MDPI

Article

EOFA: an extended version of the optimal foraging algorithm 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: The problem of finding the global minimum of a function is applicable to a multitude of real-world problems and hence a variety of computational techniques have been developed to efficiently locate it. Among these techniques, evolutionary techniques play a central role, which seek, through the imitation of natural processes, to efficiently obtain the global minimum of multidimensional functions. An evolutionary technique that has recently been introduced is the optimal foraging algorithm, which is a swarm - based algorithm, and it is notable for its reliability in locating the global minimum. In this work, a series of modifications are proposed that aim to improve the reliability and speed of the above technique, such as a termination technique based on stochastic observations, an innovative sampling method and a technique to improve the generation of offspring. The new method was tested on a series of problems from the relevant literature and a comparative study was made against other global optimization techniques with promising results.

Keywords: Global optimization; Stochastic techniques; Evolutionary methods; Swarm based methods

1. Introduction

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

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

with S:

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

Global optimization is a core part of applied mathematics [1] and computer science [2]. Additionally, it finds application in various fields such as physics [3–5], chemistry [6–8], economics [9,10], medicine [11,12] etc. Optimization methods are divided into deterministic and stochastic [13]. Deterministic methods guarantee finding the global optimal solution and are usually applied to small problems [14]. The most commonly appeared techniques of the first category are the so - called interval techniques [15,16], which divide the initial domain of the function until it discovers a promising subset that may contain the global minimum. On the other hand, stochastic methods do not guarantee finding the global minimum and are mainly applied to large problems, but are inefficient when nonlinear equality constraints are involved. Stochastic methods include a variety of techniques, such as Controlled Random Search methods [17–19], Simulated Annealing methods [20–22], methods based on the Multistart technique [23,24] etc. Also, in the recent literature, a variety of hybrid methods have appeared that combine different optimization techniques

Citation: Kyrou, G.; Charilogis, V.; Tsoulos, I.G. EOFA: an extended version of the optimal foraging algorithm for global optimization problems. *Journal Not Specified* **2024**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

Copyright: © 2024 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

[25,26]. Furthermore, due to the widespread usage of parallel programming architectures, a variety of methods have been proposed that take advantage of such systems [27–30].

The above techniques present various problems that had to be addressed by the researchers, such as, for example, the initialization of the possible solutions, the termination rules, etc. A number of researchers have dealt with the topic of initialization of candidate solutions and have proposed techniques such, as the repulsion method [31], an adaptive sampling method using the Kriging model [32], Monte Carlo based sampling methods [33], a method which utilizes an Radial Basis Function (RBF) network [34] to obtain samples [35] etc. Additionally, in the direction of termination methods, a series of techniques have been proposed, such as sequential stopping rules [36], a stopping rule based on asymptotic considerations [37], a stopping rule that utilizes the likelihood of obtaining the specific set of solutions [38] etc.

An important group of stochastic techniques that has attracted the interest of many researchers is the group of evolutionary techniques. The branch of evolutionary computing includes the study of foundations and applications of computational techniques based on the principles of natural evolution. Evolution in nature is responsible for the "design" of all living things on earth and the strategies they use to interact with each other. This branch of methods includes Differential Evolution methods [39,40], Particle Swarm Optimization (PSO) methods [41–43], Ant Colony optimization methods [44,45], Genetic algorithms [46,47] etc. Zhu and Zhang proposed a novel method in this area, that tackles optimization problems with a strategy based on the animal foraging behavior [48]. This Optimal Foraging Algorithm (OFA) is a swarm-based algorithm motivated by animal behavioral ecology. Also, Fu et al. proposed a variation of the OFA algorithm with the incorporation of the Differential Evolution technique [49]. Additionally, Jian and Zhu proposed a modification of the OFA algorithm with direction prediction [50] in order to enhance the performance of the original algorithm. Recently, Chen Ding and Guang Yu Zhu introduced an improved quasi-adversarial optimal social behavior search algorithm (QOS-OFA) to handle optimization problems [51].

The current work introduces a series of modifications to the OFA algorithm in order to speed up the process and to increase the effectiveness of the algorithm. These modifications include:

- Incorporation of a sampling technique that utilizes the K-Means method [52]. The points that will be sampled will more accurately and quickly lead to the global minimum of the function and additional points that might have been close enough using another sampling technique will be discarded.
- Use a termination technique based on stochastic observations. At each iteration of the algorithm, the smallest functional value is recorded. If this remains constant for a predetermined number of iterations, then the method terminates. This way, the method will terminate in time without needlessly wasting computing time on iterations that will simply produce the same result.
- Improving the offspring produced by the method using a few steps of a local minimization method. In the current work, a BFGS variant [53] was used as a local search procedure.

The rest of this paper is divided into the following sections: in section 2 the proposed method is fully described. In section 3 the benchmark functions are presented as well as the experimental results, and finally in section 4 some conclusions and guidelines for future work are discussed.

