Len$, $rage2PMult$, $rage2SMult$. optional $copyRate$
if $sinceBest \geq rageStagnThr$ and $rageTimer$=0 $rageTimer$=$rageLen$
else if $rageTimer$>0 $rageTimer$= $rageTimer$ -1
if $sinceBest \geq rage2_s tagn_t hr$ and $hyperTimer$=0 $hyperTimer$=$rage2Len$
else if $hyperTimer$>0 $hyperTimer$ = $hyperTimer$ -1
if $rageTimer$>0 $pBase$=$pBase \cdot ragePMult$, $scale$=$scale \cdot rageSMult$
if $hyperTimer$>0 $pBase$=0.999 ,$scale$=$scale \cdot rage2SMult$
if ($rageTimer$>0 and $rageIgnorBandage$) or $hyperTimer$>0 $bypassBandage$=true
if ($rageTimer$>0 or $hyperTimer$>0) and $U(0,1)$<$copyRate$ $x_{i,j} = x_{best_j}$

When the stagnation counter sinceBest exceeds predefined thresholds, temporary timers for RAGE and Hyper-RAGE ($rageTimer$, $hyperTimer$) are triggered. While a timer is active, the injury phase is amplified: in RAGE the base probability and disturbance scale are boosted ($pBase \cdot ragePMult$, $scale \cdot rageSMult$), whereas in Hyper-RAGE the wounding probability is driven near certainty and the scale is increased even further (pBase≈1, scale·rage2SMult). Optionally, bandage protection can be bypassed ($rageIgnoreBandage$) so recently improved coordinates are not shielded, and components may occasionally be copied from the incumbent best ($copyRate$) to inject direction. The timers count down and, once expired, the system returns to normal. The aim is to jolt the search out of local minima during stagnation without altering the algorithm's core loop.

- Lévy-Wounds (Mantegna)

Algorithm 5 Lévy-Wounds (Mantegna): Heavy-tailed jumps for rare long-range moves

Params: $levyAlpha$, $levyScale$
Precompute: $levySigmaU$
levyStep() u=$levySigmaU \cdot N(0,1)$, v=$N(0,1)$
return $levyScale \cdot (\frac{u}{|v|^{(\frac{1}{levyAlpha})}})$
stochasticStep() return (levyEnabled ? levyStep() : $N(0,1)$)

This mechanism draws heavy-tailed random steps to enable rare, long jumps that help escape local minima. The $levyAlpha$ parameter controls tail heaviness (lower values yield more frequent large jumps), while $levyScale$ sets the step magnitude. The constant $levySigmaU$ is precomputed (Mantegna scheme), and levyStep() samples two Gaussian variables to return a Lévy-type increment. The stochasticStep() then switches between Lévy and standard Gaussian noise, allowing the Injury phase to mix aggressive exploration with steadier local moves, with feasibility preserved via clamping.

- Alpha-Strike

Algorithm 6 Alpha-Strike: Targeted, rare, large jump on a few coordinates

Params: $alphaStrikeRate$, $alphaStrikeScale$, $hotK$
State: $bestSample$, $scarMomentum_j$
if $U(0,1) \geq alphaStrikeRate$: return
S = (hot nonempty ? pick max(1,$\frac{hotK}{2}$) : {rand j})
for j in S
    step = (1+ | stochasticStep() |)$\cdot alphaStrikeScale \cdot (upper_j - lower_j)$
    dir = choose{towardBest, momentumSign, random}
    x_j = clamp($x_j + ws \cdot step \cdot dir$, $lower_j$, $upper_j$)

Alpha-Strike triggers with a small probability ($alphaStrikeRate$) and applies a strong, targeted jump to a small subset of coordinates, preferably drawn from the current "hot" set ($hotK$). The step magnitude scales with $alphaStrikeScale$, the variable range, and the current stochastic term (stochasticStep()), while direction is chosen intelligently: either toward the incumbent best (towardBest using $bestSample$), along the momentum sign (momentumSign derived from scarMomentum_j), or randomly when diversification is desired. All updates are modulated by the current wound intensity ws and clamped to the feasible bounds. The aim is to vault past barriers and local minima swiftly, keeping the move rare yet highly impactful without disrupting the algorithm's core cadence

- Catastrophic Micro-Reset

Algorithm 7 Catastrophic Micro-Reset: Targeted mini-restart around the incumbent best

Params: $catResetThr$, $catResetFrac$, $catSigma$
State: $sinceBest$, $elite$, NP
01 if $sinceBest$<$catResetThr$ return
02 $elite$ = $argmin(fit)$
03 $nreset$ = max(1, round($catResetFrac \cdot NP$))
04 repeat k=1..$nreset$ ($i \neq elite$):
05 for j $cand_j$ = clamp($bestSample_j$ + $catSigma \cdot (upper_j - lower_j) \cdot N(0,1)$, $lower_j$, $upper_j$)
06 f = f($cand$)
07 if $f < fit_i$: x_i=$cand$, fit_i=f , updateGlobalBestIfNeeded()

When the search stalls for long enough ($sinceBest$ exceeds $catResetThr$), a gentle, targeted restart is triggered on a small fraction of the population. A number of individuals proportional to $catResetFrac$ is selected (always excluding the elite), and for each, a candidate is sampled near the current $bestSample$ by adding Gaussian noise scaled by catSigma and each dimension's range. Values are clamped to the bounds and evaluated, improvements are accepted greedily and the global best is updated if necessary. This injects controlled diversity around promising regions without wiping the population, helping the search escape local minima quickly with minimal risk.

