processes

MDPI

Article

Diversity-Based Evolutionary Population Dynamics: A New Operator for Grey Wolf Optimizer

Farshad Rezaei [1], Hamid R. Safavi [1], Mohamed Abd Elaziz [2,3,4], Laith Abualigah [5,6,7,8,9,*], Seyedali Mirjalili [10,11] and Amir H. Gandomi [12,13,*]

[1] Department of Civil Engineering, Isfahan University of Technology, Isfahan 8415683111, Iran
[2] Department of Mathematics, Faculty of Science, Zagazig University, Zagazig 44519, Egypt
[3] Faculty of Computer Science and Engineering, Galala University, Suez 435611, Egypt
[4] Artificial Intelligence Research Center (AIRC), Ajman University, Ajman P.O. Box 346, United Arab Emirates
[5] Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan
[6] Prince Hussein Bin Abdullah College for Information Technology, Al Al-Bayt University, Mafraq 130040, Jordan
[7] Faculty of Information Technology, Middle East University, Amman 11831, Jordan
[8] Faculty of Information Technology, Applied Science Private University, Amman 11931, Jordan
[9] School of Computer Sciences, Universiti Sains Malaysia, Pulau Pinang 11800, Malaysia
[10] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, Brisbane, QLD 4006, Australia
[11] YFL (Yonsei Frontier Lab), Yonsei University, Seoul 03722, Republic of Korea
[12] Faculty of Engineering & Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia
[13] University Research and Innovation Center (EKIK), Óbuda University, 1034 Budapest, Hungary
* Correspondence: aligah.2020@gmail.com (L.A.); gandomi@uts.edu.au (A.H.G.)

Abstract: Evolutionary Population Dynamics (EPD) refers to eliminating poor individuals in nature, which is the opposite of survival of the fittest. Although this method can improve the median of the whole population of the meta-heuristic algorithms, it suffers from poor exploration capability to handle high-dimensional problems. This paper proposes a novel EPD operator to improve the search process. In other words, as the primary EPD mainly improves the fitness of the worst individuals in the population, and hence we name it the Fitness-Based EPD (FB-EPD), our proposed EPD mainly improves the diversity of the best individuals, and hence we name it the Diversity-Based EPD (DB-EPD). The proposed method is applied to the Grey Wolf Optimizer (GWO) and named DB-GWO-EPD. In this algorithm, the three most diversified individuals are first identified at each iteration, and then half of the best-fitted individuals are forced to be eliminated and repositioned around these diversified agents with equal probability. This process can free the merged best individuals located in a closed populated region and transfer them to the diversified and, thus, less-densely populated regions in the search space. This approach is frequently employed to make the search agents explore the whole search space. The proposed DB-GWO-EPD is tested on 13 high-dimensional and shifted classical benchmark functions as well as 29 test problems included in the CEC2017 test suite, and four constrained engineering problems. The results obtained by the proposal upon implemented on the classical test problems are compared to GWO, FB-GWO-EPD, and four other popular and newly proposed optimization algorithms, including Aquila Optimizer (AO), Flow Direction Algorithm (FDA), Arithmetic Optimization Algorithm (AOA), and Gradient-based Optimizer (GBO). The experiments demonstrate the significant superiority of the proposed algorithm when applied to a majority of the test functions, recommending the application of the proposed EPD operator to any other meta-heuristic whenever decided to ameliorate their performance.

Keywords: Grey Wolf Optimizer; evolutionary population dynamics; hybrid algorithms; meta-heuristic algorithms; swarm-intelligence techniques

check for updates

Citation: Rezaei, F.; Safavi, H.R.; Abd Elaziz, M.; Abualigah, L.; Mirjalili, S.; Gandomi, A.H. Diversity-Based Evolutionary Population Dynamics: A New Operator for Grey Wolf Optimizer. *Processes* **2022**, *10*, 2615. https://doi.org/10.3390/pr10122615

Academic Editor: Andreas Håkansson

Received: 19 October 2022
Accepted: 28 November 2022
Published: 6 December 2022

1. Introduction

Optimization algorithms in evolutionary computation can be divided into two classes individual-based and population-based. Individual-based algorithms start the optimization by a single randomly generated search agent in the search space attempting to seek and detect the global optimum of the optimization problem. This category of algorithms benefits from some advantages and suffers from some disadvantages. The main shortcoming of these algorithms is premature convergence emerging in two different types of optimization problems, called uni-modal and multi-modal functions. In uni-modal functions, premature convergence usually occurs when the algorithm's convergence rate during the optimization process is too slow.

In contrast, the major cause for premature convergence in multi-modal optimization problems is known as the local optima entrapment. Individual-based algorithms begin the optimization from an initial random point in the search space. Usually, they can only search for the optimal solution in the proximity of that initial random point without escaping from that region to search for other potentially high-fitness regions in the problem domain. On the contrary, the algorithms of type population-based commence the optimization process by generating a series of random solutions being improved throughout iterations. This category of algorithms is less likely to be stuck in the local optima in multi-modal problems and also benefits from a more rapid search process in uni-modal problems. However, having more computational costs and needing much more objective function evaluations are two significant disadvantages; this category suffers compared to the individual-based algorithms.

Among the stochastic population-based optimization algorithms, the Genetic Algorithm (GA) [1], Particle Swarm Optimization (PSO) [2], and Differential Evolution (DE) [3] are the pioneers, based on which a vast number of modified and improved algorithms have been developed so far and applied to a wide range of the practical problems in the science and engineering. There are a lot of nature-inspired and physics-based stochastic optimization algorithms in the literature as well, including but not limited to Artificial Bee Colony (ABC) algorithm [4], Firefly Algorithm (FA) [5], Bat Algorithm (BA) [6], Gravitational Search Algorithm (GSA) [7], Krill Herd (KH) algorithm [8], Sine Cosine Algorithm (SCA) [9], Monarch Butterfly Optimization (MBO) algorithm [10], and Invasive Weed Optimization (IWO) algorithm [11].

1.1. Hybrid Meta-Heuristic Optimization Algorithms

An effective way to enhance the performance of the optimization algorithms is the combination or hybridization of them with different operators and search mechanisms employed in other optimizers. Due to the consolidation of benefits from different algorithms during the hybridization process, many improved versions of the algorithms have been proposed so far. Some of these algorithms are described in the following paragraphs.

Tuba and Bacanin et al. [12] hybridized the ABC with the FA to improve the convergence rate of the ABC and also to well adjust the exploration-exploitation balance in the ABC algorithm. Wang et al. [13] introduced the Pitch adjustment operator employed in HS into the CS. This operator can act as a mutation operator to increase the cuckoo population diversity and compensate for the weakness of the Lévy flight incorporated in the CS, as the Lévy flight can lead the CS not to be able to converge to the best-fitted solutions whenever the size of the steps taken is too large. As a result, this hybridization can impede premature convergence. In order to improve the DE algorithm when attempting to balance local and global search processes while impeding the time-consuming control parameter tuning procedure, a new hybrid DE algorithm is proposed by Yi et al. [14]. This method divides the population into two sections to apply the most appropriate control parameters and mutation operators. Tuba and Bacanin [15] improved the global search procedure of the BA by incorporating the onlooker mechanism from the ABC algorithm to enhance the exploitation process of BA and applied the hybridized algorithm to solve a multi-objective radio frequency identification network planning problem. A hybrid PSO and GSA abbreviated as PSO-GSA was introduced by Das et al. [16], benefiting from a cooperative contribution of PSO velocity and GSA acceleration. In this hybridized algorithm, the PSO uses memory to save the best solutions found so far, and this ability is added to the ability of GSA to adjust the acceleration using the fitness value. Furthermore, in this algorithm, the global best particle of PSO can lead the other particles to move towards the best solution and accomplish exploitation slowly. A hybrid strategy is proposed by Abualigah et al. [17], aimed at increasing the KH's population's diversity by incorporating the HS operator using a new parameter named distance factor for fine-tuning the positions of the search agents before updating them. Nenavath and Jatoth [18] propose the hybridization of the SCA and DE. This hybrid algorithm improves the capability of the search agents to escape from the local optima and enhances the convergence speed compared to SCA and DE when operating separately. The proposed algorithm was then employed to effectively solve an object tracking problem as a real-life optimization problem. Ghanem proposes a new hybrid approach, and Jantan [19] to alter the butterfly tuning operator used in MBO while utilizing this improved operator in the role of a mutation operator to be applied as an alternative to the employee stage of the ABC. Another hybridization process has been carried out between the IWO algorithm and the FA. As any hybridization scheme aims at highlighting the best properties of a couple of the other algorithms to compensate for their drawbacks, the hybrid IWO-FA is proposed by Panda et al. [20] to improve both algorithms. As there is no known mechanism to hinder the fast movements of the search agents in IWO, the FA can also face local optima entrapments due to its dependence on the light intensity for attraction, and the IWO-FA can be taken into account as an approach to ameliorate these shortcomings simultaneously. Teng et al. [21] benefited from the high diversification ability of the GWO and high intensification ability of the PSO in local search to propose the hybrid PSO_GWO algorithm. In this algorithm, the best individual position experienced is added to the best wolves to guide the search agents in the population. The Tent chaotic sequence was utilized better to diversify the wolf pack in the algorithm initialization step. The use of the entire MBO algorithm while not using the Lévy flight that typically acts as a crossover operator for DE, proposed by Ibrahim and Tawhid [22], is another hybridized scheme named DEMBO. A modified version of the KH was proposed by Abualigah et al. [23], in which the genetic operators are employed to improve the KH performance highly. In this modified algorithm, the crossover and mutation processes are invoked after updating the krill positions because the nature of the search space of most optimization problems is ragged and deep. A hybrid method combining the SCA and ABC, called SCABC, is proposed by Gupta and Deep [24]. In the SCABC, first, the major problem of the SCA is the intensive diversification of the solutions in the early iterations while leaving the solutions less diversified in the later iterations. Then, this problem was mitigated by modifying its search procedures by integrating memory-based information. Then, the modified SCA search procedures are merged to be applied to the employed bee phase to enhance the global and local search abilities of the ABC algorithm. A new algorithm hybridizing the PSO and GWO is also proposed by Şenel et al. [21]. The hybridized scheme can replace some particles in PSO with a small probability by some other improved search agents obtained by GWO. The proposed mechanism can hamper the PSO from being trapped in local optima utilizing the exploration capability of GWO. Gupta and Deep [25] hybridized the Lévy-flight mechanism and the original GWO, yielding GLF-GWO. The Lévy-flight strategy helps better local search around the three leading wolves in GWO to enable the wolves to avoid local optima. Furthermore, a greedy selection mechanism is employed to prevent the wolves from diverging from the promising solutions during the optimization. A new hybrid HSCA is proposed by Gupta and Deep [26], mainly to impede the solutions of the SCA to skip the fitted regions of the decision space, to prevent the solutions from being trapped in local optima, and to balance the exploration-exploitation transition in the classical SCA more desirably. In the HSCA, the SCA and simulated quenching algorithm are hybridized to impart the elitism mechanism to the algorithm while easing the transition from exploration to exploitation. Mohammed and Rashid [27] hybridized the WOA with GWO to enhance the exploitation capability of the WOA. They found the exploitation capability of the GWO much stronger than the mechanism dedicated to exploitation in the WOA. That was their motivation to substitute the two mechanisms applied in two algorithms. A hybridization of SMA and AOA was proposed by Zheng et al. [28]. This hybrid algorithm replaces the contraction formula employed to perform exploration in the SMA is replaced by the multiplication and division operators used in the exploration phase of the AOA, as the position of each search agent may tend to zero in the late iterations, which, in, turn, can highly debilitate the performance of the SMA when solving an optimization problem with an unknown search space. Rezaei et al. [29] added the velocity term existing in the position updating formula of the PSO or some other meta-heuristics into the updating procedure of the GWO and proposed the VAGWO algorithm. This hybridized scheme can highly improve the exploration capability of the GWO by keeping push the search agents to move forwards to detect the high-fitness positions as much as possible. In addition, a decreasing radius circle is made around the elite agents to impede the potential drifts to occur when the agents are tending to the high-fitness positions. A hybridized Whale and Moth-Flame Optimizer, named WMFO, was proposed by Nadimi-Shahraki et al. [30]. The goal of this hybridization is enhancing the exploitation capability of the MFO via enforcing the flames in the MFO to move towards an average of the best-so-far agents found by the algorithm in the exploitation phase of the hybrid algorithm to expedite the agents' access to the high-fitness region in the search space.

No Free Lunch (NFL) theorem [31] allows new evolutionary algorithms to be proposed, as based on this theory, all algorithms have the same performance when applied to a vast range of optimization problems. As a result, an algorithm may show too effective performance when solving a series of problems but suffers from serious shortcomings when going over solving other problems. Consequently, creating new effective algorithms via hybridizing them with the other ones or equipping them with some efficient operators can highly improve the performance of such algorithms to better solve a series of optimization problems the other optimizers have difficulty handling.

Selection, combination, and mutation are usually considered the widely used evolutionary operators incorporated into the meta-heuristics to enhance their performance, as illustrated by Lewis et al. [32]. However, another evolutionary operator, Evolutionary Population Dynamics (EPD), has been found to manipulate the whole population. EPD is inspired by the theory of Self-Organizing Critically (SOC), first introduced by [33]. Based on this theory, the critical state in nature can be reached dynamically. For instance, small perturbations can make operations precisely balance different features of the population while not imposing any external force on them. Extremal Optimization (EO), proposed by Boettcher and Percus [34], is a meta-heuristic inspired by the self-organizing critical model utilizing EPD. In EO, the worst solutions in the population are eliminated and repositioned around the best ones. This process can evolve the worst solutions found during the optimization course, in contrast to the typical process in the Genetic Algorithm (GA), where the best solutions are always combined and evolved during the optimization.

