Applied Soft Computing 23 (2014) 333–345

Contents lists available at ScienceDirect

Applied Soft Computing

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

A novel particle swarm optimization algorithm with Levy flight

Hüseyin Haklı[*], Harun Uğuz

Department of Computer Engineering, Selcuk University, 42075 Konya, Turkey

ARTICLE INFO

Article history:
Received 5 August 2013
Received in revised form 20 June 2014
Accepted 22 June 2014
Available online 28 June 2014

Keywords:
Particle swarm optimization
Levy flight
Levy distribution
Optimization

ABSTRACT

Particle swarm optimization (PSO) is one of the well-known population-based techniques used in global optimization and many engineering problems. Despite its simplicity and efficiency, the PSO has problems as being trapped in local minima due to premature convergence and weakness of global search capability. To overcome these disadvantages, the PSO is combined with Levy flight in this study. Levy flight is a random walk determining stepsize using Levy distribution. Being used Levy flight, a more efficient search takes place in the search space thanks to the long jumps to be made by the particles. In the proposed method, a limit value is defined for each particle, and if the particles could not improve self-solutions at the end of current iteration, this limit is increased. If the limit value determined is exceeded by a particle, the particle is redistributed in the search space with Levy flight method. To get rid of local minima and improve global search capability are ensured via this distribution in the basic PSO. The performance and accuracy of the proposed method called as Levy flight particle swarm optimization (LFPSO) are examined on well-known unimodal and multimodal benchmark functions. Experimental results show that the LFPSO is clearly seen to be more successful than one of the state-of-the-art PSO (SPSO) and the other PSO variants in terms of solution quality and robustness. The results are also statistically compared, and a significant difference is observed between the SPSO and the LFPSO methods. Furthermore, the results of proposed method are also compared with the results of well-known and recent population-based optimization methods.

Introduction

In recent years, many nature-inspired algorithms have been developed to solve complex and difficult non-linear problems. One of the reasons of this is that it take too much time to solve real world problems with traditional optimization methods, and that they cannot be solved effectively. Nature-inspired algorithms which are also known as swarm intelligence algorithms have been developed through inspiration from the behaviors of the living things in the nature. For instance, artificial bee colony optimization algorithm was developed by being motivated from bee colonies [1], ant colony optimization simulates the behavior of real ants between nest and food sources [2]. Being inspired social behaviors of a flock of fishes or birds, PSO was first proposed in 1995 by Eberhart and Kennedy [3].

PSO is a robust stochastic optimization and population-based technique based on the movement and intelligence of swarms. Easy-to-perform methods and negligible parameter settings have recently made the algorithm very popular and started to be applied in many fields.

By virtue of these advantages, PSO is effectively used for function optimization [4], filter design [5,6], fuzzy PID control [7], predicting power allocation [8], feature selection [9,10], artificial neural networks [11], image segmentation [12], scheduling and sequencing problems [13,14], logic circuit design [15,16], human tremor analysis [17], other scientific, engineering problems, etc.

In order to overcome the problem that the basic PSO is lack of producing good results due to its deficiency in velocity control mechanism and to ensure the balance between exploration and exploitation, Shi and Eberhart [18] added the inertia weight to the velocity update procedure in the basic PSO. Inertia weight enabled the PSO algorithm work more effectively by ensuring the balance between global search and local search.

In the original PSO algorithm, update procedures may be performed according to the best value found by each particle until the iteration at that moment (*pbest*) and the best value found by all particles until the iteration at that moment (*gbest*). The principle behind the PSO is that each particle owns the learning ability from itself (*pbest*) and its best neighbor (*gbest*). The PSO performs velocity change through being affected by both local and global conditions. Although this circumstance, since particles resemble each other after a certain number of iterations (loss of diversity), velocity changes drop to very little values and lead to loss of global

[*] Corresponding author. Tel.: +90 332 223 37 28; fax: +90 332 241 06 35.
E-mail address: hhakli@selcuk.edu.tr (H. Haklı).

http://dx.doi.org/10.1016/j.asoc.2014.06.034
1568-4946/© 2014 Elsevier B.V. All rights reserved.

search ability. This causes trapping of the PSO in local minima, one of its biggest problems. There are many studies in the literature aimed at preventing this problem (such as change of velocity updates or using in hybridization with other algorithms). Liang et al. [19] diversified the swarm and targeted to prevent early convergence by making velocity update using *gbest*, or particle's best or *pbests* of different particles and selecting one of them randomly instead of learning from *pbest* and *gbest* of the particles in the original PSO. Xinchao [20] changed the velocity update procedure to prevent loss of diversity, and proposed perturbed particle swarm algorithm based on the perturbed *gbest* updating strategy. In another study, different velocity update techniques were combined, and it was ensured to continue use of the technique by which update is made better [21]. Tsoulos [22] added stopping rule, similarity check and a conditional application of some local search method modifications to enhance velocity and effectiveness of PSO. Unlike the velocity update changes, the PSO that performed local search well was used in hybridization with other nature-inspired algorithms, thus, hybrid algorithms performing both global search and local search were proposed [23–25]. Many other PSO variants such as ILPSO, GPSO, Orthogonal PSO, etc. were proposed to solve the premature convergence problem of PSO.

To strengthen global search of PSO and overcome the problem of being trapped in local minima as in the above mentioned methods, PSO was combined with Levy flight in this study. A Levy flight is a class of random walk, which is generalized Brownian motion to include non-Gaussian randomly distributed step sizes for the distance moved [26]. There are many natural and artificial facts that can be depicted by Levy flight, such as fluid dynamics, earthquake analysis, the diffusion of fluorescent molecules, cooling behavior, noise, etc. [27]. Levy flight was also used by Pereyra and Hadj in the field of Ultrasound in Skin Tissue [28], and by Al-teemy [26] in Ladar Scanning. Levy flight also took an important part in many fields in computer sciences besides these fields. Levy flight is used for Internet Traffic Models by Terdik and Gyres [29], Delay and Disruption Tolerant Network by Chen [27], Multi Robot Searching procedure by Sutantyo et al. [30] and Rhee [31] utilized Leyv walk on human mobility fields.

Moreover, Levy flight that resembles food searching path of many animals like albatross, bumblebees and deer was added to nature-inspire algorithms to ensure improvement of the algorithms [32,33]. Yang and Deb [34] used Levy flight distribution to create new cuckoo in Cuckoo Search. Also, Yang [35] introduced a new version of Firefly Algorithm-FA, Levy-flight Firefly algorithm (LFA), which combined Levy-flight with the search strategy via the Firefly for improving the randomization of FA. In their Evolution Algorithm, Lee and Yao [36] created 4 different states of β parameter of Levy flight and 4 candidate solutions, and took the one that gave the best result among these candidate values and used it to perform the mutation procedure. Also, Levy flight was used as diversification tools for ant colony optimization [37–39].

In this study, the long jumps are performed through Levy distribution, and more effective use of the search space compared to the PSO is ensured. A limit value is determined for each particle, and in case the particles could not improve self-solutions as much as the limit value given, the particles are redistributed with Levy flight such that *gbest* would be affected, and being trapped in local minima is prevented. As in many studies previously conducted to improve the PSO, in the proposed method, it is ensured with random walks that PSO performs global search more effective. It is intended to be more consistent by ensuring its being affected by *gbest* while performing these random walks.

When compared the proposed method that are examined on various types of benchmark functions with the SPSO, it is observed to be effective particularly for solving multimodal functions and as the dimensions increased, and to converge earlier. The results are also statistically compared with non-parametric Wilcoxon test, and a significant difference is seen between the LFPSO and the SPSO methods.

The rest of the paper is divided as follows. In PSO and Levy Flights, original PSO algorithm and Levy flight method are presented. The proposed approach is detailed in The Proposed Algorithm LFPSO. In Experiments and Results, the experimental results and comparison of the methods are presented. In Results and Discussion provides discussion of the present work. As a final, the paper is concluded with the future works.

PSO and Levy flights

Original PSO algorithm

The PSO algorithm has been proposed through inspiration from social behaviors of the individuals in bird and fish swarms [3]. Individuals in the swarms are referred to as particles, and each particle consists of *D-dimensional* values. For a *D-dimensional* state, position and velocity expressions of particle *i* are represented as follows.

$X_i = \{X_{i1}, X_{i2}, X_{i3}, \ldots, X_{iD}\}$ and $V_i = \{V_{i1}, V_{i2}, V_{i3}, \ldots, V_{iD}\}$

Intelligent interaction among the swarm is provided with best value of each particle (*pbest*) and best value of all particles (*gbest*) until at the current iteration. For a *D-dimensional* search space, *pbest* of particle *i* is represented as $pbest = \{P_{i1}, P_{i2}, P_{i3}, \ldots, P_{iD}\}$, *gbest* is represented as $gbest = \{G_1, G_2, G_3, \ldots, G_D\}$. Since PSO will perform update procedures according to these values, *pbest* values for each particle and the *gbest* value, which is the best value for the entire swarm, should be kept. PSO consists of two stages as beginning and calculation. In the beginning stage, all particles are distributed randomly in the search space within the determined boundaries. In calculation stage, velocities and positions of the particles are updated. Velocity of a particle is calculated as follows [3]:

$$V_{i,d}^{t+1} = V_{i,d}^{t} + c_1 rand_1 (pbest_{i,d}^{t} - X_{i,d}^{t}) + c_2 rand_2 (gbest_{d}^{t} - X_{i,d}^{t}) \tag{1}$$

where $V_{i,d}^{t+1}$ is velocity of particle *i* at iteration *t + 1* with respect to the *d*th dimension, $X_{i,d}^{t}$ is position value of the *i*th particle with respect to the *d*th dimension, *c1* is cognitive weighting factor, *c2* is social weighting factors are acceleration coefficients, *r1* and *r2* values are stochastic components of the algorithm, which are in the interval [0, 1]. *c1* and *c2* values which are generally determined identical and as 2 are set as desired, and it may be ensured that particles affected more either locally or globally. While the fact that acceleration coefficients take big values causes the particles to move away from each other and separate, their taking small values causes limitation of the movements of the particles, and not being able to scan the solution space adequately [40].

