Fractional Lévy flight bat algorithm for global optimisation

Article *in* International Journal of Bio-Inspired Computation · January 2020

DOI: 10.1504/IJBIC.2020.106441

3 authors, including:

Diego Oliva
University of Guadalajara, CUCEI

Fractional Lévy flight bat algorithm for global optimisation

Redouane Boudjemaa*

Department of Mathematics,
Faculty of Sciences,
University M'Hamed Bougara of Boumerdes,
35000 Boumerdes, Algeria
Email: rboudjemaa@univ-boumerdes.dz
*Corresponding author

Diego Oliva

Departamento de Ciencias Computacionales,
Universidad de Guadalajara, CUCEI,
Av. Revolucion 1500, Guadalajara, Jal, Mexico
Email: diego.oliva@cucei.udg.mx

Fatima Ouaar

Department of Mathematics,
Faculty of Sciences,
University Mohamed Kheider of Biskra,
07000 Biskra, Algeria
Email: fatimaouaar@yahoo.fr

Abstract: A well-known metaheuristic is the bat algorithm (BA), which consists of an iterative learning process inspired by bats echolocation behaviour in searching for prays. Basically, the BA uses a predefined number of bats that collectively move on the search space to find the global optimum. This article proposes the fractional Lévy flight bat algorithm (FLFBA), which is an improved version of the classical BA. In the FLFBA the velocity is updated through fractional calculus and a local search procedure that uses a random walk based on Lévy distribution. Such modifications enhance the ability of the algorithm to escape from local optimal values. The FLFBA has been tested using several well-known benchmark functions and its convergence is also compared with other evolutionary algorithms from the state-of-the-art. The results indicate that the FLFBA provided in several cases better performance in comparison to the selected evolutionary algorithms.

Keywords: fractional calculus; bat algorithm; Lévy flight; non-parametric statistical tests.

Reference to this paper should be made as follows: Boudjemaa, R., Oliva, D. and Ouaar, F. (2020) 'Fractional Lévy flight bat algorithm for global optimisation', *Int. J. Bio-Inspired Computation*, Vol. 15, No. 2, pp.100–112.

Biographical notes: Redouane Boudjemaa received his BSc degree in Computing Mathematics at the University of Hertfordshire and a PhD in Applied Mathematics at the University of Leeds. He is an Associate Professor at the Department of Mathematics, University of M'Hamed Bougara of Boumerdes. His research interests include optimisation, metaheuristics and numerical methods. He is author of articles in journals, conference proceedings as well as book chapters.

Diego Oliva received his BS degree in Electronics and Computer Engineering from the Industrial Technical Education Center (CETI) of Guadalajara, Mexico in 2007, MSc degree in Electronic Engineering and Computer Sciences from the University of Guadalajara, Mexico in 2010. He obtained his PhD in Informatics in 2015 from the Universidad Complutense de Madrid. Currently, he is an Associate Professor at the University of Guadalajara in Mexico. His research interest includes evolutionary and swarm algorithms, hybridisation of evolutionary and swarm algorithms, and computational intelligence.

Fatima Ouaar received an Engineering degree in Statistics and a Magister diploma in Applied Mathematics at University Mohamed Kheider of Biskra. She is currently pursuing her PhD in Mathematics at the University Mohamed KHEIDER of Biskra. Her research interests include numerical analysis, applied mathematics, meta-heuristics and bio-inspired techniques.

1 Introduction

Different problems of science and engineering require accurate solutions that in most of the cases are computationally expensive. Optimisation algorithms are then used to overcome this situation but traditional optimisation techniques, including heuristic approaches that still being insufficient. In this context, nature is considered as a huge and vast source of inspiration for solving a variety of complicated problems, it always finds the optimal solution using different balanced mechanisms. This is the main motivation for using nature inspired computational methods. The techniques that mimic or use some natural behaviours to solve complex optimisation problems are part of the metaheuristic algorithms (MA).

MA's perform a stochastic search of the best solution of a specific problem; the main idea of these methods is the collective behaviour that exists between the candidate solutions, which generates a simple procedure to solve an optimisation problem.

The bat algorithm (BA) was introduced in 2010 as an alternative method for numerical optimisation (Yang, 2010). The BA is based on the mechanism of echolocation in bats, it consists of the use of a sonar to guide the bats during the flight. This behaviour also helps bats in hunting, using the echolocation they can identify the preys in the dark. The operators of the BA have a good balance between exploration and exploitation that is desirable for a MA. However, it has been proved that the performance of BA is good only in problems with a reduced number of dimensions (Yilmaz and Küçüksille, 2015; Fister et al., 2013). In this sense, different modifications have been proposed for improving the performance of BA. A recent work by Wang et al. (2019) introduced multiple strategies coupling to BA and the performance of their algorithm was evaluated using the Wilcoxon and Friedman tests. To improve the global search ability of BA with large-scale problems, Cui et al. (2018) proposed two new variants using principal component analysis. Another work by Xie et al. (2013) proposed a BA using DE operators and Lévy Flights during the optimisation process. In 2017 it was published a directional BA (Chakri et al., 2017), in which is proposed directional echolocation to improve the exploration of BA. Another interesting improvement was proposed in Fister et al. (2013) where the BA is hybridised with DE. The standard BA has also been modified using chaotic maps instead of normal distribution to increase the search capabilities (Jordehi, 2015; Gandomi and Yang, 2014). In the work of Cai et al. (2016), the local search of BA was improved by an optimal forage strategy while a random disturbance strategy was employed to extend the search pattern globally. There has also been introduced a modification of BA that considers the GA and the invasive weed optimisation (IWO) (Yilmaz and Küçüksille, 2015).

On the other hand, the fractional calculus (FC) is a mathematical tool commonly used in engineering and applied sciences (Oldham and Spanier, 1974; Sabatier et al., 2007). FC is an extension of classical mathematics it has been applied in fields like electronics, signal processing, fractals, and chaos to mention some (Edelman, 2010; Rabei et al., 2009; Machado, 2002). In this context, the FC is an excellent alternative to introduce concepts as memory (fractional derivative) in different processes; such feature generates more realistic models that integer-based models (Couceiro and Sivasundaram, 2016). Besides, the Lévy Flights (LF) have been extensively used to improve different MA (Yang and Deb, 2009; Bhateja et al., 2015). LF can be defined as random walks whose step lengths are not constant, and the values are selected from a probability distribution (Viswanathan et al., 1996). As indicated by Zhou et al. (2015) the Brownian walk and Lévy flight strategies ignore the learning process of the visited solutions during the waiting time between the successive movement steps. It is considered one of the disadvantages of this approach.

This article presents a hybrid version of the BA that improves its performance in global optimisation. The first contribution is the introduction of the fractional Lévy flights (FLF) that use the fractional calculus to avoid the drawbacks of standard LF. The modifed algorithm proposed in this paper combines the FLF and the DE to to improve the performance of the standard BA, this is the main contribution. The proposed algorithm is called fractional Lévy flight bat algorithm (FLFBA).

The remaining sections are organised as follows: in Section 2, it analysed the related work. Section 3 presents the proposed FLFBA. In Section 4, it described the experimental results of FLFBA compared to the ones obtained by cuckoo search algorithm (CS) (Yang, 2014), fractional-order Darwinian particle swarm optimisation (FDPSO) (Couceiro et al., 2012), moth-flame optimisation algorithm (MFO) (Mirjalili, 2015), ant colony optimisation algorithm (ACO) (Dorigo et al., 1996), shuffled frog-leaping algorithm (SFLA) (Eusuff et al., 2006) and a novel bat algorithm with habitat selection and Doppler effects (NBA) (Meng et al., 2015). Statistical analysis of the results obtained by the four algorithms is provided in Section 5. Finally, Section 6 discusses the conclusions and recommendations.

2 Related work

2.1 Basic bat algorithm

The bat-inspired algorithm, which mimics the echolocation navigation system in detecting and pursuing their preys, was firstly proposed by Yang (2010). By emitting loud sound pulses, the echoes that bounce back from different surrounding objects help bats identify not only their size but also their exact distances when flying in darkness. Microbats emits from 10 to 20 ultrasonic sound bursts a second with constant frequency (25 KHz to 150 KHz) but as they get closer to their preys they are increased to up to 200 pulses per second (Yang, 2014). Emitted pulses are as loud as 110 dB but as they get closer to their preys they become quieter.

