TELKOMNIKA, Vol.11, No.4, December 2013, pp. 699~714
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v11i4.1639

Optimization of Membership Functions for the Fuzzy Controllers of the Water Tank and Inverted Pendulum with Differents PSO Variants

Resffa Fierro, Oscar Castillo, Fevrier Valdez, Patricia Melin*
Tijuana Institute of Technology, Tijuana, México
*Corresponding author, e-mail: pmelin@tectijuana.mx

Abstrak

Pada paper ini metaheuristik dari optimasi particle swarm dan dua jenis variannya, yaitu bobot inersia dan koefisien konstriksi digunakan sebagai strategi optimisasi untuk perancangan fungsi keanggotaan yang optimal pada sistem kendali fuzi untuk permasalahan tangki air dan pendulum terbalik. Setiap varian memiliki keuntunngannya masing-masing pada sisi algoritmanya, memungkinkan ekslporasi dan eksploitasi dengan cara yang berbeda-bede, selanjutnya hal ini memungkinkan ditemukannya solusi yang optimal denagn cara yang lebih baik

***Kata kunci**: optimasi, kendali fuzi, varian optimasi particle swarm, tangki air dan pendulum terbalik*

Abstract

In this paper the particle swarm optimization metaheuristic and two of its variants (inertia weight and constriction coefficient) are used as an optimization strategy for the design of optimal membership functions of fuzzy control systems for the water tank and inverted pendulum benchmark problems. Each variant has its own advantages in the algorithm, allowing the exploration and exploitation in different ways and this allows finding the optimal solution in a better way.

***Keywords**: optimization, fuzzy control, particle swarm optimization variants, water tank and inverted pendulum*

1. Introduction

At present time and within the computer science area, optimization is a key issue specifically in real-world problems. In analyzing these problems there is the difficulty of ensuring a good solution in a reasonably short time.

There are a variety of methods used to solve optimization problems. In particular, it is interesting to find heuristics that are approximate methods of solution, using an iterative process to guide the search for solutions, thus combining different concepts derived from fields such as Artificial Intelligence, Biological Evolution and Collective Intelligence.

This work is no exception in that we use the metaheuristic of Optimization by Swarm of Particles (PSO - Particle Swarm Optimization) developed by Kennedy and Eberhart in 1995 [1], where this is inspired by the behavior of flocks of birds, of fish and bees. This algorithm will be considered in this paper along with two of its variants, inertia and constriction. In the literature it has been mentioned that since this metaheuristic appeared in 1995 to the present it has had a major impact in the optimization of complex problems. Fuzzy Logic was initially proposed in 1965, at the University of California at Berkeley, by Lotfi A. Zadeh [2]. Fuzzy Logic represents the common knowledge of linguistic qualitative and not necessarily quantitative mathematical language through fuzzy set theory and functions associated with them.

The rest of the paper is organized as follows. Section 2 and 3 discusses the variants of PSO (inertia and constriction coefficient) giving a brief definition of how they work and how their equations are used. Section 4 and 5 shows the architecture and methodology of how they work, also tables and graphs the best error found and a comparison chart with other algorithms are presented. In Section 6 shows simulation results of water tank and inverted pendulum. In Section 7 shows results using only inertia in the case of the water tank. In Section 8 we can find

Received August 30, 2013; Revised October 19, 2013; Accepted October 27, 2013

the Conclusions.

2. Metaheuristic Optimization by Swarms of Particles

Basically the PSO metaheuristic is based on an iterative algorithm where a population of individuals is called Swarm and the individuals are called particles, which fly over a search space to find optimal solutions, where each particle is a possible solution to a particular problem.

More precisely we have particles flying in a multidimensional space, where the position of each particle is adjusted according to its own experience and that of their neighbors. It is a technique inspired by the flight patterns of birds; this simulates the movements of a flock trying to find food and it was developed initially by Kennedy and Eberhart in 1995 [1], [3-6].

We have that x_i (t) is the position of the particle in the search space in a time t, and unless otherwise indicated, t denotes the time steps. The position of the particle is the change in velocity v_i (t), to the current position, i.e.:

$$x_i\ (t+1)=x_i(t)+v_i(t+1) \quad (1)$$

$_i$: index of particles
t: time index
x_i: position of the particle
v_i: velocity of the particle

To calculate the velocity of the particle the following equation is used:

$$v(t+1) = v_{ij}\ (t) + c_1 r_{1j}\ (t)[y_{ij}(t)-x_{ij}(t)] + c_2\ r_{2j}\ (t)[\hat{y}_j\ (t)-x_{ij}\ (t)] \quad (2)$$

Where we have the following parameters forming this Equation:

i: index of particles
t: time index
v_i: velocity of the particle
x_i: position of the particle
y_{ij}: best position found by particle i (personal best)
$\hat{y}_j$: best position found by the swarm (best overall)
c_1,c_2: constant acceleration
r_1,r_2: random numbers in the interval [0,1] applied to the particle.

Table 1 shows the classical PSO algorithm and Figure 1 illustrates the particle movement based on the algorithm.

Table 1. Classical PSO Algorithm

```
Create and Initialize the swarm
t= 0
Swarm ← Initializing particle swarm
While not at the stop condition do
t=t+1
For i = 1 to size (Swarm) to
Evaluate each particle xi of the Swarm
If fitness xi is better than fitness_mejorposi then
mejorposi ← xi
fitness _mejorposi ← fitness_ xi
end If
If fitness _mejorposi is better than fitness _mejorposi
Then mejorpos ← mejorposi
fitness _mejorpos ← fitness _mejorposi
end If
For i = 1 to size (Swarm) to
Calculate the speed of vi xi based on the values (2)
Calculate the new position of xi, its current value and vi (1)
F in for
end while
Output: Return the best solution found
```

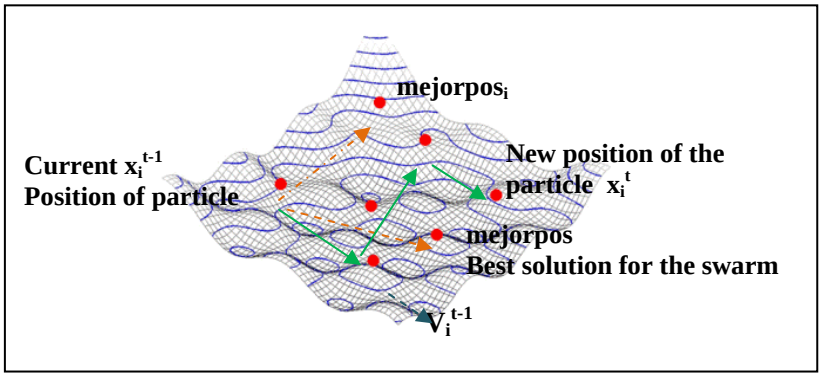

Figure 1. Particle movement taking into account the algorithm of Table 1.