2. The proposed method

The OFA algorithm uses nature as a source of inspiration. The OFA algorithm has as its main purpose the evolution of the population using some iterative steps which will be presented in more detail below. The quality of the initial population can be improved, allowing the search space to be explored more efficiently. The first major modification introduced to the above technique is the use of a sophisticated technique of sampling

points from the objective function. This sampling is based on the K-means technique and is presented in detail in the subsection 2.2. The second modification is to apply a few steps of a local search procedure in offspring creation, if an offspring is not consistent with the bounds $[a, b]$ of the objective function $f(x)$. With this modification, it is ensured that a new point will be within the field definition of the function on the one hand, and on the other hand, it will have a slightly lower value than its predecessor. The third modification is to use an efficient termination rule to terminate the global optimization method effectively. This termination rule is outlined in subsection 2.3.

2.1. The main steps of the algorithm

The main steps of the used global optimization method are the following:

1. **Initialization step**.
   (a) **Define** as N_c the number of elements in the population.
   (b) **Define** as N_g the maximum number of allowed iterations.
   (c) **Initialize** randomly the members of the population in set S. This forms the initial population denoted as P. The sampling is performed using the procedure of subsection 2.2.
   (d) **Create** the QOP population, that stands for the quasi-opposite population of P, by calculating the quasi-opposite positions of P .
   (e) **Set** $t = 0$, the iteration number.
   (f) **Select** the N_c fittest solutions from the population set $\{P, \text{QOP}\}$ using the QOBL method proposed in [51].
2. **Calculation step**.
   (a) **Produce** the new offspring solutions x^{t+1} based on the ranking order using the following equation:

$$x_j^{t+1} = \begin{cases} x_j^t + K(t) \times (r_1 - r_2) \times \left(x_j^t - x_b^t\right) + (1 - K(t)) \times \left(x_1^t - x_r^t\right) & , j = 2, \ldots, N_c \\ x_1^t + K(t) \times (r_1 - r_2) \times \left(x_1^t - x_{N_c}^t\right) & , j = 1 \end{cases} \tag{2}$$

   where r_1, r_2 are random numbers in $[0, 1]$ and x_r^t is a randomly selected solution from the population. The function $K(t)$ is defined as

$$K(t) = \cos\left(\frac{\pi t}{2N_g}\right) \tag{3}$$

   (b) **For** $j = 1, \ldots, N_c$ **do**
      i. **If** $x^{t+1} \notin S$ then $x^{t+1} = \text{LS}(x^t, B_i)$. This step applies a limited number of steps of a local search procedure LS(x) to the current point. The parameter B_i denotes the number of steps for the local search procedure. In the current work the BFGS local optimization procedure have been used, but other local search methods could be employed such as Gradient Descent [54], Steepest Descent [55] etc.
      ii. **if** $\frac{\lambda_j^{t+1} \times f\left(x^{t+1}\right)}{1 + (t+1) \times \lambda_j^{t+1}} < \frac{f\left(x^t\right)}{t}$ then select x^{t+1} for the next population else select x^t. The values λ_j^{t+1} are uniformly distributed random numbers in $[0, 1]$.
   (c) **End For**
   (d) **Sort** all solutions in the current population fro the best to worst according to the function value.
3. **Termination check step**.
   (a) **Set** $t = t + 1$

(b) **Calculate** to stopping rule in the work of Charilogis [56]. This termination method is described in subsection 2.3.

(c) **If** the termination criteria are not met then go to Calculation step, **else** terminate.

The steps of the proposed method are also graphically shown in Figure 1.

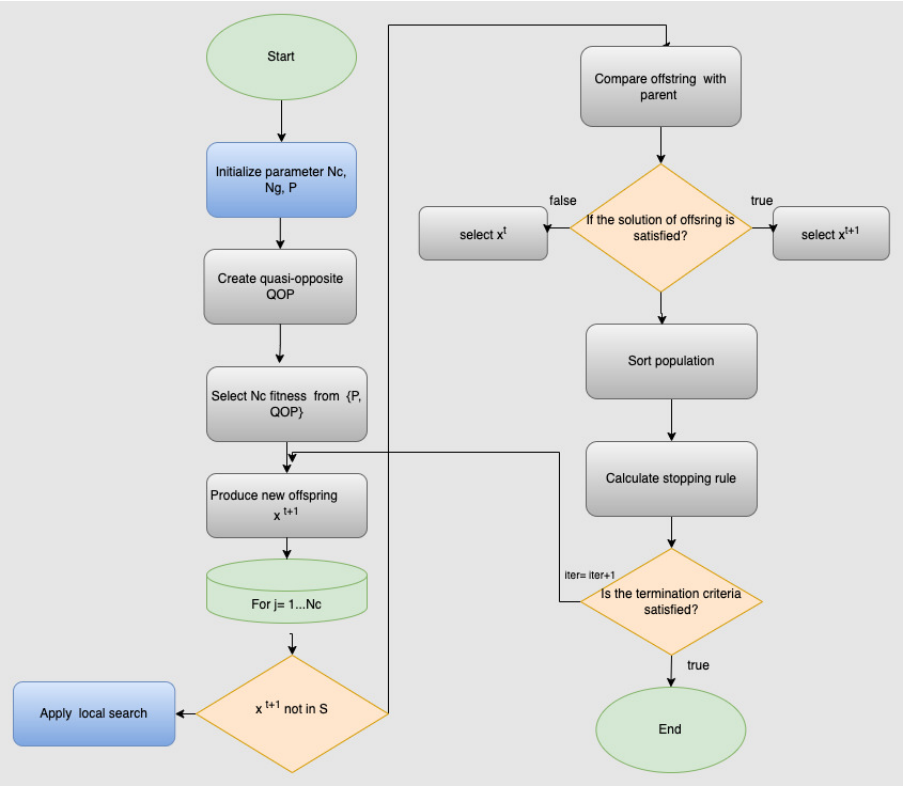

Figure 1. The steps of the proposed method as a series of steps.