The algorithm is based on the following three idealised rules (Yang, 2014):

1 Bats use echolocation to measure distance as well as differentiate between food/prey and background barriers.

2 Bats fly randomly with velocity v_i from position x_i using a frequency f_{min}, varying wavelength λ and loudness A_0 to search for prey. Based on their proximity to target, bats can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$.

3 Although the loudness can vary in many ways, it is assumed that loudness varies from a large predefined (positive) value A_0 to a minimum constant value A_{min}.

2.1.1 Initialisation

A randomly distributed population of N virtual bats is firstly generated. Such initial positions are produced in D-dimensional bounded search space as follows:

$$x_{ij}^0 = x_{min} + (x_{max} - x_{min}) * rand \quad (1)$$

where $i \in [1, \cdots, N]$, $j \in [1, \cdots, D]$ and $rand$ is a random vector with uniformly distributed elements generated in the range $[0, 1]$. The vectors x_{max} and x_{min} contain the upper and lower boundaries in each dimension j, respectively. The initial velocities v_i^0 are generally set to zero.

2.1.2 Generation of new solutions

Bats navigate by selecting new directions to optimal solutions through the combination of their own and other bats best experience. At each iteration t, a new solutions x_i^{t+1} and velocities v_i^{t+1} are updated as follows:

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \quad (2)$$

$$v_i^{t+1} = v_i^t + (x_i^t - x^g) f_i, \quad (3)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \quad (4)$$

where $\beta \in [0, 1]$ is uniformly distributed random vector and f_{min}, f_{max} are the minimum/ maximum frequency of emitted pulse by the bats. The value x^g represents the best global location found so far which is obtained by comparing all the solutions of all N bats at iteration t.

2.1.3 Local search

After new solutions are generated, it is used a local search that is based on random walks. It considers an i^{th} bat on the condition that its pulse emission rate r_i is smaller than a random number. The old position x_{old} is modified to obtain a new position x_{new} by,

$$if\ (rand > r_i)\ then,\ x_{new} = x_{old} + \epsilon \bar{A}^t, \quad (5)$$

where $\epsilon \in [-1, 1]$ is a random number and $\bar{A}^t = \langle A_i^t \rangle$ is the average loudness of all bats at time step t.

2.1.4 Solutions and parameters update

As indicated in Yang (2010), BA can be considered as a balanced combination of a classical PSO and the intensive local search controlled by both loudness and pulse emission rate.

When a bat looks for a prey it decreases its loudness while increasing the rate of pulse emission. For simplicity, the algorithm starts with an initial set loudness A_0 which is reduced at each iteration until it reaches an A_{min} near zero which represents a bat catching its prey. The bat is guided toward an optimal solution using the following two design equations,

$$If(rand < A_i^t)\ and\ f(x_i^{t+1}) < f(x_i^t)$$

$$A_i^{t+1} = \alpha A_i^t, \quad (6)$$

$$r_i^{t+1} = r_i^0 \left[1 - \exp(-\gamma t)\right], \quad (7)$$

where α and γ are constants. The value α, which is similar to a cooling factor of the simulated annealing cooling schedule (Kirkpatrick et al., 1983), lies between 0 and 1 while γ is greater than 0. The value A_i^{t+1} represent an updated value of the loudness A_i^t of bat i at time step t. As the time step t tends toward infinity, the rate of pulse emission converges to the initial rate of pulse emission r_i^0 whereas the average loudness of the bat approaches zero expressed as follows,

$$A_i^t \to 0, \quad r_i^t \to r_i^0, \quad as\ t \to \infty \quad (8)$$

2.2 Fractional-order calculus

Definition 2.1: The Riemann-Liouville fractional derivative of an order $\alpha > 0$ of a continuous function $x : (0, +\infty) \to \mathbb{R}$ is given by

$$^{RL}D_{0^+}^{\alpha} x(t) = \frac{1}{\Gamma(m-\alpha)} \frac{d^m}{dt^m} \int_0^t \frac{x(s)}{(t-s)^{\alpha-m+1}} ds, \quad (9)$$

where $\Gamma(x)$ is the gamma function, $m - 1 \le \alpha < m$, and the right-hand side is point-wise defined on $(0, +\infty)$.

Definition 2.2: Starting with the assumption that a function $x(s)$ satisfies some smooth condition for a finite interval $(0, t)$, the *Grüwald-Letnikov* fractional derivative definition, which is based on finite difference, with respect to a fractional coefficient $\alpha \in \mathbb{R}$ in an equidistant grid in $[0, t]$ such that:

$$\begin{aligned} 0 &= s_0 < \cdots < s_i \\ &= (i+1)h < \cdots < s_{n+1} = t \\ &= (n+1)h, \end{aligned}$$

is given by

$$
\begin{aligned}
{}^{RL}D_{0^+}^{\alpha}x(t) &= \lim_{h\to 0}\frac{1}{h^{\alpha}}\Delta_h^{\alpha}x(t) \\
&= \frac{1}{h^{\alpha}}\Delta_h^{\alpha}x(t) + O(h),
\end{aligned} \tag{10}
$$

where

$$
\begin{aligned}
\frac{1}{h^{\alpha}}\Delta_h^{\alpha}x(t) = \frac{1}{h^{\alpha}}\Bigg(& x(s_{n+1}) \\
& + \sum_{i=1}^{n+1}(-1)^i\binom{\alpha}{i}x(s_{n+1-i})\Bigg)
\end{aligned} \tag{11}
$$

$$
D^{\alpha}x(t) = \lim_{h\to 0}\left[\frac{1}{h^{\alpha}}\sum_{k=0}^{+\infty}\frac{(-1)^k\Gamma(\alpha+1)x(t-kh)}{\Gamma(k+1)\Gamma(\alpha-k+1)}\right] \tag{12}
$$

2.3 Bat algorithm modified equations

Following the same principles and steps as in Couceiro and Sivasundaram (2016), the bat algorithm is described both in discrete and continuous form. In discrete form, the general bat algorithm is given by

$$
\begin{aligned}
v[t+1] &= v[t] + (x[t] - x^g[t])f \\
x[t+1] &= x[t] + v[t+1]
\end{aligned} \tag{13}
$$

and in continuous form by

$$
\begin{aligned}
v'[t] &= v[t] + (x[t] - x^g[t])f \\
x'[t] &= v[t]
\end{aligned} \tag{14}
$$

The derivatives in equation (14) can be rewritten as fractional derivative in the following form,

$$
\begin{aligned}
{}^{GL}D_{0^+}^{\alpha}v[t] &= v[t] + (x[t] - x^g[t])f \\
{}^{GL}D_{0^+}^{\alpha}x[t] &= v[t]
\end{aligned} \tag{15}
$$

As a result, equation (15) is rewritten in a form which allows simple numerical computation,

$$
\begin{aligned}
v[t^{n+1}] &= h^{\alpha}\left\{v[t^n] + (x[t^n] - x^g[t^n])f\right\} \\
&\quad - \sum_{k=1}^{n+1} s_k v[t_{n+1} - kh], \\
x[t^{n+1}] &= h^{\alpha}v[t^{n+1}] - \sum_{k=1}^{n+1} s_k v[t_{n+1} - kh]
\end{aligned} \tag{16}
$$

where the coefficients s_k are computed in a recursive scheme as follows

$$
\begin{aligned}
s_0 &= 1, \\
s_k &= \left(1 - \frac{\alpha+1}{k}\right)s_{k-1},\ k > 0
\end{aligned} \tag{17}
$$

and $0 = t^0 < \cdots < t^i = ih < \cdots < t^{n+1} = (n+1)h = T$.

2.4 Lévy flight

It is shown in previous studies, that different animals and insects follow a Lévy flight behaviour in their search for preys or when flying in swarms. The process is defined as a non-Gaussian stochastic random walk kind in which the random step lengths are computed using a Lévy distribution.

