Advances in Engineering Software 121 (2018) 147–166

Contents lists available at ScienceDirect

Advances in Engineering Software

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

Research paper

Pity beetle algorithm – A new metaheuristic inspired by the behavior of bark beetles

Nikos Ath. Kallioras[a], Nikos D. Lagaros[a,*], Dimitrios N. Avtzis[b]

[a] Institute of Structural Analysis & Antiseismic Research, Department of Structural Engineering, School of Civil Engineering, National Technical University of Athens, 9, Heroon Polytechniou Str., Zografou Campus, GR-15780 Athens, Greece

[b] Forest Research Institute, National Agricultural Research Foundation, Hellenic Agricultural Organization Demeter, GR-57006 Vassilika, Thessaloniki, Greece

ARTICLE INFO

Keywords:
Nature-inspired algorithms
Optimization problems
Benchmarking
Pityogenes chalcographus
Derivative free
Metaheuristics

ABSTRACT

In the past years a great variety of nature-inspired algorithms have proven their ability to efficiently handle combinatorial optimization problems ranging from design and form finding problems to mainstream economic theory and medical diagnosis. In this study, a new metaheuristic algorithm called Pity Beetle Algorithm (PBA) is presented and its efficiency against state-of-the-art algorithms is assessed. The proposed algorithm was inspired by the aggregation behavior, searching for nest and food, of the beetle named *Pityogenes chalcographus*, also known as six-toothed spruce bark beetle. This beetle has the ability to locate and harvest on the bark of weakened trees into a forest, while when its population exceeds a specific threshold it can infest healthy and robust trees as well. As it was proved in this study, PBA can be applied to NP-hard optimization problems regardless of the scale, since PBA has the ability to search for possible solutions into large spaces and to find the global optimum solution overcoming local optima. In this work, PBA was applied to well-known benchmark uni-modal and multi-modal, separable and non-separable unconstrained test functions while it was also compared to other well established metaheuristic algorithms implementing also the CEC 2014 benchmark and complexity evaluation tests.

1. Introduction

Engineers are challenged to pursue optimal solutions in complex systems, exploiting the ever increasing available computational power. Such systems must be as economic as possible yet efficient and/or strong enough to withstand the most demanding functional requirements arising during their service life. The traditional trial-and-error design approach is not sufficient to determine economic designs satisfying also durability criteria. On the other hand, design optimization provides a numerical procedure that can replace the traditional design approach with an automated one. During the last decades many numerical methods have been developed to meet the demands of engineering design optimization. These methods can be classified in two categories, gradient-based and derivative-free ones. Oft-used brute force design methodologies are systematically replaced by state-of-the-art nature-inspired algorithms, which are able to solve combinatorial problems. Heuristic and metaheuristic algorithms are nature-inspired or bio-inspired search procedures and belong to the derivative-free class of search algorithms. In nature, evolution is achieved by following rules such as the survival of the fittest or selective reproduction, while several species have also developed some sophisticated techniques concerning their search for food or nests. By studying these behaviors several algorithms referred to as heuristic and metaheuristic algorithms were inspired. Some of the modern and well established metaheuristic algorithms are: particle swarm optimization, inspired by the social behavior of bird flocking (PSO) [1], ant colony optimization, based on the foraging behavior of ants [2], artificial bee colony algorithm [3], firefly algorithm, inspired by the flashing behavior of fireflies [4], cuckoo search algorithm, mimicking the brood parasitism of cuckoo species [4], simulated annealing, based on the process of slowly cooling heated object to avoid defects [5], harmony search, inspired by the improvisation procedure of jazz bands [6], differential evolution [7], ray optimization, inspired by Snell's light refraction law [8], genetic algorithms, inspired by natural selection [9], bat algorithm, inspired by the echolocation behavior of microbats [9], dolphin echolocation, inspired by the biological sonar that dolphin use to navigate and hunt [10], krill herd, based on the simulation of the herding behavior of krill individuals [11]; variants of existing methods [12] chaotic swarming of particles, combining swarm intelligence and chaos theory [13] or the thermal exchange optimization, based on Newton's law of cooling [14].

Swarm optimization characterize a stochastic, population-based group of algorithms inspired by the social behavior of birds flocking,

* Corresponding author.
E-mail addresses: nkallioras@yahoo.com (N.A. Kallioras), nlagaros@central.ntua.gr (N.D. Lagaros), dimitrios.avtzis@fri.gr (D.N. Avtzis).

https://doi.org/10.1016/j.advengsoft.2018.04.007
Received 17 May 2017; Received in revised form 5 March 2018; Accepted 22 April 2018
Available online 03 May 2018
0965-9978/

fish schooling etc. [15]. Briefly, an initial population of particles (birds, fish, etc.) are positioned randomly into the multidimensional search space examined [15]. Every particle, whose position represents a solution, "travels" into the search space seeking for a better position/solution. During the iterations of the algorithm each particle adjusts its position based on its own experience, built by memorizing the best position encountered, as well as that of neighboring particles. PSO algorithms combine local search (self-experience) with global search (neighboring experience), aiming to balance exploration and exploitation. This procedure continues until the termination criterion is satisfied [16]. PSO algorithms have attracted a significant amount of interest in the past years [17–19], since it was proved efficient in handling real-world optimization problems too [20].

In this work, a new metaheuristic algorithm called pity beetle algorithm (PBA) [21] that belongs to the swarm optimization class of algorithms is presented and its efficiency is studied against state-of-the-art metaheuristic algorithms. In addition to the work presented in [21], a detailed description of the algorithm is presented regarding: i) the biology of the beetle, ii) the random sampling technique, iii) the sensitivity analyses over its parameters, and iv) the performance of the algorithm over various tests. The proposed algorithm was inspired by the aggregation behavior of *Pityogenes chalcographus* (Coleoptera, Scolytinae), also known as six-toothed spruce bark beetle. This beetle has the ability to infest large forest areas starting from a single brood (colony), employing a strategy that is more or less common among the species of Ipini tribe [22]. In particular, it usually attacks less healthy trees first, which are more susceptible, however, when its population exceeds a specific threshold, it can also attack robust trees as well. Initially, a number of male bark beetles (commonly called pioneer beetles) fly through the forest – searching in space for suitable (weakened) trees – solution vectors. These pioneer beetles are searching randomly into the forest for suitable hosts (trees), when a host is found, pheromone is spread, inviting other beetles to the host. Each male beetle creates a nest in the tree while feeding from the bark of the tree. Female beetles are attracted to the nests and new generations of beetles are created. Every offspring is then taking the role of pioneer beetle searching for a new host either randomly inside the forest or at close range from its birth position. Once a new host is identified, the procedure described above is repeated. As it was proved in this study, PBA can be applied to NP-hard optimization problems regardless of the scale, since PBA has the ability to search for possible solutions into large spaces and to find the global optimum solution while overcoming local optima. In this work, PBA was applied to well-known benchmark unimodal and multi-modal, separable and non-separable unconstrained test functions while it was also compared to other well established metaheuristic algorithms. In particular, the performance of the algorithm is examined for thirteen benchmarking test function while sensitivity analysis on the performance of the algorithm with reference to its parameters is also presented. Furthermore, the algorithm is compared against various modern and well-established metaheuristic algorithms in order to further evaluate its efficiency implementing also the CEC 2014 benchmark [23] and complexity evaluation tests.

2. Pity beetle algorithm

In this section the biology of the six-toothed bark beetle along with its numerical implementation are described.

2.1. Pityogenes chalcographus

Within the subfamily of Scolytinae, belong some of the most important forests pests worldwide. Despite the great differences among the various species, a common feature for all of them is the employment of a rather sophisticated system of communication that is based on chemical signals (pheromones). By using these signals, bark beetles are often able to cultivate mass population outbreaks with devastating effects on forest health. Among the different Eurasian bark beetle species, the six-toothed spruce bark beetle, *Pityogenes chalcographus* (Coleoptera, Scolytinae) is one of the most common and important bark beetles in Europe [24] that infests mainly Norway spruce (*Picea abies*) together with pines (*Pinus* sp.) and larch (*Larix decidua*) [22,25–27]. *P. chalcographus* occurs widely in central and northern Europe, while it has been also found in Elatia (Drama, Greece) [28]. *P. chalcographus* is able to establish two generations per year, depending on the environmental temperature [29]. However, under favorable conditions a third generation may also be established while in higher elevation, the number of generations is restricted to only one [25,30–32]. It exhibits a polygamous mating behavior, with the male mating with 3 to 6 females. The males of *P. chalcographus* bore into the phloem of already weakened trees excavating there a nuptial chamber. While feeding, they transform host terpenes into pheromones, attracting females, with which they mate in the nuptial chamber. From this chamber, in a star-like arrangement, females deposit 40–70 eggs in egg niches [31].

In summary, the process followed by *P. chalcographus* particles can be divided into stages. Initially, pioneer male beetles locate a suitable host (searching stage) by exploiting the chemical traits that weakened trees emit. When these beetles start feeding on the host, they produce an aggregation pheromone that attracts males and females (aggregation stage), increasing locally the population. Once a specific population threshold is surpassed, then the defence mechanisms of the host can no longer contain this mass infestation, while at this population level, both robust and weakened trees can be attacked. However, as an overcrowded host tree can have a negative impact on infestation (less feeding space, higher probability of infectious diseases), when the infestation density becomes too high then feeding beetles emit an anti-aggregation pheromone that discourages more beetles to attack this tree, turning them to other trees (anti-aggregation stage). By doing that, infestation expands gradually into the forest creating groups of dying or dead trees around the initially attacked tree.

2.2. Numerical implementation of PBA

The mathematical formulation of an optimization problem can be expressed in standard mathematical terms as a non-linear programming problem, which in can be stated as follows:

$$\min F(\boldsymbol{x}),\ \boldsymbol{x} = [x_1, x_2, ..., x_D]^T$$
$$L_i \le x_i \le U_i,\ i = 1, 2, ..., D \tag{1}$$

where $F(\boldsymbol{x})$: $R^D \rightarrow R$ is a real-valued objective function to be minimized (without loss of generality) and R is the set of real numbers, $\boldsymbol{x} \in R^D$ is the design variables vector of dimension D, while L_i and U_i are the lower and upper bounds of the ith design variable.

In this section, the numerical simulation of the six-toothed bark beetle's peculiar new generation creation behavior is provided. In particular, PBA is based on the implementation of the suitable host search and reproduction behavior of the bark beetles. Contrary to PSO the swarm in PBA is called *population*. In a D-dimensional search space, the jth member of the population at the gth search step (generation) of the algorithm is defined by its current position vector $\boldsymbol{x}_j^{(g)} \in R^D$. PBA consists of three basic steps: *Initialization, New Hypervolume Selection Pattern* and *Update Population Position*, while a population consists of males and females; some males act as *pioneer* particles that search for the most suitable host. In the first step of the algorithm, the first population (gallery/colony) position is generated randomly into the search space (first generation). In the second step, particles of the initial population move to other positions inside different hypervolumes, in order to create there new populations (second generation). In every generation new populations are generated and in the third step, new populations replace the previous ones. A flowchart of PBA is presented in Fig. 1, the algorithmic parameters are provided in Table 1, while the steps of the algorithm are described in the next sections. The three-step procedure is

```
1.  Begin
2.     g:=0
3.     Initialize() Eq. (6)
4.     Repeat
5.      For k := 1 To N_broods Do Begin
6.           For j := 1 To N_pop Do Begin
7.             NewHypervolumeSelectionPattern()
8.             CalculateF()
9.             FE:=FE+1
10.          End
11.          UpdatePopulationLocation()
12.       End
13.    Until TerminationCriterion(FE>FE_total)
14. End
```

Fig. 1. Pseudo-code of the pity beetle algorithm.

Table 1
Algorithmic parameters of PBA and their value range.

Parameter	Description	Value range	Proposed value range
N_{pop}	Population of pioneer particles	[10, 100]	[10, 50]
f_{nb}	Neighboring factor	[0.01, 0.20]	[0.01, 0.10]
f_{tn}	Fine tuning factor	[0.005, 0.05]	[0.005, 0.05]
f_{ms}	Mid-scale factor	[0.10, 1.00]	[0.40, 1.00]
f_{ls}	Large-scale factor	[1, 100]	[1, 100]
pr	Probability for choosing large-scale search or memory consideration	[0, 1]	[0, 1]
f_{FE}	Function evaluations multiplication factor	[0.05, 0.25]	[0.05, 0.25]

repeated until the termination criterion is satisfied. In our implementation the termination criterion is defined with the maximum number of function evaluations allowed (i.e. the total function evaluations permitted, FE_{total}).

2.2.1. Random sampling – hypervolume generation

Before describing the three main steps of the proposed algorithm a random sampling technique (RST) that is used into the steps of the algorithm is described. According to this sampling technique it is ensured that all portions of the search space are properly represented into N_{pop} samples. According to RST the uniform distribution function for each variable is divided into a number of segments of equal marginal probability. These segments are randomly selected for each variable and randomly shuffled among different variables to define the samples. According to RST a sample is constructed by dividing first the range of each of the D variables into N_{pop} non-overlapping equal segments, and then a sample with dimension D is created by randomly pairing the values of all parameters. Thus, the D-space is partitioned into N_{pop}^{D} segments and a single value is selected randomly from each cell, producing a set of size D. The values from the N_{pop}^{D} space are randomly matched to create N_{pop} samples. Consider the problem of generating N_{pop} samples composed by D independent uniformly distributed variables$x_{i,j} \in [l_i u_i]$, $i = 1, 2, \ldots, D$, $j = 1, 2, \ldots, N_{pop}$. According to RST the definition range for each of the D variables is divided into N_{pop} equal intervals, resulting in $\prod_{i=1}^{D} N_{pop} = N_{pop}^{D}$ cells, the requirement that each hyper-row and hyper-column contain only one sample is satisfied with the following expression used for determining the sample values:

$$x_{i,j} = F_x^{-1}\left(\frac{i - 1 + rx_i/(u_i - l_i)}{N_{pop}}\right) \tag{2}$$

where $rx_i \in [l_i u_i]$ is random number and F_x is the cumulative distribution function of the uniform distribution for $x_{i,j}$. The maximum number of combinations of the sampling procedure for the case of N_{pop} segments and D variables are:

$$\left(\prod_{d=0}^{N_{pop}-1}(N_{pop} - d)\right)^{D-1} = (N_{pop}!)^{D-1} \tag{3}$$

In general, the variables x_i are defined into the search space given in Eq. (1) that is defined by the global bounds $\boldsymbol{L}$ and $\boldsymbol{U}$; nevertheless, based on the steps of the proposed algorithm that will be described in the next sections of this study the bounds for the gth pursuit step might be different from the global ones. In particular, $l_i^{(g)}$ and $u_i^{(g)}$ are the lower and upper bounds of the ith dimension for the gth pursuit step, while the maximum sample distance $len_{\max}^{(g)}$ (also called pursuit distance, since RST will be incorporated into the steps of the proposed search algorithm) is calculated from the following expression:

$$len_{\max}^{(g)} = \|\boldsymbol{u}^{(g)} - \boldsymbol{l}^{(g)}\| = \sqrt{\sum_{i=1}^{D}\left(u_i^{(g)} - l_i^{(g)}\right)^2} \tag{4}$$

while the implementation of the sampling technique for the gth pursuit step will be denoted as:

$$\boldsymbol{x}_j = RST(\boldsymbol{l}^{(g)}, \boldsymbol{u}^{(g)}, D, N_{pop}) \tag{5}$$

In order to satisfy the condition that every point, sampled in the range $[\boldsymbol{l}^{(g)}, \boldsymbol{u}^{(g)}]$ for the g^{th} pursuit step, should also lie into the range of the global bounds, i.e. $x_{i,j} \in [L_i U_i]$, any point outside this range is corrected, Fig. 2a and b demonstrates schematically RST implementing a correction procedure where $\Delta len_i = u_i^{(g)} - l_i^{(g)}$.

