FUZZY INFERENCE

Siti Zaiton Mohd Hashim, PhD
Fuzzy Inference

- Introduction
- Mamdani-style inference
- Sugeno-style inference
- Building a fuzzy expert system
Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy theory of fuzzy sets.

The process of fuzzy inference:
- membership functions,
- fuzzy logic operators, and
- fuzzy if-then rules.

Two known types of fuzzy inference systems (in the Fuzzy Logic Toolbox):
- Mamdani-type
- Sugeno-type.
Introduction

- Fuzzy inference systems have been successfully applied in fields:
  - automatic control,
  - data classification,
  - decision analysis,
  - expert systems, and
  - computer vision.
Because of its multidisciplinary nature, fuzzy inference systems are associated with a number of names:

- fuzzy-rule-based systems,
- fuzzy expert systems,
- fuzzy modeling,
- fuzzy logic controllers, and
- simply (and ambiguously) fuzzy systems.
Mamdani’s Fuzzy Inference

- Mamdani's fuzzy inference method is the most commonly seen fuzzy methodology and was among the first control systems built using fuzzy set theory.

- It was proposed in 1975 by Ebrahim Mamdani [Mam75] as an attempt to control a steam engine and boiler combination by synthesizing a set of linguistic control rules obtained from experienced human operators.

- Mamdani's effort was based on Lotfi Zadeh's 1973 paper on fuzzy algorithms for complex systems and decision processes [Zad73].
Mamdani’s Fuzzy Inference

- Mamdani-type inference process is performed in four steps:
  1. Fuzzification of the input and output variables
  2. Fuzzy logical operations
  3. Implication method
  4. Aggregation
  5. Defuzzification

- Mamdani-type inference expects the output membership functions to be fuzzy sets.
Sample problem

We examine a simple two-input one-output problem that includes three rules:

**Rule: 1**

IF \( x \) is \( A_3 \)
OR \( y \) is \( B_1 \)
THEN \( z \) is \( C_1 \)

**Rule: 2**

IF \( x \) is \( A_2 \)
AND \( y \) is \( B_2 \)
THEN \( z \) is \( C_2 \)

**Rule: 3**

IF \( x \) is \( A_1 \)
THEN \( z \) is \( C_3 \)

**Rule: 1**

IF \( \) project_funding is adequate
OR \( \) project_staffing is small
THEN risk is low

**Rule: 2**

IF \( \) project_funding is marginal
AND \( \) project_staffing is large
THEN risk is normal

**Rule: 3**

IF \( \) project_funding is inadequate
THEN risk is high
Fuzzification

The first step is to take the crisp inputs, \( x_1 \) and \( y_1 \) (project funding and project staffing), and determine the degree to which these inputs belong to each of the appropriate fuzzy sets.

\[
\begin{align*}
\mu(x = A_1) &= 0.5 \\
\mu(x = A_2) &= 0.2 \\
\mu(y = B_1) &= 0.1 \\
\mu(y = B_2) &= 0.7
\end{align*}
\]
After fuzzification, we know the degree to which each part of the antecedent has been satisfied for each rule.

If the antecedent of a given rule has more than one part, the **fuzzy operator** is applied to obtain one number that represents the result of the antecedent for that rule.

This number will then be applied to the output function.

- The input to the fuzzy operator is two or more membership values from fuzzified input variables.
- The output is a single truth value.
As an example, take the fuzzified inputs, $\mu_{(x=A1)} = 0.5$, $\mu_{(x=A2)} = 0.2$, $\mu_{(y=B1)} = 0.1$ and $\mu_{(y=B2)} = 0.7$, and apply them to the antecedents of the fuzzy rules.

If a given fuzzy rule has multiple antecedents, the fuzzy operator (AND or OR) is used to obtain a single number that represents the result of the antecedent evaluation. This number (the truth value) is then applied to the consequent membership function.
To evaluate the **disjunction** of the rule antecedents, we use the **OR** fuzzy operation. Typically, fuzzy expert systems make use of the classical fuzzy operation **union**:

\[ \mu_A \cup_B (x) = \text{max} \ [\mu_A(x), \mu_B(x)] \]

Similarly, in order to evaluate the conjunction of the rule antecedents, we apply the **AND** fuzzy operation **intersection**:

\[ \mu_A \cap_B (x) = \text{min} \ [\mu_A(x), \mu_B(x)] \]
**Mamdani-style RULE EVALUATION**

### Rule 1:

**IF**  
- **x** is **A3** (0.0)  
- **y** is **B1** (0.1)  
**THEN**  
- **z** is **C1** (0.1)

### Rule 2:

**IF**  
- **x** is **A2** (0.2)  
- **y** is **B2** (0.7)  
**THEN**  
- **z** is **C2** (0.2)

### Rule 3:

**IF**  
- **x** is **A1** (0.5)  
**THEN**  
- **z** is **C3** (0.5)
Implication method is the process to determine the output of each fuzzy rule’s consequent.

