Improved Gazelle Optimization algorithm using Insights from Particle Swarm Optimization Algorithm

Jyotiraditya Mishra[a], Varun Pratap Singh[a], Vishal[a]

[a]*Netaji Subhas University of Technology, Dwaraka, Delhi, India*

Abstract

There has been a notable increase in the development of optimization algorithms in recent years. Despite their potential, these algorithms often suffer from inefficiencies such as slow convergence rates. This research introduces a hybrid metaheuristic approach that integrates the exploration mechanisms of the Gazelle Optimization Algorithm (GoA) with the exploitation strategies of Particle Swarm Optimization (PSO). This combination significantly enhances efficiency, reducing the number of iterations required for convergence by an approximate factor of 10 to 12. The proposed hybrid algorithm not only accelerates convergence but also maintains robust solution quality. The results demonstrate a substantial improvement over existing methods, suggesting valuable directions for future research to further refine and expand the dynamic capabilities of the algorithm.

Keywords: Gazelle Optimization Algorithm (GoA), Particle Swarm Optimization (PSO), Hybrid Metaheuristic, Algorithm Convergence, Computational Efficiency, Optimization Techniques

1. Introduction

Optimization algorithms are fundamental to solving complex problems across various disciplines, including engineering, computer science, and operations research. These algorithms are designed to find the best possible solution or a sufficiently good solution to a problem within a reasonable time frame. In recent years, the Gazelle Optimization Algorithm (GOA) has emerged as a promising new metaheuristic technique, drawing attention for its potential in handling multidimensional and multimodal optimization problems. However, like many other algorithms, GOA faces significant chal-

lenges in terms of computational efficiency and convergence rate, particularly when applied to complex real-world scenarios.

Recent advancements have highlighted the efficiency of Particle Swarm Optimization (PSO), renowned for its fast convergence and simplicity in solving optimization problems. This paper proposes a hybrid metaheuristic approach that combines the exploration capabilities of GOA with the exploitation strategies of PSO to enhance the overall performance of the GOA. By integrating these two powerful techniques, the proposed algorithm aims to significantly reduce the number of iterations required for convergence, thus addressing one of the critical bottlenecks in the application of GOA.

The significance of this research lies in its potential to transform the operational framework of optimization algorithms. The enhanced GOA not only achieves faster convergence but also maintains the quality of solutions, which is crucial for its application in more complex and dynamic problems. This research contributes to the field by:

- Proposing a novel hybrid algorithm that combines the strengths of GOA and PSO to effectively reduce convergence times and improve computational efficiency.
- Demonstrating the applicability of the hybrid algorithm in various optimization scenarios, thereby broadening the scope of its use in practical applications.
- Providing a comprehensive comparative analysis of the hybrid algorithm against existing methodologies, highlighting its superior performance in terms of both speed and solution quality.

2. Related Works

Recent advancements in optimization algorithms have led to significant developments in both the Gazelle Optimization Algorithm (GOA) and Particle Swarm Optimization (PSO). This section provides an overview of seminal and recent works, assesses their strengths and weaknesses, and identifies the gaps that this research aims to address.

2.1. Seminal Works in GOA and PSO

The Gazelle Optimization Algorithm, detailed in foundational works by its creators, draws inspiration from the evasive maneuvers of gazelles when

escaping predators. This biological inspiration is modeled through unique mathematical approaches like Levy flights for exploration and Brownian motion for exploitation phases [1]. These mechanisms allow GOA to adaptively navigate the search space, balancing between global exploration and local exploitation effectively.

In contrast, Particle Swarm Optimization, introduced by Kennedy and Eberhart, mimics social behavior patterns observed in flocks of birds or schools of fish [2]. Each particle in the swarm adjusts its trajectory based on its own best position and the best-known positions of its neighbors, combining cognitive and social components to converge towards optimal solutions.

2.2. Integration of Hybrid Meta-heuristic Approaches

Hybrid approaches that combine features of both GOA and PSO have started to emerge, aiming to capitalize on the strengths of each. One example is the integration of GOA's effective exploration tactics with PSO's efficient exploitation strategies. This synthesis aims to overcome the individual limitations of each algorithm and enhance overall algorithmic performance [3].

2.3. Strengths and Weaknesses of Current Approaches

Strengths: Both GOA and PSO have shown considerable success in various optimization tasks. GOA's strength lies in its robust exploration capabilities, allowing it to avoid local optima effectively. PSO, on the other hand, excels in rapid convergence towards high-quality solutions thanks to its exploitation strategy driven by collective intelligence.

Weaknesses: Despite their advantages, both algorithms exhibit specific weaknesses. GOA, while effective in exploration, can suffer from slow convergence in complex landscapes due to its random walk exploitation strategy. PSO, although fast at converging, can prematurely converge to suboptimal solutions if not properly balanced between exploration and exploitation.

2.4. Identified Gaps and Research Objectives

While recent literature has explored the potential of hybridizing GOA and SA [4] [5], there remains a significant gap in systematically combining their exploration and exploitation mechanisms to enhance efficiency and solution quality simultaneously. Most hybrid approaches have focused on leveraging either exploration or exploitation but not both in a balanced manner.

This research aims to fill this gap by developing a novel hybrid algorithm that integrates the exploration capabilities of GOA with the robust exploitation strategies of a different algorithm PSO. The goal is to create an algorithm that not only converges faster but also maintains a high quality of solutions across diverse optimization problems. This will potentially reduce computational costs and improve applicability in real-world scenarios.

Research Objectives:

1. To develop a hybrid GOA-PSO algorithm that minimizes convergence time while ensuring high-quality solutions.
2. To evaluate the performance of the hybrid algorithm against standard GOA and PSO in terms of speed, efficiency, and solution accuracy.
3. To demonstrate the applicability of the hybrid algorithm in solving complex, real-world optimization problems.

By addressing these objectives, this research not only contributes to the theoretical understanding of hybrid metaheuristic algorithms but also enhances practical applications in optimization tasks.

3. Gazelle Optimization Algorithm (GoA)

3.1. Overview

Gazelle Optimization Algorithm is a metaheuristic algorithm that models the survival ability of the Gazelles being among the top prey for predators in their natural habitat. It is an Evolutionary algorithm based on Darwin's law of evolution and competence to solve optimization problems.[1] To Solve optimization Problems a population of of Gazelles is modelled using a matrix lets call it X . The members of the modelled population perform either of two characteristic behaviours that help us solve problems.

