Fuzzy systems

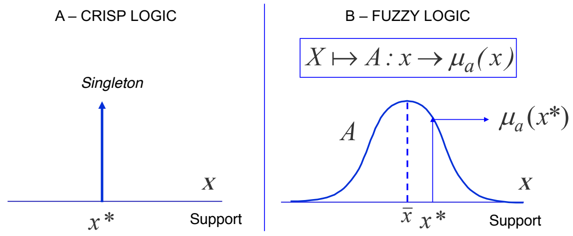
Simulation of a fuzzy system

Michela Mulas

Important concepts - A recap

In fuzzy logic the boundary between true and false becomes blurred:

- A concept can be anything between the extremes of true or false, with its **degree of truth varying smoothly from zero (totally false) to one (totally true)**.
- A concept is hardly ever completely true or completely false, but it is rather somewhere in between these two extremes.



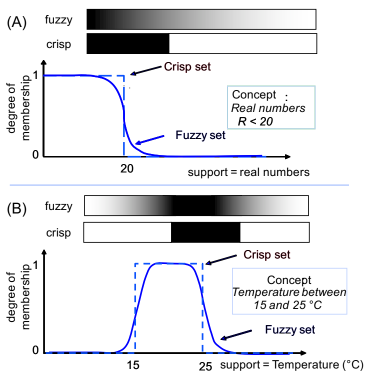
Important concepts - A recap

Consider the statements:

A. Real numbers smaller than 20
B. Temperature between 15 and 25 °C

in the crisp and fuzzy contexts.

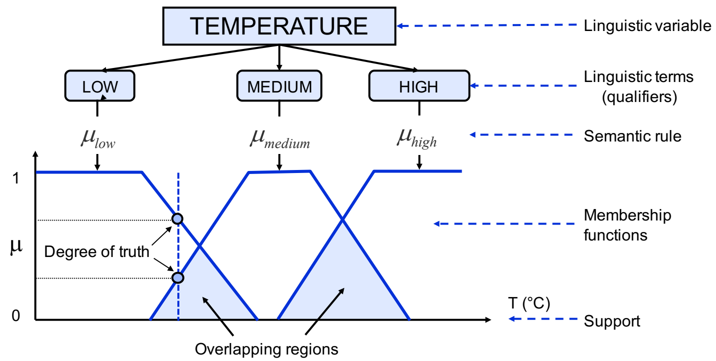
- The crisp logic makes a hard distinction by classifying each entry as either true (black) or false (white)
- The fuzzy logic ranks the truth of each statement, depending on the relative similarity with the reference concept.



In broad terms, Fuzzy Logic could be defined as an **approximate reasoning method dealing with vaguely defined concepts**.

Important concepts - A recap

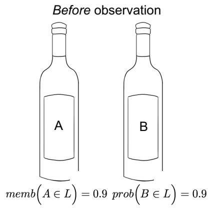
Example: A concept like "Temperature" can be defined using linguistic terms (Low, Medium, High) and translated into precise mathematical terms through a set of suitably selected reference membership functions.



Important concepts - A recap

Fuzziness Vs. Probability: Suppose that you, or the “weary traveller” in Bezdek (1993)[1] example, are confronted with two bottles containing an unknown liquid.

- A is labelled “Drinkable with fuzziness = 0.9”
- The label on B reads “Drinkable with probability = 0.9”.
- **Which bottle would you drink from?**



In glass A there is for sure some liquid that resembles drinking water to a high degree.

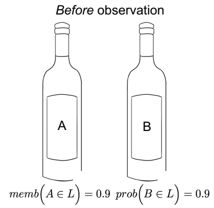
- The statement “Drinkable with fuzziness = 0.9” means that the liquid inside has been tested and found drinkable to a high degree.

[1] Bezdek, IEEE Trans. Fuzzy Syst., vol. 1, 1993

Important concepts - A recap

Fuzziness Vs. Probability: Suppose that you, or the “weary traveller” in Bezdek (1993)[1] example, are confronted with two bottles containing an unknown liquid.

- A is labelled “Drinkable with fuzziness = 0.9”
- The label on B reads “Drinkable with probability = 0.9”.
- **Which bottle would you drink from?**



The label on glass B sounds a bit mysterious.

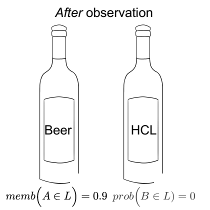
- It says that the bottle contains some liquid that over a large number of trials of similar bottles proved to be drinkable in 90% of the cases.
- There is one chance out of ten that it could be anything from dreadful to poisonous.

[1] Bezdek, IEEE Trans. Fuzzy Syst., vol. 1, 1993

Important concepts - A recap

Fuzziness Vs. Probability: Suppose that you, or the “weary traveller” in Bezdek (1993)[1] example, are confronted with two bottles containing an unknown liquid.

- A is labelled “Drinkable with fuzziness = 0.9”
- The label on B reads “Drinkable with probability = 0.9”.
- **Which bottle would you drink from?**



Bezdek concludes that it is advisable to drink from the bottle A.

- **Fuzziness identifies a certain a-posteriori quality, whereas probability represents an uncertain a priori expectation**.

