Fuzzy and Neuro-Fuzzy Systems Report

Francisco Jesus - 2020140671
Pedro Louro - 2020173394

2020-12-11

Part A -Fuzzy Control

Introduction

A fuzzy control system is a control system based on fuzzy logic.

Fuzzy logic analyses analog input values in terms of logical variables that take values between 0 and 1, and these values are continuous.

The inputs are mapped to fuzzy sets that are basically membership functions.

Fuzzyfication corresponds to the process of converting an input into a fuzzy value.

Fuzzy controllers contains three phases: input stage, processing stage and output stage. The input stage corresponds to the phase where the controller maps the inputs to the appropriate membership functions and truth values. In the processing stage it is chosen the appropriate rule for each specific case and generates the results for each one and combines them. In the output stage the result from the processing stage is converted into a specific control output value.

The transfer function used in our project is:

$$\frac{18}{s^3 + 6.5s^2 + 13.5s + 9}$$

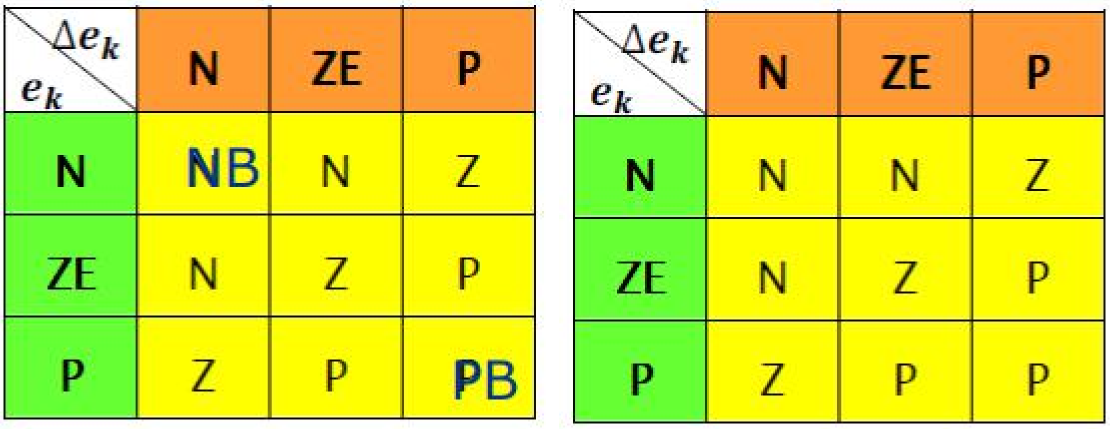
In this project we chose two membership functions. The first one chosen was the triangular function since is the simplest function to use and is somewhat similar to the gaussian function, which corresponds to our second membership function. We chose the second because many things in our world can be represented with gaussian distribution.

Structure of the Fuzzy controllers

We created fuzzy controllers for two types: mandani and sugeno. Each controller has two inputs (error and error variation) and one output (control variation). Each type of controller has controllers with 9, 25 and 49 rules. There is a special controller that uses 9 rules but has 5 possible values for the output.

The defuzzyfication function used in mandani was centroid and in the sugeno was wtaver.

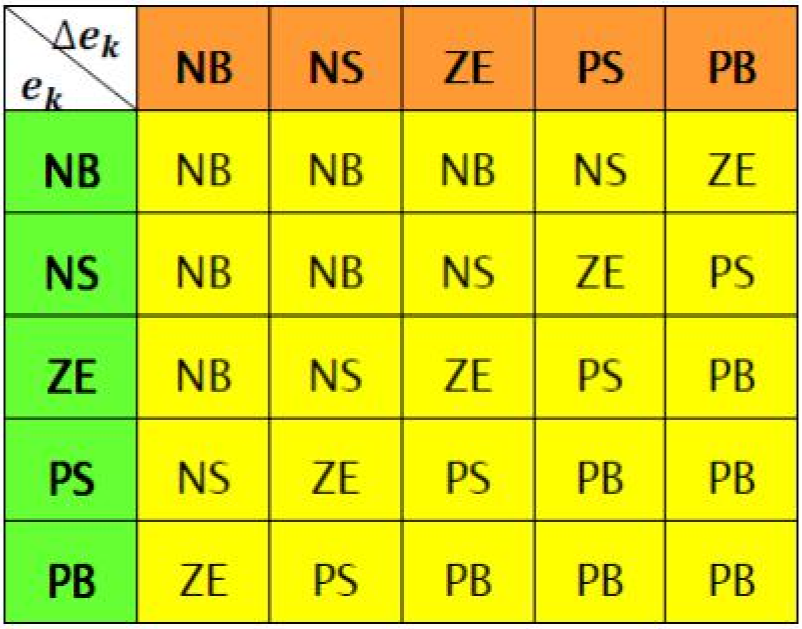
The rules used for each controller can be seen in the following tables (taken from powerpoint shown in class):



e_k \ Δe_k	N	ZE	P
N	NB	N	Z
ZE	N	Z	P
P	Z	P	PB

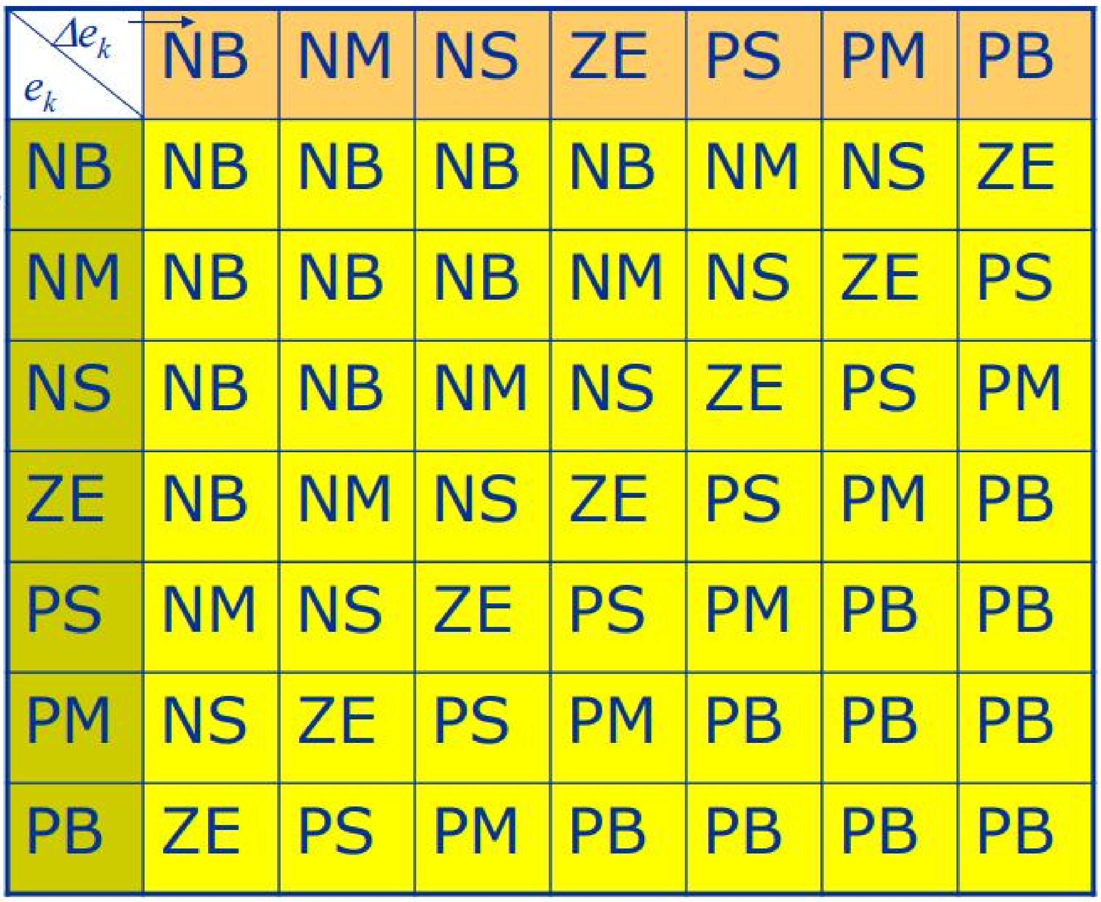
e_k \ Δe_k	N	ZE	P
N	N	N	Z
ZE	N	Z	P
P	Z	P	P

