Evaluating the Effectiveness of Swarm-Based Computational Intelligence Algorithms Using CEC 2022 Benchmark Functions

Abstract—**Swarm-based computational intelligence techniques have shown great promise in solving difficult optimization problems with many variables. This research assesses the effectiveness of four notable swarm-based algorithms: Frilled Lizard Optimizer (FLO), Grey Wolf Optimizer (GWO), Whale Optimization Algorithm (WOA), and Harris Hawks Optimization Algorithm (HHOA) with the CEC 2022 benchmark functions. It tests these algorithms on a set of standard benchmark functions on MATLAB software, unimodal, multimodal, and composite functions that are meant to check how well convergence works, how accurate the solution is, and how strong it is and to see how well they enable exploration and exploitation. The simulation findings show that GWO can search the whole space and swiftly converge on benchmark functions that are all functions. Each algorithm run over 30 times, and used statistical indicators like mean, standard deviation, Friedman ranking, convergence, and boxplot analysis to see how well they work. The results suggest that the algorithms have different strengths: FLO has a strong ability to search the whole world, while HHO and WOA are better at converging on complicated multimodal landscapes. This thorough review gives a systematic way to compare swarm-based metaheuristics, which helps to choose the best optimization methods for engineering and scientific purposes.**

Keywords: Swarm Intelligence, Frilled Lizard Optimizer, Grey Wolf Optimizer, Whale Optimization Algorithm, Harris Hawks Optimization, CEC 2022, Benchmarking, Metaheuristics.

I. Introduction

In many fields of science and engineering, optimization is a very significant method. The purpose is to choose the best response from a set of choices [1]. In real-world applications, nonlinear, non-differentiable, or multimodal challenges are prevalent. However, typical methods for optimizing arithmetic problems do not work well for these types of problems [2]. B. Tabbara [3] assessed the efficacy of the optimization methods utilizing merely two benchmark functions, hence constraining the scope of the results. Additionally, the study does not investigate dimensional scalability or evaluate the methods in high-dimensional testing environments. V. Dharmapuri [4] identified a method for classifying and analyzing benchmark functions; nevertheless, the study does not include experimental validation for the suggested test functions, hindering the assessment of their practical efficacy. Y. Liu [5] studied algorithms just for their best objective value, not their mean or standard deviation. The work lacks converging analysis of graphs, which is problematic. G. sun's work [6] does not discuss tuning parameters or analyzing sensitivity for the evaluated methods. This renders the comparison unfair and unreliable.

To address these difficulties, this study discusses four swarm intelligence-based optimizers. The Frilled Lizard Optimizer (FLO) [7], Whale Optimization Algorithm (WOA) [8], Gray Wolf Optimizer (GWO) [9], and Harris Hawks Optimizer (HHO) [10]. FLA replicates frilled lizards' territorial defense and fast movement when threatened. HHO mimics Harris hawks' cooperative hunting and surprise assaults on escaping victims. GWO copies how grey wolves hunt and lead, and WOA copies the way the whales feed by generating bubbles.

The purpose of this research is to use basic numerical benchmark tools in MATLAB to construct and compare four swarm intelligence algorithms that are based on nature. This study employs statistical performance measurements such as mean, standard deviation, boxplot, and convergence behavior to evaluate the positive and negative characteristics of each method. It determines the effectiveness of each method in maintaining a harmony between exploring and exploiting throughout optimization. High-dimensional test scenarios assess how well a method can handle problems and get better. Results from experiments show that GWO was better and more stable than FLO, WOA, and HHO in several ways. When working with real-valued variables, GWO should be the first choice for solving numerical problems. The GWO Algorithm outperformed competing algorithms in several benchmark functions, particularly regarding the stability and precision of the results. FLO, WOA, and GWO also performed well, however each has its own strengths for various types of tasks or levels of difficulty. This study offers the following key contributions:

A thorough analysis of optimizers using numerical benchmarks.

- Use of mean and SD across independent runs.
- Descriptive convergence behavior and boxplot data for each algorithm.

