# Old System Design Notes:

## System Creation & Plan Appendices

### 1. Define the Fuzzy System Objectives

Primary Objective: To accurately estimate the mass of dark matter particles by analysing the gravitational lensing phenomena, with a special focus on the texture or "fuzziness" as predicted by the Fuzzy Dark Matter (FDM) theoretical model. This involves:

##### Data Collection and Analysis:

Gather high-resolution observational data of gravitational lensing events, emphasising the identification and quantification of the "fuzziness" or wave-like distortions in the lensing patterns, which are indicative of the presence of FDM.

Analyse the angular resolution and the distortion patterns of gravitational arcs within the lensing data to correlate specific characteristics with varying dark matter particle masses.

With this initial project I will be creating a pseudo set of data based on actual data for ease of data pre-processing, my would be to continue on with actual data on a larger scale post project completion.

## Fuzzy Logic Model Development:

Develop a comprehensive set of input variables for the FLS, derived from observational data characteristics such as angular resolution, density distortions, and the distribution of gravitational arcs.

Construct and calibrate membership functions for each input and output variable, capturing the nuances of the gravitational lensing phenomena and the inferred properties of dark matter.

Formulate a robust rule base that logically connects the observed lensing characteristics with the potential mass range of dark matter particles, incorporating the theoretical underpinnings of FDM.

##### Simulation and Validation:

Simulate the fuzzy logic model using a variety of input scenarios to predict dark matter particle mass, testing the model's sensitivity and accuracy in reflecting the expected outcomes based on the FDM model.

Validate the model predictions against established data and theoretical predictions, refining the model parameters and rules based on empirical evidence and expert feedback.

### 2. Identify Input and Output Variables

Inputs: The input variables for the FLS could be derived from the characteristics of the gravitational lensing observations, such as:

Angular resolution of the observation (milli-arcseconds).

Density of the gravitational lens (could be represented as a function of the distortion patterns observed).

The width and distribution of gravitational arcs observed.

Output: The output variable would be the estimated mass of the dark matter particle, likely expressed in electronvolts (eV).

##### Inputs and Outputs Definition

##### Angular Resolution of the Observation (milli-arcseconds):

Type: Continuous

Description: Measures the smallest angular separation that the observation can distinguish, crucial for detecting the fine details in gravitational lensing.

##### Density of the Gravitational Lens:

Type: Continuous

Description: Quantifies the distortion patterns observed in the lensing, indicative of the mass distribution within the lens galaxy.

##### Width and Distribution of Gravitational Arcs Observed:

Type: Continuous

Description: Relates to the spatial extent and pattern of the arcs produced by lensing, providing insights into the structure and distribution of dark matter.

##### Estimated Mass of the Dark Matter Particle (electronvolts, eV):

Type: Continuous

Description: Represents the inferred mass range for dark matter particles based on the input observations.

### 3. Construct Membership Functions

For each input and output variable, define appropriate membership functions. These functions translate the quantitative inputs into qualitative descriptors (e.g., "low," "medium," "high") based on the domain knowledge:

Angular Resolution: High, Medium, Low

Density Distortion: Strong, Moderate, Weak

Arc Distribution: Concentrated, Dispersed

Dark Matter Particle Mass: Very Light, Light, Medium, Heavy

# Old Rulebase Development:

IF Angular Resolution is High AND Density Distortion is Strong AND Arc Distribution is Concentrated THEN Dark Matter Particle Mass is Heavy.

IF Angular Resolution is Low OR Density Distortion is Weak THEN Dark Matter Particle Mass is Very Light.

##### Plain English Rules:

Here is a systematic enumeration of all the rules, considering three linguistic values ('Low', 'Medium', 'High') for each of the three inputs, resulting in 3^3 = 27 rules:

If (AR is Low) and (Density is Low) and (Width is Narrow), then Mass is Light.

If (AR is Low) and (Density is Low) and (Width is Medium), then Mass is Light.

If (AR is Low) and (Density is Low) and (Width is Wide), then Mass is Light.

If (AR is Low) and (Density is Medium) and (Width is Narrow), then Mass is Medium.

If (AR is Low) and (Density is Medium) and (Width is Medium), then Mass is Medium.

If (AR is Low) and (Density is Medium) and (Width is Wide), then Mass is Light.