2.2. The proposed sampling procedure

The proposed sampling technique produces initially a series of samples from the objective function and subsequently, through the application of the K-means technique, only the located centers are considered as the final samples. The method was introduced by James MacQueen[57] and it is a clustering algorithm that has been used in data analysis and machine learning in a variety of research papers [58,59]. The algorithm seeks to estimate the centers of possible teams in a set of samples and its main steps are listed subsequently:

1. **Define** as k the number of clusters.
2. **Draw** randomly N_m initial points $x_i,\ i = 1, \ldots, N_m$ from the objective function.
3. **Assign** randomly each point $x_i,\ i = 1, ..., N_m$ in a cluster $S_j,\ j = 1, \ldots, k$.
4. **For** every cluster $j = 1..k$ **do**
   (a) **Set** as M_j the number of points in S_j
   (b) **Compute** the center of the cluster c_j as

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

5. **EndFor**
6. **Repeat**
   (a) Set $S_j = \{\},\ j = 1..k$
   (b) **For** each point $x_i,\ i = 1, ..., N_m$ **do**

i. **Set** $j^* = \text{argmin}_{m=1}^{k}\{D(x_i, c_m)\}$. The function $D(x, y)$ is the Euclidean distance of points (x, y).
ii. **Set** $S_{j^*} = S_{j^*} \cup \{x_i\}$.

(c) **EndFor**
(d) **For** each center c_j, $j = 1..k$ **do**
i. **Update** the center c_j as

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

(e) **EndFor**

7. **If** there is no significant change in centers c_j **terminate** the algorithm and return the k centers as the final set of samples.

2.3. The used termination rule

At every iteration t, the difference between the current best value $f_{min}^{(t)}$ and the previous best value $f_{min}^{(t-1)}$ is computed:

$$\delta^{(t)} = \left| f_{\min}^{(k)} - f_{\min}^{(k-1)} \right| \tag{4}$$

The algorithm terminates when $\delta^{(t)} \leq \epsilon$ for series of predefined consecutive iterations N_k, where ϵ is a small positive number, for example 10^{-6}.

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 [60,61] 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 [62–66]. The definitions of these functions are given below:

- **Ackley** function:

$$f(x) = -a \exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1)$$

where a=20.0.

- **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.
- **ExtendedF10** function:

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

- **F12** function:

$$f(x) = \frac{\pi}{n}\left(10\sin(\pi y_1) + \sum_{i=1}^{n-1}\left((y_i-1)^2\left(1+10\sin^2(\pi y_{i+1})\right)\right) + (y_n-1)^2\right) + \sum_{i=1}^{n} u(x_i,10,100,4)$$

where

$$y_i = 1 + \frac{x_i+1}{4}$$

and

$$u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i-a)^m, & x_i < -a \end{cases}$$

- **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:

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

- **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. $f(x) = \text{Gkls}(x, n, w)$, is a constructed function with w local minima presented in [67], with $x \in [-1, 1]^n$. For the conducted experiments the values $n = 2, 3$ and $w = 50$ were utilized.
- **Goldstein and Price function**

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

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[68]. 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.

- **Sphere** function:

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

- **Schwefel** function**:**

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

- **Schwefel 2.21** function:

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

- **Schwefel 2.22** function:

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

- **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 experiments were performed using the test functions presented previously. Every experiment was conducted 30 times using different seeds for the random generator each time. The averages of function calls were measured and presented in the following tables. The software was coded in ANSI C++ with the assistance of the OPTIMUS optimization package, which is freely available from https://github.com/itsoulos/OPTIMUS

(accessed on 2 July 2024). The values for the experimental parameters are shown in Table 1. Generally, the region of attraction $A(x^*)$ of a local minimum x^* is defined as:

$$A(x^*) = \{x : x \in S, L(x) = x^*\} \quad (5)$$

where $L(x)$ is a local search procedure, such as BFGS. Global optimization methods, usually find points which are in the region of attraction $A(x^*)$ of a local minimum x^* but not necessarily the local minimum itself. For this reason, at the end of their execution, a local minimization method is applied in order to ensure the exact finding of a local minimum. Of course, this does not imply that the minimum that will be located will be the total minimum of the function, since a function can contain tens or even hundreds of local minima.

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	500
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
B_i	Number of iterations for BFGS	3

The following applies to table 2:

1. The column FUNCTION denotes the name of the objective problem.
2. The column GENETIC denotes the application of a genetic algorithm to the objective problem. The genetic algorithm has N_c chromosomes and the maximum number of allowed generations was set to N_g.
3. The column PSO stands for the application of Particle Swarm Optimizer 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.
4. The column EOFA represents the application of the proposed method using the values for the parameters shown in Table 1.
5. 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 average function calls.

