Applied Computing and Intelligence

https://www.aimspress.com/journal/aci

Applied Computing and Intelligence
4 (1): 93–106.
DOI: 10.3934/aci.2024006
Received: 30 July 2024
Revised: 28 August 2024
Accepted: 03 September 2024
Published: 09 September 2024

Research article

Modification of coot optimization algorithm (COA) with adaptive sigmoid increasing inertia weight for global optimization

Elvis Twumasi*, Ebenezer Archer, Emmanuel O. Addo and Emmanuel A. Frimpong

Department of Electricals and Electronics Engineering, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana

* **Correspondence:** Email: etwumasi.coe@knust.edu.gh.

Academic Editor: Pasi Fränti.

Abstract: In this paper, the classical coot optimization algorithm (COA) is modified to improve its overall performance in the exploration phase by adding an adaptive sigmoid inertia weight-based method. The modified coot optimization algorithm (mCOA) was successfully assessed using 13 standard benchmark test functions, which are frequently used to evaluate metaheuristic optimization algorithms. The MATLAB software was utilized to conduct simulation tests, and the outcome was compared with the performance of the original COA, the particle swarm optimization, and the genetic algorithm reported in the literature. The findings showed that the proposed algorithm outperformed the other algorithms on ten (10) of the 13 benchmark functions, while it maintained a competitive performance on the remaining three benchmark test functions. This indicates that mCOA provides a significant improvement to the original COA, thus making it suitable for resolving optimization problems in diverse fields. As a result, the proposed algorithm is recommended for adoption to solve real-life engineering optimization problems.

Keywords: coot optimization algorithm; modification; algorithm; adaptive weight

1. Introduction

Nature-inspired algorithms are well-known algorithms for finding near-optimal solutions to optimization problems. These algorithms have successfully solved optimization problems in different domains [1,2]. The natural behaviors or events among living creatures inspire these nature-inspired algorithms Examples of such events include the mountain gazelle's social behavior, and the behavior of birds, monkeys, and other animals in the wildlife [3]. These algorithms have shown promise in solving real-world engineering problems and problems in many fields. Among the plethora of algorithms that have been designed by researchers over the years, new algorithms are still of interest among optimization enthusiasts to solve the common problem of these algorithms, which are mainly entrapped in local optima solutions and slow convergences. The *Coot Optimization Algorithm* (COA) has shown promise in solving these difficulties in classical nature-inspired algorithms. COA is inspired by the social lifestyle exhibited by coots. It is an effective and straightforward metaheuristic algorithm to find near-optimal solutions [4]. Despite the great prosperity of COA, it suffers from a slow convergence in solving very complex problems [5]. This weakness caused COA to require an extremely high number of iterations to produce substantially good results, especially on large-scale complex optimization problems [5,6].

The poor balance of the exploration and exploitation stages significantly contributes to the slow convergence [6,7]. While exploitation guarantees the refinement of solutions in attractive parts of the search space, exploration is essential for the global search and to prevent a premature convergence to the local optima [8]. An imbalance between these two stages can seriously impair the algorithm's effectiveness, thus resulting in wasted searches and higher processing expenses that ultimately yield less-than-ideal outcomes [9].

In the literature, scholars have employed a range of corrective measures to enhance the performance of certain algorithms that have comparable shortcomings [10,11]. A typical example is the truncation parameter selection technique, which was adopted to improve the general performance of the *Mountain Gazelle Optimizer* (MGO) on standard benchmark test functions [12]. To enhance the overall performance of the *Gorilla Troop Optimizer* (GTO), A. Bright et al. [13] integrated a step adaptive simulation concept into the GTO.

In the case of the COA, Aslan et al. [14] proposed a modification by integrating a randomized mutation technique into the original COA to enhance its global search ability. However, excessive randomization within algorithms causes significant instabilities. Additionally, R. R. Mostafa et al. [7] modified COA by introducing an opposition-based learning and an orthogonal-based learning approach to improve the algorithm’s performance. Moreover, authors in [15] proposed control randomization and a transition factor-based strategy to enhance the COA. This modification was geared towards solving the battery parameters estimation problem. Hence, its performance in a variety of optimization fields was not established.

Similarly, this work seeks to modify the COA to improve its general performance through an adaptive sigmoid increasing inertia weight in the "Leader Movement" phase [16]. This is a crucial stage of the COA because it determines how the coots are led by their leader coot on the surface of the water. This influences the dynamics of the search as a whole [6]. The integrated adaptive weight is intended to dynamically balance the exploration and exploitation mechanisms throughout the search process.

To achieve this aim, the exploration is dominant at the early stages of the optimization search process and then gradually improves the exploitation mechanism in the later stages. Therefore, the adaptive sigmoid rising inertia weight is designed to begin with a smaller value and gradually increase [16].

The rest of the paper is organized as follows: the original COA and the proposed modification are presented in Section 2; the performance of the simulation's outcome is discussed in Section 3; and lastly, Section 4 concludes the paper by providing a thorough synopsis of the research and suggestions for potential future studies as recommendations.

2. Adaptive sigmoid increasing inertia weight-based modification of COA

This section presents the methodological approach followed in this research, which consists of the original COA, the proposed modification, and the test implementation.