3. Variants of PSO

In this section several variants of PSO are presented [7].

3.1 Inertia factor

PSO calculates the new position of each particle as a function of the flight, updating its position after having calculated the new position. This inertia factor was introduced to reduce the influence that the search direction brings to the particle. The equation for calculating the new speed of the particle is as follows, but now accompanied by **w** that represents the inertia:

$$v_{ij}(t+1) = \mathbf{w} \cdot v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t)-x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t)-x_{ij}(t)] \quad (3)$$

Now the new speed (v_{ij} (t+1)) is determined by taking into account the following parameters:

- "w". The inertia weight or inertia factor is a value, which regulates the influence of the previous velocity of the particle v (t) in calculating the new speed v (t + 1), by way of regulating the flight of the particle making a balance between exploitation and exploration of the search space. The inertia factor under certain conditions promotes the convergence of the swarm, i.e. all the particles approach the leader of the Swarm.
- Now v (t), Represents the current speed of the particle that was previously calculated by the same equation,and refers to the direction of flight is the particle having.

The Acceleration coefficients c_1*r_{1i} () and c_2*r_{2i} () and r represents a random real number uniformly distributed between 0.0 and 1.0. The coefficient c_1 regulates the influence of cognitive knowledge of the particle and the coefficient c_2 regulates the influence of knowledge social. c_1 regulates the influence of the best position reached by the particle ($\mathbf{y}_{ij}$) to guide your new address and c_2 regulates the influence of the cluster leader ($_j$) in the search direction of the particle.

$\mathbf{y}_{ij}$. Represents the memory of the particle, its best position achieved so for in that generation.
$\hat{\mathbf{y}}_j$. Represents the best position of the swarm, i.e. the leader.
x. Represents the current position of the particle is taken as reference for calculating the new speed.

v (t+1) is the speed of the current particle will determine the new search direction of the particle in generation t + 1. After calculating the current speed v (t +1) updates the particle position using equation (1).

3.2 Constriction coefficient

Proposed by Clerc, where this variant affects the velocity and the update is calculated with the equation modified as follows:

$$v_{ij}(t+1)=X[v_{ij}(t)+\phi_1(y_{ij}(t)-x_{ij}(t))+\phi_2(\hat{y}_j(t)-x_{ij}(t))] \quad (4)$$

Where

$$x = \frac{2k}{|2-\phi\sqrt{\phi(\phi-4)}|} \quad (5)$$

with

$\phi = \phi_1+\phi_2$, $\phi = c_1r_2$, $\phi = c_2r_2$

Using constraints where $\phi \geq 4$ y $k \in [0, 1]$, k controls the exploration and exploitation capabilities of the swarm.

4. Statement of the Problem

The problem is to optimize the membership functions of the fuzzy controllers for the Benchmark cases The first case is the water tank, where the 2 variants are used to find out which one works best to find the results. Below in Figure 2 the proposed architecture is presented:

Figure 2. Architecture of PSO optimization for fuzzy control

In the case of the water tank the fuzzy system has two input variables, each with three membership functions, and an output variable that has five membership functions as shown in Figure 3.

Figure 3. Mamdani Type Fuzzy System for first case

The Input1 variable Level (level) membership functions are: Gaussian, Gaussian, triangular is as shown in Figure 4.

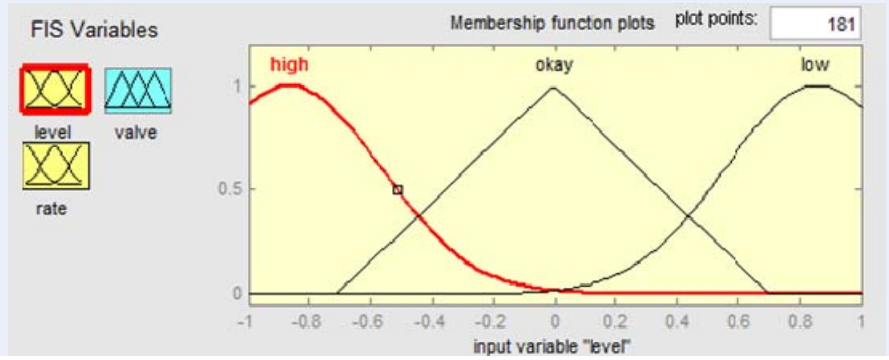

Figure 4 Variable Input

The Input Variable 2 Speed (Rate): Gaussian, triangular and Gaussian is as shown in Figure 5.

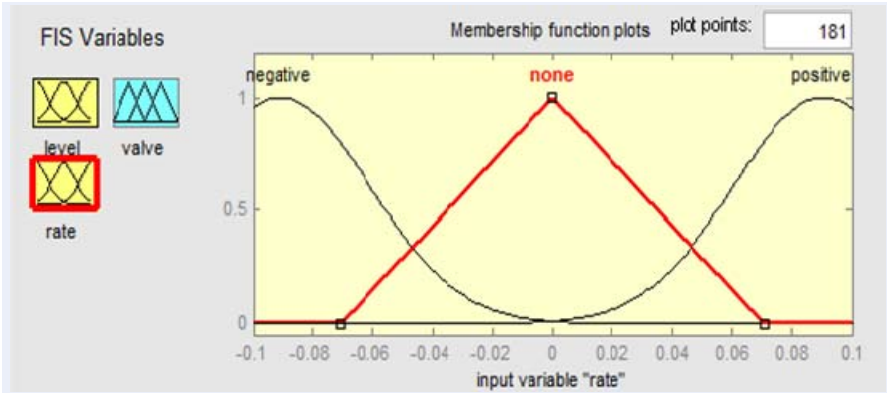

Figure 5. Variable Input 2

The Output Variable Valve (Valve): Triangular membership functions as Shown in Figure 6.

