Improving the Emperor Penguin Optimizer Algorithm Through Adapted Weighted Sum Mutation Strategy with Information Vector

Ahmed Serag[1], Hegazy Zaher[2], Naglaa Ragaa[3], Heba Sayed[4]
[1]Operations Research, faculty of graduate studies for statistical research, Cairo University, Giza, Egypt
[2]Mathematical Statistics, Faculty of Graduate Studies for Statistical Research, Cairo University
E-mail: ahmede.serag1978@gmail.com, hgsabry@cu.edu.eg, naglaa777subkiii@yahoo.com, hmhmdss@yahoo.com

Keywords: metaheuristics, emperor penguin optimizer algorithm, weighted sum

Received: February 23, 2024

The Emperor Penguin Optimizer algorithm (EPO) is a recent addition to population-based metaheuristics. However, it has been observed that the algorithm occasionally gets trapped in local optima, particularly when dealing with multi-modal functions. In this paper, we present a novel modification of the Emperor Penguin Optimizer algorithm, termed the Emperor Penguin Optimizer with Weighted Sum Procedure and Information Vector (EPOWIV). The EPOWIV algorithm combines two techniques, the weighted sum procedure and the information vector. To evaluate the effectiveness of the proposed EPOWIV algorithm, a comprehensive comparative study is conducted. This study includes a comparison with the classical EPO algorithm, the EPO algorithm with the weighted sum procedure only, and the EPO algorithm with the information vector. The comparison is carried out on 21 test optimization problems. The comparative results show superiority of the EPOWIV algorithm over its counterparts. The EPOWIV algorithm consistently exhibits superior optimization performance, effectively overcoming the stagnation issues previously associated with the EPO algorithm. It consistently delivers outstanding solutions across a diverse set of test problems.

Povzetek: Članek obravnava izboljšavo algoritma Emperor Penguin Optimizer (EPO) s prilagojeno strategijo mutacije, imenovano EPO z uteženim seštevanjem in informacijskim vektorjem (EPOWIV). Avtorji analizirajo učinkovitost predlaganega algoritma EPOWIV skozi obsežno primerjalno študijo, ki vključuje 21 testnih optimizacijskih problemov. Rezultati študije kažejo, da EPOWIV učinkovito odpravlja težave stagnacije, ki so značilne za klasični EPO, ter dosledno dosega boljše optimizacijske rezultate.

1 Introduction

In recent years, optimization algorithms have garnered significant interest in various fields due to their potential to efficiently solve complex real-world problems [1]. Nature-inspired algorithms, in particular, have gained prominence for their ability to mimic the intelligence and adaptability of biological systems [2]. Among these, the Emperor Penguin Optimizer Algorithm (EPO) has emerged as a promising optimization technique, inspired by the remarkable foraging behavior and social interactions of Emperor penguins [3]. Like many algorithms, the EPO faces certain challenges, including premature convergence and suboptimal exploration capabilities. In order to further enhance the EPO's optimization performance, this paper proposes a novel extension: the integration of a Weighted Sum Mutation Strategy and information vector into the EPO algorithm. The primary objective of this research is to utilize the strengths of relocating vectors and best solutions within the EPO, enabling improved global and local search capabilities. The WSIV introduces a controlled mutation mechanism, allowing the algorithm to strike a balance between exploration and exploitation effectively. By facilitating a more diverse search space exploration, the proposed approach aims to mitigate premature convergence issues and enhance the algorithm's convergence rate while maintaining the exploitation of valuable solutions.

In this paper, a comprehensive investigation of the novel EPOWIV algorithm's design is introduced, highlighting its key components, implementation, and mathematical formulation. We conduct an in-depth empirical analysis, employing a diverse set of benchmark functions to assess the algorithm's performance.

The remainder of this paper is structured as follows: Section 2 provides a review of related works on optimization algorithms and highlights the distinct characteristics of the EPO and its limitations. Section 3 details the proposed EPOWIV algorithm, discussing the incorporation of the weighted sum mutation strategy with the information vector and its adaptation to the EPO's behavior. In Section 4, the experimental setup is presented, evaluation metrics, and performance comparisons with other optimization methods. The results and discussions are presented in Section 5, followed by conclusions and future directions in Section 6.

2 Related work

Metaheuristics are optimization algorithms that combine stochastic and local search techniques [4]. Stochasticity enables the exploration of the search space, while local search facilitates exploitation around solutions [5]. They usually used to tackle the NP-hard problems, such as the combinatorial optimization problems. Metaheuristics are commonly employed to tackle NP-hard problems, especially combinatorial optimization problems [6]. These algorithms are designed to efficiently explore the vast solution space and find near-optimal or satisfactory solutions in a reasonable amount of time [7]. Their stochastic and adaptive nature allows them to escape local optima and search for promising regions in the solution landscape, making them particularly well-suited for challenging optimization tasks. The metaheuristics can be classified into two main categories, which are the population-based and single-based methods [8]. Population-based metaheuristics involve maintaining a population of candidate solutions throughout the optimization process [9]. These methods often use mechanisms such as evolution, mutation, and crossover to create new solutions by combining or modifying existing ones. Examples of population-based metaheuristics include Genetic Algorithms, Particle Swarm Optimization, and Differential Evolution.

On the other hand, single-solution-based metaheuristics operate with only one solution at a time and iteratively improve it to search for better solutions [10]. The Emperor Penguin Optimizer algorithm is a population-based metaheuristic first proposed by Dhiman and Kumar [11]. The EPO algorithm emulates the huddling behavior of Emperor Penguins (Aptenodytes forsteri). Its primary steps include generating the huddle boundary, computing the temperature surrounding the huddle, calculating the distance, and identifying the effective mover. Many of the previous work presented in EPO considered using the algorithm to solve real-world applications such as image segmentation, power system, and energy consumption reduction. Baliarsingh and Vipsita [12] presented a chaotic EPO to optimize machine learning for classifying microarray cancer. Cao et al. [13] presented an improved EPO to enhance the efficiency of power system. Min et al. [14] presented a quantum EPO for optimizing the minimize the energy consumption of chiller loading.

Angel and Jaya [15] adapted the EPO to solve load balancing and security enrichment in wireless sensor problem. Xing [16] improved an EPO algorithm and used it to enhance a multi-threshold image segmentation .Dhiman et al. [17] improved the EPO algorithm to make able to deal with discrete optimization problems and they used the improved version to solve feature selection .Khan et al. [18] used EPO algorithm along with deep learning model to optimize the classification of recycling waste .Cheena et al. [19] proposed EPO algorithm to optimize self-heading for sensor network based smart grid system. Babu et al. [20] developed EPO to optimize the location of AC transmission system devices in load frequency control. Serag et al. [21] enhanced EPO by incorporating an information vector, enhancing its local search process compared to the standard version. This modification enables EPO to overcome stagnation present in certain multi-modal functions.

