Bare-Bones Fuzzy Inference System

Dr. Archie Singh **Khuman**

INSTITUTE OF ARTIFICIAL INTELLIGENCE · GAMES, MATHS & ARTIFICIAL INTELLIGENCE · GH6.57

De Montfort University, Gateway House, Leicester, United Kingdom, LE1 9BH

+44 (0)116 250 6251 | arjab.khuman | G arjab singh khuman | archie khuman

"Be inspired or be inspirational - Either way, great things can be accomplished."

Outline for the Presentation...

1. Getting Started
2. Fuzzy Inference System
3. The Code

MATLAB - Getting Started

- We will create a bare-bones *fuzzy inference system* (**fis**).
- If you did install **MATLAB** on your own devices, make sure that you also included the **Fuzzy Logic Toolbox** as part of the install.
- The **Fuzzy Logic Toolbox** contains the *functions calls* that we will be making use of.
- Without the toolbox, your commands will not be executed, and your scripts will **NOT** compile & run.
- **Absolutely** make sure to also watch the accompanying *video tutorial* which can be found in the same lab folder.
- The tutorial will offer an additional *understanding*, and you can follow along in your own time.

Outline for the Presentation...

1. Getting Started
2. Fuzzy Inference System
3. The Code

Fuzzy Inference System

- For this lab we will start on creating a minimal *fuzzy system*.
- Your coursework will involve you creating *your own* fuzzy inference systems, they are all started in much the same way.
- You will get to decide on *what* your system will be doing.
- Use your creativity and choose something that genuinely *interests* you.
- Many previous students have gone on to implement fuzzy systems into their *development projects*.
- A fuzzy system is an **AI** paradigm, so *more* marks if it's applicable to your project.

Fuzzy Inference System

- Let's create a new script, simply press **CTRL+N** to open up a new script file.
- Equally, you can navigate to **Home Tab** at the top, and select **New Script**.
- We will use the editor to *create* & *populate* our fuzzy inference system.
- MATLAB has great *plotting* functionality, so do please make use of this, especially for your coursework.
- When you get to *visualise* your fuzzy system, you get a better understanding.
- We will keep coming back to this script file, but make sure to **watch** the accompanying *video tutorial*.

Fuzzy Inference System

- When you start MATLAB, please remember to navigate to a *directory* that you have *permission* to save to if working from the labs, shouldn't be an issue if installed on your own devices.
- We will begin with using the command window, the **main** window, to enter in commands following the following *prompt*: >>
- We will begin with exploring the declaration for creating a *new* fuzzy system.
- Enter the following command at the prompt, **DO NOT** include the >> in your command itself.

```
>> help newfis
```

Fuzzy Inference System

- You can see that the function call `newfis` can take several parameters and can be *overloaded*.
- In the **script file** type the following commands, each one on a new line:

```
a = newfis('Player Skill');
```

- A fuzzy system in MATLAB **CANNOT** be created without the declaration of a `newfis`.
- With the declaration made, we have a new *instance* of a fuzzy system.
- The next logical thing to do is populate the system with an *input variable*.

Fuzzy Inference System

- MATLAB writes from right to left, when I declared an instance of `newfis`, this was written to variable `a`.
- Therefore `a` will constantly be *updated* with more information when we populate the system.
- Let's assume that this system is with regards to *Player Skill*, and the first input is with regards to *Player Accuracy*.
- Inputs and outputs with regards to a fuzzy system are referred to as *variables*.
- In the **Command Window** enter the following:

```
help addvar
```

Fuzzy Inference System

- You can see from the explanation that `addvar` *adds* a variable to the system.
- You will need to pass in several key pieces of *information* when using `addvar`.
- You will first need to specify the fuzzy system this variable will be *assigned* to.
- In our case it will be `a`, the variable that holds the information for our fuzzy system.
- Next, we have to specify the `varType` of variable, this only has **2** options, an **input** or an **output**.
- We will be declaring an *input* to begin with.

Fuzzy Inference System

- You then need to declare a `varName`, a *name* for the variable contained within single quotes.
- You will also need to specify the `varBounds`, the *minimum* and *maximum* x-axis values.
- Go back to your **script file** and add the following command:

```
a=addvar(a,'input','Player Accuracy (%)',[0 100]);
```

- We have now declared a new input variable labelled *Player Accuracy*, which starts at **0** and finishes at **100**, using % as the unit of measurement.
- This is all *written* to variable `a` and updated with this new information.