If (AR is Low) and (Density is High) and (Width is Narrow), then Mass is Heavy.

If (AR is Low) and (Density is High) and (Width is Medium), then Mass is Medium.

If (AR is Low) and (Density is High) and (Width is Wide), then Mass is Medium.

If (AR is Medium) and (Density is Low) and (Width is Narrow), then Mass is Light.

If (AR is Medium) and (Density is Low) and (Width is Medium), then Mass is Light.

If (AR is Medium) and (Density is Low) and (Width is Wide), then Mass is Light.

If (AR is Medium) and (Density is Medium) and (Width is Narrow), then Mass is Medium.

If (AR is Medium) and (Density is Medium) and (Width is Medium), then Mass is Medium.

If (AR is Medium) and (Density is Medium) and (Width is Wide), then Mass is Medium.

If (AR is Medium) and (Density is High) and (Width is Narrow), then Mass is Heavy.

If (AR is Medium) and (Density is High) and (Width is Medium), then Mass is Medium.

If (AR is Medium) and (Density is High) and (Width is Wide), then Mass is Heavy.

If (AR is High) and (Density is Low) and (Width is Narrow), then Mass is Medium.

If (AR is High) and (Density is Low) and (Width is Medium), then Mass is Light.

If (AR is High) and (Density is Low) and (Width is Wide), then Mass is Light.

If (AR is High) and (Density is Medium) and (Width is Narrow), then Mass is Heavy.

If (AR is High) and (Density is Medium) and (Width is Medium), then Mass is Medium.

If (AR is High) and (Density is Medium) and (Width is Wide), then Mass is Medium.

If (AR is High) and (Density is High) and (Width is Narrow), then Mass is Heavy.

If (AR is High) and (Density is High) and (Width is Medium), then Mass is Heavy.

If (AR is High) and (Density is High) and (Width is Wide), then Mass is Heavy.

These rules assume a certain symmetry and logical consistency about how each input's "fuzziness" contributes to the final mass estimate. They also assume that 'High' inputs contribute more towards a 'Heavy' mass, while 'Low' inputs suggest a 'Light' mass.

% If (AR is Low) and (Density is Low) and (Width is Narrow), then Mass is Light.

struct('antecedent', [1 1 1], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is Low) and (Width is Medium), then Mass is Light.

struct('antecedent', [1 1 2], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is Low) and (Width is Wide), then Mass is Light.

struct('antecedent', [1 1 3], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is Medium) and (Width is Narrow), then Mass is Medium.

struct('antecedent', [1 2 1], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is Medium) and (Width is Medium), then Mass is Medium.

struct('antecedent', [1 2 2], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is Medium) and (Width is Wide), then Mass is Light.

struct('antecedent', [1 2 3], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is High) and (Width is Narrow), then Mass is Heavy.

struct('antecedent', [1 3 1], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is High) and (Width is Medium), then Mass is Medium.

struct('antecedent', [1 3 2], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Low) and (Density is High) and (Width is Wide), then Mass is Medium.

struct('antecedent', [1 3 3], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Low) and (Width is Narrow), then Mass is Light.

struct('antecedent', [2 1 1], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Low) and (Width is Medium), then Mass is Light.

struct('antecedent', [2 1 2], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Low) and (Width is Wide), then Mass is Light.

struct('antecedent', [2 1 3], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Medium) and (Width is Narrow), then Mass is Medium.

struct('antecedent', [2 2 1], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Medium) and (Width is Medium), then Mass is Medium.

struct('antecedent', [2 2 2], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is Medium) and (Width is Wide), then Mass is Medium.

struct('antecedent', [2 2 3], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is High) and (Width is Narrow), then Mass is Heavy.

struct('antecedent', [2 3 1], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is High) and (Width is Medium), then Mass is Medium.

struct('antecedent', [2 3 2], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is Medium) and (Density is High) and (Width is Wide), then Mass is Heavy.

struct('antecedent', [2 3 3], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Low) and (Width is Narrow), then Mass is Medium.

struct('antecedent', [3 1 1], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Low) and (Width is Medium), then Mass is Light.

struct('antecedent', [3 1 2], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Low) and (Width is Wide), then Mass is Light.

struct('antecedent', [3 1 3], 'consequent', [1], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Medium) and (Width is Narrow), then Mass is Heavy.

