Future Generation Computer Systems 136 (2022) 128–152

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

GOZDE: A novel metaheuristic algorithm for global optimization

Yiğit Çağatay Kuyu *, Fahri Vatansever

Electrical-Electronics Engineering Department, Bursa Uludağ University, Bursa, Turkiye

ARTICLE INFO

Article history:
Received 31 January 2022
Received in revised form 11 May 2022
Accepted 28 May 2022
Available online 6 June 2022

Keywords:
Evolutionary computation
Global optimization
Real-world problems
Metaheuristic algorithms
Population-based method
GOZDE

ABSTRACT

This study proposes a new metaheuristic algorithm, called "Geometric Octal Zones Distance Estimation" (GOZDE) algorithm to solve global optimization problems. The presented GOZDE employs a search scheme with the information sharing between the zones considering the distance of the zones utilizing median values. The whole population represents the eight zones that are the combination of different search strategies to guide knowledge dissemination from one zone to others in the search space. To demonstrate the effectiveness of the proposed optimizer, it is compared with two classes of metaheuristics, which are (1) GA, PSO, DE, CS and HS as the classical metaheuristics and (2) BWO, SSA, MVO, HHO, ChOA, AOA and EBOwithCMAR as the up-to-date metaheuristics. The search capability of the proposed algorithm is tested on two different numerical benchmark sets including low and high dimensional problems. The developed algorithm is also adapted to ten real-world applications to handle constraint optimization problems. In addition, to further analyse the results of the proposed algorithm, three well-known statistical metrics, Friedman, Wilcoxon rank-sum and Whisker-Box statistical tests are conducted. The experimental results statistically show that GOZDE is significantly better than, or at least comparable to the twelve metaheuristic algorithms with outstanding performance in solving numerical functions and real-world optimization problems.

1. Introduction

Optimization can be identified as the process of finding the best possible solutions of a function or functions, which can be done with maximization or minimization. Since numerical and engineering optimization problems have been getting more complicated day by day, researchers have been creating more talented optimization algorithms to provide a better approach to the problems. Traditional algorithms, such as gradient-based and quadratic programming, cannot provide a good solution for complex multimodal and high dimensional problems, and may be stuck at a local minimum when the derivative is zero, especially real-world scenario whose search space is unknown. Therefore, metaheuristic algorithms (MAs) are perfect alternative to the traditional algorithms as they are gradient-free, randomize, easy to use, and have an ability to escape from local minima [1]. In general, in the optimization process, firstly, the parameters of MA are identified (maximum iteration, number of function evaluation, population size, etc.) and objective function is determined. Afterwards, the search process begins with randomly generated candidate solutions, which will be updated to create next generations. Finally, an effective MA achieves an optimal solution considering constraints and problem types (continuous, discrete, multi-objective, etc.).

The MAs can be classified according to their inspirations and search procedures. Evolutionary-based MAs are inspired from biology, such as the genetic [2] and differential evolution [3] algorithms. Swarm intelligence-based techniques are inspired from the behaviour of social insects or animals, such as the particle swarm optimization [4] and artificial bee colony [5] algorithms. The rules governing a natural phenomenon, such as the gravitational search [6] and the multiverse [7] algorithms inspire physics or chemistry-based methods. Human-based MAs are inspired from the various behaviours of human beings. MAs can begin the search process with a single solution or a population. The single solution-based methods execute the search operations by generating a solution at a time and update this solution until a stopping criterion is reached. This search process can be characterized as a trajectory in the search space. Since these algorithms generate just one solution at a time, the search will likely be stuck at a local minimum. The tabu search (TS) [8], simulated annealing (SA) [9] and iterated local search (ILS) [10] are the popular members of this category. In population-based methods, as opposite to single solution-based methods, the search process is performed by using the evolution of a set of solutions. The basic summary of the classification of the MAs can be seen in Fig. 1.

In the optimization process of the population-based metaheuristics, the initial candidate solutions constitute a population which represent starting points in the search space, and then based on the search mechanism of the algorithm, the initial

* Corresponding author.
E-mail address: yigitkuyu@uludag.edu.tr (Y.Ç. Kuyu).

https://doi.org/10.1016/j.future.2022.05.022
0167-739X/

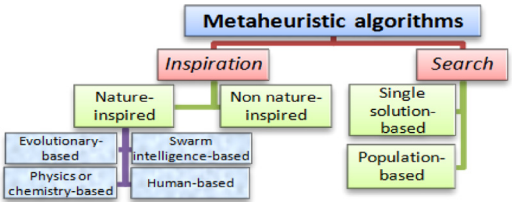


Fig. 1. The different classes of metaheuristics.

population is updated to create a new set of solutions by using some of the randomize process which enables the algorithm to generate several solutions in one iteration, and allows the algorithm to avoid local minima in the search space at each run [11]. Many population-based metaheuristic algorithms are frequently employed to solve complex numerical functions and real-world optimization problems. The genetic algorithm (GA) [2], differential evolution algorithm (DE) [3] and particle swarm optimization algorithm (PSO) [4] are the popular examples of the population-based algorithms. GA simulates natural selection and reproduction processes based on Darwinian phenomena. In GA, the population is a set of chromosomes and the genetic operators are identified to generate new candidate solutions. On the other hand, DE uses similar operators as GA used, which are the mutation, crossover and selection, in the optimization process. Mutation and crossover operators are used to discover new regions in the search space, whereas selection operator is applied to keep the better individual. PSO is inspired by the social relationship of swarm (fish, birds, bees, etc.). The candidate solutions are called as particles which are updated to fly toward new positions, and the fitness function is used to determine new search directions of the particles.

At present, in addition to the GA, PSO and DE algorithms, various researches have been conducted in the field of population-based metaheuristics as the complexity of the optimization problems are increased. In 2001, harmony search (HS) [12] inspired by the process of music improvisation was developed. In its search process, new harmonies belonging to HS are created via the pitch adjustment, memory consideration, random-based process and greed selection that is used to choose better harmonies. In 2005, artificial bee colony algorithm (ABC) [5] which mimics the cooperative foraging behaviour of honey bees was introduced. In ABC, the population consists of the bees which are classified into three groups (employed and scout bees, and onlookers). Other bees to achieve optimal solution (maximum nectar amount) follow the bee, which finds the best food source among all most likely determined search directions. In 2009, the gravitational search algorithm (GSA) [6] was introduced by inspiring the laws of gravity and mass interactions. The gravitational force to change particle motion direction performs this algorithm. The particle position and velocity are often updated due to gravitational force among them. In 2015, the moth-flame optimization algorithm (MFO) [13] inspired by the navigation manner of moths was proposed. In this algorithm, moths tend to maintain a fixed angle with respect to the moon. When the months see an artificial light, they try to maintain a similar angle with the light that causes a spiral fly path of months to achieve better solution (position). In 2016, the multiverse optimizer (MVO) [7] inspired by the theory of multi-verse in physics that simulates white, black and worm holes mathematically was developed. Population represented by universes is sorted with respect to the inflation rate and uses the roulette wheel to choose a universe with a white hole. While white and black holes are used to explore search space, wormholes are used to help to exploit the solutions. In 2017, the salp swarm algorithm (SSA) [14] inspired by the swarming behaviour of salps in oceans was proposed. The whole population is divided into two portions. One is the leader which guides to other salps, the other is the followers which move along other salps and follow the leader. The salps' positions in the chain are updated to reach optimal food sources. In 2019, the Harris hawks optimization algorithm (HHO) [15], which mimics the cooperative behaviour of Harris' hawks, was developed. During the exploration phase, the hawks (candidate solutions) are randomly on some locations for monitoring various regions. In the exploitation phase, the hawks attempt to surprise pounce to exploit the neighbourhood of intended prey. During the optimization process, the algorithm tries to keep maintaining a balance between exploration and exploitation phases. In 2021, the arithmetic optimization algorithm (AOA) [16] utilized by the distribution behaviour of the main arithmetic operators in mathematics was presented. The proposed algorithm mainly consists of three phases, which are initialization, exploration and exploitation. The optimization process begins with a set of candidate solutions in the initialization. The exploration search mechanism in AOA refers to division and multiplication operators, whereas the exploitation search mechanism uses subtraction and addition operators. There are many studies found in the literature related to the optimization and the solution methods of the optimization problems. Interested readers can refer to [17–20].

Since all the existing metaheuristics have their own pros and cons, and can have differences from one problem to another in their performances, there is always an open door for new metaheuristic algorithms. Moreover, as random factors are highly effective in the search mechanisms of the MAs, none of them can always guarantee to reach the global optima for each optimization problem. In addition to that, No Free Lunch (NFL) theorem [21] says that there is no optimization algorithm to solve every kind of optimization problem. For these reasons, to provide more versatile and efficient optimizer, a new algorithm, called "Geometric Octal Zones Distance Estimation" (GOZDE), is proposed in this paper. The motivation behind designing such a concept is to present a well-designed algorithm dedicated to lie in the fact that interactions among different sole subpopulations can afford much better income as compared with an individual optimizer, using complementary advantages of the octal zones. We said, "more versatile" that is because this algorithm is based on dividing the population into octal zones, and each zone is updated according to its own unique processes, this means eight different processes try to improve solutions in parallel. Information sharing mechanism is also utilized while each zone is working together. The reason behind this idea is to create a potential to locate the various subpopulations into different regions of search space, which is helpful for maintaining the overall diversity of the whole population, and is to get closer to an optimal solution step by step.

The major contributions of this study are summarized as follows:

i. A novel population-based metaheuristic algorithm, namely GOZDE, is proposed, which motivates from the distances of zones to each other.
ii. To the best of our knowledge, it is the first time that eight different zones (subpopulations) are executed in parallel in the field of the global optimization. This collaborative synergy is to make it easy to search for different regions in the search space, which can enable the proposal to evade from local minima.
iii. The major difference between the current work and the recently published papers is that the zones are able to determine their movement directions simultaneously and update the current knowledge via a best-so-far solution.
iv. The proposal has only one algorithm-dependent parameter that needs to be adjusted, which gains simplicity to the proposed algorithm.

v. A detailed study on the performance of the GOZDE algorithm over the standard test suites and real-world problems has been implemented to shed more light on its performance in different search space belonging to these challenging problems.

Mainly, the differences between our research and previous studies are in two aspects. First, the multi-population approach of the proposed algorithm is more sophisticated. Many subpopulations are adapted to handle numerical and real-world problems, where a search space partitioning mechanism for subpopulations is proposed to generate non-isolated sub domains of search space. Second, the proposal is structured for parallel computing, where multiple evolutionary strategies are implemented via subpopulations that use unique strategies and different control parameters to improve solutions. The efficiency of the GOZDE algorithm has been proven with different class test functions including unimodal, multimodal, composition and hybrid. 10 complex real-world problems are also integrated into comparisons. The performance of the proposed algorithm is compared with five classical (GA [6], PSO [7], DE [8], cuckoo search algorithm (CS) [22], HS [9]) and five up-to-date metaheuristics (black widow optimization algorithm (BWO) [23], SSA [14], MVO [13], HHO [15], chimp optimization algorithm (ChOA) [24]) numerically and statistically. To increase the level of difficulty of the comparisons, the winner of 2017 IEEE Congress on Evolutionary Computation (CEC), the effective butterfly optimizer with covariance matrix adapted retreat phase algorithm (EBOwithCMAR) [25], and the recently proposed arithmetic optimization algorithm (AOA) [16] are integrated for the comparisons on hybrid and composition functions. Simulation results demonstrate that GOZDE has a favourable performance compared to classical and state-of-the-art metaheuristic algorithms on both low and high dimensional problems.

The rest of this paper is organized as follows. Section 2 presents the proposed algorithm, and in Section 3 the experimental setup and the comparative performance analysis are given on numerical functions and real-world problems. Concluding remarks are given in Section 4.

2. Proposed algorithm

GOZDE is a population-based metaheuristic algorithm. The underlying concept for the proposed algorithm is performed based on a divide-conquer manner. Divide-and-Conquer approach is a war strategy pronounced firstly for Rome by Julius Caesar (100 BC) as "Divide et Impera" to maintain power by breaking up larger concentrations of power into pieces that individually have less power than the one when strategizing action [26,27]. Motivated by this strategy, the divide and conquer frameworks are adapted to the context of the proposed algorithm. Following the "divide" principle, the search space of problem is divided into many smaller distinctive regions according to fitness values of the individuals, while in the "conquer" methodology, best individuals are recombined in the most promising regions to conquer a solution of the whole search space. As similar to this philosophy, in GOZDE, all solutions (individuals) are sorted from the best to the worst with respect to their fitness values. The population is divided into eight zones according to the sorted individuals. We suppose that the best solution found so far (X_{best}) is a central and $zone_1$, which includes the best performing candidate solutions, is a reference area. The distances of the zones to $zone_1$ are calculated according to the medians of the fitness values of the corresponding zones.

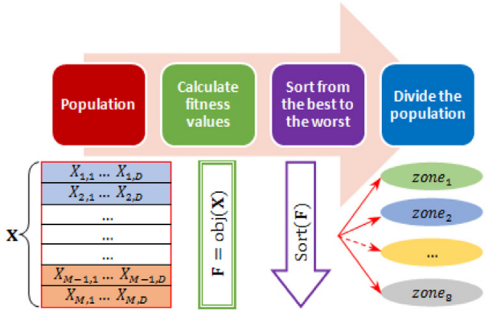


Fig. 2. Population division procedure.

2.1. Initialization

Similar to most of the metaheuristics, the initial solutions are generated with uniform distribution over the search space.

$$X = lb + rand.(ub - lb) \tag{1}$$

lb and ub are the lower and upper bounds of the problem, respectively. $rand$ is a uniform random vector in the range of [0, 1]. The initial population can be represented as follows:

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_M \end{bmatrix}, \quad X_i = \left[X_{i,1}, X_{i,2}, \dots, X_{i,D}\right], \ i = 1, \dots, M \tag{2}$$

where M is the population size and D is the dimension of the problem.

2.2. Population division

Population members are sorted from the best to the worst according to fitness values, and then are divided into eight zones ($zone_i, i = 1, \dots, 8$) (see Fig. 2). If $M \ mod \ 8 = 0$, the number of members in each zone is equal. In other case, the number of members in first seven zones is equal and is equal to $round\,(M/8)$, and the number of members in the eighth zone is $M - 7{*}round\,(M/8)$.

2.3. Updating mechanism of each zone

The term "geometric" in this algorithm reflects the finding of distance in a symbolic way. Firstly, the distance of each zone to each other is calculated by using median values (med) of their fitness values, which can be shown in the following equation:

$$D_{1j} = abs\left\{med\left[F\left(zone_1\right)\right] - med\left[F\left(zone_j\right)\right]\right\} \tag{3}$$

where D_{1j} is the distance of 1th and jth zone, F is the objective function, $j = 2, \dots, 8$. The representation of zones in the search space for a minimization problem is illustrated in Fig. 3, in which the population size is 16, the dimension of the problem is one, x-axis represents zone according to variable values of the individuals which are addressed in dark blue font while y-axis denotes the distances according to medians of fitness values of the individuals.

As the basic philosophy, we use the closest zone ($zone_1$) as a reference area in determining the new individuals of the first two zones. The zones closest the central (X_{best}) have greater coefficient values while the zones farthest to the centre have lower coefficient values. It should be noted that the opposite of the coefficient assignment process mentioned above (the farthest zones have greater coefficient values) has been done but the

$$if\left(rand>\frac{1}{2}\right)\left\{X_{zone_3}(t,d)=X_{best}(t,d)\quad X_{zone_4}(t,d)=X_{best}(t,d)\quad X_{zone_5}(t,d)=X_{best}(t,d)\quad X_{zone_6}(t,d)=X_{best}(t,d)\quad X_{zone_7}(t,d)=X_{best}(t,d)\right. \tag{5}$$

$$else\begin{cases} X_{zone_3}(t,:)=X_{best}(t,:)+\left(\frac{k}{D_{13}{}^i}\right)*\left(\frac{1}{3}\right)*c_{zone_3} \\ X_{zone_4}(t,:)=X_{best}(t,:)+\left(\frac{k}{D_{14}{}^i}\right)*\left(\frac{1}{3}\right)*c_{zone_4} \\ X_{zone_{5_t}}(t,d)=X_{zone_5}(t,d), X_{zone_5}(t,:)=\left\{X_{zone_1}(t,:)-X_{zone_m}(t,:)\right\}\left(\frac{ki}{D_{15}}\right)*\left(\frac{1}{5}\right)*c_{zone_5}, X_{zone_5}(t,d)=X_{zone_{5_t}}(t,d)+X_{zone_5}(t,d) \\ X_{zone_{6_t}}(t,d)=X_{zone_6}(t,d), X_{zone_6}(t,:)=\left\{X_{zone_1}(t,:)-X_{zone_m}(t,:)\right\}\left(\frac{ki}{D_{16}}\right)*\left(\frac{1}{5}\right)*c_{zone_6}, X_{zone_6}(t,d)=X_{zone_{6_t}}(t,d)+X_{zone_6}(t,d) \\ X_{zone_{7_t}}(t,d)=X_{zone_7}(t,d), X_{zone_7}(t,:)=\left\{X_{zone_1}(t,:)-X_{zone_m}(t,:)\right\}+X_{best}(t,:), X_{zone_7}(t,d)=\left\{X_{zone_{7_t}}(t,d)+X_{zone_7}(t,d)\right\}*\left(\frac{1}{D_{17}}\right)*c_{zone_7} \end{cases}$$

Box I.

