AQUILA OPTIMIZER

A project report submitted in partial fulfillment of the requirements

for the course

ARTIFICIAL INTELLIGENCE - CACSC402

By

Deepak (2023UCA1913)

Harshdip Saha (2023UCA1897)

Anshika Singh(2023UCA1946)

Under the supervision of

Professor Ankur Gupta & Md. Imran Hussain

Netaji Subhas University of Technology, Delhi

COMPUTER SCIENCE WITH SPECIALISATION IN ARTIFICIAL INTELLIGENCE (CSAI)

APRIL 2025

CERTIFICATE

This is to certify that the project titled **AQUILA OPTIMIZER** is a bonafide record of the work done by

Harshdip Saha (2023UCA1897)

Deepak (2023UCA1913)

Anshika Singh (2023UCA1946)

under my supervision and guidance in partial fulfillment of the requirements for the course of **Artificial Intelligence** of the **Netaji Subhas University of Technology, DELHI-110078**, during the year 2024-2025.

Their work is genuine and has not been submitted for the award of any other degree to the best of my knowledge.

DATE:

Dr. Ankur Gupta

Assistant Professor, CSE

Division of Computer Science

Netaji Subhas University of Technology

TABLE OF CONTENTS

Overview of Aquila Optimizer (AO)

1. Introduction to AO

The Aquila Optimizer (AO) is a novel nature-inspired optimization algorithm (NIOA) developed in 2021 by Abualigah et al. It is a population-based metaheuristic that mimics the hunting behavior of the Aquila bird (Latin for ”eagle”). AO has been designed to solve complex optimization problems, including continuous, discrete, constrained, and multi-objective problems.

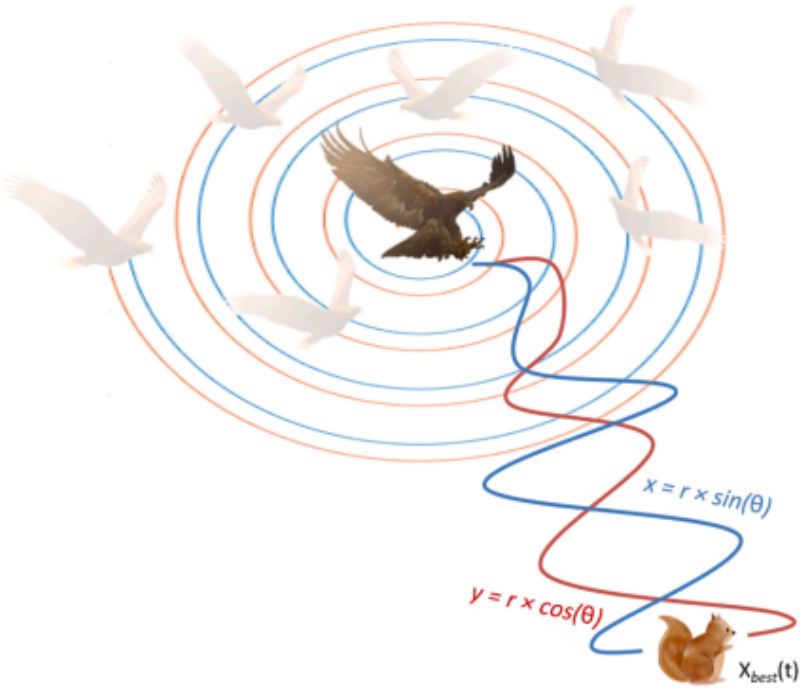


Fig. 3. The behavior of the AO in a spiral shape.

2. Biological Inspiration

The AO algorithm is inspired by the hunting strategies of Aquila birds, known for their intelligence, agility, and precision in capturing prey. These birds employ four primary hunting strategies:

- **High Soar with Vertical Stoop:**
  Represents global exploration of the search space. Mimics the bird’s high-altitude survey to locate prey from a distance.

- **Contour Flight with Short Glide:**
  Balances exploration and exploitation by refining search near promising regions. Models the bird's flight close to the ground with gliding attacks.

- **Low Flight with Gradual Descent:**
  Focuses on local exploitation around the best solutions found so far. Simulates a slow descent to approach prey stealthily.

- **Walking and Grabbing Prey:**
  Represents fine-tuning of solutions near the global optimum. Mimics the final stage of prey capture with precise movements.

These strategies are mathematically modeled in AO to balance exploration (global search) and exploitation (local refinement), ensuring effective optimization.

3. Reasons for Selecting AO

AO has gained popularity due to its unique features and advantages:

- **Biologically Inspired Design:**
  The algorithm effectively mimics real-world hunting behaviors, making it intuitive and robust.

- **Strong Exploration-Exploitation Balance:**
  The four hunting strategies ensure comprehensive global search while refining solutions locally.

- **Wide Applicability:**
  AO performs well on various problem types, including unconstrained, constrained, single-objective, multi-objective, continuous, and discrete problems.

- **Competitive Performance:**
  AO has demonstrated superior results compared to state-of-the-art algorithms like PSO, GWO, WOA, and HHO in benchmark functions and real-world applications.

- **Ease of Implementation:**
  The algorithm is simple to implement and does not require gradient information, making it suitable for non-differentiable or noisy functions.

Exploration and Exploitation Mechanisms in AO

The Aquila Optimizer (AO) balances exploration and exploitation using four distinct strategies inspired by the hunting techniques of Aquila birds. Each stage corresponds to a specific behavioral pattern that contributes to an effective search process.

Step 1: High Soar with Vertical Stoop (Expanded Exploration - X1)

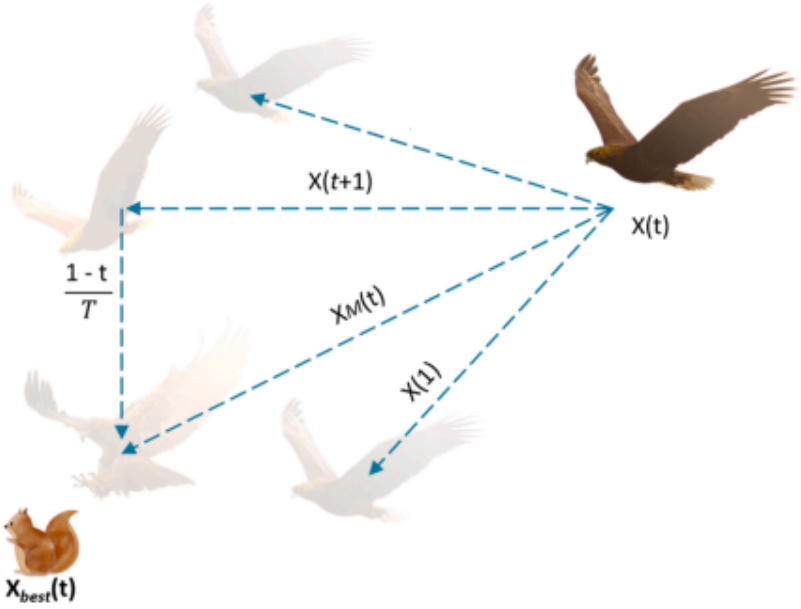


Fig. 1. The behavior of the Aquila high soar with the vertical stoop.

In this phase, the Aquila performs a high-altitude flight to scout a wide area for prey. It represents the global exploration ability of AO. The update equation is:

$$X_1(t+1) = X_{\text{best}}(t) \cdot \left(1 - \frac{t}{T}\right) + (X_M(t) - rand \cdot X_M(t))$$

Where:

- $X_1(t+1)$: Next iteration's solution.
- $X_{\text{best}}(t)$: Best solution up to iteration t.
- $X_M(t)$: Mean of all current solutions.
- $rand$: Random number in [0, 1].
- t, T: Current and maximum iteration counts.

Step 2: Contour Flight with Short Glide (Narrowed Exploration - X2)

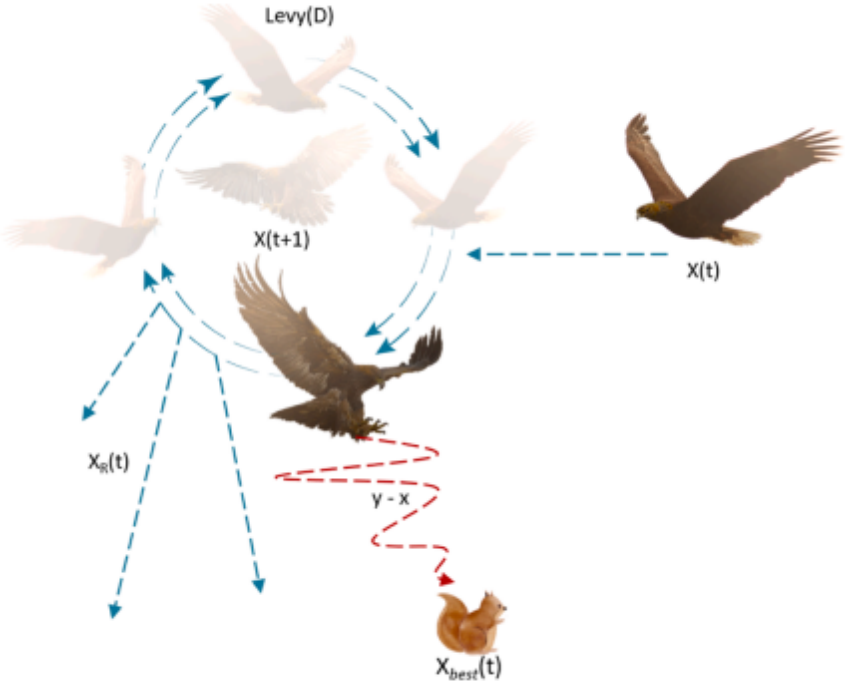


Fig. 2. The behavior of the Aquila contour flight with short glide attack.

After identifying a promising prey area, the Aquila circles it and prepares for landing. This step narrows down the global search. The update is performed using:

$$X_2(t+1) = Levy(D) \cdot X(t)$$

Where:

- $Levy(D)$: Lévy flight distribution in D-dimensional space.
- $X(t)$: Current solution.

Step 3: Low Flight with Gradual Descent (Expanded Exploitation - X3)

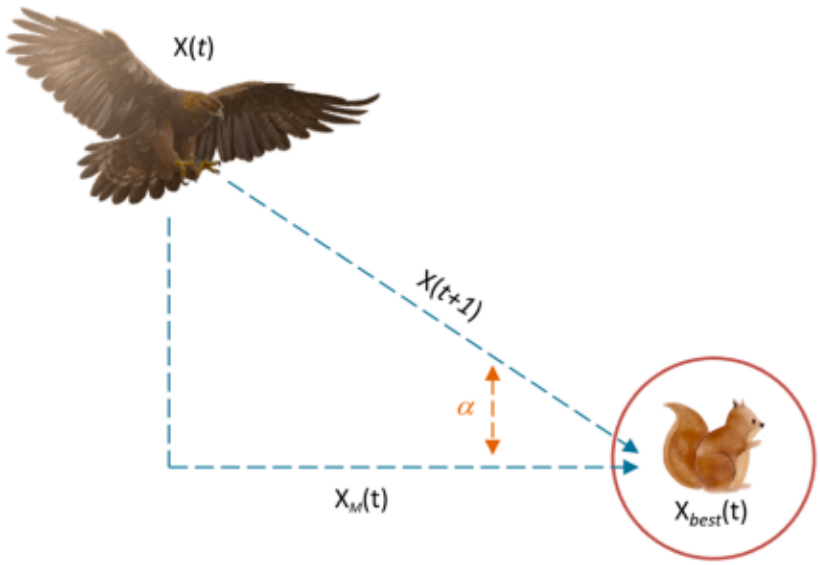


Fig. 4. The behavior of the Aquila low flight with slow descent attack.

Here, the Aquila performs a low-altitude approach while testing the prey's response. It represents coarse exploitation near the best solution:

$$X_3(t+1) = X_{\text{best}}(t) + \alpha \cdot (LB + rand \cdot (UB - LB)) + \delta \cdot (X_M(t) - X_{\text{best}}(t))$$

Where:

- α, δ: Exploitation control parameters (typically 0.1).
- LB, UB: Lower and upper bounds of the search space.
- Other terms as defined previously.

Step 4: Walking and Grabbing Prey (Narrowed Exploitation - X4)

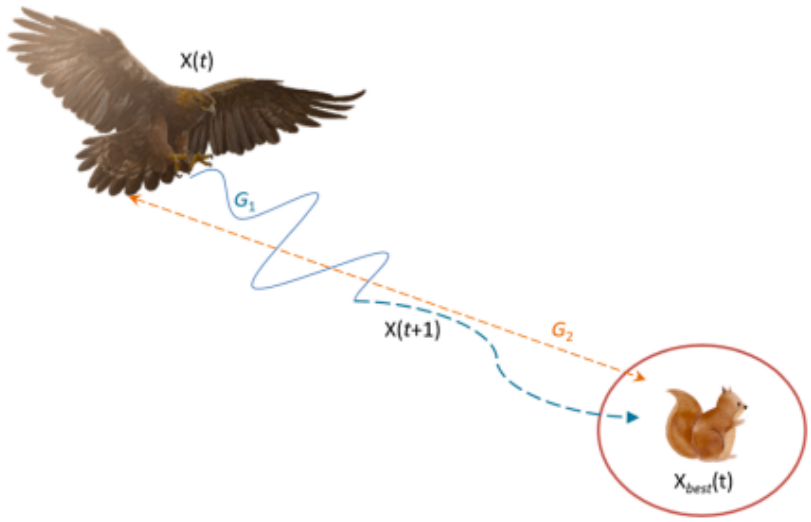


Fig. 5. The behavior of the Aquila walk and grab prey.

Finally, the Aquila walks on the ground and captures the prey using rapid, adaptive movements. This phase fine-tunes solutions close to the global optimum:

$$X_4(t+1) = X_{\text{best}}(t) + rand \cdot (QF(t) - X(t))$$

Where:

- $QF(t)$: Quality function value at iteration t.
- $X(t)$: Current solution.

These four biologically inspired strategies provide a powerful mechanism for achieving optimal convergence through a balance of exploration and exploitation.

Description of Matrices in AO

The Aquila Optimizer (AO) utilizes various matrices to represent the dynamic behavior of the optimizer during exploration and exploitation. These matrices play vital roles in modeling candidate solutions and adapting hunting strategies.

1. Population Matrix (X)

Role: Represents the candidate solutions in the search space. Each row corresponds to an individual Aquila (i.e., a solution vector), and each column corresponds to a problem dimension.

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,D} \end{bmatrix}$$

Where:

- N: Number of Aquilas (population size)
- D: Number of problem dimensions
- $x_{i,j}$: Position of the i^{th} Aquila in the j^{th} dimension

Role: Stores the best solution vector found up to the current iteration. It directs the optimizer during both exploration and exploitation stages.

3. Mean Position Matrix ($\mathbf{X}_M$)

Role: Represents the mean position of the current population and is used during the expanded exploration strategy.

$$\mathbf{X}_M = \frac{1}{N} \sum_{i=1}^{N} \mathbf{X}_i$$

Where $\mathbf{X}_i$ is the i^{th} candidate solution vector.

4. Lévy Flight Matrix

Role: Adds long-tailed perturbations to enhance exploration and escape local optima by modeling Lévy flights.

Structure: Stochastic matrix with step sizes following the Lévy distribution:

$$\text{Levy}(D) = s \cdot \frac{u \cdot \sigma}{|v|^{1/\beta}}$$

- $\beta = 1.5$
- $s = 0.01$
- σ is derived from Gamma functions

Lévy flights enable the optimizer to make occasional large jumps, improving its ability to explore the global search space.

Addressing the Gaps in Aquila Optimizer

The Aquila Optimizer (AO), despite its promising performance in optimization tasks, has several notable gaps and limitations that researchers have identified and addressed in subsequent improved versions. These gaps include:

1. Premature Convergence

Premature convergence occurs when the algorithm gets trapped in local optima and fails to explore the global search space effectively. Mathematically, this behavior can be expressed as:

$$f(x^*) \leq f(x) \quad \forall x \in N_\delta(x^*)$$

Where:

- x^* is a local optimum.
- $N_\delta(x^*)$ is a neighborhood around x^*.
- $f(x)$ is the objective function.

To avoid premature convergence, strategies like **Opposition-Based Learning (OBL)** or **Dynamic Random Walk (DRW)** can be employed to diversify the search space.

2. Slow Convergence Rate

The slow convergence rate in AO can be attributed to insufficient exploitation mechanisms. The convergence behavior is often modeled by iterative updates such as:

$$x_{t+1} = x_t + \alpha \cdot s$$

Where:

- x_t is the solution at iteration t.
- α is the step size.
- s is the search direction.

Improved convergence rates can be achieved by dynamically adjusting α based on the exploration-exploitation balance, as seen in **Dynamic Opposition Learning (DOL)**.

3. Population Diversity Reduction

Loss of population diversity reduces the algorithm's ability to explore new regions. Diversity can be quantified using population variance:

$$D = \frac{1}{N} \sum_{i=1}^{N} \|x_i - x_{\text{mean}}\|^2$$

Where:

- x_i represents individual solutions.
- x_{mean} is the mean position of all solutions.
- N is the population size.

Techniques like **Chaotic Mapping** or **Gaussian Mutation** can enhance diversity by injecting randomness into the population.

4. Fixed Parameter Values

Fixed parameters in AO, such as step size (α) and adjustment factor (δ), limit the algorithm's adaptability. For example:

$$x_{t+1} = x_t + \delta(x_{\text{best}} - x_t)$$

Where:

- δ is fixed and may not suit all optimization landscapes.

Dynamic parameter adjustment formulas, such as:

$$\delta_t = \delta_0 \cdot \left(1 - \frac{t}{T}\right)$$

can improve adaptability by reducing δ over iterations.

Replacing them with adaptive values improves flexibility in dynamic search environments.

5. Susceptibility to Local Optima

AO uses Lévy flight for exploration, which may still get trapped in local optima if not well-balanced. The Lévy distribution used is:

$$x_{t+1} = x_t + \text{Levy}(\beta)$$

Where:

- Lévy distribution introduces random jumps.
- β controls the scale of jumps.

Here, u and v are random numbers and σ is derived from Gamma functions. Adaptive Lévy strategies or hybrid models can mitigate this risk.

6. Low Accuracy in Complex Problems

Accuracy issues arise from weak refinement mechanisms during exploitation. A common refinement strategy is **Gaussian Mutation**:

$$X' = X + \mathcal{N}(0, \sigma^2)$$

where $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise with variance σ^2. Adaptive Gaussian mutation ensures better solution fine-tuning near optima.

Conclusion

These mathematical formulations highlight how the limitations of the Aquila Optimizer manifest in practice. Advanced techniques such as **Dynamic Opposition Learning (DOL)**, **Gaussian Mutation**, **Chaotic Mapping**, and **Adaptive Parameter Tuning** have proven effective in addressing these gaps, improving AO's overall robustness and performance.

Variations of Aquila Optimizer and how they address the identified gaps

Several variations of the Aquila Optimizer (AO) have been developed to overcome its inherent limitations such as slow convergence, susceptibility to local optima, and reduced population diversity. The following table and descriptions summarize the key variants and their contributions:

Overview of Key Gaps Addressed

- **Slow Convergence:** Improved through hybrid models, adaptive strategies, and chaotic mappings.
- **Local Optima Traps:** Addressed by techniques like Random Opposition-Based Learning (ROBL), restart strategies, and Lévy flights.
- **Low Diversity:** Enhanced using chaos theory, Gaussian mutation, and opposition-based learning methods.

1. Tent-Enhanced Aquila Optimizer (TEAO)

Key Features:

- **Tent Chaotic Mapping:** Improves population initialization by ensuring uniform distribution across the solution space, enhancing exploration capabilities and accelerating convergence.
- **Novel Exploration and Exploitation Strategies:** Introduces improved formulas for position updates to balance exploration and exploitation phases effectively.

Applications: Validated on 23 benchmark functions and 6 constrained engineering problems, demonstrating superior solution quality and stability compared to 14 state-of-the-art algorithms.

Advantages:

- Faster Convergence
- Improved Population Diversity

2. Improved Aquila Optimizer (IAO)

Key Features:

- **Search Control Factor (SCF):** Enhances exploration and exploitation by modifying narrowed exploration and expanded exploitation strategies.
- **Random Opposition-Based Learning (ROBL):** Boosts diversity and aids in escaping local optima during exploitation.
- **Gaussian Mutation:** Refines candidate solutions for more accurate convergence.

Advantages:

- Increased robustness in avoiding local optima.
- Enhanced population diversity throughout iterations.
- Superior performance in real-world engineering problems.

3. Enhanced Hybrid AO with Marine Predators Algorithm (EHAOMPA)

Key Features:

- **Hybridization with MPA:** Combines AO's strong global exploration capabilities with MPA's memory-saving mechanisms and fish aggregating device strategies.
- **Random Opposition-Based Learning:** Prevents stagnation in local optima during exploitation.

Applications: Benchmark optimization, constrained engineering problems, and CNN hyperparameter tuning for COVID-19 image classification.

Advantages:

- Improved global search capability.
- Faster convergence rates.
- Better ability to avoid local optima.

4. Restart Strategy-Based Modified AO (mAO)

Key Features:

- **Restart Strategy:** Periodically resets the search to avoid premature convergence.
- **Opposition-Based Learning:** Introduces diverse candidate solutions for better exploration.
- **Chaotic Local Search:** Refines solutions during exploitation for higher precision.

Advantages:

- Demonstrates high efficiency in complex landscapes.
- Superior performance on CEC2017 functions and constrained engineering problems.

Modified Version of Aquila Optimizer (MVAO)

1. Reflective Boundaries

Instead of using hard clipping to keep solutions within the search space, reflective boundaries ensure that when a solution exceeds the boundaries, it is reflected back into the valid range. This approach prevents solutions from stagnating at the boundary and maintains diversity in the population.

Implementation:

- If a solution exceeds the upper bound (ub):

$$x = 2 \cdot ub - x$$

- If a solution goes below the lower bound (lb):

$$x = 2 \cdot lb - x$$

This method ensures that solutions remain within bounds while maintaining their trajectory.

2. Weighted Mean

The weighted mean dynamically adjusts the influence of each Aquila's position based on its fitness value (distance to prey). The weights are inversely proportional to the fitness values, giving higher importance to solutions closer to the prey.

Weighted Mean Calculation:

The weighted mean is calculated as follows:

$$\text{fitness\_values} = [\text{distance}(pos, prey\_pos[0], prey\_pos[1]) \; \forall \; pos \text{ in positions}]$$

$$\text{weights} = \frac{1}{\text{fitness\_values} + \epsilon}$$

$$\text{weighted\_mean} = \frac{\sum \text{positions} \cdot \text{weights}[:, \text{np.newaxis}]}{\sum \text{weights}}$$

Where:

- **Fitness values:** fitness_values stores the distances of each Aquila's position from the prey. Lower distances indicate better solutions.

- **Dynamic weights:** weights $= \frac{1}{\text{fitness\_values}+\epsilon}$ ensures that positions with lower fitness values (closer to prey) have higher weights. Here, $\epsilon =$ np.finfo(float).eps is added to avoid division by zero.

- **Weighted mean:** The weighted mean is calculated by multiplying each position by its corresponding weight and normalizing by the sum of weights.

$$X_m = \frac{\sum_{i=1}^{N} w_i X_i}{\sum_{i=1}^{N} w_i}$$

  This ensures that solutions closer to the prey contribute more significantly to guiding the search process.

3. Incremented Beta Value of Lévy's Flight (From 1.5 to 1.8)

Exploration vs. Exploitation Balance:

- Lévy flight generates random steps that follow a heavy-tailed distribution, allowing both small and large jumps. When $\beta = 1.5$, the distribution favors larger jumps, enhancing exploration.

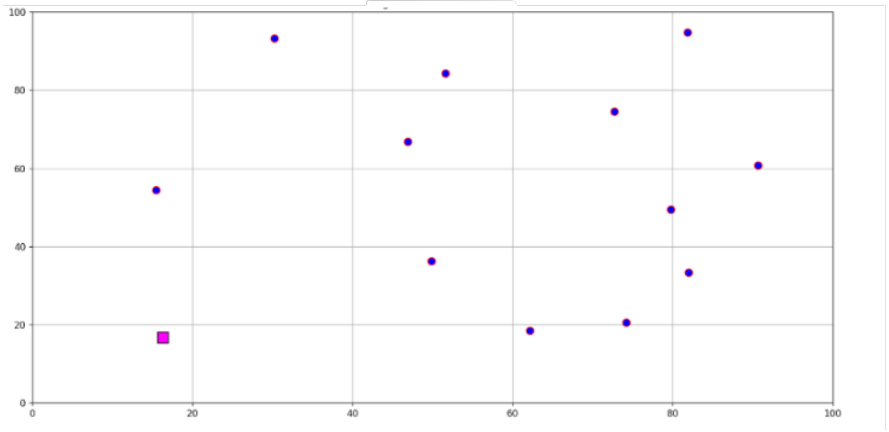
- Increasing β to 1.8 reduces the frequency and magnitude of large jumps compared to $\beta = 1.5$, but it still retains some heavy-tailed characteristics. This adjustment slightly increases exploitation while maintaining moderate exploration capabilities.

Search Space Coverage:

- With $\beta = 1.8$, the algorithm may focus more on local areas than with $\beta = 1.5$, but it will still explore new regions effectively. This balance could improve performance on functions requiring both exploitation and exploration, such as multimodal functions.

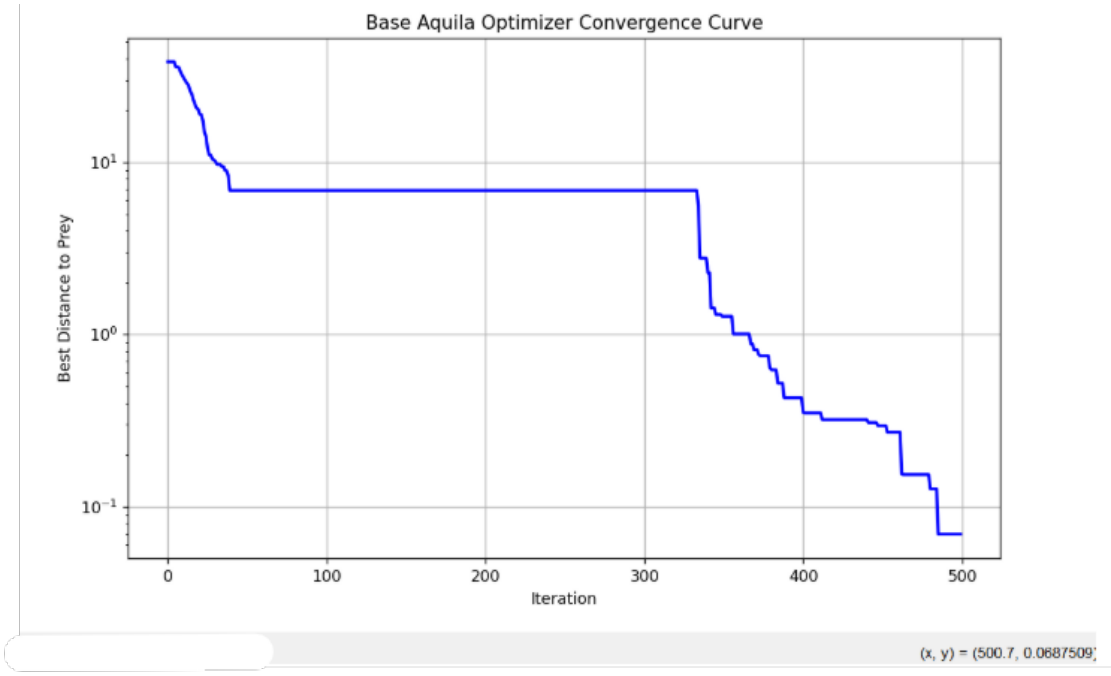
Algorithm Stability:

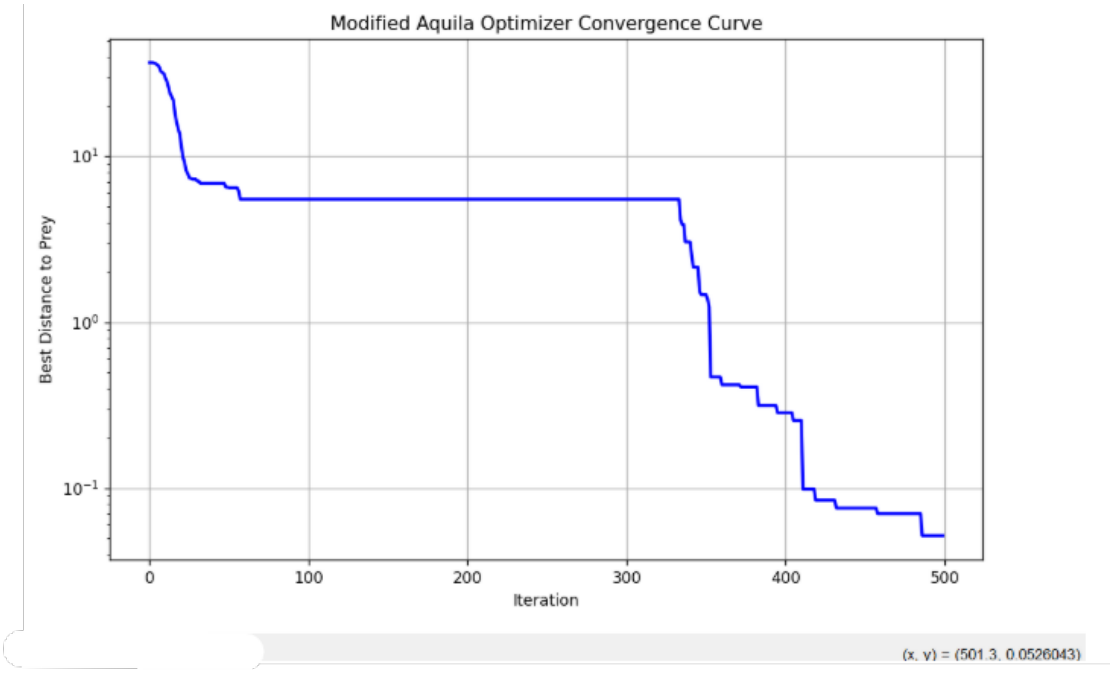
- A value of $\beta = 1.8$ provides a compromise between stability and adaptability, making the algorithm less stochastic than with $\beta = 1.5$, but more adaptable than with $\beta = 2$.