V_{max} and V_{min} parameters may be set for the velocity values determined for each particle to prevent occurrence of big changes on the particles or constant limit excesses. In this study, V_{max} and V_{min} was set as 20% of the upper and lower limits.

Inertia weight was added to PSO by Shi and Eberhart in 1998 [18] to provide the balance between exploitation and exploration:

$$V_{i,d}^{t+1} = w^t V_{i,d}^{t} + c_1 rand_1 (pbest_{i,d}^{t} - X_{i,d}^{t}) + c_2 rand_2 (gbest_{d}^{t} - X_{i,d}^{t}) \tag{2}$$

Inertia weight controls effect of previous velocity increases of the particles on the velocity value, and takes part in providing the balance between global search and local search. When the inertia weight takes large values, global search is more suitable and a small inertia weight facilitates local search [19]. Shi and Eberhart [18] proposed a linearly decreasing inertia weight over the course of search. Usually, *max* and *min* values are determined for inertia, too. In this study, inertia update is made as follows:

$$w = \frac{(Max\_iter - iter)}{Max\_iter} \quad (3)$$

where *Max_it* refers to maximum number of iteration and *iter* is the current iteration. New position values are obtained by adding the velocity updates determined by the formula given in Eq. (4) to the particles:

$$X_{i,d}^{t+1} = X_{i,d}^{t} + V_{i,d}^{t+1} \quad (4)$$

After making the velocity updates, new fitness values of the particles are calculated, and if necessary, *pbest* and *gbest* updates are performed, and the same procedure is continued until the stop criteria are provided.

Levy flights

Levy flights (or Levy motion) [41] is a class of non-Gaussian random processes whose random walks are drawn from Levy stable distribution. This distribution is a simple power-law formula $L(s) \sim |s|^{-1-\beta}$ where $0<\beta<2$ is an index. Mathematically speaking, a simple version of Levy distribution can be defined as [34,42]:

$$L(s,\gamma,\mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} \exp\left[-\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{3/2}} & \text{if } 0<\mu<s<\infty \\ 0 & \text{if } s \le 0 \end{cases}, \quad (5)$$

where μ parameter is location or shift parameter, $\gamma>0$ parameter is scale (controls the scale of distribution) parameter.

In general, Levy distribution should be defined in terms of Fourier transform

$$F(k) = \exp[-\alpha |k|^{\beta}], \quad 0<\beta\le 2, \quad (6)$$

where α is a parameter within $[-1, 1]$ interval and known as skewness or scale factor. An index of o stability $\beta€(0, 2]$ is also referred to as Levy index. The analytic form of the integral is not known for general β except for a few special cases.

In particular (or special case, for the case of), for $\beta=1$, the integral can be carried out analytically and is known as the Cauchy probability distribution.

$$F(k) = \exp[-\alpha |k|], \quad (7)$$

Another special case when $\beta=2$, the distribution correspond to Gaussian distribution.

$$F(k) = \exp[-\alpha k^2], \quad (8)$$

β and α parameters take a major part in determination of the distribution whereas γ and μ parameters take a minor part. The parameter β controls the shape of the probability distribution in such a way that one can obtain different shapes of probability distribution, especially in the tail region depending on the parameter β. Thus, the smaller β parameter causes the distribution to make longer jumps since there will be longer tail [36]. The sign of the skewness parameter α indicates the skew direction, positive to the right, and negative to the left. When $\alpha=0$, the distribution is symmetric. The last two parameters are the width γ and the shift μ of the distribution peaks [26]. The different values of the β parameter change the distribution. It makes longer jumps for smaller values whereas it makes shorter jumps for bigger values.

The proposed algorithm LFPSO

Though there are many different PSO variants in the literature, PSO's problems of premature convergence and generating inefficient results are still persisting. Levy flights method is used for solving these problems and enable PSO generate more efficient results. By this method, it is ensured that PSO, which is unable to perform global search well, perform global search more effectively and not be trapped in local minima. In LFPSO method, two changes are made in comparison to the PSO method.

- A limit value is set for every particle, and this limit value is incremented by 1 in case the particles could not improve self-solutions for each new iteration.
- The particles exceeding the limit value are redistributed in the search space using Levy flight (using Levy distribution) method.

Fig. 1, X_{max} and X_{min} represent maximum and minimum values of search range, respectively. V_{min} and V_{max} represent maximum and minimum limit of the velocity increase to be made, respectively. *NP* is number of particles, *D* is the dimension of benchmark function, *c1* and *c2* are acceleration coefficients, *trial* sets the limit value for each particle and *Max_iter* refers to maximum number of iterations.

As in the PSO, the particles are first randomly distributed to the search space, and the fitness values of particles are calculated and *pbest* for the particles and *gbest* for the swarm are determined by using the fitness values. Unlike PSO, before making velocity update, limit excess checking is performed for each particle. If the current particle has not exceeded the limit value, velocity updates are performed normally and its position is updated. If the particle has exceeded the limit value, the particle is distributed in the search space by Levy flight method, and following the distribution procedure, positions value exceeding the boundaries in the search space determined are brought to the bound values.

After these procedures are completed, the fitness value is calculated for the new particle determined, if the new fitness value calculated is better for *pbest* of the particle, they are assigned as *pbest* and its limit value is reset, if there is no improvement, limit value is incremented by 1 for the particle. Likewise, if the new fitness value is smaller (for minimize problems) than the *gbest* value, it is assigned as the particle's *gbest* value. Same procedures are performed until the number of iteration is equal to *Max_iter*.

Let us review in some more detail how distribution is performed using Levy flight. By Levy flight, new state of the particle is calculated as [34]:

$$X^{t+1} = X^{t} + \alpha \oplus \text{Levy}(\beta) \quad (9)$$

α is the step size which should be related to the scales of the problem of interest. In the proposed method α is random number for all dimensions of particles.

$$X^{t+1} = X^{t} + \text{random}(\text{size}(D)) \oplus \text{Levy}(\beta) \quad (10)$$

The product $\oplus$ means entry-wise multiplications.

A non-trivial scheme of generating step size *s* samples are discussed in detail in [34,42], it can be summarized as

$$s = \text{random}(\text{size}(D)) \oplus \text{Levy}(\beta) \sim 0.01 \frac{u}{|v|^{1/\beta}} (x_j^t - gbest^t) \quad (11)$$

where *u* and *v* are drawn from normal distributions. That is

$$u \sim N(0, \sigma_u^2) v \sim N(0, \sigma_v^2) \quad (12)$$

with

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left[\left(\frac{1+\beta}{2}\right)\right]\beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \quad \sigma_v = 1 \quad (13)$$

Here Γ is standard Gamma function.*s* value with *D-dimension* obtained by Eq. (11) is added to update of position X_i particle finds the position values of new particle. The fitness value is evaluated for

```
Initialize the parameters ($PS, D, X_{min}, X_{max}, Max\_it, c1, c2, V_{min}, V_{Max}$)
trial(keeps the limit value for each particle ) =0;
Randomly initialize the positions of all particles $X_i=\{X_{i1}, X_{i2}, X_{i3}, ..., X_{iD}\}$
Evaluate the fitness values
Set $X$ to be pbest
Set the particle with best fitness to be gbest
while iter < Max_iter do
    for i = 1:PS
        if trial(i) < limit  (if current particle is not exceeded limit value)
            Update the velocity $V_i$ of particle using Eq. (2)
            Update the position of particle using Eq. (4)
        else (if current particle is exceeded limit value)
            trial(i)=0
            Update the particle positions using Levy flight Method
            Positions value exceeding the boundaries in the search space determined are brought to the
            bound values($X_{min}, X_{max}$).
        end if
        Evaluate fitness value for new particle $X_i$
        if $X_i$ is better than $pbest_i$
            trial(i)=0;
            Set $X_i$ to be $pbest_i$
        else
            trial(i)=trial(i)+1
        end if
        if $X_i$ is better than gbest
            Set $X_i$ to be gbest
        end if
    end for
    iter = iter + 1
end while
```

Fig. 1. The detailed LFPSO algorithm.

this new particle, if particle obtains better result than its *pbest* value, updates *pbest* and trial value of this particle is set 0, otherwise, trial is incremented by 1. Then algorithm continues until the stopping criterion is met. Fig. 2 shows flowchart of LFPSO algorithm.

One of the important points to be considered while performing distribution by Levy flights is the value taken by the β parameter. As stated before, Yang and Deb [34] mentioned that the β parameter gave different results at different values in the trials conducted in the study named multiobjective cuckoo search for design optimization. Accordingly, it can be concluded that a different β parameter for each benchmark function gave a more effective result. Moreover, in the Evolutionary Algorithms with Adaptive Levy Mutations study of Lee [36], different constant values were set of the β parameter, procedures were conducted by calculating the distribution for each of these values and selecting the best of the offspring produced. As seen in these two examples, β parameter substantially affects distribution. In this study, no constant value is taken for the β parameter, but a random value in the (0, 2] interval is taken for Levy flight distribution procedure. PSO performs velocity change through being affected by both local and global conditions. Although this circumstance, since particles resemble each other after a certain number of iterations (loss of diversity), velocity changes drop to very little values and lead to loss of global search ability. The LFPSO intend to prevent loss of diversity and improve the global search ability. It performs long jumps for small values of the β parameter, and this eliminates the global search deficiency in PSO prevents which we always mention about, and prevents being trapped in local minima.

