Fuzzy Classification System Design Using PSO with Dynamic Parameter Adaptation Through Fuzzy Logic

Frumen Olivas, Fevrier Valdez and Oscar Castillo

Abstract In this paper a new method for dynamic parameter adaptation in particle swarm optimization (PSO) is proposed. PSO is a metaheuristic inspired in social behaviors, which is very useful in optimization problems. In this paper we propose an improvement to the convergence and diversity of the swarm in PSO using fuzzy logic. Simulation results show that the proposed approach improves the performance of PSO.

Keywords Fuzzy logic · Particle swarm optimization · Dynamic parameter adaptation · Fuzzy classifier · Fuzzy classification system

1 Introduction

Fuzzy logic or multi-valued logic is based on fuzzy set theory proposed by Zadeh [14], which helps us in modeling knowledge, through the use of if-then fuzzy rules.

The fuzzy set theory provides a systematic calculus to deal with linguistic information, and that improves the numerical computation by using linguistic labels stipulated by membership functions [7].

Particle swarm optimization (PSO) that was introduced by Kennedy and Eberhart in 1995 [9, 10], maintains a swarm of particles and each particle represents a possible solution. These particles "fly" through a multidimensional search space, where the position of each particle is adjusted according to your own experience and that of its neighbors [4].

PSO has recently received many improvements and applications. Most of the modifications to PSO are to improve convergence and to increase the diversity of the swarm [4]. So in this paper we propose an improvement to the convergence and diversity of PSO through the use of fuzzy logic. Basically, fuzzy rules are used to

F. Olivas · F. Valdez · O. Castillo (✉)
Tijuana Institute of Technology, Tijuana, Mexico
e-mail: ocastillo@tectijuana.mx

O. Castillo and P. Melin (eds.), *Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics*, Studies in Computational Intelligence 574,
DOI 10.1007/978-3-319-10960-2_2

control the key parameters in PSO to achieve the best possible dynamic adaptation of these parameter values.

The rest of the paper is organized as follows. Section 2 describes the proposed methodology. Section 3 shows how the experiments were performed with the proposed method and the simple method using the benchmark functions defined in Sect. 2. Section 4 shows how to perform the statistical comparison with all its parameters and analysis of results. Section 5 shows the design of fuzzy classifier. Section 6 shows the methodology to follow for the design of fuzzy classifier. Section 7 shows how the experiments were performed with the proposed method and the simple method in the design of fuzzy classifier. Section 8 shows how to perform the statistical comparison with all its parameters and analysis of results. Section 9 shows the conclusions of the design of fuzzy classifier design. Finally, the conclusions of this paper are presented.

2 Methodology for Parameter Adaptation

The dynamics of PSO is defined by Eqs. 1 and 2, which are the equations to update the position and velocity of the particle, respectively.

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{1}$$

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_1(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\widehat{y}_j(t) - x_{ij}(t)\right] \tag{2}$$

Parameters c1 and c2 were selected to be adjusted using fuzzy logic, since those parameters account for the movement of the particles.

The parameter c1 or cognitive factor represents the level of importance given the particle to its previous positions.

The parameter c2 or social factor represents the level of importance that the particle gives the best overall position.

Based on the literature [4] the recommended values for c1 and c2 must be in the range of 0.5 and 2.5, plus it is also suggested that changing the parameters c1 and c2 dynamically during the execution of this algorithm can produce better results.

In addition it is also found that the algorithm performance measures, such as: diversity of the swarm, the average error at one point in the execution of the algorithm, the iterations themselves, needs to be considered to run the algorithm, among others. In our work all the above are taken in consideration for the fuzzy systems to modify the parameters c1 and c2 dynamically changing these parameters in each iteration of the algorithm.

For measuring the iterations of the algorithm, it was decided to use a percentage of iterations, i.e. when starting the algorithm the iterations will be considered “low”, and when the iterations are completed it will be considered “high” or close to 100 %. To represent this idea we use Eq. 3.

$$Iteration = \frac{Current\ Iteration}{Maximum\ of\ Iterations} \tag{3}$$

The diversity measure is defined by Eq. 4, which measures the degree of dispersion of the particles, i.e. when the particles are closer together there is less diversity as well as when particles are separated then diversity is high. As the reader will realize the equation of diversity can be considered as the average of the Euclidean distances between each particle and the best particle.

$$Diversity(S(t)) = \frac{1}{n_s}\sum_{i=1}^{n_s}\sqrt{\sum_{j=1}^{n_x}\left(x_{ij}(t) - \bar{x}_j(t)\right)^2} \tag{4}$$

The error measure is defined by Eq. 5, which measures the difference between the swarm and the best particle, by averaging the difference between the fitness of each particle and the fitness of the best particle.

$$Error = \frac{1}{n_s}\sum_{i=1}^{n_s}(Fitness(x_i) - MinF) \tag{5}$$

Therefore for designing the fuzzy systems, which dynamically adjust the parameters of c1 and c2, the three measures described above were considered as inputs. It is obvious that for each fuzzy system the outputs are c1 and c2.

In regards to the inputs of the fuzzy systems, the iteration variable has by itself a defined range of possible values which range from 0 to 1 (0 is 0 % and 1 is the 100 %), but with the diversity and the error, we perform a normalization of the values of these to have values between 0 and 1. Equation 6 shows how the normalization of diversity is performed and Eq. 7 shows how the normalization of the error is obtained.

$$\begin{aligned} Diver\,Norm &= \begin{cases} if\ Min\,Diver = Max\,Diver\{Diver\,Norm = 0 \\ if\ Min\,Diver \neq Max\,Diver\{Diver\,Norm = Fn\,Norm \end{cases} \\ Fn\,Norm &= \frac{Diversity - Min\,Diver}{Max\,Diver - Min\,Diver} \end{aligned} \tag{6}$$

Equation 6 shows two conditions for the normalization of diversity, the first provides that where the maximum Euclidean distance is equal to the minimum Euclidean distance, this means that the particles are exactly in the same position so there is no diversity. The second condition deals with the cases with different Euclidean distances.

$$Error\,Norm = \begin{cases} if\ MinF = MaxF\{Error\,Norm = 1 \\ if\ MinF \neq MaxF\{Error\,Norm = \frac{Error - MinF}{MaxF - MinF} \end{cases} \tag{7}$$

Equation 7 shows two conditions to normalize the error, the first one tells us that when the minimum fitness is equal to the maximum fitness, then the error will be 1; this is because the particles are close together. The second condition deals with the cases with different fitness.

