CHAPTER 6

Generation of Fuzzy Membership Functions

6.1 In a Nutshell

This chapter further develops the fuzzy guided genetic algorithm (FGA) for clustering, to add the feature of automatic membership function generation in the fuzzy logic module of the algorithm. In this context, a short review of related work in the area of generation of membership functions and rules in fuzzy systems becomes necessary. The intuitive nature of the design of the fuzzy logic-based fitness finding method has been discussed. It was found that the FGA with some modification could be used to derive the membership functions though rule generation will still have to remain intuitive due to the absence of input-output training data. Next, the details of the modified algorithm and its implementation are given. The algorithm is applied to the genomes of B.subtilis and E.coli K12 and the results analyzed and compared with the intuitive cases. The conclusion reviews the features of the modified algorithm and its scope for improvement.

6.2 The Design of Fuzzy Membership Functions and Rules

The basic fuzzy inference algorithm is a generalized implementation of fuzzy logic commonly used in applications. The design elements of the basic fuzzy inference algorithm are the design of membership functions for the fuzzy variables and rules. For each fuzzy variable, the universe of discourse has to be determined. The universe of discourse represents the extent of validity of the fuzzy inference rule on each variable domain. Next the shape of the membership functions has to be fixed. Lastly, the number of membership functions (or the number of mapping categories) and their locations on the universe of discourse has to be found. In the FGA developed (Chapter 3), the membership
functions are assumed to be triangular in shape and this assumption is continued in this study.

In Chapter 2, the two basic modes of membership design, namely, data-driven and linguistic design were surveyed. In most applications, fuzzy rules and membership functions are generated by experts in the area using linguistic design. However, with increasing number of variables, the number of rules increases exponentially which makes the design difficult for the expert. If the partitioning into fuzzy sets is too coarse, a number of patterns may get misclassified. On the contrary, if it is too fine, the number of rules will increase enormously. Also, when the rules are generated from training patterns, when the partitioning is too fine, many if-then rules cannot be generated because of the lack of training examples in all the fuzzy subspaces. Therefore, the choice of the membership functions is difficult.

As it is not always easy to derive fuzzy rules from human experts, several methods have been proposed for automatically generating fuzzy rules from numerical data [Sugeno and Yasukawa 1993; Takagi and Sugeno 1985; Wang and Mendel 1992]. Generation of fuzzy rules from numerical data consists of two phases: to partition the pattern space into fuzzy subspaces and to define a fuzzy rule for each fuzzy subspace. Partitioning the pattern space and rule generation are codependent.

A straightforward method is to use clustering algorithms to partition the pattern space into subspaces with or without overlaps among them, then map the center of each cluster into a rule according to the definitions of fuzzy variables [Abe and Lan 1995; Wang and Mendel 1991]. The extracted rules will be independent of the membership functions, so there is no guarantee that the fuzzy system obtained will have sufficiently good performance.

Ishibuchi et al [1992] have proposed a fuzzy rule-based classification method using a fuzzy grid. For a two-dimensional pattern space, a grid is formed by the intersection of two fuzzy subspaces (one along the X-direction and the other along the Y-direction). The performance of the fuzzy rules depends on the choice of the fuzzy partitions. To reduce the dependency of performance on the choice of the fuzzy partitions, they have
simultaneously employed several fuzzy partitions giving rise to grids of different sizes. Corresponding to each grid, there will be a rule table. This multi rule table method has the drawback of having a large number of fuzzy rules.

Nozaki and Ishibuchi [1996] have proposed an adaptive method to construct a fuzzy rule-based classification system where an error correction based learning procedure adjusts the grade of uncertainty of each fuzzy rule by its classification performance. When a pattern is misclassified by a fuzzy rule, the grade of certainty of the rule is decreased.

A simple but powerful heuristic method for generating fuzzy rules from numerical data was proposed by Nozaki et al [1997]. The method generates fuzzy if-then rules with non-fuzzy singletons (i.e. real numbers) in the consequent parts. From m given input-output pairs of training data, a consequent real number is derived for each fuzzy if – then rule generated from the fuzzy subspaces created by assuming that the domain interval of every input variable is divided evenly into fuzzy sets. The consequent is the weighted mean of output values in the training set. A fuzzy rule is generated if there is atleast one positive weight otherwise the partitions are changed.

