Computational Intelligence

Master AI

2021-2022

Àngela Nebot

Introduction to Fuzzy Computation

Class Index

Class Index

- Introduction to Evolutionary Computation I - René
- Introduction to Evolutionary Computation II - René
- Introduction to Evolutionary Computation III - René
- Evolutionary Computation LAB – René + Àngela

Exam: 20/1/2022

Final Grade =

5% LabExerFuzzy + 5% LabExerNN + 5% LabExerEC

+ 35% Exam + 50% Practical Work

Classroom: Theory: A5001; Lab: A5S104 / A5S109

Introduction to Fuzzy Computation

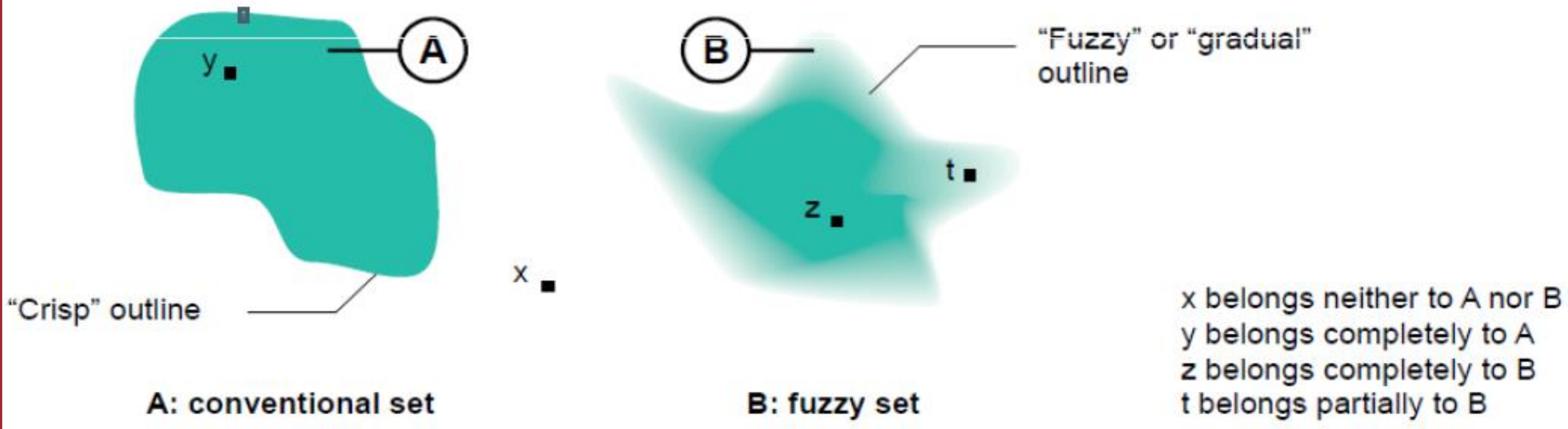
- Fuzzy logic
- Fuzzy sets
- Fuzzy rules
- Fuzzy rule-based systems (Mamdani and Sugeno)
- Adaptive Neuro-Fuzzy Inference System (ANFIS)
- Structure identification: Clustering
- Extracting Fuzzy Models from Data

Fuzzy Logic vs. Fuzzy Arithmetic

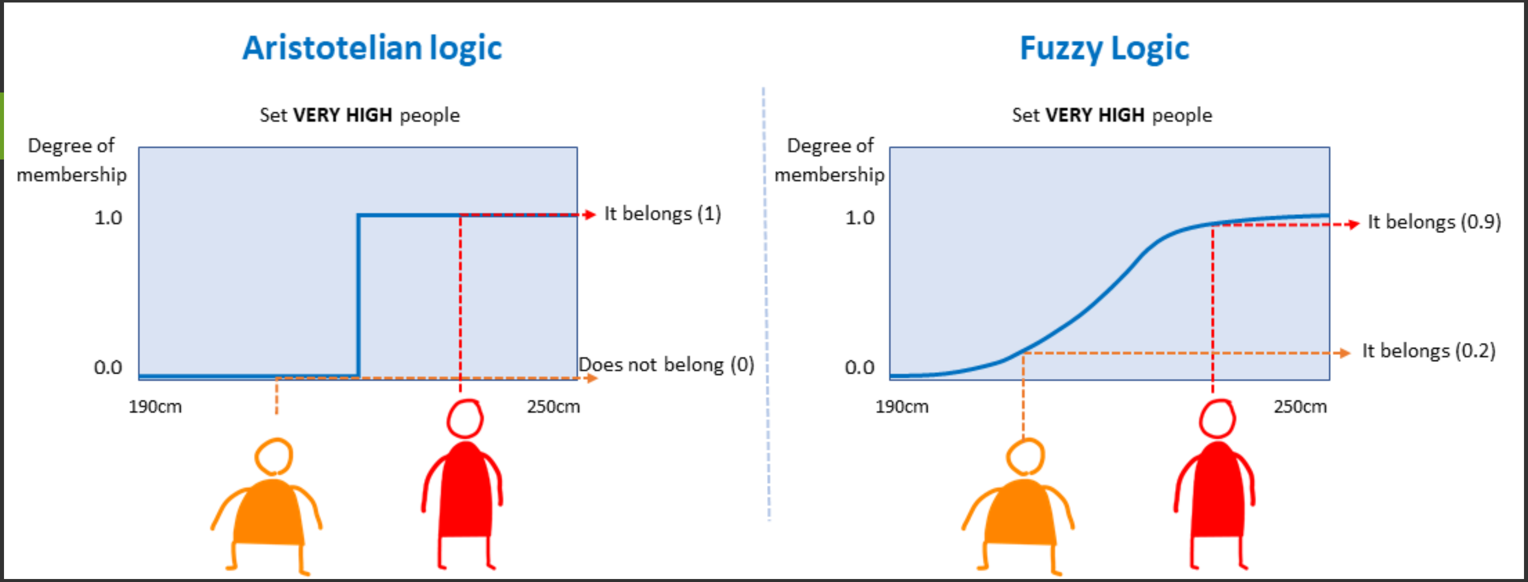
- *Fuzzy logic* is the generalization of ordinary logic (Boolean) and allows truth values other than absolute truth and utter falsity.
- *Fuzzy arithmetic* is a formalism for making calculations with numerical quantities that are imprecisely known. These quantities are known as fuzzy numbers and are defined as fuzzy sets on the real numbers.
- Fuzzy logic is comparable to fuzzy arithmetic in the same way that logic is comparable to arithmetic. They don’t have much in common in terms of the kinds of problems they’re used to address.

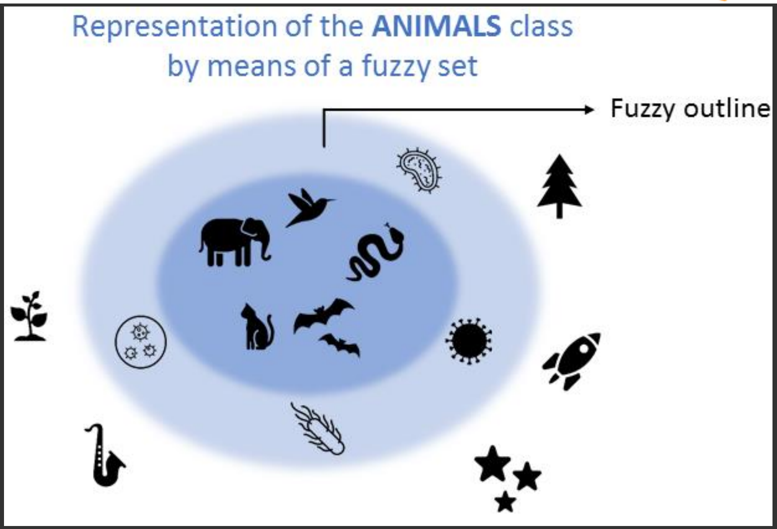
Fuzzy Logic (concept)

- Unlike Boolean logic (bivalent/Aristotelian), fuzzy logic is *multi-valued*
  - Fuzzy logic represents degrees of membership and degrees of truth
  - Things can be *part true* and *part false* at the same time


A
y
"Crisp" outline
A: conventional set
x
B
"Fuzzy" or "gradual" outline
t
z
B: fuzzy set
x belongs neither to A nor B
y belongs completely to A
z belongs completely to B
t belongs partially to B

Fuzzy Logic (concept)



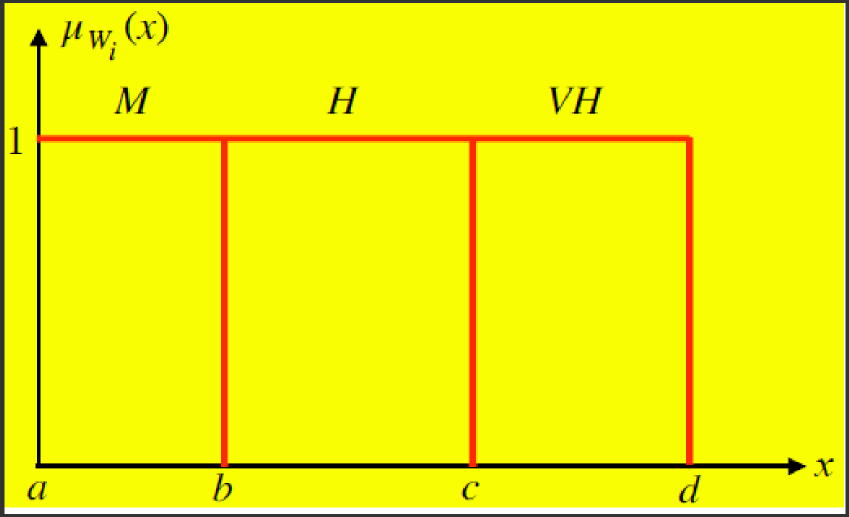


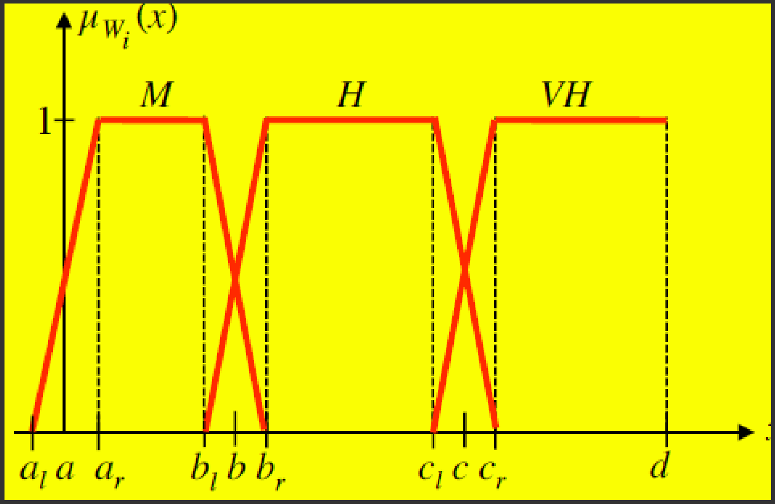
- Examples

Fuzzy Logic is based upon Fuzzy Sets

Zero-order uncertainty partition (crisp set) vs.

First-order uncertainty partition (type-1 fuzzy set)

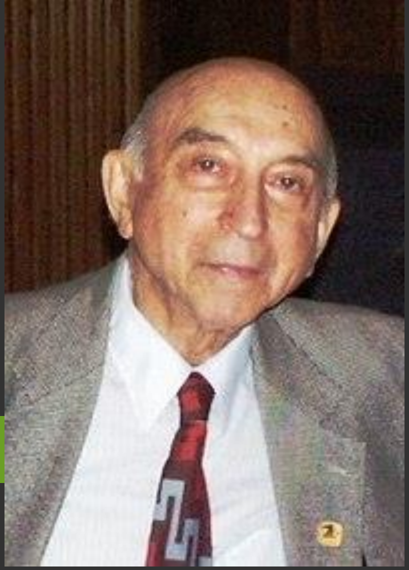




Fuzzy Logic (origins)

- ***Heraclitus*** (535-475 BC): things can be simultaneously *True* and *False*
- ***Plato*** (428-348 BC): there is a third region beyond *True* and *False*
- ***Łukasiewicz*** (1878-1956): many valued logic. First described three-valued logic, it was later generalized to *n*-valued (for all finite *n*)
- ***Knuth*** (1938): three-valued logic similar to *Łukasiewicz*. His insight, apparently missed by *Łukasiewicz*, was to use the integral range [-1, 0, +1] rather than [0, 1, 2]

Fuzzy Logic (origins)



Lotfi A. Zadeh

- was a mathematician, computer scientist, electrical engineer, artificial intelligence researcher and professor emeritus at the University of California, Berkeley
- in his paper fuzzy sets (1965), proposed using a membership function operating on the domain of all possible values and new operations for the calculus of logic and showed that fuzzy logic was a generalization of Boolean logic.
- he also proposed fuzzy numbers as a special case of fuzzy sets, as well as the corresponding rules for consistent mathematical operations (fuzzy arithmetic).

Fuzzy Logic Characteristics

- Represent vagueness and imprecision of statements in natural language
- Knowledge is interpreted as a collection of elastic/fuzzy constraints on a set of variables.
- Inference is viewed as a process of propagation of elastic constraints
- Fuzzy logic allows decision making with estimated values under incomplete or uncertain information
- Exact reasoning is viewed as a limiting case of approximate reasoning

Fuzzy Set Definition (Type-1)

Let X be a nonempty set. A fuzzy set A in X is characterized by its membership function

$$\mu_A : X \rightarrow [0, 1]$$

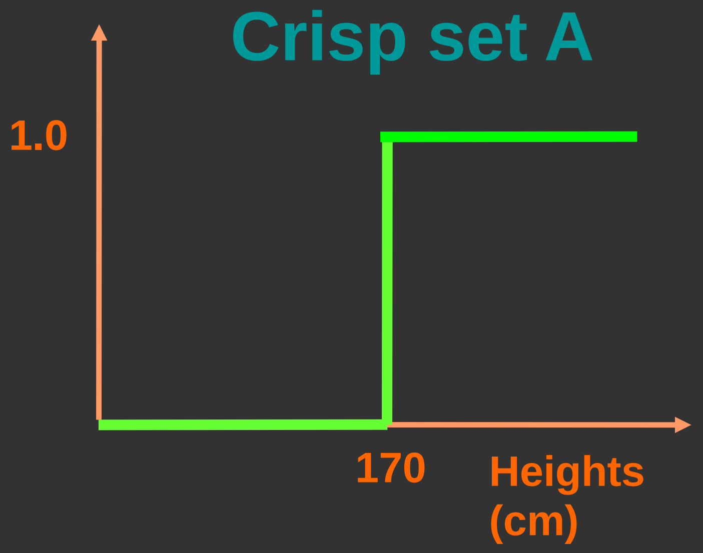
and $\mu_A(x)$ is interpreted as the degree of membership of element x in the fuzzy set A for each $x \in X$.

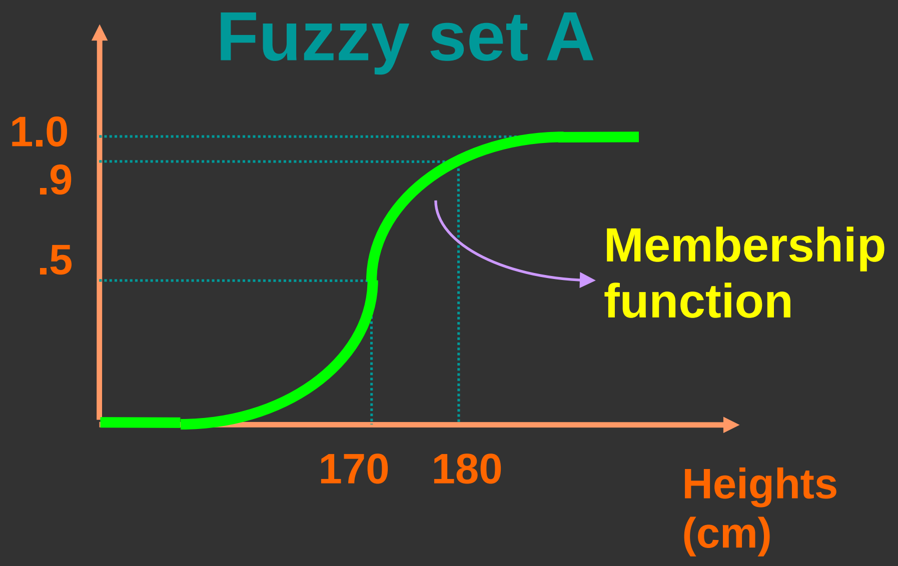
It is clear that *A* is completely determined by the set of ordered pairs:

$$A = \{(x, \mu_A(x)) | x \in X\}$$

Fuzzy Set: example

In mathematics, the membership function of a fuzzy set represents the degree of truth

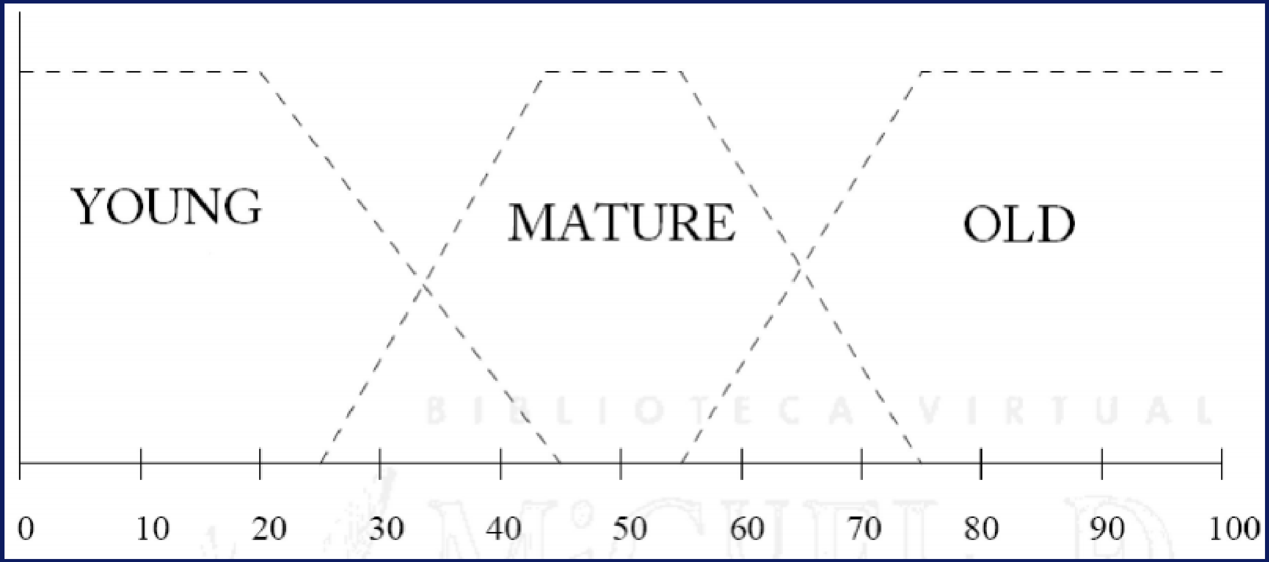




A = Set of tall people

Fuzzy Set: example

- E = {0, …, 100} (Age)
- Fuzzy sets: Young, Mature, Old



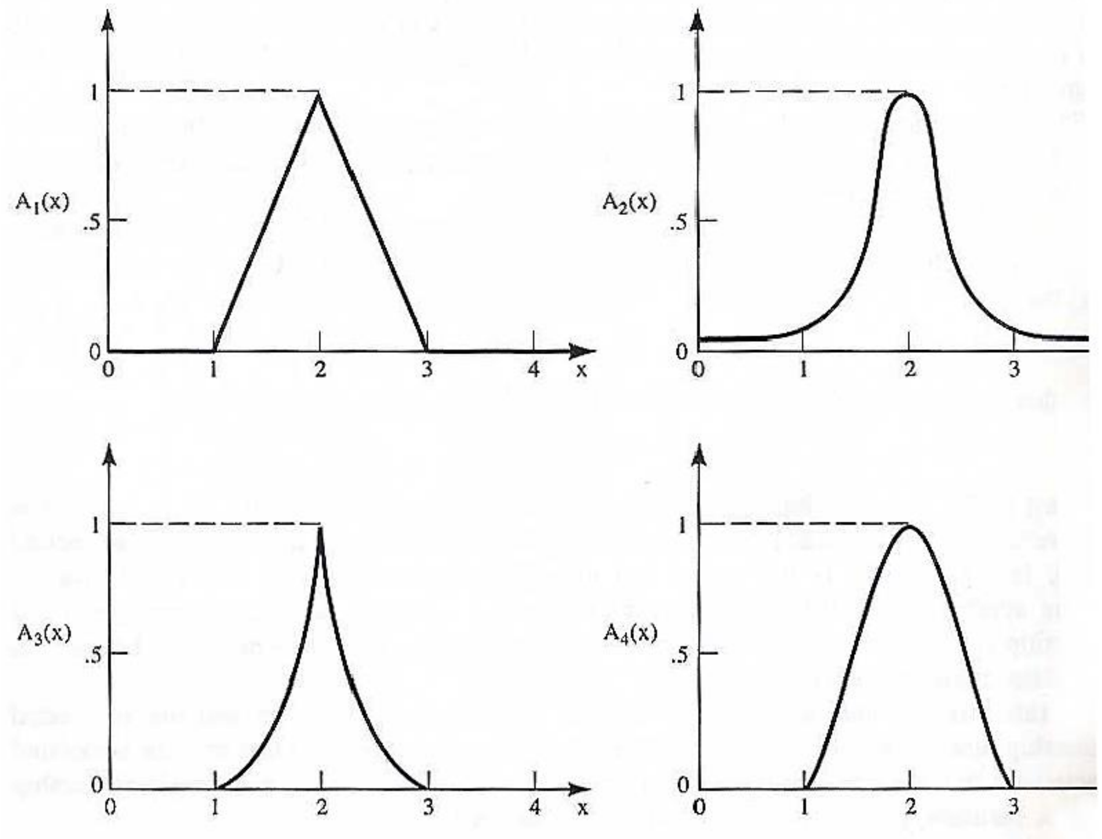
Fuzzy Sets

Fuzzy sets representing the same concept may vary considerably.

