Supplementary Topic:

Neuro-Fuzzy Systems

Brief Introduction to Fuzzy Theory

- **Classical Logic:** only two truth values *true* and *false*

  Classical Set Theory: either *is element of* or *is not element of*

- The bivalence of the classical theories is often inappropriate.

  Illustrative example: **Sorites Paradox** (greek. *sorites*: pile)

  - One billion grains of sand are a pile of sand. *(true)*
  - If a grain of sand is taken away from a pile of sand, the remainder is a pile of sand. *(true)*

  It follows:

  - 999 999 999 grains of sand are a pile of sand. *(true)*

  Repeated application of this inference finally yields

  - A single grain of sand is a pile of sand. *(false!)*

  At which number of grains of sand is the inference not truth-preserving?

Brief Introduction to Fuzzy Theory

- Obviously: There is no precise number of grains of sand,
  at which the inference to the next smaller number is false.
- Problem: **Terms of natural language are vague.**
  (e.g.. "pile of sand", "bald", "warm", "fast", "light", "high pressure")
- Note: **Vague terms are *inexact*, but nevertheless *not useless*.**
  - Even for vague terms there are situations or objects,
    to which they are *certainly applicable*, and others,
    to which they are *certainly not applicable*.
  - In between lies a so-called **penumbra** (lat. for *half shadow*) of situations, in which it is unclear whether the terms are applicable, or in which they are applicable only with certain restrictions ("small pile of sand").
  - Fuzzy theory tries to model this penumbra in a mathematical fashion
    ("soft transition" between *applicable* and *not applicable*).

Fuzzy Logic

- **Fuzzy Logic** is an extension of classical logic
  by values between *true* and *false*.
- **Truth values are any values from the real interval $[0, 1]$**,
  where $0 \mathrel{\widehat{=}}$ *false* and $1 \mathrel{\widehat{=}}$ *true*.
- Therefore necessary: **extension of the logical operators**
  - Negation classical: $\neg a$, fuzzy: $\sim a$ Fuzzy-Negation
  - Conjunction classical: $a \wedge b$, fuzzy: $\top(a, b)$ t-Norm
  - Disjunction classical: $a \vee b$, fuzzy: $\bot(a, b)$ t-Konorm
- **Basic principles of this extension:**
  - For the extreme values 0 and 1 the operations should behave exactly
    like their classical counterparts (border or corner conditions).
  - For the intermediate values the behavior should be monotone.
  - As far as possible, the laws of classical logic should be preserved.

Fuzzy Negations

A **fuzzy negation** is a function $\sim\colon [0,1] \to [0,1]$,
that satisfies the following conditions:

- $\sim 0 = 1 \quad$ and $\quad \sim 1 = 0 \qquad\qquad$ (boundary conditions)
- $\forall a, b \in [0,1]: \quad a \leq b \;\Rightarrow\; \sim a \geq \sim b \qquad$ (monotony)

If in the second condition the relations $<$ and $>$ hold instead of merely $\leq$ and $\geq$,
the fuzzy negation is called a *strict* negation.

Additional conditions that are sometimes required are:

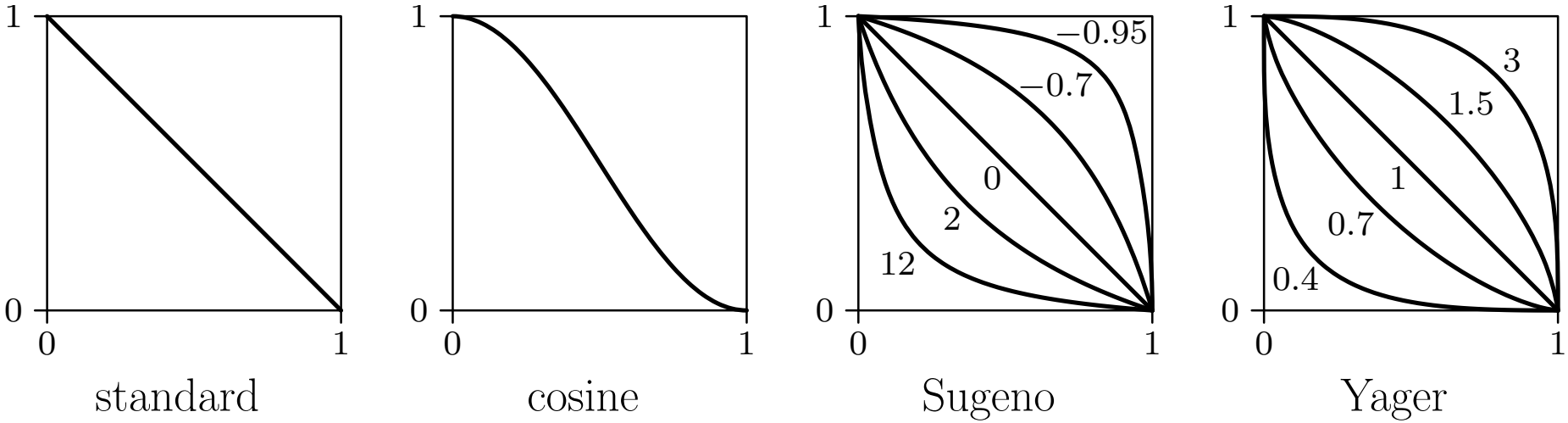
- $\sim$ is a continuous function.
- $\sim$ is *involutive*, that is, $\quad \forall a \in [0,1]: \quad \sim\sim a = a.$

Involutivity corresponds to the classical *law of identity* $\neg\neg a = a$.

The above conditions do not uniquely determine a fuzzy negation.

Fuzzy Negations

$$
\begin{array}{lllll}
\text{standard negation:} & \sim a & = & 1-a \\
\text{threshold negation:} & \sim(a;\theta) & = & \begin{cases} 1 & \text{if } x \leq \theta, \\ 0 & \text{otherwise.} \end{cases} \\
\text{cosine negation:} & \sim a & = & \frac{1}{2}(1+\cos \pi a) \\
\text{Sugeno negation:} & \sim(a;\lambda) & = & \dfrac{1-a}{1+\lambda a} \\
\text{Yager negation:} & \sim(a;\lambda) & = & (1-a^{\lambda})^{\frac{1}{\lambda}}
\end{array}
$$



t-Norms / Fuzzy Conjunctions

A **t-norm** or **fuzzy conjunction** is a function $\top : [0,1]^2 \to [0,1]$,
that satisfies the following conditions:

- $\forall a \in [0,1]: \quad \top(a,1) = a$ (boundary condition)
- $\forall a,b,c \in [0,1]: \quad b \leq c \Rightarrow \top(a,b) \leq \top(a,c)$ (monotony)
- $\forall a,b \in [0,1]: \quad \top(a,b) = \top(b,a)$ (commutativity)
- $\forall a,b,c \in [0,1]: \quad \top(a,\top(b,c)) = \top(\top(a,b),c)$ (associativity)

Additional conditions that are sometimes required are:

- $\top$ is a continuous function (continuity)
- $\forall a \in [0,1]: \quad \top(a,a) < a$ (sub-idempotency)
- $\forall a,b,c,d \in [0,1]: \quad a < b \wedge c < d \Rightarrow \top(a,b) < \top(c,d)$ (strict monotony)

The first two of these conditions (in addition to the top four)
define the sub-class of so-called *Archimedic t-norms*.

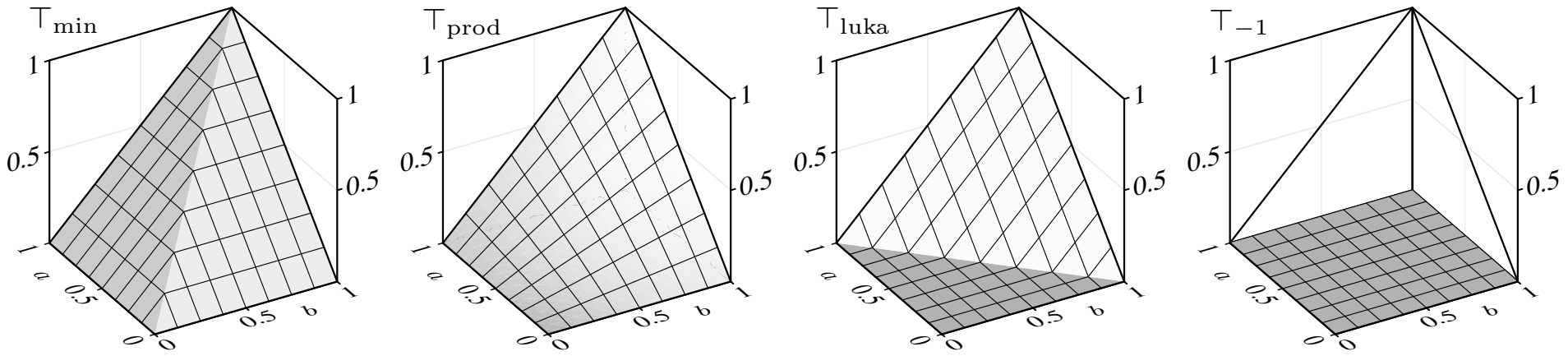
t-Norms / Fuzzy Conjunctions

standard conjunction: $\top_{\min}(a,b) = \min\{a,b\}$

algebraic product: $\top_{\text{prod}}(a,b) = a \cdot b$

Łukasiewicz: $\top_{\text{luka}}(a,b) = \max\{0, a+b-1\}$

drastic product:
$$\top_{-1}(a,b) = \begin{cases} a & \text{if } b = 1, \\ b & \text{if } a = 1, \\ 0 & \text{otherwise.} \end{cases}$$



t-Conorms / Fuzzy Disjunctions

A **t-conorm** or **fuzzy disjunction** is a function $\perp : [0,1]^2 \to [0,1]$,
that satisfies the following conditions:

- $\forall a \in [0,1]: \quad \perp(a,0) = a$ (boundary condition)
- $\forall a,b,c \in [0,1]: \quad b \leq c \Rightarrow \perp(a,b) \leq \perp(a,c)$ (monotony)
- $\forall a,b \in [0,1]: \quad \perp(a,b) = \perp(b,a)$ (commutativity)
- $\forall a,b,c \in [0,1]: \quad \perp(a,\perp(b,c)) = \perp(\perp(a,b),c)$ (assoziativity)

Additional conditions that are sometimes required are:

- $\perp$ is a continuous function (continuity)
- $\forall a \in [0,1]: \quad \perp(a,a) > a$ (super-idempotency)
- $\forall a,b,c,d \in [0,1]: \quad a < b \wedge c < d \Rightarrow \perp(a,b) < \perp(c,d)$ (strict monotony)

The first two of these conditions (in addition to the top four)
define the sub-class of so-called *Archimedic t-conorms*.

