Artificial Intelligence Review
https://doi.org/10.1007/s10462-019-09732-5

Novel meta-heuristic bald eagle search optimisation algorithm

H. A. Alsattar[1,2] · A. A. Zaidan[1] · B. B. Zaidan[1]

Abstract

This study proposes a bald eagle search (BES) algorithm, which is a novel, nature-inspired meta-heuristic optimisation algorithm that mimics the hunting strategy or intelligent social behaviour of bald eagles as they search for fish. Hunting by BES is divided into three stages. In the first stage (selecting space), an eagle selects the space with the most number of prey. In the second stage (searching in space), the eagle moves inside the selected space to search for prey. In the third stage (swooping), the eagle swings from the best position identified in the second stage and determines the best point to hunt. Swooping starts from the best point and all other movements are directed towards this point. BES is tested by adopting a three-part evaluation methodology that (1) describes the benchmarking of the optimisation problem to evaluate the algorithm performance, (2) compares the algorithm performance with that of other intelligent computation techniques and parameter settings and (3) evaluates the algorithm based on mean, standard deviation, best point and Wilcoxon signed-rank test statistic of the function values. Optimisation results and discussion confirm that the BES algorithm competes well with advanced meta-heuristic algorithms and conventional methods.

Keywords Bald eagle behaviour · Meta-heuristic algorithm · Optimisation · Unconstrained benchmark problem

1 Introduction

Optimisation remains a significant challenge in artificial computation (Sameer et al. 2019; Tariq et al. 2018; Zaidan et al. 2017). Accordingly, many algorithms have been developed to solve such a problem. However, two issues should be addressed to guarantee a successful solution to this problem: how to identify the global and local optimisation and how to preserve such optimisation until the end of the search (Qu et al. 2012).

Over the last two decades, nature-inspired computation has attracted wide interest amongst researchers, given that nature is an important source of concepts, mechanisms and ideas for

✉ A. A. Zaidan
aws.alaa@fskik.upsi.edu.my

1 Department of Computing, Faculty of Arts, Computing and Creative Industry, Universiti Pendidikan Sultan Idris, Tanjung Malim, Malaysia

2 Al-Rafidain University College, Baghdad, Iraq

Published online: 01 July 2019

designing artificial computing systems that solve many complex mathematical problems (Barhen et al. 1997). Individuals must adapt to their surrounding environments to ensure the survival and long-term preservation of their breed (Whitley 2001). This process is known as evolution (Fogel 1995, 2009; Fogel et al. 1965; Schwefel 1995; Michalewicz and Attia 1994). Maintaining the time of reproduction can also sustain the features that foster the competitiveness of individuals and remove their weak features. Only those good individuals from the surviving species can transmit genetically modified genes to their descendants. This process, which is known as natural selection, has inspired 'evolutionary algorithms' (EAs), which are amongst the most widespread and successful algorithms being applied in research. Several types of EAs have been employed in the literature, including genetic algorithms (Houck et al. 1995; Joines and Houck 1994; Kazarlis and Petridis 1998; Holland 1992), genetic programming (Koza 1992), evolutionary programming and evolutionary strategies (Yao et al. 1999; Rechenberg 1994; Whitley 2001; Yao et al. 1999b; Yao and Liu 1997).

Evolutionary optimisation techniques are the most widely used intelligent computing techniques to solve many problems, such as the combinatorial and nonlinear optimisation problems (Fiacco and McCormick 1968). These techniques easily address different types of issues by using and integrating prior information into an evolutionary search yielding process to efficiently explore a state space of possible solutions. However, EAs are unable to find the optimal solution for numerous issues despite the aforementioned advantages; therefore, many researchers have merged these algorithms with extant technologies to improve their solutions (Whitley 2001).

Swarm intelligence (SI) is another form of intelligent computing technique that includes particle swarm optimisation (PSO) (Birge 2003; Shi and Eberhart 1998; Kennedy and Eberhart 1995), which mimics the swarm behaviour of birds or fish (Li 2003); ant colony optimisation (Dorigo et al. 2006), which mimics the foraging and schooling behaviour of ants and other algorithms, such as gravitational search (Rashedi et al. 2009), grey wolf optimiser (GWO) (Mirjalili et al. 2014), artificial bee colony (Karaboga and Basturk 2007), moth–flame optimisation (Mirjalili 2015), whale optimisation (Mirjalili and Lewis 2016), group search optimiser (He et al. 2006, 2009) and ant lion optimiser algorithms (Mirjalili 2015), as well as many algorithms modified on the PSO algorithms, such as comprehensive learning particle swarm optimiser (CLPSO) (Liang et al. 2006) and fitness-distance-ratio-based particle swarm optimisation (FDR-PSO) (Peram et al. 2003) and ensemble particle swarm optimiser (EPSO) (Lynn and Suganthan 2017). SI solves many problems by simulating the normal behaviour of some animals when moving from one place to another in search of food. This technique is generally influenced by the size and nonlinearity of the problem. Despite obtaining the optimal solution to computational and combinatorial problems, the majority of the existing analytical methods are unable to converge such problems.

SI offers many advantages. For example, each individual can improve his/her search efficiency by moving from one position to another, whilst all individuals within a swarm can improve their respective positions. In EAs, the weak and inefficient individuals are neglected and replaced by highly competent individuals. A swarm continuously explores new areas within the search space to rapidly reach global optimisation areas. However, SI also has its own disadvantages. For example, collective movement may induce a state of mass decline in the local optimum and the continued failure of individuals to escape from this area can lead to the early suspension of the exploration (Del Valle et al. 2008).

Nature-inspired techniques have been developed owing to their ability to address various issues and the possibility of integrating evolutionary techniques with swarm techniques to create new technologies that can solve these problems. Such technologies maximise the advantages offered by SI in terms of searching within the best position in the swarm and capitalise on the capability of evolutionary techniques to explore the search space frequently and avoid the local optimum. Thus, the introduction of techniques such as the bald eagle search (BES), which is a nature-inspired technique for solving optimisation problems that mimics the behaviour of bald eagles when searching for food, is useful. To the best of our knowledge, an algorithm that mimics the behaviour of bald eagles, which are known for pack hunting, has yet to be developed. Bald eagles search for food in three stages. In the first stage, the eagles identify an area where they will conduct their search. In the second stage, the eagles search for food within the selected area. In the third stage, the eagles choose and attack a prey (Sörensen 2015).

The movement of bald eagles in the three stages depends on a centre point. In the first stage, the eagles move from the centre point towards the selected search area. In the second stage, the eagles search within the search space and around the centre point. In the third stage, the eagles move towards a prey from the centre point of the search area. The mean point for all the points of search was based on this point as a central point for the launch and search of the eagle. The advantages of evolutionary and swarm techniques have been integrated into the construction of the BES algorithm. The first and second stages are considerably similar to the search behaviour of evolutionary techniques. Specifically, the first stage of the BES algorithm relays and collects all search points starting from the original point to the best point. The second stage is treated as an evolution stage for all search points and maximises the preferred search points along the direction leading to the centre point. The third stage mimics the behaviour of SI whilst moving towards the best point, with the benefit of the previous site for each point of search. Considering that the centre point is a local search point enables this stage to direct the search in a large area and overcome the shortcomings in the previous stages of search.

The main objective of this study is to propose a novel nature-inspired technique for solving optimisation problems. The remainder of this paper is organised as follows. Section 2 briefly reviews the search behaviour of bald eagles and describes each stage of the proposed BES algorithm. Section 3 discusses the evaluation methodology and the results. Section 4 presents the conclusions.

2 BES algorithm

2.1 Behaviour of bald eagle during hunting

'Bald' is a derivation of the old English word *balde vgbyhuvu*, which means 'white'. Hence, bald eagles are not bald. Bald eagles are occasional predators and are at the top of the food chain only because of their size. Furthermore, bald eagles are considered scavengers that feast on any available, easy and protein-rich food. Bald eagles are an opportunistic forager that mainly select fish (alive or dead), especially salmon, as primary food. Birds that make optimal hunting decisions can evaluate the energetic cost of the hunting attempt, energy content of the prey and probability of success in various habitats utilising multiple attack methods (Todd et al. 1982). Bald eagles frequently hunt from perches but may also hunt whilst in flight. They are capable of spotting fish at enormous distances because obtaining fish from water is difficult. Thus, only 1 in 20 attempts of attack may succeed (Stalmaster and Kaiser 1997). Thereafter, bald eagles rest because hunting consumes substantial energy. Figure 1 shows the behaviour of bald eagles during hunting time. When they start to search for food over a water spot, these eagles set off in a specific direction and select a certain area to begin the search. Accordingly, finding the search space is achieved by self-searching and tracking other birds to the concentrations of fish (dead or alive) (Stalmaster 1987).

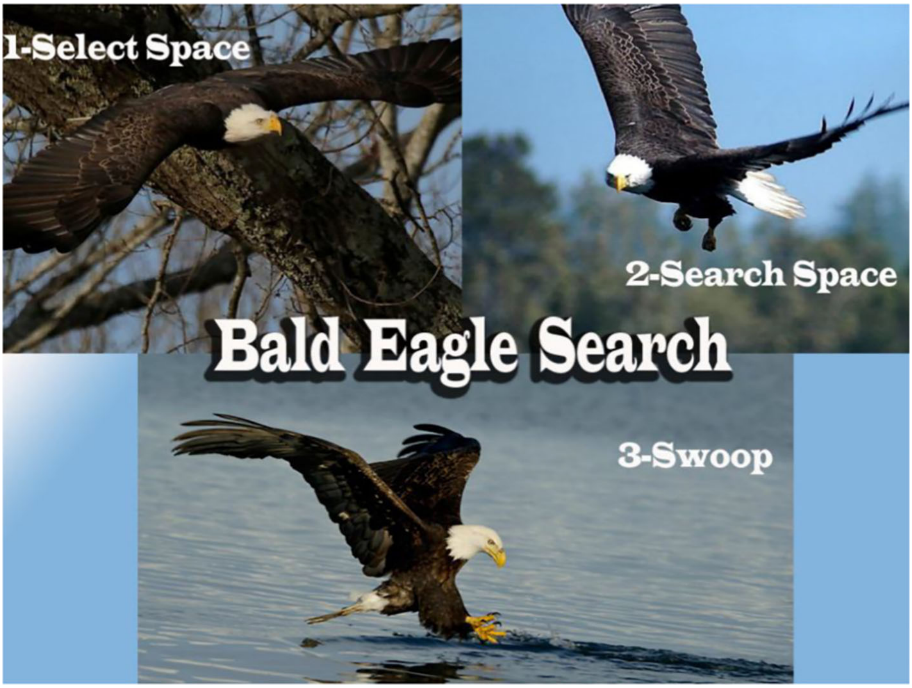

Fig. 1 Behaviour of bald eagle during hunting

Thereafter, bald eagles will go directly to the particular area. The specified search space can be justified because space selection is the first stage of hunting behaviour (see Fig. 2). Foraging success is high within the range of 5 m from the shore (47%) compared with deep in the water. That is, an important consideration for foraging habitat is bald eagles select the middle space between the land surface and deep water that is distant from the shallow water (Stalmaster and Gessaman 1982). Specifically, a pair of eagles hunt daily over 250 ha of open pasture. When the eagles reach the area, they will begin their search; the selected area is not farther than 700 m from their nest because the energy aspect is a critical factor in searching (Lasserre 2001).