3.2. Population Initialization

The Gazelle Optimization Algorithm (GOA) is a collective-based optimization strategy that commences with the random establishment of search entities, termed gazelles (X), within the solution space. These gazelles are represented by an $n \times d$ matrix of candidate solutions, where n characterizes the number of gazelles and d epitomizes the dimensions of the search domain. The initial positions of the gazelles are governed by the following equation:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,d} \\ x_{2,1} & x_{2,2} & \dots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,d} \end{bmatrix} \tag{1}$$

In this matrix, $x_{i,j}$ symbolizes the position of the i-th gazelle within the j-th dimension, formulated stochastically via the equation[1]:

$$x_{i,j} = \text{rand} \times (UB_j - LB_j) + LB_j \tag{2}$$

Herein, rand denotes a uniformly distributed random scalar between 0 and 1, while UB and LB represent the upper and lower bounds of the problem scope, respectively. The fittest gazelles, demonstrating superior detection abilities and evasion from predators, contribute to the formation of the Elite matrix, symbolizing the optimum positions located thus far:

$$\text{Elite} = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,d-1} & x_{1,d} \\ x_{2,1} & x_{2,2} & \dots & x_{2,d-1} & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,d-1} & x_{n,d} \end{bmatrix} \tag{3}$$

The Elite matrix is pivotal in the GOA as it steers the search for new potential solutions by guiding the gazelles' movement within the search terrain.

3.3. Behaviours

Behaviours of Gazelle that are relevant to us are its Grazing Behaviour and Escaping Behaviour.[6]

3.3.1. Grazing Behaviour(Exploitation)

During this phase gazelles eat grass from the ground in a random non predictable pattern. This can be modeling used a Mathematical . The best method to do so would be to use Browniam Motion[7]. A random motion in which the displacement follows the Normal (Gaussian) probability distribution function with specific mean and unit variance of $\mu = 0$ and $\sigma = 1$, respectively. At point x, the standard Brownian motion is defined in Eq. (1)

$$f_B(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \tag{4}$$

3.3.2. Escaping Behaviour (Exploration)

During the phase the gazelles when detecting the presence of a predator they run away in a randomwalk far in a straight line this can be modelled used a large step randomwalk likme levy flights function: Random walks characterized by Lévy statistics are employed in the proposed optimization approach, leveraging the advantageous properties of Lévy distributions[8] which possess a heavy tail. The mathematical formulation for the flight behavior of a gazelle in pursuit by predator is encapsulated in the given equation:

$$L(x;\alpha) = |x|^{-\alpha} \tag{5}$$

[9]

Here, x signifies the step length of the flight, and α denotes the exponent of the power law, confined within $[1, 2]$. The distribution of Lévy is expressed through an integral equation:

$$f_L(x;\alpha,\gamma) = \frac{1}{\pi}\int_0^\infty \exp(-\gamma^\alpha q^\alpha)\cos(qx)dq \tag{6}$$

In this context, α pertains to the distributional parameter dictating the trajectory of movement, while γ symbolizes the scale factor of the Lévy motion. The algorithm engages a particular method to generate stable Lévy distributions with values for α ranging from 0.3 to 1.99. The algorithm's implementation of this distribution is depicted by:

$$\text{Lévy}(\alpha) = 0.05 \times \frac{x}{|y|^{\frac{1}{\alpha}}} \tag{7}$$

The variables x and y stem from a normal distribution where x adheres to a distribution with zero mean and variance σ_x^2, while y corresponds to a similar distribution with σ_y^2. The parameters σ_x and σ_y are derived through the subsequent relations:

$$\sigma_x = \left[\Gamma(1+\alpha)\sin\left(\frac{\pi\alpha}{2}\right)\right]^{-\frac{1}{\alpha}}, \quad \sigma_y = 1 \tag{8}$$

3.4. Combine use to make a functioning algorithm

In the iterative process of the proposed optimization algorithm, each iteration confronts a decision point where the algorithm must choose between engaging in exploration or exploitation. This decision is made with equal

likelihood, thus ensuring a balanced approach throughout the optimization journey.

3.4.1. Exploitation Phase

During the exploitation phase, the algorithm emulates the gazelles' behavior when no predators are present. The gazelles move in deliberate and controlled patterns, akin to a Brownian motion, to forage the domain efficiently. This behavior is formalized in the following equation:

$$\text{gazelle}_{i+1} = \text{gazelle}_i + S \cdot R \cdot \hat{R}_B \cdot (\text{Elite}_i - R_B \cdot \text{gazelle}_i) \tag{9}$$

3.4.2. Exploration Phase

Conversely, the exploration phase is triggered when the presence of a predator is detected. The gazelles react with abrupt movements, symbolized by Lévy flights[10][11], to escape the threat. The corresponding equation is:

$$\text{gazelle}_{i+1} = \begin{cases} \text{gazelle}_i + CF \cdot (\text{Elite}_i - R_B \cdot \text{gazelle}_i), & \text{if } r \geq PSRs \\ \text{gazelle}_i + \hat{U} \cdot (LB + r \cdot (UB - LB)), & \text{if } r < PSRs[6] \end{cases} \tag{10}$$

3.4.3. Iterative Selection[12][13]

Each iteration begins with a random binary decision to determine the phase to be executed. If a random number r generated uniformly in the interval $[0, 1]$ is less than 0.5, the algorithm proceeds with exploitation; otherwise, it undertakes exploration:

$$\text{Phase} = \begin{cases} \text{Exploitation}, & \text{if } r < 0.5 \\ \text{Exploration}, & \text{otherwise} \end{cases} \tag{11}$$

Through this probabilistic approach, the algorithm ensures a versatile adaptation to the optimization landscape, mimicking the dynamic and unpredictable nature of gazelles' response to environmental cues. This keeps repeating till a convergence criteria is met.

To review the Detailed algorithm and pseudocode refer to the original seminal paper of Gazelle Optimization Algorithm [1][5]

4. Particle Swarm Optimization (PSO)

4.1. Overview

Particle Swarm Optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves optimization problems by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced primarily by its local best known position but is also guided toward the best known positions in the search space, which are updated as better positions are found by other particles.[2][3]

4.2. Population Initialization

In PSO, each particle is initialized with a random position and a random velocity. The particles explore the problem space by adjusting their positions based on their own experience and that of their neighbors. Let X_i and V_i denote the position and velocity of the i-th particle in a d-dimensional space. The initial values are given by:

$$X_i^0 = LB + \text{rand}() \cdot (UB - LB) \tag{12}$$

$$V_i^0 = -|UB - LB| + 2 \cdot \text{rand}() \cdot |UB - LB| \tag{13}$$

Here, UB and LB are the upper and lower bounds of the problem space, respectively.

