Appl Intell (2012) 37:290–304
DOI 10.1007/s10489-011-0328-6

LADPSO: using fuzzy logic to conduct PSO algorithm

Mohammad Sadegh Norouzzadeh · Mohammad Reza Ahmadzadeh · Maziar Palhang

Published online: 10 December 2011

Abstract Optimization plays a critical role in human modern life. Nowadays, optimization is used in many aspects of human modern life including engineering, medicine, agriculture and economy. Due to the growing number of optimization problems and their growing complexity, we need to improve and develop theoretical and practical optimization methods. Stochastic population based optimization algorithms like genetic algorithms and particle swarm optimization are good candidates for solving complex problems efficiently. Particle swarm optimization (PSO) is an optimization algorithm that has received much attention in recent years. PSO is a simple and computationally inexpensive algorithm inspired by the social behavior of bird flocks and fish schools. However, PSO suffers from premature convergence, especially in high dimensional multi-modal functions. In this paper, a new method for improving PSO has been introduced. The Proposed method which has been named Light Adaptive Particle Swarm Optimization is a novel method that uses a fuzzy control system to conduct the standard algorithm. The suggested method uses two adjunct operators along with the fuzzy system in order to improve the base algorithm on global optimization problems. Our approach is validated using a number of common complex uni-modal/multi-modal benchmark functions and results have been compared with the results of Standard PSO (SPSO2011) and some other methods. The simulation results demonstrate that results of the proposed approach is promising for improving the standard PSO algorithm on global optimization problems and also improving performance of the algorithm.

Keywords Particle swarm optimization · Fuzzy control · Random search · Numerical function optimization · Premature convergence

1 Introduction

There exists systems in nature which are composed of simple agents. Each agent in these systems looks to be simple and unintelligent but, the behavior of the entire system surprisingly seems intelligent. In such systems, no central control or other types of coordinators exist but the collective behavior of the system is purposeful and smart. Scientists named this behavior as swarm intelligence. In other words, in a swarm intelligent system each agent does a simple task and interacts locally with the environment and other agents, but the collective behavior of the entire system that results from these simple tasks is intelligent. Many swarm intelligent systems exist in nature, for example ants that allocate their tasks dynamically without any coordinator. As another example birds in a flock and fishes in a school organize themselves in optimal geometrical patterns.

Computer scientists have been inspired by swarm intelligent systems in nature and have tried to imitate the behavior of such systems by inventing computational swarm intelligent models. The aim of such computational models is making powerful methods for solving problems which composed of simple and computationally inexpensive agents instead of devising complex concentrated systems. Working with many simple and understandable agents is easier than working with a very complex system. Particle Swarm Optimization (PSO) algorithm is an example of such compu-

M.S. Norouzzadeh (✉) · M.R. Ahmadzadeh · M. Palhang
Electrical & Computer Engineering Department, Isfahan University of Technology, Isfahan, Iran
e-mail: m.norouzzadeh@ec.iut.ac.ir

M.R. Ahmadzadeh (✉)
e-mail: ahmadzadeh@cc.iut.ac.ir

tational models which tries to simulate behavior of the bird swarms.

PSO is a population based numerical optimization algorithm inspired by the social behavior of bird swarms. Kennedy and Eberhart first introduced PSO algorithm in 1995 [1]. Like evolutionary algorithms, PSO maintains a group of candidate solutions. In PSO each candidate's solutions is called particle and the entire population is called swarm. In order to simulating the behavior of natural bird swarms, each particle within PSO swarm does two simple tasks. First, it constantly desires to come back to its own previous best position and second, it follows its successful neighbors by flying to their successful positions. The overall behavior of the swarm results from these two simple tasks is swarm's rapid focusing on promising areas of search space.

Although PSO is a speedy and robust optimization algorithm, it suffers from premature convergence especially in problems with many variables. Premature convergence is defined as converging of the algorithm to a suboptimal solution rather than locating the global optimum solution. To date, many researchers have examined various approaches to solve this problem. We will discuss this in some important previous works on this paper.

To avoid converging to local optima in PSO, an idea of conducting the base algorithm by a fuzzy control system and also utilizing exploration ability of random search and diversification of a type of mutation operator is addressed in this paper.

This paper is organized as follows. In Sect. 2, we have talked about the basic PSO algorithm. In Sect. 3, we have reviewed previous attempts for improving PSO algorithm on global optimization problems. In Sect. 4, we have introduced our method and have described it in detail. In Sect. 5, we have explained benchmark functions and their settings then we have compared the simulation results with those of standard PSO, and some other methods. Finally, Sect. 6 is the conclusion and suggestion for some future works.

2 Particle swarm optimization

Particle swarm optimization is a stochastic optimization tool that stores a population of possible solutions to solve problems. The current solution of each particle is called position of the particle. The population initialized by randomly generated particles. The historical PSO algorithm uses uniform random numbers for initialization. During each iteration, particles within the swarm update their position in search space based on two types of knowledge. First their own personal experiences and second the experiences of their neighbors. It means that in each iteration of the algorithm, each particle flies to the direction of the best position of itself and the best position of its neighbors. The neighbors of each particle are usually determined before the start of the algorithm, basic PSO algorithm could be classified as local best PSO or global best PSO base on social structure of the swarm. In PSO algorithm, positions of particles are updated by their velocity vectors. It means that the position of each particle in the next iteration is calculated by adding its velocity vector to its current position (1). Consider that we have an optimization problem in a d-dimensional search space. Let $X_i = (x_{i1}, x_{i2}, \dots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, \dots, v_{id})$ are the ith particle's position vector and velocity vector respectively. Also suppose that $P_i = (p_{i1}, p_{i2}, \dots, p_{id})$ represents best previously visited position of the ith particle and $P_{gi} = (p_{g1}, p_{g2}, \dots, p_{gd})$ represents the global best position of the swarm. In PSO algorithm, optimization is done by the velocity equation. The velocity of each particle is computed according to (2).

$$X_i^{t+1} = X_i^t + V_i^t \tag{1}$$

$$V_i^{t+1} = wV_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (P_g^t - X_i^t) \tag{2}$$

where $d \in \{1, 2, \dots, D\}$, $i \in \{1, 2, \dots, N\}$ and N is the swarm size and the superscript t is the iteration number. In (2) w is the inertia weight which prevents particles to suddenly change their directions, r_1 and r_2 are two random vectors with values in the range $[0, 1]$ used to keep some diversity and c_1 and c_2 are the cognitive and social scaling parameters which are positive constants. The proportion of c_1 and c_2 determine how much each particle relies on its experience and how much it relies on others experiences. Usually c_1 and c_2 are equal and set to about 2.

Empirical simulations showed that in basic PSO, if we do not limit the amount of velocity to some predetermined value, then the magnitude of velocity of particles increase rapidly. In such conditions particles locations increase rapidly too and therefore the swarms is unable to do optimization [2]. Constriction coefficient model [3] is another popular model of PSO algorithm which does not need velocity clamping. Moreover if certain conditions are met, the constriction coefficient model could guarantee convergence of the swarm. In constriction coefficient model the velocity equation change to (3). Where χ called the constriction factor.

$$V_i^{t+1} = \chi[V_i^t + \varphi_1 (P_i^t - X_i^t) + \varphi_2 (P_g^t - X_i^t)] \tag{3}$$

where

$$\chi = \frac{2k}{|2 - \varphi - \sqrt{\varphi(\varphi - 4)}|},$$
$$\varphi = \varphi_1 + \varphi_2, \qquad \varphi_1 = c_1 r_1, \qquad \varphi_2 = c_2 r_2 \text{ and } k \in [0, 1]$$

The parameter k, in (3) controls the exploration and exploitation abilities of the swarm. For $k \approx 0$, fast convergence is obtained with local exploitation. The swarm exhibits an almost hill-climbing behavior. On the other hand, $k \approx 1$ results in slow convergence with a high degree of exploration.

Usually, k is set to a constant value. However it can be decreased during the execution of algorithm.

In this paper, we have used the current standard version of the PSO (SPSO-2011 [12]) as the base algorithm for our method. SPSO is a standard version of PSO approved by researchers of the field, as a base method for comparison of variations of the algorithm.

3 Related works

In optimization theory we are looking for an optimization algorithm that can locate the global optimum point on every problem regardless of its difficulty and algorithm's starting point. In 1997 Wolpert and Macready [4] proved that "averaged over all possible problems or cost functions, the performance of all search algorithms is exactly the same". This means that no algorithm is better on average than other algorithms. Although reaching to the most excellent algorithm for all problems is impossible, we are not coping with all possible problems or cost functions, therefore we are still looking for an algorithm with high accuracy on common and useful problems.