struct('antecedent', [3 2 1], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Medium) and (Width is Medium), then Mass is Medium.

struct('antecedent', [3 2 2], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is Medium) and (Width is Wide), then Mass is Medium.

struct('antecedent', [3 2 3], 'consequent', [2], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is High) and (Width is Narrow), then Mass is Heavy.

struct('antecedent', [3 3 1], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is High) and (Width is Medium), then Mass is Heavy.

struct('antecedent', [3 3 2], 'consequent', [3], 'weight', 1, 'connection', 1),

% If (AR is High) and (Density is High) and (Width is Wide), then Mass is Heavy.

struct('antecedent', [3 3 3], 'consequent', [3], 'weight', 1, 'connection', 1)

### 5. Implement Inference Engine

The inference engine evaluates the rules against the input variables to determine the output. This would involve:

Aggregating all rules based on the observed input values.

Applying the fuzzy logic operators (AND, OR, NOT) as per the rule definitions.

Determining the output fuzzy set for the dark matter particle mass.

### 6. Defuzzification

Finally, the fuzzy output set needs to be translated back into a single quantitative value (e.g., the estimated mass of the dark matter particle). Defuzzification methods include the centroid method, which computes the centre of mass of the output fuzzy set.

##### Future Considerations

Given the complexity of astrophysical data and the nuanced behavior of fuzzy dark matter, it may be beneficial to iterate on the membership functions and rule base with input from domain experts.

Incorporating machine learning techniques to refine the membership functions and rule base based on a dataset of observed gravitational lenses could enhance the system's accuracy.

### Data

1. Angular Resolution (milli-arcseconds) 2. Density of Gravitational Lens 3.Width and Distribution of Gravitational Arcs

|  |  |  |
| --- | --- | --- |
| 3.807947 | 77.393072 | 2.387743 |
| 9.512072 | 30.550558 | 1.83148 |
| 7.34674 | 85.725903 | 0.573279 |
| 6.026719 | 52.694576 | 0.332018 |
| 1.644585 | 60.551201 | 2.989731 |
| 1.644346 | 80.127638 | 4.53857 |
| 0.675028 | 69.885665 | 3.579285 |
| 8.675144 | 54.536258 | 4.092581 |
| 6.051039 | 96.604098 | 2.881796 |
| 7.109919 | 62.065729 | 4.23824 |
| 0.303786 | 41.719225 | 0.256151 |
| 9.702108 | 89.60232 | 2.506786 |
| 8.341182 | 10.650944 | 4.714007 |
| 2.202157 | 2.77079 | 0.783224 |
| 1.900067 | 73.359873 | 4.290971 |
| 1.915705 | 72.513869 | 2.425847 |
| 3.111998 | 96.325053 | 1.278922 |
| 5.295089 | 99.615092 | 0.513547 |
| 4.376256 | 86.321564 | 1.707607 |
| 2.983168 | 1.34394 | 1.21547 |
| 6.157344 | 86.350604 | 1.781765 |
| 1.480989 | 30.825888 | 4.26983 |
| 2.992232 | 50.259804 | 3.588158 |
| 3.726982 | 88.486938 | 0.173796 |
| 4.615093 | 70.859722 | 3.216444 |
| 7.873242 | 58.554452 | 2.909859 |
| 2.07677 | 54.345724 | 4.793969 |
| 5.190921 | 73.67309 | 2.969194 |
| 5.964904 | 96.652437 | 4.74438 |
| 0.559859 | 35.717862 | 4.627866 |
| 6.114694 | 49.787212 | 0.132785 |
| 1.788189 | 61.291435 | 0.403482 |
| 0.744011 | 54.709425 | 2.2615 |
| 9.493967 | 11.371053 | 3.319984 |
| 9.659757 | 0.37412 | 4.3054 |
| 8.103134 | 71.25963 | 0.598772 |
| 3.115676 | 33.606881 | 1.312239 |
| 1.066954 | 19.493899 | 4.890521 |
| 6.873907 | 21.653782 | 1.762877 |
| 4.45751 | 3.180016 | 1.818617 |
| 1.308179 | 89.448741 | 2.659654 |
| 5.002251 | 8.759426 | 0.208233 |
| 0.440446 | 18.930666 | 2.065758 |
| 9.102272 | 64.152319 | 1.507801 |
| 2.661922 | 51.195217 | 0.908674 |
| 6.658971 | 90.309015 | 4.229173 |
| 3.18594 | 31.474739 | 1.466278 |
| 5.248673 | 23.484298 | 1.104644 |
| 5.512432 | 81.45245 | 3.661966 |
| 1.930059 | 98.364588 | 0.213644 |

This is normalised in Excel.