4.3. Behaviours

It can be understood as each particle wanting to be the part of spawn hence always tends to move towards the global minima(gBest)(Cognitive). while also being best placed in its local minima (pBest)(social)

4.3.1. Exploitation (Cognitive Component)

During exploitation, each particle adjusts its velocity and position to move towards its personal best position found so far, referred to as *Pbest*. This behavior is modeled by adjusting the velocity of each particle towards *Pbest* using a cognitive coefficient c_1 and a random factor r_1:

$$V_{i+1} = w \cdot V_i + c_1 \cdot r_1 \cdot (Pbest_i - X_i) \tag{14}$$

4.3.2. Exploration (Social Component)

In the exploration phase, particles are influenced by the global best position known, called *Gbest*, encouraging them to explore new areas of the search space. This is facilitated by updating the velocity to include a social component:

$$V_{i+1} = V_i + c_2 \cdot r_2 \cdot (Gbest - X_i) \tag{15}$$

Here, c_2 is the social coefficient, and r_2 is a random number between 0 and 1. The parameter w is the inertia weight that controls the impact of the previous velocity of the particle on its current one, aiding in the balance between exploration and exploitation.

4.4. Algorithm Execution

While execution the Algorithm alternates between exploration and exploitation stages till a specific convergence criteria is met

4.4.1. Exploitation Phase

In this phase, particles update their velocities and positions primarily based on their own best experiences.

4.4.2. Exploration Phase

Conversely, in the exploration phase, the movement is largely influenced by the collective experience of the swarm, pushing particles towards the best solution found by any particle.

To review a detailed description of Particle swarm Optimization please refer to seminal papers in this field [2][3][14]

5. Hybrid Optimization Algorithm (HoA)

5.1. Motivation

In our research, we implemented the Gazelle Optimization Algorithm (GoA) in high-dimensional optimization contexts. A significant limitation identified was the algorithm's sensitivity to the scaling of its constant parameters. Specifically, values of these constants must be maintained below 0.1 to prevent exponential growth towards infinity, rendering the algorithm ineffective. This constraint necessitates extremely small step sizes, consequently increasing the number of iterations required to achieve convergence.

The operational phases of GoA also exhibit a notable imbalance. The exploration phase is sophisticated, incorporating advanced strategies such as Lévy flights, a dynamically adjusting control factor (CF) that modulates step size as iterations progress, and a predator survival rate (PSR) of 0.34, which mirrors the natural survival rate of gazelles escaping predators. In stark contrast, the exploitation phase is considerably less complex, employing a simplistic random walk reminiscent of Brownian motion.

This disparity has led us to conclude that while the exploration phase is well-developed, the simplistic nature of the exploitation phase is a critical bottleneck. Enhancing the complexity of the exploitation phase could significantly reduce the overall number of iterations required for convergence. Developing a hybrid algorithm that introduces more sophisticated mechanisms in the exploitation phase, while retaining the effective strategies of the exploration phase, could yield a more balanced and efficient tool for high-dimensional optimization challenges.

This proposed hybrid approach aims not only to address the inefficiencies identified in the current iteration of GoA but also to integrate the strengths of various optimization techniques to create a more robust and versatile solution

5.2. Methodology

5.2.1. Design Though Flow

This research introduces a hybrid algorithmic approach that integrates the strengths of the Particle Swarm Optimization (PSO) with the Gazelle Optimization Algorithm (GoA) to address inefficiencies observed in GoA's exploitation phase[3]. This methodology is specifically designed to enhance performance in high-dimensional optimization problems, aiming for reduced computational iterations and improved solution accuracy.

5.2.2. Rationale for Algorithm

5.2.2.1 Identifying the Weakness in GoA

The initial phase of our methodology focuses on pinpointing the critical weaknesses of GoA, particularly in its exploitation stage, which is primarily characterized by a simplistic random walk (Brownian Motion). This inefficiency leads to excessive computational iterations with minimal progress towards optimization.

5.2.2.2 Comparative Analysis for a Suitable Algorithm

Upon recognizing the need to augment the exploitation phase of GoA, we conducted a comparative analysis to identify an algorithm with robust exploitation capabilities. Our criteria were to find an algorithm with compatible characteristics to GoA but with a more intricate exploitation mechanism. This analysis highlighted PSO as an ideal candidate due to its advanced exploitation strategies, which include both cognitive (individual learning towards the global best, gBest) and social (collective learning towards personal bests, pBest) components.

5.2.2.3 Selection of PSO for Enhanced Exploitation

PSO's comprehensive exploitation capabilities, combined with its ability to integrate seamlessly into GoA's exploration framework, make it a superior choice for hybridization. PSO enhances the population-based approach of GoA with its sophisticated interaction between particles, effectively improving the overall exploitation phase.

5.2.2.4 Developing a Hybrid Metaheuristic Approach

With PSO identified as the enhancement tool for GoA's exploitation phase, the next step involves detailing the hybridization process. This includes defining how both algorithms will interact, determining the transition points from exploration to exploitation, and ensuring that the strengths of each are maximized within the hybrid model.

5.3. Detailed Methodology : Step by Step

1. Initialization of Algorithm Parameters : Both algorithms are initialized with their respective parameters, which are adjusted to ensure compatibility and optimal performance in their roles within the hybrid framework. This includes setting population sizes, cognitive and social coefficients for PSO, and step size controls for GoA.
2. Implementation of GoA for Global Exploration : The hybrid algorithm begins with GoA operating in the exploration phase, effectively scanning the problem space for potential regions of interest using strategies like Lévy flights coupled with predator survival instincts.
3. Transition to PSO for Local Exploitation : Once a promising area is identified by GoA, PSO takes over to refine the search. It employs its velocity and position update mechanisms to fine-tune the search

process, drawing on both individual and collective memory to approach the optimal solution efficiently.

4. Iterative Hybrid Processing and Convergence Criteria : The algorithm alternates between the GoA for broad exploration and the PSO for intensive exploitation, iterating until it meets predefined convergence criteria such as stagnation or a set number of iterations.
5. Output and Evaluation of the Optimal Solution : The process culminates in identifying the optimal solution, which is the result of an efficient blend of exploration and exploitation, achieved through the synergistic integration of GoA and PSO.

