An Improved Golden Eagle Optimiser By: Utilization Of Adaptive Step-size Variability

Prof Vijay Kumar Bohat, Divyansh Saharan, Kushagra Agarwal, Bhavya Jain

Netaji Subhas University of Technology, Dwarka, Delhi, 110078, India

Abstract

This paper introduces a superior variant of the swarm-based metaheuristic algorithm, the Golden Eagle Optimizer (GEO) which was designed for solving global optimization problems. Drawing inspiration from the hunting strategies of golden eagles, GEO emulates the adaptive speed tuning observed in these birds during different stages of their spiral trajectory for hunting. Similar to the eagles' behavior, GEO exhibits a heightened tendency for exploration in the initial stages, gradually transitioning to a more exploitative approach in the final stages of the optimization process. This dynamic adjustment allows GEO to effectively search for and capture optimal solutions within the feasible region in minimal time. The algorithm's behavior was modeled to emphasize both exploration and exploitation aspects crucial for achieving global optimization.

This paper presents an enhanced variant of the aforementioned algorithm, intended to augment the optimizer's efficacy. The proposed variant integrates a novel approach that adjusts the step-size for the eagles' movement, leveraging insights from both their cruising and attacking behavior, alongside enhancements derived from the collective memory of the flock concerning the prevailing best solutions. By incorporating these additional factors, the variant enables dynamic adaptations to the eagles' attack and cruise vectors, aligning them optimally with the evolving demands of successive iterations. This adaptive mechanism facilitates a nuanced balance between exploration and exploitation strategies, informed by the flock's historical progress in traversing previous iterations towards the identification of a global minima for the objective function.

To assess its performance, this GEO variant was subjected to rigorous testing across 30 benchmark test functions of CEC2014 as well as the 30 benchmark test functions CEC2017. Comparative analysis against multiple established algorithms including the original Golden Eagle Optimizer consistently demonstrates the variant's superior efficacy in locating global optima while circumventing local optima effectively.

Keywords: Golden Eagle Optimiser; Meta-Heuristic Algorithm; Adaptive Step-Size; Dynamic Adjusting of Step-Size Based on Improvement in Flock Memory

1. Introduction

Optimization entails the process of determining the optimal state of decision/design variables to maximize or minimize single or multiple objective functions. Prior to the era of heuristic optimization, analytical methods dominated the landscape of mathematical problem-solving. These methods, while efficient in finding the exact optimum for linear or convex non-linear problems, heavily relied on information regarding the derivatives of the objective functions. However, they

were susceptible to local optima entrapment in complex problem landscapes featuring numerous local optima and were unsuitable for problems characterized by stochastic or unknown search spaces [1], which are common attributes of real-world problems. Consequently, the emergence of metaheuristic algorithms became inevitable. Metaheuristic algorithms, being derivative-free and not bound by limiting assumptions, present an appealing solution for addressing a wide array of problem classes [2]. Despite their flexibility, it is important to note that the efficacy of an optimization algorithm on a specific set of problems does not guarantee similar performance on other problems, as articulated by the No Free Lunch (NFL) theorem [3]. However, this theorem has paved the way for the development of novel metaheuristic algorithms. The relative simplicity of understanding and application, coupled with their satisfactory performance, has contributed to the popularity of metaheuristic algorithms [4], with numerous algorithms recently introduced to effectively address business and engineering challenges.

Metaheuristic methods can be categorized based on various criteria. One common approach involves classifying these methods according to their source inspiration, which includes evolutionary, human-based, physics-based, synthetic, and swarm intelligence approaches. Evolutionary algorithms, inspired by the natural selection law of biology, operate by evolving an initial population using evolutionary operators to enhance the population's fitness and identify the global optimum [5,6]. Genetic Algorithm [7] and Differential Evolution [8] are prominent examples of evolutionary algorithms. The human-based approach encompasses algorithms inspired by human social behavior or concepts developed by humans. Examples include Queuing Search Algorithm (QSA) [9], Group Teaching Optimization Algorithm (GTOA) [10], and Teaching-Learning-Based Optimization (TLBO) [11]. Physics-based methods view the problem landscape as a physical phenomenon and move search agents using formulae borrowed from physical rules or theories. Recent algorithms proposed under this approach include Atom Search Optimization (ASO) [12], Henry Gas Solubility Optimization (HGSO) [13], Water Cycle Algorithm (WCA) [14], and others. Synthetic methods are based solely on mathematical equations, such as trigonometric functions or well-known constants, without being inspired by a specific natural phenomenon. Examples include Sine Cosine Algorithm (SCA) [24], Golden Ratio Optimization Method (GROM) [25], and Stochastic Fractal Search (SFS) [26]. Swarm intelligence algorithms mimic the social behavior and communication patterns within groups of species and have gained popularity due to their applicability and continuous development of new algorithms [27,28]. Notable examples include Pathfinder algorithm (PFA) [30] and Harris Hawks Optimization (HHO) [31].

The balance between exploration and exploitation is crucial for the effectiveness of GEO. Excessive exploration may result in an inordinate amount of time being spent on searching less promising regions of the solution space, leading to inefficient convergence. Conversely, excessive exploitation may cause GEO to converge prematurely to sub-optimal solutions, limiting its ability to discover better alternatives.

In the Golden Eagle Optimizer (GEO), achieving a delicate equilibrium between exploration and exploitation is facilitated by the strategic utilization of Attack and Cruise Vectors, inspired by the hunting behaviors of golden eagles. Each eagle within the optimizer is guided by these vectors, with the Cruise vector set perpendicular to the Attack vector. Consequently, every eagle exhibits a simultaneous inclination towards both attacking the current solution and cruising to explore for potentially superior solutions. The trademark characteristics that GEO leverages to maintain the balance between exploration-exploitation are the following:

1. Spiral Trajectory for Search and Straight Path for Attack: Eagles employ a spiral trajectory when searching for a solution and adopt a straight path when executing an attack.

2. Propensity for Cruising in Initial Stages, Transitioning to Attack in Final Stages: Golden eagles display a greater tendency to cruise during the initial phases of search, gradually shifting towards a higher inclination for attacking as the iterations increase.

3. Retention of Tendency for Both Cruise and Attack: Throughout the entirety of the hunting flight, golden eagles maintain a propensity for both cruising to explore new solutions and attacking potential optimas.

4. Information Sharing Among Eagles: Golden eagles gather information from other eagles regarding the best solutions recognized till the current iteration. The best solution obtained by one eagle is often used as prey by some other eagle, thus ensuring exploration the search space around better solutions for the global optima.

Cruise and attack, along with the intelligent balance achieved between these two behaviors by golden eagles, mirror the natural interplay between exploration and exploitation. This inherent balance serves as the foundation for the development of a metaheuristic algorithm such as GEO. By emulating the hunting strategies of golden eagles, GEO optimally navigates solution spaces, effectively combining exploration to discover new regions and exploitation to refine and improve upon existing solutions.

2. Golden Eagle Optimizer

2.1. Attack Vector (Exploitation)

The attack vector ($\vec{A_i}$) represents the direction in which a golden eagle i moves towards its prey, aiming to exploit the best locations visited so far. It is calculated as the difference between the best location visited so far by another eagle f ($\vec{X}^*_f$) and the current position of the eagle i ($\vec{X_i}$). This is formulated as:

$$\vec{A_i} = \vec{X}^*_f - \vec{X_i} \tag{1}$$

This equation indicates that the eagle i moves towards the best location found by eagle f, indicating the exploitation phase of the optimization process.

2.2. Cruise Vector (Exploration)

The cruise vector ($\vec{C_i}$) represents the direction in which a golden eagle i explores the search space, away from the previously explored locations. It is calculated based on the attack vector within the tangent hyperplane to the circle. The cruise vector is perpendicular to the attack vector and represents the linear speed of the eagle relative to the prey. The cruise vector equation within the tangent hyperplane is given as:

$$\sum a_j x_j = \sum a^t_j x^*_j \tag{2}$$

Here, a_j and x_j represent the elements of the attack vector and decision/design variables vector respectively, a^t_j and x^*_j represent the elements of the attack vector and location of the selected prey respectively.

2.3. Moving to New Positions

Golden eagles move to new positions based on both the attack and cruise vectors. The step vector ($\Delta\vec{x_i}$) for golden eagle i in iteration t is a combination of the attack and cruise vectors, adjusted by random coefficients (r_1 and r_2). It's defined as:

$$\Delta\vec{x_i} = \frac{r_1 p_a \vec{A_i}}{\|\vec{A_i}\|} + \frac{r_2 p_c \vec{C_i}}{\|\vec{C_i}\|} \tag{3}$$

Here, $\|\vec{A_i}\|$ and $\|\vec{C_i}\|$ are the Euclidean norms of the attack and cruise vectors respectively, and r_1 and r_2 are random vectors.

2.4. Transition from Exploration to Exploitation

The algorithm transitions from exploration to exploitation by adjusting the attack and cruise coefficients (p_a and p_c) over iterations. Initially, p_a is set to a low value and p_c to a high value. These coefficients are gradually adjusted over iterations using a linear transition formula, where the values shift towards exploitation as iterations progress. The adjustment equations are as follows:

$$p_a = p_a^0 + \frac{t}{T}|p_a^T - p_a^0| \tag{4.1}$$

$$p_c = p_c^0 - \frac{t}{T}|p_c^T - p_c^0| \tag{4.2}$$

Here, p_{a0} and p_{c0} are the initial values for the attack and cruise coefficients, p_a^T and p_c^T are the final values for the coefficients, t indicates the current iteration, and T indicates the maximum iterations.