The design of the input variables can be appreciated in Figs. 1, 2 and 3, which show the inputs iteration, diversity, and error respectively, each input is granulated into three triangular membership functions.

For the output variables, as mentioned above, the recommended values for c1 and c2 are between 0.5 and 2.5, so that the output variables were designed using this range of values. Each output is granulated in five triangular membership functions, the design of the output variables can be seen in Figs. 4 and 5, c1 and c2 respectively.

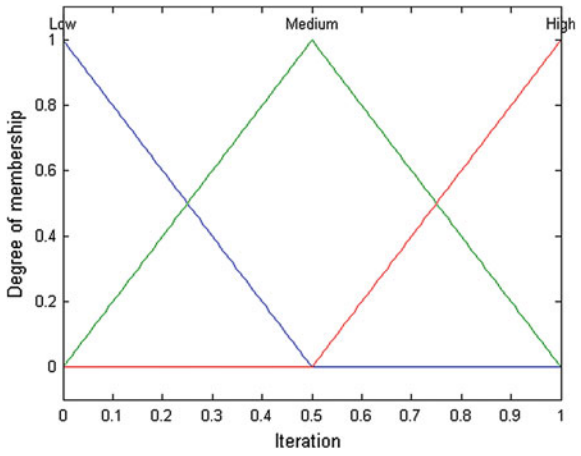

Fig. 1 Input 1: iteration

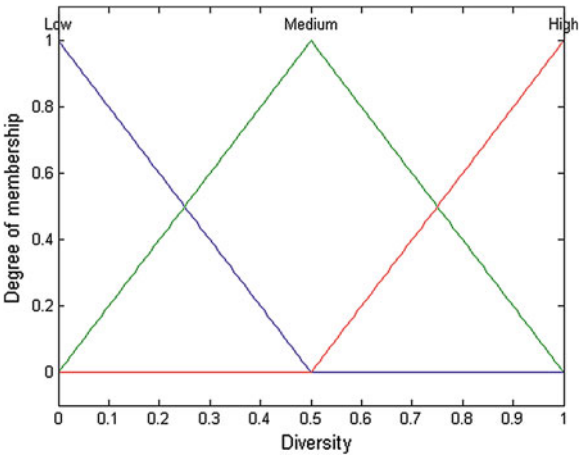

Fig. 2 Input 2: diversity

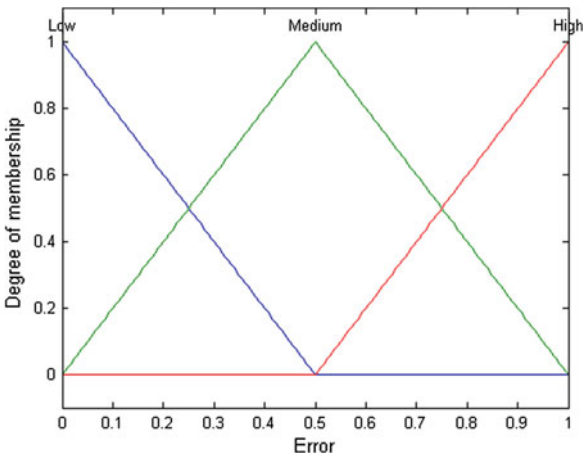

Fig. 3 Input 3: error

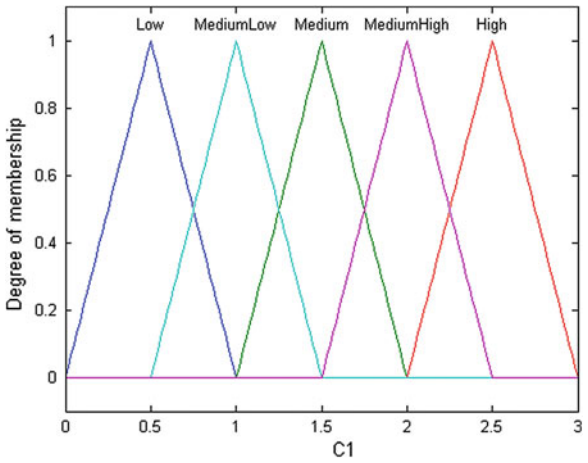

Fig. 4 Output 1: c1

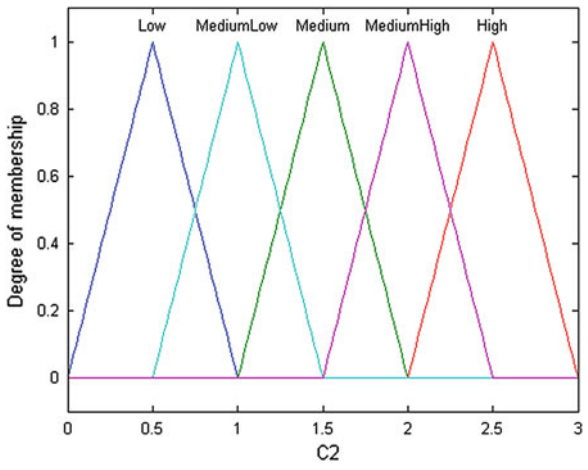

Fig. 5 Output 2: c2

Having defined the possible input variables, it was decided to combine them to generate different fuzzy systems for dynamic adjustment of c1 and c2. Based on the combinations of possible inputs, there were seven possible fuzzy systems, but it was decided to consider only the systems that have more inputs (since we previously considered fuzzy systems with only a single input), so that eventually there were three fuzzy systems which are defined below.

