TÉCNICO LISBOA

Intelligent Systems

Susana M. Vieira

Universidade de Lisboa, Instituto Superior Técnico

IS4, Center of Intelligent Systems, IDMEC, LAETA, Portugal
{susana.vieira}@tecnico.ulisboa.pt

TÉCNICO LISBOA

FUZZY MODELING

SI4 –Fuzzy Modeling

Reading:

- R. Babuska. ***Fuzzy Modeling for Control***. Kluwer Academic Publishers, 1998.
- J.M.C. Sousa and U. Kaymak. ***Fuzzy Decision Making in Modeling and Control***. World Scientific Series in Robotics and Intelligent Systems, vol. 27, Dec. 2002.

TÉCNICO LISBOA

MODELING

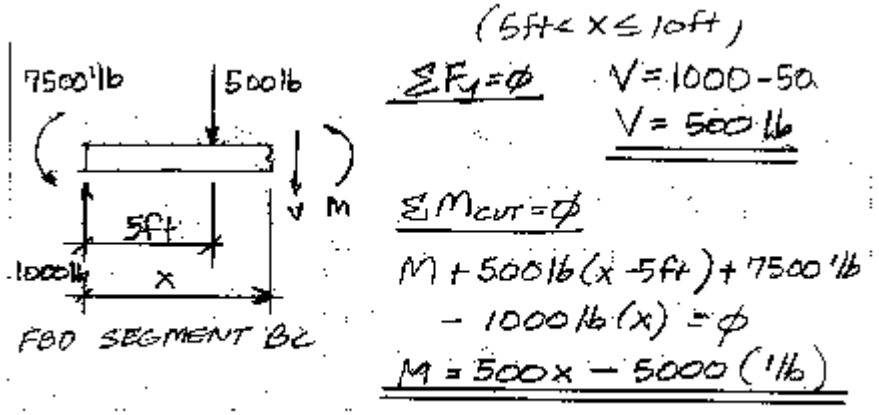
Reading:

- Katsuhiko Ogata. ***Modern Control Engineering***, 5th Edition. Prentice Hall, 2010.

Modeling techniques

- **physical modeling**
  - ***first principles***: differential equations
  - ***linearization*** around an operating point

- **system identification**
  - measure input-output data
  - postulate model structure
  - estimate model parameters from data



Mathematical models

- A representation of reality
- However, appropriateness of a model is coupled to one's goals
- A **model** consists of
  - **Structure**: variables, inputs, outputs and **types of relations** amongst them
  - **Parameters**: free variables after a structure is selected
  - **Search method**: method to derive (identify) the optimal parameters (optimization)

Modeling of dynamical systems

- **Model of a dynamic system** – set of equations that represents the system accurately.
- A mathematical model is ***not unique*** to a given system; it depends on the application of the model.
- Dynamics of many systems (mechanical, electrical, economic, biological, etc.) may be described in terms of ***differential equations***.
- **Simplicity vs. accuracy** – to improve accuracy of a model, complexity is increased.
- In general, first a simplified model is derived to get a good feeling for the solution.

Linear systems

$$y = \sum_{i=1}^{n} a_i x_i$$

- ***Principle of superposition*** is valid - the response produced by the application of two different inputs is the sum of the two individual responses.
- **Linear time-invariant systems** - differential equation where the coefficients are constants or functions only of the independent variable.
- **Linear time-variant systems** - coefficients are function of time.
  Example: spacecraft control system.

Nonlinear systems

- The ***Principle of superposition*** does not apply:

$$\frac{d^2x}{dt^2}+\left(\frac{dx}{dt}\right)^2+x=A\sin\omega t$$

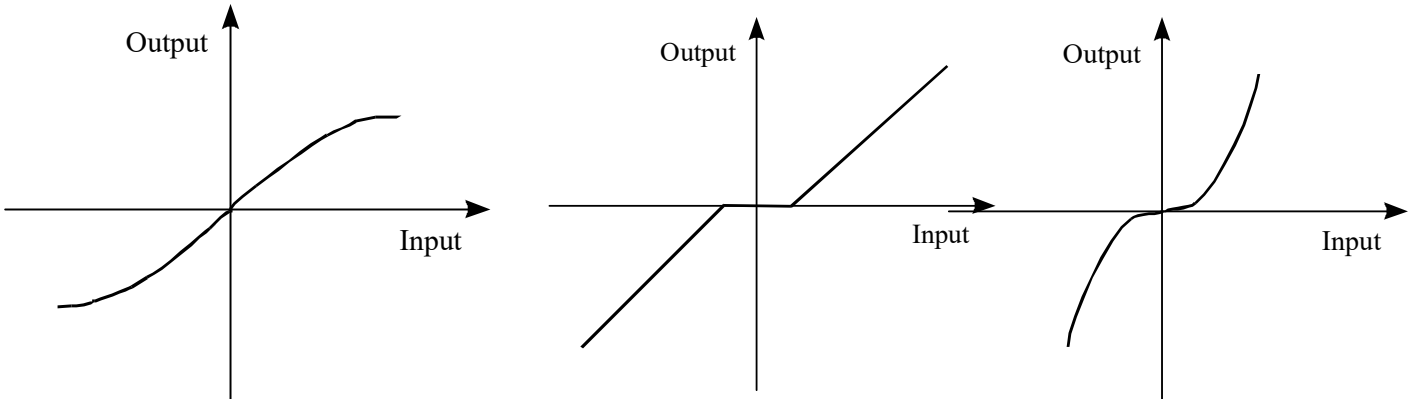
$$\frac{d^2x}{dt^2}+\left(x^2-1\right)\frac{dx}{dt}+x=0$$

$$\frac{d^2x}{dt^2}+\frac{dx}{dt}+x+x^3=0$$

- Systems are generally ***nonlinear***, but in limited operating ranges can be ***approximated by linear equations***.

Nonlinear systems

- Examples of nonlinearities are, e.g., ***saturation***, ***dead-zone***, ***square-law*** (dampers at high velocities become proportional to the square of the velocity).



Modeling in state-space

- **State variables**: n variables $x_1, x_2, \ldots, x_n$ necessary to completely describe the behavior of a system.
- **State vector**: state variables can be considered the n components of a state vector $\mathbf{x}(t)$.
- $\mathbf{x}(t)$ determines uniquely the system state for any time $t \geq 0$, given the state at $t = t_0$ and the input $\mathbf{u}(t)$ for $t \geq 0$.
- **State space**: n-dimensional space whose coordinate space consists of the $x_1, x_2, \ldots, x_n$ axis. Any state is represented by a point in the space.

State-space equations

- State-space equations:

$$\begin{aligned}
\dot{x}_1(t) &= f_1(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\
\dot{x}_2(t) &= f_2(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\
&\vdots \\
\dot{x}_n(t) &= f_n(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t)
\end{aligned}$$

- Outputs of the system:

$$\begin{aligned}
y_1(t) &= g_1(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\
y_2(t) &= g_2(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\
&\vdots \\
y_m(t) &= g_m(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t)
\end{aligned}$$

State-space vectors

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}$$

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_m(t) \end{bmatrix}$$

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_r(t) \end{bmatrix}$$

$$\mathbf{f}(\mathbf{x},\mathbf{u},t) = \begin{bmatrix} \dot{x}_1(t) = f_1(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\ \dot{x}_2(t) = f_2(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\ \vdots \\ \dot{x}_n(t) = f_n(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \end{bmatrix}$$

$$\mathbf{g}(\mathbf{x},\mathbf{u},t) = \begin{bmatrix} y_1(t) = g_1(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\ y_2(t) = g_2(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \\ \vdots \\ y_m(t) = g_m(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_r; t) \end{bmatrix}$$

State-space equations

- Equations can be written as:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, \mathbf{u}, t)$$
$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}, \mathbf{u}, t)$$

- Linearizing around a working point, and considering the system time-invariant:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

Nonlinear models

State-space models

- Continuous time:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$
$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t), t)$$

- Discrete time:

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), k)$$
$$\mathbf{y}(k) = \mathbf{g}(\mathbf{x}(k), \mathbf{u}(k), k)$$

Input-output models

- Continuous time:

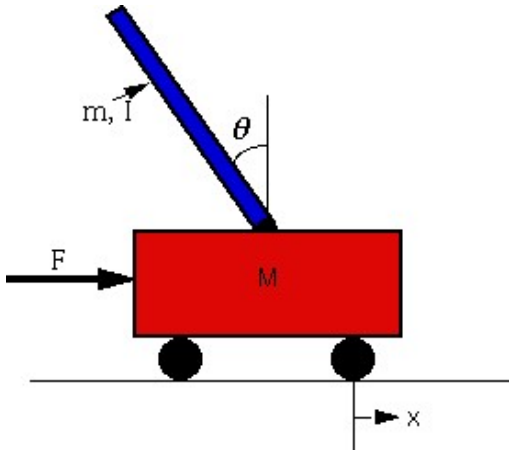
$$y^{(n)}(t) = f\left(y^{(n-1)}(t),\ldots,y(t),u^{(m)}(t),\ldots,u(t)\right)$$

- Discrete time:

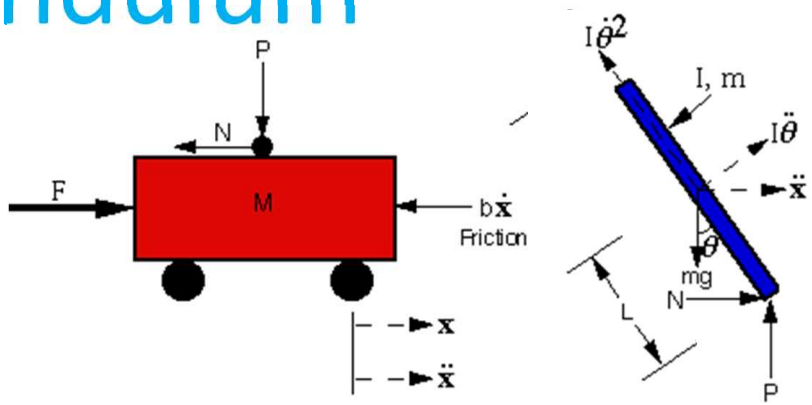
$$y(k+1) = f(y(k),y(k-1),\ldots,y(k-n_y+1),$$
$$u(k),u(k-1),\ldots,u(k-n_u+1))$$