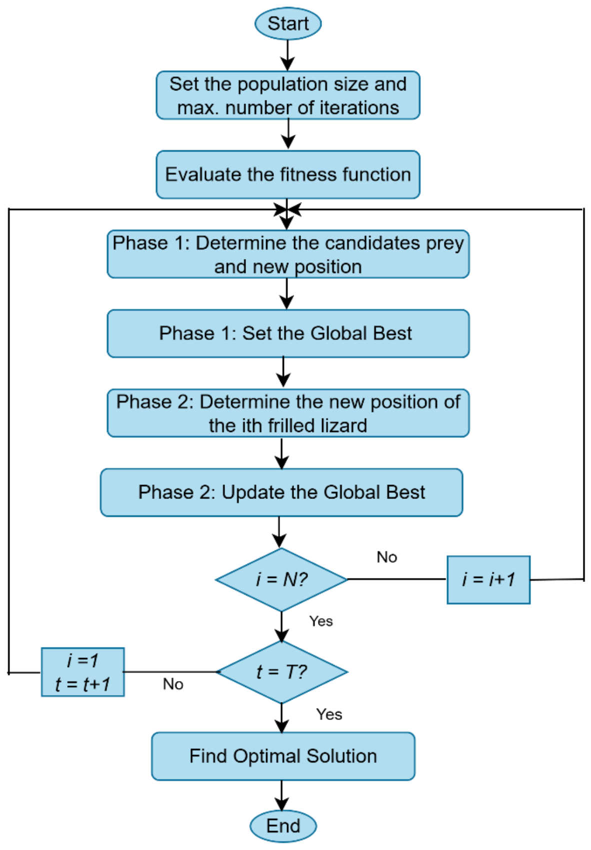
The rest of the paper is built up like this: Part II discusses the many procedures employed in this investigation. Part III talks about how the probe was set up and what the rules were. Part IV reveals what happened during the experiment. Part V then summarizes the study's findings.

II. Types of Algorithms

Natural-based metaheuristic algorithms are used to address tricky and nonlinear optimization issues. In this paper, we analyzed the four most common metaheuristic algorithms: FLO, GWO, WOA, and HHO.

A. Frilled Lizard Optimizer

The frilled lizard's place in the computational space is changed twice throughout each iteration of the FLO Algorithm shown in fig. 1. The first stage mimics the frilled lizard's movement toward prey when hunting, which makes the search area bigger and looking at new possibilities. The first



Start
Set the population size and max. number of iterations
Evaluate the fitness function
Phase 1: Determine the candidates prey and new position
Phase 1: Set the Global Best
Phase 2: Determine the new position of the ith frilled lizard
Phase 2: Update the Global Best
i = N?
No
i = i+1
Yes
i =1
t = t+1
No
t = T?
Yes
Find Optimal Solution
End

Fig.1. Flowchart of the FLO

step of FLO uses the frilled lizard's predatory method to move people around in the fix space of this problem. In the layout of FLO, each frilled lizard's position is based on the positions of the other members of the population who have a stronger aim function. The FLO design suggests that the frilled lizard will attack among these targets at random. A fresh position for each member of the population based on the way the frilled lizard moves toward its meal. If the desired function is better, this new location will take the place of the user's old one. After eating, the frilled lizard ascends a tree nearby. The algorithm's ability to search locally is improved by simulating the frilled lizard's hike to the top of the tree, which causes small changes in the population's position in the solution space. FLO's second phase adjusts the placements of the population based on how the frilled lizard climbs the tree after eating. The first spots of the frilled lizards in the solution space are set up by an arbitrary initialization utilizing in eq. (1).

$$z_{p,q}=lb_q + r.\left(ub_q - lb_q\right) \tag{1}$$

B. Gray Wolf Optimization Algorithm

This section goes into further depth on how the GWO algorithm uses calculation to explain how grey wolves live and hunt by following, surrounding, and attacking their prey. During the enclosure of the prey stage, Grey wolves' behavior of circling their prey while hunting is explained by equations (2) and (3).

$$\overrightarrow{M} = | \overrightarrow{N} . \overrightarrow{P^*}(\text{t}) - \overrightarrow{P}(t) | \tag{2}$$

$$\overrightarrow{P}(\text{t+1}) = \overrightarrow{P^*}(\text{t}) - \overrightarrow{G} . \overrightarrow{M} \tag{3}$$

To mimic this natural behavior, the system keeps track of each wolf's position in relation to the prey. In GWO, the placements of alpha, beta, and delta wolves are utilized to guess what the greatest solution (the "prey") is. The other wolves proceed ahead of this estimated prey. This method lets the group focus on sections of the solution space that look promising and make sure that everyone agrees. During the "attacking the prey" phase, wolves get closer to their prey and get ready for the final strike as the hunt goes on. In the optimization process, this phase makes the search for the optimum solutions more intense. GWO does this by slowly diminishing random exploration and raising the impact of the top wolf. So, search agents get closer to the best solutions on a global or local level. This step makes the algorithm work better and enhances the search results. The Grey Wolf Optimizer (GWO) guides other wolves (ω) in their quest for prey. When |A| > 1, wolves explore widely to find prey. Wolves exploit prey by approaching them when |A| < 1.