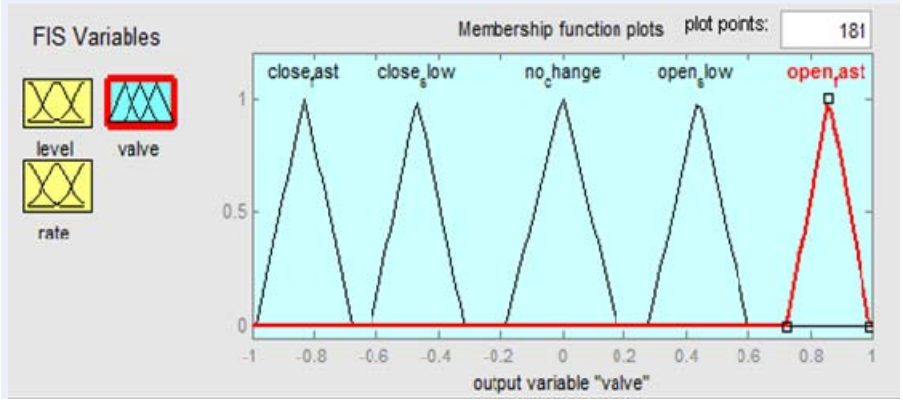

Figure 6. Variable Output

The Mamdani fuzzy system has 5 rules which are:
1. If (the water level is fine), then (valve not change)
2. If (the water level is low), then (valve open fast)
3. If (the water level is high), then (valve close fast)
4. If (the water level is good) and (rate is negative), then (valve closes slow)
5. If (the water level is good) and (rate in positive), then (valve open slow).

We are using three types of membership functions: Gaussian, triangular and generalized bell.

To evaluate the performance of the algorithm of particle swarm optimization (PSO) the equation of the mean absolute error was used, with the aim of finding the smallest error possible.

$$E_p = \sum_{i=1}^{n} \frac{| v_r^i - v_o^i |}{n} \quad (6)$$

We considered the representation of the particle as shown in Fig. 7 where there is a vector with 33 positions, the first 18 positions representing the membership functions of the two input variables, and the remaining 15 positions are the membership functions of the output variable.

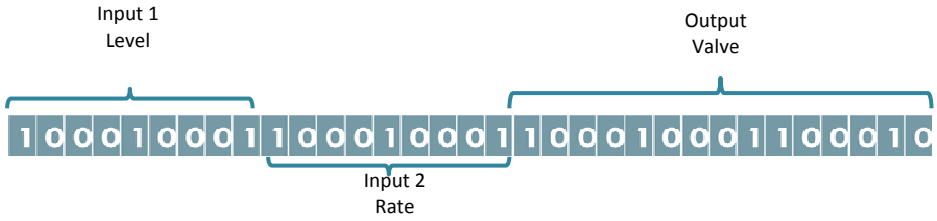

Figure7. Vector representation in PSO

It is known in advance by reviewing the literature and previous works that to develop fuzzy controllers and obtain good results optimizing membership functions is not easy, as it takes time to obtain a good design [8]-[13], [16]-[20]. That's why this work intends to make the optimization of membership functions with variants of the PSO and find out how efficient and fast they can be to find good results.

5. Inverted Pendulum

As in the previous case optimizing the membership functions of the fuzzy system the inverted pendulum is a similar problem. In this case, the control system has two inputs, each with five membership functions, and an output variable with five membership functions as shown in Figure 8.

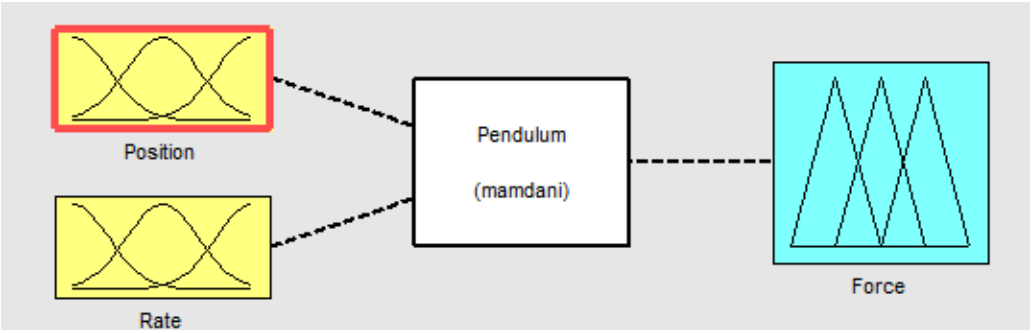

Figure 8. Mamdani Fuzzy system for the inverted pendulum

The variable Input1 Position: whit triangular membership functions, as shown in Figure 9.

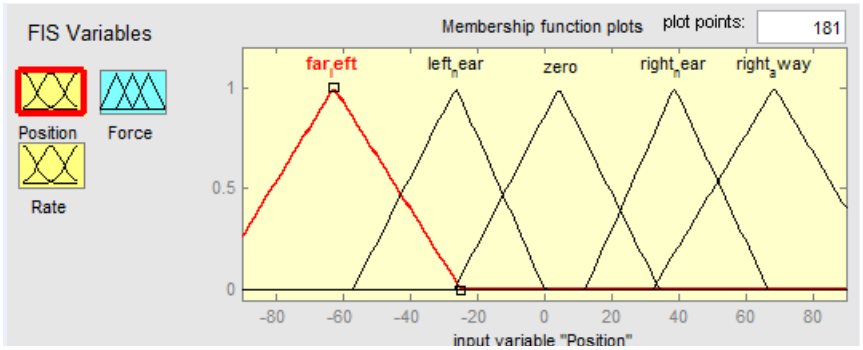

Figure 9. Membership functions of input variable (position)

The variable rate input 2: triangular as shown in Figure 10.

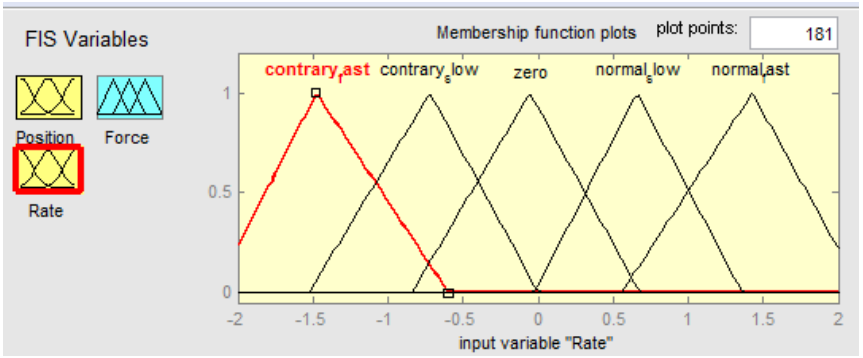

Figure 10. Membership functions of input variable 2 (rate)

The output variable force: triangular as shown in Figure 11.