Class of real numbers that are close to 2

The four fuzzy sets should possess the properties:

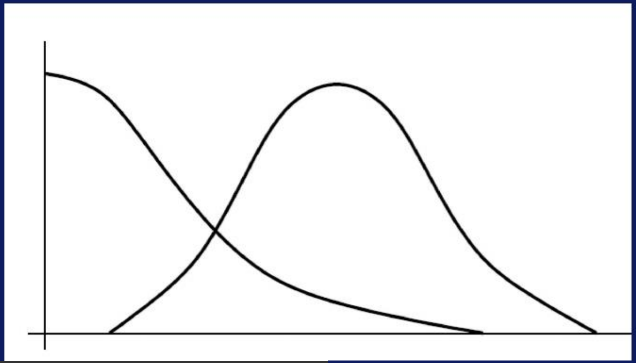
- $A_i(2)=1$ and $A_i(x)<1$, for all $x \neq 2$;
- A_i is symmetric with respect to $x=2$;
- A_i decreases monotonically from 1 to 0



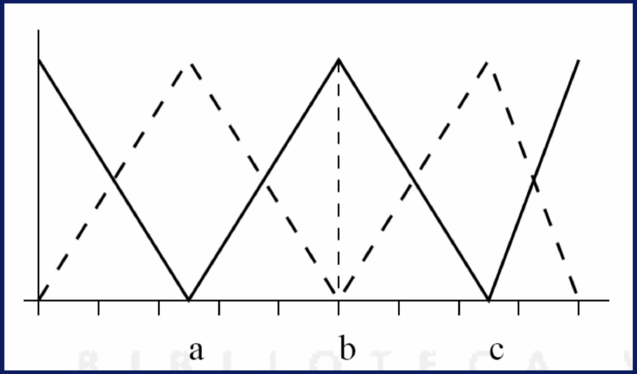
Examples of membership functions that may be used in different contexts for characterizing fuzzy sets of real numbers close to 2.

Types of Membership Functions

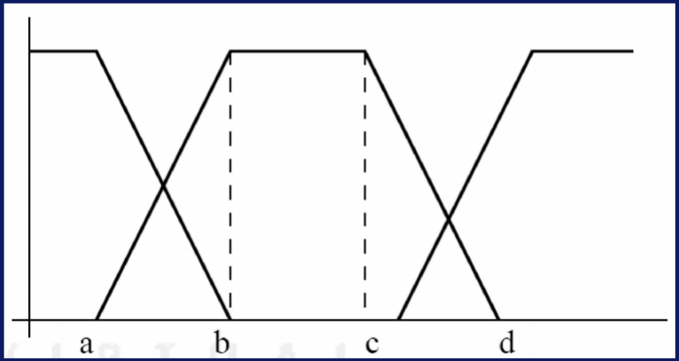
Gaussian



Triangular

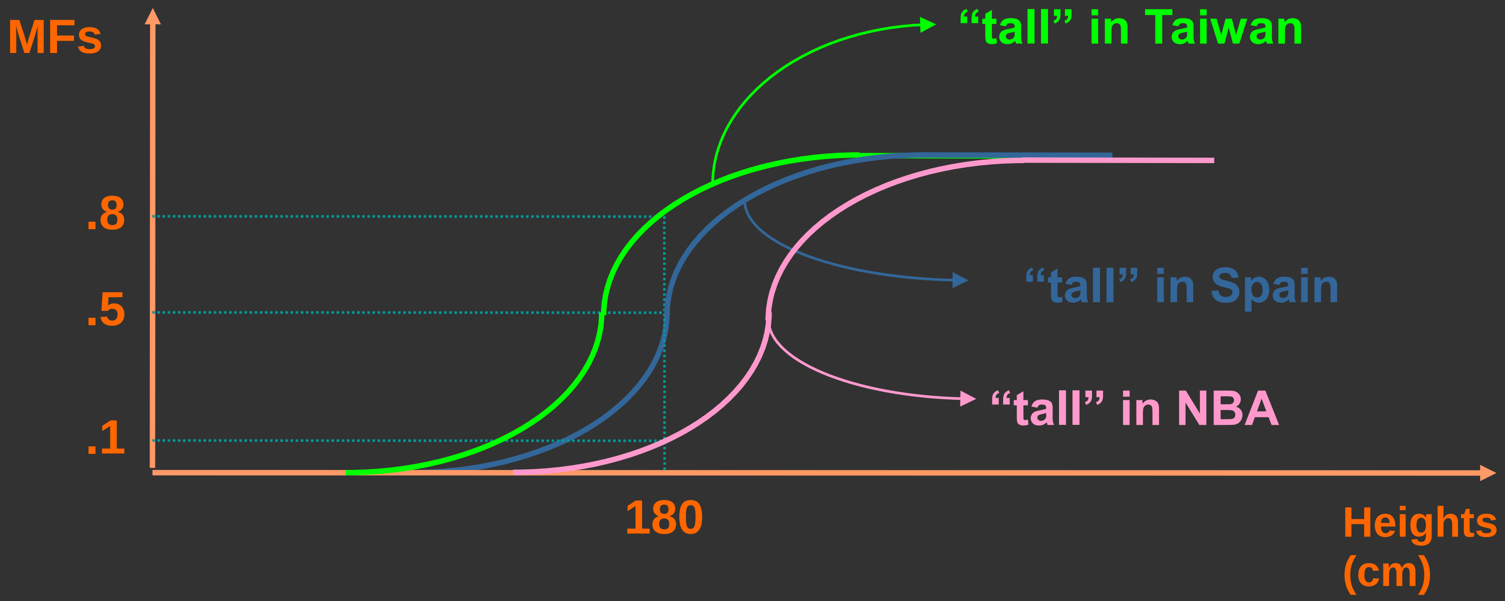


Trapezoidal

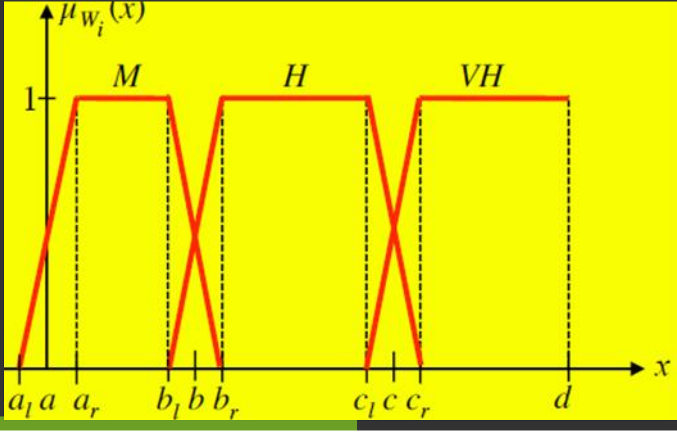


Membership Functions

- About MFs
  - Subjective measures (MFs represent distributions of possibility rather than probability)
  - Not probability functions



Fuzzy Sets: Types



Thus far we introduced only one type of fuzzy set! First-order FS or Ordinary FS or Type-1 FS

Given a relevant universal set *X*, any arbitrary fuzzy set of this type (set *A*) is defined by a function of the form:

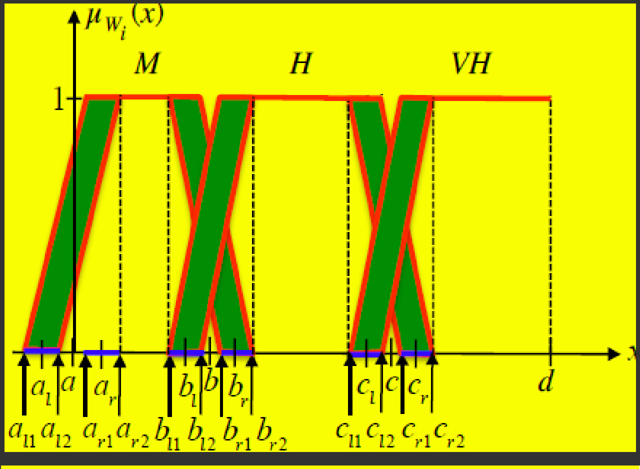
$$A : X \rightarrow [0, 1]$$

Ordinary Fuzzy Sets

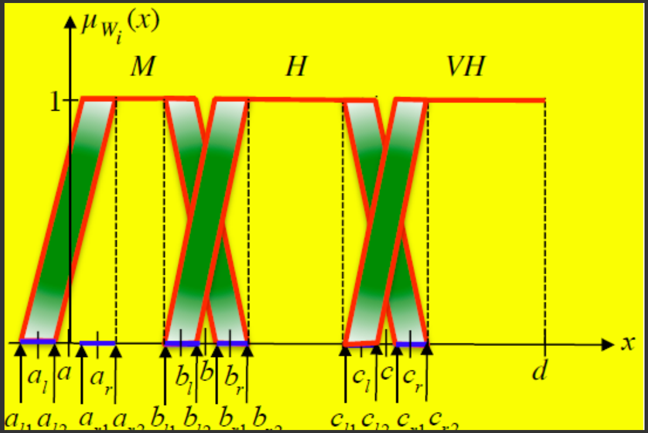
- By far the most common in the literature and applications
- Their membership functions are often overly precise, i.e. they require that each element of the universal set be assigned a particular real number

Fuzzy Sets: Types (Type-2)

- A second-order uncertainty partition of the real-variable, x, partitions it into overlapping intervals, where one is unsure about where the overlap begins and ends, so that the degree of membership in each region of overlap is an interval of real numbers that is a subset of x.


$\mu_{W_i}(x)$
M
H
VH
1
a_l a a_r b_l b b_r c_l c c_r d
a_{l1} a_{l2} a_{r1} a_{r2} b_{l1} b_{l2} b_{r1} b_{r2} c_{l1} c_{l2} c_{r1} c_{r2}

- A non-uniformly shaded footprint of uncertainty (FOU) denotes a non-uniform weighting of all of its points, and is called a non-uniformly weighted second-order uncertainty partition.


$\mu_{W_i}(x)$
M
H
VH
1
x
a_l a a_r b_l b b_r c_l c c_r d
a_{l1} a_{l2} a_{r1} a_{r2} b_{l1} b_{l2} b_{r1} b_{r2} c_{l1} c_{l2} c_{r1} c_{r2}

Fuzzy Sets: Types

Interval-valued fuzzy sets can further be generalized by allowing their intervals to be fuzzy.

Fuzzy sets of **type 2**

$$A : X \rightarrow \mathcal{F}([0, 1])$$

Fuzzy power set: Set of all ordinary fuzzy sets that can be defined within the universal set [0, 1]

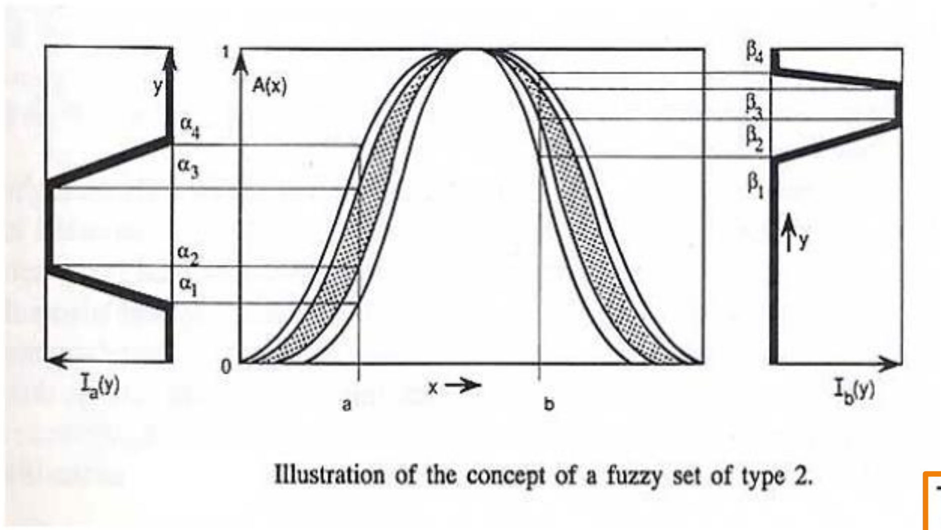
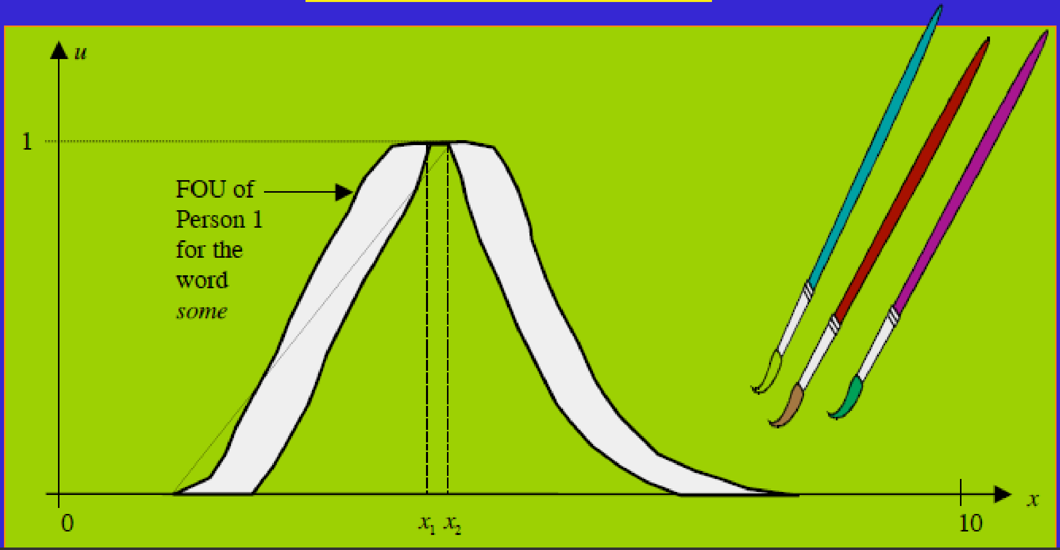


Illustration of the concept of a fuzzy set of type 2.

Types 3, 4, etc...

Examples Type-2 Fuzzy Set

- ***Intra-uncertainty*** **about a word can be modeled using a type-2** ***person fuzzy set*** $\tilde{A}(p_j), j = 1, \ldots, n_{\tilde{A}}$

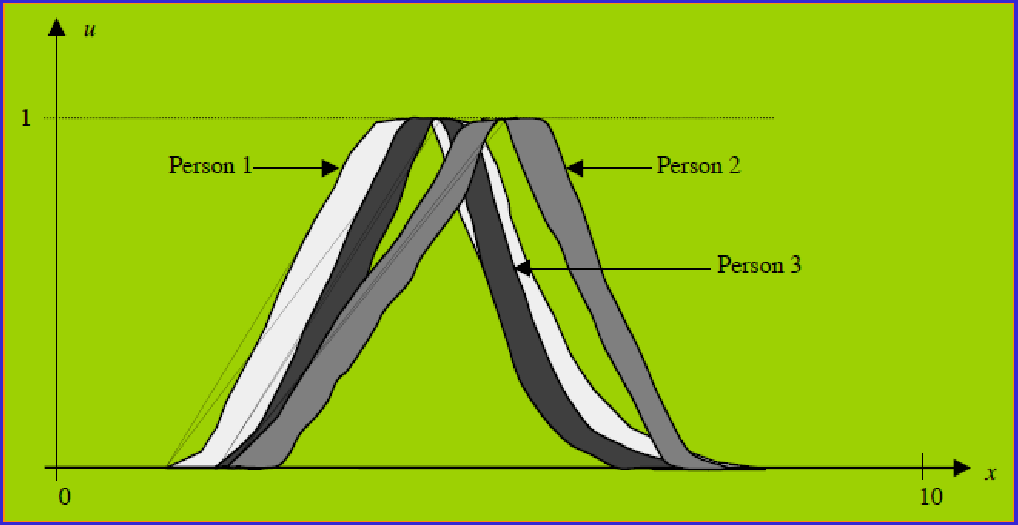


FOU: footprint of uncertainty

From J.M. Mendel IEEE-FUZZ

Examples Type-2 Fuzzy Set

- Collect person MFs from a group of people



Fuzzy Logic

= Computing with words

Linguistic Variables: variables whose values are not numbers but words or sentences in a natural or artificial language.

Lotfi Zadeh

Linguistic Variable = <X, T(X), U, G, M>

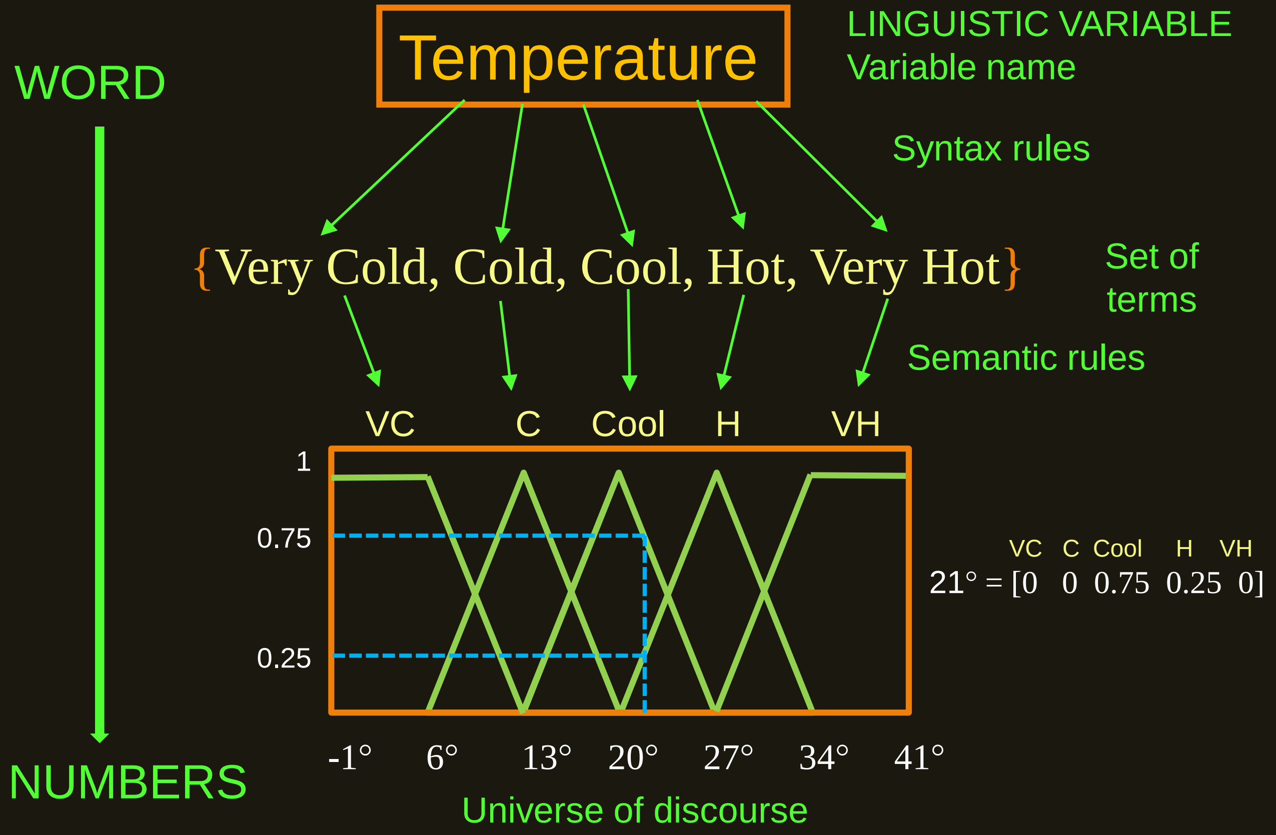
variable name

set of terms

universe of discourse

syntax rules

semantic rules



$$21° = \begin{array}{ccccc} VC & C & Cool & H & VH \\ [0 & 0 & 0.75 & 0.25 & 0] \end{array}$$

Knowledge representation: fuzzy rule-based system

- A *linguistic variable* is a fuzzy variable
  - e.g. the fact "*NBA players are tall*" implies linguistic variable "*NBA players*" takes the linguistic value "*tall*"
- Use linguistic variables to form *fuzzy rules*:

IF	'project duration' is *long*
THEN	'risk' is *high*

IF	'risk' is *very high*
THEN	'project funding' is *very low*

Fuzzy Rules

- A *fuzzy rule* is a conditional statement in the familiar form:

  IF x is A

  THEN y is B

  - x and y are linguistic variables
  - A and B are linguistic values determined by fuzzy sets on the universe of discourses X and Y, respectively.