FUNCTION	GENETIC	PSO	EOFA
ACKLEY	16754	24518	1346
BF1	10466	9912	1856
BF2	10059	9364	1738
BF3	7290	9847	1408
BRANIN	10032	5940	1479
CAMEL	11069	7132	1540
EASOM	10587	4922	1304
EXP4	10231	7382	1651
EXP8	10622	7644	1891
EXP16	10458	8050	2145
EXP32	10202	8800	2165
EXTENDEDF10	10739	14363	1361
F12	31277	23705	4340
F14	14598	26296	1617
F15	8215	13599	1620
F17	9202	11851	1438
GKLS250	10198	5488	1404
GKLS350	9861	6029	1325
GOLDSTEIN	11901	9244	1751
GRIEWANK2	13612	10315	1602
GRIEWANK10	14750	15721	3092
HANSEN	13053	7636	1559
HARTMAN3	10066	6897	1664
HARTMAN6	11119	8061	1873
POTENTIAL3	16325	10728	2093
POTENTIAL5	34284	19307	3397
RASTRIGIN	13354	9783	1528
ROSENBROCK4	12618	9266	2134
ROSENBROCK8	15019	12854	2985
ROSENBROCK16	17150	21074	4151
SHEKEL5	13927	8383	2021
SHEKEL7	13688	8491	2029
SHEKEL10	13722	8746	2120
TEST2N4	10522	7815	1608
TEST2N5	10847	8393	1658
TEST2N6	11180	9385	1782
TEST2N7	11485	10561	1876
SPHERE	3942	8330	1199
SCHWEFEL	4688	9037	1242
SCHWEFEL2.21	6233	7872	1286
SCHWEFEL2.22	81980	88227	5309
SINU4	12920	7250	1700
SINU8	12703	8202	2202
SINU16	12404	10640	2188
TEST30N3	16692	7750	1912
TEST30N4	19159	10036	1820
SUM	**651203**	**564846**	**91409**

A statistical comparison of these results is also outlined in Figure 2.

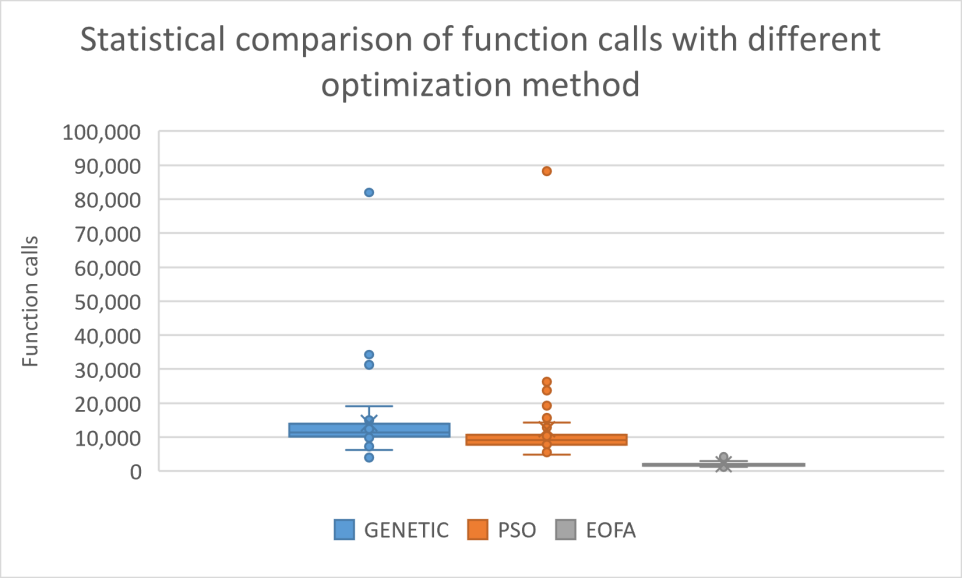

Figure 2. Statistical representation of the function calls for different optimization methods.

As can be seen, the present method drastically reduces the required number of function calls to almost all objective functions. This, of course, also results in a significant reduction of the required computing time for finding the global minimum. Also, a scatter plot for the used optimization methods is shown in Figure 3, which also depicts the superiority of the current method.

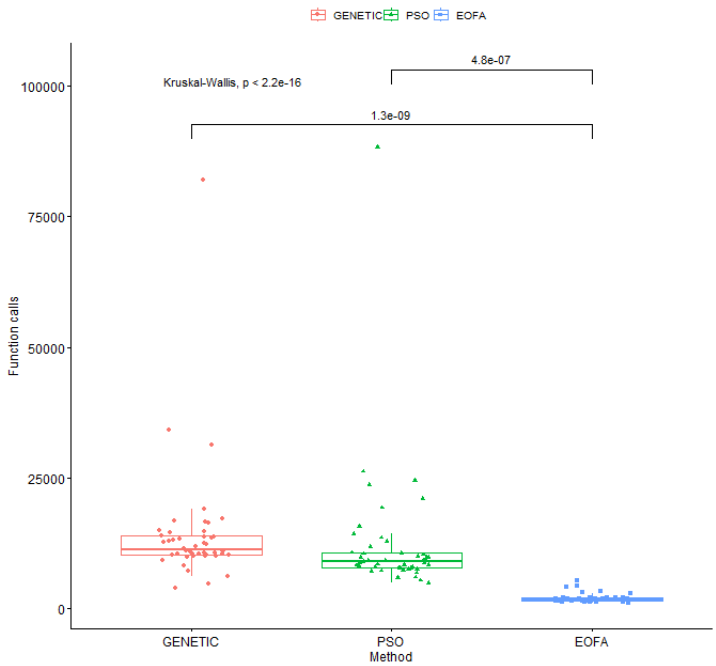

Figure 3. Scatter Plot Representation of Optimization Methods with Kruskal-Wallis Ranking

Furthermore, an additional experiment was done in order to measure the importance of the K-means sampling in the proposed method. In this test, 3 different sampling methods were used in the proposed method:

1. The column UNIFORM stands for the application of uniform sampling in the proposed method.

2. The column TRIANGULAR represents the application of the triangular distribution [69] to sample the initial points of the proposed method.
3. The column KMEANS represents the sampling method presented in the current work.