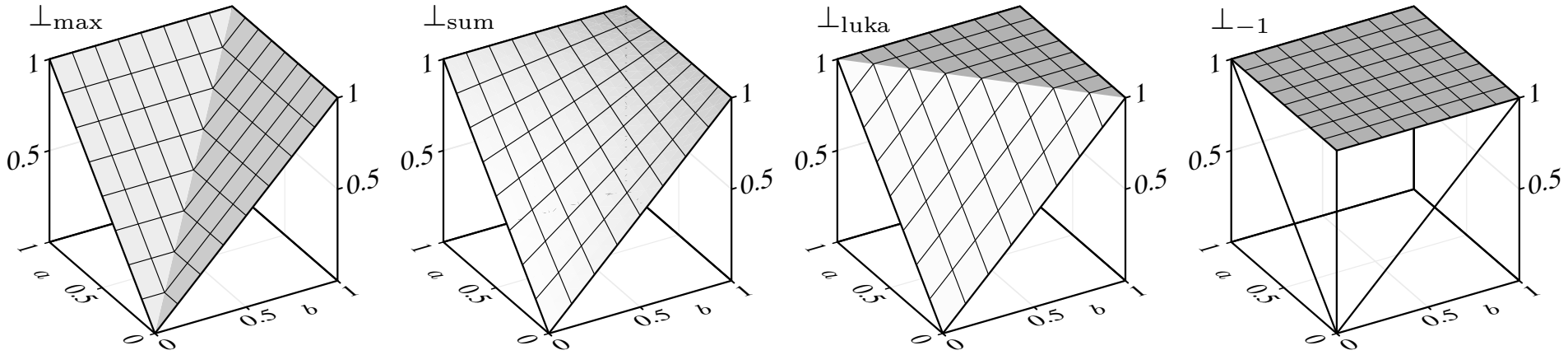
t-Conorms / Fuzzy Disjunctions

standard disjunction: $\perp_{\max}(a,b) = \max\{a,b\}$

algebraische sum: $\perp_{\text{sum}}(a,b) = a + b - a \cdot b$

Łukasiewicz: $\perp_{\text{luka}}(a,b) = \min\{1, a+b\}$

drastic sum:
$$\perp_{-1}(a,b) = \begin{cases} a & \text{if } b = 0, \\ b & \text{if } a = 0, \\ 1 & \text{otherwise.} \end{cases}$$



Interplay of the Fuzzy Operators

- It is $\forall a, b \in [0,1]: \quad \top_{-1}(a,b) \leq \top_{\text{luka}}(a,b) \leq \top_{\text{prod}}(a,b) \leq \top_{\min}(a,b)$.
  All other possible t-norms lie between $\top_{-1}$ and $\top_{\min}$ as well.
- It is $\forall a, b \in [0,1]: \quad \bot_{\max}(a,b) \leq \bot_{\text{sum}}(a,b) \leq \bot_{\text{luka}}(a,b) \leq \bot_{-1}(a,b)$.
  All other possible t-conorms lie between $\bot_{\max}$ and $\bot_{-1}$ as well.
- Note: Generally *neither* $\top(a, \sim a) = 0$ *nor* $\bot(a, \sim a) = 1$ holds.
- A set of operators $(\sim, \top, \bot)$ consisting of a fuzzy negation $\sim$, a t-norm $\top$, and a t-conorm $\bot$ is called a **dual triplet** if with these operators DeMorgan's laws hold, that is, if

$$\forall a, b \in [0,1]: \quad \sim\top(a,b) = \bot(\sim a, \sim b)$$

$$\forall a, b \in [0,1]: \quad \sim\bot(a,b) = \top(\sim a, \sim b)$$

- The most frequently used set of operators is the dual triplet $(\sim, \top_{\min}, \bot_{\max})$ with the standard negation $\sim a \equiv 1 - a$.

Fuzzy Set Theory

- Classical set theory is based on the notion "*is element of*" ($\in$).
  Alternatively the membership in a set can be described
  with the help of an *indicator function*:

  Let X be a set (base set). Then

  $$I_M : X \to \{0, 1\}, \qquad I_M(x) = \begin{cases} 1 & \text{if } x \in X, \\ 0 & \text{otherwise,} \end{cases}$$

  is called **indicator function** of the set M w.r.t. the base set X.

- In fuzzy set theory the indicator function of classical set theory
  is replaced by a *membership function*:

  Let X be a (classical/crisp) set. Then

  $$\mu_M : X \to [0, 1], \qquad \mu_M(x) \mathrel{\widehat{=}} \text{degree of membership of } x \text{ to } M,$$

  is called **membership function** of the **fuzzy set** M w.r.t. the *base set* X.

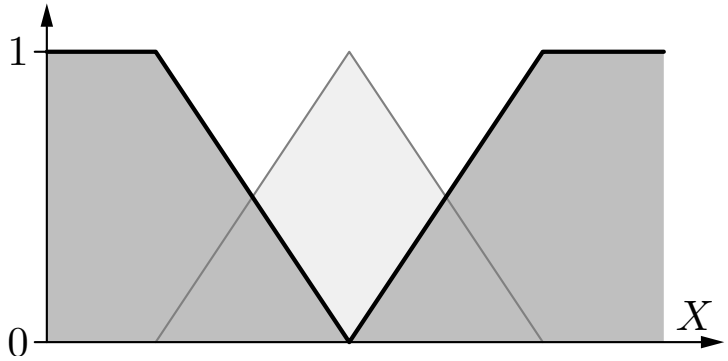
  Usually the fuzzy set is identified with its membership function.

Fuzzy Set Theory: Operations

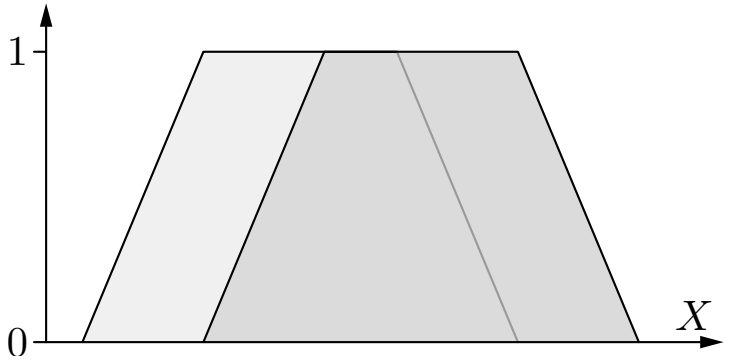
- In analogy to the transition from classical logic to fuzzy logic the transition from classical set theory to fuzzy set theory requires an **extension of the operators**.
- **Basic principle of the extension**:
  Draw on the logical definition of the operators.
  $\Rightarrow$ element-wise application of the logical operators.
- Let A and B be (fuzzy) sets w.r.t. the base set X.

complement	classical	$\overline{A} = \{x \in X \mid x \notin A\}$
	fuzzy	$\forall x \in X: \quad \mu_{\overline{A}}(x) = \sim\mu_A(x)$
intersection	classical	$A \cap B = \{x \in X \mid x \in A \wedge x \in B\}$
	fuzzy	$\forall x \in X: \quad \mu_{A\cap B}(x) = \top(\mu_A(x), \mu_B(x))$
union	classical	$A \cup B = \{x \in X \mid x \in A \vee x \in B\}$
	fuzzy	$\forall x \in X: \quad \mu_{A\cup B}(x) = \bot(\mu_A(x), \mu_B(x))$

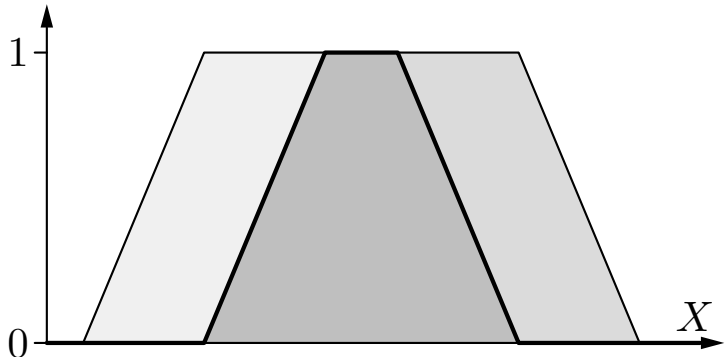
Fuzzy Set Operations: Examples



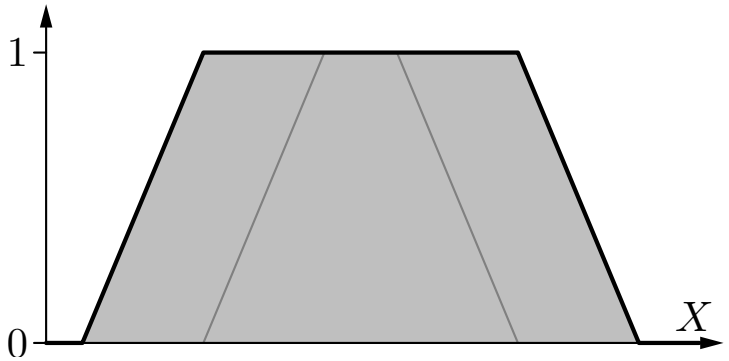
fuzzy complement



two fuzzy sets



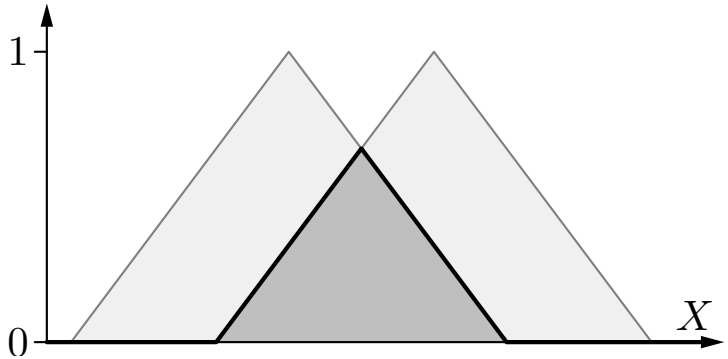
fuzzy intersection



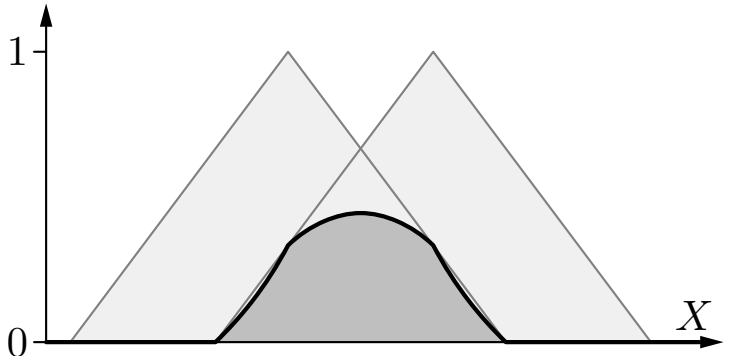
fuzzy union

- The fuzzy intersection shown on the left and the fuzzy union shown on the right are independent of the chosen t-norm or t-conorm.

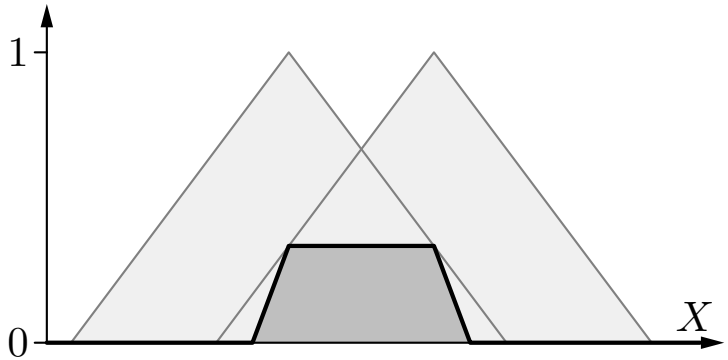
Fuzzy Intersection: Examples



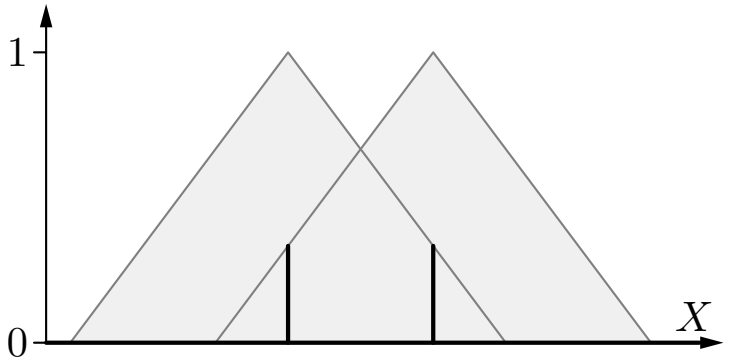
t-norm $\top_{\text{min}}$



t-norm $\top_{\text{prod}}$



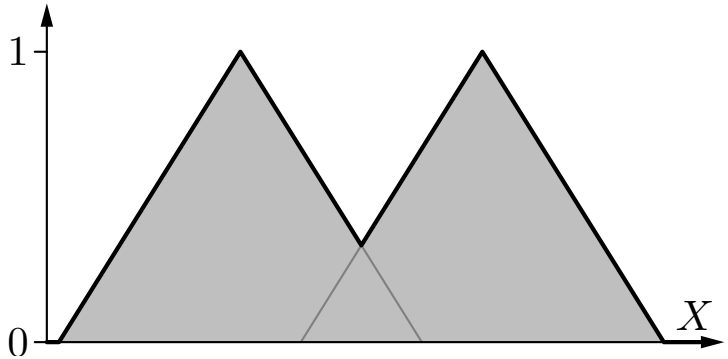
t-norm $\top_{\text{luka}}$



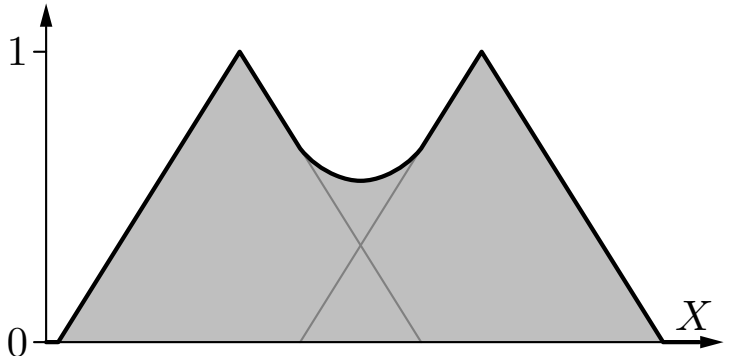
t-norm $\top_{-1}$

- Note that all fuzzy intersections lie between the one shown at the top right and the one shown at the right bottom.