5.4. Mathematically

5.4.1. Exploration

For Exploration The Mathematics remains unchanged and same as that of Gazelles Exploration. We use

$$\text{gazelle}_{i+1} = \text{gazelle}_i + S \cdot R \cdot \hat{R}_B \cdot (\text{Elite}_i - R_B \cdot \text{gazelle}_i) \tag{16}$$

and This

$$\text{gazelle}_{i+1} = \begin{cases} \text{gazelle}_i + CF \cdot (\text{Elite}_i - R_B \cdot \text{gazelle}_i), & \text{if } r \geq PSRs \\ \text{gazelle}_i + \hat{U} \cdot (LB + r \cdot (UB - LB)), & \text{if } r < PSRs \end{cases} \tag{17}$$

5.4.2. Exploitation

For Exploitation we Modifiy the PSO Exploitaion function slightly to accept Gazelles Exploration as output. We get the equation.

$$V_{i+1} = w \cdot V_i + c_1 \cdot r_1 \cdot (Pbest_i - X_i) \tag{18}$$

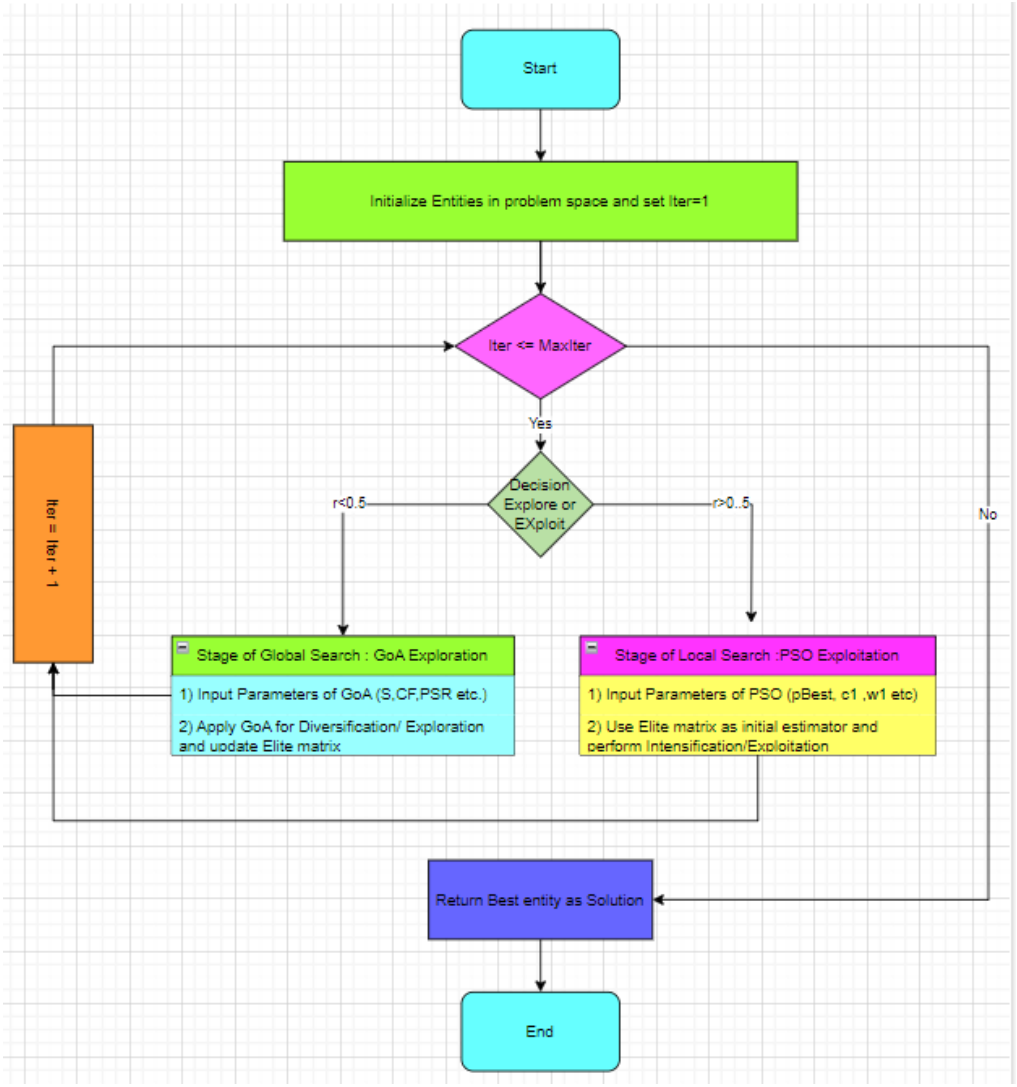
Where X_i is Elite Matrix From GoA This has made the exploitation step more complex and also introduced an new Hyper parameter c_1 (Cognitive Component) to our algorithm. In Modelling terms it has introduced a behaviour of Gazelleing not randomly looking for food while grazing but actively observing and moving towards the area where more food is present

5.5. PseudoCode

Algorithm 1 Hybrid GoA-PSO Algorithm

Input: Problem Space
Initialize entities in Problem Space
Set iteration $iter \leftarrow 1$
Set maximum iterations $MaxIter$
Output: Best Entity as Solution
while $iter \leq MaxIter$ **do**
    // Decision: Explore or Exploit
    Generate random number r
    if $r < 0.5$ **then**
        // Stage of Global Search: GoA Exploration
        Input parameters of GoA (S, CF, PSR, etc.)
        Apply GoA for Diversification/Exploration
        Update Elite matrix
    else
        // Stage of Local Search: PSO Exploitation
        Input parameters of PSO ($pBest$, $gBest$, w, etc.)
        Use Elite matrix as initial estimator
        Perform Intensification/Exploitation
    end if
    Update $iter \leftarrow iter + 1$
end while
Return Best Entity as Solution

5.6. Flow Chart



6. Results

6.1. Methodology

6.1.1. Implementation of Algorithms

In this section, we focused on evaluating the performance of a hybridized algorithm by first implementing both the proposed and original algorithms in Python, as described in the prior sections. Efforts were made to accurately

replicate the algorithms based on their descriptions in the original papers. This replication was crucial to ensure that any observed differences in performance were due to the hybridization itself rather than discrepancies in implementation.

6.1.2. Consistency of Execution

To maintain consistency in our experiments, we carefully managed the step sizes throughout the execution of the algorithms. This control was essential to prevent these parameters from affecting the results. We adopted the same set of 15 standard benchmark functions that were used in the original study, allowing for a direct comparison of performance. The Software and Hardware environment to run the experiments on are kept consistent. RandomSeed is consistent for each run for repeatability purposes.

