MODIFIED GAZELLE OPTIMIZATION ALGORITHM USING ITERATIVE DEEPENING SEARCH

Prof Vijay Kumar Bohat,Pratham Bhushan,Ashish Gupta,Krishan

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

Abstract

In this study, we present an enhanced version of the Gazelle Optimization Algorithm (GOA) by integrating predators into the optimization framework. Inspired by the dynamic interplay between predators and prey in natural ecosystems, our Improved Gazelle Optimization Algorithm (MGOA) introduces a novel predator-prey dynamic, where both predators and prey engage in exploration and exploitation behaviors. During the exploration phase, predators actively pursue prey, simulating the predatory chase observed in nature. Conversely, in the exploitation phase, predators adopt a Brownian motion-like movement pattern, mimicking the wandering behavior as they exploit local resources. This predator-prey interaction enriches the algorithm's exploration-exploitation balance, enhancing its ability to navigate complex optimization landscapes. Through rigorous experimentation on benchmark functions and engineering design problems, we assess the performance of MGOA against state-of-the-art algorithms. Our results demonstrate the efficacy of MGOA in achieving superior solution quality, convergence speed, and robustness, showcasing its potential as a versatile optimization tool across diverse domains. This research extends the frontier of metaheuristic optimization techniques by integrating predator-prey dynamics, providing practitioners with a powerful approach to address complex optimization challenges effectively.

Keywords: Gazelle optimization algorithm; GOA ; MGOA ; Metaheuristics; Optimization problems; ; Nature-inspired ; Gazelle ; Population-based

1. Introduction

Optimization problems are pervasive across numerous fields, ranging from engineering and medicine to computer science and supply chains. These problems entail the quest for optimal solutions within specified constraints, a task compounded by their inherent complexity, nonlinearity, and multimodality. Traditional optimization approaches often fall short in addressing these challenges, prompting the adoption of metaheuristic algorithms, particularly those inspired by nature.

Acknowledging the efficacy of existing optimization methods within certain contexts, it is essential to recognize their limitations and the imperative of ongoing innovation. This study

Email address: vijay.bohat@nsut.ac.in (Prof Vijay Kumar Bohat,Pratham Bhushan,Ashish Gupta,Krishan)

contributes to this discourse by presenting an enhanced metaheuristic algorithm, aiming to bridge the gap between nature-inspired principles and practical optimization challenges. By advancing the field of metaheuristic optimization, this research seeks to provide more robust and effective problem-solving strategies for diverse real-world applications.

The Gazelle Optimization Algorithm (GOA) is one such population-based optimization algorithm inspired by gazelles' survival ability in their predator-dominated environment. Introduced in 2022 by Jeffrey O. Agushaka, Absalom E. Ezugwu, and Laith Abualigah. The technical contributions of this research are summarized as follows:

- This study introduces the Improved Gazelle Optimization Algorithm (MGOA), a novel population-based metaheuristic algorithm designed to emulate the survival instincts of gazelles in their natural environment.
- Drawing inspiration from the gazelle's ability to detect and evade predators, the MGOA incorporates two distinct phases representing exploration and exploitation.
- During the exploitation phase, the algorithm emulates the calm grazing behavior of gazelles, occurring either when predators are stalking them or have yet to be detected. Upon identification of a predator, the MGOA seamlessly transitions into the exploration phase, where the gazelle actively evades and outmaneuvers the threat to find refuge. These phases are iteratively executed, governed by termination criteria, to pursue optimal solutions for various optimization problems.
- The distinct stages of the MGOA are delineated and formalized through mathematical modeling.
- The efficacy and robustness of the MGOA are assessed by solving a comprehensive set of 60 benchmark functions from the CEC2014 and CEC2017 competitions.
- Furthermore, the MGOA's performance in solving real-world engineering design problems is evaluated across five distinct scenarios.
- To gauge its effectiveness, the optimization results obtained from the MGOA are benchmarked against those of six state-of-the-art algorithms and the original Gazelle Optimization Algorithm (GOA).

In this paper, we present a modified version of the Gazelle Optimization Algorithm, where the key enhancement lies in the refinement of its exploration and exploitation strategies. The core innovation introduced in this modified algorithm is the integration of the Improved Discrete Search (IDS) method for exploration. Unlike the conventional exploration techniques employed in the original GOA, the IDS method empowers the algorithm to delve into the search space with greater granularity, thereby augmenting its capacity to traverse intricate landscapes and unearth superior solutions.

The IDS method, a novel addition to the repertoire of exploration strategies, enables the algorithm to locally scrutinize the search space, meticulously probing for potential optima that might have eluded traditional exploration approaches. By leveraging the IDS method in conjunction with the existing exploitation mechanism based on Levy flight (RL), the modified algorithm orchestrates a sophisticated dance between comprehensive exploration and strategic exploitation.

The incorporation of IDS represents a paradigm shift in the exploration phase of the Gazelle Optimization Algorithm, endowing it with a more nuanced and adaptive exploration strategy.

This strategic refinement holds the promise of catapulting the algorithm's performance to new heights, enabling it to navigate complex landscapes with dexterity and finesse. Moreover, the synergy between IDS and Levy flight fosters a symbiotic relationship wherein each component complements and reinforces the efficacy of the other, culminating in a potent optimization framework.

In essence, this paper unveils a novel rendition of the Gazelle Optimization Algorithm that transcends the confines of conventional exploration strategies, ushering in a new era of optimization prowess. Through empirical validation and comparative analyses, we endeavor to demonstrate the efficacy and superiority of the modified algorithm in tackling real-world optimization challenges. Embracing the spirit of innovation and evolution, we invite the optimization community to embark on this transformative journey, charting new frontiers in the realm of optimization methodologies.

2. MODIFIED GAZELLE OPTIMIZATION ALGORITHM

2.1. Motivation

As mentioned earlier, many researchers have proposed novel or hybrid methods to find optimal solutions to these problems. However, it is believed that room still exists to find a better solution than the existing ones by developing a more robust and efficient algorithm. Therefore, this motivated our proposed work to develop a novel population-based algorithm inspired by the activities of the gazelles. Although studies have shown that quite a number of the proposed metaheuristic optimization methods are generally inspired by the foraging and hunting behaviors of animals in their natural habitat and more so despite the interesting survival life cycle of some animals in the wild such as the gazelle, there is no modeling that conveys the attractive adaptive survival strategy of the gazelle in nature as an optimization process. The gazelle knows that if it does not outrun and outmaneuver its predators, it becomes meat for the day. The gazelles are down in the food chain and one of the most hunted prey in their habitat. They are not considered endangered; this means they are doing something right. One of those right attributes inherent in the gazelles is their escape from predators. Studies show that the predators are only successful 34paper employs this information to develop a new metaheuristic algorithm that solves real-world problems using the gazelle's survival abilities.

- Improving the GOA aims to achieve better optimization performance in terms of convergence speed, solution quality, and robustness across different types of optimization problems. By enhancing the algorithm, researchers aim to provide users with a more competitive optimization tool.

- Many real-world optimization problems are highly complex, nonlinear, and non-convex. Improving the GOA allows it to better handle such challenging problems by exploring the search space more intelligently and effectively.

- The field of optimization algorithms is highly competitive, with new methods and improvements being constantly proposed. Enhancing the GOA ensures its competitiveness with other state-of-the-art algorithms, attracting more attention from researchers and practitioners.

2.2. Inspiration and behaviour of gazelle

The gazelles belong to the genus Gazella family. About 19 different gazelles exist globally, ranging from small gazelles like Thomson's and Speke's gazelle to the large gazelle-like the Dama gazelle. Gazelles are categorized as herbivores and primarily consume vegetation such as leaves, grasses, and shoots. They exhibit a social behavior, often forming large groups for security and social interaction, with group sizes reaching up to 700 individuals. Within these groups, there can be segregation based on sex, with females and their fawns forming smaller groups, while bachelor herds consist solely of males who provide defense and support. Their reproductive cycle involves a birth rate of one or two offspring, occurring twice annually, with a gestation period of approximately six months. Breeding typically coincides with the rainy season when food and water are plentiful.

As secondary consumers, gazelles occupy a critical position in the food chain, serving as primary prey for various predators including humans, cheetahs, jackals, hyenas, wild dogs, leopards, and lions. To evade predation, gazelles employ warning signals such as tail flicking, foot stomping, and leaping, known as "stotting," when nervous or excited. They also possess remarkable agility and speed, capable of reaching speeds up to 100 km/hr, which often enables them to outrun their predators. Despite their vulnerability, gazelles maintain a survival rate of 0.66 annually, indicating that predators are successful in predation only 0.34 of the time.

These survival strategies of gazelles serve as a basis for developing a model algorithm aimed at enhancing their survival in natural environments:-

- The most spectacular aspects are grazing and running from predators.
- Predator positions are changed with respect to best position of a gazelle.
- The grazing aspect of gazelle while predator is stalking it can be used for exploration phase
- The ability to outrun spotted predators to a haven can be used for exploration.

2.3. Mathematical model of original GOA

The GOA is a population-based optimization algorithm that uses randomly initialized search agents.The search agents consist of gazelles and predators in a ratio of 4:1.The search agents are defined as a n X d matrix of candidate solutions as defined in Eq. The GOA uses the problem's constraint of upper bound (UB) and lower bound (LB) to stochastically define the range of values that the population vector can take.

$$X = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & \dots & x_{1,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \dots & x_{i,j} & \dots & x_{i,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \dots & x_{N,j} & \dots & x_{N,d} \end{bmatrix}$$

where X is the matrix of the position vector of the candidate population, each position vector is stochastically generated.,$x_{i,j}$ is the randomly generated vector position of the ith population in the jth dimension, N represents the number of search agents, and d is the defined search space (dimension) of the optimization problem. Predators are randomly allocated twenty percent of the indices in the matrix representing predator populations.

$$x_{i,j} = l_j + \text{rand} \cdot (u_j - l_j), \quad i = 1, 2, \dots, N, \quad j = 1, 2, \dots, d,$$

In every iteration, each $x_{i,j}$ produces a candidate solution. The minimum solution is taken to be the best-obtained solution so far. It is said that the strongest or fittest gazelles in nature are more talented in spotting, informing others of the treats, and running from predators. The best-obtained solution so far is selected as the top gazelle to construct an Elite N X d matrix. This matrix is utilized for searching and determining the next step for the predators, taking into account the distance from each gazelle and the distance from the elite matrix. It is also employed in updating the positions of the gazelles.

$$X = \begin{bmatrix} x'_{1,1} & \dots & x'_{1,j} & \dots & x'_{1,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x'_{i,1} & \dots & x'_{i,j} & \dots & x'_{i,d} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x'_{N,1} & \dots & x'_{N,j} & \dots & x'_{N,d} \end{bmatrix}$$

The GOA considers both predators and gazelles as search agents constituting a population of 1:4 respectively.When a predator stalks a gazelle, it employs Brownian motion tactics during the exploitation phase.The predator stealthily approaches the gazelles, moving towards elite positions.Upon detection, both predator and prey flee in unison toward the haven. As the gazelles evade capture, the predator effectively explores the search space.

2.3.1. Brownian motion

In a standard Brownian motion, the displacement follows a Normal (Gaussian) probability distribution function with a mean of $\mu = 0$ and a variance of $\sigma^2 = 1$ at point x. It can be represented by the equation:

$$f(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where $f(x|\mu, \sigma^2)$ represents the probability density function of the normal distribution.

2.3.2. Levy Flight

A Levy flight is a type of random walk where step sizes are drawn from a heavy-tailed distribution, typically a Levy distribution. In optimization algorithms, Levy flights are often used to introduce long-range exploration, allowing the algorithm to escape local optima and search the solution space more efficiently.

1.Random Levy Flight Generation: In the improved code, the function levy() generates a Levy flight vector 'RL' of size 'SearchAgents˙no'-by-'dim', where each element is drawn from a Gaussian distribution with mean 0 and standard deviation 'scale'.

□ RL(i,j)=scale×N(0,1)

Where:

- RL(i,j) represents the j-th dimension of the Levy Flight for the i-th search agent.
- N(0,1) denotes a random number drawn from a standard normal distribution.

2.Scaling Factor:

- The 'scale' parameter controls the step size of the Levy flight. Adjusting this parameter influences the exploration-exploitation balance during the optimization process.
- Larger values of 'scale' lead to longer steps in the Levy flight, promoting exploration.
- Smaller values of 'scale' result in shorter steps, favoring exploitation.

2.4. Modelling the MGOA

This MGOA algorithm simulates the survival behavior of the gazelles. The optimization process comprises grazing without a predator and running from a spotted predator to a haven. Hence, the proposed MGOA algorithm optimization process consists of two phases.

2.4.1. Exploitation

This phase assumes the gazelles are grazing peacefully without a predator or while the predator is stalking the gazelles. In this phase, the Brownian motion characterized by uniform and controlled steps was used to effectively cover neighborhood areas of the domain. The gazelles are assumed to move in Brownian motion while grazing. The mathematical model of this behavior is as shown:

$$\text{stepsize}(i, j) = RL(i, j) \cdot (Elite(i, j) - RL(i, j) \cdot gazelle(i, j))$$

$$gazelle(i, j) = gazelle(i, j) + s \cdot R \cdot \text{stepsize}(i, j)$$

where gazelle˙i+1 is the solution of the next iteration, gazelle˙i is the solution at the current iteration, s denotes the grazing speed of the gazelles,RB vector is a vector containing random numbers representing the Brownian motion, R is a vector of uniform random numbers [0,1].

2.4.2. Exploration

The exploration phase involves the exploration of the search space to discover potential solutions. This phase utilizes various techniques, including Levy flights and Iterative Deepening Search (IDS), to efficiently explore the solution space.

Levy flights are implemented using the 'levy' function, which generates Levy random numbers to facilitate random exploration. These random steps help in exploring the solution space in a more effective manner, enabling the algorithm to escape local optima and discover new regions of interest.

Additionally, the code employs Iterative Deepening Search (IDS) for exploration. In the exploration using IDS, the algorithm iterates through a predefined depth (controlled by 'IDS˙depth') to search for better solutions. It systematically evaluates potential solutions within the specified depth and selects the best among them. In the exploration phase of the GOA5 algorithm, two different strategies are employed: exploitation and exploration using the Improved Dynamic Search (IDS) method. Here are the equations used in each of these strategies:

Exploration using IDS:

$$\text{new\_pos}(j) = \text{gazelle}(i, j) + RL(i, j) \cdot \mu \cdot \text{depth}$$

$$\text{new\_pos}(j) = \max(\min(\text{new\_pos}(j), ub), lb)$$

$$\text{new\_fitness} = f_{\text{obj}}(\text{new\_pos})$$

These equations represent the movement of the gazelles in the search space during the exploration phase, either exploiting promising regions (exploitation) or conducting a local search around the current position (exploration using IDS).

Furthermore, the algorithm adapts its exploration strategy based on the current iteration. During exploitation, it focuses on intensifying the search around promising solutions by utilizing the Elite (best solution found so far) and exploiting the current best solutions using random steps controlled by the parameter 's'. On the other hand, during exploration, it leverages Levy flights and IDS to systematically search for new solutions within the solution space.

Overall, by combining random exploration through Levy flights with systematic exploration using Iterative Deepening Search, the algorithm efficiently explores the solution space, enabling it to discover high-quality solutions and converge towards optimal or near-optimal solutions.

2.5. PSEUDO CODE FOR THE MODIFIED GOA

Function: GOA8(SearchAgents_no, Max_iter, lb, ub, dim, fobj)

Inputs:

- `SearchAgents_no`: Number of search agents (gazelles)
- `Max_iter`: Maximum number of iterations
- `lb`: Lower bound for each dimension
- `ub`: Upper bound for each dimension
- `dim`: Dimensionality of the problem
- `fobj`: Objective function to be minimized

Outputs:

- `Top_gazelle_pos`: Position of the top gazelle (best solution found)
- `Top_gazelle_fit`: Fitness value of the top gazelle
- `Convergence_curve`: Array containing the fitness value of the top gazelle at each iteration

Initialize variables:

- `Top_gazelle_pos`: Array of zeros with size $(1, \texttt{dim})$
- `Top_gazelle_fit`: Infinity
- `Convergence_curve`: Array of zeros with size $(1, \texttt{Max\_iter})$
- `stepsize`: Array of zeros with size $(\texttt{SearchAgents\_no}, \texttt{dim})$ to store step sizes
- `fitness`: Array of zeros with size $(\texttt{SearchAgents\_no}, 1)$ to store fitness values of gazelles
- `gazelle`: Population of gazelles initialized using initialization function (size: $\texttt{SearchAgents\_no} \times \texttt{dim}$)
- `Xmin`: Array with each element set to `lb`, replicated `SearchAgents_no` times (size: $\texttt{SearchAgents\_no} \times \texttt{dim}$)

- `Xmax`: Array with each element set to `ub`, replicated `SearchAgents_no` times (size: `SearchAgents_no` × `dim`)
- `Iter`: Iteration counter, initialized to 0
- `PSRs`: Parameter for applying random search (0.34)
- `S`: Parameter for scaling step size (0.88)

Main loop:
While `Iter` < `Max_iter`:

- Exploration and exploitation loop (repeat 4 times per iteration):
  - For $k = 1$ to 4:
    - **Evaluate top gazelle:**
      - For each gazelle in population:
      - Check and enforce boundary constraints
      - Evaluate fitness using `fobj` function
      - Update `Top_gazelle_pos` and `Top_gazelle_fit` if a better solution is found
    - **Maintain history of best solutions:**
      - If first iteration, set `fit_old` and `Prey_old` (copies of fitness and gazelle positions)
      - Update gazelle and fitness arrays based on improvement in fitness values
    - **Update search based on top gazelle:**
      - Create an Elite array by replicating `Top_gazelle_pos` (size: `SearchAgents_no` × `dim`)
      - Calculate a scaling factor (CF) based on iteration
      - Generate a Levy random number vector (RL) using `levy` function
      - For each gazelle and dimension:
      - Randomly choose between exploitation and exploration
      - Exploitation: Update position using RL and Elite
      - Exploration:
      - Use Iterative Deep Search (IDS) to find a better solution within a depth limit
      - If no improvement found, update position with random movement
    - **Update top gazelle again (after potential improvement during exploration):**
      - Check and enforce boundary constraints for all gazelles
      - Evaluate fitness using `fobj` function
      - Update `Top_gazelle_pos` and `Top_gazelle_fit` if a better solution is found
    - **Maintain history of best solutions (again):**
      - If first iteration, set `fit_old` and `Prey_old`
      - Update gazelle and fitness arrays based on improvement in fitness values
    - **Apply random search with probability PSRs:**

- Randomly generate a binary mask (U) based on PSRs
- Update gazelle positions using Xmin, Xmax, U, and a scaling factor

- **Update iteration counter:**
  - Iter = Iter + 1
- **Record fitness of top gazelle in** Convergence_curve**:**
  - Convergence_curve(Iter) = Top_gazelle_fit

Return results:

- Top_gazelle_pos
- Top_gazelle_fit
- Convergence_curve

Function: levy(SearchAgents_no, dim, scale)

Inputs:

- SearchAgents_no: Number of search agents
- dim: Dimensionality
- scale: Scaling factor for Levy distribution

Outputs:

- RL: Array of random numbers generated using the Levy distribution (size: SearchAgents_no $\times$ dim)

Implementation details omitted (refer to original code for details)

Function: initialization(PopSize, dim, ub, lb)

Inputs:

- PopSize: Population size (number of gazelles)
- dim: Dimensionality
- ub: Upper bound for each dimension
- lb: Lower bound for each dimension

Outputs:

- Population

3. Algorithms, test problems and comparison criteria

3.1. Algorithms and comparison criteria

This section presents the design of experiments conducted to evaluate the performance of MGOA. Thirty(30) CEC 2014 test functions,thirty(30) CEC 2017 test functions, and five (5) selected optimization design problems in the engineering domain were used to evaluate the MGOA. The results obtained were compared with that of the following algorithms:-

1. Gazelle Optimization Algorithm (GOA)
2. Sand Cat Optimization Algorithm (SCSO)
3. Arithmetic Otimization Algorithm (AOA)
4. Fire Hawk Optimizer (FHO)
5. Whale Optimization Algorithm (WOA)
6. Grey Wolf Optimizer (GWO)
7. Crayfish Optimization Algorithm (COA)

The population size and the maximum number of iterations are sensitive parameters and need to be tuned. For this study, the population size is tuned to 30, and the maximum number of iterations is tuned to 1000.The stop criterium is the maximum number of iterations. The number of independent runs for each algorithm is set at 50.