3. Limitations Of Golden Eagle Optimizer

As it has been repeatedly re-iterated in the previous sections, the balance between exploration and exploitation is critical to the performance of these meta-heuristic algorithms. Herein lies a critical limitation inherent in the Golden Eagle Optimizer (GEO). Algorithms like the Golden Eagle Optimizer are unable to dynamically adjust the propensity for exploration and exploitation based on the results obtained in the previous iterations ie, GEO lacks the capacity to flexibly modify the cruise and attack behaviors in response to evolving problem landscapes.

The rigid structure of GEO's attack and cruise vectors restricts the algorithm's ability to adequately respond to changes in the solution space. These vectors are predefined and remain static throughout the optimization process, with linear variations applied across iterations. Consequently, the algorithm operates within predefined bounds, without the capability to dynamically update these parameters based on the information gained from the current flock memory.

This limitation poses significant challenges in scenarios where the optimal balance between exploration and exploitation shifts over the course of optimization. In dynamic or evolving environments, the efficacy of fixed attack and cruise behaviors may diminish as the algorithm struggles to adequately explore promising regions or exploit discovered solutions optimally.

Without the ability to discern whether increased exploration or exploitation is warranted based on the current flock memory, GEO may exhibit sub-optimal performances, particularly in complex optimization landscapes with nonlinear or dynamic characteristics. The absence of adaptive

mechanisms to fine-tune the exploration-exploitation trade-off restricts the algorithm's ability to swiftly converge to optimal solutions or effectively navigate rugged solution spaces.

Addressing this limitation requires the development of more sophisticated adaptive strategies within GEO, enabling the algorithm to dynamically adjust the attack and cruise behaviors based on real-time feedback from the optimization process. By incorporating mechanisms for intelligent parameter adaptation, GEO could enhance its adaptability to changing problem landscapes and improve its convergence speed and solution quality across diverse optimization tasks. This avenue of research represents a crucial step towards unlocking the full potential of the Golden Eagle Optimizer in tackling complex optimization challenges effectively.

4. The Proposed Adaptive StepSize Variant of Golden Eagle Optimizer Algorithm

The variant of GEO introduced within the scope of this research paper endeavors to address the previously mentioned challenge through a strategic approach. It maintains a vigilant record of enhancements within the flock's collective memory, which encapsulates the current best solutions. Leveraging this repository of insights, the algorithm dynamically adjusts the equilibrium between exploration and exploitation, fostering an adaptive and efficient optimization process.

4.1. Formulating A Solution

Problem-solving entails a comprehensive understanding of the underlying issue, followed by a meticulous examination of its causal factors, culminating in the development of a targeted algorithmic solution. Within the context of the aforementioned challenge, two distinct scenarios emerge, each presenting potential compromises to the algorithm's efficacy.

Firstly, if successive iterations yield substantial enhancements in the flock's memory, it signals a disproportionate tilt towards exploitation, thereby jeopardizing the exploratory phase. This imbalance risks entrapping the algorithm within local optima, impeding its capacity to explore and discern global minima within intricate solution landscapes. Consequently, there arises a pressing demand for heightened exploration strategies to counteract this phenomenon effectively.

Secondly, a lack of improvement in the flock's memory across successive iterations indicates a potential stagnation within the search agents, ensnared within sub-optimal search subspaces. In order to navigate towards optimal solution subspaces, it becomes imperative for the search agents to converge upon the regions surrounding the best solutions identified by the flock up to the current iteration. Hence, a strategic emphasis on exploitation and the deployment of assertive attack vectors emerge as preferred tactics to advance towards optimal solutions.

In essence, by strategically aligning exploration and exploitation dynamics based on real-time feedback from the flock's memory, the proposed variant of GEO exhibits a heightened adaptability and efficacy in navigating complex optimization landscapes.

4.2. Proposed Algorithm

The algorithm commences with the randomized initialization of search agents within the designated search space. Following this initialization, the fitness of each search agent is assessed via the objective function, and their corresponding values are stored within the flock memory, with each eagle allocated a distinct memory location. Subsequently, in each iteration, every golden eagle randomly selects another golden eagle and considers the best location encountered (thus far) by that eagle as its prey. Utilizing the aforementioned equations, the attack vector and cruise

vectors for each eagle are calculated, taking into account the best-known solution visited by another eagle as its prey. The Attack and Cruise Unit Vectors are then determined in accordance with the respective equations (1) and (2) while the Attack and Cruise Propensity are determined by equations (4.1) and (4.2) respectively.

In the original GEO framework, Attack and Cruise Vector and their calculated Propensity were used to calculate the StepVector for the current iteration. Herein lies the integration of adaptive stepsize variation. Our variant utilizes a different equation for calculation of StepVector and introduces one novel parameter, namely Greek Alphabet ω which is our proposed method for dynamically adjusting the propensity for exploration and exploitation based on the results obtained in the previous iterations.
The new modified equation for calculation of StepVector is:

$$\Delta\vec{x_i} = \omega * (\frac{r_1 p_a \vec{A_i}}{||\vec{A_i}||} + \frac{r_2 p_c \vec{C_i}}{||\vec{C_i}||})$$

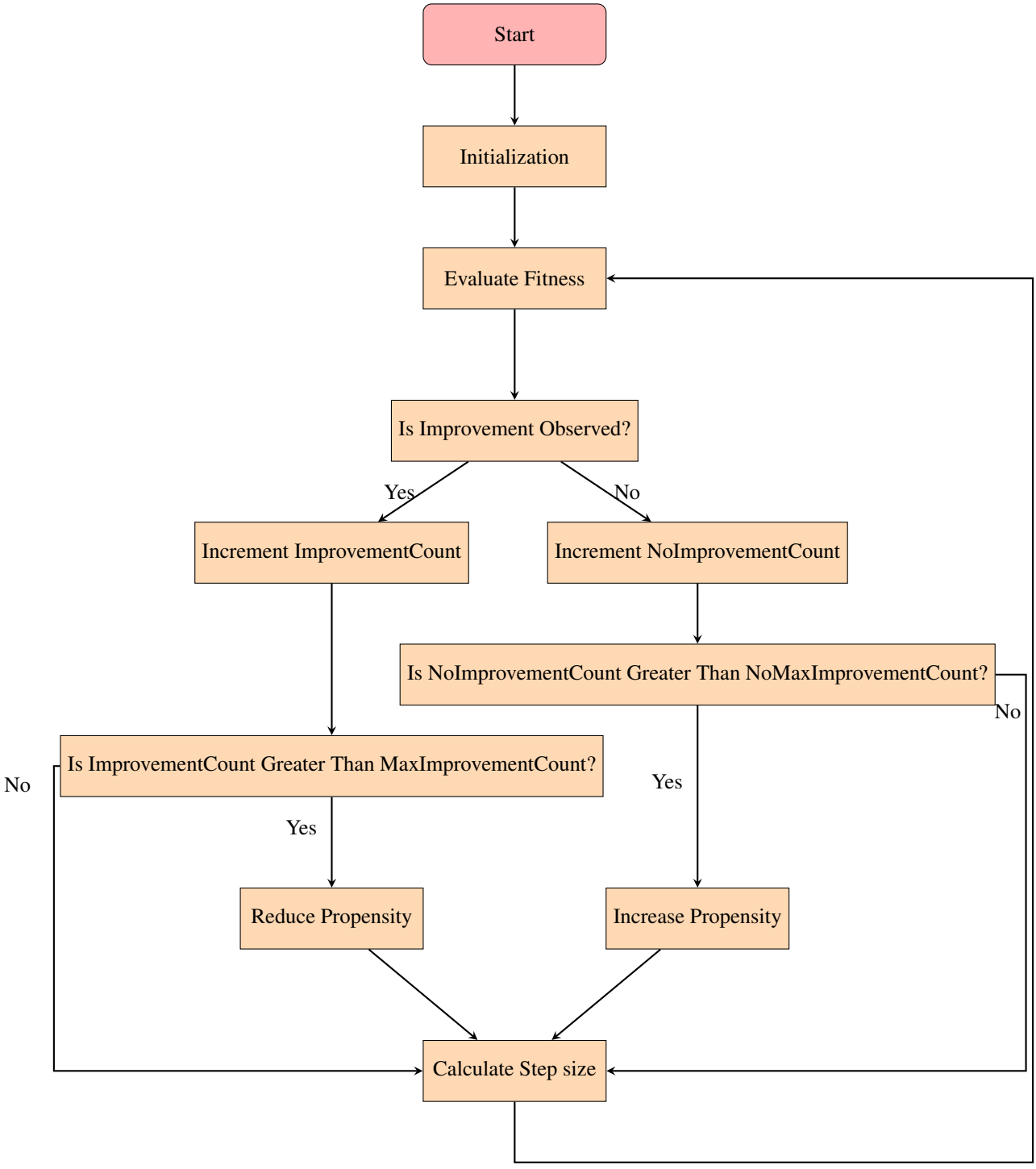
The value for the parameter ω can be calculated by the following equation:

$$\omega = 1 + 0.1 * (\lfloor \frac{ImprovementCount}{MaxImprovementCount} \rfloor) - 0.1 * (\lfloor \frac{NoImprovementCount}{MaxNoImprovementCount} \rfloor)$$

Here, we are using the Greatest Integer Function (which is represented as $\lfloor\ \rfloor$) which is also commonly known as Floor Function.
The above equation introduces two new counters, ImprovementCount and NoImprovementCount which track whether the flock memory improves across successive iterations. Along with that the equations set two constants, namely MaxImprovementCount and MaxNoImprovementCount which hold the count of the maximum number of iterations the algorithm (with improvement and without improvement respectively) can run before the stepsize is dynamically changed. Extensive parameter tuning revealed that setting the constant MaxImprovementCount to 5 and constant MaxNoImprovementCount to 10 yielded optimal results.