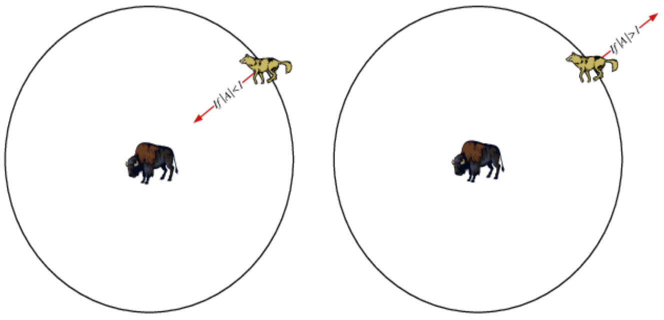


Fig.2. Exploring and Exploiting behavior of GWO

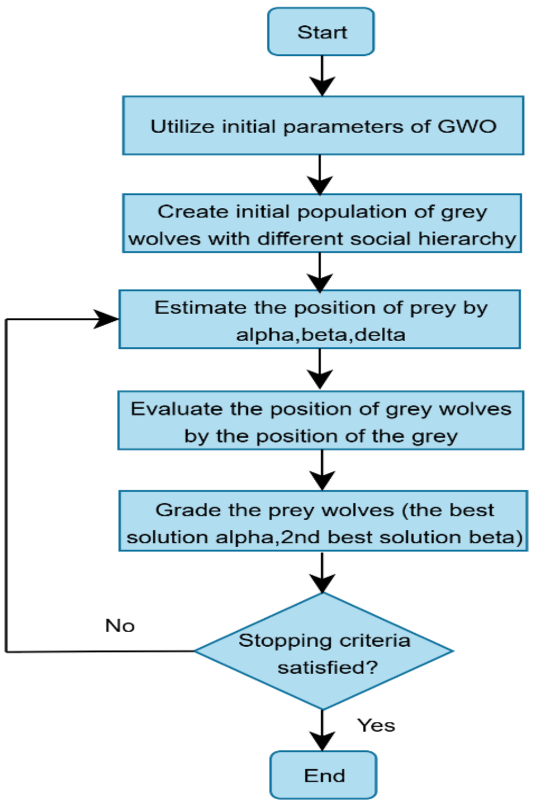

Start
Utilize initial parameters of GWO
Create initial population of grey wolves with different social hierarchy
Estimate the position of prey by alpha,beta,delta
Evaluate the position of grey wolves by the position of the grey
Grade the prey wolves (the best solution alpha,2nd best solution beta)
Stopping criteria satisfied?
No
Yes
End

Fig. 3. Flowchart of the GWO

C. *Whale Optimization Algorithm*

WOA involves exploration and exploitation. Exploration seeks better answers worldwide, while exploitation focuses on one good solution [8]. Whales eat by "bubble net attacking". They push bubbles outside the water. They whirl around the prey, shrinking as they approach. The algorithm assumes that a whale or prey nearby is the best option, although whales hunt throughout the ocean. Whales move to the best spot. Whales shrink or transform into helical transpiration when hunted. The distance between the whale and its prey is S', the spiral constant is an, and m could be anywhere from -1 to 1. Use (4)'s probability p to switch between the two.