The first fuzzy system has iteration and diversity as inputs, which is shown in Fig. 6. The second fuzzy system has iteration and error as inputs and is shown in Fig. 7. The third fuzzy system has iteration, diversity, and error as inputs, as shown in Fig. 8.

To design the rules of each fuzzy system, it was decided that in early iterations the PSO algorithm must explore and eventually exploit. Taking into account other variables such as diversity, for example, when diversity is low, that is, that the

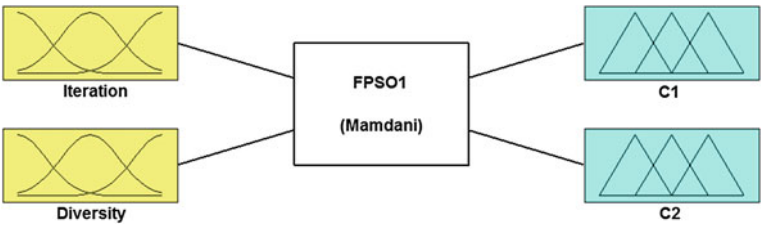

Fig. 6 First fuzzy system

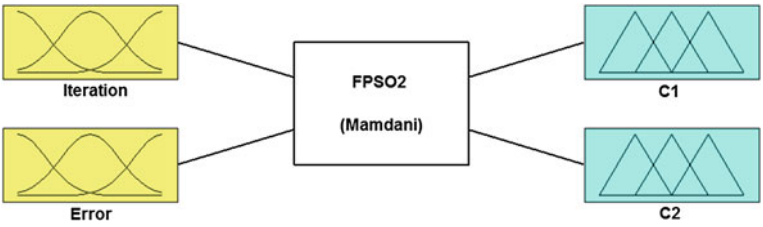

Fig. 7 Second fuzzy system

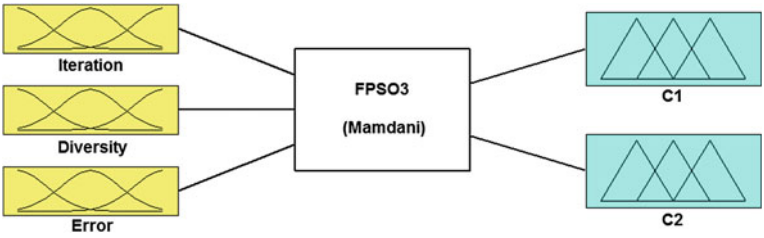

Fig. 8 Third fuzzy system

1. If (Iteration is Low) and (Diversity is Low) then (C1 is High)(C2 is Low)
2. If (Iteration is Low) and (Diversity is Medium) then (C1 is MediumHigh)(C2 is Medium)
3. If (Iteration is Low) and (Diversity is High) then (C1 is MediumHigh)(C2 is MediumLow)
4. If (Iteration is Medium) and (Diversity is Low) then (C1 is MediumHigh)(C2 is MediumLow)
5. If (Iteration is Medium) and (Diversity is Medium) then (C1 is Medium)(C2 is Medium)
6. If (Iteration is Medium) and (Diversity is High) then (C1 is MediumLow)(C2 is MediumHigh)
7. If (Iteration is High) and (Diversity is Low) then (C1 is Medium)(C2 is High)
8. If (Iteration is High) and (Diversity is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
9. If (Iteration is High) and (Diversity is High) then (C1 is Low)(C2 is High)

Fig. 9 Rules for fuzzy system 1

1. If (Error is Low) and (Iteration is Low) then (C1 is Low)(C2 is Medium)
2. If (Error is Low) and (Iteration is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
3. If (Error is Low) and (Iteration is High) then (C1 is Low)(C2 is High)
4. If (Error is Medium) and (Iteration is Low) then (C1 is MediumLow)(C2 is MediumHigh)
5. If (Error is Medium) and (Iteration is Medium) then (C1 is Medium)(C2 is Medium)
6. If (Error is Medium) and (Iteration is High) then (C1 is Medium)(C2 is High)
7. If (Error is High) and (Iteration is Low) then (C1 is High)(C2 is MediumLow)
8. If (Error is High) and (Iteration is Medium) then (C1 is MediumHigh)(C2 is Medium)
9. If (Error is High) and (Iteration is High) then (C1 is High)(C2 is Low)

Fig. 10 Rules for fuzzy system 2

particles are close together, we must use exploration, and when diversity is high we must use exploitation.

The rules for each fuzzy system are shown in Figs. 9, 10 and 11, for the fuzzy systems 1, 2 and 3, respectively.

Also for the comparison of the proposed method with respect to the PSO without parameter adaptation, we considered benchmark mathematical functions, defined in [6, 11], which are 27 in total, and in each we must find the parameters that give us the global minimum of each function. In Fig. 12 there is a sample of the functions that are used.

As indicated in Fig. 12 we only considered functions of one or two dimensions for the experiments.

So that once defined the fuzzy systems that dynamically adjust the parameters of PSO, and defined the problem in which it applies (Benchmark mathematical functions), the proposal is as shown in Fig. 13, where we can notice that c1 and c2 parameters are adjusted by a fuzzy system, and in turn this "fuzzy PSO" searches for the optimal parameters for the Benchmark mathematical functions.