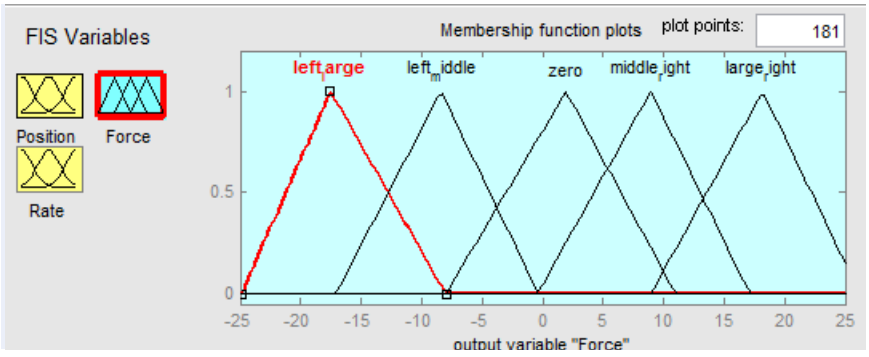

Figure 11. Membership functions of output variable (force)

The Mamdani fuzzy system has 25 rules are:
1. If (position is far right) and (rate is fast normal) then (right big Force)
2. If (position is about right) and (rate is fast normal) then (Force is great left)
3. If (position is zero) and (rate is fast normal) then (Force is great right)
4. If (position is near left) and (rate is fast normal) then (Force is half right)

5. If (position is far to the left) and (rate is fast normal) then (Force is great left)
.
.
.
25. If (position is far left) and (rate is fast contrary) then (Force is great left)
We are using two types of membership functions: triangular and trapezoidal.

To evaluate the performance of the algorithm of particle swarm optimization (PSO) the mean absolute error equation was used, with the aim of finding the smallest error as in the previous case shown in Equation 6.

We considered the representation of the particle as shown in Figure 12, where there is a vector with 45 positions, the first 30 positions represent the membership functions of the two input variables, and the remaining 15 positions are the membership functions of the output variable.

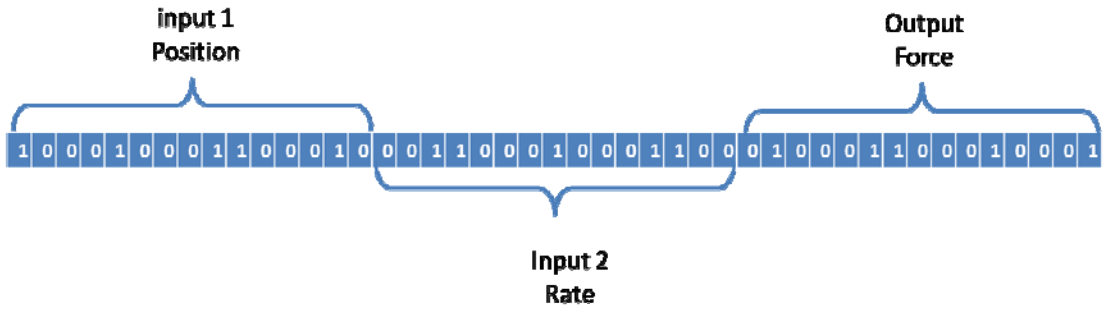

Figure 12. PSO vector representation for the second case (inverted pendulum).

It is known in advance by reviewing the literature and previous works that to develop fuzzy controllers and obtain good results optimizing membership functions is not easy, as it takes time to obtain a good design [8]-[13]. That's why this work intends to make the optimization of membership functions with variants of the PSO and find out how efficient and fast they can be to find good results.

6. Simulation Results for the water tank and the Inverted Pendulum

In this section we show results obtained from the two cases mentioned above.

6.1 Water Tank

Results are presented below in first the case (water tank). Table 2 shows results with the two variants of PSO and can be observed that only in Experiment 5 yielded a good error compared 0.011315 tank water that comes standard in Simulink without optimization that has an error of 0.0107.

Table 2 Experiments with two Variants of PSO Case Tank Water.

Experiment	*Iteration*	*Size Swarm*	*inertia*	*constriction*	*error*
1	10	20	1.0	0.2	0.42465
2	10	20	1.0	0.2	0.026217
3	10	20	1.0	0.2	0.011472
4	10	20	1.0	0.2	2.521
5	10	20	1.0	0.2	0.011315

Figure 13 shows the behavior of the PSO that as shown in iteration 4 found the best results for that run.

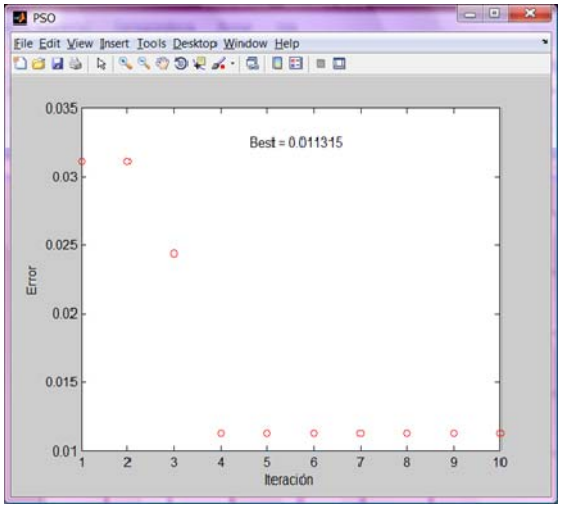

Figure 13. The perfomance of the PSO to Table 2.

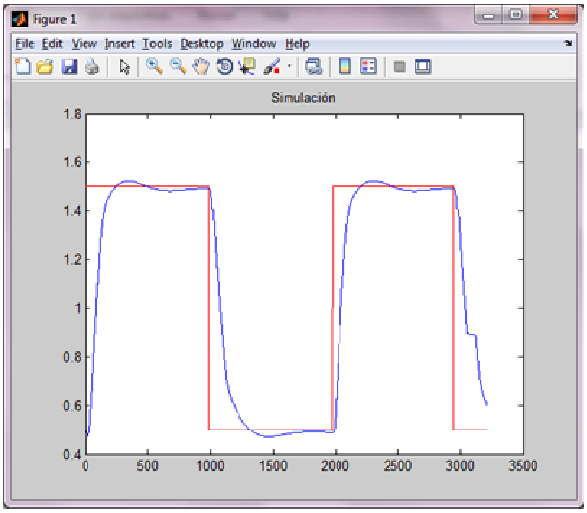

Figure 14 Benchmark simulation of water tank according to Table 2.