Fig 1 - Tables for 9 rules, the second one is the normal one, the first one is the special one with 5 different values for output (created by ex-students André Fonseca and Fábio Mestre)



e_k \ Δe_k	NB	NS	ZE	PS	PB
NB	NB	NB	NB	NS	ZE
NS	NB	NB	NS	ZE	PS
ZE	NB	NS	ZE	PS	PB
PS	NS	ZE	PS	PB	PB
PB	ZE	PS	PB	PB	PB

Fig 2 - Table for 25 rules



Δe_k / e_k	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NB	NM	NS	ZE
NM	NB	NB	NB	NM	NS	ZE	PS
NS	NB	NB	NM	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PM	PB	PB
PM	NS	ZE	PS	PM	PB	PB	PB
PB	ZE	PS	PM	PB	PB	PB	PB

Fig 3 - Table for 49 rules

Legend: NB - negative big, NM - negative medium, NS - negative small, ZE - zero, PS - positive small, PM - positive medium, PB - positive big

Fuzzy system architecture

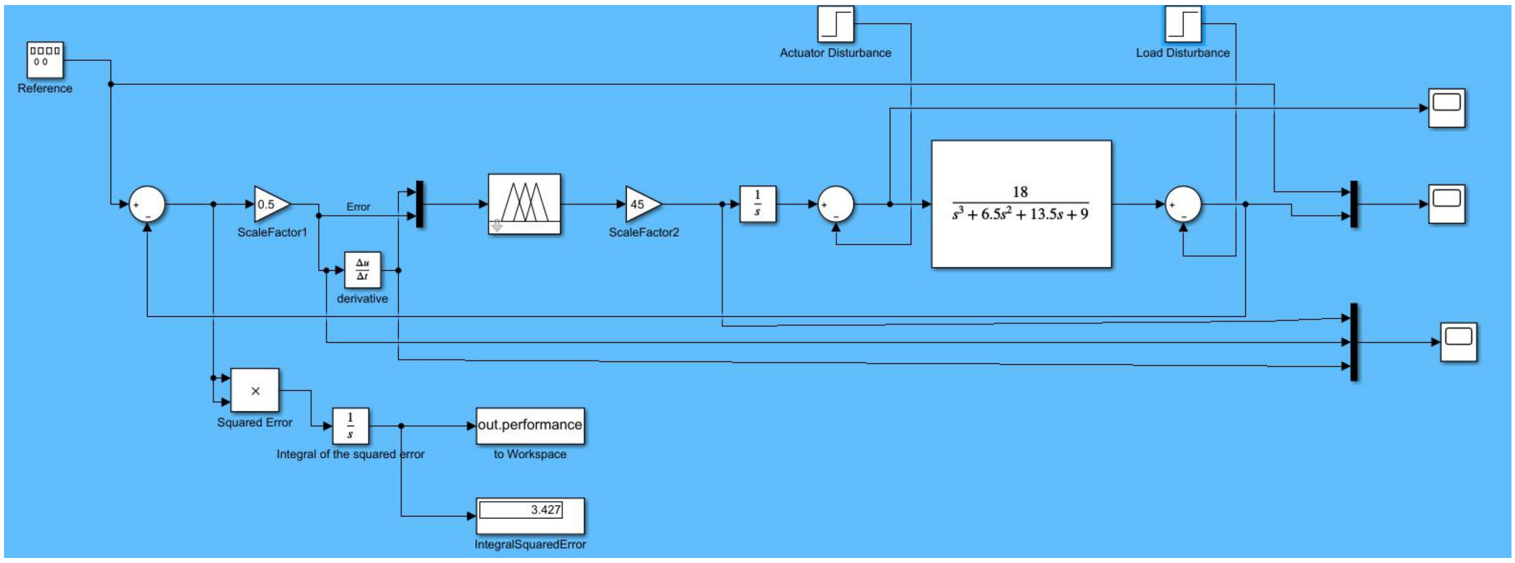


Fig 4 - Fuzzy system architecture

In Fig 4, we can see the block with controller, which corresponds to the controllers created (only one can be used at a time). We have two disturbances that are introduced to the system after a certain amount of time to guarantee that the system is stable before introducing them. The reference signal can take different functions, we used sine, sawtooth and square.

We had of the objective of getting the lowest possible squared error, for that we tracked the value of the squared error and changed the values of each scale factor to get the best results.

The disturbances were activated at t=40 (actuator disturbance) and t=80 (load disturbance). These disturbances can be seen in the graphs that will appear later in the report.

The roots of the transfer function are: there are no zeros, since the numerator is 18 only and the poles are [-3, -2, -1.5].

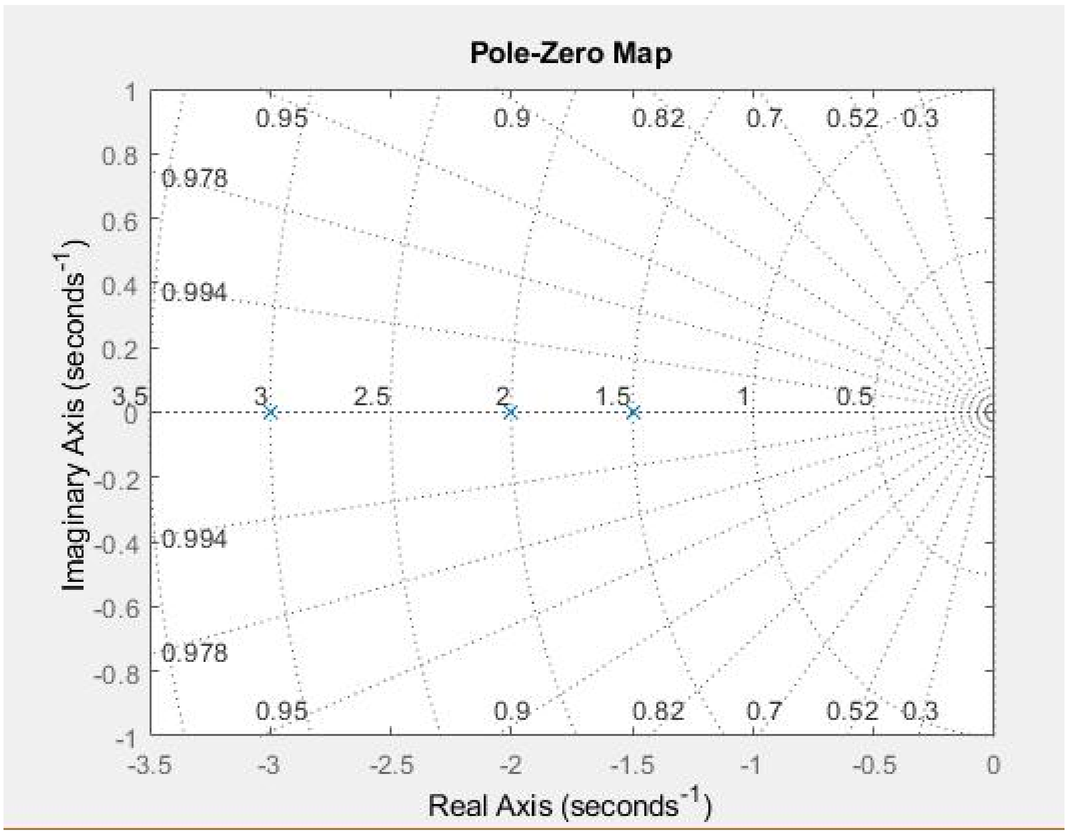


Fig 5 - Pole-zero map of the transfer function

The poles of the transfer function all belong to the stable region of the map, so the system is stable.