Fuzzy Rules

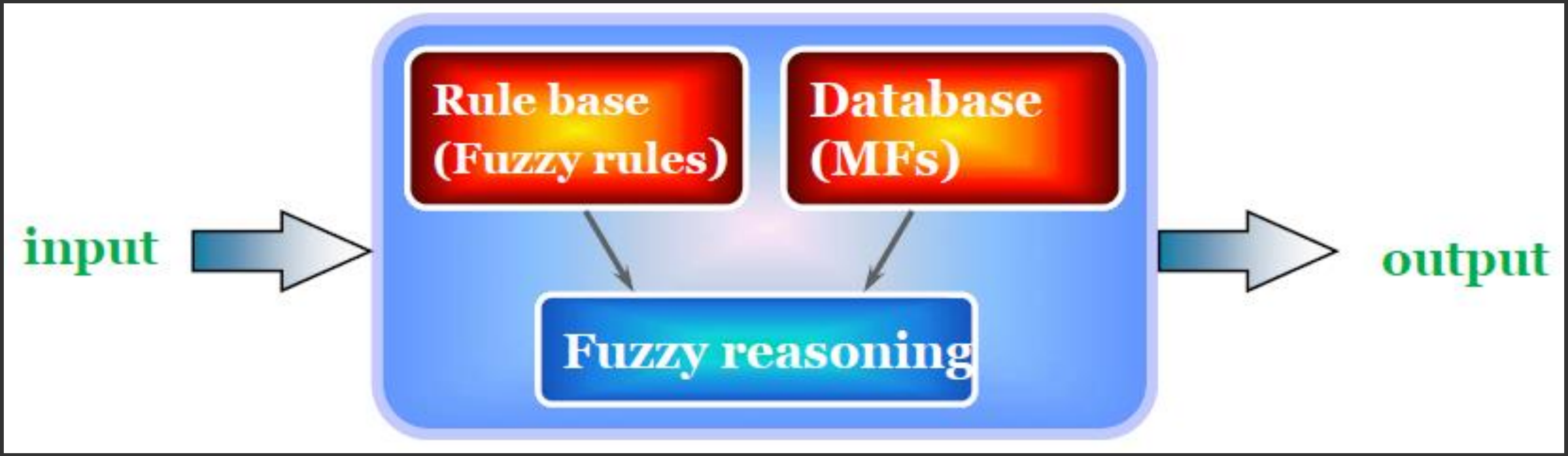
- Other examples (multiple antecedents):
  - e.g.

	IF ‘*project duration*’ is *long*
AND	‘*project staffing*’ is *large*
AND	‘*project funding*’ is *inadequate*
THEN	*risk* is *high*

  - e.g.

	IF *service* is *excellent*
OR	*food* is *delicious*
THEN	*tip* is *generous*

Fuzzy Rule-Base Systems



Extension Principle

- Suppose that f is a function from X to Y, and A is a fuzzy set on X defined as:

$$A = \{(x, \mu_A(x))\} = \{(x_1, \mu_A(x_1)), (x_2, \mu_A(x_2)), \cdots, (x_n, \mu_A(x_n))\}$$

Then the extension principle states that the image of the fuzzy set A under the mapping of f can be expressed as a fuzzy set $B \subseteq Y$.

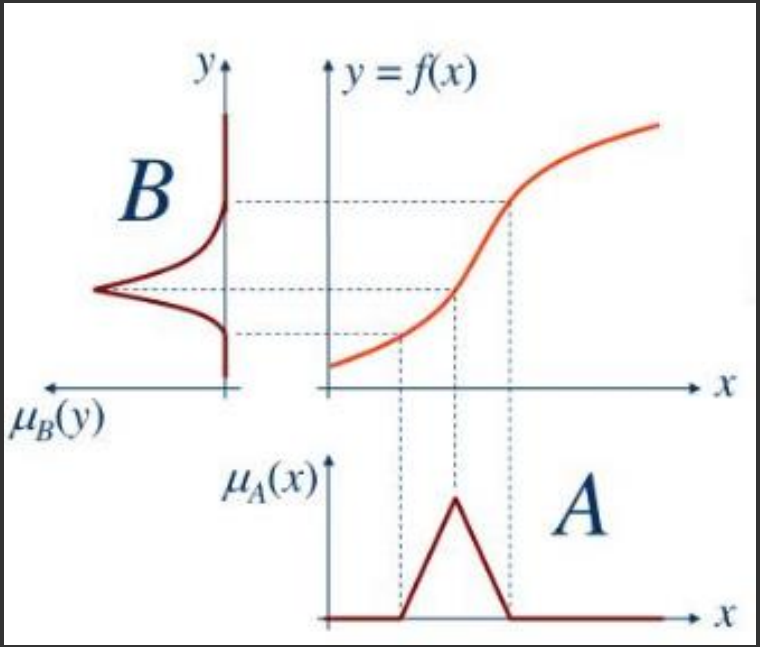
$B = f(A) = \{(y, \mu_B(y)\}$, where $\mu_B(y) = sup\{\mu_A(x) | x \in X, y = f(x)\}$

The supremum, sup, function applies if there are two or more values of X that have the same value in f.

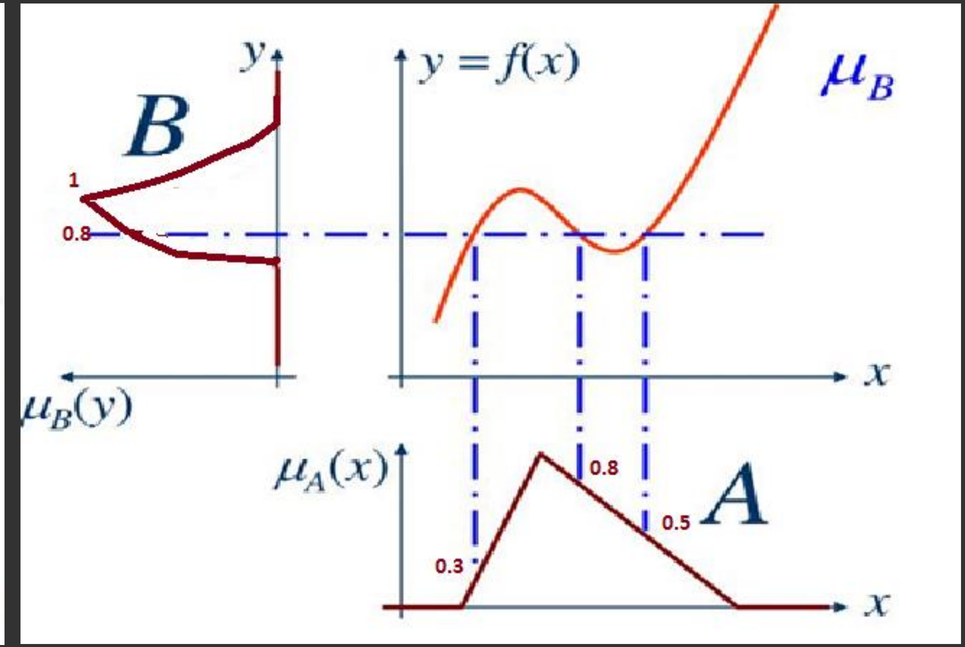
Extension Principle

- The extension principle generalizes an ordinary mapping of a function to a mapping between fuzzy sets.

monotonic function



non-monotonic function



Fuzzy Rule-Based Systems

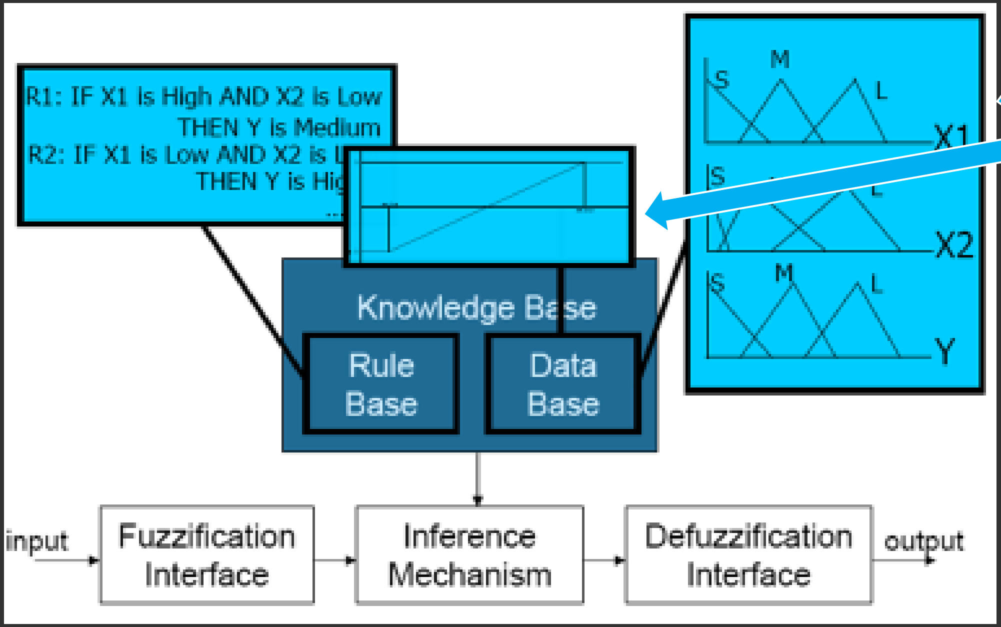
Principal FRBS for engineering problems (i.e., dealing with real-valued inputs and outputs):

- *Mamdani*
- *Takagi-Sugeno-Kang*

Fuzzy Rule-Based Systems Mamdani

- *Ebrahim Mamdani* (1974), the Mamdani method for fuzzy inference is:
  1. Fuzzify the input variables; 2.Evaluate the rules
  3. Aggregate the rule outputs; 4. Defuzzify

Linguistic rules (expert knowledge)



Fuzzy rule semantics (fuzzy sets, scalability)

Fuzzy Inference

- Classic Modus Ponens:

$$\frac{\begin{array}{l} A \\ A \rightarrow B \end{array}}{B} \qquad \frac{\begin{array}{l} \text{old person} \\ \text{rule: if person is old then lose memory} \end{array}}{\text{lose memory}}$$

- What happens if we have A', i.e. very old person?

Humans can infer a conclusion but computers can not!

- Generalized Modus Ponens:

$$\frac{\begin{array}{l} A' \\ A \rightarrow B \end{array}}{B'} \xrightarrow{B' = A' \circ R} \text{Compositional Rule of Inference:}$$

$$B'(y) = \sup_{x \in X} \min[A'(x), R(x,y)]$$

Fuzzy Inference

fuzzy relation

Modus Ponens

CRI (C'_i = max min[R', R_i])

Fact: x is A' and y is B' $\Rightarrow$ R' = A' x B'

Fuzzy Rule Set:

R_1 = if x is A_1 and y is B_1 then z is C_1 $\Rightarrow$ $R_1 = A_1 \times B_1 \times C_1$

R_2 = if x is A_2 and y is B_2 then z is C_2 $\Rightarrow$ $R_2 = A_2 \times B_2 \times C_2$

: : :

---

z is C'

$\Rightarrow$ $C'_1 = R' \circ R_1$

$C'_2 = R' \circ R_2$

:

$C' = C'_1 \cup C'_2 \cup ..$

$\Rightarrow$

$$\mu_{C'_1}(z) = \mu_{A_1}(x_0) \wedge \mu_{B_1}(y_0) \wedge \mu_{C_1}(z)$$
$$\mu_{C'_2}(z) = \mu_{A_2}(x_0) \wedge \mu_{B_2}(y_0) \wedge \mu_{C_2}(z)$$
$$\vdots$$

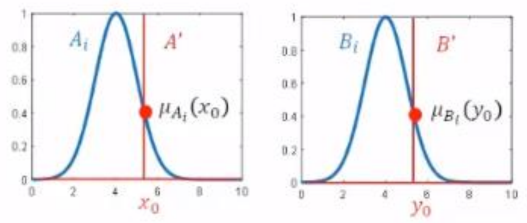
$$\mu_{C'}(z) = \mu_{C'_1}(z) \vee \mu_{C'_2}(z) \vee \ldots$$

$$\mu_{C'_i}(z) = \vee_{x,y}\left[\mu_{R'}(x,y) \wedge \mu_{R_i}(x,y,z)\right] = \vee_{x,y}\left[\mu_{A'}(x) \wedge \mu_{B'}(y) \wedge \mu_{A_i}(x) \wedge \mu_{B_i}(y) \wedge \mu_{C_i}(z)\right]$$
$$= \{\vee_x[\mu_{A'}(x) \wedge \mu_{A_i}(x)]\} \wedge \{\vee_y[\mu_{B'}(y) \wedge \mu_{B_i}(y)]\} \wedge \mu_{C_i}(z)$$

$\Downarrow$

$$\mu_{C'_i}(z) = \mu_{A_i}(x_0) \wedge \mu_{B_i}(y_0) \wedge \mu_{C_i}(z)$$

Considerando que $A' = \{x_0\}$ y $B' = \{y_0\}$



$$\vee_x[\mu_{A'}(x) \wedge \mu_{A_i}(x)] = \mu_{A_i}(x_0) \qquad \vee_y[\mu_{B'}(y) \wedge \mu_{B_i}(y)] = \mu_{B_i}(y_0)$$

t-norm operator generalizes intersection (usually min., prod)

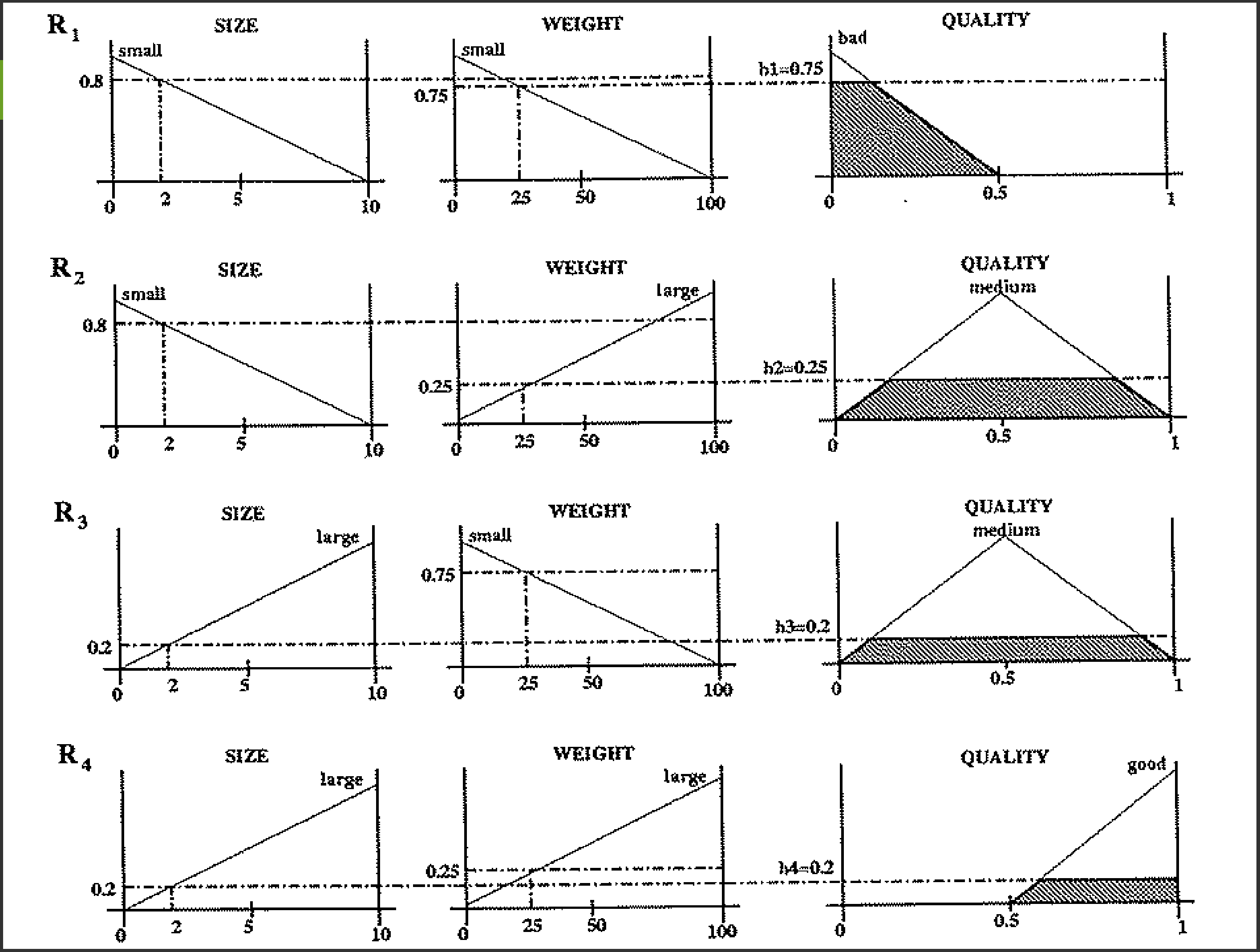
t-conorm operator generalizes union (usually max.)

Fuzzy Inference: Mamdani

Current system input = (2, 25)

Degree of firing or firing strength between the antecedent part of the i-th rule and the current inputs to the system.

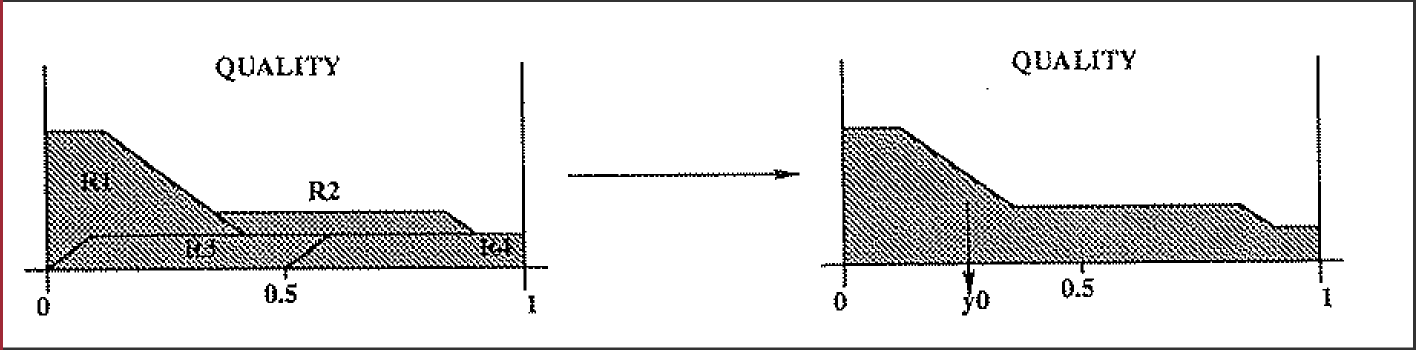
T-norm = conjunctive operators (minimum, product)



Mamdani – Defuzzification

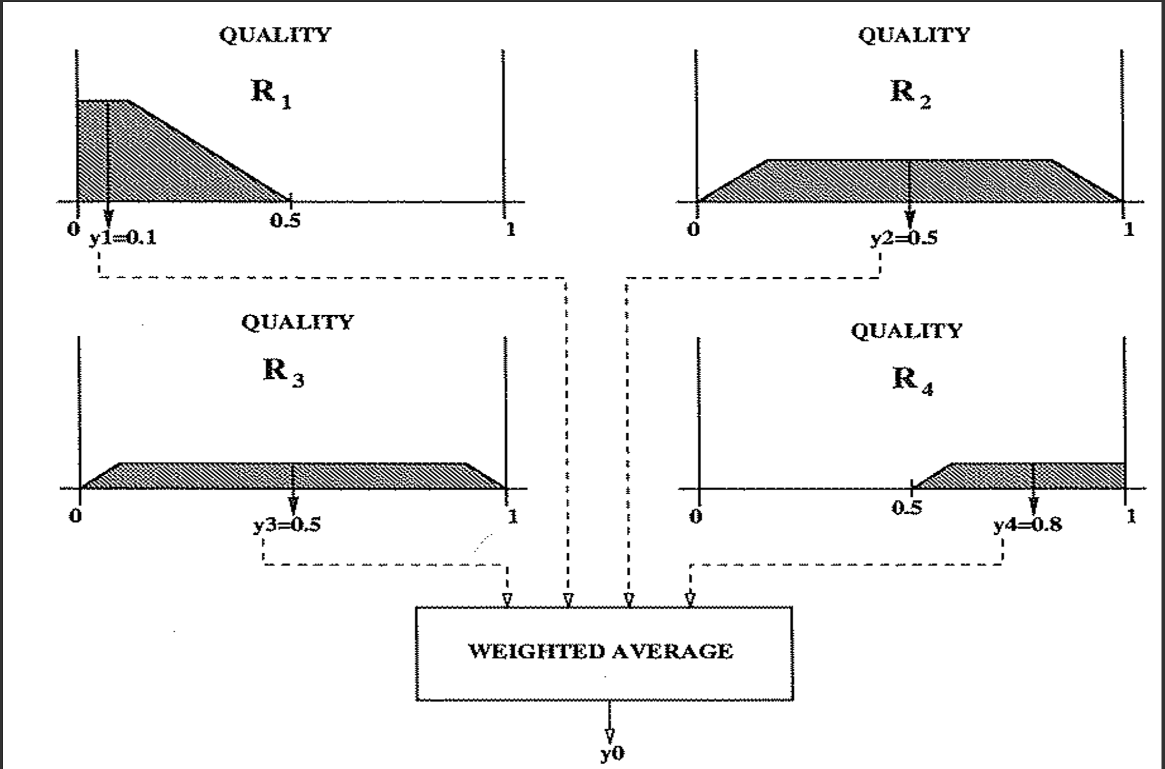
First Aggregation then Inference - FATI

(center of maximums, center of gravity, etc.)