3.2. CEC 2014 AND CEC 2017 Benchmark functions

Typology	No.	Function name	Opt.
Unimodal Functions	1	Shifted and Rotated Bent Cigar	100
	2	Shifted and Rotated Sum of Different Power	200
	3	Shifted and Rotated Zakharov	300
Simple Multimodal Functions	4	Shifted and Rotated Rosenbrock	400
	5	Shifted and Rotated Rastrigin	500
	6	Shifted and Rotated Expanded Schaffer F6	600
	7	Shifted and Rotated Lunacek Bi-Rastrigin	700
	8	Shifted and Rotated Non-Continuous Rastrigin	800
	9	Shifted and Rotated Levy	900
	10	Shifted and Rotated Schwefel	1000
Hybrid Functions	11	Zakharov; Rosenbrock; Rastrigin	1100
	12	High-conditioned Elliptic; Modified Schwefel; Bent Cigar	1200
	13	Bent Cigar; Rosenbrock; Lunacek bi-Rastrigin	1300
	14	High-conditioned Elliptic; Ackley; Schaffer F7; Rastrigin	1400
	15	Bent Cigar; HGBat; Rastrigin; Rosenbrock	1500
	16	Expanded Schaffer F6; HGBat; Rosenbrock; Modified Schwefel	1600
	17	Katsuura; Ackley; Expanded Griewank plus Rosenbrock; Schwefel; Rastrigin	1700
	18	High-conditioned Elliptic; Ackley; Rastrigin; HGBat; Discus	1800
	19	Bent Cigar; Rastrigin; Griewank plus Rosenbrock; Weierstrass; Expanded Schaffer F6	1900
	20	HappyCat; Katsuura; Ackley; Rastrigin; Modified Schwefel; Schaffer F7	2000
Composition Functions	21	Rosenbrock; High-conditioned Elliptic; Rastrigin	2100
	22	Rastrigin; Griewank; Modified Schwefel	2200
	23	Rosenbrock; Ackley; Modified Schwefel; Rastrigin	2300
	24	Ackley; High-conditioned Elliptic; Griewank; Rastrigin	2400
	25	Rastrigin; HappyCat; Ackley; Discus; Rosenbrock	2500
	26	Expanded Schaffer F6; Modified Schwefel; Griewank; Rosenbrock; Rastrigin	2600
	27	HGBat; Rastrigin; Modified Schwefel; Bent Cigar; High-conditioned Elliptic; Expanded Schaffer F6	2700
	28	Ackley; Griewank; Discus; Rosenbrock; HappyCat; Expanded Schaffer F6	2800
	29	$f_{15}; f_{16}; f_{17}$	2900
	30	$f_{15}; f_{18}; f_{19}$	3000

Figure 1: CEC2017 BENCHMARK FUNCTIONS

No.	Functions	$f(x^*)$	NO.	Functions	$f(x^*)$
F01	Rotated High Conditioned Elliptic Function	100	F16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
F02	Rotated Bent Cigar Function	200	F17	Hybrid Function 2	1700
F03	Rotated Discus Function	300	F18	Hybrid Function 2	1800
F04	Shifted and Rotated Rosenbrock's Function	400	F19	Hybrid Function 3	1900
F05	Shifted and Rotated Ackley's Function	500	F20	Hybrid Function 4	2000
F06	hifted and Rotated Weierstrass Function	600	F21	Hybrid Function 5	2100
F07	Shifted and Rotated Griewank's Function	700	F22	Hybrid Function 6	2200
F08	Shifted Rastrigin's Function	800	F23	Composition Function 1	2300
F09	Shifted and Rotated Rastrigin's Function	900	F24	Composition Function 2	2400
F10	Shifted Schwefel's Function	1000	F25	Composition Function 3	2500
F11	Shifted and Rotated Schwefel's Function	1100	F26	Composition Function 4	2600
F12	Shifted and Rotated Katsuura Function	1200	F27	Composition Function 5	2700
F13	Shifted and Rotated HappyCat Function	1300	F28	Composition Function 6	2800
F14	Shifted and Rotated HGBat Function	1400	F29	Composition Function 7	2900
F15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500	F30	Composition Function 8	3000

Search Range: $[-100, 100]^D$

Figure 2: CEC2014 BENCHMARK FUNCTIONS

3.3. Engineering Problems

F1:Speed Reducer Design Theory

A speed reducer, also known as a gearbox, is a mechanical device used to reduce the speed of an input shaft by a certain ratio to produce a slower output speed. This is commonly used in machinery where the input needs to be converted to a different output speed or torque.

Gear Ratio

The gear ratio R of a speed reducer is defined as the ratio of the number of teeth on the output gear (N_{out}) to the number of teeth on the input gear (N_{in}):

$$R = \frac{N_{\text{out}}}{N_{\text{in}}}$$

Speed Relation

The speed relation of a gear system is inversely proportional to the gear ratio. If the gear ratio is R, then the output speed (ω_{out}) is related to the input speed (ω_{in}) by:

$$\omega_{\text{out}} = \frac{\omega_{\text{in}}}{R}$$

Torque Relation

The torque relation of a gear system is directly proportional to the gear ratio. If the gear ratio is R, then the output torque (T_{out}) is related to the input torque (T_{in}) by:

$$T_{\text{out}} = R \cdot T_{\text{in}}$$

Efficiency

Gear systems are not 100% efficient, and they introduce losses due to friction and other factors. The efficiency (η) of a gearbox is typically less than 1 (often expressed as a percentage), and it affects the output speed and torque:

$$\text{Output Power} = \eta \cdot \text{Input Power}$$

$$\eta = \frac{\text{Output Power}}{\text{Input Power}}$$

Power

The power (P) in a gearbox can be calculated using:

$$P = T \cdot \omega$$

where:

P = Power (in watts)

T = Torque (in Newton-meters, Nm)

ω = Angular velocity (in radians per second, rad/s)

F5:Design of Gear Train

Gear Ratio Calculation

Let's design a gear train with the following parameters:

- Input Gear: 20 teeth
- Intermediate Gear: 30 teeth
- Output Gear: 40 teeth

The gear ratio for this gear train can be calculated as follows:

$$R = \frac{N_{\text{output}}}{N_{\text{input}}} = \frac{40}{20} = 2$$

Speed and Torque Relations

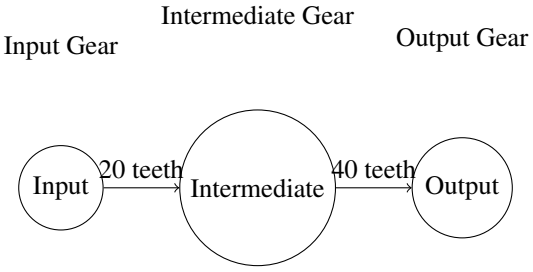
The speed relation for the gear train is:

$$\omega_{\text{output}} = \frac{\omega_{\text{input}}}{R}$$

The torque relation for the gear train is:

$$T_{\text{output}} = R \cdot T_{\text{input}}$$

Gear Train Diagram



F10:Corrugated Bulkhead Design

Geometry and Dimensions

Let's design a corrugated bulkhead with the following dimensions:

- Height of Bulkhead: $H = 2$ meters
- Depth of Corrugation: $D = 0.1$ meters
- Wave Length: $\lambda = 0.5$ meters

The profile of the corrugated bulkhead can be represented as:

$$y(x) = D \cdot \cos\left(\frac{2\pi}{\lambda}x\right)$$

Curvature and Stress

The curvature (κ) of the bulkhead can be calculated as:

$$\kappa = \frac{d^2y}{dx^2} = -\left(\frac{2\pi}{\lambda}\right)^2 D \cdot \cos\left(\frac{2\pi}{\lambda}x\right)$$

The stress (σ) due to bending can be found using the equation:

$$\sigma = E \cdot \kappa \cdot y$$

where E is the Young's modulus of the material.

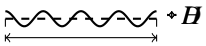
Example Calculation

Suppose we have a material with Young's modulus $E = 2 \times 10^{11}$ Pa.
Let's calculate the stress at $x = 0.25$ meters:

$$\kappa = -\left(\frac{2\pi}{0.5}\right)^2 \times 0.1 \times \cos\left(\frac{2\pi}{0.5} \times 0.25\right)$$

$$\sigma = (2 \times 10^{11}\,\text{Pa}) \times \kappa \times 0.1 \times \cos\left(\frac{2\pi}{0.5} \times 0.25\right)$$

Corrugated Bulkhead Profile



Corrugated Bulkhead Profile

F11:Car Side Impact Design

Force Calculation

Let's design the side impact protection for a car. The force (F) experienced during a side impact can be estimated using Newton's second law:

$$F = m \cdot a$$

where:

- m = mass of the car
- a = acceleration due to impact

Deceleration Distance

The deceleration distance (d) can be calculated using the equation of motion:

$$d = \frac{v^2}{2a}$$

where:

- v = initial velocity of the car

Design Requirements

Let's assume:

- Car mass (m) = 1500 kg
- Initial velocity (v) = 30 m/s
- Required deceleration (a) = 5 m/s^2

Using the equations above, we can calculate the force and deceleration distance required for the side impact protection.

Force Calculation Example

Substituting the values into the force equation:

$$F = (1500\,\text{kg}) \times (5\,\text{m/s}^2) = 7500\,\text{N}$$

Deceleration Distance Calculation Example

Substituting the values into the deceleration distance equation:

$$d = \frac{(30\,\text{m/s})^2}{2 \times 5\,\text{m/s}^2} = 90\,\text{m}$$

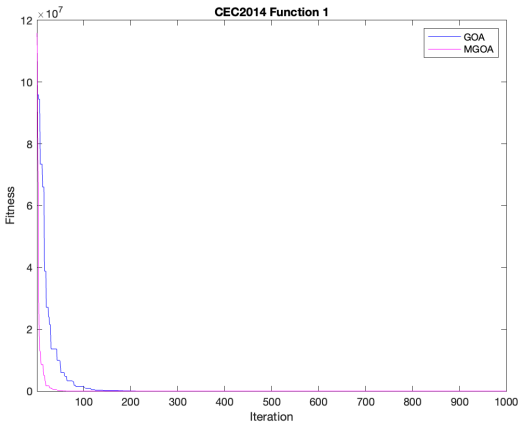
4. Experimental results and discussion

4.1. Performance Comparison of GOA and MGOA on CEC2014

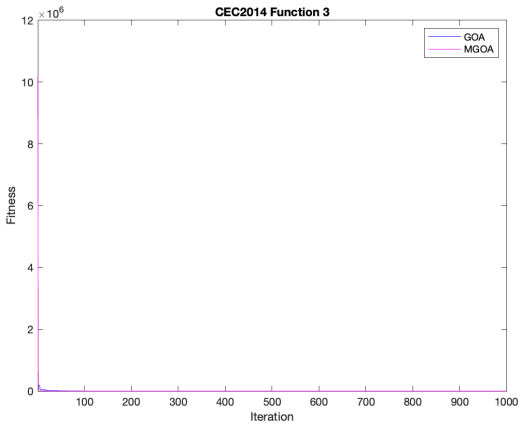
Table 1: CEC 2014 Results

Function	GOA Avg	MGOA Avg	P_VALUE	GOA Std dev	MGOA Std dev
F1	100.0002227	100.00	3.31×10^{-20}	3.46×10^{-04}	0.00×10^{00}
F2	200.009432	200.00	3.31×10^{-20}	8.86×10^{-03}	0.00×10^{00}
F3	300.0000003	300.00	3.31×10^{-20}	3.25×10^{-07}	0.00×10^{00}
F4	403.8229488	404.34	5.35×10^{-06}	1.25×10^{01}	1.03×10^{01}
F5	520.1215684	504.05	7.50×10^{-18}	2.69×10^{-01}	6.07×10^{00}
F6	600.5240172	600.00	7.44×10^{-12}	4.38×10^{-01}	3.15×10^{-01}
F7	700.1637536	715.44	1.73×10^{-15}	8.70×10^{-02}	1.67×10^{-02}
F8	802.9089046	805.31	8.98×10^{-15}	1.52×10^{00}	7.08×10^{-01}
F9	906.9275369	900.00	3.48×10^{-05}	2.37×10^{00}	2.20×10^{00}
F10	1059.508291	1096.51	6.89×10^{-15}	2.54×10^{01}	1.07×10^{01}
F11	1395.35269	1100.26	7.55×10^{-06}	1.30×10^{02}	9.53×10^{01}
F12	1200.278103	1201.87	3.32×10^{-17}	8.25×10^{-02}	2.69×10^{-02}
F13	1300.159784	1302.31	2.69×10^{-11}	3.15×10^{-02}	2.18×10^{-02}
F14	1400.134723	1400.81	1.19×10^{-06}	3.89×10^{-02}	3.26×10^{-02}
F15	1501.158673	1500.26	1.83×10^{-15}	2.76×10^{-01}	1.74×10^{-01}
F16	1602.24176	1601.33	1.75×10^{-06}	3.64×10^{-01}	4.12×10^{-01}
F17	1752.560554	1720.69	7.07×10^{-18}	1.63×10^{01}	3.99×10^{00}
F18	1801.946356	1800.27	4.46×10^{-17}	8.21×10^{-01}	2.62×10^{-01}
F19	1901.364038	1900.25	7.55×10^{-17}	2.93×10^{-01}	1.77×10^{-01}
F20	2001.880827	2002.32	8.46×10^{-18}	6.40×10^{-01}	1.16×10^{-01}
F21	2107.074944	2107.07	7.07×10^{-18}	5.54×10^{00}	2.48×10^{-01}
F22	2215.523085	2215.52	1.43×10^{-16}	7.44×10^{00}	3.53×10^{00}
F23	2629.457475	2642.66	4.73×10^{-20}	4.66×10^{01}	4.66×10^{01}
F24	2512.020663	2510.82	0.001536244878	3.30×10^{00}	2.54×10^{00}
F25	2641.344042	2622.86	1.75×10^{-06}	2.82×10^{01}	1.29×10^{01}
F26	2700.156546	2700.11	1.39×10^{-11}	3.79×10^{-02}	2.26×10^{-02}
F27	2860.332795	2780.79	7.05×10^{-07}	1.89×10^{02}	1.45×10^{02}
F28	3169.447045	3169.51	0.03881338257	3.64×10^{00}	5.63×10^{00}
F29	3115.822734	3104.92	4.67×10^{-08}	2.28×10^{01}	4.25×10^{01}
F30	3494.990892	3459.11	4.21×10^{-17}	2.29×10^{01}	1.73×10^{01}