Mandani controllers results

Reference - SINE - frequency = 0.5					
Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.5	45	3.561	3.427
	9 special	0.4	10	2.773	4.272
	25	0.2	10	2.82	4.236
	49	0.01	70	2.771	3.947
Gaussian	9	0.7	45	3.271	11.83
	9 special	0.4	10	3.965	4.77
	25	0.2	15	1.848	3.163
	49	0.01	80	1.531	3.194

Reference - SQUARE - frequency = 0.1

Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.2	30	15.96	16.75
	9 special	0.2	10	16.39	17.67
	25	0.1	10	15.51	17.53
	49	0.01	70	15.27	17.25
Gaussian	9	0.2	50	18.27	21.47
	9 special	0.2	25	14.84	16.55
	25	0.2	10	14.86	16.76
	49	0.01	80	15.16	17.98

Reference - SAWTOOTH = frequency = 0.1

Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.3	15	6.833	7.87
	9 special	0.2	10	5.714	8.11
	25	0.1	10	5.722	7.526
	49	0.01	70	5.767	7.452
Gaussian	9	0.2	60	8.305	10.42
	9 special	0.2	25	5.434	7.044
	25	0.2	10	5.343	7.131
	49	0.01	80	5.992	7.9

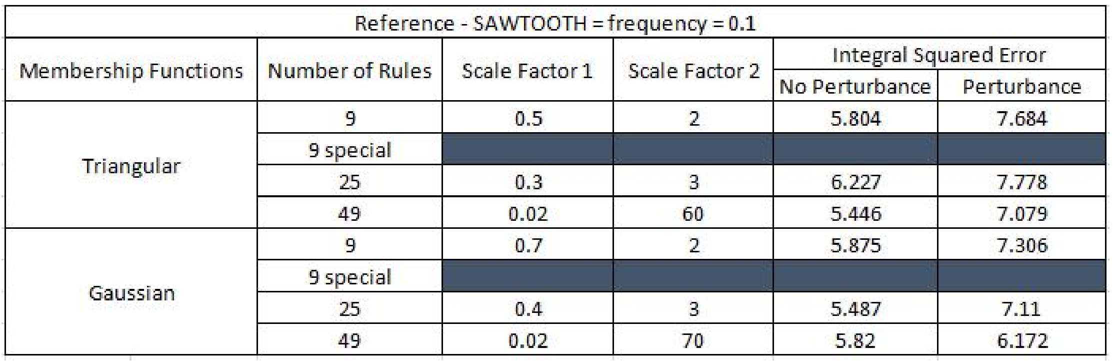
Sugeno controllers results

Reference - SINE - frequency = 0.5

Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.7	2	2.002	3.15
	9 special				
	25	0.3	5	1.757	3.066
	49	0.02	70	1.999	3.199
Gaussian	9	0.7	3	2.403	3.435
	9 special				
	25	0.4	5	2.302	3.498
	49	0.02	120	1.795	3.063

Reference - SQUARE - frequency = 0.1

Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.5	2	16.3	18.65
	9 special				
	25	0.3	3	16.26	18.43
	49	0.02	60	14.37	16.33
Gaussian	9	0.7	2	15.93	17.53
	9 special				
	25	0.4	3	14.79	14.95
	49	0.02	70	15.43	17.43



Reference - SAWTOOTH = frequency = 0.1					
Membership Functions	Number of Rules	Scale Factor 1	Scale Factor 2	Integral Squared Error	
				No Perturbance	Perturbance
Triangular	9	0.5	2	5.804	7.684
	9 special	▨	▨	▨	▨
	25	0.3	3	6.227	7.778
	49	0.02	60	5.446	7.079
Gaussian	9	0.7	2	5.875	7.306
	9 special	▨	▨	▨	▨
	25	0.4	3	5.487	7.11
	49	0.02	70	5.82	6.172

Graphs of some results from the controllers

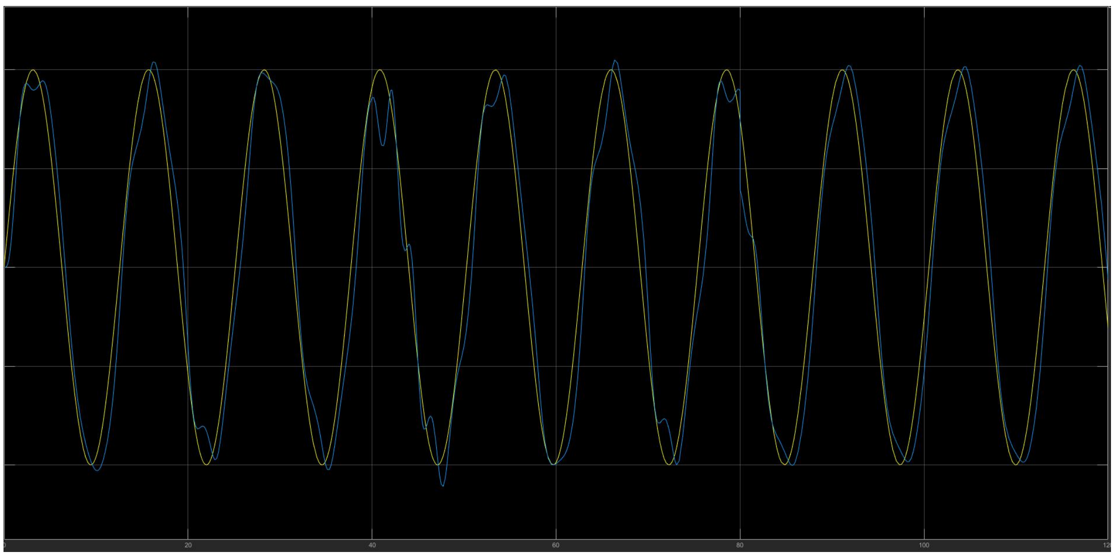


Fig 6 - Performance of the Fuzzy Controller Mandani for a Sinusoidal Reference - 9 rules triangular MSF.

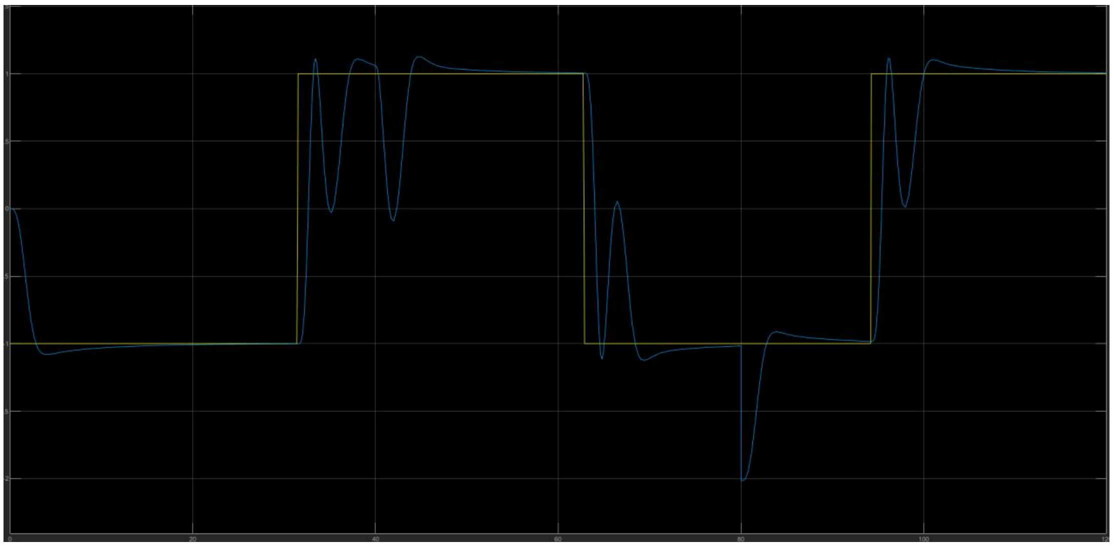


Fig 7 - Performance of the Fuzzy Controller Mandani for a Square Reference - 9 rules triangular MSF.

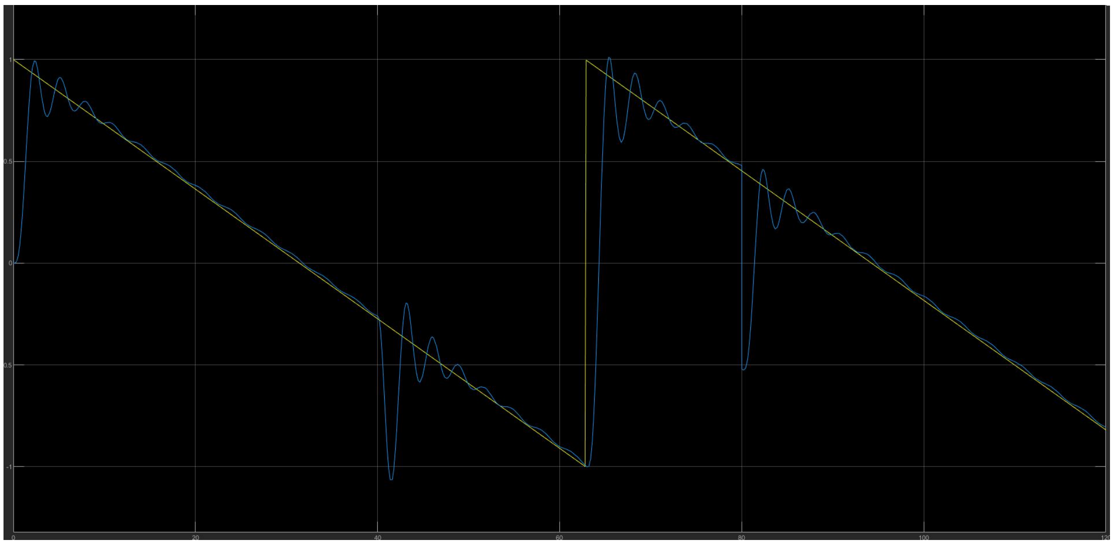


Fig 8 - Performance of the Fuzzy Controller Mandani for a Sawtooth Reference - 25 rules triangular MSF.