System: inverted pendulum

- M - mass of the cart (Kg)
- m - mass of the pendulum (Kg)
- b - friction of the cart (N/m/s)
- l - length to pendulum center of mass (m)
- F - force applied to the cart (N)
- x - cart position coordinate (m)
- θ - pendulum angle from vertical (rad)



Inverted pendulum



- **Free Body Diagram (horizontal direction):** $\sum F_x = 0$

$$F = M\ddot{x} + b\dot{x} + N$$

$$N = m\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta$$

- Substituting the second in the first equation:

$$(M+m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F$$

Inverted pendulum

- **Free Body Diagram (vertical direction):** $\sum F_y = 0$

$$P\sin\theta + N\cos\theta - mg\sin\theta = ml\ddot{\theta} + m\ddot{x}\cos\theta$$

- Sum of moments around the centroid: $\sum M = 0$

$$-Pl\sin\theta - Nl\cos\theta = I\ddot{\theta}$$

- Combining these equations:

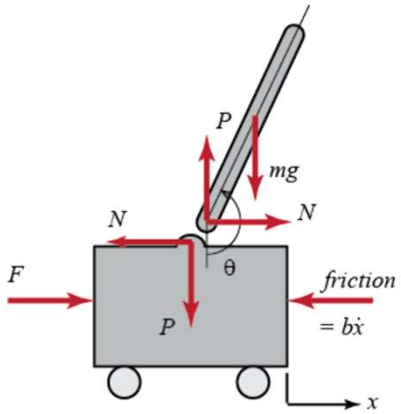
$$(I + ml^2)\ddot{\theta} + mgl\sin\theta = -ml\ddot{x}\cos\theta$$

Inverted pendulum

- **Nonlinear model:**

$$(M+m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F$$

$$(I+ml^2)\ddot{\theta} + mgl\sin\theta = -ml\ddot{x}\cos\theta$$



- **Linearization around:** $\theta = \pi$
  - $\theta = \pi + \phi$, where ϕ is a small angle from the vertical upward direction
  - And so,

$$\cos\theta = \cos(\pi+\phi) \approx -1,$$

$$\sin(\theta) = \sin(\pi+\phi) \approx -\phi,$$

$$\left(\frac{d\theta}{dt}\right)^2 = \left(\frac{d\phi}{dt}\right)^2 \approx 0$$

Inverted pendulum

- **Linearized model:**

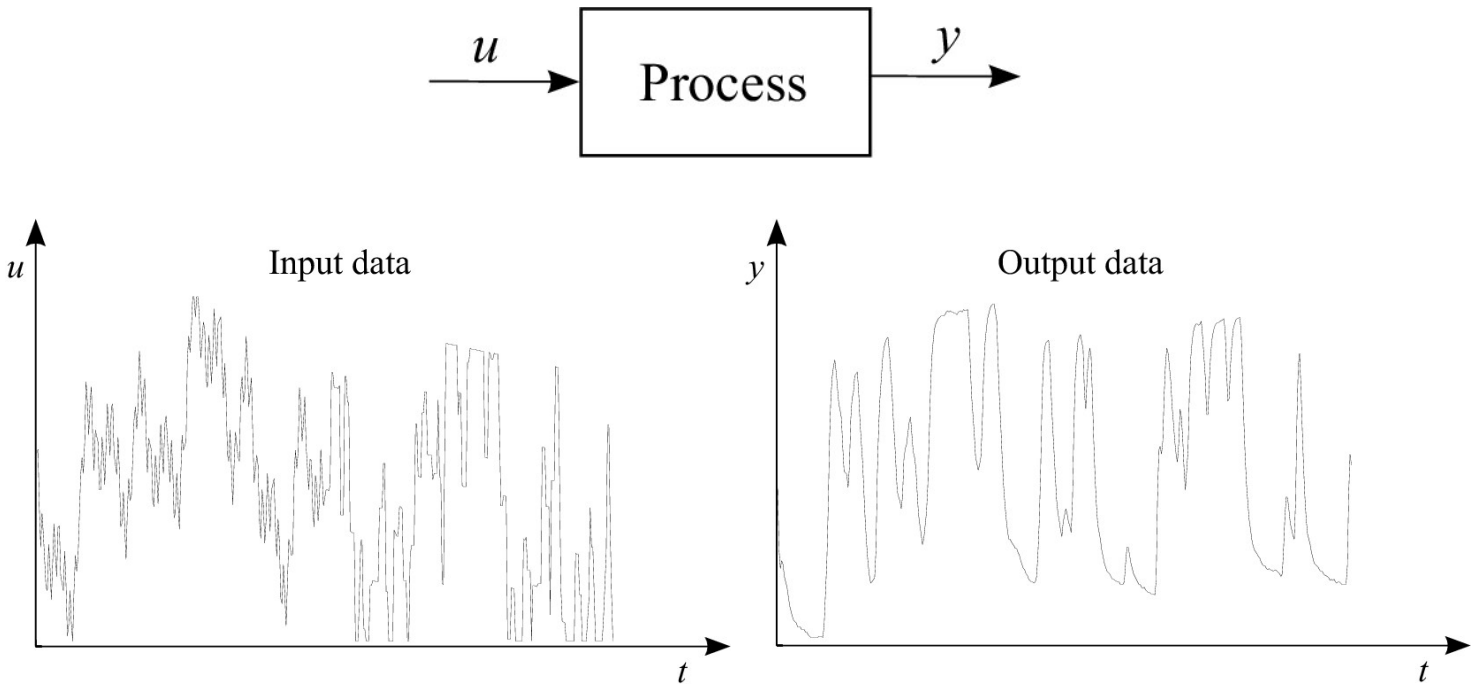
$$(I+ml^2)\ddot{\phi} - mgl\phi = ml\ddot{x}$$
$$(M+m)\ddot{x} + b\dot{x} + ml\ddot{\phi} = u$$

- **State-space model:**

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Mml^2} \\ 0 \\ \frac{ml}{I(M+m)+Mml^2} \end{bmatrix} u$$

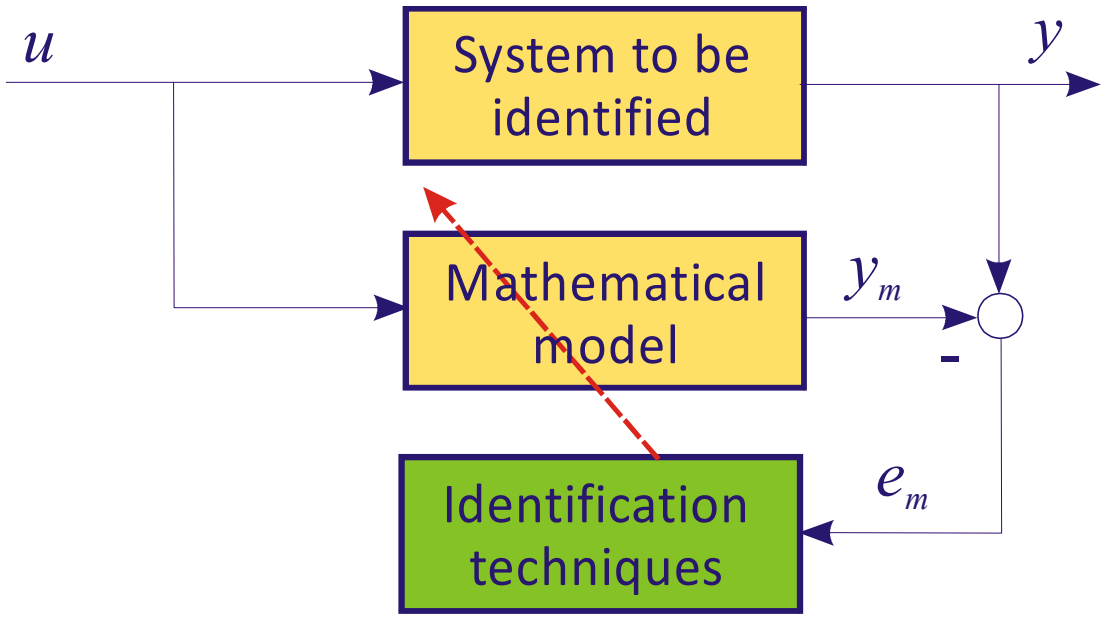
$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

System identification



$u(1), u(2),...,u(N)$ $y(1), y(2),...,y(N)$

Parameter identification



Modeling

- Models can be ***static*** or ***dynamic***
- **Why build models?**
  - For classification purposes
  - To predict a system's behaviour
  - To **explain interactions** and **relationships** between inputs and outputs
  - To simulate a system and design controllers
- **Modeling process**
  - specify and parametrize a class of models
  - perform parameter identification
  - validate model

Data based modeling

- Fuzzy systems
- Radial basis function networks
- Support vector machines
- Multi-layer perceptron
- ...

- Fuzzy systems can be **interpretable**!
- Fuzzy sets can close the gap between *symbolic processing* and *numerical computations*.

Fuzzy modeling

Parameters to be determined:

- Rule base
  (rule mapping)
- Definition of membership functions
  (inputs and outputs)
- Estimation of consequent parameters
  (for Sugeno systems)

How to obtain fuzzy models?