1.2. The Contribution of This Study

In this paper, we propose a novel EPD operator addressing the other side of the goodness of an individual in the population of the EPD-based evolutionary algorithms. In the original version of EPD, the worst individuals are considered low-fitness individuals, required to be improved by removing and repositioning them around the best (high-fitness) individuals. However, the other side of the goodness/fitness of an individual can also be interpreted as the diversity of that individual in the search space. Having this another side for fitness, we can evolve and improve the best (high-fitness) individuals by eliminating a number of them and then repositioning them around the most diversified individuals in the search space. In this way, the best individuals are getting better, not in their fitness, but in their diversity as the other aspect of fitness that is very important to address, especially in solving complex and large-scale optimization problems. Our novel EPD-based mechanism, which we name Diversity-Based EPD (DB-EPD), is applied to the Grey Wolf Optimizer

(GWO) in this paper, and the results are compared to the original GWO and the GWO-EPD proposed by Saremi et al. [35], which is addressing the application of the traditional EPD, which we name the Fitness-Based EPD (FB-EPD), to improve the GWO.

The organization of the rest of this paper is as follows. Section 2 presents the methodology, in which we first introduce the theory of GWO and then formulate the FB-EPD for GWO. Then we introduce our novel DB-EPD mechanism and its application to improve the GWO in Section 3. Section 4 presents the results and some analyses of them. Section 5 concludes the paper and summarizes the main contributions of this paper.

2. Methodology

This section introduces the traditional grey wolf optimization and Fitness-Based EPD for GWO.

2.1. Original Grey Wolf Optimization (GWO) Algorithm

The GWO algorithm was proposed by Mirjalili et al. in 2014 [36]. This algorithm imitates the hunting behavior and hierarchical leadership of grey wolves in nature. The GWO initializes the optimization process by randomly producing a population of the solutions (wolves). At each iteration, the three best-fitted wolves, named alpha, beta, and delta, are identified as the guide for the other wolves named omega. Afterward, the omega wolves encircle their guides to find the high-fitness regions in the search space. The wolves are named the search agents in the concept of swarm-intelligence optimization techniques. As every omega wolf encircles the three best-fitted agents, its final position can be obtained by simply averaging the updated positions of the alpha, beta, and delta wolves. This procedure may boost the exploratory ability of the algorithm due to adopting more than one search agent as the guide to the other agents. The mathematical formulation of updating the omega wolves is as follows:

$$\vec{D} = \left|\vec{C}.\overrightarrow{X_p(t)} - \overrightarrow{X(t)}\right| \tag{1}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X_p(t)} - \vec{A}.\vec{D} \tag{2}$$

where t is the current iteration; $\vec{A} = 2\vec{r_1}.\vec{a} - \vec{a}$; $\vec{C} = 2\vec{r_2}$; $\overrightarrow{X_p(t)}$ is the position vector of the prey; $\overrightarrow{X(t)}$ is the position vector of a grey wolf. Moreover, $\vec{r_1}$ and $\vec{r_2}$ are two random vectors in [0, 1]. Furthermore, $\vec{a}$ is linearly decreased from 2 to 0 by lapse of iterations. The factor $\vec{A}$ facilitates the optimization process by controlling a safe and reliable transition from the exploration to the exploitation phases of the algorithm. Moreover, $\vec{C}$ is multiplied by the guiding vector to help the exploration process more precisely.

The position of each omega wolf is updated through several equations formulated below:

$$\vec{D_\alpha} = \left|\vec{C_1}.\vec{X_\alpha} - \vec{X}\right| \tag{3}$$

$$\vec{D_\beta} = \left|\vec{C_2}.\vec{X_\beta} - \vec{X}\right| \tag{4}$$

$$\vec{D_\delta} = \left|\vec{C_3}.\vec{X_\delta} - \vec{X}\right| \tag{5}$$

$$\vec{X_1} = \vec{X_\alpha} - \vec{A_1}.\vec{D_\alpha} \tag{6}$$

$$\vec{X_2} = \vec{X_\beta} - \vec{A_2}.\vec{D_\beta} \tag{7}$$

$$\vec{X_3} = \vec{X_\delta} - \vec{A_3}.\vec{D_\delta} \tag{8}$$

$$\overrightarrow{X(t+1)} = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{9}$$

As Mirjalili et al. (2014) expressed, in GWO, the first half of the iterations are assigned to the exploration process when $\left|\vec{A}\right| > 1.$, and the second half is allocated to the exploitation phase of the optimization process, when $\left|\vec{A}\right| < 1$.

2.2. Fitness-Based EPD for GWO (FB-GWO-EPD)

In this approach proposed by Saremi et al. [35], half of the worst search agents are detected and eliminated from the population. Then, the omitted agents are re-initialized around the best agents in terms of fitness. This process evolves the low-fitness solutions instead of the best-fitted ones. The procedure of repositioning the worst agents around the four random positions with equal probability can be implemented via Equations (10)–(13).

$$\overrightarrow{X(t+1)} = \overrightarrow{X_\alpha(t)} \pm \left(\left(\vec{ub} - \vec{lb}\right).\vec{r_1} + \vec{lb}\right);\ \text{if } 0 \leq r_5 \leq 1/4 \tag{10}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X_\beta(t)} \pm \left(\left(\vec{ub} - \vec{lb}\right).\vec{r_2} + \vec{lb}\right);\ \text{if } 1/4 < r_5 \leq 1/2 \tag{11}$$

$$\overrightarrow{X(t+1)} = \overrightarrow{X_\delta(t)} \pm \left(\left(\vec{ub} - \vec{lb}\right).\vec{r_3} + \vec{lb}\right);\ \text{if } 1/2 < r_5 \leq 3/4 \tag{12}$$

$$\overrightarrow{X(t+1)} = \left(\left(\vec{ub} - \vec{lb}\right).\vec{r_4} + \vec{lb}\right);\ \text{if } 3/4 < r_5 \leq 1 \tag{13}$$

where, $\overrightarrow{X_\alpha(t)}$, $\overrightarrow{X_\beta(t)}$ and $\overrightarrow{X_\delta(t)}$ are the alpha, beta, and delta wolf positions; $\vec{ub}$ indicates the upper bound vector of the search space; $\vec{lb}$ denotes the lower bound vector of the search space, and $\vec{r_1}$ to $\vec{r_5}$ are five uniformly distributed random vectors in [0, 1]. Equations (10)–(12) formulate the repositioning process of the worst solutions around the best ones, and Equation (13) formulates repositioning the worst solutions around randomly positions around the search space to maintain the diversity among the diversified worst solutions to promote the exploration and avoid missing a large number of suitable solutions when the algorithm approaches the leading solutions.

3. Proposed Method

As discussed in the previous subsection, the fitness-based GWO-EPD aims at eliminating the worst (low-fitness) individuals instead of evolving the best ones. This process is well done when reproducing the worst individuals around the three elite ones in GWO: alpha, beta, and delta. However, repositioning the low-fitness individuals around the high-fitness areas in the search space can, in turn, intensify the risk of engagement in premature convergence. In other words, the low-fitness individuals may benefit from being located in the high-diversified positions in the search space, in contrast to those in the low-diversified regions in the search space. The process of repositioning the worst individuals around the best ones can further exacerbate the diversity of the solutions in the search space when knowing that the best (high-fitness) individuals are being rapidly gathered in a closed region, leading the search space to lose its diversity. Since population diversification is a key phase in any stochastic population-based optimization algorithm, such as the GWO, there is an urgent need to diversify the solutions produced by the algorithm.

In the proposed diversity-based evolutionary population dynamics for GWO, not only the best-fitted solutions are repositioned around the highly diversified areas in the search space, but also all the solutions continue to converge to the three (alpha, beta, and delta) optimal solutions found at each iteration. This way, the solutions are frequently gathered inside the high-fitness region and dispersed outside of this region so that the solutions can be circulated across the search space. In this process, the poor-fitted solutions move to the best-fitted ones, on the one hand, and the best-fitted solutions are moved to the diversified areas, on the other hand. This process can better maintain the balance between the exploration capability of the GWO, in which the solutions are to be diversified to fully cover the search space, and the exploitation capability, in which the solutions must intensify, moving to the best regions in the search space to converge to the optimum of the optimization problem finally.

In the proposed diversity-based version of the GWO-EPD algorithm, we first eliminate half of the best individuals in terms of fitness and then reposition them around the three highly diversified solutions identified in the solution space. It is noteworthy that the alpha, beta, and delta agents are maintained in the algorithm's memory to guide search agents. However, they are repositioned around the highly diversified agents if involved in the current population. The diversity of the solutions can be numerically calculated through Equation (14).

$$d_i \ = \min_j \left| f^i(X) \ - \ f^j(X) \right| \tag{14}$$

where $i \ = \ 1, \ 2, \ \ldots \ , \ N; \ j \ \in \ \{1, \ \ldots, i \ - \ 1, \ i+1, \ \ldots, \ N\}; \ N$ is the population size; $f^i(X)$ = objective (fitness) function value for the ith solution, and $f^j(X)$ = objective (fitness) function value for the jth solution. In addition, d_i is the diversity index of the ith solution. As can be seen in Equation (14), the differences between the fitness value of each solution from the fitness values of the other solutions in the population are all calculated, and then, the minimum of these differences is taken as the intersection of them to result in the d_i index. On the one hand, the differences in the fitness value of a solution from the other solutions' can properly reflect how different that solution is from the other solutions and so how diversified it is across the search space. On the other hand, the higher the intersection of several values, the higher all of these values will be, as their intersection can always be regarded as those values in common. Hence, the more the d_i value, the higher the diversity of the ith search agent will be. The three individuals of the swarm denoted by the three highest values of d_i are then identified as the most diversified individuals and considered to guide half of the best-fitted individuals in the search space. Suppose that the three most diversified individuals at the tth iteration are denoted by $\overrightarrow{X_\alpha^{div}(t)}$, $\overrightarrow{X_\beta^{div}(t)}$, and $\overrightarrow{X_\delta^{div}(t)}$. Then, half of the best-fitted solutions are to be repositioned around these solutions with equal probability based on Equations (15)–(17), as follows:

$$\overrightarrow{X(t+1)} \ = \ \overrightarrow{X_\alpha^{div}(t)} \pm \left(\left(\overrightarrow{ub} - \overrightarrow{lb} \right) . \overrightarrow{r_1} + \overrightarrow{lb} \right); \text{ if } 0 \ \leq \ r_4 \ \leq \ 1/3 \tag{15}$$

$$\overrightarrow{X(t+1)} \ = \overrightarrow{X_\beta^{div}(t)} \pm \left(\left(\overrightarrow{ub} - \overrightarrow{lb} \right) . \overrightarrow{r_2} \ + \overrightarrow{lb} \right); \text{ if } 1/3 \ < \ r_4 \ \leq \ 2/3 \tag{16}$$

$$\overrightarrow{X(t+1)} \ = \ \overrightarrow{X_\delta^{div}(t)} \pm \left(\left(\overrightarrow{ub} - \overrightarrow{lb} \right) \overrightarrow{r_3} \ + \overrightarrow{lb} \right); \text{ if } 2/3 \ < \ r_4 \ \leq \ 1 \tag{17}$$

where, $\overrightarrow{ub}$ is the upper bound vector of the search space, $\overrightarrow{lb}$ denotes the lower bound vector of the search space, and $\overrightarrow{r_1}$ to $\overrightarrow{r_3}$ are three uniformly distributed random vectors in [0, 1]. r_4 is also a random number in [0, 1], delineating which equation to be used to generate $X(t+1)$ as the new position of a solution included in the first half of the best-fitted solutions. In generating $X(t+1)$, the Equations (15)–(17) are to be used. The random number r_4 is first generated for each solution e.g. $\overrightarrow{X(t)}$ that is included in the first half of the best-fitted solutions, and then only one of these three equations is utilized to reposition that solution. Each equation is composed of two terms, including one of the three most-diversified solutions in the search space as the first term and a randomly generated solution in the search space as the second term. Note that the probability of choosing one of these three equations to reposition the focused solution is just the same as the r_4 is a uniformly distributed random number and has just the same probability of being chosen when its ranges are equal in length, just as considered in Equations (15)–(17). It is noticeable that in the proposed DB-GWO-EPD algorithm, the critical factor $\vec{a}$, introduced in Section 2.1, is linearly decreased from 2 to 0, such as the original GWO algorithm. As the last modification applied to the proposed approach, the factor $\vec{C}$ is eliminated. It is worth noticing that the factor $\vec{C}$ is used in the updating formulae in the original GWO algorithm to multiply the positions of the prey (alpha, beta, or delta wolves) and redirect the search agents to the random positions away from the accurate positions of the prey. This can help the GWO not to consider the fitness of the prey positions so certain and precise that it can, in turn, contribute to the search agents getting more diversified across the search space and avoiding local optima entrapments. The elimination of such a factor in the proposed DB-GWO-EPD algorithm is because of the well-maintaining of diversity in this algorithm, not necessitating the algorithm to take random positions around the alpha, beta, and delta wolves at each step of the optimization process, unlike the original GWO and the fitness-based GWO-EPD (FB-GWO-EPD). The rest of the calculations and updating equations are similar to the original GWO algorithm. Furthermore, since the proposed approach intrinsically maintains diversity over the whole search space, a random re-initialization process performed on the solutions in the FB-GWO-EPD is unnecessary. Eliminating this re-initialization process in the proposed DB-GWO-EPD can reduce the proposed algorithm's complexity and runtime, at least when handling global optimization problems with slightly expensive objective function evaluations. The pseudo-code of the proposed DB-GWO-EPD algorithm is described in Algorithm 1. Moreover, the flowchart of the DB-GWO-EPD is depicted in Figure 1.