First Inference then Aggregation - FITA

(maximum value)



Mamdani – Example

4. Defuzzification

– Calculate the *center of gravity* (*cog*):

$$COG = \frac{\int_a^b \mu_A(x)\, x\, dx}{\int_a^b \mu_A(x)\, dx}$$

- An easier approximation (with some imprecision…):

$$y = \frac{\sum_{j=1}^{r} \mu_j \cdot s_j}{\sum_{j=1}^{r} \mu_j}$$

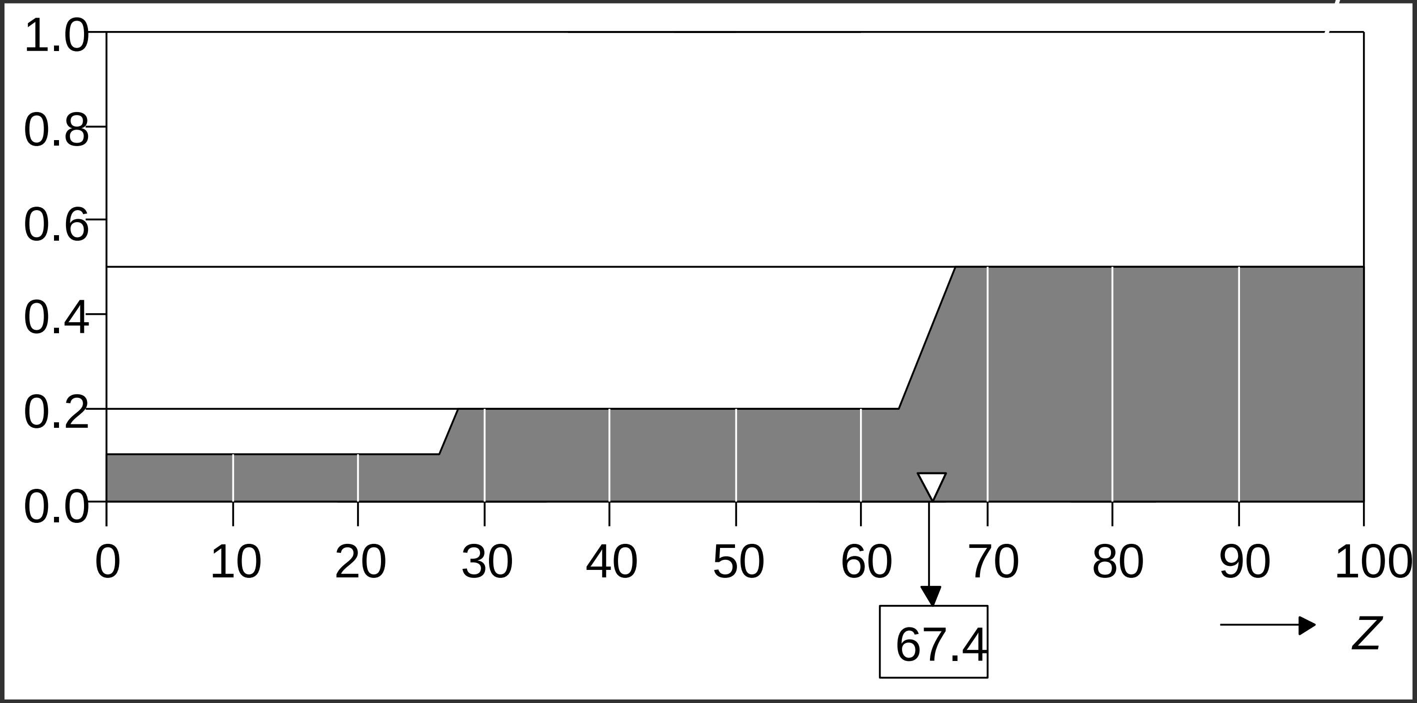
S_j is the center of gravity of set *j*

$$COG = \frac{(0+10+20)\times 0.1+(30+40+50+60)\times 0.2+(70+80+90+100)\times 0.5}{0.1+0.1+0.1+0.2+0.2+0.2+0.2+0.5+0.5+0.5+0.5} = 67.4$$

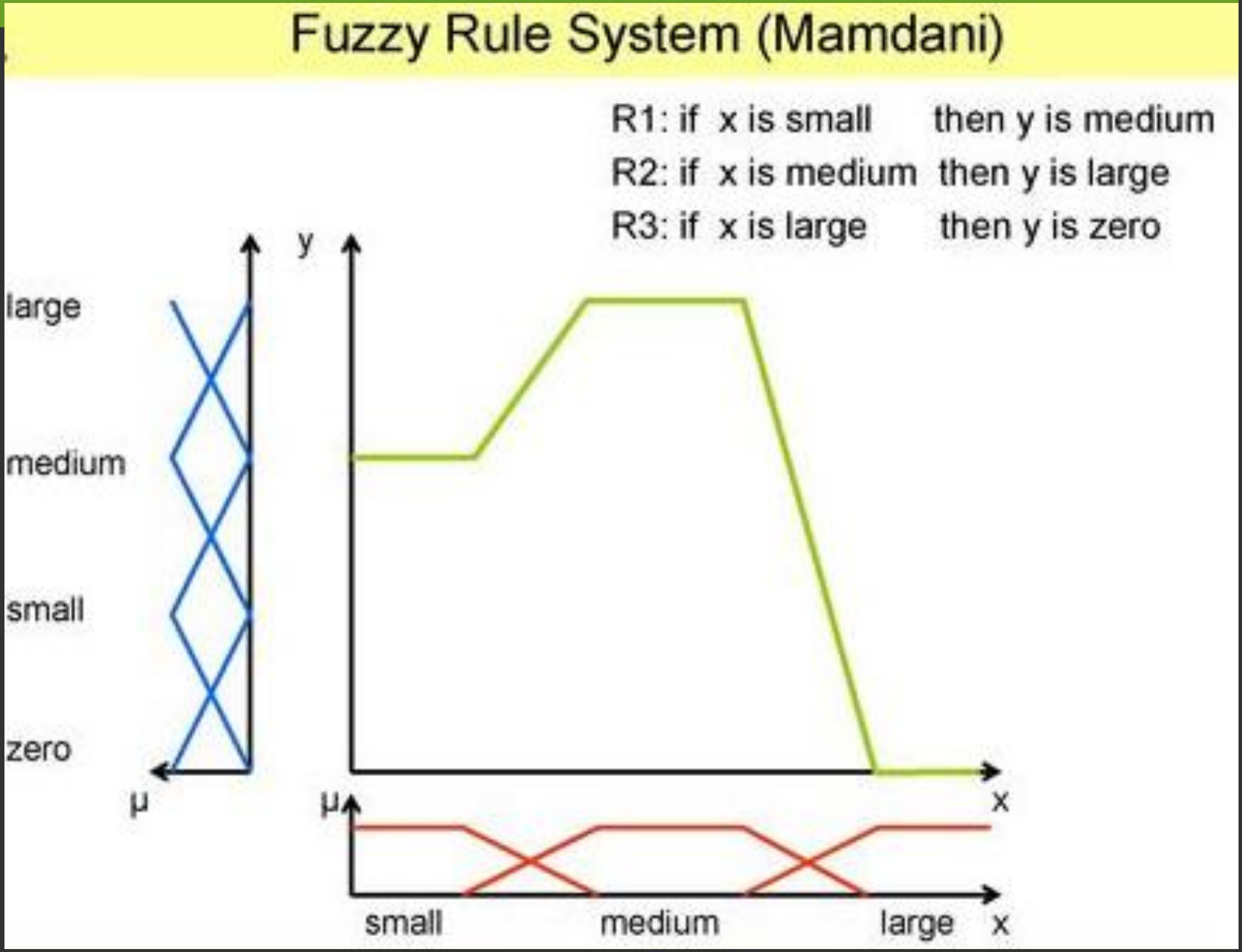
Mamdani – Example

4. Defuzzification

– Use a reasonable sampling of points



Mamdani: Visual example


Fuzzy Rule System (Mamdani)

R1: if x is small then y is medium
R2: if x is medium then y is large
R3: if x is large then y is zero

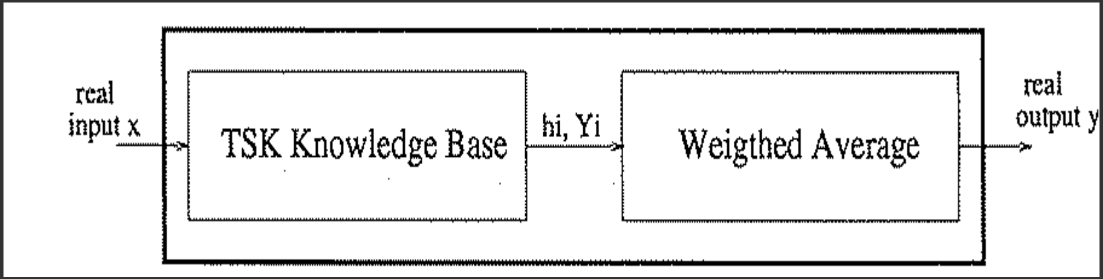
Demo Mamdani

- **Fuzzy Water Tank system**

Matlab: sltankrule

Fuzzy Inference: TSK

- Takagi-Sugeno-Kang (1985), the TSK method for fuzzy inference is:



$$\begin{aligned}&IF\ X_1\ is\ A_1\ and\ \dots\ and\ X_n\ is\ A_n\\&THEN\ Y = p_1 \cdot X_1 + \dots + p_n \cdot X_n + p_0,\end{aligned}$$

Polynomial function

Output TSK (m rules): $\dfrac{\sum_{i=1}^{m} h_i \cdot Y_i}{\sum_{i=1}^{m} h_i}$

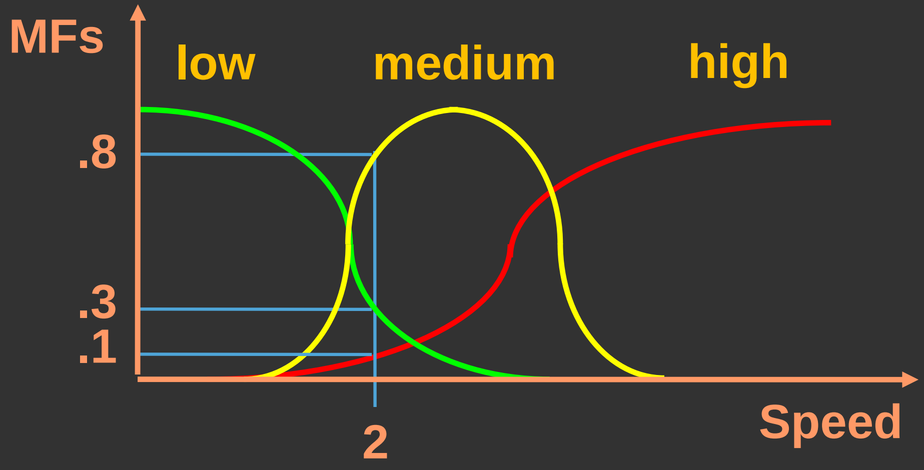
$$h_i = T(\mu_{A_{i1}}(x_1), \dots, \mu_{A_{in}}(x_n))$$

Matching degree between the antecedent part of the i-th rule and the current inputs to the system. T-norm = conjunctive operators (minimum, algebraic product)

Zero-Order TSK FRBS

Takagi-Sugeno-Kang

If speed is low then resistance = 2
If speed is medium then resistance = 4
If speed is high then resistance = 8



Rule 1: h1 = .3; Y1 = 2
Rule 2: h2 = .8; Y2 = 4
Rule 3: h3 = .1; Y3 = 8

Resistance = Σ(hi*Yi) / Σhi
= (0.3*2+0.8*4+0.1*8)/(0.3+0.8+0.1)
= 3.83

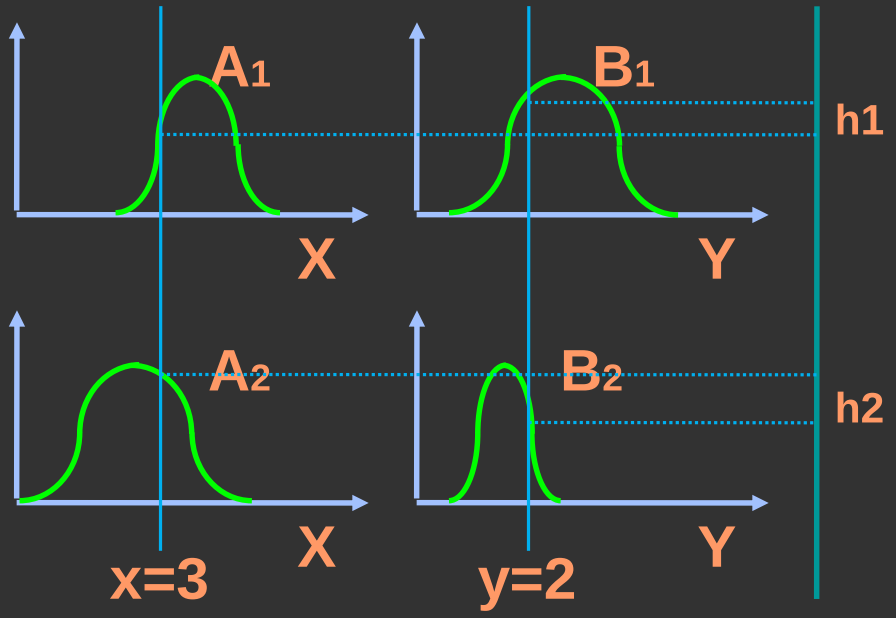
First-Order TSK FRBS

Takagi-Sugeno-Kang

Rule base

If X is A1 and Y is B1 then Z = p1*x + q1*y + r1
If X is A2 and Y is B2 then Z = p2*x + q2*y + r2

Fuzzy reasoning



$Z_1 = p_1*3+q_1*2+r_1$

$Z_2 = p_2*3+q_2*2+r_2$

$$Z = \frac{h_1*Z_1+h_2*Z_2}{h_1+h_2}$$

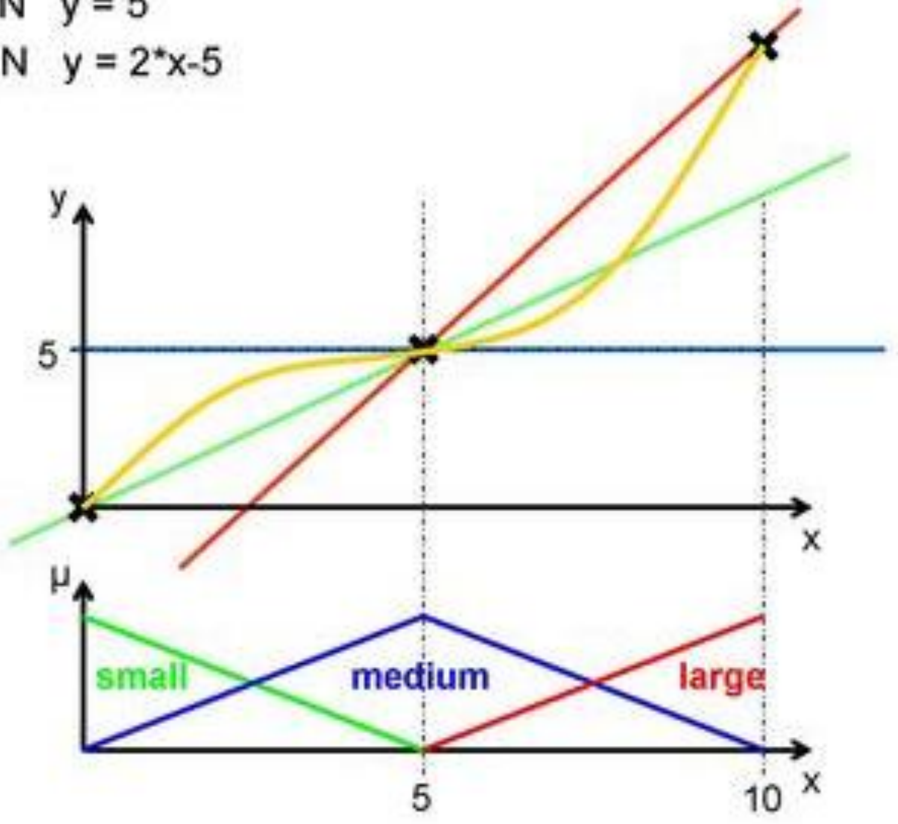
TSK: Visual example

R1: IF x IS small THEN y = x

R2: IF x IS medium THEN y = 5

R3: IF x IS large THEN y = 2*x-5

$$y = \frac{\sum_{i=1}^{r} \mu_{R_i} \cdot y_i(\vec{x})}{\sum_{i=1}^{r} \mu_{R_i}}$$



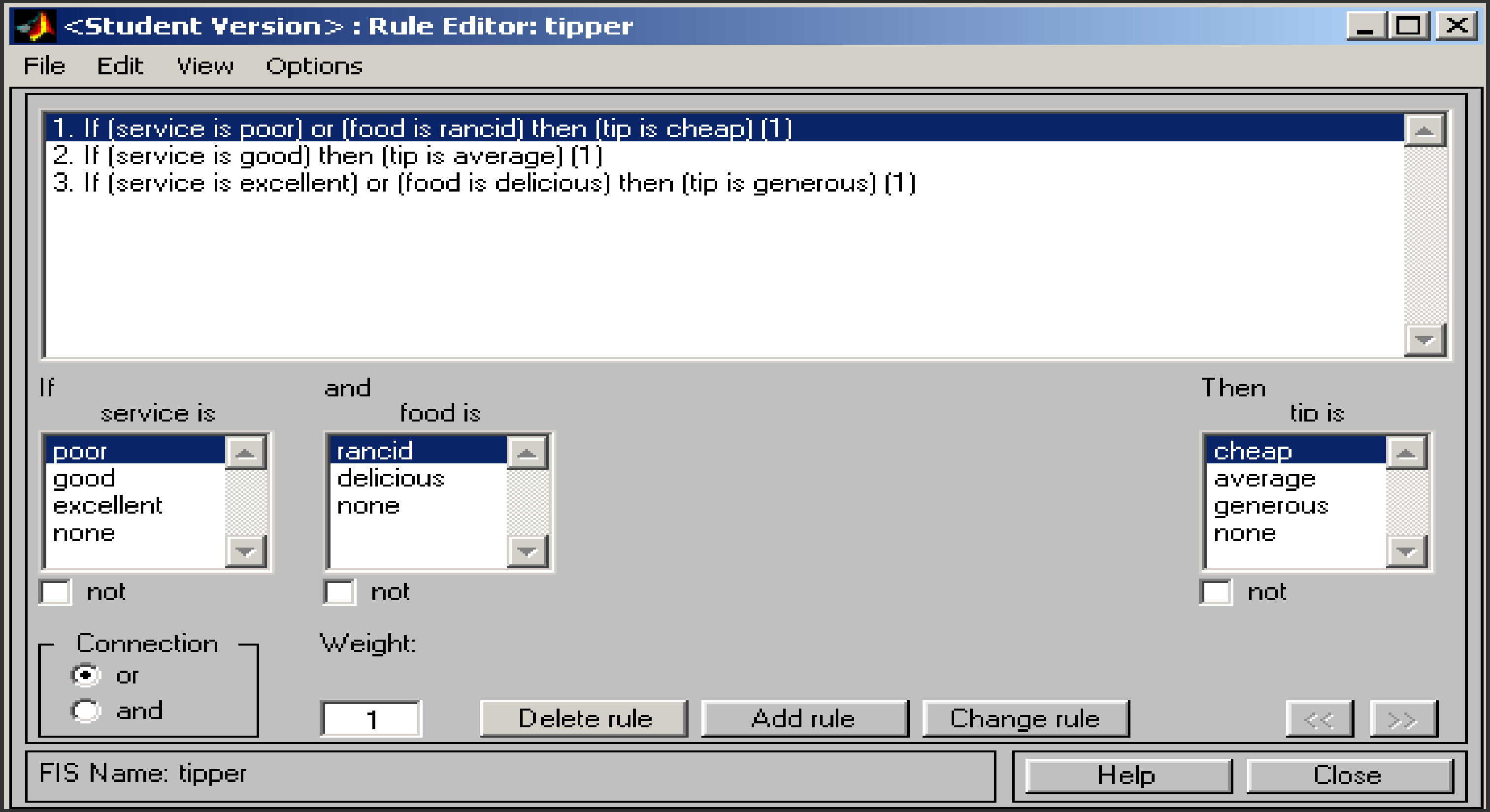
Demo TSK

Fuzzy Juggling ball system

Matlab: juggler

Example: Dinner for two

Rule Editor



1. If (service is poor) or (food is rancid) then (tip is cheap) (1)
2. If (service is good) then (tip is average) (1)
3. If (service is excellent) or (food is delicious) then (tip is generous) (1)

