


**Approximate Reasoning using Fuzzy Set Theory**  
**Prof. Balasubramaniam Jayaram**  
**Department of Mathematics**  
**Indian Institute of Technology, Hyderabad**

**Lecture - 35**  
**SBR: Mamdani Fuzzy Systems**

Hello and welcome to the 3rd of the lectures, in this week 7 of this course titled Approximate Reasoning using Fuzzy Set Theory. A course offered over the NPTEL platform. In this week of lectures we have been discussing similarity based reason.

(Refer Slide Time: 00:36)




**A quick recap ...**

- An FIS approximates a given system function ...
- ... by covering it through overlapping rule patches.
- Similarity Based Reasoning - The operations.

**Outline of this lecture**

- Mamdani Fuzzy System.
- Generalisation of a Matching Function.
- Singleton Fuzzification and Matching Functions.
- Building a Mamdani FIS using Matlab.



Balasubramaniam Jayaram    ARFST - SBR : Mamdani Fuzzy Systems

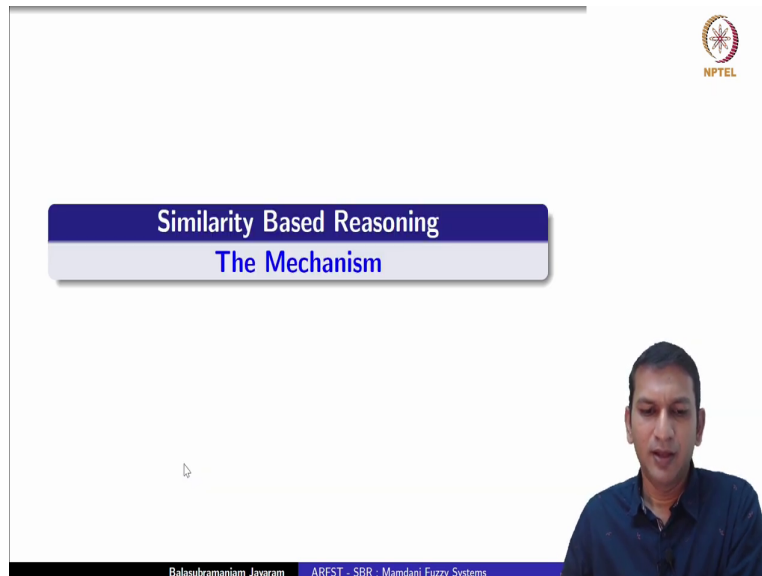
And we have seen that a fuzzy inference system approximates a given system function by covering it through overlapping rule patches. This is the most important takeaway when you want to visualize how a fuzzy inference system captures the working of a system.

We have of course, through the previous two lectures seen the operations involved in similarity based reason. In this lecture we will discuss Mamdani fuzzy system, before going towards seeing how it is implemented in the MATLAB fuzzy logic toolbox and using it to build Mamdani systems.

We need two preliminary things to look at matching function in a slight generalization and also to understand how singleton fuzzification works with such class of matching functions. Then we will go on to build one particular function through a Mamdani fuzzy inference

system that is approximating that function by building a Mamdani fuzzy inference system using the fuzzy logic toolbox in MATLAB.

(Refer Slide Time: 01:55)



NPTEL

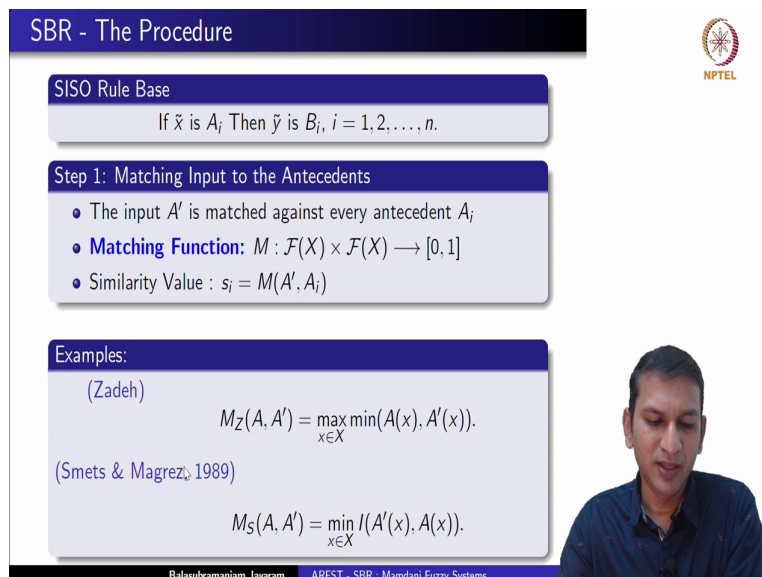
## Similarity Based Reasoning

### The Mechanism

Balasuubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

Well a quick recap of the similarity based reasoning the different steps involved.

(Refer Slide Time: 02:01)



NPTEL

## SBR - The Procedure

### SISO Rule Base

If  $\tilde{x}$  is  $A_i$  Then  $\tilde{y}$  is  $B_i$ ,  $i = 1, 2, \dots, n$ .

### Step 1: Matching Input to the Antecedents

- The input  $A'$  is matched against every antecedent  $A_i$
- Matching Function:**  $M : \mathcal{F}(X) \times \mathcal{F}(X) \rightarrow [0, 1]$
- Similarity Value :  $s_i = M(A', A_i)$

### Examples:

(Zadeh)

$$M_Z(A, A') = \max_{x \in X} \min(A(x), A'(x)).$$

(Smets & Magrez, 1989)

$$M_S(A, A') = \min_{x \in X} \max(A'(x), A(x)).$$

Balasuubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

We are given a set of single input single output rules. The first step is given an input  $A'$  we want to match it with each of the antecedents. Now, for this you are given an  $A'$  you will match it against each of the antecedents  $A_1, A_2$ . So, on to a what we need is a matching

function for this. So, essentially the matching function takes two fuzzy sets defined over  $x$  and throws out a similarity value which is typically in the  $[0,1]$  interval. So, you could look at it as a normalized matching function.