Furthermore, bald eagles take advantage of stormy air whilst flying high. Soaring is activated by increased wind speed, in which eagles consume substantial time on flying. Eagles are observed to have slithering, graceful, motionless flights for hours at a time (Stalmaster and Kaiser 1997; Hansen 1986; Hansen et al. 1984). They also have outstanding eyesight, thereby enabling them to observe fish in water or dead fish from hundreds of feet up in the air. An eagle's eye is as large as a human eye but is more powerful. Moreover, an eagle's eye has perfect vision, which is four times that of humans. Eagles can also see in two directions at the same time, forward and side views. Whilst eagled fly thousands of feet in the air, scanning becomes easy with a twisting motion and the eagle can spot a prey over an area of nearly 3 m^2 (Hansen 1986). The second stage of hunting behaviour is seeing the prey (see Fig. 2). Once the eagles see the prey, they will start the last stage of hunting behaviour, which descends with a gradual flow of motion to reach the prey at a high speed and snatch the fish from the water (see Fig. 2). A consumption card estimated at five branches of the search energy spiral is used by eagles in search of fish (Liang et al. 2006).

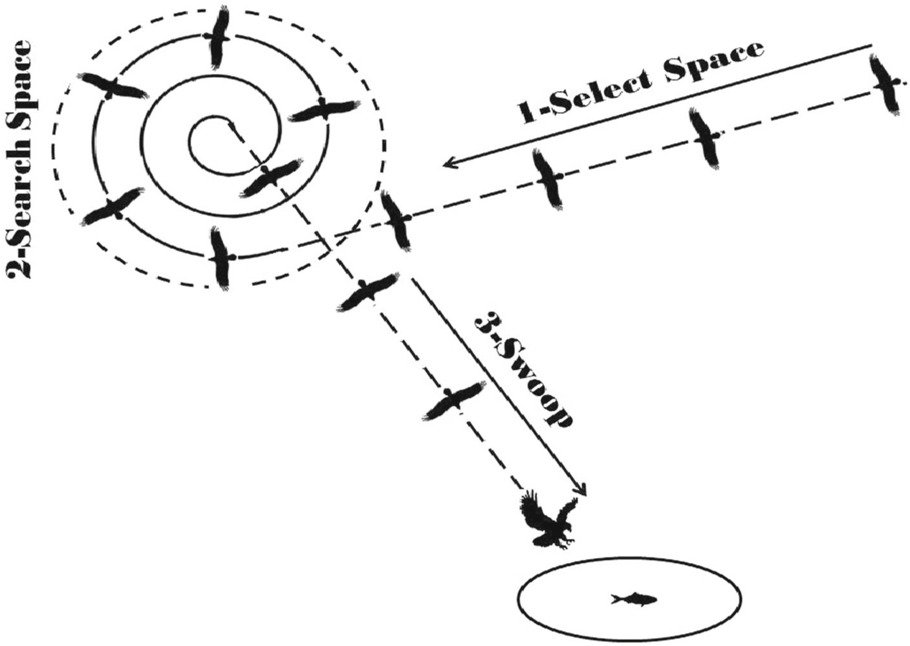

Fig. 2 Co-sequences for the three main stages of hunting by BES

2.2 BES algorithm

The proposed BES algorithm mimics the behaviour of bald eagles during hunting to justify the co-sequences of each stage of hunting. Accordingly, this algorithm can be divided into three parts, namely, selecting the search space, searching within the selected search space and swooping.

2.2.1 Select stage

In the select stage, bald eagles identify and select the best area (in terms of amount of food) within the selected search space where they can hunt for prey. Equation (1) presents this behaviour mathematically.

$$Pnew,\quad i = P_{best} + \alpha * r(P_{mean} - P_i) \tag{1}$$

where α is the parameter for controlling the changes in position that takes a value between 1.5 and 2 and r is a random number that takes a value between 0 and 1. In the selection stage, bald eagles select an area on the basis of the available information from the previous stage. The eagles randomly select another search area that differs from but is located near the previous search area. P_{best} denotes the search space that is currently selected by bald eagles based on the best position identified during their previous search. The eagles randomly search all points near the previously selected search space. Meanwhile, P_{mean} indicates that these eagles have used up all information from the previous points. The current movement of bald eagles is determined by multiplying the randomly searched prior information by α. This process randomly changes all search points (Hatamlou 2012).

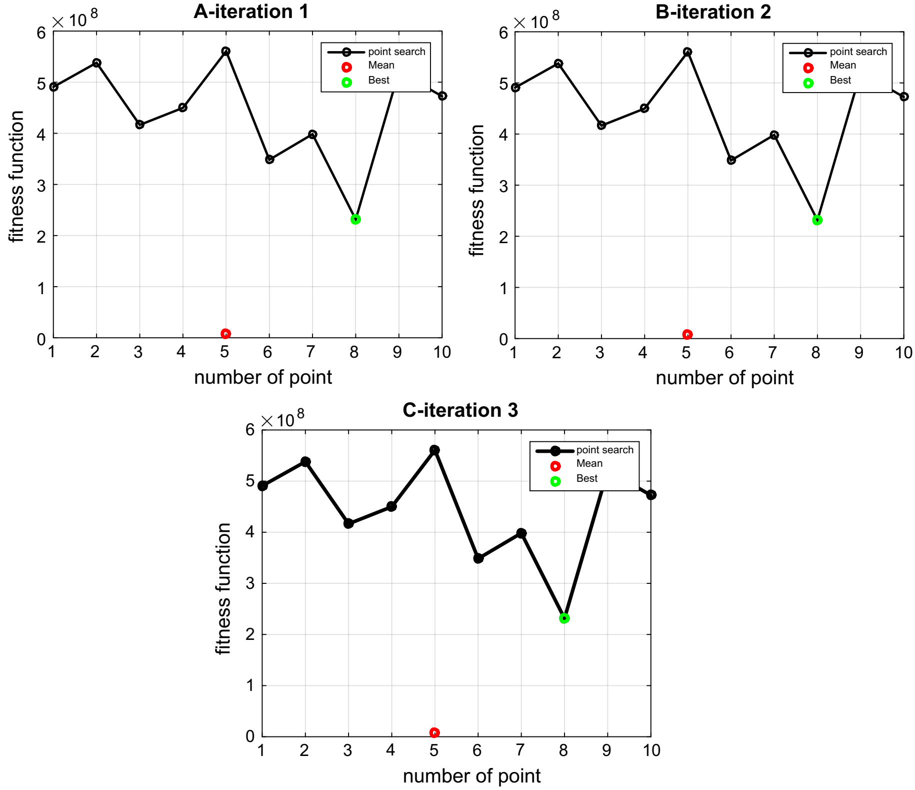

Fig. 3 Improvised selection stage: **a** selection stage after one alteration, **b** selection stage after two alterations and **c** selection stage after three alterations

We solve the Rosenbrock function in each stage based on the size of search point 10 to improve the efficiency of the random solution. Figure 3 shows that the selection stage effectively improves all solutions within the search based on the mean and best points. Figure 3a shows the location of the best and mean solutions within the search space. The mean point has the best location in the search space, whereas the selected space depends on the difference between the search and mean points. Figure 3b shows that search point 10 is located near the best point. Figure 3c shows the improvement in all points within the search space and the new best point within search point 3.

2.2.2 Search stage

In the search stage, bald eagles search for prey within the selected search space and move in different directions within a spiral space to accelerate their search. The best position for the swoop is mathematically expressed in Eq. (2).

$$P_{i,new} = P_i + y(i) * (P_i - P_{i+1}) + x(i) * (P_i - P_{mean}) \tag{2}$$

$$x(i) = \frac{xr(i)}{\max(|xr|)}, \quad y(i) = \frac{yr(i)}{\max(|yr|)} \quad \text{(a)}$$

$$xr(i) = r(i) * \sin(\theta(i)), \quad yr(i) = r(i) * \cos(\theta(i)) \quad \text{(b)}$$

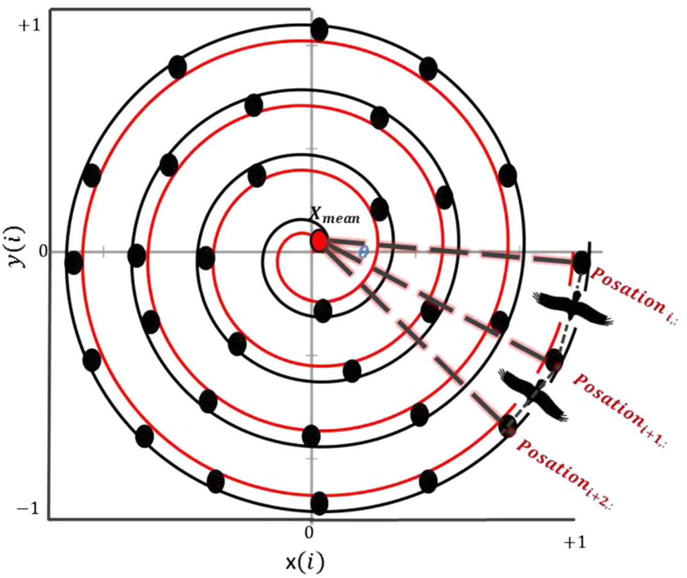

Fig. 4 Bald eagles searching within a spiral space

$$\theta(i) = a * \pi * rand \quad \ldots(\mathrm{c}) \quad and \quad r(i) = \theta(i) + R * rand \quad \ldots(\mathrm{d}),$$

where a is a parameter that takes a value between 5 and 10 for determining the corner between point search in the central point and R takes a value between 0.5 and 2 for determining the number of search cycles. Figure 4 shows that bald eagles move in a spiral direction within the selected search space and determine the best position for swooping and hunting for prey. We use a polar plot property to mathematically represent this movement. This property also enables the BES algorithm discover new spaces and increase diversification by multiplying the difference between the current and next points by the point of polar in the y-axis and by adding the difference between the current and centre points with the point of polar in the x-axis. We use the mean solution in the search point because all search points move towards the centre point. All points in the polar plot take a value between − 1 and 1 and we use a special equation for the spiral shape (a–d). Moreover, a and R represent the parameters for the change in the spiral shape. Figure 5 shows the spiral shape when these parameters are changed.

The points move around the centre point during the search stage. When parameters a and R are changed, the algorithm increases diversification to escape from the local optimum and to continuously obtain an efficient solution. Figure 6 shows the improvement in the fitness function during the search stage, whilst Fig. 6a shows the location of the best and mean points. The best point has a better location compared with the mean point in the search space. Figure 6b shows the improvement in all points and the best solution that is obtained in point 4. Figure 6c shows the new location of all points in the search space and the best point that is obtained in point 7. The search space depends on the movement of points from one location to another, whereas the mean point is based on the movement of these points around a spiral.