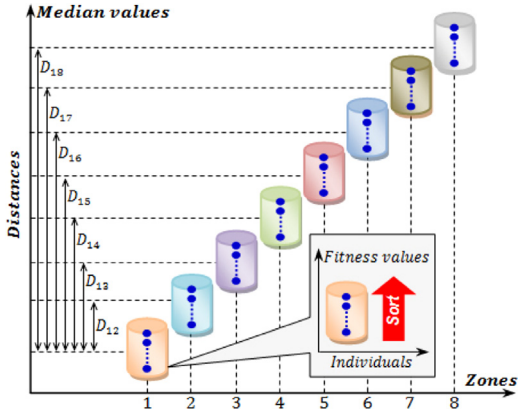


Fig. 3. Representation of the zones' positions with distances.

meaningful results have not been achieved. Each zone is assigned to a random coefficient (c_{zone_i}) in the range of $[-1, 0]$. These greater coefficients are used as a part of step sizes in $zone_1$ and $zone_2$ so as to make a balance between c_{zone_i} and D_{12}. In the early iterations, the inversely proportional D_{12} can be high because the individuals X_{zone_1} and X_{zone_2} cannot be well fit in the search space, and can be far away from each other. Hence, the division of distance D_{12} can avoid big step sizes and can exploit the solutions. By writing the distance between the zones in the form of a division, the zones would have step sizes inversely proportional to the distance. The closest first two zones ($zone_1$ and $zone_2$) are updated as below:

$$\left.\begin{aligned} X_{zone_1}&=\left(X_{zone_1}-X_{zone_2}\right)+\left(\frac{k.t}{D_{12}}\right)c_{zone_1}+X_{zone_1} \\ X_{zone_2}&=\left(X_{zone_1}-X_{zone_2}\right)+\left(\frac{k.t}{D_{12}}\right)c_{zone_2}+X_{zone_2} \end{aligned}\right\} \tag{4}$$

where t is the population index of current zone; k is the random number; c_{zone_1} and c_{zone_2} are the coefficients of the corresponding zones. As can be shown in Eq. (4), $zone_1$ and $zone_2$ search directions are updated by utilizing the distance between $zone_1$ and $zone_2$. Since these zones are located at best-so-far promising regions in the whole search space, the individuals in these zones are exploited to obtain better solutions. On the other hand, $zone_3$ and $zone_4$ are updated according to "if-else" rule. X_{zone_3} and X_{zone_4} determines the new movement directions under the influences of X_{best}, as given in Box I.

In Eq. (5), $d = randi(D)$ is a random integer number; D is the dimension of the problem; X_{best} is the best solution found so far; $X_{zone_j}(t,:)$ represents an individual of $zone_j$ population and $X_{zone_j}(t,d)$ indicates the dth element of X_{zone_j} at individual t. As can be seen in Eq. (5), X_{zone_3} and X_{zone_4} move around X_{best}, and are able to capture beneficial knowledge from X_{best}. These zones contribute to the exploitation behaviours of $zone_1$ and $zone_2$. In addition, the individuals of X_{zone_3} and X_{zone_4} are able to perform independent exploration around X_{best}, and can have long jump due to a wide variety of the range of step size. Regarding X_{zone_5} and X_{zone_6} which are calculated via Eq. (5), they interchange some elements with the best solution, and the update of the search locations of X_{zone_5} and X_{zone_6} is affected by an individual X_{zone_m}. Except for learning from the best individual, this means if the current best solution gets stuck into a local minima, X_{zone_5} and X_{zone_6} are able to reach more promising regions with the help of X_{zone_m}, which increases the opportunity to form better individuals and keep diversity of population. X_{zone_m} is a randomly chosen individual from the $zone_m$; m is a random integer number, in which $m = 1$ represents a best solution and k is a random number in the range of [0,1]. As can be seen in Eq. (5), the individuals of $zone_7$ are updated as are those in $zone_5$ and $zone_6$, but the step size of $zone_7$ is updated differently to these zones, adding X_{best} with the difference vector $\left\{X_{zone_1}-X_{zone_m}\right\}$. This process has two folds. One is that it can contribute more to exploration since $zone_7$ is the one of the worst zones, and the difference vector can make the variation large. The other is that it can contribute to making the solution more accurate due to the probability of $m = 1$, hence, the new generated solution may become closer to an optimal solution. $zone_8$ including the worst solutions is updated below:

$$\left.\begin{aligned} &d_1=rand_{perm}(pn_8), d_2=rand_{perm}(pn_8), \\ &V1_{zone_{8t}}=pop_{zone_8}(d_1,:), V2_{zone_{8t}}=pop_{zone_8}(d_2,:) \\ &X1_{zone_{8t}}=\left\{kV1_{zone_{8t}}(n,:)+(1-k)V1_{zone_{8t}}(n+1,:)\right. \\ &\quad\left.+kV2_{zone_{8t}}(n,:)+(1-k)V2_{zone_{8t}}(n+1,:)\right\}\left(\frac{c_{zone_8}}{D_{18}}\right) \\ &X2_{zone_{8t}}=\left\{kV1_{zone_{8t}}(n+1,:)+(1-k)V1_{zone_{8t}}(n,:)\right. \\ &\quad\left.+kV2_{zone_{8t}}(n+1,:)+(1-k)V2_{zone_{8t}}(n,:)\right\}\left(\frac{c_{zone_8}}{D_{18}}\right) \end{aligned}\right\} \tag{6}$$

where pn_8 is the size of $zone_8$ population; pop_{zone_8} is $zone_8$ population, $rand_{perm}$ randomly permutes of the elements of a given individual and $n = 1, \ldots, pn_8 - 1$. The individual with better objective function value between $X1_{zone_{8t}}$ and $X2_{zone_{8t}}$ attends to the population of $zone_8$, whereas the last individual of $zone_8$ is replaced with the best solution in the previous generation.

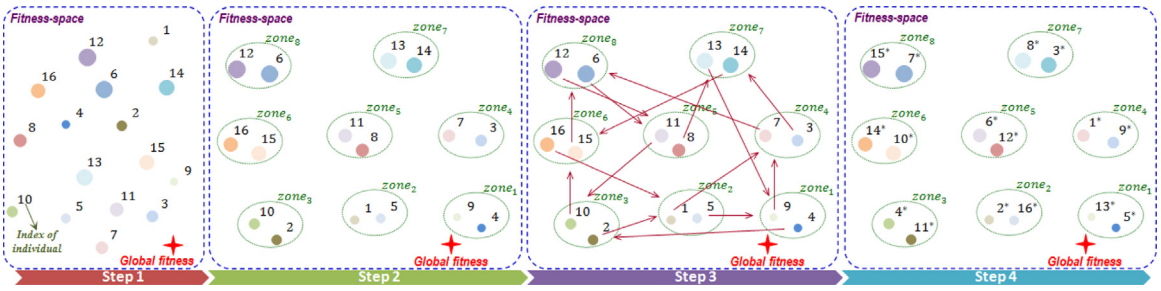


Fig. 4. The fitness-space representation of the steps of the proposed algorithm.

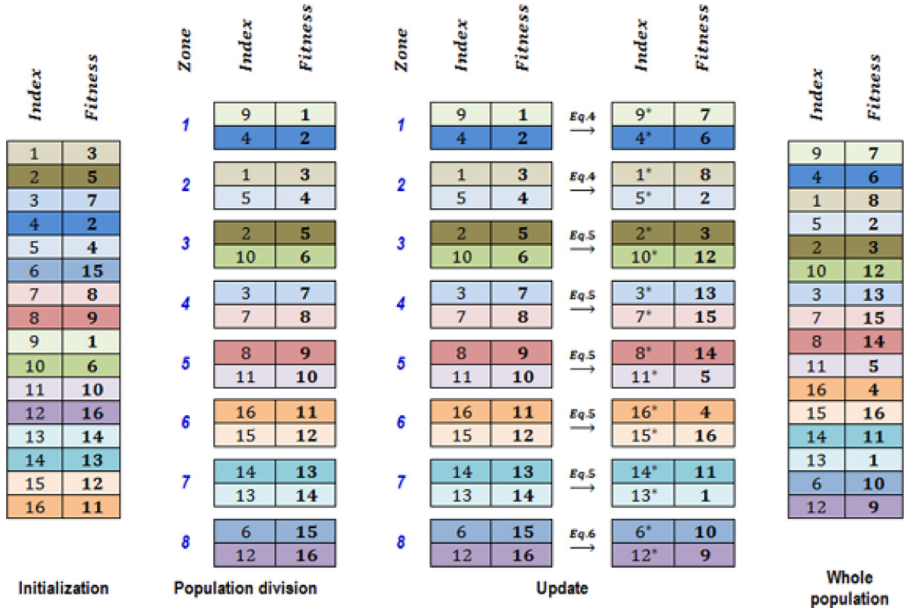


Fig. 5. The population representation of the steps of the proposed algorithm.

The solutions generated via $rand_{perm}$ can promote the exploration of the search space. The reason for applying the random permutation to the individuals of $zone_8$ is that the existence of randomness in all zones can prevent finding a good solution. However, the better movement direction between $X1_{zone_{8t}}$ and $X2_{zone_{8t}}$ can explore potential regions and more productive solutions. At the end of the iteration, the sub-populations are regrouped into one population. The sorting process is then applied to the whole population again, and the individuals of the population are sorted and arranged in order to divide the population into octal zones. This process brings the possibility to exchange solutions between subpopulations in which individuals can undergo different strategies than the previous iteration, in other words, new search directions can be generated, which can help the algorithm escape from local optima. It is worth to point out that the distances of the zones found with respect to median values are scaled according to the quantity of the distances.

Optimization of different types of complex problems such as, multimodal, hybrid and composition, can require multiple strategies since if one strategy does not improve the solution; one of the others is able to enhance the search capability of the algorithm in which it may avoid stagnation and diversity loss scenarios. In addition, due to the ever-changing nature of the problems, it is hard to analytically find which strategy improves on which ability of the algorithm for each problem, but incorporating several different strategies can enable the algorithm to open the door for effective solutions. To show the effectiveness of the proposed octal zones division, we divided all population into 2 and 4 zones in order by using the same improvement mechanisms of the zones in the proposed algorithm. 23 well-known

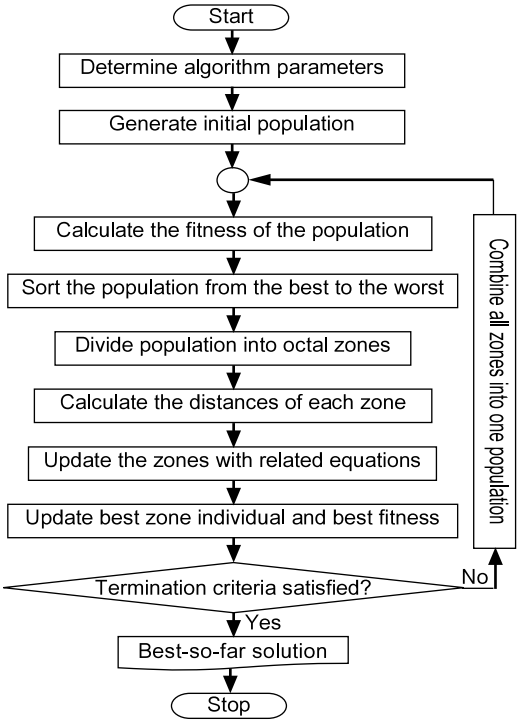


Fig. 6. The flowchart of the proposed algorithm.

numerical functions are used to demonstrate the performances when dividing the population into 2 and 4 zones, respectively, in which the corresponding improvement mechanisms of the zones stay the same as the proposed algorithm, whereas the improvement mechanisms of zones 5, 6, 7, and 8 are removed. The mean and standard deviation (Std) results of the corresponding zone numbers on 30 runs for each function can be seen in Appendix (Table A.7). It is worth mentioning here that selecting the zones as octal is the best performing option as compared to the choice of 2 or 4 zones.

Big-O notation is used for the computational complexity of the presented algorithm. The overall complexity of the given algorithm depends on four processes: initialization, function evaluations, division of zones and updating of individuals. To show the complexity of GOZDE, it is assumed that N is the number total solutions consisted of the solutions of eight zones, D is the number of dimensions, T is the number of iterations and C is the cost of function evaluation.

The computational complexity of initialization process is $O(ND)$. The computational complexity of function evaluations (FE) includes the fitness evaluation of each zone.

$$O\left(FE\right) = O\left(\sum_{i=1}^{8} FE_{zone_i}\right) = O\left(\frac{C7N}{8}\right) + O\left(\frac{C2N}{8}\right) \cong O(CN) \quad (7)$$

The process of the division of zones benefits from Quicksort (QS) algorithm in which has complexity equal to $O(NlogN)$ in the best case; while in the worst case, it is $O(N^2)$. The complexity of the updating procedure is $O(ND)$. Therefore, the total complexity of the proposed algorithm in the best case of QS is $O(ND + T(NlogN + CN + ND))$, whereas in the worst case scenario, the total complexity is $O(ND + T(N^2 + CN + ND))$. It is worth noting here that the usage of the different sorting algorithms can decrease the computational complexity of the proposed algorithm.

To highlight the steps of the proposed algorithm, let us suppose that the number of population is 16 and the obtained fitness values of the individuals are in the range [1,16] for simplicity. The proposed approach has three main steps, including population initialization, population division and population evolution. As can be seen in Fig. 4 in Step 1, the initial population of individuals is scattered over the search space randomly (see Section 2.1). Considering the minimization problem, the fitness values and the indexes of related individuals in the same initial population are illustrated in the initialization in Fig. 5. Following the population division process in Section 2.2, the initial population is divided into the eight zones according to fitness values represented in population division in Figs. 4 and 5, Step 2. Third step represents the update of the individuals. The unique search mechanisms are applied to each zone to generate new population, which is demonstrated in the update in Fig. 5 in which the indexes 1*,2*, ….,16* represent the indexes of new individuals. Fig. 4, Step 3 demonstrates the movements of individuals after updating them. Considering $zone_1$ and $zone_7$, following this figure, the individual with the index of 13 in $zone_7$ is moved toward the best solution, meaning that the mechanism denoted in Eq. (5) assists to improve this individual. Conversely, the individuals in $zone_1$ show no more improvement, and after the update of these individuals via Eq. (4), they move away from the global point. It can be time-consuming to try to apply Eq. (4) over and over to these individuals for improvement. At this point, interchangeable hierarchy between the zones comes into play. Afterwards, the zones constitute the whole population, which can be seen in Fig. 5. As can be seen in Fig. 4, Step 4, which is the objective-space representation of Fig. 5, the individuals in the population will undergo different processes for improvement since they are located at different zones. All processes are then implemented again (from Step 2 to Step 4 in Fig. 4) until the stopping criterion is reached. This hierarchy helps to maintain balance between exploration and exploration in finding the optimal solution and keeping the diversity of the candidate solutions.

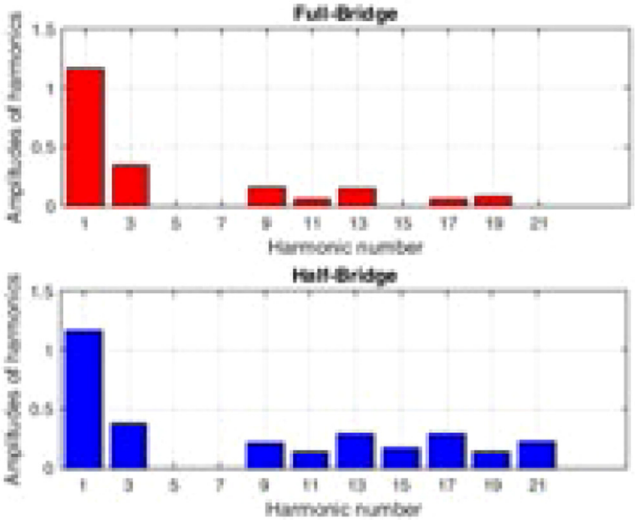


Fig. 7. The harmonics found by the GOZDE algorithm.

The basic steps of GOZDE algorithm can be summarized in the following pseudo code and its flowchart is given in Fig. 6.

Pseudo code of the proposed algorithm

1. Determine population size, coefficient value, maximum iteration, problem dimension-boundaries
2. Generate initial population
3. Calculate fitness values of the population's individuals
4. Sort individuals by fitness values from smallest to largest (best to worst)
5. Group individuals by dividing the sorted population into octal zones
6. While (maximum iteration not exceeded)
   - For each zone
     - Calculate the distances of each zone using Eq. (3)
     - Update $zone_1$ and $zone_2$ using Eq. (4)
     - Update zones from 3 to 7 using Eq. (5)
     - Update $zone_8$ using Eq. (6)

     end for
   - Calculate the fitness values of the individuals in each zone
   - Combine the octal zones into one zone
   - Sort and move the individuals into the corresponding zones
   - Calculate the distances of the zones

     end while
7. Return best-so-far solution

2.4. Parameter sensitivity analysis

Parameter tuning configuration is a crucial part of the process for the metaheuristic algorithms as the suitable parameter choice can result in superior performance and robustness. The only parameter of the GOZDE algorithm is the coefficient vector which is constant. This vector is generated in the predefined

Table 1
The compared metaheuristics in this study.