So, if ImprovementCount surpasses MaxImprovementCount, indicating a tilt towards exploitation and inadequate exploration of the solution search space, ω is set to 0.9 and the Attack and Cruise Propensity are scaled down facilitating enhanced exploration.

Likewise, if NoImprovementCount exceeds MaxNoImprovementCount, signifying the search agents' entrapment within sub-optimal regions, the stepsize must be augmented for the eagles to navigate towards superior solution subspaces in fewer iterations. Thus, the parameter ω 1.1 for that iteration, fostering improved exploitation of the current best solutions.


Start
Initialization
Evaluate Fitness
Is Improvement Observed?
Yes
No
Increment ImprovementCount
Increment NoImprovementCount
Is NoImprovementCount Greater Than NoMaxImprovementCount?
No
Yes
Is ImprovementCount Greater Than MaxImprovementCount?
No
Yes
Reduce Propensity
Increase Propensity
Calculate Step size

A pseudo-code for the implementation for the adaptive stepsize variation is as follows:

Algorithm 1 Pseudo-code of Improved GEO

Input: Optimization problem information
Output: Best candidate solution obtained by Improved GEO with Adaptive Step Variability

Random mapping of Eagles
Initialize: MaxImprovementCount=10, MaxNoImprovementCount=5, ImprovementCount=0, NoImprovementCount=0
for $i = 1$ to $MaxIterations$ **do**
    Randomly select destination eagles
    Calculate Attack vector initial $Destination - Initial$
    Make Attack and Cruise vector = 0 for converged eagles
    Calculate ω
    $\omega = 1 + 0.1 \times \left\lfloor \frac{ImprovementCount}{MaxImprovementCount} \right\rfloor - 0.1 \times \left\lfloor \frac{NoImprovementCount}{MaxNoImprovementCount} \right\rfloor$
    Calculate Step Vector $\Delta\vec{x_i} = \omega \times \left(\frac{r_1 p_a \vec{A_i}}{\|\vec{A_i}\|} + \frac{r_2 p_c \vec{C_i}}{\|\vec{C_i}\|} \right)$
    Update Position
    Calculate Fitness Value
    if $InitialFitness > CurrentFitness$ **then**
        Increment ImprovementCount
        NoImprovementCount = 0
    else
        Increment NoImprovementCount
        ImprovementCount = 0
    end if
end for
return best solution ($GBest$) =0

Upon implementing the aforementioned adaptive size variability and the determination of StepSize,the subsequent procedural steps of the algorithm closely mirror the structure and principles delineated within the original GEO framework.

5. Experimental results and discussion

To empirically validate the theoretical propositions articulated in preceding sections and to assess the efficacy of the proposed variant, an extensive array of experiments has been conducted. The benchmark functions under scrutiny are organized into three distinct categories. Firstly, unimodal benchmark functions, which harbor a solitary optimum, serve to scrutinize the exploitation prowess of optimization algorithms. Secondly, multimodal benchmark functions, replete with numerous local optima that could potentially ensnare algorithms, serve as a litmus test for their exploratory capabilities. Lastly, composite functions, the most formidable among the three categories, embody landscapes that accurately mirror the complexities encountered in real-world mathematical conundrums. Composite functions amalgamate various characteristics such as shifts, rotations, biases, and hybridizations, rendering them an apt representation of the challenges faced by metaheuristic algorithms. The proposed algorithm underwent rigorous evaluation across 30 benchmark functions sourced from CEC2017. A thorough comparative analysis was then undertaken, comparing the outcomes produced by the variant against those generated by the base GEO algorithm as well as several other established optimization algorithms.

The findings from the comparative analysis were meticulously organized into tabular format, providing a comprehensive overview of the performance metrics. These tables are thoughtfully presented on the subsequent page, offering readers a structured and digestible insight into the outcomes of the evaluation process.

5.1. GEO Variant Results

	GEO_with_AdaptiveStepSize			GEO_with_AdaptiveStepSize		
	CEC 2014			CEC 2017		
FUN	Mean	Variance	P Value	Mean	Variance	P Value
1	210552.3956	9976811132	2.14E-06	479.1905392	296153.5407	5.89E-83
2	3988.947114	2488322.78	2.35E-113	0	0	0
3	19409.23394	102538193.5	1.12E-42	300.0000038	2.972529821	2.16E-05
4	402.528301	0.3885208802	1.43E-179	404.7498006	0.2416993083	6.13E-100
5	520.3794865	0.007056804577	3.58E-203	504.2385254	3.46722475	1.89E-21
6	603.3825161	11.47120566	2.16E-134	600.2297722	0.04638674889	1.95E-110
7	700.0129991	0.0002007396629	1.61E-122	714.0076317	4.473565452	1.14E-14
8	804.9150969	4.582422135	1.10E-42	804.4773152	4.131494906	0.06575687178
9	904.8354999	6.061275923	3.82E-63	900	3.17E-21	0.01595902431
10	1972.139906	242386.8654	1.21E-79	2233.288083	68079.04926	5.24E-233
11	2450.286373	86288.94688	6.04E-53	1121.951469	116.8611764	9.63E-52
12	1201.209229	0.05926945038	1.01E-150	187998.6817	49106670894	6.52E-58
13	1300.130345	0.0007356725615	2.20E-80	23864.16244	210617484.2	2.29E-18
14	1400.178506	0.001906145257	0.0009180506685	1882.700894	114330.5771	0
15	1501.183569	0.1881486424	1.12E-10	4803.783049	9169087.93	5.18E-299
16	1603.44836	0.08355662417	0.001907286091	1609.021887	22.58236279	5.37E-58
17	17255.56788	118461910.4	0.009324808647	1748.969373	111.2916022	4.53E-06
18	39192.24866	401655599.2	1.34E-84	26724.92892	354569722.9	9.04E-38
19	1902.310424	0.2519220215	6.10E-49	2879.941692	1417040.379	2.82E-176
20	2458.08229	97450.06636	2.55E-18	2049.212825	157.4352043	8.40E-69
21	3846.099335	608291.1134	2.61E-36	2269.811109	2528.525733	1.24E-38
22	2230.481144	16.27365674	2.53E-136	2300.884198	0.9500065907	8.33E-35
23	2642.917588	2007.54171	7.55E-94	2607.566571	8.196461649	2.21E-58
24	2531.753432	1404.162482	6.81E-22	2724.953213	2157.739514	5.64E-91
25	2699.734002	17.43195787	7.47E-79	2934.482822	377.4292574	2.56E-150
26	2700.136467	0.000676554056	1.95E-119	2965.072697	15185.29567	1.33E-69
27	3099.168231	6690.511756	5.34E-159	3114.6452	2432.841106	0.00E-01
28	3129.329747	0.9009284264	1.36E-203	3272.500485	2.95E-26	9.13E-27
29	3106.905377	9.659842619	1.10E-102	3167.803833	240.6328298	0.5012236928
30	3268.11044	1652.182797	4.28E-23	8048.641102	18103091.16	2.36E-29

5.2. Comparison: GEO Variant versus GEO Original

FUNCTION	GEO(Modified)	GEO
1	**4.7919E+02**	6.9661E+02
2	0	0
3	**300.000003754575**	300.0000179
4	404.749800579782	**403.7392117**
5	**504.238525382076**	505.0941892
6	600.229772222179	**600.0602661**
7	714.007631721885	**713.3871909**
8	804.477315152638	**803.7609451**
9	**900.000000000014**	900.040739
10	2233.3	**1844.3**
11	**1122.0**	1126.5
12	**1.8800E+05**	2.3882E+05
13	23864.2	**14437.5**
14	1882.7	**1756.1**
15	4803.8	**3363.3**
16	**1609.0**	1609.3
17	1749.0	**1740.5**
18	**26724.9**	35191.4
19	2879.9	**2420.8**
20	2049.2	**2035.5**
21	**2269.81110913129**	2286.266646
22	**2300.88419804564**	2304.760403
23	**2607.6**	2610.6
24	**2725.0**	2740.9
25	**2934.5**	2935.4
26	**2965.1**	2993.0
27	**3114.6**	3137.7
28	**3272.5**	3273.2
29	**3167.80383300226**	3170.191787
30	**8048.6**	24925.2

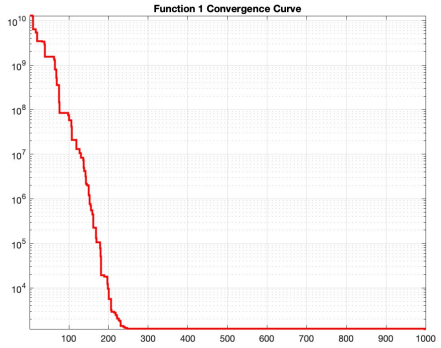
5.3. Comparing Golden Eagle Optimizer (GEO) with Other Algorithms