Genetic algorithms offer a convenient way to model fuzzy systems. GAs were first used by Karr [1991a] to design a fuzzy logic controller (FLC) for the cart pole problem. The membership functions used were Gaussian in nature and the objective was to minimize the squared difference between the cart and the center of the track that the cart is on, while keeping the pole balanced. Meredith et al [1992] have applied GAs to the fine tuning of membership functions in a FLC for a helicopter. Initial guesses for the triangular membership functions are made by the control engineer and the GAs adjust the parameters that minimize the movement of a hovering helicopter. Lee and Takagi [1993] also tackle the cart problem by taking a holistic approach that uses GAs to design the whole system (rules as well as membership functions).

A key issue in the design of fuzzy systems using GAs is their genotype representation. Thrift [1991] and Hwang and Thompson [1994] encode all the rules into chromosomes while fixing the membership functions. Using several critical points to represent each membership function while using all the possible rules, Karr and Gentry use GAs to
evolve these critical points in order to adjust the membership functions. Due to the
codependence of membership functions and rules, much work has been done in
developing them simultaneously. Homaifer and McCormick [1995] use GAs to tune the
membership functions and evolve the rule set at the same time. The base length of each
triangular membership function and all possible rules are encoded into the chromosomes.
In most applications, only a portion of all possible rules are needed. This will reduce the
length of the chromosomes greatly. Karr [1991b] considers a special case where the
number of rules is provided by an expert together with the complete rules from the rule set
and the antecedents for the remaining ones, so only the consequents of the latter part need
to be evolved and, therefore, need to be encoded in the chromosome. However, most of
the time, it is difficult to know apriori exactly how many rules are to be included in the
each rule and encoded effectiveness information for each rule and membership function.

Due to the complex and nonlinear characteristic of the problem space, uniform distribution
of the fuzzy sets is usually not optimal. Some parts of the pattern space may require fine
partition while others need only a coarse partition. To cope with this problem, Ishibuchi et
al [1995] introduced the concept of distributed if-then rules. They encode all the rules
corresponding to several different fuzzy partitions into a string and apply GAs to remove
the unnecessary rules. Nonlinear functions like Gaussian and sigmoid functions can also
be employed in addition to linear membership functions. Shi et al [1999] have proposed a
GA-based method to evolve a fuzzy expert system. The expert system is coded as fuzzy
rules. A fuzzy rule is represented by a string consisting of the start points and end points
of the fuzzy sets of all the input and output variables.

Arslan and Kaya [2001] have developed a GA to determine the membership functions for
a single-input single-output fuzzy system. The GA finds the base lengths of the triangular
membership functions. The bases are coded by a fixed number of bits. The fitness
function is calculated by minimizing a squared –error function defined by the reference
input-output values.

In the next section, it will be seen how GAs can generate membership functions from an
artificially generated set of input values for each input variable.
6.3 The Design of the Fuzzy Fitness Finder (FFF)

In the fuzzy logic part of the algorithm for clustering, it is not a physical system that is being modeled. The aim is to find the fitness values of the clusters generated by the genetic algorithm. The designer has identified a set of criteria/features whose presence in varying degrees in the cluster determines the validity of the cluster. Methods have been worked out to quantify these features which are the input variables to the FFF. There is only one output which is the quantification of the rather abstract notion of how good the cluster is. The output is a crisp value which is the cluster fitness in percentage.

The membership functions of the input variables, the output variable and the rules were all designed intuitively. The available data-driven methods for rule and membership function generation require a set of input-output training data. For the clustering problem, training data may or may not be available. In the application to the problem of gene grouping in Chapter 5, it is possible to create training data sets for E.coli K12 and B.subtilis genomes for which experimentally predicted gene clusters are available. But for M.Tuberculosis, there are no predicted clusters. Such data is difficult to generate in most cases especially when there are a large number of genome sequencing projects that are currently in progress. A possible approach is to use the experimentally predicted gene clusters of E.coli K12 to generate membership functions and rules and use the same for all other genomes under the assumption that all the genomes belong to the same class of prokaryotic genomes.

A method has been devised to generate the membership functions of the input variables from numerical data on each variable. The shape of the membership functions is assumed to be triangular. In the absence of input-output training data, the rule generation part remains intuitive. The intuitive rules should be re-designed with the new membership functions as the antecedents of the rules would change due to a change in the number and locations of the membership functions. From the purely linguistic design, this is one step towards a data-driven design option.

