Research Article

Ahmad Salim*, Wisam K. Jummar, Farah Maath Jasim, and Mohammed Yousif

Eurasian oystercatcher optimiser: New meta-heuristic algorithm

https://doi.org/10.1515/jisys-2022-0017
received November 13, 2021; accepted January 09, 2022

Abstract: Modern optimisation is increasingly relying on meta-heuristic methods. This study presents a new meta-heuristic optimisation algorithm called Eurasian oystercatcher optimiser (EOO). The EOO algorithm mimics food behaviour of Eurasian oystercatcher (EO) in searching for mussels. In EOO, each bird (solution) in the population acts as a search agent. The EO changes the candidate mussel according to the best solutions to finally eat the best mussel (optimal result). A balance must be achieved among the size, calories, and energy of mussels. The proposed algorithm is benchmarked on 58 test functions of three phases (unimodal, multimodal, and fixed-diminution multimodal) and compared with several important algorithms as follows: particle swarm optimiser, grey wolf optimiser, biogeography based optimisation, gravitational search algorithm, and artificial bee colony. Finally, the results of the test functions prove that the proposed algorithm is able to provide very competitive results in terms of improved exploration and exploitation balances and local optima avoidance.

Keywords: meta-heuristic, optimisation, Eurasian oystercatcher optimiser, Eurasian oystercatcher

1 Introduction

Optimisation may be used anywhere, as engineering design or industrial design, planning different activities, etc. Optimisation is the process of selecting the best solution for a particular problem with a large number of alternatives. Using optimisation techniques as meta-heuristics is the best option to solve multi-solutions problems in an intelligent way [1].

Meta-heuristic methods have higher level than heuristic search techniques and are used in complex research areas to search for suitable solutions to problems that do not have specific solutions. These methods strike a balance between a suitable solution and its search time. The appropriate solution is selected from a group of solutions during several cycles by applying certain rules or criteria. In any optimisation method, exploration and exploitation are the two common features used to solve the problem. Exploration is the stage of searching for solutions within the search space by expanding this field to unexplored areas, and exploitation focuses on search areas that may contain solutions close to optimal solutions [1]. Meta-heuristic algorithms have recently spread and used to solve many problems in different disciplines due to their flexibility and simplicity. The simplicity of their concepts greatly contributes to their spread. Furthermore, meta-heuristics can also be classified according to the number of solutions used at the

*** Corresponding author: Ahmad Salim,** Middle Technical University, Baghdad, Iraq, e-mail: pro.ahmadsalimiq@gmail.com, ahmadsalim@mtu.edu.iq
Wisam K. Jummar: University of Anbar, Anbar, Iraq, e-mail: wisamkhalid6@gmail.com, wisamkhalid6@uoanbar.edu.iq
Farah Maath Jasim: University of Anbar, Anbar, Iraq, e-mail: Farahmaath86@uoanbar.edu.iq
Mohammed Yousif: Ministry of Youth & Sport, Anbar, Iraq, e-mail: Mohammed.alyousif1991@gmail.com

same time, which is a single solution or population-based [2]. This research article focuses on population-based.

Generally, meta-heuristics algorithms can be classified into three main categories: the first category is observing the behaviour of animals (nature-inspired), the second category is observing the physical phenomena of materials, and the last category is the concept of natural evolution [3].

Natural animal behaviour is simulated by observing and analyzing their social behaviour, which is considered meta-heuristics based on population, such as in particle swarm optimisation (PSO) proposed by Kennedy and Eberhart [4]. Bird flow is the inspiration for this algorithm. An artificial bee colony (ABC) developed by Basturk [5] was based on bacterial foraging for distributed optimisation [6]. One of the most famous algorithms in this field is the ant colony optimisation proposed by Dorigo et al. [7], which is inspired by the method that ants use in searching for food and confirming the shortest paths that lead them to this food by using the pheromone that is secreted every time this path is used. Proposed by Mirjalili et al. [8], grey wolf optimiser (GWO) is another technique that simulates the social behaviour of living things by mimicking the hunting process of grey wolves, which is characterised by group hunting where the group members surround and continually move around the prey from all directions. In addition to the whale optimisation algorithm inspired by the social humpback whales behaviour and proposed by Mirjalili and Lewis [9], the crow swarm optimisation algorithm mimics the behaviour of American crow [10]. Majhi and Biswal presented another example of algorithms that simulate animal behaviour, in which an ant lion optimiser-based algorithm was rolled out alongside K-means. This technique has been verified after testing with eight datasets. Many other nature-inspired algorithms have been developed, such as marine predators algorithm [11], the ant lion optimiser [12], red deer algorithm [13], and doctor and patient optimisation algorithm [14]. These studies demonstrate how to learn from the behaviour of wild animals or insects and transform their behaviour into algorithms that may be used to solve many complex problems.

Observing physical rules is the second type of meta-heuristic search, in which algorithms are inspired by simulating these rules. The most famous among the many algorithms in this field are as follows: simulated annealing [15], gravitational search algorithm (GSA) [16], black hole [17], and ray optimisation [18]. Many other methods are also inspired by the same pattern.

The last type of meta-heuristics is inspired by the concept of evolution in nature. In 1992, Holland introduced a technique called genetic algorithm, one of the most important algorithms in this field [19]. This algorithm simulates Darwin's theory of evolution. In this method, a random group is created and developed through the generations by selecting the best individuals or solutions in this generation and merging them to form a new generation with better qualities than the previous one. The improvement process may continue for several generations. Many algorithms follow this approach, such as biogeography-based optimiser (BBO) [20], genetic programming [21], and evolutionary programming [22].

However, meta-heuristic optimisations until now are considered an open challenge. Thus, needing the meta-heuristic algorithm is an interesting topic. The meta-heuristic algorithm is used to discover the best solution over a set of candidate solutions to solve any optimisation problem by determining the minimum or maximum objective functions for a specific problem such as travel salesmen, train scheduling, timetabling, and shape optimisation.

