Extracting relevant information from input-output data

Carles Garriga Berga[†]

[†]Enginyeria La Salle. Universitat Ramon Llull.
cgarriga@salleURL.edu

Abstract

In this paper a set of heuristic criteria devised to address the problems encountered in designing a fuzzy system from input-output data is presented. In particular, we show how to discriminate unsignificant linguistic variables, determine the number of fuzzy sets, place them in the universe of scope, and propose a set of linguistic rules. The objective is to obtain in a simple and fast manner an algorithm simpler than the standard ones with minimum computational cost but still similar performance and more intelligibility in most cases.

Keywords: fuzzy identification, linguistic modelling.

1 Introduction

A former application of fuzzy logic was the developement of intelligent systems emulating one's linguistic capacity leading to many fuzzy identification methods based on clustering. These algorithms commonly consist in fitting a set of input-output data pairs achieving a lower error in prediction than the previous one. In most cases, the more precision they achieve the poorer linguistic intelligibility they present.

Among the popular ones, we can mention Fuzzy C-means [9] or Possibilistic C-means [6]. After clustering the transfer function, one can approximate the fuzzy sets needed to define the system. Some authors combine clustering techniques with genetic algorithms to obtain a fine tunning for the fuzzy sets [2]. In other methods ellipsoidal regions for the clusters are defined obtaining a good performance but at the price of increasing the computational cost [1, 3]. Alternative methods include taking clustering decision to obtain the minimum entropy [5]. Here we are interesed in recovering the linguistic capabilities of fuzzy identification systems rather than improving their precision and with a computational cost adequated for on-line applications.

Given a set of raw data, a number of problems arise during the design procedure such as determining the relevant variables, how the universe of scope should be partitioned or which is the best rule base. In this paper we present an enhanced version of the one presented in [4] as a set heuristic criteria devised to address the design problems. In particular, we are able to discriminate unsignificant linguistic variables, determine the number of fuzzy sets, place them in the universe of scope and propose a set of linguistic rules.

The goal is to obtain in a simple and fast manner a good starting model that explains the most significant relations between the output and inputs. These criteria can be set up in an algorithm which first computes the relevance of the variables, then determines the number and position of the necessary fuzzy sets to describe each relevant variable and finally computes a fuzzy rule base. The result is an algorithm simpler than the standard ones with minimum computational cost but still similar performance and more intelligibility in most cases.

2 Simplifying fuzzy systems

Among the several methods to model a system, those based on fuzzy logic not only try to obtain a good representation of the relations between output and inputs but also a linguistic explanation. Even inside fuzzy logic the model is not unique.

We can consider the case of a proportional (P) and derivative (D) control of a process to exemplify

the differences between fuzzy models. A PD control is a system with two inputs and one output defined by the equation

$$u = K_P e + K_D \frac{de}{dt} \quad (1)$$

where the input called e is the error defined as the difference between the current output of the process and its desired value and de/dt is the first derivative of this error. There are also two parameters called K_P and K_D that are adjusted depending on the process to control. Here we'll consider $K_P = 1$ and $K_D = 1$ to simplify the example and a domain between { -1,+1 } for the three variables.

A possible solution for modelling this system is the fuzzy model with 3 sets for each input and 5 sets for the output variable plotted in figure 1. This model with 9 linguistic rules presented in table 1 is commonly found in literature.

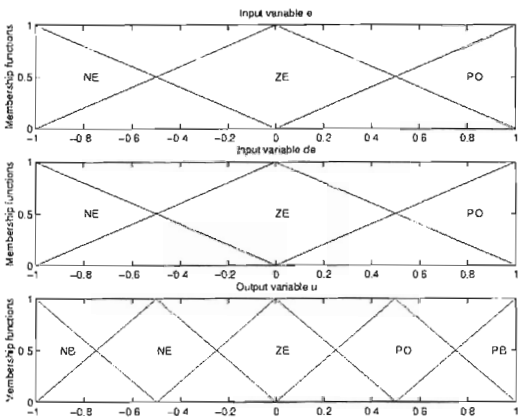

Figure 1: Fuzzy sets of the fuzzy PD

Rule matrix			
$de \setminus e$	**NE**	**ZE**	**PO**
NE	NB	NE	ZE
ZE	NE	ZE	PO
PO	ZE	PO	PB

Table 1: Linguistic rules of a typical fuzzy PD

An alternative to the previous solution is the simplified model proposed in table 2 with 4 rules and only two sets for each input variable and three sets for the output variable plotted in figure 2.