1. If (Iteration is Low) and (Diversity is Low) and (Error is Low) then (C1 is MediumLow)(C2 is Low)
2. If (Iteration is Low) and (Diversity is Low) and (Error is Medium) then (C1 is MediumHigh)(C2 is Low)
3. If (Iteration is Low) and (Diversity is Low) and (Error is High) then (C1 is High)(C2 is Low)
4. If (Iteration is Low) and (Diversity is Medium) and (Error is Low) then (C1 is Medium)(C2 is Medium)
5. If (Iteration is Low) and (Diversity is Medium) and (Error is Medium) then (C1 is MediumHigh)(C2 is MediumLow)
6. If (Iteration is Low) and (Diversity is Medium) and (Error is High) then (C1 is MediumHigh)(C2 is Low)
7. If (Iteration is Low) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is MediumLow)
8. If (Iteration is Low) and (Diversity is High) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
9. If (Iteration is Low) and (Diversity is High) and (Error is High) then (C1 is MediumLow)(C2 is Low)
10. If (Iteration is Medium) and (Diversity is Low) and (Error is Low) then (C1 is Medium)(C2 is Medium)
11. If (Iteration is Medium) and (Diversity is Low) and (Error is Medium) then (C1 is MediumHigh)(C2 is MediumLow)
12. If (Iteration is Medium) and (Diversity is Low) and (Error is High) then (C1 is MediumHigh)(C2 is Low)
13. If (Iteration is Medium) and (Diversity is Medium) and (Error is Low) then (C1 is MediumLow)(C2 is MediumHigh)
14. If (Iteration is Medium) and (Diversity is Medium) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
15. If (Iteration is Medium) and (Diversity is Medium) and (Error is High) then (C1 is MediumHigh)(C2 is MediumLow)
16. If (Iteration is Medium) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is MediumHigh)
17. If (Iteration is Medium) and (Diversity is High) and (Error is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
18. If (Iteration is Medium) and (Diversity is High) and (Error is High) then (C1 is Medium)(C2 is Medium)
19. If (Iteration is High) and (Diversity is Low) and (Error is Low) then (C1 is Low)(C2 is MediumLow)
20. If (Iteration is High) and (Diversity is Low) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
21. If (Iteration is High) and (Diversity is Low) and (Error is High) then (C1 is MediumLow)(C2 is Low)
22. If (Iteration is High) and (Diversity is Medium) and (Error is Low) then (C1 is Low)(C2 is MediumHigh)
23. If (Iteration is High) and (Diversity is Medium) and (Error is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
24. If (Iteration is High) and (Diversity is Medium) and (Error is High) then (C1 is Medium)(C2 is Medium)
25. If (Iteration is High) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is High)
26. If (Iteration is High) and (Diversity is High) and (Error is Medium) then (C1 is Low)(C2 is MediumHigh)
27. If (Iteration is High) and (Diversity is High) and (Error is High) then (C1 is Low)(C2 is MediumLow)

Fig. 11 Rules for fuzzy system 3

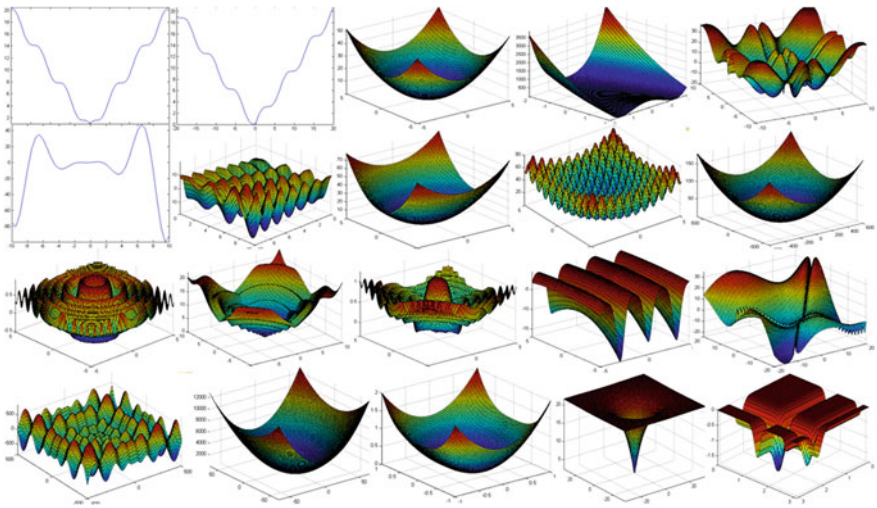

Fig. 12 Benchmark mathematical functions

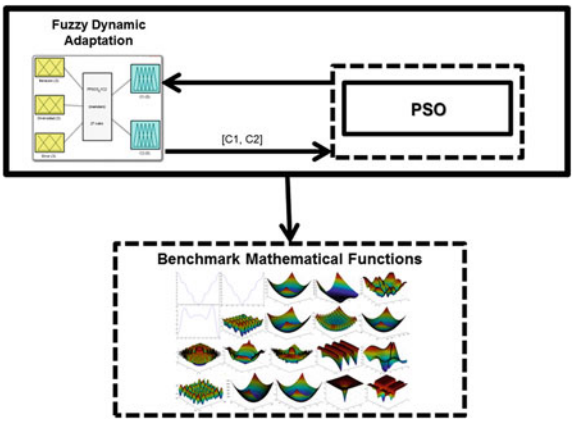

Fig. 13 Proposal for fuzzy dynamic adaptation of PSO

3 Experimentation with the Fuzzy Systems and the Benchmark Mathematical Functions

For the experiments we used the parameters contained in Table 1. Table 1 shows the parameters of the methods to be compared; in this case, we perform a comparison of the proposed method and its variations against the simple PSO algorithm.

Since functions do not have the same global minimum, for comparison it was decided to normalize the results of each function, for this it is used Eq. 8, which gives results between 0 and 1, which means that a number close to 0 is better than a number close to 1.

$$Experiment\ Norm = \left| \frac{Experiment - GlobalMin}{GlobalMax - GlobalMin} \right| \tag{8}$$

To normalize the results with Eq. 8, we need the maximum and the minimum of each benchmark mathematical function; in our case these data are known. Also, the absolute value is needed, because we want to know how much difference between the results of the experiment and the minimum value of the function. Therefore, Table 2 shows some experimental results of each method with each function.