Algorithm 1: Pseudo-code of the DB-GWO-EPD algorithm.

```
Initialize the DB-GWO-EPD parameters, $a$ = [2, 0], population size ($N$), and the maximum number of iterations ($M$)
while $t < M$ do
    if $t = 1$ then
        for ($j$ = 1:$N$) do
            Initialize random positions for the jth solution at the first iteration as follows:
            $\vec{X_j} = \left(\vec{ub} - \vec{lb}\right).\vec{r_j} + \vec{lb}$; $\vec{ub}$ and $\vec{lb}$ are the upper and lower bounds of each dimension and $\vec{r_j}$ is a random vector
        end for
    else
        for ($j$ = 1:$N$) do
            Calculate $\vec{X_1}$, $\vec{X_2}$, and $\vec{X_3}$, as the guiding solutions of the jth solution, using Equations (6)–(8)
            Calculate $\vec{X}$ as the arithmetic average of $\vec{X_1}$, $\vec{X_2}$, and $\vec{X_3}$, using Equation (9)
            Adopt $\vec{X}$ as the newly updated position of the jth solution at the tth itertaion
        end for
    end if
    Calculate the fitness function value for each solution
    Sort the fitness function values and their corresponding solutions in an ascending order for minimization purpose
    Identify the first half of the solutions in the solutions sorted as the ones to be repositioned
```

Algorithm 1: *Cont.*

```
        Identify three best-fitted solutions ($\vec{X}_{\alpha}(t)$, $\vec{X}_{\beta}(t)$ and $\vec{X}_{\delta}(t)$), and save them in the memory of the algorithm
        Save $\vec{X}_{\alpha}(t)$ as the best-fitted solution found so far and name it $\vec{X}_{best}$
        Calculate the diversity index ($d_i$) for each solution using Equation (14)
        Appoint three solution positions with the highest $d_i$ values, as the $\vec{X}^{div}_{\alpha}(t)$, $\vec{X}^{div}_{\beta}(t)$, and $\vec{X}^{div}_{\delta}(t)$
        Reposition the first half of the best-fitted solutions around $\vec{X}^{div}_{\alpha}(t)$, $\vec{X}^{div}_{\beta}(t)$, and $\vec{X}^{div}_{\delta}(t)$, randomly, using Equations (15)–(17)
        Adopt the repositioned solution positions as their new positions
        $t = t + 1$
    end while
    Return $\vec{X}_{best}$ as the final result of the optimization process
```

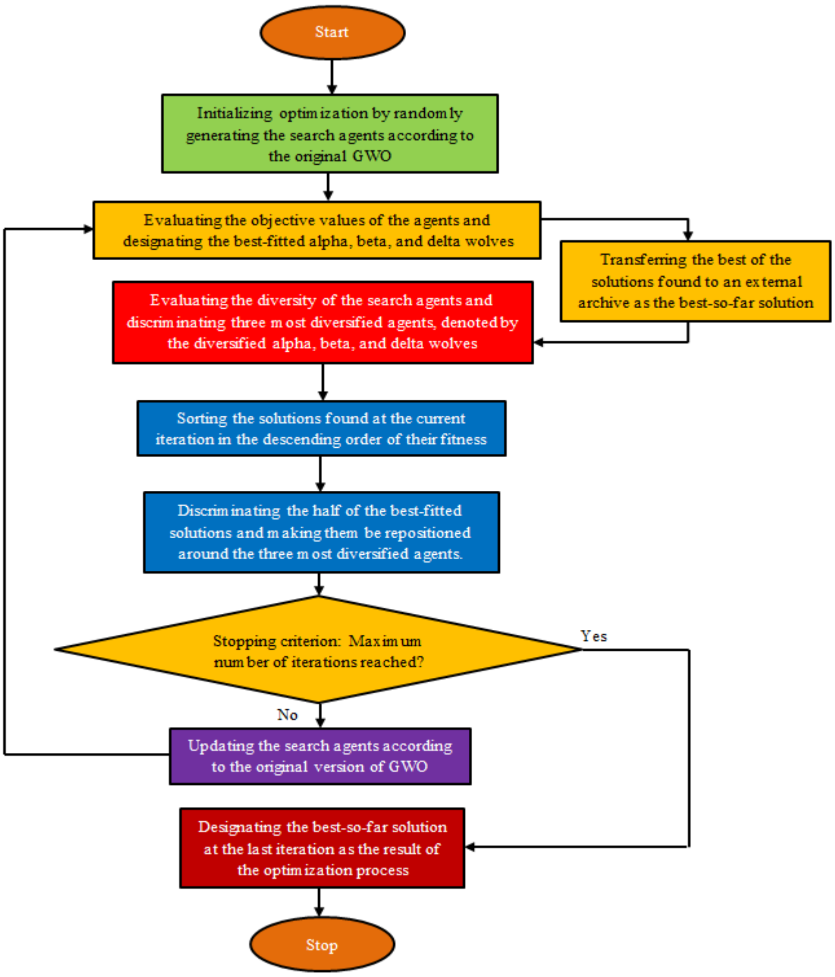
Start
Initializing optimization by randomly generating the search agents according to the original GWO
Evaluating the objective values of the agents and designating the best-fitted alpha, beta, and delta wolves
Transferring the best of the solutions found to an external archive as the best-so-far solution
Evaluating the diversity of the search agents and discriminating three most diversified agents, denoted by the diversified alpha, beta, and delta wolves
Sorting the solutions found at the current iteration in the descending order of their fitness
Discriminating the half of the best-fitted solutions and making them be repositioned around the three most diversified agents.
Stopping criterion: Maximum number of iterations reached?
Yes
No
Updating the search agents according to the original version of GWO
Designating the best-so-far solution at the last iteration as the result of the optimization process
Stop

Figure 1. Flowchart of DB-GWO-EPD.

The Pros and Cons of the Proposed DB-GWO-EPD and Its Computational Complexity

As comprehensively explained in the previous sub-section, the main advantage of the DB-GWO-EPD against the FB-GWO-EPD and the original GWO may be hidden in its highly enhanced exploration capability, as the search agents are frequently pushed towards the high-fitness region and pulled out of this region to be diversified and be enabled to discover much more good candidate positions in the search space. Another advantage is the elimination of the factor $\vec{C}$ may be so interesting to be applied to this algorithm, unlike the FB-GWO-EPD and the original GWO. This elimination is of higher importance when the runtime of these algorithms is discussed, especially when implemented on global optimization problems. The proposal also suffers from some weaknesses, the most important of which is difficulty in the intensification of the search process, especially at the later iterations of the optimization, which may result from the unbalanced contribution of the random auxiliary positions added to the diversified solutions in the repositioning procedure. This problem may be mitigated by setting an emotional weight for these random positions that highly decreases over the course of iterations.

The computational complexity of the DB-GWO-EPD is the summation of the complexity of the algorithm in four items: (1) initialization process, (2) mechanism for sorting the elite solutions, (3) calculation of the diversity each solution has in the search space, and (4) position updating procedure of the solutions. The complexity of the initialization process is $O(N \times D)$, and the complexity of the elites' sorting mechanism as well as the diversity calculation are $O(M \times N^2)$. Finally, the complexity of the updating procedure of the solutions is $O(M \times N \times D)$. Consequently, the total computational complexity of the proposed algorithm is $O(M \times N^2) + O(M \times N \times D) + O(N \times D) = O(M \times N \times (N + D))$. It is worth noticing that M is the maximum number of iterations, N is the population size, and D is the variable decision size.

4. Results and Discussion

4.1. Benchmark Functions

To test the competence of the proposed DB-GWO-EPD algorithm in solving optimization problems, a set of 13 shifted classical benchmark functions, as well as all the test functions, included in the technical report of the 2017 IEEE Congress on Evolutionary Computation (CEC), called CEC2017 test suite are adopted, and the proposal and its competitors are all implemented on these problems [37,38]. The proposed method was deliberately applied to these two sets of high-dimensional benchmark functions with shifted global optima to examine how well it can handle large-scale and hard-to-solve optimization problems compared to its competitors. The CEC2017 test suite was also chosen as another test bed to assess the competence of the optimizers to solve such hard-to-solve problems that can toughly challenge any optimizer when tackling them.

The mathematical formulation of the uni-modal and multi-modal shifted classical benchmark functions, as well as the CEC 2017 test suite, along with their optimal fitness values, can be seen in Tables 1–3. In these tables, n denotes the number of dimensions of the corresponding problem. The global optima of these functions are all shifted from the center of the domain to a position away from the center to make these problems more challenging and complex to solve and more realistic for any optimizer when solving them. The shifted positions adopted for the global optima of the classical functions are all set similar to the same in extensive experiments conducted in the literature [35,39].

Table 1. The shifted uni-modal benchmark functions.

Benchmark Function	Range	Shifted Position	f_{min}
$F_1(X) = \sum_{d=1}^{n} x_d^2$	$[-100,\ 100]^n$	$[-30,\ -30,\ \ldots,\ -30]$	0
$F_2(X) = \sum_{d=1}^{n} \lvert x_d \rvert + \prod_{d=1}^{n} \lvert x_d \rvert$	$[-10,\ 10]^n$	$[-3,\ -3,\ \ldots,\ -3]$	0
$F_3(X) = \sum_{d=1}^{n} \left(\sum_{j=1}^{d} x_j \right)^2$	$[-100,\ 100]^n$	$[-30,\ -30,\ \ldots,\ -30]$	0
$F_4(X) = \max_d \{ \lvert x_d \rvert, 1 \leq d \leq n \}$	$[-100,\ 100]^n$	$[-30,\ -30,\ \ldots,\ -30]$	0
$F_5(X) = \sum_{d=1}^{n-1} \left[100 (x_{d+1} - x_d^2)^2 + (x_d - 1)^2 \right]$	$[-30,\ 30]^n$	$[-15,\ -15,\ \ldots,\ -15]$	0
$F_6(X) = \sum_{d=1}^{n} (x_d + 0.5)^2$	$[-100,\ 100]^n$	$[-750,\ -750,\ \ldots,\ -750]$	$4.225 \times 10^5 \times n$
$F_7(X) = \sum_{d=1}^{n} \mathrm{d}x_d^4 + \text{random}\ [0, 1)$	$[-1.28,\ 1.28]^n$	$[-0.25,\ -0.25,\ \ldots,\ -0.25]$	0

Table 2. The shifted multi-modal benchmark functions.

Benchmark Functions	Range	Shifted Position	f_{min}
$F_8(X) = \sum_{d=1}^{n} - x_d \sin \sqrt{\lvert x_d \rvert}$	$[-500,\ 500]^n$	$[-300,\ -300,\ \ldots,\ -300]$	$-418.9829 \times n$
$F_9(X) = \sum_{d=1}^{n} \left[x_d^2 -\ 10\cos(2\pi x_d) + 10 \right]$	$[-5.12,\ 5.12]^n$	$[-2,\ -2,\ \ldots,\ -2]$	0
$F_{10}(X) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{d=1}^{n} x_d^2} \right) - \exp\left(\frac{1}{n}\sum_{d=1}^{n}\cos(2\pi x_d) \right) + 20 + e$	$[-32,\ 32]^n$	$[-16,\ -16,\ \ldots,\ -16]$	0
$F_{11}(X) = \frac{1}{4000}\sum_{d=1}^{n} x_d^2 - \prod_{d=1}^{n}\cos\left(\frac{x_d}{\sqrt{d}}\right) + 1$	$[-600,\ 600]^n$	$[-400,\ -400,\ \ldots,\ -400]$	0
$F_{12}(X) = \frac{\pi}{n}\left\{ 10\sin(\pi y_1) + \sum_{d=1}^{n-1}(y_d - 1)^2\left[1 + 10\sin^2(\pi y_{d+1})\right] + (y_n - 1)^2 \right\} + \sum_{d=1}^{n} u(x_d,\ 10,\ 100,\ 40)$	$[-50,\ 50]^n$	$[-30,\ -30,\ \ldots,\ -30]$	0
$y_d\ =\ \frac{x_d+1}{4}$			
$u(x_d, a,\ k,\ m) = \begin{cases} k(x_d - a)^m & x_d > a \\ 0 & -a < x_d < a \\ k(-x_d - a)^m & x_d < -a \end{cases}$			
$F_{13}(X) = 0.1\{\sin^2(3\pi x_1) + \sum_{d=1}^{n}(x_d - 1)^2\left[1 + \sin^2(3\pi x_d + 1)\right] +\ (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right] + \sum_{d=1}^{n} u(x_d,\ 5,\ 100,\ 4)$	$[-50,\ 50]^n$	$[-100,\ -100,\ \ldots,\ -100]$	*

* The minimum of this function varies as the dimensionality (n) varies, however, if n = 100, then f_{min} = 4.1006 × 10^{10}.

All of the test functions are to be minimized. The shifted classical benchmark test problems are assumed to be high-dimensional, with the dimensionality set to 100. In addition, the CEC2017 test functions are adopted to be 50-dimensional to increase the difficulty the algorithms may face when solving such hard-to-handle and complex optimization problems. All of these settings are made to examine better the proposed algorithm's real strength against the other comparative algorithms. The first test bed consists of two categories of uni-modal functions (F1–F7) and multi-modal functions (F8–F13). The first one challenges the exploitation capability of the algorithms and tests their convergence rate of them when finding out the global optimum of the problems, and the second one tests the exploration capability of the algorithms, investigating if the optimization algorithms can avoid local optima entrapments and detect all the possible good candidate solutions in the search space. The CEC2017 test suite encompasses a variety of composition, hybrid, uni-modal, and multi-modal functions. Thus, it can suitably benchmark the maximum ability of the examined algorithms to solve a wide range of problems providing great difficulties to an optimizer when attempting to search for the optimum of such problems.

Table 3. Summary of the CEC2017 test functions.