Table 1: Literature review summary table

Author	Contribution
Dhiman and Kumar [11]	Developed the first version of EPO
Baliarsingh and Vipsita [12]	Developed a chaotic EPO to optimize machine learning for classifying microarray cancer
Cao et al. [13]	Used EPO to enhance the efficiency of power system
Min et al. [14]	Developed a quantum EPO for optimizing the minimize the energy consumption of chiller loading
Angel and Jaya [15]	Solved load balancing and security enrichment in wireless sensor problem using EPO
Xing [16]	Enhanced a multi-threshold image segmentation using EPO
Dhiman et al. [17]	Solved feature selection after adopting EPO to make it deal with discrete optimization problems
Khan et al. [18]	Presented EPO to optimize the classification of recycling waste
Cheena et al. [19]	proposed EPO algorithm to optimize self-heading for sensor network based smart grid system
Babu et al. [20]	developed EPO to optimize the location of AC transmission system devices in load frequency control
Serag et al. [21]	Developed a new modification for EPO that considers information vector to improve the local search procedure of the algorithm

This paper is considered an extension for our previous work presented in [21] that solves the stagnation problem of the multi-modal functions. Therefore, we presented a new modification that utilizes weighted sum methodology along with generated information vector to update the relocating procedure of the algorithm, where the new modification, as shown in the comparative results, outperforms the algorithm proposed in [21].

3 Emperor penguin optimizer algorithm

The Emperor Penguin Optimizer (EPO) is a population-based metaheuristic inspired by the collective behavior of emperor penguins. Its operations encompass the computation of ambient temperature, distances to the emperor penguins, and the effective mover. The temperature within the group is determined by aggregating the temperatures of individual penguins (T) within a defined radius (R) surrounding the crowd. Thus, the temperature distribution surrounding the crowd can be derived using Eq. (1), which is dependent on the iteration count (Itr) and the maximum number of iterations ($MaxItr$). The following notations make the illustration of the algorithm steps easier:

Notations:

TA	The ambient temperature
T	Individual penguin temperature
R	The huddle radius
Itr	Iteration count
$MaxItr$	The maximum number of iterations
D	The distance between penguins
P_{best}	The position of best penguin
P_i	The position of penguin i
P_{i+1}	The next position of penguin i
S	Social force
f, l	Random numbers related to social force calculation
A	Movement vector
M	Movement parameter

$$TA = T - \frac{MaxItr}{Itr - MaxItr}, \forall Itr < MaxItr \quad (1)$$
$$, where\ T = \begin{cases} 0, & if\ R > 1 \\ 1, & if\ R < 1 \end{cases}$$

The calculation of the distance (D) between the penguins and their emperor involves several parameters designed to prevent collisions among the penguins. These parameters include the position of the best penguin (P_{best}), the position of each individual penguin (P), the ambient temperature (TA), and the social force (S), which compels the penguins to move towards the optimal solution. The parameter A is computed for the position P_i, utilizing the movement parameter (M), set to 2, as specified in Eq. (2).

$$A = M \times (TA + |P_{best} - P_i| \times rand) - TA \quad (2)$$

Eq. (3) serves the purpose of computing the social force. This equation takes the form of a decreasing function and relies on three variables: f, l, and Itr. Both f and l are random numbers, each constrained within its respective lower and upper bounds.

$$S = \left(\sqrt{f \,.\, e^{-Itr/l} - e^{-Itr}}\right)^2 \quad (3)$$

The parameter S plays a crucial role in the calculation of distance D. Initially, it increases during the early iterations to ensure a high degree of locality, gradually diminishing as the iterations progress, eventually leading to very low locality in the later stages. Consequently, distance D is computed using equation (4) as follows:

$$D = |S \,.\, P_i - rand\ P_{best}| \quad (4)$$

The position of the penguin in the subsequent iteration (P_{i+1}) can be determined utilizing equation (5) as follows:

$$P_{i+1} = P_i - A \,.\, D \quad (5)$$

Through the alteration of penguin positions in each iteration, the algorithm continually updates the best position until the stopping criteria are met, ultimately yielding the optimal solution. The pseudocode for the algorithm can be summarized as follows:

Algorithm 1: Pseudo code of EPO

```
Input the Populatin size (N), MaxItr, and R parameters
Generate the initial population
Evaluate each solution in population and store the best solution (P_best)
Itr = 1
While Itr ≤ MaxItr do:
      i = 1
      While i ≤ N do:
            TA = T − MaxItr / (Itr − MaxItr)
            A = M × (TA + |P_best − P_i| × rand) − TA
            S = ( sqrt( f . e^(−Itr/l) − e^(−Itr) ) )^2
            D = |S . P_i − rand P_best|
            P_(i+1) = P_i − A . D
            if f(P_(i+1)) ≤ f(P_best) then:
                  P_best = P_(i+1)
            i = i + 1
      Itr = Itr + 1
Return Gbest
```

4 Enhancing EPO with Weighted Sum Mutation and Information Vector

The new modification of the algorithm stated in this paper is related to adding weighted sum mutation procedure inside the relocation process and utilizing the information gained between the best solution and the relocated positions of the penguins. The weighted sum procedure generates a new position by combining the best solution found with the relocated vector. The weights for this combination are determined based on the evaluations of each, relative to the total evaluation of both. So, the new vector gained by the weighted sum procedure can be calculated using Eq. (6). The weighted sum method could result in certain solution components exceeding the predetermined lower and upper bounds. Therefore, a maintenance procedure should be applied to rectify these out-of-range component values. Values lower than the lower bound should be adjusted to match the lower bound, while values exceeding the upper bound should be adjusted to match the upper bound. Equations (7), and (8) shows the maintenance procedure of the weighted sum method.

$$P_{weighted} = \left(\frac{f(P_{i+1})}{f(P_{i+1}) + f(P_{best})}\right) P_{relocated} + \left(\frac{f(P_{best})}{f(P_{i+1}) + f(P_{best})}\right) P_{best} \quad (6)$$

$$P_{weighted} = \left[\max(x_1, LB), \max(x_2, LB), \ldots, \max\left(x_{\dim(P)}, LB\right)\ \right] \quad (7)$$

$$P_{weighted} = \left[\min(x_1, UB), \min(x_2, UB), \ldots, \min\left(x_{\dim(P)}, UB\right)\ \right] \quad (8)$$

The information vector is created through a user-defined threshold, typically chosen from the range between 0 and 1. This information vector, denoted as P_{IV}, is constructed using the weighted sum position, $P_{weighted}$, and the best position P_{best}. To detail the procedure, the first step is to generate a random uniform number, which will be compared against the predefined threshold. If the generated random number exceeds the threshold, component j will be selected from $P_{weighted}$; otherwise, it will be chosen from P_{best}. The subsequent steps outline the process of creating the P_{IV} position:

$j = 1$
$While\ j \leq \dim(P)\ do$:
$if\ rand > threshold$:
    $P_{IV}[j] = P_{weighted}[j]$
$else$:
    $P_{IV}[j] = P_{best}[j]$
$j = j + 1$
[1]

Now, the algorithm can be upgraded adding the weighted sum procedure and the information vector process as follows and the flowchart is shown in Figure 1:

Algorithm 2: Pseudo code of EPOWIV

$Input\ the\ Populatin\ size\ (N), MaxItr, and\ R\ parameters$
$Generate\ the\ initial\ population$
$Evaluate\ each\ solution\ in\ population\ and\ store\ the\ best\ solution\ (P_{best})$
$Itr = 1$
$While\ Itr \leq MaxItr\ do$:
  $i = 1$
  $While\ i \leq N\ do$:
    $TA = T - \frac{MaxItr}{Itr - MaxItr}$
    $A = M \times (TA + |P_{best} - P_i| \times rand) - TA$
    $S = \left(\sqrt{f\ .\ e^{-Itr/l} - e^{-Itr}}\right)^2$
    $D = |S\ .\ P_i - rand\ P_{best}|$
    $P_{relocated} = P_{i+1} = P_i - A\ .\ D$
    $P_{weighted} = \left(\frac{f(P_{i+1})}{f(P_{i+1}) + f(P_{best})}\right) P_{relocated} + \left(\frac{f(P_{best})}{f(P_{i+1}) + f(P_{best})}\right) P_{best}$
    $P_{weighted} = \left[\max(x_1, LB), \max(x_2, LB), \ldots, \max\left(x_{\dim(P)}, LB\right)\ \right]$
    $P_{weighted} = \left[\min(x_1, UB), \min(x_2, UB), \ldots, \min\left(x_{\dim(P)}, UB\right)\ \right]$
    $j = 1$
    $While\ j \leq \dim(P)\ do$:
      $if\ rand > threshold$:
        $P_{IV}[j] = P_{weighted}[j]$
      $else$:
        $P_{IV}[j] = P_{best}[j]$
      $j = j + 1$
    $if\ f(P_{i+1}) \leq f(P_{best})\ then$:
      $P_{best} = P_{i+1}$
    $i = i + 1$
  $Itr = Itr + 1$
$Return\ Gbest$

5 Comparative results

This section shows the implementation of the proposed EPO algorithm that utilizes a weighted sum procedure with information vector (EPOWIV). The algorithm is coded using python programming and uploaded to the GitHub link https://github.com/ahmedssssssA/EPOWIV. The GitHub repository also includes the code of the 21 test optimization functions used in this comparative results. The comparative results are done to compare between the EOPWIV and the classical EPO [11], the weighted sum EPO (EPOW) that combines the classical EPO with the weighted sum procedure only, and the information vector EPO (EPOIV) that combines the classical EPO with the information vector process, where the EPOIV can be found in [21] and it is one of the algorithms listed in the literature summary of this paper. The boxplots show that the proposed EPOWIV algorithm robustly outperforms the other selected algorithms in this comparative results in terms of variability and median results.

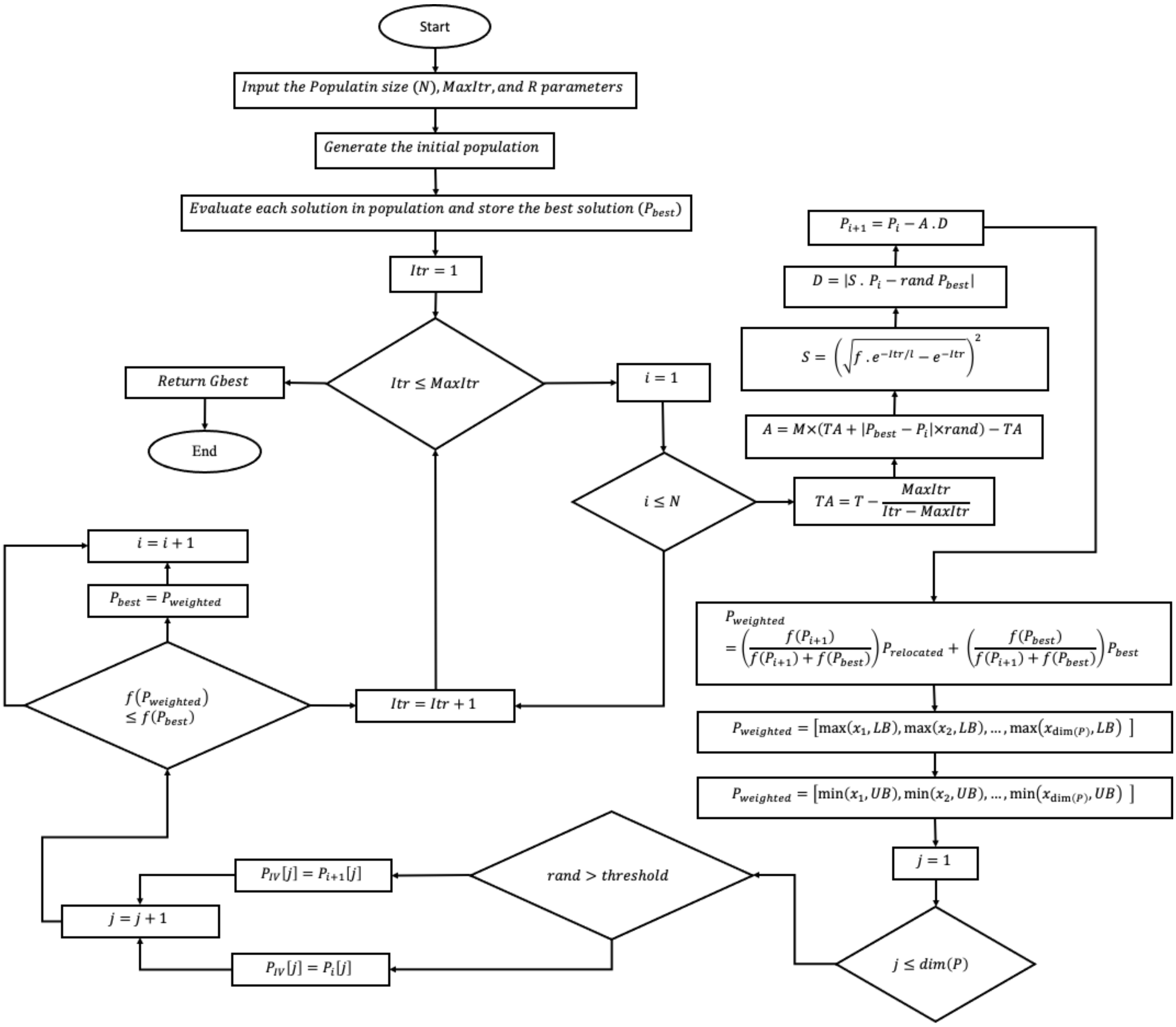

Figure 1: The flowchart of the EPOWIV algorithm

Table 2 shows the 21 test optimization functions.