For the bat algorithm a new location x_i' corresponding to an i^{th} bat is derived by combining a Lévy flight to its old position x_i as follows:

$$
x_i' = x_i + \zeta \oplus Levy(\lambda) \tag{18}
$$

where ζ is a random step size, λ is a Lévy flight distribution parameter and $\oplus$ indicates an entry-wise multiplication. The step size ζ is obtained using the Mantegna algorithm (Mantegna, 1994) such that,

$$
\zeta = v \oplus Levy(\lambda) \sim 0.01\frac{u}{|v|^{1/\beta}}(x_i - x_{opt}) \tag{19}
$$

where u and v are obtained from

$$
\begin{aligned}
& u \sim N(0, \sigma_u^2),\ \sigma_u = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\beta\Gamma[(1+\beta)/2]2^{(\beta-1)/2}}\right)^{1/\beta} \\
& v \sim N(0, \sigma_v^2),\ \sigma_v = 1
\end{aligned} \tag{20}
$$

and Γ is the gamma function defined as,

$$
\Gamma(y) = \int z^{y-1}e^{-y}dt \tag{21}
$$

2.5 DE-based location update formula

In order to overcome the entrapment of the BA on local optimal values, that occurs due to a location update equation which is based only on a global best solution. In Yilmaz and Küçüksille (2015), it is proposed an improved version based on DE. They provide a formula which allows a better search ability at both local and global levels through the following equation:

$$
v_i^{t+1} = \omega^t v_i^t + f_i\xi_1^t(x_i^t - x^g) + f_i\xi_2^t(x_i^t - x_q^t) \tag{22}
$$

$$
\begin{aligned}
& \xi_1^t + \xi_2^t = 1, \\
& \xi_1^t = 1 + (\xi_{init} - 1)\frac{(T-t)^n}{T^n},
\end{aligned} \tag{23}
$$

where x_q^t is a randomly selected solution from the population and ξ_1^t and ξ_2^t are learning factors in the range $[0,1]$. The ξ_1^t is computed based on a relation between an initial ξ_{init}, the maximum number of iteration T, the actual iteration number t, and a nonlinear index n as presented in equation (23).

This modification allows an exploitation in one hand by guiding the velocity update toward the global best x^g and on the other hand an exploration based on a third term that is based on a random position x_q^t. The balance between exploration and exploitation is governed by the changes in the term ξ_1.

3 Fractional Lévy flight bat algorithm

Even though BA has the advantage of simplicity and flexibility, just like any other metaheuristic it still lacks the mechanism of escaping local optimums. The idea is to design an algorithm capable of adjusting itself to the fitness functions landscape making it more robust and applicable to any sort of optimisation problem. An innovative bat algorithm based on fractional calculus, differential evolution and Lévy flight is introduced and explained in this section.

The proposed algorithm starts by generating a population of N random locations using equation (1) and assigning values to the initial velocity vector. The objective function is then evaluated at each position of the initial population and the first global best solution is selected. At each generation and for each bat in the population an update process of the positions is divided, with an equal probability of 50%, between two different mechanisms. Here is important to mention that the population is divided to increase the diversity of the solutions using different operators to modify each partition. As was previously mentioned the LF increases the exploration of the search space and the FLF enhances the exploitation of the prominent regions.

The first mechanism starts by producing a new location using the difference between two randomly selected local best solutions multiplied by a random value ε. This random variable is taken from a uniform distribution and then added to the i^{th} local best solution as in the following equation,

$$s = \varepsilon(x_q^l - x_p^l) \tag{24}$$

$$x_n = x_i^l + s \tag{25}$$

The fitness function is then evaluated at this new solution and then compared with function value at the i^{th} local best solution. If the new location x_n produces a better result than x_i^l then $x_i^l = x_n$. An updated solution x_i^{t+1} is obtained in the neighbourhood of the corresponding best local solution x_i^l through a Lévy flight search as follows,

$$x_i^{t+1} = x_i^l + 0.01\frac{u}{|v|^{1/\beta}}|\bar{x}^l - x_i^t| \tag{26}$$

where $\bar{x}^l$ is the mean value of the vector of local best solutions. This will assign half of the population in a local search around the selected local best solution which should result in a better exploration procedure.

A combination of DE and fractional calculus velocity update equation is applied to obtain the second half of the population. The velocity term is computed based on a fractional differential formula where,

$$\nu_i^t = \sum_{k=0}^{o} s_k v_i^{t-k} \tag{27}$$

where s_k is obtained using equation (17) and o is the order of the fractional derivative which is selected randomly between the integers 1 and 10. The above process tries to mimics to a certain degree a continuous time random walk (CTRW) where for each individual of the population a different learning period is selected based on the order o of equation (27).

As defined by Zhou et al. (2015), the CTRW strategy is a composition of the flight step lengths of a movement in a random direction with the elapsed waiting time between two successive movement steps which are both represented by two independent random variables distributed according to their probability densities. The main idea of CTRW search strategy is presented in Zhou et al. (2015) by using the probability distribution function of search length:

$$\omega(l_j) \sim l_j^{-(\alpha+1)} \tag{28}$$

where $0 < \alpha < 2$, and l_j is the search length at the j^{th} step and probability distribution function of waiting time.

$$\psi(t_j) \sim t_j^{-(\beta+1)} \tag{29}$$

where $0 < \beta < 1$, and t_j is the waiting time length before starting the j^{th} step. The Fractional random walk has been used successfully on undirected regular networks such as Riascos and Mateos (2014) and Michelitsch et al. (2017).

Algorithm 1 Pseudo code of FLFBA

N: the number of individuals (bats) in a single population
$MIter$: the maximum number of iteration
G: the frequency of renewing a section of the population
Pa: The portion of the population to be renewed
α, γ, f_{max}, f_{min}, A_0, r_0: classical BA parameters
$t = 0$; Initialise the population using equation (1)
Initialise velocity vector and local best solutions
Compute fitness and select best solution
while $t < MIter$ **do**
    $f^t = r(f_{max} - f_{min}) + f_{min}$; f^t: frequency
    Update ω^t, ξ_1^t and ξ_2^t; ω^t: inertia weight; $\xi_{1,2}^t$: learning factors
    for $j = 1, \ldots, n$ **do**
        if $\varepsilon < 0.5$ **then** $\varepsilon \in [0, 1]$
            Select two random best local bats
            Generate a new bat x_n using equation (25)
            if $F(x_n) < F(x_j^l)$ **then**
                $x_j^l = x_n$ ▷ Update local best
            end if
            Compute x_j^t using equation (26)
        else
            Compute v_i^t ▷ Using equations (27) and (22)
            $x_j^t = x_j^{t-1} + v_j^t$
        end if
        if $\varepsilon > r_j$ **then** $\varepsilon \in [0, 1]$
            Do a local search using equation (32)
        end if
        Evaluate $F(x_j^t)$
        Update local best solution
        if $F(x_j^l) < F(x^g)$ and $\varepsilon < A_i$ **then**
            $x^g = x_j^l$ ▷ Update global best
            Update A_j and r_j using equations (6) and (7)
        end if
    end for
    $t = t + 1$
end while

The velocity term computed in equation (27) is combined with an exploration and exploitation term, through a DE approach. In this process is obtained the updated velocity vector corresponding to each individual of the population. The following formula explains the computation of such value:

$$v_i^{t+1} = \omega^t \nu_i^t + f_i^t \xi_1^t (x^g - x_i^t) + f_i^t \xi_2^t (x_i^l - x_q^t) \quad (30)$$

where ω^t is an inertia weight, f_i^t is the frequency, x^g is the global best location. x_i^l is the corresponding local best solution, and x_q^t is a randomly selected solution from the population such that $q \neq i$. The new proposed formula contains several important factors, such as information from previous generations, a combination of exploitation around the global best and an exploration with respect to local best. This combination is reflected in a better and consistent convergence of the algorithm. The new computed velocity vector is then added to the actual corresponding location vector as in equation (4).

Once the second half of the population locations are updated, a local search routine, based on the one the proposed in Meng et al. (2015), is applied only to the solutions which pulse emission r_i is below a randomly generated value in the interval $[0, 1]$. The proposed local search is combined with a Lévy flight pattern around the global best solution as indicated below,

$$rA = 0.01 \frac{u}{|v|^{1/\beta}} |A_i - \bar{A}| \quad (31)$$

$$x_i^{t+1} = x^g (1 + rA) \quad (32)$$

where A_i is the i^{th} related loudness value and $\bar{A}$ is the mean value of the loudness vector.