Description	NO.	Function	f_{min}
Uni-modal Functions	1	Shifted and Rotated Bent Cigar Function	100
	2	Shifted and Rotated Sum of Different Power Function *	200
	3	Shifted and Rotated Zakharov Function	300
Simple Multi-modal Functions	4	Shifted and Rotated Rosenbrock's Function	400
	5	Shifted and Rotated Rastrigin's Function	500
	6	Shifted and Rotated Expanded Scaffer's F6 Function	600
	7	Shifted and Rotated Lunacek Bi_Rastrigin Function	700
	8	Shifted and Rotated Non-Continuous Rastrigin's Function	800
	9	Shifted and Rotated Levy Function	900
	10	Shifted and Rotated Schwefel's Function	1000
Hybrid Functions	11	Hybrid Function 1 (N = 3)	1100
	12	Hybrid Function 2 (N = 3)	1200
	13	Hybrid Function 3 (N = 3)	1300
	14	Hybrid Function 4 (N = 4)	1400
	15	Hybrid Function 5 (N = 4)	1500
	16	Hybrid Function 6 (N = 4)	1600
	17	Hybrid Function 6 (N = 5)	1700
	18	Hybrid Function 6 (N = 5)	1800
	19	Hybrid Function 6 (N = 5)	1900
	20	Hybrid Function 6 (N = 6)	2000

Table 3. *Cont.*

Description	NO.	Function	f_{min}
Composition Functions	21	Composition Function 1 (N = 3)	2100
	22	Composition Function 2 (N = 3)	2200
	23	Composition Function 3 (N = 4)	2300
	24	Composition Function 4 (N = 4)	2400
	25	Composition Function 5 (N = 5)	2500
	26	Composition Function 6 (N = 5)	2600
	27	Composition Function 7 (N = 6)	2700
	28	Composition Function 8 (N = 6)	2800
	29	Composition Function 9 (N = 3)	2900
	30	Composition Function 10 (N = 3)	3000
Search Range: $[-100, 100]^D$; D is the dimensionality of the test problems			

* F2 has been excluded because it shows unstable behavior, especially for higher dimensions, and significant performance variations for the same algorithm implemented in Matlab, C.

4.2. Comparison with Other Well-Known Algorithms

The proposed DB-GWO-EPD was compared with the FB-GWO-EPD and the original GWO along with four other popular and newly-proposed algorithms, including Aquila Optimizer (AO) [40]; Flow Direction Algorithm (FDA) [41]; Arithmetic Optimization Algorithm (AOA) [42]; and Gradient-based Optimizer (GBO) [43]. The parameter settings of all the competitive algorithms are presented in Table 4.

Table 4. Parameter settings of the DB-GWO-EPD and its competitive algorithms.

Algorithm	Parameter Settings
AO	$r_1 \in [1,\ 20]$; $U = 0.00565$; $D_1 = D$; $\omega = 0.005$; $\alpha = \delta = 0.1$; $G_2\ = [2,\ 0]$
FDA	$\beta = 1$
AOA	$\alpha = 5$; $\mu = 0.5$
GBO	$\beta_{\mathrm{min}} = 0.2$; $\beta_{\mathrm{max}} = 1.2$; $\mathrm{pr} = 0.5$
GWO	$a = [2,\ 0]$
FB-GWO-EPD	$a = [2,\ 0]$
DB-GWO-EPD	$a = [2,\ 0]$

All algorithms were run 30 independent times, and the average and standard deviation of the final results achieved during all runs, abbreviated as "Ave" and "Std", respectively, are adopted as the performance criteria and reported in Tables 5 and 6. The results corresponding to the best-performing criteria are emboldened in these tables. The convergence curves of the algorithms, when implemented on the shifted benchmark functions, are also plotted in Figures 2 and 3. Furthermore, for a fair and unbiased comparison, the population size of all the algorithms is set to 30 for the first benchmark function set and adopted to be 50 for the CEC2017 test suite. In addition, all the algorithms were implemented for 1000 iterations at each runtime. The stopping criterion is also considered fulfilled after reaching the maximum number of iterations.

4.2.1. Comparing the Algorithms on the Uni-Modal Functions

The results of applying the algorithms to the uni-modal benchmark functions show that the DB-GWO-EPD highly outperforms the original GWO and the FB-GWO-EPD as the improved version of the GWO. While the FB-GWO-EPD is slightly better than the GWO on almost all uni-modal functions, the DB-GWO-EPD performs much better than both other algorithms on the entire test functions falling in this category except for F6.

The DB-GWO-EPD can reach the slightest standard deviations among its competitors on F1, F2, F3, and F7. This can guarantee the use of this algorithm without any need to run the algorithm frequently. In other words, the user of the proposed algorithm can rely much more on the results of the DB-GWO-EPD compared to the other version of GWO.

The proposed algorithm is superior to all the examined algorithms, including the four newly-proposed algorithms on F1, F2, F3, and F5. The closest rival of the proposed DB-GWO-EPD is the AO algorithm, outperforming its competitors on F4 and F7. Overall, the proposal is superior to the other algorithms on 6 out of 14 (43%) of the entire performance criteria (average and standard deviations of the best-achieved solutions). In comparison, each of the other examinees is superior to the others on a maximum of only 2 out of 14 (14%) criteria.

The main reason behind the superior results of the proposed algorithm is hidden in having a high exploration capability which, in turn, can impede the stagnation of the solutions in the search space by frequently diversifying them to accelerate the process of finding the optimal solutions across the search space and speed up the convergence of the proposal when implemented on such a high-dimensional uni-modal function set.

4.2.2. Comparing the Algorithms on the Multi-Modal Functions

The results of the multi-modal functions indicate the superiority of the proposed DB-GWO-EPD algorithm to the FB-GWO-EPD and the original GWO on most of the benchmark functions. The high capability of the proposed algorithm in the diversification of the search agents takes effect, especially when solving F10, F11, and F12. Compared with the other examined algorithms, including the popular and newly-proposed ones, the DB-GWO-EPD shows its superiority on 5 out of 12 (42%) of the total criteria, demonstrating its high efficacy in solving this set of problems.

Table 5. Results of the DB-GWO-EPD and its rivals on the shifted benchmark functions, with $n = 100$.

	Criteria	**AO**	**FDA**	**AOA**	**GBO**	**GWO**	**FB-GWO-EPD**	**DB-GWO-EPD**
F1	Ave	6.6579×10^2	9.2465×10	8.1744×10^4	4.2565	2.6986×10^4	5.0429×10^2	$\mathbf{8.3539 \times 10^{-1}}$
	Std	3.7689×10^2	2.7576×10	3.5185×10^3	1.8961	4.6165×10^3	1.4285×10^2	$\mathbf{3.0210 \times 10^{-1}}$
F2	Ave	6.2467×10	1.2532×10	8.2193×10^{46}	7.1612	1.5454×10^2	1.0983×10^2	**1.9838**
	Std	1.0766×10	1.2938×10	1.4548×10^{47}	2.5304	1.7258×10	2.2135×10	**1.7873**
F3	Ave	6.2975×10^5	5.1412×10^4	5.2178×10^7	5.0863×10^4	6.5265×10^4	4.5943×10^4	$\mathbf{2.8288 \times 10^4}$
	Std	3.3353×10^5	1.2203×10^4	3.0106×10^7	1.1385×10^4	1.2188×10^4	$\mathbf{2.3826 \times 10^3}$	4.5466×10^3
F4	Ave	**6.4199**	5.7145×10	3.0010×10	3.0000×10	3.0000×10	3.0001×10	1.0698×10
	Std	1.0959	3.2665	1.0620×10^{-2}	**0**	1.4235×10^{-5}	1.1221×10^{-3}	2.7652
F5	Ave	7.1099×10^5	2.6441×10^4	4.2934×10^8	2.4430×10^3	8.4413×10^7	1.7271×10^3	$\mathbf{7.9457 \times 10^2}$
	Std	2.3792×10^5	1.0748×10^4	4.8797×10^6	1.0962×10^3	1.3821×10^7	$\mathbf{6.9398 \times 10^2}$	9.8350×10^2
F6	Ave	4.7087×10^7	4.2455×10^7	5.4015×10^7	$\mathbf{4.2315 \times 10^7}$	4.5852×10^7	4.2609×10^7	4.3322×10^7
	Std	1.1035×10^6	2.0457×10^5	5.9905×10^5	**0**	3.4135×10^5	1.4011×10^5	1.1990×10^5
F7	Ave	$\mathbf{5.9662 \times 10^{-2}}$	1.8381	1.9591×10	5.2178×10^{-1}	7.6576	2.0676	2.3332×10^{-1}
	Std	8.1757×10^{-2}	2.3114×10^{-1}	$\mathbf{6.6384 \times 10^{-2}}$	1.1624×10^{-1}	9.3946×10^{-1}	4.0188×10^{-1}	6.7209×10^{-2}
F8	Ave	-4.0026×10^4	-4.0001×10^4	-1.6152×10^4	$\mathbf{-4.6366 \times 10^4}$	-3.0043×10^4	-3.1018×10^4	-2.9074×10^4
	Std	5.0202×10^3	2.8268×10^3	$\mathbf{1.4243 \times 10^3}$	3.3289×10^3	2.0813×10^3	6.5483×10^3	1.0665×10^4
F9	Ave	$\mathbf{6.4745 \times 10}$	4.8795×10^2	3.9917×10^2	3.7281×10^2	3.5234×10^2	3.7875×10^2	4.2068×10^2
	Std	2.4065×10	5.6871×10	$\mathbf{3.0287 \times 10^{-2}}$	7.5392	1.5079×10	3.7176×10	2.7046×10^2
F10	Ave	9.2096	1.9712×10	1.9185×10	1.0791×10	1.7895×10	1.1858×10	**1.1055**
	Std	8.7246×10^{-1}	2.6169×10^{-1}	$\mathbf{4.3534 \times 10^{-6}}$	2.2658	9.7602×10^{-1}	7.1181	3.0835×10^{-1}
F11	Ave	2.0904×10^2	1.9947×10	2.1440×10^3	1.0766	5.7018×10^2	2.1362	$\mathbf{4.9285 \times 10^{-1}}$
	Std	2.5256×10^2	8.7853	1.2316×10^2	7.3605×10^{-2}	1.3720×10^2	6.0558×10^{-1}	$\mathbf{7.3081 \times 10^{-2}}$
F12	Ave	9.2327×10^2	1.2972×10^3	1.5442×10^9	1.4721×10	2.0258×10^8	1.2536×10	**1.5566**
	Std	2.7363×10^3	1.4015×10^3	2.8499×10^7	4.6852	3.6813×10^7	4.0517	$\mathbf{6.3168 \times 10^{-1}}$
F13	Ave	4.7531×10^{10}	$\mathbf{4.1006 \times 10^{10}}$	7.9630×10^{11}	$\mathbf{4.1006 \times 10^{10}}$	1.8052×10^{11}	4.2906×10^{10}	5.3127×10^{10}
	Std	1.5675×10^{10}	**0**	1.2203×10^{10}	**0**	3.8026×10^{10}	1.6574×10^8	1.0299×10^9

Table 6. Results of the DB-GWO-EPD and its rivals on the 50-dimensional CEC2017 test functions.