6.1.3. Data Collected and Analysis

Each algorithm was run for 1,000 iterations (Original Paper also ran for 1000 iterations), with the fitness recorded at every iteration to monitor the progression and efficiency of the algorithms over time. This granularity enabled us to plot detailed fitness curves and analyze the convergence behavior of the algorithms. We will observe the average best fitness values across 10 runs with different initial parameters and random seeds.This is done to ensure that randomness in our results can be reduced

ID	Type	Function	Dimension	Bounds	Global
F1	Unimodal	$f(x) = \sum_{i=1}^{n} x_i^2$	30	[− 100,100]	0
F2	Unimodal	$f(x) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	30	[− 10,10]	0
F3	Unimodal	$f(x) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	30	[− 100,100]	0
F4	Unimodal	$f(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	[− 100,100]	0
F5	Unimodal	$f(x) = \sum_{i=1}^{n-1} \left[100(x_i - x_{i+1})^2 + (1 - x_i)^2 \right]$	30	[− 30,30]	0
F6	Unimodal	$f(x) = \sum_{i=1}^{n} (\lvert x_i - 0.5 \rvert)^2$	30	[− 100,100]	0
F7	Unimodal	$f(x) = \sum_{i=0}^{n} i x_i^4 + \mathrm{rand}[0, 1)$	30	[− 128,128]	0
F8	Multimodal	$f(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert} \right)$	30	[− 500,500]	$-418.9829 \times n$
F9	Multimodal	$f(x) = 10 + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	30	[− 5.12,5.12]	0
F10	Multimodal	$f(x) = -a \exp\left(-0.02 \sqrt{n^{-1} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(n^{-1} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + a + e, a = 20$	30	[− 32,32]	0
F11	Multimodal	$f(X) = 1 + \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right)$	30	[− 600,600]	0
F12	Multimodal	$f(x) = \frac{\pi}{n} \{ 10 \sin(\pi y_i) \} + \sum_{i=1}^{n-1} (y_i - 1)^2 \left[1 + 10 \sin^2(\pi y_{i+1}) + \sum_{i=1}^{n} u(x_i, 10, 100, 4) \right]$ Where $y_i = 1 + \frac{x_i + 1}{4}, u(x_i, a, k, m) \begin{cases} K(x_i - a)^m & if x_i > a \\ 0 & -a \le x_i \ge a \\ K(-x_i - a)^m & -a \le x_i \end{cases}$	30	[− 50,50]	0
F13	Multimodal	$f(x) = 0.1 \left(\sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 1 + \sin^2(2\pi x_n) \right) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[− 50,50]	0
F14	Fixed-dimension multimodal	$f(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	[− 5,5]	0.00030
F15	Fixed-dimension multimodal	$f(X) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right] \times \left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \right]$	2	[− 2,2]	3

Function	Avg. Best Fitness GoA	Std. Best Fitness GoA	Avg. Best Fitness GoA_PSO	Std. Best Fitness GoA_PSO
F1	7241.46	1139.05	0.0094	0.016
F2	36.54	3.046	2.2620	2.134
F3	98116.38	15823.34	0.1368	0.22
F4	33.05	3.23	19.5227	4.53
F5	55795.00	11329.41	812.32	425.89
F6	8167.61	1170.17	0.0392	0.08
F7	11990863.74	5140868.25	25.21	19.54
F8	7872.17	609.237	5297.67	581.89
F9	182.8	12.44	127.7	22.09
F10	14.09	0.41	10.01	1.21
F11	63.37	11.92	0.7491	0.21
F12	481.94	115.82	246.6327	85.22
F13	233.36	49.19	96.925	71.87
F14	0.001	0.0005	0.001	0.001
F15	3.008	0.01	3.0000	0.001

6.3. Comparitive Performamce Table

Given this generated Raw data we can apply independent paired t-test to see which algorithm performed better. Here since lesser mean implies better fitness that is algorithm that can bring fitness closer to zero (global minimum) faster is better algorithm T-test is to be interpreted in such a way that higher the T Statistic with significant p value worse the original Algorithm performs. The equation for the paired t-test statistic is given by:

$$t = \frac{\overline{D}}{s_D/\sqrt{n}} \tag{19}$$

where $\overline{D}$ is the mean of the differences between all pairs of observations, s_D is the standard deviation of these differences, and n is the number of pairs.

Table 1: Comparison of Two Optimization Algorithms Across Functions[15]

Function	**T-statistic**	**P-value**	**Interpretation**
F: 1	19.072	1.38×10^{-8}	Second better
F: 2	27.131	6.08×10^{-10}	Second better
F: 3	18.602	1.72×10^{-8}	Second better
F: 4	6.389	0.000127	Second better
F: 5	14.844	1.24×10^{-7}	Second better
F: 6	20.940	6.05×10^{-9}	Second better
F: 7	6.997	6.34×10^{-5}	Second better
F: 8	6.970	6.54×10^{-5}	Second better
F: 9	6.219	0.000155	Second better
F: 10	8.872	9.60×10^{-6}	Second better
F: 11	15.772	7.29×10^{-8}	Second better
F: 12	4.296	0.002001	Second better
F: 13	4.711	0.001102	Second better
F: 14	NaN	NaN	Inconclusive
F: 15	1.520	0.163	No difference

6.4. Detailed Discussion of Significant Results

In our study, we employed paired t-tests to evaluate the comparative performance of two optimization algorithms across a spectrum of 15 distinct objective functions. The results significantly demonstrated that the second algorithm consistently outperformed the GoA in nearly all the tested functions. This was evidenced by significant t-statistic values in 13 out of 15 cases, with p-values well below the standard significance level of 0.05. Particularly notable were functions F2 and F6, where the t-statistics were 27.131 and 20.940, respectively, with corresponding p-values indicating extremely strong statistical significance. These results suggest a robust superiority of the second algorithm in optimizing these specific functions.

To further analyze our findings, we plotted the top five most significant functions to compare how each algorithm performed on average across ten runs. The convergence analysis revealed that the hybrid algorithm achieved convergence much earlier than the GoA, underscoring its superior convergence rate.