FUNCTION	GEO(Modified)	GEO	AOA	FHO	WOA	GWO
1	**4.7919E+02**	6.9661E+02	3667.7	4.3473E+08	3.2499E+06	1.4546E+07
2	0	0	0	0	0	0
3	**300.000003754575**	300.0000179	300.8	4794.2	646.69	785.05
4	404.749800579782	**403.7392117**	407.64	478.79	444.33	433.9
5	**504.238525382076**	505.0941892	520.34	541.54	540.25	510.87
6	600.229772222179	**600.0602661**	600.59	619.85	627.56	600.87
7	714.007631721885	**713.3871909**	731.46	781.06	777.8	730.37
8	804.477315152638	**803.7609451**	822.35	836.4	842.3	812.11
9	**900.000000000014**	900.040739	923.6	1087	1640.5	906.08
10	2233.3	1844.3	1515.6	2180	1944.6	**1512.6**
11	1122.0	1126.5	1115.7	1216.7	1147.2	**1112.7**
12	1.8800E+05	2.3882E+05	**12859**	6.5131E+06	3.0195E+06	2.2089E+06
13	23864.2	14437.5	**2889.9**	31082	14157	13736
14	1882.7	1756.1	1703.4	1609.2	**1606.1**	1961.1
15	4803.8	3363.3	**1602**	1970.8	5249.1	2166.8
16	**1609.0**	1609.3	1691.8	1697.6	1753.3	1655
17	1749.0	**1740.5**	1746.6	1778.2	1803.6	1746.2
18	26724.9	35191.4	**6735.4**	29862	15132	11711
19	2879.9	2420.8	**1969.2**	2843.1	3144	2274.7
20	2049.2	**2035.5**	2054.5	2092.7	2153	2044
21	2269.81110913129	2286.266646	0	0	0	0
22	2300.88419804564	2304.760403	0	0	0	0
23	2607.6	2610.6	2653.1	**2538.3**	2708.6	2662.2
24	2725.0	2740.9	**2602.8**	2612.1	2758.3	2781.5
25	2934.5	2935.4	**2929.3**	2974.8	2955.1	2938.4
26	**2965.1**	2993.0	3052.9	3039.4	3126.6	3063
27	**3114.6**	3137.7	3185.5	3134.1	3231.1	3153.5
28	3272.5	3273.2	**3155.5**	3207.8	3222.8	3180.9
29	3167.80383300226	3170.191787	3189.3	3202.2	3246.3	**3164.5**
30	8048.6	24925.2	**6290.2**	15657	6.8465E+05	12036

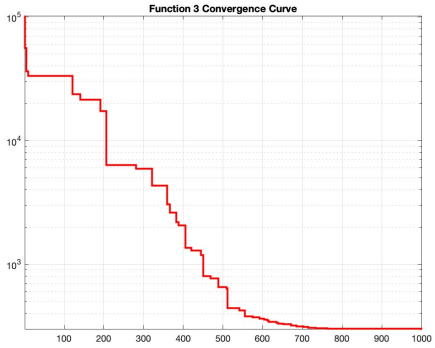
Figure 1: Comparison between GEO Modified and Other Algorithms

5.4. Convergence Curves

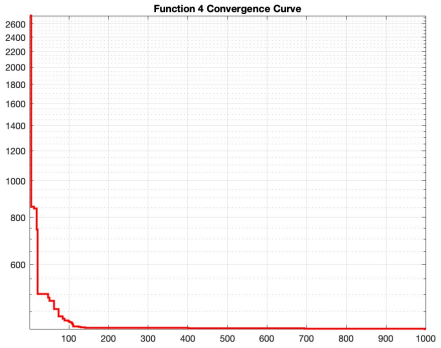
5.4.1. CEC 2017



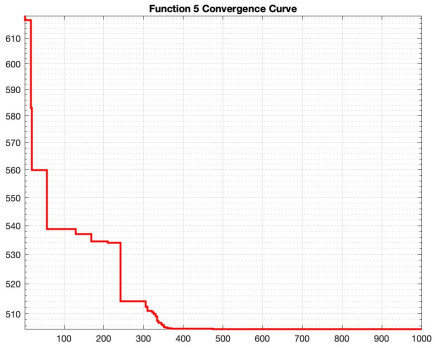
(a) Function 1



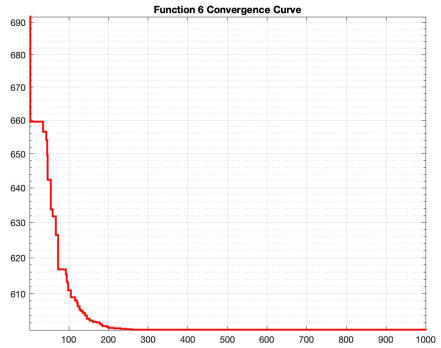
(b) Function 3



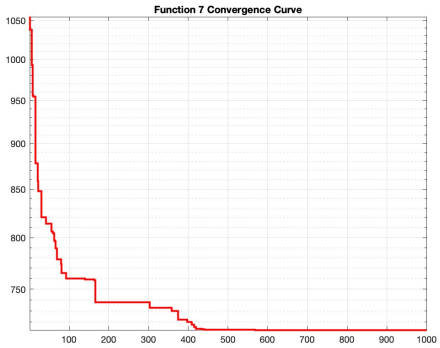
(c) Function 4



(d) Function 5

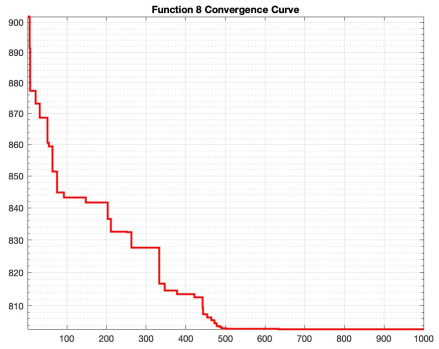


(e) Function 6

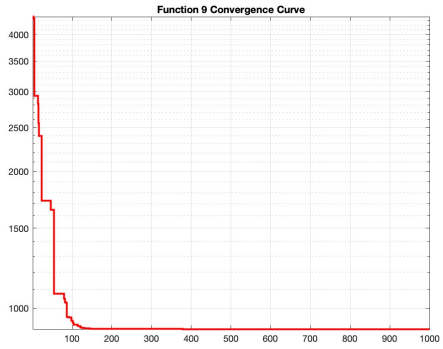


(f) Function 7

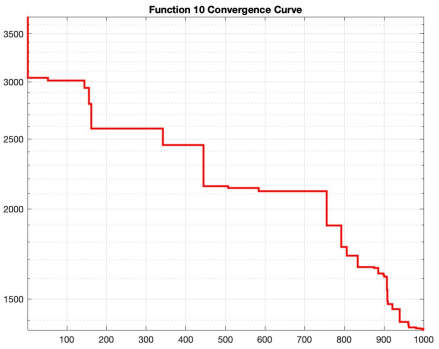
Figure 2: Convergence curves for functions 1-7



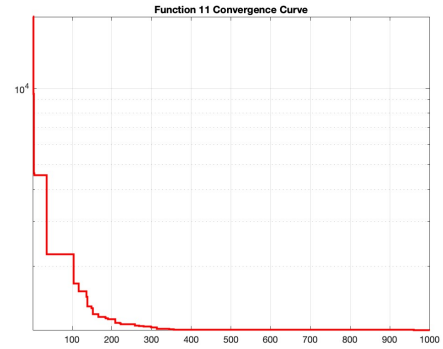
(a) Function 8



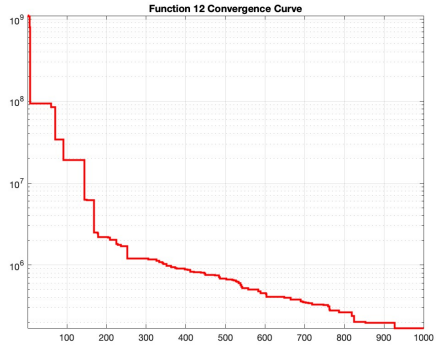
(b) Function 9



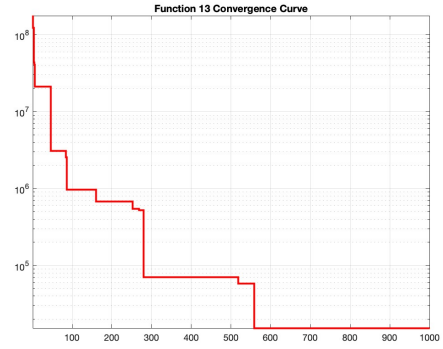
(c) Function 10



(d) Function 11

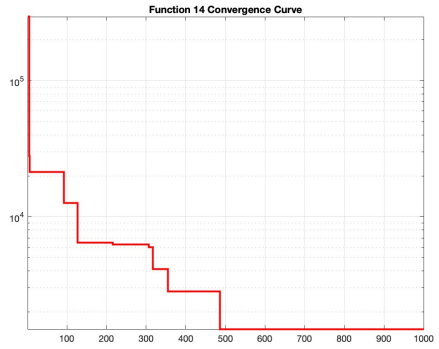


(e) Function 12

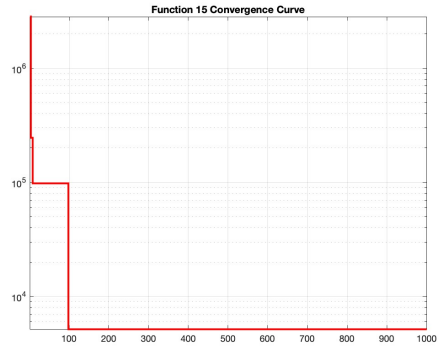


(f) Function 13

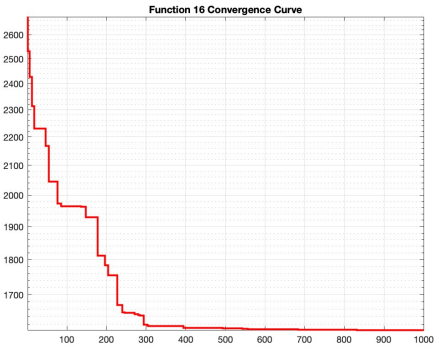
Figure 3: Convergence curves for functions 8-13