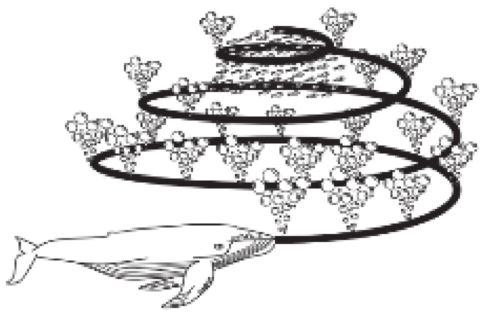


Fig. 4. Hunting strategy of the WOA

$$\vec{L}(t+1) = \begin{cases} \vec{L^{*}}(t) - \vec{M}.\ \vec{D} & \text{if } p < 0.5 \\ \vec{S'}.\ e^{am}.\ cos(2\pi m) + \vec{Y^{*}}(t), & \text{if } p \geq 0.5 \end{cases} \qquad (4)$$

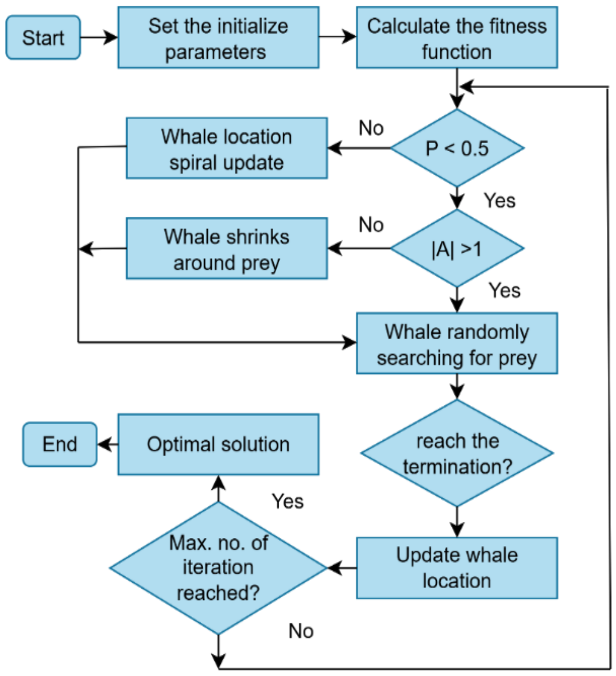

Start
Set the initialize parameters
Calculate the fitness function
P < 0.5
No
Yes
Whale location spiral update
|A| >1
No
Yes
Whale shrinks around prey
Whale randomly searching for prey
reach the termination?
Update whale location
Max. no. of iteration reached?
Yes
No
Optimal solution
End

Fig. 5. Flowchart of the WOA

D. *Harris Hawks Optimizer*

In this section, we will discuss the HHO optimizer. If we think about how Harris' hawks work, we can see that they can find and follow their prey with their strong eyes. However, sometimes the prey cannot see them. Therefore, the hawks wait, watch, and keep an eye on the desert site for prey, which could take a few hours. In HHO, the Harris' hawks are the possible solutions, and the best candidate solution at each phase is seen as the envisioned prey or close to the best one. In HHO, the hawks sit on different places and wait for prey to come by using two different methods. The prey's energy drops a lot when it tries to escape. To represent this fact, the power of a prey is depicted as:

$$E = 2E_0\left(1 - \frac{t}{T}\right) \qquad (5)$$

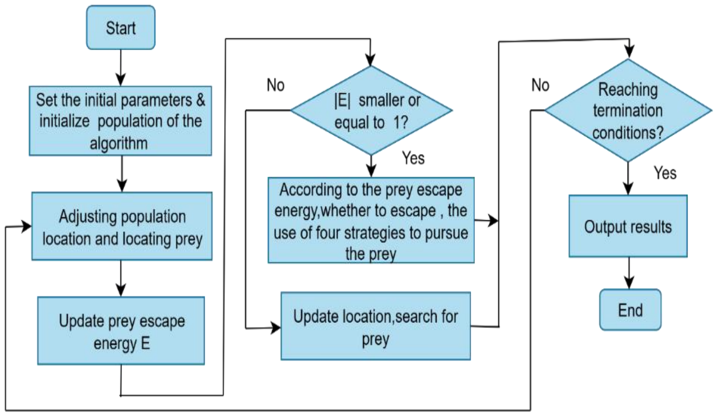

Start
Set the initial parameters & initialize population of the algorithm
Adjusting population location and locating prey
Update prey escape energy E
No
|E| smaller or equal to 1?
Yes
According to the prey escape energy,whether to escape , the use of four strategies to pursue the prey
Update location,search for prey
No
Reaching termination conditions?
Yes
Output results
End