This study proposes a new meta-heuristic nature-inspired algorithm called Eurasian oystercatcher optimiser (EOO) inspired by the food behaviour of Eurasian oystercatcher (EO) in searching for mussels. Oyster hunter optimisation, one of the algorithms of swarm, was devised based on the hypothesis of caloric maximisation developed by Meire and Ervynck [23]. The behaviour of the oystercatchers and its feeding style are observed. A balance among calories gained, lost time, and energy consumed is a criterion for finding an appropriate solution. The main contributions can be summarised as follows: propose a new meta-heuristic optimisation algorithm inspired by the food behaviour of EO and compare the proposed algorithm with other well-known meta-heuristics optimisation algorithms. The main advantages of the proposed EOO algorithm are giving a proper balance between exploration and exploitation and preventing a local minimum problem. Moreover, it is simple, unique, fast, and unbiased.

The remainder of this article is arranged as follows. Section 2 illustrates the inspiration process and mathematical model of the EOO algorithm. Section 3 discusses the experiment results. Section 4 describes the comparison of the proposed EOO algorithm with the most well-known meta-heuristic algorithms. Finally, Section 5 concludes the article with future works.

2 EOO

EOO is a nature-inspired optimisation algorithm. In this section, the inspiration process of the algorithm will be described. Moreover, the mathematical model will be explained.

2.1 Inspiration

Meire and Ervynck [23] determined the value of gain from different sizes of prey by measuring the time required to open the mussels, the energy the bird consumed in this process, and the calories it would get from this hunting. Mussels with a length of 50 mm or more provide greater amounts of calories for every minute wasted in the opening of the shells but require more time and energy to open their shell through stabbing compared with prey of smaller sizes. From the above assumptions of the model, the eater of mussels would focus on eating large sizes of prey. However, the bird does not prefer large mussels. This contradiction between reality and assumption was explained as follows [23,24]:

The first assumption is that the average profitability from large prey is less than the maximum benefit because some mussels have extremely strong shells that cannot be opened. When calculating the profitability obtained from the prey, a defect in this step was found because the researchers did not focus on the mussels that the bird was able to open. Mussel catchers sometimes choose large prey that they cannot open even with the utmost effort. The time lost in dealing with large, albeit, breakable prey reduces the average benefit of these mussels. If this factor is considered, then a new prediction model will emerge; that is, the mussel catchers' focus should be on the preys with 50 mm length rather than those of a very large size. However, mussel catchers tend to prefer prey with a size between 30 and 45 mm. Therefore, they waste time on the huge, unbreakable mussels, leading to failure in explaining the food selection behaviour of the mussel catchers [23].

The other hypothesis indicates that a layer of barnacles covers the surface of large mussels. Hence, the opening of the shell is almost impossible, and this type of prey is not preferred by the mussel catchers. This hypothesis is supported by observing the mussel catchers while eating. This bird does not eat mussels covered with barnacles even though they contain many calories. The amount of barnacle covering the mussels increases potentially with their size, thus rendering them unbreakable and therefore not a preferred option. If the time for opening the mussels is considered in the mathematical model, in addition to the time lost in unsuccessful attempts to open some prey and the range of sizes suitable for predation, then the mussel catchers must focus on the prey size in the range of 30–45 mm. Thus, the researchers were able to demonstrate that the mussel catchers eat almost perfectly according to scientific explanations and assumptions [23,24].

2.2 Mathematical model

This section presents the mathematical model of search and the appropriate mussel selected by EO. The main aim of EO is to balance their energy and the calories obtained from the mussels. Energy and calories are directly related to the size of mussels. When the mussels' length increases, the calories and the time required for opening also increase. Thus, high energy is required from EO waste. Equations (1) and (2) represent the behaviour of EO in the search process.

$$Y = T + E + L * r * (X_{\text{best}} - X_{i-1}), \tag{1}$$

$$X_i = X_{i-1} * C, \tag{2}$$

where Y represents the final energy of EO in each solution (iteration), X_i is a position of candidate mussel, L indicates the length of the mussel and is a random value ranging from 3 to 5 that represents the range of optimal length of the T is the time required to open the current mussel (solution), and its value relies on L according to equation (3). The numbers (3 and 5) have been used in equation (3) based on the size of the appropriate mussel, which was mentioned in the inspiration section. E is the EO's energy at the current time, which is obtained from equation (4) and rely on the iteration's value because the EO's value decreases every time. r is a random value between 0 and 1 to give more randomness and discover new places in the search area. C in the equation (2) represents the caloric value, which is obtained from the equation (5) and depends on the length of the mussel.

$$T = \left(\left(\frac{L-3}{5-3}\right) * 10\right) - 5, \tag{3}$$

$$E = \left(\frac{i-1}{n-1}\right) - 0.5, \quad \text{where } i > 1, \tag{4}$$

$$C = \left(\left(\frac{L-3}{5-3}\right) * 2\right) + 0.6. \tag{5}$$

Equation (3) presents the value between (5) and (−5) and value that result from the equation (5) in the range from 0.6 to 0.8. These values were selected based on trial and testing. It is important to mention that if the time is negative, this means that the time required to break the mussels may be greater than the ability of the bird and *vice versa*, which is considered as a constraint. E value was obtained from equation (4) and linearly decreased from 0.5 to −0.5, where i represents the iteration value that starts with the number of iteration and ends with one, and in last two iterations, the E value will be fixed. T and E, which are the required time and energy to open the candidate mussel (must be less than EO's energy), may consequently be negative values. Figure 1 illustrates the search process followed by EO to find appropriate mussels or prey. The T value in equation (1) and C in equation (2) rely on L, a random value that changes inconstantly. This condition allows EO to reach any position in the search space and prevent a local minimum problem, thus emphasising the exploration every time. Figure 2 illustrates the pseudo-code of the EOO to determine how EOO can solve optimisation problems.

Figure 1: EO and the candidate mussels.