2.1. The original coot optimization algorithm

The COA mimics the behavior of American coots as they navigate through seas or lakes. American coots have four distinct movement strategies: random movement, chain formation, moving toward group leader positions, or leading the group [6]. The initial generation is created randomly using Eqs (1) and (2):

$$CootPos(i) = rand(1, d) \times (ub - lb) + lb, \tag{1}$$

$$lob = [lob_1, lob_1, \dots, lob_d], uob = [uob_1, uob_2, \dots, uob_d], \tag{2}$$

where *CootPos* denotes the position of the *ith* coot, *d* represents the dimension number, *uob* and *lob* signify the upper and lower boundaries, respectively, and *rand* denotes a random vector within the range [0, 1].

Random movement:

If coots exhibit random movement, then they will consequently migrate towards a position denoted as Q, which can be determined through Eq (3):

$$Q = rand(1, d) \times (ub - lb) + lb. \tag{3}$$

To prevent getting trapped in local optimal areas, if coots encounter a failure within a local region, then they will employ a position as determined through Eq (4):

$$CootPos(i) = CootPos(i) + A \times R2 \times (Q - CootPos(i)), \tag{4}$$

where $R2 \in [0, 1]$ and A can be calculated using Eq (5):

$$A = 1 - L \times \left(\frac{1}{Iter}\right), \tag{5}$$

where *L* and *Iter* represent the current iteration and the maximum number of iterations, respectively.

Chain movement:

To mathematically represent the movement of the chain phase, we address the following equation, denoted as Eq (6):

$$CootPos(i) = 0.5 \times (CootPos(i - 1) + CootPos(i)). \tag{6}$$

Moving towards group leader:

When adjusting its position according to the leader's position, the coot's movement is governed by the following equation, denoted as Eq (7):

$$K = 1 + (i\ MOD\ NL). \tag{7}$$

Here, i represents the total number of coots, NL represents the number of leaders, and *K* denotes a specific leader. The update process utilizes the following equation, referred to as Eq (8):

$$CootPos(i) = LeaderPos(K) + 2 \times R1 \times \cos(2R\pi) \times (LeaderPos(K) - CootPos(i)). \tag{8}$$

Leading the group by the leader (Leader movement):

Finally, to update their positions, the leaders employ the following Eq (9): to update their positions.

$$LeaderPos(i) = \begin{cases} B \times R3 \times \cos(2R\pi) \times \big(gBest - LeaderPos(i)\big) + gBest, & if\ R4 < 0.5, \\ B \times R3 \times \cos(2R\pi) \times \big(gBest - LeaderPos(i)\big) - gBest, & if\ R4 \geq 0.5, \end{cases} \tag{9}$$

where *gBest* is the best position, $R \in [-1, 1]$, both $R3$ and $R4 \in [0, 1]$, B can be calculated using Eq (10):

$$B = 2 - L \times \left(\frac{1}{Iter}\right), \tag{10}$$

where L represents the current iteration, and *Iter* represents the maximum number of iterations.

2.2. Proposed modification

The original COA has a great potential of being adopted for applications in various optimization fields [7]. However, the COA has a slow convergence, which makes it require a lot of iterations to produce a good optimization result [6]. This drawback is caused by a poor exploration and exploitation balance to ensure an efficient search.

To remedy this weakness, an adoptive sigmoid increasing inertia weight [16] is incorporated in the Leader Movement phase. This phase of the original COA is expressed in Eq (9), which indicates how the leader coots lead the coots' group to move on the water surface. The proposed weight is incorporated as shown in Eq (11):

$$LeaderPos(i) = \begin{cases} B \times R3 \times \cos(2R\pi) \times \big(gBest - LeaderPos(i)\big) + gBest, & ifR4 < 0.5, \\ B \times R3 \times \cos(2R\pi) \times \big(gBest - LeaderPos(i)\big) - \omega \times gBest, & ifR4 \geq 0.5, \end{cases} \tag{11}$$

where the value of the weight, ω, is calculated using the sigmoid increasing inertia weight expressed in Eq (12):

$$\omega(i) = \omega_{min} + \frac{\omega_{max} - \omega_{min}}{1 + e^{a - b \times \frac{i}{MaxIter}}}, \tag{12}$$

where a and b are parameters for adjustment, which are carefully chosen through numerical simulations. The weights ω_{min} and ω_{max} represent the minimum and maximum weight values, respectively, and i and *MaxIter* represent the current iteration and the maximum number of iterations, respectively.

Finally, $\omega(i)$ represents the adaptive weight value at the *ith* iteration. The proposed weight is integrated in Eq (11) when $R4 \geq 0.5$ to gain the full benefit of the inertia weight technique while avoiding its drawback of a possible premature convergence. In algorithms with the inertia weight techniques, they might converge too quickly to suboptimal solutions, especially when the weight parameter is not tuned properly. However, they are good at providing an effective and fast convergence to global solutions when properly implemented. To tap the good qualities of the coot technique, the developers of the algorithm incorporated the weighting factor when $R4 \geq 0.5$. In this contribution, the eight has been designed to ensure the versatility of the algorithm during each iteration while avoiding its weaknesses. Depending on the value of $R4$, Eq (11) is executed with the proposed weight, where the good qualities can be utilized, or without the weight, where the algorithm can freely search within the space without solely focusing on the temporarily best individual members of the population.