Any optimization algorithm must do two tasks, exploration and exploitation. Exploration means that searching regions of search space to find promising regions and exploitation means that concentrating on a promising area to refine the solution. Some problems needs more exploration and less exploitation while other problems needs more exploitation and less exploration. Every good optimization method must be able to balance these two contradictive tasks. Usually algorithms are trying to explore search space in early steps and then they try to exploit hopeful areas.

Up to now many attempts have been made to deal with PSO algorithm's premature convergence. Some of these attempts were successful to improve PSO algorithm on global optimization problems. Previous works have tried different approaches for improving basic PSO algorithm. In this section we introduce studies about important previous attempts and classify them according to the changes made on the basic algorithm. In this section we have examined seven types of approaches for dealing with premature convergence.

3.1 Changing the initialization method of the algorithm

Some methods have tried to improve the initialization method of the PSO in order to have a better exploration in early iterations of the algorithms. Pant et al. [5] have used different probability distributions for initializing the swarm. In another work Pant et al. utilized low discrepancy sequences (discrepancy is deviation of the random sequence from the true uniform distribution, low discrepancy sequences are less random but more uniform than pseudo-random numbers) for better initializing particles [6]. Plowing PSO is another approach for initializing PSO effectively, which uses a kind of random search for improving initialization [7].

3.2 Changing the neighborhood pattern of the swarm

Some other methods have changed the neighborhood structure of the swarm in order to get to a compromise between exploration and exploitation. Whatever neighborhood of particles in swarm is more coupled, the flow of experiences between particles become faster and therefore premature convergence is more probable. Various methods have been suggested for changing the neighborhood structure. For example in fully informed PSO [8] each particle not only impressed by best of its neighbor but also impressed by all of its neighbors. Suganthan have used Euclidian distance for selecting neighbors [9]. Kennedy and Mendes in [10] have examined the impact of various social topology graphs on PSO performance. In basic PSO algorithm each particle converges to median of its own and its neighbor's best position, Kennedy in barebones PSO [11] has used randomly generated positions around the middle of these two positions instead of particle's best position and best swarm position in velocity equation. Standard PSO (SPSO-07, SPSO-2011) [12] is an improved version and current standard of PSO which uses a random topology for choosing the local best particles adaptively.

3.3 Adjustment of basic algorithm parameters

Some researchers have tried to achieve a compromise between exploration and exploitation by adjusting historical PSO algorithm parameters [13–16]. For example whatever the value of inertia weight is greater, the exploration ability of the swarm is more and vice versa. So some methods adjust inertia weight so that exploration in early steps is high and decrease it during running of algorithm. Constriction coefficient model [3] itself is another method that tries to adjust historical PSO parameters for improving it.

3.4 Hybrid methods

There exists various studies that have mixed PSO algorithm by other optimization methods [17–19] or other method's heuristics and concepts. For example Chen et al. have mixed PSO with Extremal optimization [17]. Angeline [20] has hybridized PSO with the selection operator of genetic algorithm and some other methods have combined PSO with crossover and mutation operators of the genetic algorithm [21–25]. The proposed method in this paper could be classified as hybrid method because it uses the mutation concept from genetic algorithms to improve the PSO algorithm.

3.5 Multi-swarm methods

Some techniques have tried the idea of utilizing more than one swarm for doing optimization. These swarms may co-operate or compete for doing their tasks. For example co-operative split PSO [26] has a number of sub-swarms that each one is responsible for optimization of a subset of the solution. In predator-prey PSO [27], Silva has inspired by predator-prey relationship in nature and introduced predator swarm to PSO which is useful for keeping diversity within the swarm. The predator swarm follows global best position of the (prey) swarm so the preys fear and do not get close to global best position very much so the swarm keeps its diversity.

3.6 Multi-start methods

One of the main reasons of premature convergence is the lack of diversity within the swarm. In PSO when a particle reaches to an optima, it rapidly attracts its neighbors and therefore, the swarm concentrates on the founded optimum. Many techniques [28–30] have been suggested for keeping diversity within swarm by injecting diversity to the swarm by reinitializing particles or their velocities. These techniques are differ based on when they inject diversity to the swarm and how they do diversification. For example some methods randomize velocity vectors while the others randomize the position of the particles. We utilized random restart technique in our proposed algorithm for injecting diversity to the swarm.

3.7 Other methods

Several methods exist that cannot simply be classified as above types. For example Blackwell et al. [31] have introduced charged PSO in which repulsion force has been introduced to PSO. In charged PSO each particle has an electrostatic charge that has impact on the behavior of the particle. Particles with the same electrostatic charge repulse each other and so we can ensure of the existence of diversity in the swarm.

4 The proposed method

Our proposed algorithm is based on the Standard PSO algorithm. It means that our suggested algorithm does all of Standard PSO computations in every iterations, moreover we have utilized two operators and a simple fuzzy control system in order to improve Standard PSO algorithm on global optimization problems. The first operator or plow operator is used for efficient initializing of the algorithm. The second operator or mutation operator is used for escaping from local optima and finally the fuzzy control system is used for conducting the search and effectively using the mutation operator, with the aim of facilitating exploration and exploitation. We will explain details of these operators and control system in addition to description of our algorithm which we named Light Adaptive PSO (LADPSO) in the following sub-sections. Note that all pseudo codes are assuming minimization problems.

```
for i=1 to Problem Dimension
    for j=1 to k
        X=particle_position;
        X(i)=uniform Random number between Upper and lower bound;
        if(fitness(X)<fitness(particle_position))then
            particle_position=X
        end
    end
end
```

Fig. 1 Pseudo code of plow operator

4.1 Plow operator

The plow operator has utilized the exploration ability of blind random search for making initialization of the PSO more effective [7]. Plow operator performs on one particle of the swarm. Plow operator tries to reach a better solution by iteratively changing the variable of one dimension in a candidate solution by uniform random numbers while other dimensions remain fixed. If the new generated position is better than the previous one, then it keeps the new position and drops it otherwise. This procedure gets done for each dimension k times. Since this process is like plowing in agriculture and prepares the algorithm to reach to a better solution, this process has been named as plow. The pseudo code of plow operator is represented in Fig. 1.

As we mentioned earlier, plow operator changes the value of each dimension k times. The original paper [7] uses the plow operator only for global best location of the swarm with $k = 50$, but we use the plow operator for all particles once at the beginning of the algorithm with $k = 20$. However plow operator may be used several times during execution of the algorithm for various reasons. For example plow operator may be used to facilitate exploration, helping the algorithm to escape from local optima or even injecting diversity to the swarm. Usefulness of the plow operator has been reported on [7].

4.2 Mutation operator

Up to now mutation has been combined with PSO in many previous works. We have referenced to some of them in pre-

```
for i=1 to problem dimension
    for j=1 to k
        X= global_best_position;
        X(j)= X(j)+randn*sigma;
        if( fitness(X) < fitness(global_best_position)) then
            global_best_position=X
        end
    end
end
```

Fig. 2 Pseudo code of the adaptive mutation operator

vious section. Several methods including uniform mutation, Gaussian mutation, Levy mutation and adaptive mutation which choose the type of mutation based on problems conditions have been suggested for performing mutation in PSO. In our work, we have added a Gaussian mutation to the particles. The amplitude of the Gaussian mutation is controlled by the fuzzy system. The control strategy designed so that the amplitude of the Gaussian noise is high in the early steps to help the algorithm to explore more effectively and it will be decreased to facilitate exploitation in the last phases.

Like the plow operator, mutation operator changes the value of one dimension k times while other variables remain constant. This change is performed by replacing the value of the variable by a random number with normal distribution with average of variable's old value and a determined variance. The value of variance and k is set automatically by the fuzzy control system. Mutation operator acts on global best location of the entire swarm. Like multi restart techniques, in our algorithm after each application of mutation operator, we have reinitialized some of population randomly (except the global best particle) to inject diversity to the swarm; the number of particles to reinitialize is determined by fuzzy control system too.

The pseudo code of the mutation operator has been depicted in Fig. 2 where "*randn*" indicates a normal random number with zero mean and unity standard deviation.