At this point, a reference has to be made to the work of Hong and Lee [1996] on the induction of fuzzy rules and membership functions from training examples. They have
used a method to cluster the given data on a set of values taken by the output variable to derive appropriate triangular membership functions for the variable. It can be asserted that data lying close together will belong to the same class with high membership values. The values are sorted in ascending order. The sequence of ordered values is converted to a difference sequence by taking the difference between adjacent values. This is converted to a similarity sequence by calculating the similarity between adjacent values from a similarity function based on the differences and the standard deviation of the differences. Using an $\alpha$ - cut of similarity, the sequence is divided into classes. The value of $\alpha$ is the threshold for two adjacent data to be thought of as belonging to the same class. Larger $\alpha$ will have smaller number of groups.

6.4 The Fuzzy Guided Genetic Algorithm for Generation of Membership Functions

The fuzzy guided genetic algorithm (FGA) for clustering has been applied to the gene grouping problem successfully. The design of the fuzzy part has been worked out on paper. But still, the predictions made were in agreement with the experimental results in terms of the high values for the area under the ROC curves. It was found that the FGA can also be applied to the design of membership functions for the fuzzy input variables that form the clustering parameters for the FGA studied so far. This is a case of an algorithm calling upon itself to improve its performance.

The intuitively designed membership functions when replaced by derived values should lead to results that are as good as, or better than the intuitive case. In the absence of input-output training data, it is not possible to compare the intuitive and derived functions. But one can only compare if there is significant difference between the results generated by the two approaches. The FGA for clustering has to be suitably modified to be useable for membership function generation from numerical data on the variables. The guiding principles of the algorithm, however, remain unchanged.
6.4.1 The Modified Algorithm

The algorithm has to be run once for each fuzzy variable. Before this, an artificial data set has to be created that consists of a representative set of values that the variable takes in its universe of discourse. This is the numerical data input to the modified FGA. The algorithm achieves a clustering of this data to yield the number and location of the membership functions. The membership function generation and rule re-construction is a three phase process (Fig. 6.1):

**Phase I**: Data set generation for each variable along its universe of discourse.

**Phase II**: The modified FGA

**Phase III**: Rule base reconstruction with the new membership functions.

![Flowchart for membership function generation using a modified FGA](image-url)
**Phase I Design**

**Aim:** To generate a representative data set artificially that covers as much as possible of the most likely values that the fuzzy variable can take in its universe of discourse.

**Step 1:** From the original sequentially ordered data set of \( N \) data items, generate \( n \) individuals where each individual is created by using a different cut-off value of the predominant criterion.

Being sequential data, each individual is a different partitioning of the data set into contiguous clusters. This step is the same as that used to create the initial population in the FGA (Algorithm 3.2, Chapter 3). The choice of the cut-off values are to be made such that the resulting clusters together constitute a representative sample of clusters that are likely to occur in reality. Cut-off values are generally chosen by the domain expert or using some heuristic criteria.

**Step 2:** For each cluster generated in Step 1, calculate the crisp value for the input variable, which is the score for that variable (Section 3.4.2, Chapter 3). If there are an average of \( m \) clusters formed from each threshold value, then the total number of clusters is equal to \( n \times m = N_D \).

The generated data set consisting of the \( N_D \) values that the variable takes in its universe of discourse is the data set which will be used for membership function generation for the variable by the modified FGA.

**Step 3:** Repeat Steps 1 and 2 for each variable.

The flowchart for Phase I is given in Fig. 6.2.
Phase II Design

Aim: Generation of membership functions using the modified FGA

Step 1: Sort the list of scores.
Let the set of scores be \( X = x_1, x_2, \ldots, x_{ND} \) where \( D_L \leq x_i \leq D_U \)
\( D_L \) and \( D_U \) are the lower and upper limits of the domain of discourse respectively. Let the
sorted list be defined as \( X' = x_1', x_2', \ldots, x_{ND}' \). Now the data is reduced to a sequence of
ordered data. The data is ordered in ascending order of scores.

Step 2: Define \( k \) cut-off values such that \( D_L \leq \text{cut}_i \leq D_U \) where \( i = 1, \ldots, k \).