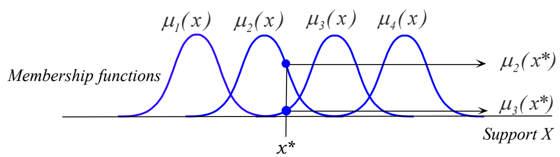
[1] Bezdek, IEEE Trans. Fuzzy Syst., vol. 1, 1993

Important concepts - A recap

Fuzzification

In order to enter the fuzzy world a concept needs to be fuzzified: its value or its semantics must be converted from absolute to relative by comparison to a set of reference concepts, mathematically represented by μ.

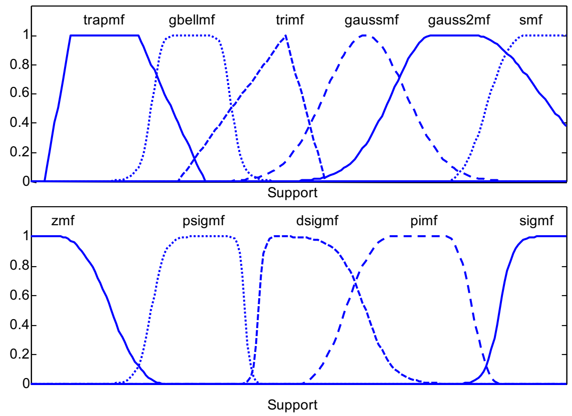
$$x^* \xrightarrow{\text{Fuzzification}} \left[0\ \mu_2(x^*)\ \mu_3(x^*)\ 0\right]$$



Fuzzification implies **substituting a vector of degrees of membership** in place of the original number or concept.

Important concepts - A recap

Analytical forms of the membership functions



Important concepts - A recap

Basic fuzzy set properties

Normality: a fuzzy set is said to be normal if there exists at least one element whose membership is equal to one. $\exists x | \mu(x) = 1$

Height: largest membership grade of any element in the set A. $h(A) = \max_x \mu(x)$

Support: the crisp subset X of elements in the universe of discourse for which all μs are nonzero. $Supp(A) = \{x | \mu(x) > 0 \text{ and } x \in X\}$

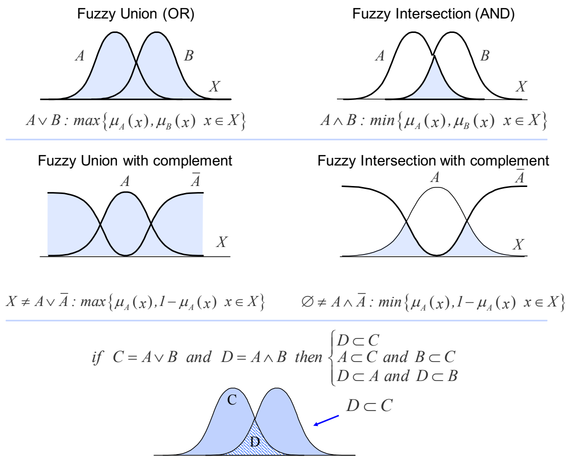
Core: the crisp subset of A containing the elements with membership equal to one. $Core(A) = \{x | \mu(x) = 1 \text{ and } x \in X\}$

Cardinality: extends the notion of conventional sets. For fuzzy sets with n elements it is defined as the sum of membership grades. $Card(A) = \sum_{i=1}^{n} \mu_A(x_i)$

α-cut: subset of A formed by the elements whose membership is greater than α $A_\alpha = \{x \in X | \mu_A(x) \geq \alpha\}$

Important concepts - A recap

Logical operations in fuzzy sets



Important concepts - A recap

Building a fuzzy inference system

An automated **fuzzy inference system** (FIS) is capable of deducing a logical result on the basis of some given premises.

- The deductive logic is based on the **modus ponens rule of inference**.
- It represents a widely accepted method for the construction of deductive reasoning, whereby given **a premise and a consequent**, the truth of the latter is determined by the truth of the former.
- In crisp logic the antecedent may only be either true or false, thus implying that the consequent is also either true or false.
- In fuzzy logic things get a lot more interesting because the **degree of truth** of the consequent is determined (actually upper-limited) from the degree of truth of the antecedent, of which it cannot be greater.

In other words, the consequent cannot be “truer” than its premises.

Important concepts - A recap

Building a fuzzy inference system

In strictly logical terms, given a (fuzzy) antecedent $x \in X$ and a (fuzzy) consequent $y \in Y$ the basic inferential logic rule R can be stated as:

$$R: \texttt{If}\ \left(x\ \texttt{is}\ A\right)\ \texttt{Then}\ \left(y\ \texttt{is}\ B\right),\quad R: X \times Y \rightarrow [0,1]$$

- A and B are fuzzy sets defined by their membership functions, so that the linguistic statements $\left(x\ \texttt{is}\ A\right)$ and $\left(y\ \texttt{is}\ B\right)$ actually produce the degree of membership $\mu_A(x)$ and $\mu_B(x)$.
- The implication R operates over the Cartesian Product of the two spaces (premise X) and (consequent Y).
- The implication `Then` is normally implemented with a T-norm: a logical `and` ($\wedge$).