So,  $M$  is from  $F(X)$  cross  $F(X)$  to  $[0,1]$  given an  $A$  dash the similarity value of  $A$  dash to  $A_i$  is what is captured by this matching function in the value  $s_i$ . We have seen a couple of matching functions in the previous lecture, we will have occasion to see these presently too. So, that is the first step. So, given an  $A$  dash you have matched it against all the antecedents and we have got similarity values.

(Refer Slide Time: 03:11)

### SBR - The Procedure

Step 2: Modifying the Consequents

- Modify each  $B_i$  with the similarity value  $s_i$
- **Modification Function:**  $J : [0,1] \times \mathcal{F}(Y) \rightarrow \mathcal{F}(Y)$
- $B'_i = J(s_i, B_i)$ , i.e.,  $B'_i(y) = J(s_i, B_i(y))$ ,  $y \in Y$ .
- In essence,  $J : [0,1] \times [0,1] \rightarrow [0,1]$ .


Examples:


(Cross & Sudkamp, 1993)

$$J_{ML}(s, B) = B'(x) = \min\{1, B(x)/s\}, x \in X.$$

(Morsi & Fahmy, 2002)

$$J_{MVR}(s, B) = B'(x) = s \cdot B(x), x \in X.$$






Balasubramaniam Jayaram
ARFST - SBR - Mamdani Fuzzy Systems

The next step is to modify using this similarity value of the corresponding rule, modify it modify the consequent of the corresponding rule. So,  $s_1$  is the similarity value between  $A$  dash and  $A_1$  we use this  $s_1$  to modify  $B_1$  to get a  $B_1$  dash similarly  $A$  dash and  $A_i$  you get a similarity value of  $s_i$  using a matching function.

Use this  $s_i$  to modify  $B_i$  to get a  $B_i$  dash. These are some of the matching functions that we have seen. In fact, we have seen that these are essentially fuzzy logic connectives. So, what you see here this is a  $t$  norm and this we know is the Gogon implication rule.

(Refer Slide Time: 03:56)



**SBR - The Procedure**

**Step 3: Aggregating the Modified Consequents**

- Aggregate all of the  $B'_i$ 's.
- **Aggregation:**  $G : \mathcal{F}(Y) \times \mathcal{F}(Y) \rightarrow \mathcal{F}(Y)$ .
- $G(B'_i, B'_j)(y) = G(B'_i(y), B'_j(y))$ ,  $y \in Y$ .
- So, again,  $G : [0, 1] \times [0, 1] \rightarrow [0, 1]$  and **associative**.


**Step 3+: Defuzzification**

- The final output  $B' \in \mathcal{F}(Y)$  is defuzzified to  $y \in Y$ .
- Centroid, Mean of Maxima, etc.
- $g : \mathcal{F}(Y) \rightarrow Y$  is any **defuzzifier**.

**Step 1-: Fuzzification**

- Input  $x \in X$  is fuzzified to  $A' \in \mathcal{F}(X)$ .
- Singleton, Gaussian, Triangular, etc.
- $h : X \rightarrow \mathcal{F}(X)$  is any **fuzzifier**.


Balazsbramiam Jayaram ARFST - SBR - Mamdani Fuzzy Systems



In the third step we aggregate all of these  $B_1$  dash  $B_2$  dash so on and so forth till  $B_n$  dash, these are the modified consequence. We aggregate all of them. Once again we have seen that it is sufficient to consider associative fuzzy logic connectives, typically t norms or u norms are used for this. Finally, there is also another step you could defuzzify this; that means, the obtained output set is a fuzzy set  $B$  dash on  $Y$ . Now, we would typically need a value in the domain of  $Y$ . So, we apply a defuzzifier and get a value  $y$  in the domain  $Y$ .


Some of the defuzzifier are the centroid mean of maxima and so on. We will see these in little more detail, visually when we implement using MATLAB. There is also another step that is there, that is typically if you are given only a value and not a fuzzy set then we do the reverse of defuzzification which is called the fuzzification. We have seen single what is to how to obtain a singleton fuzzifier fuzzified value of a given input. Also we have seen how to obtain triangular fuzzification or Gaussian fuzzification.

(Refer Slide Time: 05:24)



## Matching Function


### A Generalisation



Balazubramaniam Jayaram ARFST - SBR : Mamdani Fuzzy Systems

Well, let us look at matching function a little deep.

(Refer Slide Time: 05:29)



## SBR - The Procedure

### SISO Rule Base

If  $\tilde{x}$  is  $A_i$  Then  $\tilde{y}$  is  $B_i$ ,  $i = 1, 2, \dots, n$ .

### Step 1: Matching Input to the Antecedents


- The input  $A'$  is matched against every antecedent  $A_i$
- Matching Function:**  $M : \mathcal{F}(X) \times \mathcal{F}(X) \rightarrow [0, 1]$
- Similarity Value :  $s_i = M(A', A_i)$

### Examples:

(Zadeh)

$$M_Z(A, A') = \max_{x \in X} \min(A(x), A'(x)).$$

(Smets & Magrez, 1989)

$$M_S(A, A') = \min_{x \in X} I(A'(x), A(x)).$$


Balazubramaniam Jayaram ARFST - SBR : Mamdani Fuzzy Systems

Matching function essentially is a function from  $\mathcal{F}(X)$  cross  $\mathcal{F}(X)$  to  $[0, 1]$ . Now look at these two examples these are some of the often used matching functions.