Genetic Algorithm (GA)	**Particle Swarm Optimization (PSO)**
1. Define initial parameters (population size, mutation probability, crossover probability, etc.) **2**. Generate initial population randomly **3**. Calculate the fitness values of individuals **4**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best individual **b.** (Else) If not satisfied: • Generate new population via selection, crossover and mutation processes • Go to step 3.	**1.** Define initial parameters (number of the swarm particles, inertia weight, acceleration constants, etc.) **2**. Generate initial particles randomly **3.** Calculate the fitness values of particles **4**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best particle **b.** (Else) If not satisfied: • Update particles' velocities and positions • Go to step 3
Cuckoo Search Algorithm (CS)	**Differential Evolution (DE)**
1. Define initial parameters (population size, probability, etc.) **2**. Generate initial population randomly **3.** Calculate the fitness values of host nests **4.** Update cuckoo solutions **5.** Apply the replacing process **6.** Determine the cuckoo societies **7**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best cuckoo **b.** (Else) If not satisfied: • Go to step 4	**1**. Define initial parameters (population size, crossover rate, scaling factor, etc.) **2**. Generate initial population randomly **3**. Calculate the fitness values of individuals. **4**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best individual **b.** (Else) If not satisfied: • Generate new population via recombination and selection processes • Go to step 3
Black Widow Optimization Algorithm (BWO)	**Harmony Search (HS)**
1. Define initial parameters (number of the candidate solutions, maximum iteration, etc.) **2**. Generate initial population **3**. Evaluate the fitness of individuals **4** Is the termination criteria satisfied? **a.** If satisfied: • Choose best solution so far **b.** (Else) If not satisfied: • Select parents randomly • Apply procreate, cannibalism, mutation • Update population • Go to step 3	**1**. Define initial parameters (harmony memory size, harmony memory consideration rate, distance bandwidth, pitch adjustment rate, etc.) **2**. Initialize harmony memory randomly **3**. Calculate harmony memory solutions **4** Is the termination criteria satisfied? **a.** If satisfied: • Choose best harmony in the harmony memory **b.** (Else) If not satisfied: • Improve new harmonies • Update harmony memory • Go to step 3
Harris Hawks Optimization (HHO)	**Salp Swarm Algorithm (SSA)**
1. Define initial parameters (population size, maximum iteration, etc.) **2**. Generate initial population randomly **3**. Calculate fitness values of hawks **4**. Set the location of rabbit **5**. Update the location vector **6**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best hawk **b.** (Else) If not satisfied: • Go to step 3	**1**. Define initial parameters (population size, maximum iteration, etc.) **2**. Generate salp individuals with random positions **3**. Calculate each salp in the population **4**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best salp **b.** (Else) If not satisfied: • Decrease value of balance parameter adaptively • Update positions of salps • Evaluate fitness values of salps • Go to step 4
Multi-verse Optimizer (MVO)	**Chimp Optimization Algorithm (ChOA)**
1. Define initial parameters (universes, wormhole existence probability (WEP), travelling distance rate (TDR), etc.) **2**. Generate initial universes randomly **3**. Calculate the fitness values of universes **4**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best universe **b.** (Else) If not satisfied: • Update WEP and TDR • Update universes • Go to step 3	**1**. Define initial parameters (number of chimps, maximum iteration, decreasing shapes parameter, etc.) **2**. Generate initial chimp population **3**. Calculate the position of each chimp **4**. Divide chimps into independent groups randomly **5**. Is the termination criteria satisfied? **a.** If satisfied: • Choose best chimp **b.** (Else) If not satisfied: • Calculate the position of each chimp • Update the search agents and parameters • Go to step 5

(*continued on next page*)

range randomly and each coefficient in the vector is assigned to the zones (eight coefficients in total). To find the best range of this vector that includes coefficients, we have sampled this vector over various intervals; we then have ranked the trialled ranges according to mean values for 23 numerical functions. These 23 benchmark functions commonly used in literature [1, 13,28,29] are listed. The benchmark functions consist of three types of functions: (1) unimodal functions: F1–F7, (2) multimodal functions: F8–F13 and (3) fixed-dimension multimodal functions: F14–F23. The best range is determined according to the number of best mean values (No. best) for each algorithm with a different parameter range. It is worth to mention that since there is no chance to try all combinations of the ranges, the purpose is to determine the most optimal coefficient vector range as possible.

Table 1 (*continued*).

Effective Butterfly Optimizer with Covariance Matrix Adapted Retreat Phase (EBOwithCMAR)	**Arithmetic Optimization Algorithm (AOA)**
1. Define initial parameters (population size, maximum iteration, step size, dynamic probability, etc.) **2.** Generate subpopulations randomly **3.** Calculate probabilities and fitness values **4.** Apply data sharing, EBO, CMAR techniques according to probabilities **5.** Is the termination criteria satisfied? a. If satisfied: • Choose best individual b. (Else) If not satisfied • Go to step 3	**1.** Define initial parameters (number of solutions, maximum iteration, control and sensitive parameters, etc.) **2.** Generate the solutions' positions randomly **3.** Calculate the fitness values of solutions **4.** Apply the operators with respect to random values **5.** Is the termination criteria satisfied? a. If satisfied: • Choose best individual b. (Else) If not satisfied • Go to step 3

Table 2
The parameter settings of the algorithms used in this study.

GA	HHO	PSO	ChOA	MVO	EBOwithCMAR
Mutation rate: 0.2 Crossover rate: 0.5	Adaptive parameters	Cognitive constant: 1 Social constant : 1 Local constant: 0.3	Dynamic coefficient vector Chaotic vector based on chaotic map	Wormhole existence probability max: 1 Wormhole existence probability min: 0.2	Step size: 0.3 Dynamic probability: 0.1
DE	**SSA**	**BWO**	**CS**	**HS**	**AOA**
Crossover constant: 0.3 Scaling factor: 0.9	Adaptive parameters	Procreating rate: 0.6 Cannibalism rate: 0.44 Mutation rate: 0.4	Discovery rate of alien eggs: 0.25	Harmony memory considering rate: 0.9 Pitch adjusting rate maximum value: 0.9 Pitch adjusting rate minimum value: 0.0 Bandwidth maximum value: 1 Bandwidth minimum value: 1e−6	Control parameter: 0.5 Sensitive parameter: 5

Table 3
The experimental results of the up-to-date metaheuristics for benchmark functions.

			BWO	SSA	MVO	HHO	ChOA	GOZDE
Unimodal	F1	Mean	8.587e−04 +	1.844 +	3.407 +	8.725e+04 +	0.016 +	6.378e−06
		Std	0.002	1.413	0.597	5.362e+03	0.018	1.119e−05
		FEs	4840	24000	24000	38125	21760	19528
	F2	Mean	0.001 +	5.971 +	61.567 +	3.344e+14 +	0.006 +	1.606e−11
		Std	0.001	2.792	70.877	6.07e+14	0.005	1.046e−10
		FEs	4840	24000	24000	10500	22160	168
	F3	Mean	1.949e+03+	4.324e+03 +	1.972e+03 +	1.266e+05 +	3.637e+03 +	4.567e−05
		Std	6.199e+02	2.785e+03	5.694e+02	1.417e+04	3.327e+03	6.183e−05
		FEs	4840	24000	24000	8491	19600	17944
	F4	Mean	22.592 +	12.818 +	6.480 +	80.964 +	4.915 +	0.002
		Std	7.115	2.253	2.141	1.885	3.686	0.002
		FEs	1944	24000	24000	15433	19840	13808
	F5	Mean	7.404e+03 +	6.699e+02+	5.216e+02 +	2.690e+08 +	49.630 +	4.242
		Std	1.327e+04	5.221e+02	7.023e+02	2.900e+07	0.786	12.059
		FEs	4840	24000	24000	21493	20560	19616
	F6	Mean	2.445e−04 −	2.515 +	3.794 +	8.657e+04 +	6.765 +	0.055
		Std	2.944e−04	2.145	0.783	4.187e+03	0.902	0.058
		FEs	4840	24000	24000	47217	18400	25688
	F7	Mean	0.050 +	0.257+	0.056 +	2.052e+02 +	0.004 +	0.001
		Std	0.026	0.092	0.021	22.287	0.004	0.001
		FEs	4840	23680	23840	46509	21040	14424
Multimodal	F8	Mean	−1.616e+04 -	−1.217e+04 =	−1.282e+04 =	−5.447e+03 +	−9.306e+03 +	−1.248e+04
		Std	1.737e+03	1.012e+03	8.717e+02	2.961e+02	57.570	1.493e+03
		FEs	4840	23840	24000	29011	320	26480
	F9	Mean	0.051+	51.366 +	2.359e+02 +	6.414e+02 +	30.719 +	5.129e−05
		Std	0.242	20.239	38.201	23.795	29.224	1.075e−04
		FEs	4840	24000	24000	5001	20000	19440
	F10	Mean	0.002 +	3.668 +	2.293 +	20.237 +	19.964 +	0.001
		Std	0.001	0.979	0.434	0.115	9.734e−04	0.002
		FEs	4840	24000	24000	27593	2160	23312
	F11	Mean	0.105+	0.827 +	0.999 +	7.977e+02 +	0.087 +	1.398e−05
		Std	0.193	0.205	0.041	35.348	0.111	2.099e−05

(*continued on next page*)

Table 3 (*continued*).

			BWO	SSA	MVO	HHO	ChOA	GOZDE
		FEs	4840	24000	24000	40228	21360	17856
		Mean	0.003 +	6.401 +	3.395 +	5.189e+08 +	0.627 +	0.001
	F12	Std	0.021	2.720	1.263	1.059e+08	0.174	0.001
		FEs	4840	24000	24000	22610	17680	26304
		Mean	0.019 +	58.144 +	0.543 +	1.146e+09 +	4.686 +	0.010
	F13	Std	0.062	14.707	0.474	1.086e+08	0.129	0.016
		FEs	4840	24000	24000	1249	20880	6240
		Mean	4.999 +	1.196 +	0.998 =	1.045 +	0.998 +	0.998
	F14	Std	2.669	0.546	1.777e−11	0.111	2.774e−05	9.313e−11
		FEs	1416	24000	24000	31894	18000	8792
		Mean	0.001 +	8.329e−04 +	0.003 +	0.002 +	0.001 +	3.659e−04
	F15	Std	9.915e−04	2.614e−04	0.007	8.949e−04	4.409e−05	8.859e−05
		FEs	3384	24000	24000	34770	13040	24016
		Mean	−1.030 +	−1.031 =	−1.031 +	−1.029 +	−1.031 +	−1.031
	F16	Std	0.001	2.618e−10	1.696e−07	0.002	6.847e−06	7.430e−11
		FEs	824	24000	23920	27715	18880	18208
		Mean	0.398 =	0.397 +	0.397 +	0.400 +	0.398 +	0.397
	F17	Std	0.002	4.632e−4	3.694e−07	0.002	0.001	3.342e−07
		FEs	680	24000	24000	28228	640	4480
		Mean	4.187 +	3.000 =	3.000 +	3.101 +	3.000 +	3.000
Fixed dimension multimodal	F18	Std	4.992	1.693e−08	1.629e−06	0.124	1.204e−04	4.963e−08
		FEs	840	23920	23840	3178	18800	14952
		Mean	−3.851 +	−3.862 =	−3.862 =	−3.856 +	−3.855 +	−3.862
	F19	Std	0.011	2.197e−07	2.444e−07	0.003	0.002	1.030e−06
		FEs	2248	24000	23800	41822	18800	18208
		Mean	−3.248 +	−3.209 +	−3.254 +	−3.074 +	−2.903 +	−3.266
	F20	Std	0.077	0.038	0.060	0.062	0.234	0.060
		FEs	2376	24000	24000	45380	17920	21640
		Mean	−7.663+	−8.140 +	−7.036 +	−3.630 +	−4.380 +	−10.153
	F21	Std	3.581	2.978	2.862	1.242	1.422	5.914e−05
		FEs	1032	23920	24000	15933	17280	22608
		Mean	−8.469 +	−10.004 -	−8.836 +	−3.915 +	−4.865 +	−9.954
	F22	Std	3.276	1.527	2.695	0.907	0.748	1.693
		FEs	1576	24000	24000	29874	19840	25512
		Mean	−10.024 +	−9.531 +	−9.124 +	−3.771 +	−5.052 +	−10.089
	F23	Std	1.944	2.612	2.654	1.088	0.032	1.7002
		FEs	2136	23840	24000	23326	21200	25512
	+/=/-		20/1/2	18/4/1	20/3/0	23/0/0	23/0/0	
	Avg. Rank		4.408	4.605	5.284	10.215	6.383	3.072

Table 4
The experimental results of the classical metaheuristics for benchmark functions.

			GA	PSO	DE	CS	HS	GOZDE
		Mean	9.106e+02 +	42.061 +	9.306e+04 +	2.805e+03 +	2.984e+03 +	6.378e−06
	F1	Std	1.928e+02	22.830	1.168e+04	604.740	3.500e+02	1.119e−05
		FEs	21080	23840	17040	47120	23920	19528
		Mean	12.473 +	13.244 +	1.782e+16 +	7.426e+09 +	12.220 +	1.606e−11
	F2	Std	1.819	4.797	4.647e+16	4.366e+09	1.139	1.046e−10
		FEs	21010	24080	23920	37360	23600	168
		Mean	9.429e+03 +	7.457e+03 +	1.568e+05 +	3.495e+04 +	9.100e+04 +	4.567e−05
	F3	Std	2.531e+03	2.432e+03	1.275e+04	5.195e+03	1.043e+04	6.183e−05
		FEs	21010	24080	23920	47760	8880	17944
Unimodal		Mean	18.891 +	14.593 +	90.426 +	36.226 +	40.925 +	0.002
	F4	Std	2.353	2.264	2.376	3.528	1.644	0.002
		FEs	20870	23920	160	44080	22000	13808
		Mean	6.028e+04+	3.000e+03 +	4.008e+08 +	7.217e+05 +	1.349e+06 +	4.242
	F5	Std	2.626e+04	2.024e+03	6.888e+07	2.625e+05	2.591e+05	12.059
		FEs	21080	24080	15920	48080	21360	19616
		Mean	8.376e+02 +	38.737 +	9.308e+04 +	2.826+03 +	2.955e+03 +	0.055
	F6	Std	2.530e+02	23.221	1.010e+04	534.433	3.955e+02	0.058
		FEs	21080	24000	19760	46640	24080	25688
		Mean	0.357 +	0.186 +	2.847e+02 +	1.129 +	1.377 +	0.001
	F7	Std	0.124	0.039	43.948	0.359	0.195	0.001
		FEs	10650	22960	21360	47920	21760	14424

(*continued on next page*)

Table 4 (*continued*).