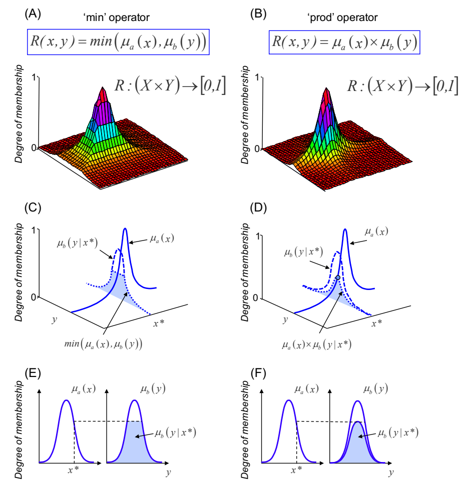
$$R: \mu_A(x^*) \wedge \mu_B(y) \rightarrow \begin{cases} \min\left(\mu_A(x), \mu_B(x)\right) \\ \mu_A \times \mu_B \end{cases}$$

Important concepts - A recap

Building a fuzzy inference system

Three different aspects of the fuzzy implication for the `min` and `prod` operations are:

- A and B: implication surface resulting from the cartesian product.
- C and D: the whole antecedent domain can be reduced to the single x^*.
- E and F: same results with independently plotted supported x and y.



Important concepts - A recap

Building a fuzzy inference system

The overall degree of truth of the premises determines the degree of truth of the consequent. For example:

$$\textbf{If}\ \left(x_1 \text{ is } A_1\right)\ \textbf{and}\ \left(x_2 \text{ is } A_2\right)\ \textbf{or}\ \left(x_3 \text{ is not } A_3\right).$$

whose computational version in terms of membership function is:

$$\mu_{a_1}(x_1) \underbrace{\wedge}_{\texttt{and}} \mu_{a2}(x_2) \underbrace{\vee}_{\texttt{or}} \underbrace{\left[1 - \mu_{a_3}(x_3)\right]}_{\texttt{not}}$$

The resulting degree of truth determines the highest degree of truth of the consequent. Thus, the general ith rule R_i would look like

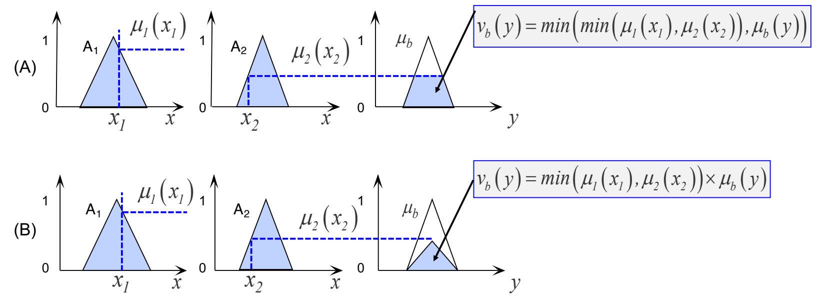
$$R_i: \texttt{If}\ \underbrace{(x_1 \text{ is } A_{1,i})\ \texttt{and}\ (x_2 \text{ is } A_{2,i})\ \texttt{and}\ \ldots \texttt{and}\ (x_n \text{ is } A_{n,i})}_{\textit{Premise}}\ \underbrace{\texttt{then}\ y\ \texttt{is}\ B_i}_{\textit{Consequent}}$$

Important concepts - A recap

Building a fuzzy inference system

A graphical representation of a rule with two premises (x_1 and x_2) and a triangular membership function.

$$\boxed{if\ (x_1\ is\ A_1)\ and\ (x_2\ is\ A_2)\ then\ (y\ is\ B)\ \Rightarrow\ v_b(y) = (\mu_1(x_1) \wedge \mu_2(x_2)) \wedge \mu_b(y)}$$



Important concepts - A recap

Building a fuzzy inference system

Fuzzy reasoning is based on a set of implications (rules) where each consequent output is determined by a certain combination of antecedents.

- The following structure of reasoning was introduced by Mamdani.

A complete FIS will involve $i = 1, \ldots, m$ rules can be combined with the new **connective else**:

$$R_i : \texttt{If } (x_1 \text{ is } A_{1,i}) \texttt{ and } (x_2 \text{ is } A_{2,i}) \texttt{ and } \ldots \texttt{and } (x_n \text{ is } A_{n,i}) \texttt{ then } y \texttt{ is } B_i$$

Each rule is composed by n premise's variables $(x_1, x_2, \ldots, x_n)$ each of which is fuzzified with q membership functions according to the fuzzification rule:

$$x_1 \texttt{ is } A_{i,k} \rightarrow \mu_{i,k}(x_i), \quad i = 1, \ldots, n$$
$$y_1 \texttt{ is } B_{i,k} \rightarrow v_{i,k}(x_i), \quad k = 1, \ldots, q$$

The maximum number of rules is $m = q^n$.