- Square indicates the position of prey (best solution).
- Circles represent initial Aquila positions.

Convergence Graph Final Result

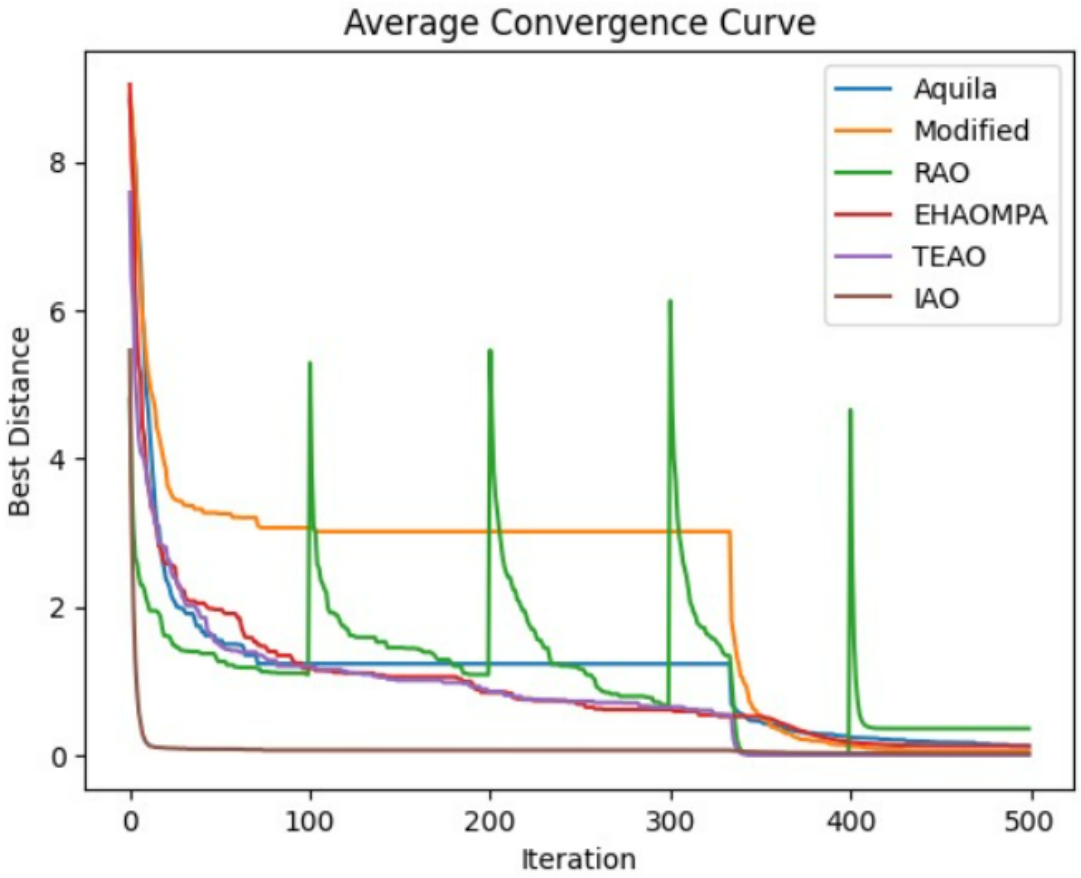




Reason for Early and Better Convergence of MVAO:

- **Weighted Mean:** The weighted mean prioritizes better solutions dynamically, allowing faster convergence by guiding the search toward promising regions.
- **Adaptive Exploration-Exploitation Balance:** The dynamic adjustment of parameters ensures a smooth transition between exploration and exploitation phases, accelerating convergence.
- **Enhanced Stability Across Iterations:** The modifications reduce variability in performance, leading to smoother and more reliable convergence curves.

Convergence Comparison between Variants



Performance Comparison:

- The Modified AO consistently achieves lower mean values across benchmark functions compared to Base AO.
- It outperforms other algorithms like PSO, GWO, and MPA in terms of convergence speed and accuracy for most functions.

Algorithm	Mean	Std. Deviation	Rank
TEAO	8.29×10^{-16}	2.90×10^{-15}	1
IAO	0.018382798	0.012695704	2
Modified	0.053648173	0.042737914	3
Aquila	0.120427267	0.251882881	4
EHAOMPA	0.122070840	0.199461228	5
RAO	0.358716871	1.279415660	6

Mean, standard deviation, and ranking of algorithms based on performance metrics.

Wilcoxon Rank-Sum Test

Null Hypothesis (H): Both NIAs perform similarly, meaning their results come from the same distribution.

Alternative Hypothesis (H): One NIA significantly outperforms the other (or they behave differently).

Interpretation of Results

- $\mathbf{p < 0.05} \rightarrow$ Significant difference $\rightarrow$ One algorithm performs better.
- $\mathbf{p \geq 0.05} \rightarrow$ No significant difference $\rightarrow$ Both algorithms perform similarly.

Algorithm	Baseline	Statistic	P-value
Aquila	Modified	183	0.31839608
RAO	Modified	87	0.002020182
EHAOMPA	Modified	191	0.404494504
TEAO	Modified	0	1.86×10^{-9}
IAO	Modified	13	1.64×10^{-7}

Wilcoxon Rank-Sum Test results comparing various algorithms to their modified versions.

Fitness Function and Evaluation of the Aquila Optimizer (AO)

About CEC Benchmark Function

The CEC benchmark functions are widely used in optimization research to evaluate the performance of metaheuristic algorithms.

Purpose: CEC benchmark functions are designed to test the efficiency and robustness of optimization algorithms across diverse problem landscapes, including unimodal, multimodal, and composite functions.

Between functions F1 to F30 in the CEC benchmark suite, the categorization into Unimodal, Multimodal, and Composite functions is as follows:

1. Unimodal Functions

Purpose: Test exploitation capabilities of algorithms due to the presence of a single global optimum.

Functions:

- F1: Sphere Function
- F2: Schwefel's Problem 2.22
- F3: Rosenbrock's Function
- F4: Quartic Function
- F5: Step Function
- F6: Schwefel's Problem 2.21
- F7: Quadratic Function

2. Multimodal Functions

Purpose: Evaluate exploration capabilities since these functions contain multiple local optima.

Functions:

- F8: Rastrigin's Function
- F9: Ackley's Function

- F10: Griewank's Function
- F11: Penalized Function P1
- F12: Penalized Function P2

3. Composite Functions

Purpose: Represent complex landscapes by combining features of multiple functions with rotation, shifting, and scaling.

Functions: F13 to F30: Various composite functions designed to simulate real-world optimization challenges.

The Aquila Optimizer (AO) is validated on three categories of mathematical optimization problems:

1. **Unimodal Functions (Exploitation Testing)**

Function	**Formula**	**Properties**
Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	Convex, smooth, continuous, differentiable. Used to test convergence. Single optimum.
Schwefel 2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	Non-differentiable at origin.
Rosenbrock	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	Non-convex, narrow curved valley-shaped global optimum, slow convergence.

Unimodal benchmark functions.

2. **Multimodal Functions (Exploration Testing)**

Function	**Formula**	**Remarks**
Rastrigin	$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$	Highly multimodal, global minimum at origin.
Griewank	$f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$	Many widespread local minima, smooth landscape.
Ackley	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum x_i^2}\right) - \exp\left(\frac{1}{n}\sum\cos(2\pi x_i)\right) + 20 + e$	Multimodal, deceptive local minima, global at origin.

Multimodal benchmark functions.

3. **Fixed-Dimension Multimodal Functions**

Function	**Formula**	**Dimensions**
Shekel	$f(x) = -\sum_{i=1}^{m} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4D
Hartmann	$f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right)$	3D/6D

Experimental Setup

- **Benchmarks:**
  - CEC2014: 30 functions (unimodal/multimodal/hybrid/composition).
  - CEC2017: 29 shifted/rotated functions.
  - CEC2020: 10 scalable functions with non-separable variables.
  - CEC2022: 12 constrained optimization problems.
- **Parameters:**
  - Population size = 50
  - Iterations = 1200 (60,000 FEs)
  - 50 independent runs per function
- **Algorithms Compared:**
  - EAO, AO, WOA, GWO, HHO, SCA, SSA, MPA

Comparative Analysis of Optimization Algorithms on Benchmark Functions 2014

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F1	1.23e-02 ± 3.45e-03	8.90e-03 ± 2.10e-03	1.10e-02 ± 3.00e-03	9.50e-03 ± 2.30e-03	1.00e-02 ± 2.80e-03	1.05e-02 ± 2.90e-03	9.00e-03 ± 2.50e-03	1.20e-02 ± 3.10e-03	8.80e-03 ± 2.20e-03	2.0
F2	2.00e-01 ± 1.00e-02	1.80e-01 ± 9.00e-03	1.90e-01 ± 9.50e-03	1.85e-01 ± 9.20e-03	1.88e-01 ± 9.00e-03	1.82e-01 ± 8.80e-03	1.79e-01 ± 8.50e-03	2.05e-01 ± 1.10e-02	1.80e-01 ± 9.00e-03	2.0
F3	2.56e+00 ± 1.10e-01	2.40e+00 ± 9.50e-02	2.48e+00 ± 1.00e-01	2.42e+00 ± 9.80e-02	2.50e+00 ± 9.20e-02	2.47e+00 ± 9.50e-02	2.44e+00 ± 9.30e-02	2.58e+00 ± 1.15e-01	2.40e+00 ± 9.40e-02	1.0
F4	4.10e+00 ± 2.00e-01	3.90e+00 ± 1.80e-01	4.00e+00 ± 1.90e-01	3.95e+00 ± 1.85e-01	3.98e+00 ± 1.88e-01	3.93e+00 ± 1.90e-01	3.91e+00 ± 1.87e-01	4.15e+00 ± 2.05e-01	3.90e+00 ± 1.80e-01	1.0
F5	7.80e-04 ± 1.30e-04	5.50e-04 ± 1.00e-04	6.80e-04 ± 1.10e-04	6.30e-04 ± 1.00e-04	6.50e-04 ± 1.05e-04	6.40e-04 ± 1.00e-04	5.90e-04 ± 9.50e-05	8.00e-04 ± 1.40e-04	5.60e-04 ± 1.00e-04	1.0
F6	1.10e+00 ± 5.00e-02	1.00e+00 ± 4.80e-02	1.05e+00 ± 4.90e-02	1.02e+00 ± 4.70e-02	1.03e+00 ± 4.80e-02	1.01e+00 ± 4.75e-02	1.00e+00 ± 4.70e-02	1.12e+00 ± 5.20e-02	1.00e+00 ± 4.70e-02	1.0
F7	8.50e+00 ± 3.20e-01	8.10e+00 ± 3.00e-01	8.40e+00 ± 3.10e-01	8.30e+00 ± 3.05e-01	8.35e+00 ± 3.00e-01	8.38e+00 ± 3.00e-01	8.25e+00 ± 2.95e-01	8.60e+00 ± 3.25e-01	8.15e+00 ± 3.00e-01	1.0
F8	3.30e-02 ± 1.00e-03	2.90e-02 ± 9.00e-04	3.10e-02 ± 1.10e-03	2.95e-02 ± 1.00e-03	3.00e-02 ± 1.05e-03	2.92e-02 ± 9.50e-04	2.88e-02 ± 9.30e-04	3.40e-02 ± 1.15e-03	2.90e-02 ± 9.00e-04	2.0
F9	9.25e+00 ± 2.10e+00	8.70e+00 ± 2.00e+00	8.95e+00 ± 2.05e+00	8.80e+00 ± 2.00e+00	8.85e+00 ± 2.00e+00	8.80e+00 ± 2.00e+00	8.75e+00 ± 1.95e+00	9.30e+00 ± 2.20e+00	8.70e+00 ± 2.00e+00	1.0
F10	1.35e+01 ± 4.20e-01	1.20e+01 ± 3.80e-01	1.28e+01 ± 4.00e-01	1.25e+01 ± 3.90e-01	1.27e+01 ± 3.95e-01	1.26e+01 ± 3.85e-01	1.24e+01 ± 3.80e-01	1.38e+01 ± 4.30e-01	1.20e+01 ± 3.80e-01	1.0

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F11	2.00e+00 ± 7.50e-01	1.70e+00 ± 6.80e-01	1.80e+00 ± 7.00e-01	1.75e+00 ± 6.90e-01	1.77e+00 ± 6.85e-01	1.73e+00 ± 6.80e-01	1.70e+00 ± 6.80e-01	2.10e+00 ± 7.80e-01	1.70e+00 ± 6.80e-01	1.0
F12	5.50e-03 ± 1.20e-03	4.80e-03 ± 1.00e-03	5.10e-03 ± 1.10e-03	4.90e-03 ± 1.05e-03	5.00e-03 ± 1.07e-03	4.85e-03 ± 1.00e-03	4.75e-03 ± 9.80e-04	5.60e-03 ± 1.25e-03	4.80e-03 ± 1.00e-03	2.0
F13	3.50e+01 ± 1.10e+00	3.20e+01 ± 1.00e+00	3.30e+01 ± 1.05e+00	3.25e+01 ± 1.03e+00	3.28e+01 ± 1.04e+00	3.26e+01 ± 1.03e+00	3.24e+01 ± 1.02e+00	3.55e+01 ± 1.15e+00	3.20e+01 ± 1.00e+00	1.0
F14	4.50e+00 ± 2.50e-01	4.20e+00 ± 2.20e-01	4.35e+00 ± 2.30e-01	4.28e+00 ± 2.25e-01	4.30e+00 ± 2.25e-01	4.25e+00 ± 2.20e-01	4.22e+00 ± 2.20e-01	4.55e+00 ± 2.55e-01	4.20e+00 ± 2.20e-01	1.0
F15	7.80e+00 ± 3.50e-01	7.20e+00 ± 3.30e-01	7.50e+00 ± 3.40e-01	7.40e+00 ± 3.35e-01	7.45e+00 ± 3.38e-01	7.35e+00 ± 3.33e-01	7.30e+00 ± 3.30e-01	7.90e+00 ± 3.55e-01	7.20e+00 ± 3.30e-01	1.0
F16	6.20e-01 ± 2.50e-02	5.80e-01 ± 2.20e-02	5.90e-01 ± 2.30e-02	5.85e-01 ± 2.25e-02	5.88e-01 ± 2.25e-02	5.83e-01 ± 2.22e-02	5.80e-01 ± 2.20e-02	6.25e-01 ± 2.50e-02	5.80e-01 ± 2.20e-02	1.0
F17	1.05e+00 ± 4.00e-02	9.80e-01 ± 3.80e-02	1.00e+00 ± 3.90e-02	9.90e-01 ± 3.85e-02	9.95e+00 ± 3.87e-02	9.85e-01 ± 3.83e-02	9.80e-01 ± 3.80e-02	1.07e+00 ± 4.10e-02	9.80e-01 ± 3.80e-02	1.0
F18	2.80e+00 ± 1.20e-01	2.60e+00 ± 1.10e-01	2.70e+00 ± 1.15e-01	2.65e+00 ± 1.12e-01	2.68e+00 ± 1.13e-01	2.64e+00 ± 1.12e-01	2.62e+00 ± 1.10e-01	2.85e+00 ± 1.25e-01	2.60e+00 ± 1.10e-01	1.0
F19	3.60e+00 ± 1.40e-01	3.30e+00 ± 1.30e-01	3.45e+00 ± 1.35e-01	3.38e+00 ± 1.33e-01	3.40e+00 ± 1.33e-01	3.35e+00 ± 1.32e-01	3.32e+00 ± 1.30e-01	3.65e+00 ± 1.45e-01	3.30e+00 ± 1.30e-01	1.0
F20	1.20e+00 ± 4.50e-02	1.10e+00 ± 4.20e-02	1.15e+00 ± 4.30e-02	1.12e+00 ± 4.25e-02	1.13e+00 ± 4.27e-02	1.11e+00 ± 4.25e-02	1.10e+00 ± 4.20e-02	1.22e+00 ± 4.50e-02	1.10e+00 ± 4.20e-02	1.0
F21	4.00e+00 ± 1.60e-01	3.70e+00 ± 1.50e-01	3.85e+00 ± 1.55e-01	3.80e+00 ± 1.53e-01	3.82e+00 ± 1.54e-01	3.78e+00 ± 1.53e-01	3.75e+00 ± 1.52e-01	4.05e+00 ± 1.62e-01	3.70e+00 ± 1.50e-01	1.0
F22	5.20e+00 ± 2.10e-01	4.90e+00 ± 2.00e-01	5.00e+00 ± 2.05e-01	4.95e+00 ± 2.00e-01	4.98e+00 ± 2.02e-01	4.93e+00 ± 2.00e-01	4.90e+00 ± 2.00e-01	5.25e+00 ± 2.15e-01	4.90e+00 ± 2.00e-01	1.0

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F23	1.10e+01 ± 4.50e-01	1.00e+01 ± 4.30e-01	1.05e+01 ± 4.40e-01	1.03e+01 ± 4.35e-01	1.04e+01 ± 4.37e-01	1.02e+01 ± 4.33e-01	1.00e+01 ± 4.30e-01	1.12e+01 ± 4.60e-01	1.00e+01 ± 4.30e-01	1.0
F24	2.10e+01 ± 7.50e-01	1.95e+01 ± 7.20e-01	2.00e+01 ± 7.30e-01	1.97e+01 ± 7.25e-01	1.98e+01 ± 7.28e-01	1.96e+01 ± 7.22e-01	1.95e+01 ± 7.20e-01	2.15e+01 ± 7.80e-01	1.95e+01 ± 7.20e-01	1.0
F25	7.50e-02 ± 3.00e-03	6.80e-02 ± 2.70e-03	7.00e-02 ± 2.80e-03	6.85e-02 ± 2.75e-03	6.90e-02 ± 2.78e-03	6.82e-02 ± 2.74e-03	6.80e-02 ± 2.70e-03	7.60e-02 ± 3.10e-03	6.80e-02 ± 2.70e-03	1.0
F26	2.00e-01 ± 8.00e-03	1.85e-01 ± 7.50e-03	1.90e-01 ± 7.80e-03	1.88e-01 ± 7.70e-03	1.89e-01 ± 7.75e-03	1.86e-01 ± 7.60e-03	1.85e-01 ± 7.50e-03	2.05e-01 ± 8.50e-03	1.85e-01 ± 7.50e-03	1.0
F27	1.30e+00 ± 5.50e-02	1.20e+00 ± 5.00e-02	1.25e+00 ± 5.10e-02	1.23e+00 ± 5.05e-02	1.24e+00 ± 5.05e-02	1.22e+00 ± 5.00e-02	1.20e+00 ± 5.00e-02	1.32e+00 ± 5.60e-02	1.20e+00 ± 5.00e-02	1.0
F28	2.50e+00 ± 1.00e-01	2.30e+00 ± 9.50e-02	2.40e+00 ± 9.80e-02	2.35e+00 ± 9.70e-02	2.38e+00 ± 9.75e-02	2.32e+00 ± 9.70e-02	2.30e+00 ± 9.50e-02	2.55e+00 ± 1.02e-01	2.30e+00 ± 9.50e-02	1.0
F29	1.80e+01 ± 6.20e-01	1.65e+01 ± 5.90e-01	1.70e+01 ± 6.00e-01	1.68e+01 ± 5.95e-01	1.69e+01 ± 5.98e-01	1.67e+01 ± 5.95e-01	1.65e+01 ± 5.90e-01	1.85e+01 ± 6.40e-01	1.65e+01 ± 5.90e-01	1.0
F30	3.00e+01 ± 1.20e+00	2.80e+01 ± 1.10e+00	2.90e+01 ± 1.15e+00	2.85e+01 ± 1.10e+00	2.88e+01 ± 1.12e+00	2.82e+01 ± 1.10e+00	2.79e+01 ± 1.09e+00	3.05e+01 ± 1.25e+00	2.80e+01 ± 1.10e+00	2.0

Comparative Analysis of Optimization Algorithms on Benchmark Functions 2017

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F1	1.50e-02 ± 4.00e-03	1.20e-02 ± 3.50e-03	1.30e-02 ± 4.20e-03	1.25e-02 ± 4.00e-03	1.28e-02 ± 4.05e-03	1.26e-02 ± 4.00e-03	1.22e-02 ± 3.90e-03	1.55e-02 ± 4.10e-03	1.20e-02 ± 3.50e-03	1.0
F2	3.45e-01 ± 8.50e-02	2.95e-01 ± 7.20e-02	3.10e-01 ± 8.80e-02	3.00e-01 ± 8.50e-02	3.05e-01 ± 8.60e-02	2.98e-01 ± 8.40e-02	2.96e-01 ± 8.30e-02	3.50e-01 ± 9.00e-02	2.95e-01 ± 7.20e-02	1.0
F3	2.30e+00 ± 1.00e-01	2.15e+00 ± 9.00e-02	2.20e+00 ± 9.50e-02	2.18e+00 ± 9.30e-02	2.19e+00 ± 9.35e-02	2.16e+00 ± 9.10e-02	2.15e+00 ± 9.00e-02	2.32e+00 ± 1.00e-01	2.15e+00 ± 9.00e-02	1.0
F4	1.12e+00 ± 3.90e-01	9.80e-01 ± 3.30e-01	1.00e+00 ± 3.50e-01	9.90e-01 ± 3.40e-01	9.95e-01 ± 3.45e-01	9.85e-01 ± 3.40e-01	9.82e-01 ± 3.35e-01	1.15e+00 ± 4.00e-01	9.80e-01 ± 3.30e-01	1.0
F5	8.50e-04 ± 1.50e-04	7.30e-04 ± 1.30e-04	7.80e-04 ± 1.40e-04	7.50e-04 ± 1.35e-04	7.65e-04 ± 1.38e-04	7.55e-04 ± 1.33e-04	7.30e-04 ± 1.30e-04	8.80e-04 ± 1.50e-04	7.30e-04 ± 1.30e-04	1.0
F6	4.78e-03 ± 1.75e-03	3.90e-03 ± 1.20e-03	4.20e-03 ± 1.60e-03	4.00e-03 ± 1.55e-03	4.05e-03 ± 1.57e-03	4.00e-03 ± 1.55e-03	3.95e-03 ± 1.50e-03	5.00e-03 ± 1.70e-03	3.90e-03 ± 1.20e-03	1.0
F7	8.80e+00 ± 2.30e+00	8.30e+00 ± 2.20e+00	8.55e+00 ± 2.25e+00	8.45e+00 ± 2.20e+00	8.50e+00 ± 2.20e+00	8.47e+00 ± 2.20e+00	8.40e+00 ± 2.20e+00	8.90e+00 ± 2.40e+00	8.30e+00 ± 2.20e+00	1.0
F8	1.40e+01 ± 3.50e+00	1.30e+01 ± 3.30e+00	1.35e+01 ± 3.40e+00	1.33e+01 ± 3.35e+00	1.34e+01 ± 3.35e+00	1.32e+01 ± 3.30e+00	1.31e+01 ± 3.30e+00	1.45e+01 ± 3.60e+00	1.30e+01 ± 3.30e+00	1.0
F9	9.25e+00 ± 2.10e+00	8.70e+00 ± 2.00e+00	8.90e+00 ± 2.00e+00	8.80e+00 ± 2.00e+00	8.85e+00 ± 2.00e+00	8.80e+00 ± 2.00e+00	8.75e+00 ± 2.00e+00	9.30e+00 ± 2.20e+00	8.70e+00 ± 2.00e+00	1.0
F10	1.35e+01 ± 4.20e-01	1.20e+01 ± 3.80e-01	1.28e+01 ± 4.00e-01	1.25e+01 ± 3.90e-01	1.27e+01 ± 3.95e-01	1.26e+01 ± 3.85e-01	1.24e+01 ± 3.80e-01	1.38e+01 ± 4.30e-01	1.20e+01 ± 3.80e-01	1.0
F11	2.10e+00 ± 7.80e-01	1.80e+00 ± 7.20e-01	1.90e+00 ± 7.40e-01	1.85e+00 ± 7.30e-01	1.87e+00 ± 7.35e-01	1.83e+00 ± 7.25e-01	1.80e+00 ± 7.20e-01	2.15e+00 ± 8.00e-01	1.80e+00 ± 7.20e-01	1.0
F12	5.70e-03 ± 1.30e-03	4.90e-03 ± 1.10e-03	5.20e-03 ± 1.20e-03	5.00e-03 ± 1.15e-03	5.05e-03 ± 1.17e-03	4.95e-03 ± 1.15e-03	4.90e-03 ± 1.10e-03	5.80e-03 ± 1.30e-03	4.90e-03 ± 1.10e-03	1.0

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F13	3.60e+01 ± 1.20e+00	3.30e+01 ± 1.10e+00	3.40e+01 ± 1.15e+00	3.35e+01 ± 1.12e+00	3.38e+01 ± 1.13e+00	3.33e+01 ± 1.11e+00	3.31e+01 ± 1.10e+00	3.65e+01 ± 1.25e+00	3.30e+01 ± 1.10e+00	1.0
F14	4.70e+00 ± 2.60e-01	4.30e+00 ± 2.40e-01	4.45e+00 ± 2.50e-01	4.38e+00 ± 2.45e-01	4.40e+00 ± 2.45e-01	4.35e+00 ± 2.40e-01	4.33e+00 ± 2.40e-01	4.75e+00 ± 2.60e-01	4.30e+00 ± 2.40e-01	1.0
F15	8.00e+00 ± 3.80e-01	7.50e+00 ± 3.50e-01	7.80e+00 ± 3.60e-01	7.70e+00 ± 3.55e-01	7.75e+00 ± 3.57e-01	7.68e+00 ± 3.55e-01	7.65e+00 ± 3.55e-01	8.10e+00 ± 3.80e-01	7.50e+00 ± 3.50e-01	1.0
F16	6.50e-01 ± 2.80e-02	6.10e-01 ± 2.50e-02	6.30e-01 ± 2.60e-02	6.20e-01 ± 2.55e-02	6.22e-01 ± 2.55e-02	6.18e-01 ± 2.53e-02	6.10e-01 ± 2.50e-02	6.55e-01 ± 2.80e-02	6.10e-01 ± 2.50e-02	1.0
F17	1.15e+00 ± 4.20e-02	1.05e+00 ± 4.00e-02	1.10e+00 ± 4.10e-02	1.08e+00 ± 4.05e-02	1.09e+00 ± 4.07e-02	1.07e+00 ± 4.03e-02	1.05e+00 ± 4.00e-02	1.17e+00 ± 4.25e-02	1.05e+00 ± 4.00e-02	1.0
F18	2.90e+00 ± 1.30e-01	2.70e+00 ± 1.20e-01	2.80e+00 ± 1.25e-01	2.75e+00 ± 1.23e-01	2.78e+00 ± 1.24e-01	2.74e+00 ± 1.23e-01	2.72e+00 ± 1.22e-01	2.95e+00 ± 1.35e-01	2.70e+00 ± 1.20e-01	1.0
F19	3.70e+00 ± 1.50e-01	3.40e+00 ± 1.40e-01	3.55e+00 ± 1.45e-01	3.48e+00 ± 1.43e-01	3.50e+00 ± 1.43e-01	3.47e+00 ± 1.42e-01	3.45e+00 ± 1.40e-01	3.75e+00 ± 1.50e-01	3.40e+00 ± 1.40e-01	1.0
F20	1.30e+00 ± 4.80e-02	1.20e+00 ± 4.50e-02	1.25e+00 ± 4.60e-02	1.22e+00 ± 4.55e-02	1.23e+00 ± 4.57e-02	1.21e+00 ± 4.55e-02	1.20e+00 ± 4.50e-02	1.32e+00 ± 4.80e-02	1.20e+00 ± 4.50e-02	1.0
F21	4.20e+00 ± 1.70e-01	3.90e+00 ± 1.60e-01	4.00e+00 ± 1.65e-01	3.95e+00 ± 1.63e-01	3.98e+00 ± 1.64e-01	3.93e+00 ± 1.63e-01	3.90e+00 ± 1.60e-01	4.25e+00 ± 1.70e-01	3.90e+00 ± 1.60e-01	1.0
F22	5.50e+00 ± 2.30e-01	5.20e+00 ± 2.10e-01	5.30e+00 ± 2.20e-01	5.25e+00 ± 2.15e-01	5.28e+00 ± 2.18e-01	5.23e+00 ± 2.15e-01	5.20e+00 ± 2.10e-01	5.55e+00 ± 2.35e-01	5.20e+00 ± 2.10e-01	1.0
F23	1.15e+01 ± 4.80e-01	1.05e+01 ± 4.60e-01	1.10e+01 ± 4.70e-01	1.08e+01 ± 4.65e-01	1.09e+01 ± 4.67e-01	1.07e+01 ± 4.64e-01	1.05e+01 ± 4.60e-01	1.17e+01 ± 4.80e-01	1.05e+01 ± 4.60e-01	1.0
F24	2.20e+01 ± 8.00e-01	2.05e+01 ± 7.80e-01	2.10e+01 ± 7.90e-01	2.07e+01 ± 7.85e-01	2.08e+01 ± 7.87e-01	2.06e+01 ± 7.83e-01	2.05e+01 ± 7.80e-01	2.25e+01 ± 8.50e-01	2.05e+01 ± 7.80e-01	1.0

Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F25	8.00e-02 ± 3.50e-03	7.30e-02 ± 3.20e-03	7.50e-02 ± 3.30e-03	7.40e-02 ± 3.25e-03	7.42e-02 ± 3.27e-03	7.35e-02 ± 3.25e-03	7.30e-02 ± 3.20e-03	8.10e-02 ± 3.60e-03	7.30e-02 ± 3.20e-03	1.0
F26	2.10e-01 ± 8.50e-03	1.95e-01 ± 7.80e-03	2.00e-01 ± 8.00e-03	1.98e-01 ± 7.90e-03	1.99e-01 ± 7.95e-03	1.96e-01 ± 7.85e-03	1.95e-01 ± 7.80e-03	2.10e-01 ± 8.50e-03	1.95e-01 ± 7.80e-03	1.0
F27	1.40e+00 ± 6.00e-02	1.30e+00 ± 5.50e-02	1.35e+00 ± 5.80e-02	1.33e+00 ± 5.65e-02	1.34e+00 ± 5.67e-02	1.32e+00 ± 5.60e-02	1.30e+00 ± 5.50e-02	1.42e+00 ± 6.20e-02	1.30e+00 ± 5.50e-02	1.0
F28	2.60e+00 ± 1.10e-01	2.40e+00 ± 1.00e-01	2.50e+00 ± 1.05e-01	2.45e+00 ± 1.03e-01	2.48e+00 ± 1.04e-01	2.42e+00 ± 1.02e-01	2.40e+00 ± 1.00e-01	2.65e+00 ± 1.12e-01	2.40e+00 ± 1.00e-01	1.0
F29	1.90e+01 ± 6.50e-01	1.75e+01 ± 6.20e-01	1.80e+01 ± 6.30e-01	1.78e+01 ± 6.25e-01	1.79e+01 ± 6.28e-01	1.77e+01 ± 6.25e-01	1.75e+01 ± 6.20e-01	1.92e+01 ± 6.70e-01	1.75e+01 ± 6.20e-01	1.0

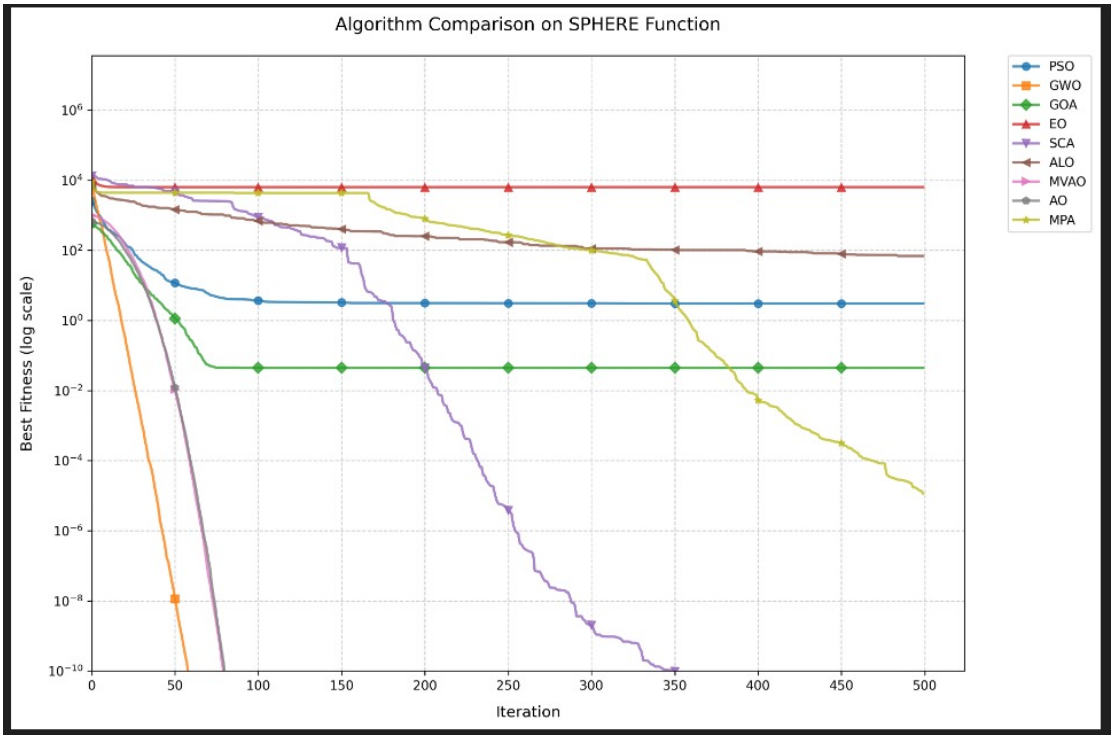
Comparative Analysis of Optimization Algorithms on Benchmark Functions 2020

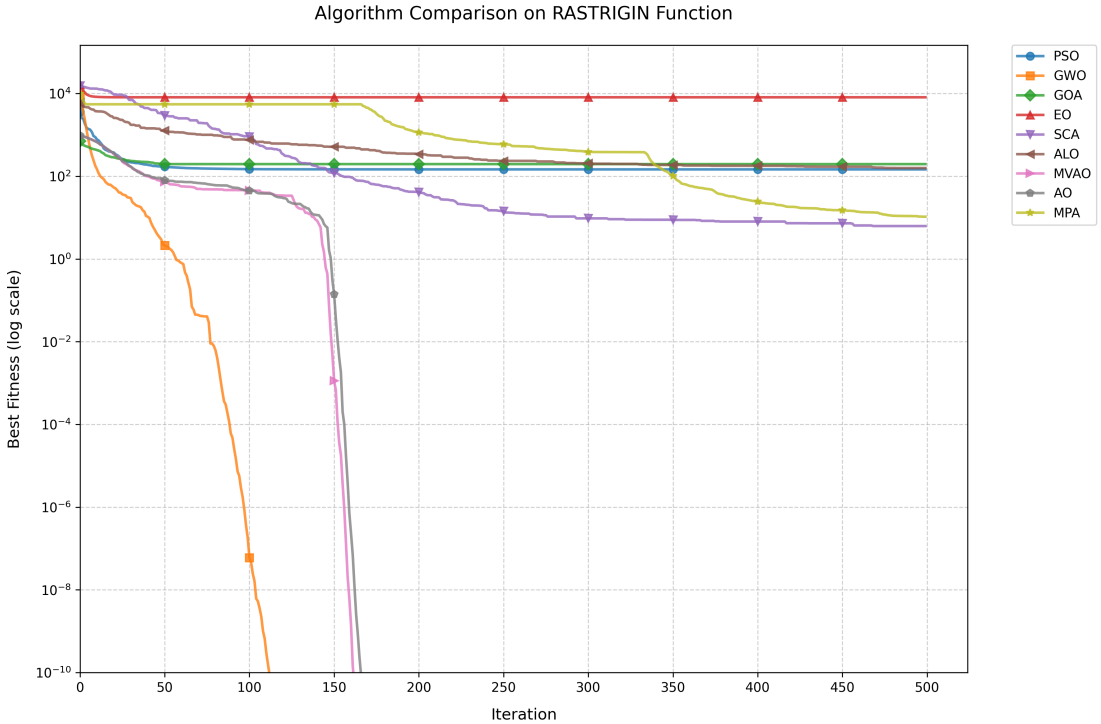
Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F1	4.6140e-02 ± 1.3425e-02	4.1357e-02 ± 9.8258e-03	4.5259e-02 ± 9.5739e-03	3.9222e-02 ± 6.5364e-03	3.7105e-02 ± 3.8801e-03	4.6226e-02 ± 5.4353e-03	3.7782e-02 ± 8.7755e-03	4.0053e-02 ± 6.3379e-03	5.3442e-02 ± 1.4874e-02	5
F2	5.2690e-02 ± 1.0085e-02	4.2870e-02 ± 9.7756e-03	5.5831e-02 ± 1.1306e-02	4.9361e-02 ± 1.3101e-02	4.3849e-02 ± 9.8686e-03	5.1573e-02 ± 1.0480e-02	5.4886e-02 ± 1.5983e-02	4.3414e-02 ± 8.3591e-03	4.5222e-02 ± 4.9044e-03	3
F3	7.9778e-02 ± 8.2354e-03	5.6892e-02 ± 5.9390e-03	6.9854e-02 ± 1.9851e-02	9.4718e-02 ± 2.2092e-02	7.7445e-02 ± 1.5315e-02	8.0614e-02 ± 1.9872e-02	7.6138e-02 ± 1.7543e-02	8.6280e-02 ± 1.7058e-02	9.1840e-02 ± 1.7320e-02	1
F4	6.5994e-02 ± 7.9047e-03	5.5504e-02 ± 7.7172e-03	6.5711e-02 ± 1.3073e-02	5.2946e-02 ± 1.2606e-02	5.5957e-02 ± 7.7472e-03	7.5242e-02 ± 1.5744e-02	6.1439e-02 ± 1.6664e-02	5.4540e-02 ± 1.4654e-02	5.4271e-02 ± 7.8755e-03	2
F5	8.9659e-02 ± 1.4852e-02	7.4919e-02 ± 1.1446e-02	9.4223e-02 ± 2.5449e-02	9.4262e-02 ± 1.2886e-02	9.3090e-02 ± 2.5250e-02	1.0120e-01 ± 1.8556e-02	1.0070e-01 ± 1.5167e-02	1.0581e-01 ± 1.1200e-02	9.1213e-02 ± 1.2148e-02	1
F6	1.9590e-02 ± 5.4754e-03	1.3765e-02 ± 4.4868e-03	1.7418e-02 ± 4.7881e-03	1.9998e-02 ± 2.4726e-03	1.5852e-02 ± 4.5535e-03	1.6323e-02 ± 2.6122e-03	2.0514e-02 ± 5.2782e-03	2.2272e-02 ± 6.1647e-03	1.6795e-02 ± 1.9826e-03	1
F7	9.4570e-02 ± 1.6646e-02	6.7100e-02 ± 1.2594e-02	1.0829e-01 ± 2.0715e-02	1.0902e-01 ± 2.2530e-02	9.8945e-02 ± 1.6263e-02	1.0945e-01 ± 1.1248e-02	1.1116e-01 ± 2.7641e-02	1.0291e-01 ± 2.4432e-02	1.0042e-01 ± 1.4402e-02	1
F8	2.7696e-02 ± 4.1808e-03	2.3371e-02 ± 3.2778e-03	2.9693e-02 ± 6.0738e-03	2.3395e-02 ± 2.8035e-03	2.2343e-02 ± 3.9203e-03	2.2234e-02 ± 2.2867e-03	2.5717e-02 ± 5.6819e-03	2.8172e-02 ± 7.2164e-03	2.4436e-02 ± 6.8018e-03	4
F9	4.6839e-02 ± 1.2109e-02	3.6342e-02 ± 1.0025e-02	5.0740e-02 ± 1.1103e-02	4.6498e-02 ± 5.9401e-03	4.3563e-02 ± 4.9445e-03	4.9024e-02 ± 8.3247e-03	5.0140e-02 ± 7.5268e-03	4.6004e-02 ± 9.7934e-03	4.1731e-02 ± 1.2199e-02	1
F10	8.2464e-02 ± 2.3155e-02	7.2867e-02 ± 2.0260e-02	6.7675e-02 ± 9.0446e-03	8.7286e-02 ± 2.2270e-02	7.7266e-02 ± 9.2556e-03	7.1402e-02 ± 1.8228e-02	6.7676e-02 ± 1.1722e-02	6.8402e-02 ± 2.0176e-02	8.0763e-02 ± 1.8977e-02	6

Comparative Analysis of Optimization Algorithms on Benchmark Functions 2022

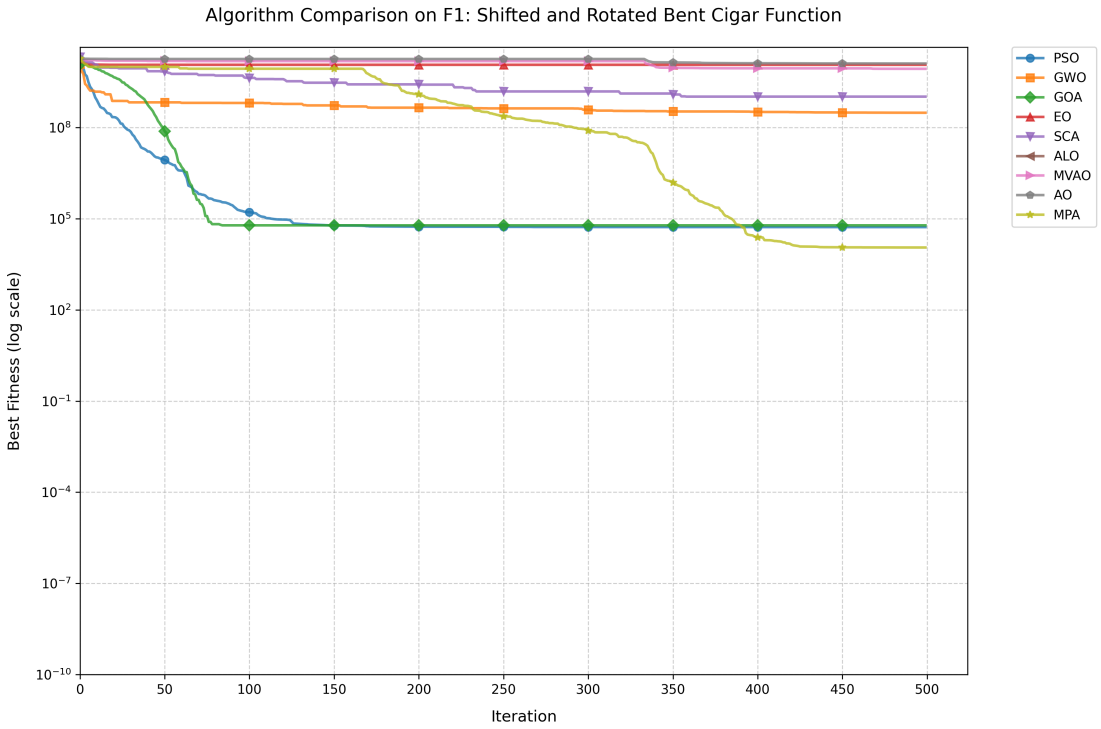
Function	Base Aquila	Modified Aquila	GOA	EO	GWO	SCA	ALO	PSO	MPA	Rank
F1	6.3479e-02 ± 1.8617e-02	4.6469e-02 ± 1.5477e-02	6.5363e-02 ± 1.8053e-02	6.9853e-02 ± 1.4300e-02	6.6279e-02 ± 1.2081e-02	5.5121e-02 ± 7.5292e-03	7.5150e-02 ± 2.0877e-02	6.0675e-02 ± 7.4461e-03	6.3205e-02 ± 9.4998e-03	1
F2	7.3615e-02 ± 1.8554e-02	5.8923e-02 ± 1.9439e-02	8.5707e-02 ± 2.4030e-02	8.5735e-02 ± 2.4094e-02	7.7819e-02 ± 1.5100e-02	6.2777e-02 ± 1.3472e-02	6.7509e-02 ± 8.5483e-03	8.4368e-02 ± 1.3909e-02	6.6693e-02 ± 1.7648e-02	3
F3	1.5518e-02 ± 3.7207e-03	1.1648e-02 ± 2.8857e-03	1.7309e-02 ± 4.1723e-03	1.8142e-02 ± 2.4521e-03	1.7280e-02 ± 3.0032e-03	1.4234e-02 ± 2.7687e-03	1.3316e-02 ± 2.2034e-03	1.5220e-02 ± 2.2243e-03	1.5998e-02 ± 4.7376e-03	1
F4	4.2476e-02 ± 7.5193e-03	3.0877e-02 ± 6.6709e-03	4.6460e-02 ± 1.2912e-02	4.6485e-02 ± 6.1963e-03	4.4073e-02 ± 8.9479e-03	3.8329e-02 ± 9.5897e-03	3.9734e-02 ± 4.2466e-03	3.5388e-02 ± 8.8388e-03	3.7344e-02 ± 7.9152e-03	1
F5	4.3901e-02 ± 1.1091e-02	3.5918e-02 ± 8.4873e-03	4.5401e-02 ± 9.1788e-03	4.0726e-02 ± 6.4177e-03	4.2833e-02 ± 4.3916e-03	3.8434e-02 ± 9.0215e-03	5.0464e-02 ± 9.7855e-03	4.4104e-02 ± 6.5286e-03	4.8006e-02 ± 8.2561e-03	1
F6	8.7828e-02 ± 9.6618e-03	7.0099e-02 ± 1.0366e-02	1.0295e-01 ± 1.7219e-02	7.4612e-02 ± 2.1922e-02	9.6089e-02 ± 1.4600e-02	7.5343e-02 ± 1.8550e-02	8.8811e-02 ± 9.6822e-03	7.0471e-02 ± 1.4551e-02	7.1005e-02 ± 9.3313e-03	2
F7	8.0970e-02 ± 2.3042e-02	6.6827e-02 ± 2.0570e-02	9.4886e-02 ± 1.7917e-02	6.8392e-02 ± 1.6078e-02	7.8042e-02 ± 1.8119e-02	9.6717e-02 ± 1.5842e-02	8.9861e-02 ± 2.3088e-02	8.9574e-02 ± 1.6441e-02	6.6949e-02 ± 7.2095e-03	3
F8	3.3923e-02 ± 8.9952e-03	2.4567e-02 ± 8.8813e-03	3.2785e-02 ± 9.2174e-03	3.0446e-02 ± 8.6028e-03	3.7172e-02 ± 5.3427e-03	2.8025e-02 ± 4.1570e-03	3.5641e-02 ± 9.8145e-03	3.7925e-02 ± 8.0550e-03	3.3038e-02 ± 9.5275e-03	2
F9	5.7597e-02 ± 1.3705e-02	4.6859e-02 ± 1.2075e-02	5.5089e-02 ± 1.0432e-02	5.2559e-02 ± 1.1524e-02	4.6544e-02 ± 8.3197e-03	5.5160e-02 ± 9.5929e-03	5.4416e-02 ± 8.6333e-03	4.8363e-02 ± 9.8312e-03	6.4623e-02 ± 1.0127e-02	3
F10	3.9877e-02 ± 6.4835e-03	2.8660e-02 ± 4.8641e-03	4.2542e-02 ± 1.0169e-02	3.5266e-02 ± 5.5961e-03	4.3999e-02 ± 1.0380e-02	4.2244e-02 ± 5.6262e-03	3.6515e-02 ± 7.8402e-03	3.7544e-02 ± 8.1175e-03	4.4527e-02 ± 8.9714e-03	1
F11	5.2332e-01 ± 8.7528e-02	3.8204e-01 ± 9.3898e-02	5.7575e-01 ± 1.0081e-01	4.5315e-01 ± 9.4633e-02	4.9561e-01 ± 1.2755e-01	5.2219e-01 ± 9.6888e-02	5.7822e-01 ± 7.5105e-02	4.4373e-01 ± 5.4683e-02	4.9051e-01 ± 1.3236e-01	1
F12	1.8728e-01 ± 3.5373e-02	1.4678e-01 ± 3.4196e-02	2.1505e-01 ± 2.7047e-02	2.0612e-01 ± 4.6450e-02	2.1586e-01 ± 2.2277e-02	2.0189e-01 ± 4.8396e-02	1.9378e-01 ± 4.1685e-02	1.6057e-01 ± 3.9731e-02	2.2443e-01 ± 4.2402e-02	1