If service is: [✓ poor] good excellent none ☐ not

and food is: [✓ rancid] delicious none ☐ not

Then tip is: [✓ cheap] average generous none ☐ not

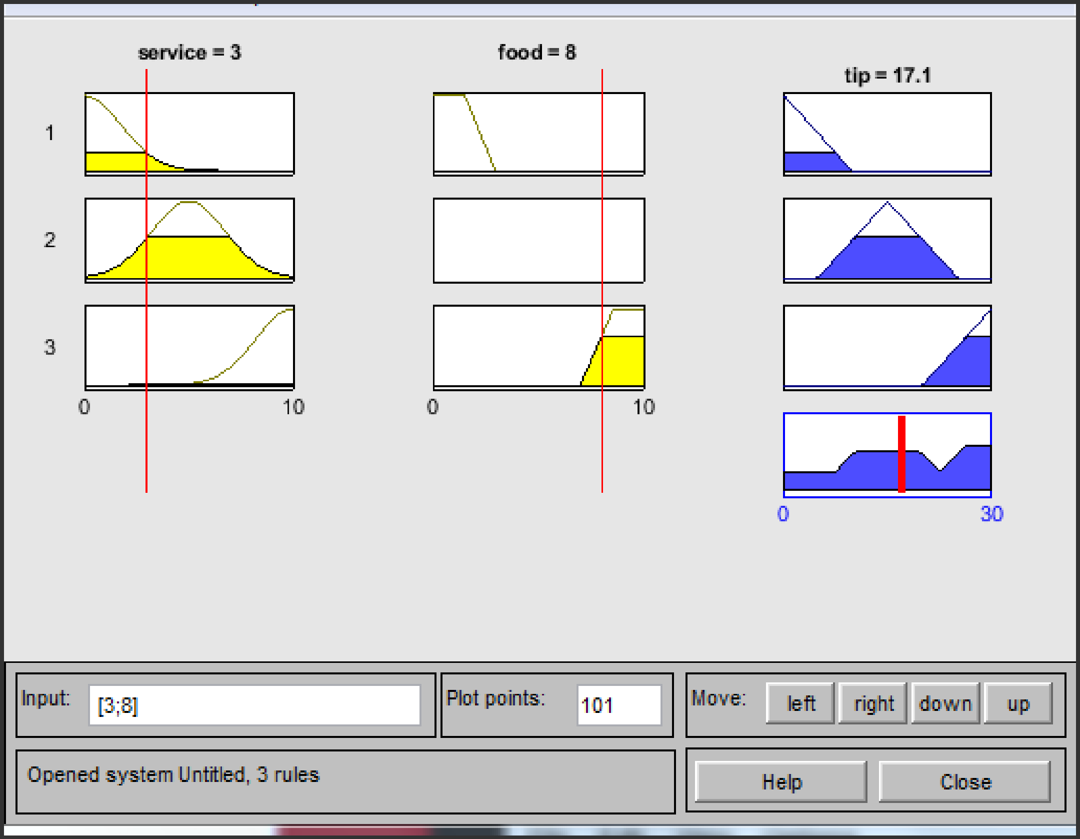
Connection: [✓ or] and

Weight: 1

FIS Name: tipper

Example: Dinner for two (Mamdani)

Rule Viewer



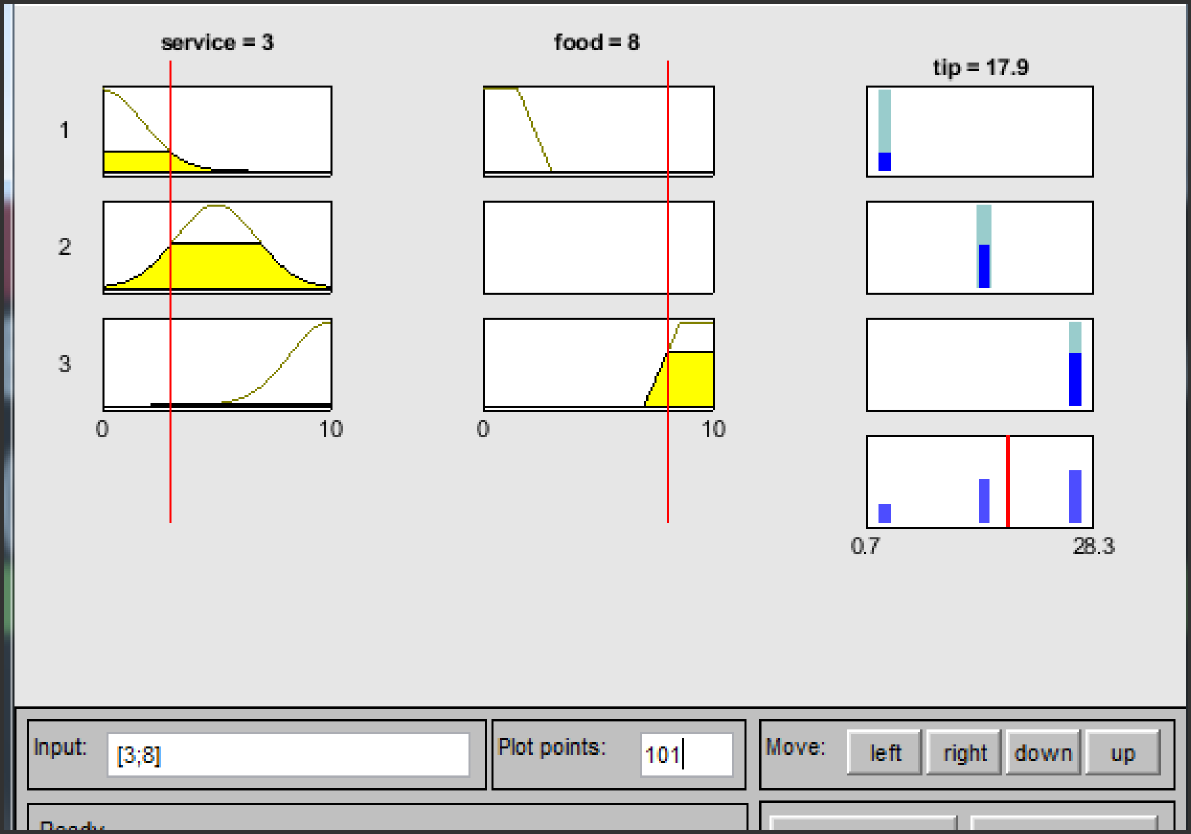
Input: [3;8]

Plot points: 101

Opened system Untitled, 3 rules

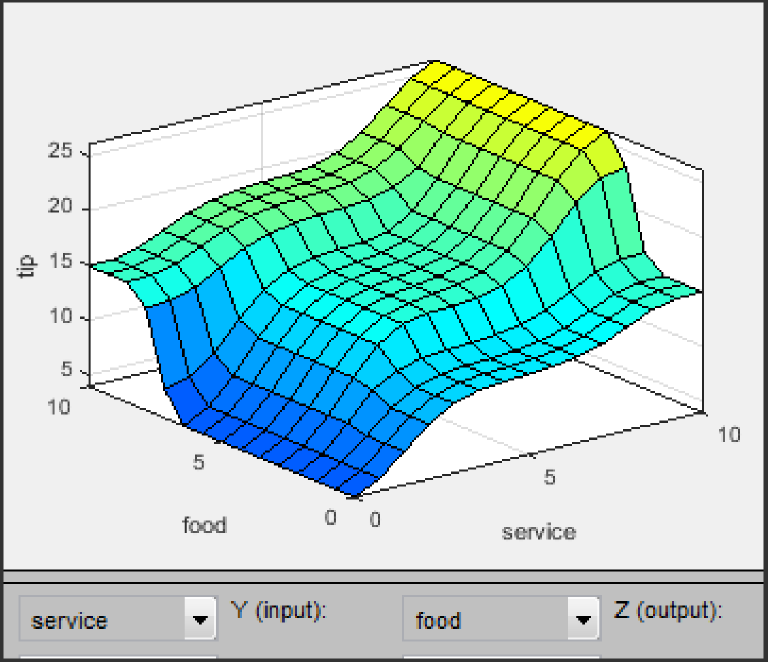
Example: Dinner for two (Sugeno)

Rule Viewer

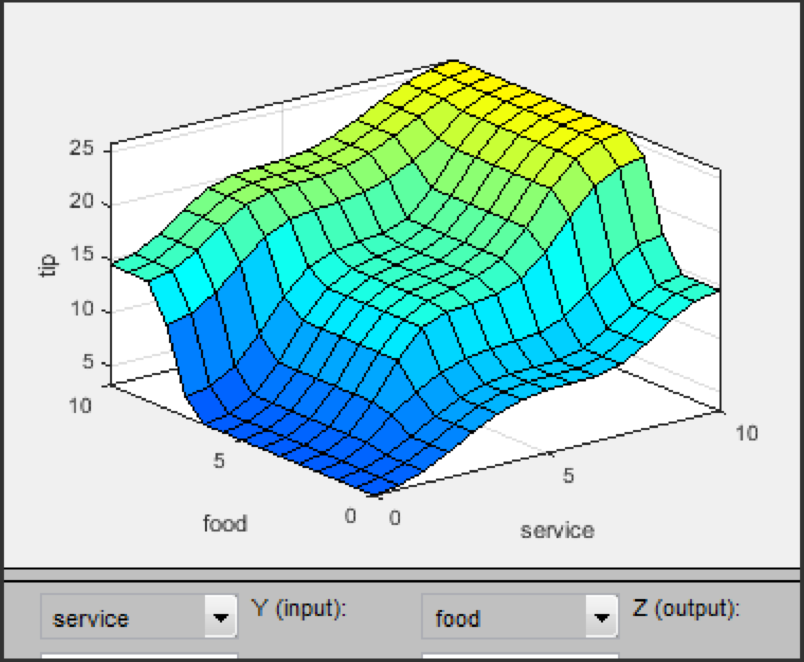


Example: Dinner for two

Mamdani



Sugeno



Mamdani vs. TSK

Advantages of the TSK Method

- It is computationally efficient.
- It works well with linear techniques (e.g., PID control).
- It works well with optimization and adaptive techniques.
- It has guaranteed continuity of the output surface.
- It is well suited to mathematical analysis.

Advantages of the Mamdani Method

- It is intuitive.
- It has widespread acceptance.
- It is well suited to human input.

Some Misconceptions on Fuzzy Logic

Fuzzy logic is the same as imprecise logic

Fuzzy logic is not any less precise than any other form of logic: it is an organized and mathematical method of handling *inherently* imprecise concepts.

Some Misconceptions on Fuzzy Logic

Fuzzy logic is a new way of expressing probability

Fuzzy logic and probability refer to different kinds of uncertainty. Fuzzy logic is specifically designed to deal with **imprecision** of facts (fuzzy logic statements), while probability deals with **chances** of that happening *(but still considering the result to be precise)*

Example:

- P(A) = 0.5 means that A may be true or may be false
- A logical value of 0.5 means both true and false at the same time

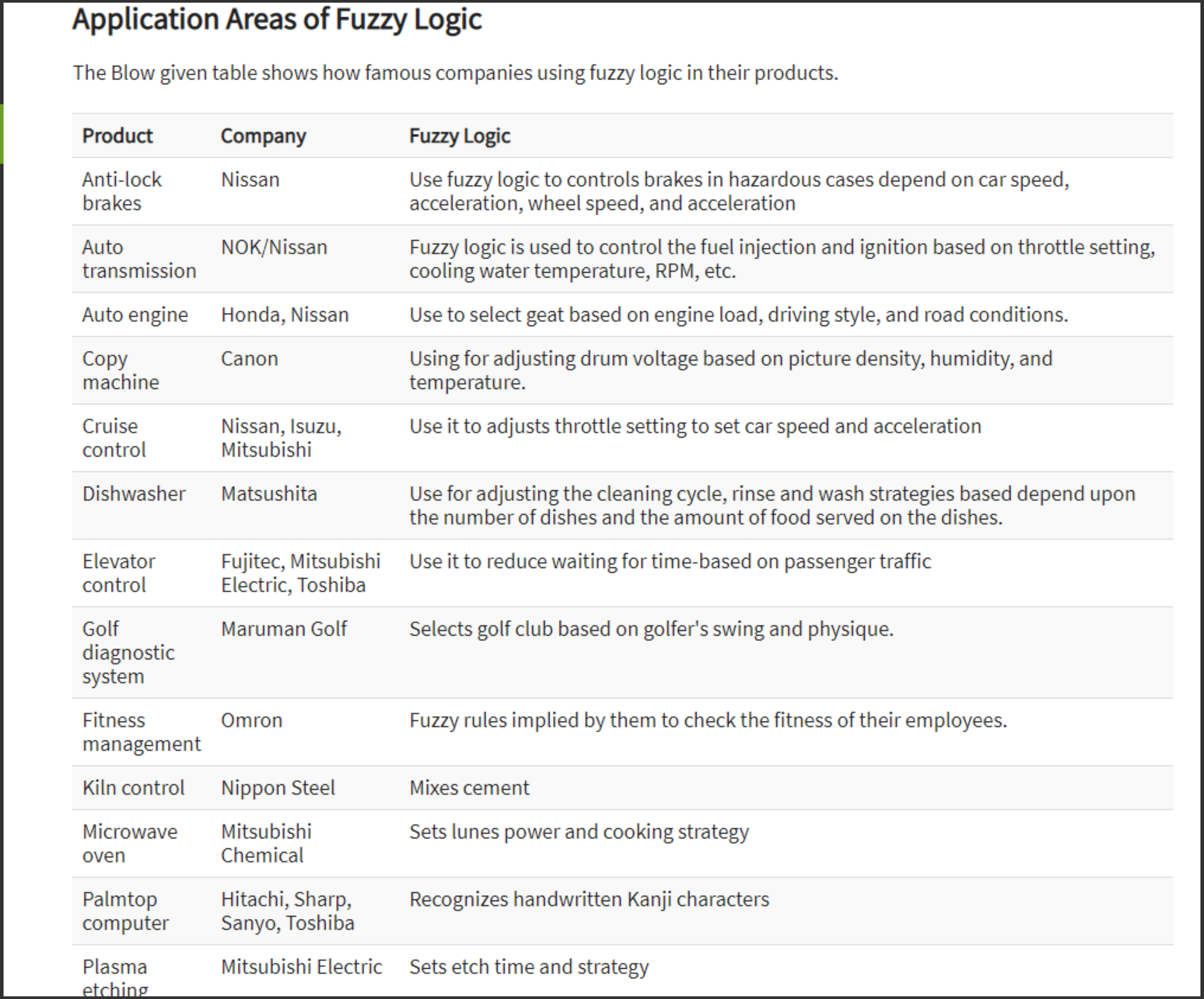
➡ However, still is a point of controversy

Applications of Fuzzy Logic

- Why use *fuzzy systems*?
  - Apply fuzziness (*and therefore accuracy*) to linguistically defined terms and rules
  - Lack of crisp or concrete mathematical models exist

- When do you avoid *fuzzy systems*?
  - Traditional approaches produce acceptable results
  - Crisp or concrete mathematical models exist and are easily implemented

Applications of Fuzzy Logic

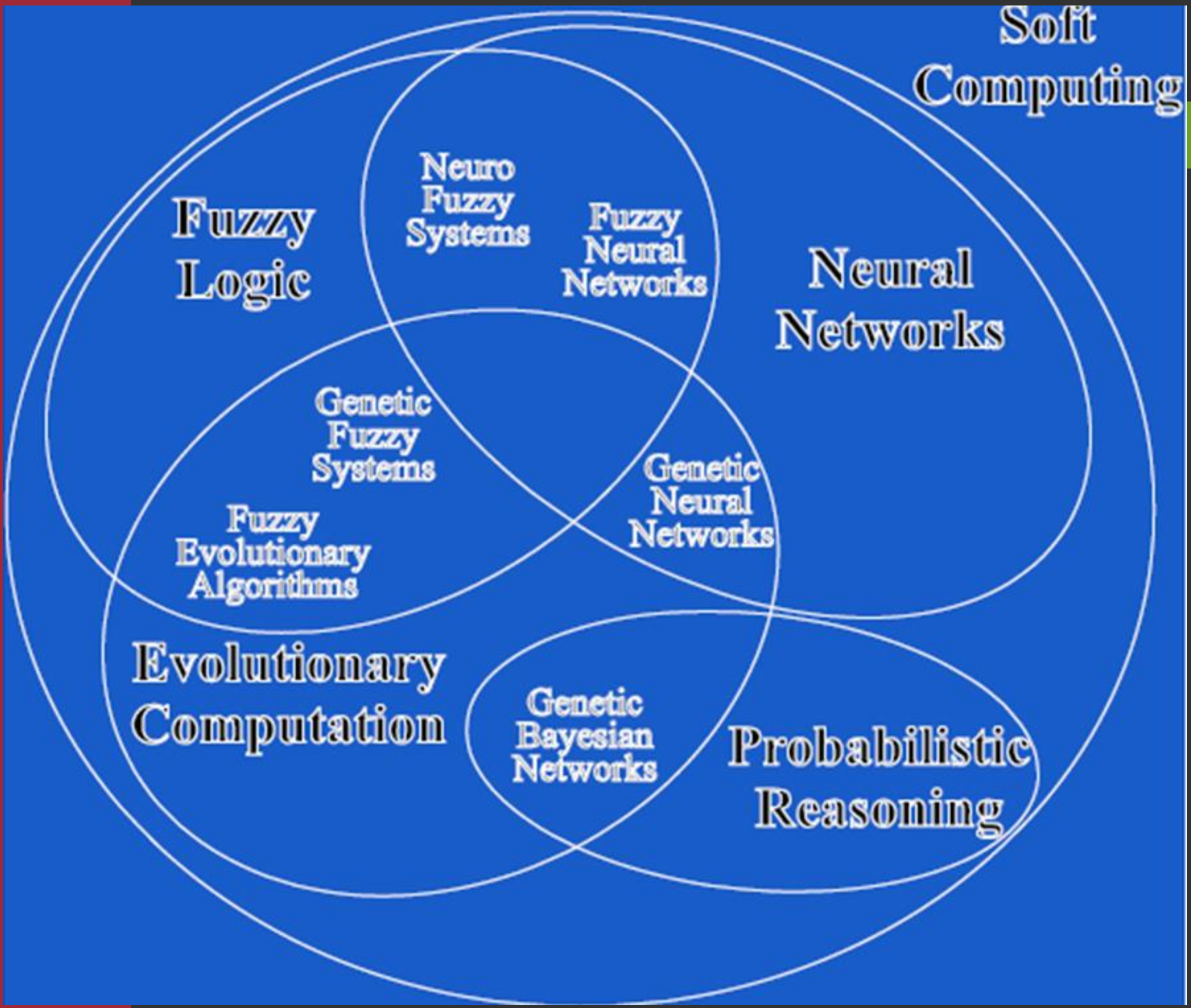


Application Areas of Fuzzy Logic

The Blow given table shows how famous companies using fuzzy logic in their products.

Product	Company	Fuzzy Logic
Anti-lock brakes	Nissan	Use fuzzy logic to controls brakes in hazardous cases depend on car speed, acceleration, wheel speed, and acceleration
Auto transmission	NOK/Nissan	Fuzzy logic is used to control the fuel injection and ignition based on throttle setting, cooling water temperature, RPM, etc.
Auto engine	Honda, Nissan	Use to select geat based on engine load, driving style, and road conditions.
Copy machine	Canon	Using for adjusting drum voltage based on picture density, humidity, and temperature.
Cruise control	Nissan, Isuzu, Mitsubishi	Use it to adjusts throttle setting to set car speed and acceleration
Dishwasher	Matsushita	Use for adjusting the cleaning cycle, rinse and wash strategies based depend upon the number of dishes and the amount of food served on the dishes.
Elevator control	Fujitec, Mitsubishi Electric, Toshiba	Use it to reduce waiting for time-based on passenger traffic
Golf diagnostic system	Maruman Golf	Selects golf club based on golfer's swing and physique.
Fitness management	Omron	Fuzzy rules implied by them to check the fitness of their employees.
Kiln control	Nippon Steel	Mixes cement
Microwave oven	Mitsubishi Chemical	Sets lunes power and cooking strategy
Palmtop computer	Hitachi, Sharp, Sanyo, Toshiba	Recognizes handwritten Kanji characters
Plasma etching	Mitsubishi Electric	Sets etch time and strategy

Hybridization

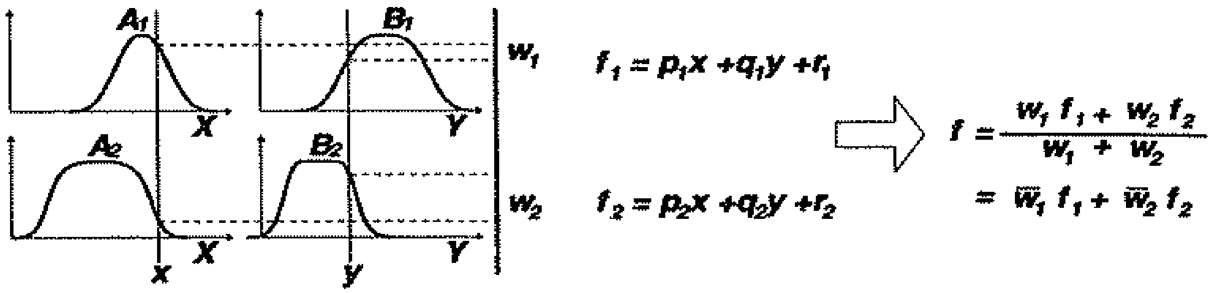


- FSs are neither capable of learning, adaptation or parallel computation, whereas these characteristics are clearly attributed to NNs.
- NNs lack flexibility, human interaction or knowledge representation, which lies at the core of FL.

ANFIS: Adaptive Neuro-Fuzzy Inference System

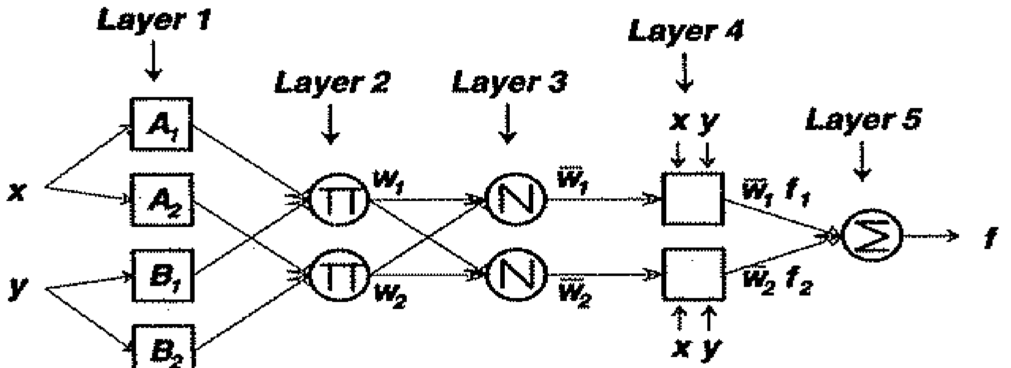
Sugeno

Rule 1: If x is A_1 and y is B_1, then $f_1 = p_1 x + q_1 y + r_1$,
Rule 2: If x is A_2 and y is B_2, then $f_2 = p_2 x + q_2 y + r_2$.