Important concepts - A recap

Building a fuzzy inference system

An alternative structure of a FIS is the **Takagi-Sugeno**. With $i = 1, \ldots, m$, the consequent could be a deterministic (crisp) quantity:

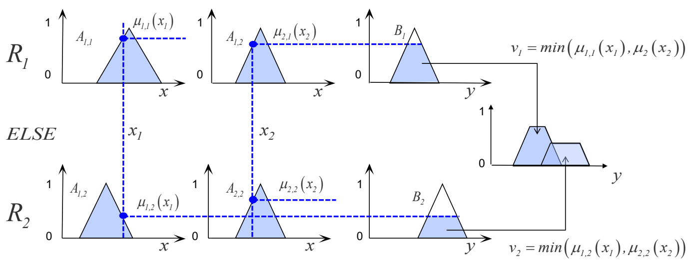
$$R_i : \texttt{If } (x_1 \text{ is } A_{1,i}) \texttt{ and } (x_2 \text{ is } A_{2,i}) \texttt{ and } \ldots \texttt{and } (x_n \text{ is } A_{n,i}) \texttt{ then } y = b_i$$

- This approach opened the way to incredible developments, given the freedom to define the consequents in the most diverse ways.
- From a numerical constant to anything from a linear function to full dynamical systems that could be blended by the antecedents degree of truth to represent any complex system behaviour in differing regions of operation.

Important concepts - A recap

Mamdani fuzzy inference rules

R_1 : if $(x_1$ is $A_{1,1})$ and $(x_2$ is $A_{2,1})$ then $(y_1$ is $B_1)$
else
R_2 : if $(x_1$ is $A_{1,2})$ and $(x_2$ is $A_{2,2})$ then $(y_2$ is $B_2)$



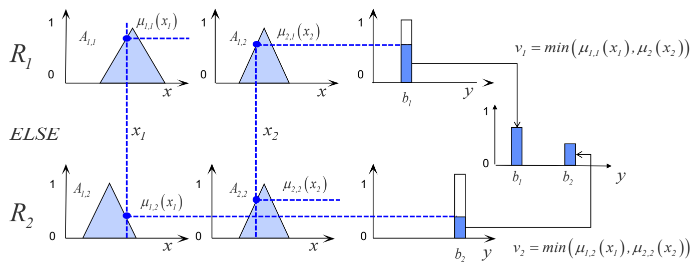
Composition of two Mamdani fuzzy inference rules: R1 and R2:

- Both the `and` and the `then` connectors were implemented with the min operator, whereas max was used for the `else` connector.

Important concepts - A recap

Sugeno fuzzy inference rules

R_1 : if $(x_1$ is $A_{1,1})$ and $(x_2$ is $A_{2,1})$ then $y_1 = b_1$
else
R_2 : if $(x_1$ is $A_{1,2})$ and $(x_2$ is $A_{2,2})$ then $y_2 = b_2$



Composition of two Sugeno inference fuzzy rules: R1 and R2.

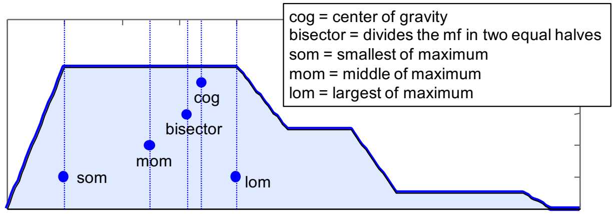
- Both the `and` and the `then` connectors are implemented with the min operator, whereas max was used for the `else` connector.

Important concepts - A recap

Defuzzification

The FIS result must then be defuzzified in order yield a unique result; that is , it should indicate the “most representative” crisp equivalent of the fuzzy reasoning outcome.

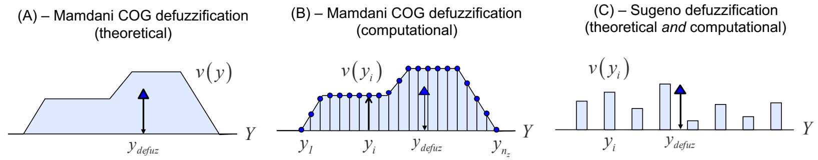
- There are several ways to compute the defuzzified output.



- The most common is the **centre-of-gravity** (COG) method.

Important concepts - A recap

Defuzzification

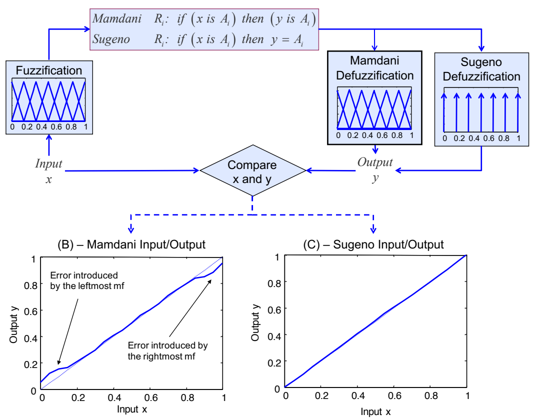


- In the Mamdani theoretical case the COG y_{defuz} can be obtained by integration of the continuous membership function.
  - In practice this is reduced to n_z values and the integral is approximated by a weighted summation.
- In the Sugeno case the output is naturally a collection of discrete values and y_{defuz} is computed by the weighted sum of the consequent values.