- **Expert knowledge driven**
  - Initialize FM using expert knowledge
  - Optimize parameters with expert knowledge
- **Data driven**
  - Partition input/output spaces and use available data to induce the rules or
  - Apply fuzzy clustering with projection of fuzzy clusters
  - Refine the parameters (e.g., neuro-fuzzy approach)
- **Combinations of above**
  - data-driven estimation of optimal parameters after a suitable expert-driven initialization
  - **…**

Steps to build fuzzy systems

- Determine the **relevant/available input** and **output variables/features**
- Determine suitable **universe of discourse** and a **term set** for the variables
- Define **membership functions** for **linguistic terms**
- Determine **fuzzy rules** for the rule base
- Determine model choices and parameters (**including inference operators**)
- Tune the system

Rule induction

- **Antecedents**
  - ***Location*** of kernels
    membership function positions
  - ***Extension*** of kernels
    spread of membership functions
  - ***Shape*** of kernels
    type of membership functions
- **Consequents**
  - Associated with kernel estimation of consequent parameters

Rule base optimization

Interpretability influenced by:

- ***Input and output variables***
  - number, relevance and interpretation
- ***Membership functions*** (MFs)
  - number and label meanings
- ***Rule base***
  - number of rules and interpretation; possibility for incomplete rules

Using expert knowledge

When designing fuzzy systems, use expert knowledge whenever available

- **Possible strategies:**
  - initialize rule base with expert knowledge and optimize with data
  - design rule base from data and optimize with expert knowledge
  - use experts to validate final rule base
  - use experts to provide boundary conditions for rule base optimization

Selecting # of antecedents

- **A priori knowledge** (experts, dynamics, etc.)
- **Regularity criterion** – based on cross-validation
  - Split training set randomly into two parts (*A* and *B*)
  - Minimize regularity criterion:

$$RC = \frac{1}{2}\left[\frac{1}{k_A}\sum_{i=1}^{k_A}\left(y_i^A - \hat{y}_i^{AB}\right)^2 + \frac{1}{k_B}\sum_{i=1}^{k_B}\left(y_i^B - \hat{y}_i^{BA}\right)^2\right]$$

  - Variables selected incrementally until regularity criterion increases
- …

Selecting # of antecedents

Feature Selection (FS)

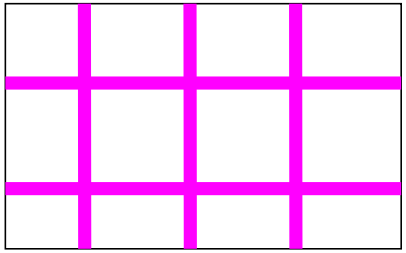
- Principal Component Analysis (PCA)
- Curvilinear Component Analysis (CCA)
- **…**
- Tree search methods
  - Bottom-up
  - Top-down
- Using metaheuristics: genetic algorithms, ant colony optimization, particle swarm optimization,…)

Input space partitioning

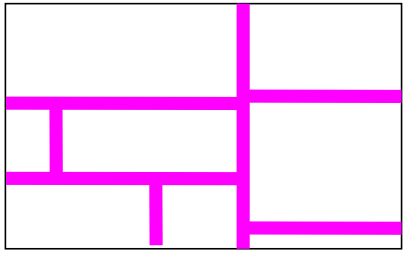
- Rule induction partitions the input space into a number of fuzzy (overlapping) regions
- The type of partition generated depends on chosen algorithm
- Optimization modifies membership parameters to define a partition that minimizes output error

Input space partitioning

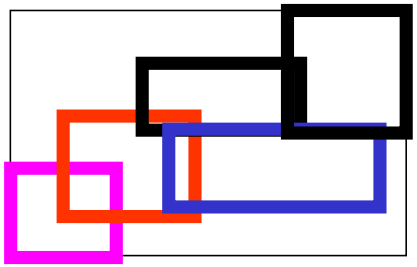
- Grid partition



- Tree partition



- Scatter partition



Partitioning algorithms

- **All rules** – **product** of number of linguistic terms for each antecedent:

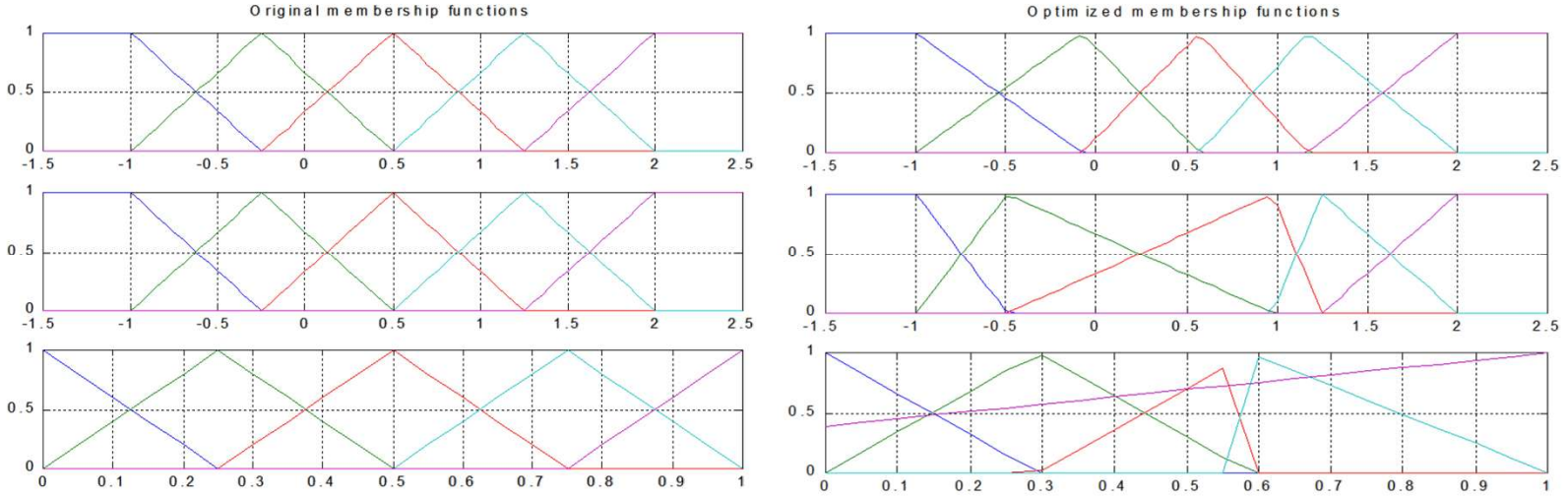
$$K = \prod_{i=1}^{n} \#\mathrm{L}X^{i}$$

- Partition **refinement**
- Partition **optimization**, using e.g. neuro-fuzzy methods or genetic algorithms.
- Decision trees (**tree partition**) based on an information criterion

➢ **Fuzzy clustering** to determine best fit of data

Example: optimize MFs

- ***Define partition*** and a suitable objective function
- ***Use optimization method*** to derive MFs
  - Triangular MFs: derivative-free methods
  - Gaussian MFs: derivative-based methods

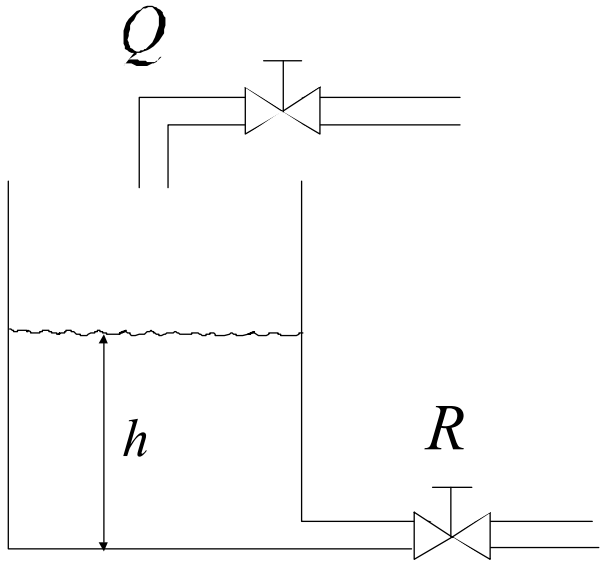


Example: liquid level revisited

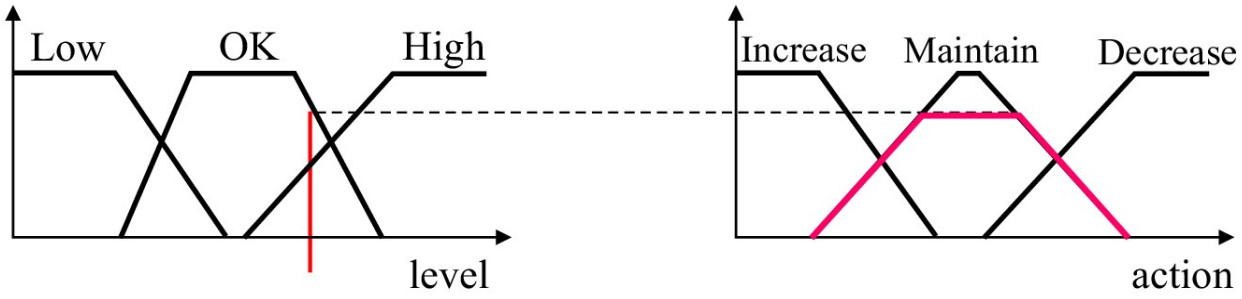
If level is *low* **then** *increase* valve opening

If level is *OK* **then** *maintain* valve opening

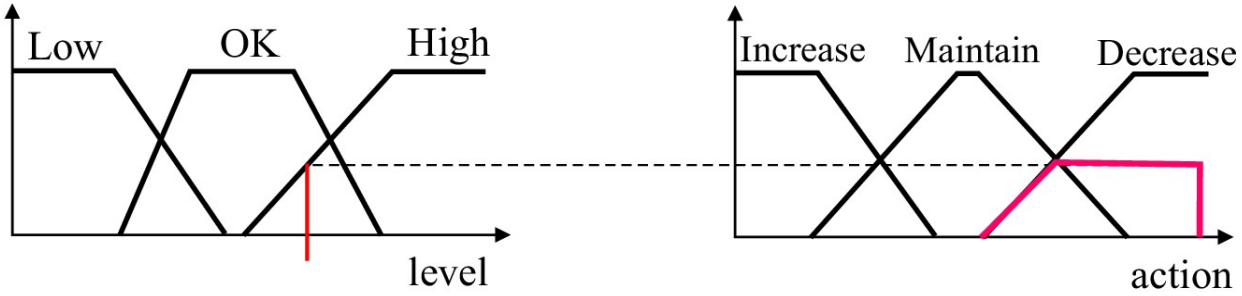
If level is *high* **then** *decrease* valve opening