Step 3: Using the \( k \) cut-off values, generate a population of \( k \) individuals by partitioning
\( X' \) into contiguous clusters. Fig. 6.3 shows how the clusters are formed from the scores
data set.
Fig. 6.3: The ordered scores of an input variable divided into clusters in k ways

It can be seen that each individual is a possible division of the domain of discourse into membership functions. The fuzzy sets are the bases of the clusters. The membership functions are assumed to be linear triangular.

**Step 4:** Clusters with zero or very small supports or bases are merged with any of its two neighbours.

**Step 5:** Overlapping clusters

Though the clusters shown in Fig. 6.3 appear to be contiguous, the fuzzy sets are actually discontinuous at the boundaries. Let \( a \) and \( b \) be two consecutive points in the ordered score data set. If \((b-a) > \text{cut-off value}\) then a new cluster (set) starts off from the point \( b \). The space between points \( a \) and \( b \) is not covered by any set. Let \( A \) be the fuzzy set containing \( a \) and \( B \) the set containing \( b \). Let set \( A \) end at point \( b \) and set \( B \) begin at point \( a \). This changes \( A \) and \( B \) from disjoint sets to overlapping sets. Overlaps are naturally required in the membership functions.

**Step 6:** The Modified FGA

The modifications in the FGA are in the crossover and mutation processes and the fuzzy fitness finder. The strategies for crossover and mutation followed in this implementation are explained.

Crossover – When two individuals are paired off by the selection process, a random number in the range \([1, N_d]\) is selected as the crossover point. The clusters (or fuzzy sets)
Generation of Fuzzy Membership Functions

up to this point are copied into the new individuals. The remaining clusters are interchanged in both the individuals resulting in two new individuals. Following crossover, there can be overlaps between fuzzy sets and sometimes some parts of the universe of discourse may not be covered by any fuzzy set (dead zones). The dead zones are covered by extending the base points of fuzzy sets as in step 5. The case where the base of one fuzzy set is completely covered by another, the subset is removed from the individual as this is a case of total overlap.

Mutation – Mutation brings about random variation within the individual in its fuzzy sets with a predefined probability. Probability of mutation ($p_m$) is defined as a percentage of the number of clusters in an individual. The number of times mutation will be performed in an individual will be equal to ($p_m^* N_D$). A cluster is chosen at random and thereafter another random number (0/1) decides whether the left base point or the right base point has to be moved. Presently, the left point can be shifted leftwards or the right point can be shifted rightwards by a small number of points. The overlap increases as a result of mutation. There can be other such mutation operations defined.

The Fuzzy Fitness Finder – A method has been evolved to estimate the fitness of a fuzzy set which is required during the selection process and whenever a fuzzy set is altered. For a symmetric triangular set, we expect more number of points closer to the center or the peak (middle) with a taper towards the sides. The cloud of points around the center should have maximum density with the cloud density thinning towards the sides of the set. The distribution of points on the support of the fuzzy set is taken as a means to evaluate the fitness of the set. Fig. 6.4 shows a possible distribution where the middle 50% should contain at least 80% of the points. This may be defined as the core of the fuzzy set. Then the tail portions should ideally contain about 10% of the points each.

![Fig. 6.4: A fuzzy set divided into three parts](image-url)
The three parameters together decide the fitness of the fuzzy set. We model the fuzzy inference engine based on the values of these parameters. Each parameter can take crisp values in the range (0,100). The range is divided into linguistic hedges (sets) like ‘very high’, ‘medium’, ‘low’ etc. A very high value for all the three parameters should make the fitness of the fuzzy set very high. The fuzzy rule base has three antecedents and one output or consequent. The fitness of an individual is taken as the average fitness of the constituent fuzzy sets or clusters.

The flowchart for Phase II is given in Fig. 6.5.

![Flowchart for Phase II](image)

**Fig. 6.5: Flowchart for membership function generation**
Phase III Design

**Aim**: To reconstruct the rule base and compare with the intuitive design.

**Step1**: The old rules become meaningless after the generation of new membership functions for each input variable. The new rule base has to be constructed with the new antecedents intuitively again as we do not have any test data but only a perception about how the input variables combine. In the process, the output sets may also be modified.