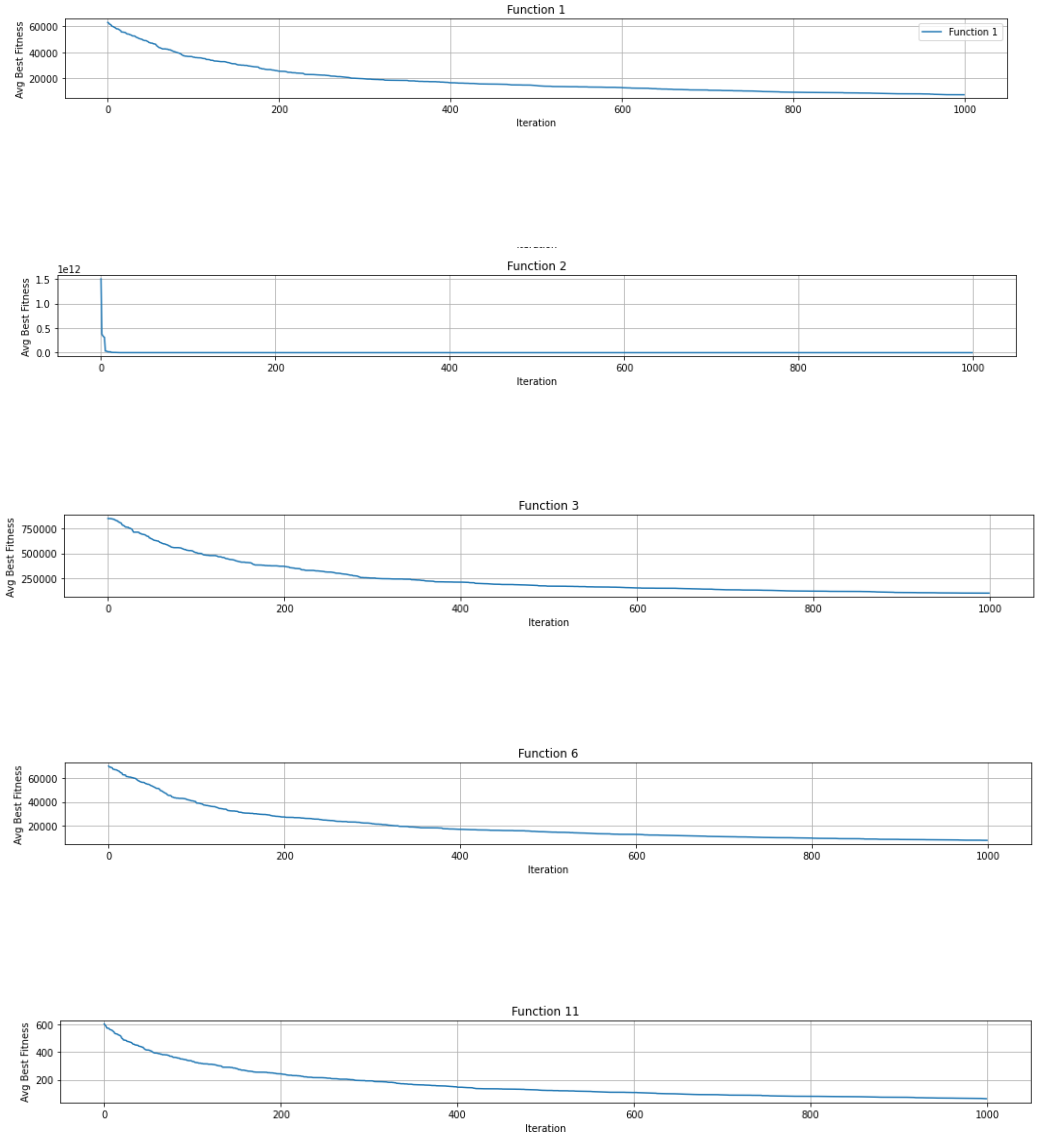


Figure 1: Plot for Average Iteration of Functions for Gazelle Algorithm

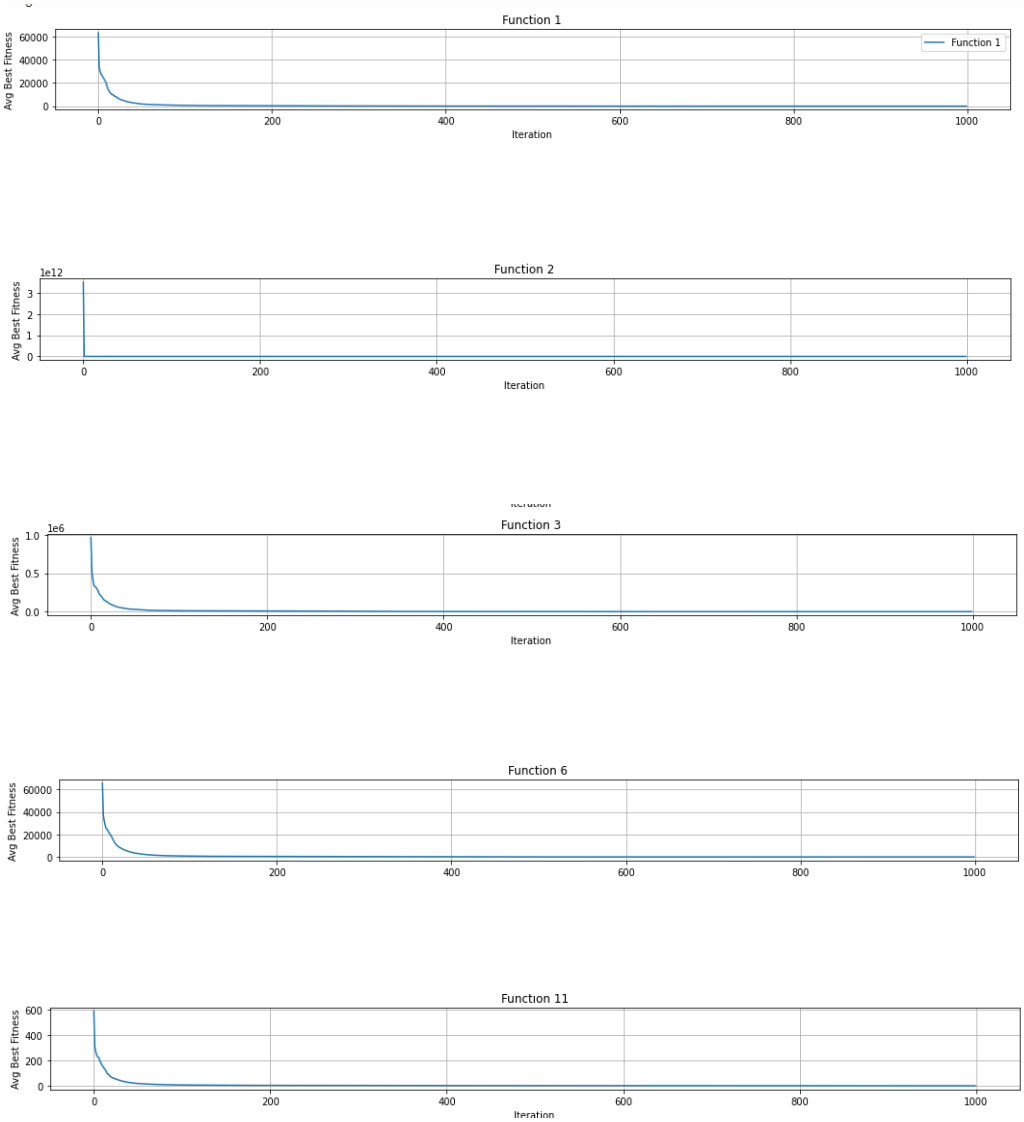


Figure 2: Plot for Average Iteration of Functions for Hybrid Algorithm

Our line graphs and heat-maps illustrate the performance disparities more vividly. The hybrid algorithm not only performs better on standard benchmark functions but also shows superior performance on custom functions. Notably, the variability in the value of the best fitness is considerably lower, suggesting that our algorithm is more generalizable than the GoA.