(Refer Slide Time: 05:43)

The Matching Function

Examples:

(Zadeh)

$$M_Z(A, A') = \max_{x \in X} \min(A(x), A'(x)).$$

(Smets & Magrez, 1989)

$$M_S(A, A') = \min_{x \in X} I(A'(x), A(x)).$$

$M_{\oplus, \otimes} : \mathcal{F}(X) \times \mathcal{F}(X) \rightarrow [0, 1]$

$$M_{\oplus, \otimes}(A, A') = \bigoplus_{x \in X} \left( A(x) \otimes A'(x) \right).$$

Balazubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

If you look at them closely what you have here is actually a function outside and a function inside. So, essentially it is more like a composition. In fact, the Zadeh's maximum matching function is essentially the max min composition of two vectors. So, when you look at it like this, it is in some sense generalization of composition of two vectors. So, you need two operations here and that is how we can look at them.

So, in the case of Zadeh's matching function the o times operator is actually min and the operation that is outside is max. In the case of Smets Magrez matching function, the operation inside is an implication and the operation outside is a is the minimum d u well. Typically these are major class of matching function that are considered.

(Refer Slide Time: 06:42)




### Matching Function & Singleton Fuzzification



Balashubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

Now, we let us look at how singleton fuzzification goes with this class of matching functions.

(Refer Slide Time: 06:49)



### Singleton Fuzzification and the Matching Function $M$

Examples:

(Zadeh)


$$M_Z(A, A') = \max_{x \in X} \min(A(x), A'(x)).$$

(Smets & Magrez, 1989)

$$M_S(A, A') = \min_{x \in X} I(A'(x), A(x)).$$

#### Singleton Fuzzification

- Fix  $x_0 \in X$ .
- $A'_{x_0} : X \rightarrow [0, 1]$ .

$$A'_{x_0}(x) = \begin{cases} 1, & \text{if } x = x_0 \\ 0, & \text{if } x \neq x_0 \end{cases}$$


Balashubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

Let us consider these two examples and a singleton fuzzifier. So, given an input  $x$  naught what we do is we create an  $A$  dash to be given as input to this fuzzy inference system which is a fuzzy set on  $X$  to on  $X$ . So,  $A$  dash of  $A$  dash at  $x$  naught with respect to  $x$  naught is defined like this it takes the value one at  $x$  is equal to  $x$  naught and 0 elsewhere. So, in some sense it is the characteristic function of the singleton classical singleton set  $x$  naught.


(Refer Slide Time: 07:27)


### Singleton Fuzzification and the Matching Function $M$

$$M_Z(A, A') = \max_{x \in X} \min(A(x), A'(x)).$$

$$A'_{x_0}(x) = \begin{cases} 1, & \text{if } x = x_0, \\ 0, & \text{if } x \neq x_0. \end{cases}$$

$$\begin{aligned} M_Z(A, A'_{x_0}) &= \max_{x \in X} \min(A(x), A'_{x_0}(x)), \\ &= \left\{ \max_{x \neq x_0} \min(A(x), A'_{x_0}(x)) \right\} \vee \min(A(x_0), A'_{x_0}(x_0)), \\ &= \left\{ \max_{x \neq x_0} \min(A(x), 0) \right\} \vee \min(A(x_0), 1) \\ &= 0 \vee A(x_0) = A(x_0). \end{aligned}$$





Balasubramaniam Jayaram
ARFST - SBR - Mamdani Fuzzy Systems

Now, consider the Zadeh's matching function and let us assume that we are giving it an input which is a singleton fuzzified input. Let us look at what happens to the similarity between  $A$  and such an  $A$  dash. So,  $M_Z$  of  $A$   $A$  dash of  $x$  naught, if you were to write it in terms of formula we substitute for  $A$  dash we substitute  $A$  dash  $x$  naught.

Now, this can be split into two parts. The first part is consider all those  $x$  in  $X$ , which are not equal to  $x$  naught and separate out when  $x$  is equal to  $x$  naught. Now, look at this region when  $x$  is not equal to  $x$  naught, what happens to  $A$  dash  $x$  naught of  $x$  it is essentially 0 when  $x$  is not equal to  $x$  naught then it is essentially 0. So, what we are looking at is this part  $A$  dash of  $x$  naught of  $x$  is actually 0.

Now, what happens to  $A$  dash  $x$  naught at  $x$  naught by this definition it is 1. Now, minimum of anything comma 0 is 0 because we know 0 is the annihilated for the  $t$  naught especially in the case of minimum. So, no matter what  $x$  you take here whatever the value of  $A$  of  $x$  may be, this whole thing turns out to be 0. Now, minimum is a  $t$  naught 1 is its neutral element. So, from here what we get is  $0 \max A$  of  $x$  naught which is essentially  $A$  of  $x$  naught. Well, let us look at what happens in the case of Smeth Magrez modification matching function.




(Refer Slide Time: 09:08)


### Singleton Fuzzification and the Matching Function $M$

$$M_S(A, A') = \min_{x \in X} I(A'(x), A(x)).$$

$$A'_{x_0}(x) = \begin{cases} 1, & \text{if } x = x_0, \\ 0, & \text{if } x \neq x_0. \end{cases}$$

$$\begin{aligned} M_S(A, A'_{x_0}) &= \min_{x \in X} I(A'_{x_0}(x), A(x)), \\ &= \left\{ \min_{x \neq x_0} I(A'_{x_0}(x), A(x)) \right\} \bigwedge I(A'_{x_0}(x_0), A(x_0)) \\ &= \left\{ \min_{x \neq x_0} I(0, A(x)) \right\} \bigwedge I(1, A(x_0)) \\ &= 1 \bigwedge A(x_0) = A(x_0). \end{aligned}$$





Balazsramaniam Jayaram
ARFST - SBR - Mamdani Fuzzy Systems


Once again let us give the input which is singleton fuzzified value at  $x$  naught. So,  $M$  s of  $A$  and  $A$  dash  $x$  naught is minimum over  $i$  of  $A$  dash  $x$  naught of  $x$  comma  $A$  of  $x$ . Note that  $A$  dash and  $A$  they have changed positions implication is not commutative. So, we take the given input at the first place and the antecedent at the second place.