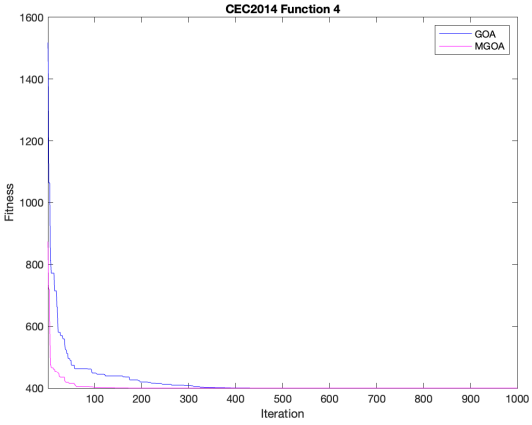
4.1.1. Convergence Curves for CEC2014



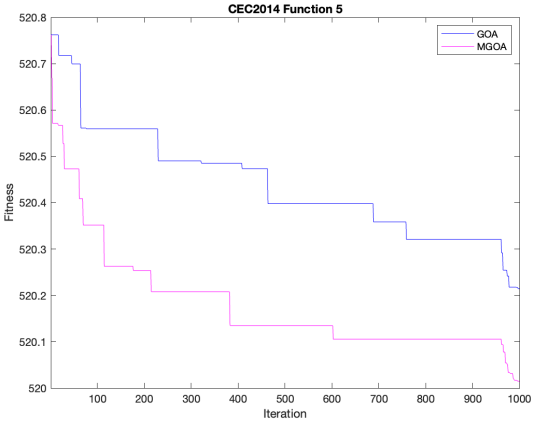
CEC2014-F1



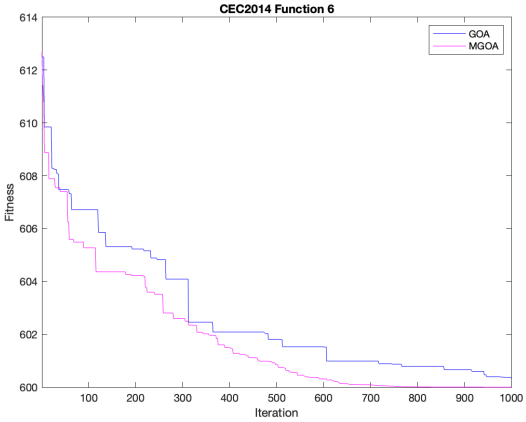
CEC2014-F3



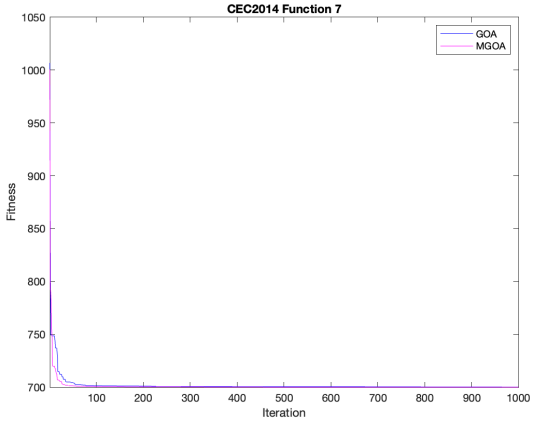
CEC2014-F4



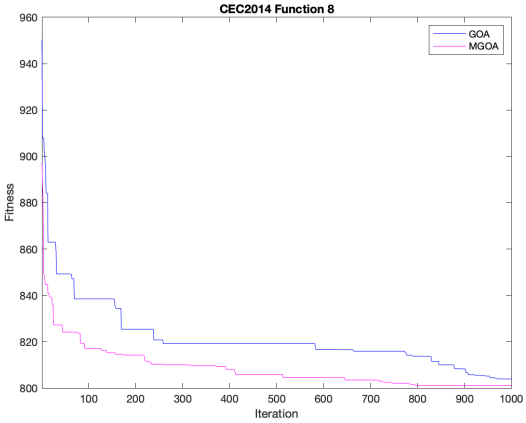
CEC2014-F5



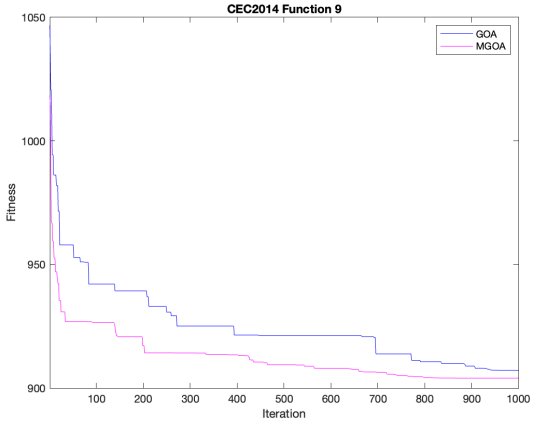
CEC2014-F6



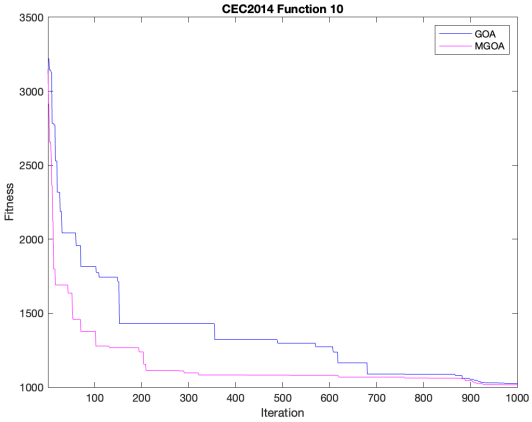
CEC2014-F7



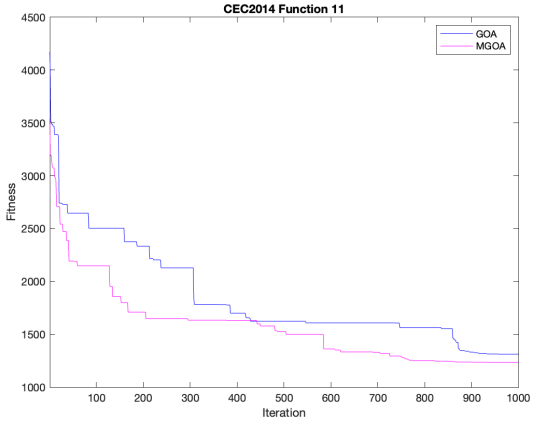
CEC2014-F8



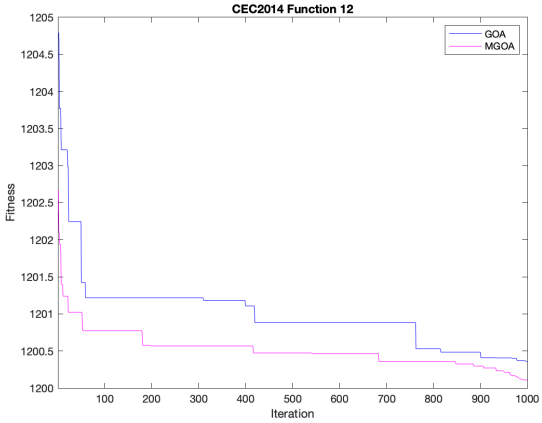
CEC2014-F9



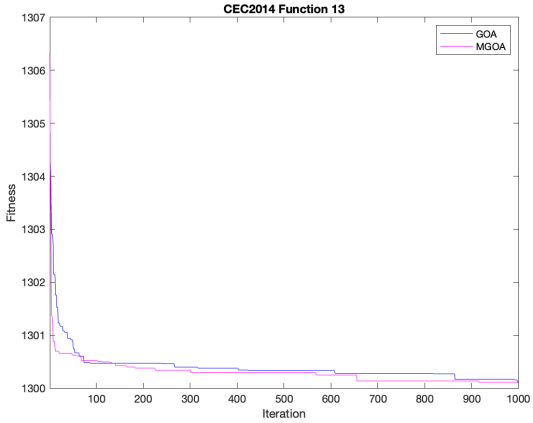
CEC2014-F10



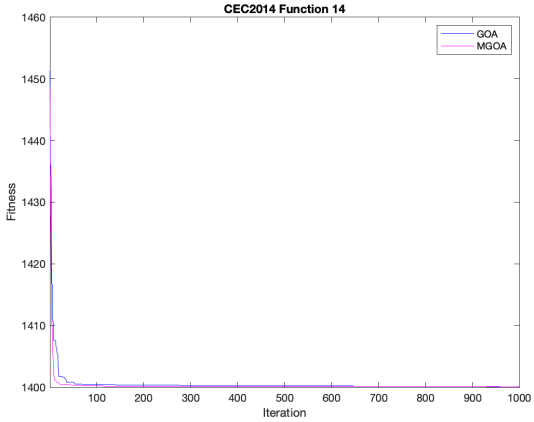
CEC2014-F11



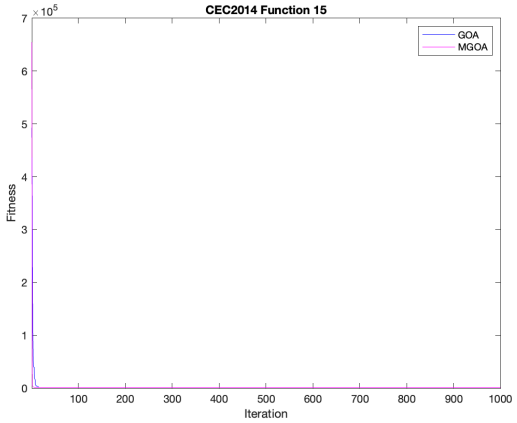
CEC2014-F12



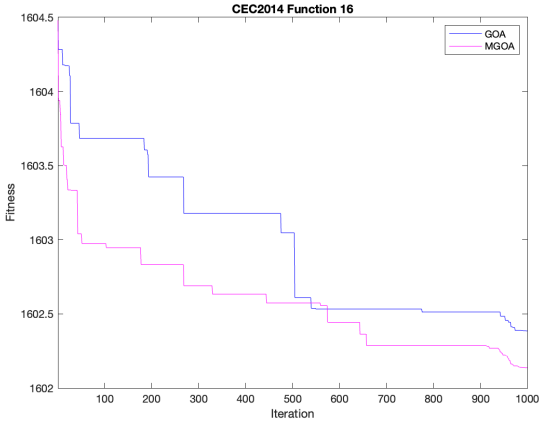
CEC2014-F13



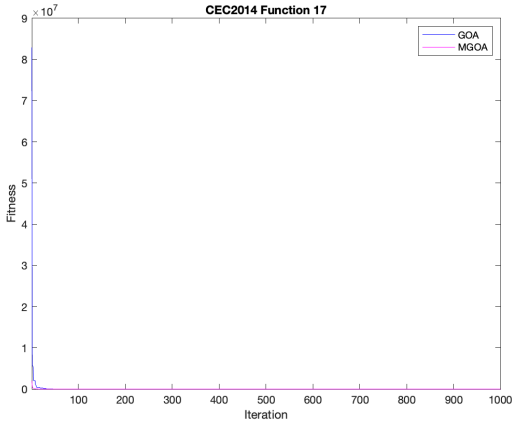
CEC2014-F14



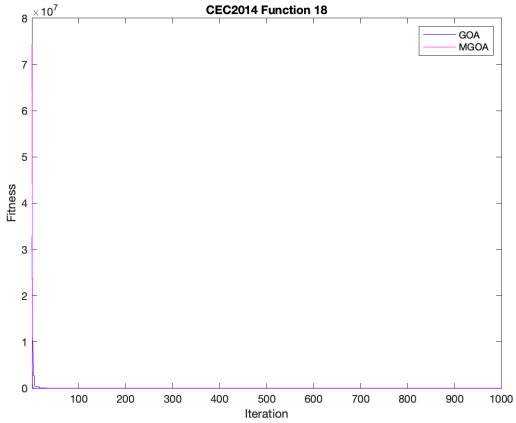
CEC2014-F15



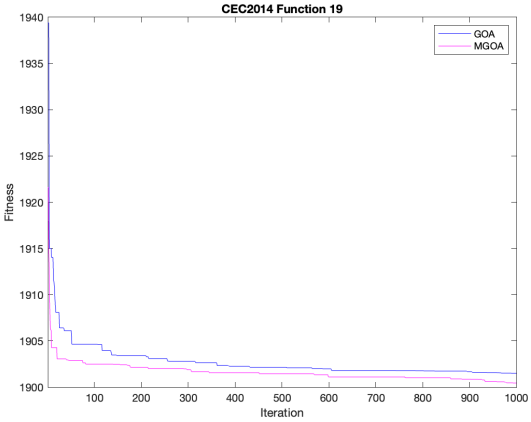
CEC2014-F16



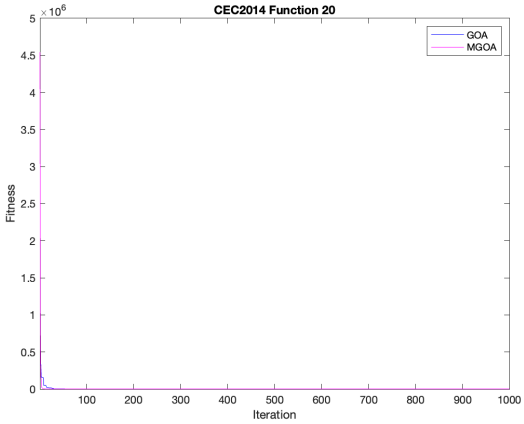
CEC2014-F17



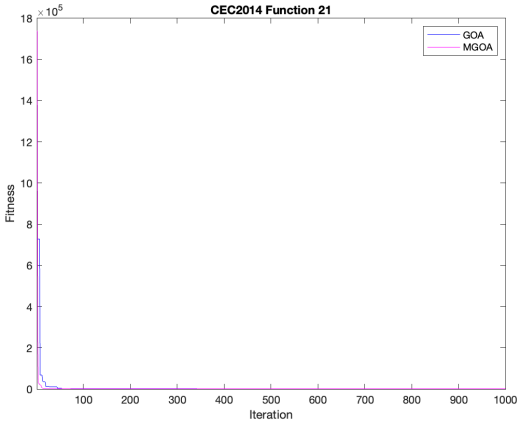
CEC2014-F18



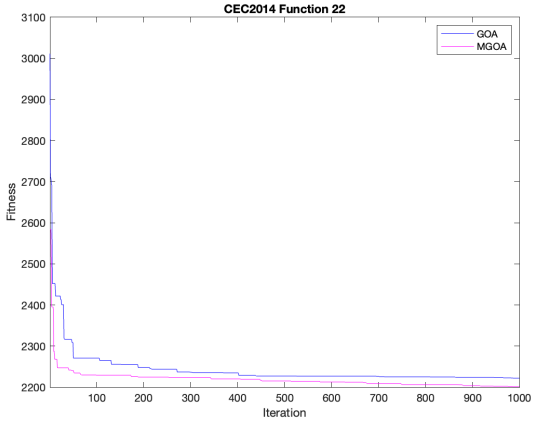
CEC2014-F19



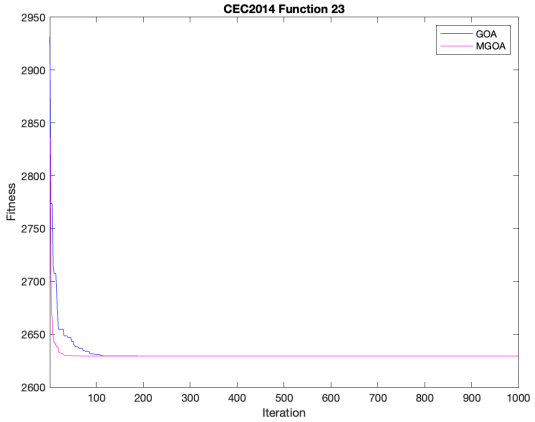
CEC2014-F20



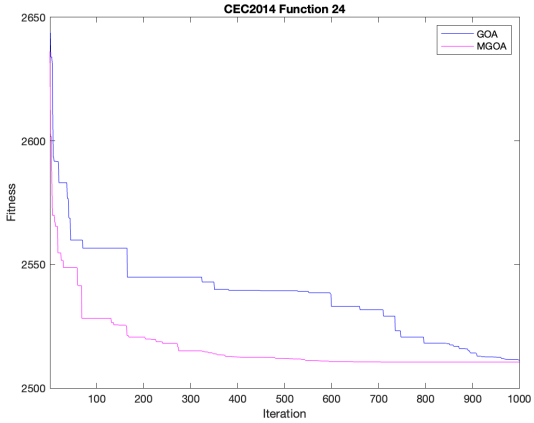
CEC2014-F21



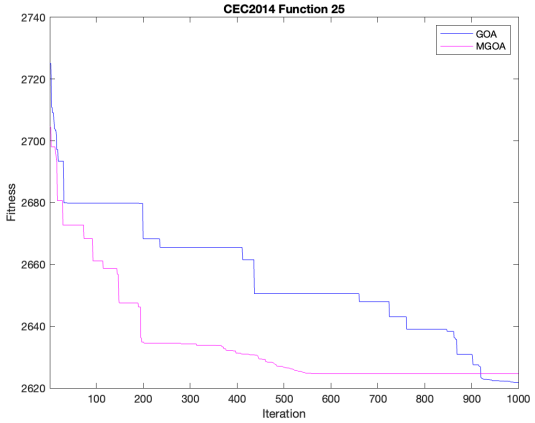
CEC2014-F22



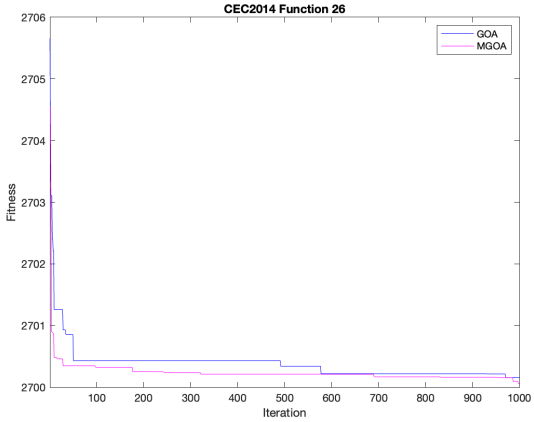
CEC2014-F23



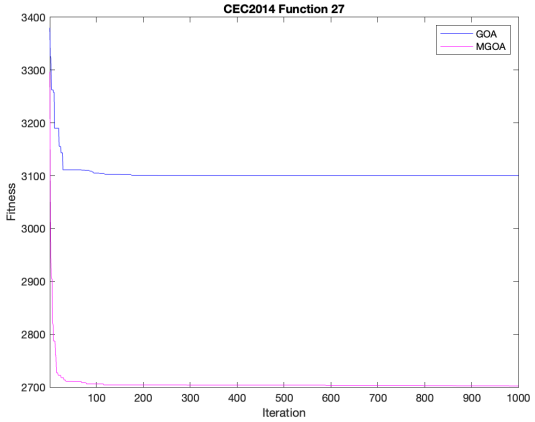
CEC2014-F24



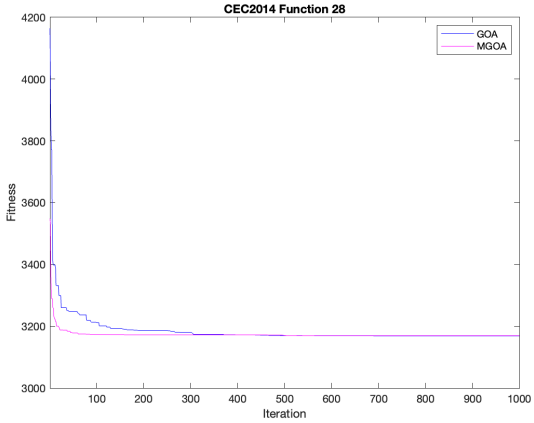
CEC2014-F25



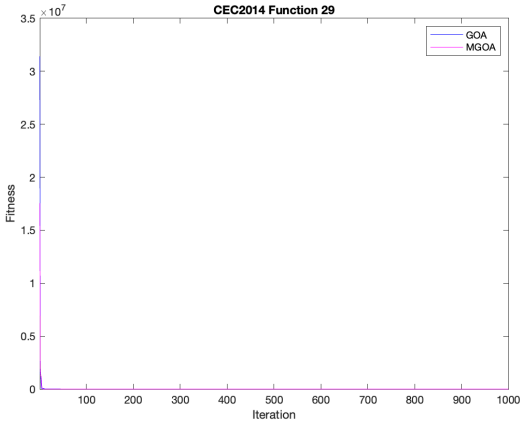
CEC2014-F26



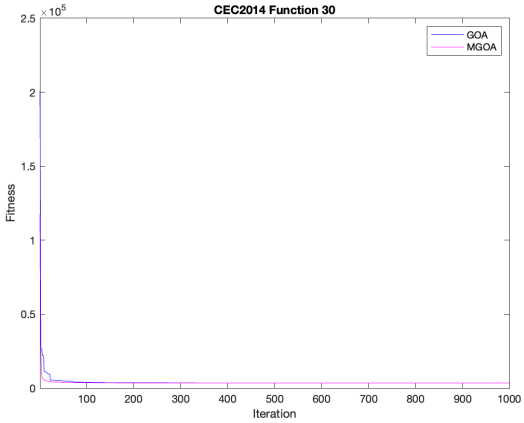
CEC2014-F27



CEC2014-F28



CEC2014-F29


CEC2014 Function 30
×10^5
2.5
2
1.5
1
0.5
0
Fitness
100
200
300
400
500
600
700
800
900
1000
Iteration
GOA
MGOA