Fig. 6. Flowchart of the HHO

III. EXPERIMENTAL SETUP AND SETTINGS

A. *Experimental setup and data sampling*

The CEC 2022 testing functions are now a well-known set of tests that consist of 12 test functions that are best used to check the suggested method. Four different kinds of functions are shown in table I. The search range goes from -100 to 100, the dimension D is 20, and the most rounds you can have is 500. Table II shows the results of the CEC 2022 study for FLO, GWO, WOA, and HHO. The findings indicate that the GWO has superior best, worst, mean, and standard deviation values for all functions, with the exception of CEC-06. Figure 7 shows the boxplot for CEC2022 also show how the CEC 2022 approaches come together. The graphic shows that the proposed GWO method swiftly converges to near-optimal solutions every time. We run each experiment 30 times on its own, and the mean and standard deviation (SD) of the results are based on those runs. After then, these figures are retained for further use in performance reviews.

B. *Friedman Rank Test*

We used the Friedman rank test on twelve benchmark functions to show that the suggested procedures did better than each other in a statistically significant way. Table III shows the overall ranks and the average scores for each method on the list. The results show that GWO does the best on average, and FLO does the second best then WOA and HHO, However, don't work as well as the other two. These results clearly show that the algorithms have statistically significant differences in how well they work.

TABLE I: THE DETAILS ON CEC 2022 TEST SUITES

Items	Function	Name	Global
CEC-01	Unimodal	Shifted and full rotated Zakharov	300
CEC-02	Basic	Shifted and full rotated Rosenbrock's	400
CEC-03		Shifted and full rotated expanded Schaffer's f_6	600
CEC-04		Shifted and full rotated Non-Continuous Rastrigin	800
CEC-05		Shifted and full rotated Levy	900
CEC-06	Hybrid	Hybrid 1 (N = 3)	1800
CEC-07		Hybrid function 2 (N = 6)	2000
CEC-08		Hybrid function 3 (N = 5)	2200
CEC-09	Composition	Composition 1 (N = 5)	2300
CEC-10		Composition (N = 4)	2400
CEC-11		Composition (N = 5)	2600
CEC-12		Composition (N = 6)	2700

TABLE II: NUMERICAL ANALYSIS OF CEC 2022

Functions	Algorithms	Mean	Std.	Rank
CEC-1	FLO	5.294992e+03	1.823834e+03	2
	GWO	4.039493e+03	3.048057e+03	1
	WOA	2.413840e+04	9.937307e+03	4
	HHO	9.622024e+03	9.524890e+02	3
CEC-2	FLO	8.220557e+02	2.738410e+02	3
	GWO	4.270600e+02	2.425674e+01	1
	WOA	4.915696e+02	9.657666e+01	2
	HHO	1.001694e+03	4.723462e+02	4
CEC-3	FLO	6.257311e+02	5.632975e+00	2
	GWO	6.014999e+02	1.599664e+00	1
	WOA	6.397748e+02	1.383836e+01	3
	HHO	6.438784e+02	1.295612e+01	4
CEC-4	FLO	8.252735e+02	6.779356e+00	2
	GWO	8.147174e+02	9.005514e+00	1
	WOA	8.411971e+02	1.636371e+01	4
	HHO	8.379934e+02	1.022987e+01	3
CEC-5	FLO	1.058348e+03	9.348999e+01	2
	GWO	9.194269e+02	2.682765e+01	1
	WOA	1.618867e+03	8.081768e+02	4
	HHO	1.397115e+03	1.525194e+02	3
CEC-6	FLO	1.407474e+07	2.368284e+07	4
	GWO	6.382354e+03	2.447005e+03	2
	WOA	4.677420e+03	1.829022e+03	1
	HHO	1.494703e+06	3.810926e+06	3
CEC-7	FLO	2.057043e+03	1.395216e+01	2
	GWO	2.029102e+03	1.116272e+01	1
	WOA	2.082629e+03	3.250243e+01	3
	HHO	2.106319e+03	3.372957e+01	4
CEC-8	FLO	2.241102e+03	3.560302e+01	3
	GWO	2.229593e+03	2.233458e+01	1
	WOA	2.236112e+03	8.463430e+00	2
	HHO	2.241630e+03	2.396184e+01	4
CEC-9	FLO	2.695418e+03	3.946632e+01	3
	GWO	2.580883e+03	4.178399e+01	1
	WOA	2.606755e+03	5.602925e+01	2
	HHO	2.738435e+03	4.667999e+01	4
CEC-10	FLO	2.584221e+03	6.936166e+01	2
	GWO	2.564568e+03	8.782647e+01	1
	WOA	2.665456e+03	3.396003e+02	3
	HHO	2.958359e+03	5.472212e+02	4
CEC-11	FLO	3.153066e+03	3.311690e+02	3
	GWO	2.985134e+03	1.604446e+02	1
	WOA	3.005841e+03	1.275071e+02	2
	HHO	3.822613e+03	5.608837e+02	4
CEC-12	FLO	2.940101e+03	3.774951e+01	3
	GWO	2.871168e+03	1.194024e+01	1
	WOA	2.912436e+03	4.653411e+01	2
	HHO	2.964702e+03	5.766556e+01	4