Table 1 Parameters for each method

Parameter	Simple PSO	Fuzzy PSO 1	Fuzzy PSO 2	Fuzzy PSO 3
Population	10	10	10	10
Iterations	30	30	30	30
C1	1	Dynamic	Dynamic	Dynamic
C2	3	Dynamic	Dynamic	Dynamic

Table 2 Simulation results

Function	Minimum	Simple PSO	Fuzzy PSO 1	Fuzzy PSO 2	Fuzzy PSO 3
1	1	0.0005	0.0000	0.0001	0.0003
2	0	0.0000	0.0000	0.0000	0.0000
3	0	0.0000	0.0009	0.0000	0.0000
4	0	0.0000	0.0000	0.0000	0.0000
5	−20	0.1042	0.0665	0.0728	0.0743
6	−100.2238	0.1275	0.1277	0.0000	0.0000
7	−18.5547	0.1929	0.2484	0.2645	0.1253
8	0	0.0000	0.0000	0.0003	0.0000
9	0	0.0017	0.0039	0.0157	0.0019
10	0	0.0000	0.0000	0.0001	0.0000

4 Statistical Comparison

To perform the statistical comparison, we have:

3 methods to compare against the simple PSO, (FPSO1, FPSO2, FPSO3).

27 Benchmark mathematical functions.

10 experiments were performed for each method by each function, so it has a total of 270 experiments for each method. Of this total, we took a random sample of 50 experiments for each method for statistical comparison.

The statistical test used for comparison is the z-test, whose parameters are defined in Table 3.

With the parameters in Table 3, we applied the statistical z-test, giving the following results (Table 4):

In applying the statistic z-test, with significance level of 0.05, and the alternative hypothesis says that the average of the proposed method is lower than the average of simple PSO, and of course the null hypothesis tells us that the average of the proposed method is greater than or equal to the average of simple PSO, with a rejection region for all values fall below −1.645. So the statistical test results are that: for the fuzzy PSO 1, there is significant evidence to reject the null hypothesis, as in the fuzzy PSO 3. But in the fuzzy PSO 2, there is no significant evidence to

Table 3 Parameters for the statistical z-test

Parameter	Value
Level of significance	95 %
Alpha	0.05 %
Ha	$\mu 1 < \mu 2$
H0	$\mu 1 \geq \mu 2$
Critical value	−1.645

Table 4 Results of applying statistical z-test

Our method	Simple method	Z value	Evidence
FPSO1	Simple PSO	−2.1937	Significant
FPSO2	Simple PSO	−0.6801	Not significant
FPSO3	Simple PSO	−2.1159	Significant

reject the null hypothesis. In conclusion, two of the proposed variants of PSO were significantly better than simple PSO.

We proposed a method for dynamic adaptation of the parameters of PSO to improve the quality of results. With the results of the statistic test, we can conclude that there is significant evidence to say that the proposed approach could help in the adaptation of parameters in PSO.

Future work includes experiments with functions with more than two dimensions, comparison with other approaches of PSO, for example, PSO with inertia weight and PSO with constriction. Also try to achieve better results for the PSO with fuzzy system 2, more specifically with the input error and the rules of the fuzzy system. In future work also we try to apply the proposed method to other types of problems, for example, optimization of fuzzy systems.

5 Fuzzy Classifier Design

To design fuzzy classifiers were used methods such as PSO simple and proposed methods with parameters adapted dynamically. These methods were applied to different dataset taken from [1, 2, 3, 5, 8, 12, 13], in which the goal is to obtain a fuzzy classifier that "classify" the data in the best way possible. Figure 14 shows an example of a dataset, in this case the Fisher's Iris dataset [5], which shows that it has 4 attributes (length and width of sepal and petal length and width), 150 records and three distinct classes. The figure shows some graphs which visually compares the attributes of Fisher's Iris dataset.

6 Methodology for Designing Fuzzy Classifiers

The methodology proposed for the design of fuzzy classifiers, defined below:

1. Given a dataset, is obtained the number of different classes, and is divided into 70 % for training and 30 % exclusively for testing.
2. From the training data are obtained some necessary features, such as: number of attributes, attribute ranges, etc.
3. It generates a fuzzy classifier of base, from the characteristics obtained.
4. Optimize the rules from fuzzy classifier using the data for training.

Num	Sepal Length	Sepal Width	Petal Length	Petal Width	Class
1	5.1	3.5	1.4	0.2	*I. setosa*
2	4.9	3.0	1.4	0.2	*I. setosa*
3	4.7	3.2	1.3	0.2	*I. setosa*
4	4.6	3.1	1.5	0.2	*I. setosa*
5	5.0	3.6	1.4	0.2	*I. setosa*
⋮	⋮	⋮	⋮	⋮	⋮
73	5.6	2.5	3.9	1.1	*I. versicolor*
74	5.9	3.2	4.8	1.8	*I. versicolor*
75	6.1	2.8	4.0	1.3	*I. versicolor*
76	6.3	2.5	4.9	1.5	*I. versicolor*
77	6.1	2.8	4.7	1.2	*I. versicolor*
78	6.4	2.9	4.3	1.3	*I. versicolor*
⋮	⋮	⋮	⋮	⋮	⋮
146	6.7	3.0	5.2	2.3	*I. virginica*
147	6.3	2.5	5.0	1.9	*I. virginica*
148	6.5	3.0	5.2	2.0	*I. virginica*
149	6.2	3.4	5.4	2.3	*I. virginica*
150	5.9	3.0	5.1	1.8	*I. virginica*