In contrast with the Levy flight, in Cuckoo optimization stepsize is calculated by subtracting two random cuckoo in Levy flight distribution, whereas, in this study *gbest* is subtracted from current particle. Thus, it is ensured that the best particle value (*gbest*) is maintained without change. If the β value randomly so taken takes small values, it allows the particle perform very long jumps in the search space and prevents constantly being trapped in local minima, if big values are attained, it continues to derive new values around *gbest*. The randomization can be more efficient as the steps obey a Levy distribution which can be approximated by a power law; therefore, the steps consist of many small steps and occasionally large-step, long-distance jumps [34]. In this study, since the β parameter will be determined randomly for each particle, it was ensured that big jumps occur more. In explaining the Cuckoo optimization in their Multiobjective Cuckoo Search for Design Optimization study, Yang and Deb have stated: "*When compared with PSO, these long jumps substantially increase search efficiency of the cuckoo search in several cases, particularly for multimodal and non-linear problems comprising many local minima but one global minimum*", indicated this as the weakness of PSO, and stated that this was an advantage for Cuckoo Optimization. In this study, global search was strengthened using random walk with Levy flight to eliminate the weakness of PSO, its being trapped in local minimums was prevented, and it was observed to give more successful results particularly for multimodal functions.

Experiments and results

All simulations are conducted in a Windows 7 Professional OS environment using a Intel i5, 2.4 GHz, 4 GB RAM and the codes were implemented in Matlab 7.9.

Results and comparison of SPSO and LFPSO

Owing to weakness and the problem of the premature convergence in the basic PSO, one of state-of-the-art PSO proposed by Omran [43] is used for the experimental results and comparison. For more information and the program code can be found at http://www.particleswarm.info/Programs.html. SPSO is the current standard which is improved by using a random topology.

Parameter settings for the methods

For the SPSO algorithm, the number of particles is determined with using the dimension of function and Eq. (14) shows this formula.

$$\mathrm{NP} = \mathrm{floor}(10 + 2 * \mathrm{sqrt}(D)) \tag{14}$$

The algorithms are examined on all benchmark functions with dimension (D) of 30 and 50. For the SPSO automatically computes a population size of 20 for 30-D and 24 for 50-D. Experiments are implemented with the population size (NP) of 40 for LFPSO. As a

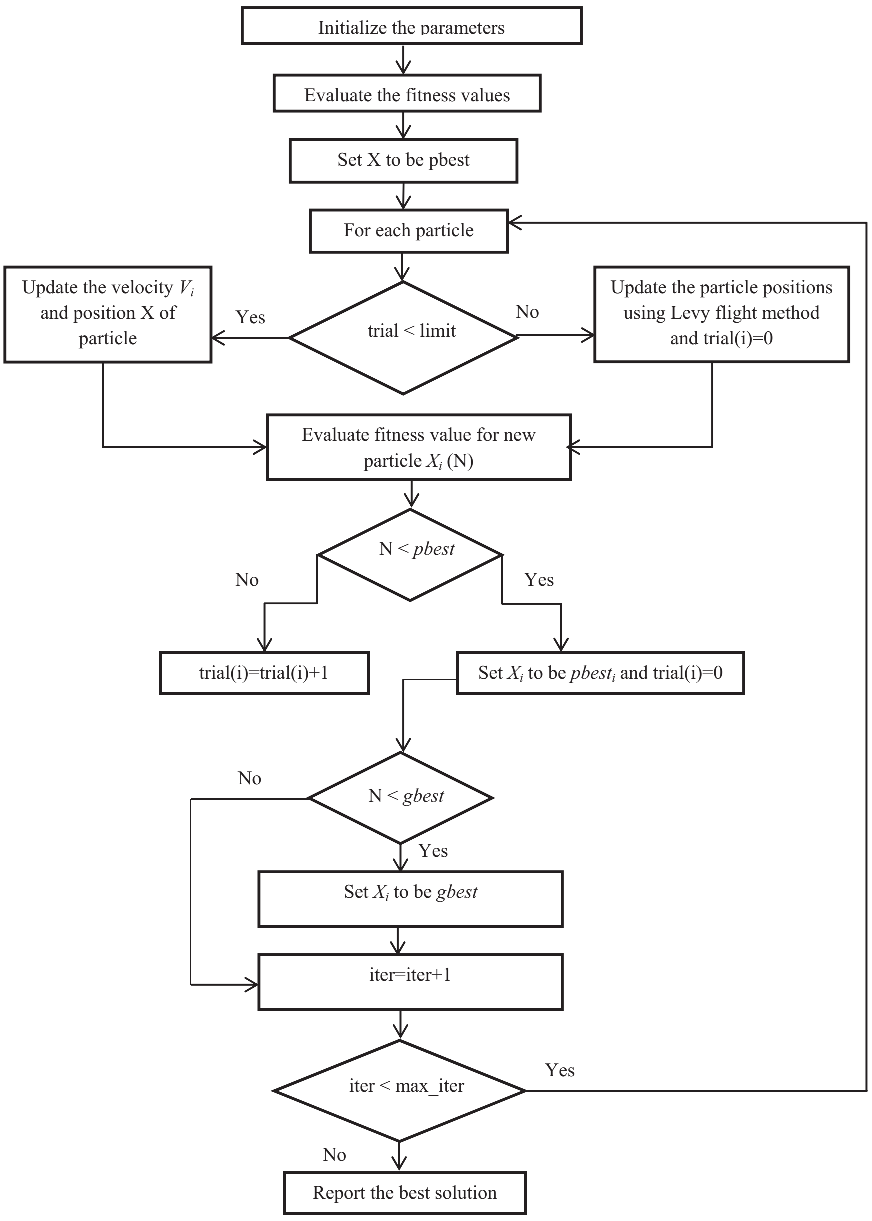
Initialize the parameters
Evaluate the fitness values
Set X to be pbest
For each particle
Update the velocity V_i and position X of particle
Yes
trial < limit
No
Update the particle positions using Levy flight method and trial(i)=0
Evaluate fitness value for new particle X_i (N)
N < pbest
No
Yes
trial(i)=trial(i)+1
Set X_i to be $pbest_i$ and trial(i)=0
No
N < gbest
Yes
Set X_i to be gbest
iter=iter+1
iter < max_iter
Yes
No
Report the best solution

Fig. 2. Flowchart of LFPSO algorithm.

stopping criterion is used the maximum number of function evaluations (FEs) and all the algorithms use the same (FEs) 2×10^5 in each run for each test function. In order to make comparison clear, the values below 10^{-18} are assumed to be 0. Inertia weight is evaluated with Eq. (15) for SPSO and Eq. (3) for LFPSO.

$$w = \frac{1}{(2 * \log(2))} \tag{15}$$

The particles are initialized in the initialization range, where X_{max} and X_{min} are the maximum and minimum value in search range. SPSO have not any criteria for the V_{min} and V_{max}, while LFPSO is used as a speed limit of 20% of the search space. For LFPSO, in addition to indication of β parameter, it is set as a random number in the interval (0, 2] for each new distribution procedure. It is found appropriate to select the limit value for LFPSO as 10 as a result of the experiments conducted. Table 1 shows the detailed parameter settings for SPSO and LFPSO.

The performance of LFPSO is compared with the SPSO. Moreover, in order to understand whether there was a statistical difference between the results obtained, the results were subjected to Wilcoxon test, and their states were indicated in result tables.

Benchmark functions

Twenty one benchmark functions listed in Table 2 are used in experimental tests. Well-known benchmark functions in Table 2, have different characteristics. The first characteristic includes

Table 1
Parameter settings of SPSO and LFPSO.

	SPSO	LFPSO
Population size	30 Dimension → 20 50 Dimension → 24	30 Dimension → 40 50 Dimension → 40
FEs	200,000	200,000
Inertia weight	$w = 0.7213$	$w = (Max\_iter - iter)/Max\_iter$
c_1, c_2	$c_1 = c_2 = 1{,}1931$	$c_1 = c_2 = 2$
Limit	–	10

Table 2
Benchmark functions (C: characteristic, U: unimodal, M: multimodal, R: rotated).