The implementation (pseudocode) of the modified COA (mCOA) is presented as shown in Algorithm 1:

Algorithm 1. The pseudo-code of mCOA.

```
Initialize the first coot population randomly using Eqs (1) and (2).
Initialize the parameters of P=0.5, NL (Number of leaders), Ncoot (Number of coots), $\omega_{\min}$ , $\omega_{\max}$, a, b, and MaxIter (max iteration).
Random selection of coot leaders.
Calculate the fitness of coots and leaders.
Find the best coot or leader as the global optimum (gBest).
While (Iter≤MaxIter)
Calculate $A$, B, and $\omega$ using Eqs (5), (10), and (12) respectively.
  If $rand < P$
   $R$1, $R$2, and $R$3 are random vectors along the dimensions of the problem.
  Else
   $R$1, $R$2, and $R$3 are random numbers.
  End
   For $i$=1 to the number of coots
     Calculate $K$ by Eq (7).
     If $rand > 0.5$
      Update the positions of the coots by Eq (8)
     Else
     If $rand < 0.5$ $i \neq 1$
      Update the positions of the coots by Eq (6)
     else
      Update the positions of the coots by Eq (4)
     End-if
   End-for
   Calculate the fitness of the coot
   If the fitness < the fitness of leader($k$)
      $Temp$ = leader($k$); leader($k$)=$coot$; $coot$=$Temp$
   End
  End-????
 For the number of leaders
   If $rand < 0.5$
      Update the position of the leader by Eq (11.1)
   Else
      Update the position of the leader by Eq (11.2)
   End-if
   If the fitness of the $leader < gBest$
      $Temp$ =$gBest$; $gBest$=$leader$; $leader$=$Temp$; (update Global optimum)
   End-if
 End
$Iter$=$Iter$+1;
End-while
```

The pseudo-code serves as a guide for the implementation of the proposed mCOA.

3. Simulation test setup on benchmark functions

The proposed mCOA was tested on thirteen commonly used standard benchmark test functions to establish its performance [17]. The test functions consist of the same functions used in the literature of the original COA, the details of which can be found in the reference [6]. The algorithm was coded in the MATLAB software (R2018a), using an HP Pavilion laptop computer with AMD A-8-6410 APU, an AMD Radeon R5 Graphics processor of a 2.00GHz clock speed, a RAM size of 4.00GB, and a 64-bit operating system. Table 1 provides detailed information on the benchmark function.

Table 1. Detail information of benchmark functions.

No.	Function	Search range	Global optimum	Dimension
1	F1	[-100, 100]	0	30
2	F2	[-10, 10]	0	30
3	F3	[-100, 100]	0	30
4	F4	[-100, 100]	0	30
5	F5	[-30, 30]	0	30
6	F6	[-100, 100]	0	30
7	F7	[-1.28, 1.28]	0	30
8	F8	[-500, 500]	-12,569	30
9	F9	[-5.12, 5.12]	0	30
10	F10	[-32, 32]	0	30
11	F11	[-600, 600]	0	30
12	F12	[-50, 50]	0	30
13	F13	[-50, 50]	0	30

The simulation parameter settings used in the simulation experiment are presented in Table 2. These contain all the parameter settings used in the simulation of the original COA in the literature to facilitate a fair comparison.

Table 2. Simulation parameters settings.

Parameter	Value
Population size	30
a	0.5
B	1.2
ω_{min}	0.001
ω_{max}	0.8
Maximum iteration	1000
Number of runs	30

On each test function, the simulation was executed thirty (30) times, and some relevant statistical indicators were calculated, including the best value (Min), the worst value (Max), the mean value (Avg), and the standard deviation (Std). This depicts the possible best performance, the worst performance, the average performance, and the possible deviation when applied to a real-world optimization problem. Since no algorithm has been deemed globally optimal, relatively better-performing algorithms were searched for based on these statistical indicators [18].

To establish the efficacy of the proposed mCOA, the simulation results were compared to the original COA and some other state-of-the-art metaheuristic algorithms in [6], namely the genetic algorithm (GA) and the particle swarm optimization (PSO) algorithm. The comparison of simulation results on the thirteen (13) test functions are presented in Table 3, where the boldened numbers represent the best performances.

Table 3. Results comparison on benchmark functions.