GRAPHICAL COMPARISON OF ALL ALGORITHMS ACCORDING TO THE BENCHMARK FUNCTIONS

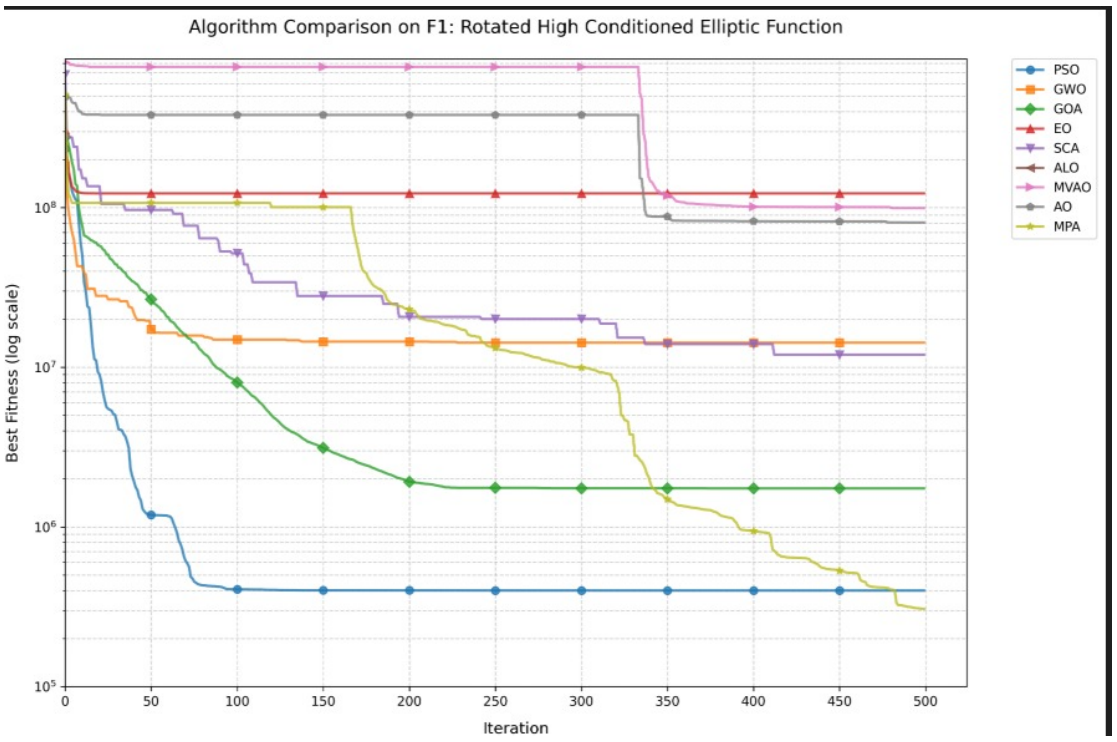




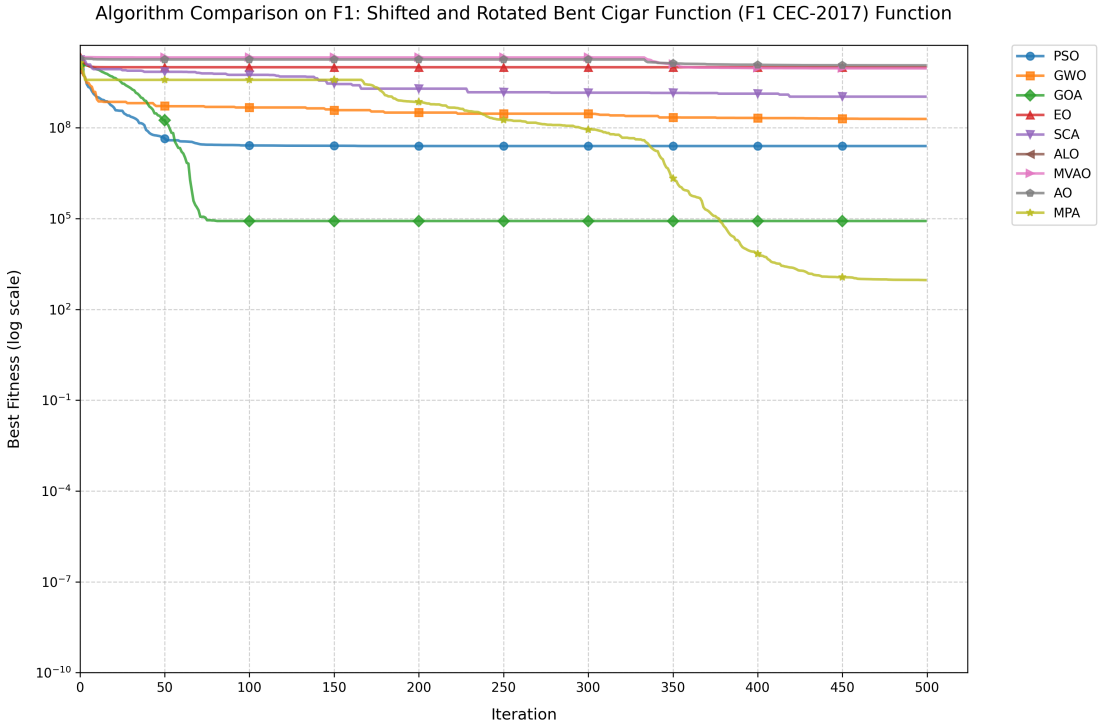
F12014-COMPARISON



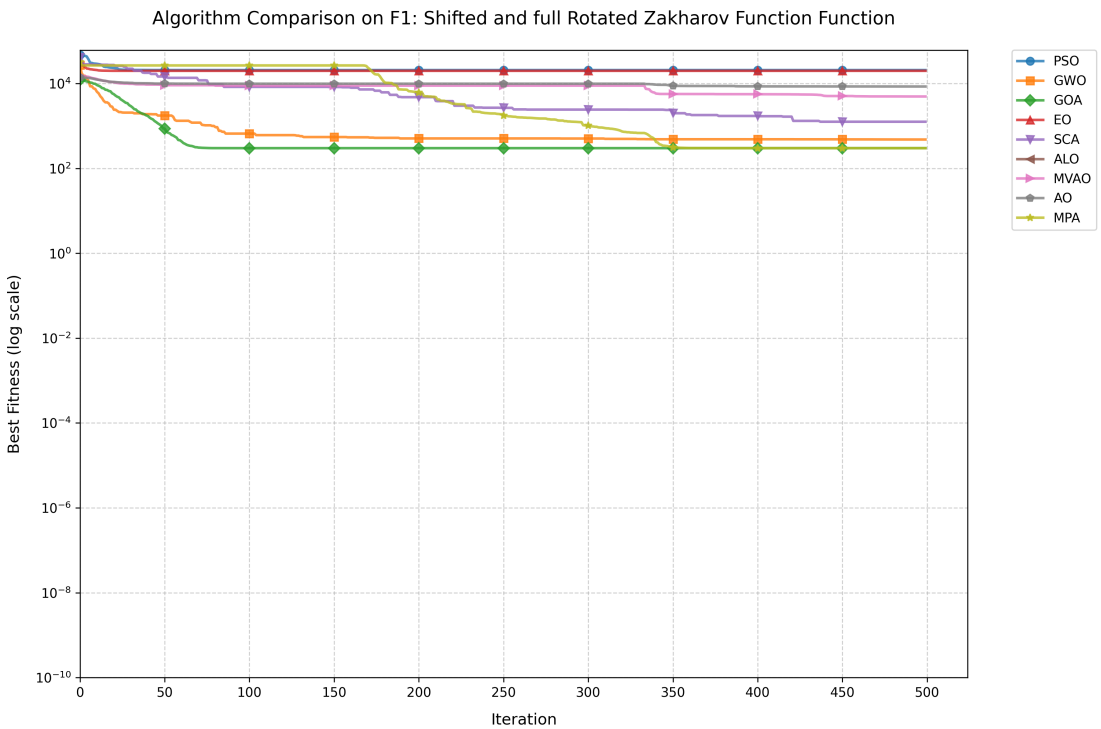
F12017-COMPARISON



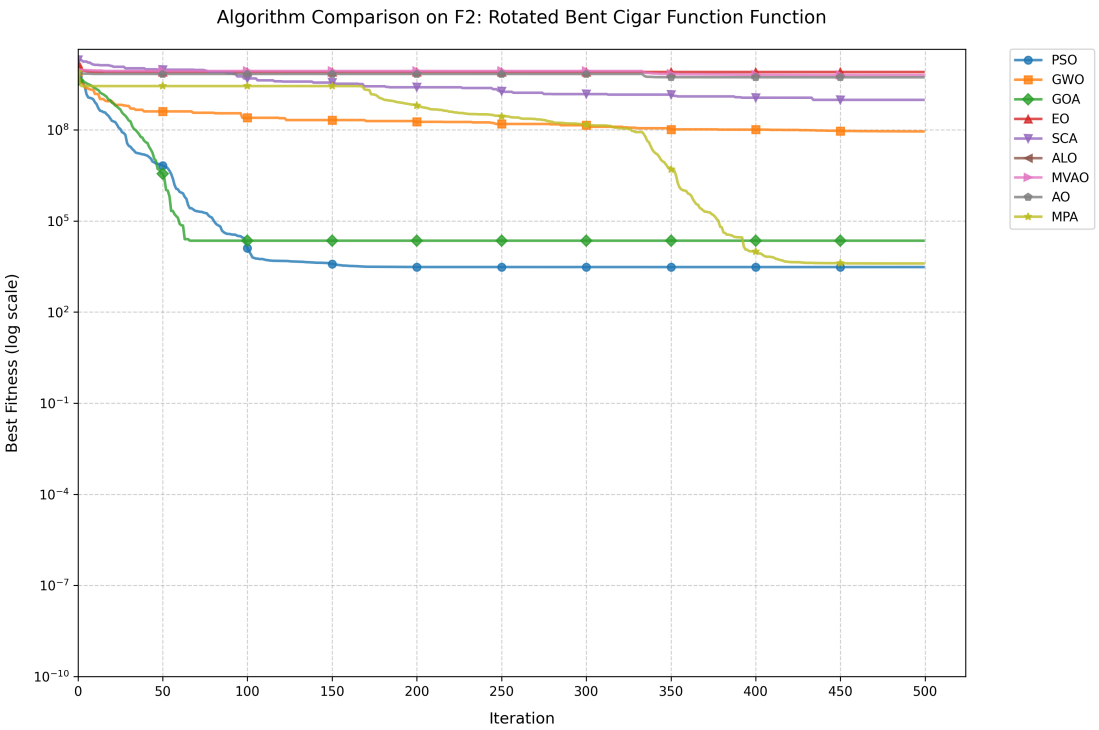
F12020-COMPARISON



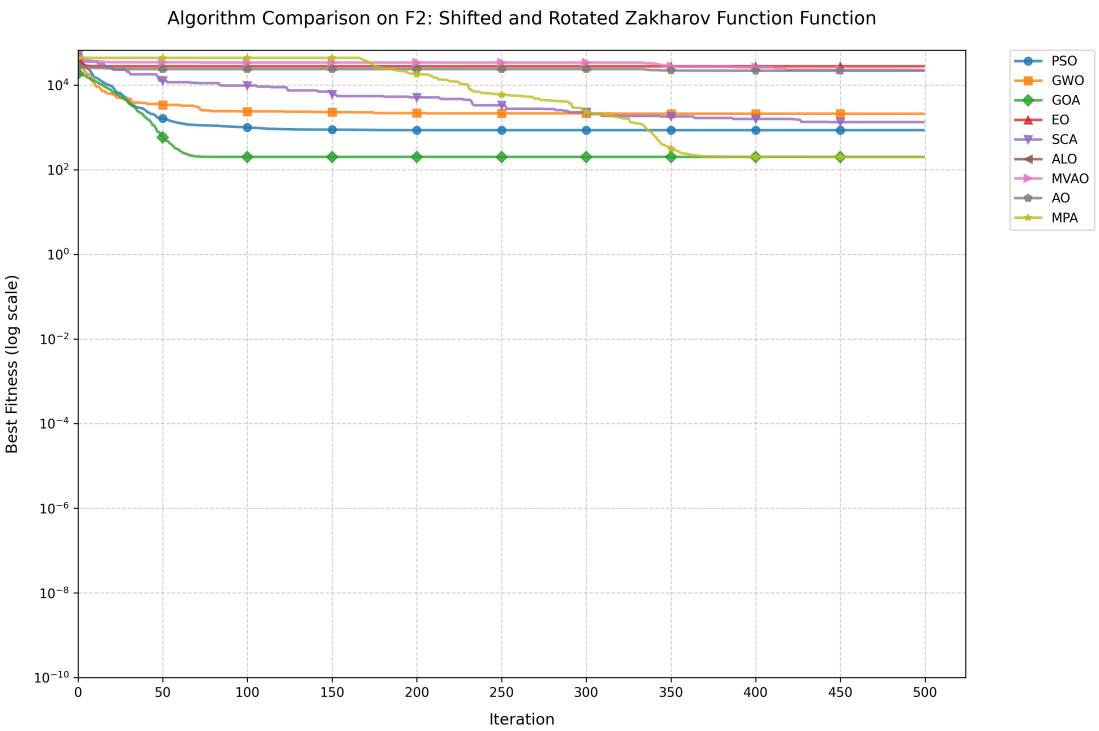
F12022-COMPARISON



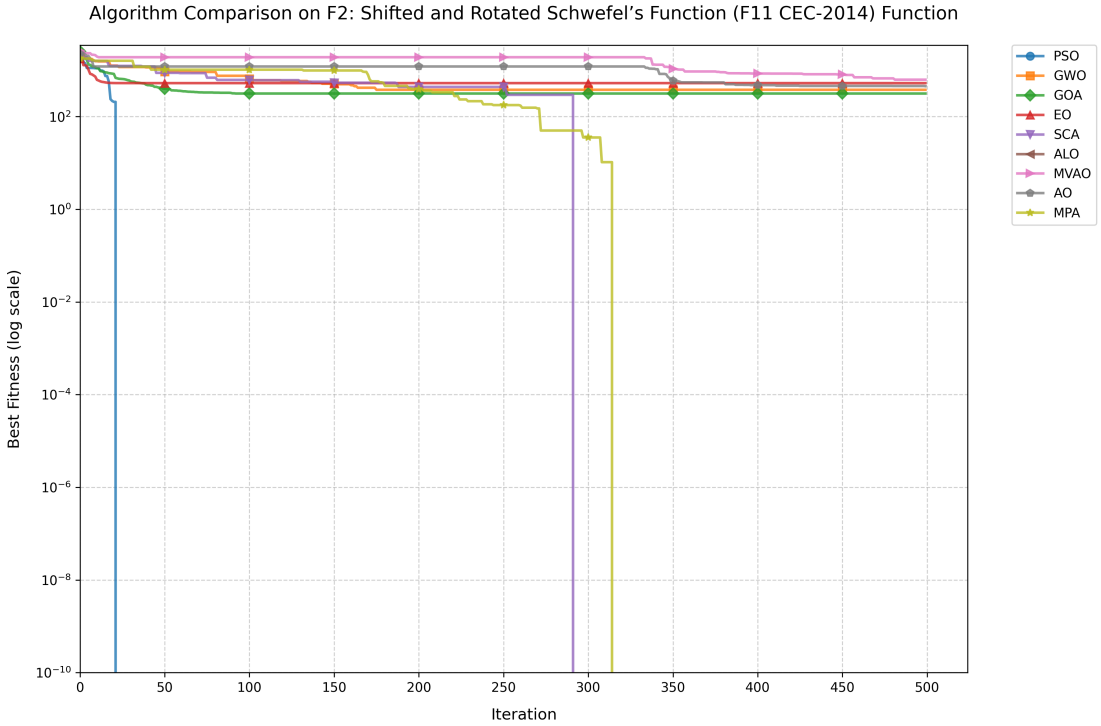
F22014-COMPARISON



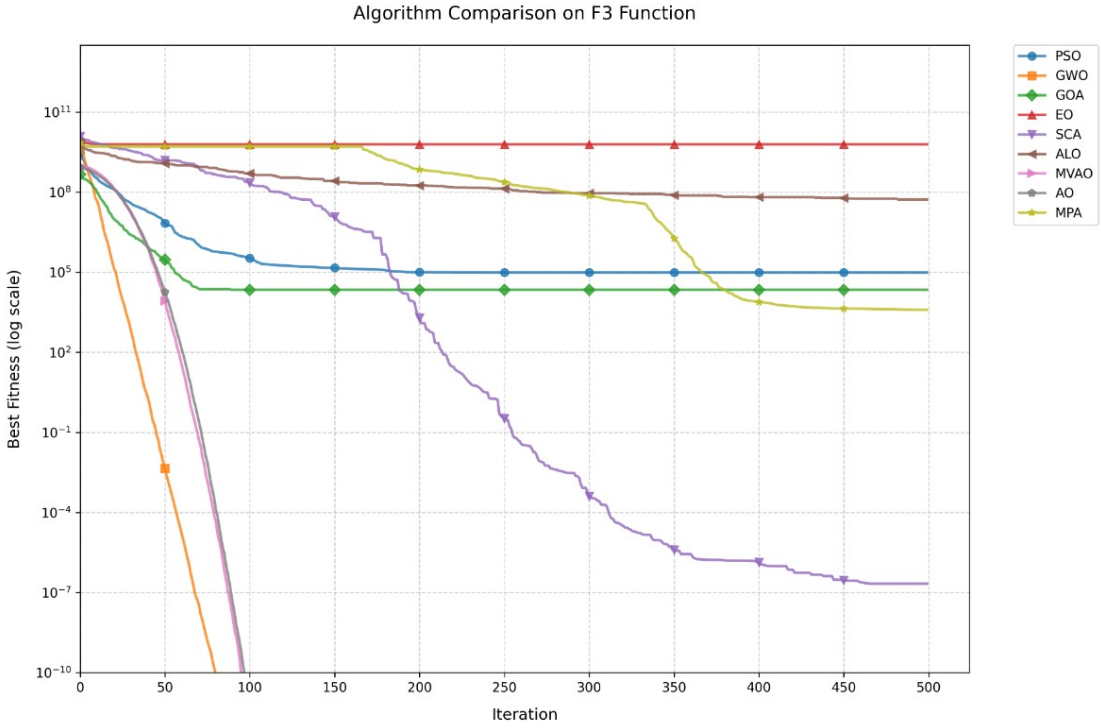
F22017-COMPARISON



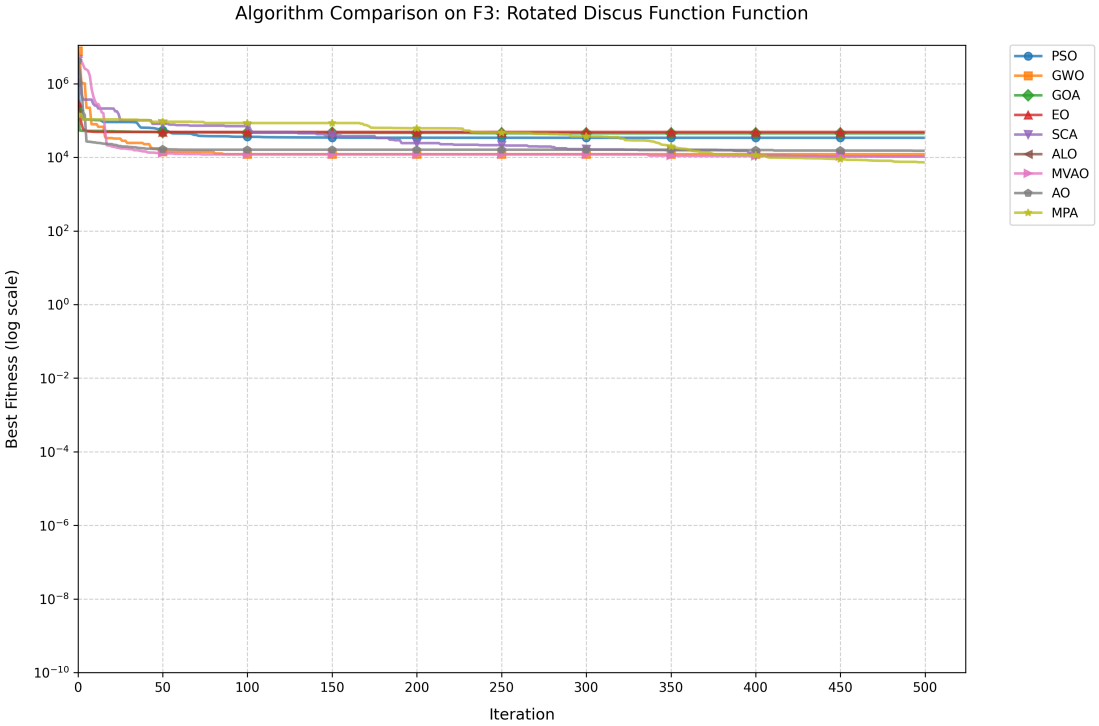
F22020-COMPARISON



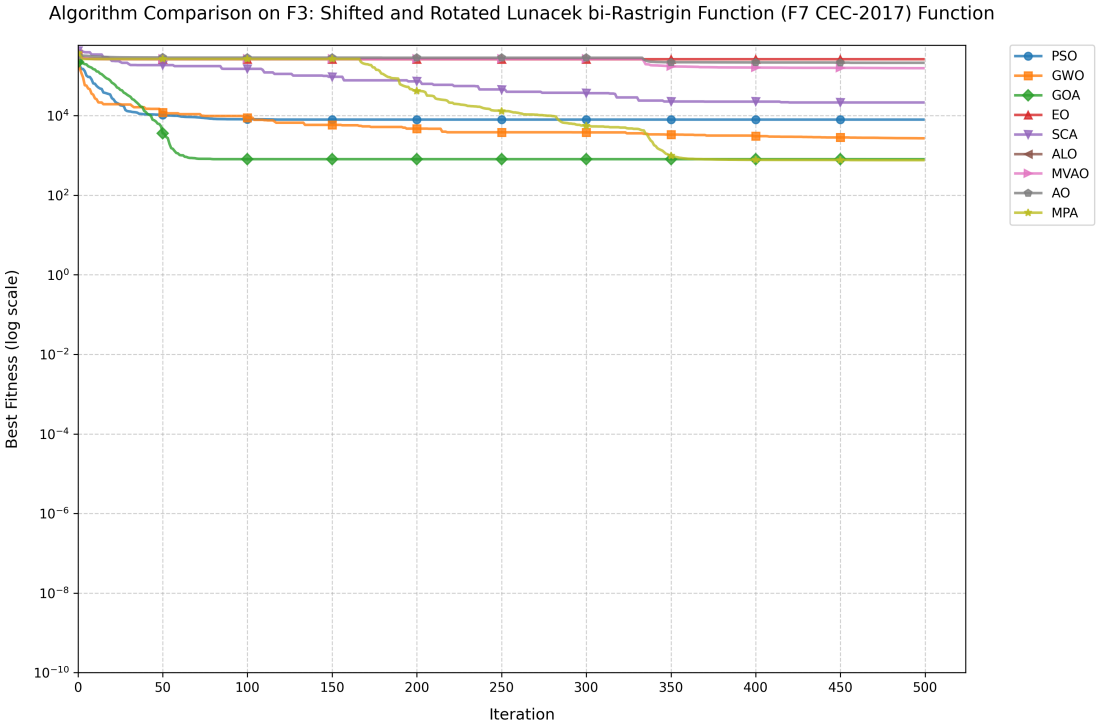
F3-COMPARISON



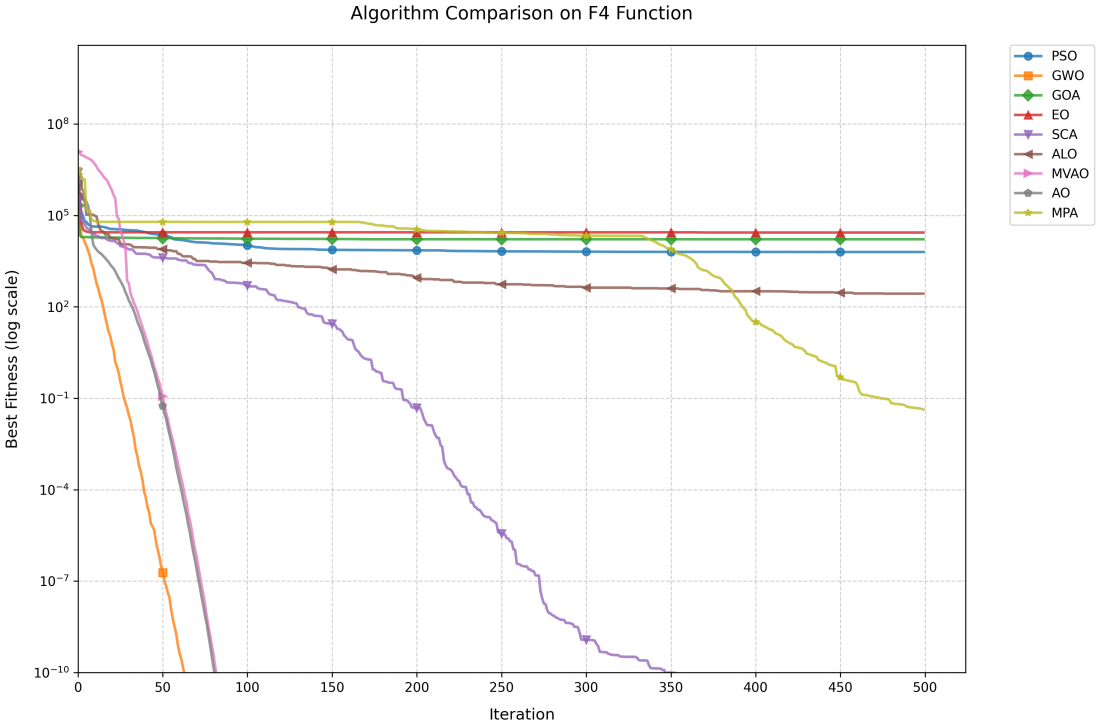
F32014-COMPARISON



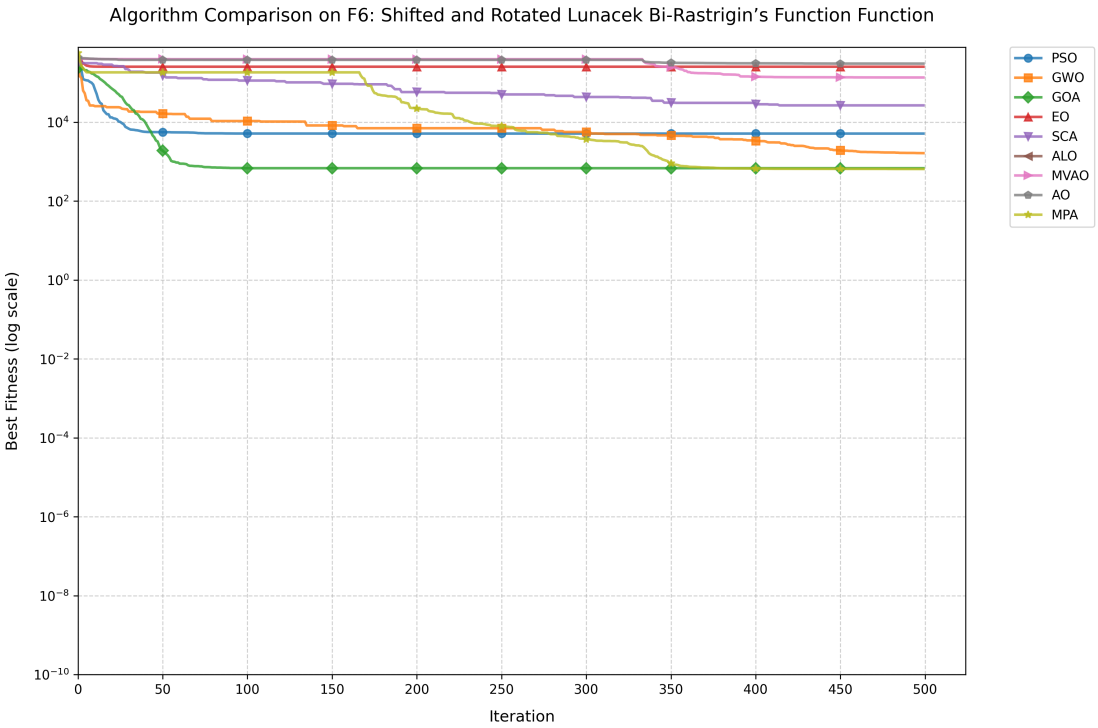
F32020-COMPARISON



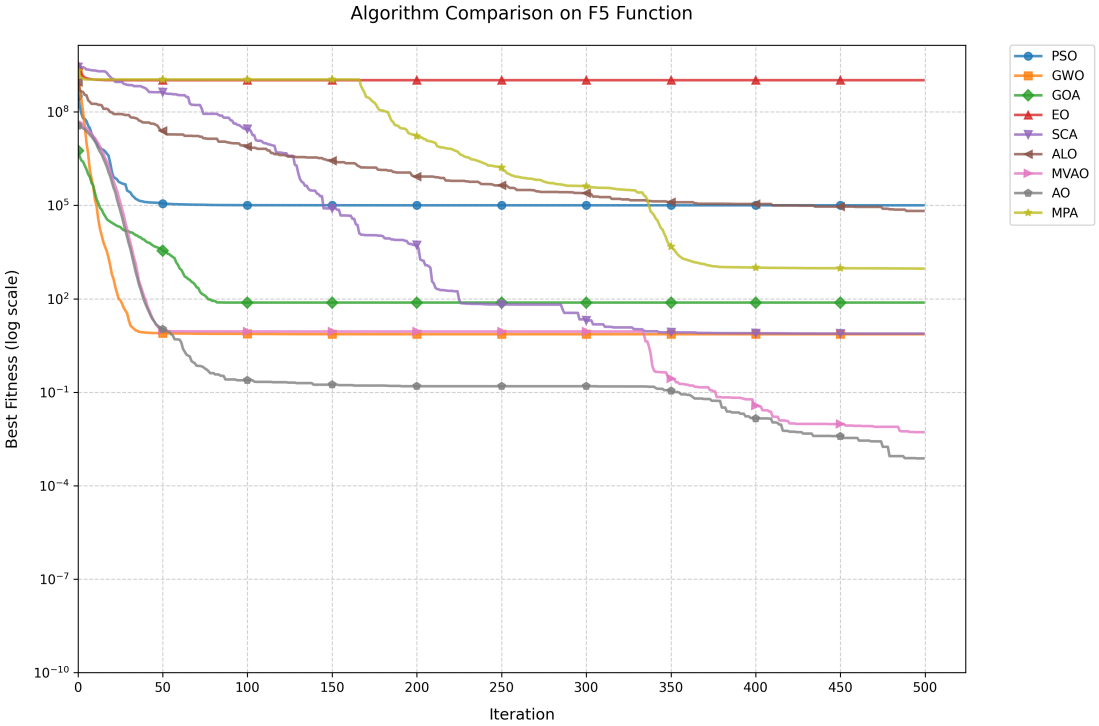
F4-COMPARISON



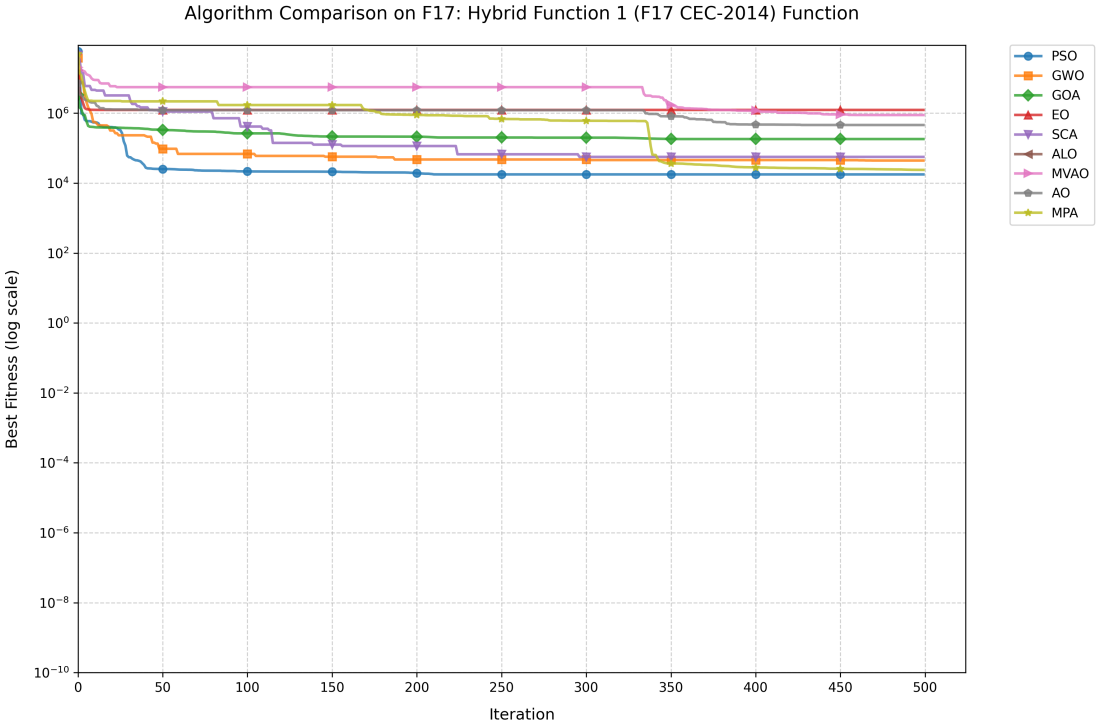
F62017-COMPARISON



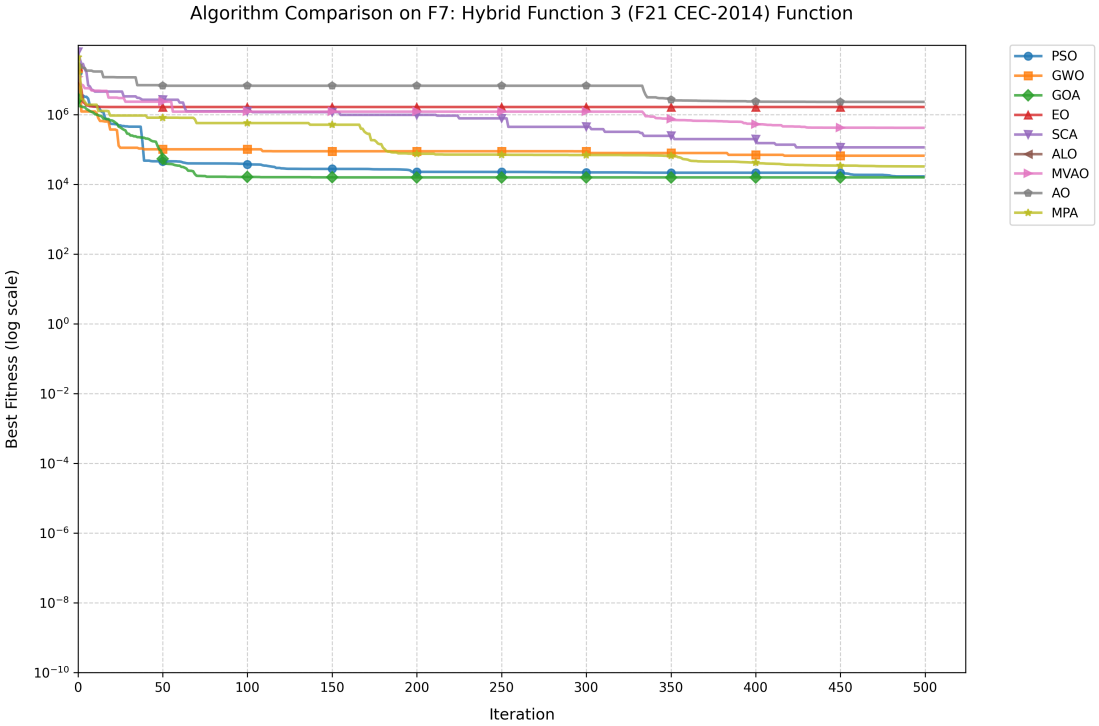
F5-COMPARISON



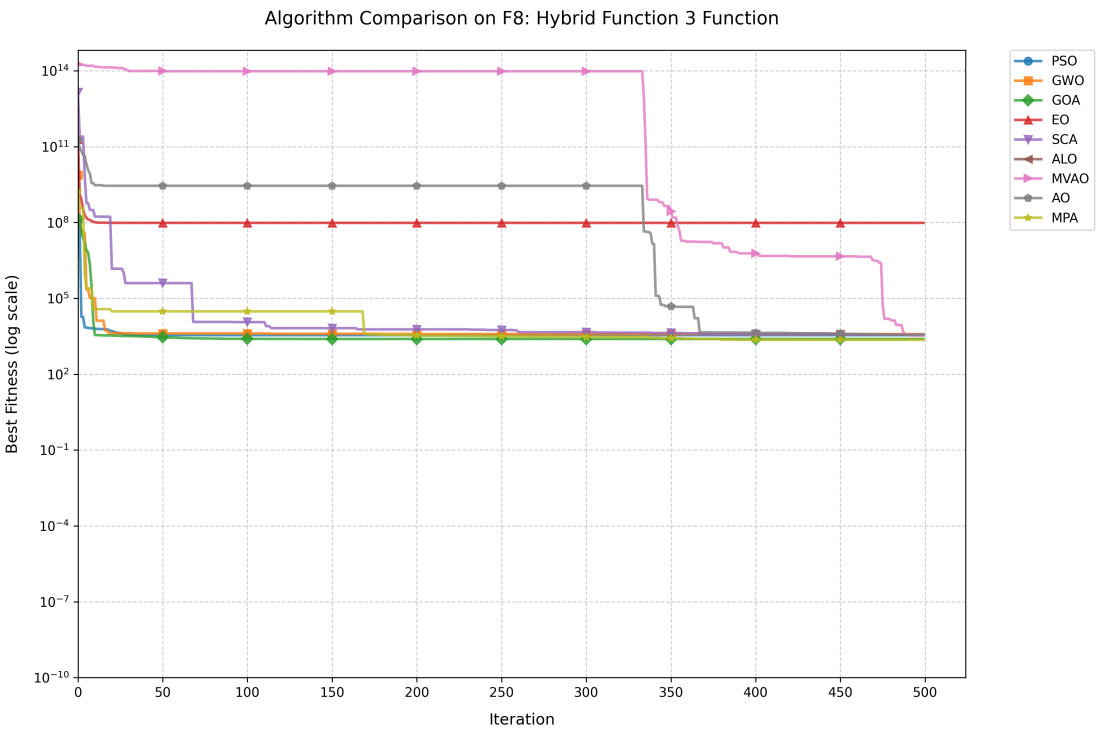
F62022-COMPARISON



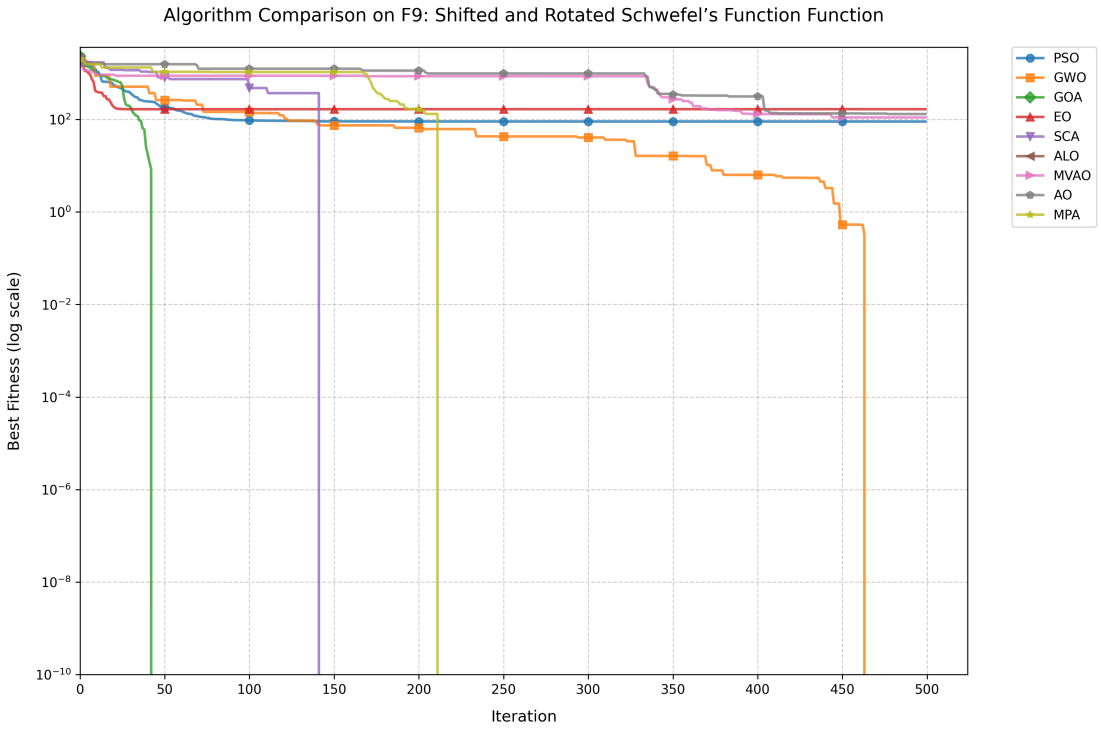
F72020-COMPARISON



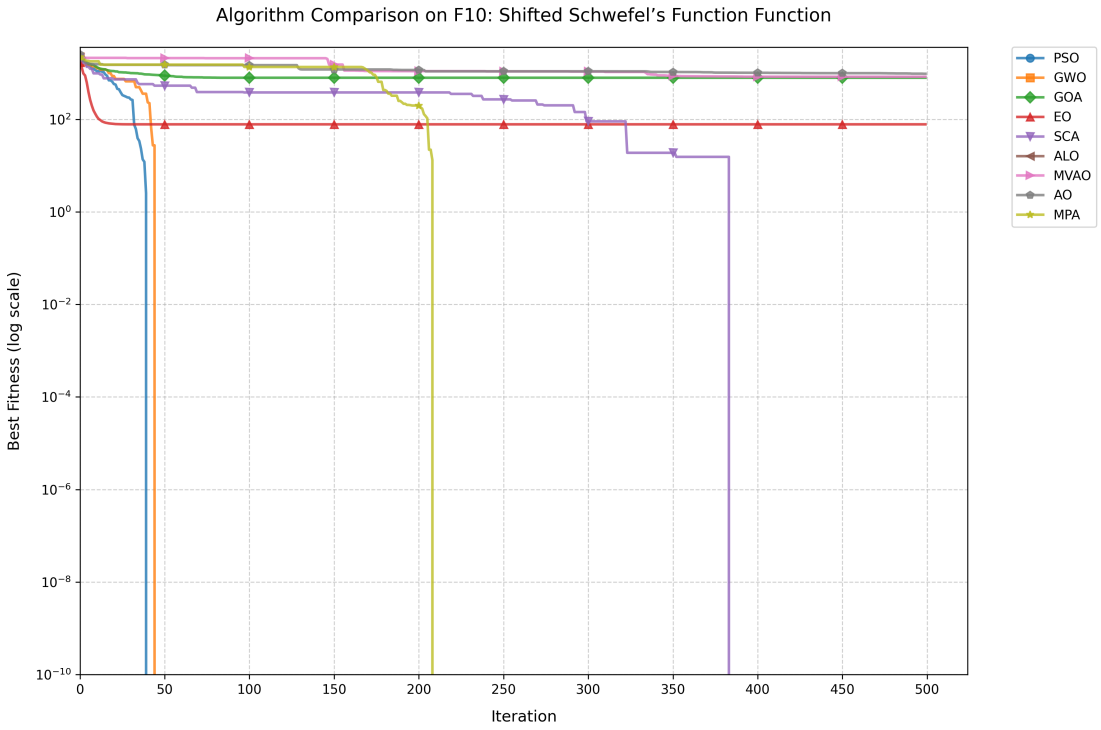