	Criteria	AO	FDA	AOA	GBO	GWO	FB-GWO-EPD	DB-GWO-EPD
F1	Ave	1.6838×10^{9}	3.8271×10^{4}	1.0679×10^{11}	**7.2920×10^{3}**	8.2896×10^{9}	5.8980×10^{7}	1.8071×10^{4}
	Std	6.2457×10^{8}	4.4200×10^{4}	1.0241×10^{10}	**8.4348×10^{3}**	4.2006×10^{9}	2.0039×10^{7}	9.2804×10^{3}
F3	Ave	2.0248×10^{5}	**2.7961×10^{4}**	1.6642×10^{5}	3.9352×10^{4}	1.0484×10^{5}	6.6347×10^{4}	5.0470×10^{4}
	Std	4.4288×10^{4}	**8.9157×10^{3}**	2.0180×10^{4}	9.7975×10^{3}	1.6965×10^{4}	1.3566×10^{4}	1.2134×10^{4}
F4	Ave	1.0630×10^{3}	**5.4393×10^{2}**	3.0754×10^{4}	5.5786×10^{2}	1.1238×10^{3}	6.6857×10^{2}	5.9875×10^{2}
	Std	1.9549×10^{2}	5.8808×10	7.7284×10^{3}	5.3041×10	2.3977×10^{2}	**4.1381×10**	5.2919×10
F5	Ave	8.6665×10^{2}	8.2013×10^{2}	1.1525×10^{3}	8.1004×10^{2}	7.2367×10^{2}	6.7674×10^{2}	**6.5307×10^{2}**
	Std	3.3186×10	6.2431×10	4.0010×10	4.9971×10	**2.7447×10**	4.7522×10	9.5217×10
F6	Ave	6.6537×10^{2}	6.5096×10^{2}	6.9011×10^{2}	6.3725×10^{2}	6.1746×10^{2}	6.1596×10^{2}	**6.0456×10^{2}**
	Std	5.3658×10	8.7769	5.1072	9.7671	5.4024	5.9317	**2.9473**
F7	Ave	1.5161×10^{3}	1.4326×10^{3}	1.9341×10^{3}	1.2644×10^{3}	1.0724×10^{3}	1.0822×10^{3}	**9.7839×10^{2}**
	Std	1.2247×10^{2}	1.1392×10^{2}	6.3770×10	9.4996×10	5.9392×10	**4.4810×10**	1.5911×10^{2}
F8	Ave	1.1804×10^{3}	1.1620×10^{3}	1.4717×10^{3}	1.1110×10^{3}	1.0259×10^{3}	1.0124×10^{3}	**9.1755×10^{2}**
	Std	3.4016×10	4.3580×10	4.7283×10	5.6324×10	5.7317×10	9.3255×10	**2.7887×10**
F9	Ave	2.1934×10^{4}	1.0603×10^{4}	2.8784×10^{4}	7.4002×10^{3}	8.0871×10^{3}	6.2582×10^{3}	**1.5274×10^{3}**
	Std	3.8379×10^{3}	2.4207×10^{3}	4.3133×10^{3}	2.4743×10^{3}	3.3907×10^{3}	2.5762×10^{3}	**1.2381×10^{3}**
F10	Ave	9.1459×10^{3}	8.4369×10^{3}	1.3462×10^{4}	7.8672×10^{3}	**7.5159×10^{3}**	9.8846×10^{3}	8.1947×10^{3}
	Std	9.7917×10^{2}	1.0053×10^{3}	**7.8133×10^{2}**	9.4913×10^{2}	1.7420×10^{3}	4.0438×10^{3}	3.5965×10^{3}
F11	Ave	2.2437×10^{3}	**1.3379×10^{3}**	2.2605×10^{4}	1.3879×10^{3}	4.3152×10^{3}	1.5887×10^{3}	1.4279×10^{3}
	Std	2.7895×10^{2}	**7.0421×10**	3.8152×10^{3}	8.3613×10	1.5839×10^{3}	1.0617×10^{2}	8.1144×10
F12	Ave	6.1213×10^{8}	**2.2862×10^{6}**	6.8321×10^{10}	2.6664×10^{6}	5.9699×10^{8}	1.5015×10^{8}	2.3364×10^{7}
	Std	4.0105×10^{8}	**1.4679×10^{6}**	1.5355×10^{10}	2.2857×10^{6}	6.4348×10^{8}	8.5096×10^{7}	1.7026×10^{7}
F13	Ave	2.3863×10^{7}	**6.1931×10^{3}**	3.9481×10^{10}	1.2570×10^{4}	4.1420×10^{8}	8.5343×10^{5}	8.4775×10^{4}
	Std	3.8111×10^{7}	**7.0134×10^{3}**	1.2296×10^{10}	9.9819×10^{3}	8.8631×10^{8}	4.4079×10^{5}	4.8870×10^{4}
F14	Ave	4.8743×10^{6}	**3.1987×10^{4}**	5.2754×10^{7}	3.9591×10^{4}	1.0451×10^{6}	3.7271×10^{5}	1.9834×10^{5}
	Std	4.3827×10^{6}	**3.1930×10^{4}**	4.4620×10^{7}	3.9226×10^{4}	1.2344×10^{6}	2.3847×10^{5}	9.5941×10^{4}
F15	Ave	6.1620×10^{5}	**1.0266×10^{4}**	4.4614×10^{9}	1.2206×10^{4}	1.8441×10^{7}	1.4188×10^{5}	4.2204×10^{4}
	Std	3.6634×10^{5}	**6.4333×10^{3}**	2.6802×10^{9}	7.4912×10^{3}	3.2540×10^{7}	1.8065×10^{5}	2.4322×10^{4}
F16	Ave	4.3612×10^{3}	3.6774×10^{3}	7.9190×10^{3}	3.4870×10^{3}	3.2102×10^{3}	2.9626×10^{3}	**2.8461×10^{3}**
	Std	5.3030×10^{2}	5.1378×10^{2}	1.2332×10^{3}	4.9718×10^{2}	4.4746×10^{2}	4.1833×10^{2}	**3.4984×10^{2}**
F17	Ave	3.6432×10^{3}	3.3813×10^{3}	9.1957×10^{3}	3.0819×10^{3}	2.9329×10^{3}	2.8919×10^{3}	**2.8425×10^{3}**
	Std	3.8545×10^{2}	3.7707×10^{2}	2.4943×10^{3}	3.4783×10^{2}	**3.4609×10^{2}**	3.9359×10^{2}	4.1003×10^{2}
F18	Ave	9.1232×10^{6}	2.2535×10^{5}	1.0356×10^{8}	**2.1870×10^{5}**	4.4394×10^{6}	2.9262×10^{6}	2.3528×10^{6}
	Std	6.4256×10^{6}	1.5052×10^{5}	4.8975×10^{7}	**1.3210×10^{5}**	5.2595×10^{6}	1.9711×10^{6}	2.3575×10^{6}
F19	Ave	2.2803×10^{6}	**1.8197×10^{4}**	2.7898×10^{9}	1.8302×10^{4}	4.1561×10^{6}	1.4102×10^{6}	8.0552×10^{5}
	Std	2.1570×10^{6}	**1.0490×10^{4}**	1.4850×10^{9}	1.1745×10^{4}	8.5345×10^{6}	1.0476×10^{6}	5.9277×10^{5}
F20	Ave	3.2709×10^{3}	3.4256×10^{3}	3.5912×10^{3}	3.2025×10^{3}	**2.9462×10^{3}**	3.1085×10^{3}	2.9489×10^{3}
	Std	**2.6175×10^{2}**	3.3007×10^{2}	2.6817×10^{2}	4.0586×10^{2}	3.6343×10^{2}	5.4458×10^{2}	4.7087×10^{2}
F21	Ave	2.7043×10^{3}	2.6240×10^{3}	3.0782×10^{3}	2.5662×10^{3}	2.5173×10^{3}	2.4638×10^{3}	**2.4115×10^{3}**
	Std	6.3075×10	6.1326×10	8.5153×10	5.1954×10	5.5484×10	5.0296×10	**2.3648×10**
F22	Ave	1.0965×10^{4}	9.9308×10^{3}	1.5926×10^{4}	9.4638×10^{3}	9.6836×10^{3}	1.0955×10^{4}	**8.7319×10^{3}**
	Std	1.6841×10^{3}	8.3742×10^{2}	**7.0190×10^{2}**	1.6273×10^{3}	2.0126×10^{3}	3.9900×10^{3}	3.0635×10^{3}
F23	Ave	3.4393×10^{3}	3.1004×10^{3}	4.4239×10^{3}	3.0558×10^{3}	2.9820×10^{3}	2.9274×10^{3}	**2.8798×10^{3}**
	Std	9.2377×10	8.1408×10	2.3112×10^{2}	7.8350×10	**6.3290×10**	8.7578×10	6.8065×10
F24	Ave	3.5218×10^{3}	3.2665×10^{3}	4.9199×10^{3}	3.1724×10^{3}	3.1932×10^{3}	3.0855×10^{3}	**3.0343×10^{3}**
	Std	1.2135×10^{2}	9.9971×10	3.1459×10^{2}	**6.0068×10**	1.1563×10^{2}	1.2804×10^{2}	7.5755×10
F25	Ave	3.4327×10^{3}	3.0855×10^{3}	1.5444×10^{4}	3.0855×10^{3}	3.5262×10^{3}	3.2034×10^{3}	**3.0668×10^{3}**
	Std	8.8058×10	**2.3492×10**	1.3989×10^{3}	2.3553×10	2.1484×10^{2}	5.9425×10	2.6992×10
F26	Ave	8.6249×10^{3}	9.0556×10^{3}	1.6923×10^{4}	7.0615×10^{3}	6.3760×10^{3}	5.8852×10^{3}	**5.0518×10^{3}**
	Std	2.4721×10^{3}	1.8166×10^{3}	1.2165×10^{3}	2.4977×10^{3}	5.2971×10^{2}	8.0009×10^{2}	**3.4836×10^{2}**
F27	Ave	4.0141×10^{3}	3.6061×10^{3}	6.7731×10^{3}	3.5950×10^{3}	3.6165×10^{3}	3.4305×10^{3}	**3.4253×10^{3}**
	Std	1.9720×10^{2}	1.5115×10^{2}	7.3983×10^{2}	1.3302×10^{2}	1.2148×10^{2}	**5.3921×10**	6.6431×10
F28	Ave	4.2899×10^{3}	3.3315×10^{3}	1.2297×10^{4}	3.3326×10^{3}	4.2560×10^{3}	3.4948×10^{3}	**3.3136×10^{3}**
	Std	2.7492×10^{2}	2.9990×10	1.3466×10^{3}	2.7941×10	4.0559×10^{2}	8.1306×10	**2.5638×10**
F29	Ave	6.1689×10^{3}	4.7010×10^{3}	3.6559×10^{4}	4.6781×10^{3}	4.6299×10^{3}	4.3602×10^{3}	**4.2874×10^{3}**
	Std	6.9454×10^{2}	4.4253×10^{2}	2.9085×10^{4}	3.7906×10^{2}	**2.8006×10^{2}**	2.8032×10^{2}	3.0278×10^{2}
F30	Ave	1.2248×10^{8}	1.1529×10^{6}	5.8046×10^{9}	**1.0718×10^{6}**	1.1702×10^{8}	9.2911×10^{7}	3.9885×10^{7}
	Std	4.9406×10^{7}	2.9984×10^{5}	2.6383×10^{9}	**2.1165×10^{5}**	5.0559×10^{7}	2.0213×10^{7}	7.6628×10^{6}

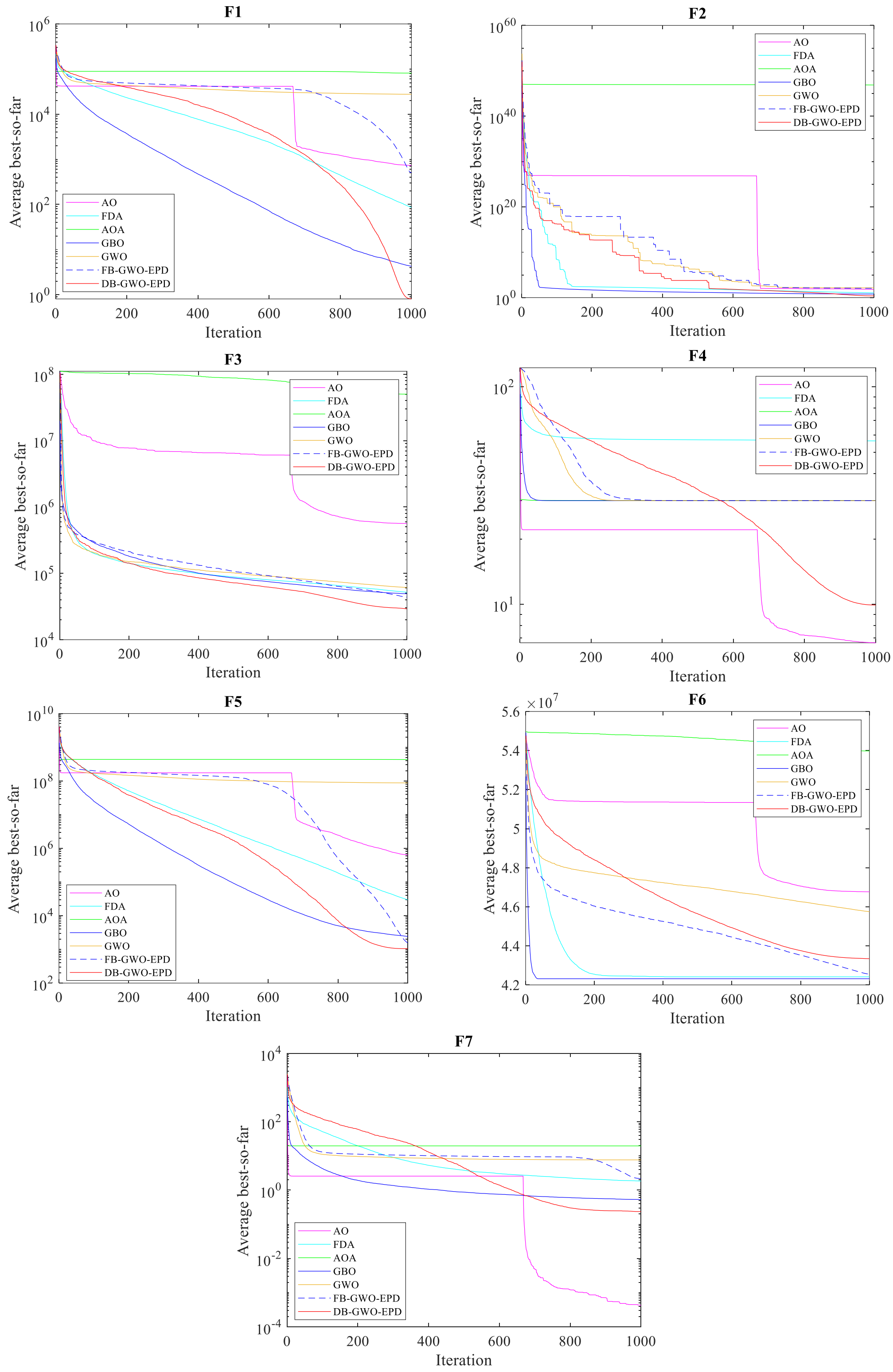

Figure 2. Convergence curves plotted for the algorithms on the uni-modal shifted benchmark functions.

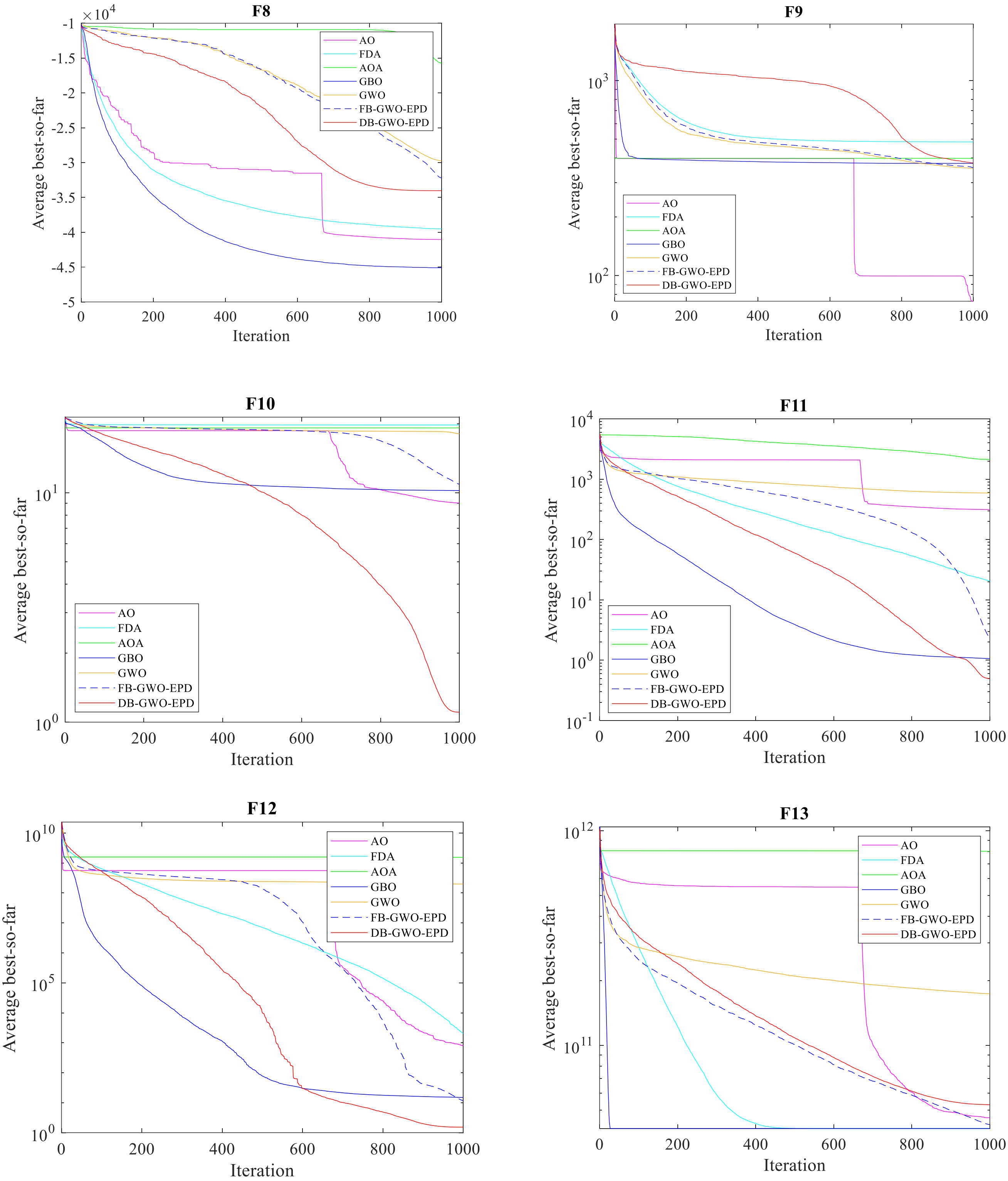

Figure 3. Convergence curves plotted for the algorithms on the multi-modal shifted benchmark functions.