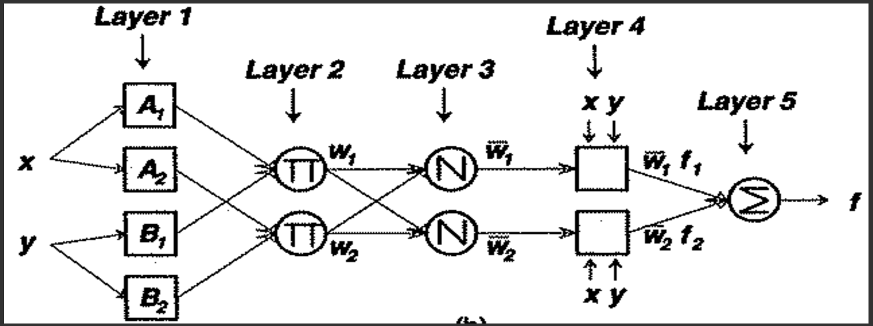
A_1
B_1
A_2
B_2
X
Y
x
y
w_1
w_2
$f_1 = p_1 x + q_1 y + r_1$
$f_2 = p_2 x + q_2 y + r_2$
$f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2}$
$= \bar{w}_1 f_1 + \bar{w}_2 f_2$

(a)



(b)

ANFIS: Architecture



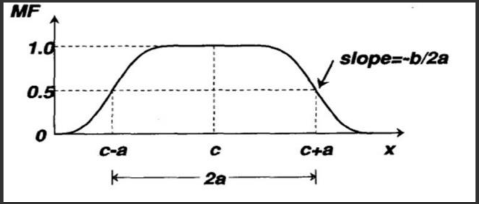
Layer 1: Every node *i* is an adaptive node with a node function:

$$O_{1,i} = \mu_{A_i}(x), \quad \text{for } i = 1, 2, \text{ or}$$
$$O_{1,i} = \mu_{B_{i-2}}(y), \quad \text{for } i = 3, 4,$$

where *x* (or *y*) is the input to node *i* and A_i (or B_{i-2}) is a linguistic label ("small", "large",…) associated with this node

The membership function for *A* or B (=A_1, A_2, B_1 or B_2) can be any appropriate parameterized membership function such as the generalized bell function:

$$\mu_A(x) = \frac{1}{1 + \left|\frac{x - c_i}{a_i}\right|^{2b}},$$

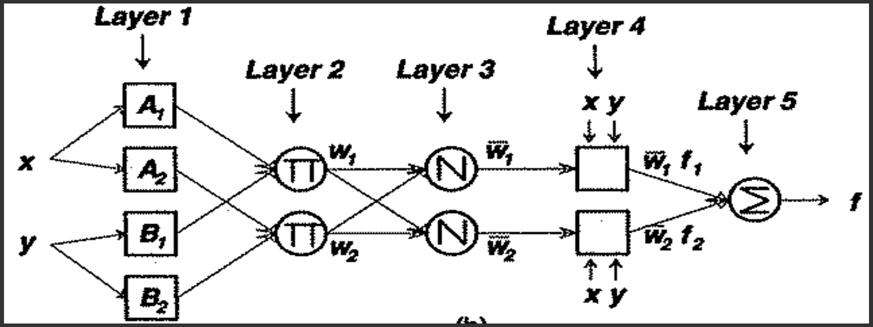


where $\{a_i, b_i, c_i\}$ is the parameter set (**premise parameters**)

Layer 2: Every node in this layer is a fixed node labeled Π, whose output is the result of applying any other T-norm operator that performs fuzzy AND

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \quad i = 1, 2.$$

ANFIS: Architecture



Layer 3: Every node in this layer is a fixed node labeled N. The *i*th node calculates the ratio of the *i*th rule's firing strength to the sum of all rules' firing strengths

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2.$$

Outputs of this layer are called **normalized firing strengths**

Layer 4: Every node in this layer is an adaptive node with a node function

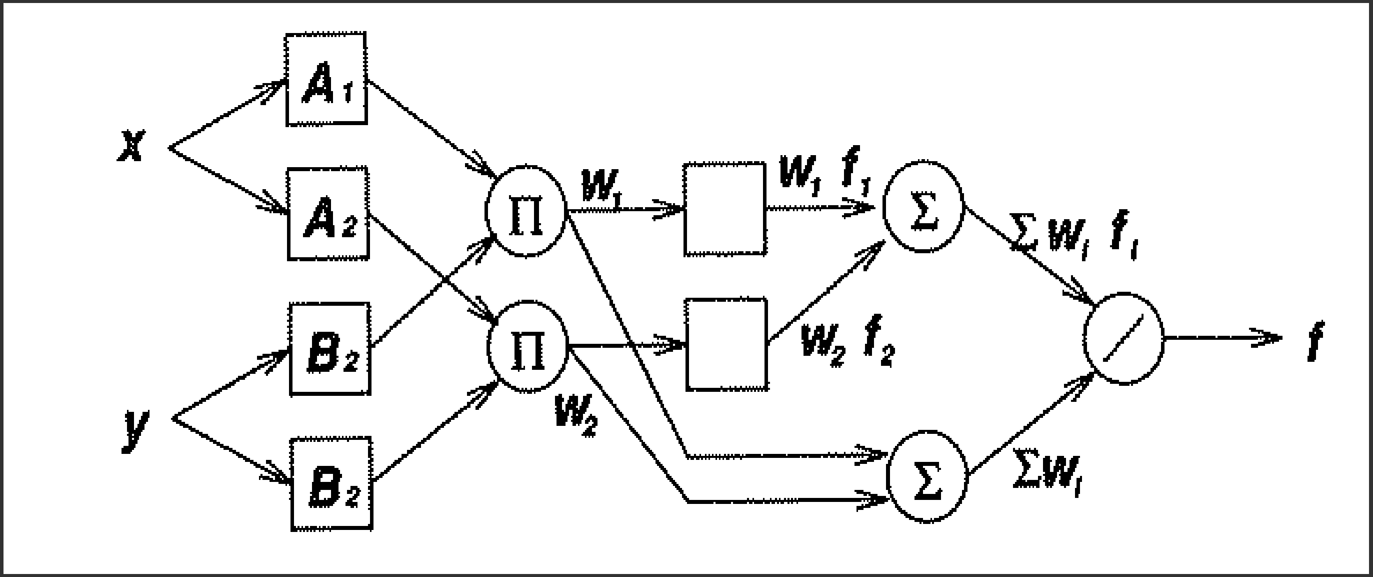
$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i),$$

where $\overline{w}_i$ is a normalized firing strength from layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set of this node (**consequent parameters**)

Layer 5: The single node in this layer is a fixed node labeled Σ, which computes the overall output as the summation of all incoming signals:

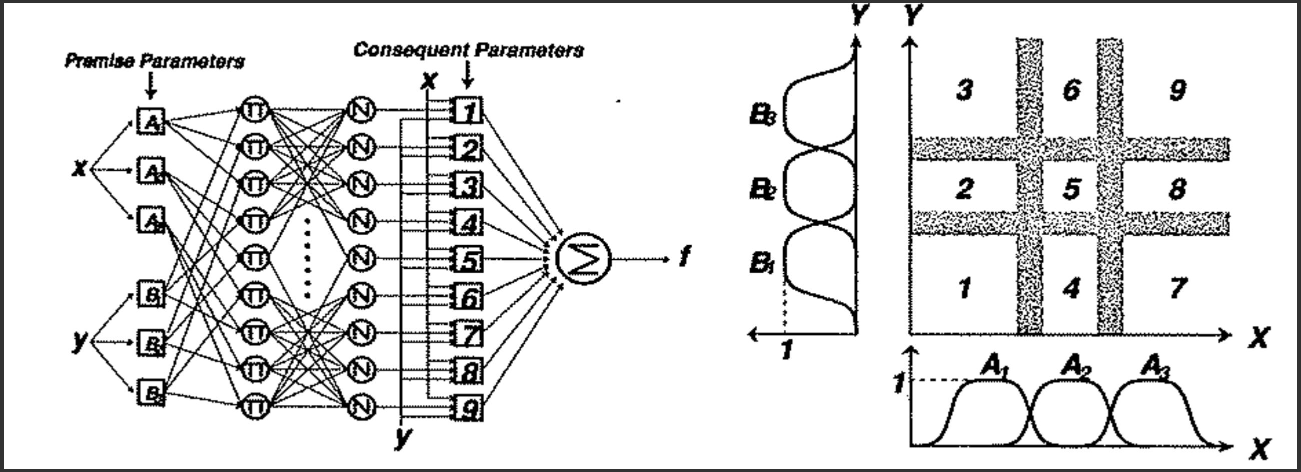
$$O_{5,1} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

ANFIS: Another Architecture

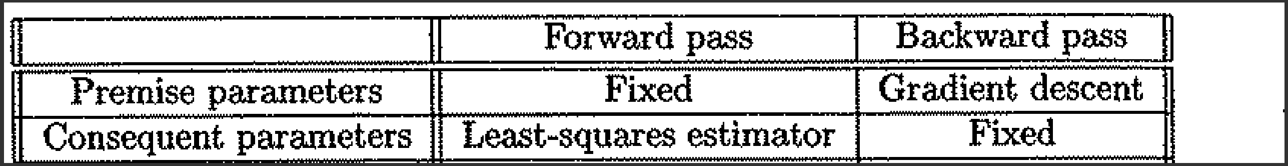


Weight normalization is performed at the very last layer

ANFIS: Hybrid Learning



The premise part of a rule defines a fuzzy region, while the consequent part specifies the output within the region



	Forward pass	Backward pass
Premise parameters	Fixed	Gradient descent
Consequent parameters	Least-squares estimator	Fixed

S1= Set of premise (nonlinear) parameters
S2= Set of consequent (linear) parameters

Example 1: Two-Input Sinc Function

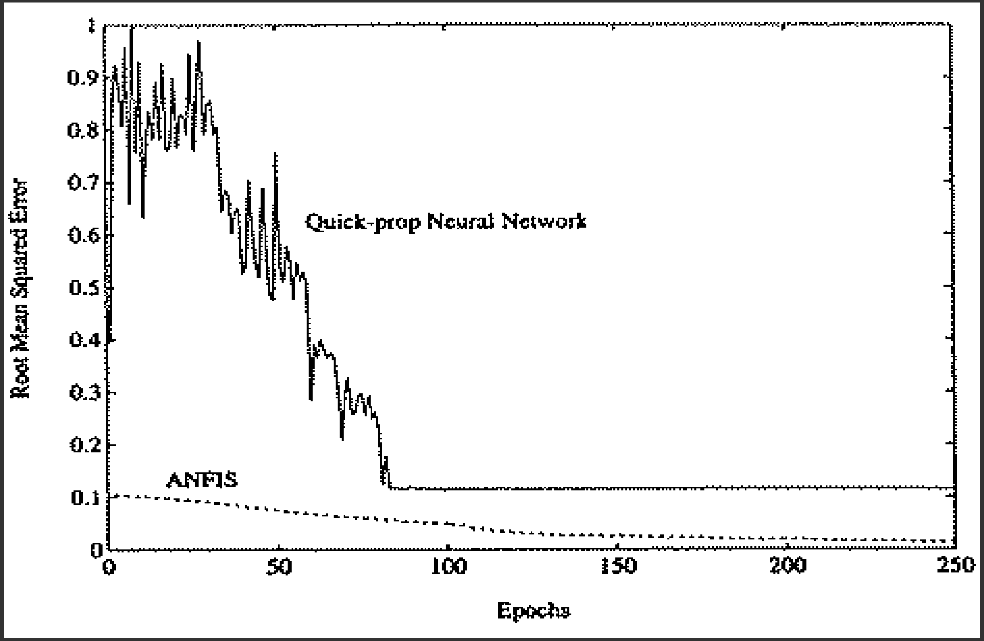
$$z = \text{sinc}(x, y) = \frac{\sin(x)\sin(y)}{xy}.$$

ANFIS

- 121 training data pairs
- 16 rules with 4 membership functions assigned to each input
- 72 total fitting parameters (24 premise and 48 consequent)

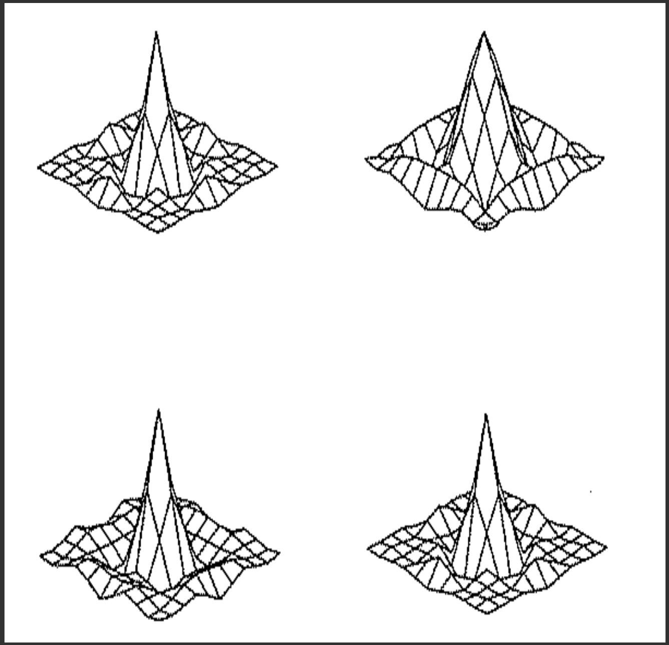
2-18-1 Backpropagation MLP

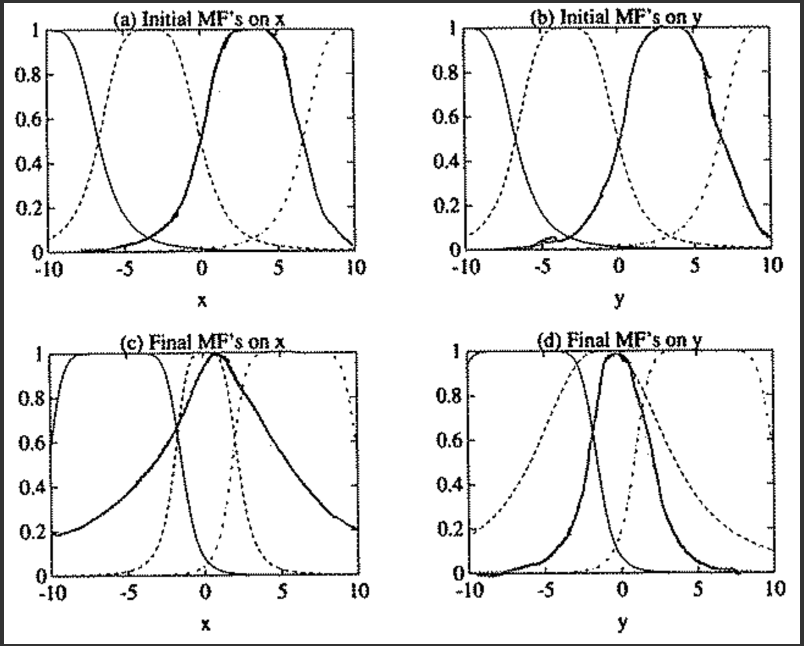
- 121 training data pairs
- Trained with quick propagation
- 73 total fitting parameters (connection weights and thresholds)



ANFIS approximates the highly nonlinear surface more effectively than the MLP

Example 1: Two-Input Sinc Function



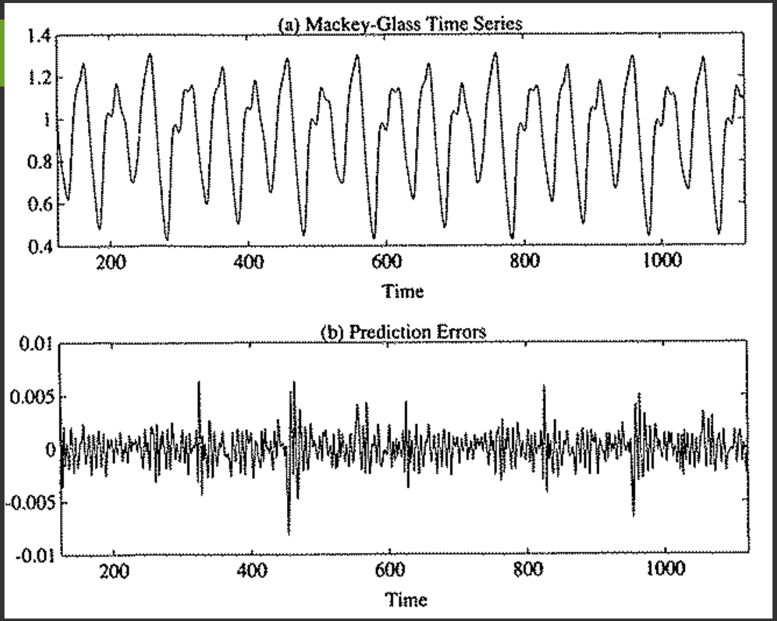


- Training data and reconstructed surfaces at different epochs: 0.5, 99.5 and 249.5

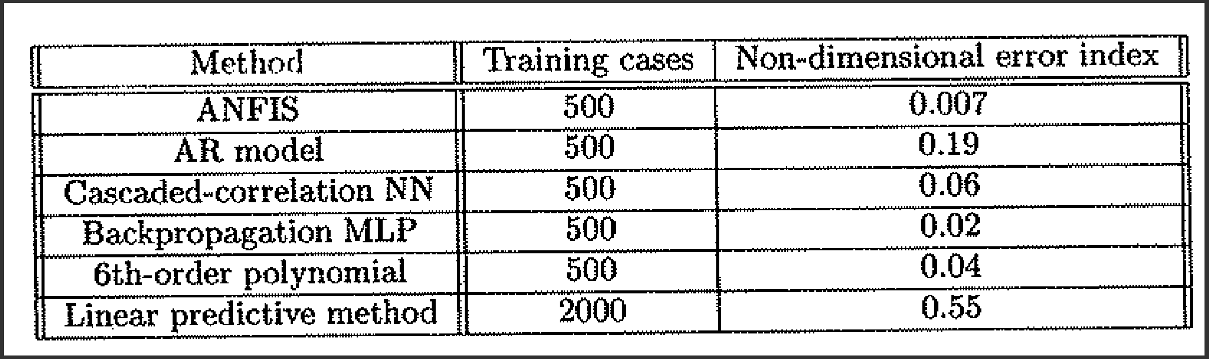
- Initial and final membership functions

Example 2: Predicting Mackey-Glass Chaotic Time Series

- Desired and predicted values for both training and test data are essentially the same



NDEI: root mean square error divided by the standard deviation of the target series;
Prediction: 500 values test data set



Method	Training cases	Non-dimensional error index
ANFIS	500	0.007
AR model	500	0.19
Cascaded-correlation NN	500	0.06
Backpropagation MLP	500	0.02
6th-order polynomial	500	0.04
Linear predictive method	2000	0.55

PARAM
(104)

(693)
(540)

Demo ANFIS

Synthetic Data

Matlab:

```
>> load fuzex1trnData.dat (single input - single output)
>> anfisedit
```

Demo ANFIS

Automobile Data

Matlab:

```
>> load auto.mat (multiple input - single output)
>> anfisedit
```

This example uses demographic and trip data from 100 traffic analysis zones in Delaware. Contains five demographic factors as input variables: population, number of dwelling units, vehicle ownership, median household income, and total employment and one output variable, number of automobile trips.

Advantages of ANFIS

The remarkable generalization capability of ANFIS is due to:

- ANFIS can achieve a highly nonlinear mapping, far superior to MLP and common linear methods of similar complexity.
- The initial parameters are intuitively reasonable and all the input space is covered properly (fast convergence).
- ANFIS requires fewer adjustable parameters than those required in other Neural Network structures and, specifically, backpropagation MLPs.
- ANFIS consists of fuzzy rules that are local mappings instead of global ones, then facilitate the minimal disturbance principle (the adaptation should not only reduce the output error for the current training pattern but also minimize disturbance to response already learned).

Disadvantages of ANFIS

- ANFIS only deals with **parameter identification**, we still need methods for **structure identification** to determine an initial ANFIS architecture.
- High number of rules when the number of inputs is high, reducing the interpretability.
- Curse of dimensionality and high computational cost (but allows parallelization).

Subtractive Clustering

- **Clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense) to each other than to those in other groups (clusters).

- Subtractive clustering is a fast one-pass algorithm for estimating the number of clusters and the cluster centers in a data set.