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

FUNCTION	UNIFORM	TRIANGULAR	KMEANS
ACKLEY	1803	1330	1346
BF1	2637	2146	1856
BF2	2470	2011	1738
BF3	1863	1366	1408
BRANIN	2023	1518	1479
CAMEL	2174	1671	1540
EASOM	1755	1264	1304
EXP4	2428	1844	1651
EXP8	2500	1939	1891
EXP16	2568	2026	2145
EXP32	2534	1976	2165
EXTENDEDF10	1812	1360	1361
F12	6085	4196	4340
F14	2225	1691	1617
F15	1950	1512	1620
F17	1932	1424	1438
GKLS250	1895	1403	1404
GKLS350	1789	1249	1325
GOLDSTEIN	2516	2013	1751
GRIEWANK2	2242	1711	1602
GRIEWANK10	3776	3416	3092
HANSEN	2163	1678	1559
HARTMAN3	2289	1757	1664
HARTMAN6	2541	2065	1873
POTENTIAL3	2841	2366	2093
POTENTIAL5	4029	3960	3397
RASTRIGIN	2105	1779	1528
ROSENBROCK4	3255	2949	2134
ROSENBROCK8	4059	3468	2985
ROSENBROCK16	4963	4391	4151
SPHERE	1533	1033	1199
SCHWEFEL	1635	1139	1242
SCHWEFEL2.21	1724	1230	1286
SCHWEFEL2.22	8363	8663	5309
SHEKEL5	2945	2515	2021
SHEKEL7	3008	2567	2029
SHEKEL10	3160	2683	2120
TEST2N4	2339	1804	1608
TEST2N5	2388	1847	1658
TEST2N6	2478	1962	1782
TEST2N7	2552	2044	1876
SINU4	2444	1954	1700
SINU8	2881	2388	2202
SINU16	3694	3300	2188
TEST30N3	2189	1654	1912
TEST30N4	2282	2374	1820
SUM	**124837**	**102636**	**91309**

The statistical comparison for the previous experiment is graphically outlined in Figure 4. The proposed sampling method significantly outperforms the remaining sampling techniques in almost all problems. Furthermore, the proposed global optimization technique outperforms the rest of the compared techniques, even if a different sampling technique is used.

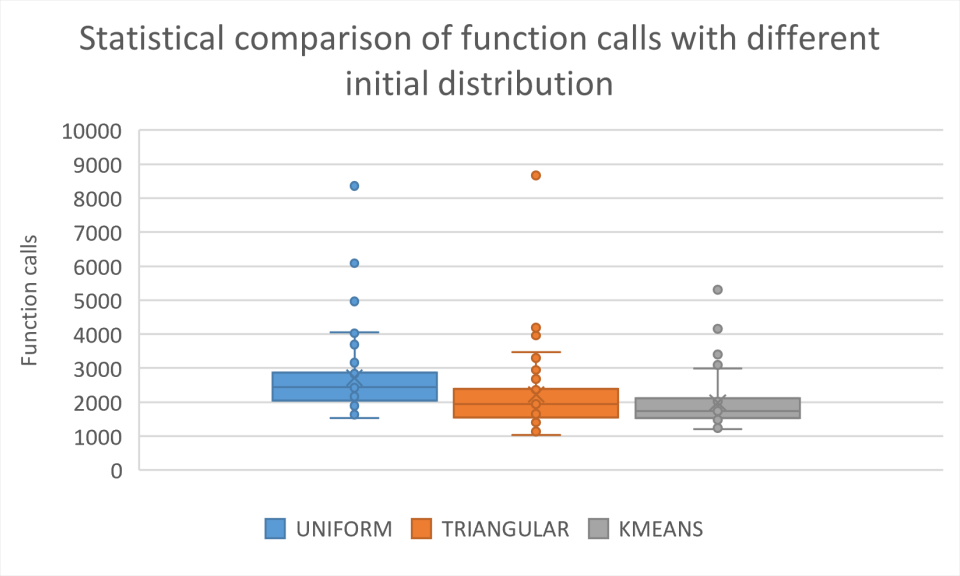

Figure 4. Statistical representation for different sampling methods.

Moreover, a scatter plot representation of the used Distributions with Kruskal-Wallis Ranking is depicted in Figure 5.