Theoretically, the following points explain the main features of EOO that help in solving optimisation problems:

```
Initialize the EO population $X_i$ (i=1,2,…..,n)
Calculate the fitness of each search agent
$X_{best}$ = the best solution in search agent
While (i>0)
    For each solution in search agent
        L=random (3,5)
        Calculate T, E and C based on equations 3, 4 and 5
        Update the position of solution based on equations 1 and 2
    End for
    Calculate the fitness of each search agent
    $X_{best}$ = the best solution in search agent
End while
Return $X_{best}$
```

Figure 2: Pseudo code of EOO algorithm.

Adopting the time required to open the mussel, which is calculated using the bird's energy and the size of the mussel as parameters to measure the expected position of the target food, thereby improving the quality of selection.

The random values entered throughout optimisation help explore new areas at each iteration. Thus, prevent a local minimum problem.

Exploration and exploitation are guaranteed by the random values used in each stage of optimisation.

3 Results and discussion

Matlab 2018B was applied on Dell Core i7 processor and 16 GB RAM to test the algorithm and measure its efficiency. Benchmark test functions were adopted to measure the algorithm's efficiency and ability to solve optimisation problems.

EOO was tested by 58 benchmark functions classified into three types: unimodal, multimodal, and fixed-diminution multimodal. The unimodal test functions are listed in Table 1, where D represents the dimension of the test function, and Range indicates the accepted range from upper to lower value in each function and Opt is the optimal value. Multimodal and fixed multimodal benchmark functions are presented in Table 2, where Type indicates the type of function F for fixed and N for N-dimensional multimodal. The test functions were as follows: continuous, convex, differentiable, and separable. In the continuous function, the small changes in the input must be sufficient to produce a large change in output. The function is differentiable when the function derivable and this function is continuous at each point in the domain. In the evaluation of algorithms, Separable indicates a measure of the difficulty of solving a problem.

Table 3 (average and standard deviation) confirms that EOO achieved the optimal value in nine benchmark functions (F4, F5, F6, F9, F11, F13, F14, F15, and 19), and Table 4 shows that EOO produced the optimal value in 22 multimodal test functions. Ultimately, EOO achieved the optimal value in 31 benchmark functions and best value in 49 test functions from all 58 test functions in two tables. Generally, unimodal benchmarks functions are suitable for exploitation problem, and multimodal functions are appropriate for the exploration ability of the algorithm. Therefore, EOO performance was superior in terms of exploitation (Table 3) and exploration (Table 4). Finally, EOO algorithm produced 9 optimal values in unimodal benchmark functions, 11 in N-dimensional multimodal functions and 11 in fixed multimodal functions, implying its good balance between exploitation and exploration.

Table 1: Unimodal Benchmark functions

Function	Equation	Test name	*D*	Range	Opt
F1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	Sphere	30	100 to −100	0
F2	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	Schwefel 2.22	2	100 to −100	0
F3	$f_3(x) = \max_i\{\lvert x_i \rvert, 1 \le i \le n\}$	Schwefel 2.21	2	100 to −100	0
F4	$f_4(x, y) = -200e^{-0.2}\sqrt{x^2 + y^2}$	Ackley 2	2	32 to −32	−200
F5	$f_5(x) = x_1^2 + x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$	Bohachevskyn N.1	2	100 to −100	0
F6	$f_6(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	Booth	2	10 to −10	0
F7	$f_7(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$	Zakharov	2	5.12 to −5.12	0
F8	$f_8(x) = (x_1 - 1)^2 + \sum_{i=2}^{d} i\,(2x_i^2 - x_{i-1})^2$	Dixon-Price	2	10 to −10	0
F9	$f_9(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right)$	Exponential	30	1 to −1	−1
F10	$f_{10}(x) = 0.26({x_1}^2 + {x_2}^2) - 0.48x_1x_2$	Matyas	2	10 to −10	0
F11	$f_{11}(x) = \sum_{i=1}^{n} x_i^{10}$	Schwefel 2.23	2	100 to −100	0
F12	$f_{12}(x) = \sum_{i=1}^{n} \lvert x_i \rvert$	Schwefel 2.20	2	100 to −100	0
F13	$f_{13}(x) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	Schaffer N2	2	36 to −36	0
F14	$f_{14}(x) = 0.5 + \frac{\sin^2(\cos(\lvert x^2 - y^2 \rvert)) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	Schaffer N3	2	100 to −100	0.00156
F15	$f_{15}(x) = 0.5 + \frac{\sin^2(x_1^2 - x_2^2)^2 - 0.5}{[1 + 0.001(x_1^2 - x_2^2)]^2}$	Schaffer N1	2	100 to −100	0
F16	$f_{16}(x) = \sum_{i=1}^{d}\left[\left(\sum_{j=1}^{d} x_j^i\right) - b_i\right]^2$	Power sum	4	4, 0	0
F17	$f_{17}(x) = \frac{1 + \cos(12\sqrt{x_1^2 + x_2^2})}{0.5(x_1^2 + x_2^2) + 2}$	Drop wave	2	5.12 to −5.12	−1
F18	$f_{18}(x, y) = (x + 10)^2 + (y + 10)^2 + e^{-x^2 - y^2}$	Brent	2	0, 20	10–200
F19	$f_{19}(x, y) = 100(y - x^3)^2 + (1 - x)^2$	Leon	2	0, 10	0
F20	$f_{20}(x) = f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} ix_i^2$	Sum squares	10	−10 to 10	0

4 Comparing with the related work

This section demonstrates the comparison of the proposed EOO algorithm with the most well-known meta-heuristic algorithms such as PSO, GWO, BBO, GSA, and ABC. In the test process, each benchmark function run for 30 times. Tables 3 and 4 illustrate the results of the proposed algorithm and its comparison with other meta-heuristic algorithms such as PSO, GWO, BBO, GSA, and ABC. The results proved that EOO is competitive against other meta-heuristic algorithms.

However, for Table 2, the min–max normalisation approach was used to replicate the findings and examine the discussion. Then it generates Figures 3 and 4, respectively. Figures 3 and 4 illustrate a comparison of the optimal value both of the unimodal function test and the multimodal function test with results of the proposed method EOO, additional with the PSO, GWO, BBO, GSA, and ABC algorithms results. Figure 3 shows a large match between the optimal values and the results of the proposed EOO, with GSA and ABC. Consequently, this convergence has been backed up by positive outcomes to the proposed EOO, whereas Figure 4 shows that the proposed EOO algorithm's ability to achieve values close to the optimal values, similar to the PSO algorithm.