Knowing that the multi-modal functions with many dimensions, as examined in this study, have a lot of local optima in their search space, an algorithm should have a powerful exploration capability to overcome these natural difficulties the multi-modal functions offer. Moreover, in all of the benchmark functions employed in this study, the optimal solution is shifted in all dimensions, and this alteration applied to these functions can make them significantly harder to solve. It can be inferred from the results obtained that the proposed DB-GWO-EPD algorithm is versatile and robust, tackling the difficulties mentioned for the high-dimensional multi-modal functions as the most realistic representatives of real-world optimization problems.

4.2.3. Comparing the Algorithms on the CEC2017 Test Suite

The CEC2017 benchmark functions are rated as a test bed that can challenge any optimization algorithm comprehensively. The problems in this set seem more realistic and more similar to the real-world optimization problems as compared to those in the first 13-problem set used to examine the eligibility of the proposed algorithm and its rivals. This test suite comprises a variety of composition, hybrid, uni-modal, and multi-modal functions. Here, all the functions included in the CEC2017 test suite are set to be 50-dimensional to challenge further the capabilities of the proposal to handle such large-scale and complex optimization problems.

As the results suggest, the proposed DB-GWO-EPD significantly outperforms the other comparative algorithms on F21–F29 problems. These problems are known as the composition functions comprising a variety of shifted, rotated, and biased multi-modal functions, representing them as the most challenging category of the test problems included in the CEC2017 benchmark functions. As a result, the superiority of the proposal on these functions can affirm the power of this algorithm to solve hard-to-solve optimization problems. Although the proposed DB-GWO-EPD performs much better than the FB-GWO-EPD and the original GWO algorithms on 10 out of 10 (100%) of the F11–F20 problems, the proposal loses the competition in solving these problems to the FDA algorithm having 60% outperformance rate that overtakes the DB-GWO-EPD with an outperformance rate of 15% on the F11–F20. This problem set is called the hybrid functions, the second most challenging category in the CEC2017 test suite. The proposal also outperforms both the original GWO and the FB-GWO-EPD on the first ten functions of this test bed that can technically be divided into uni-modal and multi-modal functions.

The proposal is also superior to all rivals on 5 out of 9 (56%) of the F1–F10 problems. Note that none of the algorithms is tested on F2, as recommended by the creators of the CEC2017 test suite. FDA is rated as the second-best algorithm outperforming the other examinees on 2 out of 9 (22%) of these problems. Overall, the proposed algorithm can achieve the best average objective function values on 16 out of 29 (55%) criteria in the CEC2017 test bed. In addition, the DB-GWO-EPD tackles its rivals in 23 out of 58 (40%) cases, including the average and standard deviation of the best-reached results by the optimizers on this test suite.

The main reason for the high competence of the proposed DB-GWO-EPD algorithm may be the strong capability of the proposed algorithm to preserve the diversity of the solutions during the whole optimization process, which especially matches the nature of this test problems suite. This property incorporated into the proposal can make this algorithm tackle its competitors when facing many local optima in the search space of multi-modal problems. In this case, the proposed algorithm explores the search space as much as possible as a result of being enabled to escape from such a large number of local optima and finds the best-fitted region and, finally, the optimal solution to the problem, especially when handling complex optimization problems. Preserving diversity not only guarantees reaching the optimal solution of a multi-modal problem but could also accelerate the convergence in solving the uni-modal problems when their global optimum is shifted to the corner of the problem domain, similar to the cases examined in this paper.

4.3. Statistical Analysis

In statistical analysis, the Wilcoxon rank-sum test is used as a non-parametric test to determine whether a couple of solutions follow the same probability distribution function (García et al., 2008). This test finally renders an output named p-value. If the p-value < 0.05 for an algorithm, it is assumed to be statistically significantly different compared to its competitor. This difference can appear in both cases of superiority and non-superiority in the algorithms' results. The test results are presented in Tables 7 and 8. In these tables, the expression N/A mentioned in each row denotes that the algorithm ahead of that row is "Not Applicable" in the test, meaning this algorithm cannot be compared. On the shifted classical benchmark functions, the proposed DB-GWO-EPD is assigned a p-value more significant than the level of 5% significance versus AO and GBO, despite being superior to them on a vast majority of the functions. The critical point is that the DB-GWO-EPD cannot achieve a p-value less than 0.05 when facing the FB-GWO-EPD. However, the p-value that the test produces for DB-GWO-EPD versus the original GWO is much less than that the test offers for the FB-GWO-EPD versus GWO, implicitly demonstrating that the results of the proposed DB-GWO-EPD are more significantly superior to the FB-GWO-EPD. While on the CEC2017 test problems, the proposed DB-GWO-EPD is significantly superior to all its rivals, as the p-values generated pairwise for this algorithm against the others clearly suggest. Since the CEC2017 test suite is a more appropriate bed for evaluating the overall performance of an algorithm, having significant superiority to a set of popular algorithms can properly delineate the high competence of the DB-GWO-EPD in solving optimization problems.

Table 7. p-values calculated for the shifted classical functions (p-values ≥ 0.05 have been underlined).

Algorithms	AO	FDA	AOA	GBO	GWO	FB-GWO-EPD	DB-GWO-EPD
AO	N/A						
FDA	$\underline{7.7641 \times 10^{-1}}$	N/A					
AOA	1.0217×10^{-3}	2.5245×10^{-3}	N/A				
GBO	4.6500×10^{-2}	3.2649×10^{-2}	3.9261×10^{-5}	N/A			
GWO	1.5966×10^{-2}	1.4304×10^{-2}	$\underline{5.1522 \times 10^{-2}}$	8.1663×10^{-5}	N/A		
FB-GWO-EPD	$\underline{8.4754 \times 10^{-1}}$	$\underline{8.7630 \times 10^{-1}}$	9.8442×10^{-4}	4.6674×10^{-2}	1.4854×10^{-2}	N/A	
DB-GWO-EPD	$\underline{1.2496 \times 10^{-1}}$	$\underline{1.2324 \times 10^{-1}}$	3.3946×10^{-4}	$\underline{9.1839 \times 10^{-1}}$	9.5116×10^{-4}	$\underline{1.9343 \times 10^{-1}}$	N/A

Table 8. p-values calculated for the CEC2017 benchmark functions (p-values ≥ 0.05 have been underlined).

Algorithms	AO	FDA	AOA	GBO	GWO	FB-GWO-EPD	DB-GWO-EPD
AO	N/A						
FDA	4.4360×10^{-3}	N/A					
AOA	5.4935×10^{-15}	4.0441×10^{-17}	N/A				
GBO	2.5666×10^{-3}	$\underline{8.0087 \times 10^{-1}}$	3.2827×10^{-16}	N/A			
GWO	$\underline{9.3737 \times 10^{-2}}$	$\underline{2.2604 \times 10^{-1}}$	3.9835×10^{-17}	$\underline{1.1853 \times 10^{-1}}$	N/A		
FB-GWO-EPD	4.7753×10^{-3}	$\underline{3.5684 \times 10^{-1}}$	1.6881×10^{-15}	$\underline{3.8885 \times 10^{-1}}$	$\underline{1.5497 \times 10^{-1}}$	N/A	
DB-GWO-EPD	4.5442×10^{-10}	7.3072×10^{-3}	4.8602×10^{-18}	5.4212×10^{-4}	8.7465×10^{-8}	1.5154×10^{-6}	N/A

4.4. Comparative Results on Engineering Design Problems

An optimization algorithm may face some optimization problems having different difficulties, which in turn, can make these problems too hard-to-solve. The DB-GWO-EPD is implemented on four constrained engineering problems to validate further the proposal to tackle these difficulties. For constraint handling in these problems, the penalty function approach was adopted. In addition, 50 wolves, and 1000 iterations, are set for the DB-GWO-EPD algorithm when solving this problem set. The algorithm is run 30 times, and the best-achieved results are reported versus those of the other comparative algorithms. It is worth mentioning that the results found by the best-performing algorithm are emboldened in the tables showing the results of the optimization conducted on these problems.

4.4.1. Welded Beam Design Problem

This problem aims to minimize the construction cost of a welded beam. This problem is a high-constrained one, imposing several restrictions on the beam properties such as bending stress (σ), shear stress (τ), deflection at the end of the beam (δ), buckling load on the bar (P), and several other constraints. Four design components considered as the decision variables of the problem are illustrated in Figure 4 as $\mathrm{h}(x_1)$, $L(x_2)$, $t(x_3)$, and $\mathrm{b}(x_4)$. The problem formulation is as follows.

$$\text{Minimize}\, f\left(\overrightarrow{x}\right) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14 + x_2) \tag{18}$$

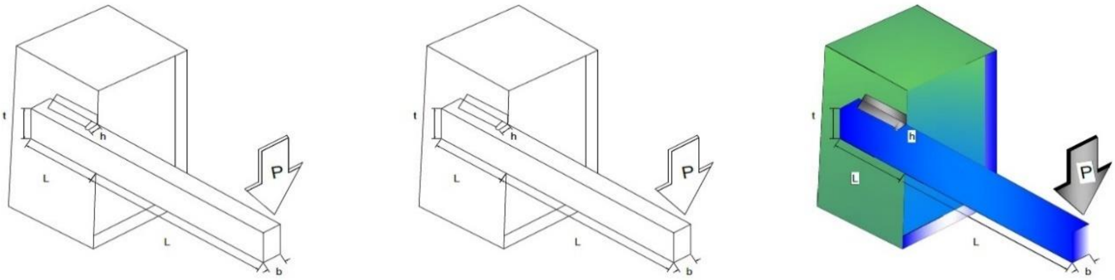

Figure 4. Welded beam scheme [44]. *"Reprinted from Publication Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer, vol. 191, 1–32, Copyright (2022), with permission from Elsevier."*

Subject to :

$$g_1\left(\overrightarrow{x}\right) = \tau(x) - \tau_{\max} \le 0 \tag{19}$$

$$g_2\left(\overrightarrow{x}\right) = \sigma(x) - \sigma_{\max} \le 0 \tag{20}$$

$$g_3\left(\overrightarrow{x}\right) = x_1 - x_4 \le 0 \tag{21}$$

$$g_4\left(\overrightarrow{x}\right) = 0.10471 x_1^2 + 0.04811 x_3 x_4 (14 + x_2) - 5 \le 0 \tag{22}$$

$$g_5\left(\overrightarrow{x}\right) = 0.125 - x_1 \le 0 \tag{23}$$

$$g_6\left(\overrightarrow{x}\right) = \delta(x) - \delta_{\max} \le 0 \tag{24}$$

$$g_7\left(\overrightarrow{x}\right) = P - P_c(x) \le 0 \tag{25}$$

$$0.1 \le x_i \le 2,\ i = 1,\ 4 \tag{26}$$

$$0.1 \le x_i \le 10,\ i = 2,\ 3 \tag{27}$$

where

$$\tau(x) = \sqrt{\left(\tau'\right)^2 + 2\tau'\tau''\frac{x_2}{2R} + \left(\tau''\right)^2},\ \tau' = \frac{P}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J} \tag{28}$$

$$M = P\left(L + \frac{x_2}{2}\right),\ R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},\ J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\} \tag{29}$$

$$\sigma(x) = \frac{6PL}{x_4x_3^2},\ \delta(x) = \frac{4PL^3}{Ex_3^3x_4},\ P_c(x) = \frac{4.013E\sqrt{\left(x_3^2x_4^6/36\right)}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right) \tag{30}$$

$$P = 26.698\ KN,\ L = 35.56\ \text{cm},\ E = 2.07 \times 10^4\ \frac{KN}{\text{cm}^2},\ G = 8270\ \frac{KN}{\text{cm}^2} \tag{31}$$

$$\tau_{max} = 9.38\ \frac{KN}{\text{cm}^2},\ \sigma_{max} = 20.7\ \frac{KN}{\text{cm}^2},\ \delta_{max} = 6.35\ \text{mm} \tag{32}$$

Table 9 shows the design variables and optimal cost to spend for designing such a welded beam when calculated by the proposed algorithm and its competitors. The results reveal that the DB-GWO-EPD can find the least cost among the other algorithms whose results are from the literature.

Table 9. Results of solving the welded beam design problem.