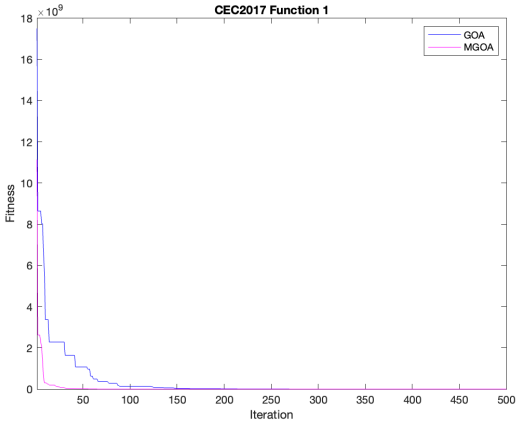
CEC2014-F30

4.2. Performance Comparison of GOA and MGOA on CEC2017

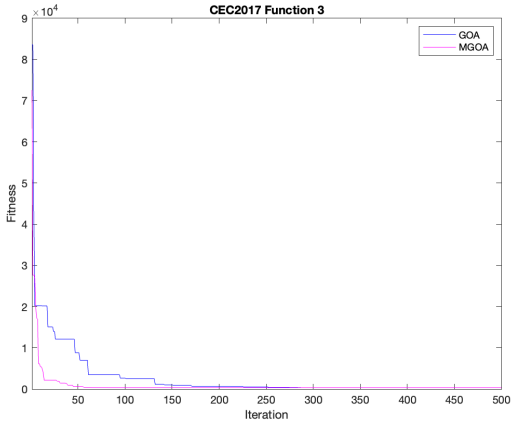
Table 2: CEC 2017 Results

Function	GOA Average	MGOA Average	p value	GOA std dev	MGOA std dev
F1	100.0169667	100	3.31×10^{-20}	0.0226923362	0
F2	0	0	0	0	0
F3	300.0000003	300	3.31×10^{-20}	2.13×10^{-07}	0
F4	405.99889	404.3431686	0.0001712532529	12.31687099	6.81917687
F5	506.3691918	504.0514196	7.03×10^{-07}	1.783013287	1.868525347
F6	600.0758535	600.0004968	7.07×10^{-18}	0.03979970257	0.001435475292
F7	721.474513	715.4423758	2.38×10^{-14}	3.782322515	2.094971953
F8	806.6523015	805.313288	0.00126506368	2.62922165	2.02110426
F9	900.0207003	900	3.31×10^{-20}	0.001492634779	0
F10	1212.479987	1096.512604	2.22×10^{-06}	119.9112966	76.20987561
F11	1102.723902	1100.258689	1.19×10^{-17}	0.9970198207	0.4845183547
F12	1321.495389	1201.867782	9.54×10^{-18}	49.86599288	0.5962528603
F13	1308.240615	1302.312182	7.07×10^{-18}	1.551048872	1.84829284
F14	1408.735054	1400.814926	1.31×10^{-17}	4.764745596	0.9872936355
F15	1502.076476	1500.261408	7.07×10^{-18}	0.9104051924	0.2434103126
F16	1603.2353	1601.329557	2.86×10^{-15}	1.222092442	0.8281845806
F17	1729.805887	1720.688394	5.33×10^{-16}	4.196571345	5.640221661
F18	1803.158898	1800.270125	3.19×10^{-16}	1.45295606	0.2199989783
F19	1901.741388	1900.249767	1.14×10^{-17}	0.3647372702	0.4203352752
F20	2026.829577	2002.324396	7.07×10^{-18}	7.135512153	3.73595461
F21	Inf	Inf	NaN	0	0
F22	Inf	Inf	NaN	0	0
F23	2655.39604	2642.663616	0.001330566117	7.609805541	7.946096936
F24	2651.467866	2554.815628	2.09×10^{-07}	0.196922716	29.28154697
F25	2894.953086	2888.837198	1.22×10^{-13}	45.18542784	45.66831271
F26	2888.02863	2866	4.97×10^{-12}	47.28545188	43.27570477
F27	3127.074458	3122.20881	0.002771717661	79.30068395	26.58166509
F28	3102.93909	3105.874297	3.08×10^{-12}	84.24499041	27.53639586
F29	3149.705945	3132.551483	1.58×10^{-13}	73.30003877	91.27234014
F30	3230.054232	3218.306967	2.32×10^{-09}	0.4027896303	0.7285246031