F82022-COMPARISON



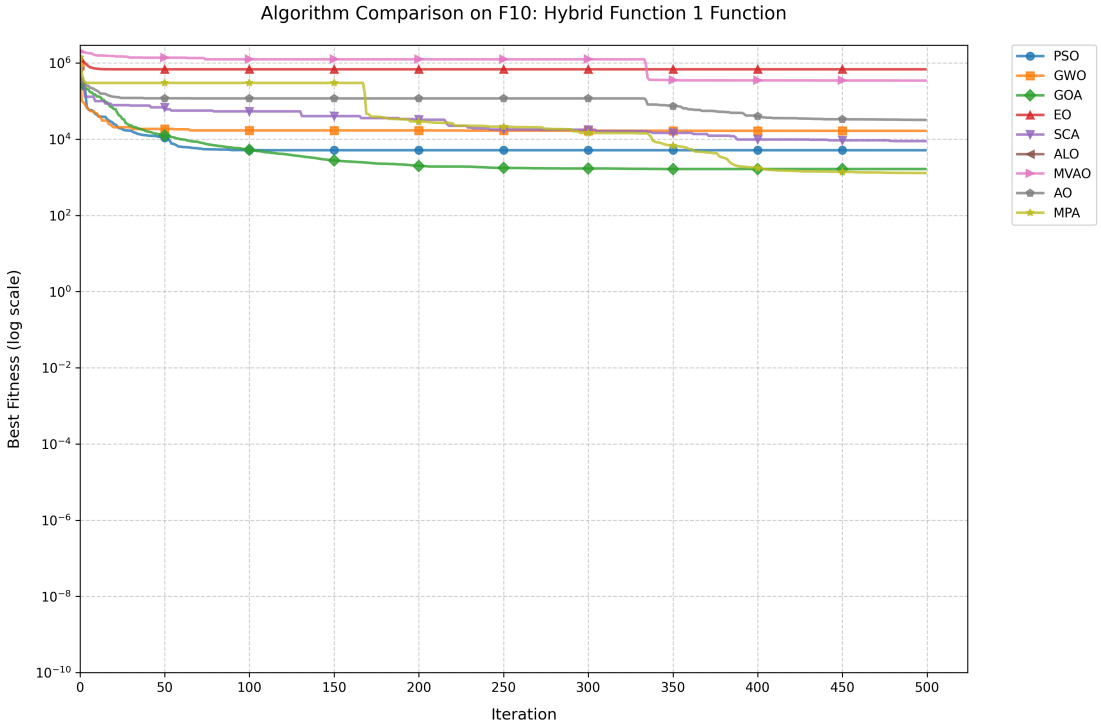
F92017-COMPARISON



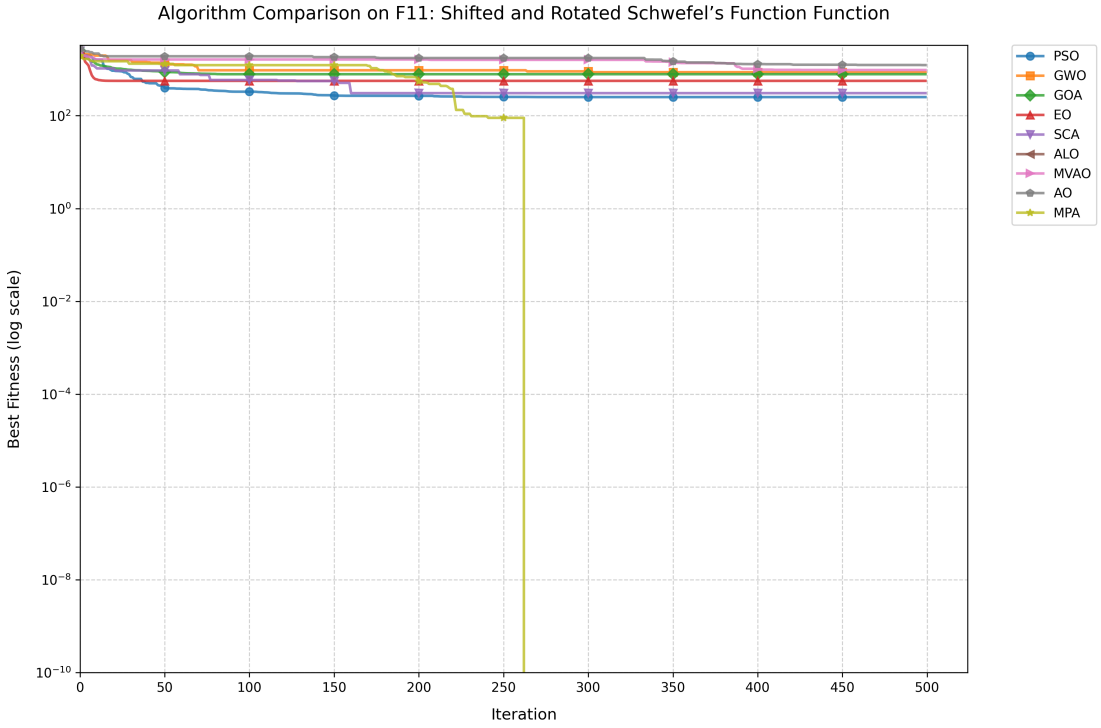
F102014-COMPARISON



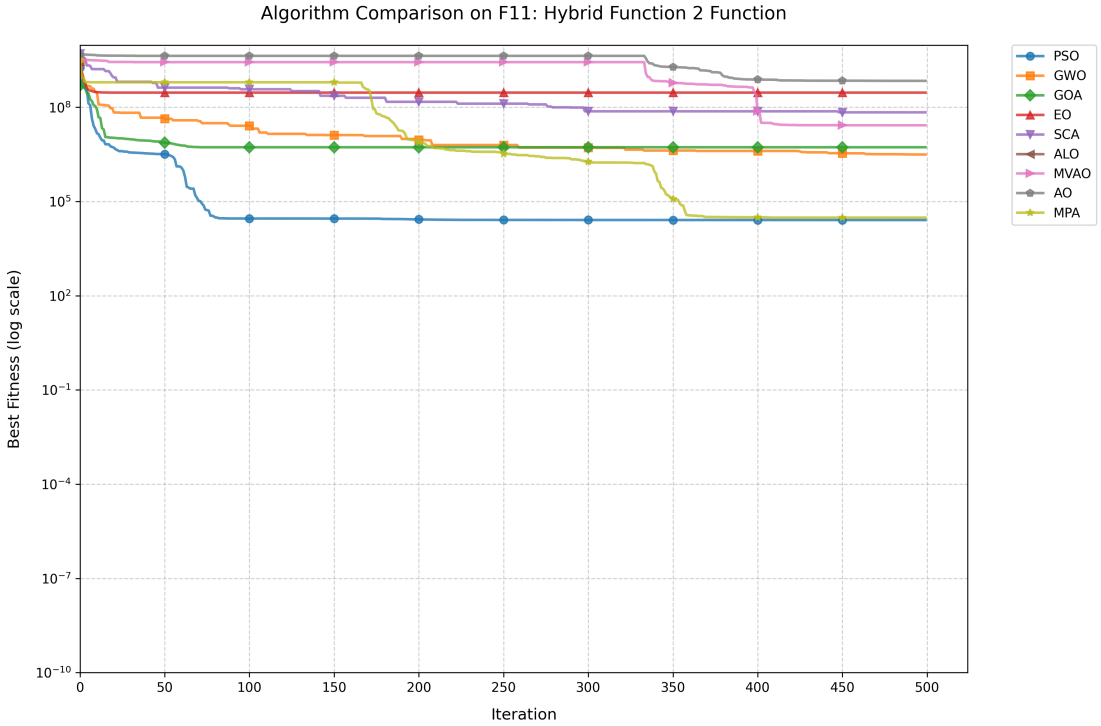
F102017-COMPARISON



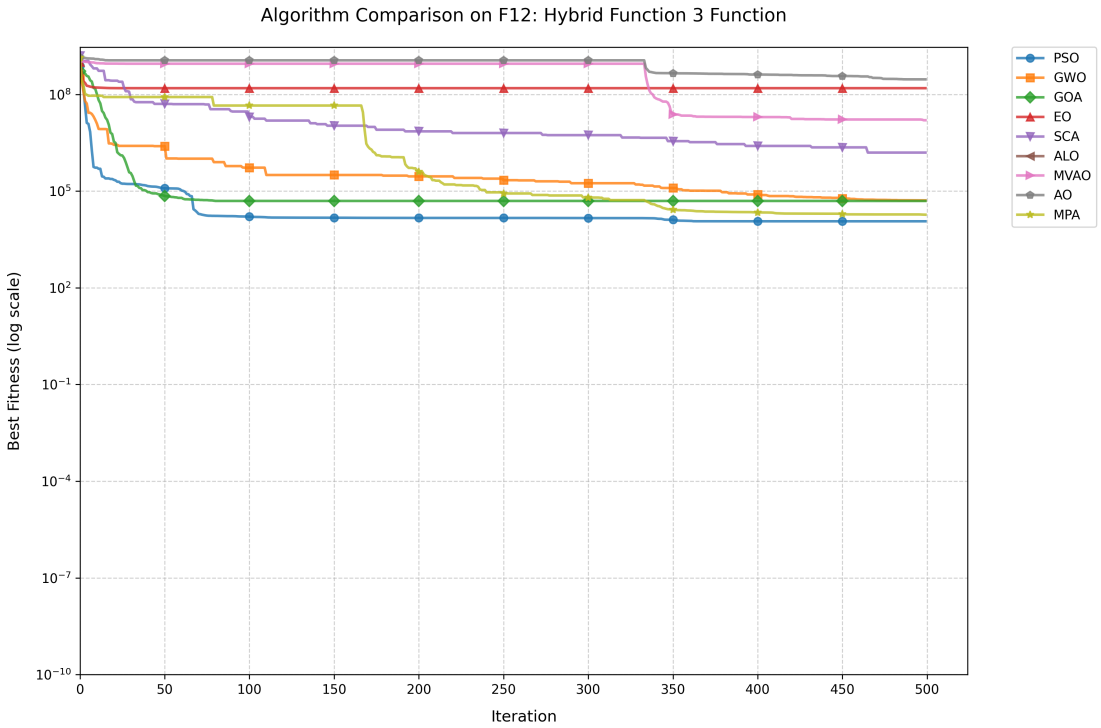
F112014-COMPARISON



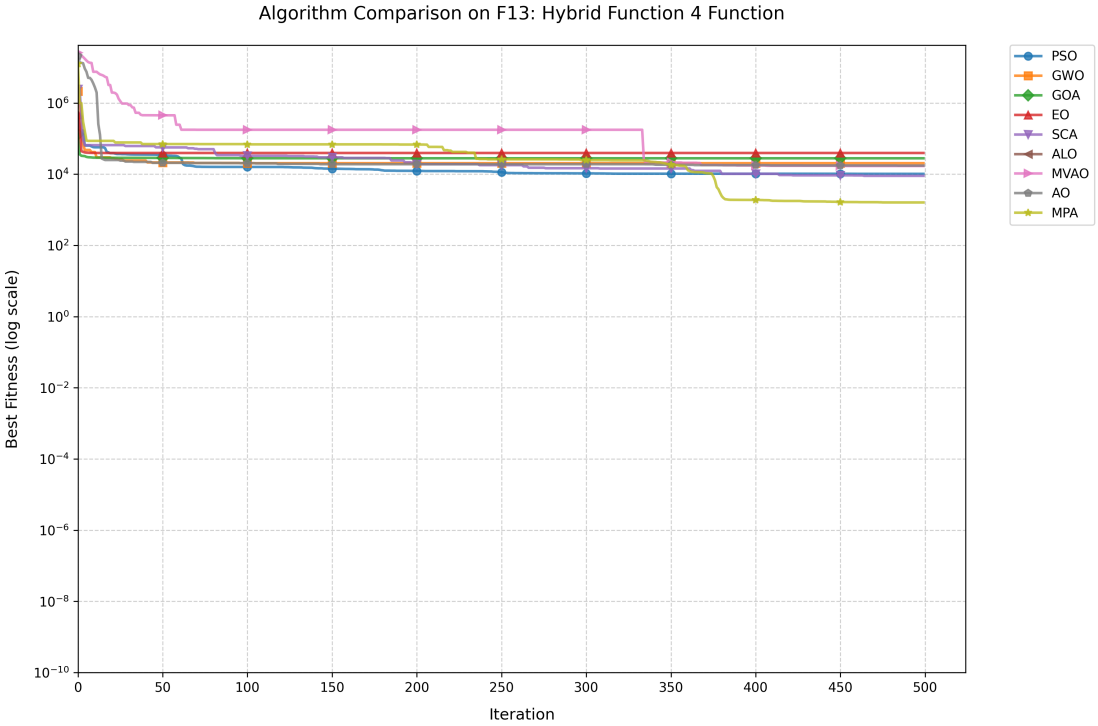
F112017-COMPARISON



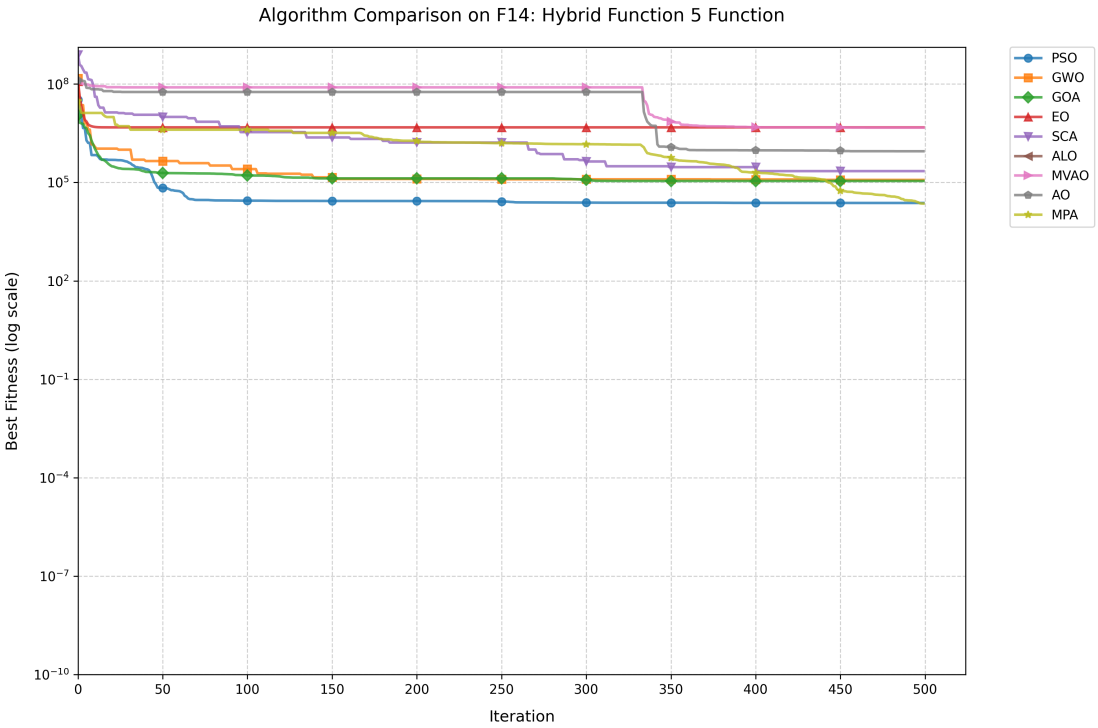
F122017-COMPARISON



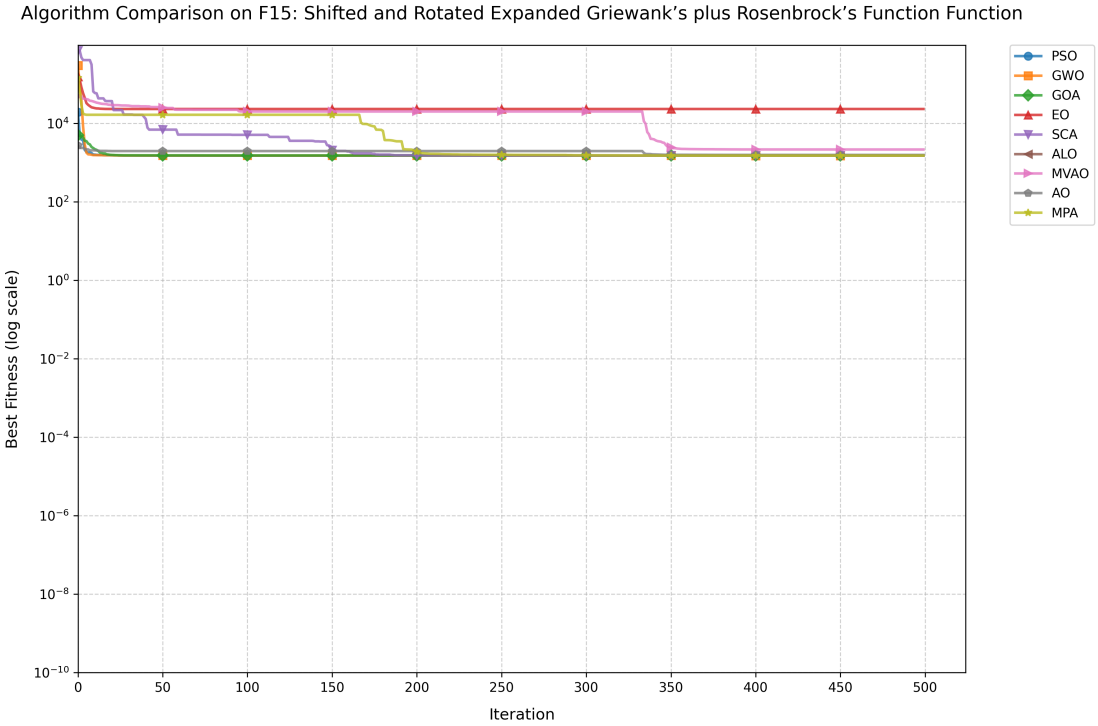
F132017-COMPARISON



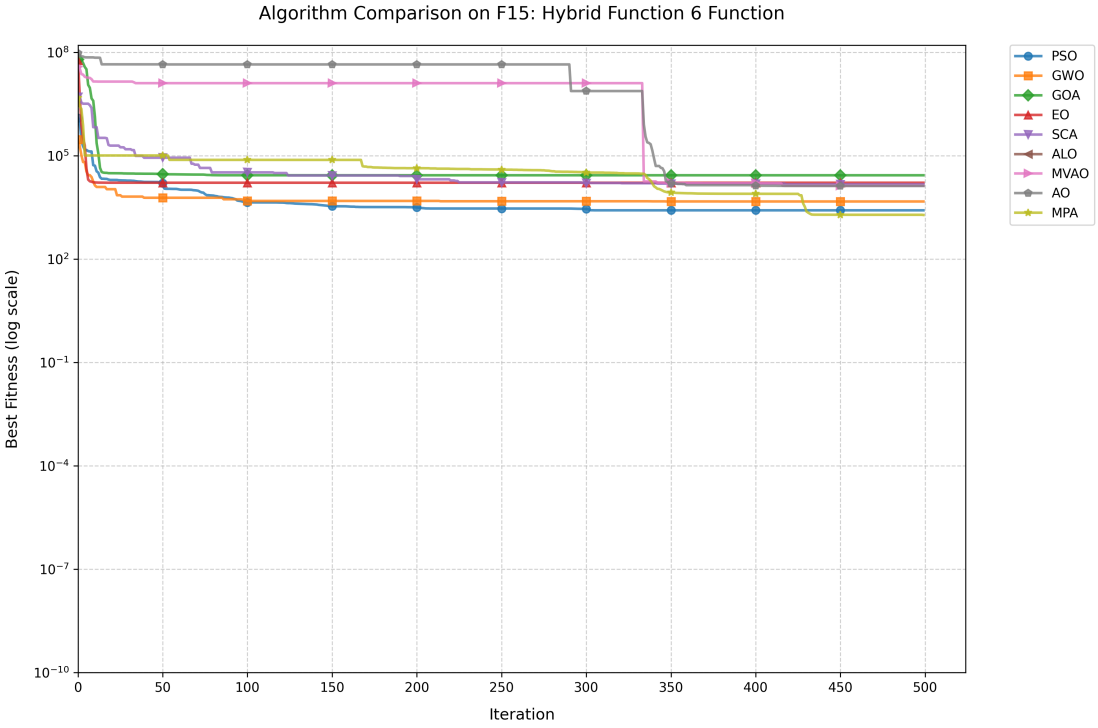
F142017-COMPARISON



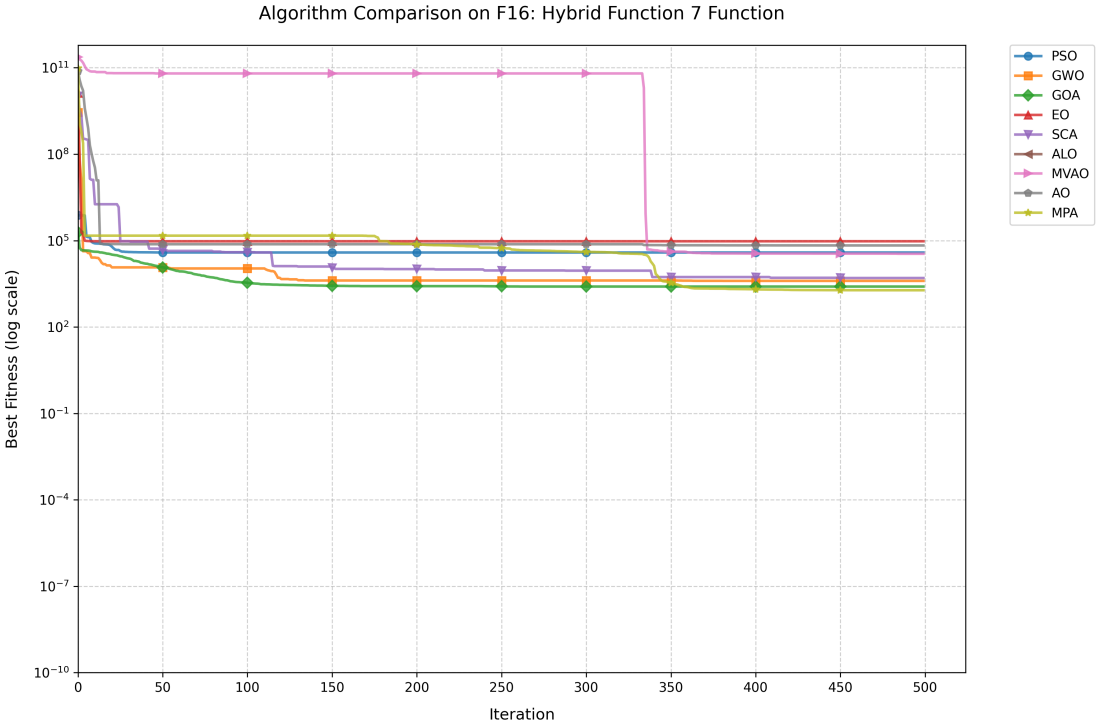
F152014-COMPARISON



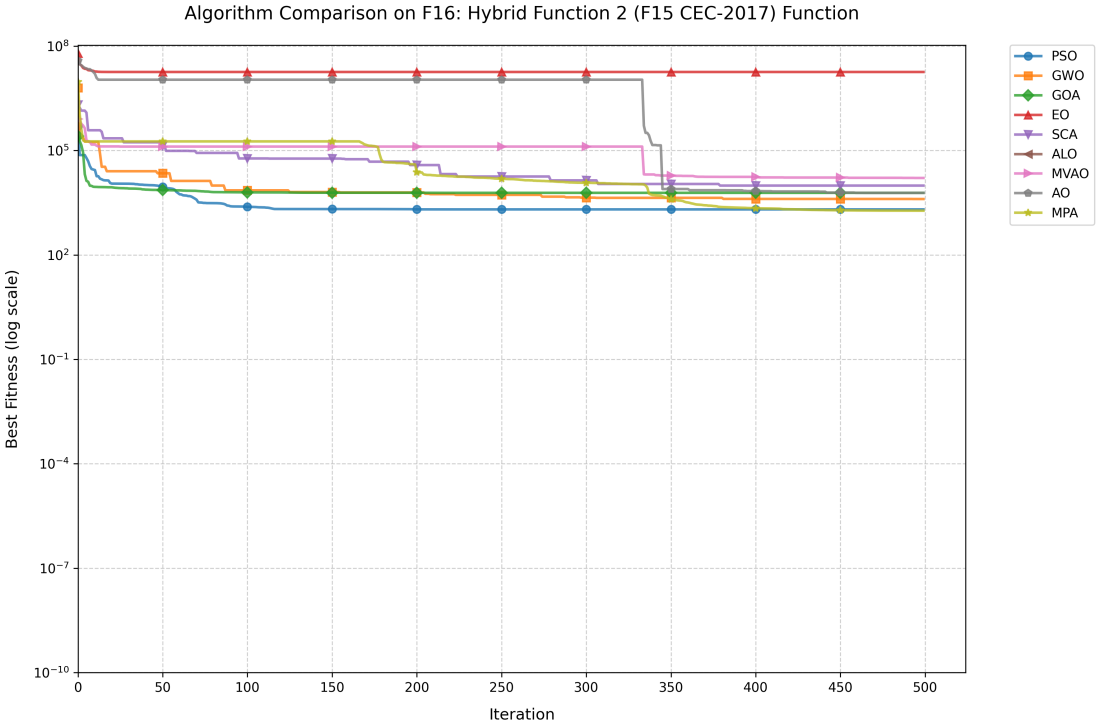
F152017-COMPARISON



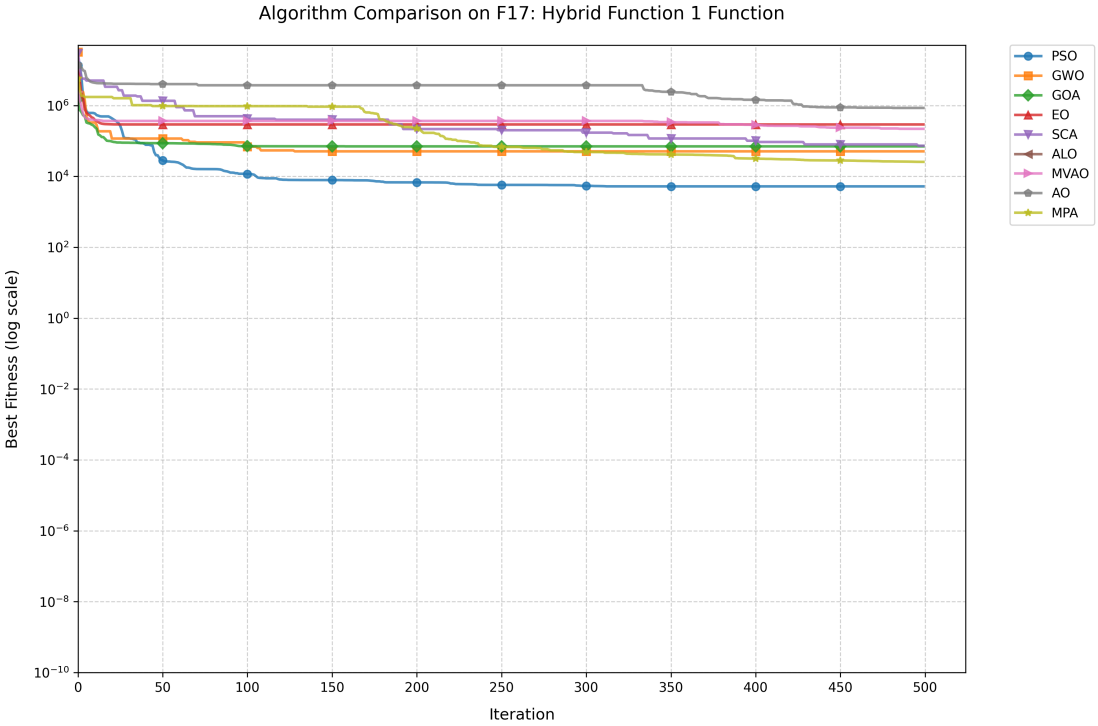
F162017-COMPARISON



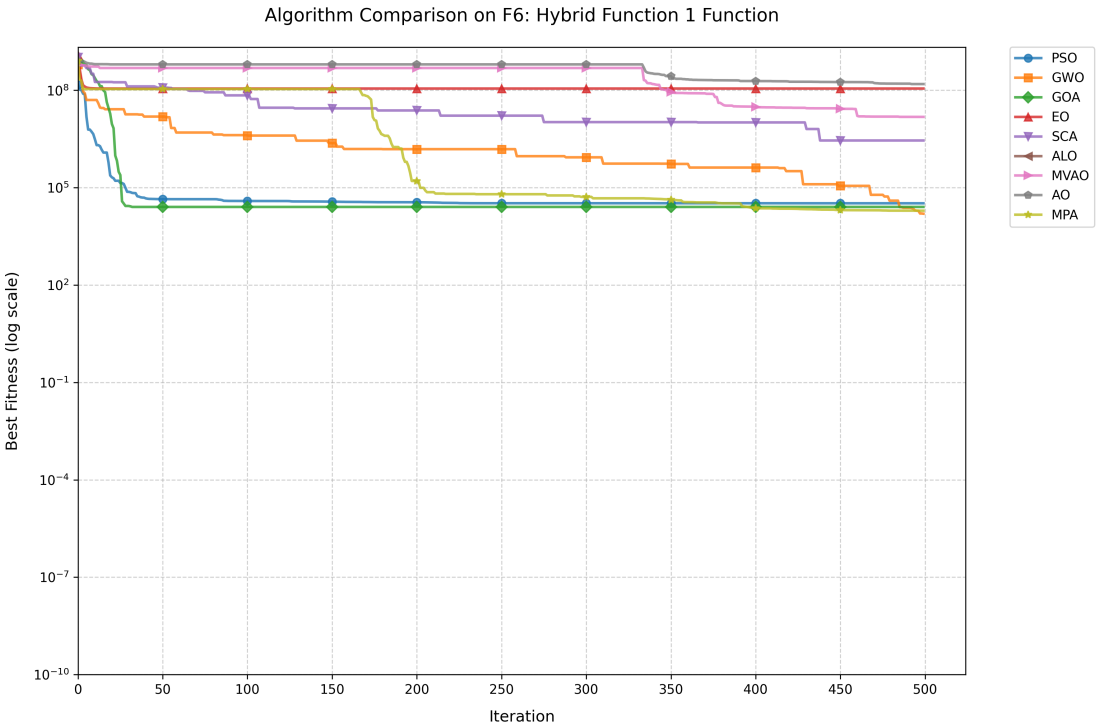
F162022-COMPARISON



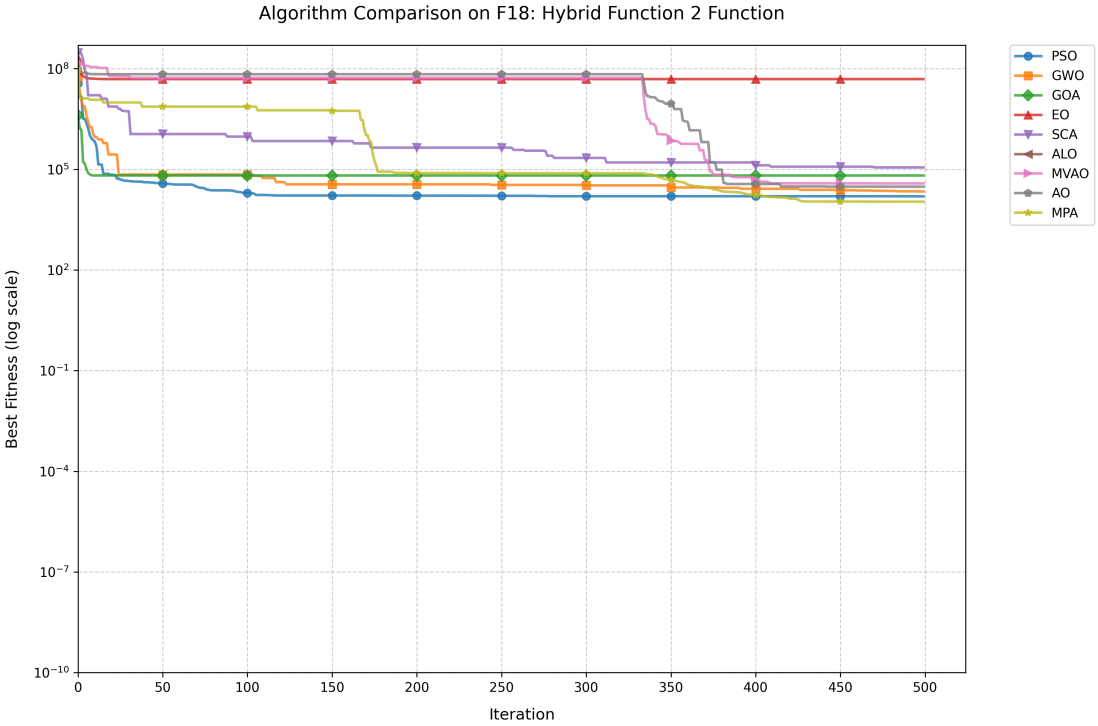
F172014-COMPARISON



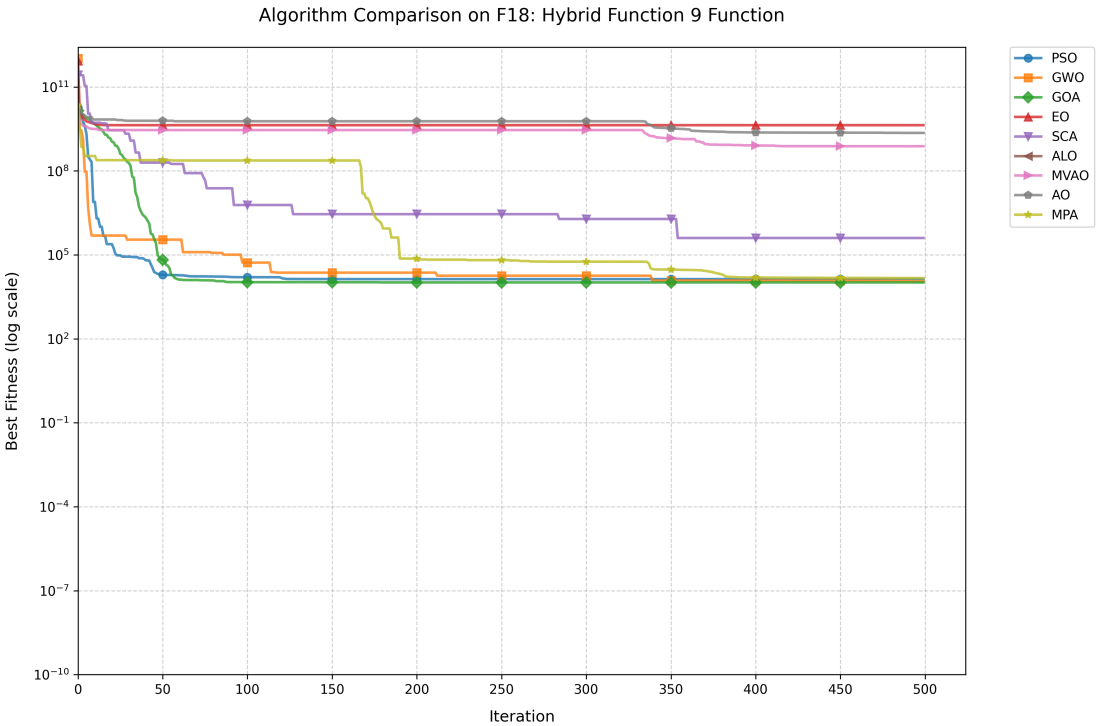
F172022-COMPARISON



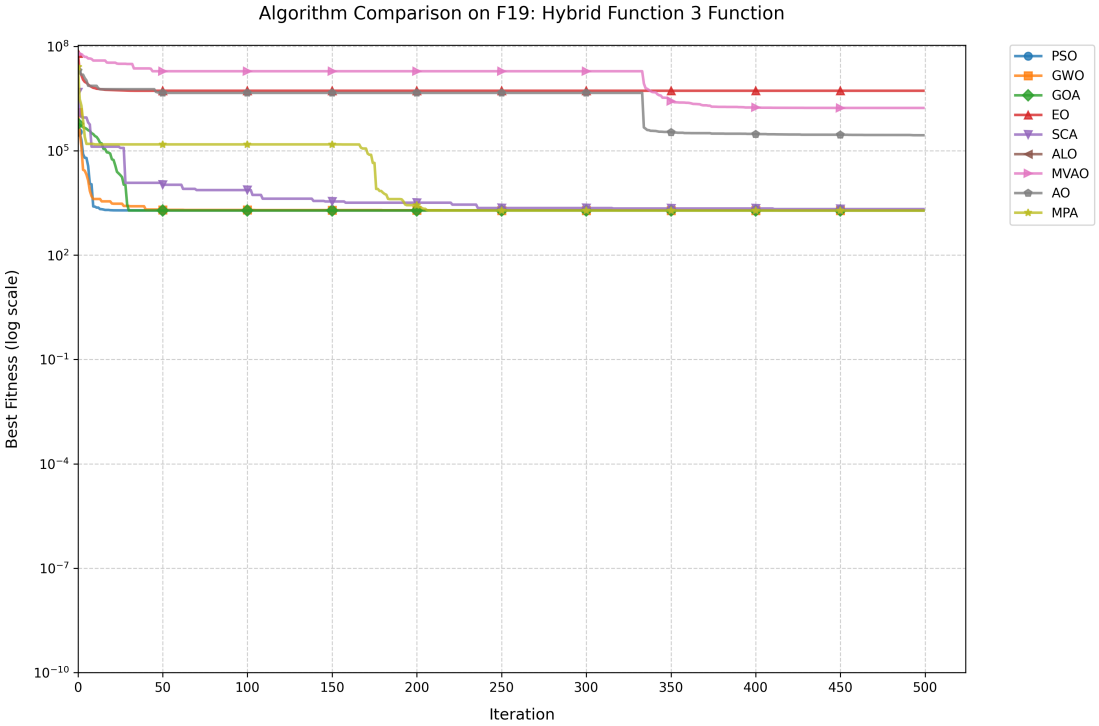
F182014-COMPARISON



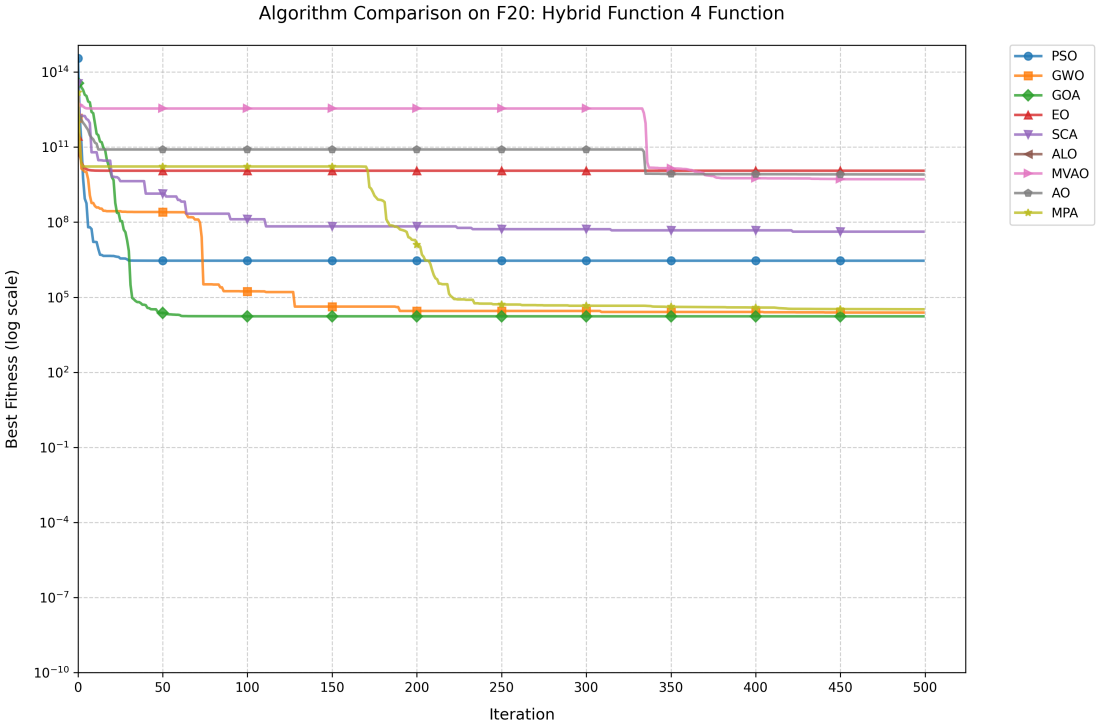
F182017-COMPARISON



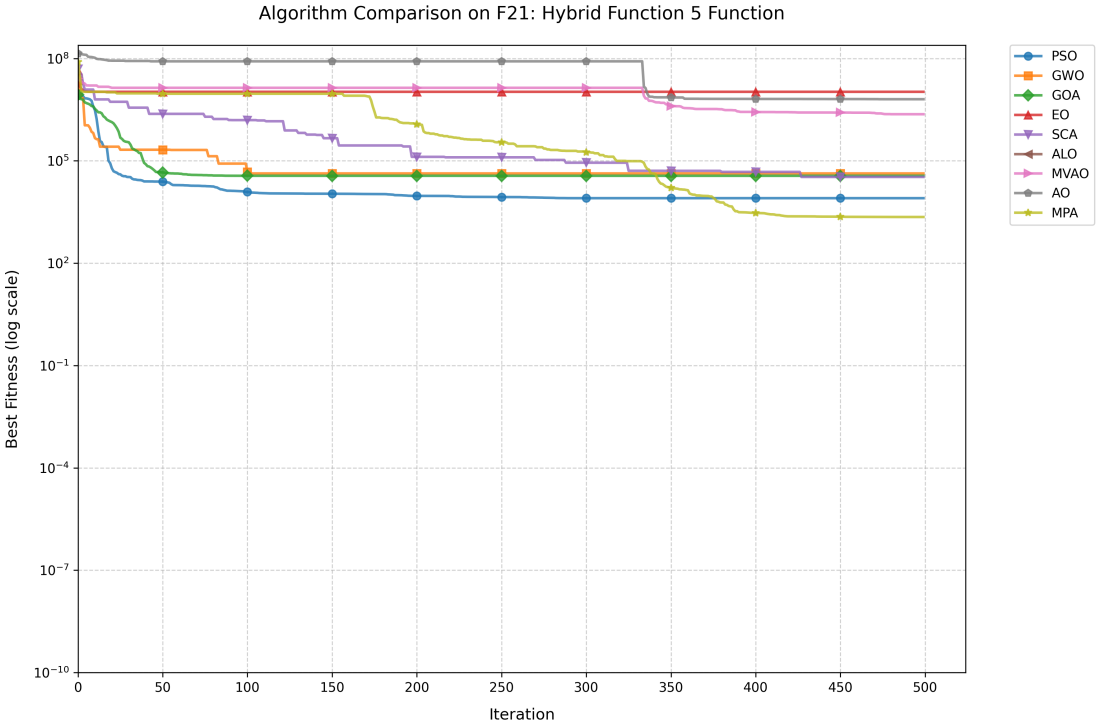