- Healing Adjustments & Cooldown

Algorithm 8 Healing Adjustments & Cooldown: Modulating healing during bursts with a smooth post-burst ramp

Params: $healRageReduce$, $healPostCool$, $cooldownLen$
State: hr, $hrate$, $rageTimer$, $hyperTimer$, $cooldown$
01 hrate = hr
02 if ($rageTimer$>0 or $hyperTimer$>0) $hrate$ = reduce($hrate$, $healRageReduce$)
03 else if $cooldown$>0 $hrate$ = increase($hrate$, $healPostCool$)
04 $alpha$ = healStep($hrate$), applyHealingWith($alpha$)
05 if $burstEnded$ $cooldown$=$cooldownLen$
06 if $cooldown$>0 $cooldown$ = $cooldown$-1

This mechanism dynamically tunes the healing rate so exploration remains effective during bursts and stabilization accelerates immediately afterward. The base hr is copied to hrate, which is temporarily reduced whenever either burst timer is active ($rageTimer$ or $hyperTimer$) via $healRageReduce$. Once bursts end, a short cooldown window controlled by cooldown boosts healing using healPostCool until the counter reaches zero. At each step, the healing increment is computed as $alpha$ = healStep($hrate$) and applied through applyHealingWith($alpha$). When a burst ends ($burstEnded$), $cooldown$ is set to $cooldownLen$, ensuring a smooth transition from aggressive exploration to controlled exploitation without abrupt shifts in the algorithm's behavior.

- Soft-Stagnation Boost

Algorithm 9 Soft-Stagnation Boost: Gentle injury amplification under early stagnation

Params: $stagnThrSoft$, $stagnBoost$
State: $sinceBest$
Factor: $woundBoost$
01 $woundBoost$ = 1
02 if $sinceBest \geq stagnThrSoft$
03 $woundBoost$ = 1 + scaled($stagnBoost$, $sinceBest - stagnThrSoft$)
04 applyInjuryWith($scale = scale \cdot woundBoost$)

This mechanism triggers when the stagnation counter $sinceBest$ exceeds a mild threshold ($stagnThrSoft$). A boost factor $woundBoost$ above one is then computed using a scaled function of the excess over the threshold and the coefficient $stagnBoost$. The factor multiplies the injury intensity in the subsequent Injury step, briefly deepening disturbances without changing probabilities or invoking aggressive bursts (RAGE/Hyper-RAGE). Once stagnation subsides, $woundBoost$ returns to 1, restoring the normal intensity. The boost is applied before per-dimension modifiers and respects bounds via clamping, providing a gentle push for exploration without abrupt shifts in behavior.