Fired fuzzy rules

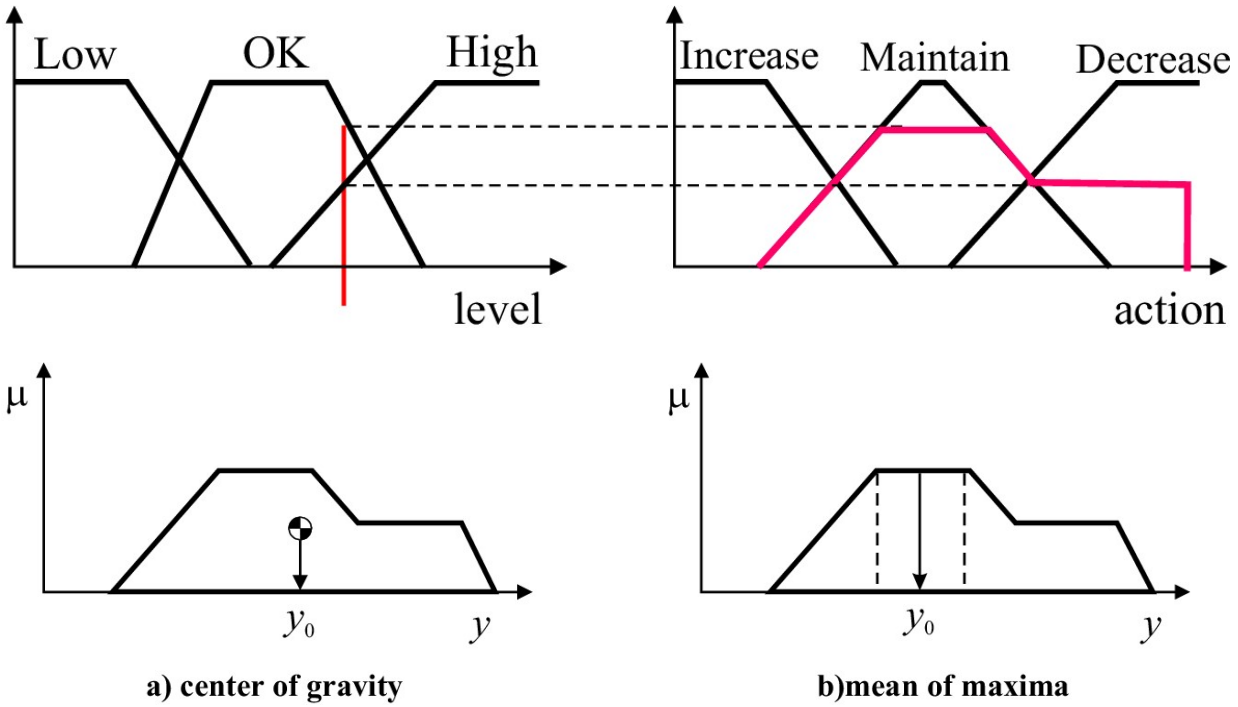


If level is *OK* **then** *Maintain* valve opening



If level is *High* **then** *Decrease* valve opening

Aggregation and defuzzifucation



a) center of gravity

b)mean of maxima

TÉCNICO LISBOA

Optimization

Reading: Part I Optimization: Chapter 5,6,and 7

- J.-S. Jang, C.-T. Sun and E. Mizutani. ***Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence***. Prentice Hall, New Jersey, 1997.

Optimization

- **Derivative-based optimization**
  - Steepest descent (gradient) methods
  - Newton's method
- **Derivative-free optimization**
  - Simplex
  - Simulated Annealing
  - Genetic Algorithms
  - Ant Colony Optimization
  - ...

Derivative-based optimization

- **Goal**: Solving optimization (nonlinear) problems using derivative information
- **Methods**:
  - Gradient based optimization
  - Steepest descent
  - Newton methods
  - Conjugate gradient
  - Nonlinear least-squares problems

Derivative-based optimization

- Methods used in:
  - Deriving membership functions
  - Deriving consequents of Takagi-Sugeno models
  - Neural network learning
  - Regression analysis in nonlinear models
  - Optimization of nonlinear neuro-fuzzy models
  - **…**
- ***Methods will be introduced as needed in the next lectures.***

TÉCNICO LISBOA

DATA CLUSTERING

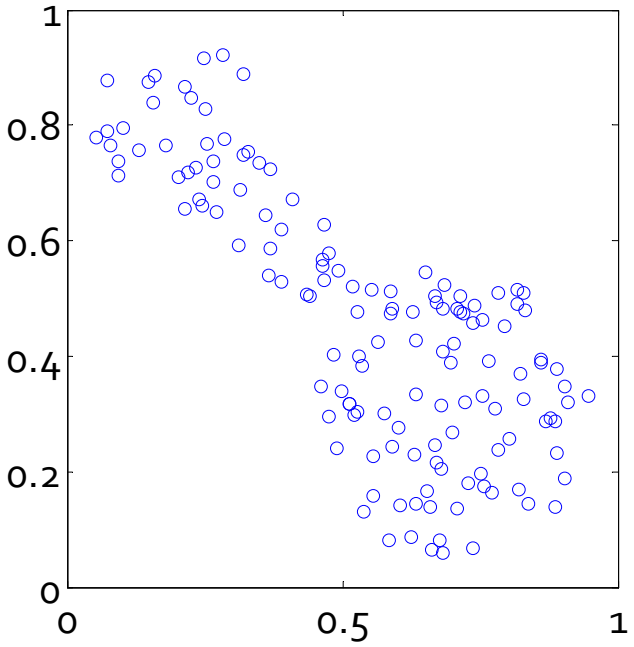
Reading:

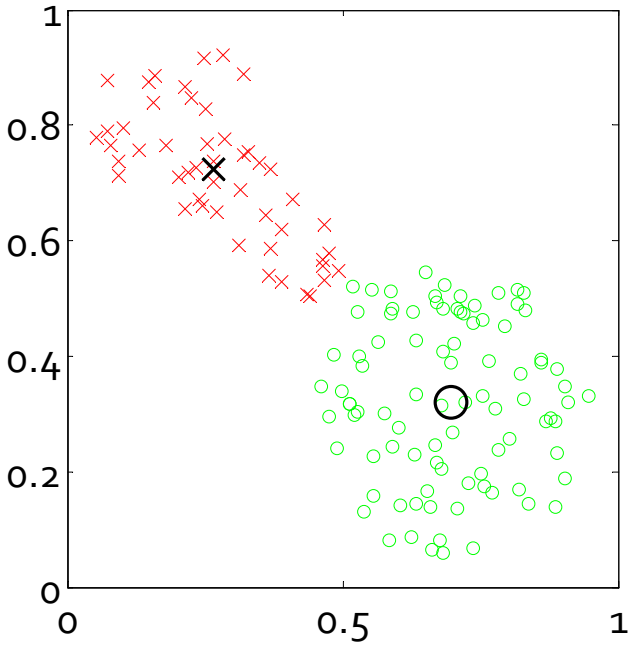
- R. Babuska. ***Fuzzy Modeling for Control***. Kluwer Academic Publishers, 1998.
- J. Valente de Oliveira and W. Pedrycz (Edts). ***Advances in Fuzzy Clustering and its Applications***. Wiley, 2007.

Data clustering

- Extensively used for
  - data categorization (e.g. classification)
  - data compression
  - model construction
- Two types
  - Hierarchical clustering
  - Objective function-based clustering
- Partitions data into several groups such that within-group similarity is larger than between-group similarity

Example





Hierarchical clustering

- Proceed by a series of successive mergers of data points (bottom-up) or successive division of data set (top-down)
- Induce a tree-like structure in the data set
- No pre-determined number of clusters
- Clusters are induced after partitioning by thresholding the induced tree based on **similarity**

Similarity

- Based on some notion of distance
- Inputs are usually normalized to the range $[0,1]$
- Inversely related to distance, e.g.

$$S(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{1 + d(\mathbf{x}_i, \mathbf{x}_j)}$$

- Similarity is a reflexive and symmetric relation
- Suitable distance metrics can be defined, even for nominal data

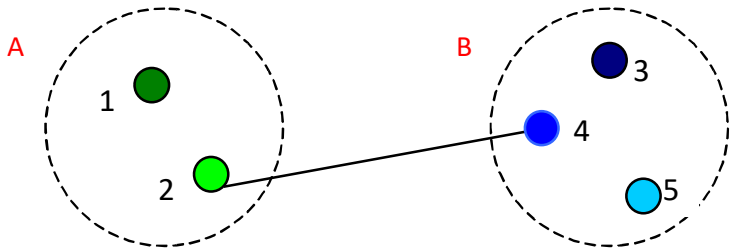
Linkage algorithms

Given the data $\mathbf{x}_k = [x_{1k}, x_{2k}, \ldots, x_{nk}]^T, \quad k = 1, \ldots, N$

1. Start with N clusters and compute the $N \times N$ matrix of similarities
2. Determine most similar clusters i^*, j^*
3. Merge clusters i^* and j^* for form new cluster i'.
4. Delete the rows and columns corresponding to i^* and j^* in the similarity matrix.
5. Determine the similarity between i' and all other remaining clusters.