4.3 Proposed algorithm (LADPSO)

We have utilized two above operators in order to improving standard PSO algorithm. As we mentioned earlier the plowing operator has been used for effective initialization and the mutation operator has been used for escaping from local optima. In LADPSO algorithm plowing is used only once at beginning of the algorithm, on the other hand the mutation operator acts when a fuzzy control system triggers it. We have used control system with the aim of helping the algorithm to converge to global optimum and optimal use of computational resources. Usually on similar works adjunct operators have been used regularly for example one time on every few iterations. This approach does not seem efficient because on some conditions the algorithm uses operators unduly or the algorithm could not use them when it needs to use them. In this paper, we have used a computationally inexpensive fuzzy control system for determining the exact timing of operator's usage and configuration of the operator. This control system specifies whether we must use or not use the mutation operator based on swarm's conditions. Also the fuzzy system specifies the type of mutation operator. The details of our control system will be explained in next section.

```
Initialize swarm;
Calculate best location;
Plow;
Do
    Calculate PSO velocities;
    Calculate particles new locations;
    Calculate personal best and global best for all particles;
    Calculate swarm radius;
    If (output of fuzzy system is mutation) then
        Mutate;
        Reset some of Population;
    If( solution is not enhanced) then
                Stagnation Count++;
    End
While (stopping criteria not met)
```

Fig. 3 Pseudo code of the LADPSO algorithm

In LADPSO, in each iteration, if the control system decides that we need to perform the mutation operator, then it will be done. When the swarm starts to converge we use the mutation operator to help the swarm to escape from local optima. The proposed algorithm first tries to reach to a good start point by performing the plow operator on the entire swarm. In consequent iterations, if the swarm starts to converge then the fuzzy control system decides to do mutation, during the mutation the algorithm tries to escape from local optimum and to inject some diversity to the swarm. If after some mutations the swarm was unable to get better result, the amplitude of Gaussian mutation is decreased to help the algorithm to do more exploitation. In fact the aim of the fuzzy system is to help both the exploration and exploitation when the algorithm needs to them.

Pseudo code and the flowchart of the LADPSO are shown in Figs. 3 and 6.

4.4 Fuzzy control system

We have used a fuzzy approach to cope with premature convergence in this paper. Our fuzzy control system tries to detect premature convergence by examining swarm conditions and use necessary reaction (mutation operator timing and settings) to deal with it. In this section we will explain the details of our fuzzy control. Our control system has been designed based on fuzzy approach; we have defined two input variables for our system. The first one is swarm radius;

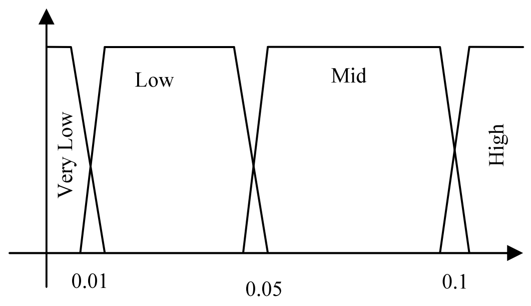

Fig. 4 Membership function of swarm radius

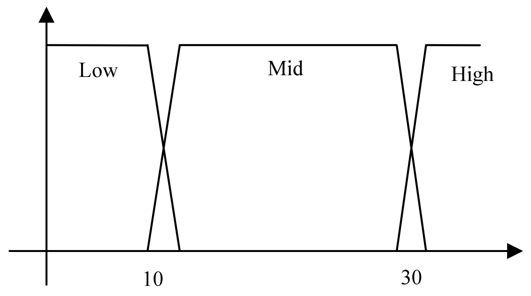

Fig. 5 Membership of stagnation

swarm radius is mean of the standard deviations of the dimensions of particles within swarm. The swarm radius has been computed based on following equations (4) and (5). The second input variable is the number of iterations which the swarm does not success to improve the solution or the number of stagnant iterations.

$$radius(D) = STD(D) \tag{4}$$

$$Swarm\_Radius = \frac{\sum_{D=1}^{\dim} STD(D)}{\dim} \tag{5}$$

To deal with normal variables for all problems we have defined the membership functions of input variables as showed on Figs. 4 and 5.

We have designed a table-based inference engine for our proposed fuzzy system. The inference table which defines our system strategy is depicted on Table 1. We have planned two setting for the mutation operator. These setting are designed so that it could help exploration in early phases and also assist exploitation on the last phases. The details of these settings are shown on Table 2, where k is the k parameter on the mutation operator, sigma is multiplied to search space boundaries (upper limit-lower limit) and produces the variance of the Gaussian mutation. The reinitialize proportion is the proportion of the swarm that must be randomly reinitialized. Our suggested algorithm does not have any extra parameters for setting by user. The values of decision table and membership functions of fuzzy variables are determined based on our experimental simulations. We have tried to tune these parameters accurately, but by using more

Table 1 Decision table of fuzzy controller

Stagnation	Swarm radius			
	Very low	Low	Mid	High
Low	M1	M1	NOP	NOP
Mid	M2	M1	M1	NOP
High	M2	M2	M1	M1

Table 2 Parameters of mutation

	k	Sigma	Reinitialize proportion
M1	10	0.05	0.05
M2	10	0.01	0.00

computational resources it is possible to do a Comprehensive experimental simulation to set these parameters more accurately. Note that in Table 1, NOP means that do not any operation; just leave the standard PSO to go further.

5 Results and discussion

We have organized our simulations in two sections. On the first section we have evaluated the algorithm on first ten functions designed for the special session on real optimization of CEC 2005 [32]. These benchmark functions are largely used and result of many various algorithm are available for them. Also we have compared the results with the results of standard PSO 2011. On the second section we have applied our algorithm on a real life optimization problem and compared it with results of some other methods. Note that all simulations of the proposed algorithm were run on Intel Pentium Dual core with 1.7 GHz CPU and 2 GB of memory.

5.1 CEC 2005 functions

For this section, we have set the number of dimensions to ($D =$)10 and ($D =$)30, the swarm size is set to 40, and maximum function evaluation number is set to $10000 * D$, the other parameters of algorithm is like standard PSO algorithm[12]. We have reported the results of standard PSO algorithm on Tables 4 and 5. Also we have shown the results of our method on Tables 6 and 7. We have reported the result based on settings suggested by [32]. Our results are competitive with some other algorithms which they tested on these benchmark functions for example [33–35] we have quoted results of [33] as an example for comparison on Tables 8 and 9. Also we have reported Algorithm Complexity based on [32] in Table 8, where T_0 is the Computing time for the code on Fig. 7, T_1 is computing time of the function 3 for 200000 evaluations and $\hat{T}_2$ is the average of the complete

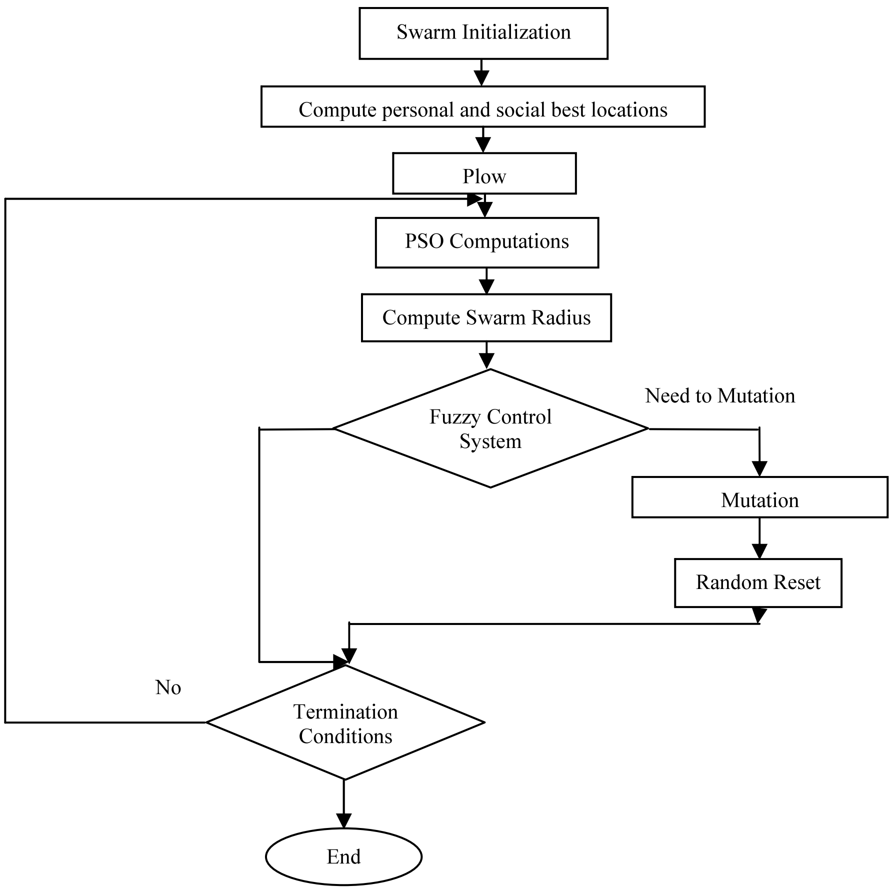

Fig. 6 Flowchart of LADPSO algorithm

```
for i=1 to 1000000
        x= (double) 5.55;
       x=x + x; x=x./2; x=x*x; x=sqrt(x); x=ln(x);
       x=exp(x);  y=x/x;
end
```

Fig. 7 The code for evaluating T_0

computing time for the algorithm with 200000 evaluations for 5 times.