Before applying the implication method, we must take care of the rule's weight.

Every rule has a weight (a number between 0 and 1), which is applied to the number given by the antecedent. Generally this weight is 1 (as it is for this example) and so it has no effect at all on the implication process. From time to time you may want to weight one rule relative to the others by changing its weight value to something other than 1.
Implication method is the process of reshaping the consequent part of each fuzzy rules.

The consequent is reshaped using a function associated with the antecedent (a single number).

Before applying the implication method, we must take care of the rule's weight.

- Every rule has a weight (a number between 0 and 1), which is applied to the number given by the antecedent.
- Generally this weight is 1 (as it is for this example) and so it has no effect at all on the implication process.
A consequent is a fuzzy set represented by a membership function, which weights appropriately the linguistic characteristics that are attributed to it.

The implication process is implemented for each rule:

- The input: a single number given by the antecedent
- The output: fuzzy set.

Two built-in methods are supported:

- $\text{min}$ (minimum), which truncates the output fuzzy set
- $\text{prod}$ (product), which scales the output fuzzy set.
Now the result of the antecedent evaluation can be applied to the membership function of the consequent.

The most common method of correlating the rule consequent with the truth value of the rule antecedent is to cut the consequent membership function at the level of the antecedent truth. This method is called **clipping**. Since the top of the membership function is sliced, the clipped fuzzy set loses some information. However, clipping is still often preferred because it involves less complex and faster mathematics, and generates an aggregated output surface that is easier to defuzzify.
While clipping is a frequently used method, **scaling** offers a better approach for **preserving** the original shape of the fuzzy set. The original membership function of the rule consequent is adjusted by multiplying all its membership degrees by the truth value of the rule antecedent. This method, which generally loses less information, can be very useful in fuzzy expert systems.

\[(i.e. \ 0.2 \times (x_1=10), \ 0.2\times(y_1=30)) = 2,6\]
Clipped and scaled membership functions

Degree of Membership

1.0

0.2

0.0

Z

C2

Degree of Membership

1.0

0.2

0.0

Z

C2
Aggregation of the rule outputs

Aggregation is the process of unification of the outputs of all rules. We take the membership functions of all rule consequents previously clipped or scaled and combine them into a single fuzzy set.

The input of the aggregation process is the list of clipped or scaled consequent membership functions, and the output is one fuzzy set for each output variable.
Aggregation of the rule outputs

- z is C1 (0.1)
- z is C2 (0.2)
- z is C3 (0.5)

\[ z \] is \( \sum \)
Defuzzification

The last step in the fuzzy inference process is defuzzification. Fuzziness helps us to evaluate the rules, but the **final output of a fuzzy system has to be a crisp number.** The input for the defuzzification process is the aggregate output fuzzy set and the output is **a single number.**
There are several defuzzification methods, but probably the most popular one is the **centroid technique**. It finds the point where a vertical line would slice the aggregate set into two equal masses. Mathematically this **centre of gravity (COG)** can be expressed as:

\[
COG = \frac{\int_{a}^{b} \mu_{A}(x) x \, dx}{\int_{a}^{b} \mu_{A}(x) \, dx}
\]
Centre of gravity (COG):

- Centroid defuzzification method finds a point representing the centre of gravity of the fuzzy set, \( A \), on the interval, \( ab \).
- A reasonable estimate can be obtained by calculating it over a sample of points.
Centre of gravity (COG):

\[
COG = \frac{(0 + 10 + 20) \times 0.1 + (30 + 40 + 50 + 60) \times 0.2 + (70 + 80 + 90 + 100) \times 0.5}{0.1 + 0.1 + 0.1 + 0.2 + 0.2 + 0.2 + 0.5 + 0.5 + 0.5} = 67.4
\]
Mamdani-style inference, as we have just seen, requires us to find the centroid of a two-dimensional shape by integrating across a continuously varying function. In general, this process is not computationally efficient.