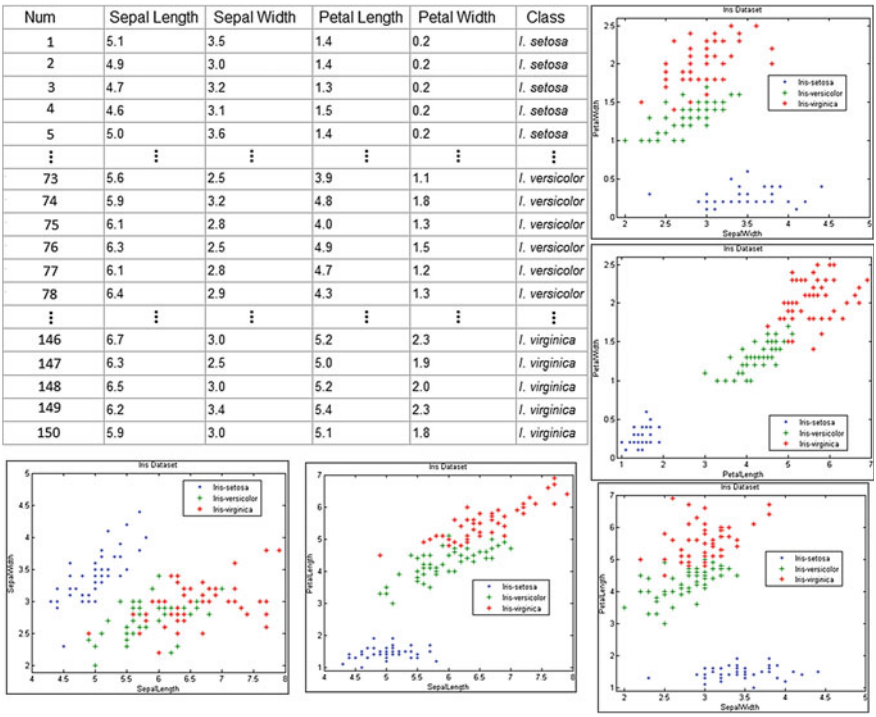

Fig. 14 Fisher's Iris dataset

5. Test the best fuzzy classifier found, so far, with the test data.
6. Optimize the membership functions of the best fuzzy classifier found, which has optimized rules, using data for training.
7. Test the best fuzzy classifier found with the test data.

The following defines each step of the methodology for the design of fuzzy classifiers.

In the first step, given a dataset, we obtain the number of different classes, then is divide the dataset, is taken 30 % of the dataset randomly without replacement, the remaining records, that is, 70 % becomes our set of data for training, but before that, are obtained the number of different classes of 70 %, and if this is less than the total of different classes of full dataset, we proceed to select a new random 30 % of records, until the 70 % of the data contains at least one record of each class, this to guarantee that the fuzzy classifier have all possible output classes.

In the second step, to obtain the necessary features from the training data, we obtain the number of attributes, and ranges are obtained from the minimum and maximum of each attribute.

In the third step, to generate a fuzzy classifier, a structure is created to define a Sugeno-type fuzzy system from scratch, where the inputs are each attribute of the dataset and the ranges of the inputs are the ranges of each attribute, and the output

of the fuzzy classification system, is given by the total number of classes. Also defines the number of membership functions per input, since the system is Sugeno each output, that is each class, will be an integer. The rules of the fuzzy classifier are all possible combinations in the antecedents, and all the consequents are the first class. Figure 15 shows a fuzzy classifier for Fisher's Iris dataset.

In Fig. 15 one can observe the fuzzy system for classification of Iris dataset. The Iris dataset has four attributes, which are: sepal length, sepal width, petal length and petal width, these attributes are reflected in the inputs of the fuzzy system.

Figure 16 shows the inputs and the output of the fuzzy system, as you can see, each entry has its own range, defined by the training data, in addition, each input has two membership functions, placed symmetrically (can be more membership functions but in the example used only two), and has every possible output dataset class.

Figure 17 shows the set of rules of the fuzzy system for classification of the Iris dataset. As can be observed the number of rules is defined by the number of inputs and their membership functions, in the example, there are 4 inputs with 2 membership functions each, so the maximum number of possible combinations in the antecedents is 16 (that is $2 \times 2 \times 2 \times 2$ or 2^4), plus all rules have class number 1 as consequent, that is for simplicity, since the next step is an optimization of these rules.

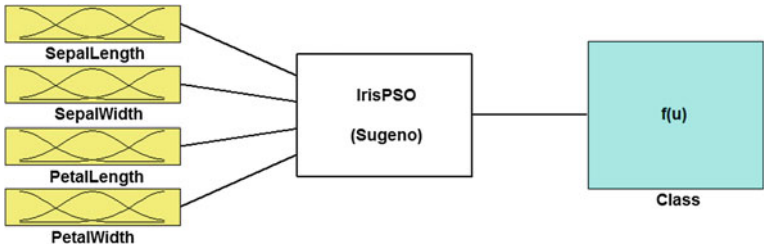

Fig. 15 Fuzzy classifier for Iris dataset

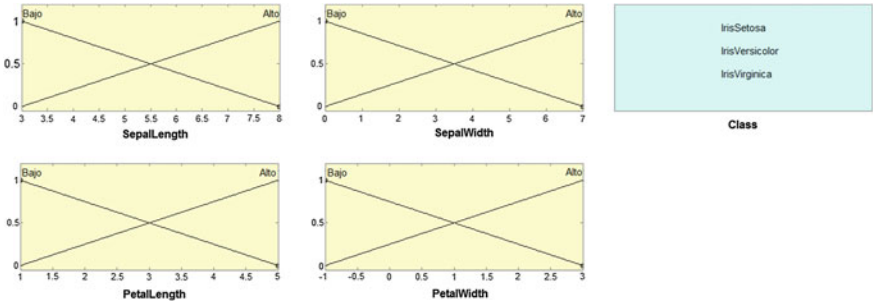

Fig. 16 Inputs and outputs from the fuzzy system for classification of Iris dataset

1. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
2. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisSetosa)
3. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
4. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisSetosa)
5. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
6. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisSetosa)
7. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
8. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisSetosa)
9. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
10. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisSetosa)
11. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
12. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisSetosa)
13. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
14. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisSetosa)
15. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
16. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisSetosa)

Fig. 17 Set of rules from the fuzzy classifier

Only for purposes of assessing the evolution of the fuzzy classifier, we applied this to the classification of the test data, that to obtain an error of classification using Eq. 9. The error in this case is 66.67 %, this because the consequent of all rules is Class 1, and since the dataset contains 3 classes with 50 records each, this is, the third part of the dataset is for a class.

$$Error\ of\ classification = \frac{Misclassified\ Records}{Total\ Records} \tag{9}$$

For the fourth step, are used the PSO methods (simple and proposed), for optimization of the rules of the fuzzy classifier previously generated. The optimization of the rules, in this case, is the modification of the consequents of the rules, so that the length of each particle depends on the number of rules to optimize, in Fig. 18 shown an example of a particle for optimizing the rules of the fuzzy classifier for Iris dataset, where you can see it has 16 positions each corresponding to each rule, and possible values are the numbers 1, 2 or 3, which correspond to the possible classes.