After implementing the Python code for various algorithms, it was observed that the EPOWIV algorithm consistently outperforms other versions of the EPO algorithm. This superiority is evident in both the mean results and the algorithm's robustness, as indicated by the standard deviation values. The superior performance of the EPOWIV algorithm is highlighted in

Table 3: Comparative results

No.	Function	EPOIV Mean	EPOW Mean	EPO Classical Mean	EPOWIV Mean	EPOIV Std	EPOW Std	EPO Classical Std	EPOWIV Std
1	Ackley	2.306	0.002	7.600	***0.000***	0.575	0.000	2.358	***0.000***
2	Bukin N. 6	0.619	0.412	1.517	***0.243***	0.385	0.586	0.725	***0.000***
3	Cross-in-Tray	***-2.063***	***-2.063***	***-2.063***	***-2.063***	0.000	0.000	0.000	***0.000***
4	Drop-Wave	-0.994	***-1.000***	-1.004	***-1.000***	0.019	***0.000***	0.001	***0.000***
5	Griewank	1.368	0.513	33.537	***0.000***	0.242	0.490	17.253	***0.000***
6	Langermann	***-4.144***	-4.136	-4.134	-4.131	0.013	0.023	0.019	***0.000***
7	Levy	***0.015***	0.346	0.487	0.183	0.029	0.151	0.158	***0.000***
8	Rastrigin	2.219	3.159	18.406	***0.000***	1.600	5.926	11.326	***0.000***
9	Schaffer N. 2	***0.000***	0.001	0.004	***0.000***	0.000	0.003	0.005	***0.000***
10	Schaffer N. 4	0.293	***0.293***	0.296	***0.293***	0.001	0.001	0.004	***0.001***
11	Shubert	***-186.727***	-185.665	-186.039	-186.632	0.008	1.136	0.598	***0.000***
12	Bohachevsky	0.006	0.058	0.380	***0.000***	0.012	0.120	0.228	***0.000***
13	Perm 0	***0.03***	214.063	0.09	0.08	***0.000***	114.381	12163.566	***0.000***
14	Rotated Hyper-Ellipsoid	21.166	0.255	1258.814	***0.000***	9.007	0.107	929.035	***0.000***
15	Sphere	***0.000***	***0.000***	0.013	***0.000***	0.000	0.000	0.008	***0.000***
16	Sum of Different Powers	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***
17	Sum Squares	0.012	***0.000***	0.001	***0.000***	0.015	0.000	0.878	***0.000***
18	Booth	***0.000***	0.003	0.001	***0.000***	0.000	0.004	0.001	***0.000***
19	Matyas	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***
20	Zakharov	0.631	0.003	10.052	***0.000***	0.803	0.002	3.693	***0.000***
21	Three-Hump Camel	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***

6 Pressure vessel design

In this section, the pressure vessel design, a real-world problem, is solved using EPOWIV. This problem is proposed by Kannan and Kramer [22] to minimize the fabrication cost. Figure 2 shows the isometric view of the pressure vessel. There are four variables needed to solve this problem, which are the thickness of the shell (T_s), the thickness of the head (T_h), the inner radius (R), and the length of the cylindrical part (L). The mathematical model of this problem is as follows:

$$Consider\ Z = [z_1, z_2, z_3, z_4] = [T_s, T_h, R, L]$$

$$Min\ f(Z) = 0.6224\, z_1 z_3 z_4 + 1.7781 z_2 z_3^2 + 3.1661 z_1^2 z_4 + 19.84 z_1^2 z_3 \quad (9)$$

Subject to:

$$g_1 = -z_1 + 0.0193 z_3 \le 0 \quad (10)$$

$$g_2 = -z_3 + 0.000953 z_3 \le 0 \quad (11)$$

$$g_3 = -\pi z_3^2 z_4 - 4\square 3\square \pi z_3^3 + 1{,}296{,}000 \le 0 \quad (12)$$

$$g_4 = z_4 - 240 \le 0 \quad (13)$$

$$i \times 0.0625 \le z_1, z_2 \le 99 \times 0.0625, \forall i = 1, 2, \dots, 99 \quad (14)$$

$$10 \le z_3, z_4 \le 200 \quad (15)$$

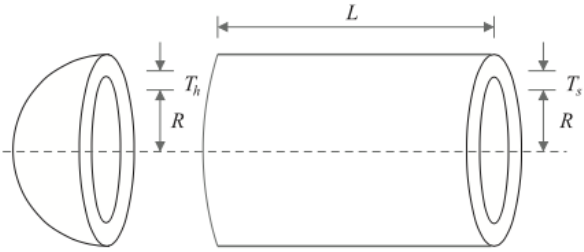

Figure 2: The pressure vessel

The problem has been solved in [23], but we found after checking the values of their solution vector that the solution doesn't satisfy the problem constraints. Their solution is $Z^* = [0.778099, 0.383241, 40.315121, 200]$

and It found that $z_1$and z_2 are not integer multipliers of 0.0625. Constraint (12) is broken, since the right-hand side of the constraint according to their solution vector is equal to 319.75, while it should be greater than 0.

We coded the problem constraints and objective, then adapted the proposed EPOWIV to solve the pressure vessel design problem. The code can be found in https://github.com/ahmedsssssA/EPOWIV/blob/main/EPO_PVD. After solving the problem, we found that best solution found by the proposed EPOWIV algorithm is 3320.58 with $Z^* = [1.3125,\ 0.0625,\ 65.7733789, 10]$. This solution satisfies all the problem constraints and shows better objective value than the infeasible solution found by [23]. The statistical results can be summarized as follows:

Best	Mean	Worst	Std. Dev.	Median
3289.4	3585.11	5466.4	523.4	3292.58

. Furthermore, the boxplots presented in figures 1: , 2, and 3 vividly illustrate the distribution of values for each algorithm. These boxplots showcase key statistical metrics, including the median, first quartile, and third quartile. Collectively, these results provide compelling evidence of the efficiency and overall superiority of the EPOWIV algorithm when compared to the other algorithms. The boxplots show that the proposed EPOWIV algorithm robustly outperforms the other selected algorithms in this comparative results in terms of variability and median results.