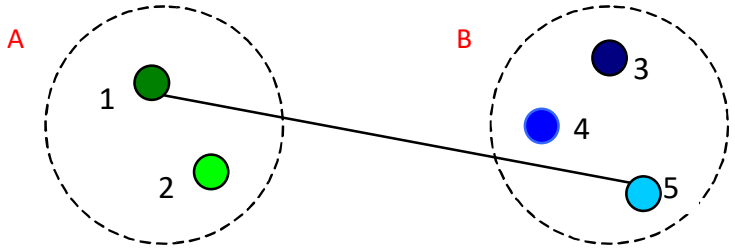
Linkage variations

Single linkage



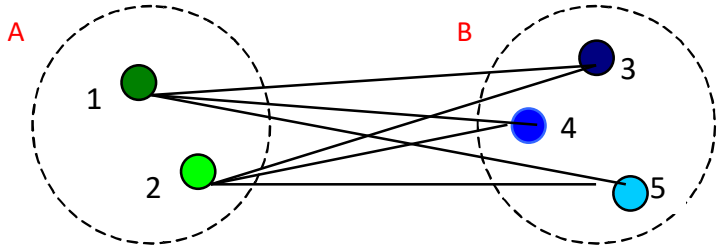
$$d(A,B) = \min_{\mathbf{x}_i \in A, \mathbf{x}_j \in B} d(\mathbf{x}_i, \mathbf{x}_j)$$

Complete linkage



$$d(A,B) = \max_{\mathbf{x}_i \in A, \mathbf{x}_j \in B} d(\mathbf{x}_i, \mathbf{x}_j)$$

Average linkage



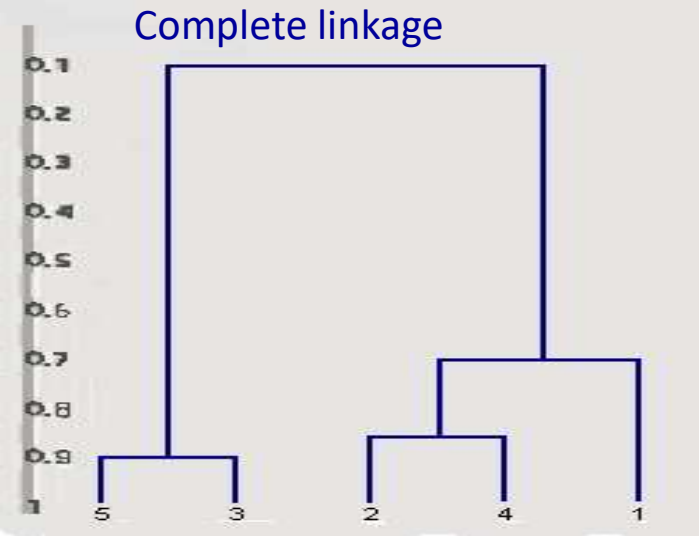
$$d(A,B) = \frac{1}{|A||B|} \sum_{\mathbf{x}_i \in A} \sum_{\mathbf{x}_j \in B} d(\mathbf{x}_i, \mathbf{x}_j)$$

Linkage example

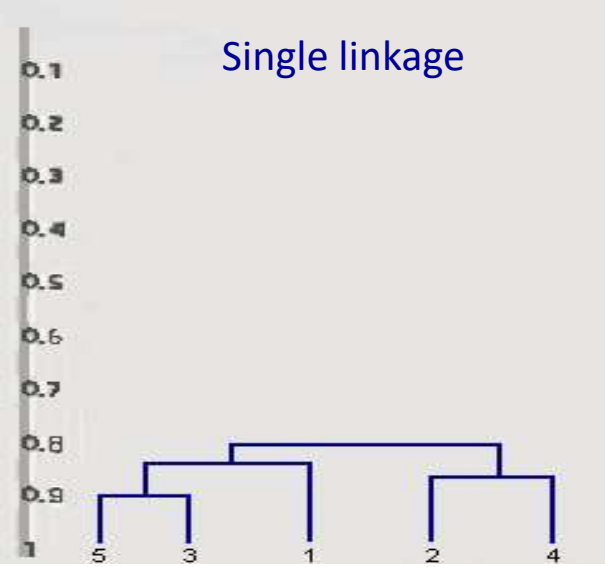
$$\mathbf{S} = \{\mathbf{s}_{ij}\} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & & & & \\ 2 & 0.8 & 1 & & & \\ 3 & 0.4 & 0.3 & 1 & & \\ 4 & 0.7 & 0.85 & 0.2 & 1 & \\ 5 & 0.83 & 0.1 & 0.9 & 0.25 & 1 \end{array}$$

Clustering 5 objects with complete and single linkage algorithms

Complete linkage



Single linkage



Cost function-based clustering

- Partition data by optimizing an objective function
- Clusters represented by cluster prototypes
- Objective function usually minimizes the distance to cluster prototypes
- Pre-determined number of clusters
- Simultaneous estimation of partition and cluster prototypes

Crisp vs. fuzzy clustering

- **Crisp clustering algorithms**
  - partition the data set into disjoint groups, i.e. each data point belongs to one cluster only
  - similarity is quantified using some metric
- **Fuzzy clustering algorithms**
  - partition the data set into overlapping groups, i.e. each data point belongs to multiple clusters with varying degree of membership
  - similarity is quantified using some metric which is modified by membership values

Clustering algorithms

- Hard c-means (K-means)
- Fuzzy c-means
- Gustafson-Kessel
- Possibilistic clustering
- Mountain clustering
- Subtractive clustering
- *And many, many others…*

K(C)-means clustering

- Partition data into ***disjoint*** sets based on similarity amongst patterns

  Given the data $\mathbf{x}_k = [x_{1k}, x_{2k}, \ldots, x_{nk}]^T \in \Re^n,\ k = 1, \ldots, N$

  Find the **crisp partition matrix**:

$$\mathbf{U} = \begin{bmatrix} \mu_{11} & \cdots & \mu_{1N} \\ \vdots & \ddots & \vdots \\ \mu_{C1} & \cdots & \mu_{CN} \end{bmatrix},\ \mu_{ij} \in \{0,1\}$$

and the cluster centres: $\mathbf{V} = \{\mathbf{v}_1, \ldots \mathbf{v}_C\},\ \mathbf{v}_i \in \Re^n$

K-means clustering

- The following cost function of dissimilarity measures is minimized

$$J = \sum_{i=1}^{C} J_i = \sum_{i=1}^{C} \left(\sum_{k, \mathbf{x}_k \in G_i} d(\mathbf{x}_k - \mathbf{v}_i) \right), \text{ often } d(\mathbf{x}_k - \mathbf{v}_i) = \left\| \mathbf{x}_k - \mathbf{v}_i \right\|^2$$

- Partition matrix can be calculated if centers are fixed

$$\mu_{ij} = \begin{cases} 1 \text{ if } \left\| \mathbf{x}_j - \mathbf{v}_i \right\|^2 \leq \left\| \mathbf{x}_j - \mathbf{v}_k \right\|^2, k \neq i \\ 0 \text{ otherwise} \end{cases}$$

- Centers can be calculated if partition matrix is fixed

$$\mathbf{v}_i = \frac{1}{|G_i|} \sum_{\substack{k \\ \mathbf{x}_k \in G_i}} \mathbf{x}_k$$

C-means algorithm

1. Initialize cluster centers $\mathbf{V}$
2. Determine the partition matrix $\mathbf{U}$
3. Compute cost function J
   Stop if J is below a threshold or if it has not improved
4. Update the cluster centers $\mathbf{V}$ and iterate from step 2.

Note that $\sum_{i=1}^{C} \mu_{ij} = 1, \ \forall j = 1, \ldots, N$ and $\sum_{i=1}^{C} \sum_{j=1}^{N} \mu_{ij} = N$

TÉCNICO LISBOA

FUZZY CLUSTERING

Reading:

- R. Babuska. ***Fuzzy Modeling for Control***. Kluwer Academic Publishers, 1998.
- J. Valente de Oliveira and W. Pedrycz (Edts). ***Advances in Fuzzy Clustering and its Applications***. Wiley, 2007.
- J.-S. Jang, C.-T. Sun and E. Mizutani. ***Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence***. Prentice Hall, New Jersey, 1997.

Fuzzy c–means

- Partition data into ***overlapping*** sets based on similarity amongst patterns

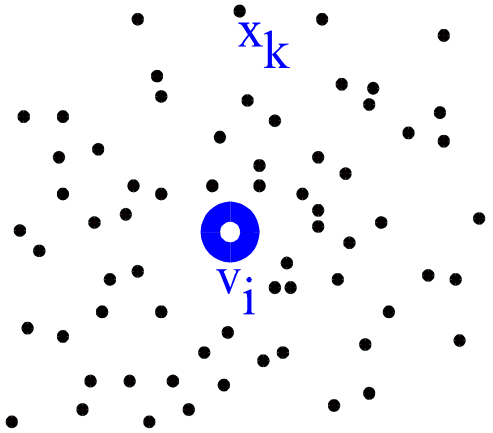
  Given the data $\mathbf{x}_k = [x_{1k}, x_{2k}, \ldots, x_{nk}]^T \in \Re^n,\ k = 1, \ldots, N$

  Find the **fuzzy partition matrix**: $\mathbf{U} = \begin{bmatrix} \mu_{11} & \cdots & \mu_{1N} \\ \vdots & \ddots & \vdots \\ \mu_{C1} & \cdots & \mu_{CN} \end{bmatrix},\ \mu_{ij} \in [0,1]$

and the cluster centres: $\mathbf{V} = \{\mathbf{v}_1, \ldots \mathbf{v}_C\},\ \mathbf{v}_i \in \Re^n$

This is a generalization of hard c-means!