### Run 1 Notes – Prior to changes being implemented

In the first run, I used centroid, the ruleviewer states that the following rules did not fire: 1, 7, 19, & 25

### PCA Variance Explained:

The PCA graph demonstrates that the three principal components account for nearly all of the variance in the dataset. The first component explains a substantial portion, but it's the combination of the three that captures the system's dynamics. This suggests that while the system has complexity, it doesn't require a high-dimensional space to capture most of its behavior.

### Rule Viewer Plot:

The Rule Viewer plot indicates how different rules are activated across the range of input values. The rules that did not fire may indicate regions of the input space that are not covered by the current rule set, or combinations of inputs that do not correspond to any of the defined outputs. This might suggest a need to re-evaluate the rule set for completeness, or it might validate that certain conditions are indeed rare or non-existent in the problem domain.

### Fuzzy Inference System Output - Surf Plot:

The Surface Plot is a visual representation of the Fuzzy Inference System's output, showing the estimated mass for a range of angular resolutions and densities with a fixed width. It shows how changes in these two variables can affect the estimated mass of dark matter. The peaks and valleys can suggest how sensitive the mass estimation is to changes in angular resolution and density. A sharp peak, for example, would indicate high sensitivity—small changes in input lead to large changes in output.

### Sensitivity Analysis for Angular Resolution Plot:

The Sensitivity Analysis for Angular Resolution indicates how changes in the angular resolution while holding other factors constant affect the FIS output. Ideally, this plot should show a smooth and consistent trend that helps to understand the system's behavior. However, the output appears to spike unexpectedly, which could indicate an issue with the rule base or the membership functions concerning angular resolution. This needs to be investigated to ensure the system's reliability.

### Conclusion and Next Steps:

From the PCA, it's evident that there is a good initial set of components capturing system variance. However, the fact that some rules did not fire suggests that either the rule base might need expansion, or certain scenarios are not applicable to the current system. The surf plot provides a good starting point for understanding the system's output but needs further refinement to understand the sensitivity thoroughly. The sensitivity analysis indicates potential issues with the rule base concerning angular resolution, which should be the focus of immediate attention.

### Adjustments Made:

The membership functions of the FIS have been carefully adjusted to enhance the overlap between them, creating a more resilient and flexible model. Specifically, the boundaries of the 'Low', 'Medium', and 'High' categories for inputs such as 'Angular Resolution', 'Density of the Gravitational Lens', and 'Width of Arcs' have been extended. This increased overlap allows for a smoother transition between membership degrees, ensuring that a broader range of input values can trigger multiple rules simultaneously. Such a configuration enables the system to better handle the intricacies of input data, providing a more nuanced and accurate output. The expanded overlap is expected to result in an FIS that can more effectively model the complex relationships inherent in the estimation of dark matter mass, thereby enhancing the system's analytical capabilities.

### Notes for further testing:

Expand the Rule Base: Investigate the conditions under which rules 1, 7, 19, & 25 should fire. Determine if additional rules are needed or if the existing rules need adjustment.

Membership Function Adjustment: Re-examine the membership functions for inputs, especially for angular resolution, to ensure they accurately represent the input space and are triggering the rules as expected.

Sensitivity Analysis Expansion: Extend the sensitivity analysis to other inputs, not just the angular resolution, to identify any other potential issues with input handling or rule triggering.

Test with Varied Input Combinations: Conduct a more thorough test with varied combinations of all inputs to ensure the FIS responds correctly across the entire input space.

Refinement and Iteration: Iterate on the FIS configuration, adjusting rules and membership functions, and reassess using the above analysis tools to measure improvements or identify further areas for enhancement.

# Adapting Angular Resolution to Fuzzy Models

### Wrap-around Membership Functions

To design wrap-around membership functions for a circular variable like angular resolution, I would define the membership functions so that they account for the circularity. For example, a "High" angular resolution could have a membership function that starts at 0.9, peaks at 1, wraps around to 0, and ends at 0.1.