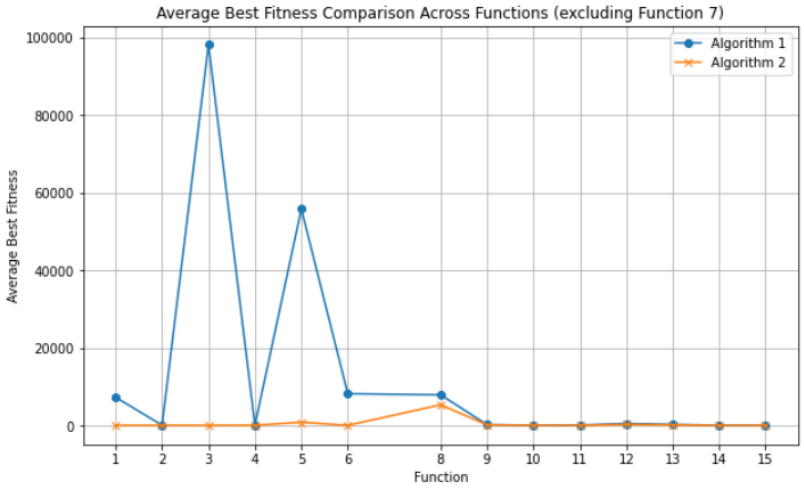


Figure 3: Line graph comparison of the average best fitness per function over runs between the GoA and the hybrid algorithm.

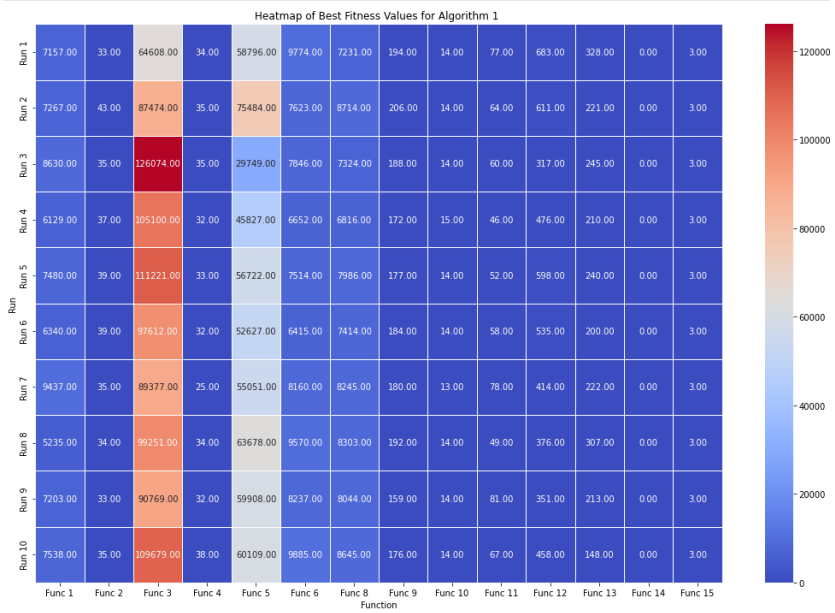


Figure 4: Heatmap of best fitness per run for each function for the GoA.

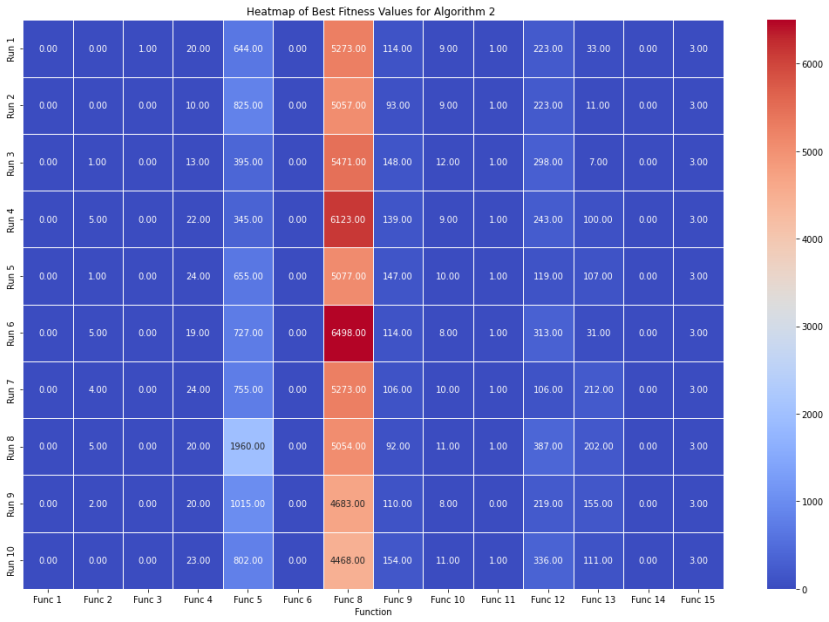


Figure 5: Heatmap of best fitness per run for each function for the hybrid algorithm.

A comparison of the heatmaps between the two algorithms reveals that the hybrid algorithm exhibits a less pronounced color gradient than the GoA, except for function F8 (the notorious Schwefel function), which poses significant challenges to optimization algorithms. The marked gradient difference in the GoA, combined with higher fitness values for each cell in the heatmap, allows us to conclude that the hybrid algorithm is more stable and reliable, exhibiting less variability in performance.

For functions F14 and F15, the results were inconclusive and insignificant, respectively, as both algorithms were able to reduce the best fitness to the true global minimum values in relatively few iterations. This outcome may be attributed to the nature of the custom function used in the original paper and the very small bounds, which are very close to the global minimum.