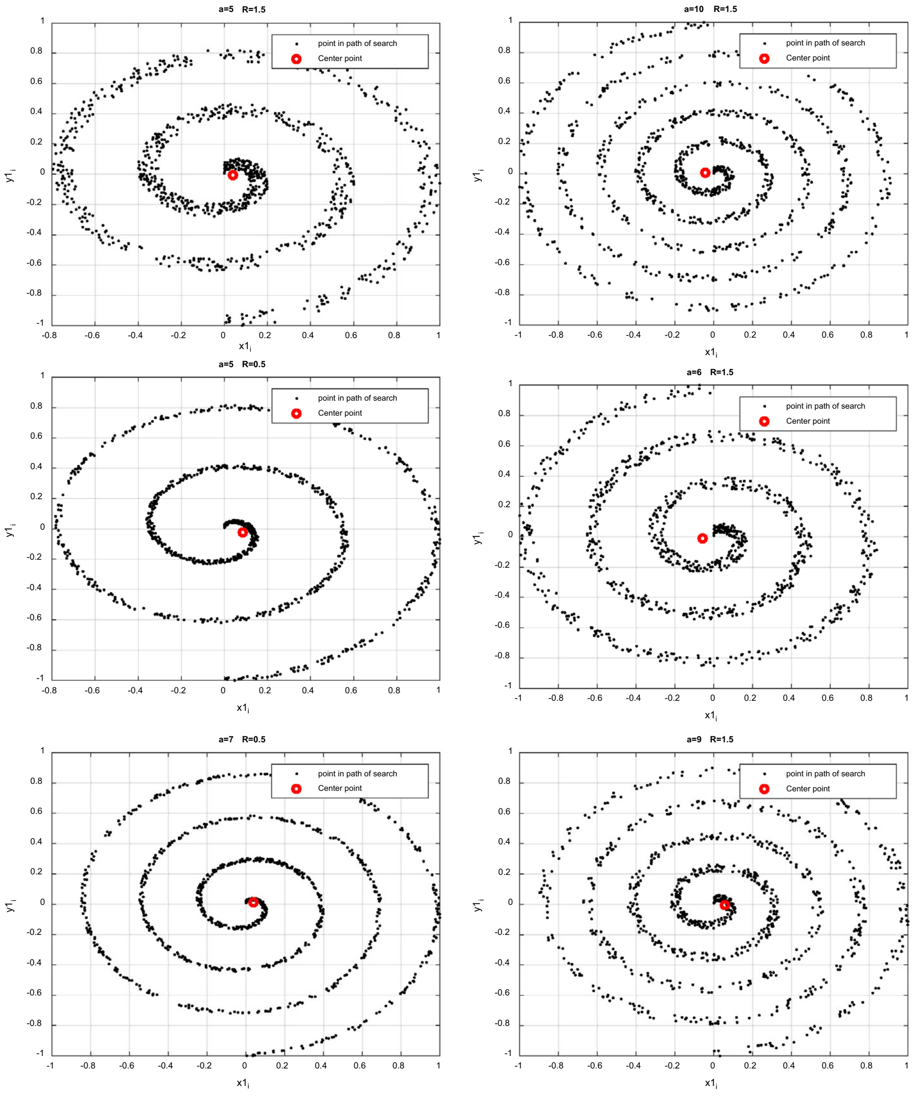

Fig. 5 Shape of the spiral when parameters a and R are changed

2.2.3 Swooping stage

In the swooping stage, bald eagles swing from the best position in the search space to their target prey. All points also move towards the best point. Equation (3) mathematically illustrates this behaviour.

$$P_{i,new} = rand * P_{best} + x1(i) * (P_i - c1 * P_{mean}) + y1(i) * (P_i - c2 * P_{best}) \quad (3)$$

$$x1(i) = \frac{xr(i)}{\max(|xr|)}, \quad y1(i) = \frac{yr(i)}{\max(|yr|)}$$

$$xr(i) = r(i) * \sinh[\theta(i))], \quad yr(i) = r(i) * \cosh[\theta(i))]$$

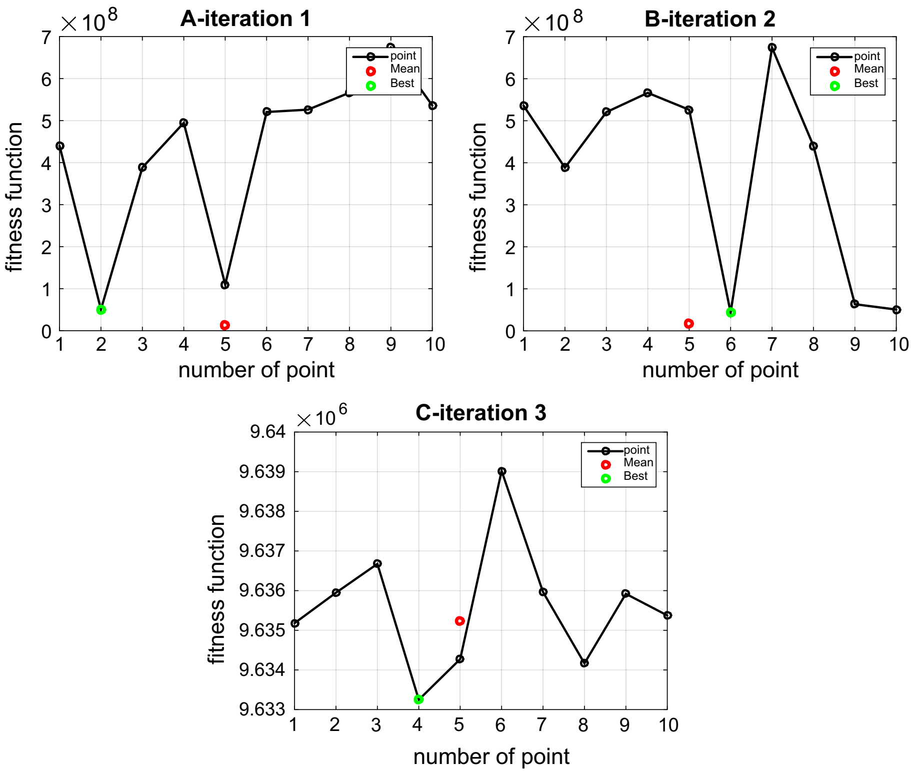

Fig. 6 Improvements in the fitness function during the search stage **a** after one alteration, **b** after two alterations and **c** after three alterations

$$\theta(i) = a * \pi * rand \quad and \quad r(i) = \theta(i)$$
$$where \quad c1, c2 \in [1, 2].$$

The movement of the eagles takes different shapes. We use a polar equation to plot the movement of these eagles whilst swooping. Additionally, we compute for the best point by multiplying the difference between the current and centre points by the point of polar in the x-axis and multiplying the difference between the current and best points by the point of polar in the y-axis. The best solution must be multiplied by a random number because parameters c1 and c2 increase the movement intensity of bald eagles towards the best and centre points (see Fig. 7).

The movement of points in the swoop equation when the parameters were changed was circular to the best point. The mean of population in this stage can help the algorithm in intensification and diversification, where all solution approaches the best solution. Figure 9 shows the improvised swoop process, whilst Fig. 9a shows the location of the mean. The best point is in the same location and the location of the point. Figure 9b shows the improvement of all points in the search space, obtains very near location from the best solution in points 1, 2, 5 and 6 and obtains the new best solution in point 6. Figure 9c shows all point to the new location, which is better compared with the previous location, and obtains the new best solution in point 5, which is better compared with the mean point. The three stages are critical in obtaining a good solution with minimum iteration, where each stage depends on two crucial

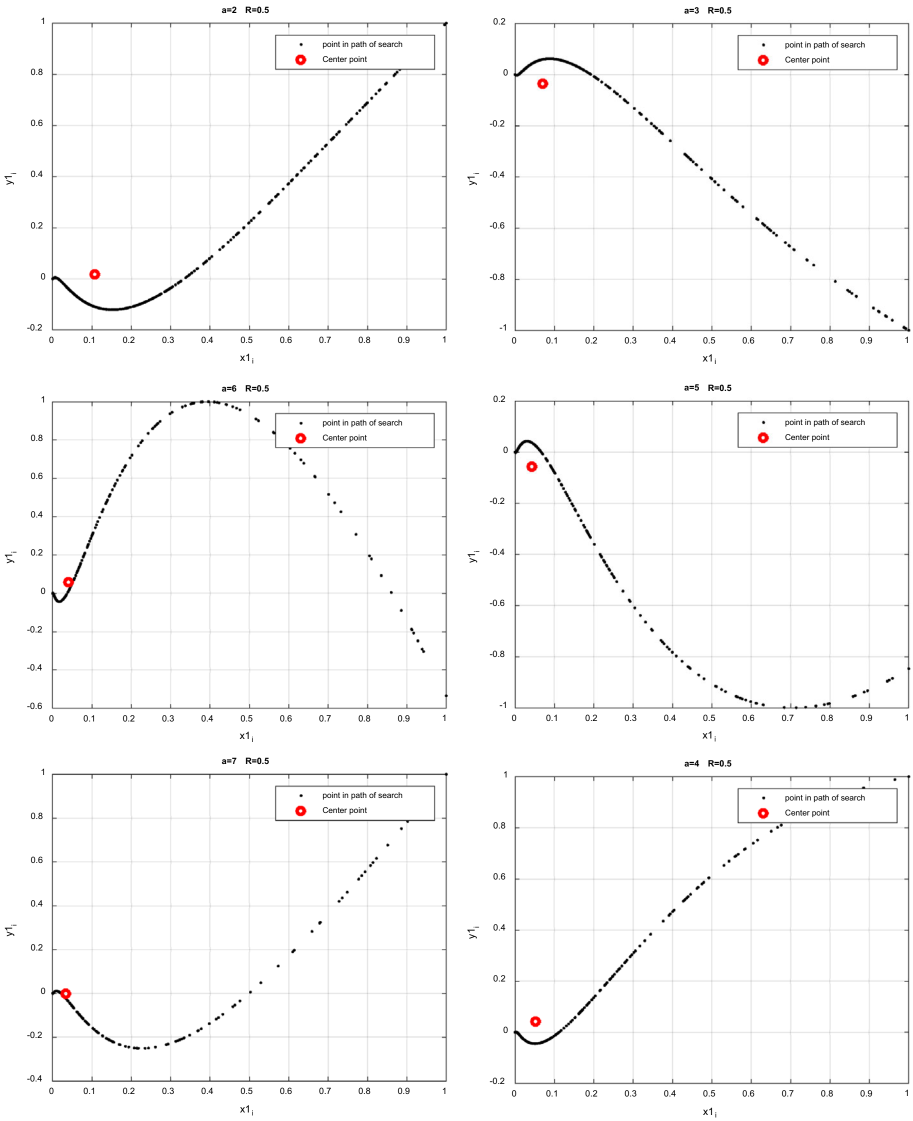

Fig. 7 Shape of swoop when parameter a is changed

characteristics, intensification and diversification. These characteristics are crucial to obtain a new solution continuously and search a round optimal solution (Fig. 8).