Both models perform well because their transfer functions are very similar and close to the original one depending on the implication and defuzzyfication methods selected. Nevertheless, the best

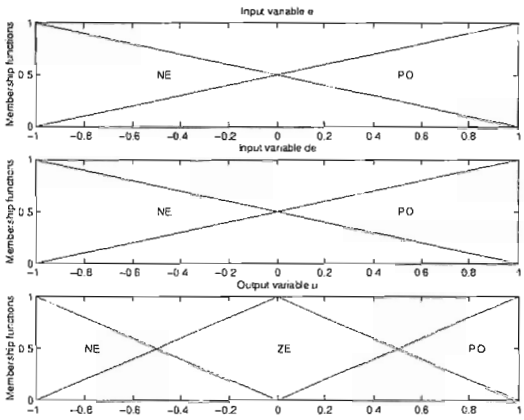

Figure 2: Fuzzy sets of the simplified fuzzy PD

Rule matrix		
$de \setminus e$	**NE**	**PO**
NE	NE	ZE
PO	ZE	PO

Table 2: Linguistic rules of the simplified fuzzy PD

linguistic model of this problem should match two characteristics: linguistic interpretation and low computational cost, and the last solution seems to be very close to it. In fact we can demonstrate that the simplest fuzzy model of a plane (a PD control has a planar transfer function) is given by the last proposed solution if we use product-product implication and weight counting defuzzyfication[1].

So a clue to obtain fuzzy models with a minimum number of rules is identifying planes (or hyperplanes for more than two input variables) each of them defining the necessary fuzzy sets. The fuzzy sets of the input variables are given by the margins of those planes (or hyperplanes) while the output sets of each possible rule are given by the value of the plane (or hyperplane) itself. So the problem when looking for a simple model is reduced to a planar decomposition of the input-output data.

3 Fuzzy curves

In general, applications related with identification appear as high computational cost methods. Any reduction in this aspect would be appreciated if online processing is required. The method we pro-

[1] Proof of assessment 1 at the end of the article

pose consist basically in identifying planes (or hyperplanes) and the conventional solutions to this problem require a high number of operations.

In fact, most algorithms applied to define those planes (or hyperplanes) have a computational cost proportional to $\prod_{i=1}^{N} b_i$ considering N input variables and each one defined with $B_{1..N}$ bits. An alternative to reduce this time is obtained using fuzzy curves [7] because the computational cost is then proportional to $\sum_{i=1}^{N} b_i$. Obviouslly, the advantage of reducing execution time is counteracted by eliminating the possible correlations between input variables.

Original fuzzy curves proposed in [7] for an input variable x_i with m input-output samples (x_{ik}, y_k) are defined as

$$fc_i(x_i) = \frac{\sum_{k=1}^{m} \phi_{ik}(x_i)\, y_k}{\sum_{k=1}^{m} \phi_{ik}(x_i)} \tag{2}$$

where $\phi_{ik}(x_i)$ represents a fuzzy set placed at x_{ik}

$$\phi_{ik}(x_i) = \exp\left(-\left(\frac{x_{ik} - x_i}{\beta}\right)^2\right) \tag{3}$$

and β is a constant proportional to the length of the universe of the scope (typically 20%). Can be easily observed that fuzzy curves work as weighted interpolation functions of the samples close to x_i similar to other methods like radial basis functions but with fuzzy approximations. Even more, in order to reduce the computational cost, exponential functions can be changed by linear functions.