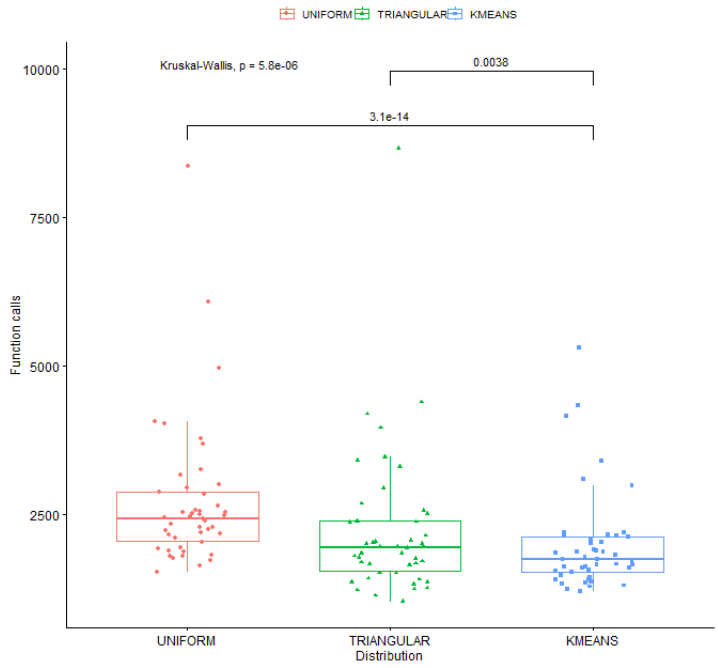

Figure 5. Scatter Plot Representation of the used Distributions with Kruskal-Wallis Ranking.

In this test, the p-value remains very low at any level of significance. Consequently, the null hypothesis is rejected, indicating that the experimental tests are statistically significant.

Furthermore, an additional test was carried out, where the function ELP was used to measure the effectiveness of the proposed method, as the dimension increased. This function is defined as:

$$f(x) = \sum_{i=1}^{n} \left(10^6\right)^{\frac{i-1}{n-1}} x_i^2$$

where the parameter n defines the dimension of the function. A comparison between the genetic algorithm and the proposed method as n increases is shown in Figure 6.

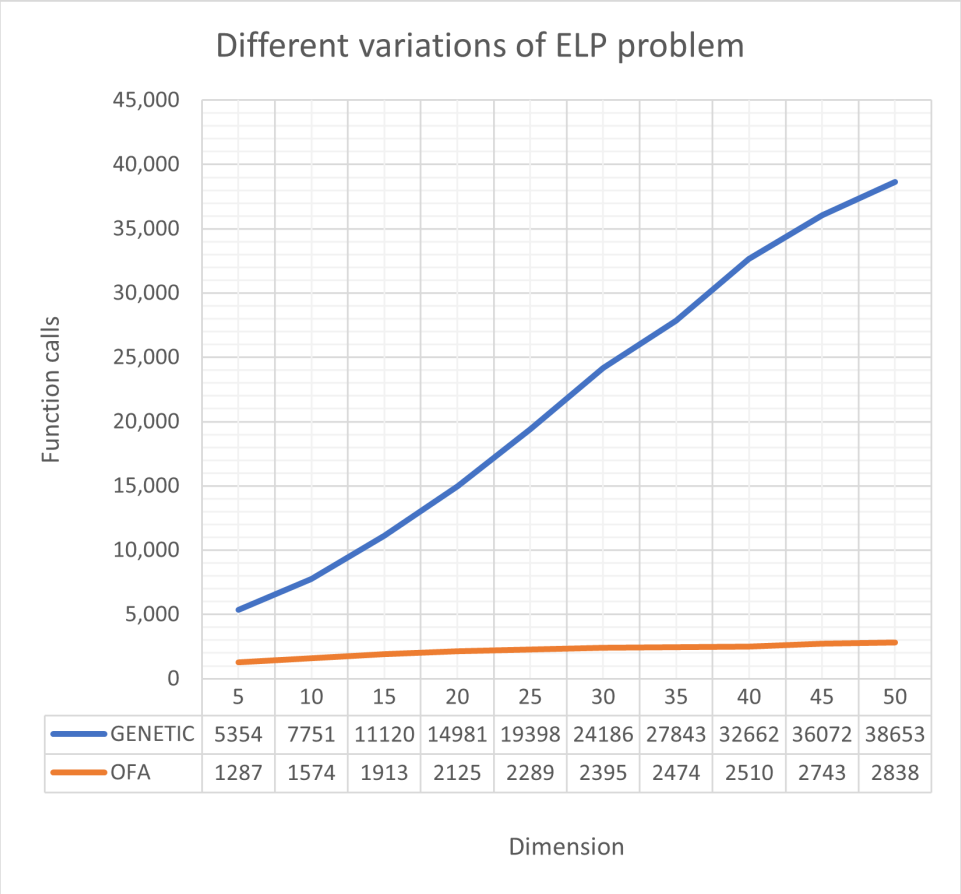

Figure 6. Different variations of the ELP problem.

The required number of calls for the current method increases at a much lower rate than in the case of the Genetic Algorithm. This means that the method can cope with large-scale problems without significantly increasing the required computing time.

