MDPI

Article

Implementing a series of modifications for the Eel and grouper optimizer for global optimization problems

Glykeria Kyrou[1], Vasileios Charilogis[2] and Ioannis G. Tsoulos[3,*]

[1] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece; g.kyrou@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece; v.charilog@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, 47150 Kostaki Artas, Greece;itsoulos@uoi.gr
* Correspondence: itsoulos@uoi.gr

Abstract: Global optimization is used in many practical and scientific problems. For this reason, various computational techniques have been developed. Particularly important are the evolutionary techniques, which simulate natural phenomena with the aim of detecting the global minimum in complex problems. A new evolutionary method is the Eel and Grouper Optimization (EGO) algorithm, inspired by the symbiotic relationship and foraging strategy of eels and groupers in marine ecosystems. In this work, modifications of this technique are proposed, such as the use of a termination technique based on stochastic observations. The proposed modifications have been tested on multidimensional functions available from the relevant literature and compared with other evolutionary methods.

Keywords: Global optimization; Metaheuristic algorithms; Stochastic methods; Evolutionary algorithms; Swarm algorithms; Termination strategies; Sampling techniques.

1. Introduction

The basic goal of global optimization is to find the global minimum by searching for the appropriate scope of the underlying objective problem. Primarily, a global optimization method aims to discover the global minimum of a continuous multidimensional function, and it is defined as

$$x^* = \arg\min_{x \in S} f(x) \tag{1}$$

with S:

$$S = [a_1, b_1] \times [a_2, b_2] \times \ldots [a_n, b_n]$$

In recent years many researchers have published important reviews on global optimization[1–3]. Global optimization is a technique of vital importance in many fields of science and applications, as it allows finding the optimal solution to problems with multiple local solutions. In mathematics [4–7], it is used to solve complex mathematical problems, in physics [8–10], it is used to analyze and improve models that describe natural phenomena, in chemistry [11–13], it analyzes and designs molecules and chemical diagnostic tools, and in medicine[16] it analyzes and designs therapeutic strategies and diagnostic tools .

Optimization methods are divided into two categories, deterministic [17–19] and stochastic [20–22]. In the first category, there are techniques aimed at identifying the total minimum with some certainty, such as interval methods [23,24] and are usually distinguished by their complex implementation. The vast majority of global optimization algorithms belong to stochastic methods that have simpler implementation and can also be applied to large -scale problems. Recently, Sergeyev et al [25] published a systematic comparison between deterministic and stochastic methods for global optimization problems.

Citation: Kyrou, G.; Charilogis, V.; Tsoulos, I.G. Implementing a series of modifications for the Eel and grouper optimizer for global optimization problems. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

An important branch of the stochastic methods are the evolutionary methods, that attempt to mimic a series of natural processes. Among these methods one can find the Differential Evolution method [26,27], Particle Swarm Optimization (PSO) methods [28–30], Ant Colony optimization methods [31,32], Genetic algorithms [33,34] etc. Additionally, since in recent years there has been an extremely wide spread of parallel computing units, many researches have proposed evolutionary methods that exploit modern parallel processing units [35–37].

Among the evolutionary techniques one finds a large group of methods that have been explored intensively in recent years, the so - called Swarm Intelligence algorithms. Swarm intelligence algorithms [38–40] are inspired by the collective behavior of insects and other animals. These algorithms mimic systems in which agents interact locally and cooperate worldwide to solve optimization problems. Swarm intelligence algorithms are very important tools for dealing with complex optimization problems in a wide range of applications [41]. Some examples of these methods are the Fast Bacterial Swarming Algorithm (FBSA) [42], the Fish Swarm Algorithm [43], the Dolphin Swarm Algorithm [44], the Whale Optimization Algorithm (WOA) algorithm 45–47, the Tunicate Swarm Algorithm [48], the Sine Cosine algorithm (SCA) algorithm [49–52], the Salp Swarm algorithm (SSA) algorithm [53–56] etc. These methods simulate a series of complex interactions between biological species [57,58], such as:

1. Naturalism: Where two species can live in an ecosystem without affecting each other.
2. Predation, where one creature dies by feeding another.
3. Parasitism: where one species causes harm to another without always killing it.
4. In competitive mode, the same or different organizations compete for resources.
5. Mutualism [59–61]: when two organisms have a beneficial interaction.

Among swarm intelligence algorithms one can find the Eel and Grouper (EGO) algorithm, which is inspired by the symbiotic interaction and foraging strategy of eels and groupers in marine ecosystems. Bshary et al. [62] consider that target ingestion, something observed in eels and groupers, is a necessary condition for interspecific cooperative hunting to occur. Intraspecific predation could increase the hunting efficiency of predators by mammals. According to Ali Mohammadzadeh and Seyedali Mirjalili the EGO optimization algorithm [63] generates a set of random answers, then stores the best answers found so far, allocates them to the target point, and changes the answers with them. As the number of iterations increases, the limits of the sine function are changed to enhance the phase of finding the best solution. This method stops the process when the iteration exceeds the maximum number. Because the EGO optimization algorithm generates and boosts a collection of random responses, it has the advantage of increased local optimum discovery and avoidance compared to individual methods.

This paper introduces some modifications to the EGO algorithm in order to improve its efficiency. The proposed amendments are presented below:

- The addition of a sampling technique based on the K-means method [67]. The sampling points will facilitate finding the global minimum of the function in the most efficient way. Additionally, by applying this method, nearby points are discarded.
- Using a termination technique that is developed with random measurements. Each time the algorithm is repeated, the minimum value is recorded. When this remains constant for a predetermined number of iterations, the process is terminated. Therefore, the method will complete in time without unnecessarily wasting time in iterations, avoiding unnecessary consumption of computing resources.
- Application of randomness in the definition of the range of the positions of candidate solutions.

The rest of this paper is divided into the following sections: in section 2, the proposed method is fully described, in section 3 the experimental results and statistical comparisons are outlined and finally in section 4 some conclusions and guidelines for future improvements are discussed.

2. The proposed method

The main steps of the base EGO algorithm and the proposed modifications are described in detail in this section.