No	Function	Search range	Initialization range	C	Formulation
1	Sphere	[−100, 100]	[−100, 50]	U	$f_1 = \sum_{i=1}^{n} x_i^2$
2	Schwefel2.22	[−10, 10]	[−10, 5]	U	$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
3	Rosenbrock	[−10, 10]	[−10, 10]	U	$f_3 = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$
4	Noise	[−1.28, 1.28]	[−1.28, 0.64]	U	$f_4 = \sum_{i=1}^{n} i x_i^4 + \mathrm{random}[0,\ 1)$
5	Schwefel2.26	[−500, 500]	[−500, 500]	M	$f_5 = 418.98288727243369 * n - \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$
6	Rastrigin	[−5.12, 5.12]	[−5.12, 2]	M	$f_6 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$
7	Ackley	[−32, 32]	[−32, 16]	M	$f_7 = -20\exp\left\{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right\} - \exp\left\{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right\} + 20 + e$
8	Griewank	[−600, 600]	[−600, 200]	M	$f_8 = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
9	Penalized1	[−50, 50]	[−50, 25]	M	$f_9 = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{1}{4}(x_i + 1) u_{x_i,a,k,m} = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(x_i - a)^m & x_i < -a \end{cases}$
10	Penalized2	[−50, 50]	[−50, 25]	M	$f_{10} = \frac{1}{10}\{\sin^2(\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2[1 + \sin^2(2\pi x_{i+1})]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$
11	Rotated Schwefel	[−500, 500]	[−500, 500]	R	$f_{11} = 418.9828 * \mathrm{n} - \sum_{i=1}^{n} z_i,$ where $z_i = \begin{cases} y_i \sin(\sqrt{\lvert y_i \rvert}), \text{ if } \lvert y_i \rvert \le 500 \\ 0, \text{ otherwise} \end{cases}, y_i = y_i' + 420.96,$ where $y' = M * (x - 420.96), M$ is an orthogonal matrix
12	Rotated Rastrigin	[−5.12, 5.12]	[−5.12, 2]	R	$f_{12} = \sum_{i=1}^{n}\left[y_i^2 - 10\cos(2\pi y_i) + 10\right]$ where $y = M * x, M$ is an orthogonal matrix
13	Rotated Ackley	[−32, 32]	[−32, 16]	R	$f_{13} = -20\exp\left\{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2}\right\} - \exp\left\{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi y_i)\right\} + 20 + e$ where $y = M * x, M$ is an orthogonal matrix
14	Rotated Griewank	[−600, 600]	[−600, 200]	R	$f_{14} = \frac{1}{4000}\sum_{i=1}^{n} y_i^2 - \prod_{i=1}^{n}\cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$ where $y = M * x, M$ is an orthogonal matrix
15	Sum Square	[−10, 10]	[−10, 10]	U	$f_{15} = \sum_{i=1}^{n} i x_i^2$
16	Step	[−100, 100]	[−100, 100]	U	$f_{16} = \sum_{i=1}^{n}\left(\lfloor x_i + 0.5 \rfloor\right)^2$
17	Quartic	[−1.28, 1.28]	[−1.28, 1.28]	U	$f_{17} = \sum_{i=1}^{n} i x_i^4$

Table 2 (*Continued*)

No	Function	Search range	Initialization range	C	Formulation
18	Levy	[−10, 10]	[−10, 10]	M	$f_{18}=\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+\sin^2(3\pi x_1)+ \lvert x_n-1\rvert\,[1+\sin^2(3\pi x_n)]$
19	Schaffer	[−100, 100]	[−100, 100]	M	$f_{19}=0.5+\frac{\sin^2(\sqrt{\sum_{i=1}^{n}x_i^2})-0.5}{(1+0.001(\sum_{i=1}^{n}x_i^2))^2}$
20	Alpine	[−10, 10]	[−10, 10]	M	$f_{20}=\sum_{i=1}^{n}\left\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\right\rvert$
21	Non-continuous Rastrigin	[−5.12, 5.12]	[−5.12, 5.12]	M	$f_{21}=\sum_{i=1}^{n}\left[y_i^2-10\cos(2\pi y_i)+10\right]\ y_i=\left\{\begin{array}{ll} x_i & \lvert x_i\rvert<\frac{1}{2} \\ \frac{\mathrm{round}(2x_i)}{2} & \lvert x_i\rvert\geq\frac{1}{2}\end{array}\right\}$

seven unimodal functions, where functions 1, 2, 3, 4, 15, 16 and 17 are unimodal, function 3 (Rosenbrock) is unimodal in a 2-D or 3-D search space but can be treated as a multimodal function in high-dimensional cases [44]. The second characteristic includes 10 complex multimodal functions with high dimensionality. The last characteristic includes four rotated multimodal functions. Since SPSO's being trapped in local minima is its most known problem, when selecting the benchmark functions, benchmark functions are determined mainly as multimodal. It was intended to show that the proposed method with this determination prevented from being trapped in local minima and derived better results. While multimodal functions have more than one local minimum, unimodal functions have only one local minimum. The determined benchmark functions are numbered *F1–F21* given in Table 2. All the functions are to be minimized. Table 2 shows search range, initialization range, characteristic (C) and formulation of the functions.

Experimental results and comparison of SPSO and LFPSO

As indicated in Table 2, 21 functions are used. All benchmark functions are operated 30 and 50 dimensions. Wilcoxon test is conducted to understand whether there is a statistically significant difference between the results. In order to make comparison clear, the values below 10^{-18} (error rate) are assumed to be 0. SPSO is compared with LFPSO method. In Table 3, in case Wilcoxon test was at 0.05 significance level, + mark is used, otherwise, − mark is used; no mark is used in case all results showed the same value or optimum value.

Table 3 shows the mean result of 30 runs with 200,000 FEs of all benchmark functions with dimension 30 and 50. The information about the mean optimum solution, standard deviation and Wilcoxon test result by SPSO and LFPSO in 30 runs over benchmark functions are given in Table 2. The best mean result and the best standard deviations of the benchmark functions obtained by the algorithms are shown in bold.

For the 30 dimension, LFPSO algorithm acquires better results compared to PSO algorithm for the 14 benchmark functions in terms of solutions. SPSO algorithm gives better results for the functions 2, 3, 11 and 20, both of the algorithms find optimum results for the remaining functions 1, 15 and 17. For the 50 dimension, while SPSO algorithm performs better than the LFPSO algorithm for the functions 1, 2, 15 and 20, both of the algorithms obtain optimum result for the function 17. The remaining 17 benchmark functions, LFPSO algorithm gives better results for the SPSO algorithm. Sphere and Schwefel2.2 are simple unimodal function, SPSO is more successful than LFPSO for these functions. But for the multimodal functions, while SPSO algorithm gets stuck the local minima, LFPSO algorithm escapes the local minima and obtains the better result than SPSO.

When a more general consideration is made for Table 3, LFPSO is observed to give more successful and robust results for most of the benchmark functions. We can conduct robustness comparisons of the algorithms by checking the standard deviations. The fact inconsistency between its results is high shows that it is not robust. With respect to standard deviations in Table 3, LFPSO is a robust algorithm compared to SPSO.

When the dimension increases, the success of the LFPSO algorithm has been continuing except a few function. While LFPSO algorithm obtains worse result for function 3 and 11 with 30 dimension, it is more successful than SPSO for functions 3 and 11 with 50 dimension. Also whereas the proposed method acquires good result for the functions 1 and 15 with 50 dimension, it does not show same success with 30 dimension.

Convergence graphs of the methods

In order to compare convergence rates of the methods, convergence graphs for SPSO and LFPSO methods are given for 30 and 50 dimensions are given in Figs. 3 and 4.

When examined Fig. 3, while SPSO algorithm convergences rapidly for unimodal Schwefel2.22 and SumSquare functions, LFPSO algorithm does not obtain successful results. In Table 3 shows that high values of standard deviation for the Rosenbrock function with 50 dimension, so SPSO algorithm does not perform robust results owing to this situation. At the convergence graph for Rosenbrock function, SPSO has early convergence, but both of algorithms reach close results. When SPSO algorithm does not improve solution, LFPSO acquires better results for unimodal Noise, multimodal Rastrigin, Griewank, Penalized2, Schaffer and rotated Rastrigin functions. Fig. 3 shows while SPSO algorithm gets stuck the local minima owing to early convergence, LFPSO algorithm escapes local minima and nears the optimum solution for multimodal functions.

When examined Fig. 4, SPSO has a problem of being trapped local minima, the proposed method continues to improve solutions for multimodal Ackley, Penalized1, Levy, non-continuous Rastrigin and Rotated Ackley functions. Due to the fact that both of the algorithms have high standard deviation values, results are not robust for Rotated Schwefel function. The result of algorithms for this function does not pass statistical test. Therefore, interpret the convergence graph would not be correct for this function. Although SPSO convergences faster, LFPSO obtains more successful result for unimodal Schwefel2.26 and Step functions.

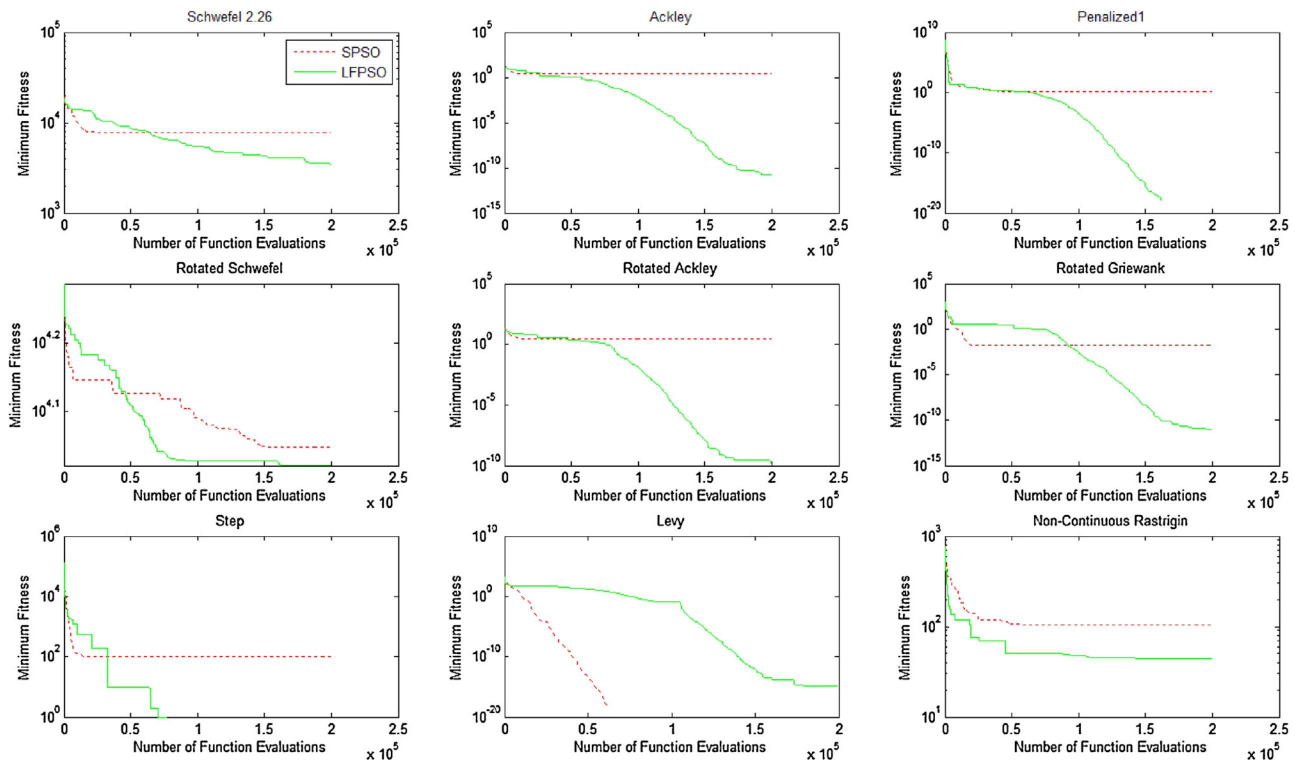

Fig. 3. Convergence graphs of functions for 50 dimensions.