**Step2**: To prove the efficacy of the new method, define an integer vector called ‘rulehits’. This vector contains the number of times each rule in the rule base was activated. The number of rulehits of each rule in the rule set in the two cases (intuitive and generated) are compared.

**Step3**: If there are output values (clusters) available for a particular case, the performance of the generated membership functions can be validated by comparing the resultant clusters against the expected results.

The test data set of scores for a variable created in design phase I can be generated using another set of cut-off values. The modified FGA can be run on different score data sets for every variable. The rules generated can be compared with the intuitive as well as the generated cases. This completes the design phase for the generation of input membership functions.

6.4.2 Implementation

The modified FGA was implemented in the two genomes of B.subtilis and E.coli K12 for which experimental operons are available to plot the ROC curves.

**Case 1: B.subtilis**

The two input variables for which data is available are intergenic distance and participation in metabolic pathways. It has been observed that typically, the intergenic distance between genes in an operon does not exceed 600 bps and is mostly in the range [-4,+4] where distance is negative for overlapped genes. Hence, in the case of intergenic
distance, the intuitively designed membership functions have reason to be accurate enough. Membership functions will therefore be generated only for the pathway criterion.

**Phase I: Generation of score data sets of input variables**

Data Set Generation for Metabolic Pathways

Number of genes in the B.subtilis genome, \( N = 4225 \)

Number of cut-off values, \( n = 10 \)

Total Number of clusters generated by the cut-off values = 545

Range of the universe of discourse for metabolic pathway scores – \([0,1]\)

Pathway scores for the 545 clusters are calculated. This forms the input data set for Phase II. The ordered sequence reads as 0, 0,…,0.0074,…,0.3,…, 0.72,…,1.

**Phase II: Generation of membership functions**

The following is a member of the initial population of membership functions for the pathway criterion.

4

\((0, 0.5), (0.357143, 0.583333), (0.535714, 0.714286), (0.644444, 1)\).

There are four fuzzy sets and the lower and upper limits of each set is given. Fig. 6.6 shows the linguistic hedges, names and limits of the fuzzy sets that were selected for the fuzzy system after the modified FGA was run.

![Fig. 6.6: Generated membership functions for pathway criterion](image-url)
Generation of Fuzzy Membership Functions

<table>
<thead>
<tr>
<th>Lower limit</th>
<th>Midvalue Limit</th>
<th>Upper Limit</th>
<th>Left score (%)</th>
<th>Mid score (%)</th>
<th>Right score (%)</th>
<th>Total score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00068</td>
<td>0.00068</td>
<td>0.2667</td>
<td>9.7682</td>
<td>80.00</td>
<td>9.768</td>
<td>90</td>
</tr>
<tr>
<td>0.20714</td>
<td>0.39024</td>
<td>0.5833</td>
<td>7.6923</td>
<td>69.231</td>
<td>10.00</td>
<td>90</td>
</tr>
<tr>
<td>0.53571</td>
<td>0.62500</td>
<td>0.64410</td>
<td>9.5238</td>
<td>76.191</td>
<td>10.00</td>
<td>90</td>
</tr>
<tr>
<td>0.61111</td>
<td>1.000000</td>
<td>1.0000</td>
<td>2.4823</td>
<td>80.00</td>
<td>2.4822</td>
<td>70</td>
</tr>
</tbody>
</table>

Table 6.1: Fuzzy sets and scores generated by the modified FGA.

Corresponding to Fig. 6.6, the fuzzy sets chosen for implementation, their scores on the three parameters (left, middle and right) and their combined scores (fitness) in percentage are given in Table 6.1. The average cluster fitness is 82.37. The first and the last sets are right triangles. In these two cases, there are only two sets instead of three, half of the middle portion and one of the tail portions. For the sake of uniformity, the percentage of points in the tail portion is divided by two to account for the other tail portion.

**Phase III: Reconstruction of Rules**

The rules have to be reconstructed with the new antecedents generated by Phase II. In the intuitive case, there were nine fuzzy sets for each input variable and hence 81 rules. In the generated case, while the membership functions remain same for intergenic distance, there are only four sets for the pathway criterion thereby reducing the number rules to 36 ($9*4$). The number of output sets were kept unchanged at nine.