The local best bats are updated and if $\exists x_i^l,\ F(x_i^l) < F(x^g),\ i = 1, \cdots, n$ and the associated loudness value A_i is greater than a random number $\varepsilon \in [0, 1]$ then $x^g = x_i^l$. This is then followed by a reduction in the corresponding i^{th} loudness A_i and an increase in the pulse emission r_i as presented in equations (6) and (7). The pseudo-code of the modified bat algorithm is presented in Algorithm 1.

4 Results and analysis

The parameters settings of the CS, FDPSO, NBA, ACO, MFO, SFLA and FLFBA are provided in Table 1. The maximum number of iterations was set to 50 times the dimension such as, for $D = 10$ it is 500, for $D = 20$ it is 1000 and finally for $D = 40$ it is 2000. The search space in all algorithms is restricted to the interval $[-5,\ 5]^D$ since the majority of the benchmark functions has the global optimum solution inside the interval $[-4,\ 4]^D$ or drawn uniformly from this compact. For a better analysis of the results, each optimisation procedure was repeated 50 times overall the functions in the three dimensions. It should be noted that the inconsistent results obtained for F_5 in all dimensions were omitted due to a probable bug in the downloaded benchmark source code.

The proposed algorithm is evaluated for performance using 24 CEC2015 benchmark functions (Qu et al., 2016).

In Table 2, the average computational time of the selected algorithms using 50 different trials for each benchmark function computed using three variables dimensions, $10 - D/20 - D/40 - D$, is presented. The first observation is the large computational time of both SFLA followed by FDPSO and, for some benchmark functions, CS algorithms which is more than 4 times that of NBA and MFO algorithms. The second observation is the similar performance of ACO and FLFBA time wise.

Table 1 Parameters settings

Parameters	*CS*	*FDPSO*	*NBA*	*ACO*	*MFO*	*SFLA*	*FLFBA*
Population size N	30	30	30	30	30	30	30
Loudness A_0	–	–	$[1, 2]$	–	–	–	$[1, 2]$
Pulse emission r_0	–	–	$[0, 1]$	–	–	–	$[0, 1]$
Frequency $[f_{min}, f_{max}]$	–	–	$[0, 1.5]$	–	–	–	$[0, 2]$
ρ, γ	–	–	0.99, 0.9	–	–	–	0.99, 0.9
Probability habitat selection P	–	–	$[0.5, 0.9]$	–	–	–	–
Compensation rates Doppler echoes C	–	–	$[0.1, 0.9]$	–	–	–	–
Contraction-expansion coefficient θ	–	–	$[0.5, 1]$	–	–	–	–
Inertia weight ω	–	0.9	$[0.4, 0.9]$	–	–	–	$[0.2, 0.9]$
Fractional coefficient α	–	0.632	–	–	–	–	0.632
Cognitive and social components	–	1.5, 1.5	–	–	–	–	–
Search counter/max iterations	–	15	–	–	–	5	–
Number of swarms $[\min, n, \max]$	–	$[1, 2, 3]$	–	–	–	–	–
Discovery rate/step-size	0.25	–	–	2	–	–	–
Deviation-distance ratio	–	–	–	1	–	–	–
Intensification factor	–	–	–	0.5	–	–	–
Memeplex/sample size	–	–	–	40	–	5	–
Offspring number	–	–	–	–	–	3	–

Table 2 Average computational time of the different algorithms using 50 trials for the benchmarks function $F_1 \cdots F_{24}$

Algorithm	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
						D – 10						
CSN	7.67	9.34	9.86	12.98	9.44	14.92	13.14	12.18	12.09	13.78	13.73	12.83
FDPSO	12.13	27.58	19.22	19.79	11.62	19.64	8.41	16.71	18.14	21.57	8.69	12.86
NBA	2.11	2.75	3.18	3.27	1.72	2.91	2.40	2.27	2.44	2.90	2.79	2.52
ACO	3.58	4.38	4.61	5.04	2.50	4.48	3.79	3.48	3.57	4.25	4.23	3.88
MFO	4.28	4.18	3.99	4.15	2.59	4.27	4.32	3.40	1.74	2.18	2.27	1.94
SFLA	14.86	14.85	14.75	15.22	12.41	15.34	15.70	12.60	12.40	15.74	16.57	12.78
FLFBA	4.11	6.77	7.05	6.90	5.85	8.58	7.37	6.78	7.45	8.46	5.56	4.20
						D – 20						
CSN	15.41	19.30	21.06	21.89	15.15	19.47	18.12	16.62	17.00	20.46	20.16	18.06
FDPSO	18.85	48.92	32.50	33.39	16.15	33.77	14.10	22.58	20.10	30.80	17.76	22.04
NBA	4.11	5.27	5.81	5.92	3.13	5.33	4.30	4.08	4.07	5.37	5.27	4.48
ACO	12.04	11.83	11.24	11.35	5.87	10.68	9.57	8.82	9	10.68	10.49	9.63
MFO	2.92	4.21	4.67	4.83	2.22	4.48	3.66	3.13	3.26	4.33	4.21	3.62
SFLA	27.64	32.95	34.59	34.50	24.55	32.14	32.14	25.11	24.49	32.25	34.61	25.51
FLFBA	11.40	19.17	14.79	16.7	8.76	16.82	15.33	15.00	18.62	18.43	18.87	15.92
						D – 40						
CSN	32.71	42.03	45.92	46.82	35.60	40.45	38.40	35.20	36.03	43.32	42.77	38.01
FDPSO	28.35	70.98	48.53	48.21	21.30	52.25	22.67	32.37	30.68	48.40	34.88	33.61
NBA	7.01	10.75	12.58	12.64	7.28	10.92	9.00	7.69	8.08	11.30	11.29	9.15
ACO	23.68	28.73	30.42	30.62	16.74	28.70	26.52	24.63	25.25	29.34	29.19	26.57
MFO	5.63	8.77	9.88	10.10	4.35	8.77	7.43	6.28	6.59	9.23	9.04	7.38
SFLA	59.91	71.40	76.01	75.77	53.66	72.66	69.47	54.53	56.20	69.08	70.77	57.26
FLFBA	15.32	19.00	21.13	21.18	10.13	18.91	16.97	20.28	26.82	22.85	29.75	26.60

Algorithm	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}	F_{21}	F_{22}	F_{23}	F_{24}
						D – 10						
CSN	12.19	12.01	14.61	16.07	14.22	10.46	8.76	9.93	11.86	11.50	9.36	8.53
FDPSO	13.15	11.48	20.41	10.24	14.80	11.62	5.90	14.42	24.97	25.07	10.70	8.16
NBA	2.33	2.37	3.21	3.58	3.17	3.03	2.26	2.61	4.01	3.51	3.87	2.40
ACO	3.46	3.57	4.96	5.20	4.56	4.41	3.54	3.99	5.27	4.87	4.43	3.67
MFO	1.69	1.81	2.53	2.78	2.34	2.35	1.73	1.98	3.06	2.69	2.22	1.77
SFLA	11.25	13.22	14.82	16.24	14.01	14.18	15.80	13.11	17.16	16.03	16.93	13.48
FLFBA	3.87	5.71	5.39	5.90	5.16	5.00	4.05	4.53	6.44	5.84	11.19	6.93
						D – 20						
CSN	17.88	16.89	21.05	22.97	20.79	21.00	16.85	19.62	25.57	23.88	19.80	18.22
FDPSO	20.79	20.64	31.78	19.59	25.38	21.50	12.59	21.66	38.35	34.70	13.55	14.92
NBA	3.94	4.21	6.06	7.47	6.08	5.96	4.33	4.81	7.25	6.38	8.57	4.27
ACO	8.72	8.94	11.44	12.71	10.97	10.85	9.27	9.61	12.69	11.78	11.08	8.92
MFO	3.07	3.22	4.83	5.72	4.60	4.52	3.28	3.66	5.84	5.14	4.43	3.30
SFLA	24.14	27.79	33.81	35.06	29.19	29.19	31.82	30.60	42.00	39.10	32.58	27.17
FLFBA	20.52	11.88	16.25	20.32	24.63	17.32	9.09	9.51	14.04	14.83	18.88	10.57
						D – 40						
CSN	34.87	35.19	46.59	52.62	46.1	45.76	35.94	41.73	58.64	49.63	42.26	37.01
FDPSO	33.08	32.03	50.65	46.96	47.38	41.65	23.79	34.15	67.42	54.93	28.65	27.83
NBA	7.92	8.56	13.20	20.50	11.89	11.83	8.04	9.32	15.73	12.78	17.46	8.59
ACO	24.73	25.46	31.60	37.38	30.25	30.15	25.35	26.50	34.47	31.42	30.70	25.31
MFO	6.24	6.67	10.54	14.10	9.87	9.85	6.67	7.38	13.21	10.72	9.87	6.83
SFLA	52.53	59.69	70.98	78.38	66.55	65.54	66.85	64.00	88.60	79.58	65.68	59.27
FLFBA	23.02	25.27	34.83	59.88	30.17	31.09	23.40	25.42	39.03	32.27	59.71	24.22

To analyse the performance of stochastic algorithms based on computational intelligence we avoid using the parametric tests because the independence, normality, and homoscedasticity assumptions cannot be satisfied, for that reason non-parametric statistical procedures are very practical in this case (Derrac et al., 2011). In this section we deal with multiple comparisons class of non-parametric analysis and post-hoc procedures to evaluate the performance of FLFBA with respect to remaining five algorithms. Moreover, this analysis helps verify if the proposed FLFBA can significantly improve the accuracy in comparison with other methods.