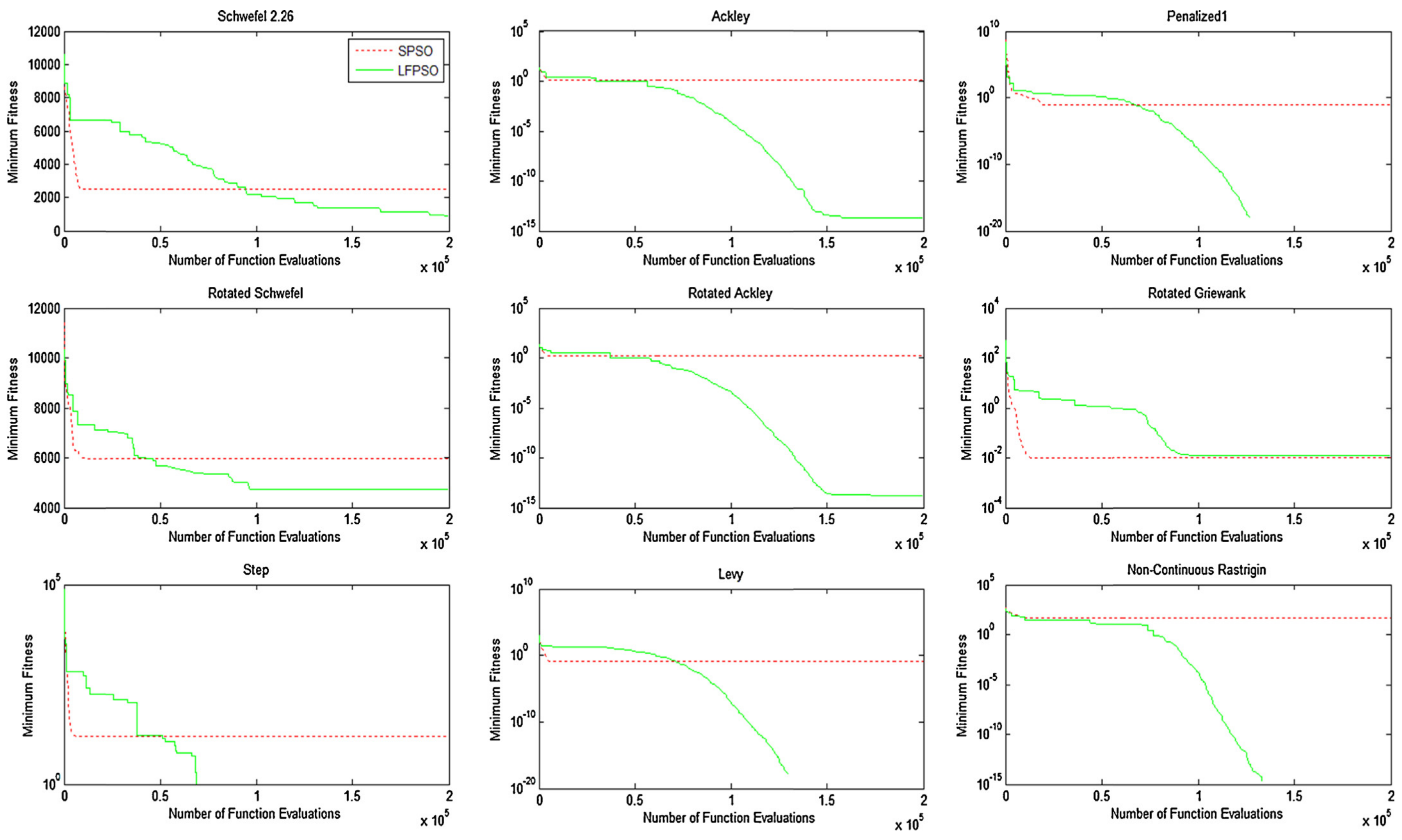

Fig. 4. Convergence graphs of functions for 30 dimensions.

Table 3
Experimental results of SPSO and LFPSO (Opt.: optimum, Std. Dev: standard deviation, Sign: statistical test sign).

F	*D*	Opt.	SPSO (PSO2007)		LFPSO		
			Mean	Std. Dev.	Mean	Std. Dev.	Sign
F1	30	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	
	50	0.00E+00	**0.00E+00**	**0.00E+00**	1.45E−14	4.88E−14	+
F2	30	0.00E+00	**0.00E+00**	**0.00E+00**	3.33E−01	1.83E+00	+
	50	0.00E+00	**0.00E+00**	**0.00E+00**	3.33E−01	1.83E+00	+
F3	30	0.00E+00	**8.91E+00**	**1.29E+01**	2.39E+01	3.26E−01	+
	50	0.00E+00	5.89E+01	3.21E+01	**4.40E+01**	**2.20E−01**	+
F4	30	0.00E+00	3.38E−03	1.55E−03	**2.42E−03**	**1.22E−03**	+
	50	0.00E+00	1.57E−02	7.76E−03	**5.75E−03**	**3.04E−03**	+
F5	30	0.00E+00	3.84E+03	6.75E+02	**1.74E+03**	**1.02E+03**	+
	50	0.00E+00	7.20E+03	1.06E+03	**4.49E+03**	**1.37E+03**	+
F6	30	0.00E+00	4.59E+01	1.40E+01	**3.47E+00**	**8.78E+00**	+
	50	0.00E+00	9.67E+01	2.50E+01	**2.17E+01**	**2.51E+01**	+
F7	30	0.00E+00	1.21E+00	9.03E−01	**1.58E−14**	**4.03E−15**	+
	50	0.00E+00	2.27E+00	5.35E−01	**2.35E−10**	**6.25E−10**	+
F8	30	0.00E+00	1.64E−02	2.54E−02	**0.00E+00**	**0.00E+00**	+
	50	0.00E+00	3.10E−02	4.87E−02	**1.26E−02**	**2.36E−02**	+
F9	30	0.00E+00	2.01E−01	4.38E−01	**0.00E+00**	**0.00E+00**	+
	50	0.00E+00	3.82E−01	4.85E−01	**1.80E−17**	**8.32E−17**	+
F10	30	0.00E+00	5.80E−02	2.93E−01	**0.00E+00**	**0.00E+00**	+
	50	0.00E+00	2.38E−01	7.32E−01	**8.78E−11**	**4.71E−10**	+
F11	30	0.00E+00	**5.29E+03**	**9.02E+02**	5.83E+03	5.17E+02	+
	50	0.00E+00	1.00E+04	9.91E+02	**9.86E+03**	**8.14E+02**	–
F12	30	0.00E+00	4.85E+01	1.36E+01	**4.38E+00**	**1.77E+01**	+
	50	0.00E+00	1.04E+02	3.91E+01	**8.05E−15**	**7.72E−15**	+
F13	30	0.00E+00	1.48E+00	8.58E−01	**1.63E−14**	**4.99E−15**	+
	50	0.00E+00	2.26E+00	5.87E−01	**3.53E−10**	**8.88E−10**	+
F14	30	0.00E+00	2.09E−02	2.39E−02	**3.02E−03**	**7.00E−03**	+
	50	0.00E+00	2.42E−02	4.56E−02	**3.69E−03**	**6.89E−03**	–
F15	30	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	
	50	0.00E+00	**0.00E+00**	**0.00E+00**	1.67E+01	9.13E+01	+
F16	30	0.00E+00	3.83E+00	7.24E+00	**0.00E+00**	**0.00E+00**	+
	50	0.00E+00	1.76E+01	3.04E+01	**0.00E+00**	**0.00E+00**	+
F17	30	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	
	50	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	
F18	30	0.00E+00	6.58E−02	1.84E−01	**0.00E+00**	**0.00E+00**	+
	50	0.00E+00	4.54E−01	1.12E+00	**4.31E−15**	**1.96E−14**	+
F19	30	0.00E+00	4.54E−02	1.67E−02	**9.07E−03**	**2.45E−03**	+
	50	0.00E+00	1.03E−01	4.28E−02	**1.82E−02**	**1.27E−02**	+
F20	30	0.00E+00	**2.69E−15**	**1.99E−15**	7.97E−08	4.36E−07	+
	50	0.00E+00	**7.59E−15**	**3.26E−15**	1.74E−05	9.05E−05	+
F21	30	0.00E+00	4.13E+01	1.18E+01	**2.07E+00**	**5.64E+00**	+
	50	0.00E+00	7.39E+01	2.28E+01	**1.41E+01**	**1.82E+01**	+

Even SPSO convergence faster algorithm, it gets stuck the local minima and performs bad result especially multimodal and rotated functions. LFPSO algorithm escapes the local minima, nears the optimum results and obtains more effective and successful results than the SPSO algorithm.