Algorithm	x_1	x_2	x_3	x_4	f_{min}
SIMPLEX [45]	0.279200	5.625600	7.751200	0.279600	2.530700
DAVID [45]	0.243400	6.255200	8.291500	0.244400	2.384100
APPROX [45]	0.244400	6.218900	8.291500	0.244400	2.381500
GA [46]	0.248900	6.173000	8.178900	0.253300	2.430000
HS [47]	0.244200	6.223100	8.291500	0.240000	2.380700
CSCA [48]	0.203137	3.542998	9.033498	0.206179	1.733461
CPSO [49]	0.202369	3.544214	9.048210	0.205723	1.728020
RO [50]	0.203687	3.528467	9.004233	0.207241	1.735344
WOA [51]	0.205396	3.484293	9.037426	0.206276	1.730499
GSA [7]	0.182129	3.856979	10.000000	0.202376	1.879950
MVO [52]	0.205463	3.473193	9.044502	0.205695	1.726450
OBSCA [53]	0.230824	3.069152	8.988479	0.208795	1.722315
AOA [42]	0.194475	2.570920	10.000000	0.201827	1.716400
GWO	0.205711	3.254161	9.035520	0.205794	1.695653
FB-GWO-EPD	0.205486	3.259422	9.036860	0.205771	1.696094
DB-GWO-EPD	**0.205699**	**3.253667**	**9.036660**	**0.205729**	**1.695281**

4.4.2. Three-Bar Truss Design Problem

This problem aims to minimize the weight of a three-bar truss subject to stress, deflection, and buckling constraints. A scheme of this problem is depicted in Figure 5. The truss focused on this problem has three cross-section areas denoted by $A1$, $A2$, and $A3$, out of which $A1$ is just equal to $A3$. Thus, there are only two areas to be optimized, which are denoted by x_1 and x_2 as the decision variables in the problem, formulation are illustrated below.

$$\text{Minimize}\, f\left(\overrightarrow{x}\right) = \left(2\sqrt{2}x_1 + x_2\right) \times l \tag{33}$$

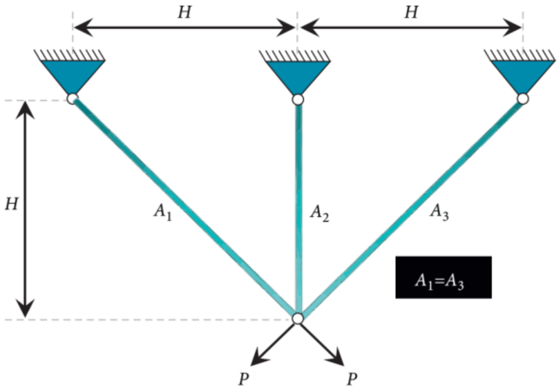

Figure 5. Three-bar truss structure [42].

Subject to :

$$g_1\left(\overrightarrow{x}\right) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0 \tag{34}$$

$$g_2\left(\overrightarrow{x}\right) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0 \tag{35}$$

$$g_3\left(\overrightarrow{x}\right) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0 \tag{36}$$

$$0 \leq x_1,\ x_2 \leq 1 \tag{37}$$

where

$$l = 100\ \text{cm},\ P = 2\frac{KN}{\text{cm}^2},\ \sigma = 2\frac{KN}{\text{cm}^2} \tag{38}$$

Table 10 shows the results of numerous algorithms attempting to solve this problem. As can be seen in the table, the results found by the first five algorithms in the table violate the constraint G_1. Hence, their optimal objective function values are overridden. All the constraints violated are underlined in the table. Among the remaining algorithms, the proposed DB-GWO-EPD offers the best objective value.

Table 10. Results of solving three-bar truss design problem.

Algorithm	x_1	x_2	g_1	g_2	g_3	f_{min}
DEDS [54]	0.788675	0.408248	$\underline{1.777971 \times 10^{-8}}$	−1.464102	−0.535898	263.895841
SSA [55]	0.788665	0.408276	$\underline{1.247906 \times 10^{-8}}$	−1.464070	−0.535930	263.895842
MBA [56]	0.788565	0.408560	$\underline{1.418869 \times 10^{-7}}$	−1.463748	−0.536252	263.895834
PSO-DE [57]	0.788675	0.408248	$\underline{1.427175 \times 10^{-7}}$	−1.464102	−0.535898	263.895825
Tsa [58]	0.788000	0.408000	$\underline{1.636731 \times 10^{-3}}$	−1.463566	−0.534798	263.680057
Rai and Saini [59]	0.795000	0.395000	-3.375515×10^{-3}	−1.480901	−0.522474	264.359956
CS [60]	0.788670	0.409020	-5.733901×10^{-4}	−1.463512	−0.537062	263.971562
MFO [61]	0.788245	0.409467	-1.190244×10^{-9}	−1.462717	−0.537283	263.895980
AOA [42]	0.793690	0.394260	-1.103007×10^{-5}	−1.480113	−0.519898	263.915432
GWO	0.788398	0.409034	-8.468448×10^{-7}	−1.463210	−0.536791	263.896012
FB-GWO-EPD	0.788911	0.407583	-5.171678×10^{-7}	−1.464858	−0.535143	263.895952
DB-GWO-EPD	**0.788539**	**0.408634**	$\mathbf{-5.484502 \times 10^{-14}}$	**−1.463663**	**−0.536337**	**263.895857**

4.4.3. Cantilever Beam Design Problem

The goal of this problem is the minimization of the weight of a cantilever beam made of five hollow blocks with constant thicknesses and variable heights, as depicted in Figure 6. Thus, the blocks' heights are the decision variables to be optimized, denoted by x_1 to x_5. The problem formulation is as follows.

$$\text{Minimize} f\left(\vec{x}\right) = 0.06224(x_1 + x_2 + x_3 + x_4 + x_5) \tag{39}$$

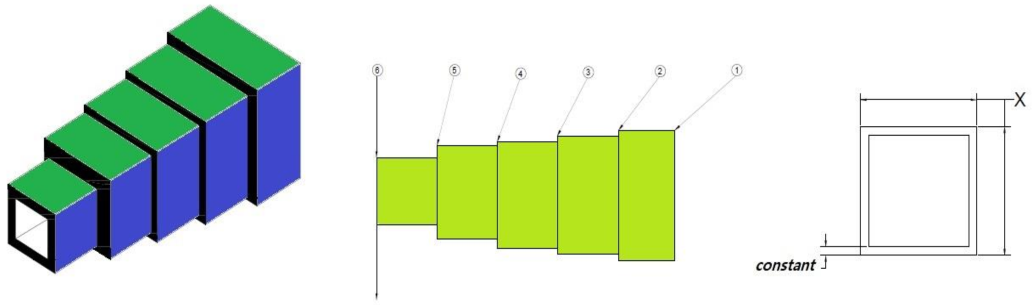

Figure 6. Cantilever beam problem [44]. *"Reprinted from Publication Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer, vol. 191, 1–32, Copyright (2022), with permission from Elsevier."*

Subject to :

$$g\left(\vec{x}\right) = \frac{61}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0 \tag{40}$$

$$0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100 \tag{41}$$

As the results in Table 11 suggest, both the DB-GWO-EPD and GWO reach the lowest weight of the cantilever beam compared to those of their competitors.

Table 11. Results of solving cantilever beam design problem.

Algorithm	x_1	x_2	x_3	x_4	x_5	f_{min}
MFO [61]	5.98487	5.31673	4.49733	3.51362	2.16162	1.33999
SOS [62]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
CS [60]	6.00890	5.30490	4.50230	3.50770	2.15040	1.33999
MMA [63]	6.01000	5.30000	4.49000	3.49000	2.15000	1.34000
GCA1 [63]	6.01000	5.30000	4.49000	3.49000	2.15000	1.34000
GCA2 [63]	6.01000	5.30000	4.49000	3.49000	2.15000	1.34000
GWO	**6.02059**	**5.31527**	**4.48660**	**3.50436**	**2.14698**	**1.33653**
FB-GWO-EPD	6.00974	5.31982	4.48908	3.51235	2.14305	1.33654
DB-GWO-EPD	**6.00305**	**5.30003**	**4.50193**	**3.51318**	**2.15568**	**1.33653**

4.4.4. Gas Transmission Compressor Design Problem

This problem is designed to minimize the cost of delivering 100 million cft gas daily. The decision variables X_1, X_2, and X_3 are the gas transmission parameters, including the distance between the two compressors, compression ratio value, and value of the gas pipe inside diameter. A scheme of this problem is shown in Figure 7. The problem formulation is presented as follows.

$$\text{Minimize}\, f\left(\overrightarrow{x}\right) = 3.69 \times 10^4 \times x_3 + 7.72 \times 10^8 x_1^{-1} x_2^{0.219} - 765.43 \times 10^6 \times x_1^{-1} + 8.61 \times 10^5 \times x_1^{\frac{1}{2}} x_2 \left(x_2^2 - 1\right)^{-\frac{1}{2}} x_3^{-\frac{2}{3}} \quad (42)$$

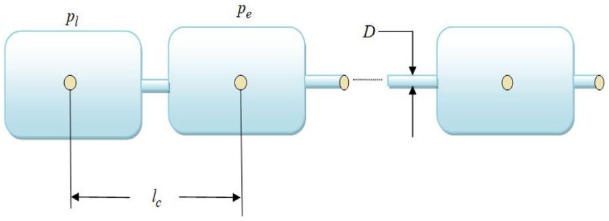

Figure 7. Gas transmission compressor design problem [64]. *"Reprinted from Publication Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer, vol. 191, 1–32, Copyright (2022), with permission from Elsevier."*

Subject to :

$$10 \le x_1 \le 55 \quad (43)$$

$$1.1 \le x_2 \le 2 \quad (44)$$

$$10 \le x_3 \le 40 \quad (45)$$

As seen in Table 12, the proposal can calculate the least value for the gas transmission cost, stressing the high capabilities of the DB-GWO-EPD to win such a highly close competition in solving this design problem.

Table 12. Results of solving gas transmission compressor design problem.

Algorithm	x_1	x_2	x_3	f_{min}
ECS-AGQPSO [64]	53.446716	1.190101	24.718579	2,964,375.495330
RCSOMGA [65]	53.446827	1.190101	24.718580	2,964,375.495330
SOMA [65]	53.347298	1.190142	24.737115	2,964,378.729000
RCGA [65]	53.520217	1.190361	24.723656	2,964,375.725000
PSO [66]	55.000000	1.195410	24.774900	2,964,460.000000
DE [66]	51.985700	1.183350	24.719500	2,964,480.000000
DE-PSO [66]	53.447400	1.190100	24.718600	2,964,375.503101
GP [67]	52.600000	1.187000	24.800000	2,964,419.625000
GWO	**53.444416**	**1.190075**	**24.719897**	**2,964,375.499725**
FB-GWO-EPD	53.440728	1.190088	24.717434	2,964,375.498014
DB-GWO-EPD	**53.446720**	**1.190101**	**24.718583**	**2,964,375.495329**

5. Conclusions

This paper introduced a novel EPD operator for improving the performance of the Grey Wolf Optimization (GWO) algorithm. In the main version of EPD, half of the worst search agents are eliminated from the search space and repositioned around the best ones to evolve the worst individuals, while in our proposed EPD method, the best agents are eliminated and repositioned around the most diversified agents. The main reason justifying this new approach to applying EPD is the low-diversity nature of the best agents in the search space. Indeed, in any meta-heuristic, the best agents are gradually gathered around each other, such that the region the best individuals are placed in is becoming increasingly closed, and thus the diversity could be soon lost in the search space. Losing diversity could contribute to premature convergence. Hence, in the proposed Diversity-Based EPD (DB-EPD) mechanism, the enhanced diversity plays a major role in conducting the other search agents and accomplishing the exploration phase. It is worth mentioning that the three best-fitted alpha, beta, and delta wolves are always kept in the algorithm's memory and appointed to guide the other wolves (agents), even if they are eliminated from the current population. In this paper, the proposed approach was applied to the GWO, and so the resulting algorithm was named DB-GWO-EPD.

The results of applying the proposed DB-GWO-EPD and its competitors to the high-dimensional shifted classical benchmark problems, as well as the CEC2017 challenging test bed, indicated that the proposed algorithm is highly superior to the original GWO and the GWO equipped with the EPD operator in its original definition, named Fitness-Based GWO-EPD (FB-GWO-EPD), as well as four other newly proposed optimization algorithms. The superior performance of the proposed DB-GWO-EPD algorithm is mainly due to its ability to desirably maintain diversity in the search space during the optimization process, helping the proposal to find as many candidate solutions in the search space as possible. This ability could help the proposed algorithm rapidly reach the single optimum in uni-modal functions while impeding the proposal from being trapped in local optima to explore as many optima as possible in the search space to find the fittest solution of the optimization problem finally.

In future work, we aim to apply the DB-EPD operator to other meta-heuristics to validate the proposed approach further to use such an EPD. The application of the DB-GWO-EPD in solving the more recently created and more complicated problems could be another field of interest through which the merits or shortcomings of the proposed algorithm may be better highlighted.

Author Contributions: Conceptualization, F.R. and M.A.E.; methodology, F.R.; software, F.R.; formal analysis, F.R., M.A.E., L.A. and S.M.; investigation, F.R., H.R.S., M.A.E., L.A. and S.M.; resources, L.A. and A.H.G.; data curation, F.R.; writing—original draft preparation, F.R., M.A.E., L.A. and S.M.; writing—review and editing, F.R., H.R.S., M.A.E., L.A. and S.M.; visualization, F.R., M.A.E., L.A., A.H.G. and S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data is available upon request from the corresponding author.