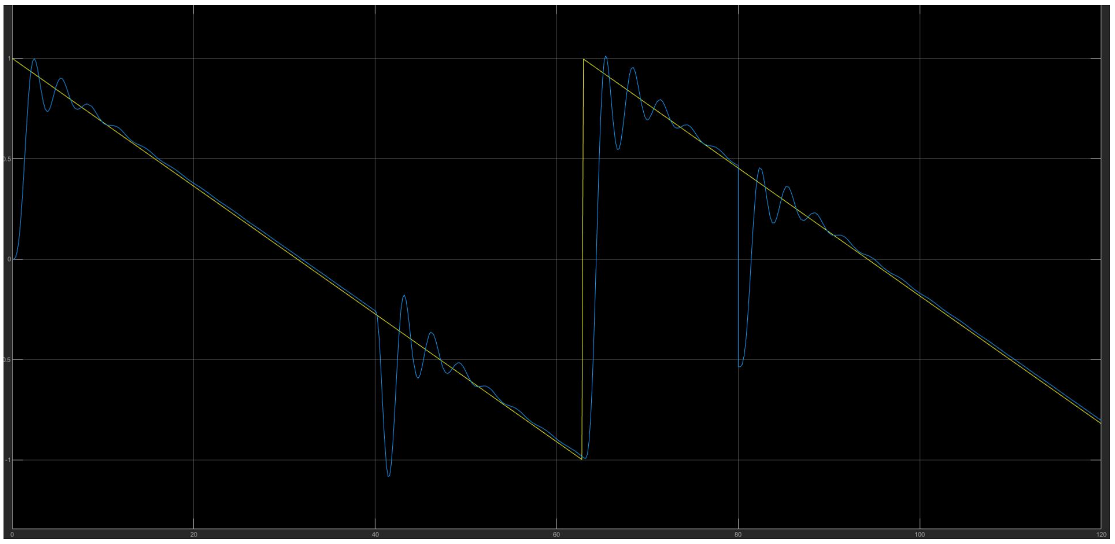


Fig 9 - Performance of the Fuzzy Controller Sugeno for a Sawtooth Reference - 9 rules triangular MSF.

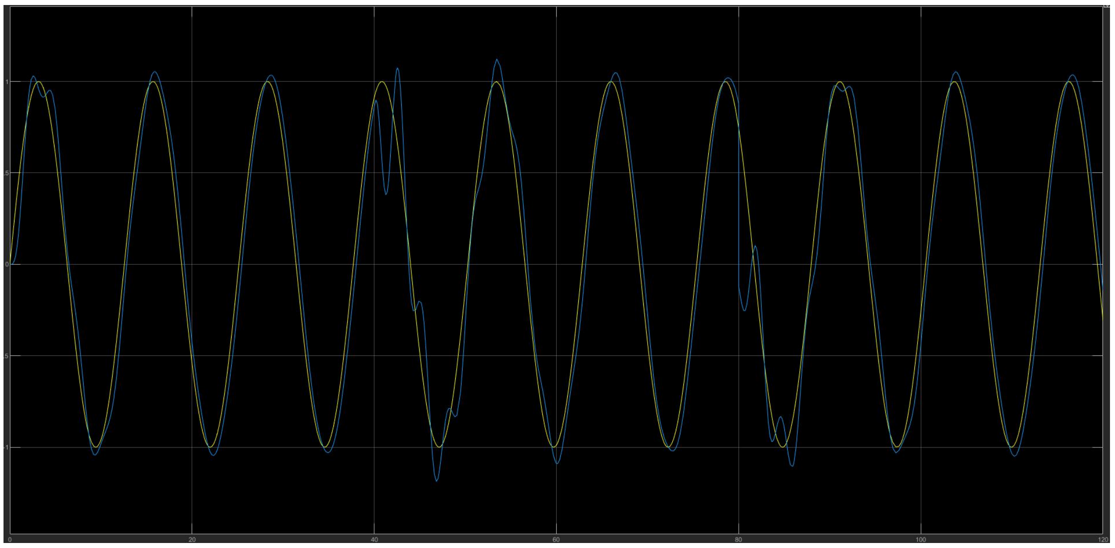


Fig 10 - Performance of the Fuzzy Controller Sugeno for a Sinusoidal Reference - 25 rules triangular MSF.

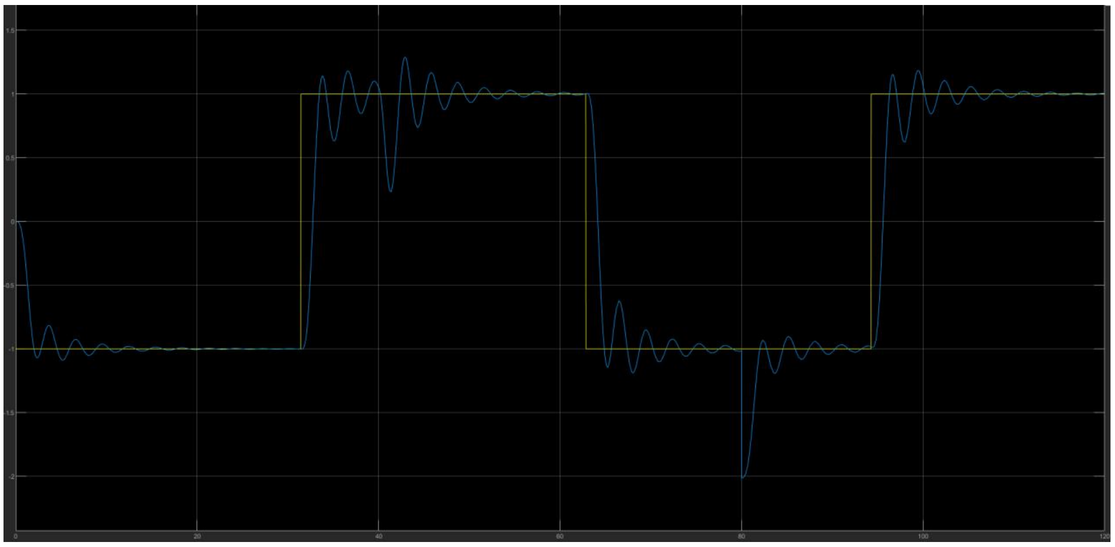


Fig 11 - Performance of the Fuzzy Controller Sugeno for a Square Reference - 49 rules triangular MSF.

Conclusion

From what we can see from the tables, we can conclude that more rules achieve a better performance, the system gets more stable. But there is also an interesting thing that we can see, is that when we use more rules in the case of 49 rules we will not get better results by a large difference, and in some cases we even get worst results. So it is necessary to study the situation where we want to use a certain fuzzy system and use a number of rules only necessary for that case.

Other interesting situation is the fact that the system with 9 rules with a different output achieves better results then the system with 9 rules with an output with 3 values.

We can also see that the worst results were found using the square signal, that is probably due to the fact that it has fast variations which makes the system have difficulties in stabilizing. The same can be said with the sawtooth. The sinusoidal signal is easier to stabilize since its variations are softer in comparison to the other two.

We also can deduce that using gaussian instead of triangular offers better results overall.

We need to keep in mind that these results were obtained using different scale factors which makes it hard to really deduce which system is really the best.

Part B - NEURO-FUZZY SYSTEMS FOR MODELLING DYNAMIC PROCESSES

Introduction

Like the name implies, neuro-fuzzy systems are a combination of neural networks and fuzzy logic. It combines the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks. They even use IF-THEN rules.

In this part of the project we created a dynamical system using our transfer function defined in the previous part. This system contains memory, this means that it uses previous outputs to calculate a certain output, this means that the system has inertia.

Defining the data set

To create our data set, we had, first, to transform our transfer function in a Discrete Transfer Function, basically we transform our function so that it has the same amount of variables in the numerators and in the denominators. Since the transfer functions has a higher order in the denominator we can assume that the function has inertia.

Since the function has a order 3 in the denominator we can say that our data matrix will have 7 columns, where 6 of them are assigned to antecedents and 1 to the desired time series

We divided the data set in 70% for training and 30% for testing.