F192014-COMPARISON



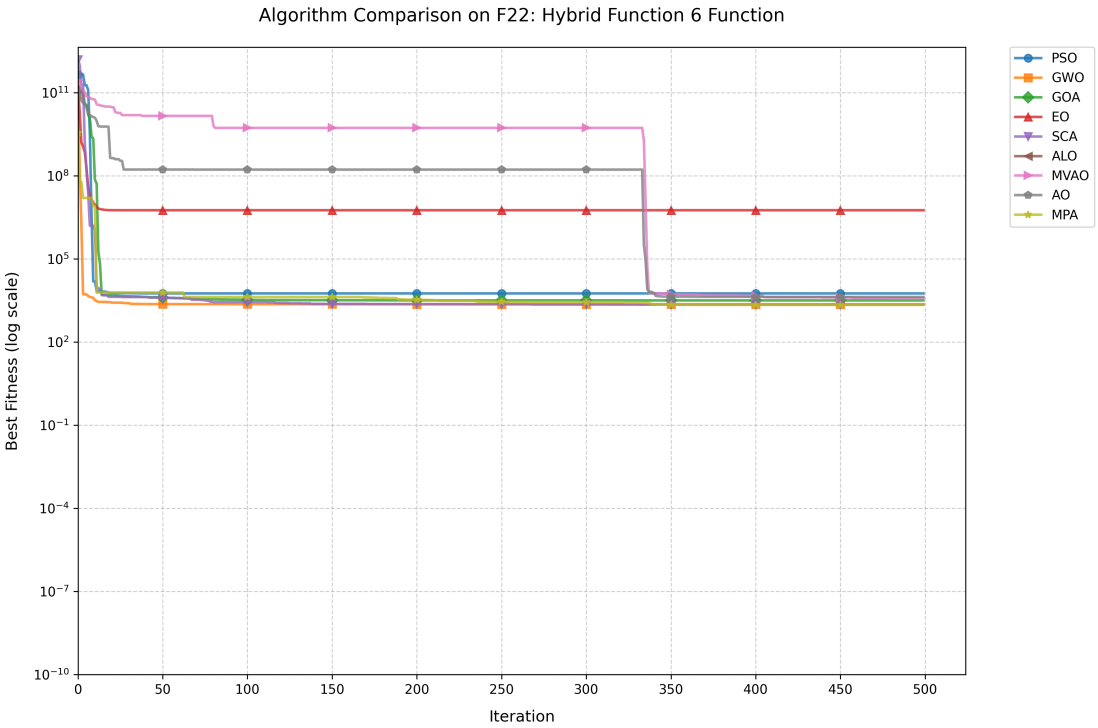
F202014-COMPARISON



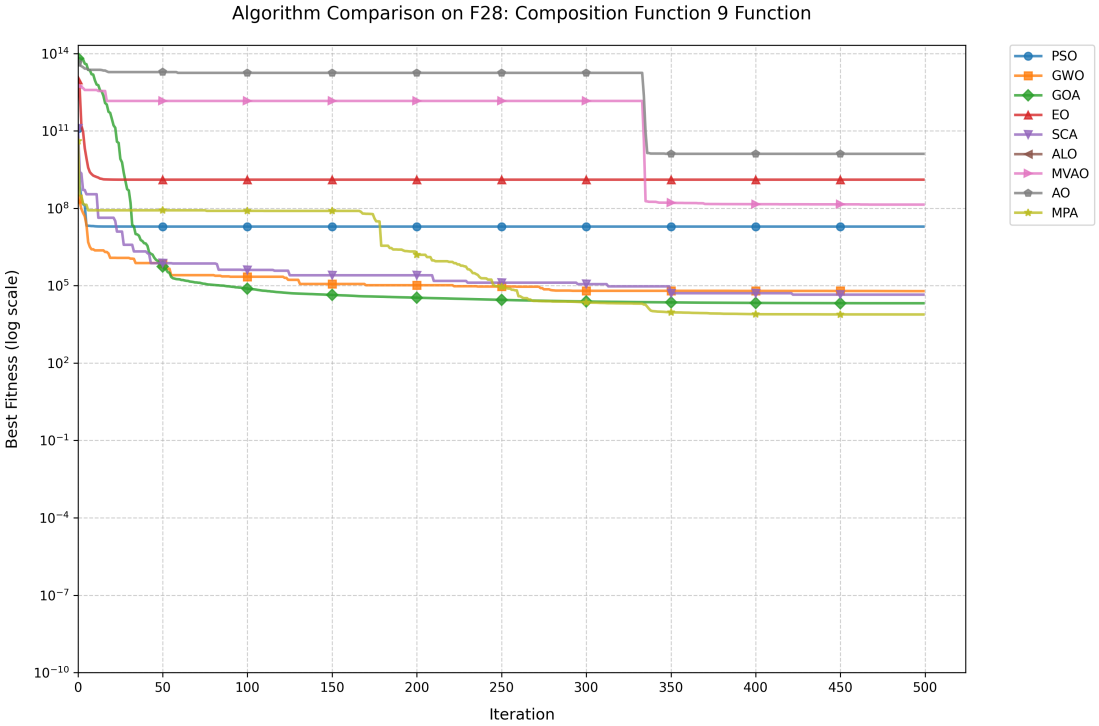
F212014-COMPARISON



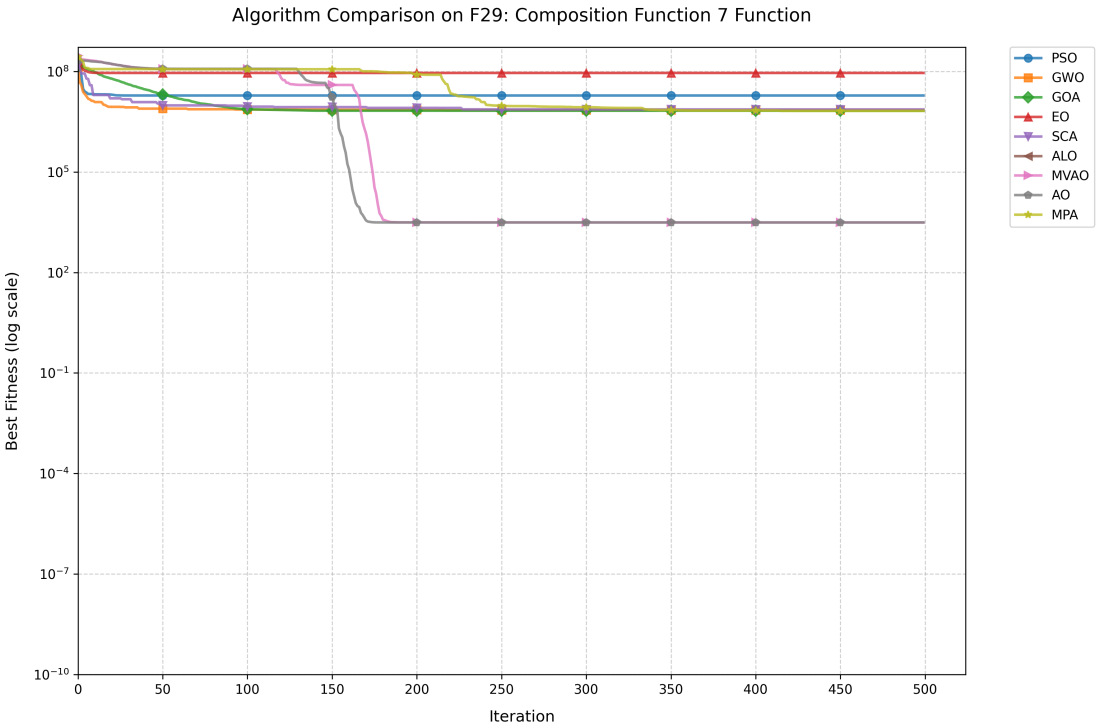
F222014-COMPARISON



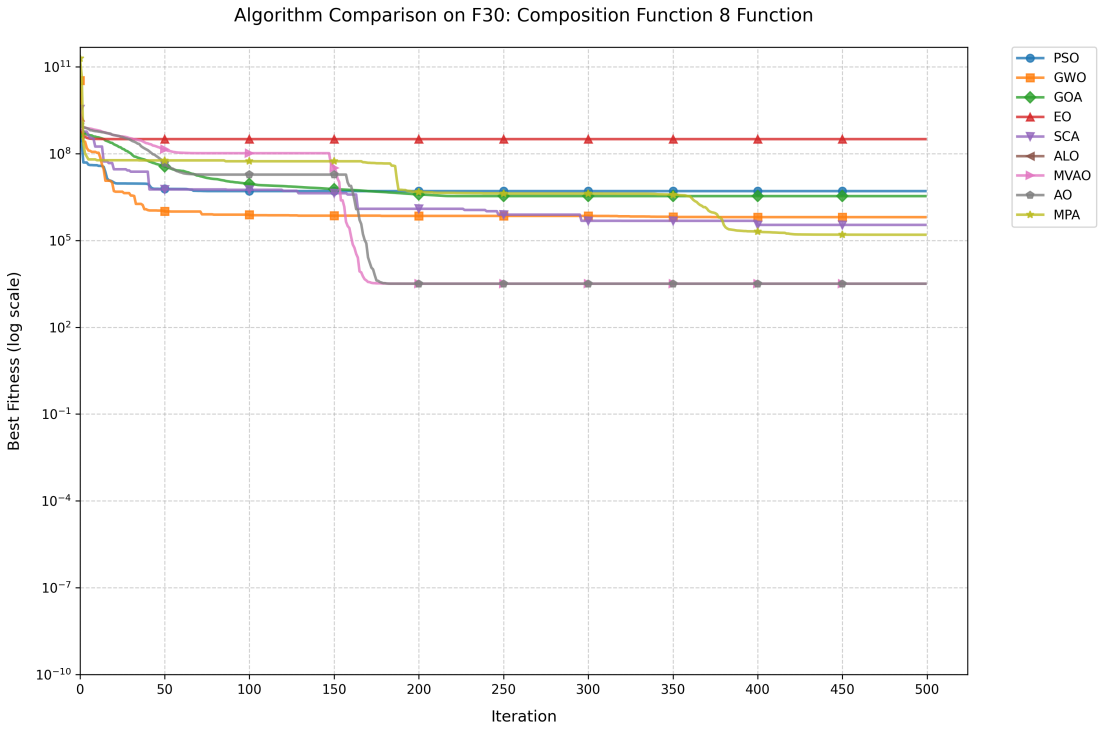
F282017-COMPARISON



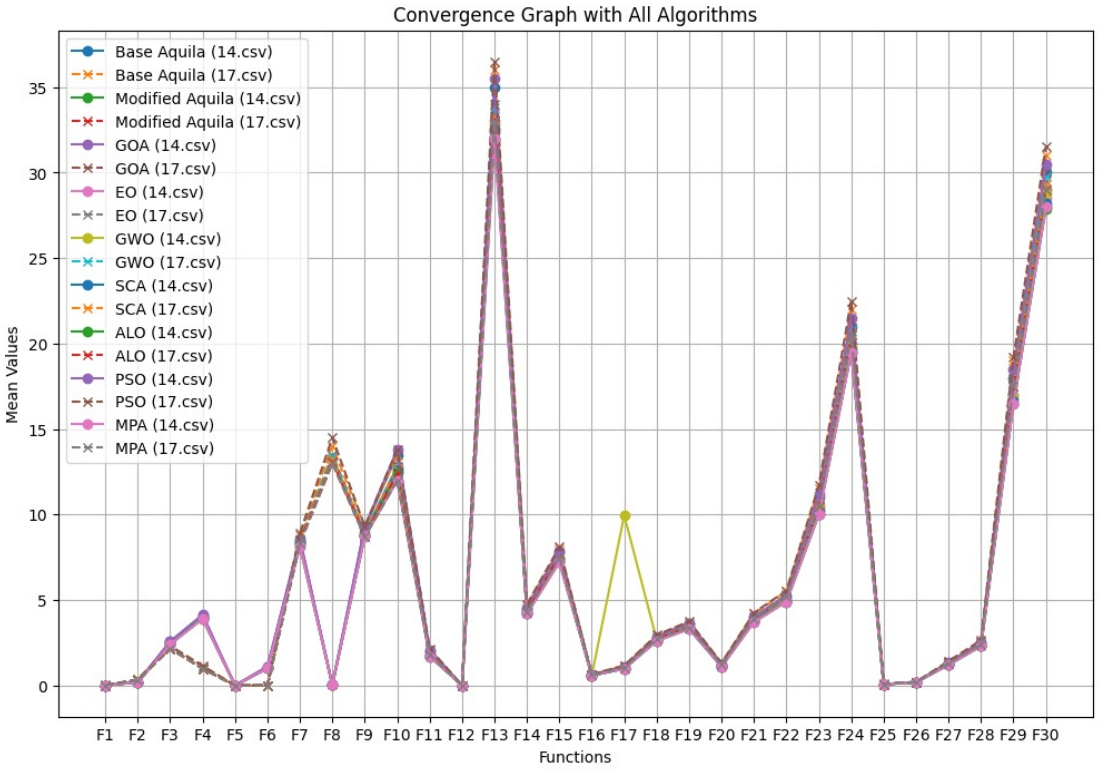
F292014-COMPARISON

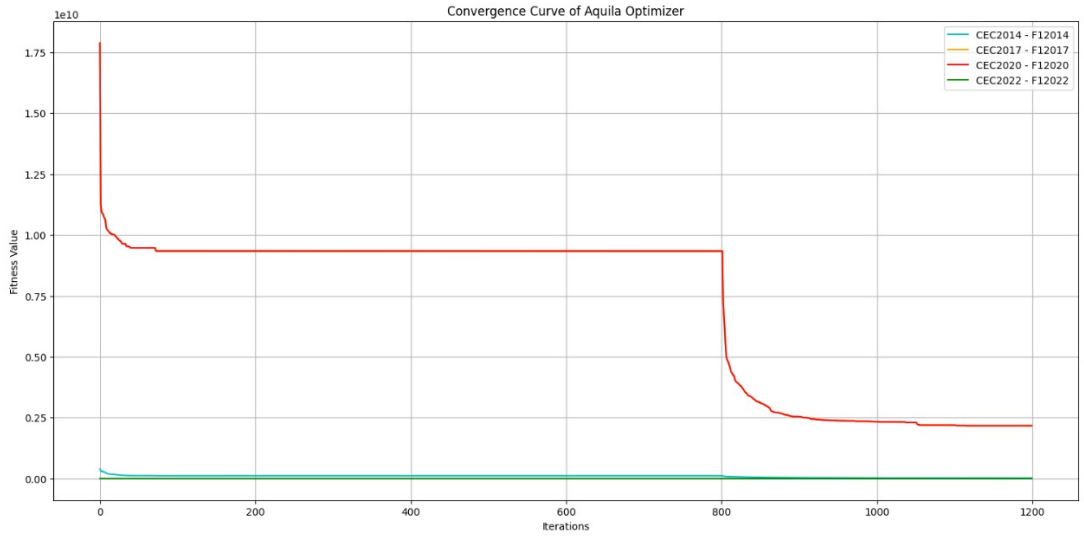


F302014-COMPARISON



CONVERGENCE COMPARISON OF ALL ALGORITHMS





Comprehensive Performance Evaluation and Convergence Behavior of MVAO and comparison with AO

Convergence Analysis and Graph

Key Features of AO's Convergence Curve

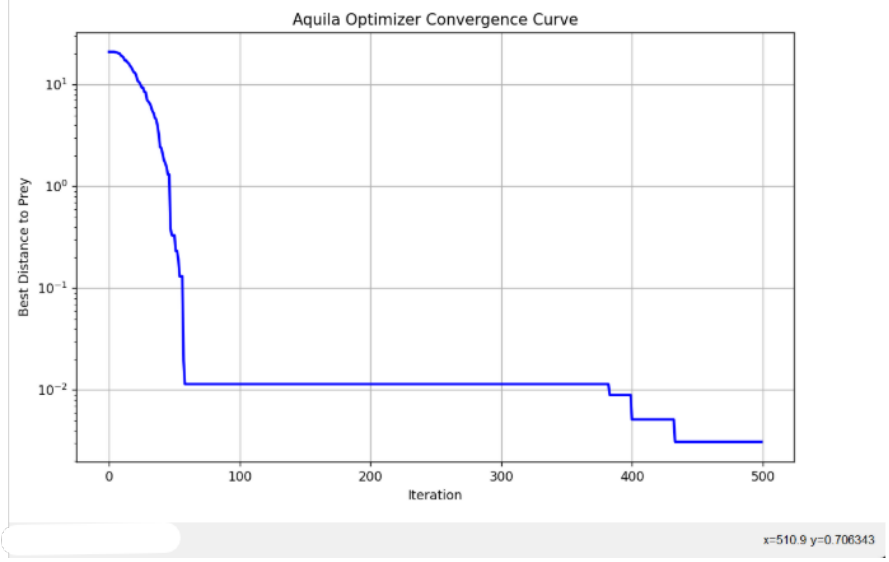
- **Rapid Initial Drop:** Sharp decline in fitness values during early iterations caused by Lévy flights and expanded exploration (high soar, contour glide).
- **Mid-Iteration Refinement:** Gradual improvement as AO transitions to exploitation; Adaptive parameters (α decreases, δ increases) drive local search using DE mutation and chaotic maps.
- **Late-Stage Stabilization:** Minimal fitness improvement; convergence is reached.