For the fifth step, simply is used the best fuzzy classifier found in the previous step, to classify the test data and obtain an error of classification in this case of 22.67 %. As can be seen by comparing errors before optimize rules (66.67 %) and once optimized, there is an improvement in the data classification.

For the sixth step, the optimization of the membership functions consist in "move" the points of the membership functions of the inputs, continuing the example, there are 4 inputs with 2 triangular membership functions each, so you should to "move" 3 points for each membership function, which gives a total of 24 points, these 24 points become the size of the particle. Figure 19 shows an example of a particle for optimizing the membership functions of the fuzzy classifier system.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	3	1	1	1	1	2	3	1	1	3	3	1	2	2	3

Fig. 18 Particle for optimizing the rules of the fuzzy classifier

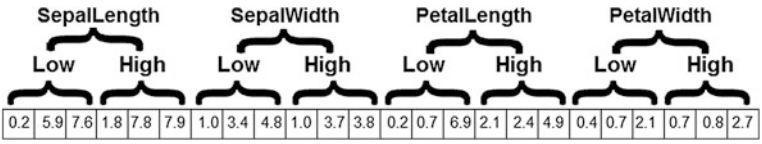

Fig. 19 Particle for optimizing the membership functions of the fuzzy classifier

For the last step uses the best fuzzy classifier found, after optimization of rules and membership functions to classify the test data and obtain a classification error. In this case the classification error is 14 % compared with the classification error optimized keeping only the rules (22.67 %), so that there is an improvement in the classification of data, and is saved the fuzzy classifier with less error.

Figure 20 shown the rules optimized of the fuzzy classifier, and Fig. 21 shown the membership functions of each input once optimized.

1. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
2. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisVersicolor)
3. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
4. If (SepalLength is Bajo) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisSetosa)
5. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
6. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisVirginica)
7. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisVirginica)
8. If (SepalLength is Bajo) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisVirginica)
9. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
10. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisVersicolor)
11. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
12. If (SepalLength is Alto) and (SepalWidth is Bajo) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisVirginica)
13. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Bajo) then (Class is IrisSetosa)
14. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Bajo) and (PetalWidth is Alto) then (Class is IrisVirginica)
15. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Bajo) then (Class is IrisSetosa)
16. If (SepalLength is Alto) and (SepalWidth is Alto) and (PetalLength is Alto) and (PetalWidth is Alto) then (Class is IrisVersicolor)

Fig. 20 Set of optimized rules of the fuzzy classifier

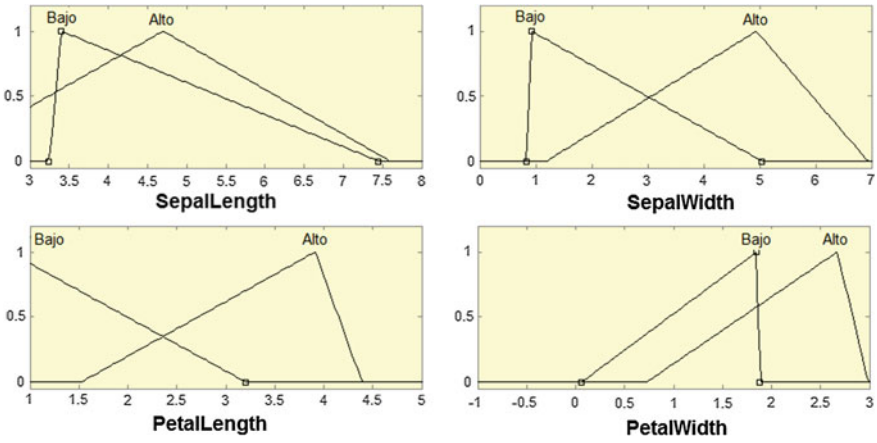

Fig. 21 Optimized membership functions of the fuzzy classifier

7 Experimentation in the Design of Fuzzy Classifiers

For experimentation in the design of fuzzy classifiers, we used dataset taken from [1, 2, 3, 5, 8, 12, 13]. Table 5 shows the main features of these dataset.

As can be seen in Table 5, the data supported by the proposed method (so far) are: numerical and categorical. Addition the proposed method can work with a varied number of attributes (inputs for fuzzy classifier), number of classes (outputs) and number of instances (records to classify).

The parameters used for each method are the same as specified by performing experiments with benchmark mathematical functions, this is, the parameters included in Table 1.

Table 6 shows some of the results of experiments in the design of fuzzy classifiers (first results were taken of each method with each dataset, and in total 10 experiments were performed for each method with each dataset). Moreover, these experiments using Eq. 9, so that have values between the ranges of 0–1.