Table 3 Average Rankings of the algorithms

Algorithm	*Friedman*	*Aligned Friedman*	*Quade*
	D – 10 CASES		
CS	4.9792	109.4375	5.02167
FDPSO	4.5625	102.9792	4.5716
NBA	2.2291	51.1041	2.1183
ACO	3.8125	80.8125	3.8066
MFO	3.1875	63.0625	2.9116
SFLA	5.666	112.3334	5.9183
FLFBA	3.5624	71.7708	3.6516
	D – 20 CASES		
CS	5.0416	107.8333	5.0633
FDPSO	4.6667	95.8333	4.6733
NBA	2.4583	62.9166	2.380
ACO	3.75	81.9583	3.7066
MFO	2.5000	46.3333	2.0466
SFLA	5.9583	110.9166	6.4366
FLFBA	3.625	85.7083	3.6933
	D – 40 CASES		
CS	5.29166	107.9166	5.2866
FDPSO	5.2083	98.6249	5.1233
NBA	2.9166	74.3333	2.71
ACO	2.0	57.6666	2.1433
MFO	2.3333	47.5416	1.9100
SFLA	6.125	108.875	6.4866
FLFBA	4.1245	96.5416	4.339

The Friedman test aims to distinguish significant differences between two or more algorithms. Its null hypothesis designates sameness of medians between the populations when the alternative hypothesis is given as the reversal of the null hypothesis, its statistic distributed according to the chi-square distribution with $(k-1)$ degrees of freedom. The Friedman aligned Ranks test is used when the number of compared algorithms is small. The statistical test is evaluated through a chi-square distribution with $(k-1)$ degrees of freedom. The Quade test considered as an alternative test of Friedman by means of the difficulty considerations. In this sense, the rankings computed on each problem could be scaled depending on the differences observed in the algorithm's performances, finding, as a consequence, a weighted ranking analysis of the results sample. Its distributed according to the Fisher distribution with $(k-1)$ and $(k-1)(n-1)$ degrees of freedom where k is the number of the tested algorithms and n is the number of problems considered.

Table 3 provides the average rankings of the algorithms achieved by the Friedman, Friedman aligned, and Quade tests for $D-10$, $D-20$ and $D-40$. Results indicate that SFLA achieves the best average rank by all three tests in all the dimensions while FLFBA was fifth in both 10 and 20 dimensions and finally NBA scored last. A different order of performance is obtained in the $D-40$ case where FLFBA was fourth and ACO came last.

Our experimental study shows that the Friedman and Friedman aligned ranks are both distributed according chi-square distribution with 6 degrees of freedom, while the Quade test is distributed according to F-distribution with 6 and 138 degrees of freedom. The Friedman statistic shows an ($F = 41.5312$, p-value = 2.2757E-7) for the $10-D$ cases, an ($F = 52.4285$, p-value = 1.5810E-9) for the $20-D$ cases and finally an ($F = 80.2857$, p-value= 4.7076E-11) for the $40-D$ cases. Iman and Davenport test indicates for the $D-10$ an ($F = 9.3220$, p-value = 1.4116E-8), for $D-20$ ($F = 13.1684$, p-value = 9.4150E-12) and for $D-40$ ($F = 28.9820$, p-value = 2.9043E-22) proposing the existence of significant differences between the tested algorithms.

Table 4 Contrast estimation

	CS	*FDPSO*	*NBA*	*ACO*	*MFO*	*SFLA*	*FLFBA*
			D – 10 CASES				
CS	0	–0.9582	–7.124	–2.193	–4.534	1.631	–2.774
FDPSO	0.9582	0	–6.165	–1.235	–3.576	2.589	–1.815
NBA	7.124	6.165	0	4.931	2.589	8.754	4.350
ACO	2.193	1.235	–4.931	0	–2.341	3.823	–0.5806
MFO	4.534	3.576	–2.589	2.341	0	6.165	1.761
SFLA	–1.631	–2.589	–8.754	–3.823	–6.165	0	–4.404
FLFBA	2.774	1.815	–4.350	0.5806	–1.761	4.404	0
			D – 20 CASES				
CS	0.000	–5.059	–19.68	–7.410	–28.00	9.945	–9.221
FDPSO	5.059	0.000	–14.62	–2.350	–22.94	15.00	–4.161
NBA	19.68	14.62	0.000	12.27	–8.321	29.62	10.46
ACO	7.410	2.350	–12.27	0.000	–20.59	17.35	–1.811
MFO	28.00	22.94	8.321	20.59	0.000	37.95	18.78
SFLA	–9.945	–15.00	–29.62	–17.35	–37.95	0.000	–19.17
FLFBA	9.221	4.161	–10.46	1.811	–18.78	19.17	0.000
			D – 40 CASES				
CS	0.000	–34.32	–86.68	–177.6	–157.2	42.20	–32.72
FDPSO	34.32	0.000	–52.36	–143.3	–122.9	76.52	1.598
NBA	86.68	52.36	0.000	–90.97	–70.56	128.9	53.96
ACO	177.6	143.3	90.97	0.000	20.41	219.8	144.9
MFO	157.2	122.9	70.56	–20.41	0.000	199.4	124.5
SFLA	–42.20	–76.52	–128.9	–219.8	–199.4	0.000	–74.92
FLFBA	32.72	–1.598	–53.96	–144.9	–124.5	74.92	0.000

Contrast estimation based on medians used to estimate the difference between the presentations of two algorithms in view of all pairwise comparisons. It is particularly helpful to estimate by how far an algorithm outperforms another one. The importance of this test can be summarised as the estimation of dissimilarity between medians of samples of results. It is important to note that this test cannot give a probability of error related to the refusal of the null hypothesis of equality. In our experimental study, we can calculate the set of estimators of medians directly from the average error results.

Table 4 shows the estimations computed for each algorithm in the $D-10$, $D-20$ and $D-40$ cases respectively. Focusing our attention in the rows of the tables, we can underline the performance of FSLA as the best performing algorithm because it have the max number of negative related estimators (attain very low error rates considering median estimators) followed by CS algorithm while FLFBA was fourth in both the $D-20/D-40$ cases.

4.1 Post-hoc procedures

Since the Friedman, Iman-Davenport, Friedman aligned, and Quade tests can just detect significant differences over the complete multiple comparisons, which makes it incapable to create accurate comparisons between some of the considered algorithms then we can progress with a post-hoc procedure that permit us to establish which algorithms are significantly better/worse.

Tables 5, 6 and 7 show the Holm/ Hochberg/ Hommel, Holland, Rom, Finner and Li procedures for all six algorithms in the $D-10$, $D-20$ and $D-40$ cases respectively for $alpha = 0.05$.

In order to better show the differences between the three tests and their respective approximations for obtaining the p-value (also named unadjusted p-values), of every hypothesis, we will compute the unadjusted p-values for the selected algorithms. Numerous dissimilarities can be clarified; Friedman test shows a lower power than the Friedman aligned test (the unadjusted p-values are considerably lower). Within a multiple comparisons tests the p-values are not appropriate because it does not consider the remaining comparisons going to the family, it just represent the probability error of a certain comparison.