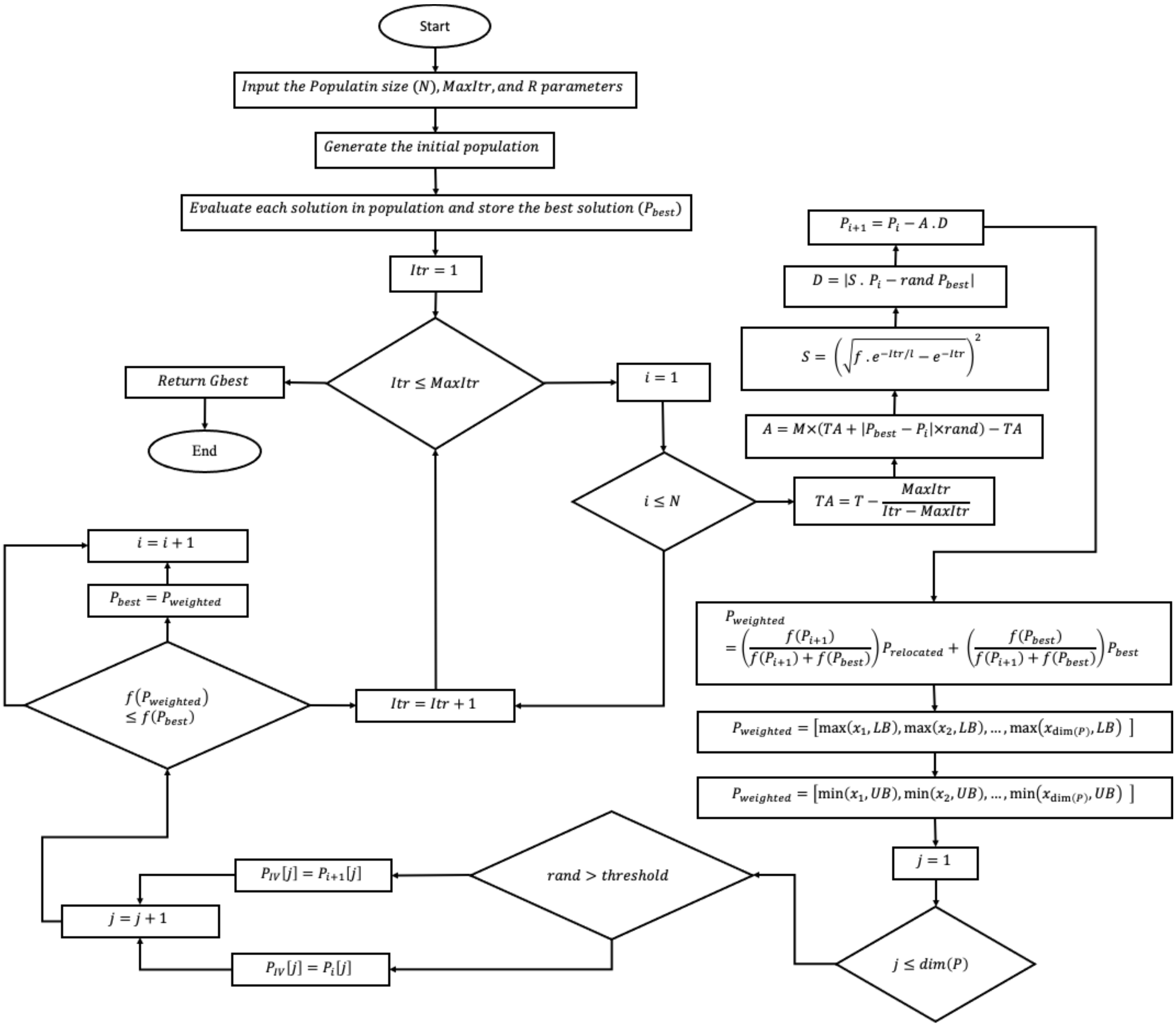

Figure 1: The flowchart of the EPOWIV algorithm

Table 2: Test optimization functions used in comparative results

No.	**Function Name**	$\boldsymbol{f(x)}$
1	Ackley	$20\left(e^{-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}}\right) - \left(e^{\frac{1}{d}\sum_{i=1}^{d} cos(2\pi x_i)}\right) + 20 + e$

No.	Function Name	$\boldsymbol{f(x)}$
2	Bukin N. 6	$f(x,y) = 100\sqrt{\lvert y - 0.01x^2 \rvert} + 0.01\lvert x+10 \rvert$
3	Cross-in-Tray	$f(x,y) = -0.0001 \vdash (\lvert \sin(x)\sin(y)\exp\left(\left\lvert 100 - \frac{\sqrt{x^2+y^2}}{\pi}\right\rvert\right)\rvert + 1)^{0.1}$
4	Drop-Wave	$f(x,y) = -\frac{1+\cos\left(12\sqrt{x^2+y^2}\right)}{0.5(x^2+y^2)+2}$
5	Griewank	$f(\boldsymbol{x}) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$
6	Langermann	$f(\boldsymbol{x}) = \sum_{i=1}^{m} c_i \cdot \exp\left(-\frac{1}{\pi}\sum_{j=1}^{n}\left(x_j - a_{ij}\right)^2\right) \cdot \cos\left(\pi\sum_{j=1}^{n}\left(x_j - a_{ij}\right)^2\right)$ $A = \begin{bmatrix} 3 & 5 \\ 5 & 2 \\ 2 & 1 \\ 1 & 4 \\ 7 & 9 \end{bmatrix}$
7	Levy	$f(\boldsymbol{x}) = \sin^2(\pi w_1) + \sum_{i=1}^{n-1}(w_i - 1)^2[1 + 10\sin^2(\pi w_i + 1)] + (w_n - 1)^2[1 + \sin^2(2\pi w_n)],$ $where\ w_i = 1 + \frac{x_i - 1}{4}$
8	Rastrigin	$f(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$
9	Schaffer N. 2	$f(x,y) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{[1 + 0.001(x^2+y^2)]^2}$
10	Schaffer N. 4	$f(x,y) = 0.5 + \frac{\cos(\sin(\lvert x^2 - y^2 \rvert)) - 0.5}{[1 + 0.001(x^2+y^2)]^2}$
11	Shubert	$f(x,y) = \sum_{i=1}^{5} i \cdot \cos\left((i+1)x + i\right) \cdot \sum_{i=1}^{5} i \cdot \cos\left((i+1)y + i\right)$
12	Bohachevsky	$f(x) = \sum_{i=1}^{n-1}(x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7)$
13	Perm 0	$f(x) = \sum_{i=1}^{n}\left(\sum_{j=0}^{d}(10+j)(x_i - 1)^j\right)^2$
14	Rotated Hyper-Ellipsoid	$f(x) = \sum_{i=1}^{n}\sum_{j=1}^{i} x_j^2$
15	Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$
16	Sum of Different Powers	$f(x) = \sum_{i=1}^{n}\lvert x_i \rvert^{i+1}$
17	Sum Squares	$f(x) = \sum_{i=1}^{n} i \cdot x_i^2$

No.	Function Name	$f(x)$
18	Booth	$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$
19	Matyas	$f(x, y) = 0.26(x^2 + y^2) - 0.48xy$
20	Zakharov	$f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$
21	Three-Hump Camel	$f(x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$

Table 3: Comparative results

No.	Function	EPOIV Mean	EPOW Mean	EPO Classical Mean	EPOWIV Mean	EPOIV Std	EPOW Std	EPO Classical Std	EPOWIV Std
1	Ackley	2.306	0.002	7.600	***0.000***	0.575	0.000	2.358	***0.000***
2	Bukin N. 6	0.619	0.412	1.517	***0.243***	0.385	0.586	0.725	***0.000***
3	Cross-in-Tray	***-2.063***	***-2.063***	***-2.063***	***-2.063***	0.000	0.000	0.000	***0.000***
4	Drop-Wave	-0.994	***-1.000***	-1.004	***-1.000***	0.019	***0.000***	0.001	***0.000***
5	Griewank	1.368	0.513	33.537	***0.000***	0.242	0.490	17.253	***0.000***
6	Langermann	***-4.144***	-4.136	-4.134	-4.131	0.013	0.023	0.019	***0.000***
7	Levy	***0.015***	0.346	0.487	0.183	0.029	0.151	0.158	***0.000***
8	Rastrigin	2.219	3.159	18.406	***0.000***	1.600	5.926	11.326	***0.000***
9	Schaffer N. 2	***0.000***	0.001	0.004	***0.000***	0.000	0.003	0.005	***0.000***
10	Schaffer N. 4	0.293	***0.293***	0.296	***0.293***	0.001	0.001	0.004	***0.001***
11	Shubert	***-186.727***	-185.665	-186.039	-186.632	0.008	1.136	0.598	***0.000***
12	Bohachevsky	0.006	0.058	0.380	***0.000***	0.012	0.120	0.228	***0.000***
13	Perm 0	***0.03***	214.063	0.09	0.08	***0.000***	114.381	12163.566	***0.000***
14	Rotated Hyper-Ellipsoid	21.166	0.255	1258.814	***0.000***	9.007	0.107	929.035	***0.000***
15	Sphere	***0.000***	***0.000***	0.013	***0.000***	0.000	0.000	0.008	***0.000***
16	Sum of Different Powers	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***
17	Sum Squares	0.012	***0.000***	0.001	***0.000***	0.015	0.000	0.878	***0.000***
18	Booth	***0.000***	0.003	0.001	***0.000***	0.000	0.004	0.001	***0.000***
19	Matyas	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***
20	Zakharov	0.631	0.003	10.052	***0.000***	0.803	0.002	3.693	***0.000***
21	Three-Hump Camel	***0.000***	***0.000***	***0.000***	***0.000***	0.000	0.000	0.000	***0.000***