Function		GA	PSO	COA	mCOA
F1	Min	1.5245E+00	9.3905E-08	2.8240E-47	**3.9156E-165**
	Max	7.4966E+00	5.0237E-05	4.3670E-23	**1.114 E-104**
	Avg	3.6872E+00	2.8706E-06	1.4570E-24	**3.7133 E-106**
	Std	1.3063E+00	9.0693E-06	7.9728E-24	**2.0339 E-105**
F2	Min	2.2110E-01	1.1259E-04	4.0984E-24	**3.6289 E-86**
	Max	6.5960E-01	1.0800E-02	9.0834E-13	**3.3722 E-51**
	Avg	4.7510E-01	1.8000E-03	5.6782E-14	**1.1241 E-52**
	Std	1.0590E-01	2.4000E-03	1.8801E-22	**6.1568 E-52**
F3	Min	2.8924E+03	3.1124E+01	9.1935E-48	**1.829E-161**
	Max	1.1739E+04	4.5637E+02	8.5873E-22	**5.1103E-76**
	Avg	5.1669E+03	1.5501E+02	2.8626E-23	**1.7034E-77**
	Std	2.1462E+03	1.0194E+02	1.5678E-22	**9.3301E-77**
F4	Min	4.9927E+00	1.0955E+00	9.1191E-25	**5.5155E-82**
	Max	2.1162E+01	6.1639E+00	3.0541E-12	**3.8707E -34**
	Avg	9.1426E+00	2.5264E+00	1.4861E-13	**1.2902E-35**
	Std	3.0510E+00	1.0886E+00	6.0000E-13	**7.0669E-35**
F5	Min	1.7212E+02	**1.6161E+01**	2.8344E+01	2.7475E+01
	Max	9.0158E+02	1.0518E+02	6.6751E+01	**2.8597E+01**
	Avg	4.1400E+02	3.7766E+01	3.1804E+01	**2.8173E+01**
	Std	1.7821E+02	2.4842E+01	7.8738E+00	**2.5579E-01**
F6	Min	1.5385E+00	**3.8856E-08**	4.7900E-02	2.8573E-03
	Max	7.5916E+00	**9.8496E-06**	5.4550E-01	2.1669E-02
	Avg	3.6049E+00	**1.5514E-06**	1.9360E-01	7.6979E-03
	Std	1.5701E+00	**2.4394E-06**	1.2700E-01	3.7623E-03
F7	Min	5.5600E-02	1.0200E-02	3.0816E-04	**9.8849E-05**
	Max	2.0180E-01	5.6100E-02	2.2100E-02	**3.7090E-03**
	Avg	1.2240E-01	2.5300E-02	4.4000E-03	**1.1750E-03**
	Std	3.9000E-02	1.0300E-02	4.8000E-03	**1.0394E-03**
F8	Min	**-1.1491E+04**	-7.8895E+03	-9.2190E+03	-8.9888E+03
	Max	**-1.0316E+04**	-4.8503E+03	-6.3040E+03	-6.3108E+03
	Avg	**-1.0872E+04**	-6.5312E-03	-7.5838E+03	-7.4229E+03
	Std	**3.0684E+02**	8.0228E+02	7.3974+02	6.1435E+02
F9	Min	2.8791E+00	9.5516E+01	**0.0000E+00**	**0.0000E+00**
	Max	1.4370E+01	9.5516E+01	5.0591E-12	**0.0000E+00**
	Avg	7.7912E+00	4.8255E+01	1.8948E-13	**0.0000E+00**
	Std	2.8269E+00	1.6229E+01	9.2219E-13	**0.0000E+00**

Continued on next page

Function		GA	PSO	COA	mCOA
F10	Min	3.2410E-01	4.6059E-05	**8.8818E-16**	**8.8818E-16**
	Max	1.6306E+00	2.8136E+00	1.9263E-08	**4.4409E-15**
	Avg	7.3590E-01	1.2718E+00	6.4475E-10	**1.2434E-15**
	Std	3.1770E-01	8.2150E-01	3.5164E-09	**1.0840E-15**
F11	Min	9.1080E-01	3.8588E-08	**0.0000E+00**	**0.0000E+00**
	Max	1.0681E+00	7.0900E-02	6.1950E-14	**0.0000E+00**
	Avg	1.0221E+00	1.4800E-02	2.1057E-15	**0.0000E+00**
	Std	3.0900E-02	1.6700E-02	1.1305E-14	**0.0000E+00**
F12	Min	2.9000E-03	3.0857E-01	1.2000E-03	**5.3217E-05**
	Max	2.4180E-01	9.3380E-01	1.3627E+00	**5.4852E-04**
	Avg	3.1500E-02	1.7650E-01	9.7700E-02	**1.4757E-04**
	Std	4.5400E-02	2.4850E-01	2.8540E-01	**1.1308E-04**
F13	Min	7.6300E-02	**3.4979E-07**	7.0200E-02	4.2545E-03
	Max	**6.1460E-01**	6.2250E-01	2.9777E+00	2.8156E+00
	Avg	3.0580E-01	**3.4600E-02**	5.4750E-01	2.3575E-01
	Std	1.3670E-01	**1.1420E-01**	5.7420E-01	5.2133E-01

4. Results

The results presented in Table 2 show the performance of the proposed mCOA compared to GA, PSO, and COA. It produced improvements for the GA, PSO, and COA on functions F1–F12. In these test functions, the mCOA produced the least values of the min, max, avg, and std, which indicates a better performance for minimization optimization problems, and represents ten (10) out of the thirteen (13) benchmark test functions tested. This represents an average performance of 76.92% on the test functions. For the cases of F6 and F13, the PSO outperformed the other algorithms, while the GA outperformed the other algorithms on F13.

The convergence of the two algorithms is presented in Figures 1–13, which indicate the convergence process from the first iteration to the last iteration. This provides a clearer comparison of the performances of the two algorithms to effectively justify the superior performance of the mCOA over the COA.