Now we show in Fig. 14, the resulting simulation using the two variants of PSO, and we can note that it follows very closely the reference. Below we show in Table 3 the updated results of the optimized water tank with PSO using only the two variants.

Table 3. New water tank experiments with only two variants of PSO.

Experiment	Iteration	Size Swarm	Inertia	Constriction	Error
1	20	50	1	0.2	0.010579
2	20	50	1	0.2	0.028229
3	20	50	1	0.2	0.022648
4	20	50	1	0.2	0.052259
5	20	50	1	0.2	0.063466
6	20	50	1	0.2	0.093932
7	20	50	1	0.2	0.0096114
8	20	50	1	0.2	0.015979
9	20	50	1	0.2	0.0020019
10	20	50	1	0.2	0.00062319
11	20	50	1	0.2	0.013211
12	20	50	1	0.2	0.0095633
13	20	50	1	0.2	0.010616
14	20	50	1	0.2	0.012901
15	**20**	**50**	**1**	**0.2**	**0.00038358**
16	20	50	1	0.2	0.008376
17	20	50	1	0.2	0.0366279
18	20	50	1	0.2	0.013028
19	20	50	1	0.2	0.0056146
20	20	50	1	0.2	0.0221167

In the experiments shown in Table 3 we have that the best error is shown in row number 15 with 0.00038353, and in this experiment we use the inertia factor of 1 and constriction of 0.2. These experiments are illustrated in the following Figures 15, 16 and 17.

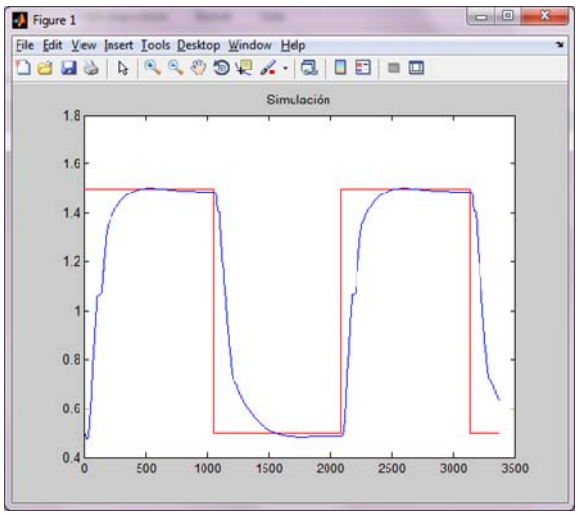

Figure 15. Benchmark simulation of water tank according to Table 3.

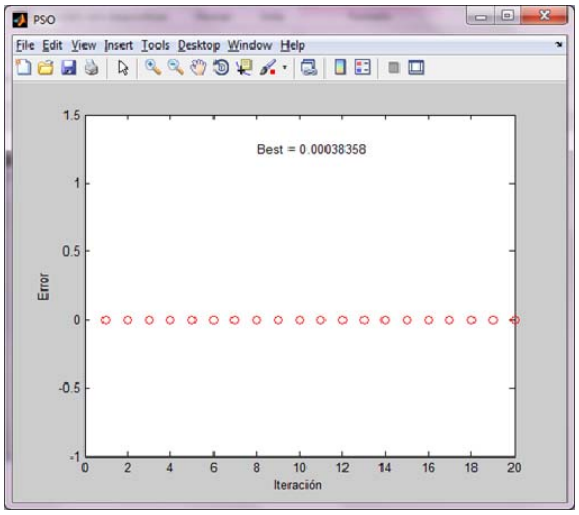

Figure 16. This figure shows the performance of PSO to Table 3.

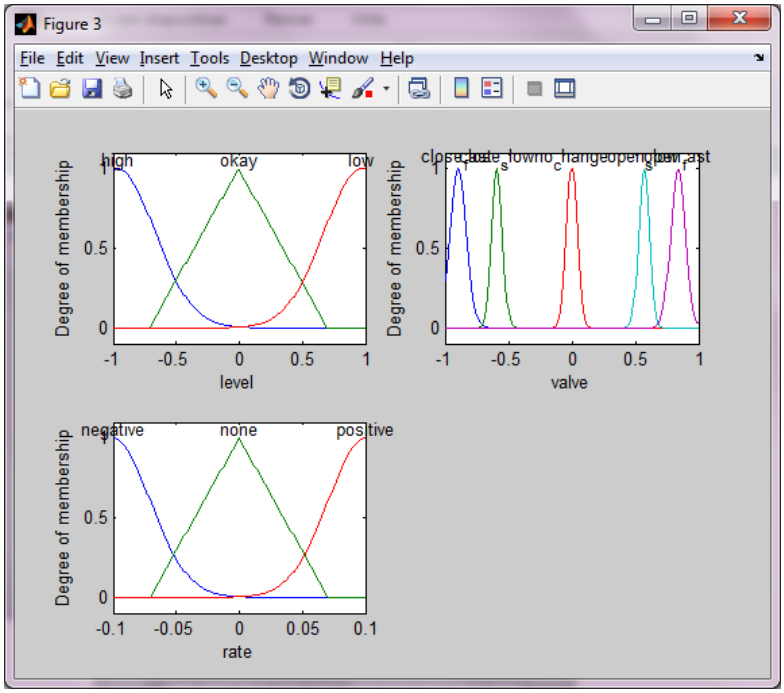

Figure 17. Membership functions optimized with PSO according to Table 3.

Table 4 shows 15 new experiments using inertia with a value of 2 and constriction of 0.8 with the best result in experiment 3 with an error of 0.009892. Below we show for experiment 3 the results in Figure 18-20. We can see in Figure 13 that control stopped short of reaching the water tank reference, in Figure 14 shows the behavior of PSO where we observe that since the first iteration it was found small error in Figure 15 shows the memberships functions and optimized with PSO variants.

Table 5 shows a comparison with the two other algorithms, which are the Fuzzy Lyapunov Synthesis and the GA, of which we can be see that we have performed well compared to these algorithms reponted in [14][15].

Table 4. New water tank experiments with only two variants of PSO.