The program is run for both the intuitive and generated cases. The number of rules and the rules themselves are different for the two cases. The rulehit vector which is a measure of the significance of the rules, has 81 elements for the intuitive case and 36 elements for the generated case.
**Case 2: E.coli K12**

**Phase I: Generation of score data sets of input variables**

Data Set Generation has been done for three criteria: Metabolic Pathways, Conservation across genomes and protein function similarity. For intergenic distance, as before the intuitive sets have been retained.

Number of genes in the E.coli-K12 genome, \( N = 4405 \)

Number of cut-off values, \( n = 10 \)

Total Number of clusters generated by the cut-off values = 4520

Range of the universe of discourse for scores on the three input variables – \([0,1]\)

Pathway, Conservation and protein similarity scores for the 4520 clusters are calculated. The three score data sets generated, form the input to Phase II.

**Phase II: Generation of membership functions**

The FGA generates five fuzzy sets for conservation, five for pathway and seven for the protein similarity criteria respectively. Fig. 6.7 shows the linguistic hedges, names and limits of the fuzzy sets for the three criteria that were selected for the fuzzy system after the modified FGA was run.
Phase III: Reconstruction of Rules

The rules were reconstructed with the new antecedents generated by Phase II. As there are four input variables, there are three rule bases as the fuzzy inferencing works in a cascading manner. Intergenic distance with nine fuzzy sets is combined with conservation having five sets. The first rule base has 45 rules (9*5). The output variable, namely fitness has nine sets, same as in the intuitive case. The output is combined with the pathway criterion which has five sets. This rule base also has 45 rules (9*5). The second output is combines with the protein criterion which has seven fuzzy sets. So, the third rule base has 63 rules (9*7).

The rulehit vector which is a measure of the significance of the rules has 243 elements (81*3) for the intuitive case and 153 elements (45+45+63) for the generated case.
6.5 Analysis of Results

In both B.subtilis and E.coli K12, the rule set is smaller in the generated case. But this need not be the case always. It is also possible that in the intuitive case, the number of membership functions was under-designed and the generated case has more sets. What is important is the usage of the rules.

Though both the old rules and the new rules are designed by intuition, the new membership functions have an effect on the quality of the rules. It was found that in the old rule base, 22% of the rules had zero rule hits. Amongst the new rules, only 8% have zero rule hits. In the old rules, 25% of the rules had rule hits less than 10 whereas the figure was only 13% with the new rules. This shows that unused rules and rarely used rules decrease in the case when the membership functions are generated.

As the experimental operons are available for B.subtilis and E.coli K12, the ROC curves are plotted for both cases. It was found that the area under the curve for intuitive case (Chapter 5) for B.subtilis with two criteria was 0.88. The ROC curve for the generated case gives an area of 0.9 (Fig. 6.8)

![ROC Curve](image-url)

Fig. 6.8: ROC curve for B.subtilis with two criteria – distance and pathway
The area under the curve for intuitive case (Chapter 5) for E.coli K12 with four criteria was 0.9. The ROC curve for the generated case gives an area of 0.93 (Fig. 6.9).

When a scores data set is generated, if the distribution of points in the universe of discourse of the variable is uniform then an equal division of the range of the variable is enough keeping the number of fuzzy sets as 5 or 7 which is generally considered as optimum in practice. In such a case, there is no need to generate the membership functions.

### 6.6 Conclusions

The fuzzy guided genetic algorithm has been used to generate membership functions. The results prove to be better than the intuitive case in terms of the ROC plots. But there is plenty of scope for further research.

There can be variations in crossover and mutation which are left to the discretion of the designer. The operators of the GA (crossover and mutation) can be augmented with more options. The parameters of the algorithm can be changed. The probability of mutation and crossover can be changed. The probabilities can also be made to vary within the
program. The number of points that the mutation operator shifts in order to change the bases of the fuzzy sets is a variable that can be changed. The break-up of the points in the fuzzy set as 10-80-10 is also not fixed. The algorithm can be tuned by changing one or all of these parameters.

The fuzzy fitness finder is presently, considering only the distribution of points along the support of the fuzzy set. Other important criteria like the permitted overlaps between sets, the number of fuzzy sets (balance between coarse partitioning and fine partitioning) etc. can also be incorporated into the fitness finder. There are no hard and fast rules for the generation of membership functions. An attempt has been made to show how the algorithm can be used for the generation of membership functions.