Michio Sugeno suggested to use a single spike, a singleton, as the membership function of the rule consequent. A singleton, or more precisely a fuzzy singleton, is a fuzzy set with a membership function that is unity at a single particular point on the universe of discourse and zero everywhere else.
Sugeno-style fuzzy inference is very similar to the Mamdani method. Sugeno changed only a rule consequent. Instead of a fuzzy set, he used a mathematical function of the input variable. The format of the Sugeno-style fuzzy rule is

\[
\text{IF } x \text{ is } A \quad \text{AND} \quad y \text{ is } B \quad \text{THEN} \quad z \text{ is } f(x, y)
\]

where \(x\), \(y\) and \(z\) are linguistic variables; \(A\) and \(B\) are fuzzy sets on universe of discourses \(X\) and \(Y\), respectively; and \(f(x, y)\) is a mathematical function.
Sugeno fuzzy inference

The most commonly used zero-order Sugeno fuzzy model applies fuzzy rules in the following form:

IF $x$ is $A$
AND $y$ is $B$
THEN $z$ is $k$

where $k$ is a constant.

In this case, the output of each fuzzy rule is constant. All consequent membership functions are represented by singleton spikes.
**Sugeno-style rule evaluation**

**Rule 1:** IF $x$ is $A_3$ (0.0) \ OR \ $y$ is $B_1$ (0.1) \ THEN \ $z$ is $k_1$ (0.1)

**Rule 2:** IF $x$ is $A_2$ (0.2) \ AND \ $y$ is $B_2$ (0.7) \ THEN \ $z$ is $k_2$ (0.2)

**Rule 3:** IF $x$ is $A_1$ (0.5) \ THEN \ $z$ is $k_3$ (0.5)
Sugeno-style aggregation of the rule outputs

\[ z \text{ is } k_1 (0.1) \rightarrow z \text{ is } k_2 (0.2) \rightarrow z \text{ is } k_3 (0.5) \rightarrow \sum \]
Weighted average (WA):

\[ WA = \frac{\mu(k1) \times k1 + \mu(k2) \times k2 + \mu(k3) \times k3}{\mu(k1) + \mu(k2) + \mu(k3)} = \frac{0.1 \times 20 + 0.2 \times 50 + 0.5 \times 80}{0.1 + 0.2 + 0.5} = 65 \]

Sugeno-style defuzzification
How to make a decision on which method to apply – Mamdani or Sugeno?

- Mamdani method is widely accepted for capturing expert knowledge. It allows us to describe the expertise in more intuitive, more human-like manner. However, Mamdani-type fuzzy inference entails a substantial computational burden.

- On the other hand, Sugeno method is computationally effective and works well with optimisation and adaptive techniques, which makes it very attractive in control problems, particularly for dynamic nonlinear systems.
A service centre keeps spare parts and repairs failed ones.

A customer brings a failed item and receives a spare of the same type.

Failed parts are repaired, placed on the shelf, and thus become spares.

The objective here is to advise a manager of the service centre on certain decision policies to keep the customers satisfied.
Process of developing an fuzzy expert system

1. Specify the problem and define linguistic variables.
2. Determine fuzzy sets (for fuzzification process).
3. Elicit and construct fuzzy rules.
4. Encode the fuzzy sets, fuzzy rules and procedures to perform fuzzy inference into the expert system.
5. Evaluate and tune the system.
Step 1: Specify the problem and define linguistic variables

There are four main linguistic variables: average waiting time (mean delay) $m$, repair utilisation factor of the service centre $\rho$, number of servers $s$, and initial number of spare parts $n$.

Input~ $m, s$ and $\rho$
Output~ $n$
### Linguistic variables and their ranges

#### Linguistic Variable: *Mean Delay, m*

<table>
<thead>
<tr>
<th>Linguistic Value</th>
<th>Notation</th>
<th>Numerical Range (normalised)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Short</td>
<td>VS</td>
<td>[0, 0.3]</td>
</tr>
<tr>
<td>Short</td>
<td>S</td>
<td>[0.1, 0.5]</td>
</tr>
<tr>
<td>Medium</td>
<td>M</td>
<td>[0.4, 0.7]</td>
</tr>
</tbody>
</table>

#### Linguistic Variable: *Number of Servers, s*

<table>
<thead>
<tr>
<th>Linguistic Value</th>
<th>Notation</th>
<th>Numerical Range (normalised)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>S</td>
<td>[0, 0.35]</td>
</tr>
<tr>
<td>Medium</td>
<td>M</td>
<td>[0.30, 0.70]</td>
</tr>
<tr>
<td>Large</td>
<td>L</td>
<td>[0.60, 1]</td>
</tr>
</tbody>
</table>