Experiment	Iteration	Size Swarm	Inertia	Constriction	Error
1	20	100	2	0.8	0.00654
2	20	100	2	0.8	0.019979
3	**20**	**100**	**2**	**0.8**	**0.009892**
4	20	100	2	0.8	0.133
5	20	100	2	0.8	0.078268
6	20	100	2	0.8	0.70393
7	20	100	2	0.8	0.098223
8	20	100	2	0.8	0.0043964
9	20	100	2	0.8	0.024171
10	20	100	2	0.8	0.0087435
11	20	100	2	0.8	0.029568
12	20	100	2	0.8	0.061995
13	20	100	2	0.8	0.050065
14	20	100	2	0.8	0.03556
15	20	100	2	0.8	0.021311

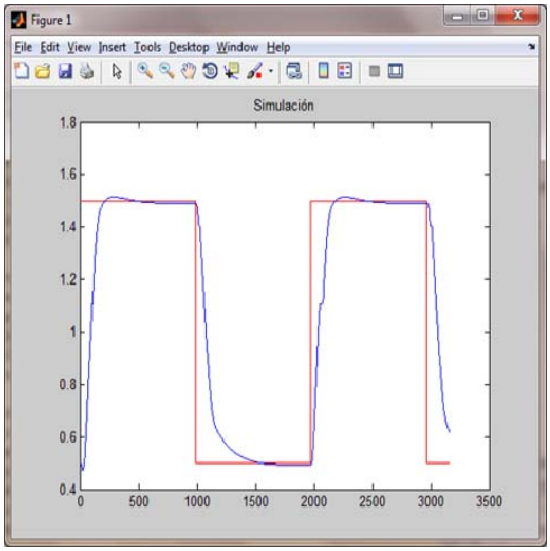

Figure 18. Benchmark simulation of water tank according to the Table 4.

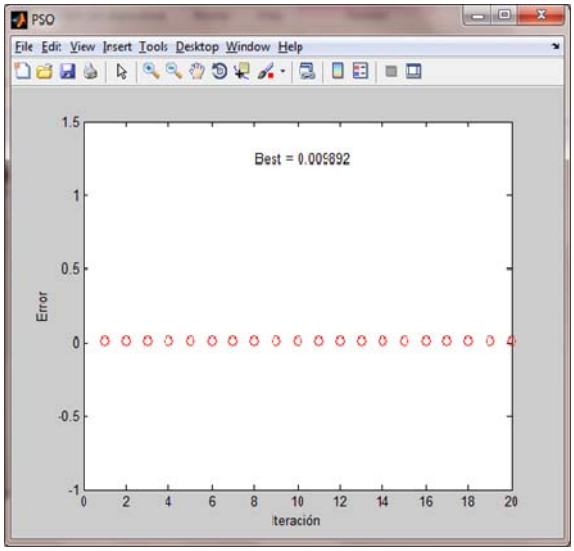

Figure 19. PSO Behavior according to Table 4.

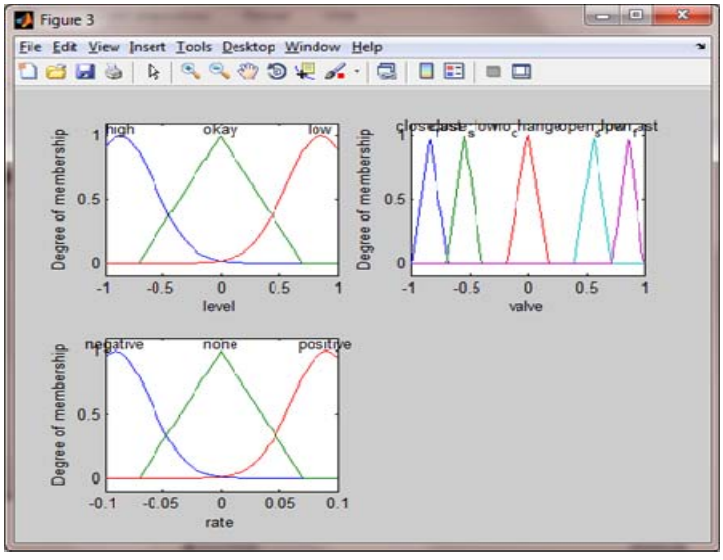

Figure 20 Membership functions optimized with PSO according to Table 4.

Table 5 Comparison table with other optimization algorithms

Comparisons of different optimization algorithms			
Optimization PSO variants MF	MF Synthesis Optimization with Fuzzy Lyapunov	GA Optimization MF	
error	error	error Michigan	error Pittsburgh
0.00038358	0.0358	0.102	0.201
0.00062319	0.084	0.103	0.204
0.0020019	0.0907	0.111	0.215
0.0095633	------	------	------

6.2 Inverted Pendulum

Below are the results in Table 6 of the Inverted Pendulum optimization

Table 6 Results of the Optimized Inverted Pendulum

Experiment	Error
1	0.00521
2	0.0269
3	0.0452
4	0.0490
5	0.0651

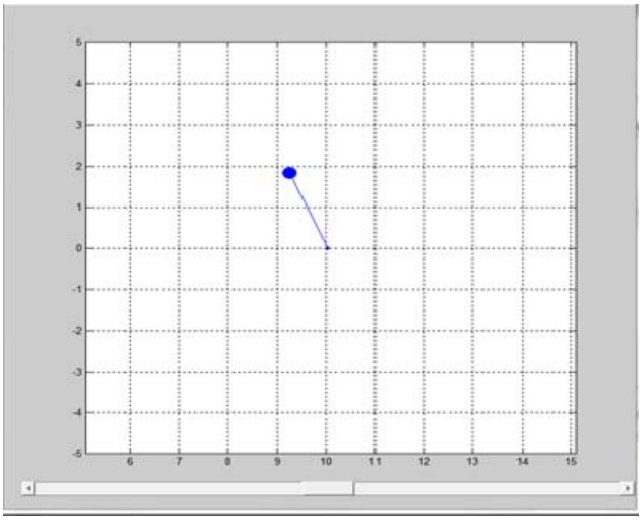

Figure 21 Pendulum simulation not optimized.

Table 7 shows results with the two variants of PSO, and it can be noted that in Experiment 19 we obtain an error of 0.0038974, which compared to the non-optimized pendulum experiment number 1 we can see that we obtain a good result.