Table 2: Multimodal Benchmark functions

Function	Equation	Type	Test name	*D*	Range	Opt
F21	$f_{21}(x) = -20\exp\left(^{-0.2}\sqrt[2]{\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	N	Ackley	2	10 to −10	0
F22	$f_{22}(x) = \sum_{i=1}^{n} \lfloor \lvert x_i \rvert \rfloor$	N	Step 1	30	100 to −100	0
F23	$f_{23}(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	N	Quartic	10	1.28 to −1.28	0 + rand
F24	$f_{24}(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	F	Six-Hump Camel	2	5 to −5	−1.0316
F25	$f_{25}(x) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $*[30 + (2x_1 + 3x_2)^2(18 - 32x_1 + 12x_1^2 - 48x_2 + 36x_1x_2 + 27x_2^2)]$	F	Goldstein Price	2	2 to −2	3
F26	$f_{26}(x) = 2x_1^2 + 1.05x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3$	F	Camel Three	2	−5 to 5	0
F27	$f_{27}(x) = x_1^2 + 2x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$	F	Bohachevskyn N.2	2	10 to −10	0
F28	$f_{28}(x) = \sin(x)\, e^{(1-\cos(y))^2} + \cos(x)\, e^{(1-\sin(x))^2} + (x - y)^2$	F	Brid	2	2pi to −2pi	−106.7645
F29	$f_{29}(x) = \left(\left\lvert \sin(x_1)\sin(x_2)\exp\left(\left\lvert 100 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right\rvert\right)\right\rvert + 1\right)^{0.1}$	N	Cross in Tiny	2	10 to −10	−2.06261
F30	$f_{30}(x) = -\frac{\sin^2(x-y)\sin^2(x+y)}{\sqrt{(x^2 + y^2)}}$	F	Keane	2	10, 0	−0.6737
F31	$f_{31}(x) = -(x_2 + 47)\sin\left(\sqrt{\lvert x_2 + \frac{x_1}{2} + 47\rvert}\right) - x_1\sin\left(\sqrt{\lvert x_1 - (x_2 + 47)\rvert}\right)$	F	Egg Holder	2	512 to −512	−959.641
F32	$f_{32}(x) = -\lvert \sin(x_1)\, exp(\lvert 1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\rvert)\rvert$	F	Holder	2	10 to −10	−19.2085
F33	$f_{33}(x) = f(x_1, x_2, \ldots, x_n) = 418.9829d - \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	F	Schwefel	2	500 to −500	0
F34	$f_{34}(x) = \sum_{i=1}^{d} \sin(x_i)\sin^{2m}\left(\frac{i x_i^2}{\pi}\right)$	N	Michalewicz	2	1.57 to 2.21	−1.8013
F35	$f_{35}(x) = f(x_1, x_2, \ldots, x_n) = 1 + \sum_{i=1}^{n}\sin^2(x_i) - 0.1e^{(\sum_{i=1}^{n} x_i^2)}$	N	Periodic	30	−10 to 10	0.9
F36	$f_{36}(x) = f(x_1, x_2, \ldots, x_n) = 1 + \sum_{i=1}^{n}(x^2 - i)^2$	N	Qing	10	−500 to 500	0
F37	$f_{37}(x) = f(x_1, x_2, \ldots, x_n) = 1 - \cos\left(\sqrt[2\pi]{\sum_{i=1}^{n} x_i^2}\right) + \sqrt[0.1]{\sum_{i=1}^{n} x_i^2}$	N	Salomon	10	−100 to 100	0
F38	$f_{38}(x) = f(x_1, x_2, \ldots, x_n) = (\sum_{i=1}^{n}\lvert x_i\rvert)\exp(-\sum_{i=1}^{n}\sin(x_i^2))$	N	Xin-She Yang N. 2	10	−2π to 2π	0
F39	$f_{39}(x) = f(x_1, x_2, \ldots, x_n) = (\sum_{i=1}^{n}\sin^2(x_i) - 0.1e^{(-\sum_{i=1}^{n} x_i^2)})e^{-\sum_{i=1}^{n}\sin^2\sqrt{\lvert x_i\rvert}}$	N	Xin-She Yang N. 4	10	−10 to 10	−1
F40	$f_{40}(x) = \sum_{i=1}^{n-1}(x_i^2)^{(x_{i+1}^2 + 1)} + x_{i+1}^2(x_i^2 + 1)$	F	Brown	1	−1 to 4	0
F41	$f_{41}(x) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1 - t)\cos(x_1) + s$	F	Branin	2	−5 to 15	0.3979

(*Continued*)