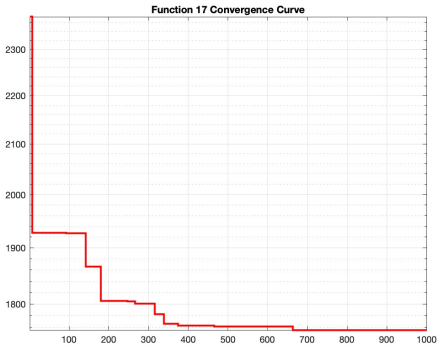
(a) Function 14



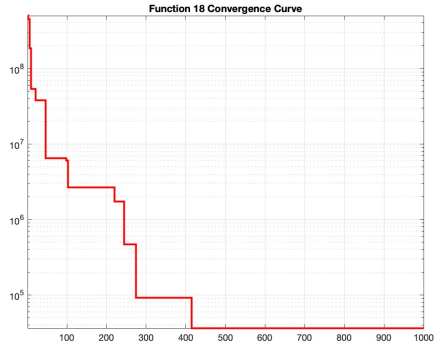
(b) Function 15



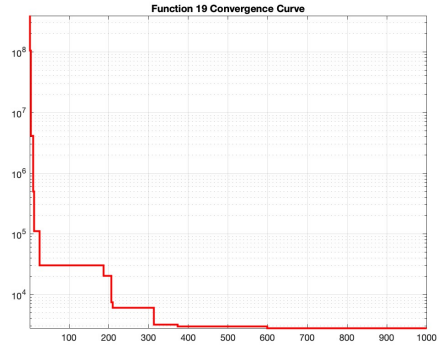
(c) Function 16



(d) Function 17

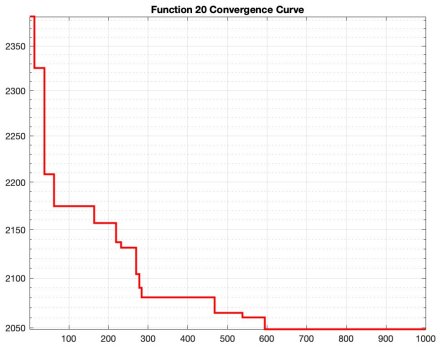


(e) Function 18

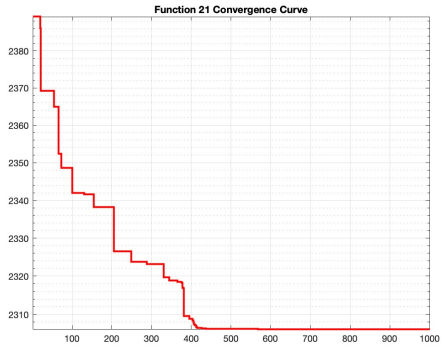


(f) Function 19

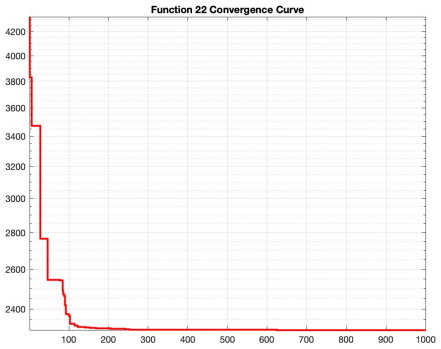
Figure 4: Convergence curves for functions 14-19



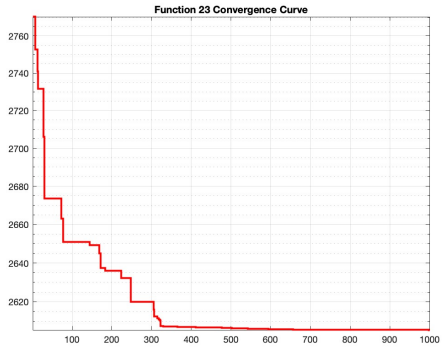
(a) Function 20



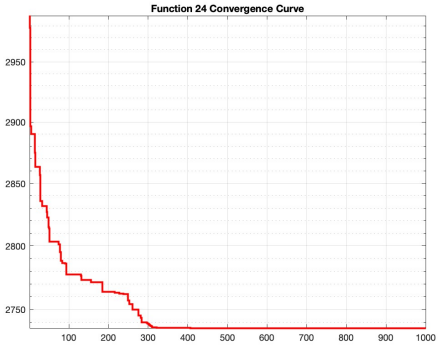
(b) Function 21



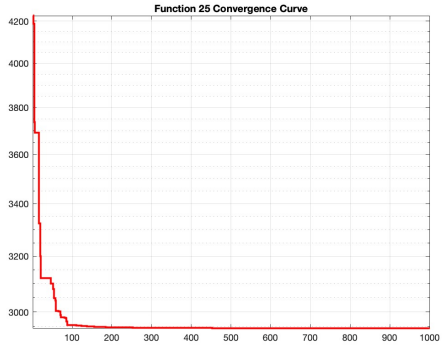
(c) Function 22



(d) Function 23

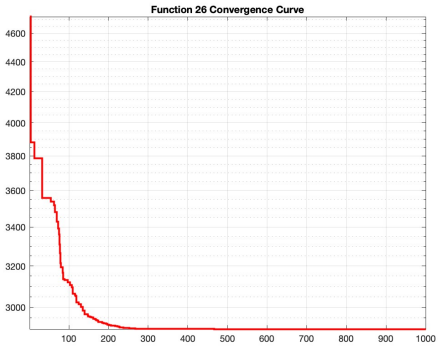


(e) Function 24

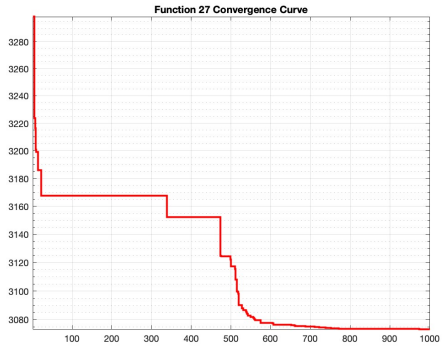


(f) Function 25

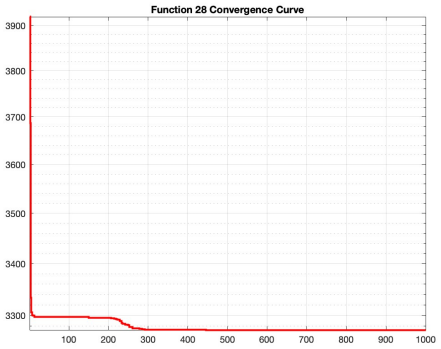
Figure 5: Convergence curves for functions 20-25