% Example of a wrap-around membership function for "High" Angular Resolution

fis = addmf(fis, 'input', 1, 'High', 'trapmf', [-0.1 0 0.1 0.2]); % This MF starts at -0.1 to represent wrapping around to the high values

### Vector-based Approach

For the vector-based approach, I would transform the angle into its sine and cosine components and use these as inputs instead:

% As 'angle' is input in degrees

sin\_input = sind(angle); % Sine component

cos\_input = cosd(angle); % Cosine component

% Add these as inputs to FIS

fis = addvar(fis, 'input', 'Sin Component', [-1 1]);

fis = addvar(fis, 'input', 'Cos Component', [-1 1]);

% ... Define the rest of the FIS as usual

### Multiple Angles Input

In a similar vein to the vector-based approach, I can use the sine and cosine as separate inputs:

% FIS inputs will now be the sine and cosine values directly

% Use the same code as above for adding these new inputs

### Domain-specific Transformation

For the domain-specific transformation, I need to find or create a linear metric that represents the circular data. This could be a derived feature based on domain knowledge.

% Example of transforming angle to a domain-specific metric (e.g., quality score)

quality\_score = someFunctionOf(angle); % This function needs to be defined based on domain knowledge

% Add this quality score as an input to your FIS

fis = addvar(fis, 'input', 'Quality Score', [0 1]);

% ... Define the rest of the FIS as usual

## Code Appendices

### Type 1 Mamdani Code

%% To suppress deprecation warnings

warning('off','fuzzy:general:warnDeprecation\_Newfis')

warning('off','fuzzy:general:warnDeprecation\_Addvar')

warning('off','fuzzy:general:warnDeprecation\_Addmf')

warning('off','fuzzy:general:warnDeprecation\_Evalfis')

%% Clear the Command Window to remove previous runs' clutter

clc

%% Creating a new Fuzzy Inference System (FIS) with specified parameters

% AND OR Impl Agg Defuzzification

fis = newfis('DarkMatterDetection','mamdani','min','max', 'min','max','centroid');

%fis = newfis('DarkMatterDetection','mamdani','min','max', 'min','max','mom');

%fis = newfis('DarkMatterDetection','mamdani','min','max', 'min','max','lom');

%fis = newfis('DarkMatterDetection','mamdani','min','max', 'min','max','som');

%fis = newfis('DarkMatterDetection','mamdani','min','max', min','max','bisector');

% AND OR Impl Agg Defuzzification

%fis = newfis('DarkMatterDetection','mamdani','prod','probor', prod','max','centroid');

%fis = newfis('DarkMatterDetection','mamdani','prod','probor', 'prod','max','mom');

%fis = newfis('DarkMatterDetection','mamdani','prod','probor', 'prod','max','lom');

%fis = newfis('DarkMatterDetection','mamdani','prod','probor', 'prod','max','som');

%fis = newfis('DarkMatterDetection','mamdani','prod','probor', 'prod','max','bisector');

%% Adding 'Angular Resolution' as an input variable

fis = addvar(fis, 'input', 'Angular Resolution (milli-arcseconds)', [0 1]);

% Defining membership functions for 'Angular Resolution'

fis = addmf(fis, 'input', 1, 'Low', 'trapmf', [0 0 0.3 0.6]);

fis = addmf(fis, 'input', 1, 'Medium', 'trimf', [0.3 0.5 0.7]);

fis = addmf(fis, 'input', 1, 'High', 'trapmf', [0.4 0.7 1 1]);

%% Adding 'Density of the Gravitational Lens' as an input variable

fis = addvar(fis, 'input', 'Density', [0 1]);

% Defining membership functions for 'Density'

fis = addmf(fis, 'input', 2, 'Low', 'trapmf', [0 0 0.3 0.6]);

fis = addmf(fis, 'input', 2, 'Medium', 'trimf', [0.3 0.5 0.7]);

fis = addmf(fis, 'input', 2, 'High', 'trapmf', [0.4 0.7 1 1]);

%% Adding 'Width and Distribution of Gravitational Arcs' as an input variable

fis = addvar(fis, 'input', 'Width of Arcs', [0 1]);