Non-parametric test results

Non-parametric statistical test has been implemented for statistically significant of between LFPSO and the SPSO. We used Wilcoxon's test with significance level 0.05, and results are shown in Table 3. In the charts, in case Wilcoxon test was at 0.05 significance level, + mark was used, otherwise, – mark was used; no mark was used in case all results showed the sale value or optimum value.

From the results in Table 3, it can be observed that the LFPSO algorithm's performance is significantly better compared to SPSO algorithm for 30 and 50 (except function 11 and 14) dimension functions. This shows that the difference between the two methods is statistically significant.

Results and comparison of LFPSO and PSO variants

To evaluate achievement of the proposed method, LFPSO algorithm is compared the other PSO variants with results in Table 4 are given. Results of the algorithms in Table 4 are received from [44]. The used functions for comparison of algorithms are first 14 benchmark functions in Table 2. The dimension of the all benchmark functions is determined as 30. Mentioned in the summary of the literature of PSO algorithm CLPSO [19], also HPSO-TVAC [45], FIPSO [46] and as well as SPSO, LPSO [47] and DMS-PSO [48] algorithms are compared the proposed method.

In the CLPSO, Liang et al. diversified the swarm and targeted to prevent early convergence by making velocity update using *gbest*, or particle's best or *pbests* of different particles and selecting one of them randomly instead of learning from *pbest* and *gbest* of the particles in the original PSO. So this strategy ensures that the diversity of the swarm and prevents to get stuck the local minima [19]. The HPSO-TVAC was proposed to provide the required momentum for particles to find the global optimum solution. In this algorithm, only the acceleration coefficients of the particle swarm strategy are considered to estimate the new velocity of each particle and particles are reinitialized whenever they are stagnated in the search space [45]. Full information of the entire neighborhood is used to guide the particle and influence the flying velocity in a fully informed particle swarm (FIPSO) [46]. So this increases the diversity of the population. In LPSO which is the local version of PSO, a particle uses the best historical experience of the particle in its neighborhood which is determined by some topological structure [44,47]. In DMS-PSO, dynamic multi-swarm was used and all population was divided in many small swarms. In order to decelerate the population's convergence velocity and increase diversity, DMS-PSO used

Table 4
Results and comparison of PSO variants on 14 benchmark functions.

Functions		CLPSO	HPSO-TVAC	FIPSO	SPSO-40	LPSO	DMS-PSO	LFPSO
1	Mean	1.58E−12	2.83E−33	2.42E−13	**2.29E−96**	3.34E−14	2.65E−31	4.69E−31
	Std. Dev.	7.70E−13	3.19E−33	1.73E−13	**9.48E−96**	5.39E−14	6.25E−31	2.50E−30
	Rank	7	2	6	1	5	3	4
2	Mean	2.51E−08	9.03E−20	2.76E−08	**1.74E−53**	1.70E−10	1.57E−18	2.64E−17
	Std. Dev.	5.84E−09	9.58E−20	9.04E−09	**1.58E−53**	1.39E−10	3.79E−18	6.92E−17
	Rank	6	2	7	1	5	3	4
3	Mean	**1.14E+01**	2.39E+01	2.51E+01	1.35E+01	2.81E+01	4.16E+01	2.38E+01
	Std. Dev.	**9.85E+00**	2.65E+01	5.10E−01	1.46E+01	2.18E+01	3.03E+01	3.17E−01
	Rank	1	4	5	2	6	7	3
4	Mean	5.85E−03	9.82E−02	4.24E−03	4.02E−03	2.28E−02	1.45E−02	**2.41E−03**
	Std. Dev.	1.11E−03	3.26E−02	1.28E−03	1.66E−03	5.60E−03	5.05E−03	**8.07E−04**
	Rank	4	7	3	2	6	5	1
5	Mean	**3.82E−04**	1.59E+03	9.93E+02	3.14E+03	3.16E+03	3.21E+03	1.37E+03
	Std. Dev.	**1.28E−07**	3.26E+02	5.09E+02	7.81E+02	4.06E+02	6.51E+02	6.36E+02
	Rank	1	4	2	5	6	7	3
6	Mean	**9.09E−05**	9.43E+00	6.51E+01	4.10E+01	3.51E+01	2.72E+01	4.54E+00
	Std. Dev.	**1.25E−04**	3.48E+00	1.34E+01	1.11E+01	6.89E+00	6.02E+00	1.03E+01
	Rank	1	3	7	6	5	4	2
7	Mean	3.66E−07	7.29E−14	2.33E−07	3.73E−02	8.20E−08	1.84E−14	**1.68E−14**
	Std. Dev.	7.57E−08	3.00E−14	7.19E−08	1.90E−01	6.73E−08	4.35E−15	**4.84E−15**
	Rank	6	3	5	7	4	2	1
8	Mean	9.02E−09	9.75E−03	9.01E−12	7.48E−03	1.53E−03	6.21E−03	**8.14E−17**
	Std. Dev.	8.57E−09	8.33E−03	1.84E−11	1.25E−02	4.32E−03	8.14E−03	**4.46E−16**
	Rank	3	7	2	6	4	5	1
9	Mean	6.45E−14	2.71E−29	1.96E−15	7.47E−02	8.10E−16	2.51E−30	**4.67E−31**
	Std. Dev.	3.70E−14	1.88E−29	1.11E−15	3.11E+00	1.07E−15	1.02E−29	**9.01E−31**
	Rank	6	3	5	7	4	2	1
10	Mean	1.25E−12	2.79E−28	2.70E−14	1.76E−03	3.26E−13	2.64E−03	**1.51E−28**
	Std. Dev.	9.45E−13	2.18E−28	1.57E−14	4.11E−03	3.70E−13	4.79E−03	**8.00E−28**
	Rank	6	2	4	7	3	5	1
11	Mean	4.39E+03	5.32E+03	4.41E+03	4.57E+03	4.50E+03	**4.04E+03**	5.51E+03
	Std. Dev.	3.51E+02	7.00E+02	9.94E+02	6.28E+02	3.97E+02	**5.68E+02**	5.64E+02
	Rank	2	6	3	5	4	1	7
12	Mean	8.71E+01	5.29E+01	1.50E+02	4.34E+01	5.34E+01	4.20E+01	**1.79E+00**
	Std. Dev.	1.08E+01	1.25E+01	1.45E+01	1.74E+01	1.40E+01	9.74E+00	**9.81E+00**
	Rank	6	5	7	3	4	2	1
13	Mean	5.91E−05	9.29E+00	3.16E−07	9.24E−02	1.55E+00	2.42E−14	**1.65E−14**
	Std. Dev.	6.46E−05	2.07E+00	1.00E−07	3.20E−01	4.50E−01	1.52E−14	**5.40E−15**
	Rank	4	7	3	5	6	2	1
14	Mean	7.96E−05	9.26E−03	**1.28E−08**	3.05E−03	1.68E−03	1.02E−02	1.48E−03
	Std. Dev.	7.66E−05	8.80E−03	**4.29E−08**	5.70E−03	3.47E−03	1.24E−02	6.17E−03
	Rank	2	6	1	5	4	7	3
Mean rank		3.93	4.36	4.29	4.43	4.71	3.93	2.36
Final rank		2	4	3	5	6	2	1
Algorithms		CLPSO	HPSO-TVAC	FIPSO	SPSO-40	LPSO	DMS-PSO	LFPSO

small neighborhoods [48]. All of the mentioned PSO variants aim to improve solution search equation in order to diversify the particle population and prevent get stuck local minima. However, the proposed method aims not only improving solution search equation using efficiency of Levy flight approach but also preventing search within inefficient spaces using limit parameter. Consequently, the LFPSO searches the solution space efficiently and effectively via limit parameter and Levy flight.

The number of particles of all algorithms is determined as 40. The number of particle of SPSO is not evaluated by Eq. (14), SPSO-40 is indicated to number of particles is 40 for the algorithm. The dimension of all benchmark functions (first fourteen functions in Table 2) is received as 30. Furthermore, the FEs is set as 200,000 for each run and all algorithms. For the purpose of reducing statistical errors, each algorithm is tested 25 times independently for every function and the mean results are used in the comparison. For results given in Table 4 is not used error rate. Algorithms run until it reaches the FEs.

Table 4 shows the mean result of 25 runs with 200,000 FEs of all benchmark functions with dimension 30. The information about the mean optimum solution, standard deviation and rank of the mean solution by LFPSO and other PSO variants in 25 runs over first 14 benchmark functions are given in Table 4. The best mean result and the best standard deviations of the benchmark functions obtained by the algorithms are shown in bold. Being compared to the mean values of seven algorithms from best to worst they are listed correctly. Algorithm finds the best result for the function ranks 1, worst algorithm ranks 7 and the other algorithms rank between 1 and 7 in terms of the results.

While SPSO-40 algorithm finds the best solution for Sphere and Schwefel2.22 unimodal functions, DMS-PSO is first rank for the Rotated Schwefel. CLPSO seems good at Rosenbrock,

Table 5
Results and comparison of other algorithms on 10 benchmark functions (Opt.: optimum. Std. Dev: standard deviation).