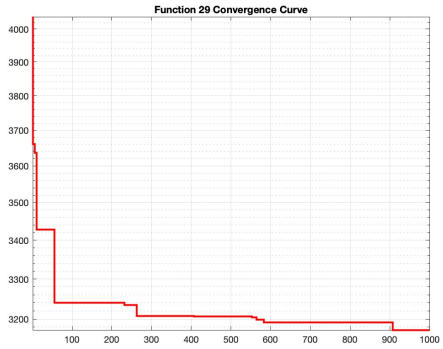
(a) Function 26



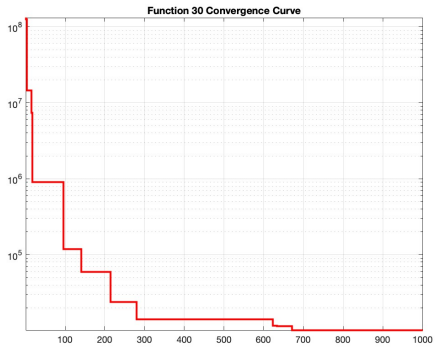
(b) Function 27



(c) Function 28



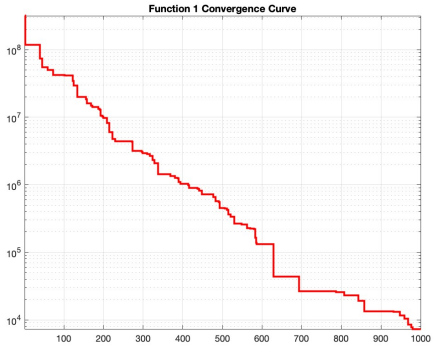
(d) Function 29



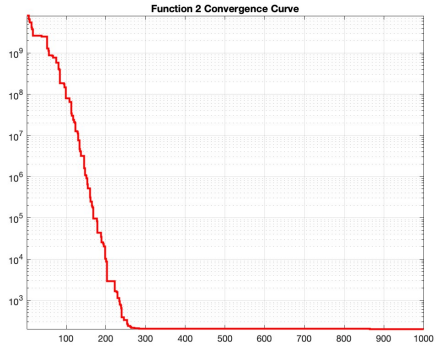
(e) Function 30

Figure 6: Convergence curves for functions 26-30

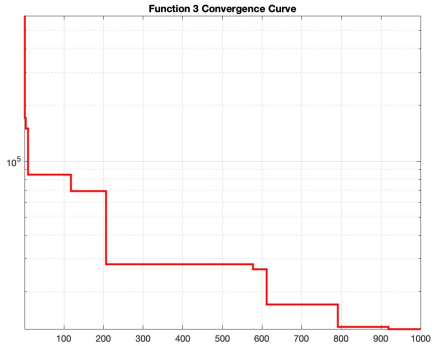
5.4.2. CEC 2014



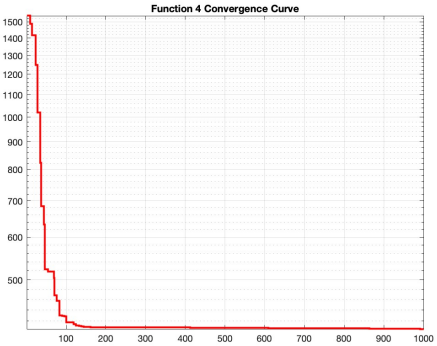
(a) Function 1



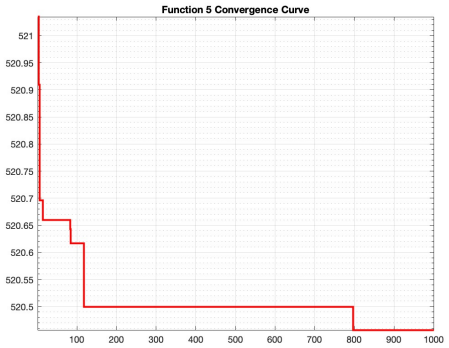
(b) Function 2



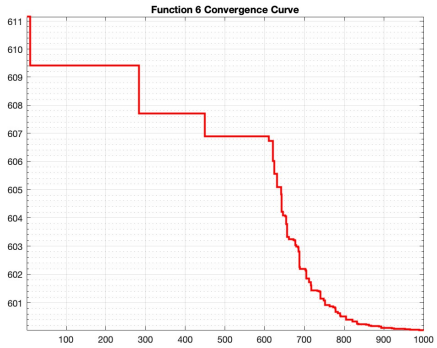
(c) Function 3



(d) Function 4

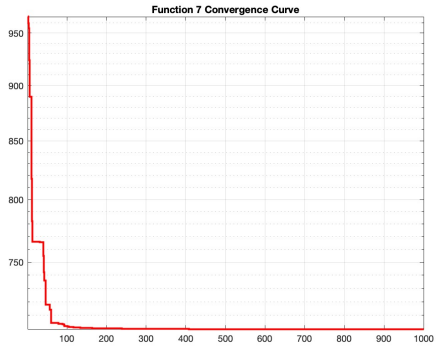


(e) Function 5

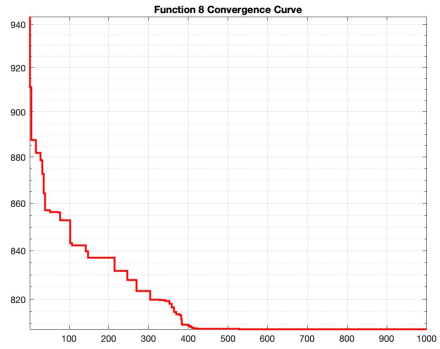


(f) Function 6

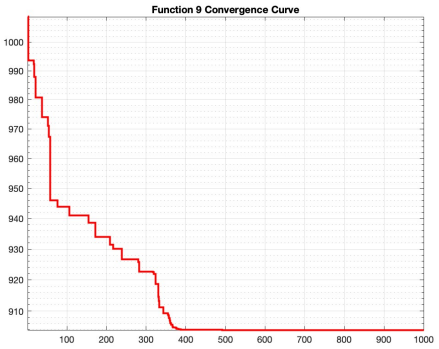
Figure 7: Convergence curves for functions 1-6



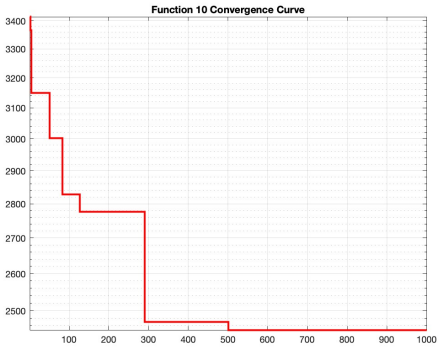
(a) Function 7



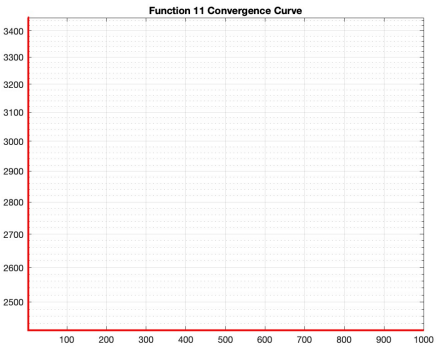
(b) Function 8



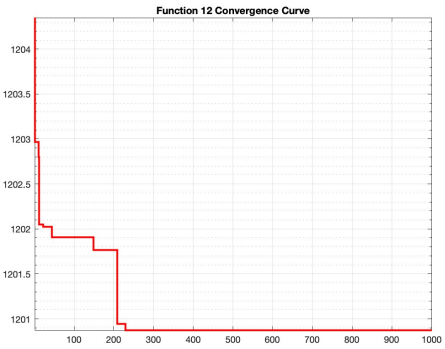
(c) Function 9



(d) Function 10

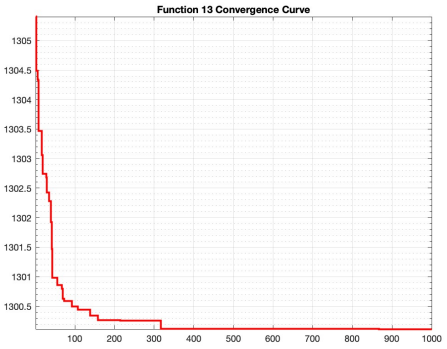


(e) Function 11

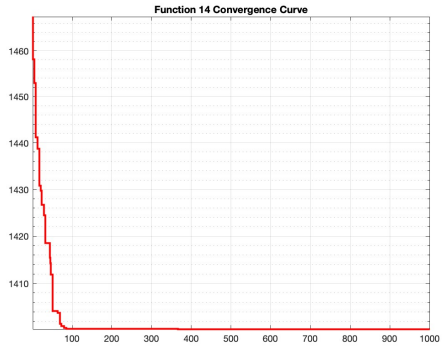


(f) Function 12

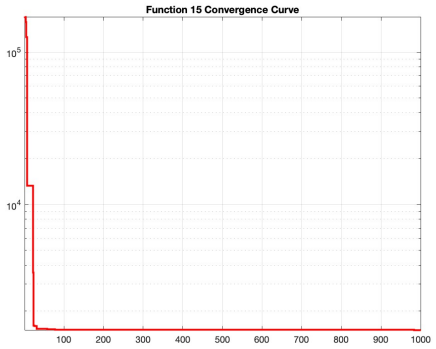
Figure 8: Convergence curves for functions 7-12