From the results above, it could be said that, our method is promising for improving the standard algorithm, and it was able to locate the global minimum on most of problems, with acceptable accuracy. Also from the results of Table 3 it is clear that our algorithm is more computationally efficient than its standard version

5.2 Real life problem

In this section we have reported the results of applying our algorithm to the Lennard-Jones potential problem. The Lennard-Jones potential (L-J potential or 6-12 potential) is a mathematically simple model that describes the interaction between a pair of neutral atoms or molecules. Minimizing the Lennard-Jones potential is an unconstrained global optimization problem with many local optimum points. Because of its many local optimums the results of algorithms on minimizing this function is very discriminant for comparison of algorithms. In minimizing this function we are seeking to a geometrical arrangement of n atoms with the lowest potential energy. We have reported the average results of our algorithm for 5 to 15 atoms on Table 10 along with the results of some other methods like [36] and its global minimum for comparison. The swarm size is 40 and the function evaluation count is set to $5000 + 3000 * n * (n - 1)$. We have reported the average of results for running the test for 20 times.

Also in Table 11 we have compared the results of our algorithm to one of the most comprehensive studies about Lennard-Jones function optimization [37]. In Table 11 the best value achieved over 10 runs, for various number of atoms has been reported. As we could see, our method has a good performance on Lennard-Jones potential problem.

5.3 Analyzing the results

We have used Mann–Whitney–Wilcoxon (MWW) method [38, 39] to compare the results of SPSO and LADPSO algorithms. MWW is a non-parametric statistical analysis for examining whether one of two samples of independent observations have a tendency to have smaller values than the

Table 4 Error values of SPSO2011 for problems 1-10 (10D)

FEs/function		1	2	3	4	5	6	7	8	9	10
10^3	min	1.1539E+02	1.1509E+03	3.1160E+06	5.3283E+02	1.8704E+03	3.9346E+05	1.3900E+01	2.0423E+01	4.2129E+01	3.2557E+01
	7	3.4795E+02	1.8323E+03	6.7645E+06	1.9885E+03	3.4304E+03	7.7151E+05	4.7139E+01	2.0689E+01	4.9606E+01	5.3311E+01
	13	4.1273E+02	2.3802E+03	1.0634E+07	3.0851E+03	3.7821E+03	1.4911E+06	6.1102E+01	2.0792E+01	5.4903E+01	6.1824E+01
	19	4.5542E+02	2.8692E+03	1.6214E+07	4.3814E+03	4.3999E+03	2.7670E+06	6.9930E+01	2.0842E+01	5.6695E+01	6.7036E+01
	max	6.2131E+02	3.2308E+03	2.4606E+07	6.9924E+03	5.3274E+03	6.4235E+06	1.3045E+02	2.0930E+01	6.7348E+01	7.2116E+01
	mean	3.9501E+02	2.3480E+03	1.1482E+07	3.2882E+03	3.8881E+03	2.0175E+06	6.1812E+01	2.0748E+01	5.3631E+01	5.9921E+01
	std	1.2293E+02	6.1874E+02	6.5327E+06	1.7212E+03	8.1726E+02	1.6213E+06	2.6267E+01	1.3632E-01	5.5598E+00	9.8830E+00
10^4	min	4.8339E-07	1.6306E-05	4.1856E+04	1.2636E-03	4.2929E-03	4.4028E+00	5.6055E-02	2.0150E+01	5.5383E+00	5.7830E+00
	7	7.3028E-07	4.7178E-04	1.3289E+05	8.9430E-03	1.4527E-02	8.5542E+00	1.6503E-01	2.0351E+01	9.8777E+00	1.1343E+01
	13	8.1489E-07	9.7080E-04	1.8476E+05	1.9373E-02	1.9517E-02	1.0229E+02	2.4216E-01	2.0375E+01	1.3254E+01	1.4659E+01
	19	8.6275E-07	2.2922E-03	3.7931E+05	3.7432E-02	4.0156E-02	8.2742E+02	2.9198E-01	2.0523E+01	1.8500E+01	1.7670E+01
	max	9.8329E-07	1.3413E-02	9.7753E+05	1.1559E+00	8.5052E-02	6.7380E+03	4.6971E-01	2.0611E+01	2.5348E+01	2.3827E+01
	mean	7.8889E-07	2.2670E-03	2.7539E+05	7.7618E-02	2.8589E-02	1.1759E+03	2.3943E-01	2.0390E+01	1.4226E+01	1.4858E+01
	std	1.2551E-07	3.3099E-03	2.0503E+05	2.2865E-01	2.1155E-02	2.0525E+03	1.0700E-01	1.3479E-01	5.2504E+00	5.0732E+00
10^5	min	5.3971E-07	4.4858E-07	4.3365E+03	6.4167E-07	6.9608E-07	1.6895E-02	9.6210E-03	2.0112E+01	1.9899E+00	2.0668E+00
	7	7.3741E-07	7.8312E-07	2.1273E+04	8.3495E-07	9.1386E-07	3.0600E-02	2.9562E-02	2.0201E+01	3.9798E+00	3.7657E+00
	13	7.9384E-07	8.7540E-07	3.5677E+04	9.3487E-07	9.4535E-07	4.3523E-02	4.6796E-02	2.0236E+01	4.7817E+00	5.2163E+00
	19	9.0362E-07	9.7090E-07	5.2422E+04	9.7272E-07	9.6729E-07	1.0220E+02	5.6552E-02	2.0302E+01	5.9698E+00	6.2968E+00
	max	9.7847E-07	9.9812E-07	9.0286E+04	9.9980E-07	9.9942E-07	2.7800E+02	5.2246E-01	2.0368E+01	1.5940E+01	9.8310E+00
	mean	8.0304E-07	8.5995E-07	3.7918E+04	8.9410E-07	9.2681E-07	5.0578E+01	6.7519E-02	2.0241E+01	5.6251E+00	5.1169E+00
	std	1.2802E-07	1.3776E-07	2.2458E+04	1.0323E-07	6.8907E-08	8.6887E+01	9.8247E-02	7.1137E-02	3.1687E+00	2.0606E+00

Table 5 Error values of SPSO2011 for problems 1-10 (30D)