2.2.4 Complete BES algorithm

The preceding sections introduced the main components of BES, which include the selection, searching and swooping stages. To describe the remaining operations and facilitate the implementation of BES, the pseudo-code of its complete algorithm is provided in Algorithm 1. The initialisation procedure is first activated in lines 1–2 of Algorithm 1. Population P is

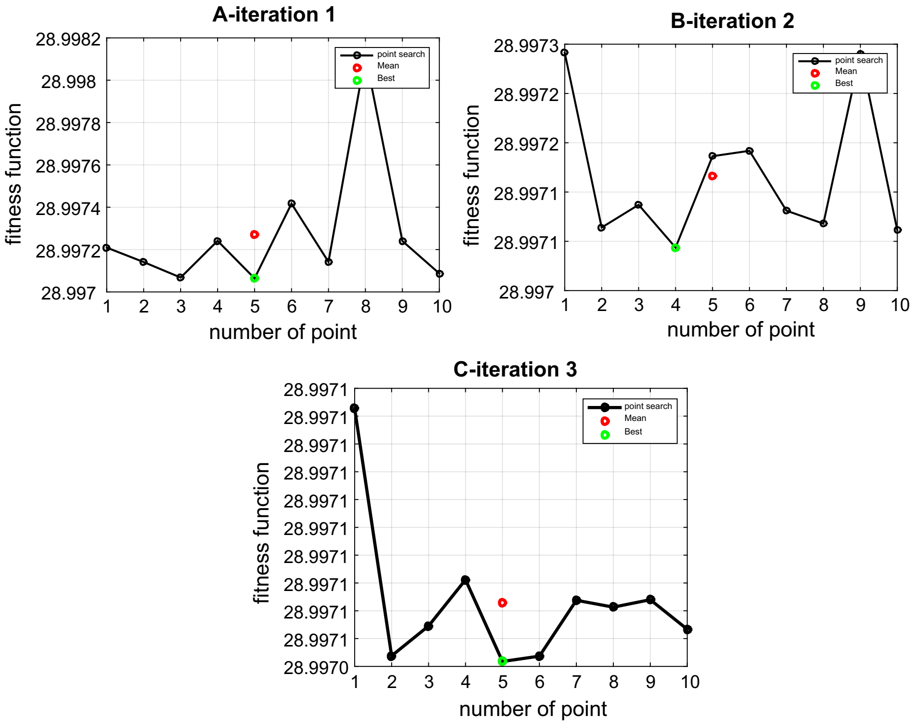

Fig. 8 Improvised swoop stage: **a** swoop stage after on alteration, **b** swoop stage after two alterations and **c** swoop stage after three alterations

initialised to be generated in the space of problems, whilst the iteration number t is set to 0. For each solution in P, the positional information is randomly generated. Thereafter, the objectives of each particle are evaluated. We execute the following steps for each solution in the population P: selection area for the searching around the best solution using lines 4–12, evaluate the new area, as well as the searching and selection areas by using spiral movement, where random number generated in two axes and two movements. The solution moves towards the next point and the central point. We evaluate the new position for hunting by using lines 13–21. Thereafter, the swoop stage begins by using the new position in the searching space to swoop towards the prey. The new solution is evaluated by using lines 22–30. The iteration counter k is increased by 2 in line 31, as three steps are run. The preceding evolutionary phase is repeated until the pre-set maximum number of iterations is achieved. Lastly, the final solutions in $\boldsymbol{P}$ are reported as the final population and the best solution obtained in the population for solving the problem.

Algorithm 1 Complete BES Algorithm

Randomly initialise Point **P**i for **n. Point**;
Calculate the fitness values of initial Point: f (**P**i);
WHILE (the termination conditions are not met)
Select space
For (each point i in the population)
$P_{new} = Pbest + \alpha * rand(P_{mean} - P_i)$
If $\mathbf{f(Pnew)} < f(Pi)$
$P_i = P_{new}$
If $\mathbf{f(Pnew)} < f(\mathbf{P_{best}})$
$P_{best} = P_{new}$
End If
End If
End For
Search in space
For (each point i in the population)
$P_{new} = P_i + y(i) * (P_i - P_{i+1}) + x(i) * (P_i - P_{mean})$
If $\mathbf{f(Pnew)} < f(Pi)$
$P_i = P_{new}$
If $\mathbf{f(Xnew)} < f(P_{best})$
$P_{best} = P_{new}$
End If
End If
End For
Swoop
For (each point i in the population)
$P_{new} = rand * P_{best} + x1(i) * (P_i - c1 * P_{mean}) + y1(i) * (P_i - c2 * P_{best})$
If $\mathbf{f(Pnew)} < f(Pi)$
$P_i = P_{new}$
If $\mathbf{f(Pnew)} < f(P_{best})$
$P_{best} = P_{new}$
End If
End If
End For
Set k:= k + 1;
END WHILE

3 Computational experiment

This section evaluates the performance of the proposed BES algorithm. Firstly, we describe the evaluation methodology and present the results of the experiments, which are conducted in different optimisation problems. Secondly, we compare the performance of the BES algorithm with that of other intelligent computational techniques. Thirdly, we discuss our findings in detail. The no free lunch (NFL) theorem indicates that 'for any algorithm, any elevated performance over one class of problems is exactly paid for in performance over another class' (Wolpert and Macready 1997). A particular meta-heuristic may yield promising results for a set of problems but may perform poorly on another set of problems. With NFL, this field of study is highly active. Consequently, the extant approaches are enhanced, whilst new meta-heuristics are proposed every year.

3.1 Experimental settings and comparative methods

Firstly, we test the performance of the proposed BES on 30 benchmark functions of the CEC 2014 Competition on Single Objective Real-Parameter Numerical Optimisation (Liang et al. 2013) and 25 benchmark functions of the CEC 2005, because those benchmark testing problems are most frequency used by other researchers in order to test thier strong points that covers the various types of function optimisation a single objective problems in most cases as shown in Tables 1 and 2. Detailed definitions of the functions can be found in Suganthan et al. (2005).

The performance of the BES algorithm is also evaluated using the CEC 2014 benchmark functions (Liang et al. 2013). The set of CEC 2014 benchmark functions consist of 30 suits THAT are classified into four categories, namely, unimodal, simple multimodal, hybrid and composition functions. Table 2 describes the search range and global optimum values of all benchmark functions.

On the test functions, we compare BES with six recent popular meta-heuristic methods:

- *Differential evolution (DE) algorithm* A basic variant of the DE algorithm works by having a population of candidate solutions (called agents). These agents are moved into the search space using simple mathematical formulas to combine the positions of existing agents in the population. If the new agent position is an improvement, then the position is accepted and is part of the population, otherwise the new position is simply discarded. The process is repeated and, in doing so, it is hoped, but not guaranteed, that a satisfactory solution will be discovered (Storn and Price 1997).
- *GWO algorithm* The GWO algorithm mimics the hierarchy of leadership and hunting mechanism of grey wolves in the wild as proposed by Mirjalili et al. (2014). Four types of grey wolf, namely, alpha, beta, delta and omega, are used to simulate the leadership hierarchy. Additionally, three main stages of hunting are implemented to perform an optimisation, namely, search, encroachment and attack of prey.
- *EPSO* A set of optimisation algorithms for particle swarms with self-adaptive mechanism is proposed by hybridising some PSO algorithms, called EPSO (Lynn and Suganthan 2017).
- *FDR-PSO* This algorithm has been proposed to solve the problem of premature convergence observed in PSOs. In comparison with PSO, FDR-PSO added a social learning component, drawing lessons from the neighbouring particle's (*nbest*) experiment. The neighbouring particles are selected on the basis of two criteria: (1) the particle must be near the particle being updated and (2) the particle must be better adapted compared with the particle being updated. Whether a neighbouring particle meets these criteria, the decision is made by the distance ratio fitness/distance one-dimensional called distance-fitness ratio (Peram et al. 2003).
- *CLPSO* In PSO, the trajectory towards the global optimum is adjusted by the *pbest* and *gbest* particles. Although *gbest* is the best experience of the population, this particle may be a lower local optimum for a multimodal problem and far from the global optimum. To solve this problem, CLPSO has been proposed in Liang et al. (2006). In CLPSO, the best experiments of all particles are used to guide the search for a particle.

Notably, fine-tuning the control parameters for each problem can improve the performance of the algorithm. However, finding separate parameter settings for each problem can take a long time. Such tuning processes can lead to an unfair comparison for each algorithm in evaluating the algorithm's overall performance over the entire test suite. Table 3 shows the recommended setting of the algorithm zones.

Table 1 CEC 2005 test functions

Benchmark functions	Initialisation range	Search range	$F(x^*)$
Unimodal functions	$[-100, 100]^D$	$[-100, 100]^D$	– 310
$F1$: Shifted sphere function	$[-100, 100]^D$	$[-100, 100]^D$	– 450
$F2$: Shifted Schwefel's problem 1.2	$[-100, 100]^D$	$[-100, 100]^D$	– 450
$F3$: Shifted rotated high conditioned elliptic function	$[-100, 100]^D$	$[-100, 100]^D$	– 450
$F4$: Shifted Schwefel's problem 1.2 with noise in fitness	$[-100, 100]^D$	$[-100, 100]^D$	– 450
$F5$: Schwefel's problem 2.6 with global optimum on bounds	$[-100, 100]^D$	$[-100, 100]^D$	– 310
Multimodal functions	$[-100, 100]^D$	$[-100, 100]^D$	– 310
$F6$: Shifted Rosenbrock's function	$[-100, 100]^D$	$[-100, 100]^D$	390
$F7$: Shifted rotated Griewank's function without bounds	$[-0, 600]^D$	$[-600, 600]^D$	– 180
$F8$: Shifted rotated Ackley's function with global optimum on bounds	$[-32, 32]^D$	$[-32, 32]^D$	– 140
$F9$: Shifted Rastrigin's function	$[-5, 5]^D$	$[-5, 5]^D$	– 330
$F10$: Shifted rotated Rastrigin's function	$[-5, 5]^D$	$[-5, 5]^D$	– 330
$F11$: Shifted rotated Weierstrass function	$[-0.5, 0.5]^D$	$[-0.5, 0.5]^D$	90
$F12$: Schwefel's problem 2.13	$[-100, 100]^D$	$[-100, 100]^D$	– 460
Expanded functions	$[-100, 100]^D$	$[-100, 100]^D$	– 310
$F13$: Expanded extended Griewank's plus Rosenbrock's function ($F8F2$)	$[-3, 1]^D$	$[-3, 1]^D$	– 130
$F14$: Shifted rotated expanded Scaffer's $F6$	$[-100, 100]^D$	$[-100, 100]^D$	– 300
Hybrid composition functions	$[-100, 100]^D$	$[-100, 100]^D$	– 310
$F15$: Hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	120
$F16$: Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	120
$F17$: Rotated hybrid composition function with noise in fitness	$[-5, 5]^D$	$[-5, 5]^D$	120
$F18$: Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	10
$F19$: Rotated hybrid composition function with a narrow basin for the global optimum	$[-5, 5]^D$	$[-5, 5]^D$	10
$F20$: Rotated hybrid composition function with the global optimum on the bounds	$[-5, 5]^D$	$[-5, 5]^D$	10
$F21$: Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	360
$F22$: Rotated hybrid composition function with high condition number matrix	$[-5, 5]^D$	$[-5, 5]^D$	360
$F23$: Non-continuous rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	360
$F24$: Rotated hybrid composition function	$[-5, 5]^D$	$[-5, 5]^D$	260
$F25$: Rotated hybrid composition function without bounds	$[-2, 5]^D$	$[-5, 5]^D$	260

In experiments, we use 30-D problems for test problems and set the maximum number of evaluations (NFE) at 100,000 for each problem algorithm to ensure a fair comparison. Each algorithm has been run 30 times (with different initial random values) on each test problem and the evaluation is based on the average performance over 60 runs.