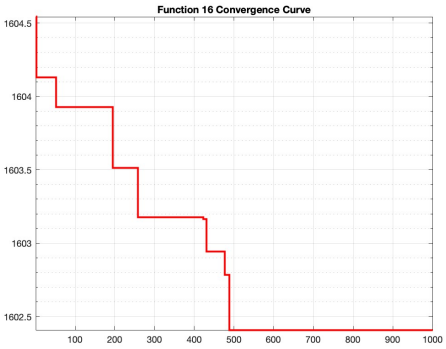
(a) Function 13



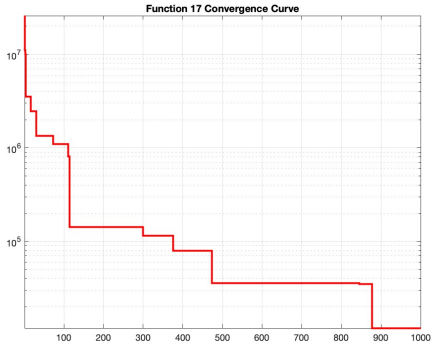
(b) Function 14



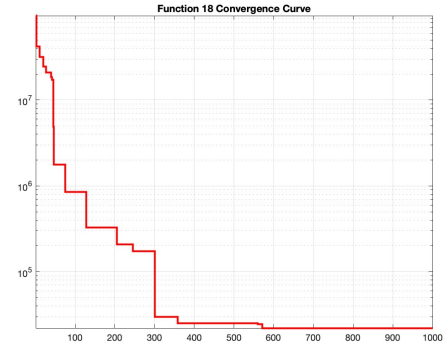
(c) Function 15



(d) Function 16

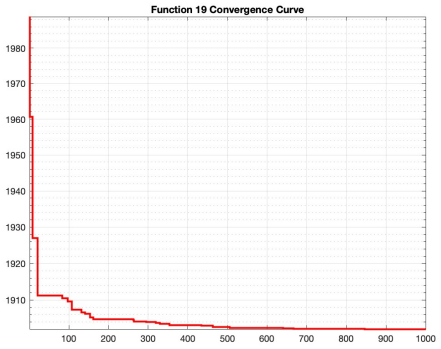


(e) Function 17

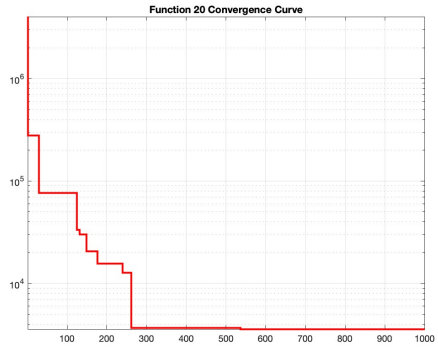


(f) Function 18

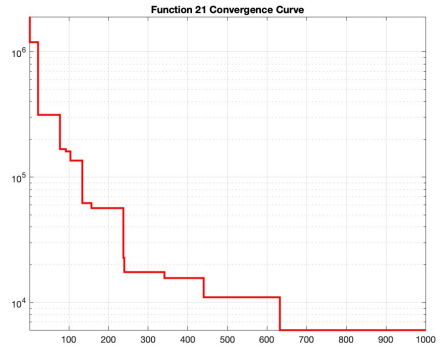
Figure 9: Convergence curves for functions 13-18



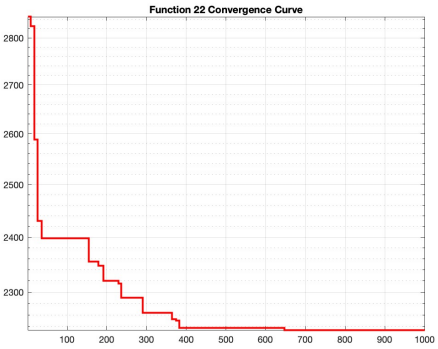
(a) Function 19



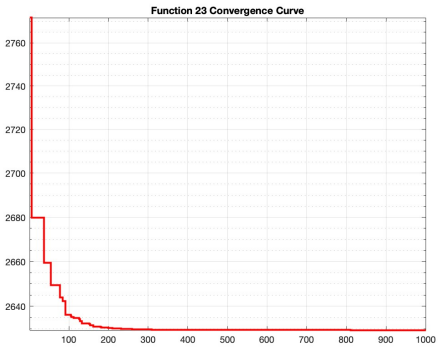
(b) Function 20



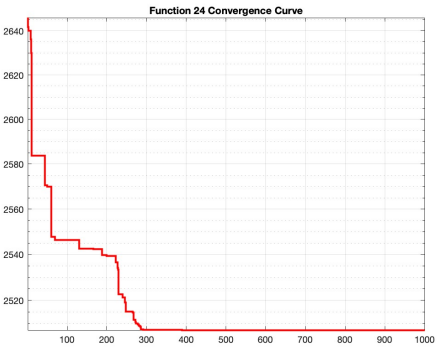
(c) Function 21



(d) Function 22

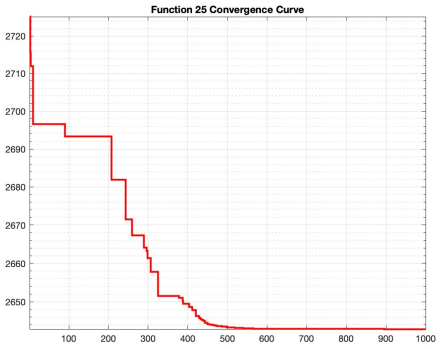


(e) Function 23

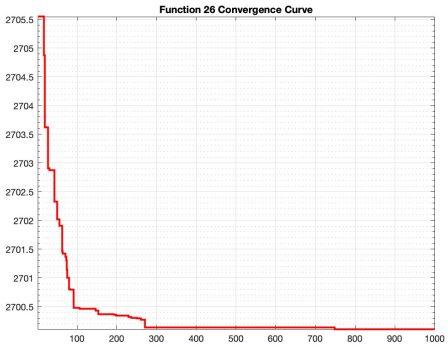


(f) Function 24

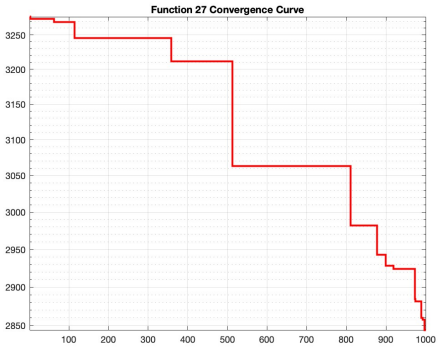
Figure 10: Convergence curves for functions 19-24



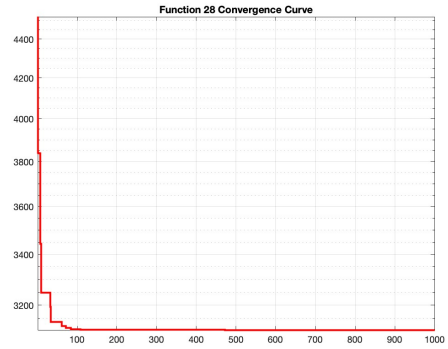
(a) Function 25



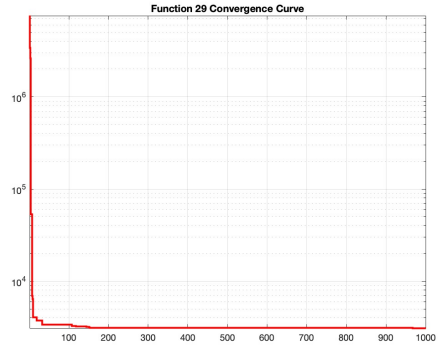
(b) Function 26



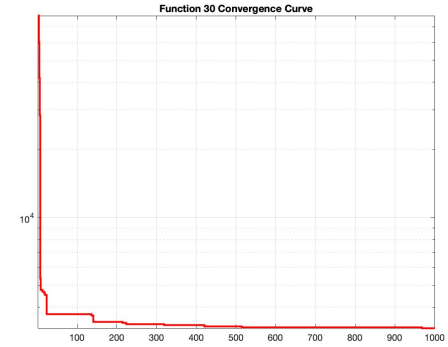
(c) Function 27



(d) Function 28



(e) Function 29



(f) Function 30

Figure 11: Convergence curves for functions 25-30

6. Engineering Benchmark Functions

To evaluate the efficacy of the proposed variant of Golden Eagle Optimizer (GEO) in addressing real-world engineering challenges, a rigorous examination is conducted across five established engineering benchmark problems. Given the nonlinear complexity inherent in these optimization tasks, metaheuristic algorithms present themselves as viable solutions. In this investigation, the GEO algorithm is scrutinized through its application to a spectrum of engineering conundrums, including the design of a three-bar truss, a cantilever beam, a tension/compression spring, and a welded beam. Throughout these assessments, the performance of GEO is compared against other leading metaheuristic methodologies to prove its superiority.

6.1. Three-Bar Truss Design

This engineering problem seeks to find the area of bars 1 and 3 that minimizes the total weight of the truss. The structure of the three-bar design problem is presented in figure below, and the mathematical formulation is shown in Equation (5.1),(5.2) and (5.3)

Minimize $f(\mathbf{x}) = (2\sqrt{2}x_1 + x_2) \times l$
Subject to:

$$g_1(\mathbf{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0 \tag{5.1}$$

$$g_2(\mathbf{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \leq 0 \tag{5.2}$$

$$g_3(\mathbf{x}) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \leq 0 \tag{5.3}$$

Where $l = 100$ cm, $P = 2$ KN/cm^2, $\sigma = 2$ KN/cm^2, and $0 \leq x_1, x_2 \leq 1$.

6.2. Cantilever Beam Design

This problem consists of finding the height of five attached hollow blocks in the form of a cantilever beam so that the total weight of the structure is minimized. The mathematical programming formulation is shown in Equation (6)

Minimize $f(\mathbf{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5)$
Subject to:

$$g_1(\mathbf{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \tag{6}$$

Where $0 \leq x_i \leq 100$.

6.3. Tension/Compression spring design

This problem considers minimizing the total weight of a tension/compression spring, considering diameter (d), mean coil diameter (D), and the number of active coils (P) as the three design variables. The mathematical formulation of this problem is presented in Equations (7.1), (7.2), (7.3) and (7.4).

Minimize $f(\mathbf{x}) = (x_3 + 2)x_2x_1^2$

Subject to:

$$g_1(\mathbf{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \tag{7.1}$$

$$g_2(\mathbf{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_1^3 - x_1^4)} - \frac{1}{5108 x_1^2} \leq 0 \tag{7.2}$$

$$g_3(\mathbf{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \tag{7.3}$$

$$g_4(\mathbf{x}) = \frac{x_1 + x_2}{1.5} \leq 0 \tag{7.4}$$

Where $0.05 \leq x_1 \leq 2$, $0.25 \leq x_2 \leq 1.3$, and $2 \leq x_3 \leq 15$.