2.2.2. Initialization

The random selection of the initial population is a common practice of the application of the metaheuristics [33]. The determination of the initial population is often disregarded in the design of a population-based metaheuristic algorithm. However, this part of the algorithm plays a crucial role in the effectiveness of the algorithm and its efficiency. Hence, one should pay more attention to this step [34]. In the generation of the initial population, the main criterion to deal with is

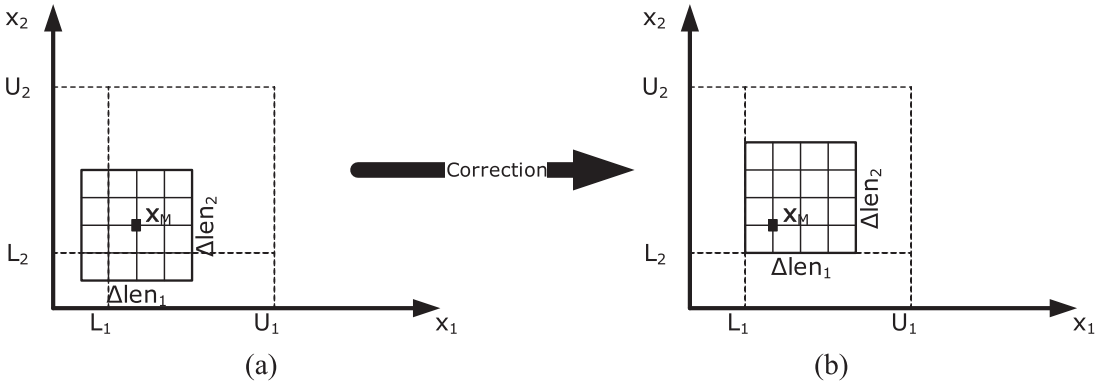


Fig. 2. Random sampling technique (a) before and (b) after correction.

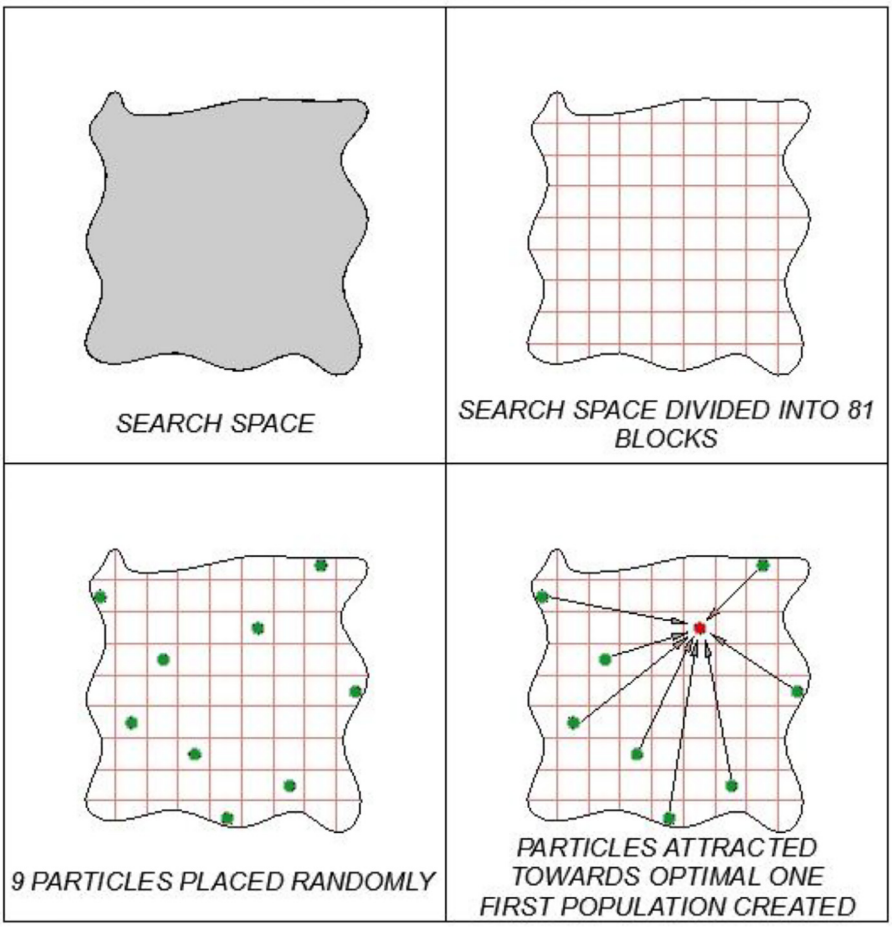


Fig. 3. Definition of the first population of pioneer particles.

diversification. If the initial population is not well diversified, a premature convergence can occur for many population-based metaheuristics. For this reason in this implementation the initial population is generated by means of RST.

In the first step of the algorithm (i.e. generation $g = 0$), a number of particles are randomly placed inside a hypervolume equal to the global search space, looking for optimal position vectors $\boldsymbol{x}_j^{(0)} \in R^D$. These particles and are named *pioneer particles*. The population (N_{pop}) of the pioneer particles is a parameter of the algorithm. Every pioneer particle of the first population $\boldsymbol{x}^{(0)} = \left[\boldsymbol{x}_1^{(0)}, \boldsymbol{x}_2^{(0)}, ..., \boldsymbol{x}_{N_{pop}}^{(0)}\right]^T$ is randomly positioned into the global search area:

$$\boldsymbol{x}_j^{(0)} = RST(\boldsymbol{L}, \boldsymbol{U}, D, N_{pop})$$

where

$$\boldsymbol{L} = [L_1, L_2, ..., L_D],\ \boldsymbol{U} = [U_1, U_2, ..., U_D] \text{ and } j = 1, 2, ..., N_{pop} \quad (6)$$

When all particles have identified a hypervolume position (solution vector), these N_{pop} solutions are compared between each other and the best particle attracts the rest ones and the first population is created in that hypervolume position. Fig. 3 depicts the initialization step for the case where N_{pop} is equal to 9 and D is equal to 2.

Six populations are created (composed of N_{pop} pioneer particles each), in the selected hypervolume position. Once all new populations are created, a new hypervolume selection pattern is decided for each

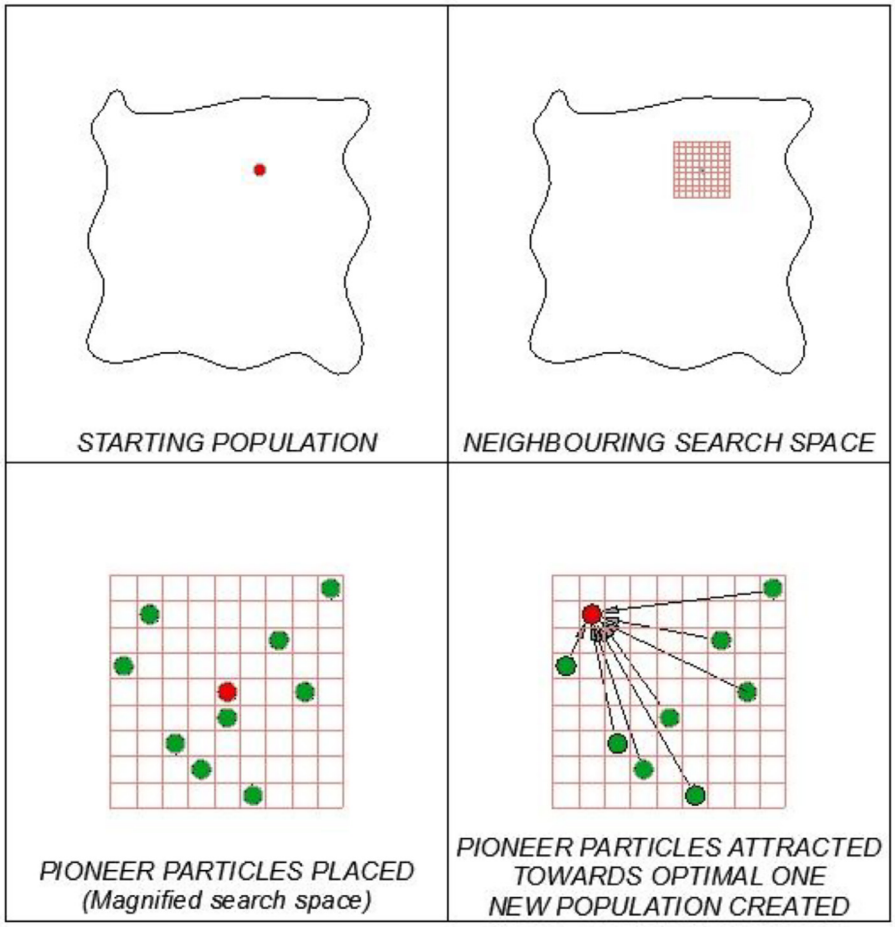


Fig. 4. Finding the new population of pioneer particles using the neighboring search hypervolume.

new population.

2.2.3. New hypervolume selection pattern

All newly-emerged particles will search inside the search space looking for a better solution in order to create their own population. Based on the behavior of the beetle described previously, five types of new hypervolume selection pattern are implemented into the proposed algorithm: (i) neighboring search volume, (ii) mid-scale search volume, (iii) large-scale search volume, (iv) global-scale search volume and (v) memory consideration search volume where the best positions found so far are used. The analytical description of their algorithmic implementation is presented below.

According to the new hypervolume selection pattern chosen, a search area is created around the generation position of the particles. For each pattern the definition of this area is implemented using a properly selected pattern factor (f_{pat}) and represents a parameter of PBA. According to every pattern, N_{pop} new pioneer particles are randomly positioned into this search area by means of RST:

$$\boldsymbol{x}_j^{(g)} = RST(\boldsymbol{l}^{(g)}, \boldsymbol{u}^{(g)}, D, N_{pop})$$

where

$$[l_i^{(g)}, u_i^{(g)}] \in [x_{birth,i}^{(g)} \cdot (1 - f_{pat}), x_{birth,i}^{(g)} \cdot (1 + f_{pat})] \quad (7)$$

where i denotes the element component of the jth individual solution vector $\boldsymbol{x}_j$, g denotes the generation (pursuit step), $x_{birth,i}^{(g)}$ is the i^{th} element of the birth position/solution vector, $u_i^{(g)}$ and $l_i^{(g)}$ are the upper and lower bounds of the i^{th} dimension for the g^{th} pursuit step, respectively.

Neighboring search hypervolume: Some of the progeny created in a new beetle brood will inevitably search for suitable position for a new brood in proximity with their starting brood. This happens either because some not robust beetles cannot fly large distances or because suitable positions (weekend trees) are present at close range to their birth position or because an overpopulation of beetles in close range can lead to an attack towards even robust trees placed inside this range. Thus, according to the *neighboring search hypervolume* a search area is created around the starting position of the particles. It should be noted that for each starting position the *neighboring search hypervolume search* is used first. The neighboring factor (f_{nb}) that is used to define the size of this area is a parameter of the PBA, the value range of f_{nb} is equal to [0.01, 0.20]. According to this pattern, the new N_{pop} pioneer particles are randomly positioned inside this search space based on the expression of Eq. (7) where f_{pat} is equal to f_{nb}. These newly defined N_{pop} solutions are compared between each other for defining the best pioneer particle. The best one is then compared to the starting position, and if better attracts the rest ones and the new population is created as follows:

$$\boldsymbol{x}_{birth}^{(g+1)} = \begin{cases} \boldsymbol{x}_{birth}^{(g)}, \text{ if } F(\boldsymbol{x}_{birth}^{(g)}) < F(\boldsymbol{x}_{j,k}^{(g)}), \forall\, j = 1, 2, \ldots, N_{pop},\ k = 1, 2, \ldots, N_{broods} \\ \boldsymbol{x}_{j,k}^{(g)}, \text{ otherwise} \end{cases} \quad (8)$$

where g denotes the generation (pursuit step), $\boldsymbol{x}_{birth}^{(g)}$ is the birth position vector and $\boldsymbol{x}_{j,k}^{(g)}$ is the new position vector found in k population (k = 1,2, …, N_{broods}), where N_{broods} is the maximum number of broods that is used a termination criterion. An example for the implementation of the *neighboring search hypervolume* can be seen in Fig. 4, for the case where N_{pop} is equal to 9 and D is equal to 2.

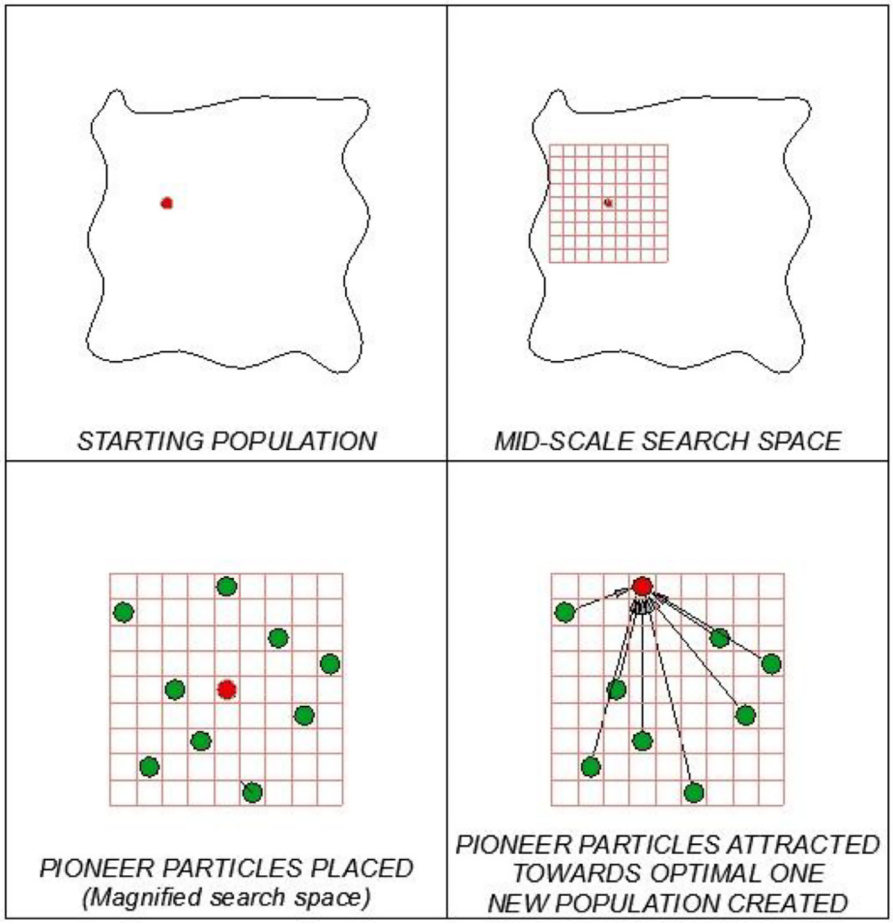


Fig. 5. Finding the new population position of pioneer particles using the mid-scale search hypervolume.