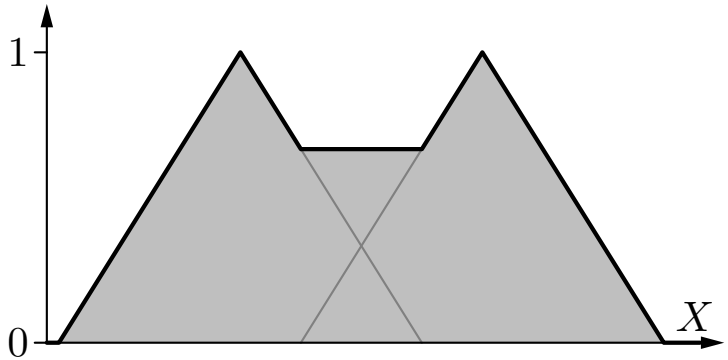
Fuzzy Union: Examples



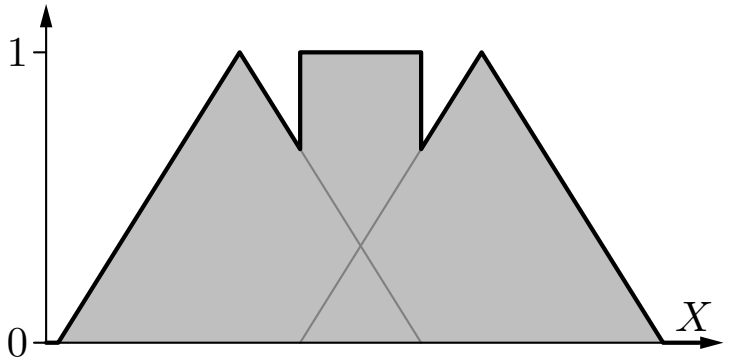
t-norm $\top_{\min}$



t-norm $\top_{\text{prod}}$



t-norm $\top_{\text{luka}}$



t-norm $\top_{-1}$

- Note that all fuzzy unions lie between the one shown at the top right and the one shown at the right bottom.

Fuzzy Partitions and Linguistic Variables

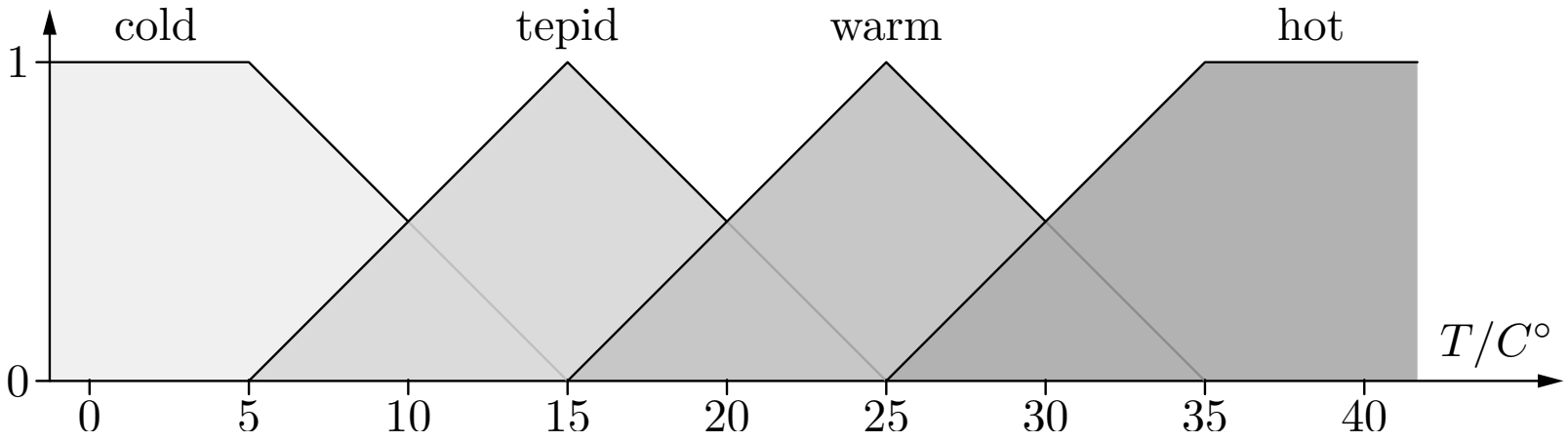
- In order to describe a domain by linguistic terms,
  it is fuzzy-partitioned with the help of fuzzy sets.

  To each fuzzy set of the partition a linguistic term is assigned.

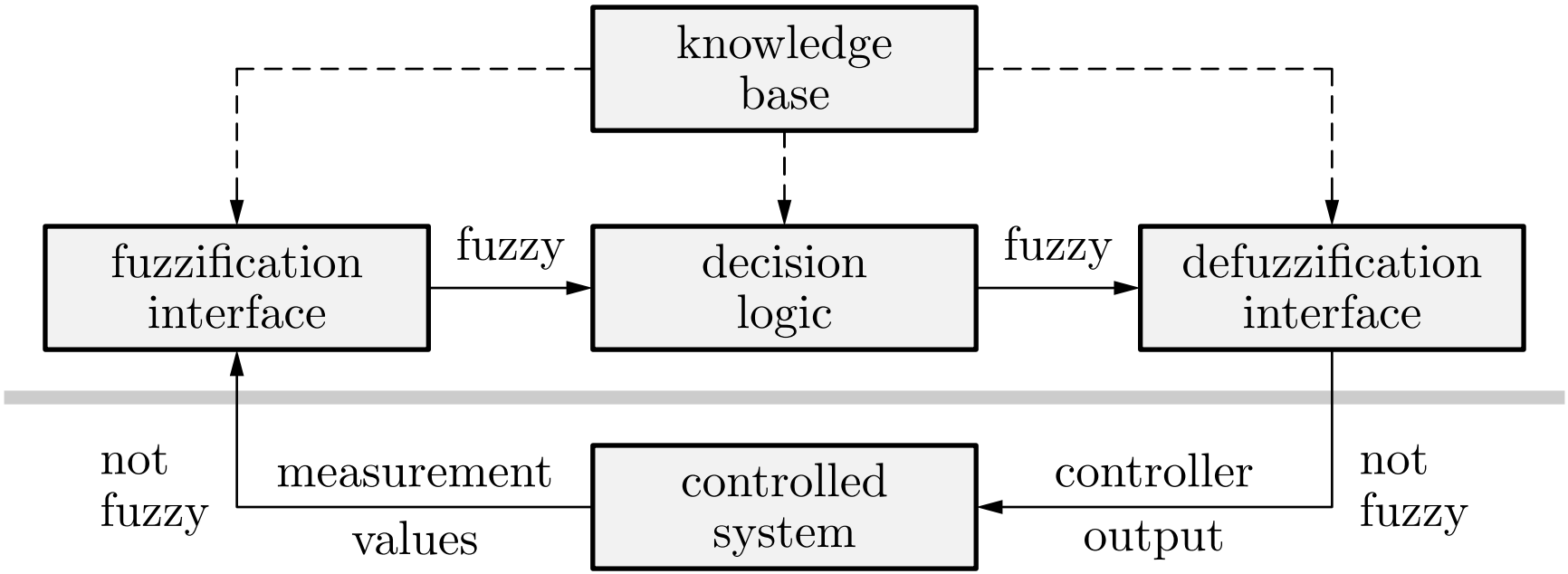
- Common condition: At each point the membership values of the fuzzy sets must sum to 1 *(partition of unity)*.

Example: **fuzzy partition for temperatures**

We define a linguistic variable with the values *cold*, *tepid*, *warm* and *hot*.

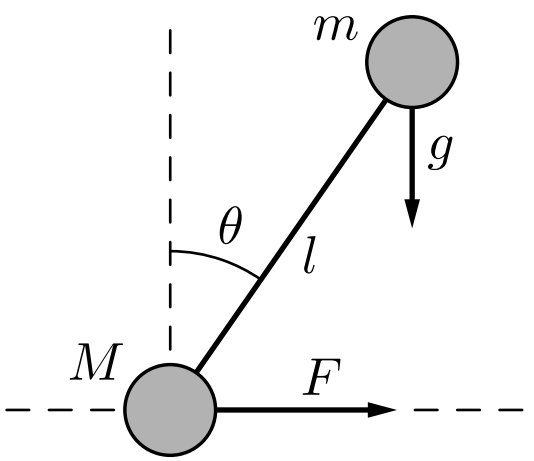


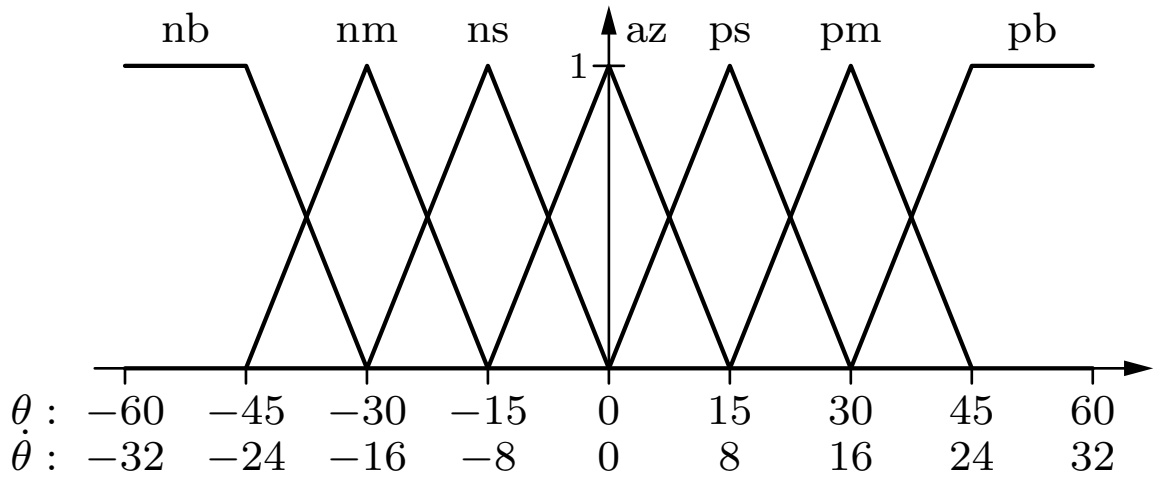
Architecture of a Fuzzy Controller



- The knowledge base contains the fuzzy rules of the controller as well as the fuzzy partitions of the domains of the variables.
- A fuzzy rule reads: **if** X_1 **is** $A_{i_1}^{(1)}$ **and** ... **and** X_n **is** $A_{i_n}^{(n)}$ **then** Y **is** B.
  $X_1, \ldots, X_n$ are the measurement values and Y is the controller output.
  $A_{i_k}^{(k)}$ and B are linguistic terms to which fuzzy sets are assigned.