4. Conclusions

Three modifications were proposed to the EOFA optimization method in this article. The modifications are primarily aimed at improving the efficiency and speed of the global optimization algorithm. The first modification proposed the application of a sampling technique which incorporates the K-Means method[52]. With the proposed modification, the points where the sampling was achieved helped to find the global minimum with the greatest accuracy and in the least possible time. The second amendment concerns termination rules. Termination rules hel terminate functions immediately without unnecessarily wasting computational time on iterations. The third modification concerns the refinement of the offspring produced using the BFGS method [53] as a local search procedure. In the experiments conducted, different sampling techniques were used for the proposed method. More specifically, the following techniques were used: Uniform, Triangular, K-means where

it was found that the K-means sampling technique yields much better results than the other two techniques and the total number of calls is extremely lower.

Experiments were also performed using different optimization methods. More specifically, the following were used: Genetic, PSO, EOFA where it was observed that the average number of EOFA calls is very limited compared to the other two.

Since the experimental results have been shown to be extremely promising, further efforts can be made to develop the technique in various areas. Among the future extensions of the application may be the use of parallel computing techniques to speed up the optimization process, such as the integration of MPI [70] or the OpenMP library [71].

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. Intriligator, M. D. (2002). Mathematical optimization and economic theory. Society for Industrial and Applied Mathematics.
2. A. Törn, M.M. Ali, S. Viitanen, Stochastic global optimization: Problem classes and solution techniques. Journal of Global Optimization **14**, pp. 437-447, 1999.
3. 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.
4. E. Iuliano, Global optimization of benchmark aerodynamic cases using physics-based surrogate models, Aerospace Science and Technology **67**, pp.273-286, 2017.
5. Q. Duan, S. Sorooshian, V. Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resources Research **28**, pp. 1015-1031 , 1992.
6. S. Heiles, R. L. Johnston, Global optimization of clusters using electronic structure methods, Int. J. Quantum Chem. **113**, pp. 2091–2109, 2013.
7. 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.
8. 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.
9. Zwe-Lee Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints, IEEE Transactions on **18** Power Systems, pp. 1187-1195, 2003.
10. C. D. Maranas, I. P. Androulakis, C. A. Floudas, A. J. Berger, J. M. Mulvey, Solving long-term financial planning problems via global optimization, Journal of Economic Dynamics and Control **21**, pp. 1405-1425, 1997.
11. Eva K. Lee, Large-Scale Optimization-Based Classification Models in Medicine and Biology, Annals of Biomedical Engineering **35**, pp 1095-1109, 2007.
12. Y. Cherruault, Global optimization in biology and medicine, Mathematical and Computer Modelling **20**, pp. 119-132, 1994.
13. L. Liberti, S. Kucherenko, Comparison of deterministic and stochastic approaches to global optimization. International Transactions in Operational Research **12**, pp. 263-285, 2005.
14. S.H. Choi, V. Manousiouthakis, Global optimization methods for chemical process design: Deterministic and stochastic approaches. Korean Journal of Chemical Engineering **19**, pp. 227-232, 2002.
15. M.A. Wolfe, Interval methods for global optimization, Applied Mathematics and Computation **75**, pp. 179-206, 1996.