Adjusted p-values can treat this problem. They are suitable to be employed since they offer more information in a statistical analysis. They assume the accumulated family error, also they can be evaluated directly with every selected significance level α. Therefore, Tables 8, 9 and 10 show the p-values obtained, using the ranks computed by the Friedman, Friedman aligned, and Quade tests, respectively for the eight considered post hoc procedures for the $D-10$, $D-20$ and $D-40$ cases respectively.

Table 5 Holm-Hochberg-Hommel (H-H-H)/Holland/Rom/Finner/Li table for $\alpha = 0.05$ in $D-10$

i	*Algorithm*	$z = \frac{R_0 - R_i}{SE}$	*p*	*H-H-H*	*Holland*	*Rom*	*Finner*	*Li*
				FRIEDMAN				
6	SFLA	5.5122	3.5425E-8	0.0083	0.00851	0.0087	0.0085	0.0461
5	CS	4.4098	1.0346E-5	0.01	0.0102	0.0105	0.0169	0.0461
4	FDPSO	3.7416	1.8281E-4	0.0125	0.0127	0.0131	0.0253	0.0461
3	ACO	2.5389	0.0111	0.01666	0.0169	0.0166	0.0336	0.0461
2	FLFBA	2.1380	0.0325	0.025	0.0253	0.025	0.0418	0.0461
1	MFO	1.5367	0.1243	0.05	0.05	0.05	0.05	0.05
				ALIGNED FRIEDMAN				
6	SFLA	4.360	1.2973E-5	0.0083	0.0085	0.0087	0.0085	0.0318
5	CS	4.1543	3.2625E-5	0.01	0.0102	0.0105	0.0169	0.0318
4	FDPSO	3.6943	2.2042E-4	0.0125	0.0127	0.0131	0.0253	0.0318
3	ACO	2.1157	0.0343	0.0166	0.0169	0.0166	0.0336	0.0318
2	FLFBA	1.4718	0.1410	0.025	0.0253	0.025	0.04184	0.0318
1	MFO	0.8516	0.3944	0.05	0.05	0.05	0.05	0.05
				QUADE				
6	SFLA	3.0777	0.002	0.0083	0.0085	0.0087	0.0085	0.0252
5	CS	2.3514	0.0186	0.01	0.0102	0.0105	0.0169	0.0252
4	FDPSO	1.9870	0.0469	0.0125	0.0127	0.0131	0.0253	0.0252
3	ACO	1.3674	0.1714	0.0166	0.0169	0.0166	0.0336	0.0252
2	FLFBA	1.2418	0.2142	0.025	0.0253	0.025	0.0419	0.0252
1	MFO	0.6425	0.5205	0.05	0.05	0.05	0.05	0.05

Table 6 Holm-Hochberg-Hommel (H-H-H)/Holland/Rom/Finner/Li table for $\alpha = 0.05$ in $D - 20$

i	*algorithm*	$z = \frac{R_0 - R_i}{SE}$	p	*H-H-H*	*Holland*	*Rom*	*Finner*	*Li*
				FRIEDMAN				
6	SFLA	5.6124	1.9944E-8	0.0083	0.0085	0.0087	0.0085	0.0028
5	CS	4.1425	3.4346E-5	0.01	0.0102	0.0105	0.0169	0.0028
4	FDPSO	3.5412	3.9829E-4	0.0125	0.0127	0.0131	0.0253	0.0028
3	ACO	2.07127	0.0383	0.0166	0.0169	0.0166	0.0336	0.0028
2	FLFBA	1.8708	0.0613	0.025	0.0253	0.025	0.0418	0.0028
1	MFO	0.0668	0.9467	0.05	0.05	0.05	0.05	0.05
				ALIGNED FRIEDMAN				
6	SFLA	4.5994	4.2365E-6	0.0083	0.0085	0.0087	0.0085	0.0401
5	CS	4.3798	1.1876E-5	0.01	0.0102	0.0105	0.0169	0.0401
4	FDPSO	3.5252	4.2310E-4	0.0125	0.0127	0.0131	0.0253	0.0401
3	FLFBA	2.8041	0.0050	0.0166	0.0169	0.0166	0.0336	0.0401
2	ACO	2.5371	0.0111	0.025	0.0253	0.025	0.0418	0.0401
1	NBA	1.1810	0.2375	0.05	0.05	0.05	0.05	0.05
				QUADE				
6	SFLA	3.5555	3.7716E-4	0.0083	0.0085	0.0087	0.0085	0.0112
5	CS	2.4432	0.0145	0.01	0.0102	0.0101	0.0169	0.0112
4	FDPSO	2.1274	0.0333	0.0125	0.0127	0.0131	0.0253	0.0112
3	ACO	1.3444	0.1787	0.0166	0.0169	0.0166	0.0336	0.0112
2	FLFBA	1.3336	0.1823	0.025	0.0253	0.025	0.0418	0.0112
1	NBA	0.2699	0.7871	0.05	0.05	0.05	0.05	0.05

Table 7 Holm-Hochberg-Hommel (H-H-H)/Holland/Rom/Finner/Li table for $\alpha = 0.05$ in $D - 40$

i	*Algorithm*	$z = \frac{R_0 - R_i}{SE}$	p	*H-H-H*	*Holland*	*Rom*	*Finner*	*Li*
				FRIEDMAN				
6	SFLA	6.6147	3.7226E-11	0.0083	0.0085	0.0087	0.0085	0.0214
5	CS	5.2784	1.3030E-7	0.01	0.0102	0.0105	0.0169	0.0214
4	FDPSO	5.1447	2.6783E-7	0.0125	0.0127	0.0131	0.0253	0.0214
3	FLFBA	3.4075	6.5541E-4	0.0166	0.0169	0.0166	0.0336	0.0214
2	NBA	1.4699	0.1415	0.025	0.0253	0.025	0.0418	0.0214
1	MFO	0.5345	0.5929	0.05	0.05	0.05	0.05	0.05
				ALIGNED FRIEDMAN				
6	SFLA	4.3679	1.2540E-5	0.0083	0.0085	0.0087	0.0085	0.0278
5	CS	4.2997	1.7101E-5	0.01	0.0102	0.0105	0.0169	0.0278
4	FDPSO	3.6379	2.7476E-4	0.0125	0.0127	0.0131	0.0253	0.0278
3	FLFBA	3.4896	4.8369E-4	0.01666	0.0169	0.0166	0.0336	0.0278
2	NBA	1.9080	0.05638	0.025	0.0253	0.025	0.0418	0.0278
1	ACO	0.7210	0.4708	0.05	0.05	0.05	0.05	0.05
				QUADE				
6	SFLA	3.7067	2.0993E-4	0.0083	0.0085	0.0087	0.0085	0.0078
5	CS	2.7348	0.0062	0.01	0.0102	0.0105	0.0169	0.0078
4	FDPSO	2.6025	0.0092	0.0125	0.0127	0.0131	0.0253	0.0078
3	FLFBA	1.9681	0.04905	0.0166	0.0169	0.0166	0.0336	0.0078
2	NBA	0.6479	0.5170	0.025	0.0253	0.025	0.0418	0.0078
1	ACO	0.1889	0.8501	0.05	0.05	0.05	0.05	0.05