FEs/function		1	2	3	4	5	6	7	8	9	10
3×10^3	min	3.1232E+02	1.2007E+04	2.4722E+07	1.8195E+04	6.1491E+03	1.9461E+06	1.4521E+02	2.0998E+01	1.7200E+02	1.8029E+02
	7	7.0195E+02	1.7960E+04	3.6941E+07	2.2750E+04	7.7983E+03	6.2680E+06	2.4781E+02	2.1101E+01	2.1750E+02	2.2878E+02
	13	8.5861E+02	1.8943E+04	4.2360E+07	2.5047E+04	8.6130E+03	1.2253E+07	3.1822E+02	2.1156E+01	2.2487E+02	2.4412E+02
	19	1.1197E+03	2.0358E+04	6.2464E+07	2.8751E+04	9.5131E+03	2.7609E+07	3.9816E+02	2.1189E+01	2.3701E+02	2.6437E+02
	max	1.7206E+03	2.7707E+04	8.7708E+07	3.8613E+04	1.1667E+04	5.9657E+07	8.2958E+02	2.1237E+01	2.5997E+02	2.9169E+02
	mean	9.1242E+02	1.9402E+04	4.7832E+07	2.5673E+04	8.6904E+03	1.8587E+07	3.2889E+02	2.1144E+01	2.2512E+02	2.4327E+02
	std	3.1847E+02	4.1023E+03	1.6673E+07	4.9048E+03	1.5265E+03	1.6349E+07	1.4270E+02	6.2536E-02	1.7861E+01	2.8578E+01
3×10^4	min	8.0126E-07	1.7629E+01	1.3434E+06	4.1280E+03	2.9619E+03	2.4856E+01	4.6281E-02	2.0807E+01	3.5818E+01	3.5818E+01
	7	8.7817E-07	3.2873E+01	2.0793E+06	5.3854E+03	4.3196E+03	2.0192E+02	1.3710E-01	2.0920E+01	4.6825E+01	5.9632E+01
	13	9.5457E-07	6.4617E+01	2.5578E+06	6.3613E+03	4.9577E+03	4.3300E+02	2.1258E-01	2.0958E+01	6.1966E+01	9.1185E+01
	19	9.7326E-07	8.7396E+01	2.9987E+06	8.2697E+03	5.3113E+03	1.1023E+03	3.0652E-01	2.1007E+01	9.5180E+01	1.2468E+02
	max	9.9903E-07	1.6048E+02	4.4525E+06	1.1701E+04	6.1753E+03	1.0310E+04	9.0820E-01	2.1076E+01	1.5628E+02	1.8210E+02
	mean	9.2528E-07	6.6370E+01	2.5926E+06	6.8501E+03	4.9010E+03	1.7795E+03	2.5857E-01	2.0957E+01	7.3631E+01	9.5081E+01
	std	6.1335E-08	3.8012E+01	7.5200E+05	2.0614E+03	8.1684E+02	2.8754E+03	2.0178E-01	7.3935E-02	3.3902E+01	4.1887E+01
3×10^5	min	7.8982E-07	9.3296E-07	7.9207E+04	7.3903E+00	3.3813E+03	1.7140E+01	9.9453E-03	2.0722E+01	2.4874E+01	2.7859E+01
	7	9.0581E-07	9.8250E-07	1.8243E+05	2.3050E+01	4.2821E+03	2.0745E+01	9.9990E-03	2.0820E+01	4.3778E+01	4.4773E+01
	13	9.4793E-07	9.8978E-07	2.2671E+05	3.6253E+01	4.6013E+03	1.4877E+02	2.4573E-02	2.0850E+01	5.6635E+01	5.1738E+01
	19	9.8749E-07	9.9408E-07	3.0744E+05	6.9441E+01	5.1330E+03	7.2051E+02	3.6894E-02	2.0891E+01	6.5667E+01	5.6713E+01
	max	9.9924E-07	9.9847E-07	5.0708E+05	2.3162E+02	6.3136E+03	4.4044E+03	8.1087E-02	2.1005E+01	1.8047E+02	7.6612E+01
	mean	9.3626E-07	9.8438E-07	2.4853E+05	5.4290E+01	4.6923E+03	4.6342E+02	2.5243E-02	2.0856E+01	6.4551E+01	5.3327E+01
	std	5.46211E-08	1.594E-08	109334.06	48.412297	712.72878	892.18201	0.0179063	0.0674975	34.209586	11.907555

Table 6 Error values of LADPSO for problems 1-10 (10D)

FEs/function		1	2	3	4	5	6	7	8	9	10
10^3	min	3.1474E+00	5.9716E+03	6.8073E+06	5.6997E+03	2.5319E+03	3.9080E+04	7.4502E+02	2.0253E+01	4.7194E+00	4.7408E+01
	7	8.3217E+00	1.0305E+04	1.0182E+07	1.2180E+04	4.6451E+03	2.7180E+05	1.1178E+03	2.0452E+01	7.1692E+00	6.3064E+01
	13	1.6061E+01	1.3250E+04	1.3308E+07	1.6530E+04	5.6153E+03	3.8131E+05	1.3390E+03	2.0494E+01	8.7066E+00	6.8110E+01
	19	4.2378E+01	1.5201E+04	1.7371E+07	2.0073E+04	6.5441E+03	4.6673E+05	1.3571E+03	2.0551E+01	9.3167E+00	7.1814E+01
	max	8.3800E+01	1.7989E+04	2.9986E+07	2.1586E+04	7.4298E+03	6.5848E+05	1.3699E+03	2.0619E+01	1.0383E+01	7.4507E+01
	mean	2.8667E+01	1.2582E+04	1.4375E+07	1.5536E+04	5.5389E+03	3.5928E+05	1.2207E+03	2.0493E+01	8.4098E+00	6.5700E+01
	std	2.7869E+01	3.4982E+03	5.9745E+06	4.9148E+03	1.3067E+03	1.7065E+05	1.9838E+02	9.7179E-02	1.4191E+00	7.8544E+00
10^4	min	3.8699E-01	7.4020E+01	7.1449E+05	1.7105E+02	6.7014E+01	1.9721E+02	1.0679E+00	2.0260E+01	3.7901E+00	1.7571E+01
	7	1.1261E+00	1.3100E+02	1.6784E+06	3.4252E+02	1.3931E+02	1.1367E+03	1.2441E+00	2.0482E+01	5.4865E+00	3.2576E+01
	13	1.8927E+00	2.0350E+02	2.5234E+06	4.9128E+02	2.7293E+02	1.7516E+03	1.3780E+00	2.0548E+01	6.2237E+00	3.9626E+01
	19	4.0834E+00	3.1862E+02	3.9012E+06	6.1154E+02	3.8794E+02	3.0385E+03	1.5380E+00	2.0571E+01	6.7404E+00	4.5093E+01
	max	5.1483E+00	5.1325E+02	5.0082E+06	8.2293E+02	6.0162E+02	6.5563E+03	2.5311E+00	2.0623E+01	8.4483E+00	5.0536E+01
	mean	2.4352E+00	2.3931E+02	2.7851E+06	4.6938E+02	2.9016E+02	2.2883E+03	1.4854E+00	2.0512E+01	6.1684E+00	3.7599E+01
	std	1.5964E+00	1.3629E+02	1.2217E+06	1.9634E+02	1.5989E+02	1.8326E+03	3.7602E-01	9.2711E-02	1.1628E+00	8.9723E+00
10^5	min	2.5605E-07	4.0498E-07	5.5884E+03	7.8660E-07	7.2006E-07	4.3130E-01	9.9967E-03	2.0048E+01	3.9119E-03	1.9899E+00
	7	7.7901E-07	9.1245E-07	2.6539E+04	9.4924E-07	8.1613E-07	1.6660E+00	2.4590E-02	2.0304E+01	7.5708E-01	2.9854E+00
	13	8.6851E-07	9.5076E-07	3.8194E+04	9.7118E-07	9.3372E-07	3.5934E+00	4.4283E-02	2.0334E+01	2.4443E+00	3.9798E+00
	19	9.2751E-07	9.8094E-07	7.1675E+04	9.8974E-07	9.7750E-07	2.3480E+01	6.8866E-02	2.0365E+01	3.3726E+00	5.9697E+00
	max	9.5170E-07	9.8620E-07	8.4803E+04	9.9955E-07	9.9602E-07	5.6094E+01	7.8775E-02	2.0381E+01	3.8897E+00	6.9651E+00
	mean	8.1585E-07	9.0734E-07	4.7083E+04	9.4849E-07	8.9888E-07	1.5314E+01	4.4214E-02	2.0316E+01	2.0849E+00	4.2733E+00
	std	1.6548E-07	1.3457E-07	2.5974E+04	6.1095E-08	8.9936E-08	1.7995E+01	2.2514E-02	7.1950E-02	1.4213E+00	1.5801E+00

Table 7 Error values of LADPSO for problems 1-10 (30D)