Implications of Findings: The practical implications of these findings are profound, especially in the context of optimization, where achieving closer to zero mean fitness signifies superior performance. The consistently lower mean fitness scores achieved by the second algorithm across a majority of the functions highlight its effectiveness and reliability under diverse conditions.

Our hybrid algorithm's significant performance margin over the GoA can

be attributed to the replacement of the GoA's inefficient exploitation stage with a more efficient step derived from insights of Particle Swarm Optimization (PSO). While most of the algorithm was retained, replacing an inefficient section with more efficient techniques led to predictable improvements, thus verifying our hypothesis that hybridizing the Gazelle Optimization Algorithm with PSO would yield significant enhancements.

7. Conclusion and Future Works

This study presented a hybridized version of the Gazelle Optimization Algorithm (GoA), enhanced by incorporating exploitation strategies from Particle Swarm Optimization (PSO). Extensive benchmark testing across multiple objective functions demonstrated that the hybrid algorithm significantly reduces the number of iterations required for convergence, achieving a reduction in convergence time by approximately 10 to 12 times compared to the original GoA. This substantial improvement not only addresses efficiency concerns in the recently developed GoA but also highlights the effective synergy between GoA's exploration mechanisms and PSO's exploitation tactics.

The improved performance underscores the potential of hybrid metaheuristic approaches in solving complex optimization problems more efficiently. This research validates the hypothesis that integrating complementary strategies from different algorithms can lead to substantial improvements in algorithmic speed and efficiency without compromising the quality of solutions.

Looking ahead, several promising avenues for further research exist. First, the increased complexity introduced by the hybridization warrants a deeper investigation into the trade-offs between computational time and iteration reduction. Future studies could explore the scalability of the hybrid algorithm across more diverse and high-dimensional problem sets to validate its robustness and versatility.

Adapting the hybrid GoA-PSO algorithm for real-world applications, such as engineering design and resource management, would provide valuable insights into its practical effectiveness. Additionally, there is potential to incorporate adaptive parameter tuning mechanisms that could dynamically adjust exploration and exploitation parameters in response to the problem's landscape, potentially enhancing the algorithm's performance further.

Moreover, exploring the integration of other metaheuristic elements, such as those from genetic algorithms or ant colony optimization, could offer new

perspectives on further enhancing the GoA. Each of these directions not only promises to expand the capabilities of GoA but also contributes to the broader field of optimization by developing more adaptive and efficient algorithms.

Appendix A. Summary

This research paper presents a significant enhancement to the Gazelle Optimization Algorithm (GoA) through a novel hybridization with Particle Swarm Optimization (PSO) techniques. The primary aim of this innovation was to reduce the number of iterations required for convergence, thereby addressing efficiency limitations observed in the newly developed GoA. By integrating the exploration mechanisms of GoA with the exploitation strategies of PSO, the modified algorithm achieved a dramatic reduction in convergence time, specifically by an approximate factor of 10 to 12.

Extensive empirical analysis was conducted, employing paired t-tests across 15 distinct objective functions to compare the performance of the original and the hybrid algorithms. The results were overwhelmingly in favor of the hybrid algorithm, which consistently outperformed the original in 13 out of 15 tested functions.

The hybrid algorithm's superior performance is primarily due to the integration of a more efficient exploitation phase derived from PSO, which replaced less efficient elements of the original GoA. This modification not only confirmed the hypothesized benefits of hybridizing GoA with PSO but also demonstrated predictable improvements due to replacing inefficient components with more robust mechanisms.

The findings have substantial practical implications, especially in fields requiring efficient optimization solutions, as the hybrid algorithm demonstrated remarkable reliability and effectiveness across diverse conditions. The paper concludes with suggestions for future research that could explore further refinements and applications of this hybrid approach, potentially extending its utility to more complex optimization challenges and real-world scenarios.

References

[1] J. O. Agushaka, A. E. Ezugwu, L. Abualigah, Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer, Neural Computing and Applications 35 (5) (2023) 4099–4131.

[2] R. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceedings of the IEEE international conference on neural networks, Vol. 4, Citeseer, 1995, pp. 1942–1948.

[3] R. Thangaraj, M. Pant, A. Abraham, P. Bouvry, Particle swarm optimization: Hybridization perspectives and experimental illustrations, Applied Mathematics and Computation 217 (12) (2011) 5208–5226.

[4] S. Kirkpatrick, C. D. Gelatt Jr, M. P. Vecchi, Optimization by simulated annealing, science 220 (4598) (1983) 671–680.

[5] S. Ekinci, D. Izci, Enhancing iir system identification: Harnessing the synergy of gazelle optimization and simulated annealing algorithms, e-Prime-Advances in Electrical Engineering, Electronics and Energy 5 (2023) 100225.

[6] K. A. Olson, E. A. Larsen, T. Mueller, P. Leimgruber, T. K. Fuller, G. B. Schaller, W. F. Fagan, Survival probabilities of adult mongolian gazelles, The Journal of wildlife management 78 (1) (2014) 35–41.

[7] A. Einstein, Investigations on the Theory of the Brownian Movement, Courier Corporation, 1956.

[8] R. N. Mantegna, Fast, accurate algorithm for numerical simulation of levy stable stochastic processes, Physical Review E 49 (5) (1994) 4677.

[9] B. H. Abed-alguni, D. Paul, Island-based cuckoo search with elite opposition-based learning and multiple mutation methods for solving optimization problems, Soft Computing 26 (7) (2022) 3293–3312.

[10] E. H. Houssein, M. R. Saad, F. A. Hashim, H. Shaban, M. Hassaballah, Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems, Engineering Applications of Artificial Intelligence 94 (2020) 103731.

[11] X.-S. Yang, S. Deb, Cuckoo search via lévy flights, in: 2009 World congress on nature & biologically inspired computing (NaBIC), Ieee, 2009, pp. 210–214.

[12] A. Tzanetos, G. Dounias, Nature inspired optimization algorithms or simply variations of metaheuristics?, Artificial Intelligence Review 54 (3) (2021) 1841–1862.

[13] X.-S. Yang, M. Karamanoglu, Nature-inspired computation and swarm intelligence: a state-of-the-art overview, Nature-Inspired Computation and Swarm Intelligence (2020) 3–18.

[14] N. M. Laskar, K. Guha, I. Chatterjee, S. Chanda, K. L. Baishnab, P. K. Paul, Hwpso: A new hybrid whale-particle swarm optimization algorithm and its application in electronic design optimization problems, Applied Intelligence 49 (2019) 265–291.

[15] E. AE, A conceptual comparison of several metaheuristic algorithms on continuous optimization problems (2020).