Now, this can once again be broken down into all those  $x$  which are not equal to  $x$  naught and  $x$  being equal to  $x$  naught. Now, if you look little deeper inside here at  $A$  dash  $x$  naught of  $x$  when  $x$  is not equal to  $x$  naught what we get is 0. So, this essentially becomes minimum over  $x$  naught equal to  $x$  naught  $i$  of 0 comma  $A$  of  $x$  with minimum of.


On the right hand side what you see after the min operation is at  $A$  dash  $x$  naught of  $x$  naught we know it is 1 and so this is what we get,  $i$  of 1 comma  $A$  of  $x$  naught. Now remember in  $i$  is an implication  $i$  of 0 comma  $y$  is always one no matter what implication you take. So, all of these for every  $x$  not equal to  $x$  naught no matter what  $A$  of  $x$  can be each of these is actually 1.

So, minimum over all of these will be 1. Now, what happens to this value here? If  $i$  has neutrality property; that means, if 1 is the neutral element left neutral element of the corresponding implication, then essentially what we get is a  $A$   $x$  naught. And we get the match value, the similarity value between a singleton fuzzified input and any set  $A$  which is a fuzzy set over  $x$  is actually the membership value of  $x$  naught in that fuzzy set  $A$ . This is exactly what we have seen also in the case of Zadeh's matching function.

(Refer Slide Time: 11:14)



## Mamdani Fuzzy Systems Implementation in Matlab




Balasubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

Well with this, let us move into looking at Mamdani fuzzy systems, how we can build one using the fuzzy logic toolbox in MATLAB.

(Refer Slide Time: 11:27)


### Mamdani FS in Matlab: Terminology



**SBR: Form**

$$\mathbb{F} = \{P_X, P_Y, R(A_i, B_j), h, M, J, G, g\}$$
$$M_{\oplus \otimes}(A, A') = \bigoplus_{x \in X} \left( A(x) \otimes A'(x) \right).$$

- Fuzzification  $h$ : **Singleton**.
- Matching Function  $M_{\oplus \otimes}$ :
  - $\otimes$  - **And method**.
  - $\oplus$  - **Or method**.
- Modification  $J$ : **Implication**.
- Aggregation  $G$ : **Aggregation**.
- Defuzzification  $g$ : **Defuzzification**.



Balasubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

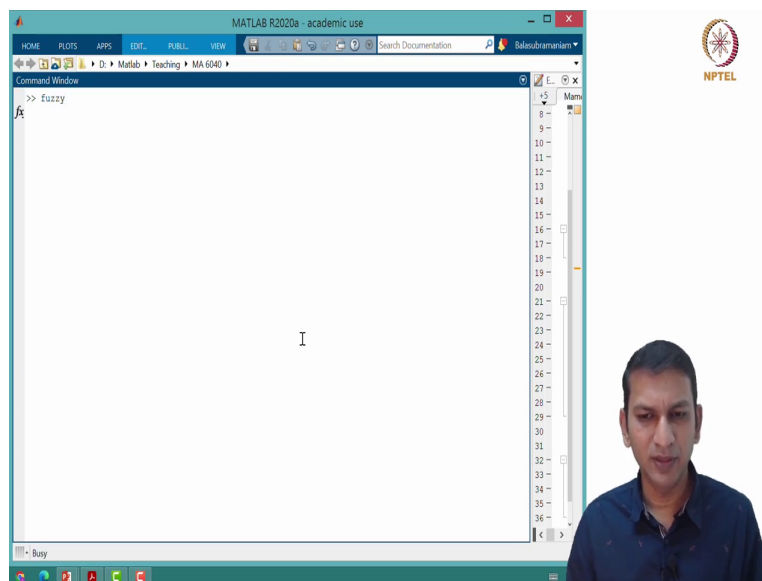
Before that a few terminologies; look at the form of similarity based reasoning fuzzy inference system. So, we have seen this  $P_X$  and  $P_Y$  they are the coverings fuzzy coverings over the domain of  $X$  and  $Y$ , this is the rule base where  $A_i$ 's and  $B_j$ 's are actually coming from the corresponding fuzzy coverings from  $X$  and  $Y$ .  $h$ , is the fuzzifier,  $M$  is the matching

function,  $J$  is the modification function,  $G$  is the aggregation function and small  $g$  is the defuzzifier.

Note that in Mamdani fuzzy system as implemented in the fuzzy logic toolbox MATLAB, this is the family of matching function that we use; that means, by specifying these two operations we are specifying what the matching function will be. What is the fuzzifier that MATLAB considers?

Typically it is a singleton fuzzifier that we will consider there and for the matching function the operation, the internal operation is called the AND method and the external operation is called the OR method. And MATLAB calls modification as implication aggregation as aggregation and defuzzification as defuzzification. So, with this terminology we are ready to move into looking at the MATLAB core.