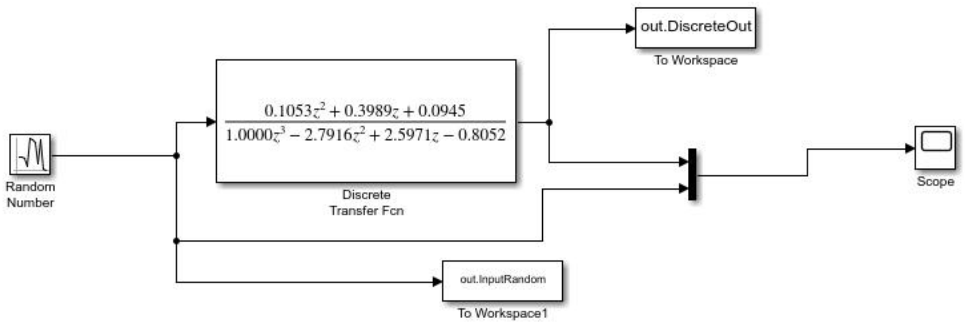


Fig 12 - Simulink Diagram used to Generate values of Data Matrix

In fig 12 we can see the Discrete Transfer Function obtained from our transfer function.

Defining the system

We used three clustering techniques: subtractive, gird-partition and fuzzy c-means. For optimization methods, we used retropropagation and hybrid, using ANFIS to make the optimization.

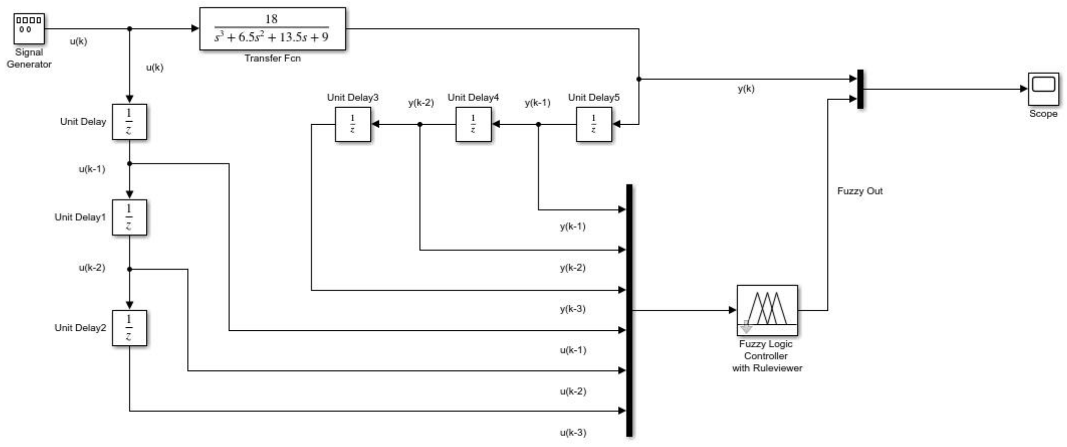


Fig 13 - Diagram of the Simulink Model for the Performance Evaluation of the different FIS.

Obtained Rules from the Created Fuzzy Inference System

The bellow figures show the obtained rules from the 3 methods of clustering presented above, each one optimized with the 2 methods presented above

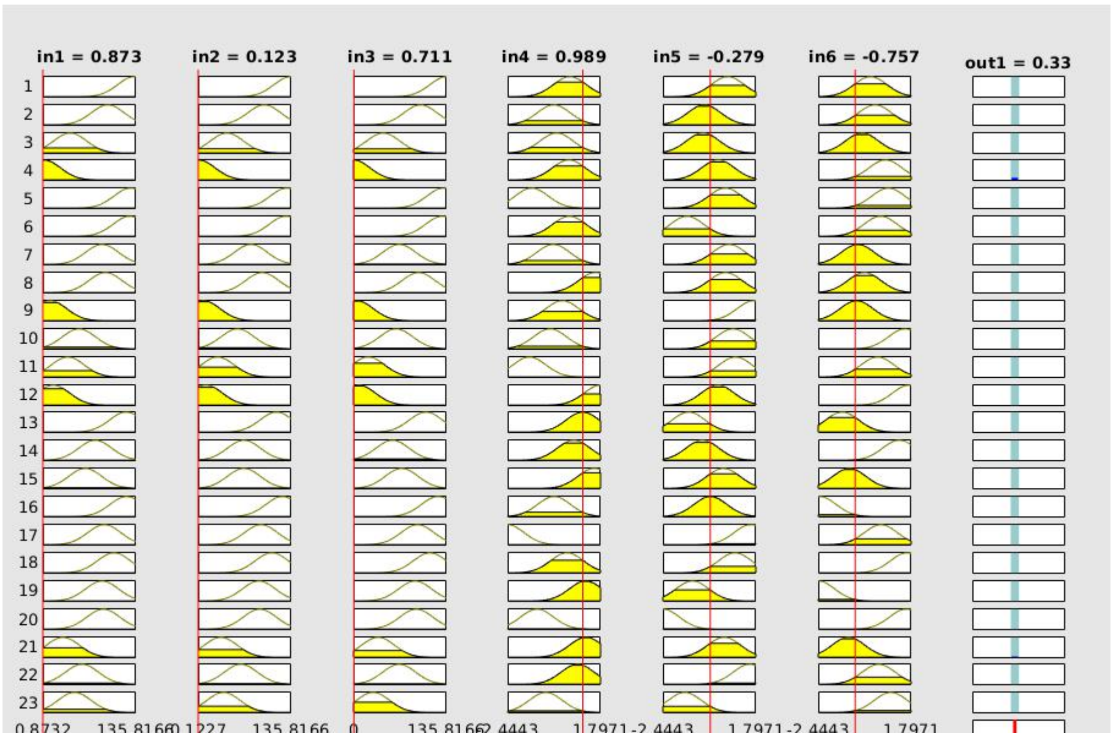


Fig 1 - Rules obtained from the FIZ using Grid Partitions optimized with Retropropagation

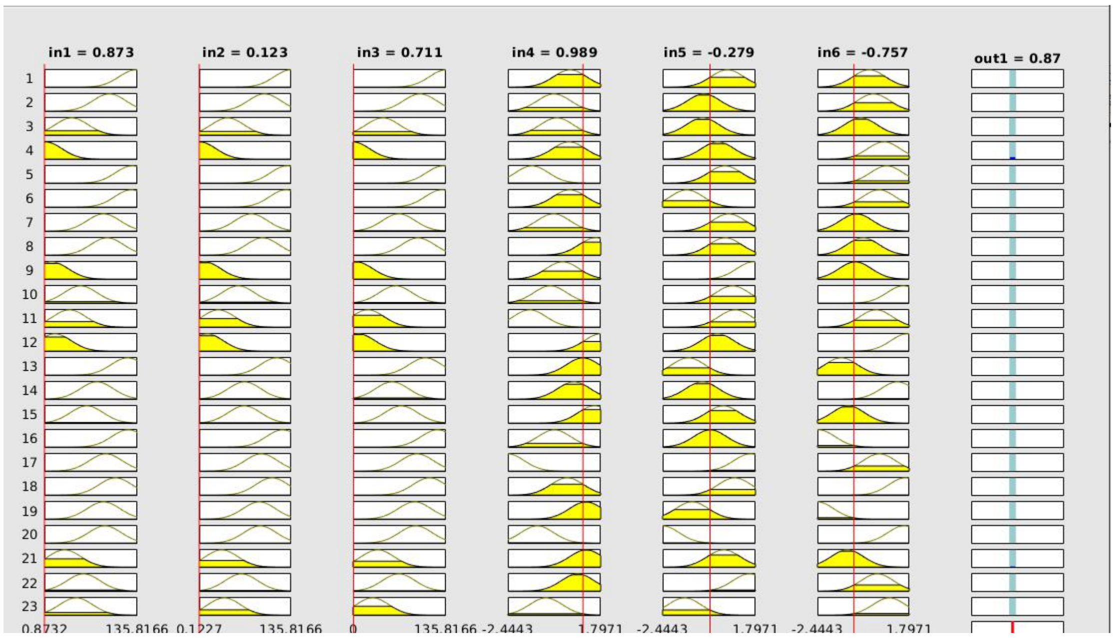


Fig 2 - Rules obtained from the FIZ using Grid Partitions optimized with the Hybrid