Example Fuzzy Controller: Inverted Pendulum



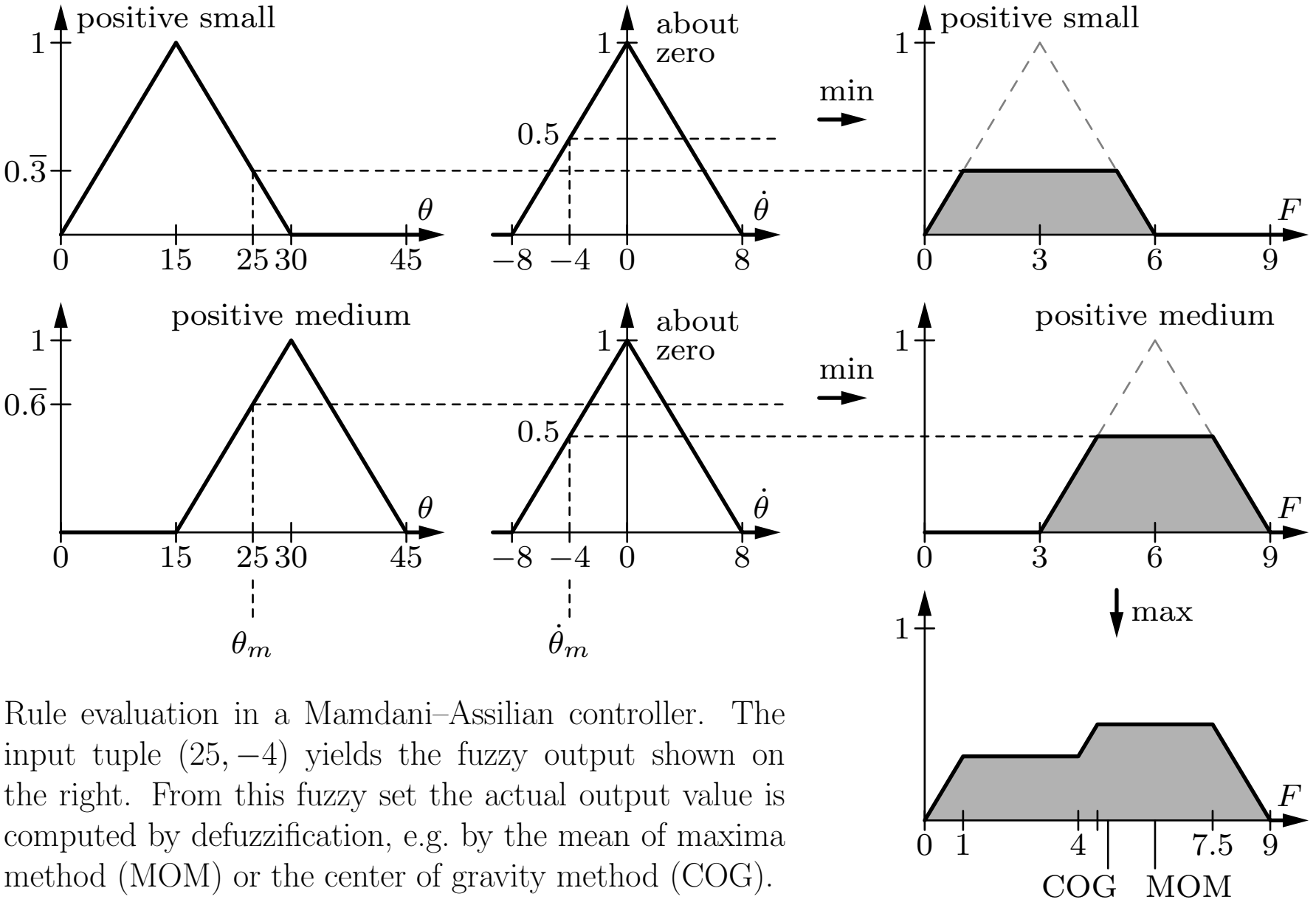


abbreviations

pb – positive big
pm – positive medium
ps – positive small
az – approximately zero
ns – negative small
nm – negative medium
nb – negative big

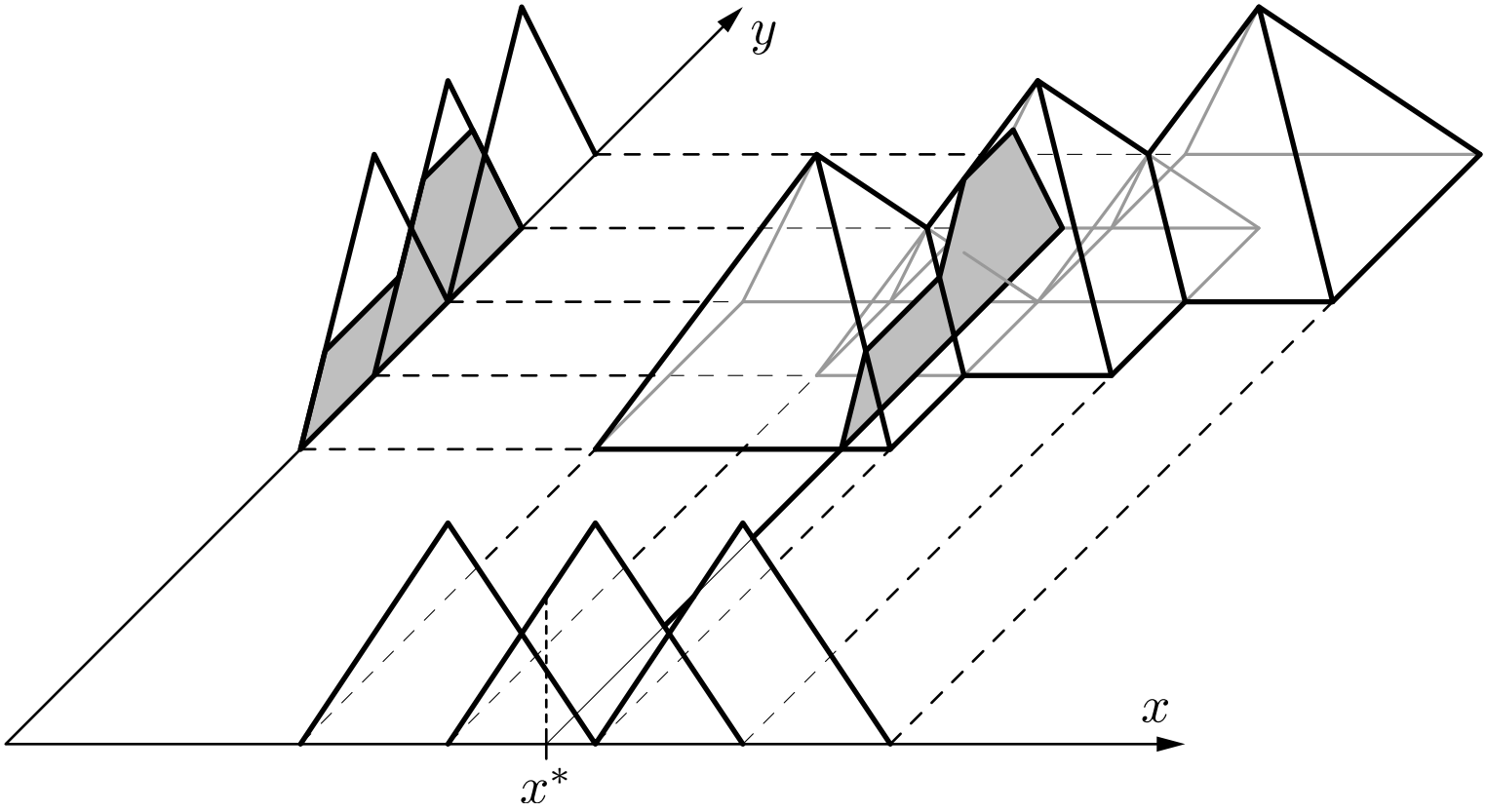
$\dot{\theta}\backslash\theta$	nb	nm	ns	az	ps	pm	pb
pb			**ps**	**pb**			
pm				**pm**			
ps	**nm**		**az**	**ps**			
az	**nb**	**nm**	**ns**	**az**	**ps**	**pm**	**pb**
ns				**ns**	**az**		**pm**
nm				**nm**			
nb				**nb**	**ns**		

Mamdani–Assilian Controller



Rule evaluation in a Mamdani–Assilian controller. The input tuple $(25, -4)$ yields the fuzzy output shown on the right. From this fuzzy set the actual output value is computed by defuzzification, e.g. by the mean of maxima method (MOM) or the center of gravity method (COG).

Mamdani–Assilian Controller



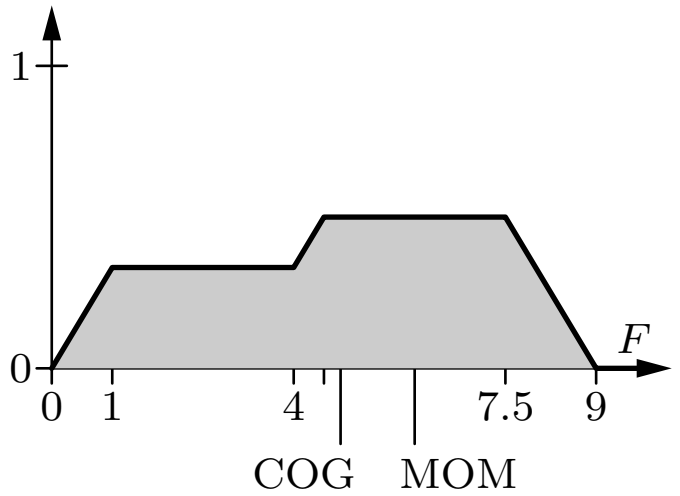
A fuzzy control system with one measurement and one control variable and three fuzzy rules. Each pyramid is specified by one fuzzy rule. The input value x^* leads to the fuzzy output shown in gray.

Defuzzification

The evaluation of the fuzzy rules yields an **output fuzzy set**.

The output fuzzy set has to be turned into a **crisp controller output**.

This process is called **defuzzification**.



The most important defuzzification methods are:

- **Center of Gravity** (COG)
  The center of gravity of the area under the output fuzzy set.
- **Center of Area** (COA)
  The point that divides the area under the output fuzzy set into equally large parts.
- **Mean of Maxima** (MOM)
  The arithmetic mean of the points with maximal degree of membership.

Takagi–Sugeno–Kang Controller (TSK Controller)

- The rules of a Takagi–Sugeno–Kang controller have the same kind of antecedent as the rules of a Mamdani–Assilian controller, but a different kind of consequent:

  R_i: **if** x_1 is $\mu_i^{(1)}$ **and** ... **and** x_n is $\mu_i^{(n)}$, **then** $y = f_i(x_1, \ldots, x_n)$.

  The consequent of a Takagi–Sugeno–Kang rule specifies a function of the inputs that is to be computed if the antecedent is satisfied.

- Let $\tilde{a}_i$ be the activation of the antecedent of the rule R_i, that is,

$$\tilde{a}_i(x_1, \ldots, x_n) = \top\left(\top\left(\ldots\top\left(\mu_i^{(1)}(x_1), \mu_i^{(2)}(x_2)\right), \ldots\right), \mu_i^{(n)}(x_n)\right).$$

- Then the output of a Takagi–Sugeno–Kang controller is computed as

$$y(x_1, \ldots, x_n) = \frac{\sum_{i=1}^{r} \tilde{a}_i \cdot f_i(x_1, \ldots, x_n)}{\sum_{i=1}^{r} \tilde{a}_i},$$

  that is, the controller output is a weighted average of the outputs of the individual rules, where the activations of the antecedents provide the weights.

Neuro-Fuzzy Systems

- **Disadvantages of Neural Networks:**
  - Training results are difficult to interpret (black box).

    The result of training an artificial neural network are matrices or vectors of real-valued numbers. Even though the computations are clearly defined, humans usually have trouble understanding what is going on.
  - It is difficult to specify and incorporate prior knowledge.

    Prior knowledge would have to be specified as the mentioned matrices or vectors of real-valued numbers, which are difficult to understand for humans.
- **Possible Remedy:**
  - Use a hybrid system, in which an artificial neural network is coupled with a rule-based system.

    The rule-based system can be interpreted and set up by a human.
  - One such approach are **neuro-fuzzy systems**.

Neuro-Fuzzy Systems

Neuro-fuzzy systems are commonly divided into cooperative and hybrid systems.

- **Cooperative Models:**
  - A neural network and a fuzzy controller work independently.
  - The neural network generates (offline) or optimizes (online) certain parameters.
- **Hybrid Models:**
  - Combine the structure of a neural network and a fuzzy controller.
  - A hybrid neuro-fuzzy controller can be interpreted as a neural network and can be implemented with the help of a neural network.
  - Advantages: integrated structure;
    no communication between two different models is needed;
    in principle, both offline and online training are possible,

Hybrid models are more accepted and more popular than cooperative models.