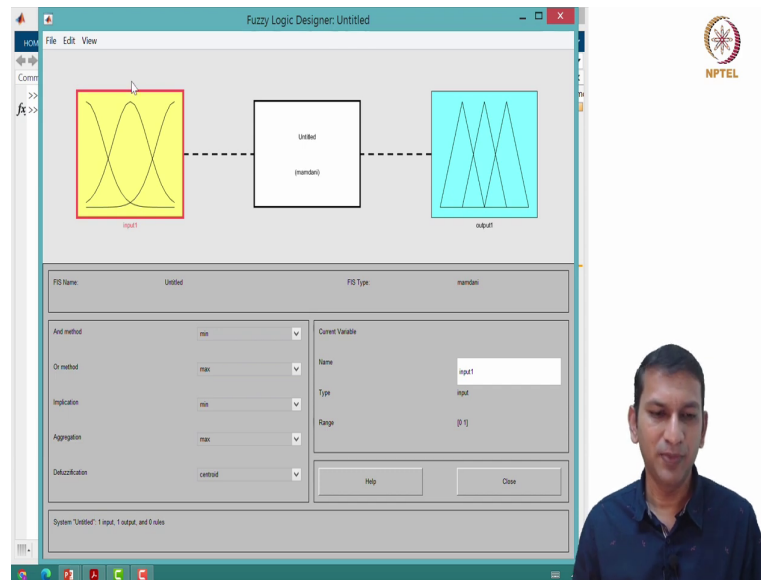
(Refer Slide Time: 13:00)



It is typical when we want to learn a new algorithm to start with a very very simple example; for instance when we have learnt how to find out the Fourier series expansion of a function. It is typical to start with the identity function and so we will also do the same. So, let us try to build a Mamdani fuzzy inference system, which will approximate the identity function over the unit interval.

So, essentially the function that we are going to consider is function from  $[0,1]$  to  $[0,1]$  and it is essentially  $F(X)$  is equal to  $x$ . Now, how do we build the system? What are the things that we need to take into consideration?

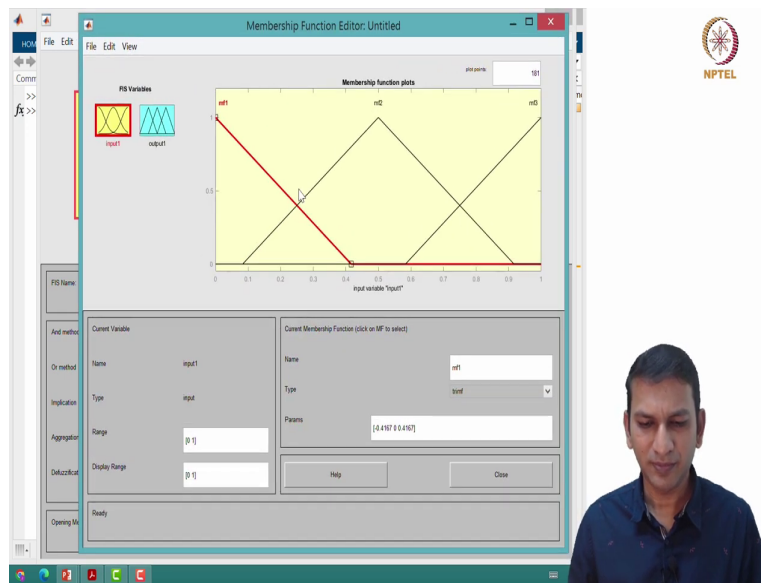
(Refer Slide Time: 13:53)



So, in MATLAB when if you have the fuzzy logic toolbox installed all you need to say is type fuzzy and you will see that it opens up the main window, wherein what you need to be specifying are immediately accessible. For instance, you see here that And Or method together they form the matching function, implication is the modification function, aggregation is the aggregation and defuzzification is the corresponding defuzzification function.

So now, when we have a clear idea of what we want to implement what function we want to approximate that is when we can start inputting the data. Now, what is it that we need to do we have a function in mind which we need to approximate and currently the function is the identity function. So, we need to first come up with the coverings on the corresponding domains of  $x$  and  $y$ . Since it is an identity function over  $[0,1]$  the domain of  $x$  is  $[0,1]$  the domain of  $y$  is also  $[0,1]$ . Now, we need to build the corresponding fuzzy sets on this domain.

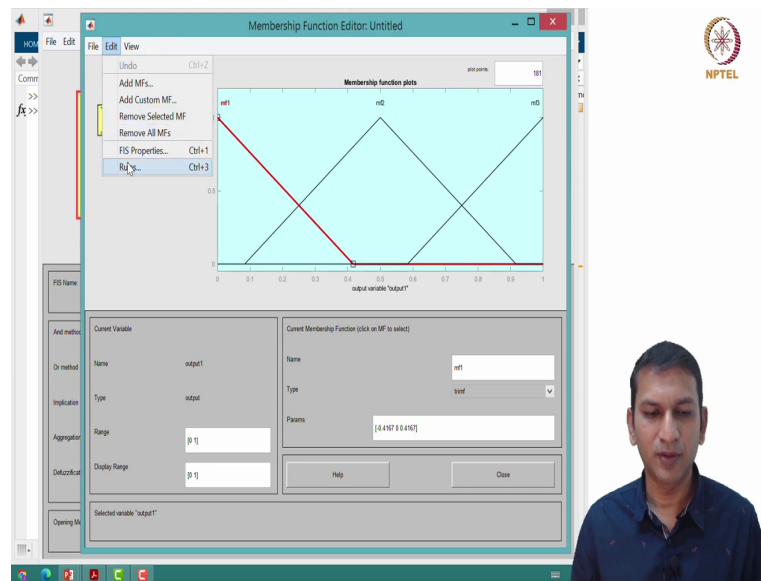
(Refer Slide Time: 15:12)



So, let us so if you double click on this you will see that it opens up another window which tells you a few things. So, you see here input variable 1 output variable 1. Now and it automatically comes up by default, it gives you three membership functions all of them are triangular in nature.