method

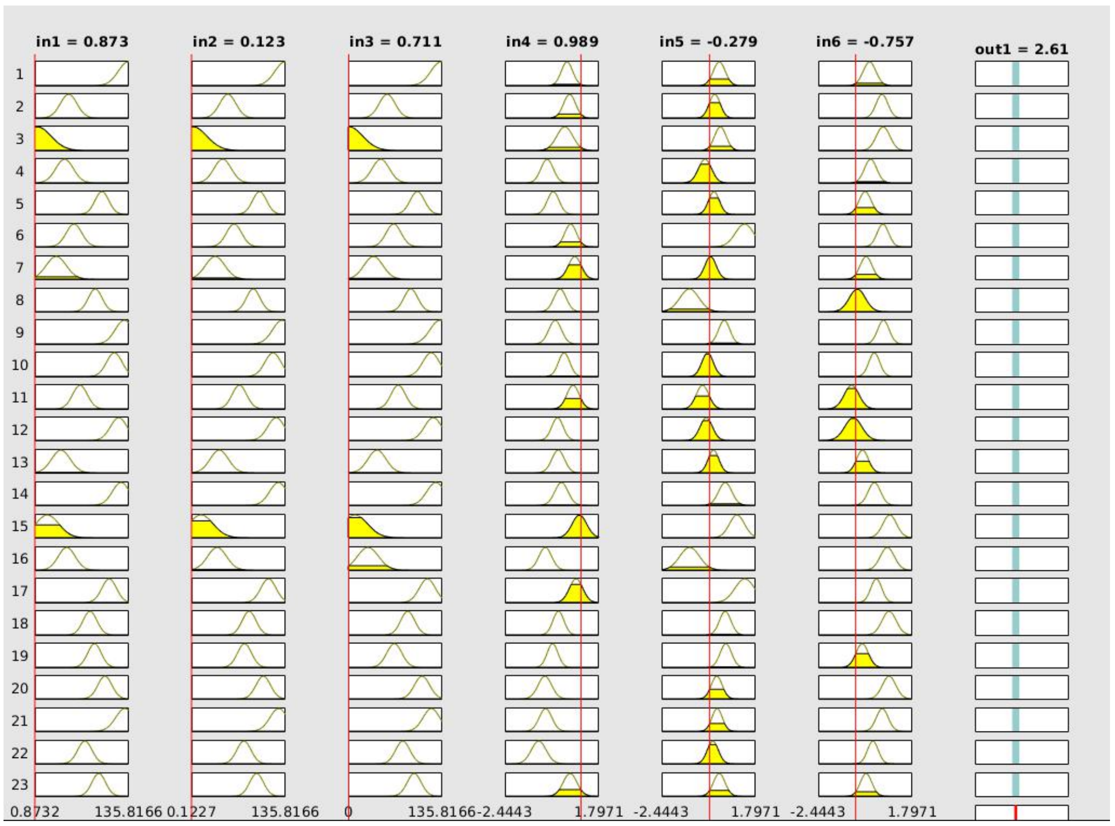


Fig 3 - Rules obtained from the FIZ using Subtractive optimized with Retropropagation

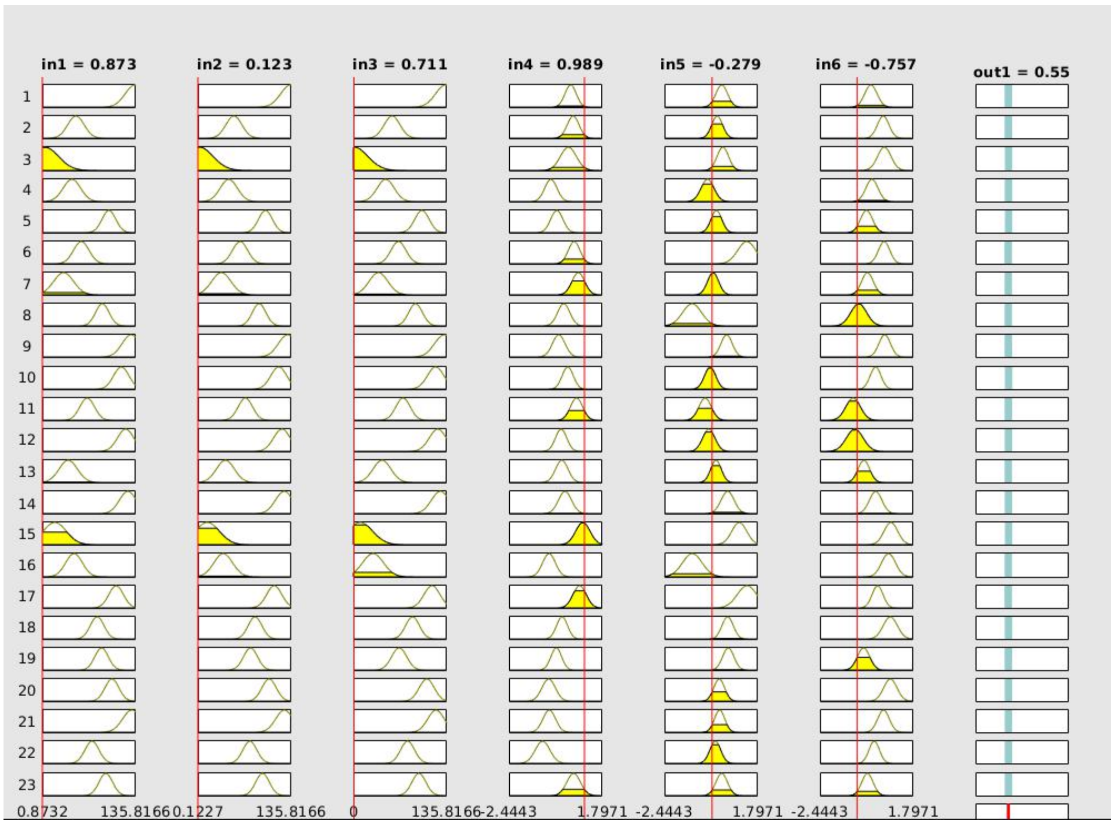


Fig 4 - Rules obtained from the FIZ using Subtractive optimized with the Hybrid method

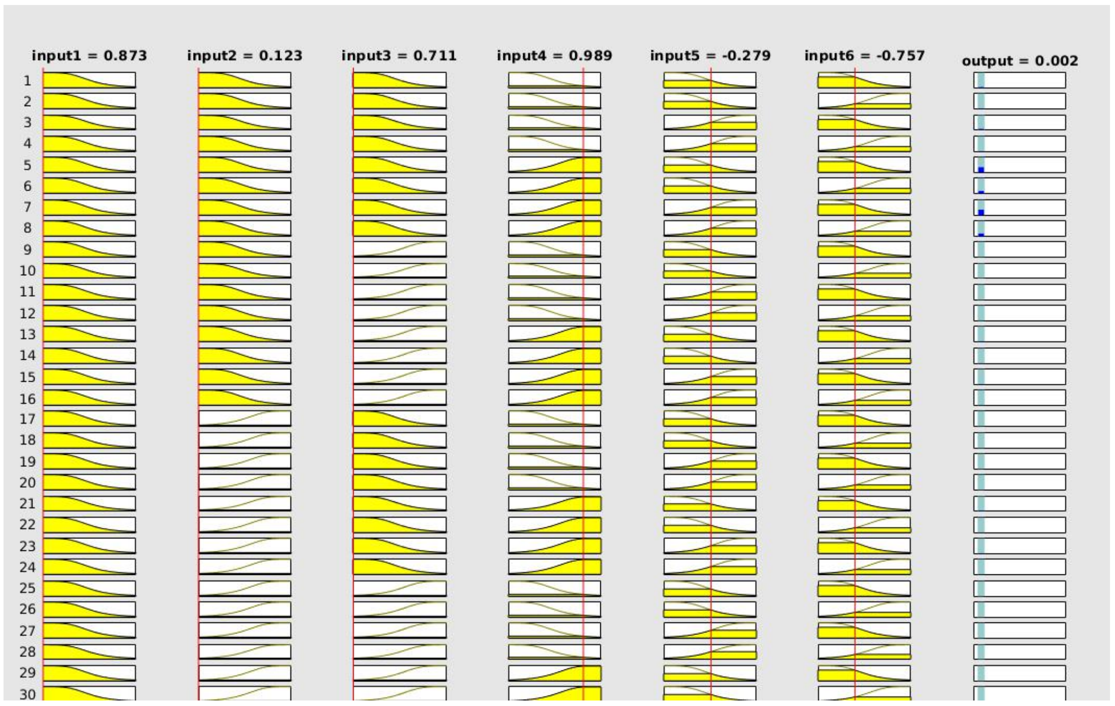


Fig 5 - Rules obtained from the FIZ using Fuzzy C-Means optimized with Retropropagation