Table 2: *Continued*

Function	Equation	Type	Test name	*D*	Range	Opt
F42	$f_{42}(x) = -\sum_{4}^{i=1} \propto_i \exp(-\sum_{3}^{j=1} A_{ij}(x_j - p_{ij})^2)$	F	Hartmann 3	3	0, 1	−3.8628
F43	$f_{43}(x) = -\sum_{i=1}^{4} \propto_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - p_{ij})^2\right)$	F	Hartmann 6	6	0, 1	−3.8628
F44	$f_{44}(x, y) = -200e^{-2\sqrt{x^2+y^2}} + 5e^{\cos(3x)+\ \sin(3y)}$	F	Ackley N. 3	2	−32 to 32	−195.629
F45	$f_{45}(x, y) = -\cos(x_1)\cos(x_2)\exp(-(x - \pi)^2 - (y - \pi)^2)$	F	Easom	2	−100 to 100	−1
F46	$f_{46}(x) = \sum_{i=1}^{d}(x_1 - i)^2 - \sum_{i=1}^{d} x_i x_{i-1}$	N	Trid	6	−36 to 36	−50
F47	$f_{47}(x) = \frac{\sin(10\pi x)}{2x} + (x - 1)^2$	F	Gramacy & Lee	1	0.5 to 2.5	−0.869
F48	$f_{48}(x, y) = \cos(x)\sin(y) - \frac{x}{y^2+1}$	N	Adjiman	2	−2 to 2	−2.02181
F49	$f_{49}(x) = f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	N	Alpine N. 1	2	−1 to 2	0
F50	$f_{50}(x) = f(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \sqrt{x_i} \sin(x_i)$	N	Alpine N. 2	2	0, 10	2.2808n
F51	$f_{51}(x, y) = \lvert x^2 + y^2 + xy \rvert + \lvert \sin(x) \rvert + \lvert \cos(y) \rvert$	F	Bartels Conn	2	−500 to 500	1
F52	$f_{52}(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$	F	Beale	2	−4.5 to 4.5	0
F53	$f_{53}(x, y) = 10^5x^2 + y^2 - (x^2 + y^2)^2 + 10^{-5}(x^2 + y^2)^4$	F	Deckkers-Aarts	2	−20 to 20	−24771.09375
F54	$f_{54}(x, y) = x^2 + y^2 + 25(\sin^2(x) + \sin^2(y))$	N	Egg Crate	2	−5 to 5	0
F55	$f_{55}(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$	N	Himmelblau	2	−6 to 6	0
F56	$f_{56}(x, y) = \sin^2(3\pi x) + (x + 1)^2(1 + \sin^2(3\pi x)) + (y + 1)^2(1 + \sin^2(2\pi x))$	F	Levi N. 13	2	−10 to 10	0
F57	$f_{57}(x, y) = \sin(x + y) + (x - y)^2 - 1.5x + 2.5y + 1$	F	McCormick	2	−1.5 to 4	−1.9133
F58	$f_{58}(x, y, z) = \frac{4}{3}(x^2 + y^2 - xy)^{0.75} + z$	N	Wolfe	3	0 to 2	0

Table 3: Results of Unimodal Benchmark functions

F	EOO_AV	EOO_STD	PSO_AV	PSO_STD	GSA_AV	GSA_STD	ABC_AV	ABS_STD	BBO_AV	BBO_STD	UB	LB	OPT	*D*
1	5.62281×10^{-18}	1.34684×10^{-17}	2.54029×10^{-8}	7.98316×10^{-8}	2.14471×10^{-17}	5.63657×10^{-18}	0.001252987	0.000811156	2.774887×10^{-3}	6.887686×10^{-4}	100	−100	0	30
2	6.633×10^{-126}	2.0973×10^{-125}5	1.0534×10^{-55}	5.14643×10^{-55}	9.31038×10^{-11}	5.16071×10^{-11}	0.000253748	0.000152438	5.84342×10^{-09}	1.14509×10^{-8}	100	−100	0	2
3	3.0525×10^{-113}	9.595×10^{-113}	2.02803×10^{-51}	1.10053×10^{-50}	6.01947×10^{-11}	2.99344×10^{-11}	0.001662766	0.001098175	4.45571×10^{-07}	6.05727×10^{-07}	100	−100	0	2
4	−200	0	−200	0	−200	0	−199.9998	0.00010171	−200	0	32	−32	−200	2
5	0	0	1.66533×10^{-16}	5.8874×10^{-16}	0	0	1.42235×10^{-05}	1.41276×10^{-05}	3.01983×10^{-12}	1.615×10^{-11}	100	−100	0	2
6	0	0	6.96827×10^{-30}	2.47526×10^{-29}	1.85221×10^{-20}	1.43471×10^{-20}	4.51821×10^{-7}	5.78899×10^{-7}	8.95268×10^{-10}	2.22824×10^{-9}	10	−10	0	2
7	8.5643×10^{-282}	0	4.0826×10^{-104}	2.0807×10^{-103}	3.11163×10^{-20}	4.09121×10^{-20}	1.8417×10^{-9}	1.69934×10^{-9}	7.82734×10^{-16}	2.00333×10^{-15}	5.12	−5.12	0	2
8	3.6978×10^{-32}	5.76992×10^{-48}	0.085226667	0.408424714	7.21111×10^{-20}	7.48794×10^{-20}	2.19797×10^{-6}	2.14305×10^{-6}	1.29442×10^{-8}	1.65908×10^{-8}	10	−10	0	2
9	−1	0	−1	0	−1	0	−0.999941333	3.8213×10^{-5}	−0.999951	1.21343×10^{-5}	1	−1	−1	30
10	2.9141×10^{-232}	0	6.77411×10^{-66}	3.71033×10^{-65}	1.56958×10^{-21}	1.18656×10^{-21}	1.32931×10^{-7}	1.36422×10^{-7}	2.80296×10^{-8}	5.51136×10^{-8}	10	−10	0	2
11	0	0	0	0	1.7161×10^{-99}	4.233×10^{-99}	2.94618×10^{-31}	9.23942×10^{-31}	2.26294×10^{-69}	1.23937×10^{-69}	100	−100	0	2
12	7.9317×10^{-143}	2.0983×10^{-142}	9.14357×10^{-46}	5.00814×10^{-45}	1.52258×10^{-10}	6.4698×10^{-11}	0.000561859	0.000278038	1.69745×10^{-8}	3.52861×10^{-8}	100	−100	0	2
13	0	0	6.95737×10^{-16}	2.22941×10^{-15}	0.005331387	0.006958944	5.26439×10^{-9}	7.65922×10^{-9}	1.6606×10^{-13}	9.08585×10^{-13}	36	−36	0	2
14	0.001566855	2.2857×10^{-19}	0.015050889	0.069811002	0.003418706	0.002052954	0.001717237	0.000144094	0.0018911	0.000475574	100	−100	0.0015	2
15	0	0	1.33227×10^{-16}	5.70145×10^{-16}	0.020150599	0.031993769	1.06425×10^{-5}	8.70544×10^{-6}	4.38922×10^{-15}	2.34985×10^{-14}	100	−100	0	2
16	0.007999503	0.009026825	1863.431315	6520.088474	0.028399943	0.033204355	0.020449223	0.010801132	0.01795016	0.015663814	4	0	0	4
17	-9.3625×10^{-1}	1.1703×10^{-16}	−1	0	−0.98456	0.013050977	−0.999206	0.000601024	−0.96175	0.034917313	5.12	−5.12	−1	2
18	1.38×10^{-87}	0.00×10^{00}	1.3839×10^{-87}	0	48.08614	0.249794351	1.16373×10^{-9}	1.5321×10^{-9}	5.38078×10^{-20}	5.7944×10^{-20}	0	−20	$\times 10^{-20}$0	2
19	1.00×10^{00}	0.00×10^{00}	9.9851×10^{-12}	1.88994×10^{-11}	0.02696	0.008670525	0.000275141	0.000371486	0.4519644	0.358633755	10	0	0	2
20	2.90×10^{-58}	4.67×10^{-58}	2.08032×10^{-41}	4.63296×10^{-41}	1.39357×10^{-20}	8.2482×10^{-21}	1.9754×10^{-7}	3.46517×10^{-7}	2.15358×10^{-7}	2.06015×10^{-7}	10	−10	0	10