Neuro-Fuzzy Systems: Hybrid Methods

- Hybrid methods map fuzzy sets and fuzzy rules to a neural network structure.
- The activation $\tilde{a}_i$ of the antecedent of a Mamdani–Assilian rule

  R_i: **if** x_1 is $\mu_i^{(1)}$ **and** ... **and** x_n is $\mu_i^{(n)}$, **then** y is ν_i.

  or of a Takagi–Sugeno–Kang rule

  R_i: **if** x_1 is $\mu_i^{(1)}$ **and** ... **and** x_n is $\mu_i^{(n)}$, **then** $y = f_i(x_1, \ldots, x_n)$.

  is computed with a t-norm $\top$ (most commonly $\top_{\min}$).
- For given input values $x_1, \ldots, x_n$ the network structure has to compute:

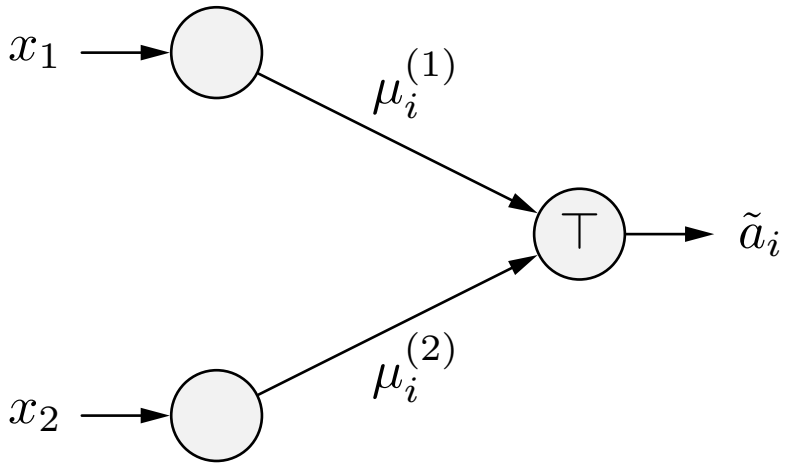
$$\tilde{a}_i(x_1, \ldots, x_n) = \top\left(\top\left(\ldots\top\left(\mu_i^{(1)}(x_1), \mu_i^{(2)}(x_2)\right), \ldots\right), \mu_i^{(n)}(x_n)\right).$$

  (Since t-norms are associative, it does not matter in which order the membership degrees are combined by successive pairwise applications of the t-norm.)

Computing the activation $\tilde{a}_i$ of the antecedent of a fuzzy rule:

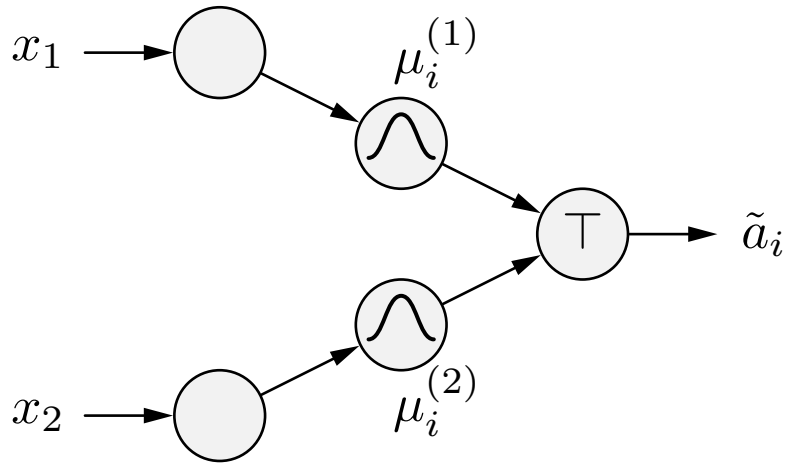
The fuzzy sets appearing in the antecedents of fuzzy rules can be modeled

as connection weights:



The neurons in the first hidden layer represent the rule antecedents and the connections from the input units represent the fuzzy sets of these antecedents.

as activation functions:



The neurons in the first hidden layer represent the fuzzy sets and the neurons in the second hidden layer represent the rule antecedents ($\top$ is a t-norm).

From Fuzzy Rule Base to Network Structure

If the fuzzy sets are represented as activation functions (right on previous slide), a fuzzy rule base is turned into a network structure as follows:

1. For every input variable x_i: create a neuron in the input layer.
2. For every output variable y_i: create a neuron in the output layer.
3. For every fuzzy set $\mu_i^{(j)}$: create a neuron in the first hidden layer and connect it to the input neuron corresponding to x_i.
4. For every fuzzy rule R_i: create a (rule) neuron in the second hidden layer and specify a t-norm for computing the rule (antecedent) activation.
5. Connect each rule neuron to the neurons that represent the fuzzy sets of the antecedent of its corresponding fuzzy rule R_i.
6. (This step depends on whether the controller is of the Mamdani–Assilian or of the Takagi–Sugeno–Kang type; see next slide.)

From Fuzzy Rule Base to Network Structure

Mamdani–Assilian controller:

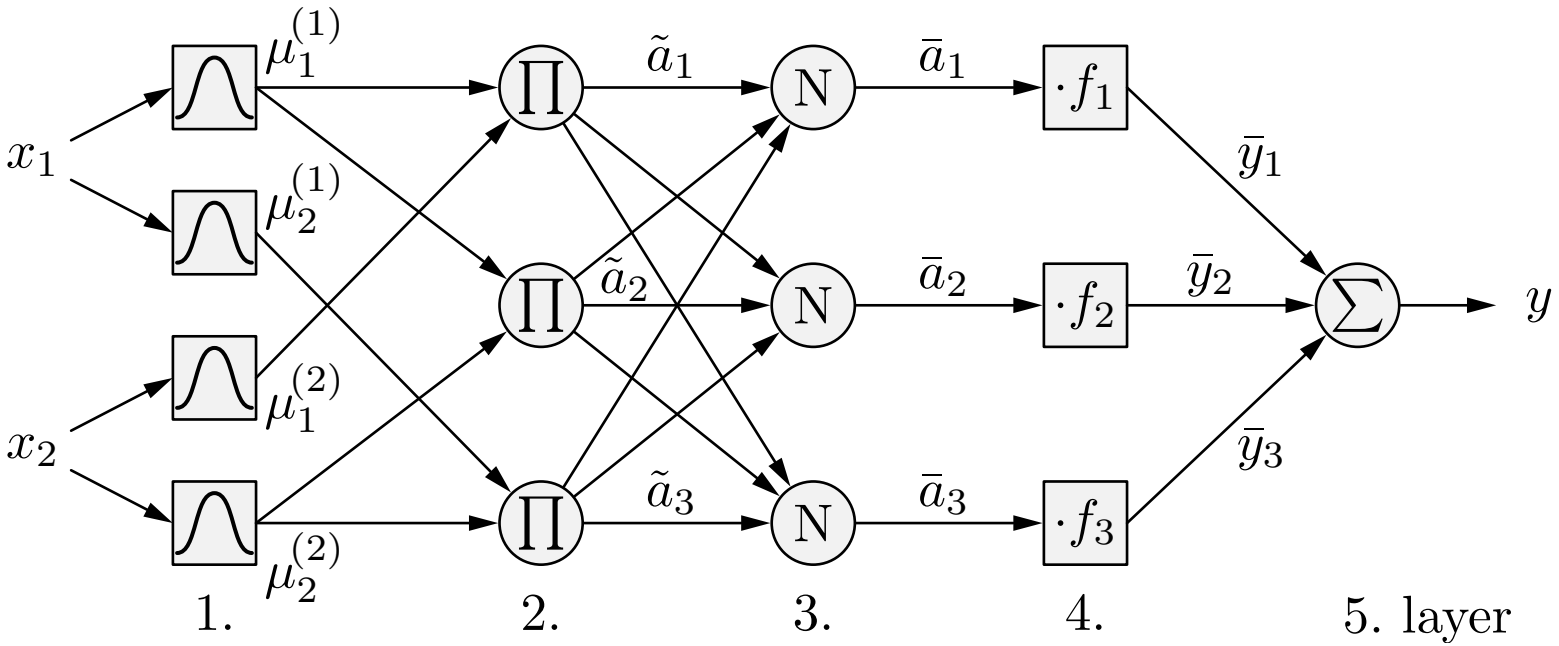
6. Connect each rule neuron to the output neuron corresponding to the consequent domain of its fuzzy rule. As connection weight choose the consequent fuzzy set of the fuzzy rule. Furthermore, a t-conorm for combining the outputs of the individual rules and a defuzzification method have to be integrated adequately into the output neurons (e.g. as network input and activation functions).

Takagi–Sugeno–Kang controller:

6. For each rule neuron, create a sibling neuron that computes the output function of the corresponding fuzzy rule and connect all input neurons to it (arguments of the consequent function). All rule neurons that refer to the same output domain as well as their sibling neurons are connected to the corresponding output neuron (in order to compute the weighted average of the rule outputs).

The resulting network structure can now be trained with procedures that are analogous to those of standard neural networks (e.g. error backpropagation).

Adaptive Network-based Fuzzy Inference Systems (ANFIS)

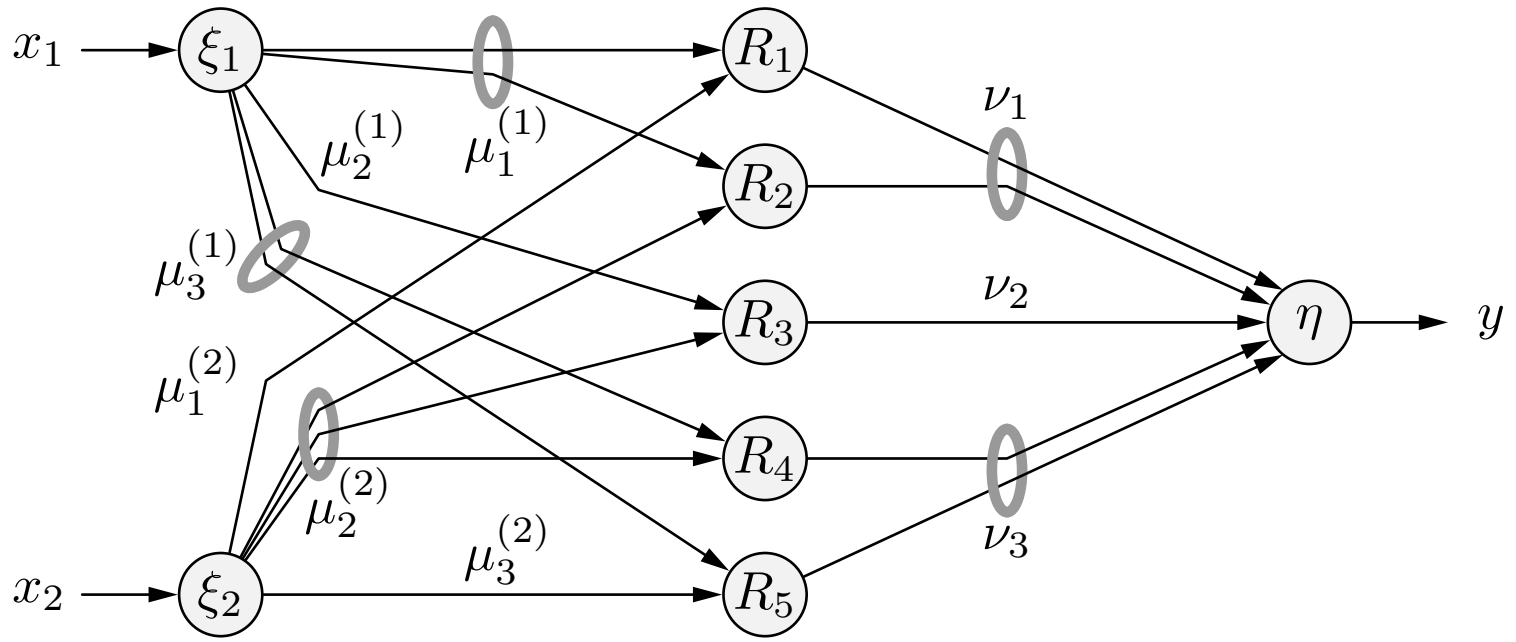


(Connections from inputs to output function neurons for f_1, f_2, f_3 not shown.)