Fuzzy curves can be used to extract the fuzzy sets of each input variable computing the first, second and/or third derivative of the fuzzy curve and looking for its values close to zero because these values match planes (or hyperplanes) similar to the real transfer function. The first derivative gives information about the maximums and minimums, the second one gives information of inflexion points and the third one gives information about sudden change of slopes.

In this point, a low-pass filter is necessary to smooth the fuzzy curve before computing its derivatives as well as a threshold to avoid noise problems when searching for the zeros.

Let's illustrate this fact through the following example. Consider the resulting fuzzy curve of an input variable plotted in figure 3 whose most significant points, namely those necessary to approximate the original function with a piecewise function, have to be found.

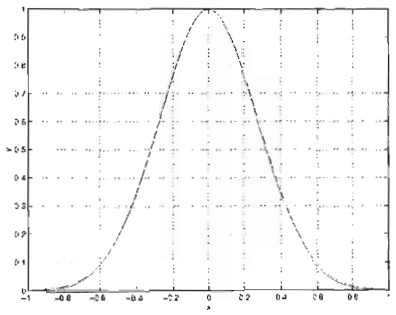

Figure 3: Example of a fuzzy curve

These points can be selected after considering the first, second and third derivative of the fuzzy curve whose values are plotted in figure 4 together with the threshold applied in each case which will be defined in the next section. From the first derivative we can observe that a fuzzy set should be placed close to 0. From the second derivative we can observe that two fuzzy sets should be placed close to -0.25 and 0.25. From the third derivative we can observe that three fuzzy sets should be placed close to -0.48, 0 and 0.48.

So with a maximum number of 7 fuzzy sets for this variable placed at -1, -0.48, -0.25, 0, 0.25, 0.48 and 1 because the limits of the scope are also included as boundary sets, we should have enough partitions to approximate the original function with acceptable precision. In fact, it's not necessary to consider all derivatives and typically only first and at the most the third derivative should be considered and consequently only 5 sets.

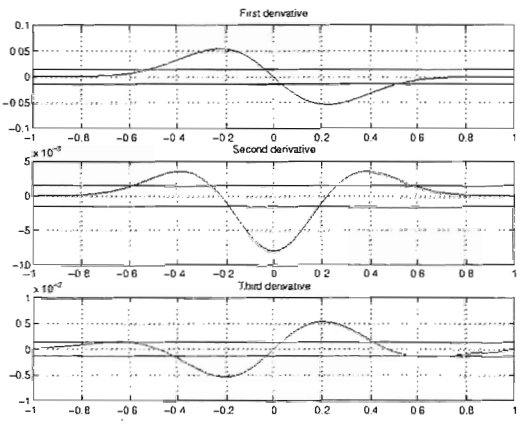

Figure 4: Derivatives of the fuzzy curve

We have also defined extended fuzzy curves to fix the output fuzzy sets for any combination of N input values as

$$efc(x_{1..N}) = \frac{\sum_{k=1}^{m} \left(\prod_{i=1}^{N} \phi_{ik}(x_i) \right) y_k}{\sum_{k=1}^{m} \left(\prod_{i=1}^{N} \phi_{ik}(x_i) \right)} \quad (4)$$

as weighted interpolation function but only computing the values we are interested in, that is the values that result of the combinations of the input fuzzy sets. In the previous example the resulting values of the extended fuzzy curve would be the values of the original fuzzy curve evaluated at -1, -0.48, -0.25, 0, 0.25, 0.48 and 1.

4 Outline of the algorithm

Let us remind in this section the basic steps necessary to implement the overall method. First we show how to evaluate the fuzzy curves necessary to compute the relevance of each linguistic variable. The result is a real number between zero and one proportional to the importance of each variable so only variables with a minimum relevance are considered. Then we compute the placement of the fuzzy sets using fuzzy curves and looking for the most significant values of it with the first, second or third derivatives close to zero that fix the edges of the fuzzy sets. The last one is the assignement of the output fuzzy sets using the extended fuzzy curves and consequently the linguistic rules. Important refinements are given at the end.