#### Linguistic Variable: *Repair Utilisation Factor, ρ*

<table>
<thead>
<tr>
<th>Linguistic Value</th>
<th>Notation</th>
<th>Numerical Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>L</td>
<td>[0, 0.6]</td>
</tr>
<tr>
<td>Medium</td>
<td>M</td>
<td>[0.4, 0.8]</td>
</tr>
<tr>
<td>High</td>
<td>H</td>
<td>[0.6, 1]</td>
</tr>
</tbody>
</table>

#### Linguistic Variable: *Number of Spares, n*

<table>
<thead>
<tr>
<th>Linguistic Value</th>
<th>Notation</th>
<th>Numerical Range (normalised)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Small</td>
<td>VS</td>
<td>[0, 0.30]</td>
</tr>
<tr>
<td>Small</td>
<td>S</td>
<td>[0, 0.40]</td>
</tr>
<tr>
<td>Rather Small</td>
<td>RS</td>
<td>[0.25, 0.45]</td>
</tr>
<tr>
<td>Medium</td>
<td>M</td>
<td>[0.30, 0.70]</td>
</tr>
<tr>
<td>Rather Large</td>
<td>RL</td>
<td>[0.55, 0.75]</td>
</tr>
<tr>
<td>Large</td>
<td>L</td>
<td>[0.60, 1]</td>
</tr>
<tr>
<td>Very Large</td>
<td>VL</td>
<td>[0.70, 1]</td>
</tr>
</tbody>
</table>
**Step 2: Determine fuzzy sets**

Fuzzy sets can have a variety of shapes. However, a triangle or a trapezoid can often provide an adequate representation of the expert knowledge, and at the same time, significantly simplifies the process of computation.
Fuzzy sets of Mean Delay $m$

![Graph showing fuzzy sets for Mean Delay $m$]
Fuzzy sets of Number of Servers $s$

![Diagram showing fuzzy sets for Number of Servers](image_url)
Fuzzy sets of Repair Utilisation Factor $\rho$

Degree of Membership

Repair Utilisation Factor

- L
- M
- H
Fuzzy sets of Number of Spares $n$

![Diagram showing fuzzy sets for different levels of spares](image)

- VS
- S
- RS
- M
- RL
- L
- VL

Degree of Membership vs. Number of Spares (normalised)
Step 3: Elicit and construct fuzzy rules

To accomplish this task, we might ask the expert to describe how the problem can be solved using the fuzzy linguistic variables defined previously.