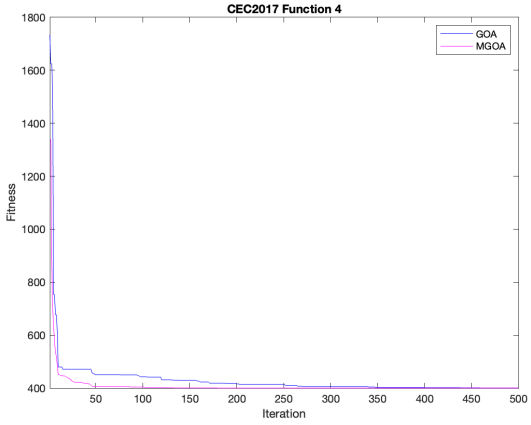
4.2.1. Convergence Curves for CEC2017



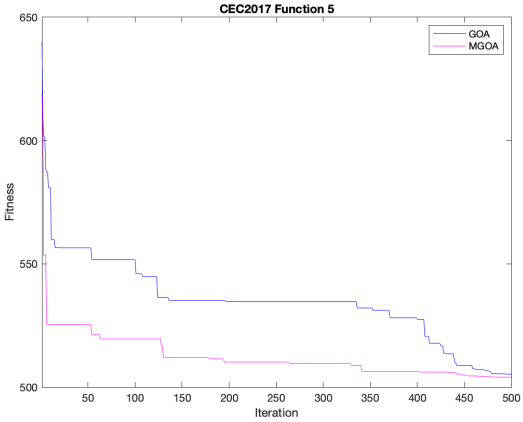
CEC2017-F1



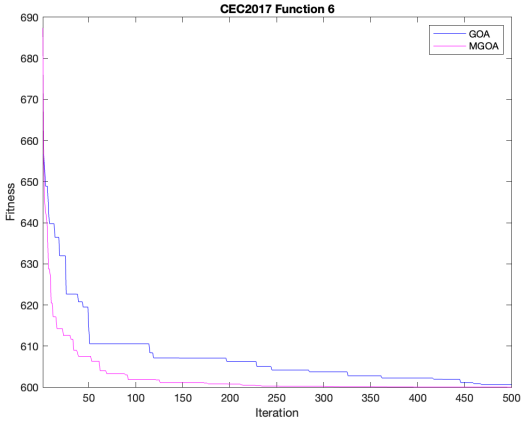
CEC2017-F3



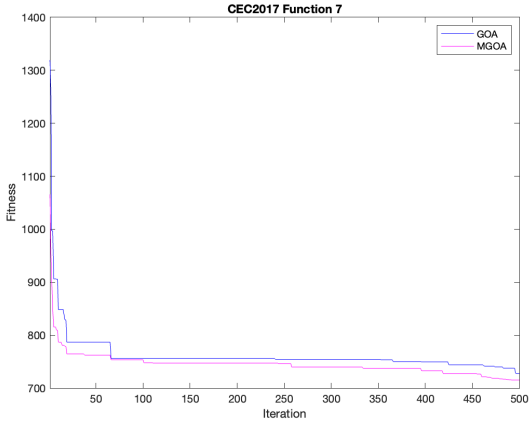
CEC2017-F4



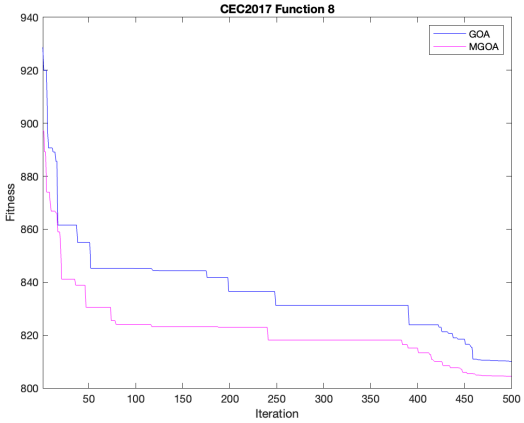
CEC2017-F5



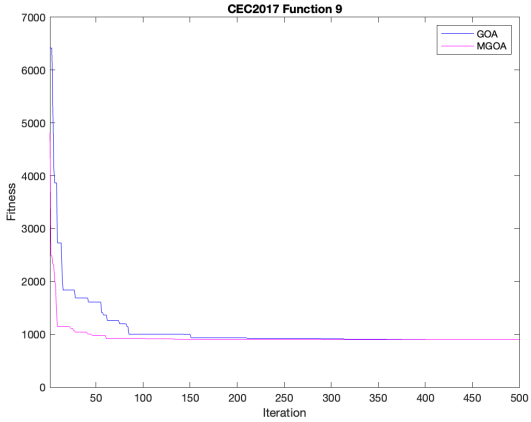
CEC2017-F6



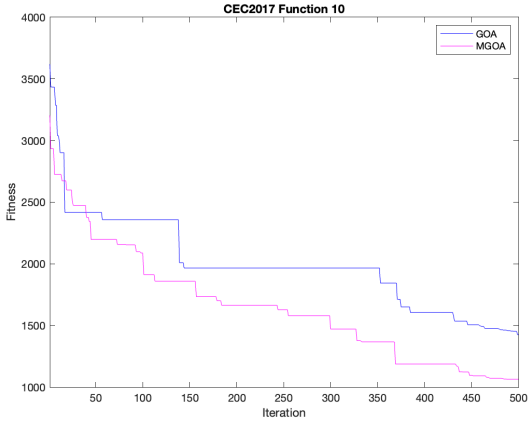
CEC2017-F7



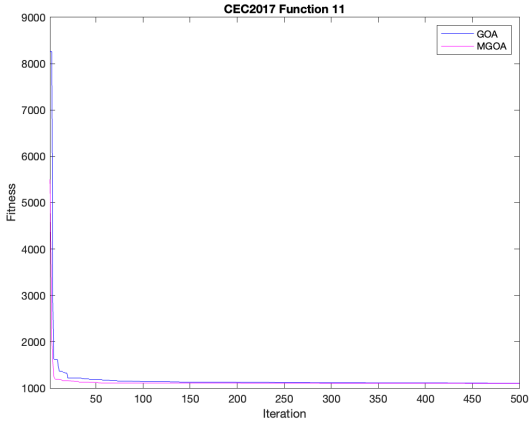
CEC2017-F8



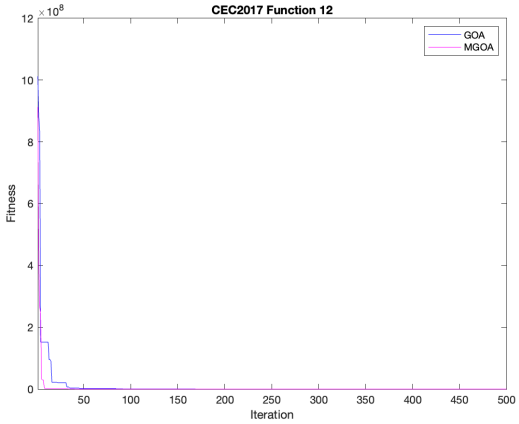
CEC2017-F9



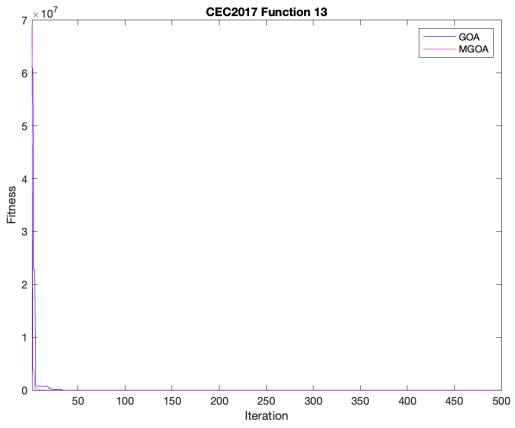
CEC2017-F10



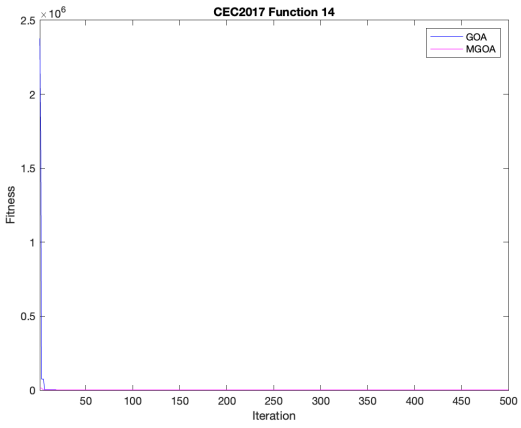
CEC2017-F11



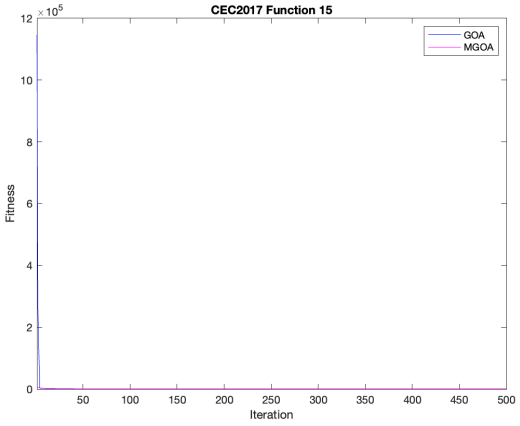
CEC2017-F12



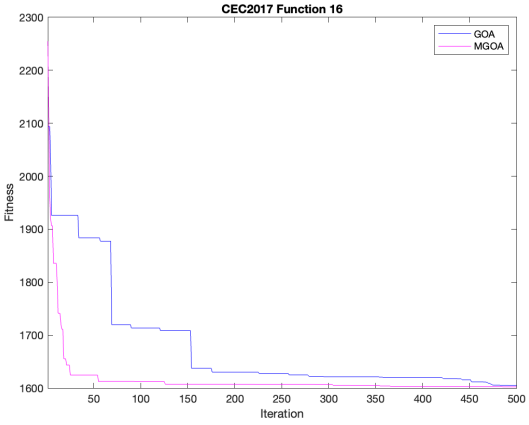
CEC2017-F13



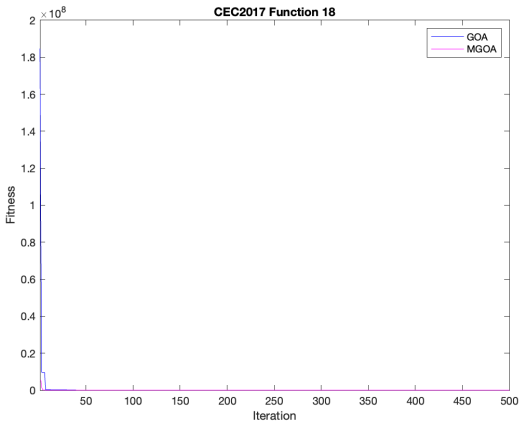
CEC2017-F14



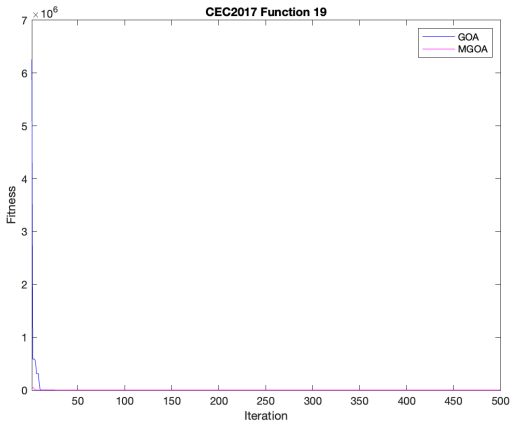
CEC2017-F15



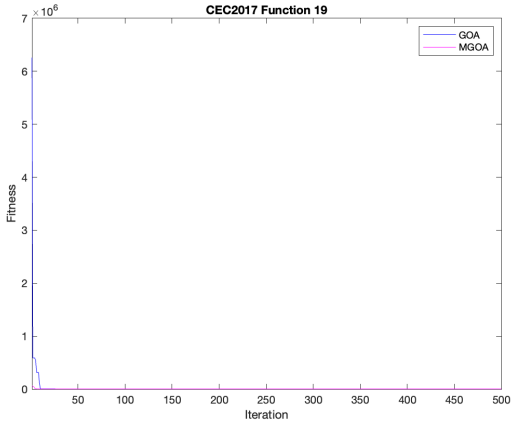
CEC2017-F16



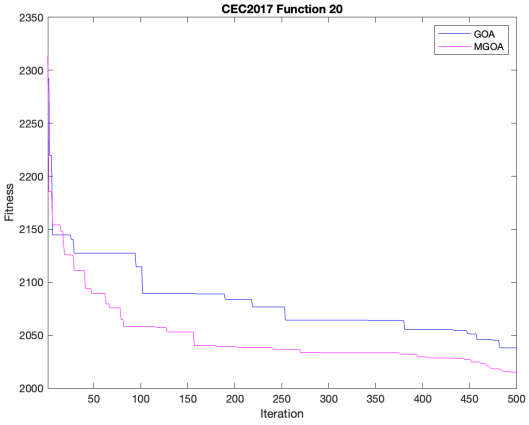
CEC2017-F17



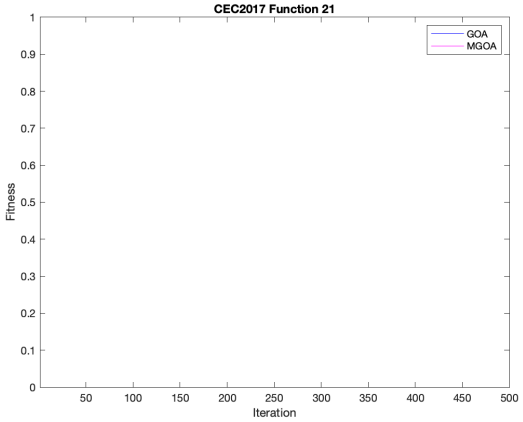
CEC2017-F18



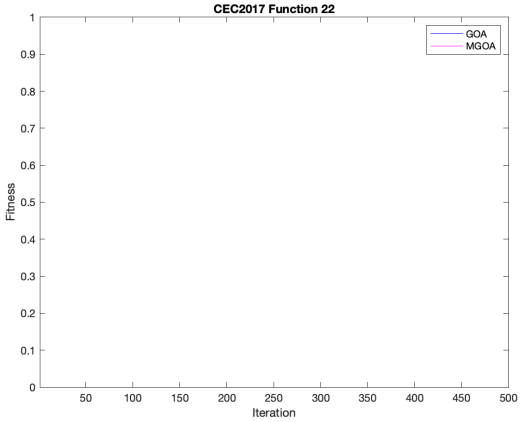
CEC2017-F19



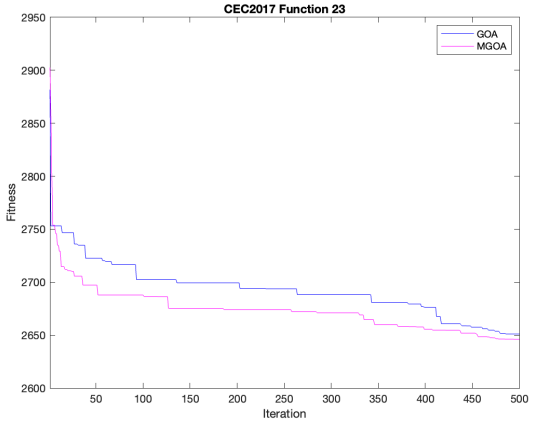
CEC2017-F20



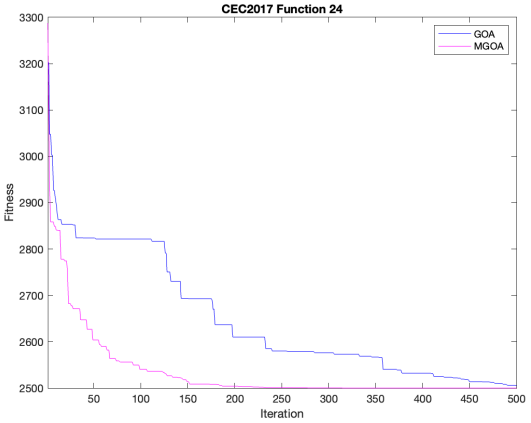
CEC2017-F21



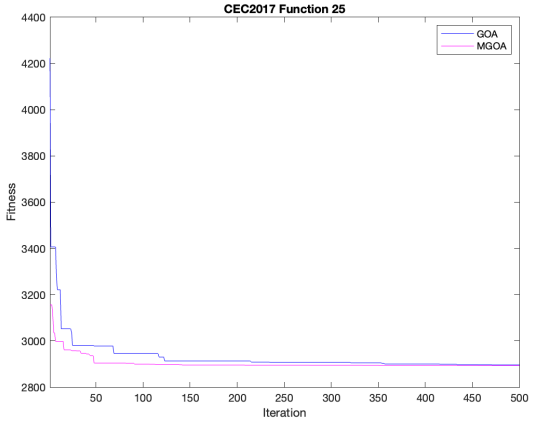
CEC2017-F22



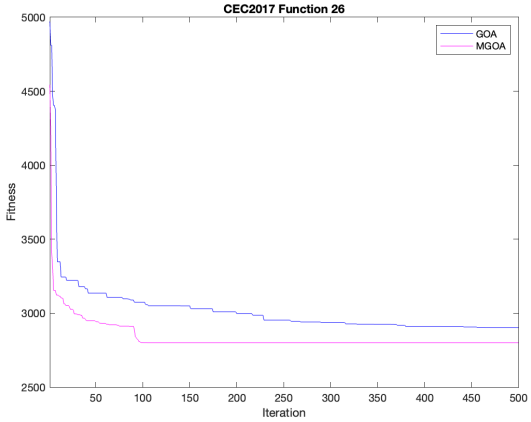
CEC2017-F23



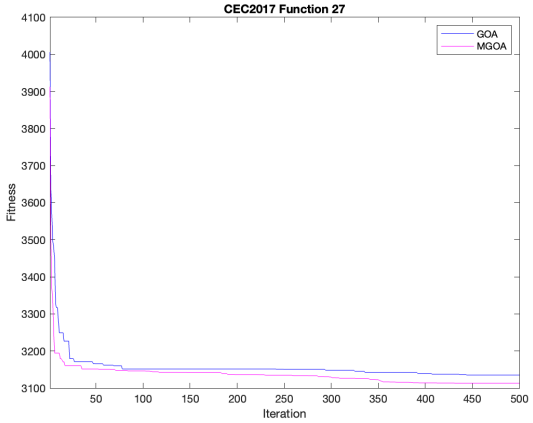
CEC2017-F24



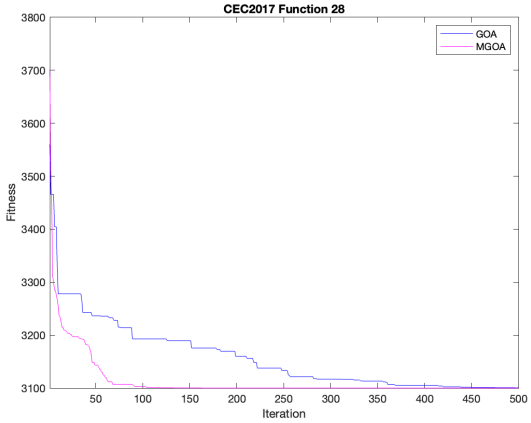
CEC2017-F25



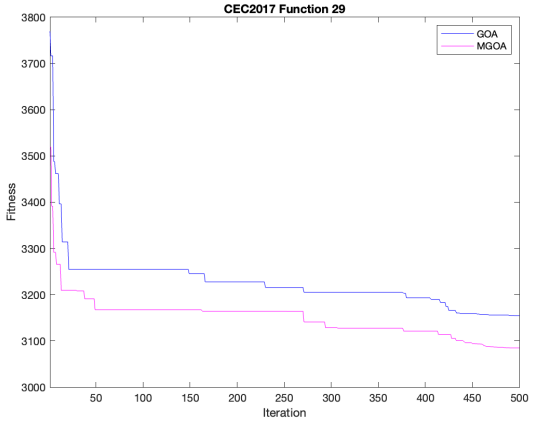
CEC2017-F26



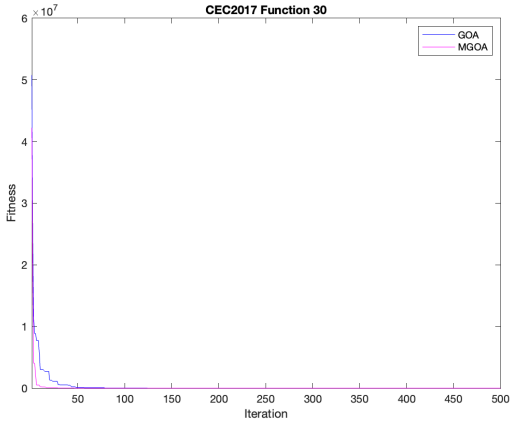
CEC2017-F27



CEC2017-F28



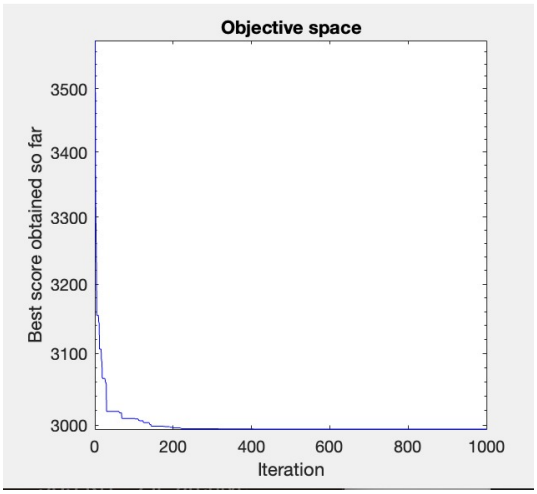
CEC2017-F29



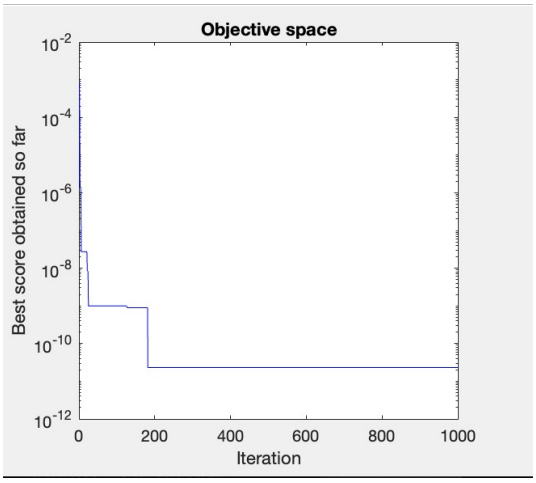
CEC2017-F30

4.3. Performance Comparison of GOA and MGOA on Engineering Problems

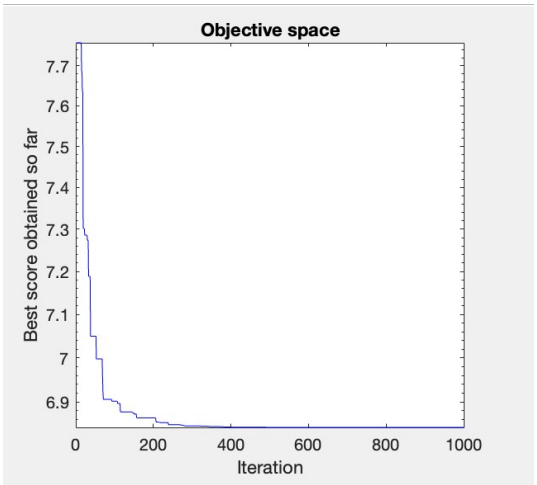
	GOA(ON 50 RUNS)		MGOA(ON 50 RUNS)	
	MEAN	VAR	MEAN	VAR
F1	2994.232551	18.371	2993.42.4466	14.025
F5	1.55E-10	3.98E-07	2.70E-12	8.78E-05
F10	6.842960551	2.85E-07	1.70E-44	1.05E-06
F11	22.84302315	9.14E-10	22.4429692	4.94E-10



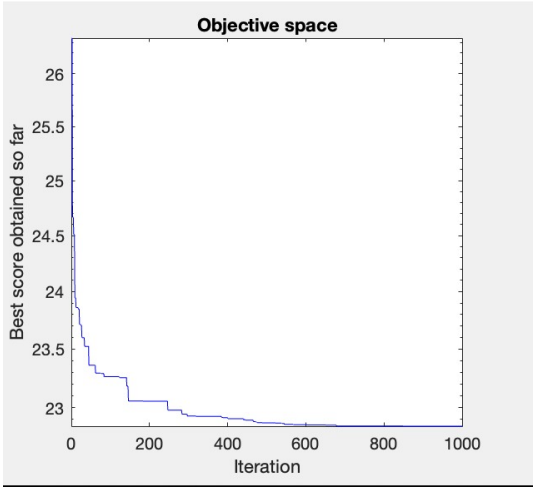
F1



F5



F10



F11

4.4. Performance Comparison with 6 state-of-the-art algorithms

4.4.1. Comparison on CEC2017 AND CEC2014

CEC 2017 (MEAN ON 50 RUNS)

	GOA	MGOA	P VALUE	SCSO	AOA	FHO	WOA	GWO	COA
F1	100.0169667	100	3.31E-20	8573797.106	23596296.13	8500796.257	19173.48917	7049145.667	154655.4668
F2	0	0	0	85108716.35	5431701480	218947527.9	203.0003564	61511567.04	4638.334372
F3	300.0000003	300	3.31E-20	5877.472543	20382.70738	4657.174783	300.2693117	7709.495985	8374.909123
F4	405.99889	404.3431686	0.0001712532529	437.6794805	1565.568065	469.1138409	421.4396884	432.2321041	427.1021836
F5	506.3691918	504.0514196	7.03E-07	520.0821198	520.1053837	520.4052666	520.0729762	520.113357	520.0592348
F6	600.0758535	600.0004968	7.07E-18	606.0483691	609.278563	605.8790067	605.8243539	602.1642442	605.3408497
F7	721.474513	715.4423758	2.38E-14	701.7935598	807.9862906	704.8380004	700.7294899	701.2978919	700.1093725
F8	806.6523015	805.313288	0.00126506368	834.2534375	838.550778	831.286157	824.714734	811.5494315	812.0597157
F9	900.0207003	900	3.31E-20	935.4651797	939.1309242	929.0275064	926.4658419	915.0830386	942.6393264
F10	1212.479987	1096.512604	2.22E-06	1706.26431	1638.755956	2548.900772	1529.010335	1308.018326	1642.495637
F11	1102.723902	1100.258689	1.19E-17	2022.795833	1977.169834	2808.639207	1909.122374	1616.478514	1948.625627
F12	1321.495389	1201.867782	9.54E-18	1200.477686	1200.476721	1201.412277	1200.493175	1200.85461	1200.418236
F13	1308.240615	1302.312182	7.07E-18	1300.364374	1302.498192	1300.398426	1300.360197	1300.212218	1300.226663
F14	1408.735054	1400.814926	1.31E-17	1400.412259	1422.394197	1400.469008	1400.391925	1400.366072	1400.340309
F15	1502.076476	1500.261408	7.07E-18	1504.540523	2488.210497	1536.547243	1504.039357	1501.878402	1501.733212
F16	1603.2353	1601.329557	2.86E-15	1603.094132	1603.558212	1603.595747	1602.731062	1602.744207	1602.706295
F17	1729.805887	1720.688394	5.33E-16	14325.32226	268712.2142	6560.801766	2116.672821	75390.52921	18835.80273
F18	1803.158898	1800.270125	3.19E-16	10497.68922	12197.45437	4288.563175	1853.21538	10626.70702	6167.511152
F19	1901.741388	1900.249767	1.14E-17	1903.888073	1929.699615	1909.577798	1902.950454	1902.450704	1902.952218
F20	2026.829577	2002.324396	7.07E-18	5701.464179	7993.225764	2755.141681	2047.22763	7598.443219	2818.301661
F21	Inf	Inf	NaN	9681.998092	58496.55654	186112.7194	2276.377263	13103.86857	4543.112866
F22	Inf	Inf	NaN	2302.202771	2393.267049	2277.681234	2293.428807	2292.210258	2238.939782
F23	2655.39604	2642.663616	0.001330566117	2500	2500	2595.302365	2500	2633.751366	2500
F24	2651.467866	2554.815628	2.09E-07	2592.434412	2584.606344	2542.13407	2574.999957	2534.073117	2598.362092
F25	2894.953086	2888.837198	1.22E-13	2698.91979	2698.494355	2693.771221	2692.745298	2693.40117	2699.196462
F26	2888.02863	2866	4.97E-12	2700.322512	2704.956369	2700.430793	2702.307138	2702.157354	2706.238355
F27	3127.074458	3122.20881	0.002771717661	2892.220279	2895.887081	2921.951774	2861.691256	3039.779342	2896.105499
F28	3102.93909	3105.874297	3.08E-12	3000	3000	3079.405597	3000	3243.917765	3000
F29	3149.705945	3132.551483	1.58E-13	44824.79697	677453.7672	1297837.747	179916.8727	161379.9152	130853.1941
F30	3230.054232	3218.306967	2.32E-09	4500.828696	20279.33275	9182.363818	4293.741165	4292.639133	4181.563819

CEC 2014 (MEAN ON 50 RUNS)