4.1 Fuzzy curves

Fuzzy curves work as interpolation functions that relate the output to each input. First of all it's necessary to decide the number of points of the fuzzy curve. These will be proportional to the number of bits of each variable which can be fixed or adjusted, for instance, doing a first analysis of the number of bits necessary to have at least one sample for each point. A simplified pseudocode version is the following :

```
For each Input variable
   Optimal_partition is FALSE
   Numer_of_bits is a constant,
   While Optimal_partition is FALSE
     Generate 2^Number_of_bits - 1 equidistant
        points (Pi) from min(Input) to max(Input)
     If some Pi does not have output samples
        Decrease Number_of_bits by one
     else
        Optimal_partition is TRUE
   Generate N=2^Number_of_bits - 1 equidistant
     points (Pi) from min(Input) to max(Input)
   Generate the fuzzy curve (Fi) with N points
   Low pass filtering of the fuzzy curve
```

4.2 Relevance of the variables

The relevance of the variables is obtained by computing the margins of the fuzzy curves (distance between the minimum value and the maximum value). Variables not correlated with the output values present a short margin and can neglected for further steps.

```
For each Input variable
   Margin Mi = max(Fi)-min(Fi)
   Relevance Ri = Mi / max ( Mi )
   Eliminate if Ri<threshold
```

4.3 Fuzzy sets

The input fuzzy sets are obtained by evaluating the first, second and/or third derivative of the fuzzy curves which are defined by $N = 2^{Number\_of\_bits} - 1$ equidistant points in the universe of the scope of each variable. Therefore, when projecting the points where the i-th derivative is close to zero onto the universe of scope, the edge points for the different fuzzy sets will be obtained. Sometimes a great number of fuzzy sets in the range of the universe of the scope can appear requiring a later adjustment. A threshold is necessary when looking for the zeros in order to reduce noise intereferences which is proportional to half of the power of the samples as $\frac{1}{2}\left(\frac{1}{N_b}\sum_{i=1}^{N_b} x_i^2\right)^{1/2}$.

```
For each Input variable
   Compute the first (d^1F), second (d^2F) and/or
     third derivative (d^3F) of Fi
   Compute the points of d^1F, d^2F and d^3F close
     to zero (with threshold)
   Compute the mean of the closest points
```

4.4 Linguistic rules from extended fuzzy curves

Extended fuzzy curves have been developed as interpolation method of systems with more than one input variable being the extension of the fuzzy curves defined with only one input. We can use extended fuzzy curves to decide the output values for each possible combination of input sets. Similarly to the input sets, this method may bring about a great number of fuzzy sets in the range of the

universe of the scope that requires joinning those that are very close. Other techniques have been tried like the popular method developed by Wang and Mendel in [10] achieving similar results but increasing the complexity of the algorithm and so its computational time.

```
For each possible rule
   Compute the output fuzzy sets focused on the
     result of the extended fuzzy curve and
     considering only the input sets of the rule
Compute the mean of the closest sets
```

4.5 Refinements

The whole method includes some refinements that must be considered. Some of them are also necessary to reduce the computational cost and optimizing memory management and only those related with the method are here presented.

Some statistical techniques have been included to discriminate some samples whose value could have been significantly altered by noise. At the beginnig, samples are ordered in quartiles in order to locate those values bigger than one and a half times the range between the first and the third quartile measured from the third quartile or lower than one and a half times this range measured from the first quartile. If these values are also very different from a linear predictor they are rejected.

Sometimes it's necessary to divide samples in different groups if an input variable x_i can mask the computation of the fuzzy curve of another variable x_j. This happen if several samples with the same value for the variable x_j have the same magnitude but different sign depending on the value of the other input x_i. An example could be the Matlab's peaks function plotted in the figure 5 where the fuzzy curve from every input will be very planar because negative values tend to compensate positive values. Another solution is working temporary with the squared values of the samples.