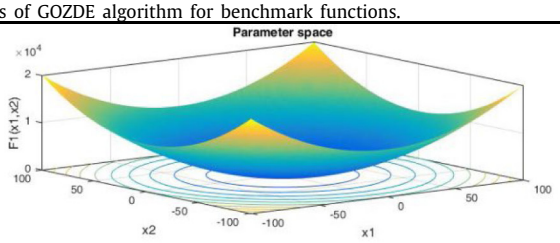
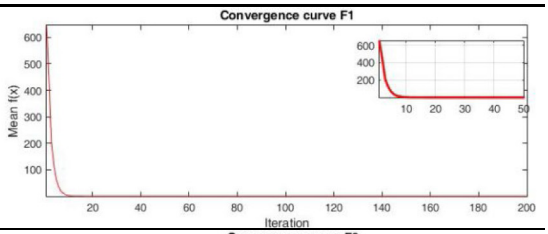
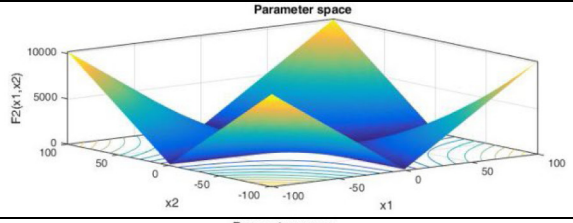
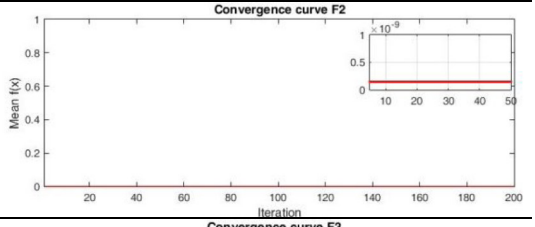
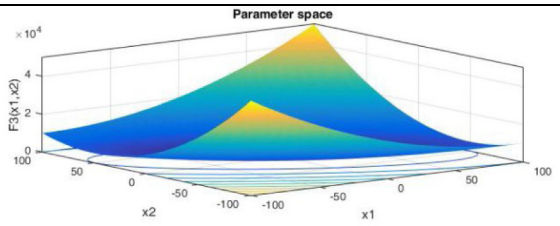
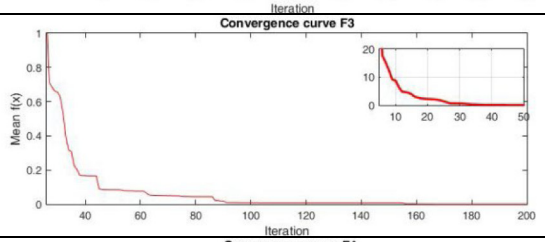
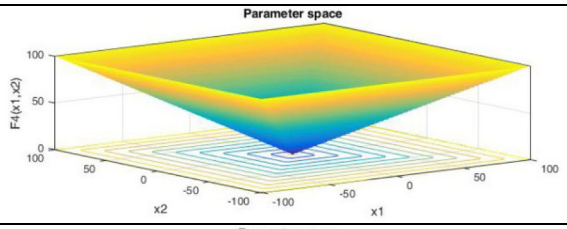
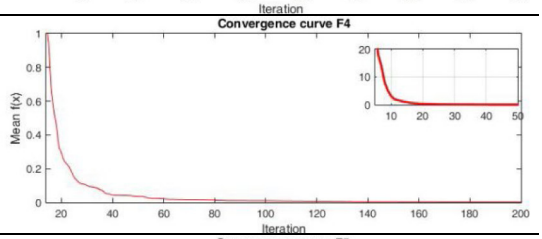
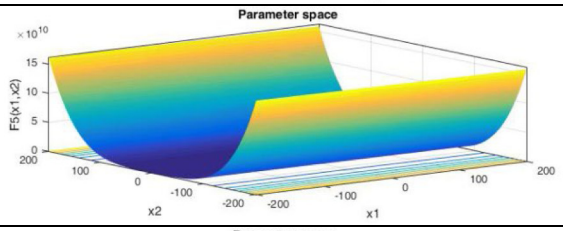
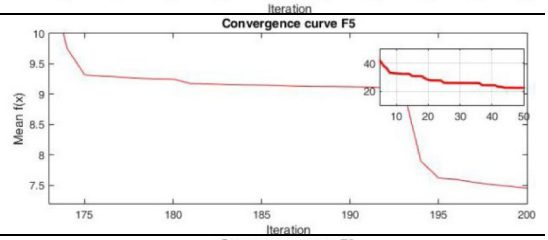
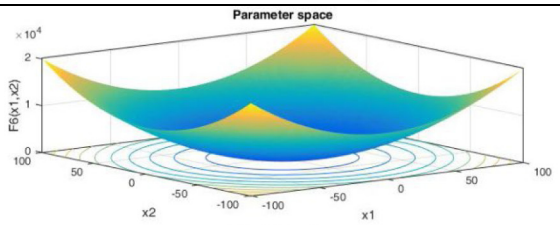
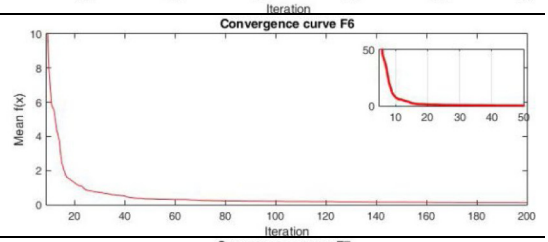
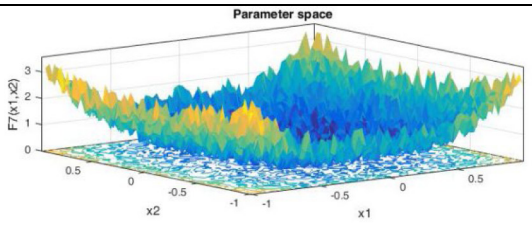
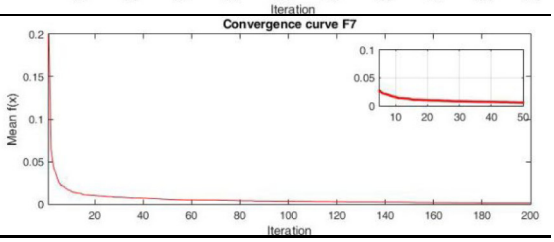
			GA	PSO	DE	CS	HS	GOZDE
Multimodal	F8	Mean	−1.214e+04 =	-8.491e+03 +	−7.250e+03 +	−4.163e+03 +	−1.194e+04 =	−1.248e+04
		Std	6.982e+02	6.781e+02	5.126e+02	2.628e+02	1.966e+02	1.493e+03
		FEs	21080	22560	22000	46320	23920	26480
	F9	Mean	1.014e+02 +	3.165e+02 +	6.963e+02+	2.983e+02 +	52.814 +	5.129e−05
		Std	16.911	26.350	25.523	20.520	5.431	1.075e−04
		FEs	20240	24080	2080	47920	22960	19440
	F10	Mean	6.147 +	3.718 +	19.958 +	16.612 +	9.213 +	0.001
		Std	0.647	0.460	0.001	1.114	0.449	0.002
		FEs	21080	24080	23680	44080	23440	23312
	F11	Mean	8.736 +	1.344 +	8.387e+02 +	26.931 +	28.334 +	1.398e−05
		Std	1.556	0.157	94.565	5.508	3.882	2.099e−05
		FEs	21080	24080	14320	47760	24000	17856
	F12	Mean	4.336 +	9.502 +	9.765e+08 +	2.051e+03+	6.943e+04 +	0.001
		Std	1.360	2.534	2.096e+08	2.664e+03	2.713e+04	0.001
		FEs	21080	24080	11360	47920	23120	26304
	F13	Mean	3.210e+02 +	76.166 +	1.797e+09 +	3.774e+05 +	1.522e+06	0.010
		Std	5.135e+02	34.984	3.814e+09	1.903e+05	3.987e+05	0.016
		FEs	21080	24080	22000	47120	24000	6240
Fixed dimension multimodal	F14	Mean	0.998 =	0.998 =	0.998 +	0.998 +	0.998 +	0.998
		Std	2.620e−09	0.001	9.737e−11	4.504e−10	1.382e−10	9.313e−11
		FEs	4630	23520	23760	45520	15440	8792
	F15	Mean	0.002 +	0.001 +	6.737e−04 =	5.549e−04 +	8.178e−04 +	3.659e−04
		Std	0.003	2.717e−04	4.562e−04	9.184e−05	9.309e−05	8.859e−05
		FEs	21080	24080	23360	34160	24000	24016
	F16	Mean	−1.031 -	−1.031 -	−1.031 +	−1.031 +	−1.031 +	−1.031
		Std	1.045e−16	1.295e−15	6.387e−06	4.222e−06	2.188e−15	7.430e−11
		FEs	6660	12320	16320	45200	23360	18208
	F17	Mean	0.397 +	0.397 -	0.397 -	0.397 +	0.397 -	0.397
		Std	3.710e−05	1.789e−13	0.000	1.305e−5	1.272e−14	3.342e−07
		FEs	1900	21120	12560	46800	19360	4480
	F18	Mean	3.000 +	3.000 -	3.000 -	3.000 +	3.000 +	3.000
		Std	1.635e−04	2.059e−15	7.691e−16	3.984e−3	3.373e−3	4.963e−08
		FEs	4630	17360	11680	47760	21120	14952
	F19	Mean	−3.862 +	−3.862 +	−3.862 -	−3.862 +	−3.862 +	−3.862
		Std	2.210e−06	2.385e−5	2.668e−15	1.108e−6	6.284e−4	1.030e−06
		FEs	3300	23680	22000	46800	23440	18208
	F20	Mean	−3.235 +	−3.216 +	−3.211 +	−3.198 +	−3.203 +	−3.266
		Std	0.051	0.033	0.030	0.064	3.486e−07	0.060
		FEs	23760	24080	24080	43440	21120	21640
	F21	Mean	−6.326 +	−10.137 +	−8.769 +	−10.141 +	−5.200 +	−10.153
		Std	3.688	0.047	0.721	0.013	3.564	5.914e−05
		FEs	12470	22640	24080	38320	23040	22608
	F22	Mean	−8.970 +	−9.526 +	−9.021+	−10.396 -	−7.853 +	−9.954
		Std	2.922	1.994	0.756	0.006	3.366	1.693
		FEs	16320	24080	9040	46960	22640	25512
	F23	Mean	−8.572 +	−9.717 +	−9.188 +	−10.534 -	−10.376 -	−10.089
		Std	3.321	1.682	0.916	0.001	0.878	1.7002
		FEs	3930	24080	22800	46000	23680	25512
	+/=/-		20/2/1	19/1/3	19/1/3	21/0/2	20/1/2	
	Avg. Rank		6.099	5.961	5.527	7.899	6.527	

The experimental result can be seen in Appendix (Tables A.1–A.6) in which the "no parameter" column denotes the usage of the proposed algorithm without the coefficient vector. The obtained results in these tables show that the optimal range of the coefficient vector is [−1, 0]. On the other hand, it seems that when the coefficient range distance between upper and lower ranges increases, the accuracies of solutions are perturbed.

3. Experimental setup

In this study, the previously mentioned 23 benchmark functions are used in the low-dimensional numerical comparisons, whereas 20 hybrid and composition functions with 100D from CEC 2017 [30] are used for the high-dimensional numerical comparisons. The default parameters for all compared algorithms (Table 1, [6–8,14–19,25]) are given in Table 2 as recommended many studies [13–15,23,24,31]. To fair comparison, population size is set to 80 and number of maximum iteration is 300. It is noteworthy to mention here that the termination criterion is chosen to be a number of iterations instead of a number of function evaluations because of the iteration-dependent property of the chimp optimization algorithm. Meanwhile, the required number of function evaluations (FEs) to achieve the best solution for each algorithm is given in Tables 3–4 and Tables 7–8. Each algorithm ran 30 times to calculate mean and standard deviation (Std) values, which can be seen in Tables 3–4 and Tables 7–8. To prove the experimental results, three statistical tests are conducted. Firstly, in Wilcoxon rank-sum test [32] at the %5 significance level, the symbols "+,-,=" mean that the proposed GOZDE performs significantly better (+) or significantly worse (-) or there is no statistical difference between the proposed algorithm and the compared algorithm (=). Secondly, Friedman test [33] is a nonparametric technique which is used for multiple comparisons

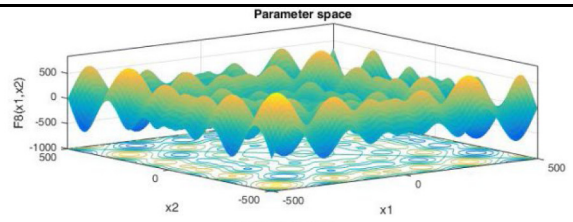
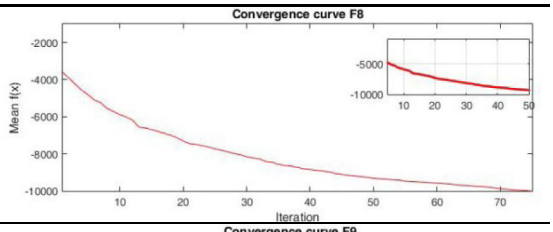
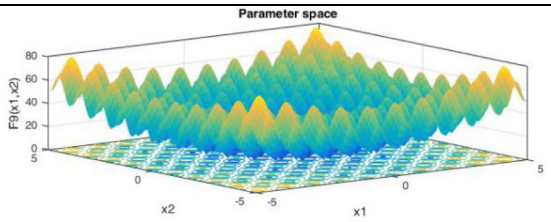
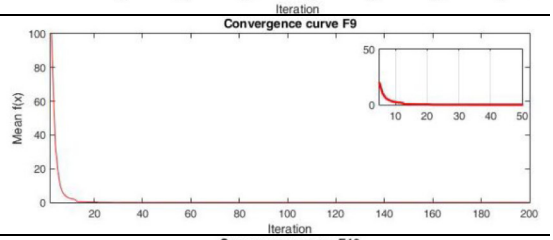
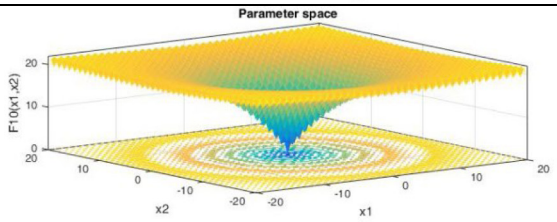
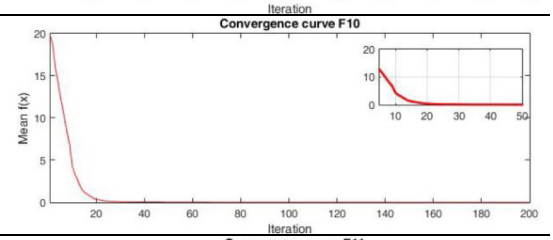
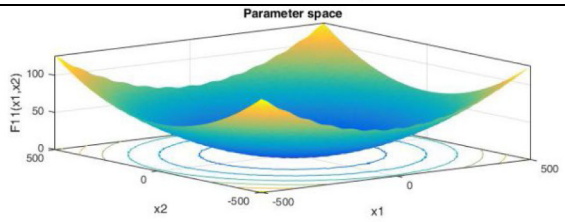
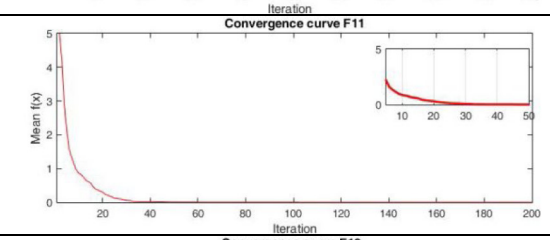
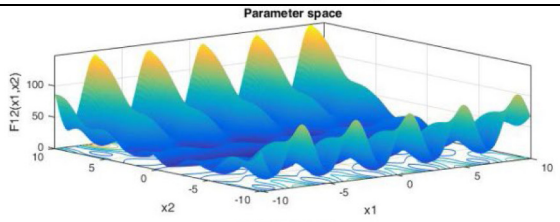
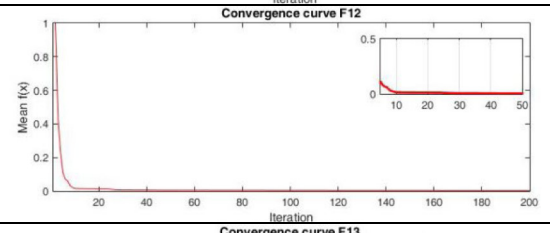
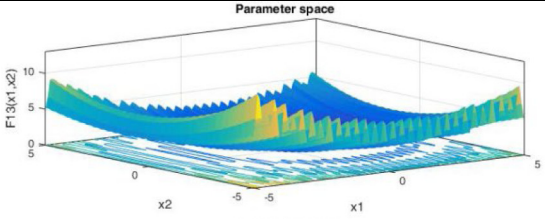
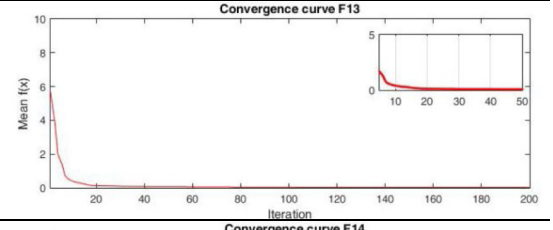
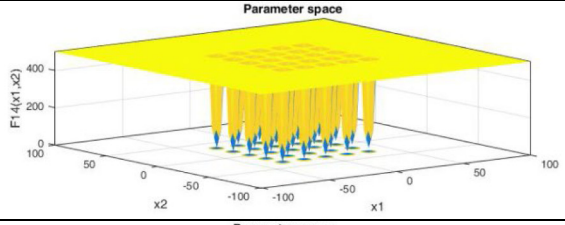
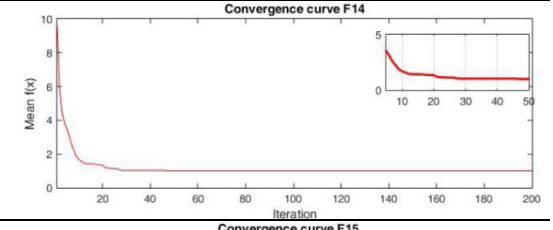
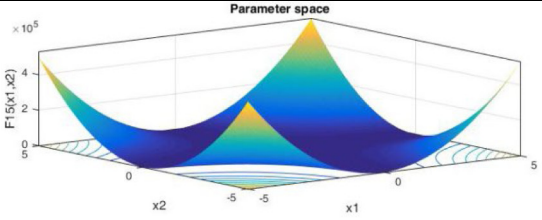
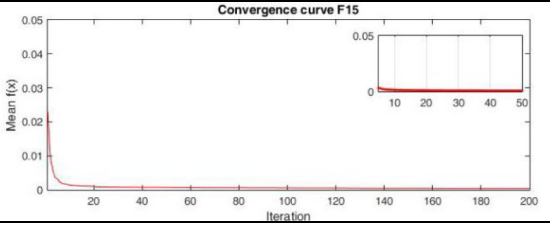
Table 5
Convergence curves of GOZDE algorithm for benchmark functions.

Unimodal	F1		
	F2		
	F3		
	F4		
	F5		
	F6		
	F7		

(*continued on next page*)

to detect difference between the performances of two or more algorithms. This statistical test ranks the algorithms from the best performing one to the worst one with 0.05 confidence level. A lowest value of rank means the best performing algorithm. The average ranks of the Friedman test and the total results of the Wilcoxon test for each algorithm are given in the last two

Table 5 (*continued*).

Multimodal	F8		
	F9		
	F10		
	F11		
	F12		
	F13		
Fixed dimension multimodal	F14		
	F15		

(*continued on next page*)

Table 5 (*continued*).