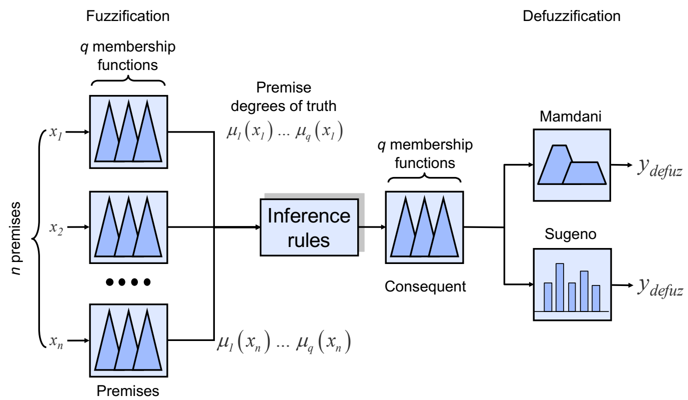
Important concepts - A recap

Fuzzification/Defuzzification errors

The dual processes of fuzzification and defuzzification (fuz/defuz) are non-linear data transforms that might introduce some error in this two-way conversion process.



Important concepts - A recap



Today's goal

Today, we are (finally) going to...

- Introduce the Fuzzy Toolbox (Matlab)
- Introduce Simulink for simulating a fuzzy model

Reading list

Marsili-Libelli, *Environmental systems analysis with Matlab*, 2015

Today's lecture (including the recap) is based on Ch. 4 of the book.

The FIS editor

The best way for the beginner to get acquainted with the world of fuzzy reasoning is to move the first steps within the Fuzzy Editor.

The editor consists of the five interconnected views:

A. **FIS definitions**: This is the first window to appear when the editor is invoked by typing `fuzzy` in the command window.

  - A previously defined FIS can be retrieved by typing fuzzy `<fisname>`, where `<fisname>` is an existing `.fis` file.
  - In the upper part of the window, the FIS structure is depicted, with the antecedents on the left, the rules in the middle (Mamdani or a Sugeno logic), and the consequent on the right.
  - In the lower part of the window the user can specify the connectors implementation and the defuzzification method.
  - By double-clicking on an antecedent icon, the control is transferred to the membership editor.

The FIS editor

The best way for the beginner to get acquainted with the world of fuzzy reasoning is to move the first steps within the Fuzzy Editor.

The editor consists of the five interconnected views:

A. **FIS definitions**.

B. **Membership editor**: the membership of each antecedent can be defined (type, support, etc.)

  - Each μ can be assigned a verbal label, which will be used in the rule definition.
  - Each μ can be manually changed by clicking and dragging the "handles" in the graph, whose numerical values are displayed in the window below.
  - Once all the μs have been defined, we can switch to the rule editor window.

The FIS editor

The best way for the beginner to get acquainted with the world of fuzzy reasoning is to move the first steps within the Fuzzy Editor.

The editor consists of the five interconnected views:

A. **FIS definitions**.

B. **Membership editor**

C. **Rule editor**: Each rule can be set up simply by clicking on the antecedent labels, selecting the connectors, and finally clicking on `add rule`.

  - Existing rules can be changed by reformulating the rule and then clicking `change rule`.
  - An unwanted rule can be discarded by highlighting the rule and clicking `delete rule`.
  - All rules, by default, have a weight of 1, but we may decide to reduce the importance of a rule by giving a smaller weight.
  - By default, the inference rules are specified in a "verbose" way.

The FIS editor

The best way for the beginner to get acquainted with the world of fuzzy reasoning is to move the first steps within the Fuzzy Editor.

The editor consists of the five interconnected views:

A. **FIS definitions**.
B. **Membership editor**
C. **Rule editor**
D. **Rule viewer**: This interactive window shows the FIS output when the vertical lines are clicked and dragged over each antecedent support.

- The degree of activation of all the membership functions are shown in colour and a thick bar shows the defuzzified output value.
- By spanning the entire antecedent supports the user can check the output changes and get an idea of the overall FIS working.

The FIS editor

The best way for the beginner to get acquainted with the world of fuzzy reasoning is to move the first steps within the Fuzzy Editor.

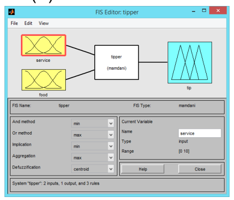
The editor consists of the five interconnected views:

A. **FIS definitions**.
B. **Membership editor**
C. **Rule editor**
D. **Rule viewer**
E. **Surface viewer**: While the previous rule viewer provides an output corresponding to a specific selection of the antecedent values, the surface presents a global view of how each combination of antecedents yields the FIS output.

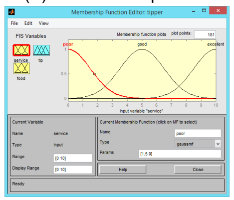
- If there is only one antecedent, the graph reduces to a curve relating the input (antecedent) to the output (defuzzified consequent).
- If there are two antecedents, the 3D graph shows the surface relating them to the defuzzified output.