TABLE III: FRIEDMAN RANK TEST FOR THE OPTIMIZERS

Function	FLO	GWO	WOA	HHO
CEC-1	1.77	1.33	3.90	3.00
CEC-2	3.27	1.10	1.97	3.67
CEC-3	2.27	1.00	3.23	3.50
CEC-4	2.13	1.20	3.40	3.27
CEC-5	2.13	1.00	3.50	3.37
CEC-6	3.43	2.20	1.73	2.63
CEC-7	2.20	1.17	3.00	3.63
CEC-8	2.33	1.53	3.03	3.10
CEC-9	3.10	1.47	1.73	3.70
CEC-10	2.53	1.77	2.40	3.30
CEC-11	2.20	1.87	2.20	3.73
CEC-12	3.23	1.03	2.40	3.30
Sum Ranks	30.59	16.67	32.49	40.20
Mean ranks	2.55	1.39	2.71	3.35

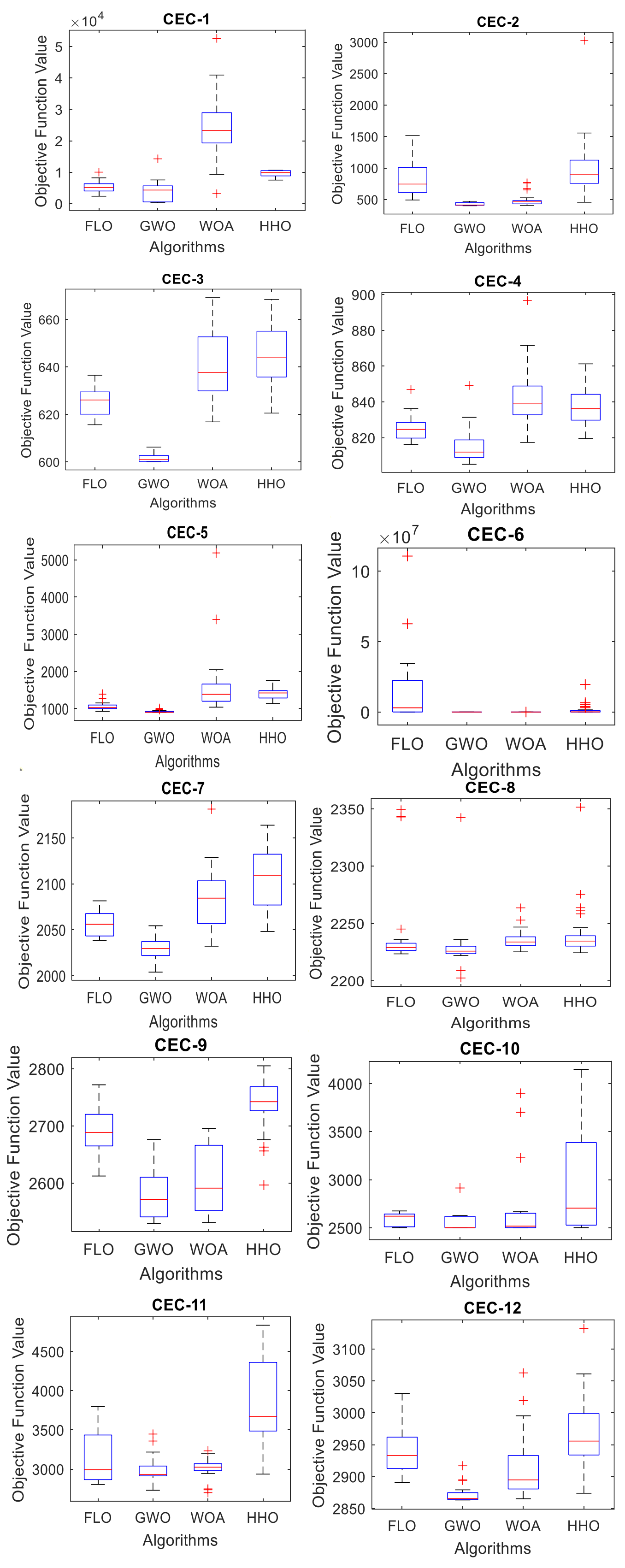


Fig. 7. Boxplot analysis for Optimizers

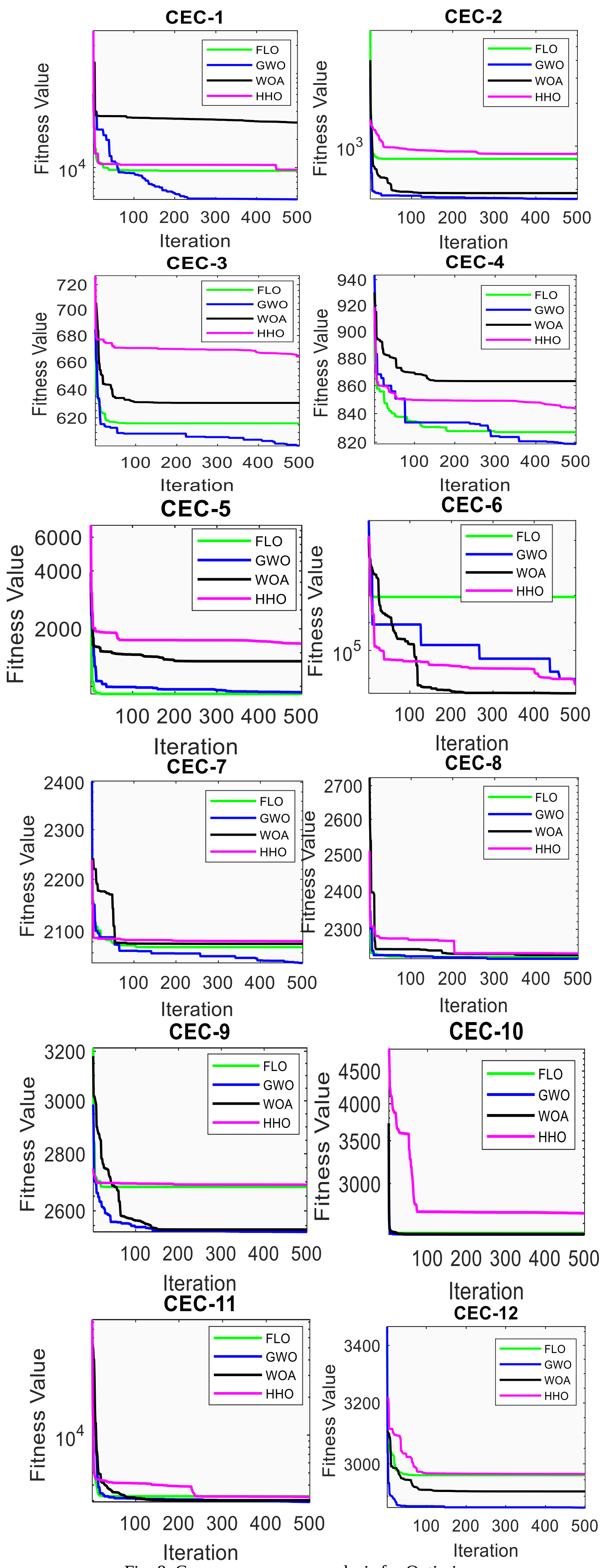


Fig. 8. Convergence curve analysis for Optimizers

C. Convergence Analysis

The convergence curve assessment of the four optimizers provides substantial insights when juxtaposed with various established optimization approaches across multiple benchmark functions. GWO always performs better, with fast convergence and high-quality solutions. For instance, in the IEEE CEC 2022 testing suites, GWO not only converges faster but also reaches lower ultimate objectives values, which shows its strong global optimization skills in fig.8. The convergence curves for GWO, on the other hand, have a steeper slope, which means that GWO gets to the best solutions faster than optimizers like FLO, WOA, and HHO.