F	Opt.	GSO		CS		FA		LFPSO	
		Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
F1	0.00E+00	3.37E+01	7.20E+00	8.91E−13	1.84E−13	1.19E−07	3.78E−08	**0.00E+00**	**0.00E+00**
F2	0.00E+00	2.35E+01	2.85E+00	3.31E−03	1.47E−03	4.57E−07	1.09E−07	**0.00E+00**	**0.00E+00**
F3	0.00E+00	2.55E+07	1.14E+07	1.97E+01	2.37E+00	**4.61E−06**	**1.48E−06**	3.15E+00	3.30E−01
F4	0.00E+00	4.23E+00	1.26E+00	1.45E−02	3.86E−03	**7.64E−09**	**2.01E−09**	1.57E−04	7.66E−05
F5	0.00E+00	2.66E+03	2.75E+02	2.99E+03	1.97E+02	1.10E−03	3.43E−04	**1.21E−13**	**3.14E−13**
F6	0.00E+00	1.05E+02	1.14E+01	6.39E+01	6.56E+00	1.34E−07	3.57E−08	**0.00E+00**	**0.00E+00**
F7	0.00E+00	1.94E+01	5.07E−01	2.19E+00	9.86E−01	4.81E−06	1.32E−06	**3.55E−15**	**0.00E+00**
F8	0.00E+00	1.14E+02	2.80E+01	1.34E−05	2.12E−05	2.43E−02	3.45E−03	**0.00E+00**	**0.00E+00**
F10	0.00E+00	1.17E+08	6.17E+07	7.77E−07	4.08E−07	1.20E−05	3.24E−06	**1.35E−32**	**5.57E−48**
F16	0.00E+00	1.22E+04	2.99E+03	**0.00E+00**	**0.00E+00**	4.51E−05	1.18E−05	**0.00E+00**	**0.00E+00**

Schwefel2.26 ve Rastrigin function, it does not show same success for Ackley, Penalized1 and Penalized2 function. FIPSO performs best for Rotated Griewank, HPSO-TVAC algorithm is failed to achieve best solution for any function.

The propose method is generally successful for multimodal functions, it performs best Noise, Ackley, Griewank, Penalized1, Penalized2, Rotated Rastrigin and Rotated Ackley function. Although LFPSO has worst solution on Rotated Schwefel function.

When examined results given in Table 4, the LFSPO method finds best mean solution seven functions and also the proposed method is located first three rank except 1, 2 and 11 functions. LFPSO algorithm is more effective on the multimodal benchmark functions. As mentioned before, this is provided improving ability of PSO's global search through Levy flight method and it escapes the local minima.

The main difference between the proposed method and the PSO variants is the proposed method controls whether the particle's position is efficient. LFPSO algorithm checks the limit parameter for each particle. If the particle position is not efficient, in other words, the particle has exceeded the limit value, LFPSO algorithm leaves this position and the particle is generated in the search space by Levy flight method. Thus the proposed method uses the search space more effectively and efficiently.

Table 4 also ranks the algorithms on performance in terms of the mean solution. It can be observed from the final rank that LFPSO offers the best overall performance, while CLPSO and DMS-PSO are the second best, followed by FIPSO, HPSO-TVAC, SPSO-40, finally LPSO. So, LFPSO algorithm is more successful than the determined PSO variants for the 14 benchmark functions.

Results and comparison of LFPSO and other algorithms

The proposed method is compared the other optimization techniques which are proposed in recent years with results in Table 5 are given. Glowworm swarm optimization (GSO) [49], cuckoo search (CS) [34] and firefly algorithm (FA) [50] algorithms are compared the proposed method. Results of the GSO algorithm in Table 5 are directly received from [51]. The program codes of CS and FA algorithms can be found at [52,53]. The used functions for comparison of algorithms are 10 benchmark functions in Table 2. The dimension of the all benchmark functions is determined as 10. Furthermore, the FEs is set as 500,000 for each run and all algorithms. For the purpose of reducing statistical errors, each algorithm is tested 30 times independently for every function and the mean results are used in the comparison. The best mean result and the best standard deviations of the benchmark functions obtained by the algorithms are shown in bold.

According to Table 5, the proposed method finds the best solution for all functions except Rosenbrock and Noise function. Furthermore, the LFPSO algorithm ensures optimum value for the five function. GSO algorithm are failed to acquire best solution for any function. CS yields optimum value only for Step function. FA algorithm obtains best result for Rosenbrock and Noise. Results in given Table 5 show that the proposed method outperforms the other methods.

Also, the LFPSO algorithm is compared the differential evolution (DE), genetic algorithm (GA) and artificial bee colony algorithm (ABC) which are mostly known optimization algorithms. Results of the GA, DE and ABC algorithms in Table 6 are received from [54]. Table 6 shows the mean result of 30 runs with 500,000 FEs of all benchmark functions with dimension 30. For results given in Table 6, the values below 10^{-12} (error rate) are assumed to be 0. The best mean result and the best standard deviations of the benchmark functions obtained by the algorithms are shown in bold.

The results show that all the algorithms yield optimum value for Sphere, Schwefel2.22, Ackley and Step function except GA. DE for Rosenbrock, ABC for Schwefel2.26 and the proposed method for Quartic find the best solutions. ABC and the proposed method ensure the optimum value for seven functions. According to Table 6, the results obtained by LFPSO are comparable with ABC algorithm and better than GA and DE algorithms.

To sum up, with regard to Tables 5 and 6, the proposed method is very efficient and effective algorithm especially multimodal and

Table 6
Results and comparison of other most-known algorithms on 10 benchmark functions (Opt.: optimum).

F	Opt.	GA	DE	ABC	LFPSO
F1	0.00E+00	1.11E+03	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	0.00E+00	1.10E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	0.00E+00	1.96E+05	1.82E+01	**8.88E−02**	2.31E+01
F4	0.00E+00	1.81E−01	1.36E−03	3.00E−02	**1.27E−03**
F5	0.00E+00	9.76E+02	2.30E+03	**1.30E−02**	1.18E+03
F6	0.00E+00	5.29E+01	1.17E+01	**0.00E+00**	**0.00E+00**
F7	0.00E+00	1.47E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**
F8	0.00E+00	1.06E+01	1.48E−03	**0.00E+00**	**0.00E+00**
F10	0.00E+00	1.25E+02	2.20E−03	**0.00E+00**	**0.00E+00**
F16	0.00E+00	1.17E+03	**0.00E+00**	**0.00E+00**	**0.00E+00**

some of unimodal functions. Also, the proposed method outperforms other algorithms and it is closely successful with the ABC algorithm.

Results and discussion

When considered the results Tables 3–6 the LFPSO method is clearly seen to be more successful than the PSO algorithm, the other PSO variants and the other algorithms. PSO has many advantages, such as simplicity, easy to implement, successful local search. PSO performs velocity change via being affected by both local and global conditions. Despite these advantages, since particles show stagnation behaviors and resemble each other during iterations (loss of diversity), velocity changes drop to very little values and lead to loss of global search ability. Thus, the original PSO loses influence and productivity. In this point, the proposed approach improves the global search capability of the algorithm by increasing diversification in the population. The LFPSO controls whether the particles improve self-solutions through limit value. If particles could not improve self-solutions, the LFPSO redistributes the particles to search space by using Levy flight. By Levy flight, it was ensured that the distributions are performed as long jumps, and that search space is used effectively. Thanks to these jumps, being trapped in local minimums was prevented particularly for multimodal functions, and the global search ability of the original PSO was strengthened. In order to conceive better the success of the method, the benchmark functions are mainly selected as multimodal functions. Based on the experimental results for multimodal functions, the LFPSO prevents being trapped in local minima through the Levy distribution. Owing to weakness and the problem of the premature convergence in the basic PSO, one of state-of-the-art PSO [43] is used for the experimental results and comparison. It is observed LFPSO algorithm performs better result than the SPSO algorithm for the most of the benchmark functions. When the standard deviations are considered, the proposed method is more robust than the state-of-the-art PSO algorithm. The non-parametric Wilcoxon test shows that performance of LFPSO is statistically significant in comparison to SPSO for majority of test functions.

In addition, to evaluate achievement of the proposed method, LFPSO algorithm is compared the other PSO variant. Also the algorithms rank on performance in terms of the mean solution. LFPSO algorithm is first with respect to final rank between PSO variants. Furthermore, the LFPSO surpasses other algorithms and it is closely successful with the ABC algorithm. The proposed method is very efficient and effective algorithm especially multimodal and some of unimodal functions.

The LFPSO method will be a good choice particularly for multimodal, rotated and some of unimodal functions.

Conclusion and future work

In order to overcome the original PSO algorithm's problem of being trapped in local minima and being unable to perform well global search due to early convergence, PSO is combined with Levy flight in this study. The comparison between the SPSO and LFPSO was performed on 21 benchmark functions with 30 and 50 dimensions. The proposed method was observed to give better average results in almost all benchmark functions tested, and to be more robust in most of them. Furthermore, in order to consider the performance of the LFPSO algorithm, it is compared the other PSO variants. When evaluated experimental results of the PSO variants, LFPSO is more successful than the other PSO variants. Moreover, the results of proposed method are also compared with the results of well-known and recent population-based optimization methods. The LFPSO outperforms other methods and it is closely successful with the ABC algorithm.

As future work, since success on the benchmark function of the proposed methods, the proposed method will be used to different optimization problems such as training neural networks, scheduling problems, image segmentation, etc. Also, because Levy flight method is a good tool for providing diversification in the population, it will be used for the other nature-inspired algorithms.

Acknowledgements

The authors thank Ömer Kaan Baykan (Ph.D.) and Mustafa Servet Kıran (Ph.D.q) for their suggestions and comments. This study has been supported by Scientific Research Project of Selcuk University.