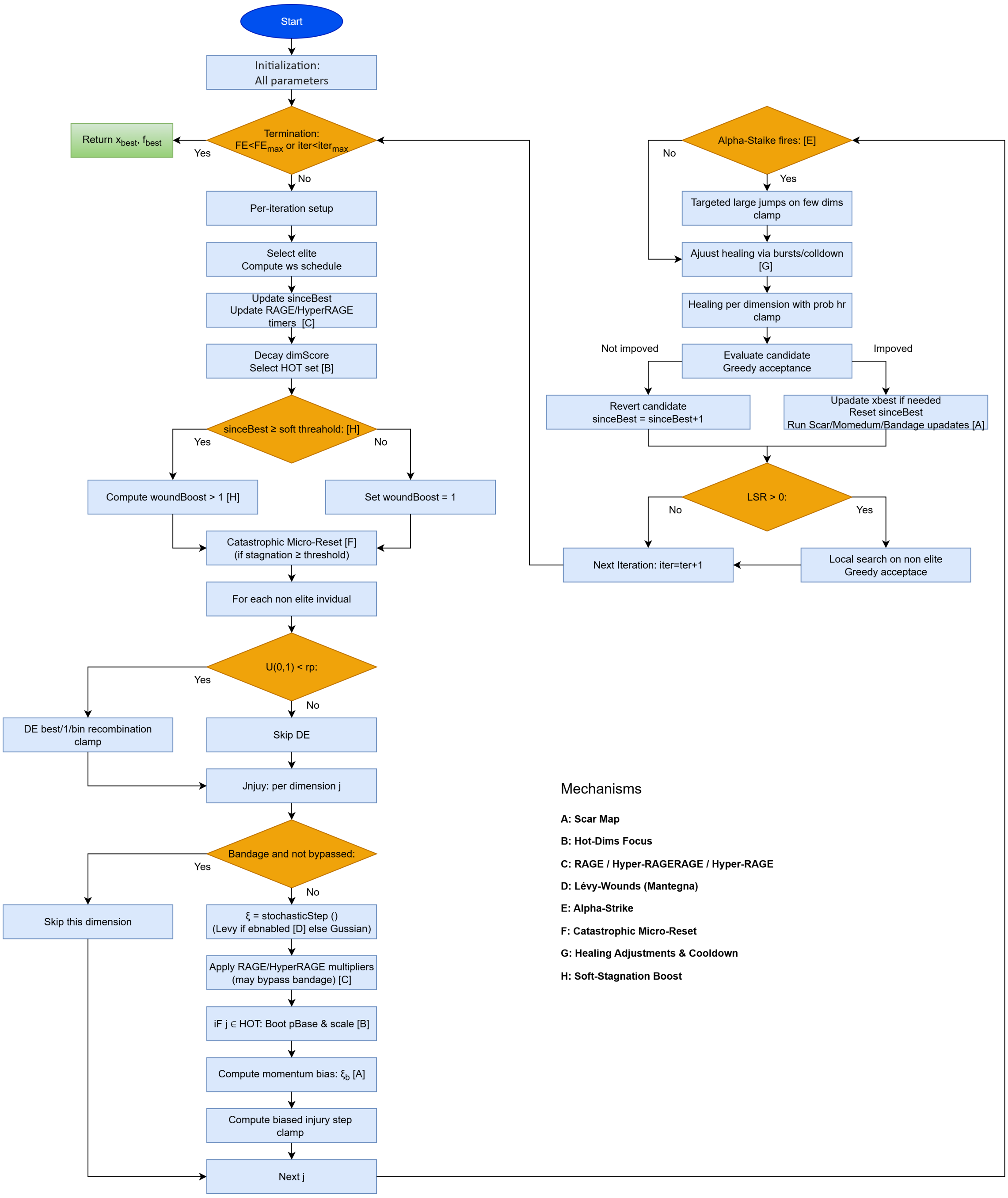

Figure 1. BHO Flowchart: Core Algorithm and Mechanism Integration

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 of BHO

Name	Value	Description
NP	100	Population size.
$iter_{max}$	500	Max iterations (also used in ws schedule).
FE_{max}	150,000	Max function evaluations (stopping).
ws_0	0.14	Initial injury intensity, decays with iter.
wp	0.38	Per-dimension wounding probability.
hr	0.8	Per-dimension healing probability.
rp	0.15	Probability of DE(best/1,bin) move.
F	0.7	DE scale factor.
CR	0.5	DE crossover rate.
Scar Map		
$scarBho$	yes	Enable Scar-Map.
$scarLR$	0.06	Scar learning rate.
$bandage_{len}$	4	Freeze improved dims (iters).
$scar_{pmin}/scar_{pmax}$	0.05 / 0.98	Bounds for per-dim wound prob.
$scar_{smin}/scar_{smax}$	0.50 / 3.00	Bounds for per-dim wound scale.
mom_{decay}	0.25	Momentum forgetting rate.
mom_{bias}	0.75	Bias strength toward momentum.
Hot-Dims Focus		
dim_{decay}	0.05	Decay rate of dimScore.
hot_k	6	Number of top dimensions.
$hotBoost_p$	1.5	Prob. boost on hot dims.
$hotBoost_s$	1.6	Strength boost on hot dims.
RAGE / Hyper-RAGE		
$rageBho$	yes	Enable RAGE bursts.
$rageStagnThr$	12	No-improve threshold for RAGE.
$rageLen$	10	RAGE duration (iters).
$ragePMmult$ / $rageSMmult$	2.2 / 2.8	Multipliers for prob/scale in RAGE.
$rageIgnoreBandage$	yes	Ignore bandage during RAGE.
$hyperRage$	yes	Enable Hyper-RAGE.
$rage2StagnThr$	28	2nd stagnation threshold.
$rage2Len$	12	Hyper-RAGE duration.
$rage2PMult/rage2SMult$	3.0 / 3.5	Multipliers in Hyper-RAGE.
Lévy-Wounds (Mantegna)		
$levyBho$	yes	Enable Lévy steps.
$levyAlpha$	1.5	Tail heaviness parameter.
$levyScale$	0.6	Lévy step scale.
Alpha-Strike		
$alphaStrikeRate$	0.1	Chance of Alpha-Strike.
$alphaStrikeScale$	0.9	Alpha-Strike step scale.
Catastrophic Micro-Reset		
$catReset$	yes	Enable catastrophic micro-reset.
$catResetThr$	40	No-improve iters → reset.
$catResetFrac$	0.08	Fraction of pop to reset.
$catSigma$	0.25	Sigma around best for reset.
Healing Adjustments & Cooldown		
$healRageReduce$	0.35	Reduce healing rate in bursts.
$healPostCool$	0.25	Increase healing after bursts.
$cooldownLen$	8	Cooldown iterations after bursts.
Soft-Stagnation Boost		
$stagnThrSoft$	10	Soft stagnation threshold.
$stagnBoost$	0.4	Extra wound factor when soft-stagnant.

Table 2. Parameters and settings of other methods

Name	Value	Description
NP	100	Population size for all methods
$iter_{max}$	500	Maximum number of iterations for all methods
CLPSO		
clProb	0.3	Comprehensive learning probability
cognitiveWeight	1.49445	Cognitive weight
inertiaWeight	0.729	Inertia weight
mutationRate	0.01	Mutation rate
socialWeight	1.49445	Social weight
CMA-ES		
NP_{CMA-ES}	$4 + \lfloor 3 \cdot \log(dim) \rfloor$	Population size
EA4Eig		
archiveSize	100	Archive size for JADE-style mutation
eig_interval	5	Recompute eigenbasis every k iterations
maxCR	1	Upper bound for CR
maxF	1	Upper bound for F
minCR	0	Lower bound for CR
minF	0.1	Lower bound for F
pbest	0.2	p-best fraction (current-to-pbest/1)
tauCR	0.1	Self-adaptation prob. for CR
tauF	0.1	Self-adaptation prob. for F
mLSHADE_RL		
archiveSize	500	Archive size
memorySize	10	Success-history memory size (H)
minPopulation	4	Minimum population size
pmax	0.2	Maximum p-best fraction
pmin	0.05	Minimum p-best fraction
SaDE		
SaDE.crSigma	0.1	Std for CR sampling
SaDE.fGamma	0.1	Scale for Cauchy F sampling
SaDE.initCR	0.5	Initial CR mean
SaDE.initF	0.7	Initial F mean
SaDE.learningPeriod	25	Iterations per adaptation window
UDE3		
minPopulation	4	Minimum population size.
memorySize	10	Success-history memory size (H).
archiveSize	100	Archive size.
pmin	0.05	Minimum p-best fraction.
pmax	0.2	Maximum p-best fraction.

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 [35–37], as listed in Table 3. 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 3. The benchmark functions used in the conducted experiments