7 Pressure vessel design

In this section, the pressure vessel design, a real-world problem, is solved using EPOWIV. This problem is proposed by Kannan and Kramer [22] to minimize the fabrication cost. Figure 2 shows the isometric view of the pressure vessel. There are four variables needed to solve this problem, which are the thickness of the shell (T_s), the thickness of the head (T_h), the inner radius (R), and the length of the cylindrical part (L). The mathematical model of this problem is as follows:

$$Consider\ Z = [z_1, z_2, z_3, z_4] = [T_s, T_h, R, L]$$
$$Min\ f(Z) = 0.6224\, z_1 z_3 z_4 + 1.7781 z_2 {z_3}^2 + 3.1661 {z_1}^2 z_4 + 19.84 {z_1}^2 z_3 \quad (9)$$

Subject to:

$$g_1 = -z_1 + 0.0193 z_3 \le 0 \quad (10)$$
$$g_2 = -z_3 + 0.000953 z_3 \le 0 \quad (11)$$
$$g_3 = -\pi {z_3}^2 z_4 - \frac{4}{3}\pi {z_3}^3 + 1{,}296{,}000 \le 0 \quad (12)$$
$$g_4 = z_4 - 240 \le 0 \quad (13)$$
$$i \times 0.0625 \le z_1, z_2 \le 99 \times 0.0625, \forall i = 1, 2, \dots, 99 \quad (14)$$
$$10 \le z_3, z_4 \le 200 \quad (15)$$

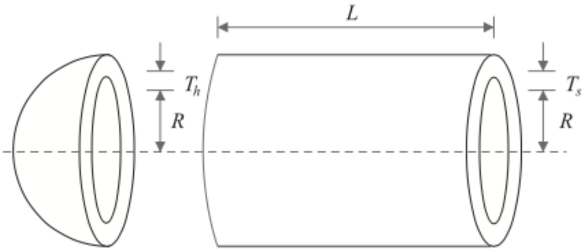

Figure 2: The pressure vessel

The problem has been solved in [23], but we found after checking the values of their solution vector that the solution doesn't satisfy the problem constraints. Their solution is $Z^* = [0.778099, 0.383241, 40.315121, 200]$

and It found that z_1 and z_2 are not integer multipliers of 0.0625. Constraint (12) is broken, since the right-hand side of the constraint according to their solution vector is equal to 319.75, while it should be greater than 0.

We coded the problem constraints and objective, then adapted the proposed EPOWIV to solve the pressure vessel design problem. The code can be found in https://github.com/ahmedsssssA/EPOWIV/blob/main/EPO_PVD. After solving the problem, we found that best solution found by the proposed EPOWIV algorithm is 3320.58 with $Z^* = [1.3125,\ 0.0625,\ 65.7733789, 10]$. This solution satisfies all the problem constraints and shows better objective value than the infeasible solution found by [23]. The statistical results can be summarized as follows:

Best	Mean	Worst	Std. Dev.	Median
3289.4	3585.11	5466.4	523.4	3292.58

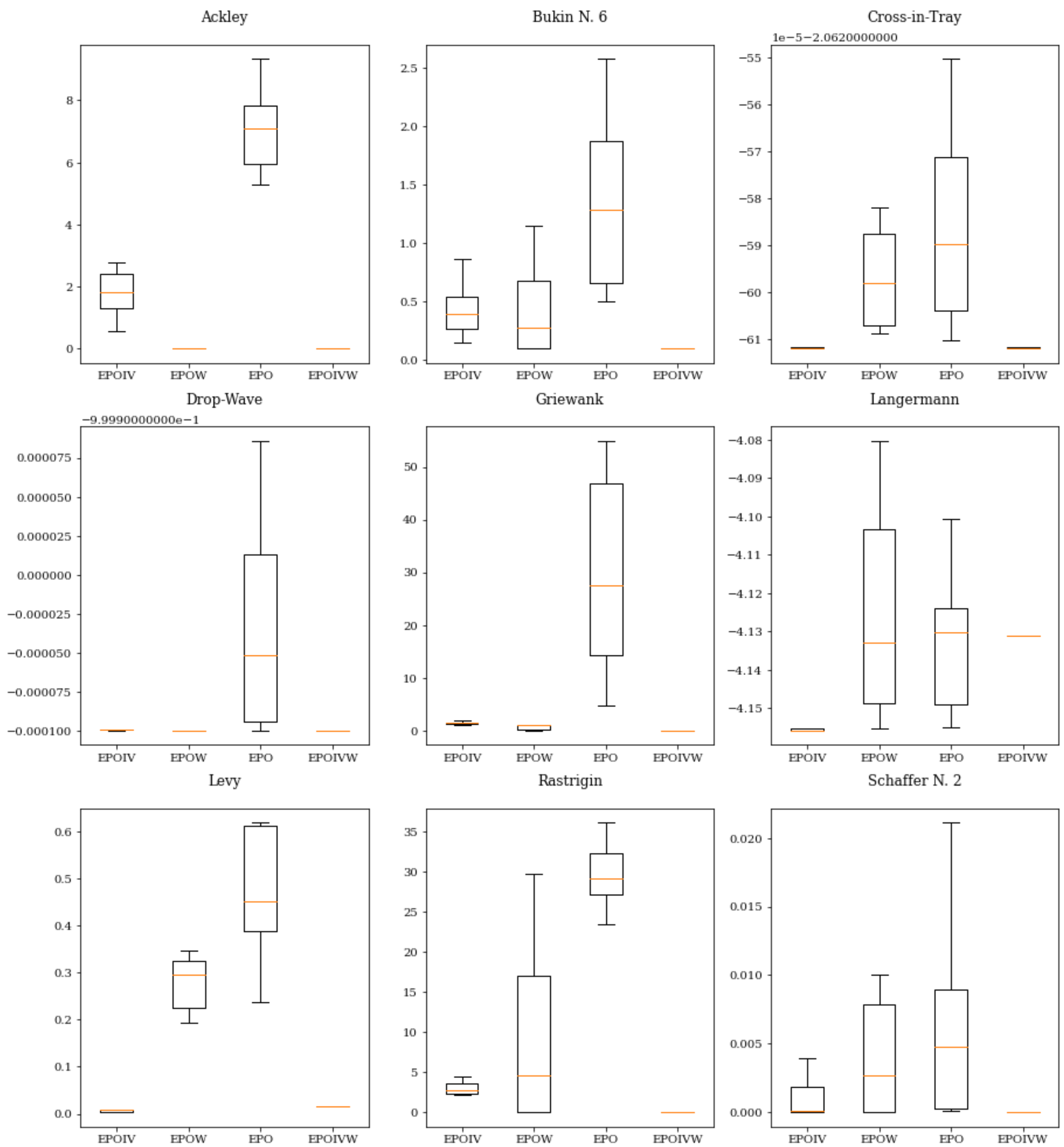

Figure 1: The boxplots of test optimization functions from 1 to 9

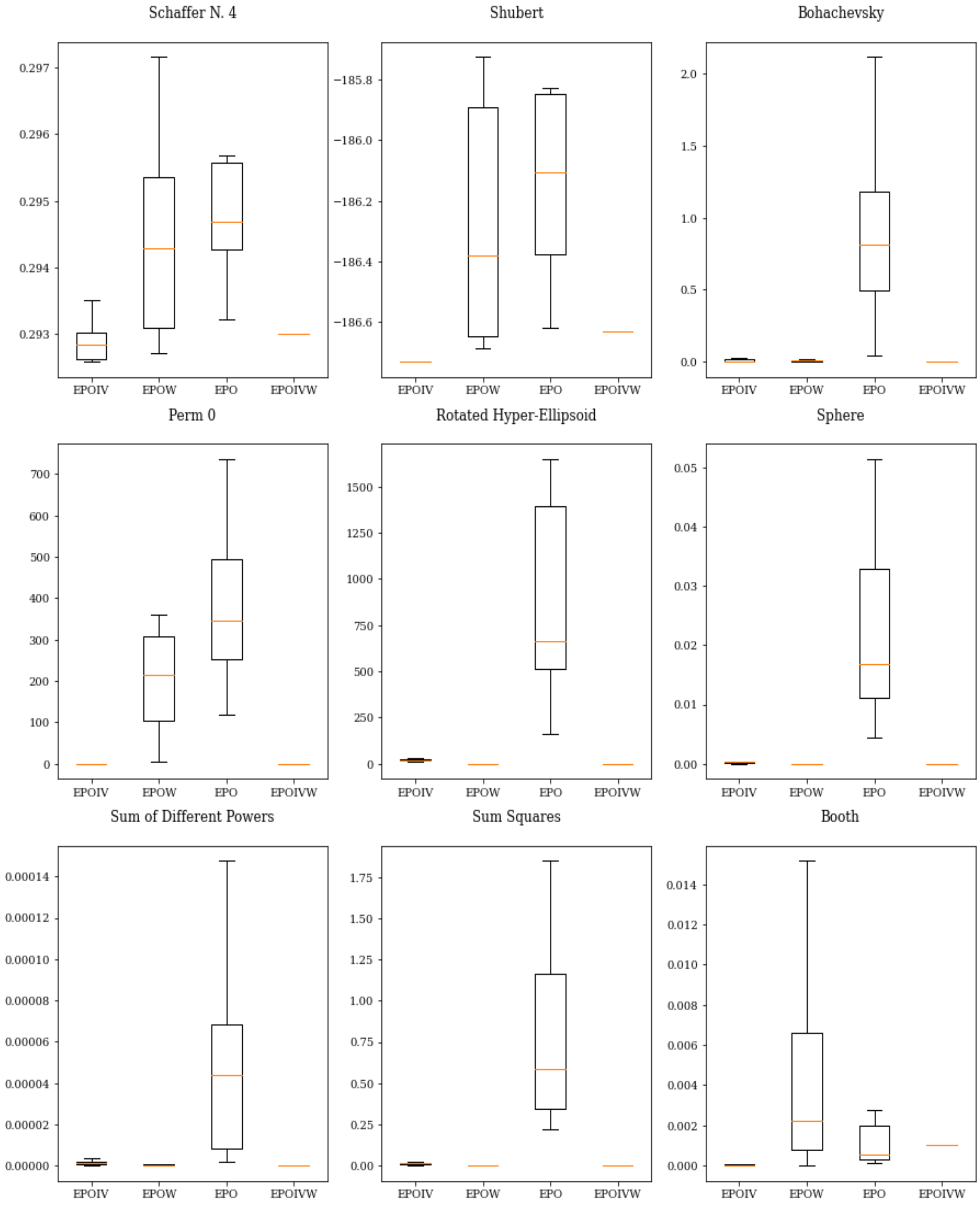

Figure 2: The boxplots of test optimization functions from 10 to 18

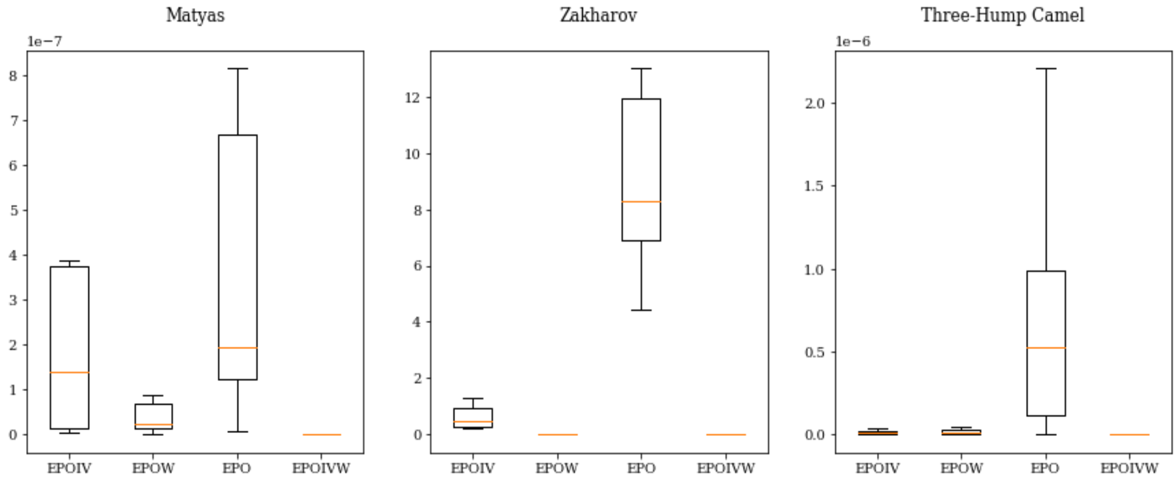

Figure 3 The boxplots of test optimization functions from 19 to 21

8 Time complexity

The population of the algorithm requires $O(PopSize \times d)$, where d is the problem dimension. The calculation of the fitness values requires $O(MaxItr \times PopSize \times d)$. The relocating procedure of the algorithm requires $O(N)$. Therefore, the total complexity of the algorithm requires $O(MaxItr \times PopSize \times d \times N)$. The space complexity requires $O(PopSize \times d)$, which represents the amount of space required at any time during running time of the algorithm.