% Defining membership functions for 'Width of Arcs'

fis = addmf(fis, 'input', 3, 'Narrow', 'trapmf', [0 0 0.3 0.5]);

fis = addmf(fis, 'input', 3, 'Average', 'trimf', [0.3 0.5 0.7]);

fis = addmf(fis, 'input', 3, 'Wide', 'trapmf', [0.5 0.7 1 1]);

%% Adding an output variable for the estimated mass of dark matter

fis = addvar(fis, 'output', 'Estimated Mass (eV)', [0 10^(-21)]);

% Defining membership functions for the output

fis = addmf(fis, 'output', 1, 'Light', 'trapmf', [0 0 2\*10^(-22) 5\*10^(-22)]);

fis = addmf(fis, 'output', 1, 'Medium', 'trimf', [2\*10^(-22) 5\*10^(-22) 8\*10^(-22)]);

fis = addmf(fis, 'output', 1, 'Heavy', 'trapmf', [5\*10^(-22) 8\*10^(-22) 10^(-21) 10^(-21)]);

% Full set of rules for the FIS

rules = [

% Antecedent (Input 1, Input 2, Input 3), Consequent (Output), Weight, Operator (AND=1)

1 1 1 1 1 1; % If AR is Low and Density is Low and Width is Narrow, then Mass is Light

1 1 2 1 1 1; % If AR is Low and Density is Low and Width is Medium, then Mass is Light

1 1 3 1 1 1; % If AR is Low and Density is Low and Width is Wide, then Mass is Light

1 2 1 2 1 1; % If AR is Low and Density is Medium and Width is Narrow, then Mass is Medium

1 2 2 2 1 1; % If AR is Low and Density is Medium and Width is Medium, then Mass is Medium

1 2 3 1 1 1; % If AR is Low and Density is Medium and Width is Wide, then Mass is Light

1 3 1 3 1 1; % If AR is Low and Density is High and Width is Narrow, then Mass is Heavy

1 3 2 2 1 1; % If AR is Low and Density is High and Width is Medium, then Mass is Medium

1 3 3 2 1 1; % If AR is Low and Density is High and Width is Wide, then Mass is Medium

2 1 1 1 1 1; % If AR is Medium and Density is Low and Width is Narrow, then Mass is Light

2 1 2 1 1 1; % If AR is Medium and Density is Low and Width is Medium, then Mass is Light

2 1 3 1 1 1; % If AR is Medium and Density is Low and Width is Wide, then Mass is Light

2 2 1 2 1 1; % If AR is Medium and Density is Medium and Width is Narrow, then Mass is Medium

2 2 2 2 1 1; % If AR is Medium and Density is Medium and Width is Medium, then Mass is Medium

2 2 3 2 1 1; % If AR is Medium and Density is Medium and Width is Wide, then Mass is Medium

2 3 1 3 1 1; % If AR is Medium and Density is High and Width is Narrow, then Mass is Heavy

2 3 2 2 1 1; % If AR is Medium and Density is High and Width is Medium, then Mass is Medium

2 3 3 3 1 1; % If AR is Medium and Density is High and Width is Wide, then Mass is Heavy

3 1 1 2 1 1; % If AR is High and Density is Low and Width is Narrow, then Mass is Medium

3 1 2 1 1 1; % If AR is High and Density is Low and Width is Medium, then Mass is Light

3 1 3 1 1 1; % If AR is High and Density is Low and Width is Wide, then Mass is Light

3 2 1 3 1 1; % If AR is High and Density is Medium and Width is Narrow, then Mass is Heavy

3 2 2 2 1 1; % If AR is High and Density is Medium and Width is Medium, then Mass is Medium

3 2 3 2 1 1; % If AR is High and Density is Medium and Width is Wide, then Mass is Medium

3 3 1 3 1 1; % If AR is High and Density is High and Width is Narrow, then Mass is Heavy

3 3 2 3 1 1; % If AR is High and Density is High and Width is Medium, then Mass is Heavy

3 3 3 3 1 1; % If AR is High and Density is High and Width is Wide, then Mass is Heavy

];

% Add rules to the FIS

fis = addrule(fis, rules);

%% Read, Normalise and save data

% Define the file and sheet name

dataFile = 'DarkMatterData.xlsx';

sheetName = 'Sheet1';

% Read the normalized data directly from the specified columns

normalisedData = readmatrix(dataFile, 'Sheet', sheetName, 'Range', 'B3:F52');