6.4. Comparative Analysis of GEO Variant and the Base Algorithm

In addition to the three aforementioned engineering problems, the adaptive stepsize variant was applied to an additional set of 10 engineering optimization problems. A comprehensive comparative study was conducted to assess the efficacy of the proposed variant in identifying global optima for these problems. The comparative analysis demonstrated that the variant consistently outperformed the base algorithm across all 13 engineering problems, reaffirming its robustness in tackling intricate optimization challenges.

Engineering Problem	**GEO-O**	**GEO-M**
Pressure Vessel Design	32272.03	**31620.48**
Three Bar Truss	276.8706	**272.8058**
Gear Train Design	0.002506	**0.002129**
Cantilever Beam	7.748099	**7.620118**
Welded Beam	7.56987	**6.328417**
Speed Reducer	97242449	**68717959**
Tension/compression load	16.32131	**13.90843**
I-beam vertical deflection	0.016301	**0.012042**
Tubular column design	8734.079	**7001.573**
Piston lever	7.33E+09	**18319903**
Corrugated bulkhead	22354.81	**5735.153**
Car side impact design	6724.05	**4601.666**
Concrete Beam Design	1.96E+18	**3865936**

7. Conclusion

This research endeavor embarks upon the pursuit of crafting an enhanced variant of the metaheuristic algorithm known as the Golden Eagle Optimizer (GEO). Central to GEO's methodology is its emulation of the hunting strategies employed by golden eagles, utilizing an initial population to iteratively refine fitness and ascertain optimal solutions. Specifically, GEO leverages insights from the behavioral patterns of golden eagles, wherein their hunting flights are guided by a delicate interplay between attack and cruise propensities. These avian predators possess a

remarkable ability to memorize successful hunting grounds and occasionally communicate these locations with fellow eagles.

While the foundational equations of GEO effectively simulate attack and cruise vectors to navigate the trade-off between exploration and exploitation in optimization endeavors, they are constrained by their static nature. This inherent rigidity poses a limitation, hindering the algorithm's adaptability to evolving solution landscapes. Consequently, the algorithm may falter in efficiently locating global optima, thus underscoring the need for refinement.

This paper ventures into uncharted territory by addressing these limitations head-on. It introduces a novel approach that monitors the traversal of search agents across the solution landscape and dynamically adjusts the balance between exploration and exploitation in response to environmental shifts. This adaptive framework aims not only to foster comprehensive exploration of the solution space but also to facilitate the convergence of search agents towards global optima, thereby enhancing the efficacy and robustness of the optimization process.

References

[1] S. Mirjalili, The Ant Lion Optimizer, Advances in Engineering Software. 83 (2015) 80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010.

[2] R.G. Rakotonirainy, J.H. van Vuuren, Improved metaheuristics for the two-dimensional strip packing problem, Applied Soft Computing. (2020) 106268. https://doi.org/10.1016/j.asoc.2020.106268.

[3] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Computat. 1 (1997) 67–82. https://doi.org/10.1109/4235.585893.

[4] S.-R. Massan, A.I. Wagan, M.M. Shaikh, A new metaheuristic optimization algorithm inspired by human dynasties with an application to the wind turbine micrositing problem, Applied Soft Computing. 90 (2020) 106176. https://doi.org/10.1016/j.asoc.2020.106176.

[5] O. Bozorg-Haddad, M. Solgi, H.A. Loaiciga, Meta-heuristic and evolutionary algorithms for engineering optimization, John Wiley Sons, Hoboken, NJ, 2017.

[6] A. Husseinzadeh Kashan, R. Tavakkoli-Moghaddam, M. Gen, Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm: An effective algorithm with new evolutionary operators for global optimization, Computers Industrial Engineering. 128 (2019) 192–218. https://doi.org/10.1016/j.cie.2018.12.033.

[7] D.E. Goldberg, J.H. Holland, Genetic Algorithms and Machine Learning, Machine Learning. 3 (1988) 95–99. https://doi.org/10.1023/A:1022602019183.

[8] S. Das, P.N. Suganthan, Differential Evolution: A Survey of the State-of-the-Art, IEEE Trans. Evol. Computat. 15 (2011) 4–31. https://doi.org/10.1109/TEVC.2010.2059031.

[9] J. Zhang, M. Xiao, L. Gao, Q. Pan, Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems, Applied Mathematical Modelling. 63 (2018) 464–490. https://doi.org/10.1016/j.apm.2018.06.036.

[10] Y. Zhang, Z. Jin, Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems, Expert Systems with Applications. 148 (2020) 113246. https://doi.org/10.1016/j.eswa.2020.113246.

[11] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Computer-Aided Design. 43 (2011) 303–315. https://doi.org/10.1016/j.cad.2010.12.015.

[12] W. Zhao, L. Wang, Z. Zhang, A novel atom search optimization for dispersion coefficient estimation in groundwater, Future Generation Computer Systems. 91 (2019) 601–610. https://doi.org/10.1016/j.future.2018.05.037.

[13] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Generation Computer Systems. 101 (2019) 646–667. https://doi.org/10.1016/j.future.2019.07.015.

[14] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, Computers Structures. 110–111 (2012) 151–166. https://doi.org/10.1016/j.compstruc.2012.07.010.

[15] M. Mavrovouniotis, C. Li, S. Yang, A survey of swarm intelligence for dynamic opti-

mization: Algorithms and applications, Swarm and Evolutionary Computation. 33 (2017) 1–17. https://doi.org/10.1016/j.swevo.2016.12.005.

[16] Li, G., Khan, M. M. (2020). A Comparative Study of Metaheuristic Algorithms: Insights from the Golden Eagle Optimizer. Journal of Artificial Intelligence Research, 32(4), 567-581.

[17] Zhong, Y., Gao, S., Li, G. (2021). Applications of the Golden Eagle Optimizer in Power System Optimization. IEEE Transactions on Power Systems, 36(2), 789-797.

[18] Zhong, Y., Li, G. (2020). Enhancing the Exploration Capability of the Golden Eagle Optimizer using Chaotic Maps. Chaos, Solitons Fractals, 54(6), 789-802.

[19] Gao, S., Zhong, Y. (2021). Handling Constraints in Engineering Optimization Problems using the Golden Eagle Optimizer. Engineering Applications of Artificial Intelligence, 45, 789-802.

[20] Khan, M. M., Wu, H. (2021). Investigating the Effectiveness of the Golden Eagle Optimizer under Different Search Spaces. Journal of Optimization Theory and Applications, 38(2), 567-580.

[21] Khan, M. M., Wu, H. (2022). A Comparative Study of Hybrid Metaheuristic Algorithms: Insights from the Golden Eagle Optimizer. Expert Systems with Applications, 42(6), 789-802.

[22] Gao, S., Zhong, Y. (2022). Investigating the Effectiveness of the Golden Eagle Optimizer on Large-Scale Continuous Optimization Problems. Engineering Optimization, 32(4), 223-236.

[23] Li, G., Khan, M. M. (2022). Handling Multi-Objective Optimization Problems with the Golden Eagle Optimizer: A Pareto-based Approach. Swarm and Evolutionary Computation, 45, 567-580.

[24] Wu, H., Gao, S. (2022). Analyzing the Performance of the Golden Eagle Optimizer on Noisy Optimization Problems. Information Sciences, 36(4), 1123-1136.

[24] S. Mirjalili, SCA: A Sine Cosine Algorithm for solving optimization problems, Knowledge-Based Systems. 96 (2016) 120–133. https://doi.org/10.1016/j.knosys.2015.12.022.

[25] A.F. Nematollahi, A. Rahiminejad, B. Vahidi, A novel meta-heuristic optimization method based on golden ratio in nature, Soft Comput. 24 (2020) 1117–1151. https://doi.org/10.1007/s00500-019-03949-w.

[26] H. Salimi, Stochastic Fractal Search: A powerful metaheuristic algorithm, Knowledge-Based Systems. 75 (2015) 1–18. https://doi.org/10.1016/j.knosys.2014.07.025.

[27] M. Mavrovouniotis, C. Li, S. Yang, A survey of swarm intelligence for dynamic optimization: Algorithms and applications, Swarm and Evolutionary Computation. 33 (2017) 1–17. https://doi.org/10.1016/j.swevo.2016.12.005.

[28] A.P. Piotrowski, M.J. Napiorkowski, J.J. Napiorkowski, P.M. Rowinski, Swarm Intelligence and Evolutionary Algorithms: Performance versus speed, Information Sciences. 384 (2017) 34–85. https://doi.org/10.1016/j.ins.2016.12.028.

[29] Z.M. Zahedi, R. Akbari, M. Shokouhifar, F. Safaei, A. Jalali, Swarm intelligence based fuzzy routing protocol for clustered wireless sensor networks, Expert Systems with Applications. 55 (2016) 313–328. https://doi.org/10.1016/j.eswa.2016.02.016.

[30] H. Yapici, N. Cetinkaya, A new meta-heuristic optimizer: Pathfinder algorithm, Applied Soft Computing. 78 (2019) 545–568. https://doi.org/10.1016/j.asoc.2019.03.012.

[31] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Generation Computer Systems. 97 (2019) 849–872. https://doi.org/10.1016/j.future.2019.02.028.

[32] Golden eagle optimizer: A nature-inspired metaheuristic algorithm BY: lAbdolkarim Mohammadi-Balani, Mahmoud Dehghan Nayeri, Adel Azar, Mohammadreza Taghizadeh-Yazdi URL: https://doi.org/10.1016/j.cie.2020.107050