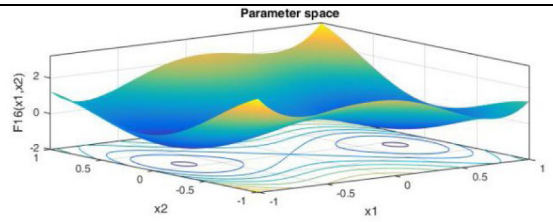
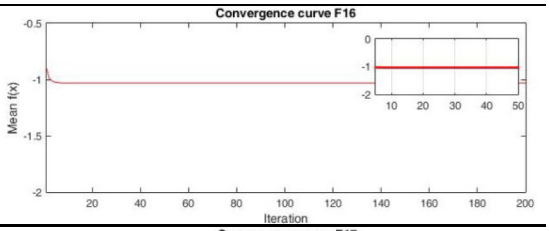
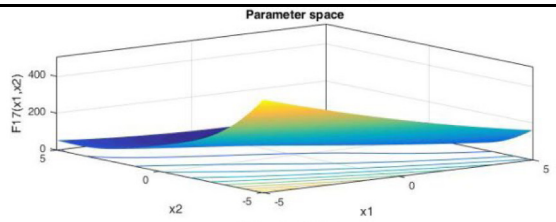
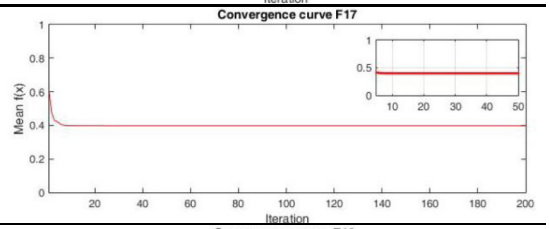
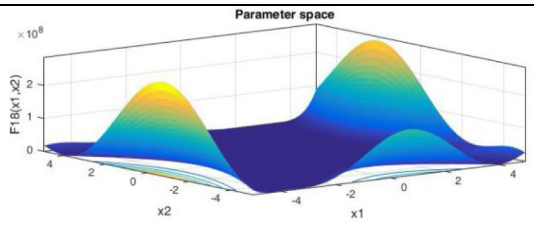
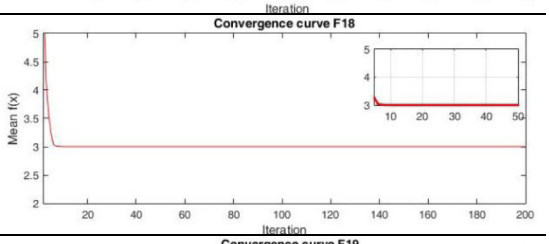
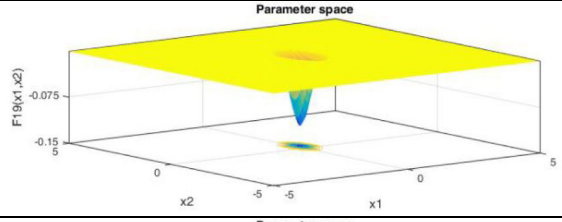
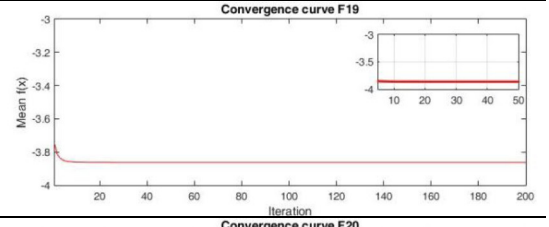
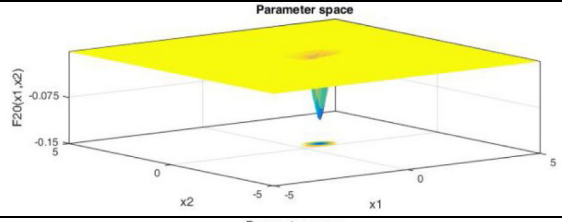
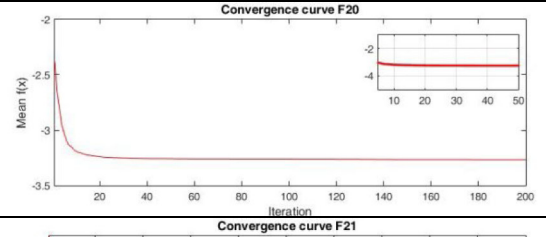
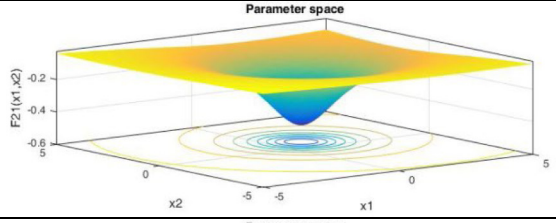
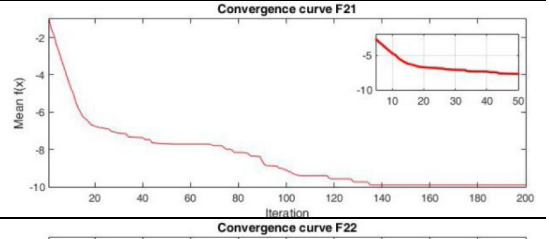
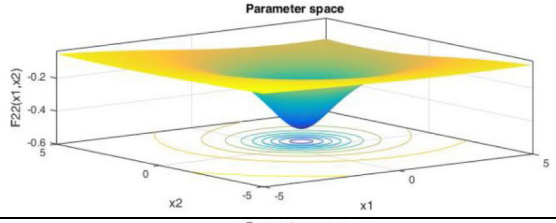
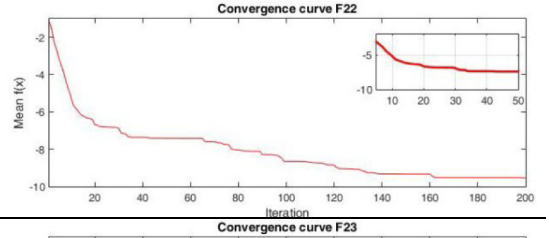
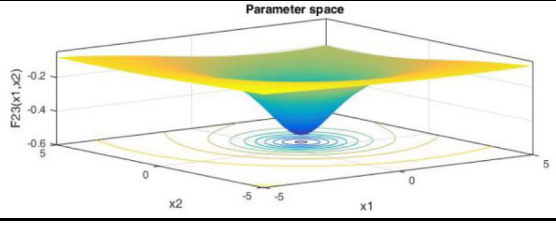
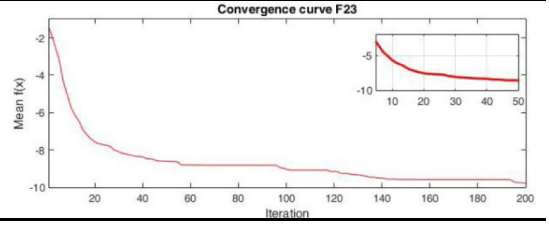
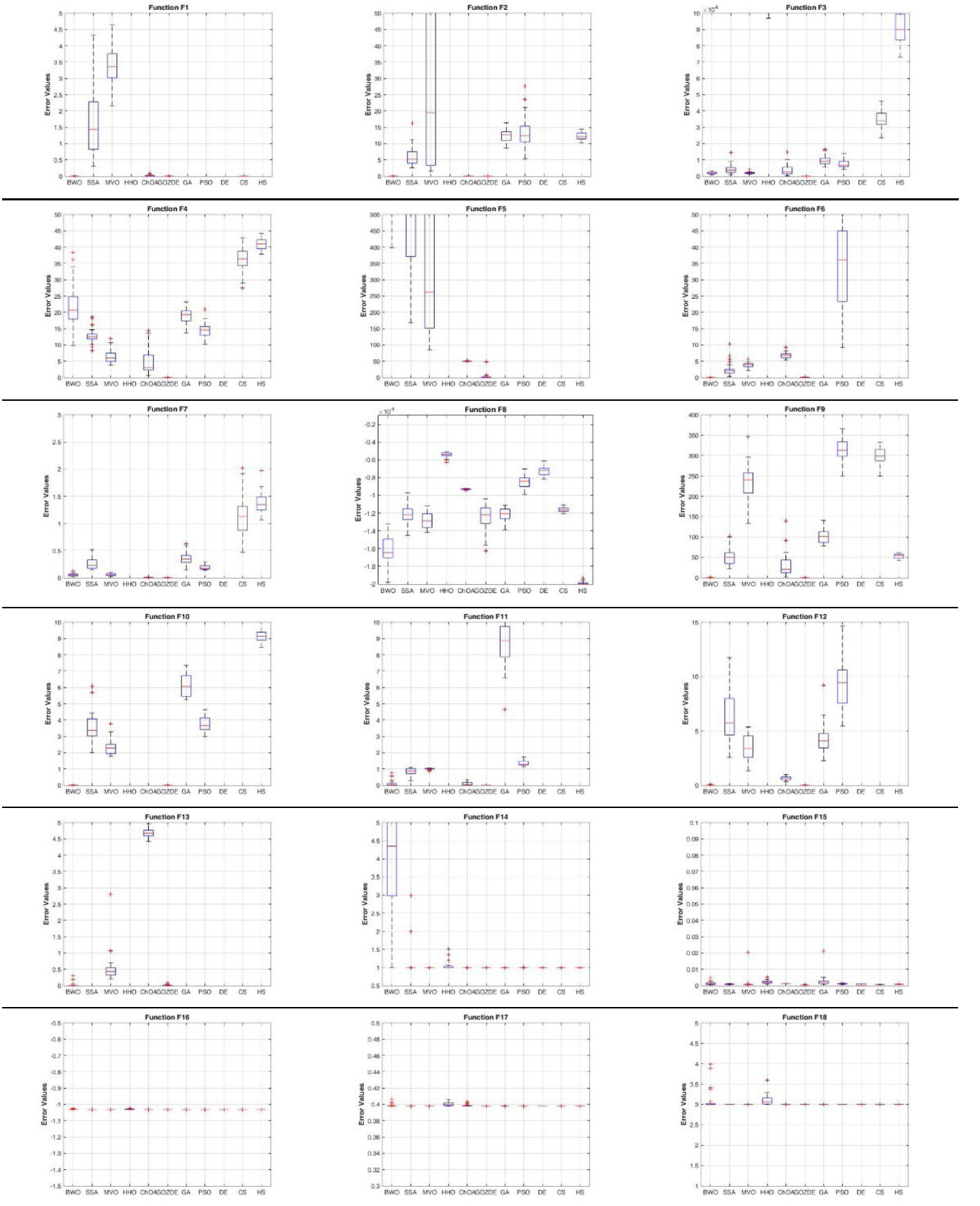
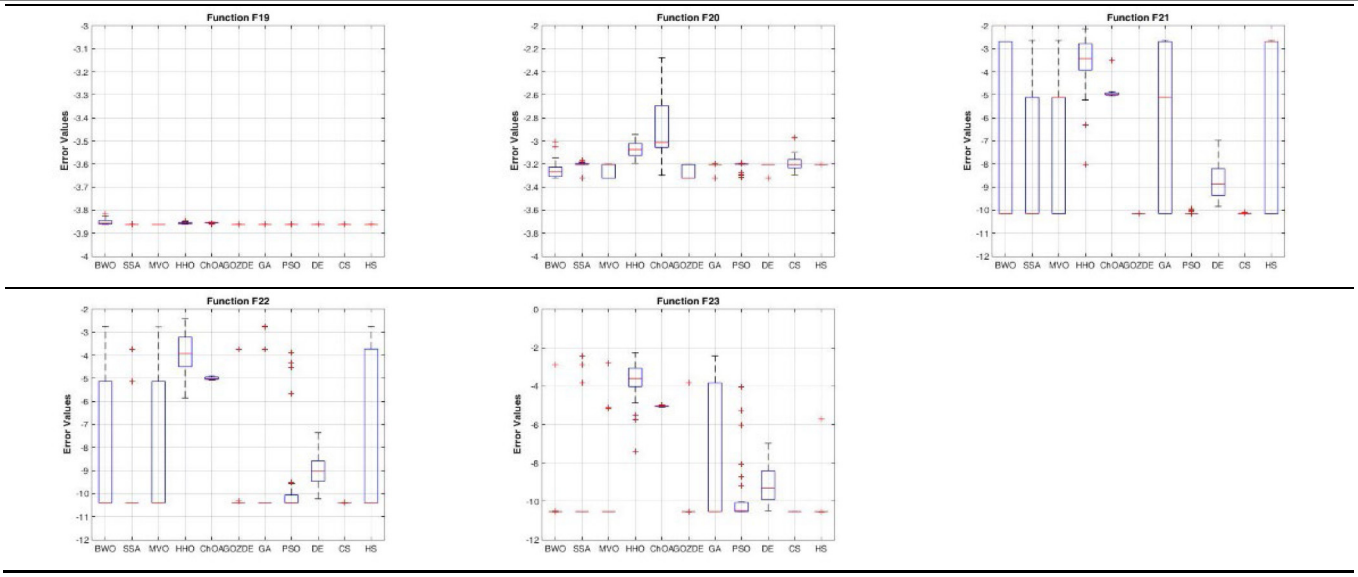
F16		
F17		
F18		
F19		
F20		
F21		
F22		
F23		

Table 6
Box-and-whisker plots of each algorithm over 30 runs.



(*continued on next page*)

Table 6 (*continued*).



rows of the Tables 3–4 and Tables 7–8. Thirdly, Whisker-box plot [34] shows the range of objective function value, in which the central box shows the values between the first and third quartiles of the fitness distribution, the red line represents the median, and outliers are marked with a star symbol. To demonstrate the effectiveness of GOZDE, five classical and five state-of-the-art metaheuristics are included in all comparisons [6–8,14,15,22–24]. In addition, EBOwithCMAR [25] and AOA [16] are added into the comparison for hybrid and composition functions. It is worth noting here that EBOwithCMAR and ChOA algorithms are represented as "EC" and "Ch", respectively in the related Whisker-box plots due to space limitations. Moreover, the shape of the benchmark functions in 3D view, and convergence graphs for the proposed algorithm are also visualized in the following subsection for low-dimensional functions.

3.1. The performance of the proposed algorithm over low-dimensional benchmark functions

As can be seen in Table 3, considering Wilcoxon test results, the proposed GOZDE algorithm shows significantly better performance than BWO on 20 functions, SSA on 18 functions, MVO on 20 functions, HHO on 23 functions and ChOA on 23 functions out of 23 benchmark functions. It is noteworthy to mentioning that GOZDE achieves the same result with BWO on only one function (F17), SSA on one function (F8) and MVO for four functions (F8, F14, F19, F20). This strong performance of GOZDE is proved to be superior to the compared state-of-the-art metaheuristics based on the average ranks of the algorithms in terms of the Friedman test which is given in the last row of Table 3. According to this test results, the proposed approach is significantly better than that of five metaheuristics, taking a minimum rank of 3.072. In comparison of the GOZDE to the classical metaheuristics given in Table 4, it can be clearly observed that the proposed algorithm produces better solutions than the classical metaheuristic algorithms in a statistical manner on most of the functions. In detail, the GOZDE algorithm achieves better results than GA on 20 functions, PSO on 19 functions, DE on 19 functions, CS on 21 functions and HS on 20 functions out of 23 benchmark functions. This demonstrates that the proposed algorithm remains its robust competitive performance over the classical metaheuristics. Generally speaking, GOZDE can be considered the most effective metaheuristic among the compared algorithms with the smallest Friedman rank value and the best Wilcoxon test results, which shows the promising search ability of GOZDE in reaching global optima.

The visualization of the convergence curves of GOZDE algorithm for each test function is given in Table 5. For unimodal functions (F1–F7), the proposed algorithm dominates the best mean results, and finds the near-global optima fast and effectively, almost in 100 iterations, except only for F5. For multimodal functions (F8–F23) including many local minima, GOZDE often jumps from local minima, and it obtains the near-global optima. For fixed dimension multimodal functions (F14–F23), GOZDE remains to show the competitive performance and it convergences rapidly to the global optima. To further verify the performance of GOZDE, Whisker-box plot results are carried out with the aforementioned 10 metaheuristic algorithms. The experimental results are visualized in Table 6. It can be seen from this table, GOZDE shows very small distributions for unimodal functions. It means GOZDE can achieve similar performances in each run and is a more stable algorithm than the others on unimodal functions. On the other hand, the distributions of some algorithms are not visible in the table for these functions because they are out of the table since their distributions of solutions are so large. As similar the results of unimodal functions, GOZDE outperforms the other algorithms in terms of median value (red line) for multimodal functions except only for F8. As comes to the fixed dimension multimodal functions, GOZDE is a good competitor which has shown to give outstanding convergence rates to optimal solutions on most of the functions with low rates of distributions. In summary, based on the results and analyses of performance of the algorithms on low-dimensional problems, the proposed GOZDE algorithm is able to obtain the superior solutions of the majority of the test functions, and it produces significantly better results statistically as compared to the both classical and state-of-the-art algorithms.