Fuzzy Inference System

- We now have an instance of a fuzzy inference system with a single input variable - *Player Accuracy*.
- Remember to keep *saving* the file after adding to it, I saved mine as **PlaySkill.m**, you can call it what you like.
- The input variable itself is empty, so the next logical thing to do is to *populate* the input.
- We will populate it with **3** fuzzy membership functions.
- In the **Command Window** enter the following:

```
help addmf
```

- You can see from the explanation that `addmf` *adds* a membership function to the system.

Fuzzy Inference System

- You will need to pass in several key pieces of *information* when using `addmf`.
- You will first need to specify the fuzzy system this variable will be *assigned* to.
- In our case it will be `a`, the variable that holds the information for a fuzzy system.
- Next, we have to specify the `VarType` of membership, this also has **2** options, an *input membership* or an *output membership*.

Fuzzy Inference System

- We also need to specify the `Index` value.
- This is the input *index* of the input we want to add the membership function to.
- As we only have **1** input variable so far, the current `Index` is **1**.
- We will also need to give the membership a `varName`, a *label*.
- You will also need to declare the *type* of membership function.
- You will then need to pass forward the *parameter values* for that membership function.
- Depending on the function you choose, will determine the *number* of parameters required.

Fuzzy Inference System

- Enter the following into the script file under your `addVar` declaration:

```
a=addmf(a,'input',1,'Poor','trapmf',[0 0 15 25]);
```

- We have created a *trapezoidal* membership with the label *Poor*.
- Compile the code by pressing **F5** while in the editor.
- Did it compile? **Yes**, but we cannot see anything.

Fuzzy Inference System

- Let us add in a *plot* function call `plotmf`.
- Similar to `plot` call, but the `plotmf` is *specifically* for membership functions.
- This will allow us to get a sense of what the system *looks* like.
- In the **script file** add the following:

```
plotmf(a, 'input', 1)
```

- Now compile & run the code, what do you see?

Fuzzy Inference System

- We can now see the *distribution* of the fuzzy set *Poor*.
- The parameter values that were chosen were *arbitrary* and *subjective*.
- For your systems, **YOU** can choose values that make sense to you.
- Let us now *add* in another fuzzy set and membership function into the system.
- *Modify* your script file so that it looks like the following code snippet:

Fuzzy Inference System

- Compile & run the following code:

```
a = newfis('Player Skill');

a=addvar(a,'input','Player Accuracy (%)',[0 100]);

a=addmf(a,'input',1,'Poor','trapmf',[0 0 15 25]);

a=addmf(a,'input',1,'Average','trimf',[20 50 80]);

plotmf(a, 'input', 1)
```

Fuzzy Inference System

- *Play* about with the membership parameter values.
- **YOU** get to define what you think is appropriate.
- Try and get the fuzzy sets to have a little more *overlap* between themselves.
- *Add* in a third fuzzy membership function with the following:

```
a=addmf(a,'input',1,'Good','trapmf',[70 90 100 100]);
```

- Compile & run the code, we now have a populated first input.
- Consider adding in *another* input variable with `addvar`.
- Call this second input *Damage Output*, with a range bound of [0, 100].

Fuzzy Inference System

- Consider adding **5** membership functions to this new input.
- **2** trapezoidal membership functions, one on either end, and **3** triangular memberships in the the middle.
- *Very Little*, *Little*, *Medium*, *High* & *Very High*.
- Choose *appropriate* values for the parameters of the functions.
- Add the following into the **script file**, under the declaration of your first input:

```
a=addvar(a,'input','Damage Output (%)',[0 100]);
```

Fuzzy Inference System

- Add in **5** membership functions:

```
a=addmf(a,'input',2,'Very Little','trapmf',[0 0 ...
10 20]);

a=addmf(a,'input',2,'Little','trimf',[15 25 35]);

a=addmf(a,'input',2,'Medium','trimf',[30 50 70]);

a=addmf(a,'input',2,'High','trimf',[65 75 85]);

a=addmf(a,'input',2,'Very High','trapmf',[80 90 ...
100 100]);
```

Fuzzy Inference System

- To see the new input, **remove** the previous instance of:

```
plotmf(a, 'input', 1)
```

- **Replace** it with the following:

```
subplot(4,1,1),plotmf(a, 'input', 1)

subplot(4,1,2),plotmf(a, 'input', 2)
```

- I have *left room* in the figure handler for you to create an output for the system.