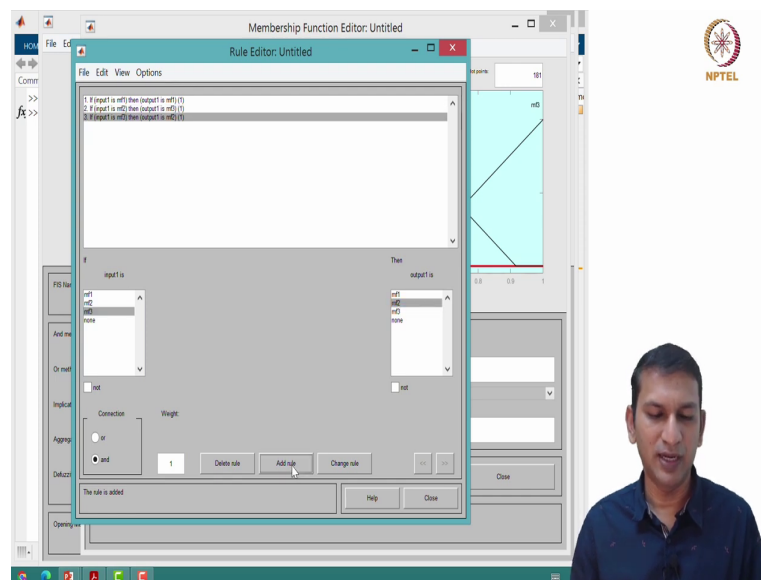
Of course, this is what we are going to modify and you see here the range automatically is [0,1] and in fact, this is what we need for the moment. Now, you could give any name M of 1 M of 2, this is the default name that is given to us. Now, we need to be able to specify the fuzzy covering on this on the input domain.

(Refer Slide Time: 15:58)



And a similar fuzzy covering on the output domain; once these are done, then we need to fix up the rules.

(Refer Slide Time: 16:10)

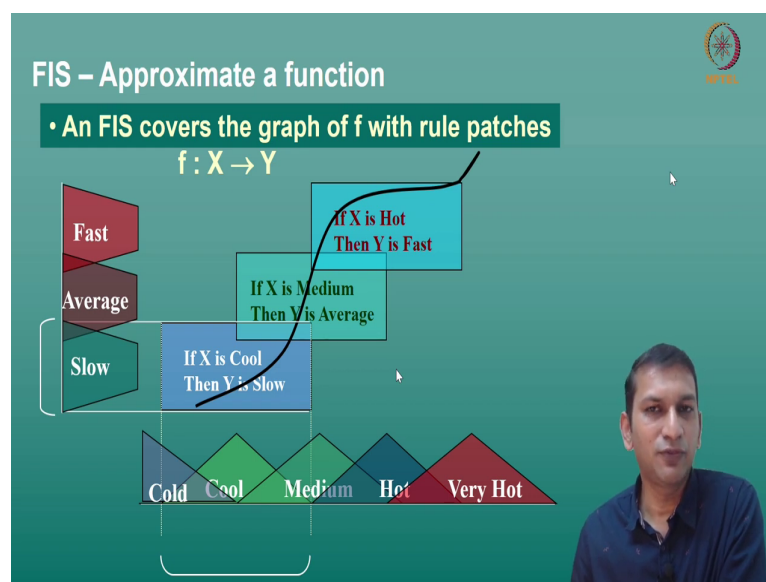


Now, how do we add the rules? So, on this side you see that we have, on this side you will see that we have all the input fuzzy sets and this side we have the output fuzzy sets. We need to relate the input fuzzy set to the corresponding output fuzzy set to get a rule. You choose two of them and say add a rule and you see that it has been added.

You take mf2 and if you want to relate it to mf3, you say add a rule and it will add a rule mf3 and mf2 may be add a rule. We can add a rule we can also delete a rule or change a rule. Well, but the most important thing is how do we come up with these membership functions. That is how do we get the fuzzy covering for this ident on this domain of  $[0,1]$  for  $x$  and  $[0,1]$  for  $y$  and.

So, so that we can relate these the fuzzy sets on  $x$  to fuzzy sets on  $y$  as rules which will actually capture the function that we are trying to approximate. Well for that we need to understand how a fuzzy inference system approximates the system function. We know it is doing it by covering it with overlapping rule patches. Let us revisit the visual reasoning that we did, visual illustration of the similarity based reasoning that we have done earlier.

(Refer Slide Time: 17:53)



Well, we know an FIS approximates a given system function  $f$  from  $X$  to  $Y$ , how did we do this? We took the domain of  $X$  in that of  $Y$ ; if this is the given function what we did was we came up with a fuzzy covering of  $X$  and a fuzzy covering of  $Y$ . And we were relating the fuzzy sets on  $X$  to fuzzy sets on  $Y$ , through rules and hence we are picking the antecedents from the covering on  $X$  and the consequence from the fuzzy covering on  $Y$ .

For instance we have seen that if you relate cool to slow through a rule, that if  $X$  is cool then  $Y$  is slow essentially what we are seeing is this that we have the support of cool and we have the support of slow and this is the rule patch that we get. Essentially, we are seeing two things

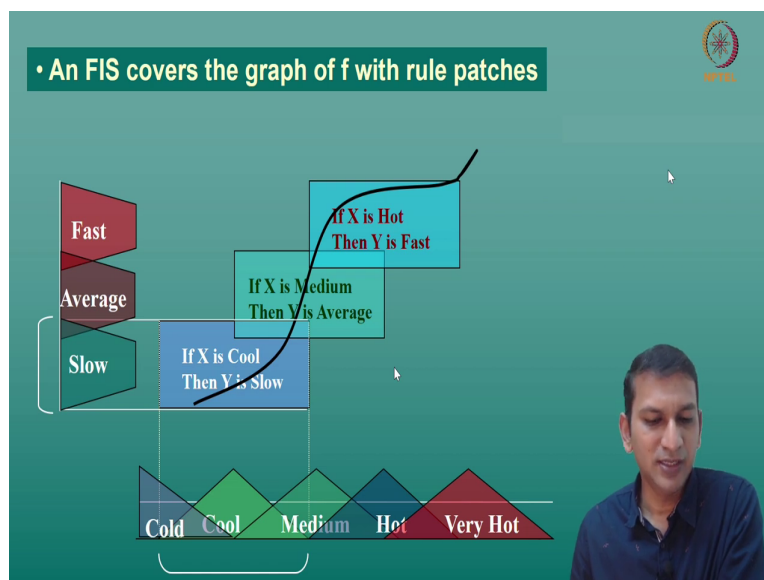
here, if an input falls in the support of cool then the  $Y$ , the output  $Y$  is essentially restricted to the support of slope.