Table 5 Dataset used in the design of fuzzy classifiers

Name	Instances	Classes	Attributes	Types of data
Abalone [12]	4,177	28	8	Numeric and categorical
Breast tissue [8]	106	6	9	Numeric
Breast cancer Wisconsin [13]	699	2	6	Numeric and categorical
Car evaluation [2]	1,728	4	6	Categorical
Iris [5]	150	3	4	Numeric
Wine [1]	178	3	13	Numeric
Wine quality red [3]	4,898	11	12	Numeric
Wine quality white [3]	4,898	11	12	Numeric

Table 6 Experiments of each method with each dataset in the design of fuzzy classifiers

Dataset	PSO simple	FPSO1	FPSO2	FPSO3
Abalone [12]	0.3556	0.1333	0.2667	0.2222
Breast tissue [8]	0.0667	0.0714	0.1238	0.0905
Breast cancer Wisconsin [13]	0.7813	0.5938	0.6563	0.6563
Car evaluation [2]	0.7399	0.7418	0.7784	0.6956
Iris [5]	0.9346	0.8900	0.8892	0.8708
Wine [1]	0.4815	0.4259	0.3704	0.4259
Wine quality red [3]	0.5208	0.4729	0.4917	0.4417
Wine quality white [3]	0.5687	0.5844	0.5531	0.5401

Table 7 Parameters for statistical z-test

Parameter	Value
Level of significance	95 %
Alpha	0.05 %
Ha	$\mu 1 < \mu 2$
H0	$\mu 1 \geq \mu 2$
Critical value	−1.645

8 Statistical Comparison for Fuzzy Classifiers

For the statistical comparison we have:

3 methods to compare against PSO simple, which are: FPSO1, FPSO2, FPSO3 with 8 datasets.

10 experiments were performed for each method with each dataset, so that it has a total of 80 experiments per method. From this total, took a random sample of 30 experiments for each method.

The statistical test used for comparison is the z-test, whose parameters are defined in Table 7.

With the results contained in Table 7, we applied the statistical z-test, obtaining the results contained in Table 8.

In applying the statistic z-test, with significance level of 0.05, and the alternative hypothesis says that the average of the proposed method is lower than the average of simple PSO, and of course the null hypothesis tells us that the average of the proposed method is greater than or equal to the average of simple PSO, with a rejection region for all values fall below −1.645. So the statistical test results are that: for the FPSO1, there is significant evidence to reject the null hypothesis. But in the FPSO2 and FPSO3, there is no significant evidence to reject the null hypothesis.

One of the main reasons that the methods FPSO2 and FPSO3, have not found sufficient statistical evidence to reject the null hypothesis, is because both use the variable input error, and given that this variable needs to know the minimum and maximum of each experiment, this made the use of this variable does not give good results.

In analyzing the results of the statistical test in the design of fuzzy classifiers can see that only the first method found statistical evidence to reject the null hypothesis, i.e., that the proposed method obtains less error in designing fuzzy classifiers.

Table 8 Results of applying statistical z-test

Our method	Simple method	Z value	Evidence
FPSO1	Simple PSO	−2.1502	Significant
FPSO2	Simple PSO	−0.8841	Not significant
FPSO3	Simple PSO	−1.4242	Not significant

The reason that only the first proposed method obtains good results is because when fuzzy systems are designed to adjust parameters, was taken as a premise, that at the beginning the optimization method (PSO here), should explore the search space to explode eventually found the best area, and to do this, the best variables to use are the iteration and diversity, which are precisely the entries of the first method and the other two methods involve variable error, it is for this reason that the first method can handle diversity and convergence in a better way than the other two methods.

9 Conclusions

We conclude that dynamically adjust parameters of an optimization method (in this case the particle swarm optimization PSO), can improve the quality of results and increase the diversity of solutions to a problem.

Three fuzzy systems were designed for adjusting the parameters for particle swarm optimization, which was obtained in two systems statistical evidence of an improvement in the quality of the results of the method of particle swarm optimization when applied in the minimization of benchmark mathematical functions.

Experiments were conducted with the proposed methods in the minimization of mathematical functions and the design of fuzzy classifiers, and a comparison was made between the method of simple particle swarm optimization and the proposed methods, i.e. with fuzzy parameters adjustment.

By comparing the proposed methods and the simple method of PSO, in the design of fuzzy classifiers was found that only the first method obtained statistical evidence to reject the null hypothesis, which says that in developing this thesis was possible to develop a method for adjusting the parameters C1 and C2 of the PSO using fuzzy logic. And in this way improve the results compared with the simple method of PSO.

References

1. Aeberhard, S., Coomans, D., de Vel O.: Comparison of classifiers in high dimensional settings. Technical Report no. 92-02, (1992), Department of Computer Science and Department of Mathematics and Statistics, James Cook University of North Queensland
2. Bohanec, M., Rajkovic, V.: Knowledge acquisition and explanation for multi-attribute decision making. In: 8th International Workshop on Expert Systems and their Applications, pp. 59–78. Avignon, France (1988)
3. Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J.: Modeling wine preferences by data mining from physicochemical properties. Decis. Support Syst. **47**(4), 547–553 (2009)
4. Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence. University of Pretoria, South Africa
5. Fisher, R.: The use of multiple measurements in taxonomic problems. Ann. Eugenics **7**, 179–188 (1936)

6. Haupt, R., Haupt, S.: Practical Genetic Algorithms, 2nd edn. A Wiley-Interscience publication, New Jersey (1988)
7. Jang, J., Sun, C., Mizutani, E.: Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. Prentice-Hall, Upper Saddle River (1997)
8. Jossinet, J.: Variability of impedivity in normal and pathological breast tissue. Med. Biol. Eng. Comput. **34**, 346–350 (1996)
9. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, IV, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
10. Kennedy, J., Eberhart, R.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
11. Marcin, M., Smutnicki, C.: Test functions for optimization needs. Available at: http://www.bioinformaticslaboratory.nl/twikidata/pub/Education/NBICResearchSchool/Optimization/VanKampen/BackgroundInformation/TestFunctions-Optimization.pdf (2005)
12. Waugh, S.: Extending and benchmarking cascade-correlation. PhD thesis, Computer Science Department, University of Tasmania (1995)
13. Wolberg, W., Mangasarian, O.: Multisurface method of pattern separation for medical diagnosis applied to breast cytology. Proc. Nat. Acad. Sci. **87**, 9193–9196 (1990)
14. Zadeh, L.: Fuzzy sets. Inf. Control **8**, 338–353 (1965)

http://www.springer.com/978-3-319-10959-6

Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics
Theory and Applications
Castillo, O.; Melin, P. (Eds.)
2015, VIII, 195 p. 98 illus., Hardcover
ISBN: 978-3-319-10959-6