PROBLEM	FORMULA	Dim	BOUNDS
Parameter Estimation for Frequency-Modulated Sound Waves	$\min_{x\in[-6.4,6.35]^6} f(x) = \frac{1}{N}\sum_{n=1}^{N}\left\lvert y(n;x) - y_{\text{target}}(n)\right\rvert^2$ $y(n;x) = x_0\sin(x_1 n + x_2\sin(x_3 n + x_4\sin(x_5 n)))$	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	$x_0 \in (0,0,0)$ $x_1, x_2 \in [0,4]$ $x_3 \in [0,\pi]$ 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)	$\min_{\mathbf{x}\in\Omega} f(\mathbf{x}) = \sum_{i=1}^{N} E(\mathbf{x}_i)$ $E(\mathbf{x}_i) = \frac{1}{2}\sum_{j\neq i} fc(r_{ij})\left[V_R(r_{ij}) - B_{ij}V_A(r_{ij})\right]$ where $r_{ij} = \lVert\mathbf{x}_i - \mathbf{x}_j\rVert$, $V_R(r) = A\exp(-\lambda_1 r)$ $V_A(r) = B\exp(-\lambda_2 r)$ $f_c(r)$: 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)	$\min_{\mathbf{x}} V(\mathbf{x}) = \sum_{i=1}^{N}\sum_{j>i}^{N} f_C(r_{ij})\left[a_{ij}f_R(r_{ij}) + b_{ij}f_A(r_{ij})\right]$ $f_C(r) = \begin{cases} 1, & r < R - D \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{\pi(r-R+D)}{2D}\right), & \lvert r - R\rvert \leq D \\ 0, & r > R + D \end{cases}$ $f_R(r) = A\exp(-\lambda_1 r)$ $f_A(r) = -B\exp(-\lambda_2 r)$ $b_{ij} = \left[1 + (\beta^n)\zeta_{ij}^n\right]^{-1/(2n)}$ $\sum_{k\neq i,j} f_C(r_{ik})g(\theta_{ijk})\exp\left[\lambda_3^3(r_{ij} - r_{ik})^3\right]$	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	$\min_{x\in X} f(x) = \max\{\lvert\varphi_1(x)\rvert, \lvert\varphi_2(x)\rvert, \ldots, \lvert\varphi m(x)\rvert\}$ $X = \{x \in \mathbb{R}^n \mid 0 \leq x_j \leq 2\pi,\ j = 1, \ldots, n\} m = 2n - 1$ $\varphi_j(x) = \begin{cases} \sum_{k=1}^{n-j}\cos(x_k - x_{k+j}) & \text{for } j = 1, \ldots, n-1 \\ n & \text{for } j = n \\ \varphi_{2n-j}(x) & \text{for } j = n+1, \ldots, 2n-1 \end{cases}$ $\varphi_j(x) = \sum_{k=1}^{n-j}\cos(x_k - x_{k+j}), \quad j = 1, \ldots, n-1$ $\varphi n(x) = n,\ \varphi_{n+\ell}(x) = \varphi_{n-\ell}(x), \quad \ell = 1, \ldots, n-1$	20	$x_j \in [0, 2\pi]$
Transmission Network Expansion Planning	$\min\sum_{l\in\Omega} c_l n_l + W_1\sum_{l\in OL}\lvert f_l - \bar{f}_l\rvert + W_2\sum_{l\in\Omega}\max(0, n_l - \bar{n}_l)$ $Sf = g - d$ $f_l = \gamma_l n_l \Delta\theta_l, \quad \forall l \in \Omega$ $\lvert f_l\rvert \leq \bar{f}_l n_l, \quad \forall l \in \Omega$ $0 \leq n_l \leq \bar{n}_l, \quad n_l \in \mathbb{Z}, \quad \forall l \in \Omega$	7	$0 \leq n_i \leq \bar{n}_l$ $n_i \in \mathbb{Z}$
Electricity Transmission Pricing	$\min_x \quad f(x) = \sum_{i=1}^{N_g}\left(\frac{C_i^{gen}}{P_i^{gen}} - R_i^{gen}\right)^2 + \sum_{j=1}^{N_d}\left(\frac{C_j^{load}}{P_j^{load}} - R_j^{load}\right)^2$ $\sum_j GD_{i,j} + \sum_j BT_{i,j} = P_i^{gen}, \quad \forall i$ $\sum_i GD_{i,j} + \sum_i BT_{i,j} = P_j^{load}, \quad \forall j$ $GD_{i,j}^{max} = \min(P_i^{gen} - BT_{i,j},\ P_j^{load} - BT_{i,j})$	126	$GD_{i,j} \in [0, GD_{i,j}^{max}]$
Circular Antenna Array Design	$\min_{r_1,\ldots,r_6,\varphi_1,\ldots,\varphi_6} \quad f(\mathbf{x}) = \max_{\theta\in\Omega} AF(\mathbf{x},\theta)$ $AF(\mathbf{x},\theta) = \left\lvert\sum_{k=1}^{6}\exp\left(j\left[2\pi r_k\cos(\theta - \theta_k) + \varphi_k\frac{\pi}{180}\right]\right)\right\rvert$	12	$r_k \in [0.2, 1]$ $\varphi_k \in [-180, 180]$
Dynamic Economic Dispatch 1	$\min_{\mathbf{P}} \quad f(\mathbf{P}) = \sum_{t=1}^{24}\sum_{i=1}^{5}\left(a_iP_{i,t}^2 + b_iP_{i,t} + c_i\right)$ $P_i^{\min} \leq P_{i,t} \leq P_i^{\max}, \quad \forall i = 1, \ldots, 5,\ t = 1, \ldots, 24$ $\sum_{i=1}^{5} P_{i,t} = D_t, \quad \forall t = 1, \ldots, 24$ $P_{\min} = [10, 20, 30, 40, 50]$ $P_{\max} = [75, 125, 175, 250, 300]$	120	$P_i^{\min} \leq P_{i,t} \leq P_i^{\max}$
Dynamic Economic Dispatch 2	$\min_{\mathbf{P}} \quad f(\mathbf{P}) = \sum_{t=1}^{24}\sum_{i=1}^{9}\left(a_iP_{i,t}^2 + b_iP_{i,t} + c_i\right)$ $P_i^{\min} \leq P_{i,t} \leq P_i^{\max}, \quad \forall i = 1, \ldots, 5,\ t = 1, \ldots, 24$ $\sum_{i=1}^{5} P_{i,t} = D_t, \quad \forall t = 1, \ldots, 24$ $P_{\min} = [150, 135, 73, 60, 73, 57, 20, 47, 20]$ $P_{\max} = [470, 460, 340, 300, 243, 160, 130, 120, 80]$	216	$P_i^{\min} \leq P_{i,t} \leq P_i^{\max}$
Static Economic Load Dispatch (1,2,3,4,5)	$\min_{P_1,\ldots,P_{N_G}} F = \sum_{i=1}^{N_G} f_i(P_i)$ $f_i(P_i) = a_iP_i^2 + b_iP_i + c_i, \quad i = 1, 2, \ldots, N_G$ $f_i(P_i) = a_iP_i^2 + b_iP_i + c_i + \lvert e_i\sin(f_i(P_i^{\min} - P_i))\rvert$ $P_i^{\min} \leq P_i \leq P_i^{\max}, \quad i = 1, 2, \ldots, N_G$ $\sum_{i=1}^{N_G} P_i = P_D + P_L$ $P_L = \sum_{i=1}^{N_G}\sum_{j=1}^{N_G} P_iB_{ij}P_j + \sum_{i=1}^{N_G} B_{0i}P_i + B_{00}$ $P_i - P_i^0 \leq UR_i \quad P_i^0 - P_i \leq DR_i$	6 13 15 40 140	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 [41], 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 Tables 1 and 2. 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.