Mid-scale search hypervolume: The second *mid-scale search hypervolume* is employed when the *neighboring search hypervolume* has failed to improve the solution compared to the starting position. Similar to the *neighboring search hypervolume* a search area is created around the current best starting position of the particles. The mid-scale factor (f_{ms}) that is used to define the size of this area is a parameter of the PBA, the value range of f_{ms} is equal to [0.10,1.00]. According to this pattern, the new N_{pop} pioneer particles are randomly positioned inside this search space based on the expression of Eq. (7) where f_{pat} is equal to f_{ms}. Similar to the *neighboring search hypervolume* these newly defined N_{pop} solutions are compared between each other for defining the best pioneer particle. The best one is then compared to the starting position, and if better attracts the rest ones and the new population is created. An example of this search hypervolume can be seen in Fig. 5, for the case where N_{pop} is equal to 9 and D is equal to 2.

Large-scale search hypervolume: Some of the new beetles created in a new brood, usually the most robust beetles, will fly large distances from their birth position in search for a suitable tree – host for a new brood. The third *large-scale search* hypervolume is employed also when the *neighboring search hypervolume* has failed to improve the solution compared to the starting position. Similar to the *neighboring search hypervolume*a search area is created around the current best starting position of the particles. The large-scale factor (f_{ls}) that is used to define the size of this area is a parameter of the PBA, the value range of f_{ls} is equal to [1,100]. According to this pattern, the new N_{pop} pioneer particles are randomly positioned inside this search space based on the expression of Eq. (7) where f_{pat} is equal to f_{ls}. Similar to the previous search hypervolume patterns these newly defined N_{pop} solutions are compared between each other for defining the best pioneer particle. An example of this, can be seen in Fig. 6, for the case where N_{pop} is equal to 9 and D is equal to 2. Based on this pattern, there is a possibility that the bounds defined in Eq. (7) coincide with the global ones (i.e. $\boldsymbol{l}^{(g)} \equiv \boldsymbol{L}$ and $\boldsymbol{u}^{(g)} \equiv \boldsymbol{U}$).

Global-scale search hypervolume: is also implemented if for large number of function evaluations a better solution was not able to be found. The maximum number of unsuccessful function evaluations (FE_{un}) that impose the use of the *global-scale search hypervolume* pattern is defined with reference to the total function evaluations (FE_{total}) using a multiplication factor (f_{FE}) that is a parameter of the algorithm. An example of this, can be seen in Fig. 7, for the case where N_{pop} is equal to 9 and D is equal to 2. The value range of f_{FE} is equal to [0.05,0.25]. According to this pattern, the new N_{pop} pioneer particles are randomly positioned inside this search space with the use of RST:

$$\boldsymbol{x}_j^{(g)} = RST(\boldsymbol{L}, \boldsymbol{U}, D, N_{pop}) \tag{9}$$

Memory consideration search hypervolume: As described in the former section of the biological behavior of the beetle, if a specific area of the search space has many suitable hosts, a large number of beetles will be attracted to that area by pheromones. This leads to an increased concentration of beetles in this area and when plenty of beetles are gathered, they can attack any tree in that area regardless its robustness. A spherical expansion of beetle colonies is achieved with this procedure. In this direction, the *memory consideration search hypervolume* pattern was formulated where the best solutions found by the algorithm so far are stored in memory (***MEM***) of size equal to N_{pop} and are subsequently used as described below. The memory ***MEM*** is initialized with the members of the initial population ($g = 0$) generated according to Eq. (6):

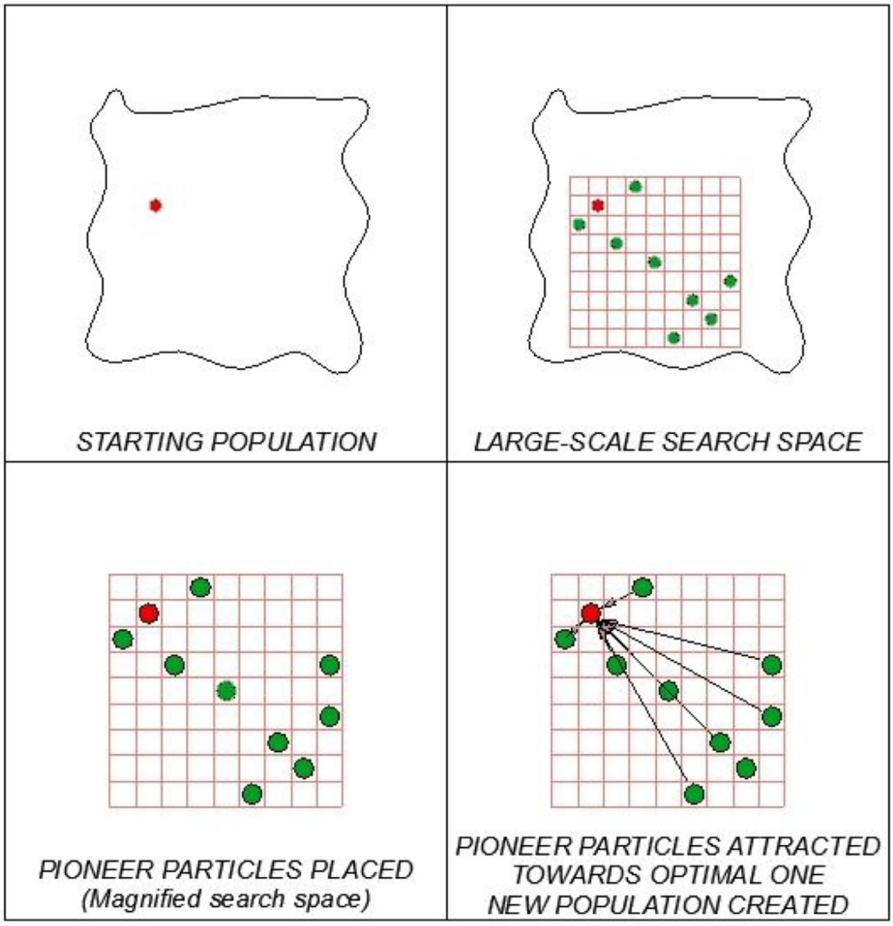


Fig. 6. Finding the new population position of pioneer particles using the large-scale search hypervolume.

$$\boldsymbol{MEM} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N_{pop}} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,N_{pop}} \\ \cdots & \cdots & \cdots & \cdots \\ x_{D,1} & x_{D,2} & \cdots & x_{D,N_{pop}} \end{bmatrix} \tag{10}$$

Contrary to the previously described search hypervolume patterns a search space is created around every member of ***MEM*** according to the following procedure: (i) select a member of ***MEM***, (ii) for every $i = 1,2, \ldots, D$, changes in the space $[L_i, U_i]$ are implemented when the rest ones are kept fixed, and (iii) if the new position is better than the worst member of ***MEM***, then it is replaced. In the case that a better position is identified, according to *memory consideration search* hypervolume pattern, a very narrow local search is performed according to a fine tuning factor (f_{tn}) used to define the size of this area and is a parameter of the PBA. The value range of f_{tn} is equal to [0.005,0.05].

2.2.4. Update location of populations

In this step the location of the broods (location of mating males and females) are updated and the past ones are dropped, except those stored in ***MEM***. In particular, all previous broods are extinct and the new locations become birth-places for the new generations. Based on the rules described previously, hypervolume patterns are selected for the new populations. This procedure continues until the termination criterion of PBA is satisfied. Collectively, the hypervolume patterns are selected in accordance to the following expression:

$$\boldsymbol{x}_{j,k}^{(g)} = \begin{cases} RST\left([x_{birth,i}^{(g)} \cdot (1 - f_{nb}),\ x_{birth,i}^{(g)} \cdot (1 + f_{nb})],\ D,\ N_{pop}\right), \\ \quad if\ \ k = 1 \\ else \\ \quad \begin{cases} RST\left(\boldsymbol{L},\ \boldsymbol{U},\ D,\ N_{pop}\right), \quad if\ \ FE > FE_{un} \\ else \\ \quad \begin{cases} RST\left([x_{birth,i}^{(g)} \cdot (1 - f_{ms}),\ x_{birth,i}^{(g)} \cdot (1 + f_{ms})],\ D,\ N_{pop}\right) \\ \qquad , \ \exists\, jf\left(\boldsymbol{x}_{j,k-1}^{(g)}\right) < f\left(\boldsymbol{x}_{birth}^{(g)}\right) \\ else \\ \quad \begin{cases} RST\left([x_{birth,i}^{(g)} \cdot (1 - f_{ls}),\ x_{birth,i}^{(g)} \cdot (1 + f_{ls})],\ D,\ N_{pop}\right), \text{ if } r < pr \\ \boldsymbol{MEM},\ otherwise \end{cases} \end{cases} \end{cases} \\ i = 1,\ 2, \ldots, D,\ j = 1,\ 2, \ldots, N_{pop} \end{cases} \tag{11}$$

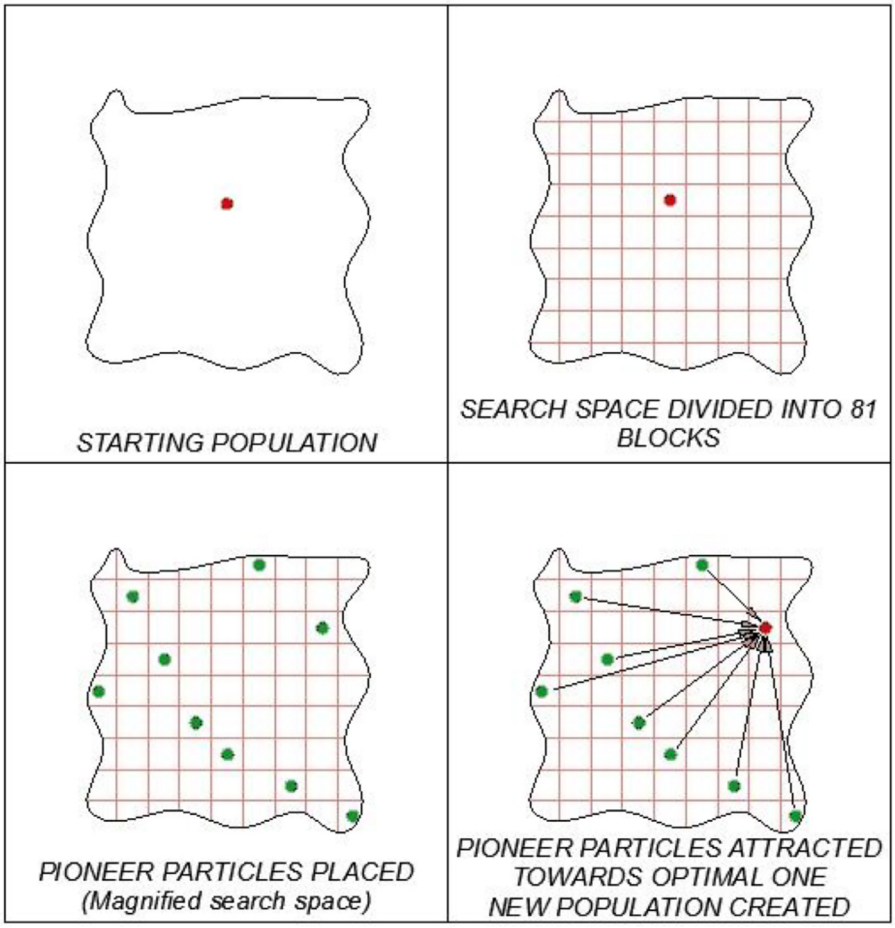


Fig. 7. Finding the new population position of pioneer particles using the global-scale search hypervolume.

and in Fig. 8, where r is a randomly generated number, $r \in (0, 1)$. In particular, Fig. 8 depicts the pattern decision procedure of the members of the kth population. According to line #2, the first population will be generated with reference to its starting position vector based on the neighboring search hypervolume pattern. Otherwise, the rest of the populations (2–N_{broods}) are generated as described in lines #4–#19. In particular, if the number of function evaluations without improvement exceeds the threshold of FE_{un} the population will be generated with reference to its starting position vector based on the global-scale search hypervolume pattern, as seen in line #6. While, for the case that the unsuccessful function evaluations have not reached this threshold, and the objective function value of the previous population is better than the starting position vector, then this vector is updated to its new value and the population will follow the mid-scale search hypervolume pattern (as denoted in line #10). If the objective function value of the previous population is not better, randomly, the population will be generated either based on a large-scale search hypervolume pattern (as it denoted in line #13) or it will take advantage of population positions memorized in MEM as can be seen in line #15.

Summarizing the flight patterns described previously, it can be stated that PBA preserves the balance between diversification and intensification not according to the population of executed function evaluations but according to the quality of the previously calculated solution in the neighborhood. Local searches are executed only around "good" solutions located by global explorations. At the same time, local searches failing to provide better solutions trigger large or global searches; this is described schematically in Fig. 8.

3. Sensitivity analysis – definition of the algorithmic parameters

The performance of metaheuristics is influenced by the values of their control parameters; however, their selection is not a straight forward procedure. In order to assess the efficiency and robustness of PBA, sensitivity analysis was performed in six test functions. Although it is not possible to define specific values for the algorithmic parameters that will be the proper ones for all problems, the values given below were found to be the proper ones as a balance between robustness and computational efficiency for the numerical tests examined in this sensitivity analysis study. In particular, aiming to assess performance and identify the best combination of these parameters for the proposed metaheuristic algorithm, 25 combinations of the parameters were generated randomly in order to calculate statistical quantities of the objective function value (i.e. mean and standard deviation of the optimized objective function value). The samples of the algorithmic parameter values used in the sensitivity analysis study are provided in Table 2. In particular, independent optimization runs were performed for each combination of the algorithmic parameters and for each test function considered.

3.1. Methodology for defining the algorithmic parameters

The algorithmic parameters are usually set empirically, in the current study those that were identified for each algorithm are based on the ranges successfully used in this study. However, in order to identify the best combination of algorithmic parameters, the following procedure is proposed: (i) The initial population is generated randomly in the range of the design space using an efficient sampling technique; then each design

1. For $k := 1$ To N_{broods} Do Begin
2. If $k == 1$ Then
3. $\boldsymbol{x}_{j,k}^{(g)} = \mathrm{RST}\left(\left[x_{birth,i}^{(g)} \cdot (1 - f_{nb}), x_{birth,i}^{(g)} \cdot (1 + f_{nb})\right], D, N_{pop}\right)$
4. Else
5. If $FE > FE_{un}$ Then
6. $\boldsymbol{x}_{j,k}^{(g)} = \mathrm{RST}\left(\boldsymbol{L}, \boldsymbol{U}, D, N_{pop}\right)$
7. Else
8. If $f(\boldsymbol{x}_{j,k-1}^{(g)}) < f(\boldsymbol{x}_{birth}^{(g)})$ Then
9. $\boldsymbol{x}_{birth}^{(g)} := \boldsymbol{x}_{j,k-1}^{(g)}$
10. $\boldsymbol{x}_{j,k}^{(g)} = \mathrm{RST}\left(\left[x_{birth,i}^{(g)} \cdot (1 - f_{ms}), x_{birth,i}^{(g)} \cdot (1 + f_{ms})\right], D, N_{pop}\right)$
11. Else
12. If $r < pr$ Then
13. $\boldsymbol{x}_{j,k}^{(g)} = \mathrm{RST}\left(\left[x_{birth,i}^{(g)} \cdot (1 - f_{ls}), x_{birth,i}^{(g)} \cdot (1 + f_{ls})\right], D, N_{pop}\right)$
14. Else
15. $\boldsymbol{x}_{j,k}^{(g)} = \boldsymbol{MEM}$
16. End If
17. End If
18. End If
19. End If
20. End

Fig. 8. Update location of populations.