Fuzzy clustering



- Minimize objective function

$$J(\mathbf{X},\mathbf{U},\mathbf{V}) = \sum_{i=1}^{C}\sum_{k=1}^{N} \mu_{ik}^{m} d^2(\mathbf{x}_k,\mathbf{v}_i)$$

- subject to

$$0 \le \mu_{ik} \le 1,\ i = 1,\ldots,C\ ,\ k = 1,\ldots,N \qquad \text{membership degree}$$

$$\sum_{i=1}^{C} \mu_{ik} = 1, \quad k = 1,\ldots,N \qquad \text{total membership}$$

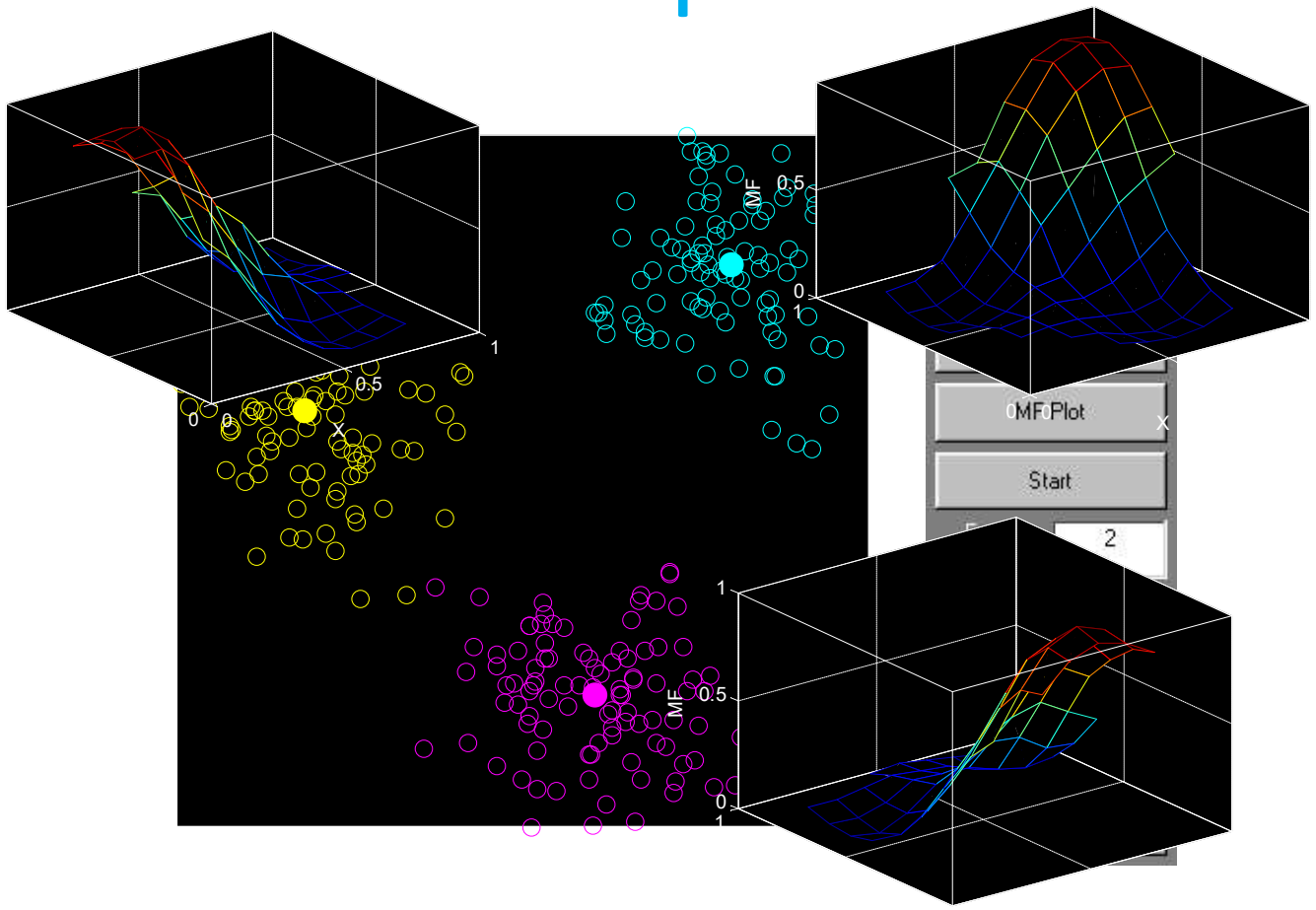
$$0 < \sum_{k=1}^{N} \mu_{ik} < N,\ i = 1,\ldots,C \qquad \text{no cluster empty}$$

$m \in (1,\infty)$ is the fuzziness parameter

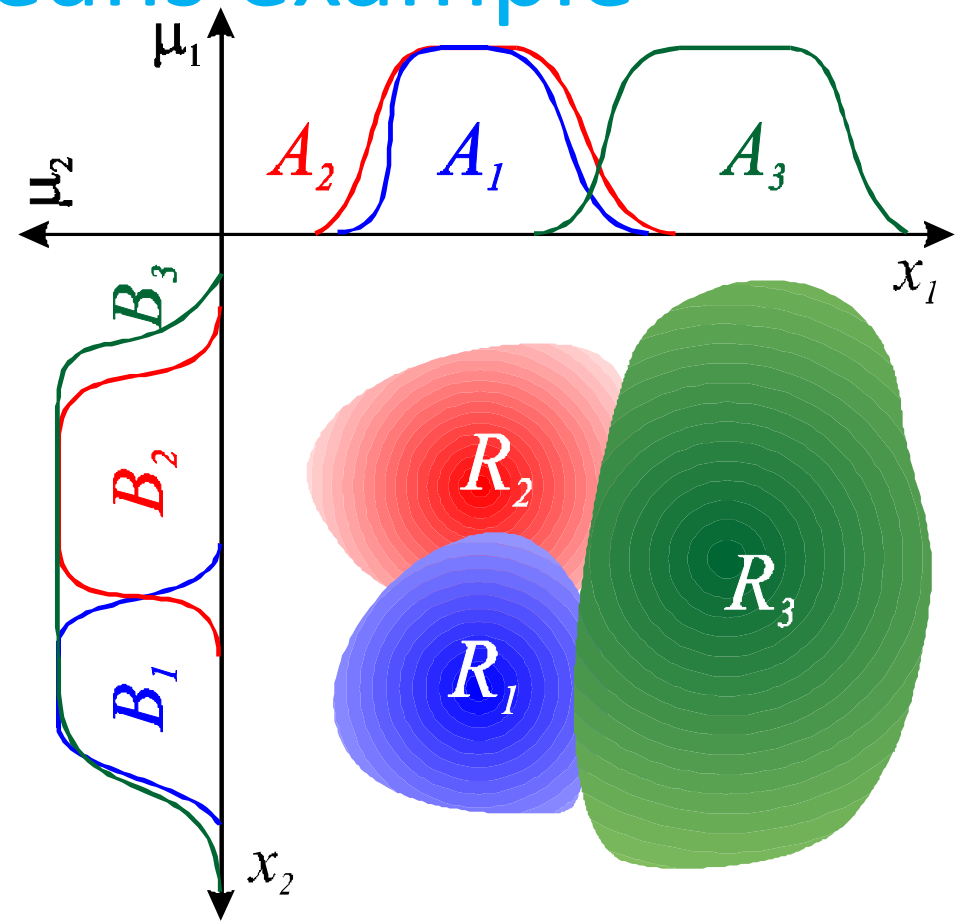
Fuzzy c-means algorithm

- Initialize $\mathbf{V}$ or $\mathbf{U}$
- **Repeat**
  1. Compute cluster centers $\mathbf{v}_i = \dfrac{\sum_{k=1}^{N} \mu_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{N} \mu_{ik}^m}$ Assumes partition matrix is fixed
  2. Calculate distances $d_{ik}^2 = (\mathbf{x}_k - \mathbf{v}_i)^T (\mathbf{x}_k - \mathbf{v}_i)$
  3. Update partition matrix $\mu_{ik} = \dfrac{1}{\sum_{j=1}^{C} (d_{ik}^2 / d_{jk}^2)^{1/(m-1)}}$ Assumes cluster centers are fixed
- **Until** $\|\Delta \mathbf{U}\| < \varepsilon$ Other stopping criteria are possible

Fuzzy c-means example



Fuzzy c-means example



Distance measures

- Euclidian norm:

$$d^2(\mathbf{x}_k, \mathbf{v}_i) = (\mathbf{x}_k - \mathbf{v}_i)^T (\mathbf{x}_k - \mathbf{v}_i)$$

- Inner-product norm:

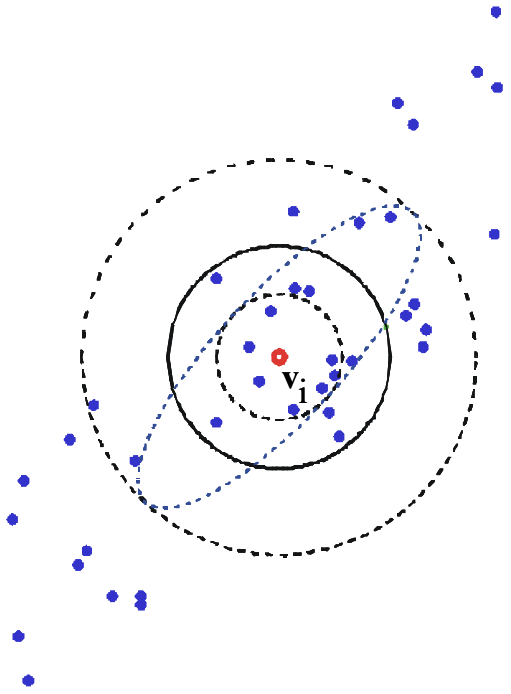
$$d^2(\mathbf{x}_k, \mathbf{v}_i) = (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{A} \ (\mathbf{x}_k - \mathbf{v}_i)$$

$\mathbf{A}$ is diagonal

- Mahalanobis norm:

$$d^2(\mathbf{x}_k, \mathbf{v}_i) = (\mathbf{x}_k - \mathbf{v}_i)^T \ \mathbf{F}_i^{-1} \ (\mathbf{x}_k - \mathbf{v}_i)$$