For the experimental evaluation of BHO, we selected the following optimization algorithms as baselines for comparison:

- EA4Eig [42] is a cooperative evolutionary algorithm with an eigen–crossover operator, introduced at IEEE CEC 2022 and evaluated on that event's benchmark suite it was essentially designed and tested within the CEC single-objective context.
- UDE3 (UDE-III) [43] is a recent member of the Unified DE family for constrained problems it is evaluated on the CEC 2024 constrained set, while its predecessor (UDE-II/IUDE) won first place at CEC 2018, giving UDE3 a clear lineage with proven competitive performance.
- mLSHADE_RL [44] is a multi-operator descendant of LSHADE-cnEpSin one of the CEC 2017 winners for real-parameter optimizationaugmenting the base with restarts and local search and being assessed on modern test suites.
- CLPSO [45] is a classic PSO variant with comprehensive learning (2006) that has long served as a strong baseline in CEC benchmarks, not tied to a single award entry but widely used in comparative studies.
- SaDE, [46] the self-adaptive DE of Qin & Suganthan, was presented with evaluation on the CEC 2005 test set and has since become a reference point for adaptive strategies.
- jDE [47] by Brest et al. introduced self-adaptation of F and CR and was extensively evaluated, including special CEC 2009 sessions on dynamic/uncertain optimization, helping popularize self-adaptation in DE.
- CMA-ES [48] is the established covariance matrix adaptation evolution strategy for continuous domains beyond its vast literature, it is a staple baseline and frequent participant/reference in BBOB benchmarking at GECCO, effectively serving as a de facto competitive black-box comparison standard.

Table 4. Algorithms' Comparison Based on Best and Mean after 1.5e+5 FEs

FUNCTION	EA4Eig best	EA4Eig mean	UDE3 best	UDE3 mean	mLSHADE_RL best	mLSHADE_RL mean	CLPSO best	CLPSO mean	SaDE best	SaDE mean	jDE best	jDE mean	CMA-ES best	CMA-ES mean	BHO best	BHO mean
Parameter Estimation for Frequency-Modulated Sound Waves	0.153993305	0.213447296	0.03808755	0.115833815	0.116157535	0.205011062	0.1314837477	0.2124981688	0.095829444	0.195600253	0.116157541	0.146008756	0.18160916	0.256863966	1.543773917e-25	0.200220764
Lennard-Jones Potential	-18.48174236	-16.3133561	-21.41786661	-17.33977959	-28.41816707	-22.49792055	-13.43649135	-10.25073403	-21.93636189	-17.95333019	-29.98126575	-27.49258505	-28.42253189	-25.78783328	-32.07742417	-24.33212306
Bifunctional Catalyst Blend Optimal Control	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591	-0.000286591
Optimal Control of a Non-Linear Stirred Tank Reactor	0.390376723	0.390376723	0.390376723	0.390376723	0.390376723	0.390376723	0.3903767228	0.3903767228	0.390376723	0.390376723	0.390376723	0.390376723	0.3903767228	0.390376723	0.3903767228	0.390376723
Tersoff Potential for model Si (B)	-29.11960284	-27.89597789	-29.44152761	-25.70627602	-28.60814558	-26.07976794	-28.23544117	-26.18834522	-27.25703406	-25.25867422	-13.51157064	-3.983690794	-29.26244222	-27.5889735	-29.03183049	-27.26630121
Tersoff Potential for model Si (C)	-33.39767521	-31.11610936	-33.12997729	-28.6603137	-32.28575942	-30.03594436	-30.85200257	-28.87349048	-31.85343594	-29.59692733	-18.76214649	-8.506037168	-33.19699356	-31.79270914	-33.38947338	-31.31864105
Spread Spectrum Radar Polly phase Code Design	0.517866993	0.838752978	1.048240196	1.265152951	0.033146096	0.625788451	1.085334991	1.343956153	0.572731322	0.844014075	1.525870558	1.812042166	0.01484822722	0.171988666	0.195096433	0.601190863
Transmission Network Expansion Planning	250	250	250	250	250	250	250	250	250	250	250	250	250	250	250	250
Electricity Transmission Pricing	13773680	13774198.28	13773582.53	13773582.53	13773567.36	13773852.63	13775010.1	13775395.07	13773468.68	13773930.93	13774627.84	14020953.78	13775841.77	13787550.18	13773334.9	13773632.45
Circular Antenna Array Design	0.006809638	0.006809638	0.006809653	0.011722944	0.006809701	0.006823547	0.006933401045	0.05181551798	0.00681287	0.008186204	0.006820072	0.017657998	0.007204797576	0.008635655364	0.007101505	0.158523549
Dynamic Economic Dispatch 1	412736103.9	421199260.5	410197836.9	410628483.3	415275891.6	418526775.3	428607927.6	435250914.5	411226317.3	413699347.4	968042312.1	1034393036	88285.6024	102776.7103	410074526.4	410079513.3
Dynamic Economic Dispatch 2	346855.5418	12332507.25	357530.5408	537163.4139	392247.7213	5966956.365	33031590.31	53906147.38	519820.5596	4090304.18	340091475.3	397471715.1	502699.4187	477720.1511	347469.862	354734.7105
StaticEconomic Load Dispatch 1	6163.560978	6170.965013	6164.766919	6266.250251	6163.546883	6353.722019	6554.672173	7668.333603	6360.353305	6464.861828	6163.749006	6778.527028	6657.613028	415917.4625	6512.525519	6772.880257
Static Economic Load Dispatch 2	14.46160489	18779.92036	18725.64707	19660.49074	17905.85383	18661.20763	19030.36081	20699.00219	18455.37286	21829.10208	1161578.904	3671587.605	763001.2185	1425815.44	18754.99866	19232.05211
Static Economic Load Dispatch 3	470023232.3	470023234.5	470023232.3	470023232.3	470023232.6	470023234.7	470192288.3	470294703.2	470023232.7	470023232.7	471058115.8	471963142.3	470023232.3	470023232.3	470023233.2	470023278
Static Economic Load Dispatch 4	71193.07649	71193.07649	168334.7003	348720.6677	71067.8441	406986.2181	884980.5569	1423887.358	862196.432	1469886.142	6482592.714	17527314.24	476053.5197	2925852.935	71089.03508	100831.805
Static Economic Load Dispatch 15	8085796774	8145304780	8070408727	8071802922	8079118012	8104455647	8105947615	8110924071	8078489742	8081680478	8453090778	8459337082	8072077963	8084017791	8072061692	74458095