Table 2
Samples of the algorithmic parameter values used in the sensitivity analysis.

Set	N_{pop}	f_{nb}	f_{tn}	f_{ms}	f_{ls}	pr	f_{FE}
1	38	0.075705	0.029182	0.563642	57	0.338595	0.161134
2	24	0.123459	0.037083	0.624188	22	0.408397	0.21086
3	16	0.068691	0.007445	0.999846	34	0.692002	0.24552
4	47	0.045784	0.032992	0.935893	40	0.149911	0.091139
5	46	0.033457	0.045939	0.436588	5	0.905509	0.113425
6	26	0.017896	0.0306	0.81412	6	0.837562	0.214723
7	92	0.134056	0.037759	0.950333	43	0.965924	0.178206
8	52	0.083214	0.017674	0.899109	96	0.340249	0.176398
9	12	0.052221	0.026661	0.274668	18	0.710718	0.187977
10	61	0.195374	0.009146	0.531219	38	0.384507	0.23957
11	21	0.090543	0.020727	0.417393	66	0.008013	0.172095
12	29	0.162232	0.04883	0.493499	49	0.435726	0.079244
13	67	0.178636	0.046596	0.97612	87	0.055815	0.144725
14	65	0.063162	0.034171	0.858121	29	0.196453	0.169105
15	35	0.080927	0.042115	0.511393	84	0.581464	0.134154
16	36	0.031623	0.031709	0.829211	37	0.225093	0.11579
17	45	0.015055	0.049438	0.72741	3	0.502583	0.06186
18	96	0.113653	0.014282	0.862913	69	0.080354	0.202934
19	28	0.070926	0.006222	0.5524	58	0.530035	0.195361
20	32	0.136118	0.043162	0.171538	24	0.128949	0.071422
21	12	0.095019	0.01706	0.686054	7	0.271986	0.124869
22	58	0.087889	0.045057	0.910344	9	0.473968	0.218572
23	86	0.130502	0.010934	0.754645	33	0.620001	0.235528
24	62	0.039123	0.019051	0.78291	86	0.246476	0.056075
25	16	0.125928	0.024675	0.294616	59	0.876725	0.201997

is assessed with reference to the objective function value and in case of constrained optimization problems the performance constraints in order to define its feasibility (fulfil the constraints) or not. (ii) Based on the sample defined by the initial population a meta-model based response surface (based on polynomial representation, neural networks, kriging, support vector machines, etc.) is calibrated (this step is proposed for the case of black box objective and constraint functions evaluation that is computationally intensive to calculate). (iii) In the next step of the procedure, random combinations (for example 25) of the algorithmic parameters are generated by means of a random sampling technique, and for each combination 100 optimization runs are performed, requiring limited computational effort since response surface is implemented for the calculation of the objective function value and in case of constrained optimization problems the feasibility assessment. Using the optimum objective function values achieved for the 100 independent optimization runs, mean and standard deviation or coefficient of variation of the optimum objective function are calculated. The implementation of the 100 independent optimization runs, performed for each combination of the algorithmic parameters, represents a necessary step since non-deterministic optimization algorithms do not yield the same results when restarted with the same parameters [35]. Although, the resulting optimization runs for defining the algorithmic parameters are equal to 25 combinations × 100 runs = 2500 optimization runs, due to the use of response surface are of very low computational cost. The combination resulting to the lower mean value (in case of minimization problem) and coefficient of variation is used for performing the optimization run with the specific algorithm.

3.2. Sensitivity analysis

For the needs of the sensitivity analysis, six frequently used in literature for benchmarking of algorithms, test functions are employed. These test functions are:

1. *Sphere function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D} x_i^2 \tag{12}$$

Table 3
Test functions employed in the study.

Test Function		Domain	Optimum	Name
$f_1(x)$	$f(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^D$	0	*Sphere*
$f_2(x)$	$f(x) = \sum_{i=1}^{n} [-x_i \sin(\sqrt{\lvert x_i \rvert})]$	$[-500, 500]^D$	−418.9829D	*Schwefel*
$f_3(x)$	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$	0	*Griewank*
$f_4(x)$	$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$	$[-100, 100]^D$	0	*Rastrigin*
$f_5(x)$	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	$[-10, 10]^D$	0	*Rosenbrock*
$f_6(x)$	$f(x) = -20\exp\left(-0.20\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + \exp(1)$	$[-32.768, 32.768]^D$	0	*Ackley*
$f_7(x)$	$f(x) = \sum_{i=1}^{n} (\sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k (x_i + 0.5))]) - n\sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k * 0.5)]$ $a = 0.5,\ b = 3,\ k_{max} = 20$	$[-0.5, 0.5]^D$	0	*Weierstrass*
$f_8(x)$	$f(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$	0	*Quadric*
$f_9(x)$	$f(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	$[-100, 100]^D$	0	*Step*
$f_{10}(x)$	$f(x) = -20\exp\left(-0.20\sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2}\right) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi y_i)) + 20 + \exp(1)$, $y = M{\cdot}x$	$[-32.768, 32.768]^D$	0	*Rotated Ackley*
$f_{11}(x)$	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} y_i^2 - \prod_{i=1}^{n} \cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$, $y = M{\cdot}x$	$[-600, 600]^D$	0	*Rotated Griewank*
$f_{12}(x)$	$f(x) = 10n + \sum_{i=1}^{n} [y_i^2 - 10\cos(2\pi y_i)]$, $y = M{\cdot}x$	$[-5.12, 5.12]^D$	0	*Rotated Rastrigin*
$f_{13}(x)$	$f(x) = 10n + \sum_{i=1}^{n} [z_i^2 - 10\cos(2\pi z_i)] - 330$, $z = (x - o){\cdot}M$	$[-5.12, 5.12]^D$	−330	*Shifted – Rotated Rastrigin*

Table 4
Statistical analysis for 10-D problems.

Set	Sphere	Schwefel's	Griewank's	Rastrigin's	Rosenbrock's	Ackley's
1	1.05E−29	−4176.76	**0.00E + 00**	**0.00E + 00**	6.21E + 00	7.99E−15
2	2.23E−72	−4171.97	1.01E−02	**0.00E + 00**	6.12E + 00	4.44E−15
3	3.38E−151	**−4188.91**	**0.00E + 00**	**0.00E + 00**	8.96E + 00	**8.88E−16**
4	9.91E−24	−4178.68	1.06E−07	**0.00E + 00**	7.17E + 00	2.92E−13
5	8.11E−16	−4146.13	1.75E-01	**0.00E + 00**	6.66E + 00	6.67E−09
6	4.78E−70	−4168.50	**0.00E + 00**	**0.00E + 00**	8.96E + 00	4.44E−15
7	1.31E−13	−4149.97	6.53E−12	2.04E−13	8.93E + 00	1.76E−07
8	4.85E−24	−4184.25	1.34E−05	**0.00E + 00**	**8.22E-02**	3.45E−12
9	5.68E−86	−4176.98	1.11E−01	**0.00E + 00**	5.94E + 00	7.99E−15
10	5.08E−08	−4187.64	2.28E−01	6.87E-08	8.03E + 00	1.18E−04
11	7.89E−56	−4181.95	1.37E−01	**0.00E + 00**	5.85E + 00	4.44E−15
12	3.97E−39	−4169.06	1.04E−01	**0.00E + 00**	5.87E + 00	4.44E−15
13	1.61E−23	−4157.17	1.00E−12	**0.00E + 00**	6.82E + 00	1.67E−12
14	4.49E−20	−4164.68	2.85E-02	**0.00E + 00**	8.72E + 00	7.45E−11
15	4.36E−23	−4159.92	1.43E-01	**0.00E + 00**	8.73E + 00	3.61E−12
16	1.70E−32	−4177.25	**0.00E + 00**	**0.00E + 00**	7.13E + 00	4.44E−15
17	8.75E−22	−4165.77	9.63E−02	**0.00E + 00**	6.47E + 00	1.58E−11
18	4.74E−08	−4184.97	1.38E−02	6.90E−08	7.39E + 00	1.90E−04
19	6.23E−43	−4188.82	**0.00E + 00**	**0.00E + 00**	8.92E + 00	4.44E−15
20	3.14E−14	−4172.84	6.27E−02	6.22E−14	5.47E + 00	3.82E−06
21	**4.00E−188**	−4185.46	7.78E−02	**0.00E + 00**	5.75E + 00	4.44E−15
22	2.25E−17	−4149.44	2.90E−05	**0.00E + 00**	8.74E + 00	7.30E−10
23	1.47E−08	−4187.48	6.54E−03	1.04E−08	6.99E + 00	3.99E−05
24	8.75E−13	−4181.78	2.37E−01	5.33E−15	7.17E + 00	2.18E−08
25	3.38E−66	−4164.91	2.72E-01	**0.00E + 00**	5.13E + 00	2.93E−14
Average	4.52E−09	−4172.85	6.81E−02	5.92E−09	6.89E + 00	1.41E−05
Standard Deviation	1.35E−08	12.72	8.48E−02	1.87E−08	1.85E + 00	4.32E−05
Minimum	4.00E−188	−4188.91	0.00E + 00	0.00E + 00	8.22E−02	8.88E−16
Maximum	5.08E−08	−4146.13	2.72E−01	6.90E−08	8.96E + 00	1.90E−04

Table 5
Statistical analysis for 30-D problems.

Set	Sphere	Schwefel's	Griewank's	Rastrigin's	Rosenbrock's	Ackley's
1	1.19E−33	−12,478.46	**0.00E + 00**	**0.00E + 00**	2.59E + 01	7.99E−15
2	1.01E−83	−12,400.32	1.70E−02	**0.00E + 00**	2.88E + 01	4.44E−15
3	4.98E−145	−12,559.84	**0.00E + 00**	**0.00E + 00**	2.88E + 01	**8.88E−16**
4	2.88E−25	−12,460.17	4.02E−02	**0.00E + 00**	2.86E + 01	7.19E−14
5	4.36E−18	−12,301.16	**0.00E + 00**	**0.00E + 00**	2.60E + 01	6.56E−10
6	2.48E−77	−12,414.98	1.93E−02	**0.00E + 00**	2.59E + 01	4.44E−15
7	3.84E−13	−12,416.15	7.76E−13	1.51E−13	2.88E + 01	1.25E−07
8	1.47E−25	−12,533.16	**0.00E + 00**	**0.00E + 00**	2.71E + 01	7.19E−14
9	1.59E−98	−12,453.65	**0.00E + 00**	**0.00E + 00**	2.86E + 01	2.58E−14
10	1.08E−07	−12,558.81	6.70E−02	4.52E-08	2.80E + 01	4.87E−05
11	6.01E−65	−12,514.33	1.26E−02	**0.00E + 00**	2.58E + 01	7.99E−15
12	8.60E−45	−12,294.99	1.09E−02	**0.00E + 00**	**2.55E + 01**	7.99E−15
13	5.11E−24	−12,332.07	**0.00E + 00**	**0.00E + 00**	2.70E + 01	3.42E−13
14	5.30E−21	−12,425.66	**0.00E + 00**	**0.00E + 00**	2.88E + 01	1.43E−11
15	4.90E−26	−12,331.18	1.33E−02	**0.00E + 00**	2.89E + 01	5.77E−14
16	4.85E−36	−12,437.18	**0.00E + 00**	**0.00E + 00**	2.79E + 01	4.44E−15
17	9.55E−23	−12,355.17	**0.00E + 00**	**0.00E + 00**	2.61E + 01	1.24E−12
18	9.05E−07	−12,540.62	3.40E−06	5.08E−07	2.69E + 01	2.00E−04
19	9.31E−49	**−12,561.68**	2.85E−02	**0.00E + 00**	2.86E + 01	4.44E−15
20	1.04E−15	−12,353.39	1.01E−02	4.97E−14	2.56E + 01	3.43E−07
21	**3.47E−206**	−12,529.82	**0.00E + 00**	**0.00E + 00**	2.86E + 01	3.44E + 00
22	3.89E−18	−12,353.25	**0.00E + 00**	**0.00E + 00**	2.80E + 01	3.50E−10
23	5.69E−09	−12,555.03	1.61E−07	1.60E−08	2.71E + 01	2.30E−05
24	8.14E−16	−12,530.70	1.33E−15	**0.00E + 00**	2.70E + 01	5.63E−09
25	1.89E−75	−12,474.81	**0.00E + 00**	**0.00E + 00**	2.88E + 01	3.58E + 00
Average	4.07E−08	−12,446.66	8.75E−03	2.27E−08	2.75E + 01	2.81E−01
Standard Deviation	1.78E−07	86.79	1.58E−02	9.94E−08	1.22E + 00	9.53E-01
Minimum	3.47E−206	−12,561.68	0.00E + 00	0.00E + 00	2.55E + 01	8.88E-16
Maximum	9.05E−07	−12,294.99	6.70E−02	5.08E−07	2.89E + 01	3.58E + 00

Table 6
PBA applied to test-functions 7 to 13 for 30-D problems.

Function	Mean best	Minimum	Maximum	Deviation
Weierstrass	7.11E−16	0.00E + 00	1.42E−14	2.81E−15
Quadric	6.66E−23	1.36E−42	2.00E−21	3.58E−22
Step	3.77E−04	2.41E−04	4.95E−04	6.79E−05
Rotated Ackley's	4.44E−15	4.44E−15	4.44E−15	0.00E + 00
Rotated Griewank's	8.59E−16	0.00E + 00	2.55E−14	4.58E−15
Rotated Rastrigin's	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00
Shifted – Rotated Rastrigin's	−3.30E + 02	−3.30E + 02	−3.30E + 02	0.00E + 00

which is a uni-modal, separable function.

2. *Schwefel's function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D} [-x_i \sin(\sqrt{|x_i|})] \tag{13}$$

which is a multi-modal, rotated, non-separable, asymmetrical function having a huge number of local optima while the second better local optimum is located far from the global optimum.

3. *Griewank's function*

$$f(\boldsymbol{x}) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{14}$$

which is a multi-modal, non-separable function.

4. *Rastrigin's function*

$$f(\boldsymbol{x}) = 10D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)] \tag{15}$$

which is a multi-modal, separable, asymmetrical function having a huge number of local optima.