2.1. The main steps of the algorithm

EGO optimization algorithm starts by initializing a population consisting of "search agents" that search to find the optimal solution. At each iteration, the position of the "prey" (optimal solution) is calculated. Agent positions are adjusted based on random variables and their distance based on the optimal position.At the end of each iteration, the current solutions are compared and it is decided whether the algorithm should continue or terminate. The main steps of the used global optimization method provided in Algorithm 1. Also, the algorithm is presented as a series of steps in the flowchart of Figure 1.

Algorithm 1 EGO Algorithm

Initialization step.

1. **Define** as N_c the number of elements in the search Agents
   (a) **Define** as N_g, the maximum number of allowed iterations.
   (b) **Initialize** randomly the search agents $x_i,\ i = 1, \ldots, N_c$ in set S.
   (c) **Set** $t = 0$, the iteration counter.
   (d) **Set** $s_r = 0$, the starvation rate of the algorithm.
   (e) **Set** $m = 1$, this parameter influences how the variables f_1, f_2 are defined, which in turn affects the calculation of new positions. When $m = 1$, it introduces randomness to the range of positions before the update, while in the inactive state, the range remains fixed.
2. **Calculation step**.
   (a) **Update** variables a and s_r:
   - $a = 2 - 2\frac{t}{N_g}$
   - $s_r = 100\frac{t}{N_g}$

   (b) **Compute** the fitness of each search agents.
   (c) **Sort** all solutions in the current population from the best to worst according to the function value.
   (d) **Set** XP the estimated position of the prey.
   (e) **Update** random variables $r_1, r_2, r_3, r_4, C_1, C_2, b$:
   - $r_1$and r_2 are random numbers in $[0, 1]$.
   - $r_3 = (a - 2)r_1 + 2$
   - $r_4 = 100r_2$
   - $C_1 = 2 * a * r_1 - a$
   - $C_2 = 2 * r_1$
   - $b = a * r_2$
   - $X_1 = e^{br_3} \sin(2\pi r_3) C_1 |\mathrm{XE} - \mathrm{XP}| + \mathrm{XE}$, where $\mathrm{XE} = C_2 x_j$
   - $X_2 = x_j + C_1 \left| x_j - \mathrm{XP} \right|$

   (f) **if** $(r4 \leq s_r)$
   - **if** $m = 1$ then set $f_1 = 0.8, f_2 = 0.2$
   - **else set** f_1 to a random number in $[0, 2]$ and f_2 to a random number in $[-2, 2]$

   (g) **Endif**
   (h) **Update** agent: $x_j = \frac{f_1 X_1 + f_2 X_2}{2}$
3. **End For**
4. **Termination check step**
   (a) **Set** $t = t + 1$
   (b) **If** $t \geq N_g$ **terminate**.
   (c) **Calculate** the stopping that proposed in the work of Charilogis [66]. In the Similarity stopping rule, at every iteration t, the absolute difference between the current located global minimum $f_{min}^{(t)}$ and the previous best value $f_{min}^{(t-1)}$ is calculated:

$$\left| f_{\min}^{(t)} - f_{\min}^{(k-1)} \right| \tag{2}$$

If the termination criteria are not met then go to Calculation step, **else** terminate and return the best solution.

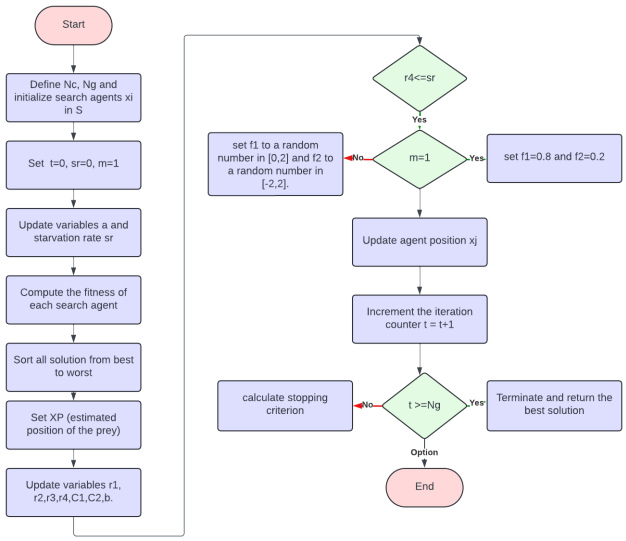

Figure 1. Flowchart of the suggested global optimization procedure.

2.2. The proposed sampling procedure

The sampling technique applied in this work initially generates samples from the objective function. Then, through the K-means method, only the recognized centers are selected as final samples. This technique, which is an achievement of James MacQueen [67], is one of the most well-known clustering algorithms in the broad research community, both in data analysis and in machine learning [68] and pattern recognition [69]. The algorithm aims to divide a data set into k clusters. The K-means algorithm tries to divide the data into groups in such a way that the internal points of each group are as close as possible to each other. At the same time, he tries to place the central points of each group in positions which are as representative as possible for the points of their group. During the past years a series of variants of this algorithm has been proposed, such as the Genetic K-means algotithm [70], the unsupervised K-means algorithm [71], the Fixed-centered K-means algorithm [72] etc. A review of K-Means clustering algorithms can be found in the work of Oti et al. [73] Next, the basic steps of the algorithm are presented in Algorithm 2. A flowchart of the K-Means procedure is also depicted in Figure 2.

Algorithm 2 K-means Algorithm

1. **Initialization**
   (a) **Set** k the number of clusters.
   (b) **Set** randomly initial samples to $x_i,\ i = 1, \ldots, N_m$
   (c) **Set** $S =_j \{\ \}$, from $j = 1, \ldots, k$.
2. Repeat
   (a) **For** each point $x_i,\ i = 1, ..., N_m$ **do**
      i. **Set** $j^* = \text{argmin}_{m=1}^{k}\{D(x_i, c_m)\}$. j^* is the nearest center from x_i
      ii. **Set** $S_{j^*} = S_{j^*} \cup \{x_i\}$.
      iii. **End For**
   (b) **For** each center $c_j,\ j = 1..k$ **do**
      i. **Set** M_j the number of samples in S_j
      ii. **Update** the center c_j as

$$c_j = \frac{1}{M_j} \sum_{x_i \in S_j} x_i$$

   (c) **End** For
3. Terminate when c_j no longer changes

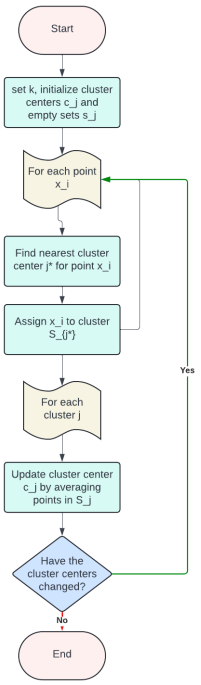

Figure 2. The flowchart of the K-Means procedure.