This ANFIS network represents the fuzzy rule base (Takagi–Sugeno–Kang rules):

$$
\begin{aligned}
R_1\text{:}\quad & \textbf{if} \quad x_1 \text{ is } \mu_1^{(1)} \quad \textbf{and} \quad x_2 \text{ is } \mu_1^{(2)}, \quad \textbf{then} \quad y = f_1(x_1, x_2) \\
R_2\text{:}\quad & \textbf{if} \quad x_1 \text{ is } \mu_1^{(1)} \quad \textbf{and} \quad x_2 \text{ is } \mu_2^{(2)}, \quad \textbf{then} \quad y = f_2(x_1, x_2) \\
R_3\text{:}\quad & \textbf{if} \quad x_1 \text{ is } \mu_2^{(1)} \quad \textbf{and} \quad x_2 \text{ is } \mu_2^{(2)}, \quad \textbf{then} \quad y = f_3(x_1, x_2)
\end{aligned}
$$

Neuro-Fuzzy Control (NEFCON)



This NEFCON network represents the fuzzy rule base (Mamdani–Assilian rules):

R_1: **if** x_1 is $\mu_1^{(1)}$ **and** x_2 is $\mu_1^{(2)}$, **then** y is ν_1

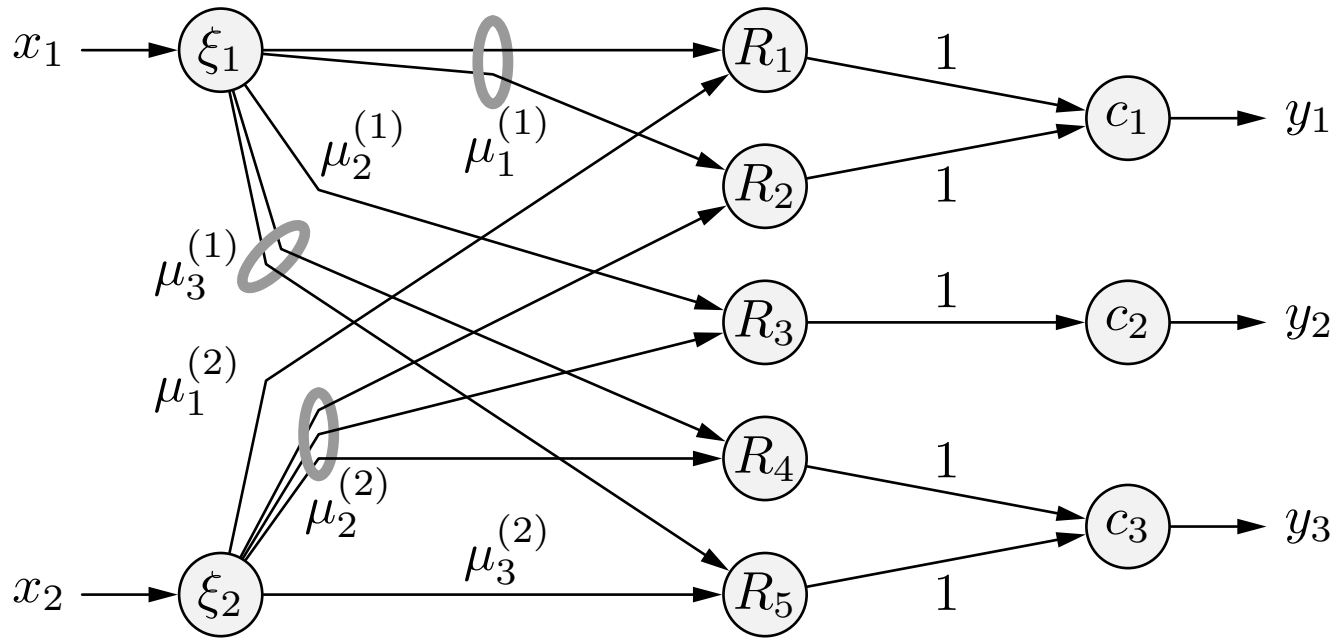
R_2: **if** x_1 is $\mu_1^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** y is ν_1

R_3: **if** x_1 is $\mu_2^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** y is ν_2

R_4: **if** x_1 is $\mu_3^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** y is ν_3

R_5: **if** x_1 is $\mu_3^{(1)}$ **and** x_2 is $\mu_3^{(2)}$, **then** y is ν_3

Neuro-Fuzzy Classification (NEFCLASS)



This NEFCLASS network represents fuzzy rules that predict classes:

R_1: **if** x_1 is $\mu_1^{(1)}$ **and** x_2 is $\mu_1^{(2)}$, **then** class c_1

R_2: **if** x_1 is $\mu_1^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** class c_1

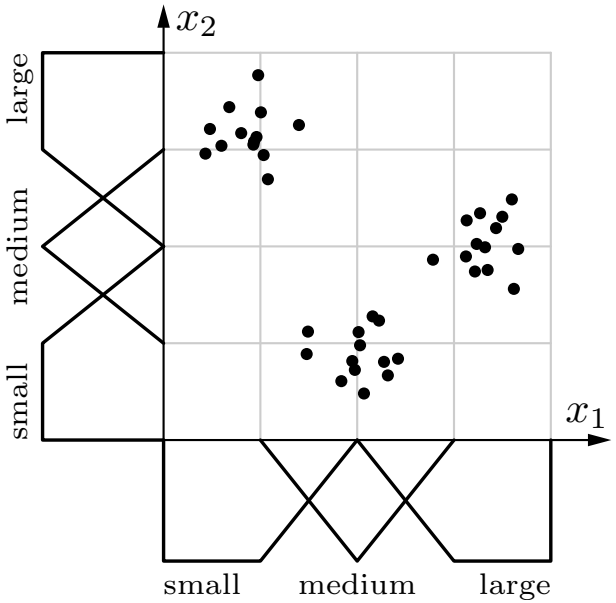
R_3: **if** x_1 is $\mu_2^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** class c_2

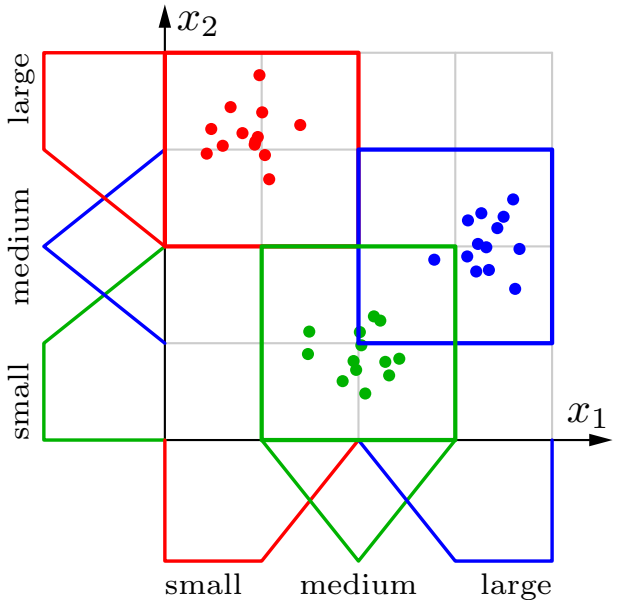
R_4: **if** x_1 is $\mu_3^{(1)}$ **and** x_2 is $\mu_2^{(2)}$, **then** class c_3

R_5: **if** x_1 is $\mu_3^{(1)}$ **and** x_2 is $\mu_3^{(2)}$, **then** class c_3

NEFCLASS: Initializing the Fuzzy Partitions

- NEFCLASS is based on modified Wang–Mendel procedure. [Nauck 1997]
- NEFCLASS first fuzzy partitions the domain of each variable,
  usually with a given number of equally sized triangular fuzzy sets;
  the boundary fuzzy sets are "shouldered" (membership 1 to the boundary).
- Based on the initial fuzzy partitions, the initial rule base is selected.





NEFCLASS: Initializing the Rule Base

$\mathcal{A} := \emptyset$; (∗ initialize the antecedent set ∗)
for each training pattern p **do begin** (∗ traverse the training patterns ∗)
  find rule antecedent A such that $A(p)$ is maximal;
  if $A \notin \mathcal{A}$ **then** (∗ if this is a new antecedent, ∗)
    $\mathcal{A} := \mathcal{A} \cup \{A\}$; (∗ add it to the antecedent set, ∗)
end (∗ that is, collect needed antecedents ∗)

$\mathcal{R} := \emptyset$; (∗ initialize the rule base ∗)
for each antecedent $A \in \mathcal{A}$ **do begin** (∗ traverse the antecedents ∗)
  find best consequent C for antecedent A; (∗ e.g. most frequent class in ∗)
  create rule base candidate $R = (A, C)$; (∗ training patterns assigned to A ∗)
  determine performance of R;
  $\mathcal{R} := \mathcal{R} \cup \{R\}$; (∗ collect the created rules ∗)
end

return rule base $\mathcal{R}$; (∗ return the created rule base ∗)

Fuzzy rule bases may also be created from prior knowledge or using
fuzzy cluster analysis, fuzzy decision trees, evolutionary algorithms etc.

NEFCLASS: Selecting the Rule Base

- In order to reduce/limit the number of initial rules, their performace is evaluated.
- The performance of a rule R_i is computed as

$$\text{Perf}(R_i) = \frac{1}{m} \sum_{j=1}^{m} e_{ij}\, \tilde{a}_i(\vec{x}_j)$$

  where m is the number of training patterns and e_{ij} is an error indicator,

$$e_{ij} = \begin{cases} +1 & \text{if class}(\vec{x}_j) = \text{cons}(R_i), \\ -1 & \text{otherwise.} \end{cases}$$

- Sort the rules in the initial rule base by their performance.
- Choose either the best r rules or the best r/c rules per class,
  where c is the number of classes.
- The number r of rules in the rule base is either provided by a user
  or is automatically determined in such a way that all patterns are covered.

NEFCLASS: Computing the Error Signal

- Let $o_k^{(l)}$ be the desired output and $\text{out}_k^{(l)}$ the actual output of the k-th output neuron (class c_k).
- Fuzzy error (k-th output/class):

$$e_k^{(l)} = 1 - \gamma\left(\varepsilon_k^{(l)}\right),$$

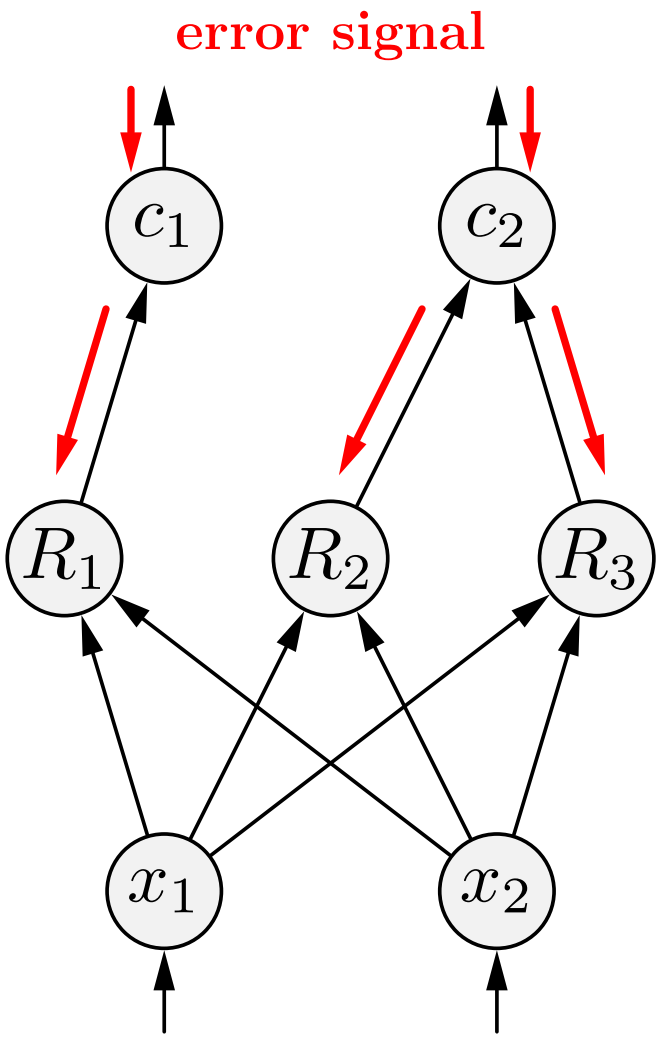
where $\varepsilon_k^{(l)} = o_k^{(l)} - \text{out}_k^{(l)}$ and

$$\gamma(z) = e^{-\beta z^2},$$

where $\beta > 0$ is a sensitivity parameter: larger β means larger error tolerance.