The FIS editor

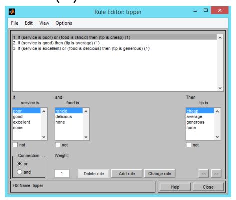
(A) – FIS definitions



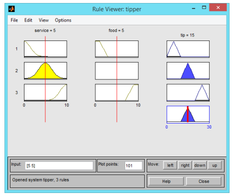
(B) – Membership editor



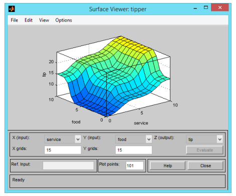
(C) – Rule editor



(D) – Rule viewer



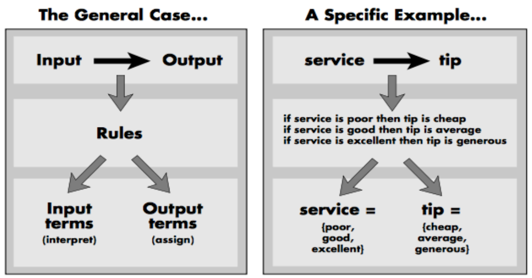
(E) – Surface viewer



The structure of the FIS file

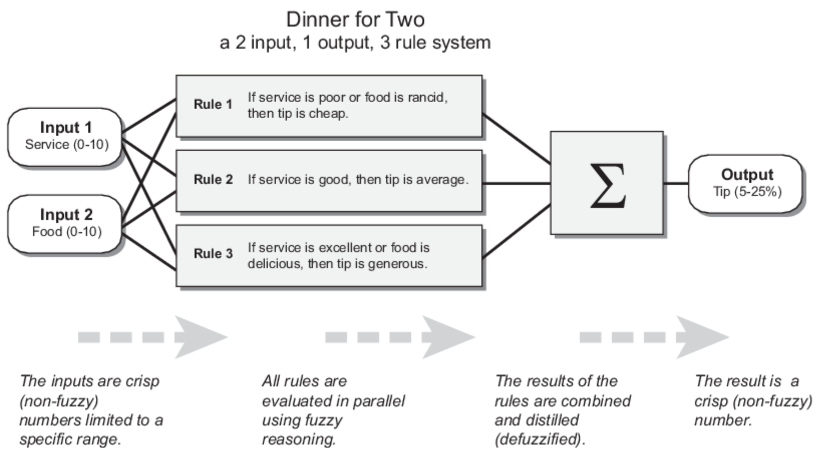
Once the FIS has been designed and tested it can be saved either to the workspace for immediate use or to the disk for permanent storage in a file with the extension `.fis`.

Tipping problem: Given a number between 0 and 10 that represents the quality of service at a restaurant (where 10 is excellent), and another number between 0 and 10 that represents the quality of the food at that restaurant (again, 10 is excellent), **what should the tip be?**



The structure of the FIS file

Tipping problem[2]



[2] Copyright, The MathWorks, Inc.

The structure of the FIS file

Tipping problem[2]

Alternatively to interactively designing the fuzzy inference system for the “Tipper problem” using Fuzzy Logic Designer, we can construct the Mamdani (or Sugeno) FIS at the MATLAB command line.

```
a=newfis('tipper');

% Add the first input variable
a=addvar(a,'input','service',[0 10]);
a=addmf(a,'input',1,'poor','gaussmf',[1.5 0]);
a=addmf(a,'input',1,'good','gaussmf',[1.5 5]);
a=addmf(a,'input',1,'excellent','gaussmf',[1.5 10]);

% Add the second input variable
a=addvar(a,'input','food',[0 10]);
a=addmf(a,'input',2,'rancid','trapmf',[-2 0 1 3]);
a=addmf(a,'input',2,'delicious','trapmf',[7 9 10 12]);
```

[2] Copyright, The MathWorks, Inc.

The structure of the FIS file

Tipping problem[2]

Alternatively to interactively designing the fuzzy inference system for the “Tipper problem” using Fuzzy Logic Designer, we can construct the Mamdani (or Sugeno) FIS at the MATLAB command line.

```
% Add the output variable
a=addvar(a,'output','tip',[0 30]);
a=addmf(a,'output',1,'cheap','trimf',[0 5 10]);
a=addmf(a,'output',1,'average','trimf',[10 15 20]);
a=addmf(a,'output',1,'generous','trimf',[20 25 30]);

% Add the rules
ruleList=[ ...
1 1 1 1 2
2 0 2 1 1
3 2 3 1 2 ];

a=addrule(a,ruleList);
```

[2] Copyright, The MathWorks, Inc.

Fuzzy modelling

We are primarily interested in dynamical models, whose behaviour by definition depends on their past history.

Given the implication structure of a FIS, we can consider discrete-time dynamical models written in the following form:

$$y_k = f\left(y_{k-1}, y_{k-2}, \ldots, y_{k-n}, u_{k-\delta}, \ldots, u_{k-\delta-n_u}\right)$$

- $n_y + n_u$ is the model order, given by the number of the past input and output sample needed to compute current output.
- δ is input/output delay, meaning that the most recent input ($u_{k-\delta}$) influencing the current output (y_k) has entered the system δ samples earlier.

Fuzzy modelling

Mamdani versus Sugeno models

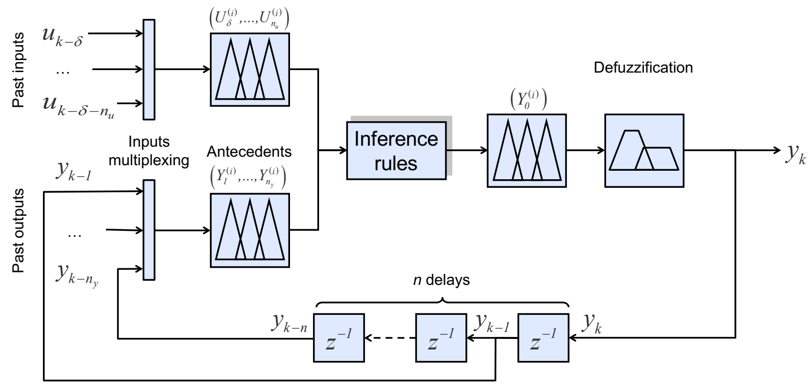
The general model structure can be represented by either a Mamdani or a Sugeno FIS.