Relating to the first, second and third derivative, a simple but fast method is recommended because we are more interested in the shape than precision. We propose the simplest method based on differences (in fact correlations with [−1 0 1] vectors) with a later windowed average. The form of the fuzzy functions seems not to be relevant when extracting the most relevant information from samples as we've observed from examples. Linear functions are prefered in order to reduce the computational cost but exponential functions perform better

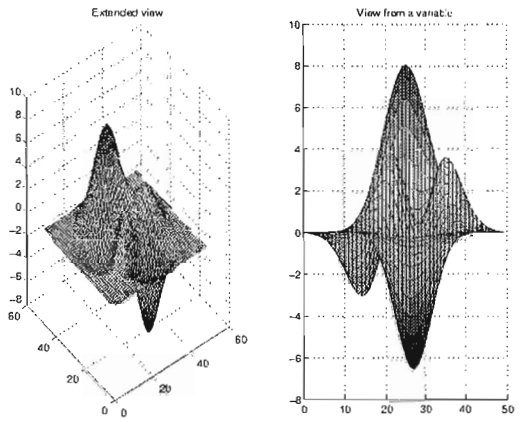

Figure 5: Matlab's peaks function

if samples are plenty of noise because of its smooth derivatives.

When fixing the fuzzy sets by identifying the planes (or hyperplanes) of the system with the derivatives of the fuzzy curves, the first and third derivatives seem to be the most important ones while the points obtained with the second derivative are typically masked.

5 Example

In order to exemplify the whole method, we'll consider the system defined by

$$z = \exp\left(-y^3 + 0.5y\right) + \tanh\left(3\left(x^3 - 0.5x\right)\right) \quad (5)$$

where the input variable $x \in [-1, 1]$ and the input variable $y \in [0, 2]$. Its transfer function is plotted in figure 6 and the model resulting should explain the dependences between the output values and both inputs. A total amount of 441 samples are considered.

First of all, our method computes the fuzzy curves of both inputs which are plotted in figures 7 and 8 together with the samples considered for its estimation. Only 16 points (N=4) have been considered for each variable. Because of the margins of both variables are similar, 1.81 and 1.11, none of them is rejected.

The first, second and third derivative of the fuzzy curves are computed in order to decide the best values of the universe of the scope where the fuzzy sets should be placed. These derivatives together with the thresholds necessaries to avoid noise problems