Table 8 Adjusted p-values (FRIEDMAN/ALIGNED FRIEDMAN/QUADE) in $D - 10$

i	*Algorithm*	*Unadjusted* p	p_{Bonf}	p_{Holm}	p_{Hoch}	p_{Homm}	p_{Holl}	p_{Rom}	p_{Finn}	p_{Li}
FRIEDMAN										
1	SFLA	3.5424E-8	2.1254E-7	2.1254E-7	2.1254E-7	2.1254E-7	2.1254E-7	2.0210E-7	2.1254E-7	4.0455E-8
2	CS	1.0346E-5	6.2076E-5	5.1730E-5	5.1730E-5	5.1730E-5	5.1729E-5	4.9195E-5	3.1038E-5	1.1815E-5
3	FDPSO	1.8281E-4	0.0010	7.3124E-4	7.3124E-4	7.3124E-4	7.3104E-4	6.9725E-4	3.6558E-4	2.0872E-4
4	ACO	0.0111	0.0667	0.03335	0.03335	0.03335	0.0329	0.0333	0.0166	0.0125
5	FLFBA	0.0325	0.1950	0.0650	0.0650	0.0650	0.0639	0.0650	0.0388	0.0357
6	MFO	0.1243	0.7461	0.1243	0.1243	0.1243	0.1243	0.1243	0.1243	0.1243
ALIGNED FRIEDMAN										
1	SFLA	1.2973E-5	7.7840E-5	7.7840E-5	7.7840E-5	7.7840E-5	7.7837E-5	7.4013E-5	7.7837E-5	2.1422E-5
2	CS	3.2625E-5	1.9575E-4	1.6312E-4	1.6312E-4	1.63125E-4	1.6311E-4	1.5513E-4	9.7873E-5	5.3871E-5
3	FDPSO	2.2042E-4	0.0013	8.8170E-4	8.8170E-4	8.8170E-4	8.8141E-4	8.4071E-4	4.4080E-4	3.6385E-4
4	ACO	0.0343	0.2062	0.1031	0.1031	0.1031	0.0995	0.1031	0.0511	0.0537
5	FLFBA	0.1410	0.8464	0.2821	0.2821	0.2821	0.2622	0.2821	0.1668	0.1889
6	MFO	0.3944	2.3664	0.3944	0.3944	0.3944	0.3944	0.3944	0.3944	0.3944
QUADE										
1	SFLA	0.0020	0.0125	0.0125	0.0125	0.0125	0.0124	0.0119	0.0124	0.0043
2	CS	0.0186	0.1121	0.0934	0.0934	0.0934	0.0900	0.0889	0.0550	0.0375
3	FDPSO	0.0469	0.2815	0.1876	0.1876	0.1876	0.1748	0.1789	0.0916	0.0891
4	ACO	0.1714	1.0289	0.5144	0.4285	0.3429	0.4312	0.4285	0.2458	0.2634
5	FLFBA	0.2142	1.2856	0.5144	0.4285	0.4285	0.4312	0.4285	0.2512	0.3088
6	MFO	0.5205	3.1231	0.5205	0.5205	0.5205	0.5205	0.5205	0.5205	0.5205

Table 9 Adjusted p-values (FRIEDMAN/ALIGNED FRIEDMAN/QUADE) in $D - 20$

i	*Algorithm*	*Unadjusted* p	p_{Bonf}	p_{Holm}	p_{Hoch}	p_{Homm}	p_{Holl}	p_{Rom}	p_{Finn}	p_{Li}
FRIEDMAN										
1	SFLA	1.9944E-8	1.1966E-7	1.1966E-7	1.1966E-7	1.1966E-7	1.1966E-7	1.1378E-7	1.1966E-7	3.7438E-7
2	CS	3.4346E-5	2.0607E-4	1.7173E-4	1.7173E-4	1.7173E-4	1.7172E-4	1.6331E-4	1.0303E-4	6.4433E-4
3	FDPSO	3.9829E-4	0.0023	0.0015	0.0015	0.0015	0.0015	0.0015	7.9642E-4	0.0074
4	ACO	0.0383	0.2299	0.1149	0.1149	0.0920	0.1106	0.1149	0.0569	0.4184
5	FLFBA	0.0613	0.3682	0.1227	0.1227	0.1227	0.1189	0.1227	0.0731	0.5353
6	MFO	0.9467	5.6803	0.9467	0.9467	0.9467	0.9467	0.9467	0.9467	0.9467
ALIGNED FRIEDMAN										
1	SFLA	4.2365E-6	2.5419E-5	2.5419E-5	2.5419E-5	2.5419E-5	2.5419E-5	2.4169E-5	2.5419E-5	5.5568E-6
2	CS	1.1876E-5	7.1259E-5	5.9382E-5	5.9382E-5	5.9382E-5	5.9381E-5	5.6472E-5	3.56295E-5	1.5577E-5
3	FDPSO	4.2310E-4	0.0025	0.0016	0.0016	0.0016	0.0016	0.0016	8.4602E-4	5.5465E-4
4	FLFBA	0.0050	0.0302	0.0151	0.0151	0.0151	0.0150	0.0151	0.0075	0.0065
5	ACO	0.0111	0.0670	0.0223	0.0223	0.0223	0.0222	0.0223	0.0133	0.0144
6	NBA	0.2375	1.4255	0.2375	0.2375	0.2375	0.2375	0.2375	0.2375	0.2375
QUADE										
1	SFLA	3.7716E-4	0.0022	0.0022	0.0022	0.0022	0.0022	0.0021	0.0022	0.0017
2	CS	0.0145	0.0873	0.0727	0.0727	0.0727	0.0706	0.0692	0.0430	0.0640
3	FDPSO	0.0333	0.2003	0.1335	0.1335	0.1335	0.1270	0.1273	0.0656	0.1356
4	ACO	0.1787	1.0727	0.5363	0.3646	0.3575	0.4461	0.3646	0.2558	0.4565
5	FLFBA	0.1823	1.0938	0.5363	0.3646	0.3646	0.4461	0.3646	0.2558	0.4613
6	NBA	0.7871	4.7230	0.7871	0.7871	0.7871	0.7871	0.7871	0.7871	0.7871

Table 10 Adjusted p-values (FRIEDMAN / ALIGNED FRIEDMAN / QUADE) in $D - 40$

i	*Algorithm*	*Unadjusted* p	p_{Bonf}	p_{Holm}	p_{Hoch}	p_{Homm}	p_{Holl}	p_{Rom}	p_{Finn}	p_{Li}
					FRIEDMAN					
1	SFLA	3.7226E-11	2.2335E-10	2.2335E-10	2.2335E-10	2.2335E-10	2.2335E-10	2.1237E-10	2.2335E-10	9.1461E-11
2	CS	1.3030E-7	7.8185E-7	6.5154E-7	6.5154E-7	6.5154E-7	6.5154E-7	6.1961E-7	3.9092E-7	3.2015E-7
3	FDPSO	2.6783E-7	1.6070E-6	1.0713E-6	1.0713E-6	1.0713E-6	1.0713E-6	1.0215E-6	5.3567E-7	6.5804E-7
4	FLFBA	6.5541E-4	0.0039	0.0019	0.0019	0.0019	0.0019	0.0019	9.8296E-4	0.0016
5	NBA	0.1415	0.8494	0.2831	0.2831	0.2831	0.2631	0.2831	0.1673	0.2580
6	MFO	0.5929	3.5578	0.5929	0.5929	0.5929	0.5929	0.5929	0.5929	0.5929
					ALIGNED FRIEDMAN					
1	SFLA	1.2540E-5	7.5243E-5	7.5243E-5	7.5243E-5	6.2702E-5	7.5241E-5	7.1544E-5	7.5241E-5	2.3699E-5
2	CS	1.7101E-5	1.0260E-4	8.5506E-5	8.5506E-5	8.5506E-5	8.5503E-5	8.1315E-5	7.5241E-5	3.2318E-5
3	FDPSO	2.7476E-4	0.0016	0.0010	0.0010	9.6738E-4	0.0010	0.0010	5.4945E-4	5.1900E-4
4	FLFBA	4.8369E-4	0.0029	0.0014	0.0014	0.0014	0.0014	0.0014	7.2544E-4	9.1328E-4
5	NBA	0.0563	0.3383	0.1127	0.1127	0.1127	0.1095	0.1127	0.0672	0.0963
6	ACO	0.4708	2.8251	0.4708	0.4708	0.4708	0.4708	0.4708	0.4708	0.4708
					QUADE					
1	SFLA	2.0993E-4	0.0012	0.0012	0.0012	0.0012	0.0012	0.0011	0.0012	0.0013
2	CS	0.0062	0.0374	0.0312	0.0312	0.0249	0.0308	0.0296	0.0186	0.0399
3	FDPSO	0.0092	0.0555	0.0370	0.0370	0.0370	0.0365	0.0352	0.0186	0.0581
4	FLFBA	0.0490	0.2943	0.1471	0.1471	0.1471	0.1400	0.1471	0.0726	0.2465
5	NBA	0.5170	3.1021	1.0340	0.8501	0.8501	0.7667	0.8501	0.5824	0.7752
6	ACO	0.8501	5.1006	1.0340	0.8501	0.8501	0.8501	0.8501	0.8501	0.8501

The Friedman test illustrates a significant performance of FSLA over the remaining algorithms while the Friedman aligned test validate its improvement for each post-hoc procedure considered except Bonferroni-Dunn which fails to emphasise the significant differences between them. It Should be noted that NBA and MFO are interchangeably omitted from the results due to their worst scores. The Finner and Li tests have the lowest p-values in the comparisons displaying the most powerful behaviour. Finally, the Quade test also confirms the order of the three first performing algorithms, i.e. FSLA, CS and FDPSO, and indicates different position between FLFBA and ACO. This result support the conclusion that, FSLA performed better than the remaining algorithms while FLFBA had an intermediate position in the scores tables.