On synthetic and physical potentials, BHO delivers particularly strong best values nearly zero error on Parameter Estimation for Frequency-Modulated Sound Waves and attains the lowest best value on the Lennard-Jones Potential however, for the mean on Lennard-Jones, jDE leads and CMA-ES follows, indicating that classical, Gaussian-driven strategies remain very stable when the landscape exhibits symmetric curvature. On the Tersoff Potential for model Si (B) and Tersoff Potential for model Si (C) the picture shifts: for Si (B) the best mean comes from EA4Eig with CMA-ES close behind, while UDE3 secures the best best for Si (C), EA4Eig has the best best and CMA-ES the best mean. Overall, advanced DE variants with stronger recombination (EA4Eig, UDE3) and CMA-ES alternate at the top, which aligns with the literature on rough but moderately structured landscapes.

In electro-economic and industrial test cases the pattern diverges in informative ways. On Electricity Transmission Pricing, BHO attains the lowest best, whereas UDE3 achieves the best mean suggesting BHO can hit excellent extremes while UDE3 maintains consistently low performance across runs. In Dynamic Economic Dispatch 1 and Dynamic Economic Dispatch 2, CMA-ES dominates the first on both best and mean, confirming its strength on smooth, nearly quadratic geometries, while in the second BHO records the lowest mean and EA4Eig the best best, indicating that BHO's injury–healing dynamics act as a variance-damping safety net. Across the static dispatch family the results are mixed: in StaticEconomic Load Dispatch 1 the best best is from mLSHADE_RL and the best mean from EA4Eig in Static Economic Load Dispatch 2 mLSHADE_RL yields the lowest mean (with EA4Eig taking the best best) in Static Economic Load Dispatch 4 mLSHADE_RL again takes the best best while EA4Eig leads on mean. Static Economic Load Dispatch 3 shows practical ties around the same value and is not very discriminative. Static Economic Load Dispatch 15 contains a notable outlier where BHO's mean is orders of magnitude lower than others this merits independent verification, such as re-running experiments or checking units, because the unusually large gap likely signals a scaling difference or an unexpected effect.

On more "classic" comparative tests, CMA-ES shows impressive robustness on smooth, well-scaled landscapes: beyond Dynamic Economic Dispatch 1, it also holds both best and mean on Spread Spectrum Radar Polly phase Code Design. EA4Eig stands out when strong anisotropy or correlation matters Circular Antenna Array Design is topped by EA4Eig on both best and mean consistent with its eigen-crossover design. UDE3 performs very well on problems with additional structure or constraints, such as the mean on Electricity Transmission Pricing, matching its remit as a unified DE for constrained scenarios. mLSHADE_RL frequently attains best-case wins in StaticEconomic Load Dispatch 1 and Static Economic Load Dispatch 4 and remains competitive on mean across several cases, underscoring the value of ensemble mutation plus restarts on fractured landscapes. SaDE, while a solid baseline, tends to trail the newer DE descendants, and CLPSO underperforms in most tables expected where anisotropy demands directional information beyond standard swarm velocity updates. jDE remains competitive on certain physical potentials, for example the mean on Lennard-Jones, but shows larger dispersion on industrial dispatch tasks.

On flat or near-flat landscapes Bifunctional Catalyst Blend Optimal Control, Transmission Network Expansion Planning, and essentially Optimal Control of a Non-Linear Stirred Tank Reactor where differences are on the order of 1e-10 all methods tie or are practically indistinguishable, so these tests add little diagnostic power. In sum, there is no single winner: CMA-ES is the reference choice for smooth, well-conditioned cases and remains very strong on mean performance EA4Eig excels where alignment with principal variance directions helps UDE3 often wins on mean under constrained or pricing structure mLSHADE_RL frequently takes best-case wins on difficult static dispatch variants and BHO shows top best values on several critical functions and competitive means in demanding settings, indicating an effective exploration-to-exploitation transition.