3. Results

This section will begin with a detailed description of the functions that will be used in the experiments, followed by an analysis of the experiments performed and comparisons with other global optimization techniques.

3.1. Test functions

The functions used in the experiments have been proposed in a series of relative works [74,75] and they cover various scientific fields, such as medicine, physics, engineering,

etc. Also, these objective functions have been used by many researchers in a variety of publications [76–80].

The definitions of these functions are given below:

- **Ackley** function:
- **Bf1** (Bohachevsky 1) function:

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$$

- **Bf2** (Bohachevsky 2) function:

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$$

- **Bf3** (Bohachevsky 3) function:

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1 + 4\pi x_2) + \frac{3}{10}$$

- **Branin** function:

$$f(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$$

with $-5 \leq x_1 \leq 10,\ 0 \leq x_2 \leq 15$.

- **Camel** function:

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, \quad x \in [-5,5]^2$$

- **Easom** function:

$$f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2-\pi)^2 - (x_1-\pi)^2\right)$$

with $x \in [-100,100]^2$.

- **Exponential** function, defined as:

$$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \leq x_i \leq 1$$

In the conducted experiments the values $n = 4, 8, 16, 32$ were used.

- **Extended F10** function:

$$f(x) = \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i)\right)$$

- **F14** function:

$$f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$$

- **F15** function:

$$f(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$$

- **F17** function:

$$f(x) = \left(1 + (x_1 + x_2 + 1)^2 \times \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right) \times \left(30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right)$$

- **Griewank2** function:

$$f(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2} \frac{\cos(x_i)}{\sqrt{(i)}}, \quad x \in [-100, 100]^2$$

- **Griewank10** function. The function is given by the equation

$$f(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

with $n = 10$.

- **Gkls** function [81]. The function $f(x) = \text{Gkls}(x, n, w)$, is a constructed function with w local minima presented in [81], with $x \in [-1, 1]^n$. For the conducted experiments the values $n = 2, 3$ and $w = 50$ were utilized.
- **Goldstein and Price function**

$$\begin{aligned} f(x) \quad = \quad & \Big[1 + (x_1 + x_2 + 1)^2 \\ & \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\Big] \times \\ & [30 + (2x_1 - 3x_2)^2 \\ & \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)] \end{aligned}$$

With $x \in [-2, 2]^2$.

- **Hansen** function: $f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i] \sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$, $x \in [-10, 10]^2$.
- **Hartman 3** function:

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0, 1]^3$ and $a = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}$$

- **Hartman 6** function:

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0,1]^6$ and $a = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}$$

- **Potential** function, this function stands for the energy of a molecular conformation of N atoms, that interacts using via the Lennard-Jones potential [82]. The function is defined as:
$$V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$
For the conducted experiments the values $N = 3,\ 5$ were used.
- **Rastrigin** function.
$$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2), \quad x \in [-1,1]^2$$
- ***Rosenbrock*** *function*.
$$f(x) = \sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \leq x_i \leq 30.$$
The values $n = 4,\ 8,\ 16$ were used in the conducted experiments.
- **Shekel 5** function.
$$f(x) = -\sum_{i=1}^{5} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$
with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{pmatrix}$
- **Shekel 7** function.
$$f(x) = -\sum_{i=1}^{7} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$
with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{pmatrix}$.
- **Shekel 10** function.
$$f(x) = -\sum_{i=1}^{10} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}$, $c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.6 \end{pmatrix}$.

- **Sinusoidal** function defined as:

$$f(x) = -\left(2.5\prod_{i=1}^{n}\sin(x_i - z) + \prod_{i=1}^{n}\sin(5(x_i - z))\right), \quad 0 \leq x_i \leq \pi.$$

  The values of $n = 4, 8, 16$ were used in the conducted experiments.
- **Schaffer** function:

$$f(x) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$$

- **Schwefel221** function:

$$f(x) = 418.9829n + \sum_{i=1}^{n} -x_i \sin\left(\sqrt{|x_i|}\right)$$

- **Schwefel222** function:

$$f(x) = \sum_{i=1}^{n}|x_i| + \prod_{i=1}^{n}|x_i|$$

- **Sphere** function:

$$f(x) = \sum_{i=1}^{n} x_i^2$$

- **Test2N** function:

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i, \quad x_i \in [-5,5].$$

  For the conducted experiments the values $n = 4, 5, 6, 7$ were used.
- **Test30N** function:

$$f(x) = \frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i - 1)^2\left(1 + \sin^2(3\pi x_{i+1})\right)\right) + (x_n - 1)^2\left(1 + \sin^2(2\pi x_n)\right)$$

  The values $n = 3, 4$ were used in the conducted experiments.

3.2. Experimental results

A series of global optimization methods were applied to the test functions presented previously. All experiments were conducted 30 times using different seed for the random generator and averages were recorded. The used software was coded in ANSI C++ using the freely available OPTIMUS optimization environment, that is available from https://github.com/itsoulos/OPTIMUS (accessed on 26 August 2024). The experiments were executed on a Debian Linux system that runs on an AMD Ryzen 5950X processor, with 128GB of RAM. In all cases the BFGS [83] local optimization method was used at the end of

each global optimization technique to ensure that an actual minimum will be discovered by the global optimization method. The values for all experimental parameters are shown in Table 1.

Table 1. Experimental settings. The numbers in cells denote the values used in the experiments for all parameters.