16. T. Csendes and D. Ratz, Subdivision Direction Selection in Interval Methods for Global Optimization, SIAM J. Numer. Anal. **34**, pp. 922–938, 1997.
17. W. L. Price, Global optimization by controlled random search, Journal of Optimization Theory and Applications **40**, pp. 333-348, 1983.
18. Ivan Krivy, Josef Tvrdik, The controlled random search algorithm in optimizing regression models, Computational Statistics & Data Analysis **20**, pp. 229-234, 1995.
19. M.M. Ali, A. Torn, and S. Viitanen, A Numerical Comparison of Some Modified Controlled Random Search Algorithms, Journal of Global Optimization **11**,pp. 377–385,1997.
20. S. Kirkpatrick, CD Gelatt, , MP Vecchi, Optimization by simulated annealing, Science **220**, pp. 671-680, 1983.
21. L. Ingber, Very fast simulated re-annealing, Mathematical and Computer Modelling **12**, pp. 967-973, 1989.
22. R.W. Eglese, Simulated annealing: A tool for operational research, Simulated annealing: A tool for operational research **46**, pp. 271-281, 1990.
23. I.G. Tsoulos, I.E. Lagaris, MinFinder: Locating all the local minima of a function, Computer Physics Communications **174**, pp. 166-179, 2006.
24. Y. Liu, P. Tian, A multi-start central force optimization for global optimization, Applied Soft Computing **27**, pp. 92-98, 2015.
25. M. Perez, F. Almeida and J. M. Moreno-Vega, "Genetic algorithm with multistart search for the p-Hub median problem," Proceedings. 24th EUROMICRO Conference (Cat. No.98EX204), Vasteras, Sweden, 1998, pp. 702-707 vol.2.
26. H. C. B. d. Oliveira, G. C. Vasconcelos and G. B. Alvarenga, "A Multi-Start Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows," 2006 Ninth Brazilian Symposium on Neural Networks (SBRN'06), Ribeirao Preto, Brazil, 2006, pp. 137-142.
27. J. Larson and S.M. Wild, Asynchronously parallel optimization solver for finding multiple minima, Mathematical Programming Computation **10**, pp. 303-332, 2018.
28. H.P.J. Bolton, J.F. Schutte, A.A. Groenwold, Multiple Parallel Local Searches in Global Optimization. In: Dongarra J., Kacsuk P., Podhorszki N. (eds) Recent Advances in Parallel Virtual Machine and Message Passing Interface. EuroPVM/MPI 2000. Lecture Notes in Computer Science, vol 1908. Springer, Berlin, Heidelberg, 2000.
29. R. Kamil, S. Reiji, An Efficient GPU Implementation of a Multi-Start TSP Solver for Large Problem Instances, Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 1441-1442, 2012.
30. Van Luong T., Melab N., Talbi EG. (2011) GPU-Based Multi-start Local Search Algorithms. In: Coello C.A.C. (eds) Learning and Intelligent Optimization. LION 2011. Lecture Notes in Computer Science, vol 6683. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25566-3_24
31. A.E. Sepulveda, L. Epstein, The repulsion algorithm, a new multistart method for global optimization, Structural Optimization **11**, pp. 145–152, 1996.
32. Z. Chen, H. Qiu, L. Gao et al, A local adaptive sampling method for reliability-based design optimization using Kriging model, Struct Multidisc Optim **49**, pp. 401–416, 2014.
33. T.H.-de-Mello,G. Bayraksan, Monte Carlo sampling-based methods for stochastic optimization, Surveys in Operations Research and Management Science **19**, pp. 56-85, 2014.
34. J. Park, I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation **3**, pp. 246-257, 1991.
35. I.G. Tsoulos, A. Tzallas, D. Tsalikakis, Use RBF as a Sampling Method in Multistart Global Optimization Method, Signals **3**, pp. 857-874, 2022.
36. W.E. Hart, Sequential stopping rules for random optimization methods with applications to multistart local search, Siam J. Optim. **9**, pp. 270–290, 1998.
37. I.G. Tsoulos, Modifications of real code genetic algorithm for global optimization Applied Mathematics and Computation **203**, pp. 598-607, 2008.
38. M. Ohsaki, M. Yamakawa, Stopping rule of multi-start local search for structural optimization, Struct Multidisc Optim **57**, pp. 595–603, 2018.
39. 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.
40. J. Liu, J. Lampinen, A Fuzzy Adaptive Differential Evolution Algorithm. Soft Comput **9**, pp.448–462, 2005.
41. 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.
42. Riccardo Poli, James Kennedy kennedy, Tim Blackwell, Particle swarm optimization An Overview, Swarm Intelligence **1**, pp 33-57, 2007.
43. Ioan Cristian Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters **85**, pp. 317-325, 2003.
44. M. Dorigo, M. Birattari and T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine **1**, pp. 28-39, 2006.
45. K. Socha, M. Dorigo, Ant colony optimization for continuous domains, European Journal of Operational Research 185, pp. 1155-1173, 2008.

46. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
47. Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer - Verlag, Berlin, 1996.
48. G.Y. Zhu, W.B. Zhang, Optimal foraging algorithm for global optimization, Applied Soft Computing **51**, pp. 294-313, 2017.
49. Y. Fu, W. Zhang, C. Qu, B. Huang, Optimal Foraging Algorithm Based on Differential Evolution, IEEE Access **8**, pp. 19657-19678, 2020.
50. Z. Jian, G. Zhu, Optimal foraging algorithm with direction prediction, Applied Soft Computing **111**, 107660, 2021.
51. C. Ding, G. Zhu, Improved optimal foraging algorithm for global optimization, Computing **106**, pp. 2293-2319, 2024.
52. M. Ahmed, R. Seraj, S.M.S. Islam, The k-means algorithm: A comprehensive survey and performance evaluation, Electronics **9**, 1295, 2020.
53. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.
54. S.I. Amari, Backpropagation and stochastic gradient descent method, Neurocomputing **5**, pp. 185-196, 1993.
55. J.C. Meza, Steepest descent, Wiley Interdisciplinary Reviews: Computational Statistics **2**, pp. 719-722, 2010.
56. Charilogis, V.; Tsoulos, I.G. Toward an Ideal Particle Swarm Optimizer for Multidimensional Functions. Information 2022, 13, 217.
57. 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.
58. Y. Li, H. Wu, A clustering method based on K-means algorithm, Physics Procedia **25**, pp. 1104-1109, 2012.
59. P. Arora, S. Varshney, Analysis of k-means and k-medoids algorithm for big data, Procedia Computer Science **78**, pp. 507-512, 2016.
60. 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.
61. 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.
62. M.M. Ali and P. Kaelo, Improved particle swarm algorithms for global optimization, Applied Mathematics and Computation **196**, pp. 578-593, 2008.
63. 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.
64. Patrick Siarry, Gérard Berthiau, François Durdin, Jacques Haussy, ACM Transactions on Mathematical Software **23**, pp 209–228, 1997.
65. I.G. Tsoulos, I.E. Lagaris, GenMin: An enhanced genetic algorithm for global optimization, Computer Physics Communications **178,** pp. 843-851, 2008.
66. 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.
67. 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.
68. J.E. Lennard-Jones, On the Determination of Molecular Fields, Proc. R. Soc. Lond. A **106**, pp. 463–477, 1924.
69. W.E. Stein, M.F. Keblis, A new method to simulate the triangular distribution, Mathematical and Computer Modelling Volume **49**, pp. 1143-1147, 2009.
70. 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.
71. Chandra, R. Parallel Programming in OpenMP; Morgan Kaufmann: Cambridge, MA, USA, 2001.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.