5. *Rosenbrock's function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \tag{16}$$

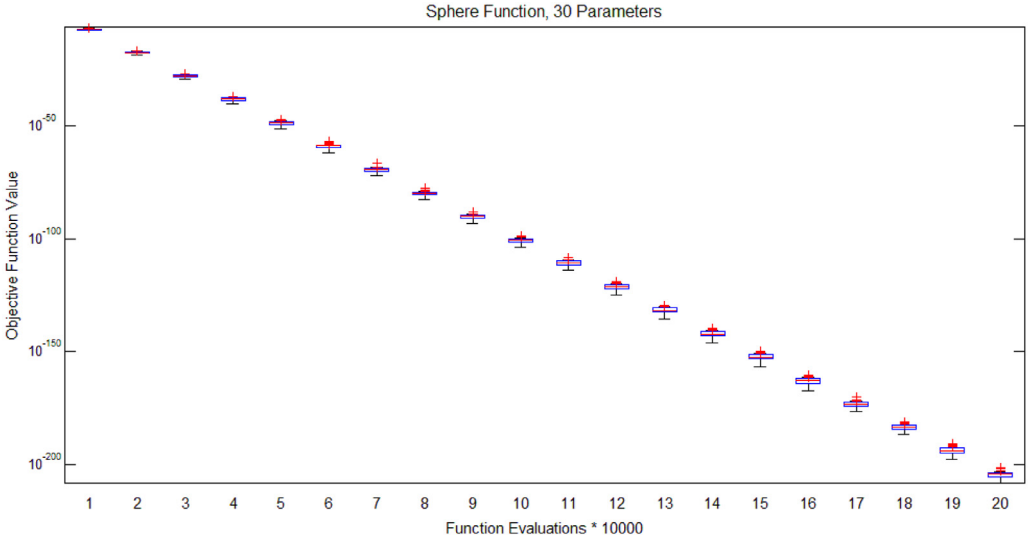
which is a multi-modal, non-separable function.

6. *Ackley's function*

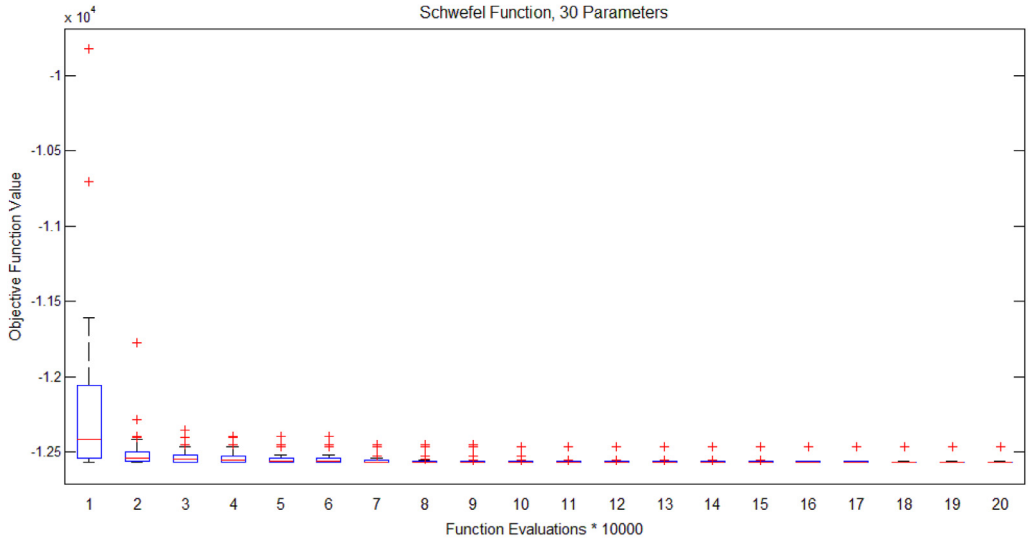
$$f(\boldsymbol{x}) = -20\exp\left(-0.20\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \exp(1) \tag{17}$$

which is a multi-modal, non-separable, asymmetrical function.

In Table 3, more information related to the test functions employed in the sensitivity analysis part (test functions 1–6) and the comparative study one (test functions 1–13) that is presented in the next section are provided with reference to the design space and the objective function value that corresponds to the global optimum. The dimension D of the test functions considered in the sensitivity analysis study is equal to 10 and 30 while the maximum function evaluations permitted for defining convergence were set to 100,000 and 200,000 for the two dimensions, respectively. The results obtained from the sensitivity analysis are



(a)



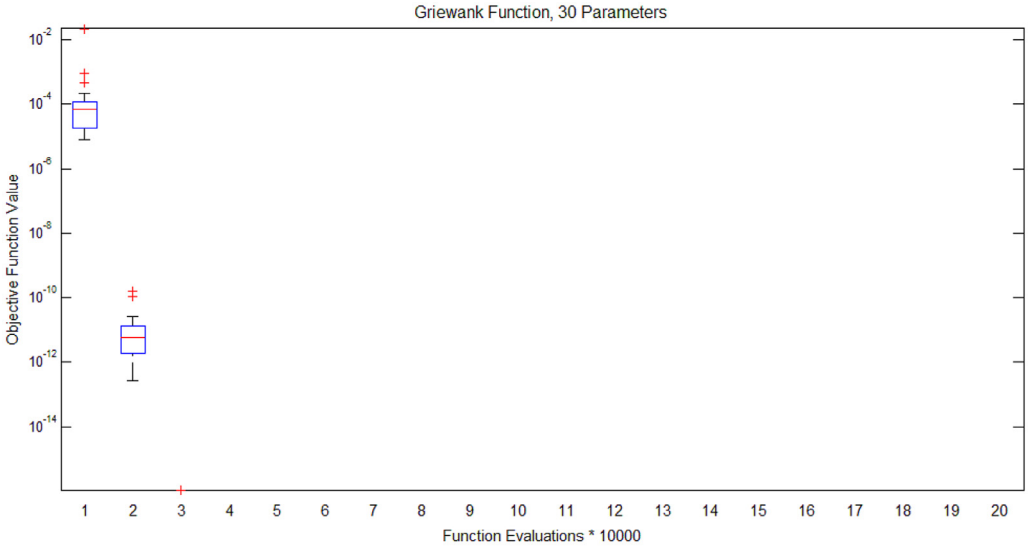
(b)

Fig. 9. Performance of the independent runs for different number of function evaluations for the test functions (a) sphere, (b) Schwefel, (c) Griewank, (d) Rastrigin, (e) Rosenbrock and (f) Ackley for 30 parameters.

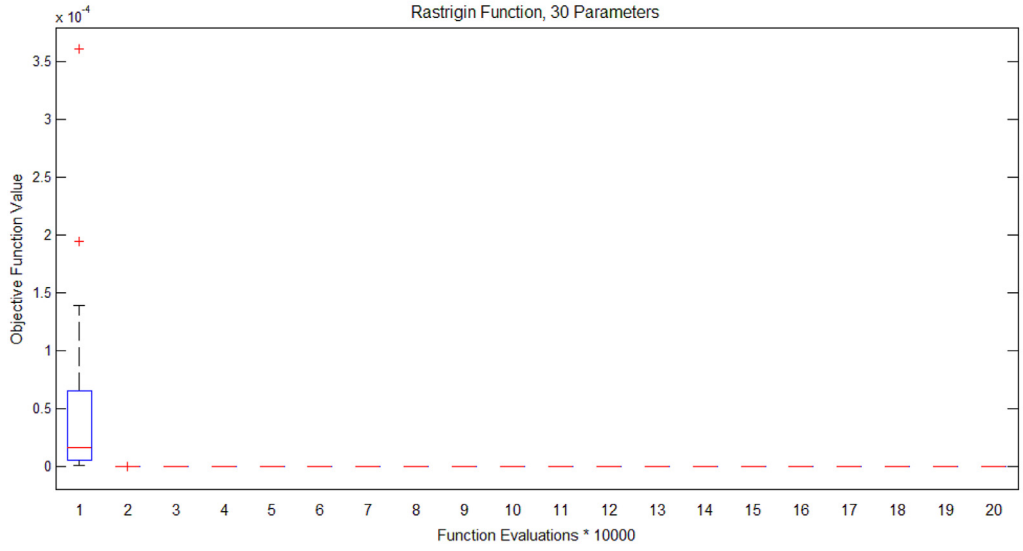
presented in Tables 4 and 5, for 10-D and 30-D problems, respectively. Specifically, the average value, standard deviation, minimum and maximum optimized objective function values achieved for the 25 independent runs (using the samples of algorithmic parameters of Table 2) for every test function examined can be found in these two tables. The results shown in the two tables denote that PBA is not influenced by the selection of the algorithmic parameters, especially for test functions 1–4 and 6. For all six test functions examined and the two different dimensions tested the best solution found coincide with the targeted optimum objective function value. The standard deviation for the five functions are very low, while all 25 parameters combinations examined converged to acceptable optimized values of the objective functions. Rosenbrock's function (f_5) described in Eq. (16) was the only one that PBA did not managed to achieve an acceptable solution for both dimensions examined. Through the sensitivity analysis performed, compromise parameter values that seem to help the algorithm in finding the optimal value for all six test functions examined were identified and are shown in Table 1 under “Suggested Value Range”.

4. Performance of PBA

In order to assess the performance of PBA, the algorithm is tested against well-established state-of-the-art metaheuristic search algorithms. In particular, the test functions described in the previous section; along with seven additional test functions which are also used in literature for benchmarking of search algorithms are considered for testing the efficiency of the proposed algorithm. The algorithms that were used for the current comparative study are those considered in the studies by Liang et al. [36], Chen et al. [37] and Chen et al. [15]. In particular, experiments were carried out to compare with (i) nine particle swam optimization algorithms taken from the study by Liang et al. [36], including the comprehensive learning particle swarm optimizer (CLPSO) algorithm proposed Liang et al. [36]: PSO with inertia weight (PSO-w) [38], PSO with constriction factor (PSO-cf) [39], local version of PSO with inertia weight (PSO-w-local) [40], local version of PSO with constriction factor (PSO-cflocal) [39], UPSO [17], fully informed particle swarm (FIPS) [18], FDR-PSO [41], CPSO-H [19], and CLPSO [36]; (ii) five algorithms taken from the study by Chen et al.



(c)



(d)

Fig. 9. (*continued*)

[37], including the IGSO proposed by Chen et al. [37]: SPSO, quantum behaved PSO (QPSO), weighted QPSO (WQPSO), group search optimizer (GSO) and improved GSO algorithm and (iii) nine particle swam optimization algorithms taken from the study by Chen et al. [15], including the aging leader and challengers (ALC-PSO) algorithm proposed Chen et al. [15]: including the global version PSO (GPSO) with a fixed inertia weight $\omega = 0.4$ [38], the GPSO that decreases the value of ω linearly from 0.9 to 0.4 [42], the local version PSO with the RPSO [40], the one with the VPSO [40], FIPS [18], hierarchical PSO with HPSO-TVAC [43], DMS-PSO [44], and CLPSO [36].

The dimensions that were considered in the sensitivity analyses performed in the previous section (i.e. 10-D and 30-D) were also used for comparing PBA with the other algorithms for the first six test function while 30-D problems are used for the case of the addition seven test functions. Worth mentioning that in all comparative tests performed, the initialization range is the same to the search space described in Table 3, since in many cases found in the literature a narrower initialization range is used (see for example in [15,36,37]). For all algorithms tested the initial population is generated randomly within the range of design space for each test example examined, while for the implementation of all seven algorithms the real valued representation of the design vectors is adopted. For comparison purposes, the termination criterion for this comparative study is the same for all metaheuristic optimization algorithms. In particular, the termination criterion used is the number of function evaluations that was set to 200,000 [45].

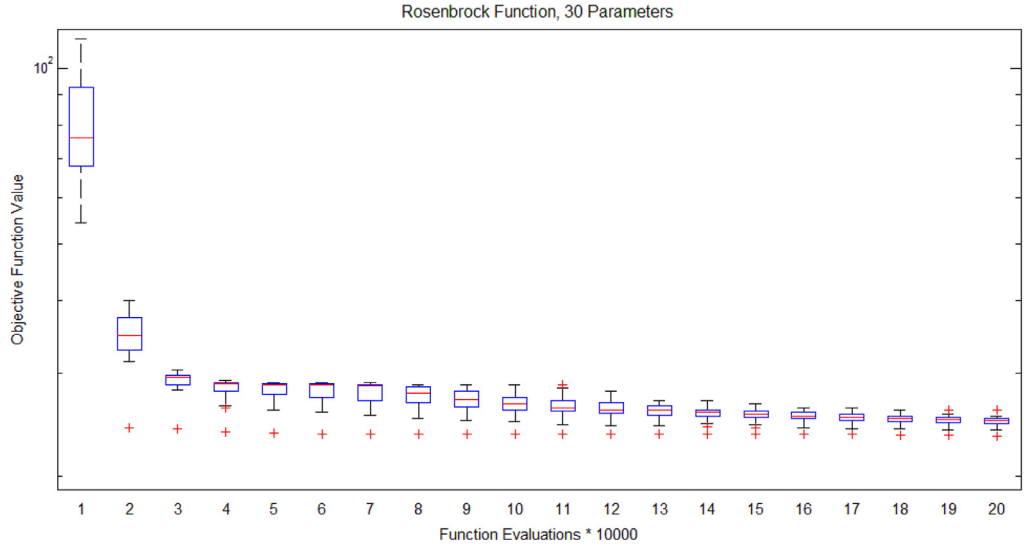
4.1. Application of PBA to seven test functions

For the needs of the comparative part of this study, seven frequently used in literature for benchmarking of algorithms test functions are employed in addition to the six ones described previously. These test functions are:

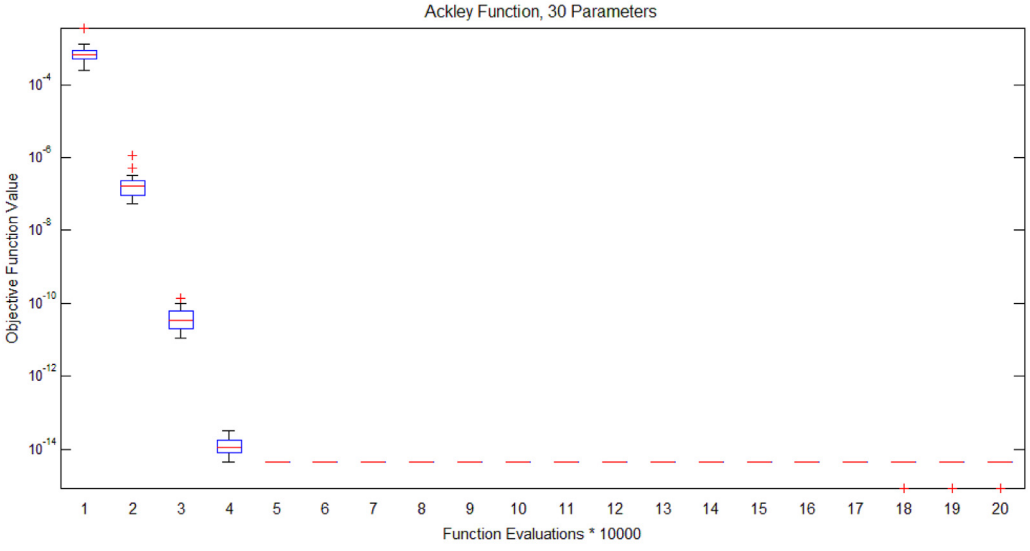
7. *Weierstrass function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k_{\max}}\left[a^k \cos(2\pi b^k (x_i + 0.5))\right]\right) - D\sum_{k=0}^{k_{\max}}\left[a^k \cos(2\pi b^k \cdot 0.5)\right]$$

$$a = 0.5,\ b = 3,\ k_{\max} = 20 \tag{18}$$



(e)



(f)

Fig. 9. (*continued*)

which is a multi-modal, non-separable, asymmetrical, continuous function but differentiable only on a set of points.