D. Boxplot Analysis

Boxplot analysis is used to statistically test and visually assess how well the suggested swarm-driven computational intelligence strategy works compared to other techniques on the CEC 2022 test functions shown in fig 7. The boxplots show how the fitness values from several independent runs are spread out. This findings show that the GWO method consistently has superior convergence behavior with a decrease across most evaluation functions than competing optimizers. This shows that the suggested strategy works well for keeping the quality of the solution while avoiding early convergence, especially in complicated function landscapes.

IV. Conclusions

This study introduces the CEC 2022 benchmark set for evaluating four swarm-based intelligent computing procedures: FLO, WOA, GWO, and HHO. The results show that each analyzer has its own advantages. GWO performed a terrific job of investigating the broad universe of multimodal functions, and it fared well in most benchmark areas with a steady and balanced performance. WOA was able to rival with other local exploiting approaches, while HHO was able to swiftly converge because its adaptive exploitation- exploration mechanism. Statistical testing indicated that the algorithms were very different from each other. This supports the argument that you should choose a strategy based on the issue instead of just utilizing one optimizer. In the future, we will focus on including changing parameter control to these procedures and testing how well they work on real-world engineering problems that have some kind of restriction, multiple goals, or both. Using deep-learning-assisted models or surrogate-based strategies may also improve the scalability and effectiveness of high-dimensional search spaces.

References

[1] Hussain, A., Muhammad, Y.S., Sajid, M.N.: Performance Evaluation of Best-Worst Selection Criteria for Genetic Algorithm. In: Mathematics and Computer Science 2(6), 89–97 2017, https://doi.org/10.11648/j.mcs.20170206.12.

[2] Zhang, J., Cheng, X., Zhao, M. *et al.* ISSWOA: hybrid algorithm for function optimization and engineering problems. *J Supercomput* 79, 8789–8842 (2023). https://doi.org/10.1007/s11227-022-04996-1.

[3] B. Tabbara, A. Tabbara, and A. Sangiovanni-Vincentelli, “Function optimizations,” in *Function/Architecture Optimization and Co-Design of Embedded Systems*, 2000, ch. 4. doi: 10.1007/978-1-4615-4359-6_4.

[4] Dharmapuri, V., Dutta, S.R. A novel cluster of improved frilled lizard optimization and multi-ladder gated networks for the detection of cyber-attacks in computer networks. *Discov Computing* 28, 120 (2025). https://doi.org/10.1007/s10791-025-09647-6.

[5] Liu, Y., As’arry, A., Hassan, M.K. *et al.* Review of the grey wolf optimization algorithm: variants and applications. *Neural Comput & Applic* 36, 2713–2735 (2024). https://doi.org/10.1007/s00521-023-09202-8.

[6] Sun, G., Shang, Y., Yuan, K. *et al.* An Improved Whale Optimization Algorithm Based on Nonlinear Parameters and Feedback Mechanism. *Int J Comput Intell Syst* 15, 38 2022, https://doi.org/10.1007/s44196-022-00092-7.

[7] Rodan, A. Enhanced Frilled Lizard Optimizer for Global Optimization and Engineering Design Problems. *Int J Comput Intell Syst* 18, 257 (2025). https://doi.org/10.1007/s44196-025-00969-3

[8] Rana, N., Latiff, M.S.A., Abdulhamid, S.M. *et al.* Whale optimization algorithm: a systematic review of contemporary applications, modifications and developments. *Neural Comput & Applic* **32**, 16245–16277 (2020). https://doi.org/10.1007/s00521-020-04849-z

[9] Rezaei, H., Bozorg-Haddad, O., Chu, X. (2018). Grey Wolf Optimization (GWO) Algorithm. In: Bozorg-Haddad, O. (eds) Advanced Optimization by Nature-Inspired Algorithms. Studies in Computational Intelligence, vol 720. Springer, Singapore. https://doi.org/10.1007/978-981-10-5221-7_9

[10] Shehab, M., Mashal, I., Momani, Z. *et al.* Harris Hawks Optimization Algorithm: Variants and Applications. *Arch Computat Methods Eng* 29, 5579–5603 2022, https://doi.org/10.1007/s11831-022-09780-1.