Subtractive Clustering

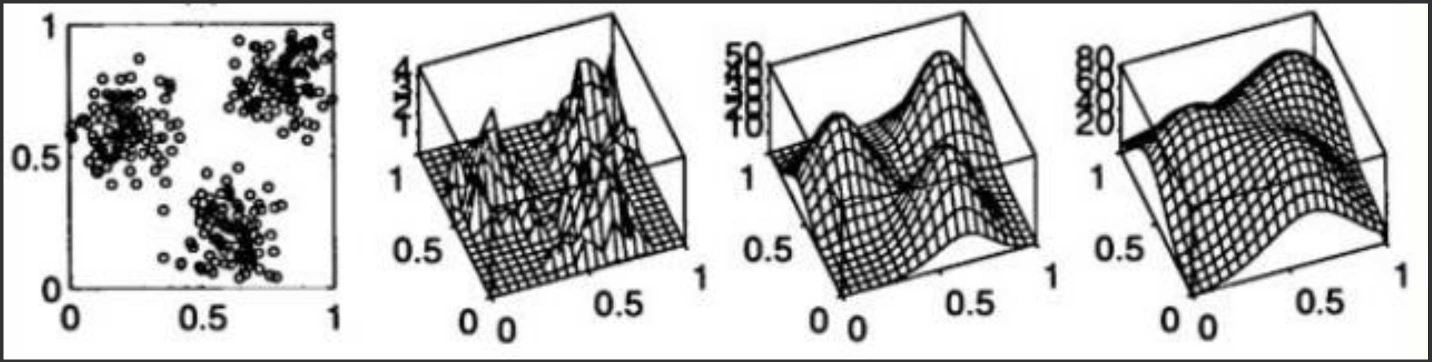
- Consider a collection of *n* data points in an *M*-dimensional space.
- Each data point is considered as a candidate for a cluster center.
- The data points are normalized within a hypercube [0,1].
- A *density measure* at data point x_i is defined as:

$$D_i = \sum_{j=1}^{n} e^{-\frac{||x_i - x_j||^2}{(r_a/2)^2}}$$

r_a (radius) positive number, defines a neighborhood; ||.|| denotes the Euclidean distance

- A data point will have a high density value if it has many neighboring data points.
- Once density measure is calculated at each data point, the one with the highest value is selected as the first cluster center.

Subtractive Clustering



r_a = 0.05 0.3 0.5

r_a may affect the smoothness of mountain functions

Subtractive Clustering

- To find the next cluster center, the density measure for each data point is revised subtracting the influence of the first cluster (data point: x_{c1}; its density measure: D_{c1})

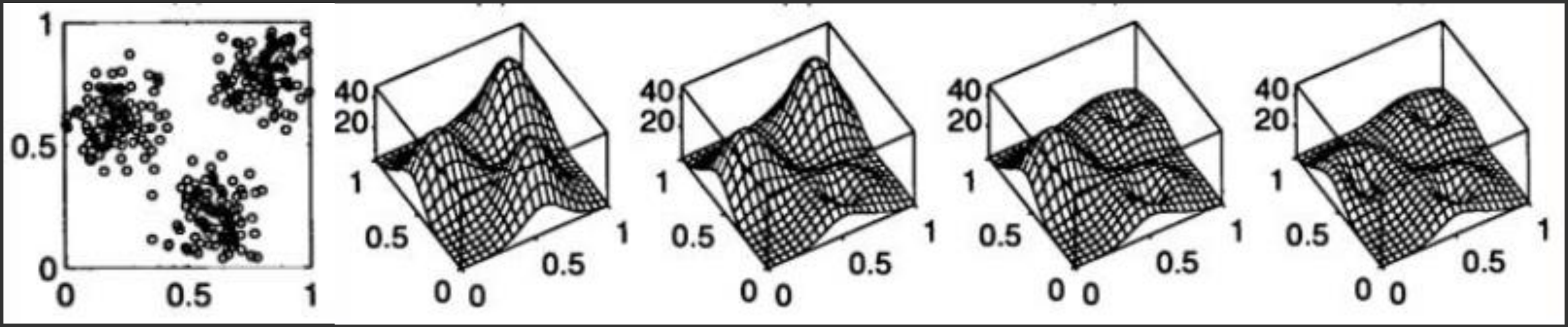
$$D_i = D_i - D_{c1} e^{-\frac{||x_i - x_{c1}||^2}{(r_b/2)^2}}$$

r_b (radius) positive number, defines a neighborhood with reductions in density measure;

$r_b = \eta\, r_a$ (squash factor > 1)

- The data points near the first cluster center x_{c1}, will have significantly reduced density measure, making the points unlikely to be selected as the next cluster
- Once the density measure for each data point is revised, the next cluster center is selected.

Subtractive Clustering



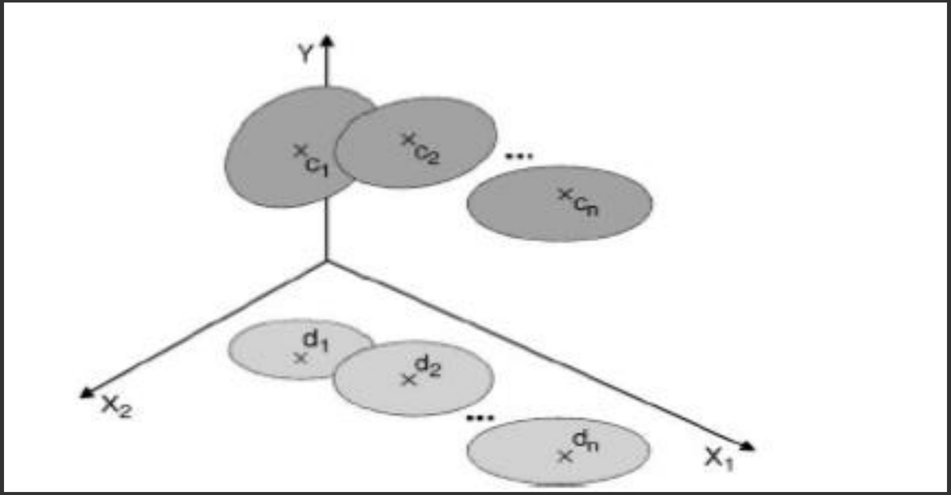
Cluster centers are selected, and mountains are destructed sequentially

- For ending the clustering process there are criteria for accepting and rejecting cluster centers that help avoid marginal cluster centers
  - If *Ratio* > *Accept ratio* then accept new cluster
  - If *Reject ratio* < *Ratio* <= *Accept ratio* and *high distance* other clusters then accept new cluster
  - If *Ratio* <= *Reject ratio* then stop

where $Ratio = D_k/D_1$; *Accept ratio* and *Reject ratio* are positive numbers [0,1]

Subtractive Clustering

- The clusters are projected onto the input space in order to find the antecedent parts of the fuzzy rules.
- The consequent parts of the rules can then be simple functions.
- In this way, one cluster corresponds to one rule of the TSK model.

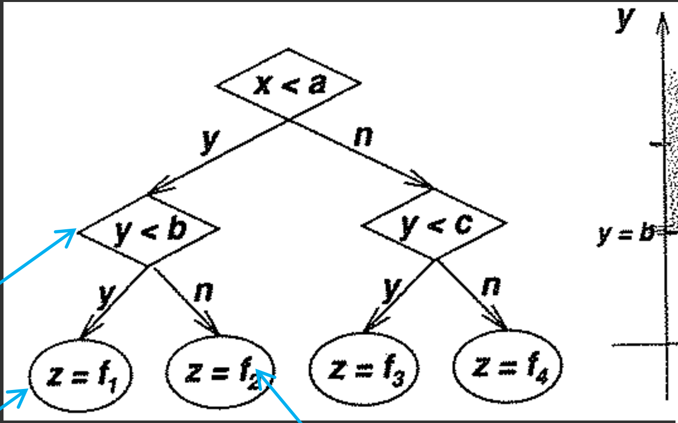


CART: Classification and Regression Trees

- CART generates a tree partitioning of the input space, which relieves the “curse of dimensionality” problem (num. of rules increasing exponentially with num. of inputs)

CART: Classification and Regression Trees

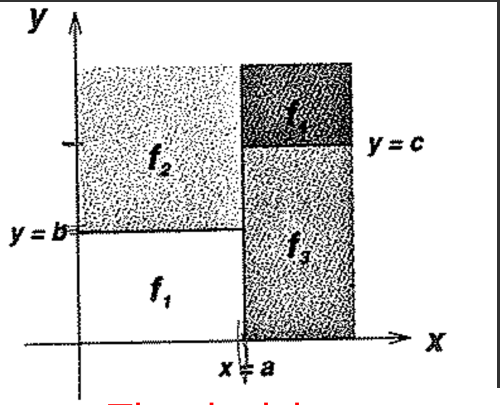
Binary regression tree with two inputs and one output



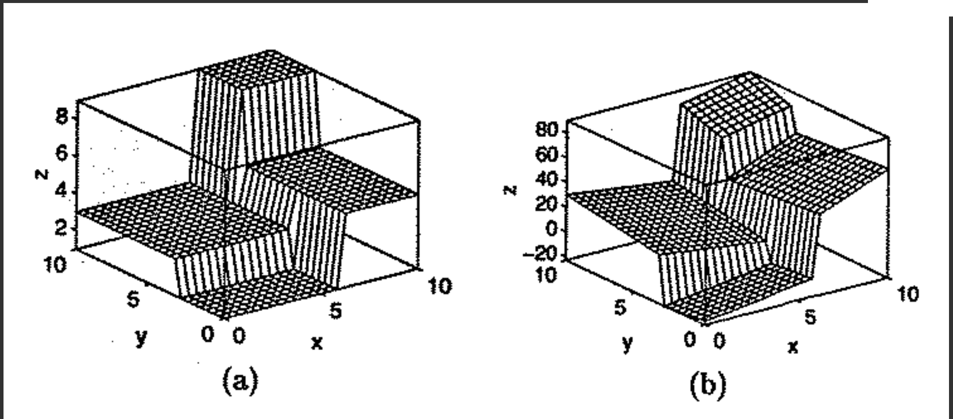
Internal node

Terminal node

Constants or equations



The decision tree partitions the input space into 4 regions



Constants: a=6, b=3, c=7
f1=1, f2=3, f3=5, f4=9

Linear functions:
f1= 2x-y-20,
f2= -2x+2y+10,
f3=6x-y+5,
f4=3x+4y+20

CART: Classification and Regression Trees

CART first **grows** the tree extensively based on a training data set, and then **prunes** the tree back based on a minimum cost-complexity principle

Tree Growing

- CART grows a decision tree by determining a succession of splits (decision boundaries) that partition the training data into disjoint subsets
- Starting from the root node (contains all the training data), an exhaustive search is performed to find the split that best reduces an error measure (cost function)
- The data set is partitioned into two subsets and the same splitting method is applied to both child nodes
- This recursive procedure terminates either when the error measure associated with a node falls below a certain tolerance level, or when the error reduction does not exceed a certain threshold value

CART: Classification and Regression Trees

For a regression tree, the error measure that quantifies the performance of a node *t* in separating data into disjoint subsets is usually taken as the square error (or residual) of a local model employed to fit the data set of the node:

$$E(t) = \min_{\boldsymbol{\theta}} \sum_{i=1}^{N(t)} (y_i - d_t(\mathbf{x}_i, \boldsymbol{\theta}))^2,$$

where $\{x_i, y_i\}$ is a data point, $d_t(\mathbf{x}_i, \boldsymbol{\theta})$ is a local model for node *t* (with modifiable parameter θ) and E(*t*) is the mean-square error of fitting the local model to the data set in the node

Constant $(d(x, \theta) = \theta)$: the minimizing of *E*(*t*) is the average value of the desired output

$$\theta^* = \frac{1}{N(t)} \sum_{i=1}^{N(t)} y_i.$$

CART: Classification and Regression Trees

Linear model (d(x, θ) = linear model with linear parameters): least-squares methods to identify the minimizing θ*

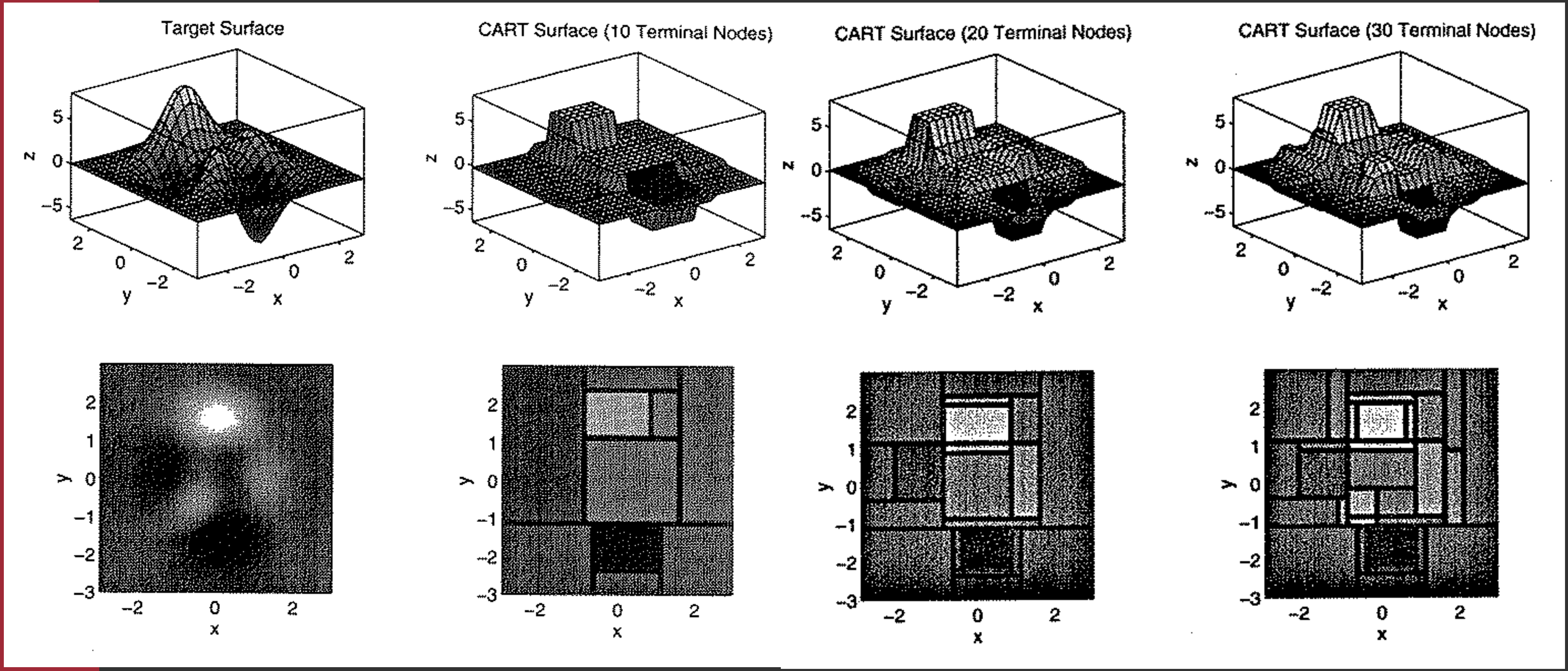
For any split *s* of node *t* into t_l and t_r, the change in the error measure is expressed as: $\Delta E(s,t) = E(t) - E(t_l) - E(t_r)$

the best split *s** is the one that maximizes the decrease in the error measure:

$$\Delta E(s^*,t) = \max_{s \in S} \Delta E(s,t).$$

The strategy for growing a regression tree is to split nodes (or data set) iteratively and thus maximize the decrease in $E(T) = \sum_{t \in \tilde{T}} E(t)$, i.e. the overall error measure (or cost) of the tree.

CART: Classification and Regression Trees



CART: Classification and Regression Trees

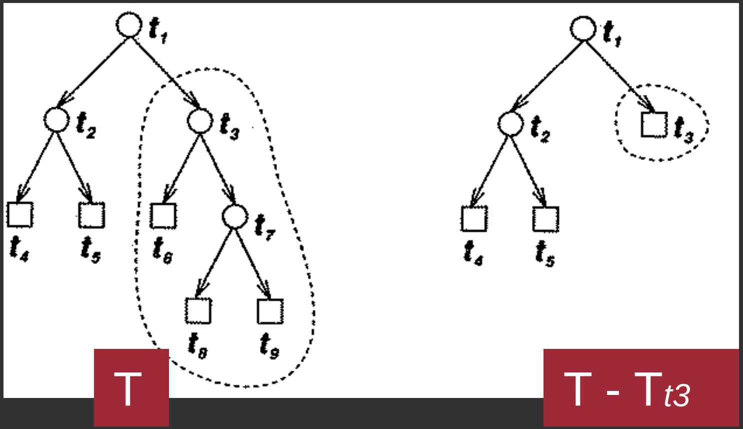
Tree Pruning

▪ The tree that the growing procedure yields is often too large and it is biased towards the training data set (overfitting)

Principle of minimum cost-complexity: prune the tree finding the weakest subtree in it, i.e. considering both the training error measure and the number of terminal nodes (measure of tree's complexity)

The internal node with the smallest α_t is chosen as the target node for shrinking

$$\alpha_t = \frac{E(t) - E(T_t)}{\underbrace{|\tilde{T}_t| - 1}_{\text{number of terminal nodes in t}}}.$$



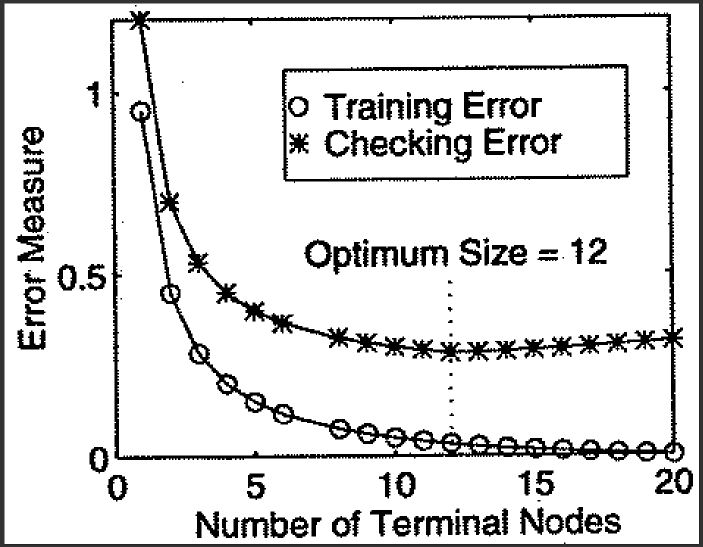
CART: Classification and Regression Trees

Therefore, a tree-pruning cycle consists of the following tasks:

1. Calculate α_t for each internal node t in Ti
2. Find the minimal α_t and choose T - T_t as the next minimizing tree

Next step: select one of these candidate trees as the optimum-size tree:

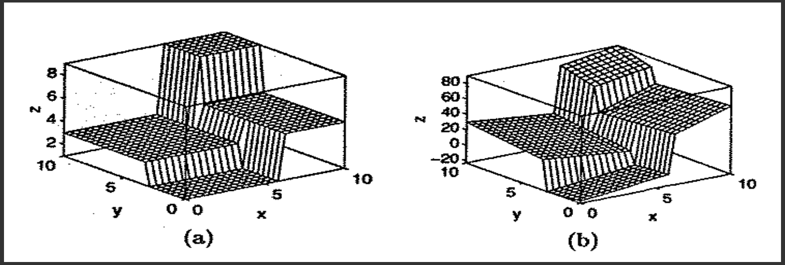
- use an independent test (checking)
- cross-validation



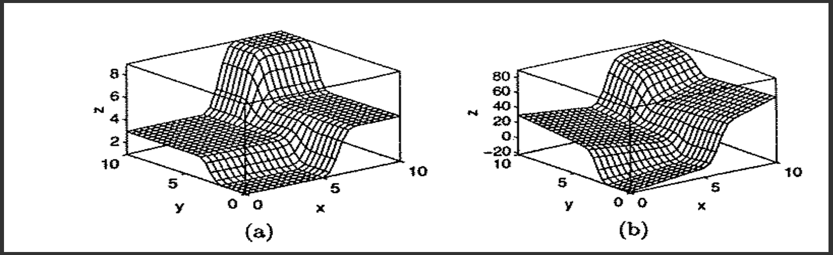
CART - ANFIS

CART is used to find the number of ANFIS rules and the initial locations of membership functions before training

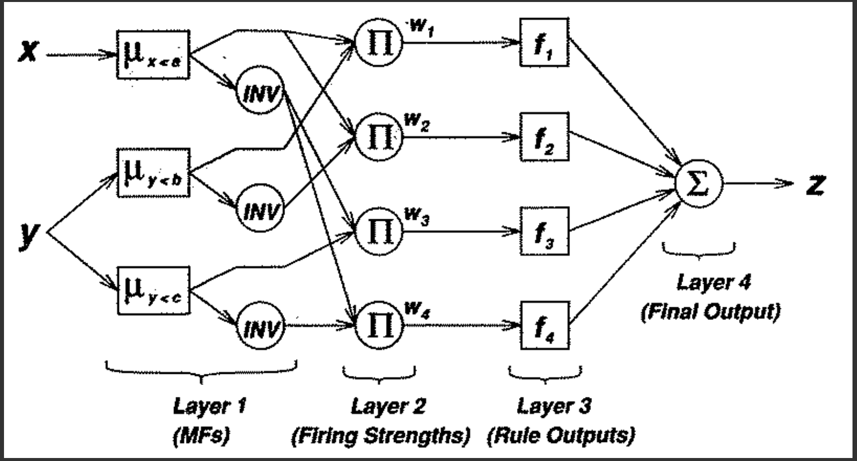
$$\begin{cases} \text{If } x < a \text{ and } y < b, \text{ then } z = f_1. \\ \text{If } x < a \text{ and } y \geq b, \text{ then } z = f_2. \\ \text{If } x \geq a \text{ and } y < c, \text{ then } z = f_3. \\ \text{If } x \geq a \text{ and } y \geq c, \text{ then } z = f_4. \end{cases}$$



Crisp: discontinuous boundaries



Fuzzy: continuous boundaries



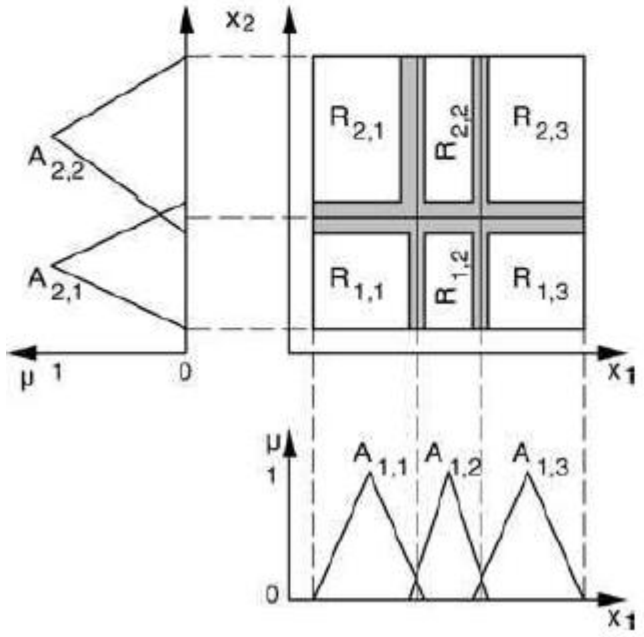
Extracting Mamdani Models from Data

- Since, besides prediction, understandability is of prime concern, the resulting fuzzy model should offer insights into the underlying system.