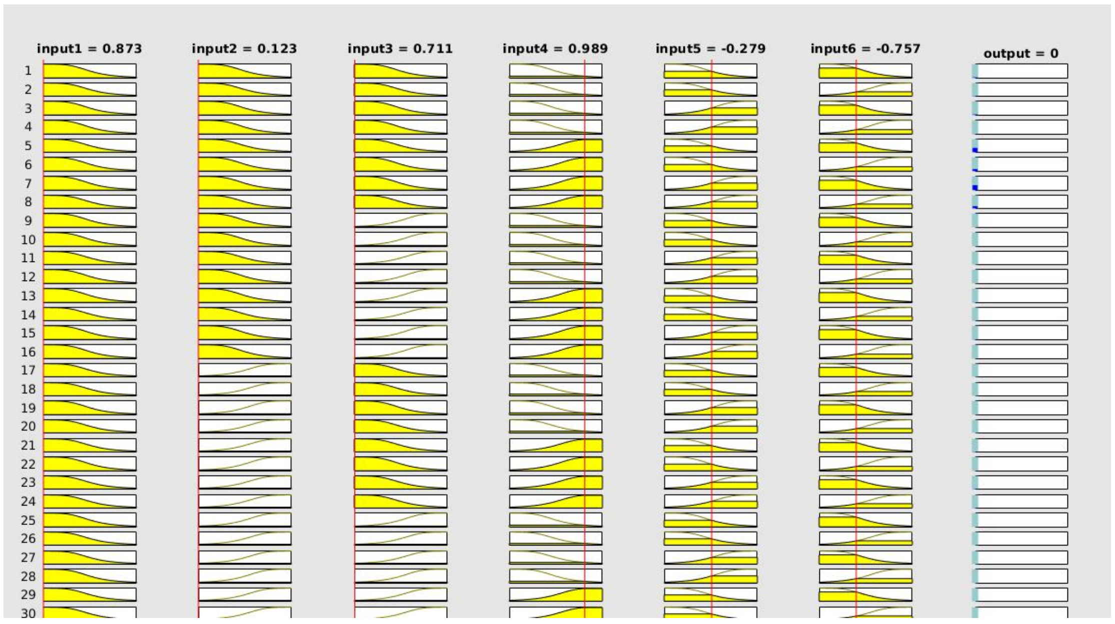


Fig 6 - Rules obtained from the FIZ using Fuzzy C-Means optimized with the Hybrid method

Obtained Performance from the Methods Used

The given transfer function produced comparatively good results with the Grid Partition and Subtractive method

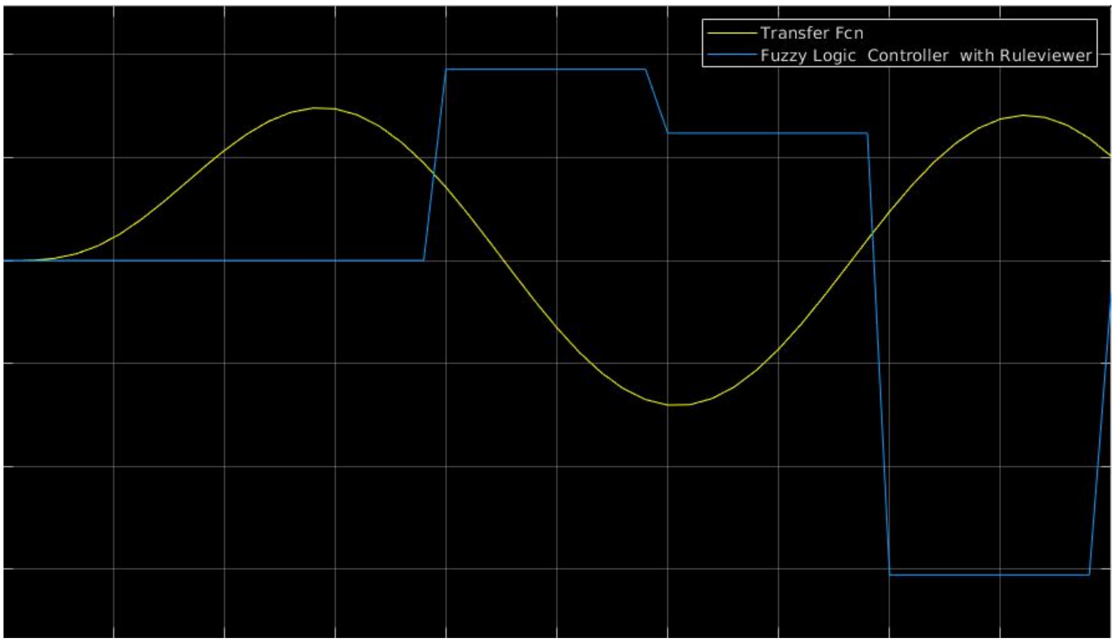


Fig 7 - Performance of the Grid Partition algorithm optimized with Retropropagation

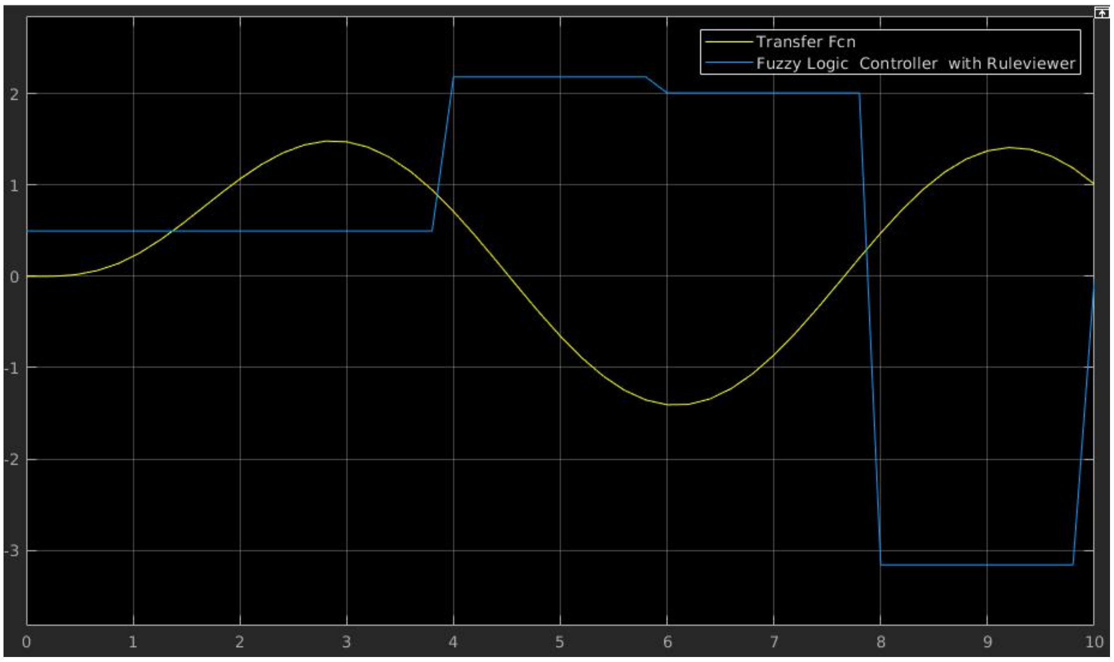


Fig 8 - Performance of the Grid Partition algorithm optimized with the Hybrid method

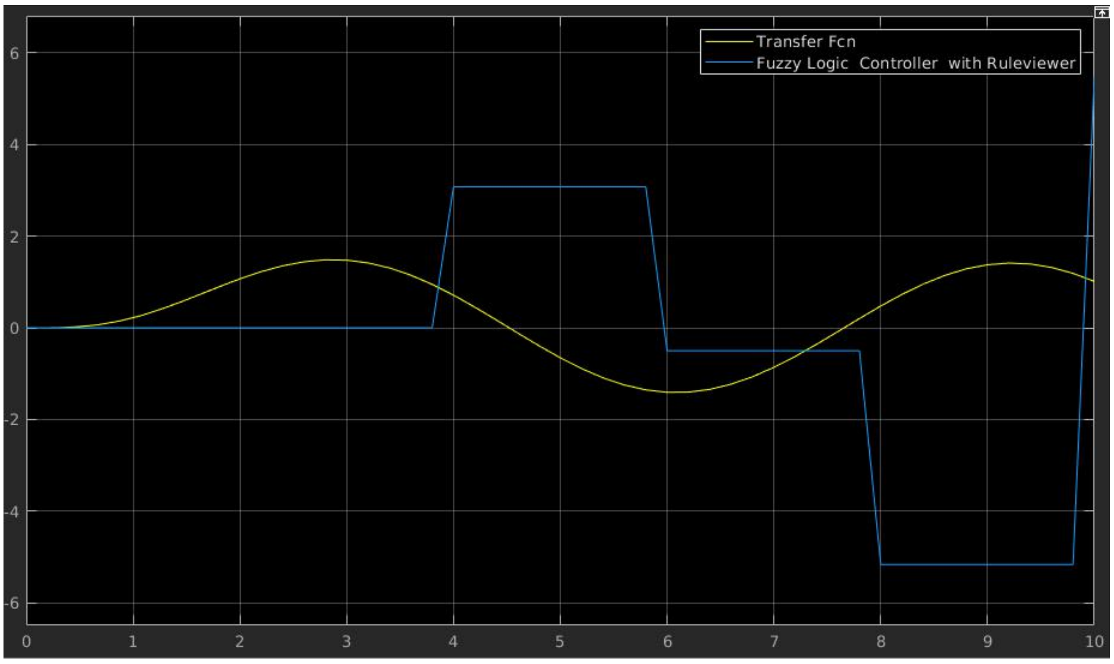


Fig 10 - Performance of the Subtractive algorithm optimized with Retropropagation