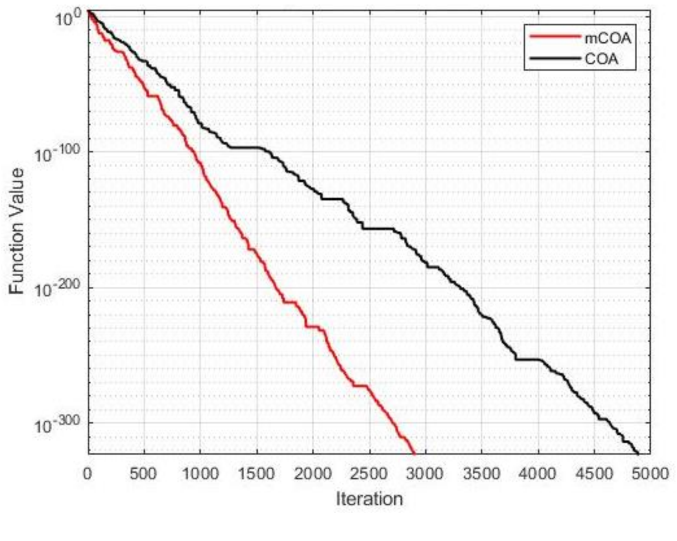

Figure 1. Convergence characteristics of function F1.

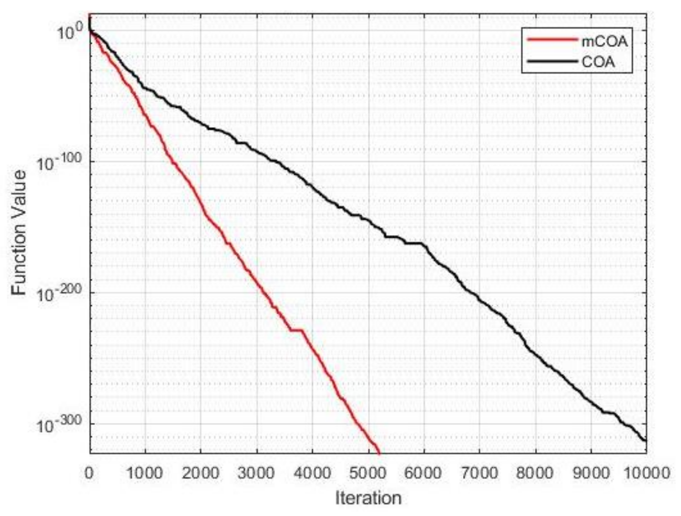

Figure 2. Convergence characteristics of function F2.

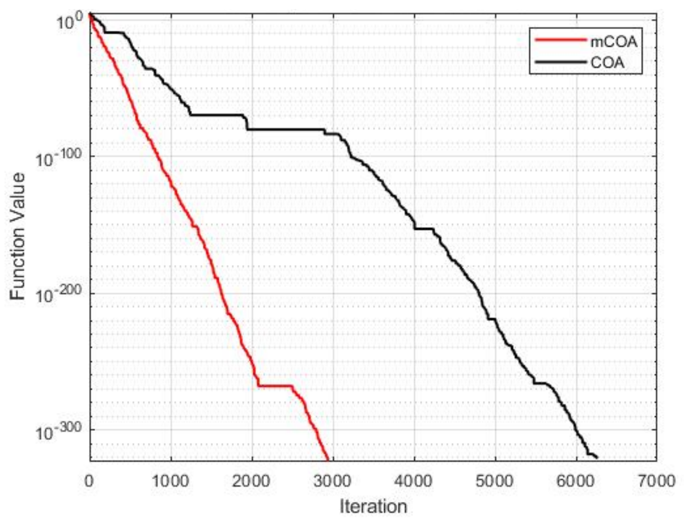

Figure 3. Convergence characteristics of function F3.

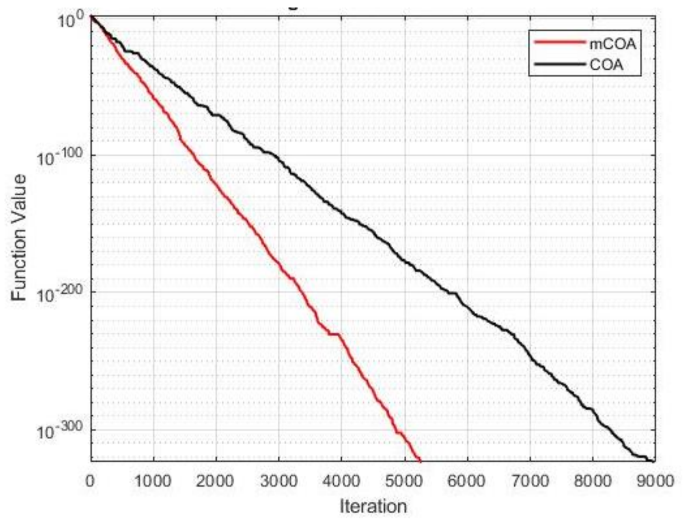

Figure 4. Convergence characteristics of function F4.

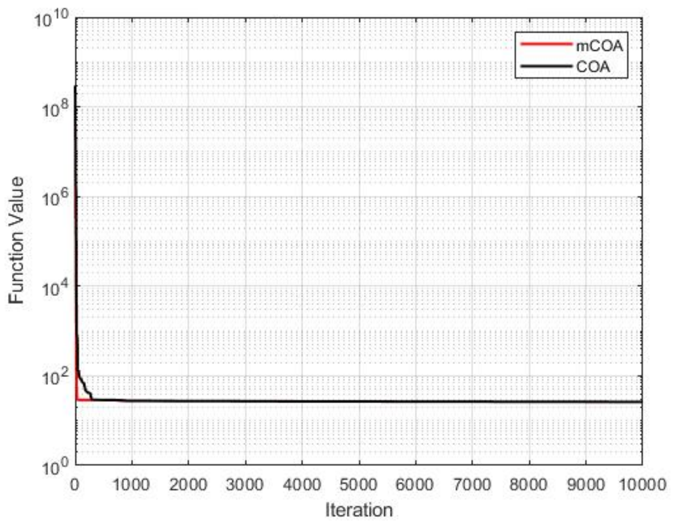

Figure 5. Convergence characteristics of function F5.