Rotated clusters



Gustafson-Kessel clustering

- Uses an adaptive distance metric

$$d^2(\mathbf{x}_k - \mathbf{v}_i) = (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{A}_i (\mathbf{x}_k - \mathbf{v}_i)$$

$$\mathbf{A}_i = \left|\mathbf{F}_i\right|^{1/n} \mathbf{F}_i^{-1}$$

- **Fuzzy covariance matrix**

$$\mathbf{F}_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m}$$

- Clusters are constrained by volume
- Clusters adapt themselves to the shape and location of data

GK algorithm

- **Repeat**
  1. Compute cluster centers $\mathbf{v}_i = \dfrac{\sum_{k=1}^{N} \mu_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{N} \mu_{ik}^m}$ Assumes partition matrix is fixed
  2. Calculate covariance matrices and distances

$$d_{ik}^2 = |\mathbf{F}_i|^{1/n} (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{F}_i^{-1} (\mathbf{x}_k - \mathbf{v}_i) \quad \mathbf{F}_i = \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m}$$

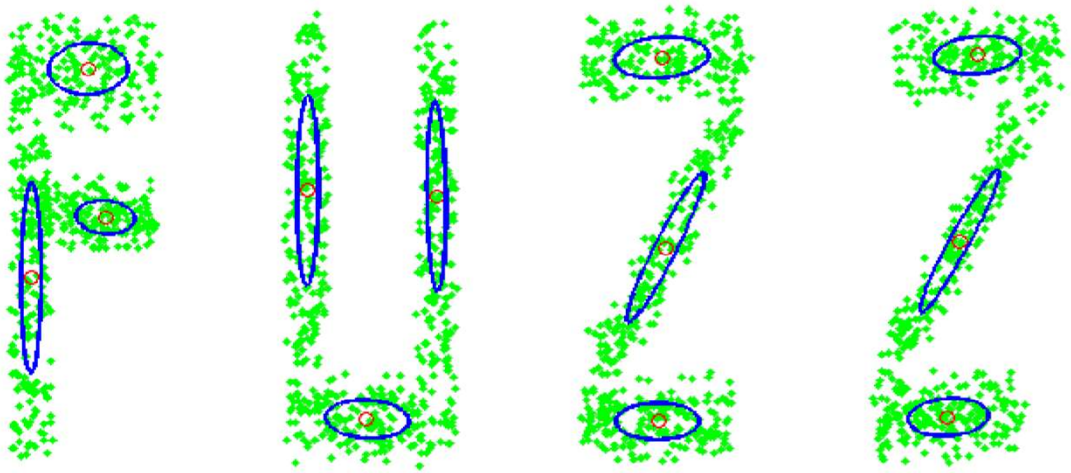
  3. Update partition matrix

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{C} (d_{ik}^2 / d_{jk}^2)^{1/(m-1)}}$$ Assumes cluster centers are fixed

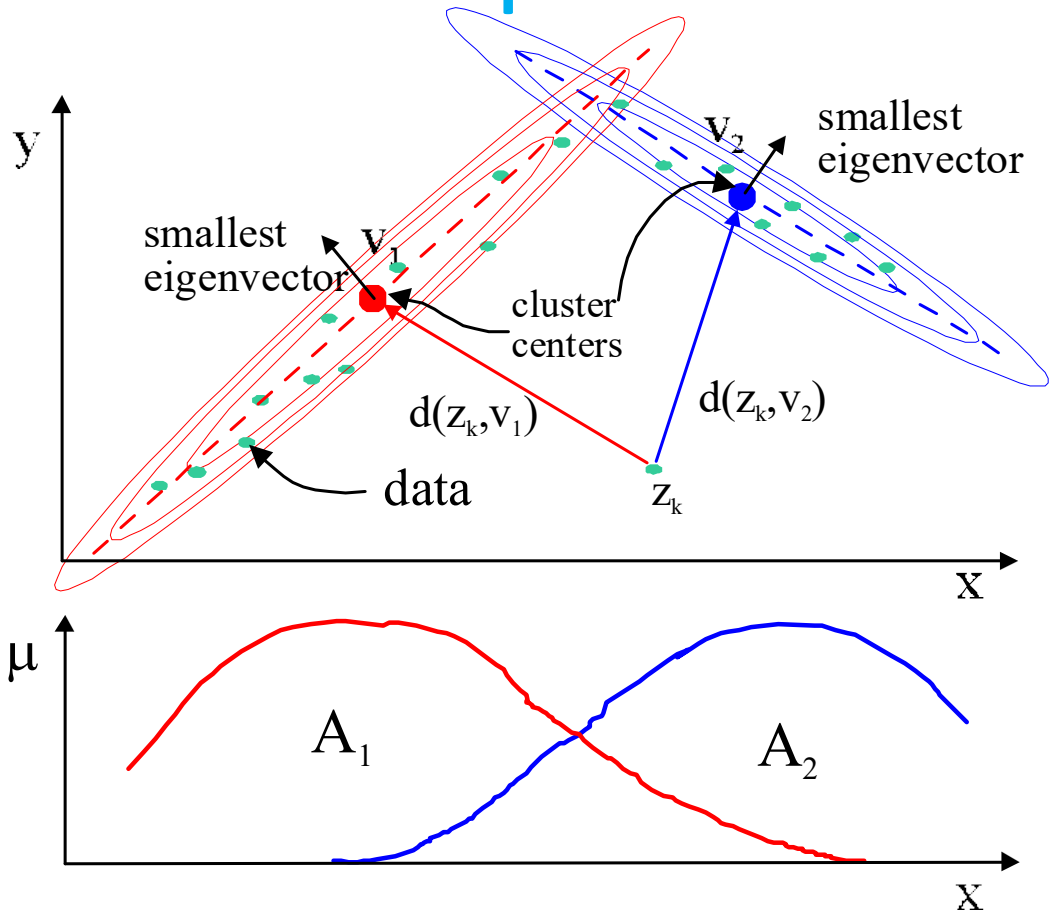
- **Until**

$\|\Delta \mathbf{U}\| < \varepsilon$ Other stopping criteria are possible

GK algorithm example



GK algorithm example



Mountain clustering

Algorithm:

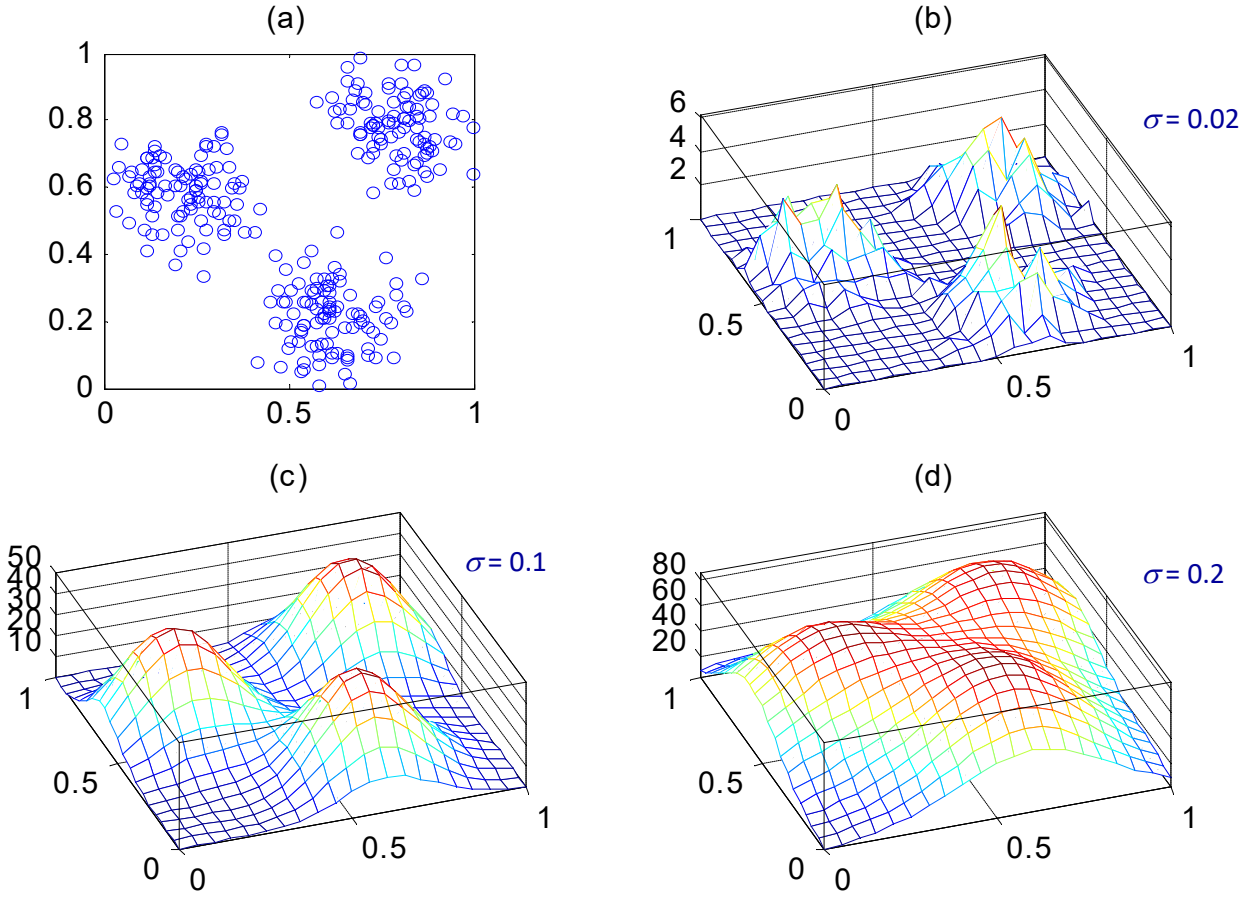
- Lay a grid (any type) on the data space
- Compute mountain functions (measure of data density)

$$f(\mathbf{v}_i) = \sum_{k=1}^{N} e^{\left(-\frac{d^2(\mathbf{x}_k - \mathbf{v}_i)}{2\sigma^2}\right)}$$