Table 2 CEC 2014 test functions

Benchmark functions	Search range	$F(x^*)$
Unimodal functions	$[-100, 100]^D$	− 310
$F1$: Rotated high conditioned elliptic function	$[-100, 100]^D$	100
$F2$: Rotated Bent Cigar function	$[-100, 100]^D$	200
$F3$: Rotated discus function	$[-100, 100]^D$	300
Simple multimodal functions		
$F4$: Shifted and rotated Rosenbrock's function	$[-100, 100]^D$	400
$F5$: Shifted and rotated Ackley's function	$[-100, 100]^D$	500
$F6$: Shifted and rotated Weierstrass function	$[-100, 100]^D$	600
$F7$: Shifted and rotated Griewank's function	$[-100, 100]^D$	700
$F8$: Shifted Rastrigin's function	$[-100, 100]^D$	800
$F9$: Shifted and rotated Rastrigin's function	$[-100, 100]^D$	900
$F10$: Shifted Schwefel's function	$[-100, 100]^D$	1000
$F11$: Shifted and rotated Schwefel's function	$[-100, 100]^D$	1100
$F12$: Shifted and rotated Katsuura function	$[-100, 100]^D$	1200
$F13$: Shifted and rotated HappyCat function	$[-100, 100]^D$	1300
$F14$: Shifted and rotated HGBat function	$[-100, 100]^D$	1400
$F15$: Shifted and rotated expanded Griewank's plus Rosenbrock's function	$[-100, 100]^D$	1500
$F16$: Shifted and rotated expanded Scaffer's $F6$ function	$[-100, 100]^D$	1600
Hybrid functions		
$F17$: Hybrid function 1 ($N = 3$)	$[-100, 100]^D$	1700
$F18$: Hybrid function 2 ($N = 3$)	$[-100, 100]^D$	1800
$F19$: Hybrid function 3 ($N = 4$)	$[-100, 100]^D$	1900
$F20$: Hybrid function 4 ($N = 4$)	$[-100, 100]^D$	2000
$F21$: Hybrid function 5 ($N = 5$)	$[-100, 100]^D$	2100
$F22$: Hybrid function 6 ($N = 5$)	$[-100, 100]^D$	2200
Composition functions		
$F23$: Composition function 1 ($N = 5$)	$[-100, 100]^D$	2300
$F24$: Composition function 2 ($N = 3$)	$[-100, 100]^D$	2400
$F25$: Composition function 3 ($N = 3$)	$[-100, 100]^D$	2500
$F26$: Composition function 4 ($N = 5$)	$[-100, 100]^D$	2600
$F27$: Composition function 5 ($N = 5$)	$[-100, 100]^D$	2700
$F28$: Composition function 6 ($N = 5$)	$[-100, 100]^D$	2800
$F29$: Composition function 7 ($N = 3$)	$[-100, 100]^D$	2900
$F30$: Composition function 8 ($N = 3$)	$[-100, 100]^D$	3000

3.2 Evaluation procedure

The experimental results will be described on the basis of the mean, standard deviation (SD), best point and Wilcoxon signed-rank test statistic of the function values.

(a) **Mean** Mean (x) is computed as the sum of all the observed outcomes from the sample divided by the total number of these outcomes.

Table 3 Parameter settings for the algorithms used in the comparative study

Algorithm	Parameter	References
BES	$c1,\ c2,\ \alpha = 2,\ a = 10,\ R = 1.5\ and\ Pop_{size} = 100$	Our proposed
DE	$Pop_{size} = 100, Cr = 0.5, F \in [0.8, 1]$	Brest et al. (2006)
GWO	$Pop_{size} = 100$	Mirjalili et al. (2014)
CLPSO	$c = 3 - 1.5, \omega = 0.9 - 0.2$	Liang et al. (2006)
EPSO	–	Lynn and Suganthan (2017)
FDR-PSO	$c1 = 1,\ c2 = 1,\ c3 = 2,\ \omega = 0.9 - 0.2,\ \chi = 0.729$	Peram et al. (2003)

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

(b) **SD** SD is a measure that quantifies the variation or dispersion of a set of data for the function values.

$$SD = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

(c) **Best point** The best point reflects the minimum value.

(d) **Wilcoxon signed-rank test** The Wilcoxon signed-rank test statistic determines the difference between two samples (Derrac et al. 2011) and provides an alternative test of location that is affected by the magnitudes and signs of these differences. This test answers the following hypotheses:

$$H_0 : mean(A) = mean(B)$$
$$H_1 : mean(A) \neq mean(B),$$

where A and B denote the results of the first and second algorithms, respectively. This test also checks whether one algorithm outperforms the other. Let d_i denote the difference between the performance scores of two algorithms in solving *ith* out of *n* problems. Let R^+ denote the sum of ranks for the problems, in which the first algorithm outperforms the second. Lastly, let R^- represent the sum of ranks for the problems in which the second algorithm outperforms the first. The ranks of $d_i = 0$ are divided evenly amongst the sums. If these sums have an odd number, then one of them is disregarded.

$$R^+ = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i)$$
$$R^- = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i)$$

We use MATLAB to find the p value for comparing the algorithms at a significant level of $\alpha = 0.05$. The null hypothesis is rejected when the p-value is less tha n the significant level. R^+ represents a high mean algorithm that shows superiority over other algorithms across different sets of experiments. When $\left(R^+ = \frac{n\times(n+1)}{2}\right)$, this algorithm outperforms all algorithms across all experiments.

Table 4 Comparative results on the unimodal benchmark functions. CEC 2005 test functions

Problems	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
*F*1	Mean	2.54E−13	*30.04215*	0.001557	1619.29	3.57E−03	8456.677	1.59E+00
	STD	9.64E−14	*114.3291*	0.000639	1061.882	5.97E−03	2653.327	8.08E−01
	Best	1.14E−13	*5.68E−14*	0.000412	207.1187	0.000333	3314.996	3.49E−01
	Winner	/	−	+	+	+	+	+
F2	Mean	*3.58E−04*	1304.298	35,520.12	13,918.57	3062.083	48,987.67	5976.6
	STD	*5.58E−04*	1336.021	6697.749	3716.508	1143.627	9091.653	2046.091
	Best	*1.78E−06*	243.5314	24,076.23	7579.364	1512.356	33,066.63	2280.343
	Winner	/	+	+	+	+	+	+
F3	Mean	*424,534.6*	13,258,785	2.28E+08	29,447,491	8,035,917	2.38E+08	17,931,228
	STD	*177,530.9*	6,956,047	47,226,682	15,421,433	3,242,596	86,784,215	6,780,924
	Best	*179,757.8*	5,738,012	1.43E+08	7,104,698	2,122,823	97,026,471	7,599,379
	Winner	/	+	+	+	+	+	+
F4	Mean	*1.18E+03*	8.20E+03	49,747.03	20,851.57	16,095.22	57,566.8	13,210.99
	STD	*9.03E+02*	5.10E+03	9485.325	4757.41	5503.273	13,882.48	3154.892
	Best	*136.9326*	1505.053	25,632.61	8712.111	8160.798	31,702.48	6741.11
	Winner	/	+	+	+	+	+	+
F5	Mean	*3808.727*	4.26E+03	4.77E+03	6067.666	6531.804	19,950.64	4498.863
	STD	*751.3042*	1.35E+03	1.41E+03	2627.344	1490.606	1951.32	1018.822
	Best	*2.75E+03*	1598.197	1.31E+03	1538.454	4011.876	15,642.48	3343.646
	Winner	/	+	+	+	+	+	+

3.3 Experimental results

Tables 4, 5, 6, 7, 8, 9, 10 and 11 present the experimental results of the unimodal, multimodal, hybrid and expanded functions, where 'Mean' and 'Best' denote the mean and minimum values, respectively, of the algorithm amongst the 30 runs; 'STD' denote standard deviation and the index greater than the median indicates the rank of the algorithm in terms of mean values amongst the six algorithms; 'Winner' indicates the winner between the BES algorithm and other algorithms by using the Wilcoxon test and Superscript + denotes that BES has significant performance improvement over the comparative method and superscript − otherwise. The best results amongst the comparative algorithms of each problem are shown in italics.

3.4 Results and discussion

3.4.1 CEC 2005 benchmark functions

Table 4 shows the results of the unimodal functions. BES obtains the best result in functions f2–f5, obtains significant result compared with other algorithms and also determines the second best result after DE/best/1.

Table 5 shows the results of the seven multimodal functions. BES obtains the best results in four functions, obtains the second best mean values in one function (f10) and ranks third in f11 and fourth in f9, thereby exhibiting the best overall performance.

Table 5 Comparative results on multimodal benchmark functions. CEC 2005 test functions

Problem	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F6	Mean	*14.59163*	16,683,589	515.4254	49,760,768	764.0062	1.67E+09	4939.805
	STD	*11.77044*	53,306,445	397.5199	92,454,010	1141.437	6.86E+08	4982.12
	Best	*0.102552*	7.664112	58.39584	230,910.7	57.12508	5.69E+08	315.4345
	Winner	/	+	+	+	+	+	+
F7	Mean	*0.01779*	2.484019	1.093496	114.1146	1.10798	637.9848	1.190527
	STD	*0.020298*	7.603139	0.054107	83.74462	0.148083	121.9135	0.188011
	Best	*2.84E−13*	0.00838	1.034927	8.415945	0.656189	395.6984	1.030021
	Winner	/	+	+	+	+	+	+
F8	Mean	*21.01276*	21.05521	21.04873	21.06576	21.08927	21.10556	21.02348
	STD	*0.050082*	0.058582	0.076623	0.038447	0.060285	0.045758	0.059419
	Best	*20.8578*	20.92556	20.83246	20.99647	20.88136	21.00448	20.91516
	Winner	/	+	+	+	+	+	+
F9	Mean	96.36371	*54.28236*	141.0055	104.8878	66.74963	185.4411	69.79172
	STD	27.40986	*14.06197*	9.380712	24.73606	15.14341	16.58373	23.11518
	Best	48.75287	*34.14963*	119.3419	55.97502	21.44116	140.0474	35.54721
	Winner	/	–	+	+	–	+	–
F10	Mean	125.0188	223.6509	227.3427	180.078	*94.93207*	362.0407	172.7515
	STD	45.55562	30.74557	11.93888	62.57511	*36.32295*	22.08793	49.2869
	Best	49.74789	129.7594	200.8293	78.26293	*53.89494*	306.642	86.34158
	Winner	/	+	+	+	–	+	+
F11	Mean	26.72306	31.22171	42.42874	*19.45602*	27.36569	36.93599	22.55814
	STD	5.551898	10.05158	1.104764	*2.871001*	3.75895	1.831779	4.178208
	Best	17.85074	11.86539	39.76568	*15.20534*	19.26707	32.77143	13.65139
	Winner	/	+	+	–	+	+	–
F12	Mean	*6920.658*	33,813.73	361,261	88,617.99	25,464.79	452,552.4	36,160.32
	STD	*9400.036*	28,571.61	62,591.25	38,089.88	13,721.59	84,126.72	17,501.06
	Best	*2.71E+02*	4746.136	234,494.5	32,123.28	9063.297	263,658.2	9404.816
	Winner	/	+	+	+	+	+	+