Table 7. Results the Inverted Pendulum Optimized

Experiment	Iterations	Size Swarm	Inertia	Constriction	Error
1	10	20	1	0.2	0.020028
2	10	20	1	0.2	0.01198
3	10	20	1	0.2	0.71938
4	10	20	1	0.2	0.21001
5	10	20	1	0.2	0.35914
6	10	20	1	0.2	0.13698
7	10	20	1	0.2	0.063951
8	10	20	1	0.2	0.15335
9	10	20	1	0.2	0.042591
10	10	20	1	0.2	0.01699
11	10	20	1	0.2	0.017881
12	10	20	1	0.2	0.14486
13	10	20	1	0.2	0.025516
14	10	20	1	0.2	0.046648
15	10	20	1	0.2	0.048371
16	10	20	1	0.2	0.17906
17	10	20	1	0.2	0.03284
18	10	20	1	0.2	0.26494
19	**10**	**20**	**1**	**0.2**	**0.0038974**
20	10	20	1	0.2	0.21848

Below is the best result figure in the above table which is experiment No. 19 with an error of 0.0038974. In figure 22 we observe the simulation of the Inverted Pendulum optimization and in Figure 23 the behavior of the PSO.

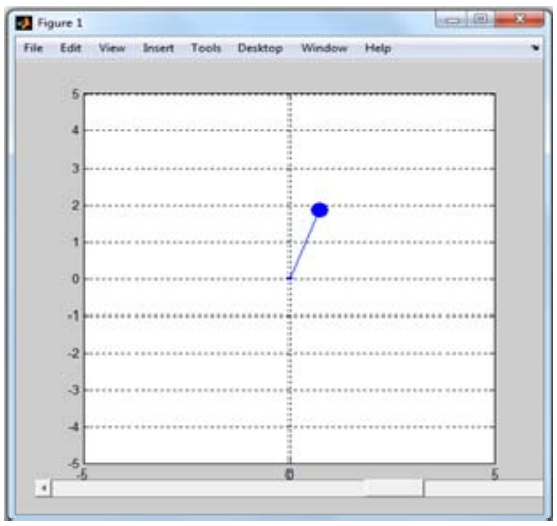

Figure 22. Inverted pendulum simulation according to Table 7.

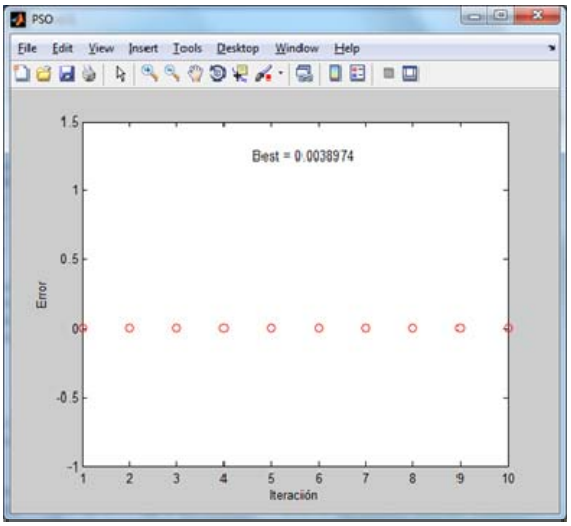

Figure 23. PSO Behavior according to Table 7.

Table 8 shows the results with the two variants of PSO, which can be noted that in Experiment an error of 6 9.326e-005 is obtained being the best result to the moment in this work.

Table 8 New Results for the Inverted Pendulum Optimization

Experiment	Iterations	Size Swarm	Inertia	Constriction	Error
1	15	50	2	0.8	0.080929
2	15	50	2	0.8	0.03305
3	15	50	2	0.8	0.032003
4	15	50	2	0.8	0.082737
5	15	50	2	0.8	0.088668
6	15	50	2	0.8	9.326e-005
7	15	50	2	0.8	0.0068039
8	15	50	2	0.8	0.0055893
9	15	50	2	0.8	0.0031543
10	15	50	2	0.8	0.021335
11	15	50	2	0.8	0.003152
12	15	50	2	0.8	0.019987
13	15	50	2	0.8	0.014184
14	15	50	2	0.8	0.087817
15	15	50	2	0.8	0.0075345

In Figure 24 shows the simulation of the optimization of the Inverted vertical pendulum and Figure 25 the PSO behavior with the good error mentioned above.

7. Results with only one variant (Inertia) of PSO for the Water Tank

The experiments carried out are to the water plant with a value of inertia of 0.7 and 0.9 to take these recommended values in the literature. Table 9 shows the results of experiments using only a value of 0.7 inerciacon and Table 10 shows the results of 10 experiments using inertia with a value of 0.9. In paragraph 7.1 in Table 11 and 12 show results using only 1 value constriction in Table 11 and worth 2 in Table 12.

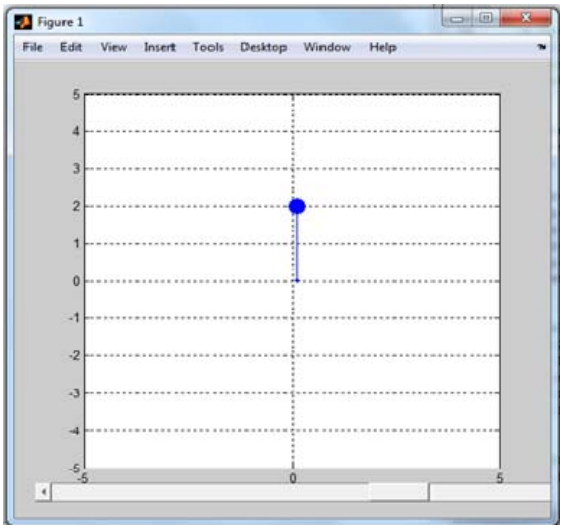

Figure 24 Pendulum simulation according to Table 8.

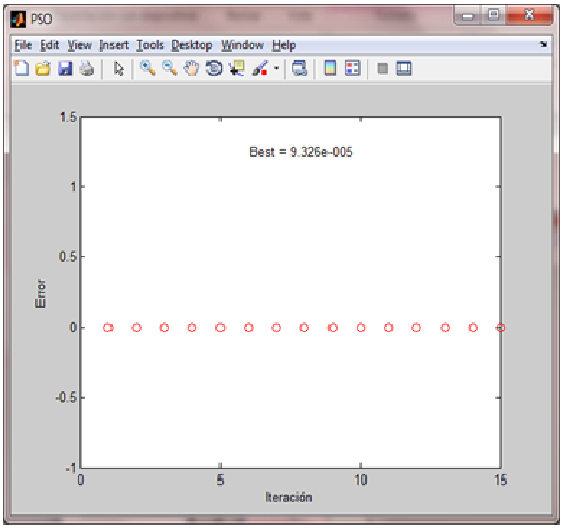

Figure 25. PSO Behavior according to Table 8.