- Error signal (k-th output/class):

$$\delta_k^{(l)} = \text{sgn}\left(\varepsilon_k^{(l)}\right)\, e_k^{(l)}.$$



NEFCLASS: Computing the Error Signal

- Rule error signal (rule R_i for c_k):

$$\delta_{R_i}^{(l)} = \mathrm{out}_{R_i}^{(l)} \left(1 - \mathrm{out}_{R_i}^{(l)}\right) \delta_k^{(l)},$$

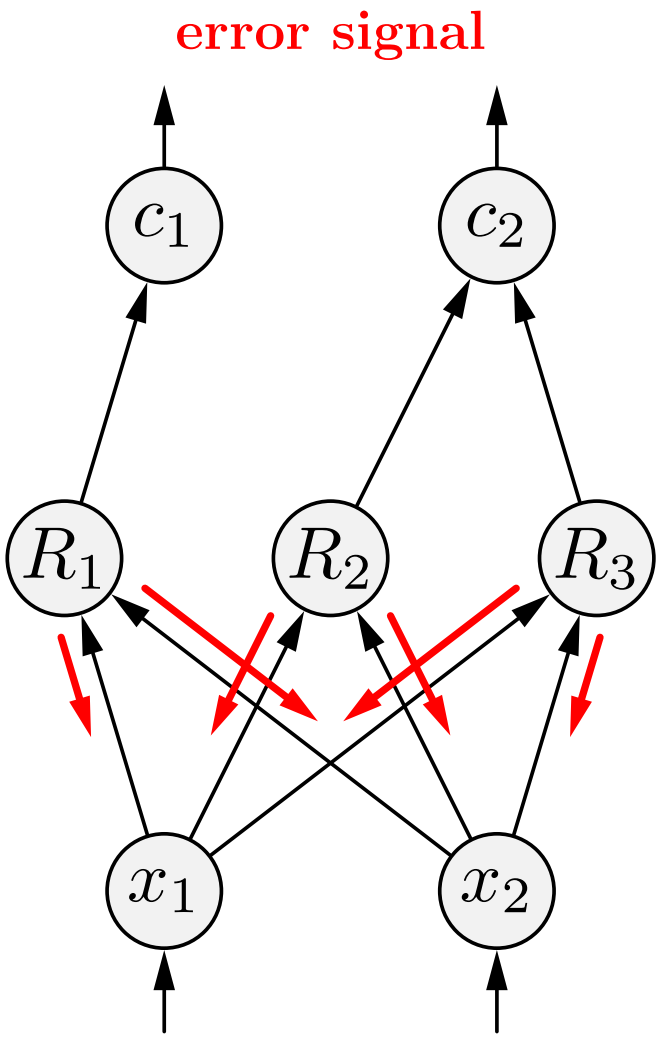
(the factor "out$(1 - \mathrm{out})$" is chosen in analogy to multi-layer perceptrons).

- Find input variable x_j such that

$$\mu_i^{(j)}\left(\imath_j^{(l)}\right) = \tilde{a}_i\left(\vec{\imath}^{(l)}\right) = \min_{\nu=1}^{d} \mu_i^{(\nu)}\left(\imath_\nu^{(l)}\right),$$

where d is the number of inputs (find antecedent term of R_i giving the smallest membership degree, which yields the rule activation).

- Adapt parameters of the fuzzy set $\mu_i^{(j)}$ (see next slide for details).



NEFCLASS: Training the Fuzzy Sets

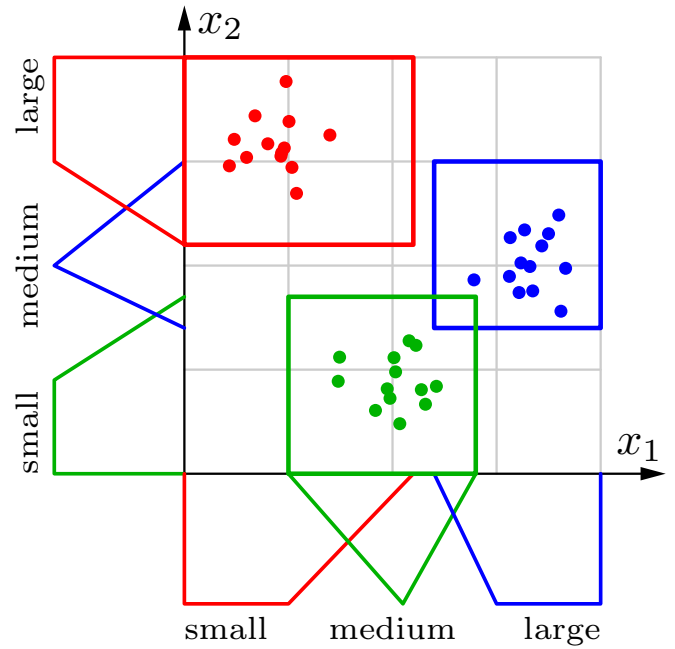
- Triangular fuzzy set as an example:

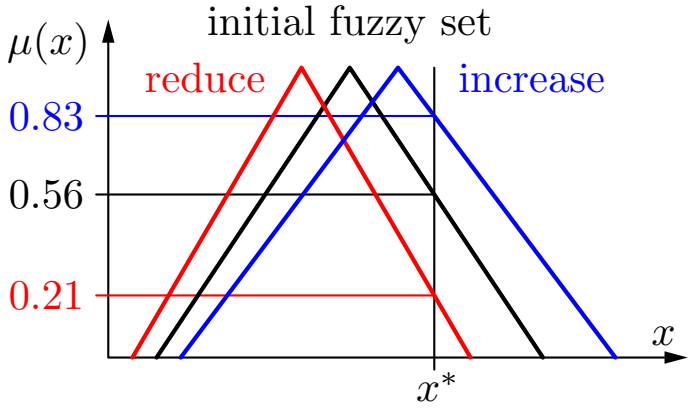
$$\mu_{a,b,c}(x) = \begin{cases} \frac{x-a}{b-a} & \text{if } x \in [a,b), \\ \frac{c-x}{c-b} & \text{if } x \in [b,c], \\ 0 & \text{otherwise.} \end{cases}$$

- Parameter changes (learning rate η):

$$\begin{aligned} \Delta b &= +\eta \cdot \delta_{R_i}^{(l)} \cdot (c-a) \cdot \operatorname{sgn}\left(\imath_j^{(l)} - b\right) \\ \Delta a &= -\eta \cdot \delta_{R_i}^{(l)} \cdot (c-a) + \Delta b \\ \Delta c &= +\eta \cdot \delta_{R_i}^{(l)} \cdot (c-a) + \Delta b \end{aligned}$$

- Heuristics: the fuzzy set to train is moved away from x^* (towards x^*) and its support is reduced (increased) in order to reduce (increase) the degree of membership of x^*.



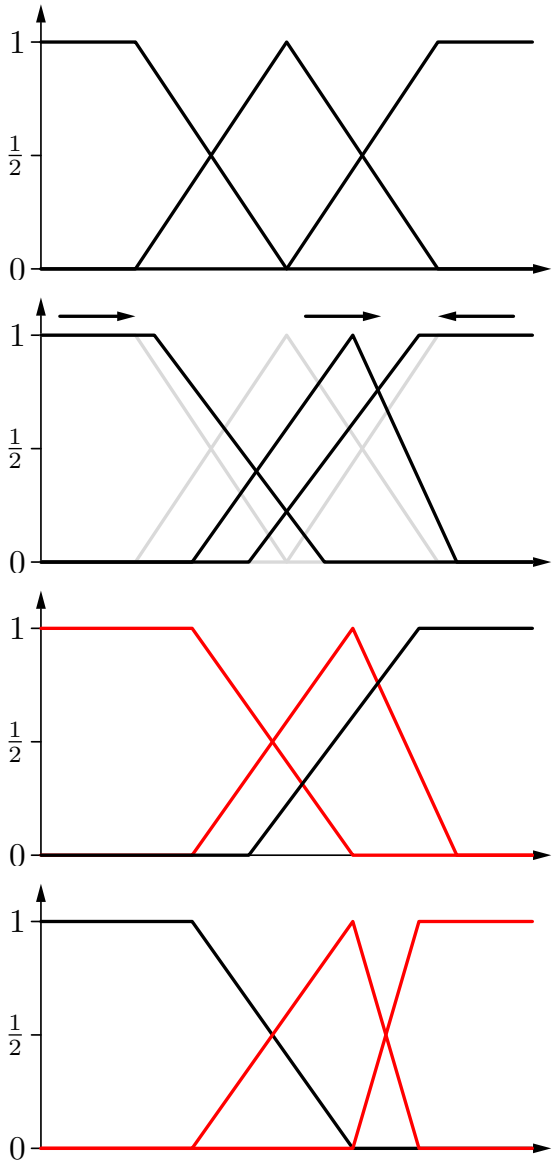


NEFCLASS: Restricted Training of Fuzzy Sets

When fuzzy sets of a fuzzy partition are trained, restrictions apply, so correction procedures are needed to

- ensure valid parameter values
- ensure non-empty intersections of neighboring fuzzy sets
- preserve relative positions
- preserve symmetry
- ensure partition of unity (membership degrees sum to 1 everywhere)

On the right: example of a correction of a fuzzy partition with three fuzzy sets.



NEFCLASS: Pruning the Rules

Objective: Remove variables, rules and fuzzy sets,
in order to improve interpretability and generalization ability.

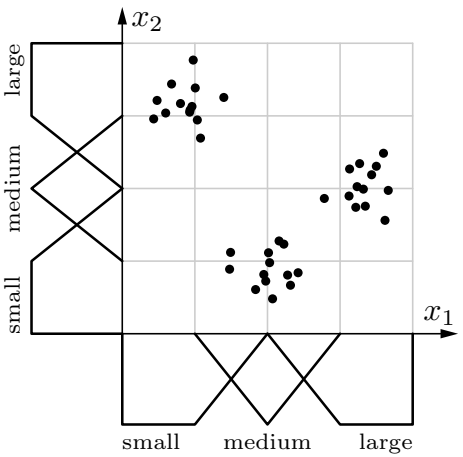
repeat
  select pruning method;
  repeat
    execute pruning step;
    train fuzzy sets;
    if no improvement
    then undo step;
  until there is no improvement;
until no further method;

1. Remove variables
   (correlation, information gain etc.)
2. Remove rules
   (performance of a rule)
3. Remove antecedent terms
   (satisfaction of a rule)
4. Remove fuzzy sets

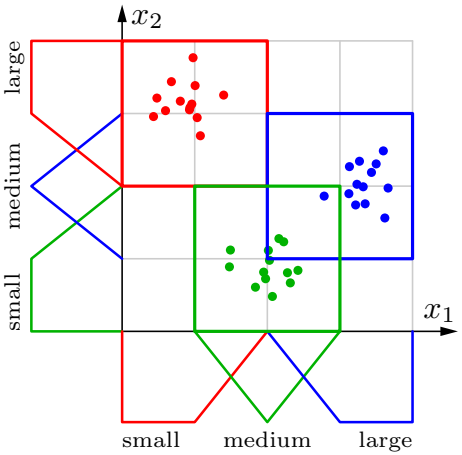
- After each pruning step the fuzzy sets need to be retrained,
  in order to obtain the optimal parameter values for the new structure.
- A pruning step that does not improve performance,
  the system is reverted to its state before the pruning step.