5 Conclusions

This paper introduces an hybrid version of BA that is called FLFBA. The proposed algorithm is based on FC and LF techniques with DE strategies for solving optimisation problems. FLFBA has been validated using several benchmarks functions and compared to five algorithms that are CS, FDPSO, SFLA, ACO, MFO and NBA. Several non-parametric statistical tests using an average of the difference between the computed optimal fitness function value and the true global optimum function value were conducted in order to analyse the performance of the FLFBA algorithm.

FLFBA showed a distinguished performance in comparison to MFO and NBA but failed to provide similar or better results than FSLA, CS, and FDPSO. Also studies on the time taken to perform the iterations for each algorithm indicate that the FLFBA was in most of the cases much faster than FSLA, CS and FDPSO, but slower than NBA and MFO.

As shown with our experiments FLFBA uses a balanced combination of the advantages of the successful proprieties of FC, LF and DE which provides a superior performance than the NBA algorithms in terms of accuracy and efficiency.

The effects of parameters settings on the performance of FLFBA will be conducted in future studies. An investigation on the effects of different population update mechanisms or their combinations should be established. Furthermore, FLFBA will be applied to several real applications in sciences and engineering.

References

Bhateja, A.K., Bhateja, A., Chaudhury, S. and Saxena, P.K. (2015) ‘Cryptanalysis of vigenere cipher using cuckoo search’, *Applied Soft Computing*, January, Vol. 26, No. C, pp.315–324, ISSN: 1568-4946.

Cai, X., Gao, X-Z. and Xue, Y. (2016) ‘Improved bat algorithm with optimal forage strategy and random disturbance strategy’, *Int. J. Bio-Inspired Comput.*, January, Vol. 8, No. 4, pp.205–214, ISSN 1758-0366.

Chakri, A., Khelif, R., Benouaret, M. and Yang, X-S. (2017) 'New directional bat algorithm for continuous optimization problems', *Expert Systems with Applications*, Vol. 69, Supplement C, pp.159–175.

Couceiro, M. and Sivasundaram, S. (2016) 'Novel fractional order particle swarm optimization', *Applied Mathematical Computing*, June, Vol. 283, No. C, pp.36–54.

Couceiro, M.S., Rocha, R.P., Ferreira, N.M.F. and Machado, J.A.T. (2012) 'Introducing the fractional-order Darwinian PSO', *Signal, Image and Video Processing*, September, Vol. 6, No. 3, pp.343–350.

Cui, Z., Li, F. and Zhang, W. (2018) 'Bat algorithm with principal component analysis', *International Journal of Machine Learning and Cybernetics*, March 2019, Vol. 10, No. 3, pp.603-622.

Derrac, J., García, S., Molina, D. and Herrera, F. (2011) 'A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms', *Swarm and Evolutionary Computation*, Vol. 1, No. 1, pp.3–18.

Dorigo, M., Maniezzo, V. and Colorni, A. (1996) 'Ant system: optimization by a colony of cooperating agents', *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, February, Vol. 26, No. 1, pp.29–41.

Edelman, D.C. (2010) 'Branding in the digital age: you're spending your money in the wrong places', *Harvard Business Review*, Vol. 88, No. 12, pp.16–23.

Eusuff, M., Lansey, K. and Pasha, F. (2006) 'Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization', *Engineering Optimization*, Vol. 38, No. 2, pp.129–154.

Fister, I., Fister, D. and Yang, X-S. (2013) 'A hybrid bat algorithm', *CoRR*, abs/1303.6310.

Gandomi, A.H. and Yang, X-S. (2014) 'Chaotic bat algorithm', *Journal of Computational Science*, Vol. 5, No. 2, pp.224–232.

Jordehi, A.R. (2015) 'Chaotic bat swarm optimisation (CBSO)', *Appl. Soft Comput.*, January, Vol. 26, No. C, pp.523–530.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983) 'Optimization by simulated annealing', *Science*, Vol. 220, No. 4598, pp.671–680.

Machado, J.A. (2002) 'System modeling and control through fractional-order algorithms', *Nonlinear Dynamics, Chaos, Control, and Their Applications to Engineering Sciences*, Vol. 4, pp.99–116.

Mantegna, R.N. (1994) 'Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes', *Phys. Rev. E*, May, Vol. 49, No. 5, pp.4677–4683.

Meng, X.B., Gao, X.Z., Liu, Yu and Zhang, H. (2015) 'A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization', *Expert Systems with Applications*, Vol. 42, Nos. 17–18, pp.6350–6364.

Michelitsch, T.M., Collet, B.A., Riascos, A.P., Nowakowski, A.F. and Nicolleau, F.C.G.A. (2017) 'Fractional random walk lattice dynamics', *Journal of Physics A: Mathematical and Theoretical*, Vol. 50, No. 5, p.055003.

Mirjalili, S. (2015) 'Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm', *Knowledge-Based Systems*, Vol. 89, No. C, pp.228–249.

Oldham, K.B. and Spanier, J. (1974) 'Chapter 1: Introduction', in Oldham, K.B. and Spanier, J. (Eds.): *The Fractional Calculus*, Vol. 111 of *Mathematics in Science and Engineering*, pp.1 – 24, Elsevier.

Qu, B.Y., Liang, J.J., Wang, Z.Y., Chen, Q. and Suganthan, P.N. (2016) 'Novel benchmark functions for continuous multimodal optimization with comparative results', *Swarm and Evolutionary Computation*, Vol. 26, No. C, pp.23–34.

Rabei, E.M., Altarazi, I.M.A., Muslih, S.I. and Baleanu, D. (2009) 'Fractional WKB approximation', *Nonlinear Dynamics*, July, Vol. 57, No. 1, pp.171–175.

Riascos, A.P. and Mateos, J.L. (2014) 'Fractional dynamics on networks: emergence of anomalous diffusion and Lévy flights', *Phys. Rev. E*, September, Vol. 90, No. 3, p.032809.

Sabatier, J., Agrawal, O.P. and Machado, J.A.T. (2007) *Advances in Fractional Calculus: Theoretical Developments and Applications in Physics and Engineering*, Springer, Netherlands.

Viswanathan, G.M., Afanasyev, V., Buldyrev, S.V., Murphy, E.J., Prince, P.A. and Stanley, H.E. (1996) 'Lévy flight search patterns of wandering albatrosses', *Nature*, Vol. 381, No. 6581, pp.413–415.

Wang, Y., Wang, P., Zhang, J., Cui, Z., Cai, X., Zhang, W. and Chen, J. (2019) 'A novel bat algorithm with multiple strategies coupling for numerical optimization', *Mathematics*, Vol. 7, No. 2, Article ID 135.

Xie, J., Zhou, Y. and Chen, H. (2013) 'A novel bat algorithm based on differential operator and Lévy flights trajectory', *Computational Intelligence and Neuroscience*, Vol. 2013, Article ID 453812.

Yang, X.S. and Deb, S. (2009) 'Cuckoo search via Lévy flights', *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*, December, pp.210–214.

Yang, X-S. (2001) 'A new metaheuristic bat-inspired algorithm', In González, J.R., Pelta, D.A., Cruz, C., Terrazas, G. and Krasnogor, N. (Eds.): *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp.65–74, Springer Berlin Heidelberg, Berlin, Heidelberg.

Yang, X-S. (2014) *Nature-Inspired Optimization Algorithms*, 1st ed., Elsevier Science Publishers B.V., Amsterdam, The Netherlands.

Yilmaz, S. and Küçüksille, E.U. (2015) 'A new modification approach on bat algorithm for solving optimization problems', *Applied Soft Computing*, March, Vol. 28, No. C, pp.259–275.

Zhou, Y., Ionescu, C. and Machado, J.A.T. (2015) 'Fractional dynamics and its applications', *Nonlinear Dynamics*, June, Vol. 80, No. 4, pp.1661–1664.