FEs/function		1	2	3	4	5	6	7	8	9	10
3×10^3	min	2.0779E+01	5.6683E+03	8.7951E+06	7.4908E+03	1.5929E+04	3.4928E+06	4.6781E+03	2.0773E+01	3.4684E+01	3.2397E+02
	7	3.5562E+01	9.2440E+03	1.6708E+07	1.0049E+04	1.9252E+04	4.3747E+06	5.1484E+03	2.0972E+01	3.8951E+01	3.5364E+02
	13	4.8565E+01	1.1642E+04	2.0291E+07	1.1824E+04	2.1243E+04	6.1217E+06	5.1837E+03	2.1014E+01	4.0360E+01	3.7186E+02
	19	5.3055E+01	1.2758E+04	2.2463E+07	1.3230E+04	2.2523E+04	7.2519E+06	5.2024E+03	2.1047E+01	4.2480E+01	3.9072E+02
	max	1.0123E+02	1.6971E+04	2.7974E+07	1.5612E+04	2.4208E+04	8.3499E+06	5.2395E+03	2.1073E+01	4.3898E+01	4.0774E+02
	mean	4.7321E+01	1.1217E+04	1.9784E+07	1.1639E+04	2.0927E+04	5.9427E+06	5.1536E+03	2.1002E+01	4.0496E+01	3.7342E+02
	std	1.7208E+01	2.7911E+03	4.7431E+06	2.1348E+03	2.3593E+03	1.5731E+06	1.1360E+02	6.3800E-02	2.2145E+00	2.2832E+01
3×10^4	min	4.3690E-01	3.4490E+01	7.0254E+05	9.4690E+02	4.5795E+03	2.9143E+02	1.3042E+00	2.0847E+01	9.0018E+00	3.6760E+01
	7	7.3626E-01	5.4928E+01	1.3057E+06	1.3878E+03	5.3579E+03	4.8023E+02	1.9922E+00	2.0985E+01	1.9135E+01	5.6085E+01
	13	9.9280E-01	7.0801E+01	1.7703E+06	1.7559E+03	5.5637E+03	7.1231E+02	3.0081E+00	2.1016E+01	2.3402E+01	8.1358E+01
	19	1.1016E+00	7.6666E+01	2.2772E+06	1.9259E+03	6.5134E+03	1.4329E+03	3.4617E+00	2.1044E+01	2.7773E+01	8.9431E+01
	max	1.9413E+00	9.6410E+01	2.9027E+06	2.2841E+03	7.1684E+03	6.5288E+03	4.8318E+00	2.1088E+01	3.6874E+01	9.7108E+01
	mean	1.0498E+00	6.6754E+01	1.7700E+06	1.6702E+03	5.9192E+03	1.6087E+03	2.7810E+00	2.1011E+01	2.3740E+01	7.2203E+01
	std	4.1162E-01	1.6525E+01	6.2000E+05	4.0073E+02	7.6900E+02	1.9146E+03	9.3192E-01	5.5064E-02	6.5601E+00	1.8960E+01
3×10^5	min	7.1895E-07	8.2668E-07	4.0245E+04	3.5302E+00	2.5340E+03	4.1711E+00	9.9342E-03	2.0750E+01	8.8636E-03	2.4874E+01
	7	9.3850E-07	1.5025E-06	6.2722E+04	5.4150E+00	3.8358E+03	8.0561E+01	9.9972E-03	2.0870E+01	1.2714E+01	3.6813E+01
	13	9.7278E-07	1.9957E-06	7.1059E+04	7.4276E+00	4.1526E+03	9.4653E+01	9.9998E-03	2.0897E+01	1.5968E+01	4.2783E+01
	19	9.8254E-07	2.6610E-06	8.3794E+04	9.1478E+00	4.4287E+03	1.3062E+02	2.4573E-02	2.0939E+01	1.8953E+01	5.1738E+01
	max	9.9193E-07	4.1496E-06	1.0888E+05	1.2313E+01	5.0847E+03	1.6340E+02	2.9552E-02	2.0966E+01	2.3695E+01	6.5668E+01
	mean	9.5249E-07	2.1342E-06	7.3452E+04	7.4768E+00	4.0612E+03	9.4867E+01	1.6313E-02	2.0896E+01	1.4859E+01	4.5014E+01
	std	5.6076E-08	9.3099E-07	1.7199E+04	2.4764E+00	6.1298E+02	4.8863E+01	7.7682E-03	5.3580E-02	6.2313E+00	1.1263E+01

Table 8 Error values of [32] for problems 1-10 (10D)

FES	Prob.	1	2	3	4	5	6	7	8	9	10
10^3	min	1.88E-3	6.22E+00	1.07E+06	8.86E+01	6.71E+00	1.12E+01	6.51E-1	2.05E+01	5.02E+00	6.11E+00
	7th	7.56E-3	2.40E+01	5.73E+06	5.81E+02	1.25E+01	1.27E+02	7.28E-1	2.07E+01	1.43E+01	3.63E+01
	med.	1.34E-2	4.35E+01	1.14E+07	1.58E+03	2.16E+01	1.91E+03	9.06E-1	2.08E+01	2.75E+01	4.56E+01
	19th	2.46E-2	8.36E+01	2.39E+07	4.28E+03	3.76E+01	3.15E+03	9.74E-1	2.08E+01	4.70E+01	5.14E+01
	max	4.68E-2	1.70E+02	5.67E+07	1.75E+04	6.71E+01	9.89E+05	1.11E+00	2.09E+01	5.91E+01	6.37E+01
	mean	1.70E-2	5.83E+01	1.68E+07	3.00E+03	2.81E+01	4.30E+04	8.69E-1	2.08E+01	3.07E+01	4.17E+01
	std	1.20E-2	4.74E+01	1.59E+07	3.83E+03	1.85E+01	1.97E+05	1.38E-1	9.93E-2	1.72E+01	1.57E+01
10^4	min	1.84E-9	2.21E-9	2.21E-9	1.71E-9	2.46E-9	3.29E-9	9.31E-10	2.03E+01	1.99E+00	2.98E+00
	7th	3.75E-9	3.27E-9	4.61E-9	3.85E-9	5.02E-9	4.49E-9	2.81E-9	2.05E+01	4.97E+00	4.97E+00
	med.	5.65E-9	4.53E-9	5.51E-9	4.78E-9	6.33E-9	7.37E-9	5.46E-9	2.06E+01	5.97E+00	6.96E+00
	19th	6.42E-9	5.71E-9	6.58E-9	6.46E-9	8.60E-9	3.99E+00	7.77E-9	2.06E+01	7.96E+00	7.96E+00
	max	9.34E-9	7.67E-9	9.66E-9	7.80E-9	9.84E-9	2.63E+02	1.48E-2	2.07E+01	1.09E+01	1.59E+01
	mean	5.20E-9	4.70E-9	5.60E-9	5.02E-9	6.58E-9	1.17E+01	2.27E-3	2.05E+01	6.21E+00	7.16E+00
	std	1.94E-9	1.56E-9	1.93E-9	1.71E-9	2.17E-9	5.24E+01	4.32E-3	8.62E-2	2.10E+00	3.12E+00
10^5	min	1.84E-9	2.21E-9	2.21E-9	1.71E-9	2.46E-9	1.44E-9	6.22E-10	2.00E+01	1.52E-10	1.50E-10
	7th	3.75E-9	3.27E-9	4.61E-9	3.85E-9	5.02E-9	3.81E-9	1.65E-9	2.00E+01	3.46E-10	3.34E-10
	med.	5.65E-9	4.53E-9	5.51E-9	4.78E-9	6.33E-9	4.69E-9	2.84E-9	2.00E+01	6.14E-10	5.64E-10
	19th	6.42E-9	5.71E-9	6.58E-9	6.46E-9	8.60E-9	5.67E-9	5.46E-9	2.00E+01	3.50E-9	1.08E-9
	max	9.34E-9	7.67E-9	9.66E-9	7.80E-9	9.84E-9	8.13E-9	7.77E-9	2.00E+01	9.95E-1	9.95E-1
	mean	5.20E-9	4.70E-9	5.60E-9	5.02E-9	6.58E-9	4.87E-9	3.31E-9	2.00E+01	2.39E-1	7.96E-2
	std	1.94E-9	1.56E-9	1.93E-9	1.71E-9	2.17E-9	1.66E-9	2.02E-9	3.89E-3	4.34E-1	2.75E-1

Table 9 Error values of [32] for problems 1-10 (30D)