NEFCLASS: Full Procedure

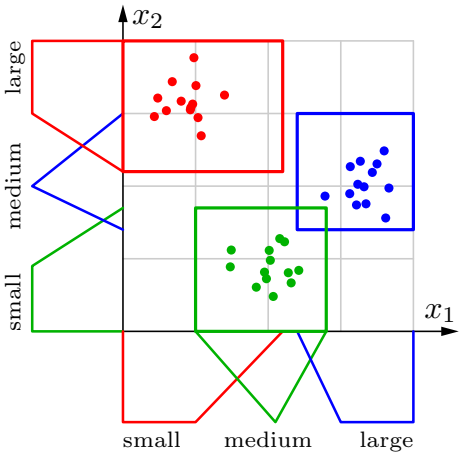
fuzzy partitions



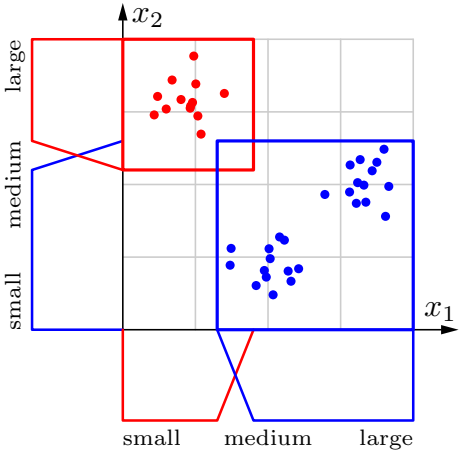
initial rule base

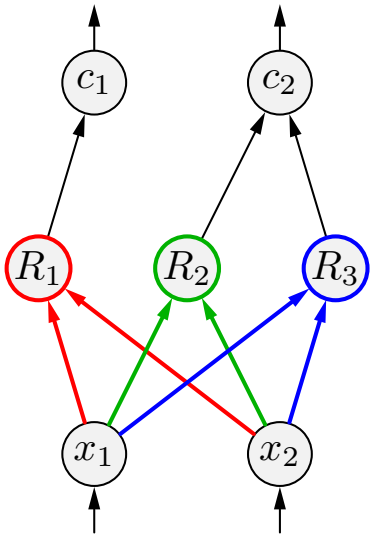


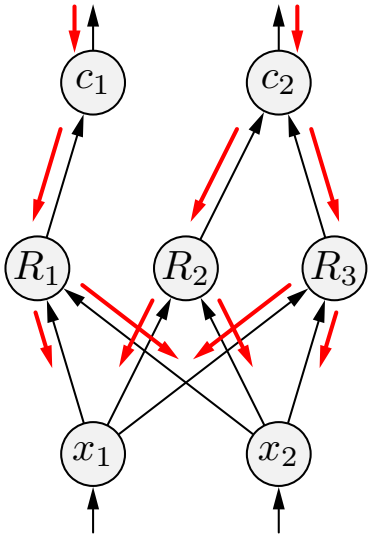
trained rule base

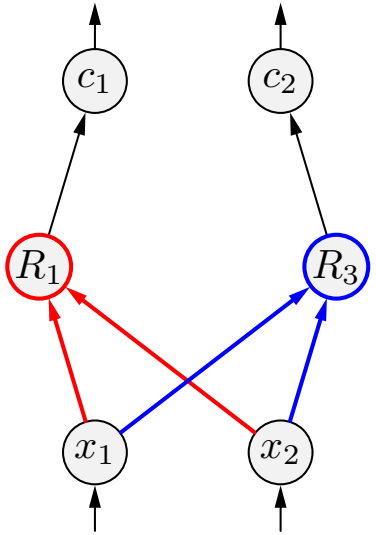


pruned rule base









Neuro-Fuzzy Systems in Finance

Stock Index Prediction (DAX) [Siekmann 1999]

- Prediction of the daily relative changes of the German stock index (Deutscher Aktienindex, DAX).
- Based on time series of stock indices and other quantities between 1986 and 1997.

Input Variables:

- DAX (Germany)
- Composite DAX (Germany)
- Dow Jones industrial index (USA)
- Nikkei index (Japan)
- Morgan–Stanley index Germany
- Morgan–Stanley index Europe
- German 3 month interest rate
- return Germany
- US treasure bonds
- price to income ratio
- exchange rate DM / US-$
- gold price

DAX Prediction: Example Rules

- trend rule:
  if DAX is decreasing **and** US-$ is decreasing
  then DAX prediction is decreasing
  with high certainty

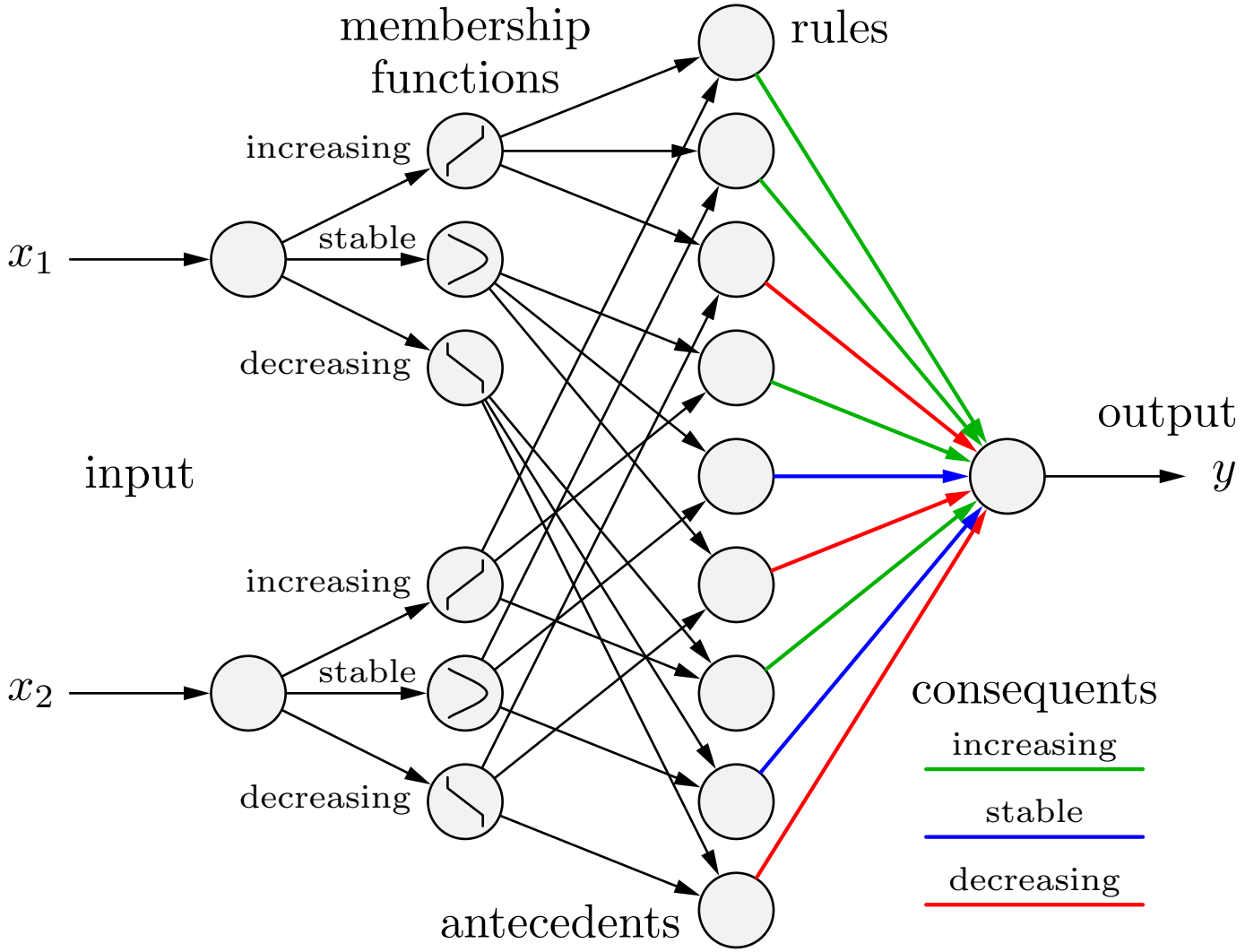
- turning point rule:
  if DAX is decreasing **and** US-$ is increasing
  then DAX prediction is increasing
  with low certainty

- delay rule:
  if DAX is stable **and** US-$ is decreasing
  then DAX prediction is decreasing
  with very high certainty

- general form:
  if x_1 is μ_1 **and** x_2 is μ_2 **and** ... **and** x_n is μ_n
  then y is ν
  with certainty c

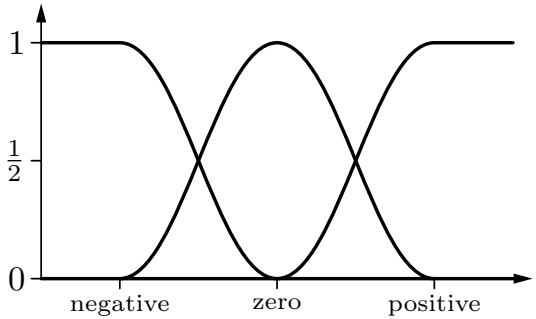
Initial rules may be provided by financial experts.

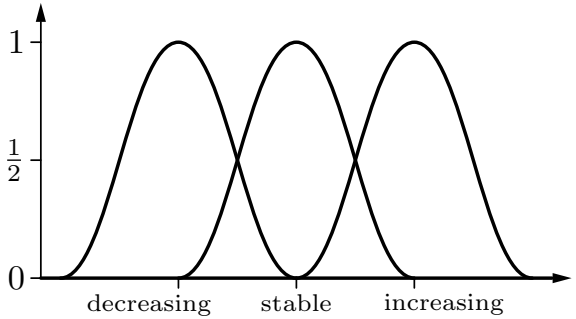
DAX Prediction: Architecture



DAX Prediction: From Rules to Neural Network

- Finding the membership values (evaluate membership functions):





- Evaluating the rules (computing the rule activation for r rules):

$$\forall j \in \{1, \ldots, r\}: \qquad \tilde{a}_j(x_1, \ldots, x_d) = \prod_{i=1}^{d} \mu_j^{(i)}(x_i).$$

- Accumulation of r rule activations, normalization:

$$y = \sum_{j=1}^{r} w_j \, \frac{c_j \, \tilde{a}_j(x_1, \ldots, x_d)}{\sum_{k=1}^{r} c_k \, \tilde{a}_k(x_1, \ldots, x_d)}, \qquad \text{where} \qquad \sum_{j=1}^{r} w_j = 1.$$

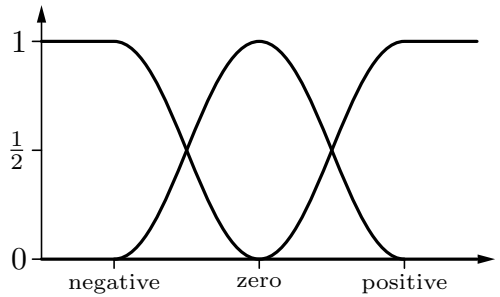
DAX Prediction: Training the Network

- Membership degrees of different inputs share their parameters, e.g.

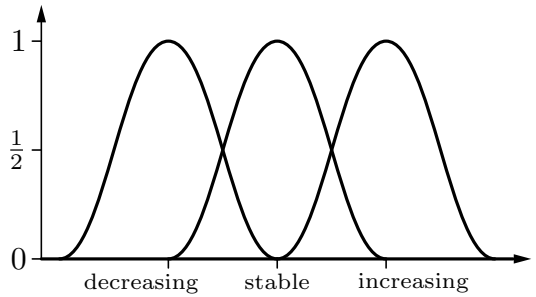
$$\mu_{\text{stable}}^{(\text{DAX})} = \mu_{\text{stable}}^{(\text{Composite DAX})}$$

Advantage: number of free parameters is reduced.

- Membership functions of the same input variable must not "pass each other", but must preserve their original order:



$\mu_{\text{negative}} < \mu_{\text{zero}} < \mu_{\text{positive}}$



$\mu_{\text{decreasing}} < \mu_{\text{stable}} < \mu_{\text{increasing}}$

Advantage: optimized rule base remains interpretable.

- The parameters of the fuzzy sets, the rule certainties, and the rule weights are optimized with a backpropagation approach.

- Pruning methods are employed to simplify the rules and the rule base.

Neuro-Fuzzy Systems: Summary

- Neuro-fuzzy systems can be useful for **discovering knowledge** in the form of rules and rule bases.
- The fact that they are **interpretable**, allows for plausibility checks and improves acceptance.
- Neuro-fuzzy systems exploit tolerances in the underlying system in order to find near-optimal solutions.
- Training procedures for neuro-fuzzy systems have to be able to cope with restrictions in order to preserve the semantics of the original model.
- **No (fully) automatic model generation.**
  $\Rightarrow$ A user has to work and interact with the system.
- Simple training methods support **exploratory data analysis**.