% Debug: Display the size of the normalisedData matrix

disp('Size of normalisedData matrix:');

disp(size(normalisedData));

% Extract only the normalised columns, which are columns 1, 3, and 5 of the normalisedData matrix

inputData = normalisedData(:, [1, 3, 5]);

% Debug: Display the inputData to ensure its correct

disp('Input data to be used:');

disp(inputData);

%% Defuzz Method

% Now inputData contains normalised values that you can pass to your FIS

evaluateAndWriteFIS(fis, inputData, dataFile, sheetName, 'H3');

%% Visualising the system - for a detailed analysis and debugging

figure('Name','Input 1 - Angular Resolution');

subplot(3,1,1), plotmf(fis, 'input', 1);

title('Angular Resolution Membership Functions');

subplot(3,1,2), plotmf(fis, 'input', 2);

title('Density Membership Functions');

subplot(3,1,3), plotmf(fis, 'input', 3);

title('Width of Arcs Membership Functions');

figure('Name','Output - Estimated Mass');

subplot(1,1,1), plotmf(fis, 'output', 1);

title('Estimated Mass Membership Functions');

%% Surf Plot

% Define the fixed width value

fixedWidthValue = 0.5; % adjust as necessary [0 1]

% Call the generateSurfPlot function

generateSurfPlot(fis, fixedWidthValue);

%% Rule Viewer Plot

ruleview(fis);

%% Sensitivity Analysis:

% Define ranges for each input variable with appropriate resolution

arRange = linspace(0, 0.4, 100); % Will generate 100 points

densityRange = linspace(0, 0.3, 1); % Assuming constant for analysis

widthRange = linspace(0, 0.2, 1); % Assuming constant for analysis

% Keeping density and width constant at their median values

densityMedian = median(densityRange);

widthMedian = median(widthRange);

% Evaluate the FIS output while varying Angular Resolution only

% Will evaluate the FIS output while changing this to other methods to fully test % the system

output = zeros(size(arRange)); % Pre-allocate for performance

for i = 1:length(arRange)

output(i) = evalfis([arRange(i), densityMedian, widthMedian], fis);

end

% Plot the results

figure;

plot(arRange, output);

xlabel('Angular Resolution (normalised)');

ylabel('FIS Output (eV)');

title('Sensitivity Analysis for Angular Resolution');

%% Principal Component Analysis (PCA):

% Prepare:

% Define the minimum and maximum values for each input

minAngularResolution = 0;

maxAngularResolution = 1;

minDensity = 0;

maxDensity = 1;

minWidth = 0;

maxWidth = 1;

% Define the range for each input

numSamples = 100; % For example, 100 samples for each input

angularResolutionRange = linspace(minAngularResolution, maxAngularResolution, numSamples);

densityRange = linspace(minDensity, maxDensity, numSamples);

widthRange = linspace(minWidth, maxWidth, numSamples);

% Generate a grid of input combinations

[ARgrid, DensityGrid, WidthGrid] = ndgrid(angularResolutionRange, densityRange, widthRange);

% Flatten the grids into vectors for evalfis

ARvector = ARgrid(:);

DensityVector = DensityGrid(:);

WidthVector = WidthGrid(:);

% Combine into a single matrix for evalfis

inputMatrix = [ARvector, DensityVector, WidthVector];

% Evaluate the FIS for each combination of inputs

outputVector = evalfis(fis, inputMatrix);

% Specify the file and sheet name for reading normalized data

dataFile = 'DarkMatterData.xlsx';

sheetName = 'Sheet1';

% Read the normalised data from specified columns and rows

normalisedData = readmatrix(dataFile, 'Sheet', sheetName, 'Range', 'B3:F52', 'UseExcel', false); % 'H3:L52'

% Extract columns B, D, and F only for PCA

inputDataForPCA = normalisedData(:, [1, 3, 5]); % 1, 2, 3, 4, 5

% Execute:

% where the last column is the output

% coeff contains the coefficients for each principal component

% score contains the observations projected into the principal component space

% latent contains the variance explained by each principal component

[coeff, score, latent] = pca(inputDataForPCA);

% Plot the variance explained by each principal component

figure;

explainedVar = cumsum(latent)./sum(latent) \* 100;

bar(explainedVar);

xlabel('Principal Components');

ylabel('Variance Explained (%)');

title('PCA Variance Explained');