Table 4: Results of Multimodal Benchmark functions

F	EOO_AV	EOO_STD	PSO_AV	PSO_STD	GSA_AV	GSA_STD	ABC_AV	ABS_STD	BBO_AV	BBO_STD	UB	LB	OPT	*D*
21	8.88178×10^{-16}	0	1.954×10^{-15}	1.65589×10^{-15}	1.70833×10^{-10}	1.07462×10^{-10}	0.000346327	0.000177635	4.73617×10^{-9}	1.64321×10^{-8}	10	−10	0	2
22	9.28938×10^{-13}	1.33932×10^{-12}	1.18715×10^{-8}	1.94462×10^{-8}	2.22948×10^{-17}	5.39243×10^{-18}	0.542361333	0.457034324	1.063618333	0.207327884	100	−100	0	30
23	0.026719186	0.008754312	0.695013333	0.77173046	0.004470523	0.002057632	0.00892749	0.002970026	0.001438262	0.000917321	1.28	−1.28	0	10
24	−1.031628453	0	15.75971404	0.008859934	−1.0316	6.77522×10^{-16}	−1.0316	6.77522×10^{-16}	−1.0316	6.77522×10^{-16}	5	−5	−1.0316	2
25	3	0	3	0.463283793	3	0	3	0	3	0	15	−5	0.3979	2
26	7.487×10^{-267}	0	2.2713×10^{-100}	5.5674×10^{-48}	6.2521×10^{-21}	7.70769×10^{-21}	2.49639×10^{-9}	2.05859×10^{-9}	0.039818667	0.103253557	500	−500	−12569.5	30
27	0	0	2.96059×10^{-17}	3.56623×10^{-5}	0	0	3.18426×10^{-5}	2.95654×10^{-5}	0.065493007	0.101752452	10	−10	0	2
28	0	0	−103.1151095	5.78238×10^{-17}	−106.7645	7.2269×10^{-14}	−106.7645	7.2269×10^{-14}	−106.1160467	3.551725182	2pi	2pi	−106.765	2
29	−2.062611871	4.68111×10^{-16}	−2.062338712	19.98873646	−2.0626	1.35504×10^{-15}	−2.0626	1.35504×10^{-15}	−2.0626	1.35504×10^{-15}	10	−10	−2.06261	2
30	−0.673667521	1.17028×10^{-16}	−0.6737	0	−0.670896667	0.00417228	−0.67367	0	−0.519266	0.192338314	10	0	−0.6737	2
31	−959.6406627	0	95.54238667	0	−727.0281867	123.8680268	-6.3842×10^{111}	2.287×10^{112}	−791.28165	147.3338192	512	−512	−959.641	2
32	−19.20850257	3.74489×10^{-15}	−15.14022386	0	−19.17958667	0.042476839	N/A	N/A	−18.88504	1.771663385	1	−10	−19.2085	2
33	2.54551×10^{-5}	3.5714×10^{-21}	711.5143067	5.42017×10^{-15}	206.868	81.44365973	-1.9688×10^{113}	1.0349×10^{114}	46.71747279	64.84744095	500	−500	0	2
34	−1.80130341	0	−1.801086667	447.0917171	−1.8013	6.77522×10^{-16}	−1.878846667	0.078420492	−1.8013	2.942408×10^{1}	2.21	1.57	−1.8013	2
35	1	0	1.8879	2.48253×10^{-16}	1	0.537239841	7.76328	0.537239841	1.01135	0.001438749	10	−10	0.9	30
36	1.22668×10^{-29}	2.91652×10^{-30}	1.22272×10^{-29}	1.76386×10^{-30}	6.5934×10^{-17}	8.5558629	15.67116	8.5558629	0.00217097	0.001867631	500	−500	0	10
37	0.736497025	0.127799749	0.27762	0.070280524	0.285	0.053416719	0.364346	0.053416719	0.19987	0	100	−100	0	10
38	0.002569467	6.55414×10^{-5}	0.00173669	0.001895051	0.00056607	0.002339459	0.0077256	0.002339459	0.00087561	0.000210189	2pi	2pi	0	10
39	4.81432×10^{-33}	3.69286×10^{-33}	4.06806×10^{-31}	5.57028×10^{-31}	3.73555×10^{-22}	6.56764×10^{-5}	0.00026413	6.56764×10^{-5}	2.86356×10^{-12}	3.54836×10^{-12}	10	−10	10	−1
40	1.26689×10^{-38}	1.46288×10^{-38}	1.67448×10^{-41}	2.27058×10^{-41}	7.83788×10^{-18}	5.56283×10^{-10}	6.66508×10^{-10}	5.56283×10^{-10}	7.41788×10^{-9}	4.23×10^{-9}	4	−1	0	1
41	0.397887358	5.85139×10^{-17}	0.482471019	91.46533617	0.3979	0	0.39789	1.6938×10^{-16}	0.39789	1.6938×10^{-16}	15	−5	0.3979	2
42	−3.862779787	0	−3.862782148	0	−3.862796667	1.82574×10^{-5}	−3.8628	3.16177×10^{-15}	−3.8628	3.16177×10^{-15}	1	0	−3.8628	3
43	−3.271279686	0.063718337	−3.270474819	1.35504×10^{-15}	−3.3224	1.35504×10^{-15}	−3.3224	1.35504×10^{-15}	−3.27472	0.059394127	1	0	−3.3224	6
44	−195.6290283	0	−195.6290218	8.6681×10^{-100}	−195.629	5.78152×10^{-14}	−195.629	5.78152×10^{-14}	−195.629	5.78152×10^{-14}	32	−32	−195.629	2
45	−1	0	−1	0.001496152	−1	0	−1	0	−1	0	100	−100	−1	2
46	−50	0	−50	1.1292×10^{-16}	−50	0	−49.99740333	0.002460934	−50	0	36	−36	−50	6
47	−0.869011135	0	0.0625	376.9030802	−0.869	1.1292×10^{-16}	−2.8739	9.03362×10^{-16}	−0.769653	0.145673467	2.5	0.5	−0.869	1
48	-2.02×10^{00}	4.68×10^{-16}	−1.8758	0.277501784	−5.12666	0.000896103	-7.8498×10^{102}	1.754×10^{103}	−2.0218	0	2	−2	−2.02181	2
49	0	0	2.9976×10^{-15}	0.000359821	1.54218×10^{-11}	2.86025×10^{-6}	1.0436×10^{-5}	2.86025×10^{-6}	1.10005×10^{-11}	1.94804×10^{-11}	2	−1	0	2
50	−6239.018731	1421.18718	43.40452	193.6002591	−4035.32	3.9127×10^{100}	-2.7292×10^{116}	3.9127×10^{100}	−6.1295	0	10	0	2.2808n	2
51	1	0	1	0	1	0.000240832	1.00044	0.000240832	1	0	500	−500	1	2
52	0	0	0	0	2.74851×10^{-20}	2.66782×10^{-6}	2.58458×10^{-6}	2.66782×10^{-6}	1.66699×10^{-6}	1.87613×10^{-6}	4.5	−4.5	0	2