So, the ordered pairs which actually will belong to this particular rule are given by this shaded region this is the first thing. So, similarly when we construct for each of these rules we get these patches. And now the second thing is we clearly see that with these patches we are trying to approximate the function, we are covering the function.

But also note one thing, that when we have this at the end of it when we give an input we would get an output, it is a modified consequent of the slow average and fast. Finally, the aggregated output is a fuzzy set and we need to defuzzify. Now, depending on the defuzzification that is chosen it could fall anywhere within this region. So, it is always better to finely granularize it so that we have smaller and smaller patches which cover this function.

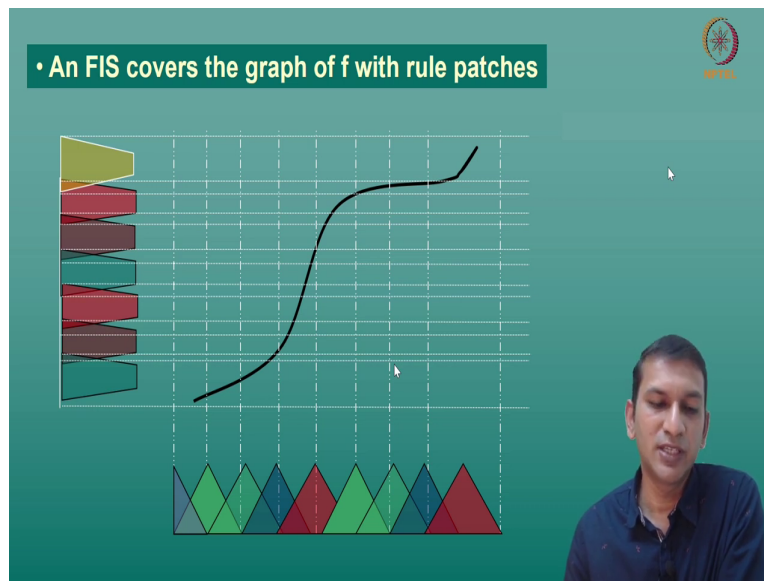
Of course, there is also a downside. If you want smaller patches then what do we need to do? We need to have more fuzzy sets; that means, the granularity on the input and output domains should go up. For instance, this is what we had in this case. Remember once again we are looking at fuzzy inference systems which cover the graph of  $F$  with rule patches.

(Refer Slide Time: 20:29)





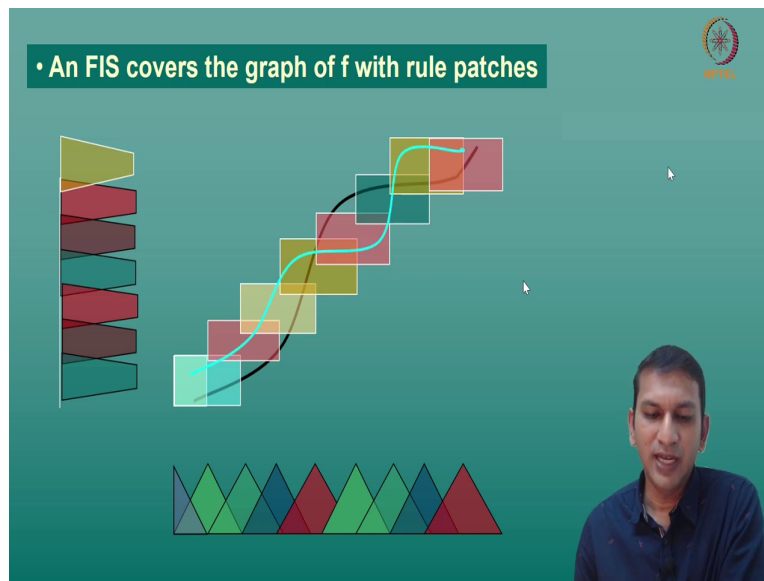
(Refer Slide Time: 20:31)



Now, instead of having just 5 fuzzy sets on  $X$ , let us assume that we have almost 9 and on the output side we have 7. Now, let us look at how we are going to approximate this function through a fuzzy inference system. So, essentially it means coming up with rules, but we understand rules are actually going to patch up this function, come up with overlapping patches of this function. What is our first point? So now, let us look at the supports of the input fuzzy sets or the fuzzy covering that we have on the input set  $X$ .

This is the fuzzy covering that we have, these are the supports of the fuzzy covering that we have on  $X$ . Similarly, the supports of the fuzzy covering we have on  $Y$ . Now, we will use this grid to see how to relate a particular fuzzy set on  $X$  to a fuzzy set on  $Y$ . For instance consider this fuzzy set and this fuzzy set.

(Refer Slide Time: 21:41)



If you relate them, what you get is this patch; that means, all the values all the ordered pairs  $UV$  that excite this rule will fall within this patch. So, if you have an input  $X$  which excites this fuzzy set here which falls in the support of this fuzzy set, then the  $y$  values will definitely belong only to the support of the fuzzy set you see here.