PARAMETER	MEANING	VALUE
N_c	Number of chromosomes/particles	200
N_g	Maximum number of allowed iterations	200
N_m	Number of initial samples for K-means	$10 \times N_c$
N_k	Number of iterations for stopping rule	5
p_s	Selection rate for the genetic algorithm	0.1
p_m	Mutation rate for the genetic algorithm	0.05

The experimental results for the test functions and a series of optimization methods are shown in Table 2, where the following applies to this table:

- The column FUNCTION denotes the name of the objective problem.
- The column GENETIC denotes the application of a genetic algorithm [33,34] to the objective problem. The genetic algorithm has N_c chromosomes and the maximum number of allowed generations was set to N_g. For the conducted experiments a modified version of the genetic algorithm for global optimization problems that was suggested by Tsoulos [84] was used.
- The column PSO stands for the application of Particle Swarm Optimizer [29,30] to every objective problem. The number of particles was set to N_c and the maximum number of allowed iterations was set to N_g. For the conducted experiments, the improved PSO method as proposed by Charilogis and Tsoulos was used [85].
- The column DE refers to the Differential Evolution method [26,27].
- The column EEGO represents the application of the proposed method using the values for the parameters shown in Table 1.
- The row SUM represents the sum of function calls for all test functions.

Table 2. Experimental results using different optimization methods. Numbers in cells represent sum function calls.

FUNCTION	GENETIC	PSO	DE	EEGO
ACKLEY	6749	6885	10220	4199
BF1	4007	4142	8268	3228
BF2	3794	3752	7913	2815
BF3	3480	3306	10270	2501
BRANIN	2376	2548	4101	1684
CAMEL	2869	2933	5609	2262
EASOM	1958	1982	2978	1334
EXP4	2946	3404	5166	2166
EXP8	3120	3585	5895	2802
EXP16	3250	3735	6498	3279
EXP32	3561	3902	7606	3430
EXTENDEDF10	4862	3653	5728(0.87)	2609
F14	6686	5498	5279(0.63)	6063
F15	4373	6696	5874(0.80)	4397
F17	3667	3805	10441	2766
GKLS250	2280	2411	3834	1603
GKLS350	2613	2234	3919	1298
GOLDSTEIN	3687	3865	6781	2784
GRIEWANK2	4501	3076 (0.73)	7429	2589 (0.96)
GRIEWANK10	6410 (0.97)	8006	18490	7435
HANSEN	3210	2856	4185	2484
HARTMAN3	2752	3140	5190	1793
HARTMAN6	3219	3710	5968	2478
POTENTIAL3	4352	4865	6118	4081
POTENTIAL5	7705	9183	9119	8886
RASTRIGIN	4107	3477	6216	2304
ROSENBROCK4	3679	6372	8452	4019
ROSENBROCK8	5270	8284	11530	6801
ROSENBROCK16	8509	11872	17432	11996
SHEKEL5	3325	4259	6662	2495
SHEKEL7	3360	4241	6967	2432
SHEKEL10	3488	4237	6757	2516
SCHAFFER	18787	15176	6315	23531
SCHWEFEL221	2667	2529	5415	2203
SCHWEFEL222	33725	42898	12200	38876
SPHERE	1588	1521	3503	1162
TEST2N4	3331	3437	6396	2277
TEST2N5	4000	3683	6271	2734 (0.96)
TEST2N6	4312 (0.93)	3781	5410 (0.93)	2905 (0.86)
TEST2N7	4775 (0.90)	4060	7074 (0.97)	3559 (0.73)
SINU4	2991	3504	5953	2005
SINU8	3442	4213	6973	3158
SINU16	4320	5019	6979	5891
TEST30N3	3211	4610	6168	2362
TEST30N4	3679	4629	7006	2978
SUM	**220993**	**244974**	**322558**	**208170**

In figure 3 we present the total function calls of every optimization method is presented graphically. The proposed method has excellent results compared to the other optimization techniques according to the experiment we conducted. As we can observe it has the least number of calls than all the other techniques.

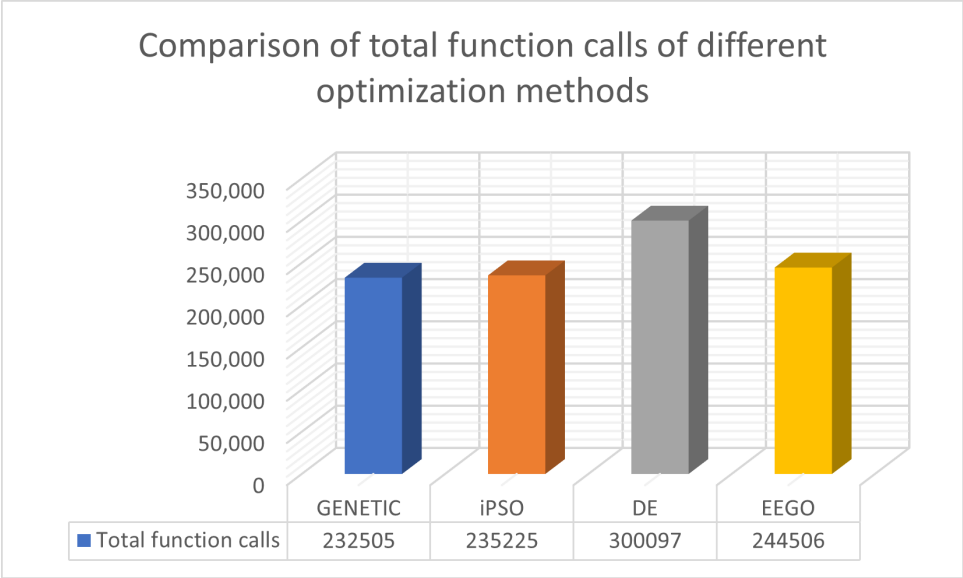

Figure 3. Total function calls for the different optimization methods, using proposed initial distribution

Also, a statistical comparison between the optimization methods is shown graphically in Figure 4.