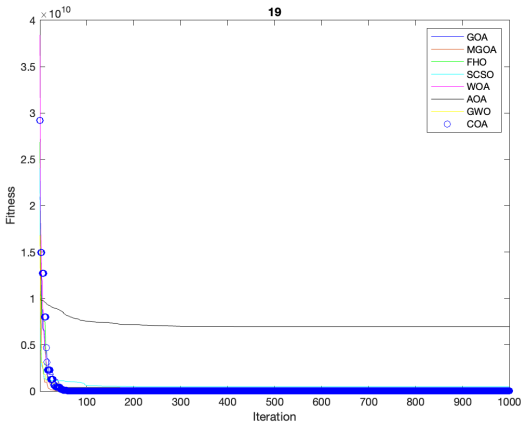
	GOA	MGOA	P VALUE	SCSO	AOA	FHO	WOA	GWO	COA
F1	100.0002227	100	3.31E-20	8573797	23596296	8500796	19173.48917	7049146	154655.5
F2	200.009432	200	3.31E-20	85108716	5.43E+09	2.19E+08	203.0003564	61511567	4638.334
F3	300.0000003	300	3.31E-20	5877.473	20382.71	4657.175	300.2693117	7709.496	8374.909
F4	403.8229488	403.560855	5.35E-06	437.6795	1565.568	469.1138	421.4396884	432.2321	427.1022
F5	520.1215684	519.2137252	7.50E-18	520.0821	520.1054	520.4053	520.0729762	520.1134	520.0592
F6	600.5240172	600.129735	7.44E-12	606.0484	609.2786	605.879	605.8243539	602.1642	605.3408
F7	700.1637536	700.0266969	1.73E-15	701.7936	807.9863	704.838	700.7294899	701.2979	700.1094
F8	802.9089046	800.4123317	8.98E-15	834.2534	838.5508	831.2862	824.714734	811.5494	812.0597
F9	906.9275369	905.0703153	3.48E-05	935.4652	939.1309	929.0275	926.4658419	915.083	942.6393
F10	1059.508291	1023.644954	6.89E-15	1706.264	1638.756	2548.901	1529.010335	1308.018	1642.496
F11	1395.35269	1257.424385	7.55E-06	2022.796	1977.17	2808.639	1909.122374	1616.479	1948.626
F12	1200.278103	1200.093677	3.32E-17	1200.478	1200.477	1201.412	1200.493175	1200.855	1200.418
F13	1300.159784	1300.109851	2.69E-11	1300.364	1302.498	1300.398	1300.360197	1300.212	1300.227
F14	1400.134723	1400.098677	1.19E-06	1400.412	1422.394	1400.469	1400.391925	1400.366	1400.34
F15	1501.158673	1500.614806	1.83E-15	1504.541	2488.21	1536.547	1504.039357	1501.878	1501.733
F16	1602.24176	1601.871163	1.75E-06	1603.094	1603.558	1603.596	1602.731062	1602.744	1602.706
F17	1752.560554	1702.809829	7.07E-18	14325.32	268712.2	6560.802	2116.672821	75390.53	18835.8
F18	1801.946356	1800.176143	4.46E-17	10497.69	12197.45	4288.563	1853.21538	10626.71	6167.511
F19	1901.364038	1900.322454	7.55E-17	1903.888	1929.7	1909.578	1902.950454	1902.451	1902.952
F20	2001.880827	2000.2373	8.46E-18	5701.464	7993.226	2755.142	2047.22763	7598.443	2818.302
F21	2107.074944	2100.537876	7.07E-18	9681.998	58496.56	186112.7	2276.377263	13103.87	4543.113
F22	2215.523085	2202.463907	1.43E-16	2302.203	2393.267	2277.681	2293.428807	2292.21	2238.94
F23	2629.457475	2622.868325	4.73E-20	2500	2500	2595.302	2500	2633.751	2500
F24	2512.020663	2510.818946	0.001536244878	2592.434	2584.606	2542.134	2574.999957	2534.073	2598.362
F25	2641.344042	2622.857526	1.75E-06	2698.92	2698.494	2693.771	2692.745298	2693.401	2699.196
F26	2700.156546	2700.106558	1.39E-11	2700.323	2704.956	2700.431	2702.307138	2702.157	2706.238
F27	2860.332795	2780.786105	7.05E-07	2892.22	2895.887	2921.952	2861.691256	3039.779	2896.105
F28	3169.447045	3169.514622	0.03881338257	3000	3000	3079.406	3000	3243.918	3000
F29	3115.822734	3104.919592	4.67E-08	44824.8	677453.8	1297838	179916.8727	161379.9	130853.2
F30	3494.990892	3459.10889	4.21E-17	4500.829	20279.33	9182.364	4293.741165	4292.639	4181.564

4.4.2. Comparison on Engineering Problems

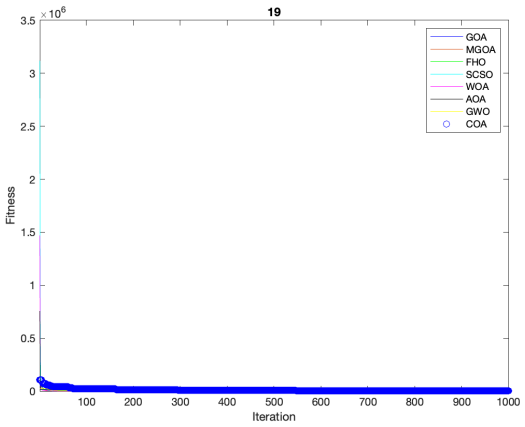
	GOA	MGOA	SCSO	FHO	AOA	COA	GWO
F1	2997.581713	2993.42.4466	3001.791475	2352.343276	2709.231621	2994.432128	3025.502769
F5	1.11E-10	2.70E-12	5.44E-10	0	12.356	1.04E-09	1.49E-09
F10	3.01372554	1.70E-44	5.893314397	0	0.0832159767	5.202963945	6.693826807
F11	22.87128418	22.4429692	23.16963199	15.515	19.85483857	22.97405261	23.84189217