(*Continued*)

Table 4: *Continued*

F	EOO_AV	EOO_STD	PSO_AV	PSO_STD	GSA_AV	GSA_STD	ABC_AV	ABS_STD	BBO_AV	BBO_STD	UB	LB	OPT	*D*
53	−24776.51834	3.83477×10^{-12}	−24777	0	−39.6307	0.003626293	−24776.5159	0.003626293	−24776.5183	0	20	−20	−24771.09375	2
54	0	0	1.4323×10^{-103}	3.2027×10^{-103}	2.4452×10^{-19}	9.74048×10^{-7}	1.28548×10^{-6}	9.74048×10^{-7}	1.03361×10^{-14}	2.31075×10^{-14}	5	−5	0	2
55	0	0	4.73316×10^{-31}	4.32076×10^{-31}	2.56433×10^{-19}	3.2854×10^{-5}	5.02648×10^{-5}	3.2854×10^{-5}	2.06358×10^{-14}	4.59257×10^{-14}	6	−6	0	2
56	1.34978×10^{-31}	0	1.3498×10^{-31}	0	3.37151×10^{-20}	4.52321×10^{-7}	5.97401×10^{-7}	4.52321×10^{-7}	2.05588×10^{-14}	4.59686×10^{-14}	10	−10	0	2
57	−1.910507547	0	−0.6411	0	−1.91022	193.9339097	−384.9026	193.9339097	−1.9105	0	4	−1.5	−1.9133	2
58	0	0	0.3054	0.523729969	0.01294	7.11711×10^{88}	-5.2568×10^{104}	7.11711×10^{88}	0	0	2	0	0	3

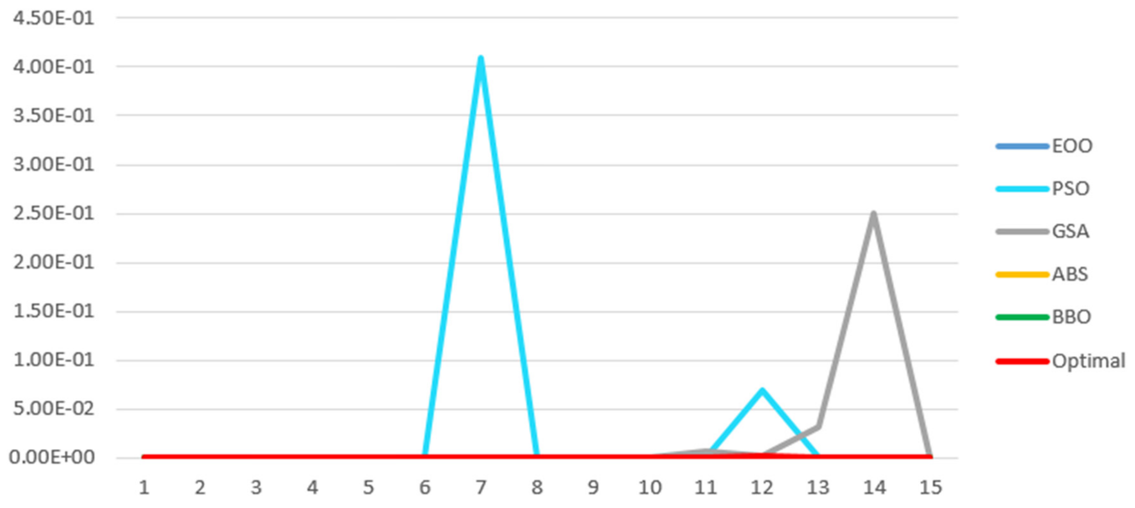

Figure 3: A comparison of the optimal value of the unimodal function test with results of the proposed method EOO, PSO, GWO, BBO, GSA, and ABC algorithms result.