FES	Prob.	1	2	3	4	5	6	7	8	9	10
3×10^3	min	4.49 E+2	1.12 E+5	3.84 E+8	6.13 E+5	6.21 E+3	3.26 E+6	4.10 E+1	2.12 E+1	2.19 E+2	2.43 E+2
	7th	5.48 E+2	1.73 E+5	8.00 E+8	1.12 E+6	9.57 E+3	7.31 E+6	9.39 E+1	2.12 E+1	2.45 E+2	2.65 E+2
	med.	7.40 E+2	2.35 E+5	1.00 E+9	1.44 E+6	1.09 E+4	1.23 E+7	1.20 E+2	2.12 E+1	2.50 E+2	2.74 E+2
	19th	1.04 E+3	2.94 E+5	1.38 E+9	1.87 E+6	1.24 E+4	1.99 E+7	1.59 E+2	2.13 E+1	2.66 E+2	2.88 E+2
	max	1.61 E+3	3.83 E+5	2.07 E+9	3.29 E+6	1.42 E+4	6.81 E+7	3.26 E+2	2.13 E+1	2.87 E+2	3.08 E+2
	mean	8.16 E+2	2.39 E+5	1.07 E+9	1.55 E+6	1.07 E+4	1.77 E+7	1.39 E+2	2.12 E+1	2.53 E+2	2.77 E+2
	std	3.01 E+2	7.80 E+4	4.43 E+8	6.15 E+5	2.13 E+3	1.62 E+7	7.17 E+1	4.35 E-2	1.65 E+1	1.90 E+1
3×10^4	min	3.98 E-9	2.29 E-3	1.24 E+6	4.88 E+2	5.00 E-2	1.77 E+1	3.93 E-9	2.10 E+1	2.39 E+1	3.08 E+1
	7th	4.70 E-9	1.60 E-2	3.41 E+6	1.46 E+3	1.00 E+3	2.28 E+1	4.85 E-9	2.11 E+1	4.28 E+1	4.38 E+1
	med.	5.20 E-9	2.57 E-2	4.90 E+6	3.51 E+3	1.32 E+3	2.58 E+1	5.69 E-9	2.11 E+1	4.88 E+1	5.27 E+1
	19th	6.10 E-9	3.99 E-2	8.21 E+6	5.18 E+4	2.04 E+3	2.22 E+2	6.95 E-9	2.11 E+1	5.47 E+1	5.87 E+1
	max	7.51 E-9	7.49 E-2	1.42 E+7	2.88 E+5	3.20 E+3	2.66 E+3	2.46 E-2	2.12 E+1	7.96 E+1	8.26 E+1
	mean	5.42 E-9	2.73 E-2	6.11 E+6	4.26 E+4	1.51 E+3	4.60 E+2	1.77 E-3	2.11 E+1	4.78 E+1	5.14 E+1
	std	9.80 E−10	1.79 E-2	3.79 E+6	7.43 E+4	8.82 E+2	8.29 E+2	5.52 E-3	4.04 E-2	1.15 E+1	1.25 E+1
3×10^5	min	3.98 E-9	4.48 E-9	4.07 E-9	6.06 E-9	7.15 E-9	4.05 E-9	1.76 E-9	2.00 E+1	2.98 E+0	9.95 E-1
	7th	4.70 E-9	5.59 E-9	4.78 E-9	8.75 E-9	8.06 E-9	5.31 E-9	4.59 E-9	2.02 E+1	4.97 E+0	5.97 E+0
	med.	5.20 E-9	6.13 E-9	5.44 E-9	1.93 E+1	8.61 E-9	6.32 E-9	5.41 E-9	2.09 E+1	6.96 E+0	6.96 E+0
	19th	6.10 E-9	6.85 E-9	6.16 E-9	2.72 E+3	9.34 E-9	7.52 E-9	6.17 E-9	2.10 E+1	8.95 E+0	8.95 E+0
	max	7.51 E-9	8.41 E-9	8.66 E-9	1.57 E+5	2.51 E-6	3.99 E+0	7.81 E-9	2.11 E+1	1.19 E+1	1.09 E+1
	mean	5.42 E-9	6.22 E-9	5.55 E-9	1.27 E+4	1.08 E-7	4.78 E-1	5.31 E-9	2.07 E+1	6.89 E+0	6.96 E+0
	std	9.80 E-10	8.95 E−10	1.09 E-9	3.59 E+4	4.99 E-7	1.32 E+0	1.41 E-9	4.28 E-1	2.22 E+0	2.45 E+0

Table 3 Computational complexity of LADPSO

Dim	T_0	T_1	$\hat{T}_2$		$(\hat{T}_2 - T_1)/T_0$	
Method	–	–	SPSO	LADPSO	SPSO	LADPSO
10	1.559	40.200	79.056	65.886	24.912	16.468
30	1.559	44.841	146.546	82.844	65.209	24.366
50	1.559	51.661	222.774	92.938	109.711	26.465

Table 10 Results of LADPSO for the Lennard-Jones potential problem

N/Method	LADPSO	SPSO	Results of [34]	Minimum
5	−9.09772	−9.01407	−9.10385	−9.10385
6	−12.5239	−11.811	−12.7121	−12.7121
7	−16.2533	−14.4687	−16.5054	−16.5054
8	−19.4855	−17.1723	−19.2084	−19.8215
9	−23.2127	−19.5568	−22.9690	−24.1134
10	−27.3535	−22.1338	−26.8424	−28.4225
11	−31.6974	−23.8806	−31.5629	−32.7656
12	−36.4019	−24.399	−36.3692	−37.9676
13	−40.9243	−27.2597	−41.1379	−44.3268
14	−45.3636	−29.5903	−44.3883	−47.8452
15	−49.7374	−32.774	−49.7777	−52.3226

other. The test gives a p-value which indicates whether we could reject the null hypothesis (two samples have equal medians) or not. We have performed MWW for all benchmark problems separately on significance level of 0.05. The results of our statistical analysis are shown on Table 12. We have reported p-value of the test along with rejection or not rejection of null hypothesis. In Table 12, $h = 0$ indicates failure of the rejection and $h = 1$ indicates rejection of hypothesis on significance level of 0.05.

Based on Table 12, we could see that on most cases the difference between results is significant and it gets more tangible when the complexity of problem increased. This observation suggests that the LADPSO seems promising to improve the base algorithm especially on high dimensional problems.

6 Conclusion and future works

In this paper, we proposed a novel extension to PSO optimization method, called LADPSO algorithm, through combining two operators to the standard PSO. The suggested approach utilizes the fuzzy logic to conduct the standard PSO on global optimization problems. We also compared accuracy and robustness of our approach with standard PSO, and some other methods by using some well-known benchmark functions. It has been depicted that our algorithm

Table 11 Results of LADPSO for the Lennard-Jones potential problem

N	App II	App III.	Steep	BFGS	DFP	LGO BB	LGO Rd	Ref. E	SPSO2011	LADPSO
6	−1.2303E+01	−1.2712E+01	−1.2303E+01	−1.2303E+01	−1.2303E+01	−1.2712E+01	−1.2303E+01	−1.2712E+01	−1.2711E+01	−1.2712E+01
7	−1.5533E+01	−1.6505E+01	−1.5499E+01	−1.5500E+01	−1.5500E+01	−1.5533E+01	−1.5533E+01	−1.6505E+01	−1.6505E+01	−1.6505E+01
8	−1.9822E+01	−1.9822E+01	−1.8829E+01	−1.8829E+01	−1.8829E+01	−1.8778E+01	−1.9822E+01	−1.9822E+01	−1.9821E+01	−1.9821E+01
9	−2.4113E+01	−2.4113E+01	−2.2181E+01	−2.2181E+01	−2.2181E+01	−2.3044E+01	−2.2156E+01	−2.4113E+01	−2.2162E+01	−2.4113E+01
10	−2.7545E+01	−2.8423E+01	−2.6943E+01	−2.6955E+01	−2.6939E+01	−2.7274E+01	−2.6623E+01	−2.8423E+01	−2.5095E+01	−2.7556E+01
11	−3.2766E+01	−3.2766E+01	−3.2766E+01	−3.2766E+01	−3.2766E+01	−2.9997E+01	−3.0954E+01	−3.2766E+01	−2.8856E+01	−3.1914E+01
12	−3.6178E+01	−3.7968E+01	−3.7968E+01	−3.7968E+01	−3.7968E+01	−3.5171E+01	−3.5462E+01	−3.7968E+01	−2.9423E+01	−3.7968E+01
13	−4.1394E+01	−4.4327E+01	−4.4327E+01	−4.4327E+01	−4.4327E+01	−3.7238E+01	−4.1356E+01	−4.4327E+01	−3.1473E+01	−4.4327E+01
14	−4.7845E+01	−4.7845E+01	−4.7845E+01	−4.7845E+01	−4.7845E+01	−4.4874E+01	−4.6935E+01	−4.7845E+01	−3.2713E+01	−4.7845E+01
15	−5.2323E+01	−5.2323E+01	−5.2323E+01	−5.2323E+01	−5.2323E+01	−4.9222E+01	−4.7583E+01	−5.2323E+01	−3.8195E+01	−5.2323E+01

Table 12 Results of Mann–Whitney–Wilcoxon test (with significance level of 0.05) on benchmark problems