Mean Results computed on 50 runs

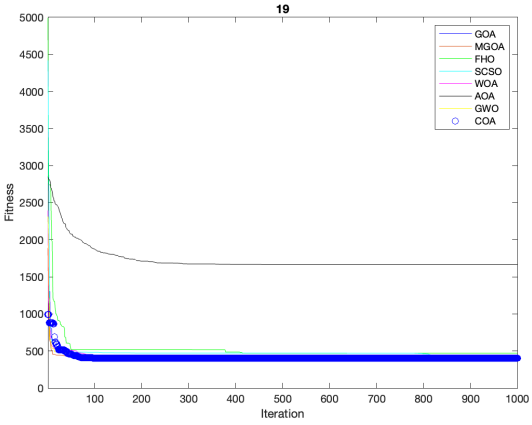
4.4.3. Convergence Curves for CEC2017



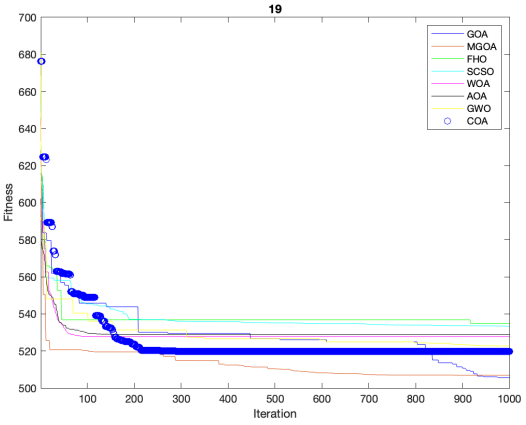
CEC2017-F1



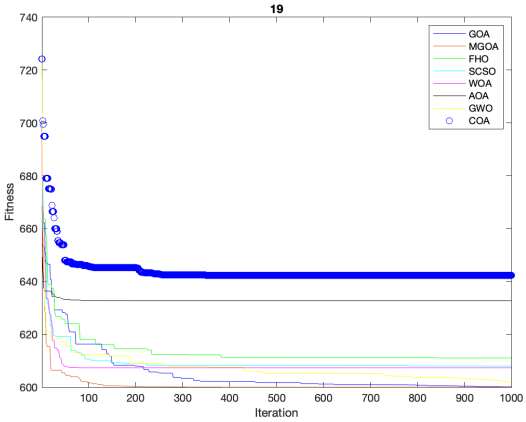
CEC2017-F3



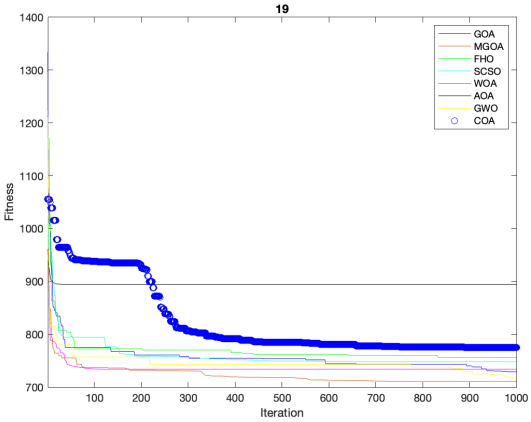
CEC2017-F4



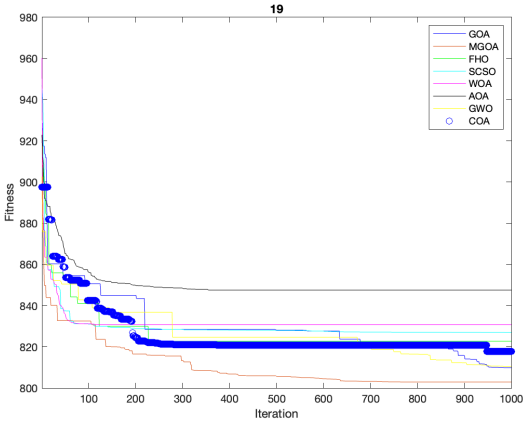
CEC2017-F5



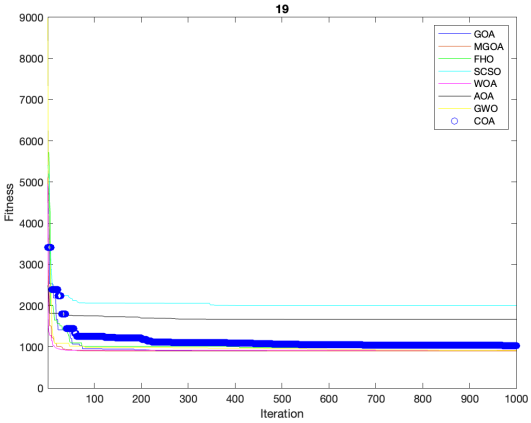
CEC2017-F6



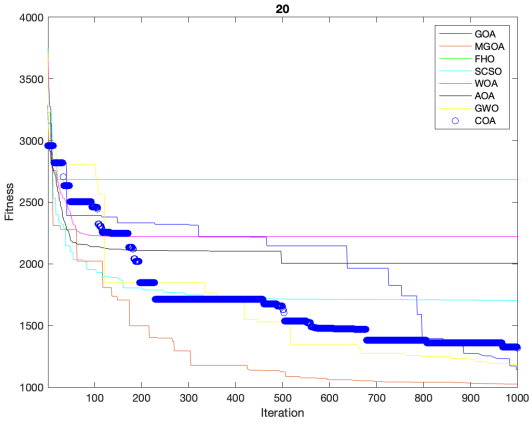
CEC2017-F7



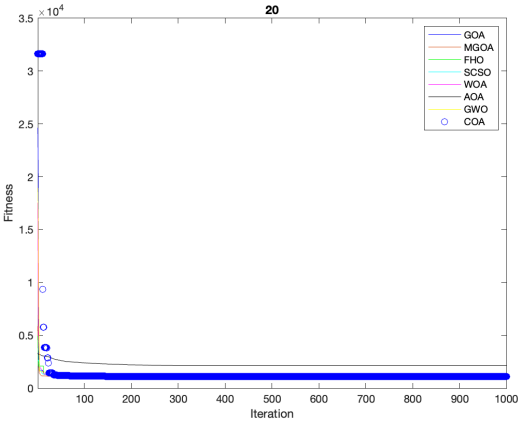
CEC2017-F8



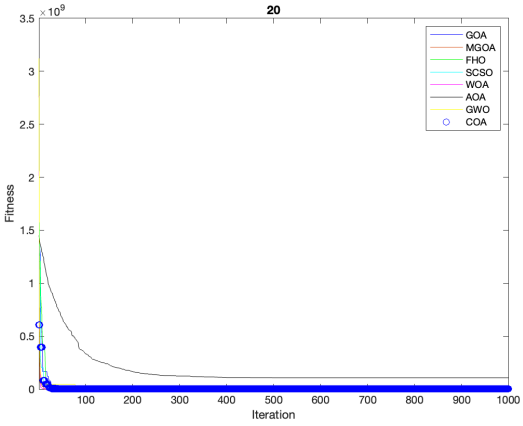
CEC2017-F9



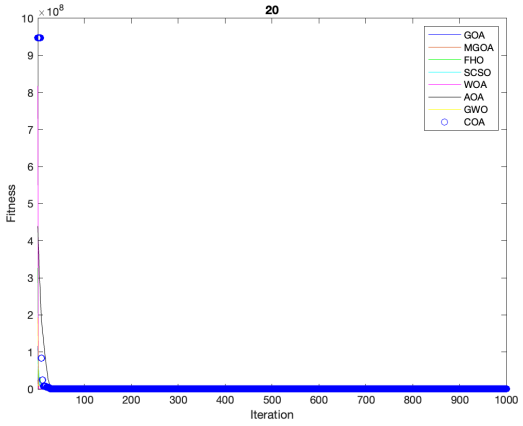
CEC2017-F10



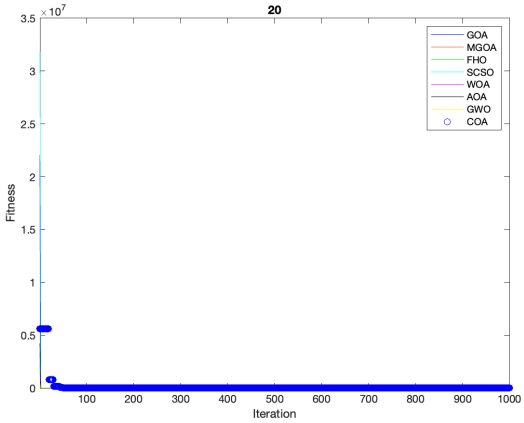
CEC2017-F11



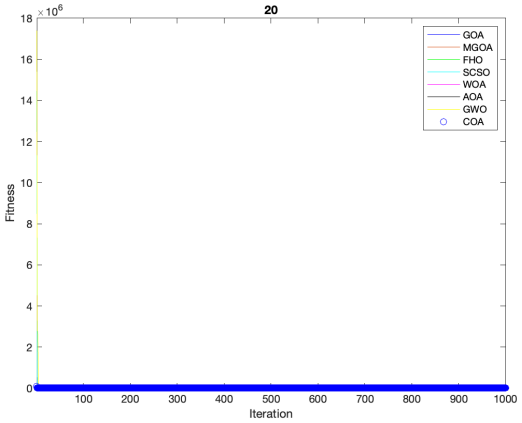
CEC2017-F12



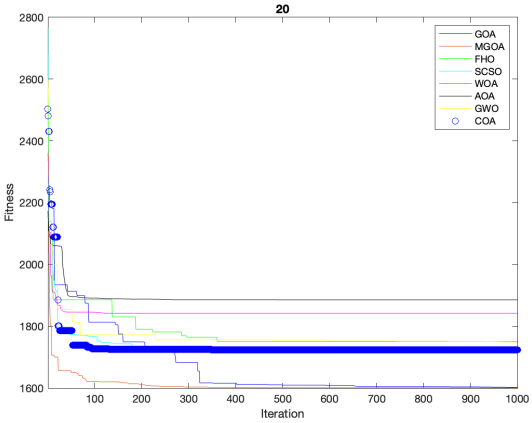
CEC2017-F13



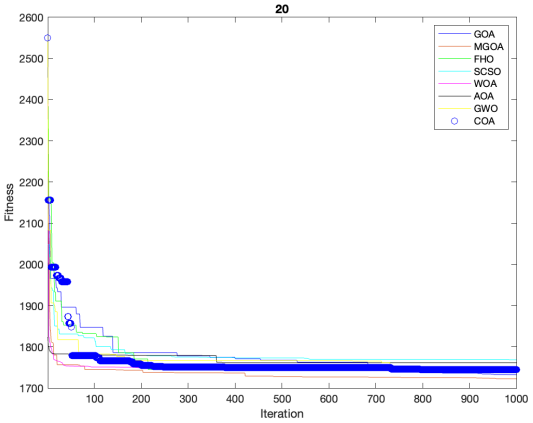
CEC2017-F14



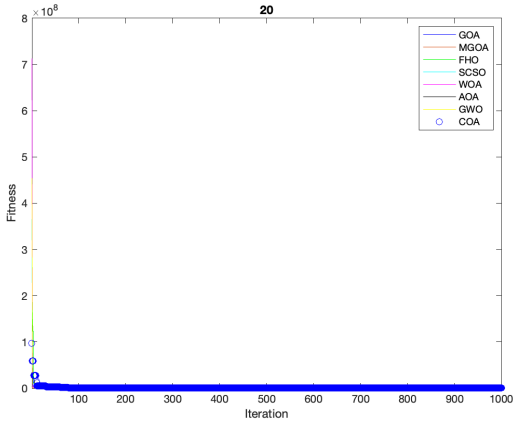
CEC2017-F15



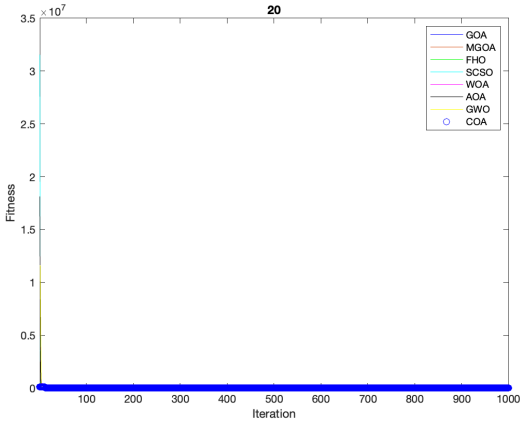
CEC2017-F16



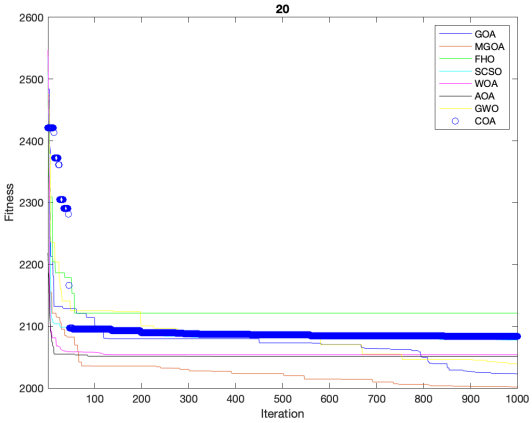
CEC2017-F17



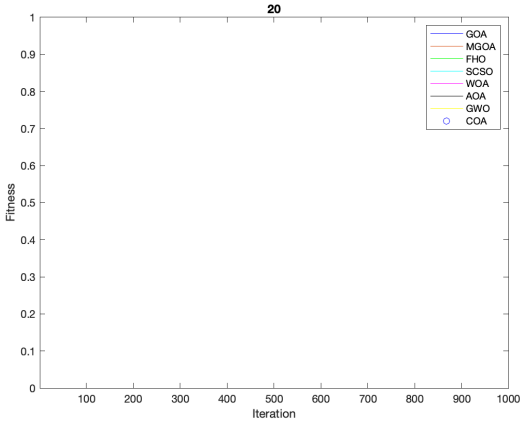
CEC2017-F18



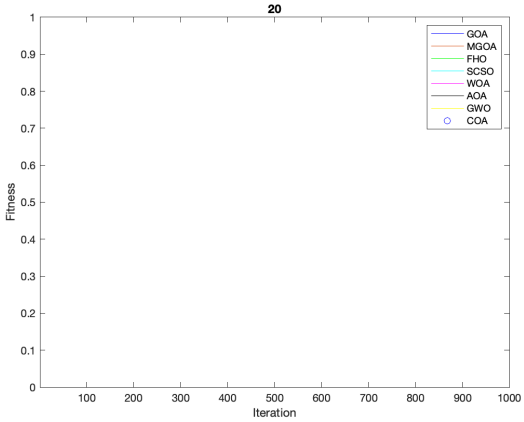
CEC2017-F19



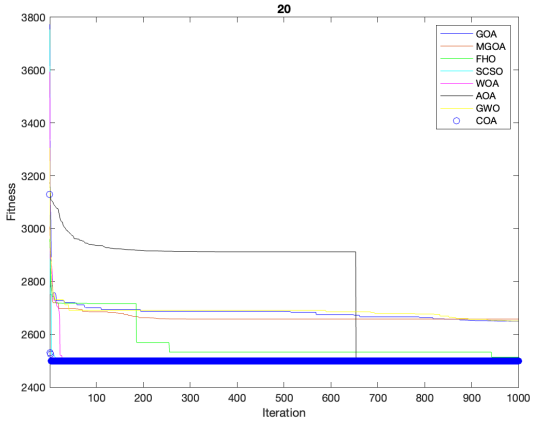
CEC2017-F20



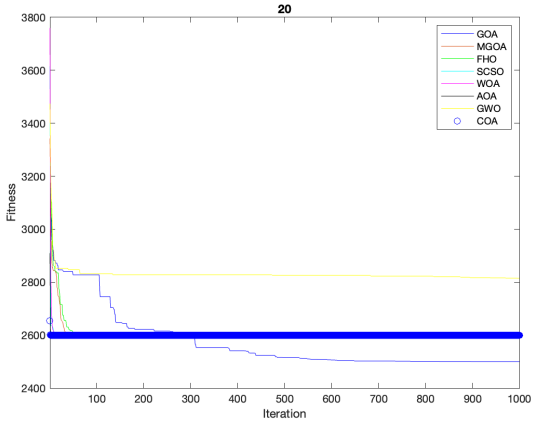
CEC2017-F21



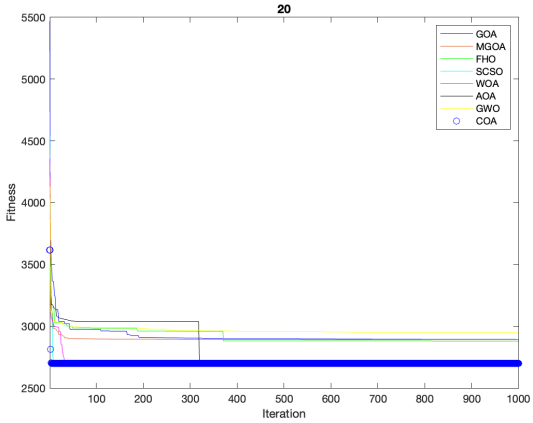
CEC2017-F22



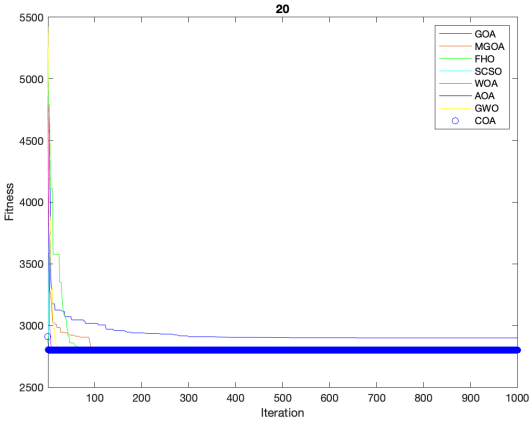
CEC2017-F23



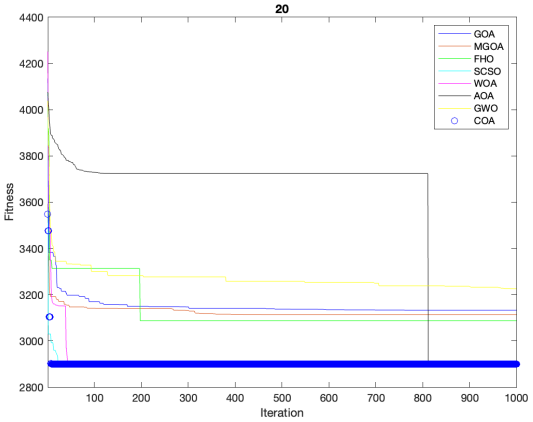
CEC2017-F24



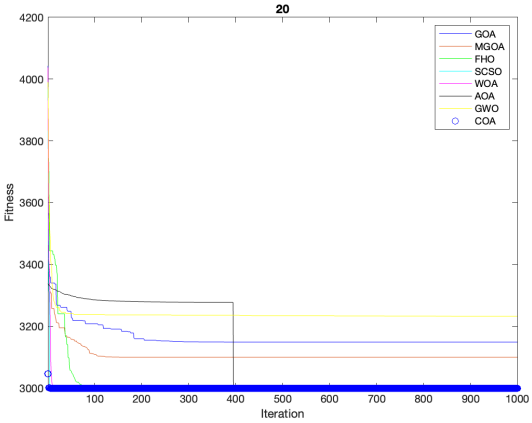
CEC2017-F25



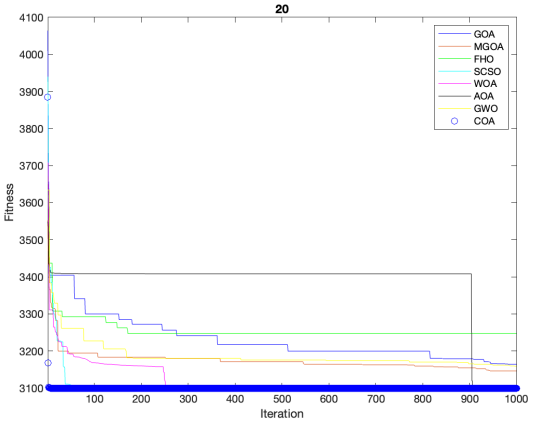
CEC2017-F26



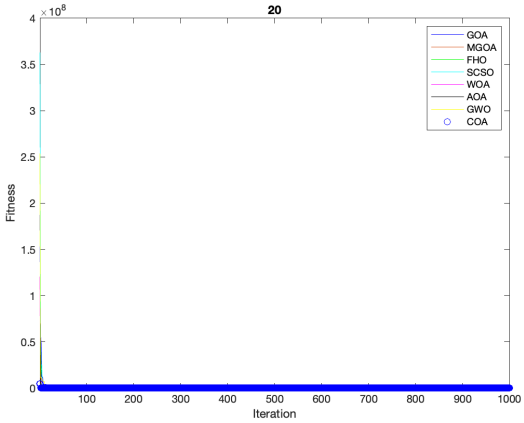
CEC2017-F27



CEC2017-F28

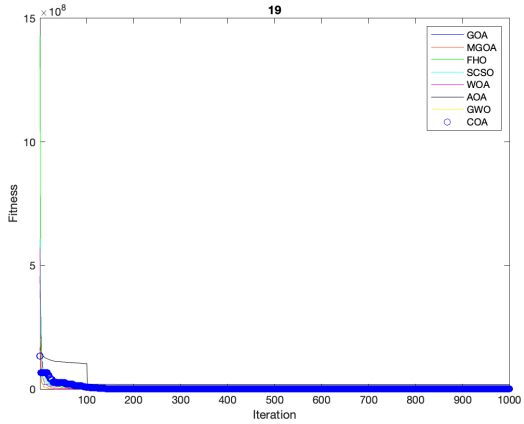


CEC2017-F29

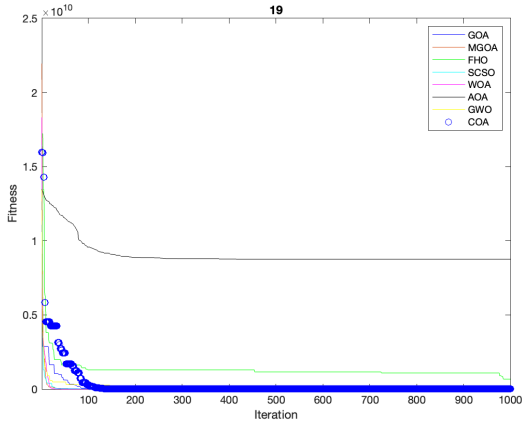


CEC2017-F30

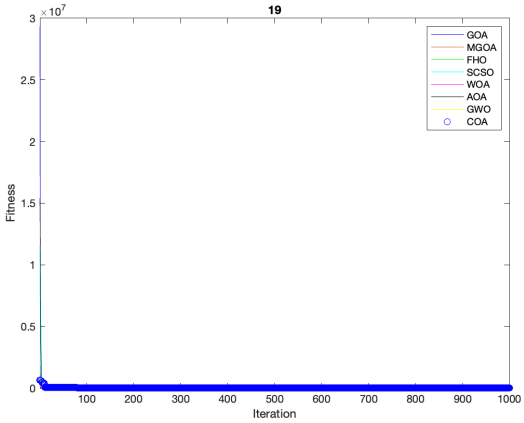
4.4.4. Convergence Curves for CEC2014



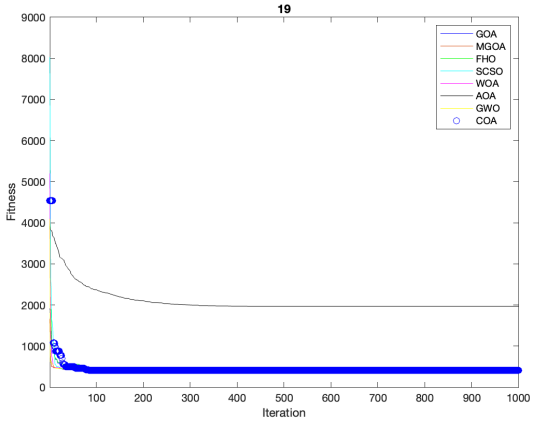
CEC2014-F1



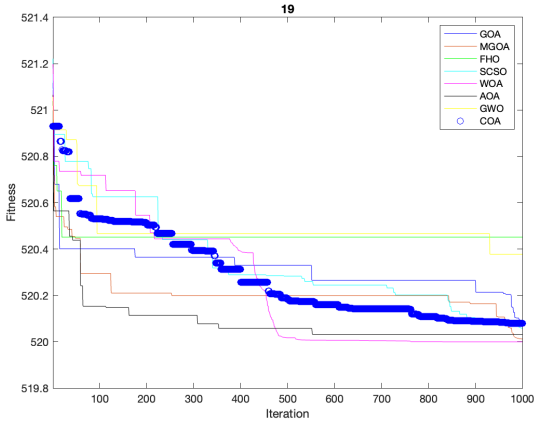
CEC2014-F2



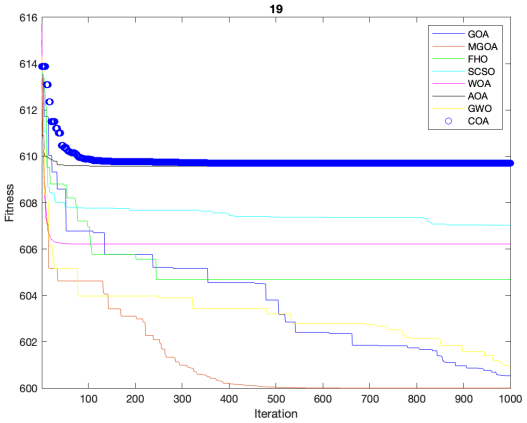
CEC2014-F3



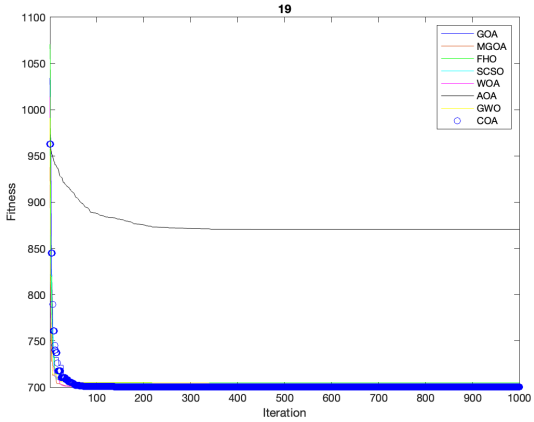
CEC2014-F4



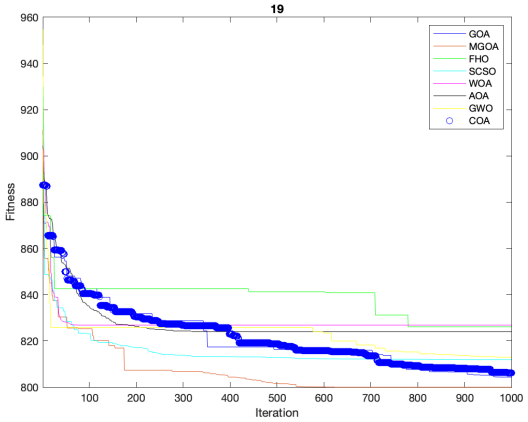
CEC2014-F5



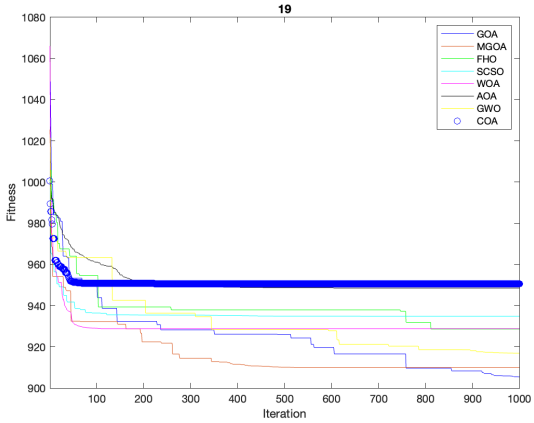
CEC2014-F6



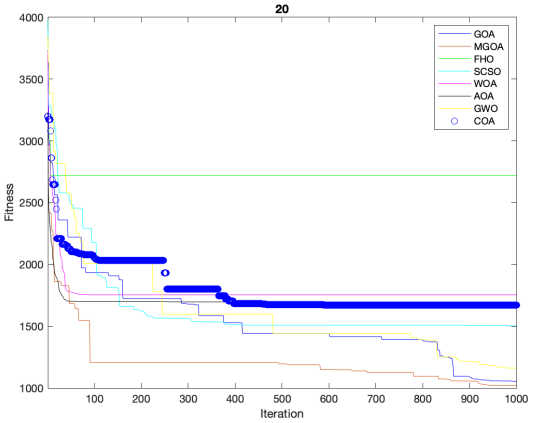
CEC2014-F7



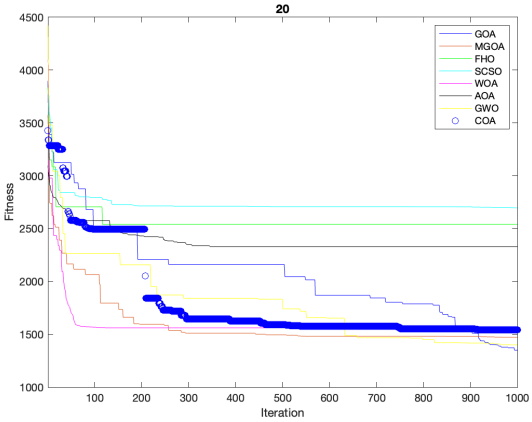
CEC2014-F8



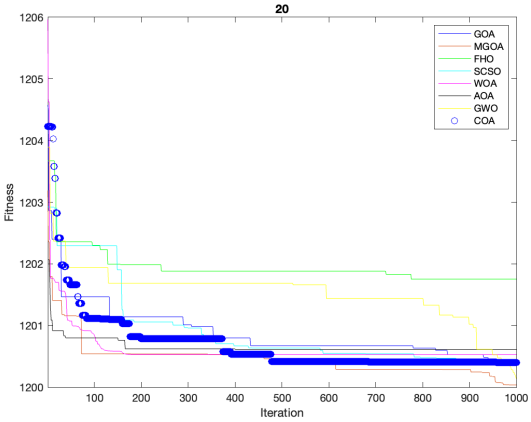
CEC2014-F9



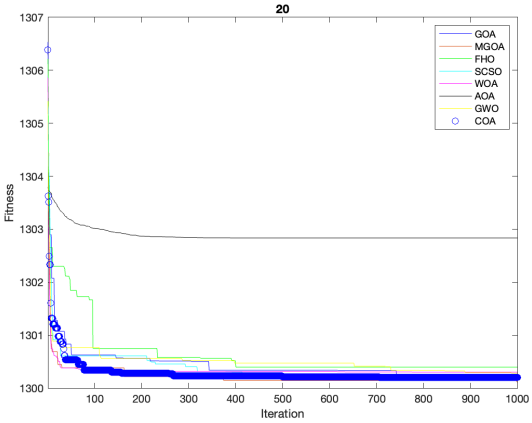
CEC2014-F10



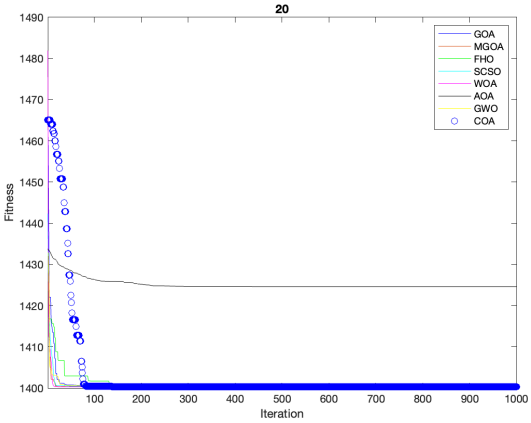
CEC2014-F11



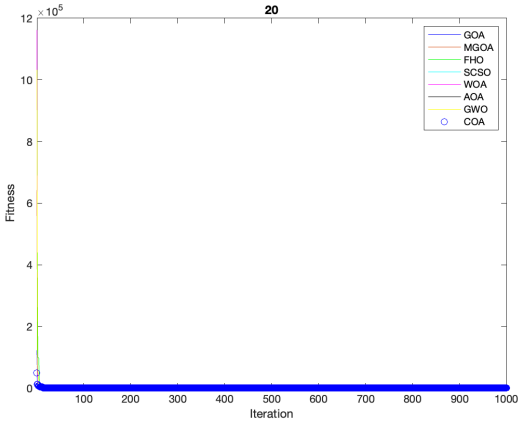
CEC2014-F12



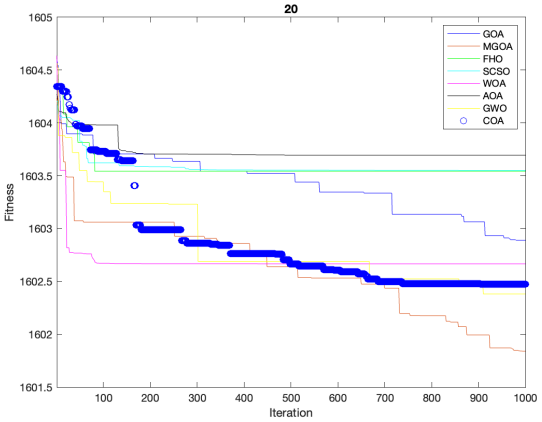
CEC2014-F13



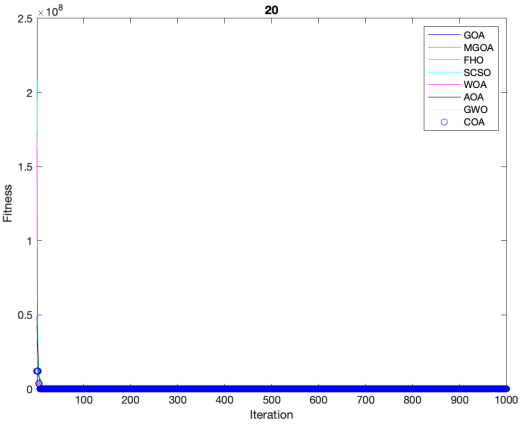
CEC2014-F14



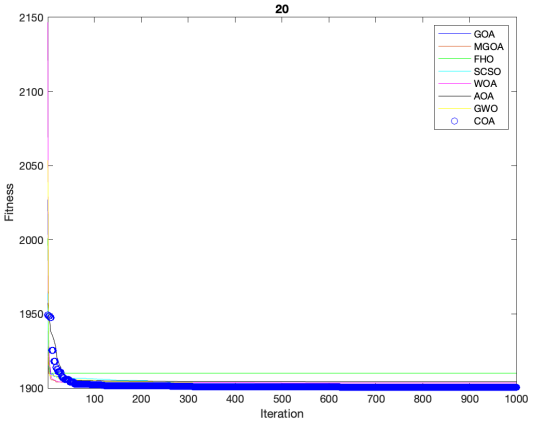
CEC2014-F15



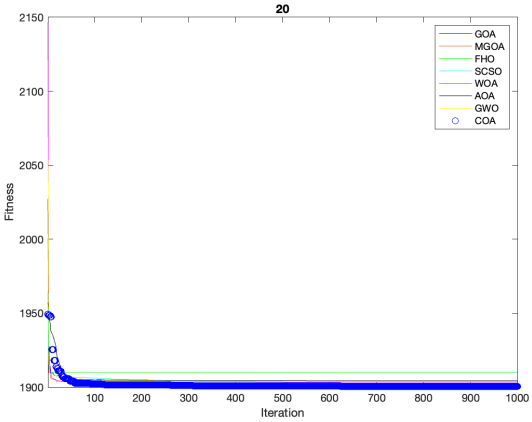
CEC2014-F16



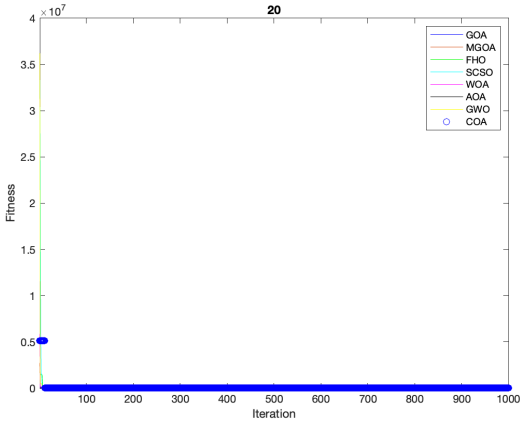
CEC2014-F17



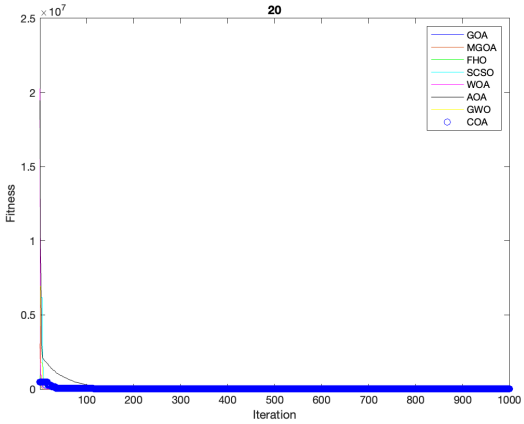
CEC2014-F18



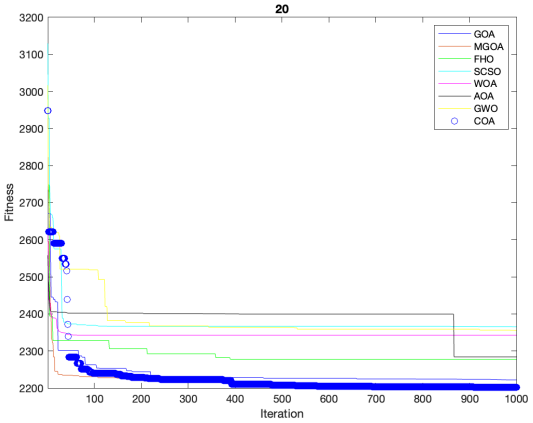
CEC2014-F19



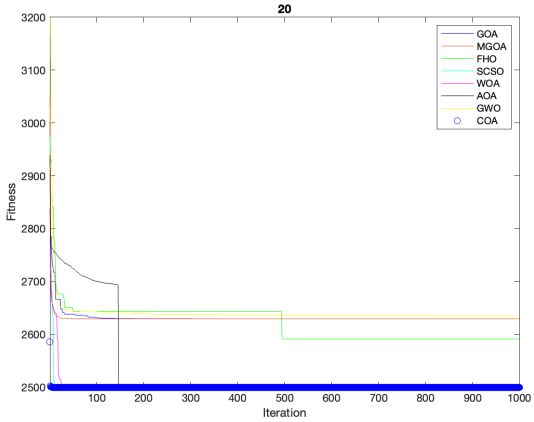
CEC2014-F20



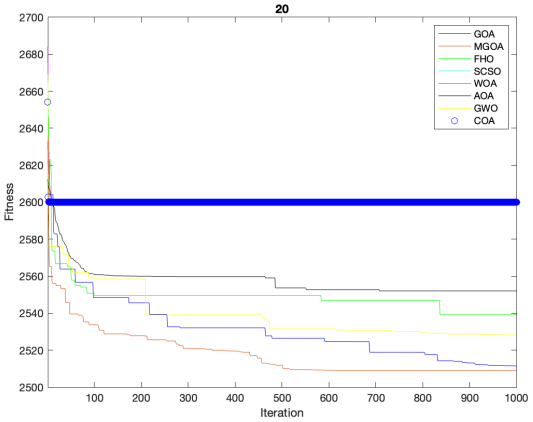
CEC2014-F21



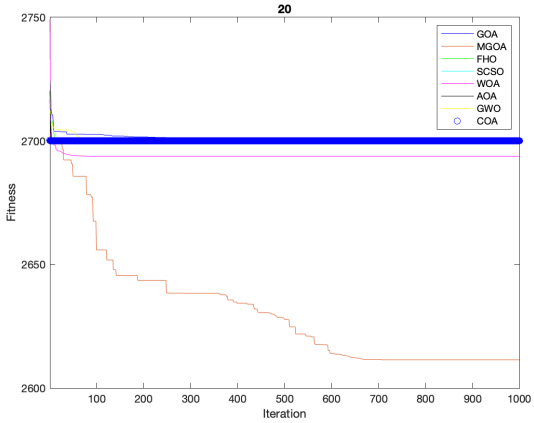
CEC2014-F22



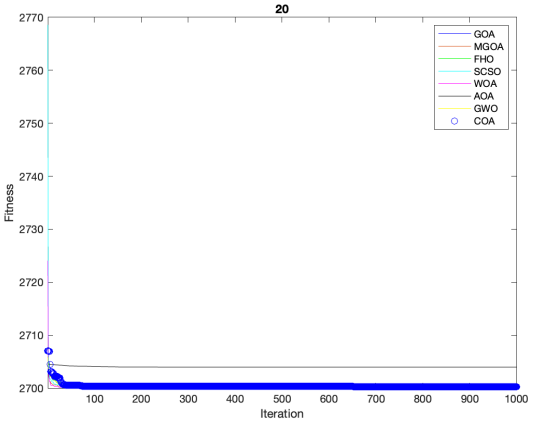
CEC2014-F23



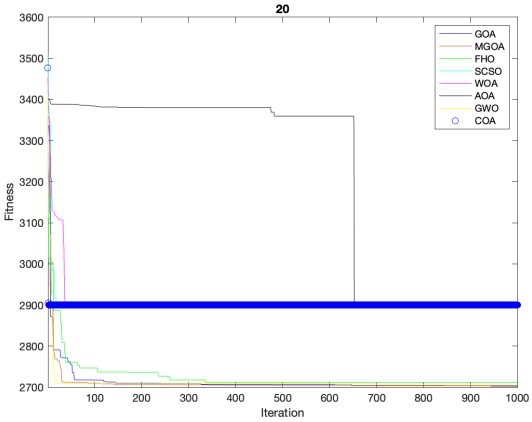
CEC2014-F24



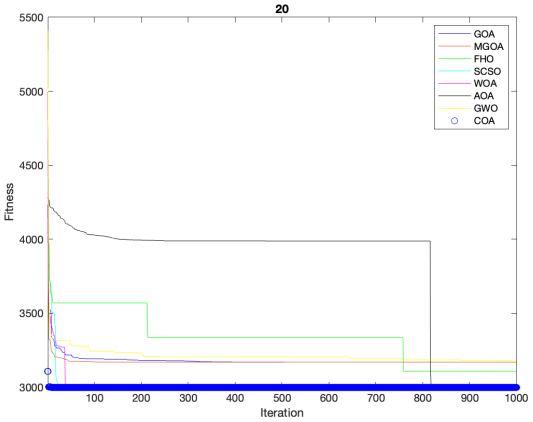
CEC2014-F25



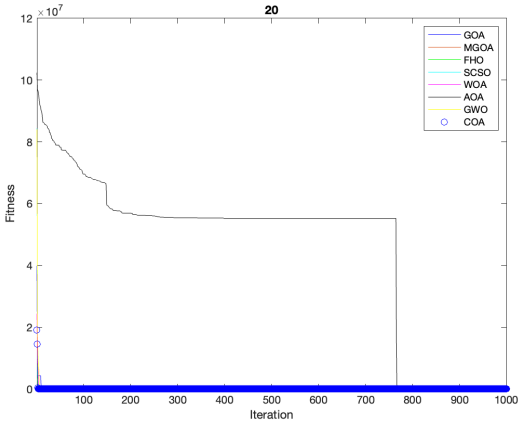
CEC2014-F26



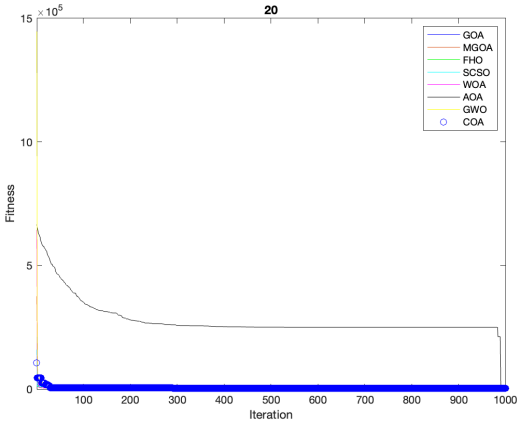
CEC2014-F27



CEC2014-F28



CEC2014-F29



CEC2014-F30

5. Conclusion and future work

The simulation results of the Gazelle Optimization Algorithm (GOA) and its variant, the Iterated Gazelle Optimization Algorithm (MGOA), showcase their robustness across a spectrum of benchmark suites and engineering problems. From the Comprehensive Economic and Engineering Problems to the CEC 2014 and 2017 suites, these algorithms consistently outshine their counterparts. Notably, MGOA stands out among six state-of-the-art algorithms documented in the literature, including the original GOA. This comprehensive performance evaluation demonstrates the algorithm's superiority, positioning it as a formidable contender in the realm of metaheuristic techniques.

In-depth statistical analysis further validates the effectiveness of MGOA, with the Wilcoxon test confirming the significance of its performance. The algorithm's remarkable ability to yield optimal solutions across a majority of the considered problems is particularly noteworthy. Even in scenarios where competitiveness is required, MGOA rises to the challenge, showcasing its versatility and adaptability. These findings not only highlight the algorithm's prowess but also underscore its potential to tackle a wide range of real-world problems with efficiency and efficacy.