8. *Quadric function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{19}$$

which is a uni-modal function.

9. *Step function*

$$f(\boldsymbol{x}) = \sum_{i=1}^{D} (x_i + 0.5)^2 \tag{20}$$

which is a uni-modal, separable function.

10. *Rotated Ackley's function*

$$f(\boldsymbol{x}) = -20 \exp\left(-0.20 \sqrt{\frac{1}{D} \sum_{i=1}^{D} y_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi y_i) \right) + 20 + \exp(1),$$
$$\boldsymbol{y} = \boldsymbol{M} \cdot \boldsymbol{x} \tag{21}$$

which is a multi-modal, rotated, non-separable, asymmetrical function.

11. *Rotated Griewank's function*

$$f(\boldsymbol{x}) = \frac{1}{4000} \sum_{i=1}^{D} y_i^2 - \prod_{i=1}^{D} \cos\left(\frac{y_i}{\sqrt{i}} \right) + 1,$$
$$\boldsymbol{y} = \boldsymbol{M} \cdot \boldsymbol{x} \tag{22}$$

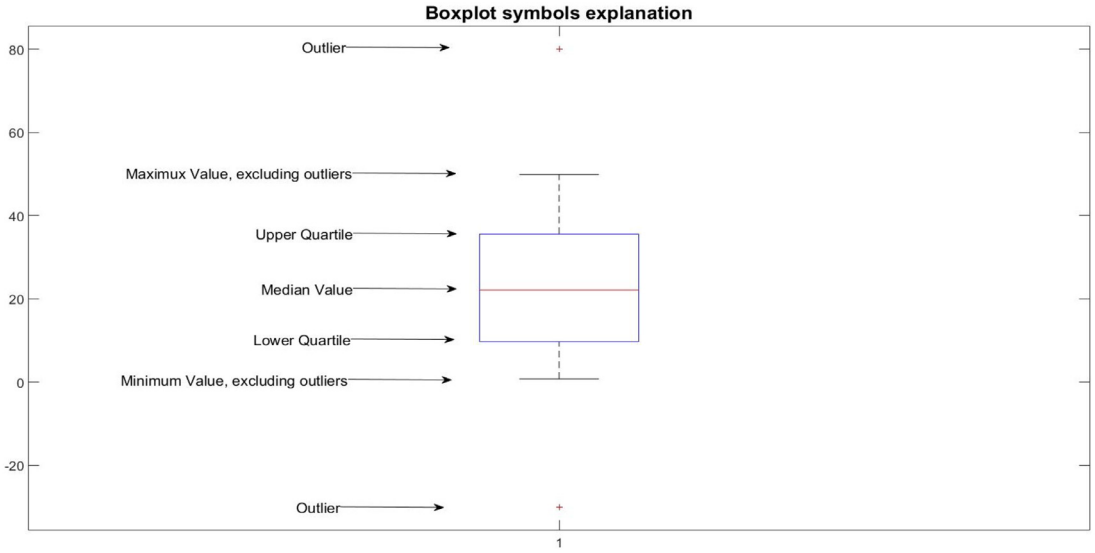


Fig. 10. Explanation of boxplots symbols.

which is a multi-modal, rotated, non-separable function.

12. *Rotated Rastrigin's function*

$$f(\boldsymbol{x}) = 10D + \sum_{i=1}^{D} [y_i^2 - 10\cos(2\pi y_i)],$$
$$\boldsymbol{y} = \boldsymbol{M}\cdot\boldsymbol{x} \tag{23}$$

which is a multi-modal, rotated, separable, asymmetrical function having a huge number of local optima.

13. *Shifted - Rotated Rastrigin's function*

$$f(\boldsymbol{x}) = 10D + \sum_{i=1}^{D} [z_i^2 - 10\cos(2\pi z_i)] - 330,$$
$$\boldsymbol{z} = \boldsymbol{M}\cdot(\boldsymbol{x} - \boldsymbol{o}) \tag{24}$$

which is a multi-modal, sifted-rotated, separable, asymmetrical function having a huge number of local optima.

Rotation means that a linear and orthogonal transformation matrix $\boldsymbol{M}$ is applied such that $F(\boldsymbol{M}, \boldsymbol{s})$ is calculated, where the orthogonal (rotation) matrix $\boldsymbol{M}$ was generated using standard normally distributed entries by Gram-Schmidt orthonormalization [46]. The transformation matrix $\boldsymbol{M}$ is a pure rotation that does not change the function's structure. The results obtained implementing PBA to the test functions 7 to 13 for 30-D problems are summarized in Table 6. Specifically, the average value, standard deviation, minimum and maximum optimized objective function values achieved for the 30 independent runs using the algorithmic parameters that presented the best performance for every test function examined. These results confirmed the findings of the sensitivity analysis part of this study i.e. that is robust and efficient. In particular, for all seven test functions the best solution found coincide with the targeted optimum objective function value. Furthermore, the worst optimization run converged to acceptable optimized values of the objective functions. In order to study the performance of the proposed algorithm with respect to the number of function evaluations, the optimization history for test functions 1–6 is depicted in Fig. 9 for 30-D problems. In Fig. 9 the variance of the optimized objective function value is also presented, where as it can be observed the algorithm shows a quick convergence trend. Explanations of the symbols used in Fig. 9 can be seen in Fig. 10.

4.2. Comparison of PBA with state-of-the-art metaheuristics

In this part of this experimental study both 10-D and 30-D problems are solved while the maximum function evaluations were set equal to 100,000 and 200,000 respectively. For every test function 30 independent runs are performed using the algorithmic parameters that presented the best performance in the parametric study performed previously and the mean values are calculated in order to be compared with those achieved by the state-of-the-art metaheuristics considered. The results are presented in Tables 7 and 8 for the 10-D and 30-D problems, respectively. As it can be seen that for the test functions used (i.e. f_1 to f_6) PBA performed equally well or even outperformed the other algorithms for both dimensions considered. In particular, in functions f_1 to f_4 and f_6 PBA achieved better mean best objective function values than those obtained by the other algorithms and better standard deviation. Worth mentioning also that with respect to its performance for the Rosenbrock's function (f_6) performed equally with the rest of the algorithms both in terms of mean best objective function value and standard deviation.

In addition to this comparative study, the performance of the proposed algorithm was assessed against the best algorithms that were identified during the CEC 2013 competition presented in the “Special Session & Competition on Real-Parameter Single Objective Optimization” [47]. In this comparative part of this study, test functions 1, 2, 4, 10, 11 and 12 with a small modification on their expression (compared to that given in Table 3) a real number (F_k) was added in this expression, in particular $F_1 = -1400$ was added to test function 1, and $F_2 = -100$, $F_4 = -400$, $F_{10} = -700$, $F_{11} = -500$ and $F_{12} = -300$ was added to the rest of the test functions, respectively. The results of this parametric study are provided in Table 9, where PBA is compared against to iCMAES-ILS and NBIPOP-ACMA-ES with reference to the mean error values. iCMAES-ILS and NBIPOP-ACMA-ES are the top two performing algorithms at the CEC 2013 competition [48]. The average error values correspond to the errors measured at the maximum number of function evaluations. Worth mentioning that for some of the test functions the design space used in CEC 2013 competition was wider while the function evaluations permitted for the 30-D problems employed in this comparative study were 300,000. In particular, in all these test functions (1, 2, 4, 10, 11 and 12) PBA achieved better

Table 7
Comparative study of PBA with state-of-the-art metaheuristics for test-functions 1 to 6 and 10-D problems.

Algorithm	Sphere function		Schwefel's function		Griewank's function		Rastrigin's function		Rosenbrock's function		Ackley's function	
	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.
PSO-w [37]	7.96E−51	NA	−3.87E + 03	NA	9.69E−02	NA	5.82E + 00	NA	3.08E + 00	NA	1.58E−14	NA
PSO-cf [38]	9.84E−105	NA	−3.20E + 03	NA	1.19E−01	NA	1.25E + 01	NA	**6.98E−01**	NA	9.18E−01	NA
PSO-w-local [39]	2.13E−35	NA	−3.86E + 03	NA	7.80E−02	NA	3.88E + 00	NA	3.92E + 00	NA	6.04E−15	NA
PSO-cflocal [39]	1.37E−79	NA	−3.31E + 03	NA	2.80E−02	NA	9.05E + 00	NA	8.60E−01	NA	5.78E−02	NA
UPSO [17]	9.84E-118	NA	−3.11E + 03	NA	1.04E−01	NA	1.17E + 01	NA	1.40E + 00	NA	1.33E + 00	NA
FDR-PSO [40]	2.21E−90	NA	−3.34E + 03	NA	9.24E−02	NA	7.51E + 00	NA	8.67E + 00	NA	3.18E−14	NA
FIPS [18]	3.14E−30	NA	−4.12E + 03	NA	1.31E−01	NA	2.12E + 00	NA	2.78E + 00	NA	3.75E−15	NA
CPSO−H [19]	4.98E−40	NA	−3.98E + 03	NA	4.07E−02	NA	**0.00E + 00**	NA	1.53E + 00	NA	1.49E−14	NA
CLPSO [35]	5.15E−29	NA	**−4.19E + 03**	NA	4.56E−03	NA	**0.00E + 00**	NA	2.46E + 00	NA	4.32E−14	NA
SPSO [36]	3.27E−18	5.75E−18	−3.39E + 03	9.06E + 00	8.70E−02	5.70E−02	2.45E + 00	1.63E + 00	3.56E + 01	5.69E + 01	NA	NA
QPSO [36]	7.40E−104	7.39E−103	−3.48E + 03	6.04E + 00	3.58E−04	2.80E−02	2.31E + 00	2.00E−02	7.43E + 00	3.20E−01	NA	NA
WQPSO [36]	8.37E−106	1.08E−106	−3.77E + 03	4.33E + 00	1.63E−04	1.63E−04	1.84E + 00	1.00E−02	1.04E + 01	**2.50E−01**	NA	NA
GSO [36]	6.75E−18	1.88E−18	−4.03E + 03	7.10E + 00	1.60E−01	4.40E−02	2.57E + 00	1.82E + 00	4.38E + 00	1.97E + 00	NA	NA
IGSO [36]	7.33E−41	1.89E−41	**−4.19E + 03**	3.10E + 00	4.38E−05	1.40E−02	6.70E−01	4.20E−01	7.60E−01	5.60E−01	NA	NA
PBA	**5.37E−187**	**0.00E + 00**	**−4.19E + 03**	**1.02E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	5.81E + 00	1.00E + 00	**8.88E−16**	**0.00E + 00**

performance than those obtained by the two algorithms.

4.3. Performance of PBA based on CEC 2014 benchmarks and complexity evaluation

In this part of the experimental study various 10-D, 30-D, 50-D and 100-D problems are solved according to the CEC 2014 test-suite [23] that is composed of 30 test functions. The functions included are unimodal (test functions No 1–3), simple multimodal (test functions No 4 to 16), hybrid functions (test functions No 17 to 22) and composite functions (test functions No 23–30). In all test functions the search space is $[-100,100]^D$ while the maximum function evaluations were set equal to 10,000 × D. According to the guidelines of the CEC 2014 competition, 51 independent runs were performed for each test function and the average error values were obtained. The results achieved for the CEC 2014 test-suite are provided in Tables 10 and 11 for the 10-D and 30-D, 50-D and 100-D problems, respectively. As seen in results, PBA deals successfully with most of the test functions while not performing very well with few of them. Out of the two Tables, it can be seen that PBA appears not to be significantly affected by the dimension of the problem. In particular, with the increase of the dimensionality it is expected that the performance is decreased, however, this drop is relatively small in the case of PBA as it can be seen in Tables 10 and 11. As differences may be noticed between some test-functions used previously and in CEC 2014 test, it must be pointed out that they are the result of different search space and different rotation methods used in CEC 2014.

In addition, the Wilcoxon ranksum test [49] was also employed in order to evaluate the performance of PBA against well-established algorithms that were also tested according to the standards of CEC 2014. In particular, simultaneous optimistic optimization (SOO) [50], the fireworks algorithm with differential mutation (FWA-DM) [51], the adaptive differential evolution (ADE), an improved variant of ADE with the use of partial opposition-based learning (POBL-ADE) [52] and the L-SHADE [53] algorithms where considered for this comparative study. The results of the Wilcoxon ranksum test are given in Table 12; *Better* denotes the number of functions (out of the 30 ones) where a specific algorithm performs better compared to PBA, *Worst* denotes the number of functions where it performs worst and *Equal* denotes the number of functions where it has the same performance with PBA. It should be underlined that PBA appears to be less affected by the dimension of the problem compared to most of the other algorithms.

The complexity of PBA is also calculated according to the CEC 2014 benchmark suite guide [23], and is presented in Table 13 for dimensionality of 10-D, 30-D, 50-D and 100-D. As suggested by the complexity test, T_0 is the time required to perform the following problem:

$$\begin{aligned} &for\ i = 1{:}\ 10^6 \\ &x = 0.55 + (double)i;\ x = x + x;\ x = x/2; \\ &x = x{*}x;\ x = sqrt(x);\ x = \log(x); \\ &x = \exp(x);\ x = x/(x + 2); \\ &end \end{aligned} \tag{25}$$

T_1 is the time to execute 200,000 evaluations of test-function No 18 and T_2 is the time that the algorithm needs to execute 200,000 optimization runs of test-function No 18. T_1 and T_2 are scaled linearly with dimensions as by the linear growth of $(T_2 - T_1)/T_0$. All test runs were executed on a computer with Windows 7 OS, Intel i7 3610QM CPU, 12 GB RAM using Matlab programming language. The complexity of the PBA is presented in Table 13, as seen the computational effort of PBA is rather small as it has the ability to execute 200,000 function evaluation of a 100-D problem in less than 10 s.

Table 8
Comparative study of PBA with state-of-the-art metaheuristics for test-functions 1 to 6 and 30-D problems.