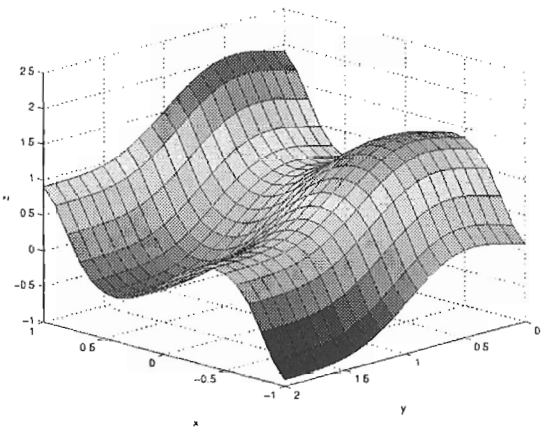

Figure 6: Real transfer function

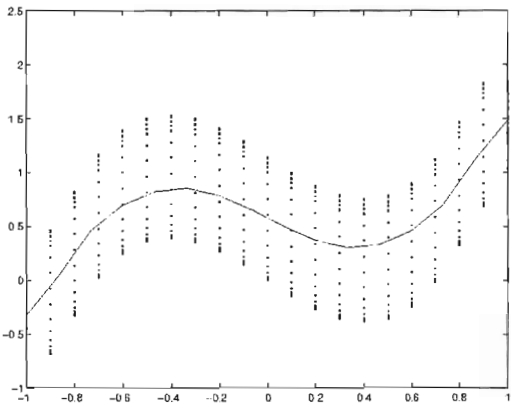

Figure 7: Fuzzy curve of the variable x

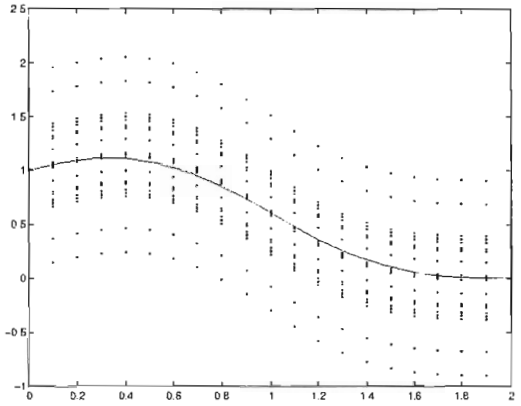

Figure 8: Fuzzy curve of the variable y

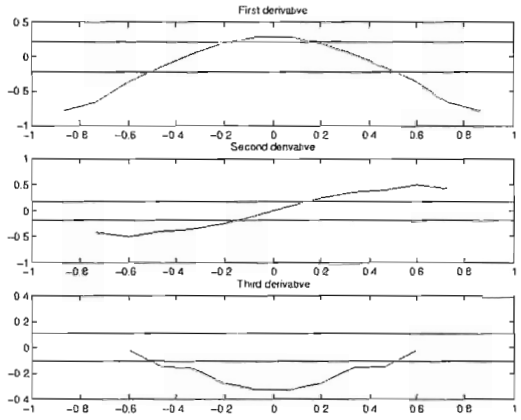

Figure 9: Derivatives of the variable x

are plotted in figures 9 and 10. For variable x we will consider 5 sets placed at -1 (limit), -0.33 (minimum), 0 (inflexion point), 0.33 (maximum) and 1 (limit) while for variable y we will consider 3 sets placed at 0 (limit), 1 (inflexion point) and 2 (limit). A lower threshold would give a higher number of points but decreasing the intelligibility of the model and also reducing noise immunity.

The output fuzzy sets are computed with the extended fuzzy curves evaluated for each combination of input sets giving the values presented in table 3. A last processing has joined those values which were very close and finally only 9 sets have been considered for the output variable placed at -9.0e-1, -2.9e-1, +4.8e-2, +3.0e-1, +6.7e-1, +9.3e-1, +1.3e+0, +1.5e+0 and +2.0e+0.

So the final model proposed has 15 rules with 5 sets for the variable x, 3 sets for the variable y and 9 sets for the output variable and its transfer function is plotted in figure 11 considering linear fuzzy sets and computing the implication with the product and the defuzzyfication with weight counting. A function smoother than this could have been obtained if applying exponential functions but increasing the processing time without improving linguistic intelligibility. In figure 12 the error defined as the difference between the original transfer function and the modelled one has been plotted observing a non biased result. Thus, errors are only located inside the planes so the more planes the more precision but decreasing linguistic intelligibility.

6 Discussion

An algorithm that is able to give a first-approach of the fuzzy systems and that relates input-output pairs has been presented and discussed. Its main feature is that it is quite simple and extracts relevant linguistic information in a fast manner. This

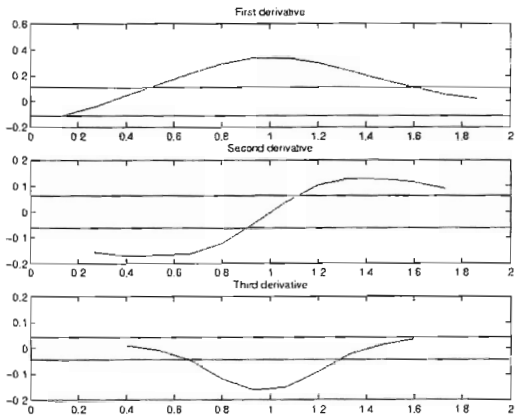

Figure 10: Derivatives of the variable y

Extended fuzzy curve			
$x \setminus y$	**0**	**1**	**2**
-1	+9.48e-2	-2.97e-1	-9.04e-1
-0.33	+1.27e+0	+8.84e-1	+2.77e-1
0	+1.00e+0	+6.07e-1	+9.11e-4
0.33	+7.23e-1	+3.31e-1	-2.75e-1
1	+1.90e+0	+1.51e+0	+9.06e-1

Table 3: Values of the extended fuzzy curve

is obviously a first-approach and the resulting system could be fine tuned using other, more powerful tecniques. We suggest the gradient descent algorithm [8] because if we have a good approach of the result, this algorithm converges fast to the optimal solution. The whole algorithm will be optimized and presented in a forthcoming work.