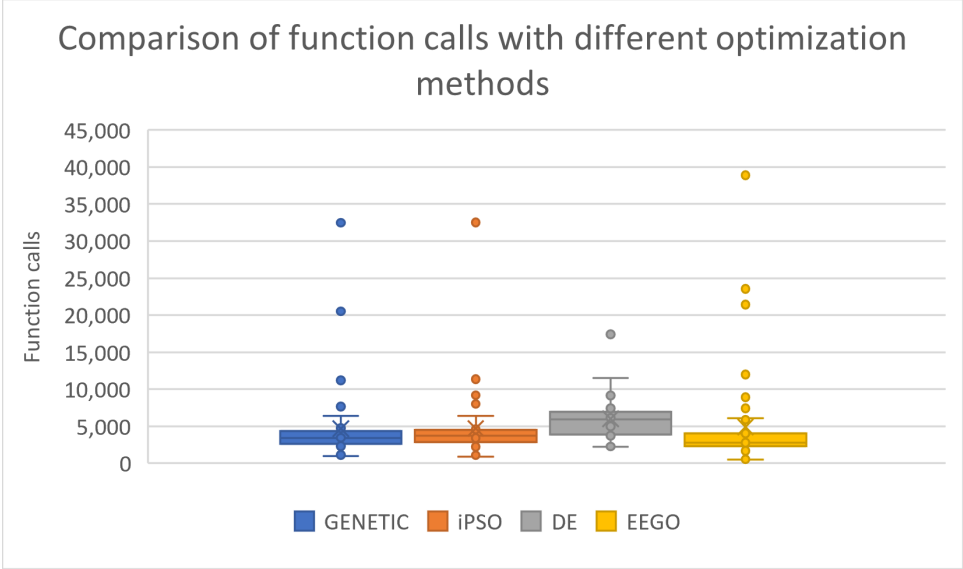

Figure 4. Comparison of function calls for the different optimization methods, using proposed initial distribution

One more experiment which was performed with the ultimate goal of measuring the importance of K-means sampling in the proposed method. The results for this experiment are outlined in Table 3 and the following sampling methods were used:

1. The column UNIFORM represents the application of uniform sampling in the current method.
2. The column TRIANGULAR stands for the usage of the triangular distribution [86] for sampling.
3. The column MAXWELL stands for the application of the Maxwell distribution [87] to produce initial samples for the used method.
4. The column KMEANS represents the usage of the method described in subsection 2.2 to produce initial samples for the used method.

Table 3. Experiments using different sampling techniques for the proposed method.

FUNCTION	UNIFORM	TRIANGULAR	MAXWELL	KMEANS
ACKLEY	6118	5912	5986	4199
BF1	4513	4318	4055	3228
BF2	3959	3879	3587	2815
BF3	3506	3344	3129	2501
BRANIN	2282	2131	2066	1684
CAMEL	3156	2919	2848	2262
EASOM	1756	1650	1321	1334
EXP4	3438	3273	3194	2166
EXP8	3432	3387	3152	2802
EXP16	3369	3326	3291	3279
EXP32	3216	3225	3344	3430
EXTENDEDF10	4304	3913	3718	2609
F14	7552	7573	6148	6063
F15	6806	6466	6543	4397
F17	3805	3700	3543	2766
GKLS250	2268	2023	1778	1603
GKLS350	2151	1841	2069	1298
GOLDSTEIN	3855	3731	3530	2784
GRIEWANK2	4310	4510	4035	2589
GRIEWANK10	8640	8773	8232	7435
HANSEN	3329	3071	2734	2484
HARTMAN3	2849	2673	2678	1793
HARTMAN6	3456	3249	3119	2478
POTENTIAL3	4554	5095	3928	4081
POTENTIAL5	8356	10032	7504	8886
RASTRIGIN	3310	3187	2751	2304
ROSENBROCK4	6566	6353	5588	4019
ROSENBROCK8	8379	8717	7783	6801
ROSENBROCK16	11921	12471	11677	11996
SHEKEL5	3946	3731	3859	2495
SHEKEL7	3990	3646	3944	2432
SHEKEL10	3836	3630	3694	2516
SCHAFFER	55131	40715	60717	23531
SCHWEFEL221	2724	2901	2799	2203
SCHWEFEL222	53118	54593	55354	38876
SPHERE	1346	1188	1084	1162
TEST2N4	3345	3233	2867	2277
TEST2N5	3937	3742	3094	2734
TEST2N6	4008	4473	3266	2905
TEST2N7	4545	4612	3549	3559
SINU4	3128	2879	3559	2005
SINU8	4126	3767	5637	3158
SINU16	6774	5977	7739	5891
TEST30N3	3704	3384	3175	2362
TEST30N4	4262	4327	3491	2978
SUM	**297076**	**281540**	**291159**	**208170**

Initial distributions play a critical role in a wide range of applications, including optimization, statistical analysis, and machine learning. Table 3 presents the proposed distribution alongside other established distributions. The uniform distribution is widely used due to its ability to evenly cover the search space, making it suitable for initializing optimization algorithms [88]. The triangular distribution is applied in scenarios where there

is knowledge of the bounds and the most probable value of a phenomenon, making it useful in risk management models [89]. The Maxwell distribution, although originating from physics, finds applications in simulating communication networks, where data transfer speeds can be modeled as random variables [90]. Finally, the k-means method is used for data clustering, with k-means++ initialization offering improved performance compared to random distributions, particularly in high-dimensional problems [91]. As observed in Table 3, the choice of an appropriate initial distribution can significantly affect the performance of the algorithms that utilize them.

In the scatter plot in figure 5, the critical parameter "p" was found to be very small, leading to the rejection of the null hypothesis and indicating that the experimental results are highly significant.Scatter plot for different initial distributions.