9 Conclusion

In conclusion, this paper introduced the Emperor Penguin Optimizer with Weighted Sum Procedure and Information Vector, a novel modification of the Emperor Penguin Optimizer algorithm. Through a comprehensive comparative study, we demonstrated the significant enhancement in optimization capabilities achieved by EPOWIV when compared to the classical EPO algorithm, EPO with the weighted sum procedure only, and EPO with the information vector. Across 27 diverse test optimization problems, EPOWIV consistently outperformed its counterparts, showcasing its efficiency and effectiveness in exploring and exploiting complex search spaces. The success of EPOWIV can be attributed to the synergistic combination of the weighted sum procedure and the information vector, which empowers the algorithm to navigate complex landscapes and converge to superior solutions. These results underscore the potential of EPOWIV as a valuable tool for solving optimization problems in various domains. The future points of research may include:

- Parameter Tuning and Sensitivity Analysis: Further research can explore the sensitivity of EPOWIV to its hyperparameters and investigate methods for automated parameter tuning to adapt to different problem types and complexities.
- Real-World Applications: Apply EPOWIV to real-world applications and case studies across diverse fields such as engineering, finance, and healthcare to assess its performance in practical settings.

Hybridization: Explore opportunities for hybridizing EPOWIV with other optimization algorithms to create more robust and versatile optimization frameworks.

References

[1] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm Evol. Comput.*, vol. 33, pp. 1–17, 2017, doi: 10.1016/j.swevo.2016.12.005.

[2] X. S. Yang, "Firefly algorithms for multimodal optimization," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5792 LNCS, pp. 169–178, 2009, doi: 10.1007/978-3-642-04944-6_14.

[3] O. W. Khalid, N. A. M. Isa, and H. A. Mat Sakim, "Emperor penguin optimizer: A comprehensive review based on state-of-the-art meta-heuristic algorithms," *Alexandria Eng. J.*, vol. 63, pp. 487–526, 2023, doi: 10.1016/j.aej.2022.08.013.

[4] A. Grasas, A. A. Juan, and H. R. Lourenço, "SimILS: A simulation-based extension of the iterated local search metaheuristic for stochastic combinatorial optimization," *J. Simul.*, vol. 10, no. 1, pp. 69–77, 2016, doi: 10.1057/jos.2014.25.

[5] A. A. Juan *et al.*, "A review of the role of heuristics in stochastic optimisation: from metaheuristics to learnheuristics," *Ann. Oper. Res.*, vol. 320, no. 2, pp. 831–861, 2023, doi: 10.1007/s10479-021-04142-9.

[6] A. Hertz and M. Widmer, "Guidelines for the use of meta-heuristics in combinatorial optimization," *Eur. J. Oper. Res.*, vol. 151, no. 2, pp. 247–252, 2003, doi: 10.1016/S0377-2217(02)00823-8.

[7] J. Crispim and J. Brandão, “Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls,” *J. Oper. Res. Soc.*, vol. 56, no. 11, pp. 1296–1302, 2005, doi: 10.1057/palgrave.jors.2601935.

[8] R. Elshaer and H. Awad, “A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants,” *Comput. Ind. Eng.*, vol. 140, 2020, doi: 10.1016/j.cie.2019.106242.

[9] G. Jaradat, M. Ayob, and I. Almarashdeh, “The effect of elite pool in hybrid population-based meta-heuristics for solving combinatorial optimization problems,” *Appl. Soft Comput. J.*, vol. 44, pp. 45–56, 2016, doi: 10.1016/j.asoc.2016.01.002.

[10] J. F. Goycoolea, M. Inostroza-Ponta, M. Villalobos-Cid, and M. Marin, “Single-solution based metaheuristic approach to a novel restricted clustering problem,” *Proc. - Int. Conf. Chil. Comput. Sci. Soc. SCCC*, vol. 2021-Novem, 2021, doi: 10.1109/SCCC54552.2021.9650429.

[11] G. Dhiman and V. Kumar, “Emperor penguin optimizer: A bio-inspired algorithm for engineering problems,” *Knowledge-Based Syst.*, vol. 159, pp. 20–50, 2018, doi: 10.1016/j.knosys.2018.06.001.

[12] S. K. Baliarsingh and S. Vipsita, “Chaotic emperor penguin optimised extreme learning machine for microarray cancer classification,” *IET Syst. Biol.*, vol. 14, no. 2, pp. 85–95, 2020, doi: 10.1049/iet-syb.2019.0028.

[13] Y. Cao, Y. Wu, L. Fu, K. Jermsittiparsert, and N. Razmjooy, “Multi-objective optimization of a PEMFC based CCHP system by meta-heuristics,” *Energy Reports*, vol. 5, pp. 1551–1559, 2019, doi: 10.1016/j.egyr.2019.10.029.

[14] S. Min, Z. Tang, and B. Daneshvar Rouyendegh, “Inspired-based optimisation algorithm for solving energy-consuming reduction of chiller loading,” *Int. J. Ambient Energy*, vol. 43, no. 1, pp. 2313–2323, 2022, doi: 10.1080/01430750.2020.1730954.

[15] M. A. Angel and T. Jaya, “An Enhanced Emperor Penguin Optimization Algorithm for Secure Energy Efficient Load Balancing in Wireless Sensor Networks,” *Wirel. Pers. Commun.*, vol. 125, no. 3, pp. 2101–2127, 2022, doi: 10.1007/s11277-022-09647-5.

[16] Z. Xing, “An improved emperor penguin optimization based multilevel thresholding for color image segmentation,” *Knowledge-Based Syst.*, vol. 194, 2020, doi: 10.1016/j.knosys.2020.105570.

[17] G. Dhiman *et al.*, “BEPO: A novel binary emperor penguin optimizer for automatic feature selection,” *Knowledge-Based Syst.*, vol. 211, p. 106560, 2021, doi: 10.1016/j.knosys.2020.106560.

[18] A. I. Khan, A. S. Almalaise Alghamdi, Y. B. Abushark, F. Alsolami, A. Almalawi, and A. Marish Ali, “Recycling waste classification using emperor penguin optimizer with deep learning model for bioenergy production,” *Chemosphere*, vol. 307, 2022, doi: 10.1016/j.chemosphere.2022.136044.

[19] K. Cheena, T. Amgoth, and G. Shankar, “Emperor penguin optimised self-healing strategy for WSN based smart grids,” *Int. J. Sens. Networks*, vol. 32, no. 2, pp. 87–95, 2020, doi: 10.1504/IJSNET.2020.104924.

[20] N. R. Babu, T. Chiranjeevi, R. Devarapalli, and S. K. Bhagat, “Optimal location of FACTs devices in LFC studies considering the application of RT-Lab studies and emperor penguin optimization algorithm,” *J. Eng. Res.*, vol. 11, no. 2, 2023, doi: 10.1016/j.jer.2023.100060.

[21] A. Serag, H. Zaher, N. Ragaa, and H. Sayed, “A Modified Emperor Penguin Algorithm for Solving Stagnation in Multi-Model Functions,” *Inform.*, vol. 47, no. 10, pp. 71–78, 2023, doi: 10.31449/inf.v47i10.5273.

[22] B. K. Kannan and S. N. Kramer, “An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design,” *Proc. ASME Des. Eng. Tech. Conf.*, vol. Part F1679, pp. 103–112, 1993, doi: 10.1115/DETC1993-0382.

[23] G. Dhiman *et al.*, “BEPO: A novel binary emperor penguin optimizer for automatic feature selection,” *Knowledge-Based Syst.*, vol. 211, 2021, doi: 10.1016/j.knosys.2020.106560.