In both cases,

- The premises are defined as a collection of membership functions.
- The consequent may be another fuzzy set (Mamdani) or a deterministic quantity (Sugeno) ranging from a constant to a specific dynamical model, which best describes the system behaviour in a specific operating region.

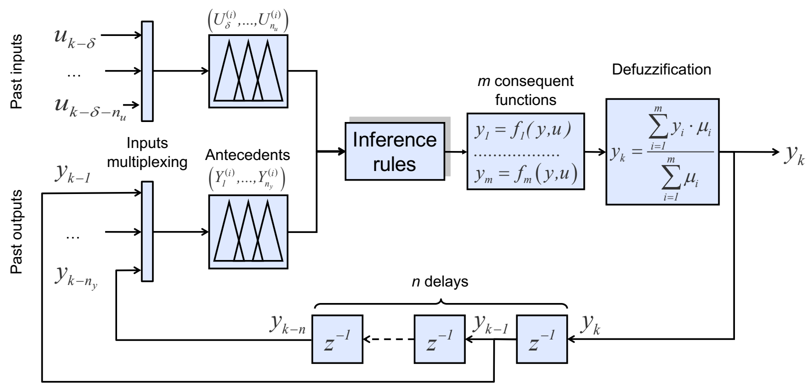
Fuzzy modelling

Mamdani fuzzy dynamic model



Fuzzy modelling

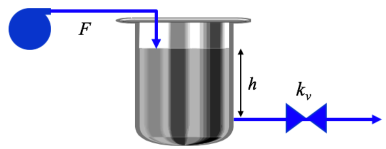
Sugeno fuzzy dynamic model



Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

The pressure flow of a liquid out of a tank can be modelled with the Mamdani fuzzy logic.

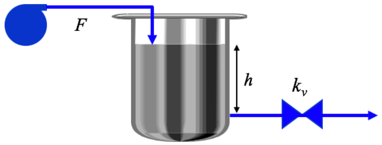


- The model is set up by just looking at the input-output data, as applies to data-driven models.
- The mechanistic model developed is used only to generate the data for training and validation.

Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

The pressure flow of a liquid out of a tank can be modelled with the Mamdani fuzzy logic.



- The pressure flow equation is

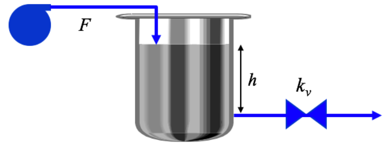
$$A\frac{dh}{dt} = F - k_v\sqrt{h}$$

  - A is the tank cross-section.
  - k_v is the friction coefficient of the outlet.
  - F is the input flow.
  - h is the height of the liquid in the tank.

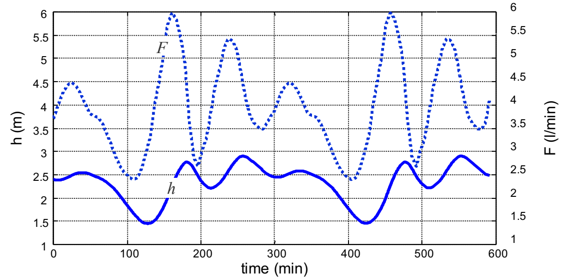
Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

The pressure flow of a liquid out of a tank can be modelled with the Mamdani fuzzy logic.



- The input-output data used to train the fuzzy model are shown in figure.



Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

To build the fuzzy model, the following operations are sequentially performed:

1. Definition of the “meaningful” regions of the data for support delimiting.
   - We must check that our training data are exhaustive and span the entire domain of operation.
2. Definition of the number and type (triangular, Gaussian, etc.) of the mfs.
   - We need to cover the data range with a “complete” set of μs.
   - Then the mathematical form of the μs can be selected and each of them may be given a linguistic label.
3. Definition of the model structure (number of past inputs and outputs) and of the rules of inference.
   - We should set the number of past inputs and outputs that contribute to the current output.

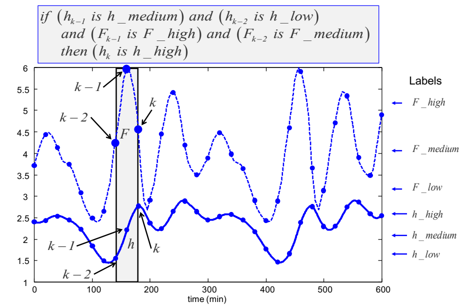
Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

We include the last two inputs and outputs and the model structure looks like:

$$h_k = f\left(h_{k-1}, h_{k-2}, F_{k-1}, F_{k-1}\right)$$

- We also assume that there are three membership functions
- We propose a heuristic method to define the rules



With 3 membership functions and 4 premises we expect 3^4=81 rules, but this example is not likely to generate the whole lot, nor we will need all of them.

Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

R1: If (h1 is h_medium) and (f1 is f_high) and (f2 is f_medium) and (h2 is h_low) then (h is h_high)

R2: If (h1 is h_medium) and (f1 is f_high) and (f2 is f_low) and (h2 is h_medium) then (h is h_high)

R3: If (h1 is h_low) and (f1 is f_high) and (f2 is f_low) and (h2 is h_low) then (h is h_low)

R4: If (h1 is h_medium) and (f1 is f_high) and (f2 is f_medium) and (h2 is h_low) then (h is h_high)

R5: If (h1 is h_medium) and (f1 is f_medium) and (f2 is f_low) and (h2 is h_medium) then (h is h_low)

R6: If (h1 is h_high) and (f1 is f_medium) and (f2 is f_high) and (h2 is h_high) then (h is h_high)

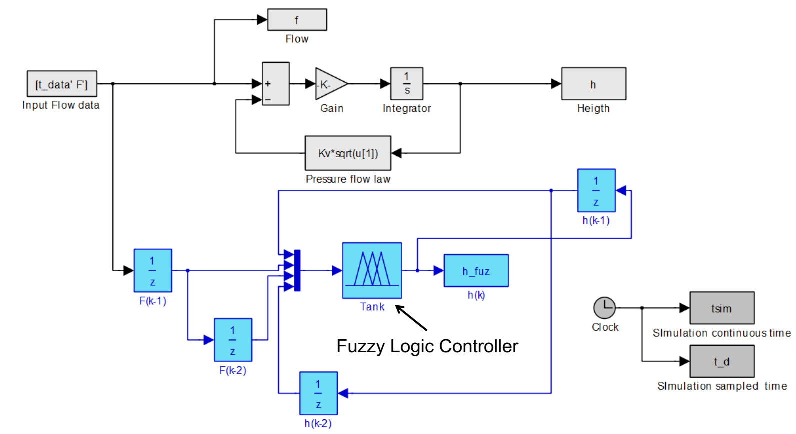
R7: If (h1 is h_high) and (f1 is f_high) and (f2 is f_medium) and (h2 is h_high) then (h is h_high)

R8: If (h1 is h_medium) and (f1 is f_low) and (f2 is f_medium) and (h2 is h_high) then (h is h_low)

Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

The Simulink model is as follows[3]

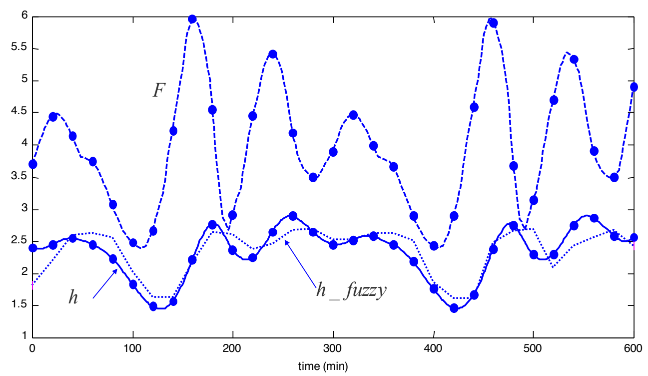


[3] Marsili-Libelli, Environmental Systems Analysis with Matlab

Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

The performance of the fuzzy model is compared to the deterministic model that it was supposed to mimic.



Fuzzy modelling

An example of a heuristic Mamdani fuzzy model

Though the model agreement is far from satisfactory, some useful conclusions may be drawn from this naive exercise.

Pros

- It is not necessary to know the system structure, nor its internal (mechanistic) working.
- The (data-driven) model is derived only on the basis of observed behaviours.

Cons

- No previous knowledge is available to assist in the definition of the model structure (e.g. its order, number and shape of μs, number and nature of the rules).
- The model validity is limited to the range of the observations.

Fuzzy systems in R

An example[1]

```
library(sets)
## set universe
sets_options("universe", seq(from = 0, to = 25, by = 0.1))
## set up fuzzy variables
variables <-
set(service =
    fuzzy_partition(varnames =
                    c(poor = 0, good = 5, excellent = 10),
                    sd = 1.5),
    food = fuzzy_variable(rancid =
                   fuzzy_trapezoid(corners = c(-2, 0, 2, 4)),
                   delicious =
                   fuzzy_trapezoid(corners = c(7, 9, 11, 13))),
    tip =fuzzy_partition(varnames =
                    c(cheap = 5, average = 12.5, generous = 20),
                    FUN = fuzzy_cone, radius = 5)
                    )
```

[1] Source: Documentation of `fuzzyinference` in R (`library(sets)`)

Fuzzy systems in R

An example[1]

```
## set up rules
rules <- set(
    fuzzy_rule(service %is% poor || food %is% rancid,
               tip %is% cheap),
    fuzzy_rule(service %is% good,
               tip %is% average),
    fuzzy_rule(service %is% excellent || food %is% delicious,
               tip %is% generous)
    )

## combine to a system
system <- fuzzy_system(variables, rules)
print(system)
plot(system) ## plots variables

## do inference
fi <- fuzzy_inference(system, list(service = 3, food = 8))

## plot resulting fuzzy set
plot(fi)

## defuzzify
gset_defuzzify(fi, "centroid")
```

[1] Source: Documentation of `fuzzyinference` in R (`library(sets)`)