CEC 2005 (dim = 10)			CEC 2005 (dim = 30)			Lennard-Jones		
Problem	p	h	Problem	p	h	Number of atoms	p	h
1	4.4922E-01	0	1	2.6874E-01	0	5	2.6162E-01	0
2	2.9475E-01	0	2	3.3440E-07	1	6	1.4359E-02	1
3	2.3656E-01	0	3	3.6690E-09	1	7	7.4064E-05	1
4	5.0032E-02	0	4	6.5743E-09	1	8	2.0616E-06	1
5	5.0945E-01	0	5	2.3172E-03	1	9	7.8980E-08	1
6	6.5289E-02	0	6	1.2532E-01	0	10	6.7956E-08	1
7	6.9080E-01	0	7	7.4230E-02	0	11	6.7956E-08	1
8	1.5460E-04	1	8	1.1029E-02	1	12	6.7956E-08	1
9	2.4478E-07	1	9	1.4157E-09	1	13	6.7956E-08	1
10	1.6231E-01	0	10	1.7005E-02	1	14	6.7956E-08	1
–	–	–	–	–	–	15	6.7956E-08	1

(LADPSO) has better performance in accuracy, stability and robustness. The future work includes the research on using similar approaches for solving multi-objective and dynamic optimization problems. Also researching about designing more intelligent systems for controlling the usage of additional operators within standard algorithm seems to be fruitful. In this paper we have used a fuzzy control system to improve standard PSO algorithm with 2 operators, it is possible to examine this approach for other operators and optimization algorithms. Furthermore it seems that some good researches could be done about improving the suggested algorithm to solve constrained optimization problems.

Acknowledgement The authors are very grateful to the referees for their excellent and many helpful comments, suggestions and corrections.

References

1. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks, Piscataway, NJ, pp 1942–1948
2. Engelbrecht AP (2007) Computational intelligence, 2nd edn. Wiley, New York
3. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73
4. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput
5. Pant M, Radha T, Singh VP (2007) Particle swarm optimization: experimenting the distributions of random numbers. In: 3rd Indian int conf on artificial intelligence (IICAI 2007), pp 412–420
6. Pant M, Thangaraj R, Abraham A (2008) Improved particle swarm optimization with low-discrepancy sequences. In: IEEE cong on evolutionary computation (CEC 2008), Hong Kong
7. Norouzzadeh MS, Ahmadzadeh MR, Palhang M (2010) Plowing PSO: anovel approach to effectively initializing particle swarm optimization. In: Proceeding of 3rd IEEE international conference on computer science and information technology, Chengdu, China, vol 1, pp 705–709
8. Mendes R, Kennedy J, Neves J (2005) The fully informed particle swarm: simpler, maybe better. IEEE Trans Evol Comput, 1(1)
9. Suganthan PN (1999) Particle swarm optimiser with neighborhood operator. In: Proceedings of the IEEE congress on evolutionary computation, pp 1958–1962
10. Kennedy J, Mendes R (2002) Population structure and particle performance. In: Proceedings of the IEEE congress on evolutionary computation, pp 1671–1676
11. Kennedy J (2003) Bare bones particle swarms. In: Proceedings of the IEEE swarm intelligence symposium, pp 80–87
12. Standard PSO 2007 and 2011, http://particleswarm.info
13. Shi Y, Eberhart RC (2001) Fuzzy adaptive particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation, pp 101–106
14. Venter G, Sobieszczanski-Sobieski J (2003) Particle swarm optimization. J Am Inst Aeronaut Astronaut 41(8):1583–1589
15. Clerc M (2001) Think locally, act locally: the way of life of cheap-PSO, an adaptive PSO. http://clerc.maurice.free.fr/pso/. Technical report
16. Zheng Y, Ma L, Zhang L, Qian J (2003) On the convergence analysis and parameter selection in particle swarm optimization. In: Proceedings of the international conference on machine learning and cybernetics, pp 1802–1807

17. Chen M-R, Lu Y-Z, Luo Q (2010) A novel hybrid algorithm with marriage of particle swarm optimization and extremal optimization. Appl Soft Comput J 10(2):367–373
18. Lim A, Lin J, Xiao F (2007) Particle swarm optimization and hill climbing for the bandwidth minimization problem. Int J Appl Intell 26:175–182
19. Shuang B, Chen J, Li Z (2011) Study on hybrid PS-ACO algorithm. Int J Appl Intell 34:64–73
20. Angeline PJ (1998) Using selection to improve particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation, pp 84–89
21. Pant M, Thangaraj R, Abraham A (2007) A new PSO algorithm with crossover operator for global optimization problems. In: 2nd international workshop on hybrid artificial intelligence systems
22. Higashi H, Iba H (2003) Particle swarm optimization with gaussian mutation. In: Proceedings of the IEEE swarm intelligence symposium, pp 72–79
23. Pant M, Radha T, Singh VP A new diversity based particle swarm optimization using Gaussian mutation. Int J Math Model, Simul Appl
24. Li C, Yang S, Korejo I An adaptive mutation operator for particle swarm optimization
25. Li C, Liu Y, Zhou A, Kang L, Wang H A fast particle swarm optimization algorithm with Cauchy mutation and natural selection strategy
26. van den Bergh F, Engelbrecht AP (2000) Cooperative learning in neural networks using particle swarm optimizers. S Afr Comput J 26:84–90
27. Silva A, Neves A, Costa E (2002) An empirical comparison of particle swarm and predator prey optimisation. In: Proceedings of the thirteenth Irish conference on artificial intelligence and cognitive science, pp 103–110
28. Xie X, Zhang W, Yang Z (2002) Adaptive particle swarm optimization on individual level. In: Proceedings of the sixth international conference on signal processing, pp 1215–1218
29. Xie X, Zhang W, Yang Z (2002) A dissipative particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation, pp 1456–1461
30. van den Bergh F (2002) An analysis of particle swarm optimizers. In PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa
31. Blackwell TM, Bentley PJ (2002) Dynamic search with charged swarms. In: Proceedings of the genetic and evolutionary computation conference, pp 19–26
32. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen Y-P, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Nanyang Technological University, Singapore and Kanpur Genetic Algorithms Laboratory, IIT Kanpur, Technical Report
33. Auger A, Hansen N (2005) A restart CMA evolution strategy with increasing population size. In: IEEE congress on evolutionary computation (CEC2005), vol 2, pp 1785–1791
34. Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: IEEE congress on evolutionary computation (CEC2005), vol 2, pp 1785–1791
35. Qin AK, Suganthan PN (2005) Dynamic multi-swarm particle swarm optimizer with local search. In: IEEE congress on evolutionary computation (CEC2005), vol 1, pp 522–528
36. Gockenbach MS, Kearsley AJ, Symes WW (1997) An infeasible point method for minimizing the Lennard-Jones potential. Comput Optim Appl 8(3):273–286
37. Fan E (2002) Global optimization of Lennard-Jones atomic clusters. MSc thesis, McMaster University
38. Gibbons JD (1985) Nonparametric statistical inference. Marcel Dekker, New York
39. Hollander M, Wolfe DA (1999) Nonparametric statistical methods. Wiley, Hoboken

Mohammad Sadegh Norouzzadeh received his B.S. degree in Software Engineering from the Tarbiat Moallem University, Iran, in 2007 and the Master degree in Artificial Intelligence from Isfahan University of Technology, in 2009. Currently he is Ph.D. student of Artificial Intelligence in Amirkabir University of Technology. His current research interests include computational intelligence, machine learning and computer vision.

Mohammad Reza Ahmadzadeh received his B.Sc. degree in Electronic Engineering from the Ferdowsi University of Mashhad, Iran in 1989 and the M.Sc. degree in Electronic Engineering from the University of Tarbiat Modarres, Tehran in 1992. He received his Ph.D. from University of Surrey, UK in 2001. He was a lecturer at Shiraz University from 1993–1997, and an Assistant Professor from 2001–2004. He is an Assistant Professor of Electrical Engineering at Isfahan University of Technology, Iran from 2004. His research interests include reasoning with uncertainty, pattern recognition, image processing, expert systems, information fusion and neural networks.

Maziar Palhang received his B.Sc. in Computer Hardware from Sharif University of Technology, Tehran, Iran. He received his M.Comp.Sc. and Ph.D. from the University of New South Wales, Sydney, Australia. He was a Postdoctoral fellow in UNSW as well. He later joined Electrical and Computer Engineering Department of Isfahan University of Technology, Isfahan, Iran. He has been the Chairman of Humanoid league of IranOpen Competitions. His research interests are Machine Learning, Computer Vision, and Robotics.