In spite of the surprising results that this and other identification algorithms can offer, the most important thing one must bear in mind when trying to identify a system, is the necessity of a good set of samples of all the variables because they will be at the end the responsible of the result.

We are currently working on applications where linguistic interpretation is required such as economy modelling, social analysis and scientific studies. Other applications related with on-line control processes of chemical reactions are being considered.

Acknowledgments

Author would like to thank Dr. Xavier Vilasís for his help on writting this work.

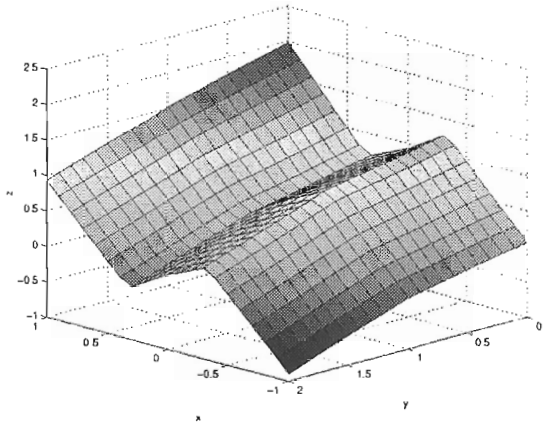

Figure 11: Transfer function of the model

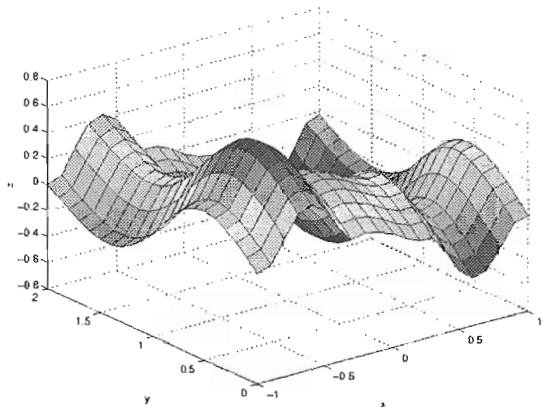

Figure 12: Error between transfer functions

Proof of assessment 1

Statement: A planar transfer function can be obtained with a four rule fuzzy system with two sets for each input variable located at the margins of their domains.

Consider a plane defined by

$$y = x + y \tag{6}$$

where $x \in [X_A, X_B]$, $y \in [Y_A, Y_B]$ and then $u \in [X_A + Y_A, X_B + Y_B]$.

An exact model can be obtained with two fuzzy sets placed at X_A and X_B, two fuzzy sets placed at Y_A and Y_B and four output fuzzy sets placed at $U_A = X_A + Y_A$, $U_B = X_A + Y_B$, $U_C = X_B + Y_A$ and $U_D = X_B + Y_B$ as following

$$\mu_{X_1}(x) = \frac{X_B - x}{X_B - X_A} = 1 - \mu_{X_2}(x) \tag{7}$$

$$\mu_{X_2}(x) = \frac{x - X_A}{X_B - X_A} = 1 - \mu_{X_1}(x) \tag{8}$$

$$\mu_{Y_1}(y) = \frac{Y_B - y}{Y_B - Y_A} = 1 - \mu_{Y_2}(y) \quad (9)$$

$$\mu_{Y_2}(y) = \frac{y - Y_A}{Y_B - Y_A} = 1 - \mu_{Y_1}(y) \quad (10)$$

$$\mu_{U_1}(u) = \begin{cases} 1 & u = U_A \\ 0 & otherwise \end{cases} \quad (11)$$

$$\mu_{U_2}(u) = \begin{cases} 1 & u = U_B \\ 0 & otherwise \end{cases} \quad (12)$$

$$\mu_{U_3}(u) = \begin{cases} 1 & u = U_C \\ 0 & otherwise \end{cases} \quad (13)$$

$$\mu_{U_4}(u) = \begin{cases} 1 & u = U_D \\ 0 & otherwise \end{cases} \quad (14)$$

$$(15)$$

The output set applied to the rule defined by X_A and Y_A is obviously U_A, the set applied to the rule defined by X_A and Y_B is U_B, the set applied to the rule defined by X_B and Y_A is U_C and the set applied to the rule defined by X_B and Y_B is U_D.