Looking ahead, the implications of MGOA's performance are profound. Its demonstrated superiority over established algorithms and competitive edge against more advanced methodologies position it as a promising solution for various applications. From parallel machine scheduling to economic load dispatch in electronic sciences, and even medical image processing, MGOA presents itself as a potent tool for optimization tasks. As the algorithm continues to evolve and adapt, future research may uncover even more applications and further enhance its capabilities for addressing complex real-world challenges.

References

1. Abed-alguni BH, Paul D (2022) Island-based Cuckoo Search with elite opposition-based learning and multiple mutation methods for solving optimization problems. Soft Comput 26(7):3293–3312
2. Abed-Alguni BH, Paul D, Hammad R (2022) Improved Salp swarm algorithm for solving single-objective continuous optimization problems. Appl Intell. https://doi.org/10.1007/s10489-022-03269-x
3. Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609
4. Agushaka JO, Ezugwu AE (2020). Diabetes classification techniques: a brief state-of-the-art literature review. In: International Conference on Applied Informatics (pp. 313–329). Ogun: Springer, Cham
5. Agushaka JO, Ezugwu AE (2021) Advanced arithmetic optimization algorithm for solving mechanical engineering design problems. PLoS ONE. https://doi.org/10.1371/journal.pone.0255703
6. Agushaka JO, Ezugwu AE (2022) Influence of probability distribution initialization methods on the performance of advanced arithmetic optimization algorithm with application to unrelated parallel machine scheduling problem. Concurr Comput Pract Exper 34:e6871
7. Agushaka JO, Ezugwu AE, Abualigah L (2022) Dwarf Mongoose Optimization Algorithm. Comput Methods Appl Mech Eng 391:114570
8. Agushaka J, Ezugwu A (2020) Influence of initializing krill herd algorithm with low-discrepancy sequences. IEEE Access 8:210886–210909

9. Akay B, Karaboga D (2012) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci 192:120–142
10. Akay B, Karaboga D (2012) Artificial bee colony algorithm for large-scale problems and engineering design optimization. J Intell Manuf 23(4):1001–1014
11. Alatas B (2011) ACROA: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 38(10):13170–13180
12. Arora S, Singh S (2019) Butterfly optimization algorithm: a novel approach for global optimization. Soft Comput 23(3):715–734
13. Atashpaz-Gargari E, Lucas C (2007). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In 2007 IEEE congress on evolutionary computation (pp. 4661–4667). IEEE
14. Biswas A, Mishra K, Tiwari S, Misra A (2013). Physics-inspired optimization algorithms: a survey. Journal of Optimization, 2013
15. Braik MS (2021) Chameleon swarm algorithm: a bio-inspired optimizer for solving engineering design problems. Expert Syst Appl 174:114685
16. Rahkar Farshi T (2021) Battle royale optimization algorithm. Neural Comput Appl 33(4):1139–1157
17. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248
18. Rather S, Bala P (2019). Hybridization of constriction coefficient based particle swarm optimization and gravitational search algorithm for function optimization. In: International Conference on Advances in Electronics, Electrical, and Computational Intelligence (ICAEEC- 2019). Elsevier
19. Rechenberg I (1978). Evolutionary strategies. In simulation methods in medicine and biology, Berlin, Heidelberg, 83–114
20. Sarzaeim P, Bozorg-Haddad O, Chu X (2018). Teaching-learning-based optimization (TLBO) algorithm. Advanced Optimization by Nature-Inspired Algorithms. Singapore, Asia: Springer, 51–58
21. Shabani A, Asgarian B, Salido M, Gharebaghi SA (2020) Search and rescue optimization algorithm: a new optimization method for solving constrained engineering optimization problems. Expert Syst Appl 161:113698
22. Simon D (2008) Biogeography based optimization. IEEE Trans Evol Comput 12(6):702–713
23. Slowik A, Kwasnicka H (2020) Evolutionary algorithms and their applications to engineering problems. Neural Comput Appl 32(16):12363–12379
24. Storn R, Price K (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
25. Tzanetos A, Dounias G (2021) Nature inspired optimization algorithms or simply variations of metaheuristics? Artif Intell Rev 54(3):1841–1862
26. Uymaz SA, Tezel G, Yel E (2015) Artificial algae algorithm (AAA) for nonlinear global optimization. Appl Soft Comput 31:153–171
27. Wang GG, Guo L, Gandomi AH, Hao GS, Wang H (2014) Chaotic krill herd algorithm. Inf Sci 274:17–34
28. Xie Q, Cheng G, Zhang X, Peng L (2020) Feature selection using improved forest optimization algorithm. Inf Technol Control 49(2):289–301

29. Xing B, Gao W (2014). Invasive weed optimization algorithm. Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms. Cham, Switzerland: Springer, 177–181
30. Yang X S, Deb S (2009). Cuckoo search via Le´vy flights. In 2009 World congress on nature biologically inspired computing (NaBIC) (pp. 210–214). IEEE