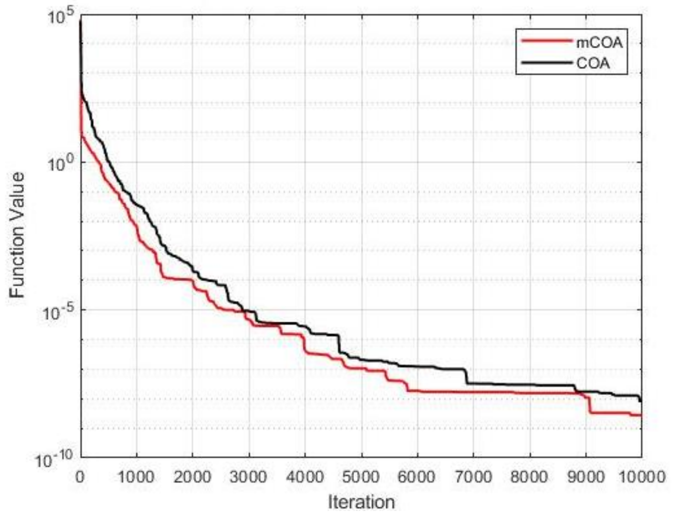

Figure 6. Convergence characteristics of function F6.

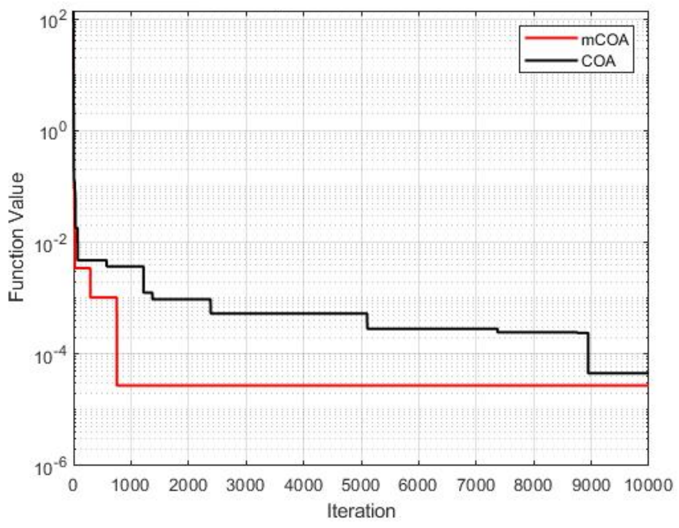

Figure 7. Convergence characteristics of function F7.

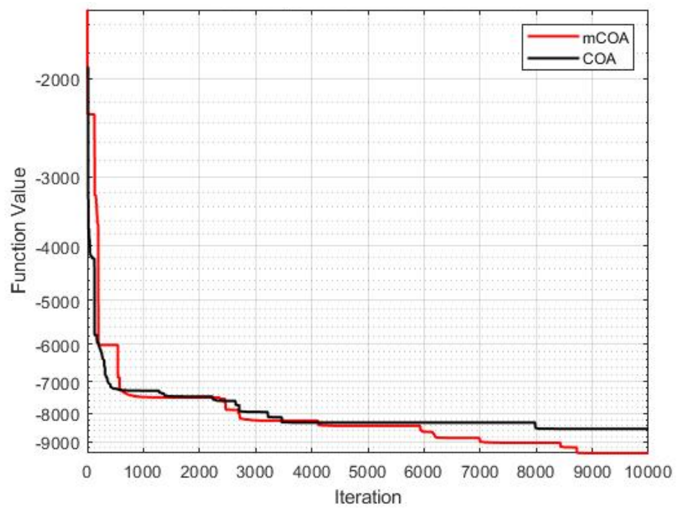

Figure 8. Convergence characteristics of function F8.

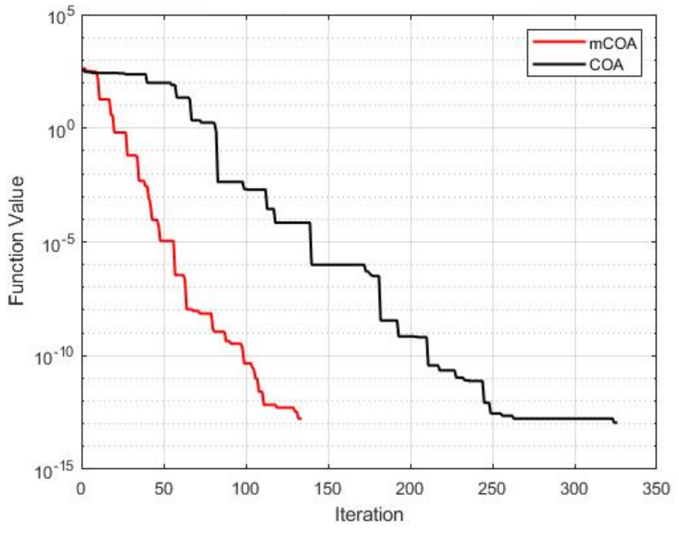

Figure 9. Convergence characteristics of function F9.