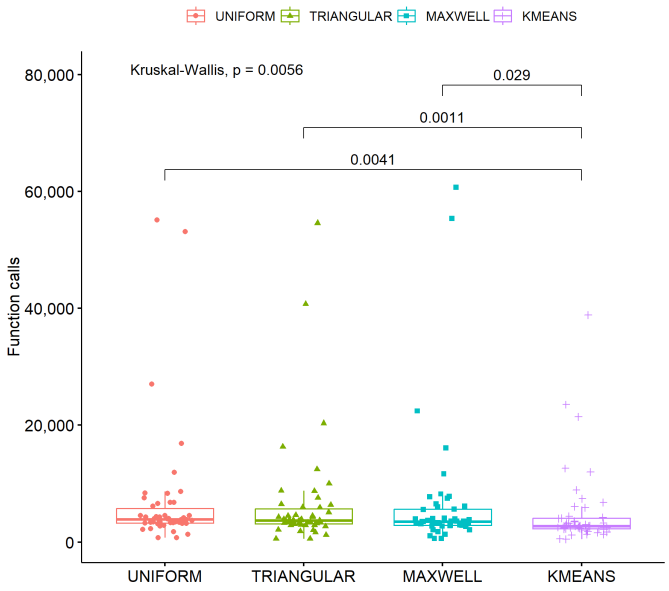

Figure 5. Scatter plot for different initial distributions

4. Conclusions

The article proposed some modifications to the EEGO optimization method, aimed at improving the overall performance and increasing the speed of the global optimization algorithm. The first modification concerns the application of a sampling technique based on the K-Means method. This technique allowed us to significantly minimize the number of function calls to find the global minimum and further improved the accuracy with which it is located. In particular, the application of the K-Means method accelerated the finding of a solution, as it more efficiently located the points of interest in the search space and led to the fastest convergence to the global minimum. Compared to other methods based on random distributions, the proposed technique proved its superiority, especially in multidimensional and complex functions.

The second proposed amendment concerns the termination rule based on similarity of solutions during iterations. The main purpose of this rule is to stop the optimization process when the iterated solutions are too close to each other, thus preventing pointless iterations that do not provide any significant improvement. It therefore avoids wasting computing time in cases where the process is already very close to the desired result. The use of the termination rule significantly improves the efficiency of the algorithm.

In addition, other modifications were proposed in the article, which help to reduce the number of function calls and increase the success rate of the algorithm. These modifications

enhanced the efficiency of the EEGO algorithm, making it more efficient in a wide range of optimization problems.

The experimental results presented in the article were particularly encouraging. Therefore, the proposed method can be applied in various fields. The increased speed and improved accuracy of the optimization process make it suitable for a wide range of applications in fields such as machine learning and artificial intelligence.

A future direction suggested for further improvement of the method is the utilization of parallel computing systems to speed up the optimization process. Examples of such systems are the use of MPI technology[92] or the integration of the OpenMP library[93], which allows multithreaded execution of the algorithm on modern multicore systems. This perspective can lead to a significant speed improvement of the EEGO method, making it even more effective and efficient for applications that require high computing power.

Author Contributions: G.K., V.C. and I.G.T. conceived of the idea and the methodology, and G.K. and V.C. implemented the corresponding software. G.K. conducted the experiments, employing objective functions as test cases, and provided the comparative experiments. I.G.T. performed the necessary statistical tests. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: This research has been financed by the European Union: Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan, under the call RESEARCH–CREATE–INNOVATE, project name “iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques” (project code: TAEDK-06195).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. A. Törn, M.M. Ali, S. Viitanen, Stochastic global optimization: Problem classes and solution techniques. Journal of Global Optimization **14**, pp. 437-447, 1999.
2. Floudas, C. A., & Pardalos, P. M. (Eds.). (2013). State of the art in global optimization: computational methods and applications
3. Horst, R., & Pardalos, P. M. (Eds.). (2013). Handbook of global optimization (Vol. 2). Springer Science & Business Media.
4. Intriligator, M. D. (2002). Mathematical optimization and economic theory. Society for Industrial and Applied Mathematics.
5. Cánovas, M. J., Kruger, A., Phu, H. X., & Théra, M. (2020). Marco A. López, a Pioneer of Continuous Optimization in Spain. Vietnam Journal of Mathematics, 48, 211-219.
6. Mahmoodabadi, M. J., & Nemati, A. R. (2016). A novel adaptive genetic algorithm for global optimization of mathematical test functions and real-world problems. Engineering Science and Technology, an International Journal, 19(4), 2002-2021.
7. Li, J., Xiao, X., Boukouvala, F., Floudas, C. A., Zhao, B., Du, G., ... & Liu, H. (2016). Data-driven mathematical modeling and global optimization framework for entire petrochemical planning operations. AIChE Journal, 62(9), 3020-3040.
8. E. Iuliano, Global optimization of benchmark aerodynamic cases using physics-based surrogate models, Aerospace Science and Technology **67**, pp.273-286, 2017.
9. Q. Duan, S. Sorooshian, V. Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resources Research **28**, pp. 1015-1031 , 1992.
10. L. Yang, D. Robin, F. Sannibale, C. Steier, W. Wan, Global optimization of an accelerator lattice using multiobjective genetic algorithms, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **609**, pp. 50-57, 2009.
11. S. Heiles, R. L. Johnston, Global optimization of clusters using electronic structure methods, Int. J. Quantum Chem. **113**, pp. 2091–2109, 2013.
12. W.H. Shin, J.K. Kim, D.S. Kim, C. Seok, GalaxyDock2: Protein–ligand docking using beta-complex and global optimization, J. Comput. Chem. **34**, pp. 2647– 2656, 2013.
13. A. Liwo, J. Lee, D.R. Ripoll, J. Pillardy, H. A. Scheraga, Protein structure prediction by global optimization of a potential energy function, Biophysics **96**, pp. 5482-5485, 1999.
14. Eva K. Lee, Large-Scale Optimization-Based Classification Models in Medicine and Biology, Annals of Biomedical Engineering **35**, pp 1095-1109, 2007.