Table 6 Comparative results on the expanded benchmark functions. CEC 2005 test functions

Problems	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F13	Mean	*8.487249*	11.92546	17.43139	7.166694	7.726158	32.99436	14.84338
	STD	*3.658165*	3.374358	1.007515	2.802968	3.215878	3.983259	2.542399
	Best	*3.11751*	3.432903	15.624	4.44821	4.25801	24.89331	8.778688
	Winner	/	+	+	+	+	+	+
F14	Mean	*12.71934*	13.53093	13.81469	12.67162	13.21311	13.68121	13.14486
	STD	*0.225599*	0.230809	0.136006	0.531075	0.314512	0.152146	0.342664
	Best	*12.21889*	12.99864	13.44302	11.5179	12.46994	13.3699	12.45767
	Winner	/	+	+	+	+	+	+

Table 7 Comparative results on hybrid benchmark functions. CEC 2005 test functions

problem	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F15	Mean	424.7033	390.9729	*315.3441*	484.9082	346.7444	702.3434	394.7499
	STD	92.4719	103.5547	*85.81975*	88.37767	105.5986	60.39995	159.2829
	Best	180.8602	137.6303	*203.7066*	357.4855	126.0998	596.7025	172.8625
	Winner	/	+	–	+	+	+	+
F16	Mean	*349.8725*	311.8774	280.1378	294.8007	245.7533	536.9206	267.6681
	STD	*147.3519*	102.7791	47.45666	166.0575	168.4909	46.23015	163.6782
	Best	*122.6517*	160.1561	228.3058	119.1686	82.59358	411.0915	57.17688
	Winner	/	+	+	+	+	+	+
F17	Mean	*261.1338*	349.543	323.9705	376.8497	259.0219	586.6951	388.1169
	STD	*159.2747*	93.30077	56.84546	158.7857	142.7435	74.82946	181.4352
	Best	*108.146*	232.7708	249.8718	122.1886	106.8439	447.2026	164.64
	Winner	/	+	+	+	+	+	+
F18	Mean	934.9513	924.1293	907.0517	956.9986	951.1687	1111.869	*903.0941*
	STD	36.28793	17.59137	0.854958	22.89824	21.96692	28.90185	*47.22523*
	Best	800	907.922	905.9079	915.7854	918.8487	1063.307	*800.0262*
	Winner	/	–	–	+	+	+	–
F19	Mean	935.1817	918.347	*907.2287*	957.7005	940.2793	1107.025	930.8864
	STD	50.65561	11.40584	*0.755932*	17.83637	32.39813	26.51699	11.59625
	Best	800	907.9179	*906.499*	920.6906	800.0085	1058	912.5285
	Winner	/	–	–	+	+	+	+
F20	Mean	945.9586	916.9416	*907.0598*	951.0267	932.5527	1103.514	921.248
	STD	25.71717	7.503387	*0.539631*	22.61306	39.79584	31.6637	24.46258
	Best	900	906.0478	*905.9592*	918.8813	800.0114	1025.034	800.0444
	Winner	/	–	–	+	+	+	–
F21	Mean	*731.4421*	618.5815	500.0009	908.2824	816.667	1228.001	646.8711
	STD	*333.38*	233.3088	0.000959	216.6567	344.443	26.19435	273.9703
	Best	*500*	500	500.0003	505.1438	500.0001	1149.816	500.0406
	Winner	/	+	+	+	+	+	+
F22	Mean	998.6506	957.0858	*926.9182*	1009.381	1054.05	1268.658	1015.344
	STD	38.43831	39.59547	*16.7434*	53.88538	40.4425	49.65976	29.3925
	Best	938.2246	895.7831	*888.3293*	919.5867	953.3575	1183.691	961.9779
	Winner	/	–	–	+	+	+	+
F23	Mean	839.5792	874.9123	*534.1654*	934.4245	711.6103	1227.433	624.3454
	STD	291.2889	176.9421	*0.000582*	182.645	266.3846	22.33277	187.0287
	Best	537.2922	602.8678	*534.1643*	572.3199	534.1753	1161.971	534.1643
	Winner	/	+	–	+	+	+	–
F24	Mean	*345.9346*	668.644	876.4201	774.5896	326.3784	1298.387	271.4754
	STD	*378.4717*	349.8999	235.7744	344.7307	337.9403	27.94609	270.15
	Best	*200*	200	200.0049	203.1733	200.0005	1214.368	200.0665
	Winner	/	+	+	+	+	+	+
F25	Mean	*339.939*	592.8892	941.6735	853.1318	271.2072	1304.863	271.8627
	STD	*363.0738*	332.4363	140.218	315.1444	270.9198	33.91557	271.3796
	Best	*200*	200	200.0153	298.119	200.0006	1245.324	200.0661
	Winner	/	+	+	+	+	+	+

Table 8 Comparative results on the unimodal benchmark functions. CEC 2014 test functions

problem	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F1	Mean	*3.66E+03*	3338818	1.11E+08	53,846,624	3.36E+05	33,781,186	3.23E+05
	STD	*3.61E+03*	6,714,714	26,076,081	43,004,178	2.17E+05	10,233,995	2.20E+05
	Best	*8.36E+01*	197,657.3	57,688,916	14,100,054	90,677.98	15,555,720	4.82E+04
	Winner		+	+	+	+	+	+
F2	Mean	*1.40E−10*	15.38533	609.6707	1.62E+09	120.5307	21,940.74	3117.584
	STD	*4.16E−10*	84.26893	1845.188	1.5E+09	182.3628	12,971.31	2588.538
	Best	*2.84E−13*	5.68E−13	1.11E+00	1.06E+08	0.067181	2700.568	73.36015
	Winner		+	+	+	+	+	+
F3	Mean	*2.57E−10*	0.134084	128.2374	37,426.69	56.18939	715.1103	1514.171
	STD	*4.68E−10*	0.284771	52.92267	9393.975	131.0564	714.7068	1786.227
	Best	*3.41E−13*	5.49E−07	50.197	18,879.99	0.013271	33.71222	35.39964
	Winner		+	+	+	+	+	+

- DE/best/1 ranks first in f9, EPSO ranks first in f10 and GWO obtains the best mean values in f11. BES ranks fourth and surpasses three other algorithms in f9, ranks second and surpasses five other algorithms in f10 and ranks third and surpasses four other algorithms in f11.
- DE/best/1, EPSO and FDR-PSO rank first, second and third, respectively, in f9, followed by BES, which is significantly different from (better than) DE/rand/1, GWO and CLPSO. f9, the Rastrigin function shifted and rotated, has a very large number of local optima, thereby making it difficult for the algorithms to obtain the global optimum in at least one executions.
- Of the remaining four benchmarks, BES constantly obtains the best results and its performance is extremely different from that of the other algorithms.

Particularly, BES results reach or substantially approximate the real optimum in such functions as f6 and f7, which is a narrow peak (or have an extremely narrow valley ranging from the local optimum to the global optimum).

Table 6 shows the results of the expanded functions. BES obtains the best result. In two functions, BES also obtains significant result compared with the other algorithms.

Table 7 shows the results of the 11 hybrid functions. BES occupies the first rank in five functions (i.e. f16, f1, f21, f24 and f25), third in function f22, fourth in functions f18, f19 and f23 and fifth in function f15. The relatively low performance of BES in f15 is partially consistent with those of the subfunctions, including f11, because hybrid functions also have many local optima. In summary, the overall performance of BES is the best amongst the six algorithms on the benchmark suite, including unimodal, multimodal, hybrid and expanded functions.

3.4.2 CEC 2014 benchmark functions

Table 8 shows the results of the unimodal functions. BES obtains the best result in all functions and also obtains significant result compared with other algorithms. Notably, numerous algorithms, such as GWO, work well with unimodal (Mirjalili et al. 2014) but have lost their

Table 9 Comparative results on simple multimodal benchmark functions. CEC 2014 test functions

Problems	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F4	Mean	*2.31E+00*	6.64E+01	76.9344	236.8712	40.74908	114.2538	38.29721
	STD	*1.19E+01*	3.90E+01	5.056233	69.24669	42.58184	23.00387	36.24049
	Best	*5.26E−06*	0.115627	72.5197	123.7805	0.001727	64.89047	0.000288
	Winner		+	+	+	+	+	+
F5	Mean	20.86845	2.10E+01	2.09E+01	21.00382	*20.39839*	20.43222	20.8303
	STD	0.061067	5.03E−02	6.02E−02	0.042132	*0.076816*	0.042371	0.199972
	Best	2.07E+01	20.811	2.08E+01	20.92822	*2.03E+01*	20.34403	20.24299
	Winner		+	+	+	–	–	+
F6	Mean	23.34192	11.85587	29.99691	14.13764	13.31339	16.17393	*6.521669*
	STD	3.047271	3.067448	5.070855	3.428804	3.080206	1.471535	*2.545097*
	Best	15.6302	6.432108	8.44128	8.706738	8.784592	12.7068	*2.78886*
	Winner		–	+	–	–	–	–
F7	Mean	0.019991	0.017199	*1.14E−12*	14.09249	0.016863	0.045864	0.011479
	STD	0.018088	0.01589	*1.97E−12*	16.17478	0.016947	0.021889	0.012675
	Best	1.25E−12	4.55E−13	*1.14E−13*	2.775252	2.27E−13	0.018348	4.55E−13
	Winner		+	–	+	+	+	+
F8	Mean	87.95414	43.28241	104.2255	77.20764	11.30973	*4.21E−05*	36.68078
	STD	23.85544	12.55066	8.683879	20.30793	3.09877	*2.03E−05*	8.782298
	Best	45.76804	26.86388	87.3676	50.17775	4.974795	*1.48E−05*	23.87901
	Winner		–	+	+	–	–	–
F9	Mean	107.5547	65.41736	190.9359	99.69832	*54.92164*	78.96062	60.80355
	STD	19.11751	25.15243	11.07124	30.89929	*20.53798*	8.471923	17.46669
	Best	76.61167	21.88909	169.1393	63.35056	*21.88909*	58.21998	34.82354
	Winner		–	+	+	–	–	–
F10	Mean	2395.764	1087.293	2899.698	2402.956	91.63457	*5.670689*	934.9101
	STD	670.425	348.3981	275.1467	594.0224	83.30696	*2.1848*	334.7705
	Best	1313.635	501.8853	2388.886	1077.178	13.08199	*1.28E+00*	255.8953
	Winner		–	+	+	–	–	–
F11	Mean	3098.091	4975.222	6862.322	3008.655	*2604.073*	2913.618	2802.815
	STD	492.9754	1824.373	314.5081	540.8616	*363.7578*	272.0616	681.6785
	Best	2165.823	1591.716	6075.617	1707.095	*1903.92*	2445.099	1406.739
	Winner		+	+	+	–	+	+
F12	Mean	1.319921	2.207899	2.323063	2.089164	*0.279357*	0.462208	0.793526
	STD	0.246955	0.277922	0.277316	1.179885	*0.137024*	0.067729	0.478633
	Best	8.25E−01	1.515601	1.558147	0.092044	*0.064829*	0.311574	0.261251
	Winner		+	+	+	–	–	–
F13	Mean	*0.36879*	0.463613	0.450412	0.592605	0.336463	0.307429	0.318912
	STD	*0.073468*	0.09945	0.066037	0.547487	0.079248	0.045056	0.083561
	Best	*0.213096*	0.266073	0.306414	0.232018	0.200541	0.206259	0.15523
	Winner		+	+	+	+	+	+
F14	Mean	*0.265301*	0.525637	0.444774	2.984409	0.265005	0.271494	0.259482