References

[1] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Comput. Eng. Dep. (2005).
[2] M. Dorigo, G.D. Caro, Ant colony optimization: a new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, 1999, pp. 1470–1477.
[3] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39–43.
[4] M. Li, D. Lin, J. Kou, A hybrid niching PSO enhanced with recombination-replacement crowding strategy for multimodal function optimization, Appl. Soft Comput. 12 (2012) 975–987.
[5] A. Sarangi, R.K. Mahapatra, S.P. Panigrahi, DEPSO and PSO-QI in digital filter design, Expert Syst. Appl. 38 (2011) 10966–10973.
[6] Y.-P. Chang, C.-N. Ko, A PSO method with nonlinear time-varying evolution based on neural network for design of optimal harmonic filters, Expert Syst. Appl. 36 (2009) 6809–6816.
[7] J.-S. Chiou, S.-H. Tsai, M.-T. Liu, A. PSO-based adaptive fuzzy PID-controllers, Simul. Model. Pract. Theory 26 (2012) 49–59.
[8] X. Yang, J. Yuan, J. Yuan, H. Mao, An improved WM method based on PSO for electric load forecasting, Expert Syst. Appl. 37 (2010) 8036–8041.
[9] H.-C. Yang, S.-B. Zhang, K.-Z. Deng, P.-J. Du, Research into a feature selection method for hyperspectral imagery using PSO and SVM, J. China Univ. Mining Technol. 17 (2007) 473–478.
[10] L.-Y. Chuang, H.-W. Chang, C.-J. Tu, C.-H. Yang, Improved binary PSO for feature selection using gene expression data, Comput. Biol. Chem. 32 (2008) 29–38.
[11] M.A. Cavuslu, C. Karakuzu, F. Karakaya, Neural identification of dynamic systems on FPGA with improved PSO learning, Appl. Soft Comput. 12 (2012) 2707–2718.
[12] Y. Zhang, D. Huang, M. Ji, F. Xie, Image segmentation using PSO and PCM with Mahalanobis distance, Expert Syst. Appl. 38 (2011) 9036–9040.
[13] Q.-K. Pana, M.F. Tasgetiren, Y.-C. Liang, A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem, Comput. Oper. Res. 35 (2008) 2807–2839.
[14] M.F. Tasgetiren, Y.-C. Liang, M. Sevkli, G. Gencyilmaz, A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem, Eur. J. Oper. Res. 177 (2007) 1930–1947.
[15] J.-P. Yang, C.-K. Kung, F.-T. Liu, Y.-J. Chen, C.-Y. Chang, R.-C. Hwang, Logic circuit design by neural network and PSO algorithm, in: 2010 First International Conference on Pervasive Computing Signal Processing and Applications (PCSPA), Harbin, China, 2010, pp. 456–459.
[16] C.A. Coello Coello, E.H. Luna, A.H. Aguirre, A comparative study of encodings to design combinational logic circuits using particle swarm optimization, in: Proceedings of 2004 NASA/DoD Conference on Evolvable Hardware, Seattle, Washington, 2004, pp. 71–78.
[17] R.C. Eberhart, X. Hu, Human tremor analysis using particle swarm optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, 1999, pp. 1927–1930.
[18] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: The 1998 IEEE International Conference on Evolutionary Computation Proceedings, Anchorage, AK, 1998, pp. 69–73.
[19] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. (2006) 281–295.
[20] Z. Xinchao, A perturbed particle swarm algorithm for numerical optimization, Appl. Soft Comput. 10 (2010) 119–124.
[21] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, Q. Tian, Self-adaptive learning based particle swarm optimization, Inf. Sci. 181 (2011) 4515–4538.
[22] I.G. Tsoulos, A. Stavrakoudis, Enhancing PSO methods for global optimization, Appl. Math. Comput. 216 (2010) 2988–3001.
[23] X.H. Shi, Y.C. Liang, H.P. Lee, C. Lu, L.M. Wang, An improved GA and a novel PSO-GA-based hybrid algorithm, Inf. Process. Lett. 93 (2005) 255–261.

[24] S.C. Chiam, K.C. Tan, A.A. Mamun, A memetic model of evolutionary PSO for computational finance applications, Expert Syst. Appl. 36 (2009) 3695–3711.
[25] M.S. Kıran, M. Gunduz, O.K. Baykan, A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum, Appl. Math. Comput. 219 (2012) 1515–1521.
[26] A.A. Al-Temeemy, J.W. Spencer, J.F. Ralph, Levy flights for improved ladar scanning, in: 2010 IEEE International Conference on Imaging Systems and Techniques (IST), Thessaloniki, Greece, 2010, pp. 225–228.
[27] Y. Chen, Research and simulation on Levy flight model for DTN, in: 2010 3rd International Congress on Image and Signal Processing, Yantai, China, 2010, pp. 4421–4423.
[28] M.A. Pereyra, H. Batatia, A Levy flight model for ultrasound in skin tissues, in: 2010 IEEE on Ultrasonics Symposium (IUS), San Diego, CA, 2010, pp. 2327–2331.
[29] G. Terdik, T. Gyires, Lévy flights and fractal modeling of internet traffic, in: IEEE/ACM Transactions on Networking, 2009, pp. 120–129.
[30] D.K. Sutantyo, S. Kernbach, P. Levi, V.A. Nepomnyashchikh, Multi-robot searching algorithm using Levy flight and artificial potential field, in: 2010 IEEE International Workshop on Safety Security and Rescue Robotics (SSRR), Bremen, Germany, 2010, pp. 1–6.
[31] I. Rhee, M. Shin, S. Hong, K. Lee, S.J. Kim, S. Chong, On the Levy-walk nature of human mobility, in: IEEE/ACM Transactions on Networking, 2011, pp. 630–643.
[32] A.M. Edwards, R.A. Phillips, N.W. Watkins, M.P. Freeman, E.J. Murphy, V. Afanasyev, S.V. Buldyrev, M.G.E. da Luz, E.P. Raposo, H.E. Stanley, G.M. Viswanathan, Revisiting Lévy flight search patterns of wandering albatrosses, bumblebees and deer, Nature 449 (2007) 1044–1048.
[33] G.M. Viswanathan, V. Afanasyev, S.V. Buldyrev, E.J. Murphy, P.A. Prince, H.E. Stanley, Lévy flight search patterns of wandering albatrosses, Nature 381 (1996) 413–415.
[34] X.-S. Yang, S. Deb, Multiobjective cuckoo search for design optimization, Comput. Oper. Res. 40 (2013) 1616–1624.
[35] X.-S. Yang, Firefly algorithm, Levy flights and global optimization, in: M. Bramer, R. Ellis, M. Petridis (Eds.), Research and Development in Intelligent Systems XXVI, Springer, London, 2010, pp. 209–218.
[36] C.-Y. Lee, X. Yao, Evolutionary algorithms with adaptive Levy mutations, in: Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, South Korea, 2001, pp. 568–575.
[37] R. Candela, G. Cottone, G.F. Scimemi, E.R. Sanseverino, Lévy flights for ant colony optimization in continuous domains, in: Mathematical Theory and Computational Practice Fifth Conference on Computability in Europe, Heidelberg, Germany, 2009, pp. 79–88.
[38] R. Candela, G. Cottone, G.F. Scimemi, E.R. Sanseverino, Composite laminates buckling optimization through Lévy based ant colony optimization, in: 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, Cordoba, Spain, 2010, pp. 288–297.
[39] G. Cottone, A. Pirrotta, G.F. Scimemi, E.R. Sanseverino, Damage identification by Lévy ant colony optimization, Reliab. Optim. Struct. Syst. (2010) 37–44.
[40] Y. Ortakçı, Comparison of Particle Swarm Optimization Methods in Applications, Karabuk University, Graduate School of Natural and Applied Sciences, 2011.
[41] A.V. Chechkin, R. Metzler, J. Klafter, V.Y. Gonchar, Introduction to the theory of Lévy flights, in: R. Klages, G. Radons, I.M. Sokolov (Eds.), Anomalous Transport: Foundations and Applications, John Wiley & Sons, 2008, pp. 129–162.
[42] X.-S. Yang, Engineering Optimization an Introduction with Metaheuristic Applications, first ed., John Wiley & Sons, New Jersey, 2010.
[43] M. Omran, SPSO 2007 Matlab, 2007 http://www.particleswarm.info/Programs.html (Last Access: 12.03.14 14:50).
[44] Z.-H. Zhan, J. Zhang, Y. Li, Y.-H. Shi, Orthogonal learning particle swarm optimization, IEEE Trans. Evol. Comput. 15 (2011) 832–846.
[45] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evol. Comput. 8 (3) (2004) 240–255.
[46] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evol. Comput. 8 (3) (2004) 204–210.
[47] J. Kennedy, R. Mendes, Population structure and particle swarm performance, IEEE Congr. Evol. Comput., Honolulu (2002) 1671–1676.
[48] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, Swarm Intelligence Symposium, California, 2005, pp. 124–129.
[49] K.N. Krishnanand, D. Ghose, Detection of multiple source locations using a glowworm metaphor with applications to collective robotics, in: IEEE Swarm Intelligence Symposium, Pasadena, CA, USA, 2005, pp. 84–91.
[50] X.-S. Yang, Firefly algorithms for multimodal optimization, in: 5th International Symposium SAGA, Sapporo, Japan, 2009, pp. 169–178.
[51] B. Wu, C. Qian, W. Ni, S. Fan, The improvement of glowworm swarm optimization for continuous optimization problems, Expert Syst. Appl. 39 (2012) 6335–6342.
[52] http://www.mathworks.co.uk/matlabcentral/fileexchange/29809-cuckoo-search-cs-algorithm (last accessed 12.03.14 14:50).
[53] http://www.mathworks.com/matlabcentral/fileexchange/29693-firefly-algorithm (last accessed 12.03.14 14:50).
[54] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 (2009) 108–132.