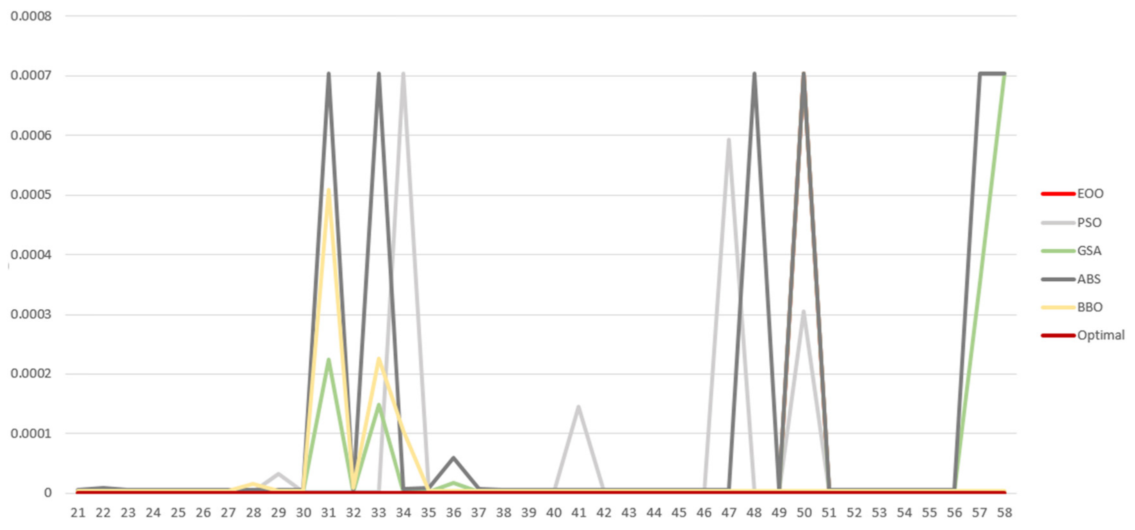

Figure 4: A comparison of the optimal value of the multimodal function test with results of the proposed EOO, PSO, GWO, BBO, GSA, and ABC algorithms result.

It is worth to note that the proposed algorithm EOO has achieved the best results compared to the other meta-heuristics algorithms that distinguished in the unimodal functions and achieved good competition results with the algorithms that distinguished in the multimodal functions. In other words, this common distinction demonstrates that the proposed EOO algorithm met both exploitation and exploration criteria, indicating that it is a competitive algorithm.

5 Conclusion

This study proposed a new nature-inspired meta-heuristic optimisation algorithm called EOO inspired from the behaviour of EO. This algorithm has a high exploratory and exploitative search technique because the randomisation used in the equations enhance the worst solutions by relying on the best solutions. Fifty-eight benchmark test functions were used to evaluate the performance of this algorithm. EOO achieved the optimal value in 31 benchmark test functions and 49 functions. Comparison with other meta-heuristic algorithms, such as PSO, GWO, BBO, GSA, and ABC, verified that EOO is a competitive meta-heuristic optimisation algorithm. Unimodal and multimodal results confirmed that EOO is superior in terms of exploitation and exploration. Future studies should focus on using EOO to solve some complex problems such as travel salesman and timetabling.

Conflict of interest: The authors declare no conflict of interest.

References

[1] Hussain K, Salleh MNM, Cheng S, Shi Y. Metaheuristic research: a comprehensive survey. Artif Intell Rev. 2019;52(4):2191–233.
[2] Beheshti Z, Shamsuddin SMH. A review of population–based meta-heuristic algorithms. Int J Adv Soft Comput Appl. 2013;5(1):1–35.
[3] Agrawal P, Abutarboush HF, Ganesh T, Mohamed AW. Metaheuristic algorithms on feature selection: a survey of one decade of research (2009–2019). IEEE Access. 2021;9:26766–91.
[4] Kennedy J, Eberhart R. Particle swarm optimization. Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 4; 1995. p. 1942–8.
[5] Basturk B. An artificial bee colony (ABC) algorithm for numeric function optimization; 2006.
[6] Passino KM. Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag. 2002;22(3):52–67.
[7] Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Comput Intell Mag. 2006;1(4):28–39.
[8] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61.
[9] Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95:51–67.
[10] Yousif M, Al-Khateeb B. A novel metaheuristic algorithm for multiple traveling salesman problem. Adv Res Dyn Control Syst. 2018;10(13):2113–22.
[11] Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH. Marine predators algorithm: a nature-inspired Metaheuristic. Expert systems with applications; 2020. p. 113377.
[12] Mirjalili S. The ant lion optimizer. Adv Eng Softw. 2015;83:80–98.
[13] Fathollahi-Fard AM, Hajiaghaei-Keshteli M, Tavakkoli-Moghaddam R. Red deer algorithm (RDA): a new nature-inspired meta-heuristic. Soft Comput. 2020;24(19):14637–65.
[14] Dehghani M, Mardaneh M, Guerrero JM, Malik OP, Ramirez-Mendoza RA, Matas J, et al. A new 'doctor and patient' optimization algorithm: an application to energy commitment problem. Appl Sci. 2020;10(17):5791.
[15] Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science. 1983;220(4598):671–80.
[16] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm. Inf Sci. 2009;179(13):2232–48.
[17] Hatamlou A. Black hole: A new heuristic optimization approach for data clustering. Inf Sci. 2013;222:175–84.
[18] Kaveh A, Khayatazad M. A new meta-heuristic method: ray optimization. Computers Struct. 2012;112:283–94.
[19] Holland JH. Genetic algorithms. Sci Am. 1992;267(1):66–73.
[20] Simon D. Biogeography-based optimization. IEEE Trans Evolut Comput. 2008;12(6):702–13.
[21] Koza JR, Koza JR. Genetic programming: on the programming of computers by means of natural selection. Vol. 1. MIT press; 1992.
[22] Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Trans Evolut Comput. 1999;3(2):82–102.
[23] Meire PM, Ervynck A. Are oystercatchers (*Haematopus ostralegus*) selecting the most profitable mussels (*Mytilus edulis*)? Anim Behav. 1986;34(5):1427–35.
[24] Craeymeersch JA, Herman PMJ, Meire PM. Secondary production of an intertidal mussel (Mytilus edulis L.) population in the Eastern Scheldt (SW Netherlands). Hydrobiologia. 1986;133(2):107–15.