However, notice that we are having a fuzzy covering; that means, there are fuzzy sets on the input domain which are overlapping. So, we need to consider them also in the overall when finding the overall output. So, consider the second fuzzy set here and let us relate that to the second fuzzy set on the output side. What we get is another rule patch and now it is overlapping clearly.

Now, if you go along this logic and start to relate input fuzzy sets to output fuzzy sets, perhaps this is one kind of a rule patch covering that you we can obtain a 5th rule, a 6th rule, a 6th rule, 8th rule and a 9th rule. So, you see here there are 9 fuzzy sets here and each one of them has been related to some fuzzy set or the other. As you can see on the output domain  $y$  we have only 7 fuzzy sets, on the input we have 9.

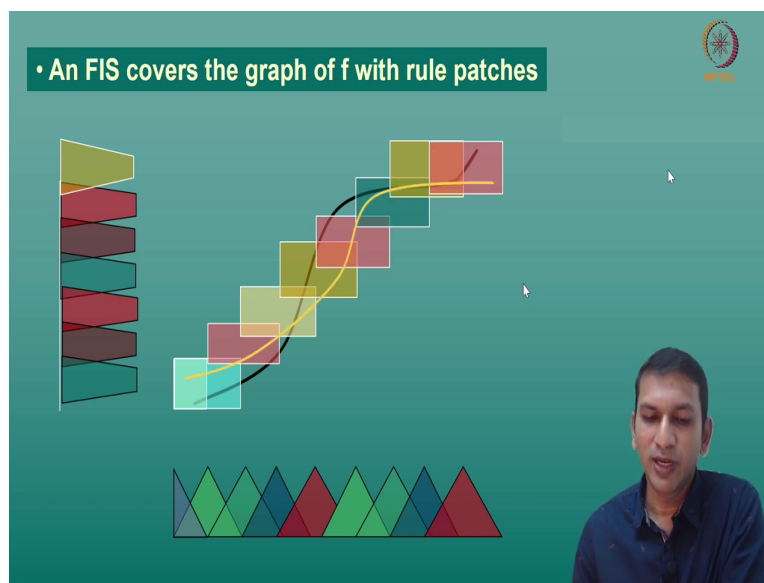
So, typically as was already mentioned you will have depending on the function that we have, you would tend to always have more input fuzzy sets than output fuzzy sets. Now, there are a few crucial things to note here. Firstly, look at this we said that it is trying to cover approximate the function by overlapping rule patches, but now you see here these parts of the function are actually not covered.

So, no matter what defuzzification mechanism you use, the output will lie within these overlapping boxes patches. And so, it is not going to approximate the function very well here that is the first thing. The second thing that you would see is, when you apply any defuzzification or whatever choice of the defuzzification function that you take, even if this function were to remain within these rule patches the output that you would get, the final function that you would get may not exactly match this.

Of course, that is because originally to begin with we did not know the function. If you knew the function then we can tweak it, for instance we only had an inkling that this is how the function should work or the air conditioner should work that is where we saw this monotonic function. So, if you have the ideal situation in your mind and if you are trying to approximate it, this is how you would approximate it.

But depending on the defuzzification scheme and all other operation similarity based reasoning inference operation that you use. What you would get actually is that a function of this kind perhaps as you will see, it will remain within these overlapping rule patches or you could also get a function of this type.

(Refer Slide Time: 24:58)




Both of them are in some sense approximating the original ideal function that you have in your mind. So, what we need to do is in fact, take  $q$  from this and come up with patches which are small in nature which are likely to fall on the function on the graph of the function

which means we need to accordingly granularize, the domain of X and relate and also the domain of Y.

And intelligently relate these two that is what we would like to do. Note, once again that a Mamdani fuzzy inference system is a similarity based reasoning system, you could use any of the fuzzification for h, any matching function for M and of course, any modification aggregation or defuzzification functions for the corresponding operation. However, what we have seen or what we are seeing right now on the slide is how MATLAB fuzzy logic toolbox implements it.

And since we are going to make use of that to build a Mamdani fuzzy inference system this terminology was used; otherwise Mamdani fuzzy inference system is a similarity based reasoning system inference system, which does not put any restriction on the kind of fuzzification that you can have or the matching function you can have.

(Refer Slide Time: 26:24)




A quick recap ...

- An FIS approximates through overlapping rule patches.
- Mamdani Fuzzy System.
- Intro to building Mamdani FIS using Matlab FL Toolbox.

Next Lecture:

**Building a Mamdani FS in Matlab using Fuzzy Logic Toolbox**



Balasubramaniam Jayaram ARFST - SBR - Mamdani Fuzzy Systems

So, far in this lecture the importance of how a fuzzy inference system approximates a fuzzy or system function is hard to exaggerate, it does it through overlapping rule patches. We saw what a Mamdani fuzzy inference system is and we had a short introduction to building Mamdani fuzzy inference system using MATLAB fuzzy logic toolbox.

The idea was to show the environment of MATLAB fuzzy logic toolbox, looking at it was clear what are the components of the fuzzy inference system that we need to come up with,

that we need to be ready with so that we can build a fuzzy inference system Mamdani one that.

And we have seen it means to be able to have a clear idea of the function that we are trying to approximate and intelligently sample the fuzzy set which will form the covering over  $x$  and  $y$  and also to relate the fuzzy sets on  $X$ , to fuzzy sets on  $Y$  as rules so that we can cover the function that we have in our mind through overlapping rule patches.

And this is what we will be doing in the next lecture. We will take a couple of functions and see how exactly to build this fuzzy inference system, Mamdani fuzzy inference system which would more or less approximate the function that we are considering. Glad you could join us today for this lecture. Hope to see you soon in the next lecture.

Thank you again.