15. Y. Cherruault, Global optimization in biology and medicine, Mathematical and Computer Modelling **20**, pp. 119-132, 1994.
16. Houssein, E. H., Hosney, M. E., Mohamed, W. M., Ali, A. A., & Younis, E. M. (2023). Fuzzy-based hunger games search algorithm for global optimization and feature selection using medical data. Neural Computing and Applications, 35(7), 5251-5275.
17. Ion, I. G., Bontinck, Z., Loukrezis, D., Römer, U., Lass, O., Ulbrich, S., ... & De Gersem, H. (2018). Robust shape optimization of electric devices based on deterministic optimization methods and finite-element analysis with affine parametrization and design elements. Electrical Engineering, 100(4), 2635-2647.
18. Cuevas-Velásquez, V., Sordo-Ward, A., García-Palacios, J. H., Bianucci, P., & Garrote, L. (2020). Probabilistic model for real-time flood operation of a dam based on a deterministic optimization model. Water, 12(11), 3206.
19. Pereyra, M., Schniter, P., Chouzenoux, E., Pesquet, J. C., Tourneret, J. Y., Hero, A. O., & McLaughlin, S. (2015). A survey of stochastic simulation and optimization methods in signal processing. IEEE Journal of Selected Topics in Signal Processing, 10(2), 224-241.
20. Hannah, L. A. (2015). Stochastic optimization. International Encyclopedia of the Social & Behavioral Sciences, 2, 473-481.
21. Kizielewicz, B., & Sałabun, W. (2020). A new approach to identifying a multi-criteria decision model based on stochastic optimization techniques. Symmetry, 12(9), 1551.
22. Chen, T., Sun, Y., & Yin, W. (2021). Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization. IEEE Transactions on Signal Processing, 69, 4937-4948.
23. M.A. Wolfe, Interval methods for global optimization, Applied Mathematics and Computation **75**, pp. 179-206, 1996.
24. T. Csendes and D. Ratz, Subdivision Direction Selection in Interval Methods for Global Optimization, SIAM J. Numer. Anal. **34**, pp. 922–938, 1997.
25. Y.D. Sergeyev, D.E. Kvasov, M.S. Mukhametzhanov, On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Sci Rep **8**, 453, 2018.
26. R. Storn, K. Price, Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization **11**, pp. 341-359, 1997.
27. J. Liu, J. Lampinen, A Fuzzy Adaptive Differential Evolution Algorithm. Soft Comput **9**, pp.448–462, 2005.
28. J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.
29. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
30. Ioan Cristian Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.
31. M. Dorigo, M. Birattari and T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine **1**, pp. 28-39, 2006.
32. K. Socha, M. Dorigo, Ant colony optimization for continuous domains, European Journal of Operational Research 185, pp. 1155-1173, 2008.
33. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
34. Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.
35. Y. Zhou and Y. Tan, "GPU-based parallel particle swarm optimization," 2009 IEEE Congress on Evolutionary Computation, pp. 1493-1500, 2009.
36. L. Dawson and I. Stewart, "Improving Ant Colony Optimization performance on the GPU using CUDA," 2013 IEEE Congress on Evolutionary Computation, 2013, pp. 1901-1908, doi: 10.1109/CEC.2013.6557791.
37. Barkalov, K., Gergel, V. Parallel global optimization on GPU. J Glob Optim 66, 3–20 (2016).
38. Hassanien, A. E., & Emary, E. (2018). Swarm intelligence: principles, advances, and applications. CRC press.
39. Tang, J., Liu, G., & Pan, Q. (2021). A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. IEEE/CAA Journal of Automatica Sinica, 8(10), 1627-1643.
40. Brezočnik, L., Fister Jr, I., & Podgorelec, V. (2018). Swarm intelligence algorithms for feature selection: a review. Applied Sciences, 8(9), 1521.
41. Tang, J., Liu, G., Pan, Q.: A review on representative swarm intelligence algorithms for solving optimization problems:applications and trends. IEEE/CAA J. Autom. Sin. 8(10),1627–1643 (2021).
42. Ying Chu, Hua Mi, Huilian Liao, Zhen Ji and Q. H. Wu, "A Fast Bacterial Swarming Algorithm for high-dimensional function optimization," 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), Hong Kong, China, 2008, pp. 3135-3140, doi: 10.1109/CEC.2008.4631222.
43. M. Neshat, G. Sepidnam, M. Sargolzaei, et al, Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications, Artif Intell Rev **42**, pp. 965–997, 2014.
44. Tq. Wu, M. Yao, Jh. Yang, Dolphin swarm algorithm, Frontiers Inf Technol Electronic Eng **17**, pp. 717–729, 2016.
45. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Adv. Eng. Softw. 95, 51–67 (2016)
46. Nasiri, J., & Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach for clustering. Cogent Mathematics & Statistics, 5(1), 1483565.
47. Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey: Whale Optimization Algorithm and its applications. Swarm and Evolutionary Computation, 48, 1-24.