3.2. The performance of the proposed algorithm over high-dimensional benchmark functions

Unlike the other experiments, here, two challenging algorithms called AOA and EBOwithCMAR have been added for comparisons over high-dimensional benchmark functions. As can be

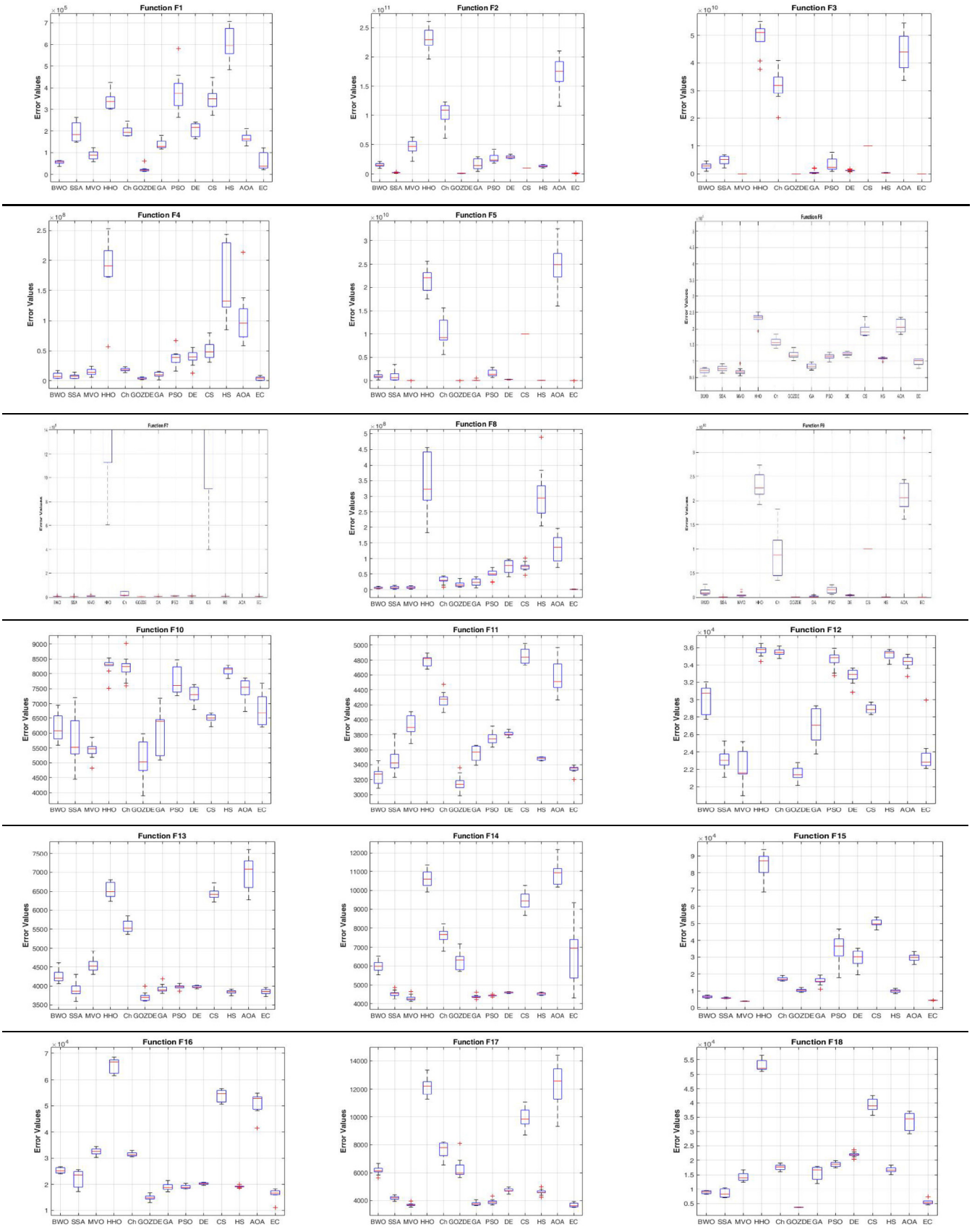
Table 7
The experimental results of the up-to-date metaheuristics for benchmark functions.

			BWO	SSA	MVO	HHO	ChOA	EBOwithCMAR	AOA	GOZDE
Hybrid	F1	Mean	5.448e+04+	1.958e+05+	8.820e+04+	3.396e+05+	2.015e+05+	5.458e+04+	1.652e+05+	2.317e+04
		Std	9.023e+03	4.634e+04	2.051e+04	3.804e+04	2.503e+04	3.784e+04	2.416e+04	1.353e+04
		FEs	5235	16000	16100	29505	13600	14384	5880	26480
	F2	Mean	1.497e+10+	2.040e+09+	4.554e+10+	2.297e+11+	1.022e+11+	7.949e+08=	1.712e+11+	7.754e+08
		Std	3.938e+09	8.576e+08	1.304e+10	2.067e+10	1.951e+10	3.776e+08	2.667e+10	2.430e+08
		FEs	5100	16405	15115	4964	11200	14792	10040	15255
	F3	Mean	2.736e+09+	4.792e+09+	1.675e+06+	4.920e+10+	3.172e+10+	5.215e+04-	4.373e+10+	1.014e+05
		Std	1.221e+09	1.667e+09	4.376e+05	5.695e+09	5.504e+09	1.784e+04	6.755e+09	4.342e+04
		FEs	5205	16100	14907	1937	11900	13093	5840	17063
	F4	Mean	8.902e+06+	8.210e+06+	1.525e+07+	1.853e+08+	1.906e+07+	4.354e+06=	1.057e+08+	4.134e+06
		Std	4.578e+06	3.595e+06	5.588e+06	5.219e+07	2.888e+06	3.587e+06	4.494e+07	1.947e+06
		FEs	5200	14107	16105	3567	14500	13860	10920	21110
	F5	Mean	1.026e+09+	1.118e+09+	4.321e+05+	2.161e+10+	1.027e+10+	1.323e+04-	2.446e+10+	1.010e+05
		Std	6.322e+08	1.251e+09	1.451e+05	2.659e+09	3.291e+09	2.849e+03	4.676e+09	6.473e+04
		FEs	4905	14105	15420	19794	14850	12585	8360	16801
	F6	Mean	7.008e+03-	7.676e+03-	6.823e+03-	2.323e+04+	1.580e+04+	2.631e+04=	2.079e+04+	1.195e+04
		Std	8.290e+02	9.056e+02	1.070e+03	1.586e+03	1.376e+03	3.547e+04	2.101e+03	1.147e+03
		FEs	5120	15905	16200	20470	11500	12714	10400	24231
	F7	Mean	7.377e+03+	6.047e+03+	9.764e+03+	3.165e+06+	2.722e+04+	6.961e+03+	7.071e+06+	5.359e+03
		Std	2.714e+03	7.505e+02	3.631e+03	2.052e+06	1.719e+04	4.220e+02	3.749e+06	3.263e+02
		FEs	5003	16100	15310	4956	12400	12749	5840	20440
	F8	Mean	5.580e+06-	6.623e+06-	7.174e+06=	3.344e+08+	3.004e+07+	1.030e+06-	1.324e+08+	1.728e+07
		Std	2.902e+06	4.398e+06	3.880e+06	1.002e+08	1.185e+07	5.461e+05	4.291e+07	9.184e+06
		FEs	4427	15313	15005	187	13050	13738	10400	18981
	F9	Mean	1.161e+09+	3.258e+07+	4.962e+08+	2.310e+10+	9.032e+09+	1.017e+04-	2.163e+10+	2.371e+07
		Std	7.313e+08	2.040e+07	4.573e+08	2.524e+09	4.918e+09	2.569e+03	4.673e+09	1.042e+07
		FEs	5220	16090	15801	1786	11450	13253	10760	23035
	F10	Mean	6.197e+03+	5.776e+03=	5.435e+03=	8.250e+03+	8.216e+03+	6.818e+03+	7.468e+03+	5.066e+03
		Std	4.585e+02	8.208e+02	2.973e+02	2.882e+02	4.025e+02	5.284e+02	3.451e+02	6.306e+02
		FEs	4321	15601	15403	18432	11400	14144	5920	26392
Composition	F11	Mean	3.260e+03=	3.453e+03+	3.909e+03+	4.795e+03+	4.268e+03+	3.336e+03+	4.560e+03+	3.150e+03
		Std	1.132e+02	1.868e+02	1.386e+02	71.909	1.072e+02	51.718	2.162e+02	1.081e+02
		FEs	5207	16110	15922	20257	12300	14292	5600	22302
	F12	Mean	3.010e+04+	2.312e+04+	2.233e+04=	3.566e+04+	3.546e+04+	2.364e+04+	3.431e+04+	2.154e+04
		Std	1.676e+03	1.162e+03	1.927e+03	6.036e+02	3.872e+02	2.327e+03	7.807e+02	8.189e+02
		FEs	5221	17056	16100	27524	11900	13372	10360	20391
	F13	Mean	4.254e+03+	3.908e+03+	4.556e+03+	6.539e+03+	5.577e+03+	3.851e+03+	6.983e+03	3.720e+03
		Std	1.666e+02	1.949e+02	1.945e+02	2.103e+02	1.667e+02	73.291	4.414e+02	1.189e+02
		FEs	5221	16907	15900	3355	12100	13189	7240	21205
	F14	Mean	6.003e+03=	4.552e+03-	4.302e+03-	1.060e+04+	7.555e+03+	6.723e+03=	1.095e+04+	6.278e+03
		Std	3.238e+02	1.782e+02	1.724e+02	4.458e+02	4.319e+02	1.629e+03	6.539e+02	5.072e+02
		FEs	5305	17502	17005	13403	10400	13242	10440	23300
	F15	Mean	6.588e+03-	5.854e+03-	3.681e+03-	8.495e+04+	1.725e+04+	4.240e+03-	2.948e+04+	1.044e+04
		Std	6.504e+02	3.383e+02	82.619	7.316e+03	1.195e+03	1.479e+02	2.124e+03	8.945e+02
		FEs	4980	16000	16407	20901	11850	13576	5220	20991
	F16	Mean	2.527e+04+	2.204e+04+	3.244e+04+	6.542e+04+	3.144e+04+	1.625e+04+	5.107e+04+	1.479e+04
		Std	1.070e+03	3.375e+03	1.214e+03	2.734e+03	7.820e+02	1.963e+03	4.046e+03	1.049e+03
		FEs	4825	16609	16710	18725	11250	11053	5720	19655
	F17	Mean	6.173e+03=	4.191e+03-	3.698e+03-	1.215e+04+	7.615e+03+	3.681e+03-	1.240e+04+	6.293e+03
		Std	3.099e+02	1.467e+02	1.121e+02	6.301e+02	5.806e+02	1.491e+02	1.615e+03	7.428e+02
		FEs	5127	16105	15905	32294	11500	13576	10640	17300
	F18	Mean	8.883e+03+	8.586e+03+	1.426e+04+	5.300e+04+	1.764e+04+	5.399e+03+	3.359e+04+	3.656e+03
		Std	5.056e+02	1.436e+03	1.461e+03	1.961e+03	9.571e+02	8.155e+02	2.989e+03	48.884
		FEs	5335	16405	16207	2778	10950	13326	7240	20562
	F19	Mean	1.362e+04+	1.147e+04+	9.216e+03=	6.123e+05+	7.632+04+	9.407e+03+	3.178e+05+	8.992e+03
		Std	1.798e+03	1.463e+03	1.008e+03	4.249e+05	4.769e+04	3.877e+02	2.013e+05	6.333e+02
		FEs	5189	16502	16013	8569	12050	14079	5600	22302
	F20	Mean	2.655e+09+	5.708e+08+	6.988e+09+	3.805e+10+	1.974e+10+	2.331e+06-	3.832e+10+	2.214e+08
		Std	1.105e+09	2.824e+08	3.540e+09	2.718e+09	1.906e+09	8.456e+05	8.113e+09	7.689e+07
		FEs	5105	16910	16555	25484	13200	14079	8640	23257
	+/ = /−		14/3/3	14/1/5	12/4/4	20/0/0	20/0/0	9/4/7	20/0/0	
	Avg. Rank		5.580	4.560	4.830	12.415	9.615	2.870	11.270	2.835

Table 8
The experimental results of the classical metaheuristics for benchmark functions.

			GA	PSO	DE	CS	HS	GOZDE
Hybrid	F1	Mean	1.376e+05+	3.856e+05+	2.081e+05+	3.479e+05+	6.017e+05+	2.317e+04
		Std	2.113e+04	8.923e+04	3.002e+04	5.035e+04	7.977e+04	1.353e+04
		FEs	30050	11850	13700	28550	15050	26480
	F2	Mean	1.670e+10+	2.599e+10+	2.902e+10+	1.000e+10+	1.306e+10+	7.754e+08
		Std	9.248e+09	7.045e+09	2.693e+09	0.000	1.775e+09	2.430e+08
		FEs	29950	14800	14650	150	12700	15255
	F3	Mean	6.441e+08+	3.272e+09+	1.199e+09+	1.00e+10+	3.537e+08+	1.014e+05
		Std	7.043e+08	2.300e+09	2.617e+08	0.000	8.362e+07	4.342e+04
		FEs	30155	15050	14350	150	14555	17063
	F4	Mean	1.073e+07+	3.888e+07+	3.868e+07+	5.157e+07+	1.619e+08+	4.134e+06
		Std	4.736e+06	1.277e+07	1.215e+07	1.627e+07	6.143e+07	1.947e+06
		FEs	27957	14950	13505	29050	2655	21110
	F5	Mean	6.867e+07+	1.595e+09+	2.048e+08+	1.000e+10+	1.930e+07+	1.010e+05
		Std	1.842e+08	7.188e+08	4.245e+07	0.00	9.011e+06	6.473e+04
		FEs	23955	15000	14855	150	14950	16801
	F6	Mean	8.434e+03-	1.135e+04=	1.221e+04=	1.961e+04+	1.079e+04-	1.195e+04
		Std	7.887e+02	8.418e+02	6.521e+02	2.080e+03	4.010e+02	1.147e+03
		FEs	22905	15050	14900	28850	9805	24231
	F7	Mean	7.061e+03+	1.034e+04+	1.003e+04+	1.849e+06+	8.044e+03+	5.359e+03
		Std	5.210e+02	1.079e+03	5.506e+02	1.059e+06	3.553e+02	3.263e+02
		FEs	19551	15355	14750	30050	10707	20440
	F8	Mean	2.409e+07=	4.984e+07+	7.544e+07+	7.469e+07+	3.040e+08+	1.728e+07
		Std	1.182e+07	1.590e+07	2.187e+07	1.520e+07	8.303e+07	9.184e+06
		FEs	22350	14800	13855	29055	12750	18981
	F9	Mean	1.227e+08+	1.450e+09+	3.744e+08+	1.000e+10+	4.017e+07+	2.371e+07
		Std	1.444e+08	6.591e+08	8.352e+07	0.000	1.839e+07	1.042e+07
		FEs	27605	15050	14600	150	14709	23035
	F10	Mean	6.118e+03+	7.771e+03+	7.293e+03+	6.494e+03+	8.106e+03+	5.066e+03
		Std	7.038e+02	4.651e+02	2.679e+02	1.410e+02	1.381e+02	6.306e+02
		FEs	29950	12100	14250	29850	8350	26392
Composition	F11	Mean	3.559e+03+	3.754e+03+	3.811e+03+	4.856e+03+	3.483e+03+	3.150e+03
		Std	97.999	82.122	35.371	1.092e+02	21.000	1.081e+02
		FEs	25980	15605	14957	29050	12055	22302
	F12	Mean	2.700e+04+	3.463e+04+	3.270e+04+	2.893e+04+	3.518e+04+	2.154e+04
		Std	1.895e+03	1.049e+03	8.702e+02	5.388e+02	5.354e+02	8.189e+02
		FEs	24503	15105	15205	29750	13850	20391
	F13	Mean	3.938e+03+	3.969e+03+	3.980e+03+	6.435e+03+	3.849e+03+	3.720e+03
		Std	1.100e+02	62.786	28.478	1.449e+02	48.782	1.189e+02
		FEs	27855	15750	14755	29107	14100	21205
	F14	Mean	4.385e+03-	4.454e+03-	4.615e+03-	9.430e+03+	4.551e+03+	6.278e+03
		Std	1.037e+02	38.749	35.187	4.868e+02	58.551	5.072e+02
		FEs	22755	14800	14505	29555	13657	23300
	F15	Mean	1.582e+04+	3.509e+04+	2.930e+04+	4.998e+04+	9.997e+03=	1.044e+04
		Std	2.319e+03	8.450e+03	4.779e+03	2.468e+03	8.541e+02	8.945e+02
		FEs	29050	15505	14650	30050	14401	20991
	F16	Mean	1.901e+04+	1.907e+04+	2.030e+04+	5.386e+04+	1.919e+04+	1.479e+04
		Std	1.411e+03	7.513e+02	3.670e+02	2.376e+03	3.583e+02	1.049e+03
		FEs	27057	15850	12301	29007	14150	19655
	F17	Mean	3.795e+03-	3.905e+03-	4.734e+03-	9.930e+03+	4.601e+03-	6.293e+03
		Std	1.324e+02	1.788e+02	1.600e+02	6.814e+02	2.125e+02	7.428e+02
		FEs	29950	14800	14900	29255	13600	17300
	F18	Mean	1.576e+04+	1.858e+04+	2.196e+04+	3.921e+04+	1.664e+04+	3.656e+03
		Std	2.189e+03	8.414e+02	8.549e+02	2.057e+03	9.705e+02	48.884
		FEs	20050	14850	14805	29650	13905	20562
	F19	Mean	1.024e+04+	1.878e+04+	1.348e+04+	2.381e+05+	1.097e+04+	8.992e+03
		Std	9.653e+02	1.787e+03	7.823e+02	1.130e+05	4.336e+02	6.333e+02
		FEs	25551	15855	12405	29605	12100	22302
	F20	Mean	4.618e+08=	2.743e+09+	3.816e+08=	1.000e+10+	2.062e+08=	2.214e+08
		Std	3.678e+08	9.151e+08	5.721e+07	0.000	6.722e+07	7.689e+07
		FEs	29955	15250	15050	150	14605	23257
	+/ = /−		15/2/3	18/1/2	16/2/2	20/0/0	18/2/2	
	Avg. Rank		4.740	7.760	7.550	10.135	6.710	

Table 9
Box-and-whisker plots of each algorithm over 30 runs.



(*continued on next page*)

Table 9 (*continued*).