- To achieve this, different approaches exist that construct grid-based rule sets defining a global granulation of the input space, as well as fuzzy graph based structures.

Extracting Grid-Based Fuzzy Models from Data

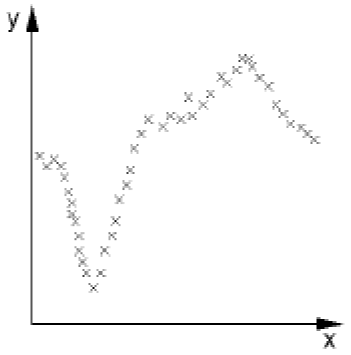
Grid-based rule sets model each input variable through a usually small set of linguistic values. The resulting rule base uses all or a subset of all possible combinations of these linguistic values for each variable, resulting in a global granulation of the feature space into "tiles":

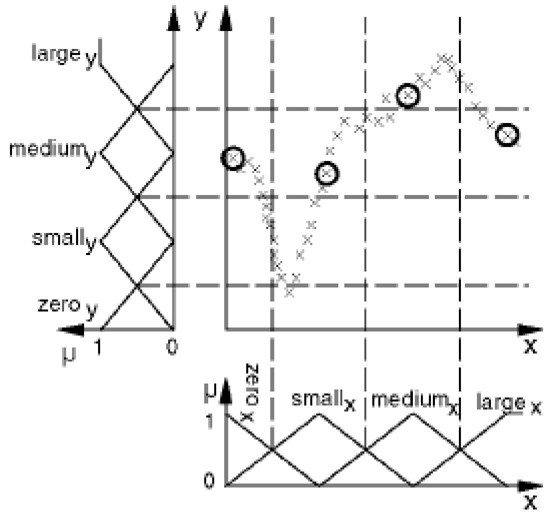


Extracting Grid-Based Fuzzy Models from Data: Wang & Mendel

- Wang & Mendel presented a straightforward approach to learning fixed-grid Mamdani models.
- First, a predefinition of the granulation of all input variables and also the output variable is performed.
- Then, a sweep through the entire dataset determines the closest example to the geometrical center of each rule, assigning the closest output fuzzy value to the corresponding rule.

Extracting Grid-Based Fuzzy Models from Data: Wang&Mendel



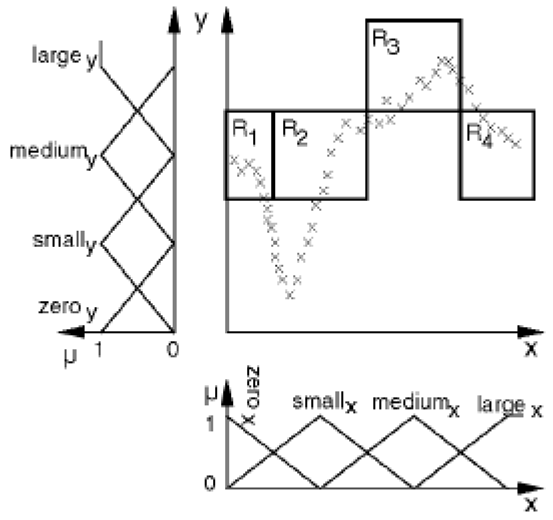


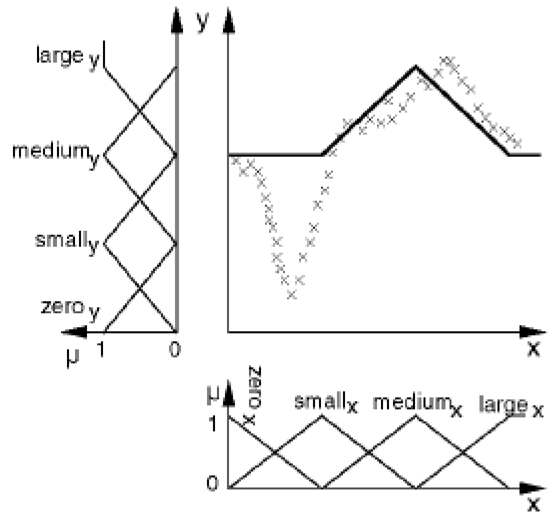
R_1 : IF x IS $zero_x$ THEN y IS $medium_y$

R_2 : IF x IS $small_x$ THEN y IS $medium_y$

R_3 : IF x IS $medium_x$ THEN y IS $large_y$

R_4 : IF x IS $large_x$ THEN y IS $medium_y$

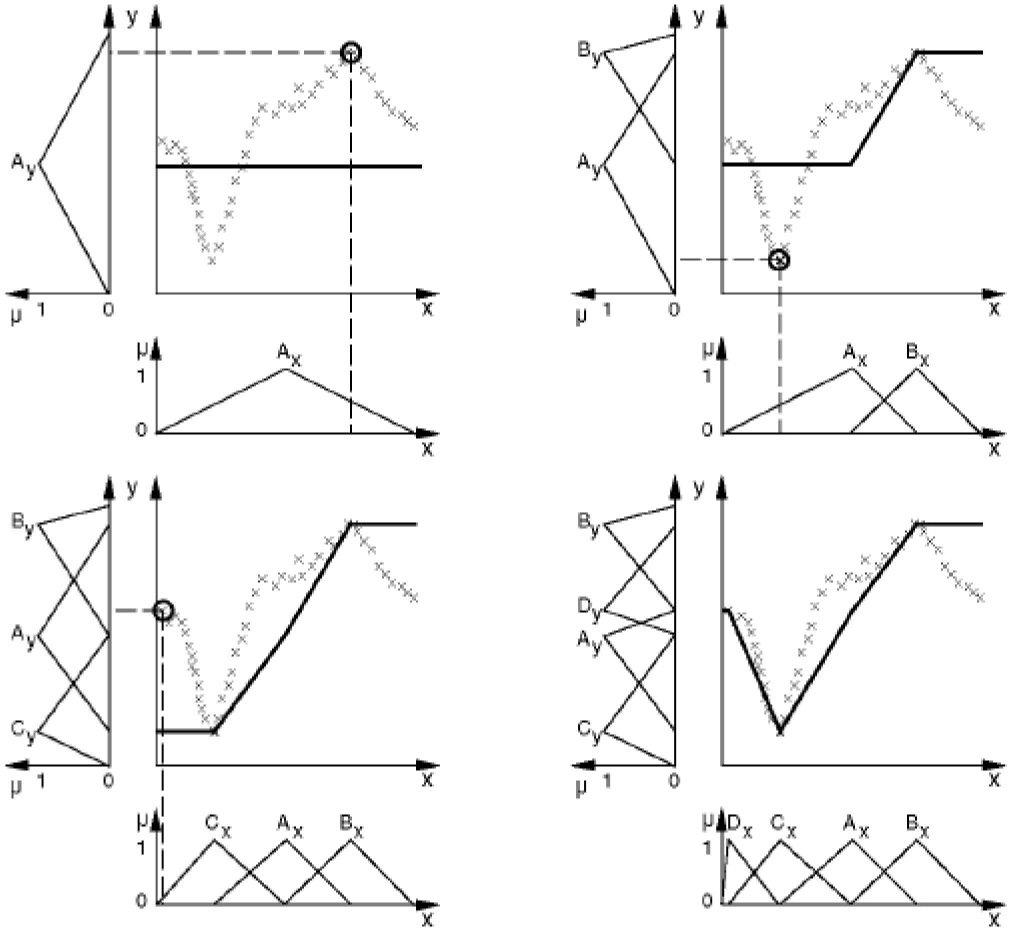




Extracting Grid-Based Fuzzy Models from Data: Higgins & Goodman

- For practical applications, it is clear that using such a predefined fixed grid, results in a fuzzy model that will either not fit the underlying functions very well or consist of a large number of rules.
- Higgins & Goodman approach automatically determine the granulations of both input and output variables.
- Initially only one MF is used to model each of the input variables as well as the output variable, resulting in one large rule covering the entire feature space.
- Subsequently, new MF are introduced at points of maximum error.
- This is done until a maximum number of divisions is reached or the approximation error remains below a certain threshold.

Extracting Grid-Based Fuzzy Models from Data: Higgins&Goodman



IF x IS A_x THEN y IS A_y

IF x IS B_x THEN y IS B_y

IF x IS C_x THEN y IS C_y

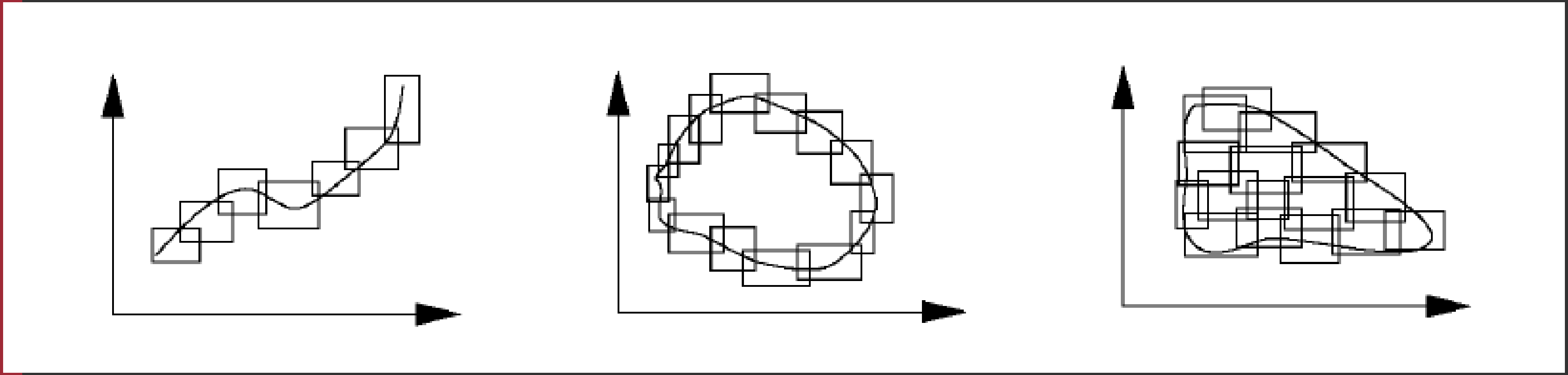
IF x IS A_x THEN y IS D_y
IF x IS B_x THEN y IS B_y
IF x IS C_x THEN y IS C_y
IF x IS D_x THEN y IS D_y

This approach is able to model extrema much better than Wang & Mendel algorithm, but has a strong tendency to concentrate on outliers

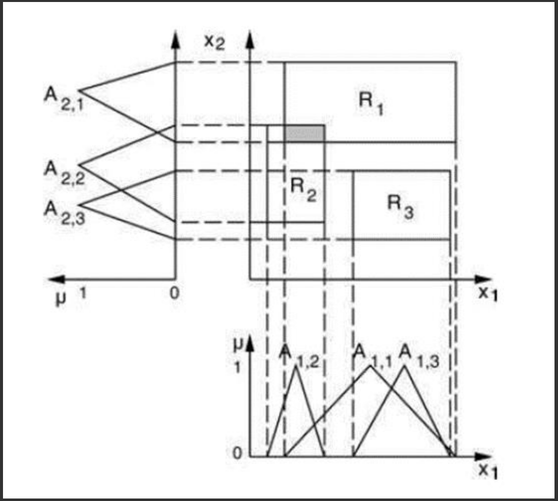
Extracting Fuzzy Graphs from Data

- We mentioned before that especially in high-dimensional feature spaces a global granulation results in a large number of rules.
- For these tasks a fuzzy graph based approach is more suitable.
- A possible disadvantage is the potential loss of interpretation.

Fuzzy Graphs

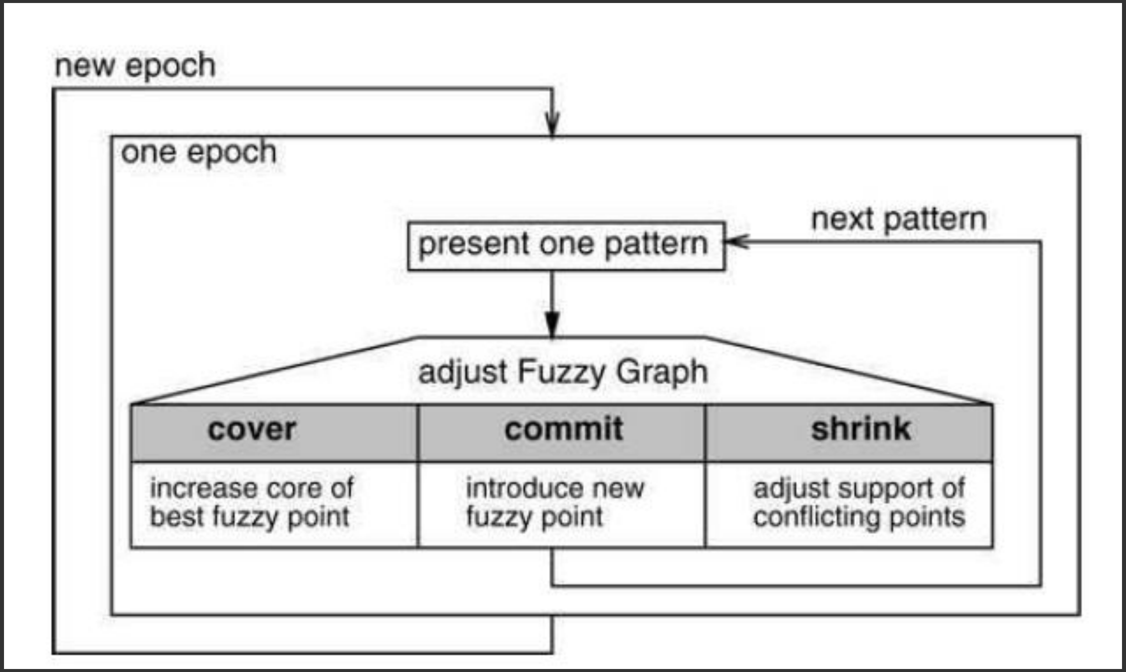


In contrast to a global grid, specified through linguistic variables, fuzzy graphs employ **individual membership functions for each fuzzy rule**; that is, the membership function of the constraint on the joint variable is defined individually for each rule.



Extracting Fuzzy Graphs from Data: Berthold and Huber

- The only parameter specified by the user is the granulation of the output variable *y*; that is, the number and shape of the MFs of *y* have to be determined manually.
- In most applications this is no disadvantage, because it enables the user to define areas of interest where a finer granulation is desired.


new epoch
one epoch
present one pattern
next pattern
adjust Fuzzy Graph
cover
commit
shrink
increase core of best fuzzy point
introduce new fuzzy point
adjust support of conflicting points

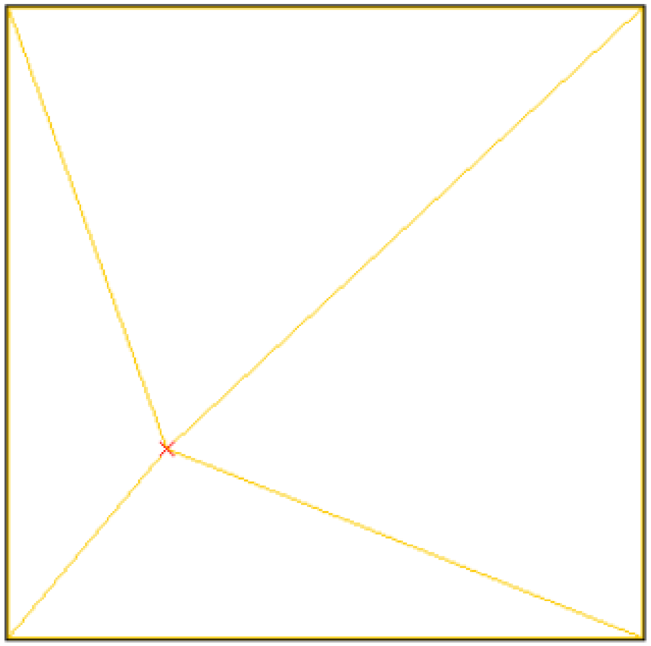
Extracting Fuzzy Graphs from Data: Berthold and Huber

The three main steps of the algorithm introduce new fuzzy points when necessary and adjust the core and support regions of existing ones in case of conflict:

- **Covered**: if the new training pattern lies inside the support-region of an already existing fuzzy point and it belongs to the correct output-region, the core-region of this fuzzy point is extended to cover the new pattern.
- **Commit**: if a new pattern is not covered, a new fuzzy point belonging to the correct output-region will be introduced. The new example pattern is assigned to its core, whereas the support-region is initialized “infinite”; that is, the new fuzzy point covers the entire domain.
- Shrink: if a new pattern is correctly covered by an already existing fuzzy point of a conflicting region, this fuzzy point’s support-area will be reduced so that the conflict is solved.

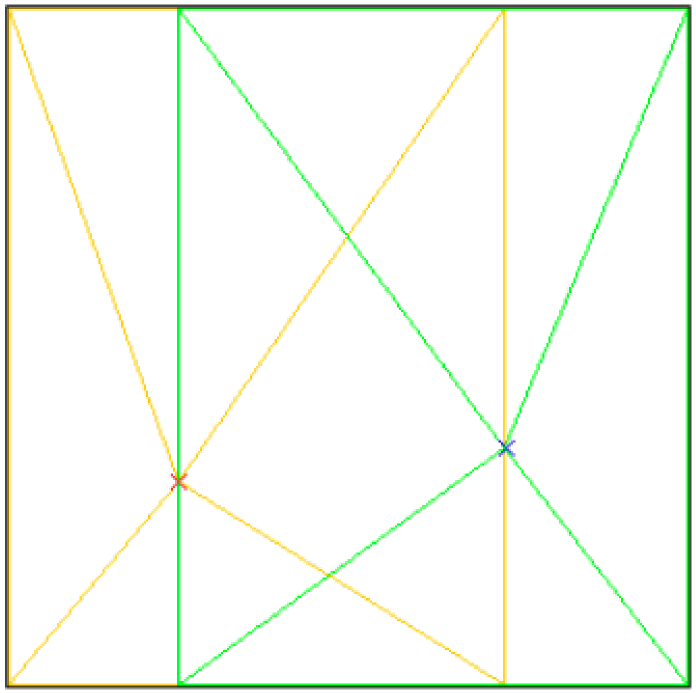
The algorithm terminates after few iterations over the set of example patterns.

Extracting Fuzzy Graphs from Data: Berthold and Huber



- insert general rule for first example pattern

Extracting Fuzzy Graphs from Data: Berthold and Huber

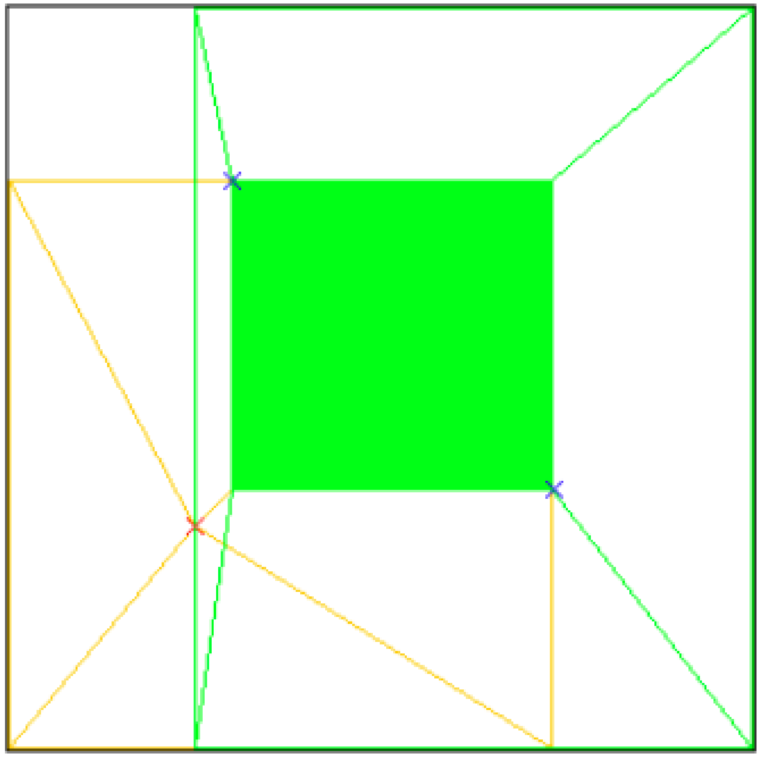


- suppose that 2nd pattern is from different class

$\Rightarrow$ new rule is inserted for 2nd pattern

- also adjust 1st (conflicting) rule

Extracting Fuzzy Graphs from Data: Berthold and Huber



- suppose that 3rd pattern is from same class as 2nd one

$\Rightarrow$ adjust free feature to avoid conflict with 3rd pattern

- and so on. . .

➢ The final set of fuzzy points forms a fuzzy graph, where each fuzzy point is associated with one output region.