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

FUNCTION	EA4Eig	UDE3	mLSHADE_RL	CLPSO	SaDE	jDE	CMA-ES	BHO
Parameter Estimation for Frequency-Modulated Sound Waves	7	2	4	6	3	5	8	1
Lennard-Jones Potential	7	6	4	8	5	2	3	1
Bifunctional Catalyst Blend Optimal Control	1	1	1	1	1	1	1	1
Optimal Control of a Non-Linear Stirred Tank Reactor	4	4	4	1	4	4	1	1
Tersoff Potential for model Si (B)	3	1	5	6	7	8	2	4
Tersoff Potential for model Si (C)	1	4	5	7	6	8	3	2
Spread Spectrum Radar Polly phase Code Design	4	6	2	7	5	8	1	3
Transmission Network Expansion Planning	1	1	1	1	1	1	1	1
Electricity Transmission Pricing	5	4	3	7	2	6	8	1
Circular Antenna Array Design	1	2	3	6	4	5	8	7
Dynamic Economic Dispatch 1	5	3	6	7	4	8	1	2
Dynamic Economic Dispatch 2	1	3	4	7	6	8	5	2
StaticEconomic Load Dispatch 1	2	4	1	7	5	3	8	6
Static Economic Load Dispatch 2	1	4	2	6	3	8	7	5
Static Economic Load Dispatch 3	1	1	4	7	5	8	1	6
Static Economic Load Dispatch 4	3	4	1	7	6	8	5	2
Static Economic Load Dispatch 15	6	1	5	7	4	8	3	2
Total	**53**	**51**	**55**	**98**	**71**	**99**	**66**	**47**

Table 6. Detailed Ranking of Algorithms Based on Mean after 1.5e+5 FEs

FUNCTION	EA4Eig	UDE3	mLSHADE_RL	CLPSO	SaDE	jDE	CMA-ES	BHO
Parameter Estimation for Frequency-Modulated Sound Waves	7	1	5	6	3	2	8	4
Lennard-Jones Potential	7	6	4	8	5	1	2	3
Bifunctional Catalyst Blend Optimal Control	1	1	1	1	1	1	1	1
Optimal Control of a Non-Linear Stirred Tank Reactor	2	2	2	1	2	2	2	2
Tersoff Potential for model Si (B)	1	6	5	4	7	8	2	3
Tersoff Potential for model Si (C)	3	7	4	6	5	8	1	2
Spread Spectrum Radar Polly phase Code Design	4	6	3	7	5	8	1	2
Transmission Network Expansion Planning	1	1	1	1	1	1	1	1
Electricity Transmission Pricing	5	1	3	6	4	8	7	2
Circular Antenna Array Design	1	5	2	7	3	6	4	8
Dynamic Economic Dispatch 1	6	3	5	7	4	8	1	2
Dynamic Economic Dispatch 2	6	3	5	7	4	8	2	1
StaticEconomic Load Dispatch 1	1	2	3	7	4	6	8	5
Static Economic Load Dispatch 2	2	4	1	5	6	8	7	3
Static Economic Load Dispatch 3	4	1	5	7	3	8	1	6
Static Economic Load Dispatch 4	1	3	4	5	6	8	7	2
Static Economic Load Dispatch 15	7	2	5	6	3	8	4	1
Total	**59**	**54**	**58**	**91**	**66**	**99**	**59**	**48**

Table 7. Comparison of Algorithms and Final Ranking

Algorithm	Best	Mean	Overal	Averange	Rang
BHO	47	48	95	2.794	**1**
UDE3	51	54	105	3.088	**2**
EA4Eig	53	59	112	3.294	**3**
mLSHADE_RL	55	58	113	3.323	**4**
CMA-ES	66	59	125	3.676	**5**
SaDE	71	66	137	4.029	**6**
CLPSO	98	91	189	5.558	**7**
jDE	99	99	198	5.823	**8**

The evaluation distinguishes peak performance ("best") from reliability ("mean") over seventeen problems. Summing per-function ranks gives BHO the lowest totals in both views (best: 47, mean: 48) and therefore the best overall score (overall: 95, average rank: 2.794). UDE3 follows at close range (best/mean: 51/54, overall: 105, average rank: 3.088). EA4Eig and mLSHADE_RL form the next tier with near-equal totals (112 and 113, average ranks: 3.294 and 3.323). CMA-ES sits mid-pack (overall: 125, average rank: 3.676), SaDE trails it (overall: 137, average rank: 4.029), while CLPSO and jDE rank lowest overall (overall: 189 and 198, average ranks: 5.558 and 5.823), reflecting consistently higher placements across most functions.

The detailed "best" table shows BHO frequently taking first place on demanding tasks (e.g., Parameter Estimation for Frequency-Modulated Sound Waves, Lennard–Jones Potential, Electricity Transmission Pricing), indicating strong exploration and high upside. UDE3 records many top-three finishes and several wins, especially where structure/constraints are prominent, explaining its overall second place and good generalization. EA4Eig and mLSHADE_RL trade advantages on anisotropic or fractured landscapes (e.g., Tersoff Si(B)/Si(C), Static Economic Load Dispatch), consistent with eigen-guided recombination in the former and ensemble mutation with restarts in the latter. CMA-ES shows the expected resilience on smooth, well-conditioned geometries: its mean rank is often better than its best rank, a sign of low variance rather than aggressive extremes. SaDE remains a sturdy baseline but lags newer DE descendants. CLPSO underperforms (typical under strong anisotropy). jDE exhibits occasional peaks but larger dispersion, which inflates its ranks in both best and mean.

Contrasting best and mean exposes consistency: BHO's lead in best does not come at the expense of reliability its mean total is also the lowest so top outcomes are not isolated "lucky" runs. Likewise, UDE3's small best–mean gap indicates stable performance across repeats. EA4Eig and mLSHADE_RL display complementary behaviors (alignment with principal directions for the former, collective mutations and restarts for the latter). CMA-ES often "wins" on mean where smoothness enforces small fluctuations.

Three problems are practically non-discriminative: in Bifunctional Catalyst Blend Optimal Control and Transmission Network Expansion Planning all methods tie, and in Optimal Control of a Non-Linear Stirred Tank Reactor differences are negligible. These dilute separability without altering the final ordering. By contrast, problems such as Electricity Transmission Pricing and the Dynamic/Static Economic Load Dispatch families yield substantive differences that drive the clear BHO–UDE3 lead and the tight EA4Eig–mLSHADE_RL contest for third–fourth.