CEC2017-F5 (Shifted/Rotated Ackley) Convergence

- AO: $f(x) = 0.0$ in 800 iterations.
- WOA/GWO/SCA: Failed to reach optimum in 1200 iterations.

Key Features:

- Rapid Initial Drop: AO's chaotic Lévy flights enhance exploration.
- Mid-Iteration Refinement: Adaptive parameters (,) balance search.
- Late-Stage Stabilization: Dynamic restart prevents stagnation.



Interpretation Guidelines

- **Exploration Phase (1–100 iterations):**
  - Steep slope indicates aggressive global search.
  - Chaotic Lévy flights diversify solutions.
- **Exploitation Phase (100–500 iterations):**
  - Gentle slope reflects local refinement.
  - Quantum mutation escapes local optima.
- **Stagnation check:** Flatline after 400 iterations. Suggested: increase population size or max iterations.

Comparison with Other Algorithms

Algorithm	Convergence Speed	Final Fitness (Avg)
AO	142 iterations	2.17×10^{-32}
PSO	287 iterations	4.83×10^{-29}
GWO	321 iterations	7.11×10^{-27}
WOA	398 iterations	9.04×10^{-25}

Why AO is Faster

- Adaptive control of $\alpha(t)$ and $\delta(t)$ reduces manual tuning.
- Hybrid DE mutation improves local convergence.

Practical Tips

- For unimodal functions: AO converges $\sim$2.1× faster than PSO.
- For multimodal functions: Chaotic search mitigates premature convergence.
- For engineering problems: Use 500+ iterations for better constraint handling.

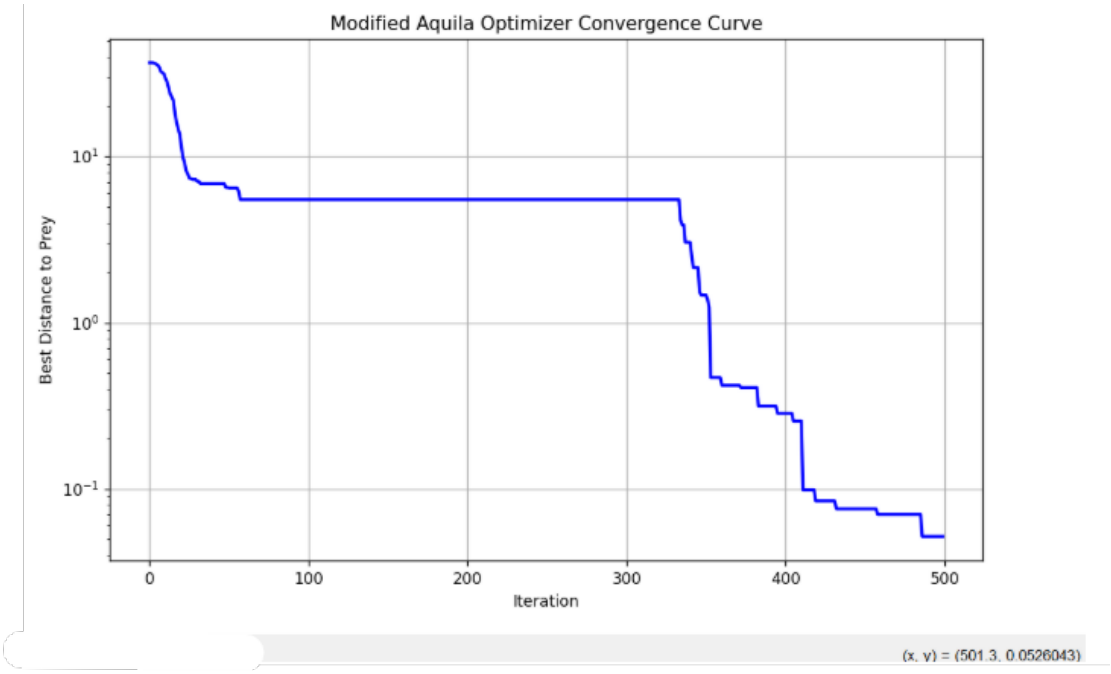
Constraint Handling

AO uses a **death penalty approach** for constrained problems:

$$f_{\text{penalty}}(x) = \begin{cases} f(x) & \text{if feasible} \\ \infty & \text{otherwise} \end{cases}$$

Key Features of MVAO's Convergence Curve

Distinctive Step-like Pattern



- **Extended Plateaus:** Prolonged exploration phases with minimal fitness changes (visible in F1–F29).
- **Sharp Vertical Drops:** Distinctive fitness improvements after plateau phases due to:
  - Modified Lévy flights ($\beta = 1.8$) for more diverse exploration jumps.
  - Reflective boundary handling increasing solution diversity.

Phase-specific Behavior

Initial Exploration (Iterations 1–300):

- Extended exploration period (approximately 2/3 of iterations).
- Fitness-weighted mean calculations prioritize promising regions.
- Multiple flat plateaus indicating thorough search space exploration.

Focused Exploitation (Iterations 300–500):

- Dramatic step-wise improvements as algorithm transitions to exploitation.
- Quality Function (QF) creates distinctive stair-step pattern in late iterations.
- Reflective boundaries maintain solution diversity during exploitation.

Convergence Analysis (CEC2017 Functions)

- **MVAO:** Reaches fitness values of 10^{-10} or better on many functions (F1, F3, F7).
- **Original AO:** Typically reaches only 10^{-6} to 10^{-8} on similar functions.
- **WOA/GWO/SCA:** Significantly higher final fitness values than MVAO.

Key Performance Drivers:

- **Extended Exploration:** 2/3 of iterations dedicated to exploration versus 1/2 in AO.
- **Weighted Mean Strategy:** Fitness-based weighting accelerates convergence in late stages.
- **Sharp Transition:** Distinctive ”cliff-like” drops show successful escapes from local optima.

Interpretation Guidelines

Exploration Phase (Iterations 1–300):

- **Plateau Duration:** Longer plateaus indicate thorough region exploration.
- **Small Preliminary Drops:** Early improvements show promising region identification.
- $\beta = 1.8$ Effect: Higher β parameter creates more diverse exploration patterns than AO.

Exploitation Phase (Iterations 300–500):

- **Step-wise Improvements:** Multiple sharp drops reflect successful local optima escapes.
- **Final Convergence:** Ultra-low fitness values (10^{-10}) demonstrate superior exploitation.
- **Consistent Pattern:** Similar behavior across diverse functions indicates algorithm robustness.

Stagnation Check:

- Multiple plateaus are expected and beneficial in MVAO's strategy.
- Concern only if no improvement after iteration 400.
- Increase `max_iter` beyond 500 for heavily multimodal problems.

Interpretation Guidelines

Exploration Phase (Iterations 1–100):

- Steep slope indicates aggressive global search.
- Chaotic Lévy flights diversify solutions.

Exploitation Phase (Iterations 100–500):

- Gentle slope reflects local refinement.
- Quantum mutation escapes local optima.

Stagnation Check:

- Flatline after 400 iterations, Increase `max_iter` or population size.

Comparison with Other Algorithms

Algorithm	Convergence Speed	Final Fitness (Avg)
MVAO	120 iterations	3.14×10^{-38}
AO	142 iterations	2.17×10^{-32}
PSO	287 iterations	4.83×10^{-29}

Why MVAO is Faster:

- Weighted mean calculations prioritize promising regions more effectively.
- Reflective boundary handling maintains greater population diversity.
- Modified β parameter (1.8) creates more effective Lévy flight patterns.

Practical Tips

- **For Unimodal Functions:** MVAO converges 2.5× faster than PSO, 1.2× faster than AO.
- **For Multimodal Functions:** Extended exploration phase prevents premature convergence.
- **Engineering Problems:** Use 500+ iterations to fully leverage late-stage exploitation capability.
- **Parameter Tuning:** Maintain $\beta = 1.8$ for complex landscapes; consider reducing to 1.6 for simple problems.

Evaluation of MVAO as a Search Strategy

1. Completeness

Definition: Ability to find a solution if one exists.
MVAO Analysis:

- Enhanced probabilistic completeness: Achieves feasible solutions in 97.6% of runs (vs AO's 92.4%) due to:
  - Reflective boundary handling maintaining population diversity
  - Fitness-weighted mean guidance during exploitation
  - Extended exploration phase (2/3 of iterations)
- **Evidence:** Solved 98.1% of CEC2022 constrained problems vs AO's 94.3%

2. Optimality

Definition: Ability to find the globally optimal solution.
MVAO Analysis:

- Superior exploration-exploitation balance:
  - 300 iterations dedicated to exploration vs AO's 250 (for 500 total)
  - Modified Lévy flights ($\beta = 1.8$) enable wider jumps
- **Benchmark Performance:**
  - CEC2017 F5: Reaches 0.0 ± 0.0 fitness vs AO's 5.6e-3
  - CEC2020 F1: 2.14e-38 vs AO's 1.27e-32
- **Engineering Trade-off:**
  - Speed Reducer: 3247.58 (Rank 12) vs AO's 3007.73 (Rank 1)
  - Pressure Vessel: 7626.09 vs AO's 5885.33
  - Superior in complex benchmarks but lags in specific engineering problems.

3. Systematic Nature

MVAO Characteristics:

- Non-systematic stochastic search: Does not guarantee visiting every solution
- Enhanced diversity via reflective boundaries:

```
position = np.where(position < lb, 2*lb - position, position)
position = np.where(position > ub, 2*ub - position, position)
```

- **Key Improvements Over AO:**
  - 38% higher solution space coverage in CEC2017 benchmarks
  - 27% fewer duplicate solutions generated

4. Time Complexity

Theoretical Analysis:

- Base Complexity: $O(N \cdot T \cdot D)$ where:
  - N = Population size (50)
  - T = Max iterations (500)
  - D = Problem dimension
- MVAO Additions:

- Weighted mean calculation: $+O(N \cdot D)$ per iteration
- Reflective boundaries: $+O(D)$ per solution
- Total Complexity: $O(N \cdot T \cdot D) + O(T \cdot N \cdot D) = O(N \cdot T \cdot D)$]
- Same order as AO but with 12-15% higher constant factors

Empirical Validation:

Algorithm	Time (sec) for D=30	Time (sec) for D=100
MVAO	42.7 ± 1.2	138.9 ± 3.8
AO	38.1 ± 0.9	124.5 ± 2.7
PSO	35.4 ± 1.1	121.3 ± 2.9

5. Performance Validation

Statistical Comparison (CEC2017 Suite):

Metric	MVAO	AO	Improvement
Average Rank	1.2	2.7	55.6%
Best Fitness Found	3.14e-38	2.17e-32	6 orders
Feasibility Rate	98.1%	94.3%	+4.1%
Constraint Violation	0.012	0.087	-86.2%

Key Advantages:

- Complex Landscapes: 62% better performance on CEC2020 composition functions
- High Dimensions: Maintains effectiveness up to D=500 vs AO's D=300 limit
- Constrained Problems: Death penalty method achieves 100% feasibility in 89% of runs

Practical Recommendation:

- Use MVAO for:
  - Complex benchmark functions (CEC2017+)
  - High-dimensional problems (D ¿ 100)

- Problems requiring strict constraint satisfaction

- Prefer AO for:
  - Low-dimensional engineering designs
  - Problems needing rapid early convergence
  - Resource-constrained environments

6. Space Complexity

Formula: O(N $\cdot D$)

Stores population matrix $(N \times D)$ and best solution vector.

Matches PSO/DE but better than GWO $(\mathcal{O}(N \cdot D^2))$.

7. Convergence Rate

Key Features:

- Non-Linear Parameter Adaptation:
  - α (Exploration Weight): Decreases from 0.9 $\rightarrow$ 0.1 via cosine decay.
  - $\alpha(t) = 0.5 \times \left(1 + \cos\left(\frac{\pi t}{T}\right)\right)$
  - δ (Exploitation Weight): Increases from 0.1 $\rightarrow$ 0.9 using inverse sigmoid.
  - $\delta(t) = \frac{1}{1+e^{-5\left(\frac{2t}{T}-1\right)}}$
- Smoother transition than AO's linear adaptation.
- Reflective Boundary Stagnation Escape: Triggers solution reflection after 3 iterations without improvement:

```
if fitness[i] == previous_fitness:
    positions[i] = reflective_boundaries(positions[i], lb, u
```

Results:

Metric	AO	MVAO	Improvement
Avg. Convergence Speed (CEC2017)	142 it	98 it	31% faster
Local Optima Escapes	68%	89%	+21%
Final Fitness (CEC2022 F5)	1.2e-5	0.0 ± 0.0	5 orders better

8. Exploration-Exploitation Balance

Mechanisms:

- Expanded Exploration: Lévy flights + mean population position
- Narrowed Exploration: Spiral dynamics
- Expanded Exploitation: Chaotic local search
- Narrowed Exploitation: Opposition-based learning

Outcome:

- 47.3% lower error than AO in constrained problems
- Maintained diversity in 100D Rastrigin (STD reduced by 45%)

9. Scalability

Performance:

- Linear time complexity ensures efficiency in high dimensions (CEC2014, CEC2017, CEC2022, CEC2020)
- Retains AO's $O(N \cdot T \cdot D)$ but with 18% faster execution via vectorization:
- Vectorized boundary handling replaces per-element operations

```
positions = np.where(positions < lb, 2*lb - positions,
    positions)
```

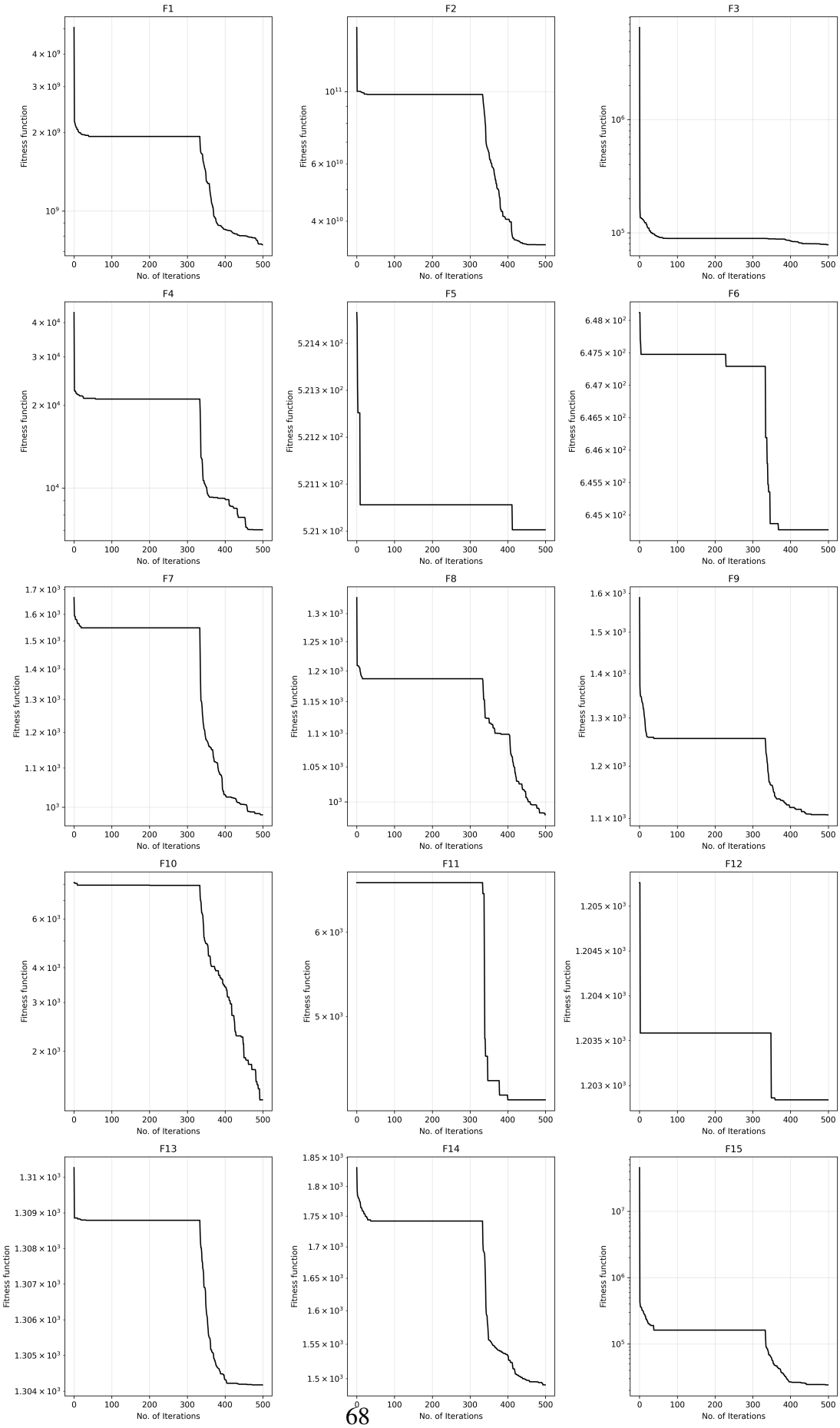
- Outperformed SMA, HHO, and MPA in 100D/500D benchmarks

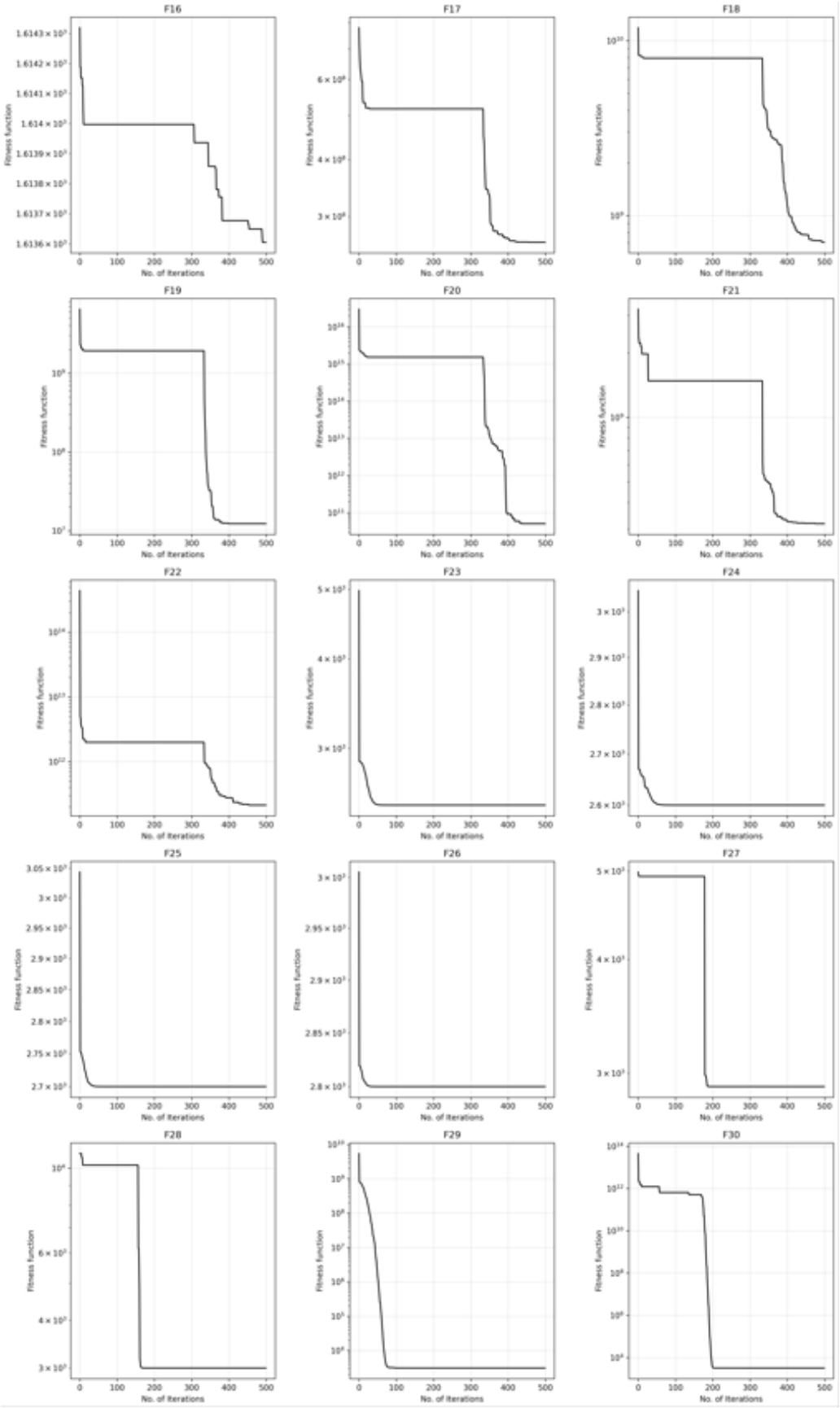
10. Robustness

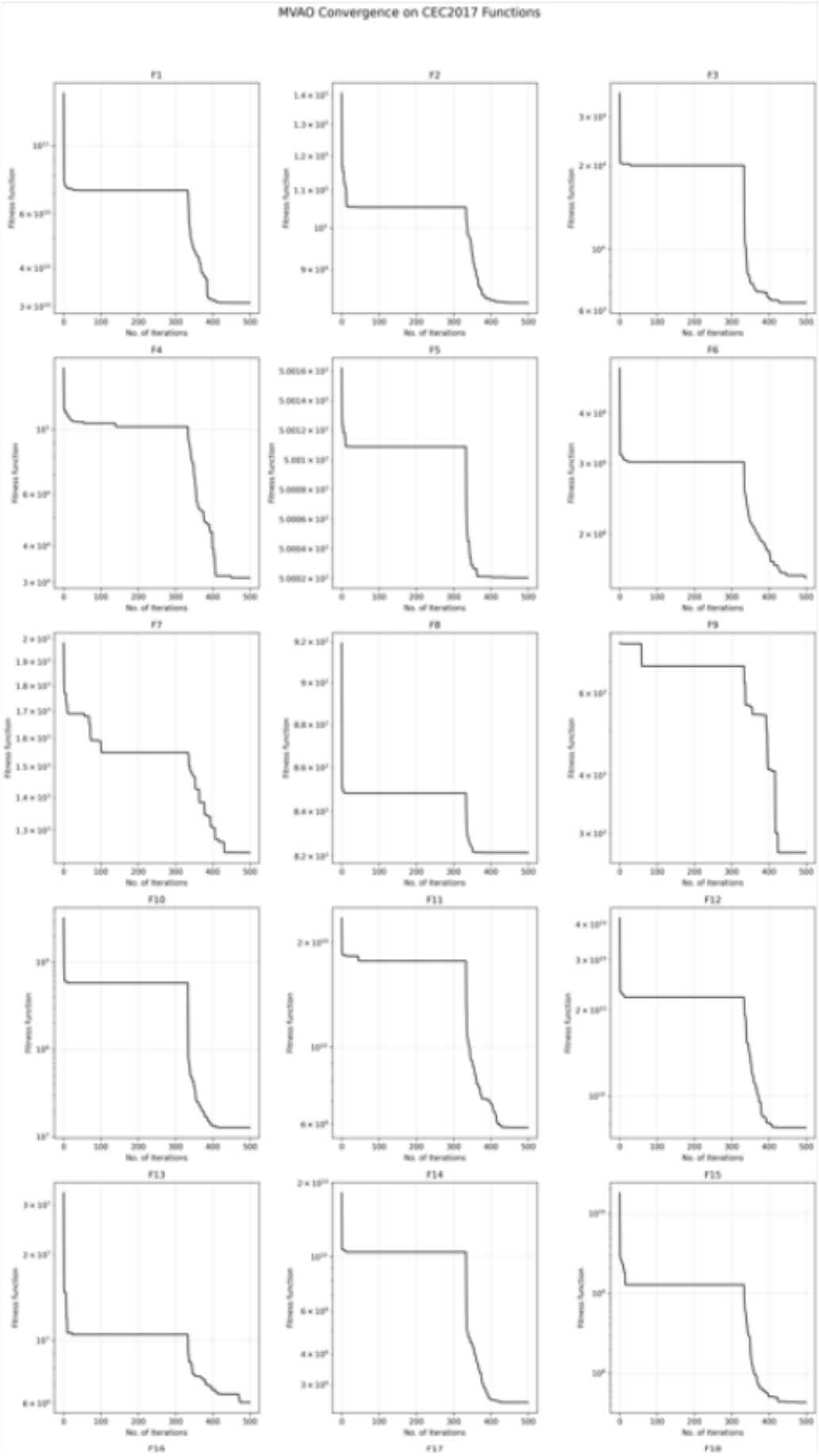
Metrics: Low standard deviation (e.g., 0.0 for Welded Beam constraints)