Fuzzy Inference System

- In much the same way as you declared an input, you can *declare* an output as follows:

```
a=addvar(a,'output','Player Skill',[0 100]);
```

- As we have moved onto declaring an output, we **restart** the indexing from **1**.
- You can have *multiple outputs*, each new output would *increment* the index.
- In much the same way, we will now *populate* the output with membership functions.

Fuzzy Inference System

- Add the following to the **script file**:

```
a=addmf(a,'output',1,'Low Skill','trapmf',[0 0 10 ...
25]);

a=addmf(a,'output',1,'Average Skill','trapmf',[20 ...
40 50 70]);

a=addmf(a,'output',1,'High Skill','trapmf',[65 75 ...
100 100]);
```

- To view this new output, **modify** the subplots:

```
subplot(4,1,1),plotmf(a, 'input', 1)

subplot(4,1,2),plotmf(a, 'input', 2)

subplot(4,1,4),plotmf(a, 'output', 1)
```

Fuzzy Inference System

- You have effectively created *non-functioning* fuzzy inference system (**fis**).
- We say non-functioning because there is **NO** *inference engine* or *rule-base*.
- We will introduce these aspects in the *next* lab session.
- Play about with the parameter values and change the *distribution* of the fuzzy sets.
- Change the *type* of membership functions being made use of.
- Play about with it and gain a better understanding of what is *going on*.

Outline for the Presentation...

1. Getting Started
2. Fuzzy Inference System
3. The Code

The Code

```
% A declaration of new FIS

a = newfis('Player Skill');

% Declaring a new variable - this is an INPUT(1)

a=addvar(a,'input','Player Accuracy (%)',[0 100]);

% Populating the 1st input variable with ...
membership functions

a=addmf(a,'input',1,'Poor','trapmf',[0 0 15 25]);

a=addmf(a,'input',1,'Average','trimf',[20 50 80]);

a=addmf(a,'input',1,'Good','trapmf',[70 90 100 100]);
```

The Code

```
% Declaring a new variable - this is another INPUT(2)

a=addvar(a,'input','Damage Output (%)',[0 100]);

% Populating the 2nd input variable with ...
membership functions

a=addmf(a,'input',2,'Very Little','trapmf',[0 0 ...
10 20])

a=addmf(a,'input',2,'Little','trimf',[15 25 35]);

a=addmf(a,'input',2,'Medium','trimf',[30 50 70]);

a=addmf(a,'input',2,'High','trimf',[65 75 85]);

a=addmf(a,'input',2,'Very High','trapmf',[80 90 ...
100 100]);
```

The Code

```
% Declaring a new variable - this is an OUTPUT(1)

a=addvar(a,'output','Player Skill',[0 100]);

% Populating the output variable with membership ...
functions

a=addmf(a,'output',1,'Low Skill','trapmf',[0 0 10 ...
25]);

a=addmf(a,'output',1,'Average Skill','trapmf',[20 ...
40 50 70]);

a=addmf(a,'output',1,'High Skill','trapmf',[65 75 ...
100 100]);
```

The Code

```
% The subplots to visualise the system

subplot(4,1,1),plotmf(a, 'input', 1)

subplot(4,1,2),plotmf(a, 'input', 2)

subplot(4,1,4),plotmf(a, 'output', 1)
```

Bare-Bones Fuzzy Inference System

Dr. Archie Singh **Khuman**

INSTITUTE OF ARTIFICIAL INTELLIGENCE · GAMES, MATHS & ARTIFICIAL INTELLIGENCE · GH6.57

De Montfort University, Gateway House, Leicester, United Kingdom, LE1 9BH

+44 (0)116 250 6251 | arjab.khuman | G arjab singh khuman | archie khuman

"Be inspired or be inspirational - Either way, great things can be accomplished."