Algorithm	Sphere function		Schwefel's function		Griewank's function		Rastrigin's function		Rosenbrock's function		Ackley's function	
	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.	Mean best	St. var.
ALC-PSO [15]	1.68E−161	8.21E−161	−1.25E + 04	5.41E + 01	1.22E−02	1.58E−02	2.53E−14	1.38E−14	7.61E + 00	6.66E + 00	1.15E−14	2.94E−15
GPSO [37]	2.65E−161	2.38E−161	−8.86E + 03	5.20E + 02	1.51E−02	1.75E−02	5.76E + 01	1.46E + 01	1.17E + 01	1.50E + 01	1.60E + 00	1.03E + 00
GPSO-2 [26]	7.08E−53	1.71E−52	−1.12E + 04	3.33E + 02	1.65E−02	1.69E−02	2.52E + 01	5.21E + 00	2.55E + 01	2.59E + 01	1.10E−14	2.27E−15
VPSO [39]	1.90E−38	3.99E−38	−9.88E + 03	5.20E + 02	2.41E−02	2.25E−02	2.92E + 01	9.66E + 00	2.95E + 01	2.47E + 01	1.52E−14	4.10E−15
RPSO [39]	5.58E−29	1.42E−28	−9.68E + 03	3.44E + 02	8.17E−03	1.78E−02	3.88E + 01	8.63E + 00	2.06E + 01	1.25E + 01	2.66E−14	5.45E−14
CLPSO [35]	1.39E−27	2.05E−27	**−1.26E + 04**	3.61E + 01	2.01E−14	8.67E−14	2.44E−14	5.98E−14	1.70E + 01	1.28E + 01	2.49E−14	4.18E−15
HPSO-TVAC [42]	1.45E−41	4.64E−41	−1.10E + 04	2.61E + 02	1.39E−02	1.39E−02	1.43E + 00	2.96E + 00	1.20E + 01	1.61E + 01	8.95E−11	4.46E−10
FIPS [18]	2.60E−30	3.24E−30	−1.05E + 04	3.87E + 02	2.47E−04	2.47E−04	2.87E + 01	1.46E + 01	2.25E + 01	4.39E−01	7.58E−15	6.49E−16
DMS-PSO [44]	1.95E−54	8.43E−54	−9.63E + 03	4.78E + 02	1.07E−02	1.60E−02	2.78E + 01	7.57E + 00	1.95E + 01	1.20E + 01	9.23E−15	1.79E−15
PSO-w [37]	9.78E−30	NA	−1.15E + 04	NA	8.13E−03	NA	2.90E + 01	NA	2.93E + 01	NA	3.94E−14	NA
PSO-cf [38]	5.88E−100	NA	−8.79E + 03	NA	2.06E−02	NA	5.62E + 01	NA	1.11E + 01	NA	1.12E + 00	NA
PSO-w-local [39]	5.35E−100	NA	−1.10E + 04	NA	5.91E−03	NA	2.72E + 01	NA	2.39E + 01	NA	9.10E−08	NA
PSO-cflocal [39]	7.70E−54	NA	−8.79E + 03	NA	5.91E−03	NA	4.53E + 01	NA	1.71E + 01	NA	5.33E−15	NA
UPSO [17]	4.17E−87	NA	−7.73E + 03	NA	1.66E−03	NA	6.59E + 01	NA	1.51E + 01	NA	**1.22E−15**	NA
FDR-PSO [40]	4.88E−102	NA	−8.96E + 03	NA	1.01E−02	NA	2.84E + 01	NA	**5.39E + 00**	NA	2.84E−14	NA
FIPS [18]	2.69E−12	NA	−1.05E + 04	NA	1.16E−06	NA	7.30E + 01	NA	2.45E + 01	NA	4.81E−07	NA
CPSO−H [19]	1.16E−113	NA	−1.15E + 04	NA	3.63E−02	NA	**0.00E + 00**	NA	7.08E + 00	NA	4.93E−14	NA
CLPSO [35]	4.46E−14	NA	**−1.26E + 04**	NA	3.14E−10	NA	4.85E−10	NA	2.10E + 01	NA	0.00E + 00	NA
SPSO [36]	3.26E−12	5.33E−12	−9.79E + 03	4.20E + 01	1.60E−02	9.00E−03	2.92E + 01	1.11E + 01	2.13E + 02	2.80E + 02	NA	NA
QPSO [36]	1.68E−51	6.70E−52	−1.02E + 04	3.11E + 01	5.47E−05	1.40E−02	1.60E + 01	2.00E−02	5.35E + 01	1.87E + 00	NA	NA
WQPSO [36]	3.56E−60	2.74E−62	−1.12E + 04	2.82E + 01	3.03E−05	1.58E−05	1.51E + 01	4.00E−02	5.11E + 01	**3.50E−01**	NA	NA
GSO [36]	8.78E−12	2.13E−12	−1.23E + 04	2.17E + 01	3.30E−02	3.00E−03	1.46E + 01	4.38E + 00	4.07E + 01	2.09E + 01	NA	NA
IGSO [36]	2.16E−21	1.22E−21	−1.25E + 04	**9.17E + 00**	4.09E−05	1.40E−02	8.79E + 00	2.33E + 00	1.05E + 01	1.37E + 00	NA	NA
PBA	**5.32E−203**	**0.00E + 00**	**−1.26E + 04**	1.87E + 01	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	2.48E + 01	4.93E−01	3.73E−15	**1.45E−15**

Table 9
Comparative study of PBA with state-of-the-art metaheuristics [47] for 30-D problems.

Algorithm	Sphere function	Schwefel's function	Rastrigin's function	Rotated Ackley's function	Rotated Griewank's function	Rotated Rastrigin's function
iCMAES-ILS [47]	1.00E−08	7.08E + 02	2.25E + 00	2.09E + 01	1.00E−08	1.72E + 00
NBIACMA [47]	1.00E−08	8.10E + 02	3.04E + 00	2.09E + 01	1.00E−08	2.91E + 00
PBA	**5.32E−203**	**5.22E + 00**	**0.00E + 00**	**4.44E−15**	**8.59E−16**	**0.00E + 00**

Table 10
Performance of PBA using the CEC 2014 test suite (10-D and 30-D).

Func.	10-D					30-D				
	Best	Worst	Median	Mean	Std.	Best	Worst	Median	Mean	Std.
F1	1.13E + 05	9.16E + 06	6.20E + 05	1.66E + 06	2.06E + 06	7.09E + 06	1.00E + 08	2.94E + 07	3.50E + 07	2.16E + 07
F2	4.53E + 05	5.12E + 07	9.27E + 06	1.09E + 07	8.64E + 06	4.83E + 07	9.61E + 08	2.60E + 08	3.05E + 08	1.89E + 08
F3	4.11E + 02	4.34E + 03	1.19E + 03	1.26E + 03	6.80E + 02	9.05E + 02	2.13E + 04	7.07E + 03	6.97E + 03	3.96E + 03
F4	4.02E + 02	4.43E + 02	4.14E + 02	4.20E + 02	1.49E + 01	4.97E + 02	6.53E + 02	5.84E + 02	5.78E + 02	3.62E + 01
F5	5.05E + 02	5.20E + 02	5.20E + 02	5.18E + 02	4.57E + 00	5.20E + 02	5.21E + 02	5.21E + 02	5.21E + 02	5.26E−02
F6	6.02E + 02	6.06E + 02	6.03E + 02	6.03E + 02	9.72E−01	6.11E + 02	6.21E + 02	6.16E + 02	6.16E + 02	2.40E + 00
F7	7.01E + 02	7.02E + 02	7.01E + 02	7.01E + 02	1.68E−01	7.01E + 02	7.08E + 02	7.04E + 02	7.04E + 02	1.72E + 00
F8	8.03E + 02	8.15E + 02	8.07E + 02	8.08E + 02	2.73E + 00	8.29E + 02	8.94E + 02	8.53E + 02	8.56E + 02	1.48E + 01
F9	9.06E + 02	9.21E + 02	9.14E + 02	9.15E + 02	3.47E + 00	9.77E + 02	1.04E + 03	1.01E + 03	1.01E + 03	1.33E + 01
F10	1.03E + 03	1.39E + 03	1.07E + 03	1.09E + 03	6.25E + 01	1.30E + 03	2.87E + 03	1.84E + 03	1.89E + 03	3.68E + 02
F11	1.23E + 03	1.88E + 03	1.48E + 03	1.48E + 03	1.33E + 02	3.36E + 03	5.45E + 03	4.54E + 03	4.49E + 03	5.35E + 02
F12	1.20E + 03	1.20E + 03	1.20E + 03	1.20E + 03	1.14E−01	1.20E + 03	1.20E + 03	1.20E + 03	1.20E + 03	1.43E−01
F13	1.30E + 03	1.30E + 03	1.30E + 03	1.30E + 03	6.64E−02	1.30E + 03	1.30E + 03	1.30E + 03	1.30E + 03	9.49E−02
F14	1.40E + 03	1.40E + 03	1.40E + 03	1.40E + 03	7.34E−02	1.40E + 03	1.40E + 03	1.40E + 03	1.40E + 03	4.57E−02
F15	1.50E + 03	1.51E + 03	1.50E + 03	1.50E + 03	8.25E−01	1.51E + 03	1.53E + 03	1.52E + 03	1.52E + 03	3.36E + 00
F16	1.60E + 03	1.60E + 03	1.60E + 03	1.60E + 03	4.11E−01	1.61E + 03	1.61E + 03	1.61E + 03	1.61E + 03	3.78E−01
F17	2.14E + 03	6.13E + 05	5.09E + 04	1.76E + 05	2.08E + 05	4.94E + 05	8.42E + 06	2.76E + 06	3.40E + 06	2.12E + 06
F18	1.88E + 03	2.01E + 04	4.46E + 03	7.27E + 03	5.26E + 03	1.31E + 05	5.23E + 06	1.60E + 06	1.70E + 06	1.06E + 06
F19	1.90E + 03	1.90E + 03	1.90E + 03	1.90E + 03	3.72E−01	1.91E + 03	1.94E + 03	1.91E + 03	1.91E + 03	5.23E + 00
F20	2.02E + 03	1.14E + 04	2.30E + 03	3.08E + 03	2.18E + 03	3.26E + 03	2.04E + 04	7.12E + 03	8.77E + 03	4.31E + 03
F21	2.39E + 03	3.11E + 04	6.33E + 03	7.74E + 03	5.41E + 03	4.11E + 04	2.09E + 06	3.35E + 05	4.39E + 05	3.40E + 05
F22	2.21E + 03	2.24E + 03	2.23E + 03	2.23E + 03	4.92E + 00	2.29E + 03	2.82E + 03	2.53E + 03	2.53E + 03	1.08E + 02
F23	2.50E + 03	2.63E + 03	2.51E + 03	2.54E + 03	5.44E + 01	2.50E + 03	2.62E + 03	2.62E + 03	2.60E + 03	3.88E + 01
F24	2.51E + 03	2.54E + 03	2.52E + 03	2.52E + 03	5.68E + 00	2.60E + 03	2.62E + 03	2.61E + 03	2.61E + 03	3.98E + 00
F25	2.61E + 03	2.70E + 03	2.65E + 03	2.66E + 03	2.84E + 01	2.70E + 03	2.71E + 03	2.70E + 03	2.70E + 03	1.41E + 00
F26	2.70E + 03	2.70E + 03	2.70E + 03	2.70E + 03	6.96E−02	2.70E + 03	2.80E + 03	2.70E + 03	2.70E + 03	1.93E + 01
F27	2.70E + 03	3.10E + 03	2.71E + 03	2.87E + 03	1.88E + 02	3.11E + 03	3.42E + 03	3.12E + 03	3.14E + 03	6.53E + 01
F28	3.18E + 03	3.32E + 03	3.23E + 03	3.23E + 03	2.93E + 01	3.00E + 03	4.20E + 03	3.94E + 03	3.90E + 03	2.16E + 02
F29	3.20E + 03	3.79E + 03	3.36E + 03	3.37E + 03	1.23E + 02	1.16E + 04	1.09E + 05	2.99E + 04	3.59E + 04	2.18E + 04
F30	3.62E + 03	4.87E + 03	4.04E + 03	4.04E + 03	2.43E + 02	7.26E + 03	2.93E + 04	1.45E + 04	1.55E + 04	4.24E + 03

4.4. Advantages and differences of PBA against other metaheuristics

Following the testing procedure that is employed by most of the developers of new evolutionary and swarm type of algorithms, the advantages of the proposed PBA were tested in three levels. In particular the proposed algorithm was assessed into the following levels: (i) In the first level the sensitivity of the performance was assessed with reference to its algorithmic parameters (the results of this test were presented in Section 3). (ii) In the second level the algorithm was assessed in comparison with other state-of-the-art algorithms and in particular against the best algorithms that were identified during the CEC 2013 competition presented in the "Special Session & Competition on Real-Parameter Single Objective Optimization" [47] (the results of this test were in Section 4.2). (iii) In the last level the performance of the algorithm was assessed based on the CEC 2014 benchmark test suite [23] (the results of this test were presented in Section 4.3). In general it can be said that the main advantage of PBA is robustness and efficiency. In particular, as it was identified through these tests the algorithms is not influenced by its algorithmic parameters, the computational effort required internally is low, it is not influenced significantly by the increase of the dimensionality while in most of the test functions outperforms many state-of-the-art metaheuristics.

Several researchers have presented variants of existing evolutionary and swarm algorithms, the proposed algorithm, although it belong to the swarm category of metaheuristic algorithms, to the authors' knowledge there is no similarity with any other existing algorithm. More specifically, the rules based on which new position vectors are generated are completely different compared to existing evolutionary based and to swarm optimization algorithms. As far as the evolutionary type of algorithms, there is no crossover/recombination and mutation operators; while compared to swarm optimization algorithms, there is no velocity vector neither exchange of experience using the cognitive and social parameters and the similarity is limited to the terms used for the position vectors (particles).

5. Conclusions

In this study a new metaheuristic search algorithm inspired by the *Pityogenes chalcographus* beetle called pity beetle algorithm is presented. The proposed algorithm has the ability to search large areas of possible solutions for an optimal global solution, while overcoming local optima. The current study underlines the importance that nature-inspired algorithms can have in providing solutions for NP-hard optimization problems. For the purposes of the current study the performance of the proposed algorithm is assessed in thirteen unimodal and multi-modal, separable and non-separable test functions found in the literature. In particular, a sensitivity analysis study is performed with reference to its algorithmic parameters along with a comparative test versus well established state-of-the-art metaheuristics. In the first part of the study it was found that the algorithm is not influenced by the selection of the

Table 11
Performance of PBA using the CEC 2014 test suite (50-D and 100-D).