A product-product implication with defuzzyfication with weight counting gives the following result for the output variable u

$$u = \frac{num}{den} \quad (16)$$

where *num* corresponds to the following expression

$$\begin{aligned} &\mu_{X_1}(x)\,\mu_{Y_1}(y)\,U_A + \\ &\mu_{X_1}(x)\,\mu_{Y_2}(y)\,U_B + \\ &\mu_{X_2}(x)\,\mu_{Y_1}(y)\,U_C + \\ &\mu_{X_2}(x)\,\mu_{Y_2}(y)\,U_D = \\ &(x+y)\,(X_A - X_B)\,(Y_A - Y_B) \end{aligned} \quad (17)$$

and *den* corresponds to the following expression

$$\begin{aligned} &\mu_{X_1}(x)\,\mu_{Y_1}(y) + \\ &\mu_{X_1}(x)\,\mu_{Y_2}(y) + \\ &\mu_{X_2}(x)\,\mu_{Y_1}(y) + \\ &\mu_{X_2}(x)\,\mu_{Y_2}(y) = \\ &(X_A - X_B)\,(Y_A - Y_B) \end{aligned} \quad (18)$$

so finally

$$u = \frac{(x+y)\,(X_A - X_B)\,(Y_A - Y_B)}{(X_A - X_B)\,(Y_A - Y_B)} = x + y \quad (19)$$

References

[1] S. Abe and R. Thawonmas, A fuzzy classifier with ellipsoidal regions, *IEEE Transactions on fuzzy systems*, vol. 5 num. 3 (1997), pp. 358-368.

[2] M. Delgado and F. Gomez-Skarmeta and F. Martin, A fuzzy clustering-based rapid prototyping for fuzzy rule-based modeling, *IEEE Transactions on fuzzy systems*, vol. 5 num. 2 (1997), pp. 223-232.

[3] J. Dickerson and B. Kosko, Fuzzy function approximation with ellipsoidal rules, *IEEE Transactions on systems, man and cybernetics*, vol. 26 num. 4 (1996), pp. 542-560.

[4] C. Garriga, Automatic Process for the synthesis of fyzzt systems from input-output data. *Proceedings of the EUSFLAT-ESTYLF 1999 Joint Conference*, 1999, pp. 143-146.

[5] C.J. Kim, An algorithmic approach for fuzzy inference, *IEEE Transactions on fuzzy systems*, vol. 5 num. 4 (1997), pp. 585-598.

[6] R. Krishnapuram and J.M. Keller, The possibilistic c-means algorithm: insights and recommendations, *IEEE Transactions on fuzzy systems*, vol. 4 num. 3 (1996), pp. 385-395.

[7] Y. Lin and G.A. Cunningham III, A new approach to fuzzy-neural system modeling, *IEEE Transactions on fuzzy systems* vol. 3 num. 2 (1995), pp. 190-197.

[8] H. Nomura and I. Hayashi and N. Wakami, A self-tuning method of fuzzy control by descent method, *Proceedings of the IFSA'91 Brussels*, 1991, pp. 155-158.

[9] N.R. Pal and J.C. Bezdek, On cluster validity for the fuzzy c-means model, *IEEE Transactions on fuzzy systems*, vol. 3 num. 3 (1995), pp. 370-379.

[10] L.X. Wang and J.M. Mendel, Generating fuzzy rules by learning from examples, *IEEE Transactions on systems, man and cybernetics*, vol. 22 num. 6 (1992), pp. 1414-1427.