48. S. Kaur, L.K. Awasthi, A.L. Sangal, G. Dhiman, Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization **90**, 103541, 2020.
49. Mirjalili, S.: SCA: a sine cosine algorithm for solving optimization problems. Knowl.-Based Syst. 96, 120–133 (2016)
50. Hao, Y., Song, L., Cui, L., & Wang, H. (2019). A three-dimensional geometric features-based SCA algorithm for compound faults diagnosis. Measurement, 134, 480-491.
51. Sahu, P. C., Prusty, R. C., & Panda, S. (2022). Optimal design of a robust FO-Multistage controller for the frequency awareness of an islanded AC microgrid under i-SCA algorithm. International Journal of Ambient Energy, 43(1), 2681-2693.
52. Zivkovic, M., Stoean, C., Chhabra, A., Budimirovic, N., Petrovic, A., & Bacanin, N. (2022). Novel improved salp swarm algorithm: An application for feature selection. Sensors, 22(5), 1711.
53. Wan, Y., Mao, M., Zhou, L., Zhang, Q., Xi, X., & Zheng, C. (2019). A novel nature-inspired maximum power point tracking (MPPT) controller based on SSA-GWO algorithm for partially shaded photovoltaic systems. Electronics, 8(6), 680.
54. Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M.: Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv. Eng. Softw. 114, 163–191 (2017)
55. Bairathi, D., & Gopalani, D. (2019). Salp swarm algorithm (SSA) for training feed-forward neural networks. In Soft Computing for Problem Solving: SocProS 2017, Volume 1 (pp. 521-534). Springer Singapore.
56. Abualigah, L., Shehab, M., Alshinwan, M., & Alabool, H. (2020). Salp swarm algorithm: a comprehensive survey. Neural Computing and Applications, 32(15), 11195-11215.
57. Usman, M.J.: A survey of symbiotic organisms search algorithms and applications. Neural Comput. Appl. 32(2), 547–566 (2020)
58. . Ezugwu, A.E., Adeleke, O.J., Akinyelu, A.A., Viriri, S.: A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems. Neural Comput. Appl. 32(10), 6207–6251 (2020)
59. Wang, Y., & DeAngelis, D. L. (2012). A mutualism-parasitism system modeling host and parasite withmutualism at low density. Mathematical Biosciences & Engineering, 9(2), 431-444.
60. Aubier, T. G., Joron, M., & Sherratt, T. N. (2017). Mimicry among unequally defended prey should be mutualistic when predators sample optimally. The American Naturalist, 189(3), 267-282.
61. Addicott, J. F. (1985). Competition in mutualistic systems. The biology of mutualism: ecology and evolution. Croom Helm, London, UK, 217-247.
62. Bshary, R., Hohner, A., Ait-el-Djoudi, K., Fricke, H.: Interspecific communicative and coordinated hunting between groupers and giant moray eels in the Red Sea. PLoS Biol. 4(12), e431 (2006)
63. Ali Mohammadzadeh, Seyedali Mirjalili. (2024) Eel and grouper optimizer: a nature-inspired optimization algorithm. Springer Science+Business Media, LLC, part of Springer Nature 2024
64. P. Arora, S. Varshney, Analysis of k-means and k-medoids algorithm for big data, Procedia Computer Science **78**, pp. 507-512, 2016.
65. Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. Electronics, 9(8), 1295.
66. Charilogis, V.; Tsoulos, I.G. Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions. Information 2022, 13, 217.
67. J.B. MacQueen, Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. Vol. 1. University of California Press. pp. 281–297. MR 0214227. Zbl 0214.46201, 1967.
68. Y. Li, H. Wu, A clustering method based on K-means algorithm, Physics Procedia **25**, pp. 1104-1109, 2012.
69. Ali, H. H., & Kadhum, L. E. (2017). K-means clustering algorithm applications in data mining and pattern recognition. International Journal of Science and Research (IJSR), 6(8), 1577-1584.
70. K. Krishna, M. Narasimha Murty, Genetic K-means algorithm, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **29**, pp. 433-439, 1999.
71. K. P. Sinaga, M. -S. Yang, Unsupervised K-Means Clustering Algorithm, IEEE Access **8**, pp. 80716-80727, 2020.
72. M. Ay, L. Özbakır, S. Kulluk, B. Gülmez , G.Öztürk, S. Özer, FC-Kmeans: Fixed-centered K-means algorithm, Expert Systems with Applications **211**, 118656, 2023.
73. E.U. Oti, M.O. Olusola, F.C. Eze, S.U. Enogwe, Comprehensive review of K-Means clustering algorithms, Criterion **12**, pp. 22-23, 2021.
74. M. Montaz Ali, Charoenchai Khompatraporn, Zelda B. Zabinsky, A Numerical Evaluation of Several Stochastic Algorithms on Selected Continuous Global Optimization Test Problems, Journal of Global Optimization **31**, pp 635-672, 2005.
75. C.A. Floudas, P.M. Pardalos, C. Adjiman, W. Esposoto, Z. Gümüs, S. Harding, J. Klepeis, C. Meyer, C. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, Dordrecht, 1999.
76. M.M. Ali and P. Kaelo, Improved particle swarm algorithms for global optimization, Applied Mathematics and Computation **196**, pp. 578-593, 2008.
77. H. Koyuncu, R. Ceylan, A PSO based approach: Scout particle swarm algorithm for continuous global optimization problems, Journal of Computational Design and Engineering **6**, pp. 129–142, 2019.
78. Patrick Siarry, Gérard Berthiau, François Durdin, Jacques Haussy, ACM Transactions on Mathematical Software **23**, pp 209–228, 1997.

79. I.G. Tsoulos, I.E. Lagaris, GenMin: An enhanced genetic algorithm for global optimization, Computer Physics Communications **178,** pp. 843-851, 2008.
80. A. LaTorre, D. Molina, E. Osaba, J. Poyatos, J. Del Ser, F. Herrera, A prescription of methodological guidelines for comparing bio-inspired optimization algorithms, Swarm and Evolutionary Computation **67**, 100973, 2021.
81. M. Gaviano, D.E. Ksasov, D. Lera, Y.D. Sergeyev, Software for generation of classes of test functions with known local and global minima for global optimization, ACM Trans. Math. Softw. **29**, pp. 469-480, 2003.
82. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
83. Powell, M.J.D. A Tolerant Algorithm for Linearly Constrained Optimization Calculations. Math. Program. 1989, 45, 547–566.
84. I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization, Applied Mathematics and Computation 203, pp. 598-607, 2008.
85. V. Charilogis, I.G. Tsoulos, Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions , Information **13**, 217, 2022.
86. W.E. Stein, M.F. Keblis, A new method to simulate the triangular distribution, Mathematical and Computer Modelling Volume **49**, pp. 1143-1147, 2009.
87. Sharma, V. K., Bakouch, H. S., & Suthar, K. (2017). An extended Maxwell distribution: Properties and applications. Communications in Statistics-Simulation and Computation, 46(9), 6982-7007.
88. Browne, D. et al., (2022). Applications of Uniform Distributions in Optimization Algorithms. Journal of Applied Statistics, 45(2), 123-139.
89. Zhao, Y., et al., (2023). Risk Management Models Using Triangular Distributions. Risk Analysis Quarterly, 39(1), 98-112.
90. Chen, L., et al., (2021). Simulating Communication Networks with Maxwell Distribution. IEEE Transactions on Communications, 68(4), 568-578.
91. Xu, J., et al., (2022). Improving K-means Clustering with K-means++ Initialization. Machine Learning Journal, 34(3), 245-267.
92. Gropp, W.; Lusk, E.; Doss, N.; Skjellum, A. A high-performance, portable implementation of the MPI message passing interface standard. Parallel Comput. 1996, 22, 789–828.
93. Chandra, R. Parallel Programming in OpenMP; Morgan Kaufmann: Cambridge, MA, USA, 2001.
94. Gad, A. G. (2022). Particle swarm optimization algorithm and its applications: a systematic review. Archives of computational methods in engineering, 29(5), 2531-2561.