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

Nomenclature

$\overrightarrow{a}$	Acceleration coefficient varying from 2 to 0
$\overrightarrow{A}$	Acceleration factor randomly generated for every leading wolf
$\overrightarrow{C}$	Multiplier of the leading wolves that are randomly generated (Equations (1) and (2))
$\overrightarrow{D}$	Distance between each wolf and the leading wolves (Equations (1) and (2))
$\overrightarrow{A_i}$	Acceleration factor randomly generated for ith leading wolf (Equations (6)–(8))
$\overrightarrow{C_i}$	Multiplier of the ith leading wolf that is randomly generated (Equations (3)–(5))

$\overrightarrow{D_i}$	Distance between each wolf and the leading wolves (Equations (6)–(8))
$\overrightarrow{X_p}(t)$	Prey position at the tth iteration (Equation (2))
$\overrightarrow{X}(t)$	Position of a wolf (solution) at the tth iteration (Equation (1))
$\overrightarrow{X}(t+1)$	Position of a wolf (solution) at the (t + 1)th iteration
$\overrightarrow{X_\alpha}$	Position of the alpha wolf (solution) (Equations (3)–(8))
$\overrightarrow{X_\beta}$	Position of the beta wolf (solution) (Equations (3)–(8))
$\overrightarrow{X_\delta}$	Position of the delta wolf (solution) (Equations (3)–(8))
$\overrightarrow{D_\alpha}$	Distance between each wolf and the alpha wolf (Equations (6)–(8))
$\overrightarrow{D_\beta}$	Distance between each wolf and the beta wolf (Equations (6)–(8))
$\overrightarrow{D_\delta}$	Distance between each wolf and the delta wolf (Equations (6)–(8))
$\overrightarrow{ub}$	Upper bound vector of the decision variables (Equations (10)–(13))
$\overrightarrow{lb}$	Lower bound vector of the decision variables (Equations (10)–(13))
$f^i(X)$	Fitness function value of the ith solution (i = 1, 2, . . . , N) (Equation (14))
$f^j(X)$	Fitness function value of the jth solution (j $= 1, \ldots, i-1, i+1, \ldots, N$) (Equation (14))
d_i	Diversity index for the ith solution (Equation (14))
$\overrightarrow{X_\alpha^{div}}(t)$	The first (alpha) most diversified wolf (solution) at the tth iteration (Equations (15)–(17))
$\overrightarrow{X_\beta^{div}}(t)$	The second (beta) most diversified wolf (solution) at the tth iteration (Equations (15)–(17))
$\overrightarrow{X_\delta^{div}}(t)$	The third (delta) most diversified wolf (solution) at the tth iteration (Equations (15)–(17))
$\overrightarrow{r_i}$	ith uniformly distributed random number generated for giving a random position

References

1. Holland, J.H. Genetic Algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
2. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
3. Storn, R.; Price, K. Differential Evolution: A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
4. Mete, Ç.; Karaboğa, D.; Köylü, F. Artificial bee colony data miner (ABC-Miner). In Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications, Istanbul, Turkey, 15–18 June 2011.
5. Gandomi, A.H.; Yang, X.S.; Talatahari, S.; Alavi, A.H. Firefly algorithm with chaos. *Commun. Nonlinear Sci. Numer. Simul.* **2013**, *18*, 89–98. [CrossRef]
6. Yang, X.S. A new metaheuristic Bat-inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Studies in Computational Intelligence; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74. [CrossRef]
7. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
8. Ahmed, K.; Ewees, A.A.; Abd El Aziz, M.; Hassanien, A.E.; Gaber, T.; Tsai, P.-W.; Pan, J.-S. A hybrid krill-ANFIS model for wind speed forecasting. In Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016, Cairo, Egypt, 24–26 November 2016; Springer: Cham, Switzerland, 2017. [CrossRef]
9. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
10. Wang, G.G.; Deb, S.; Cui, Z. Monarch butterfly optimization. *Neural Comput. Appl.* **2015**, *31*, 1995–2014. [CrossRef]
11. Karimkashi, S.; Kishk, A.A. Invasive Weed Optimization and its Features in Electromagnetics. *IEEE Trans. Antennas Propag.* **2010**, *58*, 1269–1278. [CrossRef]
12. Tuba, M.; Bacanin, N. Artificial Bee Colony Algorithm Hybridized with Firefly Algorithm for Cardinality Constrained Mean-Variance Portfolio Selection Problem. *Appl. Math. Inf. Sci.* **2014**, *8*, 2831–2844. [CrossRef]
13. Wang, G.-G.; Gandomi, A.H.; Zhao, X.; Chu, H.C.E. Hybridizing harmony search algorithm with cuckoo search for global numerical optimization. *Soft Comput.* **2016**, *20*, 273–285. [CrossRef]
14. Yi, W.; Gao, L.; Li, X.; Zhou, Y. A new differential evolution algorithm with a hybrid mutation operator and self-adapting control parameters for global optimization problems. *Appl. Intell.* **2015**, *42*, 642–660. [CrossRef]
15. Tuba, M.; Bacanin, N. Hybridized bat algorithm for multi-objective radio frequency identification (RFID) network planning. In Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC), Sendai, Japan, 25–28 May 2015; pp. 499–506. [CrossRef]
16. Das, P.K.; Behera, H.S.; Panigrahi, B.K. A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning. *Swarm Evol. Comput.* **2016**, *28*, 14–28. [CrossRef]

17. Abualigah, L.M.; Khader, A.T.; Hanandeh, E.S.; Gandomi, A.H. A novel hybridization strategy for krill herd algorithm applied to clustering techniques. *Appl. Soft Comput.* **2017**, *60*, 423–435. [CrossRef]
18. Nenavath, H.; Jatoth, R.K. Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking. *Appl. Soft Comput.* **2018**, *62*, 1019–1043. [CrossRef]
19. Ghanem, W.A.H.M.; Jantan, A. Hybridizing artificial bee colony with monarch butterfly optimization for numerical optimization problems. *Neural Comput. Appl.* **2018**, *30*, 163–181. [CrossRef]
20. Panda, M.R.; Dutta, S.; Pradhan, S. Hybridizing Invasive Weed Optimization with Firefly Algorithm for Multi-Robot Motion Planning. *Arab. J. Sci. Eng.* **2018**, *43*, 4029–4039. [CrossRef]
21. Singh, N.; Singh, S.B. A novel hybrid GWO-SCA approach for optimization problems. *Eng. Sci. Technol. Int. J.* **2017**, *20*, 1586–1601. [CrossRef]
22. Ibrahim, A.M.; Tawhid, M.A. A hybridization of differential evolution and monarch butterfly optimization for solving systems of nonlinear equations. *J. Comput. Des. Eng.* **2019**, *6*, 354–367. [CrossRef]
23. Abualigah, L.M.; Khader, A.T.; Hanandeh, E.S. Modified Krill Herd Algorithm for Global Numerical Optimization Problems. In *Advances in Nature-Inspired Computing and Applications*; Springer: Cham, Switzerland, 2019; pp. 205–221. [CrossRef]
24. Gupta, S.; Deep, K. Hybrid sine cosine artificial bee colony algorithm for global optimization and image segmentation. *Neural Comput. Appl.* **2020**, *32*, 9521–9543. [CrossRef]
25. Gupta, S.; Deep, K. Enhanced leadership-inspired grey wolf optimizer for global optimization problems. *Eng. Comput.* **2020**, *36*, 1777–1800. [CrossRef]
26. Gupta, S.; Deep, K. A novel hybrid sine cosine algorithm for global optimization and its application to train multilayer perceptrons. *Appl. Intell.* **2020**, *50*, 993–1026. [CrossRef]
27. Mohammed, H.; Rashid, T. A novel hybrid GWO with WOA for global numerical optimization and solving pressure vessel design. *Neural Comput. Appl.* **2020**, *32*, 14701–14718. [CrossRef]
28. Zheng, R.; Jia, H.; Abualigah, L.; Liu, Q.; Wang, S. Deep Ensemble of Slime Mold Algorithm and Arithmetic Optimization Algorithm for Global Optimization. *Processes* **2021**, *9*, 1774. [CrossRef]
29. Rezaei, F.; Safavi, H.R.; Abd Elaziz, M.; El-Sappagh, S.H.A.; Al-Betar, M.A.; Abuhmed, T. An Enhanced Grey Wolf Optimizer with a Velocity-Aided Global Search Mechanism. *Mathematics* **2022**, *10*, 351. [CrossRef]
30. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S.; Oliva, D. Hybridizing of Whale and Moth-Flame Optimization Algorithms to Solve Diverse Scales of Optimal Power Flow Problem. *Electronics* **2022**, *11*, 831. [CrossRef]
31. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [CrossRef]
32. Lewis, A.; Mostaghim, S.; Randall, M. Evolutionary Population Dynamics and Multi-Objective Optimisation Problems. In *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*; IGI Global: Hershey, PA, USA, 2008; pp. 185–206. [CrossRef]
33. Bak, P. *How Nature Works: The Science of Self-Organized Criticality*; Oxford University Press: Oxford, UK, 1997.
34. Boettcher, S.; Percus, A.G. Extremal optimization: Methods derived from co-evolution. *arXiv* **1999**, arXiv:math/9904056. [CrossRef]
35. Saremi, S.; Mirjalili, S.Z.; Mirjalili, S.M. Evolutionary population dynamics and grey wolf optimizer. *Neural Comput. Appl.* **2015**, *26*, 1257–1263. [CrossRef]
36. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
37. Yao, X.; Liu, Y.; Lin, G. Evolutionary Programming Made Faster. *IEEE Trans. Evol. Comput.* **1999**, *3*, 82–102.
38. Wu, G.; Mallipeddi, R.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*; National University of Defense Technology: Changsha, China, 2017.
39. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
40. Abualigah, L.; Yousri, D.; Abd Elaziz, M.; Ewees, A.A.; Al-qaness, M.A.A.; Gandomi, A.H. Aquila Optimizer: A novel meta-heuristic optimization algorithm. *Comput. Ind. Eng.* **2021**, *157*, 107250. [CrossRef]
41. Karami, H.; Anaraki, M.V.; Farzin, S.; Mirjalili, S. Flow Direction Algorithm (FDA): A Novel Optimization Approach for Solving Optimization Problems. *Comput. Ind. Eng.* **2021**, *156*, 107224. [CrossRef]
42. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The Arithmetic Optimization Algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
43. Ahmadianfar, I.; Bozorg-Haddad, O.; Chu, X. Gradient-based optimizer: A new metaheuristic optimization algorithm. *Inf. Sci.* **2020**, *540*, 131–159. [CrossRef]
44. Abualigah, L.; Elaziz, M.A.; Sumari, P.; Geem, Z.W.; Gandomi, A.H. Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer. *Expert Syst. Appl.* **2021**, *191*, 116158. [CrossRef]
45. Ragsdell, K.M.; Phillips, D.T. Optimal Design of a Class of Welded Structures Using Geometric Programming. *J. Eng. Ind.* **1976**, *98*, 1021–1025. [CrossRef]
46. Deb, K. Optimal design of a welded beam via genetic algorithms. *AIAA J.* **1991**, *29*, 2013–2015. [CrossRef]
47. Lee, K.S.; Geem, Z.W. A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Comput. Methods Appl. Mech. Eng.* **2005**, *194*, 3902–3933. [CrossRef]

48. Huang, F.-Z.; Wang, L.; He, Q. An effective co-evolutionary differential evolution for constrained optimization. *Appl. Math. Comput.* **2007**, *186*, 340–356. [CrossRef]
49. He, Q.; Wang, L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Eng. Appl. Artif. Intell.* **2007**, *20*, 89–99. [CrossRef]
50. Kaveh, A.; Khayatazad, M. A new meta-heuristic method: Ray Optimization. *Comput. Struct.* **2012**, *112–113*, 283–294. [CrossRef]
51. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
52. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-Verse Optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [CrossRef]
53. Abd Elaziz, M.; Oliva, D.; Xiong, S. An improved Opposition-Based Sine Cosine Algorithm for global optimization. *Expert Syst. Appl.* **2017**, *90*, 484–500. [CrossRef]
54. Zhang, M.; Luo, W.; Wang, X. Differential evolution with dynamic stochastic selection for constrained optimization. *Inf. Sci.* **2008**, *178*, 3043–3074. [CrossRef]
55. 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.* **2017**, *114*, 163–191. [CrossRef]
56. Sadollah, A.; Bahreininejad, A.; Eskandar, H.; Hamdi, M. Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Appl. Soft Comput.* **2013**, *13*, 2592–2612. [CrossRef]
57. Liu, H.; Cai, Z.; Wang, Y. Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Appl. Soft Comput.* **2010**, *10*, 629–640. [CrossRef]
58. Tsai, J.-F. Global optimization of nonlinear fractional programming problems in engineering design. *Eng. Optim.* **2005**, *37*, 399–409. [CrossRef]
59. Ray, T.; Saini, P. Engineering Design Optimization Using a Swarm with an Intelligent Information Sharing Among Individuals. *Eng. Optim.* **2001**, *33*, 735–748. [CrossRef]
60. Gandomi, A.H.; Yang, X.-S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Eng. Comput.* **2013**, *29*, 17–35. [CrossRef]
61. Jangir, N.; Pandya, M.H.; Trivedi, I.N.; Bhesdadiya, R.H.; Jangir, P.; Kumar, A. Moth-Flame optimization Algorithm for solving real challenging constrained engineering optimization problems. In Proceedings of the 2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS), Bhopal, India, 5–6 March 2016. [CrossRef]
62. Cheng, M.-Y.; Prayogo, D. Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Comput. Struct.* **2014**, *139*, 98–112. [CrossRef]
63. Mortazavi, A. Interactive fuzzy Bayesian search algorithm: A new reinforced swarm intelligence tested on engineering and mathematical optimization problems. *Expert Syst. Appl.* **2022**, *187*, 115954. [CrossRef]
64. Kumar, N.; Mahato, S.K.; Bhunia, A.K. Design of an efficient hybridized CS-PSO algorithm and its applications for solving constrained and bound constrained structural engineering design problems. *Results Control Optim.* **2021**, *5*, 100064. [CrossRef]
65. Duary, A.; Rahman, M.S.; Shaikh, A.A.; Niaki, S.T.A.; Bhunia, A.K. A new hybrid algorithm to solve bound-constrained nonlinear optimization problems. *Neural Comput. Appl.* **2020**, *32*, 12427–12452. [CrossRef]
66. Pant, M.; Thangaraj, R.; Abraham, A. DE-PSO: A new hybrid meta-heuristic for solving global optimization problems. *New Math. Nat. Comput.* **2011**, *7*, 363–381. [CrossRef]
67. Beightler, C.S.; Phillips, D.T. *Applied Geometric Programming*; Wiley: Hoboken, NJ, USA, 1976.