Table 9 continued

Problems	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
	STD	*0.115124*	0.271945	0.189488	4.944169	0.03983	0.037788	0.045261
	Best	*0.154129*	0.216997	0.230968	0.196436	0.194277	0.18306	0.15637
	Winner		+	+	+	+	+	+
F15	Mean	16.10439	15.40795	17.05153	138.0471	5.984547	12.0924	*4.85754*
	STD	6.990639	5.143547	1.300929	369.2851	2.082659	1.740555	*1.193944*
	Best	6.777028	3.313125	13.59799	9.621125	2.511469	8.559943	*2.669869*
	Winner		+	+	+	–	+	–
F16	Mean	*10.82171*	12.07852	12.87917	11.24053	10.59807	10.60224	10.84561
	STD	*0.603008*	0.327701	0.232058	0.63931	0.526349	0.459981	0.497321
	Best	*9.725559*	11.38094	12.2683	9.325099	9.71136	9.602978	9.717548
	Winner		+	+	+	+	+	+

Table 10 Comparative results on the hybrid benchmark functions. CEC 2014 test functions

problem	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F17	Mean	*2210.814*	44,3994.9	3,319,317	2,271,602	78,044.64	2,835,460	86,263.1
	STD	*1637.401*	407,856	1,279,977	2,576,033	49,587.78	1,270,462	63,505.04
	Best	*675.6954*	32,406	1,126,824	143,404.6	19,750.75	868,162.6	8326.808
	Winner		+	+	+	+	+	+
F18	Mean	*2608.092*	9617.067	45,769.54	7,642,787	2382.131	710.7192	1750.337
	STD	*2524.524*	9985.268	91,579.42	17,223,279	2814.034	495.2381	1642.422
	Best	*150.4415*	293.7062	505.7654	1533.875	108.5105	170.8821	123.5649
	Winner		+	+	+	+	+	+
F19	Mean	15.21444	8.939514	7.619889	46.21017	8.507054	11.80276	*6.943834*
	STD	18.90164	2.037473	0.858237	26.781	1.925868	1.554026	*1.832491*
	Best	5.489727	4.813866	5.849286	12.10587	5.598836	9.027436	*4.223448*
	Winner		+	–	+	+	+	–
F20	Mean	*220.9628*	2706.475	7127.191	19,733.45	1365.607	6641.315	6409.78
	STD	*191.7257*	6259.003	3720.62	11,521.12	1838.436	3475.523	4156.745
	Best	*96.52486*	458.1002	2358.113	4625.688	202.7854	972.6185	647.059
	Winner		+	+	+	+	+	+
F21	Mean	*1550.811*	74,935.51	640,503.3	1,062,936	43,146.02	375,699.8	58,747.33
	STD	*839.2444*	59,608.75	284,054.9	2,095,392	24,076.9	192,067.8	38,984.23
	Best	*581.2392*	8412.206	254,961.8	70,085.64	1756.285	87,221.12	11,776.67
	Winner		+	+	+	+	+	+
F22	Mean	*333.7552*	386.9185	305.1935	448.7551	353.3228	300.3243	333.1851
	STD	*152.6302*	187.7473	85.56478	164.2229	116.4933	97.55436	136.5961
	Best	*27.01889*	35.35308	149.3847	159.8945	148.1649	160.0519	140.1542
	Winner		+	+	+	+	+	+

Table 11 Comparative results on the composition benchmark functions. CEC 2014 test functions

problem	Statistics	BES	DE/best/1	DE/rand/1	GWO	EPSO	CLPSO	FDR-PSO
F23	Mean	*284.5123*	315.5549	315.2441	334.6579	315.2441	315.4332	315.2441
	STD	*51.83408*	7.62E−01	4.37E−10	8.359714	1.40E−12	0.124629	5.44E−12
	Best	*200*	315.2441	315.2441	322.1923	315.2441	315.2999	315.2441
	Winner		+	+	+	+	+	+
F24	Mean	*200*	244.2448	224.7383	200.0082	229.3168	227.0902	225.3499
	STD	*9.41E−07*	5.902141	2.140124	0.003444	4.955529	0.842493	2.695402
	Best	*200*	226.8997	222.4876	200.0035	224.2898	225.4053	222.0449
	Winner		+	+	+	+	+	+
F25	Mean	*200*	205.8454	224.8012	212.422	212.2191	209.9482	208.5969
	STD	*0*	2.187855	4.769287	2.227545	2.680017	1.265677	2.509102
	Best	*200*	202.8076	212.7112	207.9581	203.8049	207.7148	203.7192
	Winner		+	+	+	+	+	+
F26	Mean	*156.8014*	130.0926	100.4594	113.7454	107.0267	100.3773	150.2324
	STD	*50.24401*	67.68126	0.050006	34.43978	25.28844	0.078374	50.66672
	Best	*100.2133*	100.3335	100.331	100.2734	100.2174	100.1795	100.2125
	Winner		+	+	+	+	+	+
F27	Mean	811.2298	599.9518	382.1877	678.0352	*410.1491*	425.2634	493.4649
	STD	208.9184	151.1121	84.46454	133.7809	*33.09232*	11.16921	85.47025
	Best	401.2241	400.8831	300.741	416.6879	*400.8492*	410.5939	369.72
	Winner		–	–	–	–	–	–
F28	Mean	1503.785	1154.765	*845.7761*	1116.426	1087.282	1004.397	1617.405
	STD	326.2962	240.2191	*29.49447*	221.6198	140.1024	66.43646	752.626
	Best	1006.089	883.819	*789.5174*	862.4582	866.1551	885.9148	878.4516
	Winner		–	–	–	–	–	+
F29	Mean	*289,353.1*	308,356.8	283,097.4	872,829.6	1139.867	4975.601	1351.865
	STD	*1,578,238*	1,623,985	1,533,502	2,300,167	342.8063	2873.029	375.1286
	Best	*738.8932*	1119.632	1322.552	6892.181	672.4576	1648.231	806.1212
	Winner		+	+	+	+	+	+
F30	Mean	*2376.079*	5426.9	3572.649	52,771.45	2398.848	9028.825	2339.541
	STD	*881.5484*	5424.438	762.7805	35,428	601.3887	3101.532	553.7704
	Best	*1069.026*	1217.581	1768.253	8317.771	1100.716	3204.835	1260.93
	Winner		+	+	+	+	+	+

performance on these functions. BES can be effective for solving these functions compared with other algorithms.

Table 9 shows the results of the 13 multimodal functions. BES obtains the best results in four functions (f4, f13, f14 and f16). Additionally, EPSO obtains the best results in four functions (f5, f9, f11 and f12), CLPSO obtains the best results in two function (f8 and f10), FDR-PSO obtains the best results in two functions (f6 and f15) and DE/rand/1 obtains the best result in f7. The results of BES were poor in these functions. The reason is that the number of localisation areas is extremely large, which makes approaching global optimisation difficult compared with other functions.

Table 10 shows the results of the six hybrid functions. BES obtains the best result in five functions (i.e. f17, f18, f20, f21 and f22). The statistical tests show that BES performance is significantly different from the other five algorithms. Note that in this group of hybrid functions, the variables are randomly divided into subcomponents, whilst the different basic functions are used for different subcomponents, thereby resulting in a significant reduction in the performance of algorithms (e.g. GWO and DE) but the performance of BES remains as competitive as the basic features.

Table 11 shows the results of the eight composition functions. BES obtains the first rank in six functions (i.e. f23, f24, f25, f26, f29 and f30). The relatively low performance of BES in f27 and f28 is partially consistent with those of the subfunctions, including f9, f6 and f11, given that compositional functions have numerous local optima.

In summary, the overall performance of BES is the best amongst the six comparative algorithms of the benchmark suite, including the two sessions of CEC 2005 and CEC 2014. On some test functions with many local optima, the performance of BES is not very satisfactory.

We mainly used a linearly reduced BES population size in our experiments. In later iterations, the number of solutions was reduced to a single digit. Thus, veering away from the local optimum is difficult. We also tested the use of BES. The relatively large population size fixed in BES can effectively improve the performance for this test function but loses performance in many other test functions. Generally, the strategies to reduce population size facilitate the improvement of the overall performance of BES but an effective algorithm for certain problems. However, we must compare these two strategies and choose the best strategy for the majority of real optimisation problems.

Amongst the other five algorithms, BES showed the best performance throughout the suite. However, in all test functions, all algorithms are not consistently better compared with the others. Each algorithm achieves the best result in some functions. BES ranks first amongst the 33 functions. DE/best/1, DE/rand/1, GWO, EPSO, CLPSO and FDR-PSO immediately complete 2, 7, 1, 6, 2 and 4 functions, respectively. Each algorithm shows advantages and disadvantages of its benchmark suite, which we consider to be the same for various real-world problems. Therefore, when choosing an EA for a new optimisation problem, using terms to describe and quantify the boundaries of effective algorithm performance is important. We solve the characteristics of problem instances by using objective measurements and tools.

Notably, BES performance is not considerably competitive compared with the top ranked algorithms in the CEC 2005 and CEC 2014 competitions. The majority of these algorithms use complex search mechanisms, such as blending operators, history memory, replacement strategies and super heuristic controllers, as well as fine-tuning settings for the test suite. However, our goal is simply to test the performance of BES on a test suite by using simple frameworks and parameters. We expect that BES will also significantly improve its performance by introducing more complex mechanisms and by combining powerful operators with other heuristics. Figures 9 and 10 show the overall performance of BES compared with other algorithms. Accordingly, we can observe the superiority of BES amongst the six algorithms.