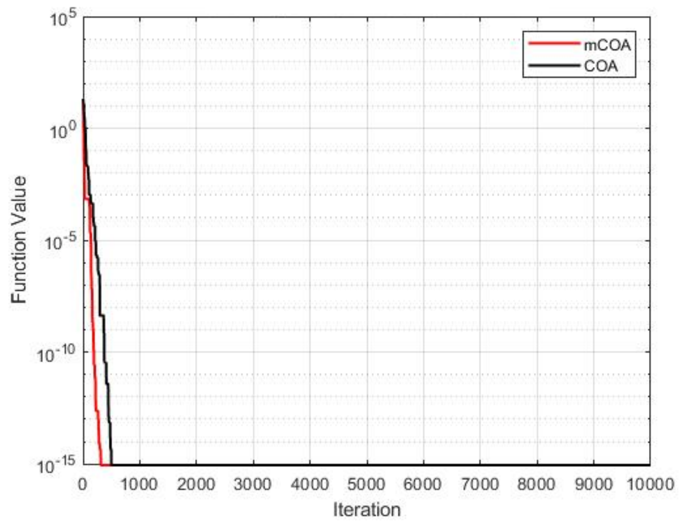

Figure 10. Convergence characteristics of function F10.

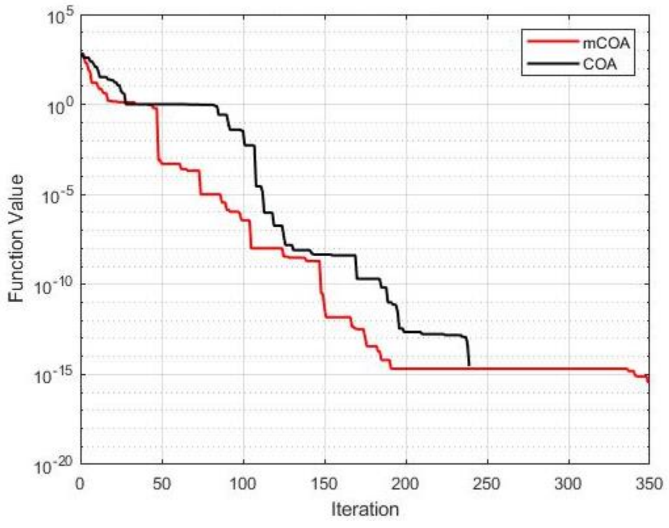

Figure 11. Convergence characteristics of function F11.

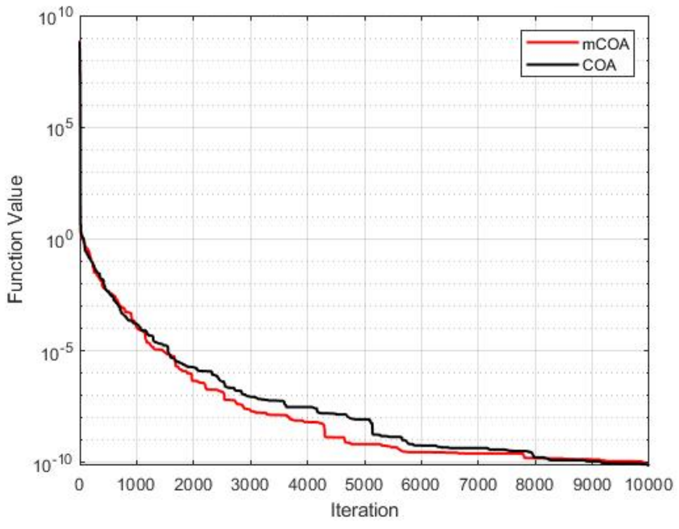

Figure 12. Convergence characteristics of function F12.

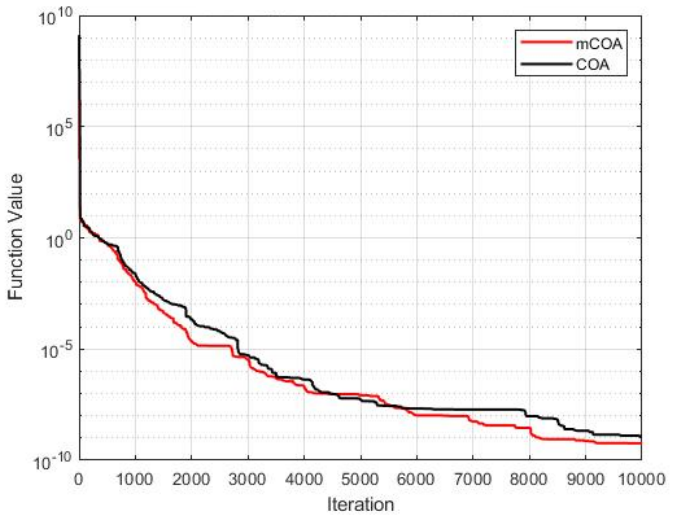

Figure 13. Convergence characteristics of function F13.

In F1–F4, F7, F9, and F10, the mCOA has better convergence characteristics than the COA with a great margin. It starts performing better effectively from the initial iteration to the final iteration, thus producing a better final optimization value on these benchmark functions. In the cases of F6, F8, and F11–F13, the mCOA produced a slightly better convergence, on average, than the COA. The two algorithms, mCOA and COA, produced fast convergences in F5.