Func.	50-D					100-D				
	Best	Worst	Median	Mean	Std.	Best	Worst	Median	Mean	Std.
F1	1.37E + 07	1.33E + 08	3.28E + 07	3.67E + 07	1.87E + 07	7.61E + 07	3.03E + 08	1.36E + 08	1.47E + 08	4.65E + 07
F2	1.24E + 08	2.04E + 09	7.75E + 08	8.22E + 08	4.71E + 08	4.09E + 08	6.37E + 09	2.04E + 09	2.31E + 09	1.38E + 09
F3	5.39E + 03	3.17E + 04	1.54E + 04	1.59E + 04	4.72E + 03	1.46E + 04	4.94E + 04	2.79E + 04	2.96E + 04	6.94E + 03
F4	5.86E + 02	8.44E + 02	7.04E + 02	7.03E + 02	5.50E + 01	8.57E + 02	1.38E + 03	1.07E + 03	1.06E + 03	1.30E + 02
F5	5.21E + 02	5.21E + 02	5.21E + 02	5.21E + 02	4.56E−02	5.21E + 02	5.21E + 02	5.21E + 02	5.21E + 02	3.73E−02
F6	6.23E + 02	6.40E + 02	6.32E + 02	6.32E + 02	3.45E + 00	6.55E + 02	6.92E + 02	6.81E + 02	6.78E + 02	9.55E + 00
F7	7.03E + 02	7.20E + 02	7.08E + 02	7.09E + 02	4.57E + 00	7.05E + 02	7.66E + 02	7.22E + 02	7.24E + 02	1.45E + 01
F8	8.60E + 02	1.00E + 03	9.18E + 02	9.20E + 02	3.28E + 01	9.78E + 02	1.27E + 03	1.12E + 03	1.11E + 03	6.99E + 01
F9	1.07E + 03	1.15E + 03	1.13E + 03	1.13E + 03	1.83E + 01	1.34E + 03	1.61E + 03	1.52E + 03	1.51E + 03	4.89E + 01
F10	1.88E + 03	4.98E + 03	3.34E + 03	3.36E + 03	7.68E + 02	4.09E + 03	1.22E + 04	8.09E + 03	7.87E + 03	2.00E + 03
F11	6.76E + 03	9.65E + 03	8.60E + 03	8.43E + 03	6.21E + 02	1.63E + 04	2.15E + 04	1.94E + 04	1.94E + 04	1.17E + 03
F12	1.20E + 03	1.20E + 03	1.20E + 03	1.20E + 03	1.67E−01	1.20E + 03	1.20E + 03	1.20E + 03	1.20E + 03	1.61E−01
F13	1.30E + 03	1.30E + 03	1.30E + 03	1.30E + 03	8.29E−02	1.30E + 03	1.30E + 03	1.30E + 03	1.30E + 03	6.15E−02
F14	1.40E + 03	1.40E + 03	1.40E + 03	1.40E + 03	4.84E−02	1.40E + 03	1.40E + 03	1.40E + 03	1.40E + 03	4.82E−02
F15	1.53E + 03	1.57E + 03	1.54E + 03	1.54E + 03	8.71E + 00	1.57E + 03	1.73E + 03	1.59E + 03	1.61E + 03	3.52E + 01
F16	1.62E + 03	1.62E + 03	1.62E + 03	1.62E + 03	4.77E−01	1.64E + 03	1.64E + 03	1.64E + 03	1.64E + 03	6.17E−01
F17	2.00E + 06	2.43E + 07	7.27E + 06	7.72E + 06	3.93E + 06	1.08E + 07	6.82E + 07	3.21E + 07	3.34E + 07	1.26E + 07
F18	2.38E + 06	4.58E + 07	1.05E + 07	1.25E + 07	7.99E + 06	1.31E + 07	1.74E + 08	6.45E + 07	6.78E + 07	3.76E + 07
F19	1.92E + 03	2.04E + 03	1.95E + 03	1.95E + 03	2.49E + 01	1.96E + 03	2.12E + 03	2.07E + 03	2.07E + 03	2.86E + 01
F20	8.94E + 03	3.13E + 04	1.85E + 04	1.76E + 04	4.26E + 03	3.34E + 04	9.53E + 04	7.83E + 04	7.30E + 04	1.69E + 04
F21	5.41E + 05	5.81E + 06	2.45E + 06	2.60E + 06	1.34E + 06	3.56E + 06	3.59E + 07	1.43E + 07	1.54E + 07	6.75E + 06
F22	2.56E + 03	3.57E + 03	3.18E + 03	3.14E + 03	2.14E + 02	3.88E + 03	5.53E + 03	4.93E + 03	4.87E + 03	3.49E + 02
F23	2.51E + 03	2.68E + 03	2.66E + 03	2.64E + 03	4.20E + 01	2.50E + 03	2.75E + 03	2.67E + 03	2.66E + 03	6.40E + 01
F24	2.60E + 03	2.67E + 03	2.62E + 03	2.62E + 03	1.06E + 01	2.60E + 03	2.77E + 03	2.65E + 03	2.66E + 03	2.75E + 01
F25	2.70E + 03	2.73E + 03	2.70E + 03	2.70E + 03	3.60E + 00	2.70E + 03	2.75E + 03	2.71E + 03	2.71E + 03	6.88E + 00
F26	2.70E + 03	2.80E + 03	2.70E + 03	2.70E + 03	1.38E + 01	2.70E + 03	2.80E + 03	2.80E + 03	2.77E + 03	4.65E + 01
F27	3.18E + 03	4.05E + 03	3.87E + 03	3.83E + 03	1.49E + 02	4.03E + 03	5.35E + 03	4.87E + 03	4.81E + 03	2.94E + 02
F28	3.19E + 03	5.59E + 03	4.86E + 03	4.81E + 03	3.54E + 02	3.45E + 03	9.91E + 03	7.78E + 03	7.70E + 03	1.40E + 03
F29	4.23E + 04	8.02E + 05	2.23E + 05	2.87E + 05	1.81E + 05	1.96E + 05	3.63E + 06	1.09E + 06	1.51E + 06	1.06E + 06
F30	2.16E + 04	1.11E + 05	3.85E + 04	4.16E + 04	1.53E + 04	4.00E + 04	7.59E + 05	2.58E + 05	2.64E + 05	1.36E + 05

Table 12
PBA compared to other algorithms – Wilcoxon's ranksum test.

With reference to PBA		10-D	30-D	50-D	100-D
SOO [49]	Better	12	12	14	13
	Worst	17	15	15	16
	Equal	1	3	1	1
FWA-DM [50]	Better	26	22	21	14
	Worst	3	5	7	15
	Equal	1	3	2	1
ADE [51]	Better	–	–	16	22
	Worst	–	–	14	8
	Equal	–	–	0	0
ADE-POBL [51]	Better	27	19	20	22
	Worst	2	8	9	7
	Equal	1	3	1	1
L-SHADE [52]	Better	27	25	24	24
	Worst	2	3	5	5
	Equal	1	2	1	1

Table 13
PBA complexity in seconds.

	T_o	T_1	T_2	$(T_2-T_1)/T_o$
$D = 10$	0.14	1.53	2.52	7.16
$D = 30$	0.14	1.84	3.77	14.00
$D = 50$	0.14	2.20	5.05	20.62
$D = 100$	0.14	3.77	9.27	39.91

algorithmic parameters, while a cost efficient methodology for the selection of algorithmic parameters for the case of cost demanding black box optimization problems was proposed. In the second part of the study, it was found that for the test functions considered the proposed algorithm outperformed the well-established, state-of-the-art metaheuristics. In particular, the proposed algorithm outperformed most of these metaheuristics (some of them found in the literature and the best ones identified during the CEC 2013 and 2014 competitions in the "Special Session & Competition on Real-Parameter Single Objective Optimization"). As far as the Rosenbrock's function is concerned, it performed equally with the rest of the algorithms both in terms of mean best objective function value and standard deviation.

The proposed algorithm was found to be capable of handling the well-established single objective optimization test functions proposed in modern literature. The algorithm was also performed significantly well in the CEC 2014 Single Objective Unconstrained Optimization Contest test suite. It is also notable that PBA is not demanding from a computational or memory usage point of view while it can easily be applied on various engineering problems. The behavior of PBA in constrained optimization problems and multi-objective ones will be assessed while an improved and parallel implementation of PBA will also be presented in future works.

Acknowledgments

This research has been supported by the OptArch project: "Optimization Driven Architectural Design of Structures" (No: 689983) belonging to the Marie Skłodowska-Curie Actions (MSCA) Research and Innovation Staff Exchange (RISE) H2020-MSCA-RISE-2015.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.advengsoft.2018.04.007.

References

[1] Kennedy J, Eberhart RC, Shi Y. Swarm intelligence. Morgan Kaufmann Publishers; 2001. Series in evolutionary computation.
[2] Dorigo M, Stützle T. Ant colony optimization. The MIT Press; 2004.

[3] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput J 2008;8(1):687–97.
[4] Yang XS. Nature-inspired metaheuristic algorithms. Frome: Luniver Press; 2008.
[5] van Laarhoven PJ, Aarts EH. Simulated Annealing: theory and applications (Mathematics and its applications). Kluwer Academic Publishers; 2010.
[6] Geem ZW. Recent advances in harmony search algorithm. Studies in computational intelligence. Springer; 2010.
[7] Das S, Suganthan PN. Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 2011;15(1):4–31. art. no. 5601760.
[8] Kaveh A, Khayatazad M. A new metaheuristic method: ray optimization. Comput Struct 2012;112–113:283–94.
[9] Michalewicz Z. Genetic algorithms + data structures = evolution programs. 3rd ed. Springer; 2012.
[10] Kaveh A, Farhoudi N. A new optimization method: dolphin echolocation. Adv Eng Softw 2013;59:53–70.
[11] Gandomi AH, Alavi AH. Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 2012;17(12):4831–45.
[12] Sarker RA, Elsayed SM, Ray T. Differential evolution with dynamic parameters selection for optimization problems. IEEE Trans Evol Comput 2014;18(5):689–707. art. no. 6600821.
[13] Kaveh A, Talatahari S, Sheikholeslami R, Keshvari M. Chaotic swarming of particles: a new method for size optimization of truss structures. Adv Eng Softw 2014;67:136–47.
[14] Kaveh A, Dadras A. A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Softw 2017;110:69–84.
[15] Chen W-N, Zhang J, Lin Y, Chen N, Zhan Z-H, Chung HS-H, Li Y, Shi Y-H. Particle swarm optimization with an aging leader and challengers. IEEE Trans Evol Comput 2013;17(2):241–58.
[16] Li X, Yao X. Cooperatively coevolving particle swarms for large scale optimization. IEEE Trans Evol Comput 2012;16(2):210–24. art. no. 5910380.
[17] Parsopoulos KE, Vrahatis MN. UPSO: a unified particle swarm optimization scheme, Lecture Series on Computational Sciences, Vol. 1, Proc. Int. Conf. Comput. Meth. Sci. Engin. (ICCMSE 2004), VSP International Science Publishers, Zeist, The Netherlands, 2004, 868-73.
[18] Mendes R, Kennedy J, Neves J. The fully informed particle swarm: simpler, maybe better. IEEE Trans Evol Comput 2004;8(3):204–10.
[19] van den Bergh F, Engelbrecht AP. A cooperative approach to participle swam optimization. IEEE Trans Evol Comput 2004;8(3):225–39.
[20] Poli R, Kennedy J, Blackwell T. Particle swarm optimization. Swarm Intell 2007;1(1):33–57.
[21] Kallioras N, Lagaros NC, Avtzis D. A new metaheuristic inspired by bark beetles for solving engineering problems. Proceedings of the 11th international congress on mechanics, HSTAM. 2016.
[22] Schwerdtfeger F. Ein beitrag zur fortpflanzungsbiologie des borkenkafers pityogenes chalcographus. L Z Angew Entomol 1929;15:335–427.
[23] Liang JJ, Qu B-Y, Suganthan PN. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization Singapore: Computational Intelligence Laboratory, Zhengzhou University; 2013. Technical Report 201311 Zhengzhou China and Technical Report, Nanyang Technological University December.
[24] Knizek M, Stauffer C, Avtzis DN, Wegensteiner R. Pityogenes chalcographus. Forestry compendium. Wallingford, UK: CAB International; 2005. www.cabicompendium.org/fc CD-ROM version ISBN: 0 85199 031 2.
[25] Postner M. Scolytidae, borkenkafer. In: Schwenke W, editor. Die forstschädlinge Europas. Band 2, Berlin: Paul Parey; 1974.
[26] Pfeffer, A., (1995). Zentral und westpalaarktische borken und kernkafer. Pro Entomologia, c/o Naturhistorisches Museum Basel.
[27] Wood SL, Bright DE. A catalog of scolytidae and platypodidae (coleoptera). *Great Basin Nat* Mem 1992(13):835–1553. A 1-833 & B.
[28] Avtzis DN, Arthofer W, Stauffer C, Avtzis N, Wegensteiner R. *Pityogenes chalcographus* (Coleoptera, Scolytinae) at the southernmost borderline of Norway spruce in Greece. Entomol Hell 2010;19:3–13.
[29] Vité JP. Ist die vorbeugende Begiftung von Fangbäumen zweckmässig. Allg Forstz Waldwirtsch Umweltvorsorge 1965;20:438–9.
[30] Novak V. Atlas of insects harmful to forest trees 1. Praha: Elsevier Scientific Publishing Company; 1976.
[31] Zumr V, Zoldán T. Reproductive cycles of *Ips typographus, I. Amitinus* and *pityogenes chalcographus* (coleoptera, scolytidae). Acta Entomol Bohemoslov 1981;78:280–9.
[32] Zuber M. Ökologie der borkenkäfer. Biol unserer Zeit 1994;3:144–52.
[33] Talbi El-G. Metaheuristics: from design to implementation. Hoboken, New Jersey: John Wiley & Sons; 2009.
[34] Maaranen H, Miettinen K, Penttinen A. On initial populations of a genetic algorithm for continuous optimization problems. J Global Optim 2007;37:405–36.
[35] Le Riche R, Haftka RT. On global optimization articles in SMO. Struct Multidiscip Optim 2012;46(5):627–9.
[36] Liang JJ, Qin AK, Suganthan PN, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 2006;10(3):281–95.
[37] Chen D, Wang J, Zou F, Hou W, Zhao C. An improved group search optimizer with operation of quantum-behaved swarm and its application. Appl Soft Comput J 2012;12(2):712–25.
[38] Shi Y, Eberhart RC. Modified particle swarm optimizer. Proceedings of the IEEE conference on evolutionary computation, ICEC. 1998. p. 69–73.
[39] Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 2002;6(1):58–73.
[40] Kennedy J, Mendes R. Population structure and particle swarm performance. Proceedings of the 2002 congress on evolutionary computation. 2. 2002. p. 1671–6.
[41] Peram T, Veeramachaneni K, Mohan CK. Fitness-distance-ratio based particle swarm optimization. *Proceedings of the IEEE swarm intelligence symposium,* SIS 2003. 2003. p. 174–81.
[42] Shi Y, Eberhart RC. Particle swarm optimization with fuzzy adaptive inertia weight. Proceedings of the workshop on particle swarm optimization. 2001. p. 101–6.
[43] Ratnaweera A, Halgamuge SK, Watson HC. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evol Comput 2004;8(3):240–55.
[44] Liang JJ, Suganthan PN. Dynamic multi-swarm particle swarm optimizer. Proceedings of IEEE swarm intelligence symposium, SIS 2005. 2005. p. 124–9.
[45] Kang F, Li J, Ma Z. Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf Sci 2011;181(16):3508–31.
[46] Salomon R. Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms. BioSystems 1996;39(3):263–78.
[47] Liang JJ, Qu B-Y, Suganthan PN, Hernández-Díaz AG. Problem definitions and evaluation criteria for the CEC 2013 special session and competition on real-parameter optimization Singapore: Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China, Nanyang Technological University; 2013. Technical Report 201212 January.
[48] Loshchilov I, Stuetzle T, Liao T. Ranking results of CEC'13 special session & competition on real-parameter single objective optimization. 2013 *IEEE congress on evolutionary computation,* CEC. Cancun, Mexico; 2013. June 20-23.
[49] Derrac J, García S, Molina D, Herrera F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 2011;1(1):3–18.
[50] Preux P, Munos R, Valko M. Bandits attack function optimization, *Proceedings of the 2014 IEEE congress on evolutionary computation*, CEC 2014, art. no. 6900558, pp. 2245–2252, 2014.
[51] Yu C, Kelley L, Zheng S, Tan Y. Fireworks algorithm with differential mutation for solving the CEC 2014 competition problems. *Proceedings of the 2014 IEEE congress on evolutionary computation*, CEC 2014. 2014. p. 3238–45. art. no. 6900590.
[52] Hu Z, Bao Y, Xiong T. Partial opposition-based adaptive differential evolution algorithms: evaluation on the CEC 2014 benchmark set for real-parameter optimization. *Proceedings of the 2014 IEEE congress on evolutionary computation*, CEC 2014. 2014. p. 2259–65. art. no. 6900489.
[53] Tanabe R, Fukunaga AS. Improving the search performance of SHADE using linear population size reduction. *Proceedings of the 2014 IEEE congress on evolutionary computation*, CEC 2014. 2014. p. 1658–65. art. no. 6900380.