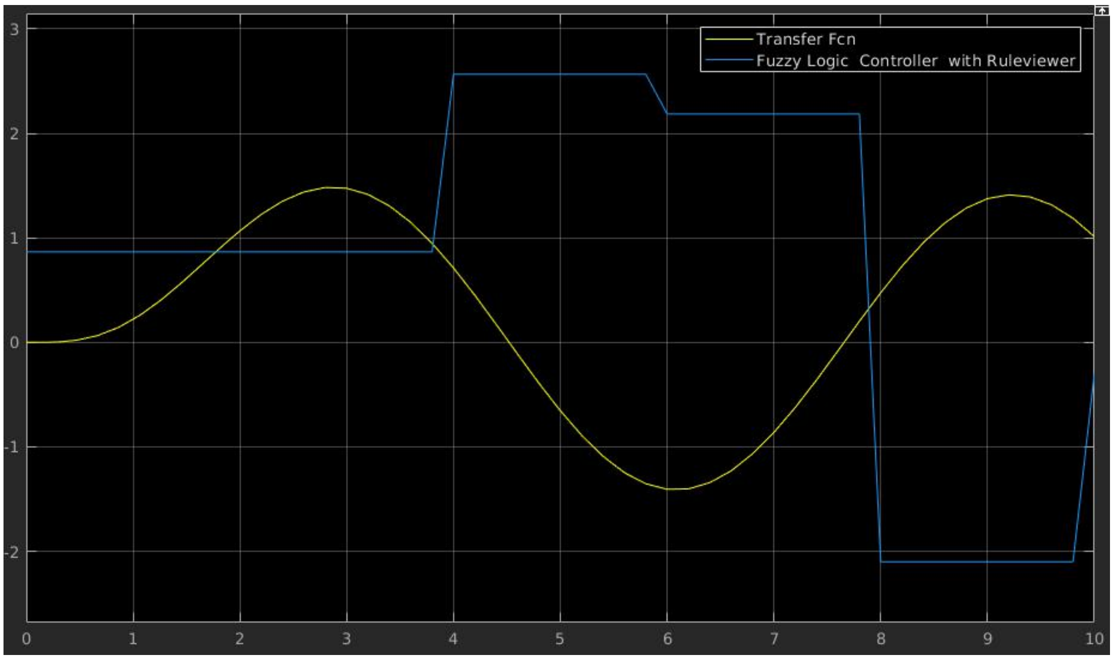


Fig 11 - Performance of the Subtractive algorithm optimized with the Hybrid method

All of the above where shown with a sine reference since none of them match the reference or the other reference possibilities (sawtooth and square)

These were comparatively good, since the performance of the Fuzzy C-Means algorithm with both methods produce the following results

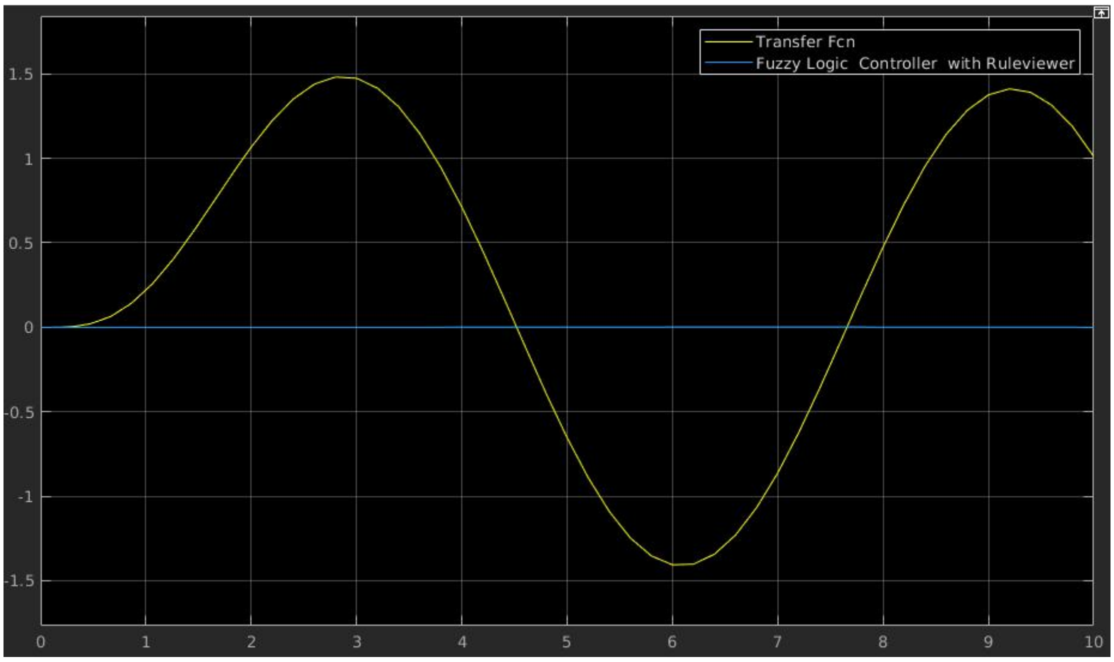
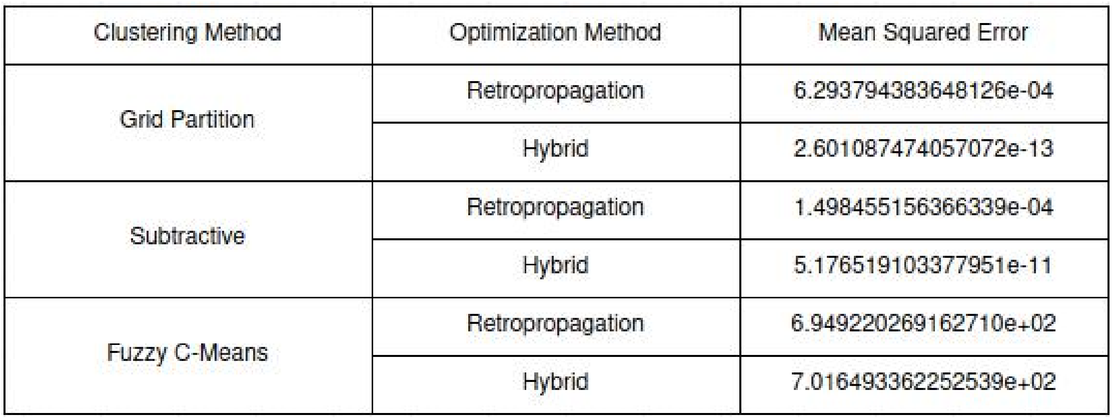


Fig 12 - Performance of the Fuzzy C-Means algorithm optimized with Retropropagation

(the graph depicting these algorithm optimized with the Hybrid method are the same)

Mean Square Error Results

These results were obtained by calculating the means square error between the results of the created Fuzzy Inference Systems with the created testing predictors dataset and the created testing target dataset



Clustering Method	Optimization Method	Mean Squared Error
Grid Partition	Retropropagation	6.293794383648126e-04
	Hybrid	2.601087474057072e-13
Subtractive	Retropropagation	1.498455156366339e-04
	Hybrid	5.176519103377951e-11
Fuzzy C-Means	Retropropagation	6.949220269162710e+02
	Hybrid	7.016493362252539e+02

Conclusion

Based on the performance results and the calculated mean square error, we can conclude that for the given transfer function, the best performance was obtained trough the Grid Partition and Subtractive clustering algorithms.

We can also conclude that the best optimization method is the Hybrid method.