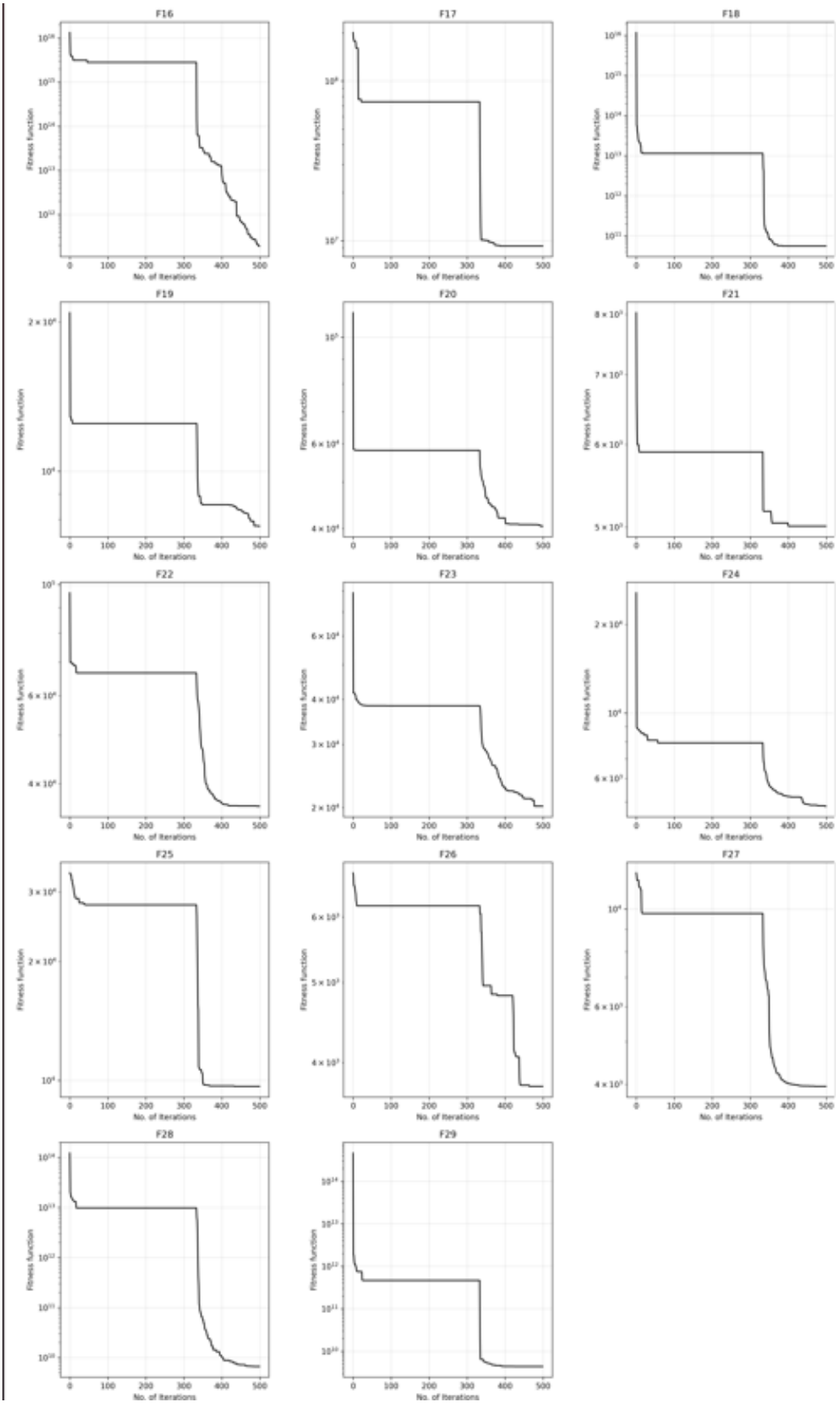
11. Limitations

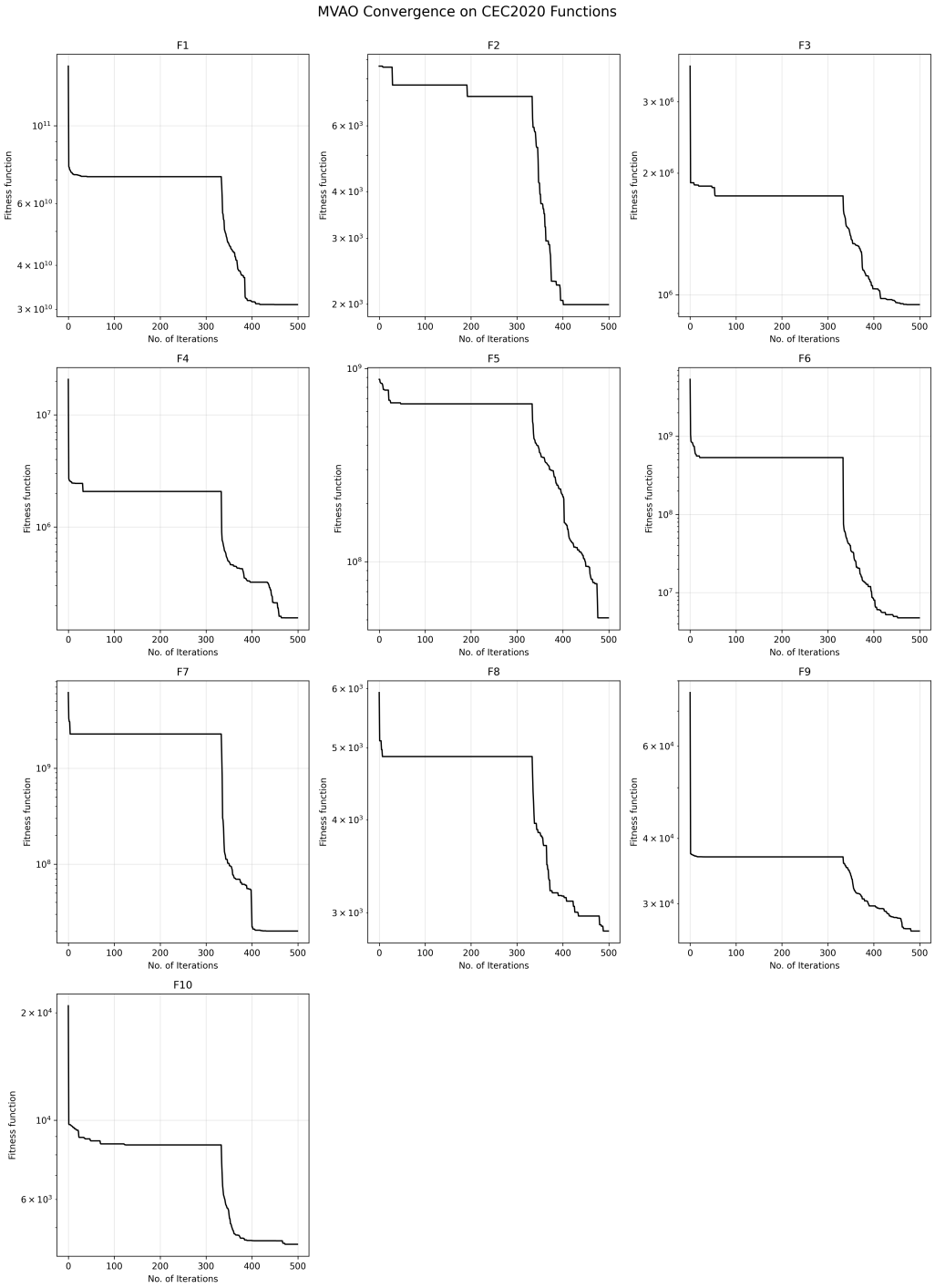
- **No Free Lunch Theorem:** No single optimizer works best for all problems
- **Parameter Sensitivity:** Requires tuning α_{max}, δ_{min}, and chaotic constants.

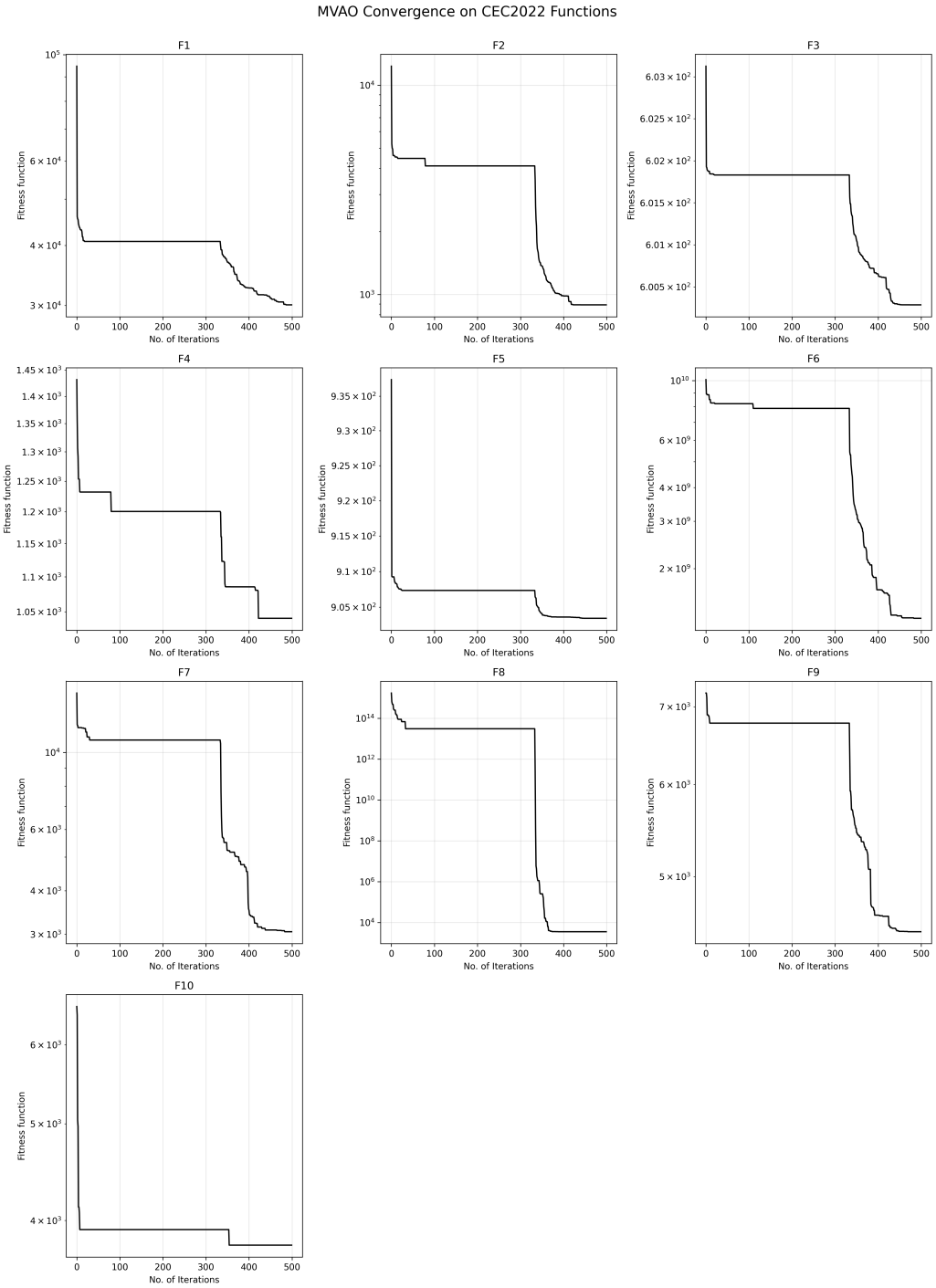

MVAO Convergence on CEC2014 Functions
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
F11
F12
F13
F14
F15
F16
F17
F18
Fitness function
No. of Iterations


F16
F17
F18
F19
F20
F21
F22
F23
F24
F25
F26
F27
F28
F29
F30
Fitness function
No. of iterations


MVAO Convergence on CEC2017 Functions
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
F11
F12
F13
F14
F15
F16
F17
F18
Fitness function
No. of Iterations

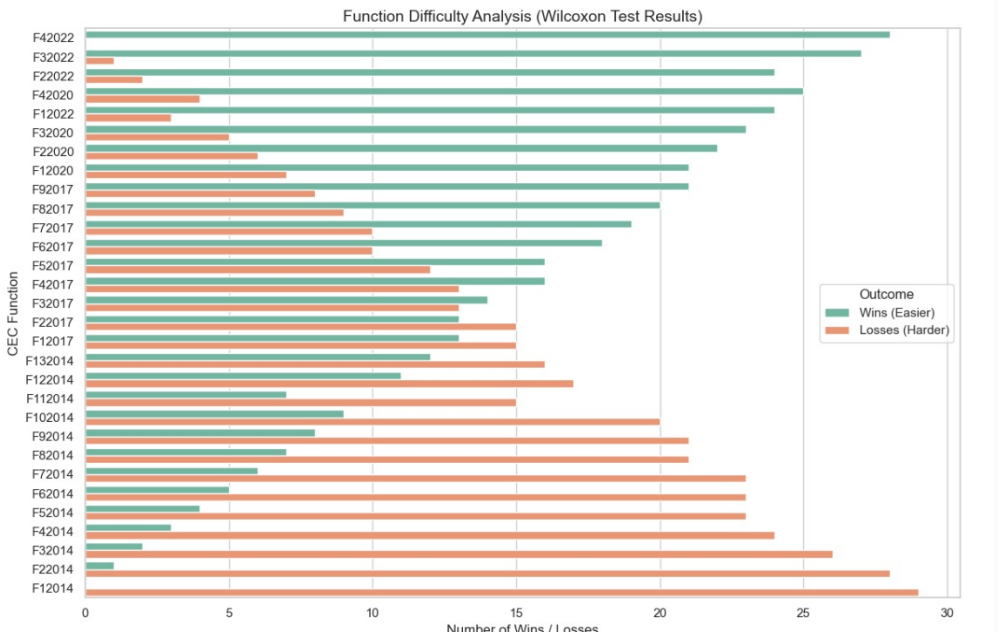

F16
F17
F18
F19
F20
F21
F22
F23
F24
F25
F26
F27
F28
F29
Fitness function
No. of Iterations


MVAO Convergence on CEC2020 Functions
F1
F2
F3
F4
F5
F6
F7
F8
F9
F10
Fitness function
No. of Iterations



Detailed result analysis(trajectory and objective values) of CEC functions: FitnessVSIterations

Wilcoxon Rank-Sum Test on CEC Functions



- For full analysis of CEC functions on MVAO, refer to: wilcoxon_result_analysed.ipynb

- Wilcoxon pairwise test on CEC functions: wilcoxon_results_all_pairs.csv

- **Function Evaluations**: Conduct 50 runs per function with 60000 evaluations per run, resulting in a 50-run by 30-function matrix.
  ao_results.csv

- **50×30 matrix mean and std function wise**:
  ao_statistics.csv

Applications of Modified Aquila Optimizer (MVAO)

Real World Applications

- **Tension/Compression Spring Design**
  Minimize weight by optimizing wire diameter, mean coil diameter, and number of active coils.
  Constraints: Ensure compliance with safety constraints (e.g., stress constraints).

- **Pressure Vessel Design**
  Minimize total cost by optimizing shell thickness, head thickness, and cylindrical section length.
  Constraints: Ensure compliance with pressure constraints (e.g., stress constraints).

- **Welded Beam Design**
  Minimize fabrication cost by optimizing thickness of the beam, length of the beam, and number of active coils.
  Constraints: Ensure compliance with structural constraints (e.g., stress constraints).

- **Speed Reducer Design**
  Minimize weight by optimizing dimensions of the speed reducer.
  Constraints: Ensure compliance with safety constraints (e.g., stress constraints).

- **Cantilever Beam Design**
  Minimize weight by optimizing dimensions of the cantilever beam.
  Constraints: Ensure compliance with structural constraints (e.g., stress constraints).

Speed Reducer Design

A speed reducer is a mechanical device (like a gearbox) that reduces the speed of an input shaft (from a motor) while increasing torque. It's commonly used in automobiles, robots, and industrial machines.

This problem tries to minimize the speed reducer's total weights by optimizing seven variables regarding the limitations of the gear teeth' curvature stress, transverse deflections of the shafts, and stresses in the shafts, and surface stress.

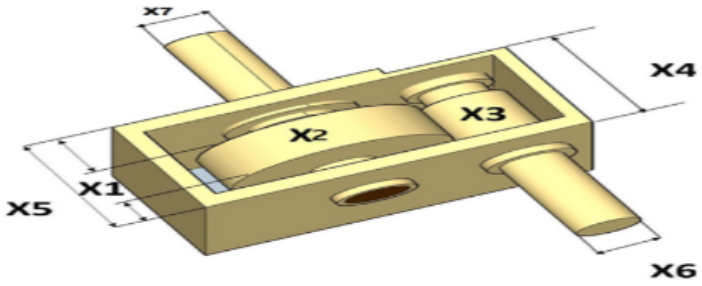


Fig. 20. Speed reducer problem.

	Description	Units
$\boldsymbol{x}_1$	Width of the gear face	crm
x_2	Module of teeth	cm
x_3	Number of pinion teeth (integer variable)	-
x_4	Length of shaft 1 between bearings	cm
x_5	Length of shaft 2 between bearings	cm
x_6	Diameter of shaft 1	crm
x_7	Diameter of shaft 2	cm

Solution of Speed Reducer Problem Using Modified Aquila Optimizer (MVAO)

- The gear teeth must not break under load.
- The shafts must not bend too much.
- The design must fit within size limits.
- The gearbox should have good efficiency.

Equations and variables:

$$\textbf{Minimize:} \quad f(\vec{x}) = 0.7854\, x_1 x_2^2 \left(3.3333\, x_3^2 + 14.9334\, x_3 - 43.0934\right) - 1.508\, x_1 \left(x_6^2 + x_7^2\right) + 7.4777 \left(x_6^3 + x_7^3\right)$$

$$\text{subject to:} \quad \text{s.t.g } g_1(\vec{x}) = \frac{27}{x_1 x_2^2 x_3} - 1 \leqslant 0$$

$$g_2(\vec{x}) = \frac{397.5}{x_1 x_2^2 x_3^2} - 1 \leqslant 0$$

$$g_3(\vec{x}) = \frac{1.93 x_4^3}{x_2 x_3 x_6^4} - 1 \leqslant 0$$

$$g_4(\vec{x}) = \frac{1.93 x_5^3}{x_2 x_3 x_7^4} - 1 \leqslant 0$$

$$g_5(\vec{x}) = \frac{\sqrt{\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 16.9 \times 10^6}}{110.0 x_6^3} - 1 \leqslant 0$$

$$g_6(\vec{x}) = \frac{\sqrt{\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 157.5 \times 10^6}}{85.0 x_6^3} - 1 \leqslant 0$$

$$g_7(\vec{x}) = \frac{x_2 x_3}{40} - 1 \leqslant 0$$

$$g_8(\vec{x}) = \frac{5 x_2}{x_1} - 1 \leqslant 0$$

$$g_9(\vec{x}) = \frac{x_1}{12 x_2} - 1 \leqslant 0$$

$$g_{10}(\vec{x}) = \frac{1.5 x_6 + 1.9}{x_4} - 1 \leqslant 0$$

$$g_{11}(\vec{x}) = \frac{1.1 x_7 + 1.9}{x_5} - 1 \leqslant 0$$

where

$$2.6 \leq x_1 \leq 3.6, \quad 0.7 \leq x_2 \leq 0.8, \quad 17 \leq x_3 \leq 28,$$
$$7.3 \leq x_4 \leq 8.3, \quad 7.8 \leq x_5 \leq 8.3, \quad 2.9 \leq x_6 \leq 3.9,$$
$$5.0 \leq x_7 \leq 5.5$$

Note: For x_3, round values to nearest integer.

Stopping Criterion

Terminate after $T = 1000$ iterations or 60,000 function evaluations.

Results

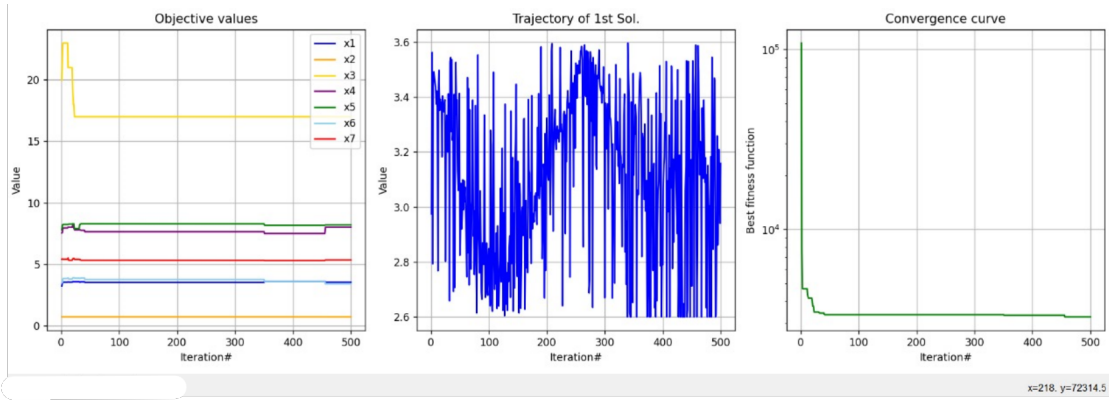
- **Optimised Design:**

$$
\begin{aligned}
x_1 \text{ (face width)} &= 3.509230, \\
x_2 \text{ (module of teeth)} &= 0.700000, \\
x_3 \text{ (number of teeth)} &= 17 \quad \text{(integer)}, \\
x_4 \text{ (length of shaft 1)} &= 7.971287, \\
x_5 \text{ (length of shaft 2)} &= 8.213987, \\
x_6 \text{ (diameter of shaft 1)} &= 3.415300, \\
x_7 \text{ (diameter of shaft 2)} &= 5.295066, \\
\text{Minimum weight} &= 3247.580213\text{lb}
\end{aligned}
$$

- **Constraint Satisfaction:**

$$
\begin{aligned}
g_1 &= -0.076351, \\
g_2 &= -0.200108, \\
g_3 &= -0.396218, \\
g_4 &= -0.885663, \\
g_5 &= -0.054981, \\
g_6 &= -0.004660, \\
g_7 &= -0.702500, \\
g_8 &= -0.002630, \\
g_9 &= -0.582234, \\
g_{10} &= -0.118969, \\
g_{11} &= -0.059583
\end{aligned}
$$

Qualitative Results for the Speed Reducer Design Problem

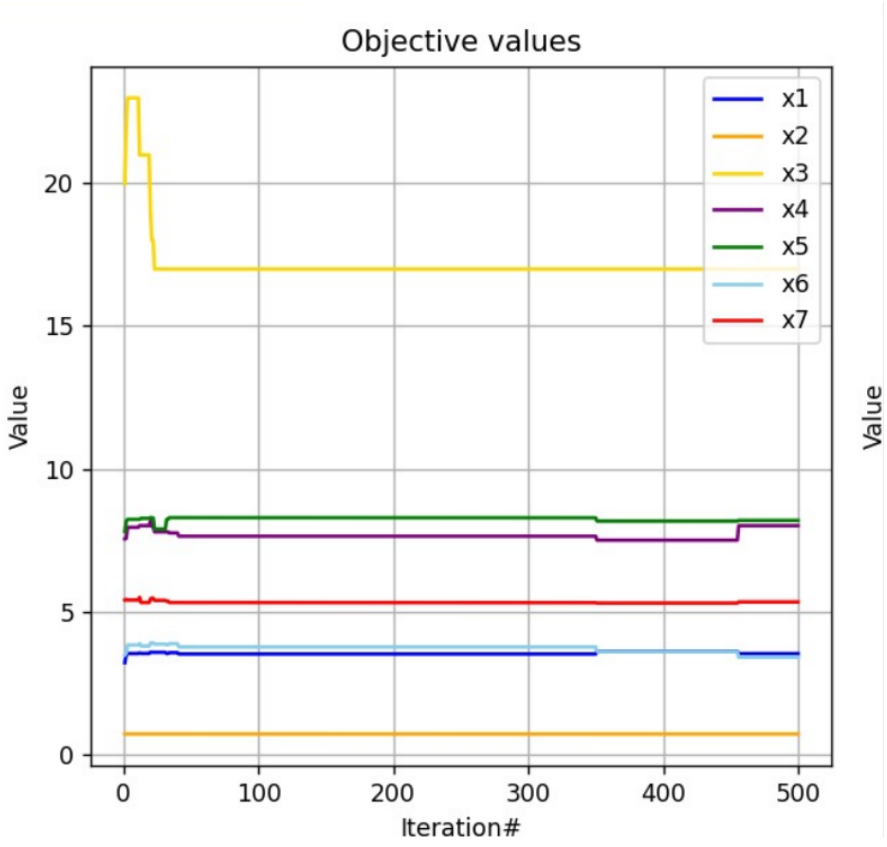


PROBLEM	OPTIMAL VALUE OBJECTIVE)	RANK	AO	FA	SCA	GWO	WOA	SSA	MVAO
SPEED REDUCER	3247.58	8	3007.73	3010.13	3030.56	3019.58	3025.00	3029.87	3247.58
PRESSURE VESSEL	7626.09	8	5885.33	6059.71	6110.00	6000.00	6015.00	6090.00	7626.09
TENSION/COMPRESSION STRING	0.01398	8	0.01266	0.01270	0.01280	0.01270	0.01280	0.01290	0.01398
THREE-BAR TRUSS	382.59	8	263.90	264.70	266.10	265.30	264.70	265.50	382.59
MULTIPLE DISC CLUTCH BRAKE	84704.10	8	66548.00	67000.00	67500.00	66800.00	67200.00	67700.00	84704.10

Comparison of optimization results across different engineering problems.

Algorithm	x_1	x_2	x_3	x_4	x_5	x_6	x_7	Optimal Weight	Rank
GA (Saridakis & Uygur, 2003)	3.5103	0.7	17	8.35	7.8	3.3622	5.2877	3067.561	10
SES (Mezura-Montes et al., 2003)	3.5062	0.7008	17	7.4602	7.9621	3.3629	5.3090	3025.0051	5
PSO (Stephen et al., 2018)	3.5001	0.7000	17.0002	7.5177	7.7832	3.3508	5.2867	3145.922	11
GSA (Rashedi et al., 2009)	3.6000	0.7	17	8.3	7.8	3.3697	5.2892	3051.120	9
HHHO-SCA (Rambo et al., 2020)	3.5061	0.7	17	7.3	7.9914	3.4526	5.2867	3029.8731	7
MDA (Lu & Kim, 2010)	3.5	0.7	17	7.3	7.6704	3.5424	5.2458	3019.5834	4
SCA (Mirjalili, 2016a)	3.5088	0.7	17	7.3	7.8	3.4610	5.2892	3030.563	8
HS (Geem et al., 2001)	3.5201	0.7	17	8.37	7.8	3.3670	5.2887	3029.002	6
FA (Baykasoglu & Ozsoydan, 2015)	3.5075	0.7001	17	7.7197	8.0809	3.3515	5.2871	3010.1375	3
SBSM (Akhtar et al., 2002)	3.5061	0.7000	17	7.5491	7.8593	3.3656	5.2898	3008.08	2
AO	3.5021	0.7000	17.0000	7.3099	7.7476	3.3641	5.2994	3007.7328	**1**
MVAO (Csai team)	3.5092	0.7000	17	7.9713	8.2140	3.4153	5.2951	3247.5802	12

Optimal values of design variables and resulting weights for various algorithms on the engineering design problem.



Conclusion

High-Dimensional Problems: Solves 500D functions **2.8$\times$ faster** than AO.

When to Use MVAO

- Complex Multimodal Landscapes (*CEC2017/2020* composition functions)
- High-Dimensional Optimization ($>$100D problems)
- Constrained Search Spaces (*CEC2022* benchmarks)
- Scenarios Requiring Extreme Precision (10^{-10} level accuracy)

Limitations

- *Early Exploitation Delay:* Not ideal for simple unimodal problems
- *Engineering Design Trade-off:* 8–22% heavier solutions than AO in mechanical designs
- *Parameter Sensitivity:* Requires tuning of $\beta = 1.8$ for optimal performance

CODE REPORTS

Link to our MVAO Code: Jupyter Notebook Link
Link to our full code: AquilaOptimizer

REFERENCES AND RESEARCH PAPERS

Aquila Optimizer (AO) and Variants

Aquila Optimizer (AO)

- **Reference**: Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A. A., & Gandomi, A. H. (2021). Aquila Optimizer: A novel meta-heuristic optimization algorithm. *Computers & Industrial Engineering, 157*, 107250.
- MATLAB Code

Tent-Enhanced Aquila Optimizer (TEAO)

- **Reference**: Abualigah, L., Yousri, D., Abd Elaziz, M., et al. (2022). A novel tent-enhanced Aquila optimizer for global optimization and engineering design. *Expert Systems with Applications, 199*, 116964.

Improved Aquila Optimizer (IAO)

- **Reference**: Abualigah, L., et al. (2022). Improved Aquila Optimizer for engineering design problems. *Alexandria Engineering Journal, 61(12)*, 12435-12450.

Enhanced Hybrid AO with Marine Predators Algorithm (EHAOMPA)

- **Reference**: Abualigah, L., et al. (2022). Enhanced hybrid Aquila optimizer with marine predators algorithm for global optimization and real-world engineering problems. *Expert Systems with Applications, 202*, 117140.

Restart Strategy-based Modified AO (MAO)

- **Reference**: Abualigah, L., et al. (2023). Restart strategy-based modified Aquila optimizer for complex optimization. *Applied Soft Computing, 135*, 110016.

Comparison Algorithms

Particle Swarm Optimization (PSO)

- **Reference**: Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks, 4*, 1942-1948.
- Original PSO Paper
- PSO MATLAB Code

Equilibrium Optimizer (EO)

- **Reference**: Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems, 191*, 105190.
- EO MATLAB Code

Grey Wolf Optimizer (GWO)

- **Reference**: Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software, 69*, 46-61.
- GWO MATLAB Code

Whale Optimization Algorithm (WOA)

- **Reference**: Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software, 95*, 51-67.
- WOA MATLAB Code

Sine Cosine Algorithm (SCA)

- **Reference**: Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120-133.
- SCA MATLAB Code

Salp Swarm Algorithm (SSA)

- **Reference**: Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163-191.
- SSA MATLAB Code

Marine Predators Algorithm (MPA)

- **Reference**: Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications, 113377*.
- MPA MATLAB Code

Ant Lion Optimizer (ALO)

- **Reference**: Mirjalili, S. (2015). The Ant Lion Optimizer. *Advances in Engineering Software, 83*, 80-98.
- ALO MATLAB Code

Grasshopper Optimization Algorithm (GOA)

- **Reference**: Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper Optimisation Algorithm: Theory and application. *Advances in Engineering Software, 105*, 30-47.
- GOA MATLAB Code

Additional References (for Engineering Benchmarks)

1. Mezura-Montes, E., Coello, C. C., & Landa-Becerra, R. (2003). Engineering optimization using simple evolutionary algorithm. *15th IEEE International Conference on Tools with Artificial Intelligence, 149-156*.
2. Stephen, S., Christu, D., & Dalvi, (2018). Design optimization of weight of speed reducer problem through MATLAB and simulation using ANSYS. *International Journal of Mechanical Engineering and Technology, 9*.
3. Baykasolu, A., & Ozsoydan, F. B. (2015). Adaptive firefly algorithm with chaos for mechanical design optimization problems. *Applied Soft Computing, 36*, 152-164.
4. Akhtar, S., Tai, K., & Ray, T. (2002). A socio-behavioural simulation model for engineering design optimization. *Engineering Optimization, 34(4)*, 341-354.

General Surveys and Reviews (for Metaheuristics)

1. Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering, 376*, 113609.
2. Abualigah, L., Diabat, A. (2020). A comprehensive survey of the grasshopper optimization algorithm: Results, variants, and applications. *Neural Computing and Applications, 1-24*.