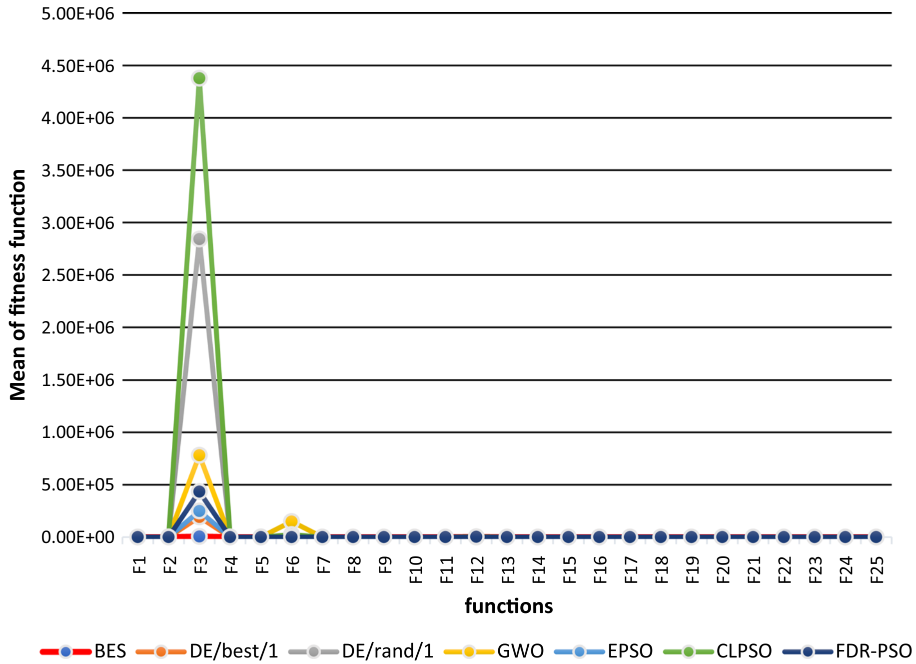

Fig. 9 Comparison amongst the algorithms by using CEC 2005

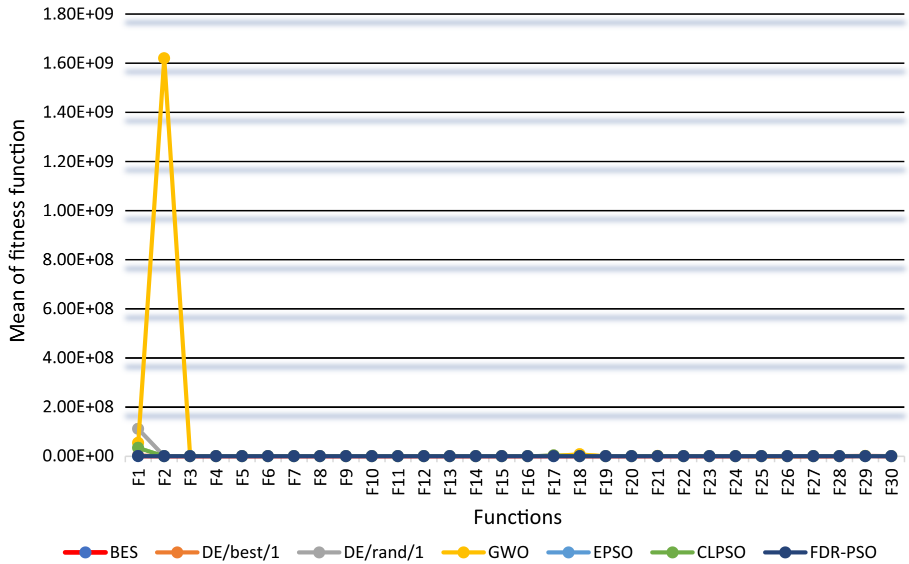

Fig. 10 Comparison amongst the algorithms by using CEC 2014

4 Conclusion

This study proposed a novel optimisation algorithm that mimics the hunting strategy, social hierarchy and behaviour of bald eagles. The optimisation results and discussion confirm that the BES algorithm is the best competitor amongst the six comparative algorithms of the benchmark suite in two sessions of CEC 2005 and CEC 2014. These algorithms include GWO, DE/best/1, DE/rand/1, EPSO, FDR-PSO and CLPSO. On some test functions with numerous local optima, the performance of BES is not very satisfactory because we used a linearly reduced BES population size in our experiments. In later iterations, the number of solutions was reduced to a single digit. Thus, breaking away from local optimum was difficult. We also tested the use of BES. The relatively large population size fixed in BES can effectively improve the performance for this test function but loses many other test functions. Generallly, strategies to reduce population size help improve the overall performance of BES, but an effective algorithm for certain problems. However, we must compare these two strategies and choose the best strategy for the majority of the real optimisation problems. Amongst the other five comparison algorithms, BES showed the best performance throughout the suite. However, in all test functions, all algorithms are not consistently better compared with the others. In fact, each algorithm achieves the best result on some functions. BES ranks first in 33 functions. DE/best/1, DE/rand/1, GWO, EPSO, CLPSO and FDR-PSO quickly complete 2, 7, 1, 6, 2 and 4 functions, respectively. Each algorithm shows the advantages and disadvantages of its benchmark suite, which we consider to be the same for various real-world problems. Therefore, when choosing an EA for a new optimisation problem, using terms to describe and quantify the boundaries of effective algorithm performance is important. We solve the characteristics of problem instances by using objective measurements and tools. Notably, BES performance is not very competitive compared with the top-ranked algorithms in the CEC 2005 and CEC 2014 competitions. The majority of these algorithms use complex search mechanisms, such as blending operators, history memory, replacement strategies and super heuristic controllers, as well as fine-tuning settings for the test suite. However, our goal is simply to test the performance of BES on a test suite by using simple frameworks and parameters. We expect that BES will also significantly improve its performance by introducing more complex mechanisms and by combining powerful operators with other heuristics. Future studies may examine the potential of using the prey identification process of bald eagles to minimise energy consumption by taking advantage of several factors, such as wind and gravity, amongst others. Moreover, some searching patterns, such as cross-based ones, may outperform others in certain computational environments.

References

Barhen J, Protopopescu V, Reister D (1997) Trust: a deterministic algorithm for global optimization. Science 276:1094–1097

Birge B (2003) PSOt—a particle swarm optimization toolbox for use with MATLAB. In: Proceedings of 2003 IEEE swarm intelligence symposium, pp 182–186

Brest J, Greiner S, Boškovic B, Mernik M, Žumer V (2006) Self- adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 10(6):646–657

Del Valle Y et al (2008) Particle swarm optimization: basic concepts, variants and applications in power systems. IEEE Trans Evol Comput 12(2):171–195

Derrac J et al (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18

Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Magaz 1:28–39

Fiacco AV, McCormick GP (1968) Nonlinear programming: sequential unconstrained minimization techniques. Wiley, New York

Fogel DB (1995) Evolutionary computation: toward a new philosophy of machine intelligence. IEEE Press, New York

Fogel D (2009) Artificial intelligence through simulated evolution. Wiley, Hoboken

Fogel LJ, Owens AJ, Walsh MJ (1965) Artificial intelligence through a simulation of evolution. In: Proceedings of 2nd cybernetics science symposium on biophysics cybernetics systems. Spartan Books, Washington, pp 131–155
Hansen AJ (1986) Fighting behavior in bald eagles: a test of game theory ecological society of America. Ecology 67(3):787–797
Hansen AJ, Boeker EL, Hodges JI, Cline DR (1984) Bald eagles of the Chilkat Valley, Alaska: ecology, behavior, and management. National Audubon Society and U.S. Fish & Wildlife, Service, Juneau
Hatamlou A (2012) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175
He S, Wu QH, Saunders JR (2006) A novel group search optimizer inspired by animal behavioural ecology. In: IEEE congress on evolutionary computation, 2006. CEC 2006. IEEE
He S, Wu QH, Saunders JR (2009) Group search optimizer: an optimization algorithm inspired by animal searching behavior. IEEE Trans Evol Comput 13(5):973–990
Holland JH (1992) Genetic algorithms. Sci Am 267:66–72
Houck C, Joines J, Kay M (1995) A genetic algorithm for function optimization: a MATLAB implementation. North Carolina State University, Raleigh, NC, Technical report NCSU-IE-TR-95-09
Joines J, Houck C (1994) On the use of nonstationary penalty functions to solve nonlinear constrained optimization problems with GAs. In: Proceedings of IEEE international conference on evolutionary computation. IEEE, Piscataway, NJ, pp 579–584
Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
Kazarlis S, Petridis V (1998) Varying fitness functions in genetic algorithms: studying the rate of increase in the dynamic penalty terms. In: Parallel problem solving from nature. Lecture notes in computer science. Springer, Berlin, vol 1498, pp 211–220
Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, pp 1942–1948
Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge
Lasserre JB (2001) Global optimization with polynomials and the problem of moments. SIAM J Optim 11(3):796–817
Li X (2003) A new intelligent optimization-artificial fish swarm algorithm. Doctor thesis, Zhejiang University of Zhejiang, China
Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10:281–295
Liang JJ, Qu BY, Suganthan PN (2013) Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and technical report, Nanyang Technological University, Singapore
Lynn N, Suganthan PN (2017) Ensemble particle swarm optimizer. Appl Soft Comput 55:533–548
Michalewicz Z, Attia N (1994) Evolutionary optimization of constrained problems. In: Sebald AV, Fogel LJ (eds) Proceedings of the 3rd annual conference on evolutionary programming. World Scientific, River Edge, pp 98–108
Mirjalili S (2015a) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249
Mirjalili S (2015b) The ant lion optimizer. Adv Eng Softw 83:80–98
Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
Peram T, Veeramachaneni K, Mohan CK (2003) Fitness-distance-ratio based particle swarm optimization. In: Proceedings of the 2003 swarm intelligence symposium, 2003. SIS '03. IEEE, pp 174–181
Qu B-Y, Liang JJ, Suganthan PN (2012) Niching particle swarm optimization with local search for multi-modal optimization. Inf Sci 197:131–143
Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
Rechenberg I (1994) Evolution strategy. In: Zurada M, Marks RJ, Robinson CJ (Eds) Computational intelligence: lmitating life, vol 1. IEEE Press, Piscataway, NJ, pp 147–159
Sameer FO, Abu Bakar MR, Zaidan AA, Zaidan BB (2019) A new algorithm of modified binary particle swarm optimization based on the Gustafson–Kessel for credit risk assessment. Neural Comput Appl 31(2):337–346
Schwefel H-P (1995) Evolution and optimum seeking. Wiley, New York
Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: Proceedings of IEEE international conference on evolutionary computation, 1998, pp 69–73

Sörensen K (2015) Metaheuristics—the metaphor exposed. Int Trans Oper Res 22:3–18
Stalmaster MV (1987) The bald eagle. Universe Books, New York
Stalmaster MV, Gessaman JA (1982) Food consumption and energy requirements of captive Bald Eagles. J Wildl Manag 46(3):646–654
Stalmaster PA, Kaiser KH (1997) Winter ecology of bald eagles on the Nisqually River Drainage, Washington. Northwest Sci 71:214–223
Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11:341–359
Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report, 2005005
Tariq I, AlSattar HA, Zaidan AA, Zaidan BB, Abu Bakar MR, Mohammed RT, Albahri OS, Alsalem MA, Albahri AS (2018) MOGSABAT: a metaheuristic hybrid algorithm for solving multi-objective optimisation problems. Neural Comput Appl 30(2):1–15
Todd CS, Young LS, Owen RB Jr, Gramlich FJ (1982) Food habits of bald eagles in Maine. J Wildl Manag 46:636–645
Whitley D (2001) An overview of evolutionary algorithms: practical issues and common pitfalls. Inf Softw Technol 43(14):817–831
Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evolut Comput 1:67–82
Yao X, Liu Y (1997) Fast evolution strategies. In: Proceedings of evolutionary programming VI. Springer, Berlin, pp 151–161
Yao X, Liu Y, Lin G (1999a) Evolutionary programming made faster. IEEE Trans Evolut Comput 3:82–102
Yao X, Liu Y, Liu G (1999b) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102
Zaidan AA, Kalaf BA, Abu Bakar MR, Zaidan BB (2017) A new hybrid algorithm of simulated annealing and simplex downhill for solving multiple-objective aggregate production planning on fuzzy environment. Neural Comput Appl 28(8):1–12