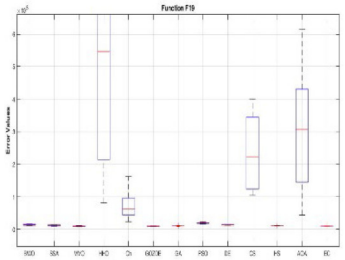
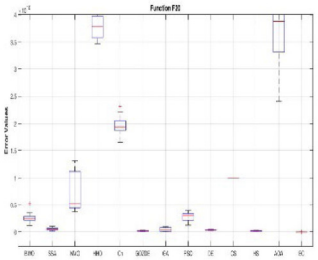



Table 10
The experimental results of the up-to-date metaheuristics for first nine real-world problems.

		BWO	SSA	MVO	HHO	ChOA	GOZDE
R1	Mean	24.981	19.073	15.001	24.749	22.323	14.574
	Std	1.800	4.360	5.662	1.234	2.828	6.803
R2	Mean	2.287e+02	2.639e+02	2.215e+02	2.459e+02	4.205e+02	2.200e+02
	Std	33.122	34.343	4.833	16.773	2.675e+02	0.000
R3	Mean	3.354	1.871	1.739	2.188	1.828	1.756
	Std	0.665	0.133	0.010	0.184	0.023	0.027
R4	Mean	24.314	37.4184	29.038	45.062	47.556	21.388
	Std	3.950	10.287	6.772	5.828	6.740	4.837
R5	Mean	29.062	30.862	27.947	52.859	39.062	26.709
	Std	1.829	4.419	4.278	1.318	3.407	4.021
R6	Mean	3.131e+04	1.553e+04	1.549e+04	1.661e+04	1.560e+04	1.545e+04
	Std	2.200e+04	44.841	24.478	1.239e+03	1.219e+02	1.046
R7	Mean	1.904e+04	1.907e+04	1.931e+04	2.105e+04	2.184e+04	1.917e+04
	Std	2.317e+02	1.621e+02	1.735e+02	1.703e+03	2.514e+03	2.559e+02
R8	Mean	6.081e−04	3.663e−07	6.679e−10	2.784e−09	1.950e−04	2.317e−10
	Std	0.002	2.025e−06	5.734e−10	5.915e−09	4.553e−04	9.540e−10
R9	Mean	0.019	0.155	0.060	0.143	0.173	0.017
	Std	0.008	0.030	0.025	0.014	0.035	0.006

Table 11
The experimental results of the classical metaheuristics for first nine real-world problems.

		GA	PSO	DE	CS	HS	GOZDE
R1	Mean	20.862	21.659	19.620	16.374	15.254	14.574
	Std	4.113	2.445	1.433	1.950	4.188	6.803
R2	Mean	2.210e+02	2.417e+02	2.286e+02	2.233e+02	2.231e+02	2.200e+02
	Std	5.659	11.796	10.307	4.845	10.721	0.000
R3	Mean	2.955	1.907	2.211	1.902	2.333	1.756
	Std	0.485	0.093	0.211	0.132	0.203	0.027
R4	Mean	23.691	55.330	45.726	30.931	23.405	21.388
	Std	5.299	8.186	4.237	3.148	1.944	4.837
R5	Mean	28.362	39.890	35.684	30.754	28.170	26.709
	Std	2.672	4.446	3.776	1.788	3.819	4.021
R6	Mean	1.547e+04	1.551e+04	1.545e+04	1.545e+04	1.547e+04	1.545e+04
	Std	23.271	30.449	6.149	3.118	17.136	1.046
R7	Mean	1.955e+04	1.904e+04	1.961e+04	1.907e+04	2.049e+04	1.917e+04
	Std	8.170e+02	1.704e+02	5.287e+02	1.125e+02	1.065e+03	2.559e+02
R8	Mean	0.0014	1.425e−04	8.854e−04	2.801e−05	2.064e−07	2.317e−10
	Std	0.002	3.074e−04	0.003	4.419e−05	4.003e−07	9.540e−10
R9	Mean	0.229	0.222	0.117	0.156	0.256	0.017
	Std	0.056	0.024	0.014	0.021	0.019	0.006

seen from the results of 100D CEC problems given in Table 7, the introduced GOZDE algorithm performs significantly better than the compared up-to-date metaheuristics on the majority of test functions. More specifically, it is better than BWO on 14 functions, SSA on 14 functions, MVO on 12 functions, HHO on 20 functions, ChOA on 20 functions, AOA on 20 functions and EBOwithCMAR on 9 functions out of 20 benchmark functions according to the Wilcoxon test results. It is worth mentioning that the proposed algorithm illustrates similar performance with BWO on three functions (F11, F14, F17), SSA on one function (F10), MVO on four functions (F8, F10, F12, F19) and EBOwithCMAR on four functions (F2, F4, F6, F14). The mean solutions obtained by the GOZDE algorithm are totally better than the mean results produced by the HHO, ChOA and AOA. From the same table, the Friedman test results indicate that the proposal has the minimum average rank compared to the BWO, SSA, MVO, HHO and ChOA algorithms. It is apparent that the presented GOZDE algorithm is the best performing algorithm with a rank of 2.835 according

Table 12
Comparison results of harmonic elimination problem for single-phase full-bridge inverter.

	α_1	α_2	Best Error
BWO	15.42512	87.42819	1.09383e−08
SSA	15.40335	87.42329	6.02218e−07
MVO	15.38214	87.28504	2.04612e−05
HHO	15.48645	87.52902	1.20699e−05
ChOA	15.74909	87.25908	1.23004e−04
GA	15.42741	87.42826	1.30223e−09
PSO	15.40607	87.36609	3.96858e−06
DE	15.39579	87.09147	1.03474e−04
CS	14.79265	87.19389	4.17579e−04
HS	15.36275	87.57650	2.45155e−05
GOZDE	15.42857	87.42857	9.29190e−17

Table 13
Comparison results of harmonic elimination problem for single-phase half-bridge inverter.

	α_1	α_2	Best Error
BWO	10.46467	88.37026	2.68023e−04
SSA	10.19792	88.50981	2.07405e−08
MVO	10.25297	88.49358	9.76037e−06
HHO	10.20329	88.70067	1.38135e−04
ChOA	10.18536	88.44779	1.68943e−05
GA	10.19743	88.51067	8.83265e−09
PSO	10.34630	88.52983	6.88847e−05
DE	10.00000	87.09147	1.03474e−04
CS	10.41750	87.80152	0.00196
HS	10.18583	88.53361	2.03298e−06
GOZDE	10.19771	88.51214	4.08694e−16

to the Friedman test. It can be concluded that the collaboration of the zones gives the proposed method ability to provide more quality solutions than the compared up-to-date methods.

From the comparative results of the classical metaheuristics given in Table 8, the proposed GOZDE maintains its superior performance against the classical metaheuristics for high-dimensional problems. In fact, it achieves better results than GA on 15 functions, PSO on 18 functions, DE on 16 functions, CS on 20 functions and HS on 20 functions out of 20 benchmark functions. As it can be seen from the results of the Friedman test, GOZDE seems to be very robust with a lower rank value than those obtained by the classical metaheuristics. The statistical analyses also prove that the proposed approach still provides more efficient solutions than classical ones for 100D problems. The results are also in good agreement with what was found by the earlier analyses.

In order to draw a comprehensive picture of the results obtained, the comparative Whisker-box plot is presented for the aforementioned 12 metaheuristic algorithms, which shows the upper and the lower quartiles of the results for each algorithm. The whisker-box plots are given in Table 9. For hybrid functions (F1–F10), the distribution of the results obtained by GOZDE is very low, is almost near to the lower limit and shows a little increase only for F6 and F10. The results of the plots express that the proposed algorithm demonstrates a smaller variance (variety of solutions) than the compared algorithms on the most of the hybrid functions. Regarding composition functions (F11–F20), it can be observed that the range of the results obtained by GOZDE slightly increases, but the median values (red line) of GOZDE are still less than the medians of the compared algorithms on the majority of functions. It is clear that the performance of the GOZDE algorithm on hybrid and composition functions is promising, and is similar or superior to those found with the compared algorithms. To sum up, these findings validate that the proposed approach does not only generate effective solutions for low-dimensional problems, but also preserves the efficiency of its performance on high-dimensional problems even if the dimensions of the problems increases.

3.3. Real-world problems

In this section, to show the potential of the GOZDE algorithm on the special problems with different search spaces, GOZDE is adapted to 10 real-world problems which have different search spaces, equality and inequality constraints. The parameters of all algorithms are kept the same with the previous experiments. The real-world problems considered in this study are: (R1) parameter estimation of FM sound waves [34], (R2) transmission network expansion planning [34], (R3) welded beam design [35,36], (R4) messenger: spacecraft trajectory optimization problem [34,37], (R5) Cassini 2: spacecraft trajectory optimization problem [34], (R6–R7) static economic load dispatch (ELD) problems [34,38], (R8) FIR filter design problem [31], (R9) IIR filter design problem [39,40] and (R10) harmonic elimination problem [41–44].

According to the results in Tables 10–11, GOZDE comes up with the best solutions for the problems R1 and R2. In particular, for R2, the solution found by GOZDE has a great difference from the other algorithms, getting a best mean value of 220.000 with zero standard deviation. As for R3, GOZDE fails to achieve the best solution but it is still robust and takes the title of the second-best algorithm. As this problem includes many constraints, it may be difficult to jump from out of local minima in solving this problem for the GOZDE algorithm. Among the compared metaheuristics, none of the algorithms are capable of providing superior performance than the GOZDE algorithm on the problems R4, R5, R6, R8 and R9. In fact, the proposed algorithm does not only give satisfactory solutions but also obtains the smallest Std values on most of the real-world problems. The efficient results obtained by GOZDE are mainly because the search mechanisms of octal zones working together may allow investigating different regions in the complex search spaces of the real-world problems. The comparative results for the problems R8 and R9 also demonstrate that the proposed algorithm can obtain better filter design performances than its opponents. The results obtained by the proposed algorithm for single-phase full and half bridge inventers are given in Tables 12–13. According to these tables, GOZDE is able to better quality of final solutions than that of the compared algorithms on both scenarios. The harmonics found by GOZDE are visualized in Fig. 7, it can be seen from this figure that GOZDE suppresses the selected harmonics effectively. From the results of the real-world problems, we can draw the conclusion that GOZDE does not only solve the real-world problems with good accuracy but also demonstrates more stable and robust performance.

4. Conclusions

This paper presents a novel population-based method called the geometric octal zones distance estimation optimization algorithm (GOZDE) to address the global optimization problems. In GOZDE, the whole population consists of the eight zones operating cooperatively via unique search mechanisms. These search mechanisms can make GOZDE gain the ability to discover different regions simultaneously in the search space. The sets of low and high dimensional benchmarks were used to evaluate the performance of GOZDE that is compared with the twelve well-known metaheuristic methods. The robustness of the proposed GOZDE is also tested on 10 real-world problems. Moreover, to

Table A.1
Experiments of parameter estimation (1/6).

			No parameter	[0 1]	[0 2]	[0 3]	[0 4]	[0 5]
Unimodal	F1	Mean	1.076e−05	1.882e−05	5.020e−05	2.461e−04	0.139	0.011
	F2	Mean	1.954e−10	2.806e−11	4.246e−11	4.226e−11	7.654e−11	8.281e−11
	F3	Mean	2.535e−06	1.999e−05	0.013	0.032	0.023	0.017
	F4	Mean	0.002	0.006	0.022	0.026	0.031	0.042
	F5	Mean	11.171	0.291	0.232	0.917	0.751	0.859
	F6	Mean	0.116	0.105	0.074	0.077	0.092	0.420
Multimodal	F7	Mean	0.002	1.152e−04	3.007e−04	7.734e−04	0.001	0.002
	F8	Mean	−1.110e+04	−1.156+04	−1.244e+04	−1.228e+04	−1.207e+04	−1.110e+04
	F9	Mean	4.565e−06	8.207e−05	7.159e−05	1.898e−04	4.932e−04	1.0897
	F10	Mean	0.001	0.003	0.031	0.017	0.219	0.561
	F11	Mean	3.278e−05	6.6352e−05	6.193e−04	0.002	0.009	0.011
	F12	Mean	0.007	0.001	0.001	0.001	0.001	0.001
	F13	Mean	0.038	0.003	0.002	0.004	0.003	0.003
Fixed-dimension multimodal	F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998
	F15	Mean	5.688e−04	5.194e−04	6.674e−04	3.902e−04	4.378–04	4.477e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.398	0.398	0.398	0.398	0.398	0.398
	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.235	−3.245	−3.183	−3.260	−3.244	−3.212
	F21	Mean	−10.151	−10.152	−9.400	−10.151	−10.149	−10.148
	F22	Mean	−9.065	−9.639	−9.734	−10.401	−8.396	−9.724
	F23	Mean	−8.523	−9.866	−10.535	−10.531	−10.534	−10.529
No. best			2	3	1	0	0	0

Table A.2
Experiments of parameter estimation (2/6).

			[−1 1]	[−1 0]	[−1 2]	[−1 3]	[−1 4]	[−1 5]
Unimodal	F1	Mean	3.861e−06	6.378e−06	2.850e−05	7.230e−05	9.076e−05	1.980e−04
	F2	Mean	1.422e−10	1.606e−11	9.827e−11	1.243e−10	7.428e−11	4.847e−11
	F3	Mean	2.106e−05	4.567e−05	0.002	0.002	0.026	0.025
	F4	Mean	0.002	0.002	0.004	0.014	0.025	0.058
	F5	Mean	3.862	4.242	0.640	0.450	0.717	2.141
	F6	Mean	0.103	0.055	0.135	0.124	0.540	0.233
	F7	Mean	0.001	0.001	0.001	0.001	0.001	0.001
Multimodal	F8	Mean	−1.179e+04	−1.248e+04	−1.181e+04	−1.241e+04	−1.175e+04	−1.130e+04
	F9	Mean	8.994e−05	5.129e−05	8.550e−05	2.454e−04	4.184e−04	0.001
	F10	Mean	0.001	0.001	0.01	0.026	0.139	0.253
	F11	Mean	2.727e−04	1.398e−05	4.531e−05	6.985e−05	0.002	0.037
	F12	Mean	0.002	0.001	0.001	0.001	0.002	0.001
	F13	Mean	0.012	0.010	0.006	0.001	0.009	0.005
Fixed-dimension multimodal	F14	Mean	1.031	0.998	0.998	0.998	0.998	0.998
	F15	Mean	3.994e−04	3.659e−04	4.311e−04	4.744e−04	4.510e−04	4.740e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.397	0.397	0.397	0.398	0.398	0.398
	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.246	−3.266	−3.250	−3.243	−3.248	−3.232
	F21	Mean	−10.151	−10.153	−9.400	−10.151	−10.007	−10.146
	F22	Mean	−9.335	−9.954	−9.402	−10.398	−10.399	−10.393
	F23	Mean	−10.087	−10.089	−10.535	−10.532	−9.862	−10.532
No. best			2	9	0	3	1	1

verify the experimental outcomes, three statistical tests were conducted. The experimental results clearly demonstrate that the proposed algorithm exhibits more effective performance than the compared algorithms in solving both the numerical and real-world problems. There are some limitations in this study. First, the parameter vector can need to be more analysing in order to obtain a better configuration. Second, the performance of the proposed algorithm should be measured on higher dimensional functions, such as 1000 and 10000. Third, the first zone, which represents the reference area, can be replaced with one of the rest of the zones to observe the effect of the change on the solution qualities. Lastly, since the proposed algorithm can tend to be stuck in local minima for the problems that include many constraints, the different improvement mechanisms, such as the chaos-based and opposition-based, can be integrated with GOZDE to enhance its performance. The next task could be to overcome these limitations. Apart from this, the proposal of discrete or multi-objective versions of the developed algorithm could be some significant contributions as well.

CRediT authorship contribution statement

Yiğit Çağatay Kuyu: Conceptualization, Methodology, Software, Formal analysis, Validation, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Fahri Vatansever:** Conceptualization, Methodology, Supervision, Writing – review & editing, Investigation, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table A.3
Experiments of parameter estimation (3/6).

			[−2 0]	[−2 1]	[−2 2]	[−2 3]	[−2 4]	[−2 5]
Unimodal	F1	Mean	5.471e−06	4.513e−06	7.136e−05	1.545e−05	3.533e−04	0.011
	F2	Mean	1.954e−10	2.197e−10	3.342e−10	2.952e−10	1.844e−10	8.586e−11
	F3	Mean	0.065	0.015	0.066	0.019	0.014	0.068
	F4	Mean	0.004	0.004	0.005	0.006	0.042	0.011
	F5	Mean	0.857	2.310	1.530	1.506	1.438	1.220
	F6	Mean	0.072	0.092	0.181	0.156	0.089	0.101
	F7	Mean	0.002	0.001	0.001	0.003	0.003	0.006
Multimodal	F8	Mean	−1.226e+04	−1.206+04	−1.202e+04	−1.206e+04	−1.208e+04	−1.218e+04
	F9	Mean	7.216e−05	4.791e−04	1.641e−04	8.656e−05	0.694	0.003
	F10	Mean	0.004	0.004	0.005	0.004	0.069	0.161
	F11	Mean	1.374e−04	1.966e−05	5.462e−05	6.215e−05	0.001	0.007
	F12	Mean	0.003	0.004	0.003	0.004	0.004	0.002
	F13	Mean	0.018	0.030	0.022	0.019	0.026	0.014
Fixed-dimension multimodal	F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998
	F15	Mean	4.733e−04	4.702–04	3.856e−04	4.689e−04	5.337e−04	7.538e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.398	0.398	0.398	0.398	0.398	0.398
	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.242	−3.202	−3.250	−3.250	−3.237	−3.248
	F21	Mean	−10.153	−10.152	−10.129	−10.151	−10.151	−10.147
	F22	Mean	−9.067	−10.401	−9.734	−9.734	−9.830	−9.727
	F23	Mean	−10.536	−9.865	−9.865	−10.534	−10.533	−9.856
No. best			0	0	0	0	0	0

Table A.4
Experiments of parameter estimation (4/6).