Required knowledge also can be collected from other sources such as books, computer databases, flow diagrams and observed human behaviour.
The square FAM representation
<table>
<thead>
<tr>
<th>Rule</th>
<th>$m$</th>
<th>$s$</th>
<th>$\rho$</th>
<th>$n$</th>
<th>Rule</th>
<th>$m$</th>
<th>$s$</th>
<th>$\rho$</th>
<th>$n$</th>
<th>Rule</th>
<th>$m$</th>
<th>$s$</th>
<th>$\rho$</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VS</td>
<td>S</td>
<td>L</td>
<td>VS</td>
<td>10</td>
<td>VS</td>
<td>S</td>
<td>M</td>
<td>S</td>
<td>19</td>
<td>VS</td>
<td>S</td>
<td>H</td>
<td>VL</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>S</td>
<td>L</td>
<td>VS</td>
<td>11</td>
<td>S</td>
<td>S</td>
<td>M</td>
<td>VS</td>
<td>20</td>
<td>S</td>
<td>S</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>S</td>
<td>L</td>
<td>VS</td>
<td>12</td>
<td>M</td>
<td>S</td>
<td>M</td>
<td>VS</td>
<td>21</td>
<td>M</td>
<td>S</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>VS</td>
<td>M</td>
<td>L</td>
<td>VS</td>
<td>13</td>
<td>VS</td>
<td>M</td>
<td>M</td>
<td>RS</td>
<td>22</td>
<td>VS</td>
<td>M</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>5</td>
<td>S</td>
<td>M</td>
<td>L</td>
<td>VS</td>
<td>14</td>
<td>S</td>
<td>M</td>
<td>M</td>
<td>S</td>
<td>23</td>
<td>S</td>
<td>M</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>VS</td>
<td>15</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>VS</td>
<td>24</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>VS</td>
<td>L</td>
<td>L</td>
<td>S</td>
<td>16</td>
<td>VS</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>25</td>
<td>VS</td>
<td>L</td>
<td>H</td>
<td>RL</td>
</tr>
<tr>
<td>8</td>
<td>S</td>
<td>L</td>
<td>L</td>
<td>S</td>
<td>17</td>
<td>S</td>
<td>L</td>
<td>M</td>
<td>RS</td>
<td>26</td>
<td>S</td>
<td>L</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>9</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>VS</td>
<td>18</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>S</td>
<td>27</td>
<td>M</td>
<td>L</td>
<td>H</td>
<td>RS</td>
</tr>
</tbody>
</table>
1. If (utilisation_factor is L) then (number_of_spares is S)
2. If (utilisation_factor is M) then (number_of_spares is M)
3. If (utilisation_factor is H) then (number_of_spares is L)
4. If (mean_delay is VS) and (number_of_servers is S) then (number_of_spares is VL)
5. If (mean_delay is S) and (number_of_servers is S) then (number_of_spares is L)
6. If (mean_delay is M) and (number_of_servers is S) then (number_of_spares is M)
7. If (mean_delay is VS) and (number_of_servers is M) then (number_of_spares is RL)
8. If (mean_delay is S) and (number_of_servers is M) then (number_of_spares is RS)
9. If (mean_delay is M) and (number_of_servers is M) then (number_of_spares is S)
10. If (mean_delay is VS) and (number_of_servers is L) then (number_of_spares is M)
11. If (mean_delay is S) and (number_of_servers is L) then (number_of_spares is S)
12. If (mean_delay is M) and (number_of_servers is L) then (number_of_spares is VS)
Cube FAM of Rule Base 2
Step 4: Encode the fuzzy sets, fuzzy rules and procedures to perform fuzzy inference into the expert system

To accomplish this task, we may choose one of two options: to build our system using a programming language such as C/C++ or Pascal, or to apply a fuzzy logic development tool such as MATLAB Fuzzy Logic Toolbox or Fuzzy Knowledge Builder.
Step 5: Evaluate and tune the system

The last, and the most laborious, task is to evaluate and tune the system. We want to see whether our fuzzy system meets the requirements specified at the beginning.

Several test situations depend on the mean delay, number of servers and repair utilisation factor.

The Fuzzy Logic Toolbox can generate surface to help us analyse the system’s performance.
Three-dimensional plots for Rule Base 1

mean_delay

number_of_servers

number_of_spares

mean_delay

number_of_servers

0.2
0.4
0.6
0.2
0.4
0.6
0.8
1.0

0.6
0.5
0.4
0.3
0.2
0.1
0.0

0.2
0.3
0.4
0.5
0.6
Three-dimensional plots for Rule Base 1

Three-dimensional plots for Rule Base 1
Three-dimensional plots for Rule Base 2

The diagram shows the relationship between three variables:
- Number of servers
- Mean delay
- Number of spares

The X-axis represents the number of servers, the Y-axis represents the mean delay, and the Z-axis represents the number of spares.
Three-dimensional plots for Rule Base 2

- number_of_spares
- mean_delay
- utilisation_factor
However, even now, the expert might not be satisfied with the system performance.

To improve the system performance, we may use additional sets — *Rather Small* and *Rather Large* — on the universe of discourse *Number of Servers*, and then extend the rule base.
Modified fuzzy sets of Number of Servers $s$
Cube FAM of Rule Base 3
Three-dimensional plots for Rule Base 3

number_of_servers

mean_delay

number_of_spares
Three-dimensional plots for Rule Base 3
Tuning fuzzy systems

1. Review model input and output variables, and if required redefine their ranges.

2. Review the fuzzy sets, and if required define additional sets on the universe of discourse. The use of wide fuzzy sets may cause the fuzzy system to perform roughly.

3. Provide sufficient overlap between neighbouring sets. It is suggested that triangle-to-triangle and trapezoid-to-triangle fuzzy sets should overlap between 25% to 50% of their bases.
4. Review the existing rules, and if required add new rules to the rule base.

5. Examine the rule base for opportunities to write hedge rules to capture the pathological behaviour of the system.

6. Adjust the rule execution weights. Most fuzzy logic tools allow control of the importance of rules by changing a weight multiplier.

7. Revise shapes of the fuzzy sets. In most cases, fuzzy systems are highly tolerant of a shape approximation.