In summary, with a fixed budget of 1.5×10^5 evaluations, BHO is the strongest overall method on both peak and average performance, UDE3 follows closely with high consistency, EA4Eig and mLSHADE_RL come very close behind, leveraging different mechanisms. CMA-ES remains a reliable reference baseline, while SaDE, CLPSO, and jDE underperform under the present conditions.

4. Conclusions

This work introduces BioHealing Optimization (BHO), a population-based metaheuristic that integrates stochastic "injury," guided "healing" toward the incumbent best, and an optional DE(best/1,bin) recombination step, augmented by adaptive, per-dimension mechanisms (scar map and momentum, hot-dimension focusing, RAGE/Hyper-RAGE bursts, Lévy steps, and healing modulation). The resulting architecture self-regulates exploration and exploitation without altering the core loop of elite selection, recombination, disturbance, greedy acceptance, and restoration.

The experimental protocol used 30 independent runs per problem with a fixed budget of $1.5 \cdot 10^5$ function evaluations and a harmonized parameter disclosure across all compared methods. Rank aggregation over 17 problems separates peak performance (best of runs) from reliability (mean of runs). Under this protocol, BHO attains the lowest total rank in both views (best: 47, mean: 48), yielding the top combined score (overall: 95, average rank: 2.794). The next methods are UDE3 (overall: 105), EA4Eig (112), and mLSHADE_RL (113), followed by CMA-ES, SaDE, CLPSO, and jDE. These results indicate that BHO combines high upside with consistent average performance across repetitions.

Per-problem observations are consistent with the aggregated ranking. BHO frequently secures first place on demanding tasks such as parameter estimation for frequency-modulated sound waves, Lennard–Jones potential, and electricity transmission pricing, reflecting an effective transition from exploration to exploitation. UDE3 performs strongly where structural constraints are prominent EA4Eig excels when alignment with principal variance directions matters mLSHADE_RL often achieves leading "best" values on difficult static dispatch cases and CMA-ES remains a dependable reference on smooth, well-conditioned landscapes. Some benchmarks bifunctional catalyst control and transmission network expansion planning, as well as the stirred-tank reactor with differences on the order of 1e−10 are effectively non-discriminative and do not affect the overall ordering.

Overall, within the stated evaluation budget and set of problems, BHO achieves the best combined ranking among strong baselines without dominating every individual case. The evidence supports that the combination of stochastic injury, guided healing, and DE-style recombination enriched with adaptive, per-dimension controls is an effective strategy for challenging continuous optimization, with performance that is competitive in both best-case and mean outcomes. Interpretation of the results should account for the problem characteristics (smoothness, anisotropy, constraints) and the fixed evaluation budget used here.

5. Future Research Directions

BioHealing Optimization (BHO) couples a DE(best/1, bin) recombination path with a disturbance–restoration dynamic stochastic injury plus guided healing augmented by per-dimension controllers (scar map and momentum, hot-dimension focusing, RAGE/Hyper-RAGE bursts, Lévy steps, healing modulation). From this architecture and the reported evidence, several technically grounded avenues follow. Convergence under nonstationary stochastic dynamics can be tightened via a nonhomogeneous Markov view on the extended state (population, incumbent best, scar/momentum/bandage), establishing drift/minorization under decaying injury, and clarifying when heavy-tailed Lévy perturbations speed basin escape without harming late-stage stability. Mechanism-level attribution is enabled by budget-matched ablations with nonparametric inference (Friedman, Nemenyi, Wilcoxon with effect sizes), isolating when RAGE, hot-dims, or Lévy dominate across smooth, ill-conditioned, multimodal, or constrained regimes. Endogenous control of (w_s, w_p, h_r), RAGE triggers, and hot-dim boosts can use bandit/feedback policies (and classic DE/ES self-adaptation for F, CR) to reduce hyperparameter sensitivity. High-dimensional scaling motivates subspace search, scar-weighted directions, and lightweight preconditioning (diagonal/low-rank) for partial rotational invariance. Constraints, noise, and nonstationarity can be handled by projection/prox in healing, noise-robust acceptance and budget-aware re-evaluation, and change-detector-driven RAGE activation. Hybridization with cheap, bound-aware local moves and straightforward parallelism (GPU, island models sharing elites and scar/hot-dim information) can raise evaluation efficiency. Broader benchmarking with budget sweeps and invariance-rich families (e.g., BBOB) would map where BHO excels, where UDE-type or CMA-ES baselines prevail, and where hybrids are preferable. Finally, healing admits a contractive/proximal interpretation, while injury with decaying variance resembles annealing; scar momentum acts as a signed, exponentially weighted directional prior. These directions remain within BHO's current envelope, aiming at provable properties, clear attribution of gains, robustness in practical settings, and scalable performance.

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. Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Chapters 3 & 5.
35. 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
36. 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.
37. 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.
38. 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.
39. Lennard-Jones, J. E. (1924). On the Determination of Molecular Fields. Proceedings of the Royal Society of London. Series A, 106(738), 463–477.
40. 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).
41. 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.
42. Bujok, P., & Kolenovský, P. (2022, July). Eigen crossover in cooperative model of evolutionary algorithms applied to CEC 2022 single objective numerical optimisation. In 2022 IEEE Congress on Evolutionary Computation (CEC) (pp. 1–8). IEEE. https://doi.org/10.1109/CEC55065.2022.9870433.
43. Trivedi, A., & Chauhan, D. (2024). UDE-III: An enhanced unified differential evolution algorithm for constrained optimization problems. arXiv preprint arXiv:2410.03992.
44. Chauhan, D. (2024). A multi-operator ensemble LSHADE with restart and local search mechanisms for single-objective optimization. arXiv preprint arXiv:2409.15994.
45. References 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
46. References 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
47. References 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
48. References 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. 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
50. 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
51. 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
52. 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
53. 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