			[−3 0]	[−3 1]	[−3 2]	[−3 3]	[−3 4]	[−3 5]
Unimodal	F1	Mean	3.169e−04	2.080e−04	1.255e−04	1.630e−04	6.383e−05	0.023
	F2	Mean	7.031e−10	4.445e−10	6.060e−10	2.395e−10	1.732e−10	2.727e−10
	F3	Mean	0.137	0.072	0.030	0.091	0.021	0.105
	F4	Mean	0.004	0.015	0.004	0.008	0.018	0.032
	F5	Mean	2.377	1.022	1.178	1.542	0.599	1.432
	F6	Mean	0.090	0.077	0.080	0.122	0.122	0.164
	F7	Mean	0.003	0.003	0.003	0.005	0.008	0.005
Multimodal	F8	Mean	−1.310e+04	−1.340e+04	−1.494e+04	−1.344e+04	−1.358e+04	−1.256e+04
	F9	Mean	0.225	0.005	0.704	0.591	0.097	0.074
	F10	Mean	0.010	0.004	0.003	0.015	0.014	0.092
	F11	Mean	8.780e−04	8.294e−05	4.831e−04	5.684e−05	8.778e−04	0.002
	F12	Mean	0.005	0.004	0.006	0.006	0.004	0.001
	F13	Mean	0.023	0.034	0.022	0.057	0.015	0.023
Fixed-dimension multimodal	F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998
	F15	Mean	4.990e−04	5.414e−04	4.336e−04	4.429e−04	4.615e−04	4.842e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.398	0.398	0.398	0.398	0.398	0.398
	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.280	−3.202	−3.201	−3.203	−3.211	−3.241
	F21	Mean	−10.153	−9.400	−10.152	−9.392	−10.151	−10.145
	F22	Mean	−10.402	−9.734	−9.734	−9.734	−9.734	−10.388
	F23	Mean	−9.999	−10.536	−10.536	−10.534	−10.535	−10.429
No. best			2	0	0	0	0	0

Table A.5
Experiments of parameter estimation (5/6).

			[−4 0]	[−4 1]	[−4 2]	[−4 3]	[−4 4]	[−4 5]
Unimodal	F1	Mean	0.007	1.020e−04	8.813e−04	3.833e−04	0.016	5.637e−04
	F2	Mean	5.678e−10	4.369e−10	5.366e−10	5.074e−10	3.581e−10	3.477e−10
	F3	Mean	0.368	0.012	0.187	0.159	0.388	0.081
	F4	Mean	0.011	0.008	0.008	0.009	0.011	0.022
	F5	Mean	2.806	2.181	3.030	3.434	2.198	3.315
	F6	Mean	0.097	0.093	0.092	0.104	0.172	0.519
	F7	Mean	0.009	0.007	0.007	0.008	0.010	0.010
Multimodal	F8	Mean	−1.646e+04	−1.356e+04	−1.410e+04	−1.279e+04	−1.377e+04	−1.541e+04
	F9	Mean	6.394	1.913	1.633	3.577	0.424	0.983
	F10	Mean	0.029	0.011	0.022	0.013	0.044	0.040
	F11	Mean	4.005e−04	1.158e−04	1.353e−04	0.001	0.001	0.004
	F12	Mean	0.003	0.004	0.006	0.007	0.006	0.007
	F13	Mean	0.058	0.051	0.111	0.060	0.033	0.040

(*continued on next page*)

Table A.5 (*continued*).

			[−4 0]	[−4 1]	[−4 2]	[−4 3]	[−4 4]	[−4 5]
	F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998
	F15	Mean	4.231e−04	4.487e−04	5.147e−04	4.814e−04	7.070e−04	5.819e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.398	0.398	0.398	0.398	0.398	0.398
Fixed-dimension multimodal	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.230	−3.238	−3.250	−3.213	−3.216	−3.249
	F21	Mean	−10.152	−10.152	−10.151	−10.152	−10.150	−10.147
	F22	Mean	−9.735	−9.734	−10.402	−9.734	−10.402	−8.966
	F23	Mean	−10.536	−10.536	−9.195	−9.865	−10.110	−10.534
No. best			1	0	0	0	0	0

Table A.6
Experiments of parameter estimation (6/6).

			[−5 0]	[−5 1]	[−5 2]	[−5 3]	[−5 4]	[−5 5]
	F1	Mean	2.509e−04	8.810e−04	4.882e−04	9.029e−04	0.005	0.001
	F2	Mean	5.356e−10	7.148e−10	5.318e−10	5.550e−10	1.642e−10	1.566e−10
	F3	Mean	0.299	0.279	0.388	0.327	0.505	0.109
Unimodal	F4	Mean	0.014	0.017	0.022	0.027	0.047	0.028
	F5	Mean	1.832	5.328	5.630	1.890	7.322	4.107
	F6	Mean	0.080	0.087	0.078	0.085	0.115	0.114
	F7	Mean	0.008	0.008	0.009	0.012	0.012	0.009
	F8	Mean	−1.419e+04	−1.558e+04	−1.557e+04	−1.427e+04	−1.379e+04	−1.479e+04
	F9	Mean	3.214	5.073	4.487	0.440	4.068	2.037
	F10	Mean	0.075	0.019	0.008	0.009	0.101	0.111
Multimodal	F11	Mean	6.977e−05	0.009	1.170e−04	3.495e−04	0.001	0.001
	F12	Mean	0.006	0.005	0.005	0.005	0.007	0.005
	F13	Mean	0.081	0.064	0.048	0.096	0.099	0.037
	F14	Mean	0.998	0.998	0.998	0.998	0.998	0.998
	F15	Mean	5.602e−04	6.196e−04	6.386e−04	6.771e−04	7.218e−04	7.514e−04
	F16	Mean	−1.031	−1.031	−1.031	−1.031	−1.031	−1.031
	F17	Mean	0.398	0.398	0.398	0.398	0.398	0.398
Fixed-dimension multimodal	F18	Mean	3.000	3.000	3.000	3.000	3.000	3.000
	F19	Mean	−3.862	−3.862	−3.862	−3.862	−3.862	−3.862
	F20	Mean	−3.213	−3.233	−3.255	−3.250	−3.214	−3.220
	F21	Mean	−10.149	−10.147	−10.146	−10.141	−10.141	−10.140
	F22	Mean	−10.302	−9.734	−8.970	−10.396	−9.734	−10.390
	F23	Mean	−10.430	−10.531	−10.532	−9.865	−10.509	−10.522
No. best			0	0	0	0	0	0

Table A.7
Numerical results of the proposed algorithm implemented with 2 and 4 zones.

Zone number		2	4			2	4
F1	*Mean*	4.310e+03	1.379e+04	F13	*Mean*	2.041e+09	6.832e+08
	Std	4.864e+03	8.540e+03		*Std*	2.972e+08	3.105e+08
F2	*Mean*	1.737e+22	2.638e+16	F14	*Mean*	1.301	0.998
	Std	6.176e+22	1.088e+17		*Std*	1.255	1.523e−15
F3	*Mean*	1.302e+05	9.668e+04	F15	*Mean*	0.001	7.873e−04
	Std	9.446e+04	3.767e+04		*Std*	0.001	2.102e−04
F4	*Mean*	15.038	42.978	F16	*Mean*	−1.030	−1.031
	Std	10.135	8.400		*Std*	0.001	5.709e−07
F5	*Mean*	3.894e+08	1.542e+08	F17	*Mean*	0.464	0.397
	Std	9.199e+07	5.250e+07		*Std*	0.212	3.364e−07
F6	*Mean*	5.015e+03	1.116e+04	F18	*Mean*	3.018	3.000
	Std	1.314e+04	5.106e+03		*Std*	0.060	1.642e−07
F7	*Mean*	48.153	1.673	F19	*Mean*	−3.861	−3.862
	Std	29.186	1.768		*Std*	0.001	9.887e−06
F8	*Mean*	−5.505e+03	−7.652e+03	F20	*Mean*	−3.160	−3.269
	Std	5.879e+02	1.080e+03		*Std*	0.094	0.059
F9	*Mean*	5.735e+02	4.281e+02	F21	*Mean*	−5.152	−10.153
	Std	54.863	55.689		*Std*	2.087	1.135e−04
F10	*Mean*	13.379	15.749	F22	*Mean*	−4.930	−8.367
	Std	4.729	1.260		*Std*	2.203	3.167
F11	*Mean*	3.357e+02	1.112e+02	F23	*Mean*	−5.731	−9.127
	Std	1.032e+02	48.793		*Std*	2.386	2.874
F12	*Mean*	9.582e+08	2.938e+08				
	Std	1.932e+08	1.316e+08				

Appendix

See Tables A.1–A.7

References

[1] S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: Theory and application, Adv. Eng. Softw. 105 (2017) 30–47, http://dx.doi.org/10.1016/j.advengsoft.2017.01.004.

[2] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, USA, 1989.

[3] R. Storn, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.

[4] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, 1995, pp. 39–43, http://dx.doi.org/10.1109/mhs.1995.494215.

[5] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005, -TR06.

[6] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248, http://dx.doi.org/10.1016/j.ins.2009.03.004.

[7] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2) (2016) 495–513, http://dx.doi.org/10.1007/s00521-015-1870-7.

[8] M. Gendreau, An introduction to Tabu search, in: F. Glover, G.A. Kochenberger (Eds.), Handbook of Metaheuristics, Springer, Boston, 2003, pp. 37–54, http://dx.doi.org/10.1007/0-306-48056-5_2.

[9] D. Henderson, S.H. Jacobson, A.W. Johnson, The theory and practice of simulated annealing, in: F. Glover, G.A. Kochenberger (Eds.), Handbook of Metaheuristics, Springer, Boston, 2003, pp. 287–319, http://dx.doi.org/10.1007/0-306-48056-5_10.

[10] H.R. Lourenco, O.C. Martin, T. Stützle, Iterated local search, in: F. Glover, G.A. Kochenberger (Eds.), Handbook of Metaheuristics, Springer, Boston, 2003, pp. 320–353.

[11] C.A.C. Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, second ed., Springer, Boston, 2007.

[12] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: Harmony search, Simulation 76 (2) (2001) 60–68, http://dx.doi.org/10.1177/003754970107600201.

[13] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249, http://dx.doi.org/10.1016/j.knosys.2015.07.006.

[14] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, C. Saremi, H. Faris, S.M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191, http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.

[15] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872, http://dx.doi.org/10.1016/j.future.2019.02.028.

[16] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, A.H. Gandomi, The arithmetic optimization algorithm, Comput. Methods Appl. Mech. Engrg. 376 (2021) 113609, http://dx.doi.org/10.1016/j.cma.2020.113609.

[17] O. Abu Arqub, Z. Abo-Hammour, S. Momani, N. N. Shawagfeh, Solving singular two-point boundary value problems using continuous genetic algorithm, Abstr. Appl. (2012) http://dx.doi.org/10.1155/2012/205391.

[18] Z. Abo-Hammour, O.A. Arqub, S. Momani, N. Shawagfeh, Optimization solution of troesch's and bratu's problems of ordinary type using novel continuous genetic algorithm, Discrete Dyn. Nat. Soc. (2014) http://dx.doi.org/10.1155/2014/401696.

[19] Z. Abo-Hammour, O.A. Arqub, O. Alsmadi, S. S. Momani, A. Alsaedi, An optimization algorithm for solving systems of singular boundary value problems, Appl. Math. Inf. Sci. 8 (6) (2014) 2809, http://dx.doi.org/10.12785/amis/080617.

[20] O.A. Arqub, Z. Abo-Hammour, Numerical solution of systems of second-order boundary value problems using continuous genetic algorithm, Inform. Sci. 279 (2014) 396–415, http://dx.doi.org/10.1016/j.ins.2014.03.128.

[21] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82, http://dx.doi.org/10.1109/4235.585893.

[22] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Cuckoo search algorithm: a meta-heuristic approach to solve structural optimization problems, Eng. Comput. 29 (2013) 17–35, http://dx.doi.org/10.1007/s00366-011-0241-y.

[23] H. Vahideh, A.A.P. Kazem, Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems, Eng. Appl. Artif. Intell. 87 (2020) 103249, http://dx.doi.org/10.1016/j.engappai.2019.103249.

[24] M. Khishe, M.R. Mosavi, Chimp optimization algorithm, Expert Syst. Appl. 149 (2020) 113338, http://dx.doi.org/10.1016/j.eswa.2020.113338.

[25] A. Kumar, R.K. Misra, D. Singh, Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase, in: IEEE Congress on Evolutionary Computation, CEC, 2017, pp. 1835–1842, http://dx.doi.org/10.1109/CEC.2017.7969524.

[26] T.R. Holmes, Caesar's Conquest of Gaul, first ed., Clarendon Press, UK, 1911.

[27] A.D. Kahn, The Education of Julius Caesar: A Biography, a Reconstruction, iUniverse, USA, 2000.

[28] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67, http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

[29] S. Mirjalili, The ant lion optimizer, Adv. Eng. Softw. 83 (2015) 80–98, http://dx.doi.org/10.1016/j.advengsoft.2015.01.010.

[30] G. Wu, R. Mallipeddi, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the Cec 2017 Competition on Constrained Real-Parameter Optimization, Technical Report.R., National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, 2016.

[31] Y.Ç. Kuyu, F. Vatansever, A new intelligent decision making system combining classical methods, evolutionary algorithms and statistical techniques for optimal digital FIR filter design and their performance evaluation, AEU-Int. J. Electron. Commun. 70 (12) (2016) 1651–1666, http://dx.doi.org/10.1016/j.aeue.2016.10.004.

[32] E. Nikolic-Dorić, K. Čobanović, Z. Lozanov-Crvenković, Statistical graphics and experimental data, in: 7th International Conference on Teaching Statistics, 2006.

[33] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evolut. Comput. 1 (1) (2011) 3–18, http://dx.doi.org/10.1016/j.swevo.2011.02.002.

[34] S. Das, P.N. Suganthan, Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems, Technical Report, Jadavpur University and Nanyang Technological University, 2010.

[35] S.S. Rao, Engineering Optimization: Theory and Practice, fourth ed., Wiley, 1996.

[36] A. Kaveh, V.R. Mahdavi, Colliding bodies optimization: a novel meta-heuristic method, Comput. Struct. 139 (2014) 18–27, http://dx.doi.org/10.1016/j.compstruc.2014.04.005.

[37] B. Addis, A. Cassioli, M. Locatelli, F. Schoen, Global optimization for the design of space trajectories, Comput. Optim. Appl. 48 (3) (2011) 635–652, http://dx.doi.org/10.1007/s10589-009-9261-6.

[38] Y.Ç. Kuyu, N. Erdem, F. Vatansever, G. Yılmaz, Comparison of the evolutionary algorithm's performances on power flow analysis, Pamukkale Univ. J. Eng. Sci. 24 (2) (2018) 167–172, http://dx.doi.org/10.5505/pajes.2016.89266.

[39] E. Ifeachor, B. Jervis, Digital Signal Processing, a Practical Approach, second ed., Pearson Education, UK, 2002.

[40] Y.Ç. Kuyu, F. Vatansever, Design of IIR digital filters using the current evolutionary algorithm, in: 5th International Symposium on Innovative Technologies in Engineering and Science, 2017, pp. 359–367, http://dx.doi.org/10.1109/ISMSIT.2018.8567060.

[41] R. Rajaram, K. Palanisamy, S. Ramasamy, P. Ramanathan, Selective harmonic elimination in PWM inverter using fire fly and fire works algorithm, Int. J. Innov. Res. Adv. Eng. 1 (8) (2015) 55–62.

[42] T.S. Babu, K. Priya, D. Maheswaran, K.S. Kumar, N. Rajasekar, Selective voltage harmonic elimination in PWM inverter using bacterial foraging algorithm, Swarm Evol. Comput. 20 (2015) 74–81, http://dx.doi.org/10.1016/j.swevo.2014.11.002.

[43] F. Vatansever, Y.Ç. Kuyu, The harmonic elimination in inverters with metaheuristic approaches, Uludag Univ. J. Faculty Eng. 24 (2019) 383–396, http://dx.doi.org/10.17482/uumfd.595233.

[44] H.S. Patel, R.G. Hoft, Generalized techniques of harmonic elimination and voltage control in thyristor inverters: part I - harmonic elimination, IEEE Trans. Ind. Appl. IA- 9 (3) (1973) 310–317, http://dx.doi.org/10.1109/tia.1973.349908.

Yiğit Çağatay Kuyu received B.Sc. and M.Sc. degrees in Electronics Engineering from the Bursa Uludağ University, Bursa, Turkey, in 2013 and in 2016. He is currently a Ph.D. candidate at the Bursa Uludağ University, Engineering Faculty, Electrical-Electronics Engineering Department. His interests include digital signal processing, evolutionary computation and algorithm development.

Fahri Vatansever received the B.Sc., M.Sc., and Ph.D. degrees in Electrical-Electronics Engineering from Sakarya University, Turkey. He has been with the Bursa Uludağ University, Engineering Faculty, Electrical-Electronics Engineering Department, Turkey as a Professor. He performs research in the areas of circuits and systems, computational and artificial intelligence, algorithms, software development and engineering education.