5. Conclusions

A modification of the COA was developed to improve its global performance by incorporating an adaptive sigmoid inertia weight-based technique in the exploration phase. The mCOA was tested on the same 13 standard benchmark test functions used for the original COA. The simulation outcome using the MATLAB software was compared to that of the COA, PSO algorithm, and the GA. The mCOA outperformed the other algorithms on most of the test functions, with a score of 10 out of the 13 functions, while maintaining a competitive performance on the other 3 test functions. Therefore, an enhanced version of the COA was developed for a better global performance. Based on the exceptional simulation performance of the mCOA, it is recommended for applications in real-life optimization problems, especially in the field of engineering without any reservation.

All tests were obtained by Matlab software publicly available in https://github.com/etwumasi/code_appendix.

Conflict of interest

All authors declare no conflict of interest regarding the publication of this paper.

References

1. V. Soni, A. Sharma, V. Singh, A critical review on nature inspired optimization algorithms, *IOP Conf. Ser.: Mater. Sci. Eng.*, **1099** (2021), 012055. https://doi.org/10.1088/1757-899x/1099/1/012055
2. A. F. Seini Yussif, T. Seini, Improved F-parameter mountain gazelle optimizer (IFMGO): a comparative analysis on engineering design problems, *IRJET*, **10** (2023), 810–816.
3. N. Khodadadi, E. S. M. El-Kenawy, F. De Caso, A. H. Alharbi, D. S. Khafaga, A. Nanni, The mountain gazelle optimizer for truss structures optimization, *Appl. Comput. Intell.*, **3** (2023), 116–144. https://doi.org/10.3934/aci.2023007
4. R. Rani, S. Jain, H. Garg, A review of nature-inspired algorithms on single-objective optimization problems from 2019 to 2023, *Artif. Intell. Rev.*, **57** (2024), 126. https://doi.org/10.1007/s10462-024-10747-w
5. P. Agarwal, S. Mehta, Nature-inspired algorithms: state-of-art, problems and prospects, *International Journal of Computer Applications*, **100** (2014), 14–21. https://doi.org/10.5120/17593-8331
6. I. Naruei, F. Keynia, A new optimization method based on COOT bird natural life model, *Expert Syst. Appl.*, **183** (2021), 115352. https://doi.org/10.1016/j.eswa.2021.115352
7. R. R. Mostafa, A. G. Hussien, M. A. Khan, S. Kadry, F. A. Hashim, Enhanced COOT optimization algorithm for dimensionality reduction, *Proceedings of Fifth International Conference of Women in Data Science at Prince Sultan University*, 2022, 43–48. https://doi.org/10.1109/WiDS-PSU54548.2022.00020

8. P. K. Mandal, A review of classical methods and nature-inspired algorithms (NIAs) for optimization problems, *Results in Control and Optimization*, **13** (2023), 100315. https://doi.org/10.1016/j.rico.2023.100315
9. M. Jain, V. Saihjpal, N. Singh, S. B. Singh, An overview of variants and advancements of PSO algorithm, *Appl. Sci.*, **12** (2022), 8392. https://doi.org/10.3390/app12178392
10. A. F. Seini Yussif, E. Twumasi, E. A. Frimpong, Modified mountain gazelle optimizer based on logistic chaotic mapping and truncation selection, *IRJET*, **10** (2023), 1769–1776.
11. A. F. Seini Yussif, E. Twumasi, E. A. Frimpong, Performance enhancement of elephant herding optimization algorithm using modified update operators, *Jurnal Nasional Teknik Elektro*, **2** (2023), 109–118. https://doi.org/10.25077/jnte.v12n2.1124.2023
12. N. K. Prah II, E. A. Frimpong, E. Twumasi, Modified individual experience mayfly algorithm, *Carpathian Journal of Electrical Engineering*, **16** (2022), 62–74.
13. A. Bright, A. K. Emmanuel, T. Elvis, F. A. Emmanuel, Enhanced adaptive simulated based artificial gorilla troop optimizer for global optimisation, *IJEEAS*, **6** (2023), 63–76.
14. M. Aslan, İ. Koç, Modified coot bird optimization algorithm for solving community detection problem in social networks, *Neural Comput. Appl.*, **36** (2024), 5595–5619. https://doi.org/10.1007/s00521-024-09567-4
15. E. H. Houssein, F. A. Hashim, S. Ferahtia, H. Rezk, Battery parameter identification strategy based on modified coot optimization algorithm, *J. Energy Storage*, **46** (2022), 103848. https://doi.org/10.1016/j.est.2021.103848
16. Z. Chen, Y. Wang, T. H. T. Chan, X. Li, S. Zhao, A particle swarm optimization algorithm with sigmoid increasing inertia weight for structural damage identification, *Appl. Sci.*, **12** (2022), 3429. https://doi.org/10.3390/app12073429
17. T. Seini, A. F. S. Yussif, I. M. Katali, Enhancing mountain gazelle optimizer (MGO) with an improved F-parameter for global optimization, *IRJET*, **10** (2023), 921–930.
18. N. Moniz, H. Monteiro, No free lunch in imbalanced learning, *Knowl.-Based Syst.*, **227** (2021), 107222. https://doi.org/10.1016/j.knosys.2021.107222