Table 9 Results with Inertia 0.7 for Water Tank

Experiment	Iterations	Size Swarm	Inertia	Error
1	10	50	0.7	0.0159
2	10	50	0.7	0.0610
3	10	50	0.7	0.0649
4	10	50	0.7	0.0772
5	10	50	0.7	0.0959
6	10	50	0.7	0.0503
7	10	50	0.7	0.0138
8	10	50	0.7	0.0373
9	10	50	0.7	0.0799
10	10	50	0.7	0.0072

Table 10 Results with inertia 0.9 for Water Tank

Experiment	Iterations	Size Swarm	Inertia	Error
1	15	100	0.9	0.0885
2	15	100	0.9	0.0995
3	15	100	0.9	0.0617
4	15	100	0.9	0.0839
5	15	100	0.9	0.0774
6	15	100	0.9	0.0409
7	15	100	0.9	0.9806
8	15	100	0.9	0.0884
9	15	100	0.9	0.0683
10	15	100	0.9	0.0797

Table 11 Results with Constriction 1 for Water Tank

Experiment	Iterations	Size Swarm	Constriction	Error
1	15	100	1	0.02541
2	15	100	1	0.145289
3	15	100	1	0.24891
4	15	100	1	0.012492
5	15	100	1	0.545810
6	15	100	1	0.239487
7	15	100	1	0.430021
8	15	100	1	0.002544
9	15	100	1	0.870025
10	15	100	1	0.100256

Table 12 Results with constriction value of 2 for the Water Tank problem

Experiment	Iterations	Size Swarm	Constriction	Error
1	15	100	2	0.125412
2	15	100	2	0.025841
3	15	100	2	0.00213
4	15	100	2	0.018900
5	15	100	2	0.054521
6	15	100	2	0.039724
7	15	100	2	0.056482
8	15	100	2	0.001628
9	15	100	2	0.016617
10	15	100	2	0.047989

8. Conclusions

In this paper the design of fuzzy control systems has been illustrated with the application of two different PSO variants, this with the idea of verifying if PSO can be an efficient method for this fuzzy system design problem. It is noted that the simulation results show the potential use of this of type bio-inspired optimization methods in fuzzy control problems. Future work includes considering other control plants (more complex) and trying other variants of PSO, to make a formal comparative study of different PSO variants in the area of Fuzzy Control. Other future work could be a comparison with other bio-inspired optimization methods, like genetic algorithms, ant colony optimization, and artificial bee optimization.

References

[1] J. Kennedy, R. Eberhart, 1995, Particle Swarm Optimization, from Proc. IEEE int´l. Conf. On Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV: 1942-1948.
[2] L. Zadeh, (1965), “Fuzzy Logic”, IEEE Computer, pp. 83-92.
[3] J. Kennedy, Y. Shi, 2001, Swarm Intelligence, Academic Press, Inc.
[4] J. Kennedy, Shi Y., “Swarm Intelligence”. San Francisco: Morgan Kaufmann Publishers, 2001.
[5] J. Kennedy, “The Particle Swarm: Social Adaptation of Knowledge”. IEEE International Conference on Evolutionary Computation, pp. 303-308, 1997.
[6] P. Andries, “Fundamentals of Computational Swarm Intelligence”, Wiley, J England, pp. 85-129, 2005.
[7] Y. Del Valle, V. Kumar G., S. Mohagheghi, J.C. Hernández, R. G. Harley: Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems. April 2008
[8] R. Martinez, O. Castillo, and J. Soria: Particle Swarm Optimization Applied to the Design of Type-1 and Type-2 Fuzzy Controllers for an Autonomous Mobile Robot. Bio-inspired Hybrid Intelligent Systems for Image Analysis and Pattern Recognition 2009: pp. 247-262
[9] S. J., Marmolejo, O. Castillo: Optimization of Membership Functions for Type-1 and Type-2 Fuzzy Controllers of an Autonomous Mobile Robot Using PSO. Ed. Springer 2013: pp. 97-104.
[10] F. Olivas, O. Castillo: Particle Swarm Optimization with Dynamic Parameter Adaptation Using Fuzzy Logic for Benchmark Mathematical Functions. Ed. Springer 2013: pp. 247-258.
[11] K. Zielinski, P. Weitkemper, R. Laur, K. Kammeyer, Optimization of Power Allocation for Interference Cancellation With Particle Swarm Optimization, IEEE transactions on evolutionary computation, vol. 13, no.1, February 2009.
[12] E. Lizárraga, O. Castillo, J. Soria, A Method to Solve the Traveling Salesman Problem Using Ant Colony Optimization Variants with Ant Set Partitioning, Recent Advances on Hybrid Intelligent Systems, Ed.Springer 2013, pp.237-246
[13] R. Fierro, O. Castillo, Design of Fuzzy Different PSO Variants, Recent Advances on Hybrid Intelligent Systems, Ed. Springer 2013, pp. 81-88.
[14] M. C. Ibarra, O. Castillo, J. Soria, Designing Systematic Stable Fuzzy Logic Controllers by Fuzzy Lyapunov Synthesis, Recent Advances on Hybrid Intelligent System, Ed. Springer 2013, pp. 63-79
[15] R. Aranda, Optimization of Membership Functions stable fuzzy controllers using genetic algorithms, Master thesis, Tijuana Institute of Technology 2012.
[16] Eberhart, R.C. & Shi, Y. Parameter Selection in Particle Swarm Optimization. Evolutionary Programming VII Springer, Lecture Notes in Computer Science, 1998. 1447, p.591-600.
[17] Kennedy, J. & Eberhart, R. C. Particle swarm optimization. en IEEE International Conference on Neural Networks. Piscataway, New York, USA, 1995.Vol. 4: p. 1942-1948.

[18] Li, X., Tian, M. & Kong, P. Novel particle swarm optimization for constrained optimization problems, Lecture Notes in Computer Science, , 2005. Vol. 3809, p. 1305–1310.
[19] Matias Javier Micheletto, Optimizador de Funciones Multivariadas por Enjsmbre de Particulas, Dpto. Ingeniería Eléctrica y Computadoras Universidad Nacional del Sur, Junio 18 2013, http://www.41jaiio.org.ar/sites/default/files/39_EST_2012.pdf
[20] Yenny Noa Vargas., Estrategias para mejorar el Balance entreExploración y Explotación en Optimización de Enjambre de Partículas, Universidad de La Habana, Diciembre 2011, Disponible en: http://www.yorku.ca/sychen/research/theses/2011_Yenny_MSc.pdf