- Sequentially destroy mountain function after selecting grid point with highest value

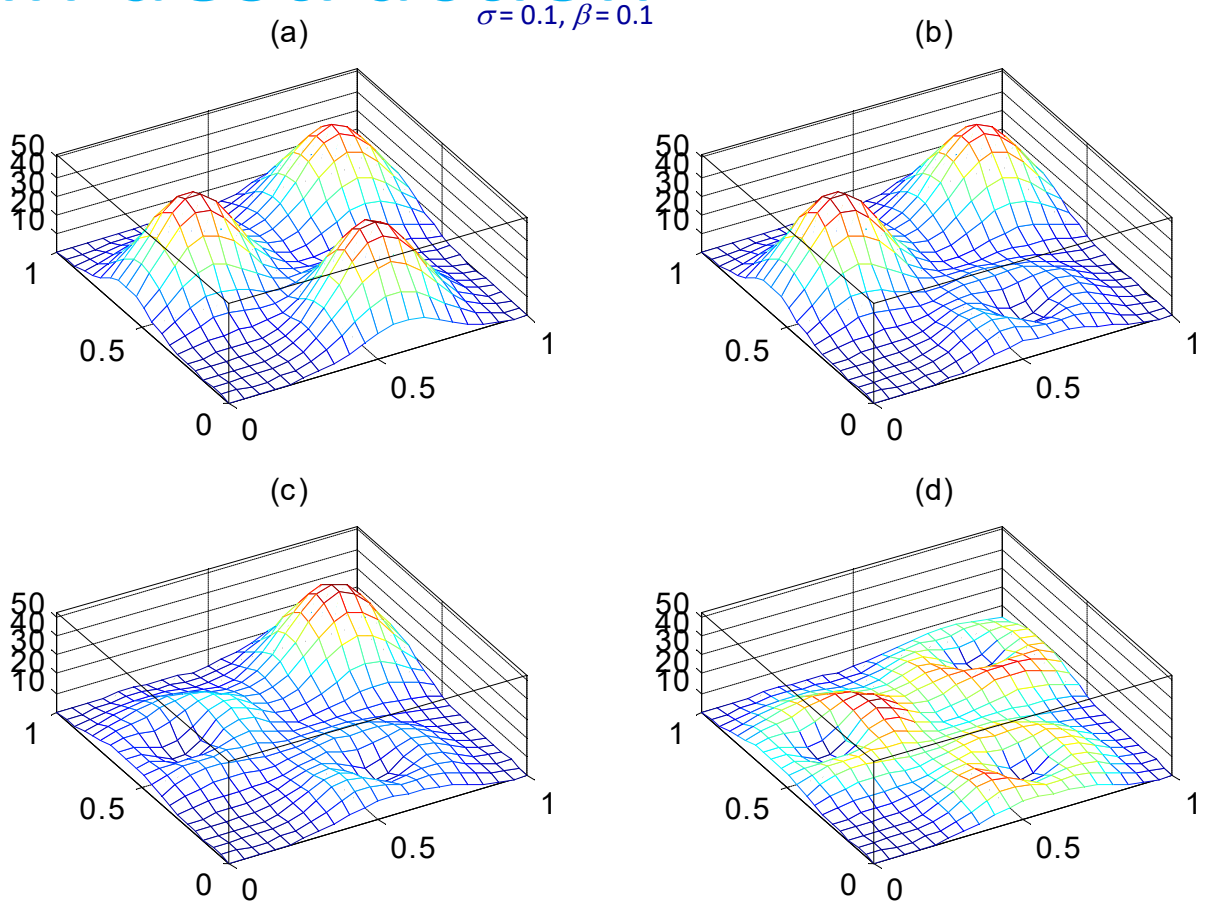
$$f^{(C)}(\mathbf{v}_i) = f(\mathbf{v}_i) - \sum_{j=1}^{C} f^{(j)}(\mathbf{v}_j) \quad e^{\left(-\frac{d^2(\mathbf{v}_i - \mathbf{v}_j)}{2\beta^2}\right)}$$

Mountain construction



Mountain destruction

$\sigma = 0.1$, $\beta = 0.1$



Subtractive clustering

- Mountain clustering with four variables and 10 grid lines results in 10^4 grid points to be evaluated.
- In **subtractive clustering**, the grid consists of the data points themselves.
- Computational complexity independent of the dimension of the data vectors

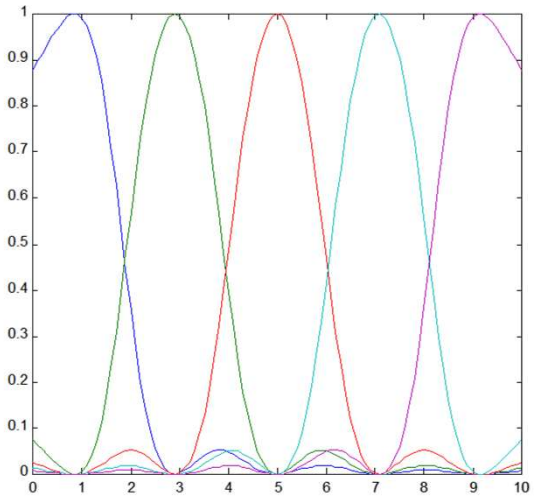
- ***Rule of thumb*: $\beta = 1.5\ \sigma$**

Effect of probabilistic constraint

Probabilistic constraint:

$$\sum_{i=1}^{C} \mu_{ik} = 1, \quad k = 1, \ldots, N$$

- Problematic if a data point lies far away from all clusters
  (e.g. outliers)
- Leads to nonconvex clusters



Possibilistic constraint:

$$\sum_{i=1}^{C} \mu_{ik} > 0, \quad k = 1, \ldots, N$$

Possibilistic clustering

- **Minimize objective function:**

$$J(\mathbf{X},\mathbf{U},\mathbf{V},\boldsymbol{\eta}) = \sum_{i=1}^{C}\sum_{k=1}^{N} \mu_{ik}^{m}\, d^2(\mathbf{x}_k,\mathbf{v}_i) + \sum_{i=1}^{C} \eta_i \sum_{k=1}^{N} (1-\mu_{ik})^m$$

- $m \in (1,\infty)$ is the fuzziness parameter
- η determine the size of the clusters
- suitable values from average inter-cluster distance

$$\eta_i = \frac{\sum_{k=1}^{N} \mu_{ik}^m d_{ik}^2}{\sum_{k=1}^{N} \mu_{ik}^m}$$

- ***The optimization problem can now be decomposed into C independent optimization problems.***

Possibilistic clustering algorithm

Repeat:

1. Compute cluster centers $$\mathbf{v}_i = \frac{\sum_{k=1}^{N} \mu_{ik}^m \mathbf{x}_k}{\sum_{k=1}^{N} \mu_{ik}^m}$$

2. Calculate distances $$d_{ik}^2 = (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{A} (\mathbf{x}_k - \mathbf{v}_i)$$

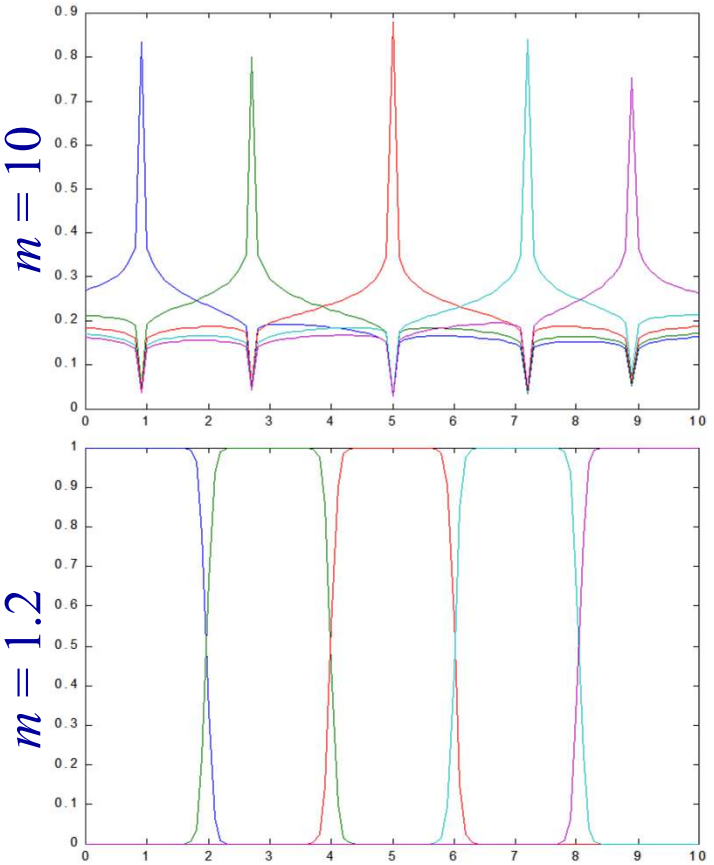
3. Update partition matrix $$\mu_{ik} = \frac{1}{1 + \left(\frac{d_{ik}^2}{\eta_i^2} \right)^{\frac{1}{m-1}}}$$

Membership value does not depend on the membership of other clusters

Until $$\|\Delta \mathbf{U}\| < \varepsilon$$

Effect of fuzziness index

- As m increases, clusters overlap more; their centres become more isolated
- As m decreases, clusters overlap less; they become crisp
- Often $m = 2$ is selected



Issues in fuzzy clustering

- Normalization
- Initialization
- Cluster volumes
- Number of clusters
- Cluster convexity
- Selection of parameters (e.g. fuzziness m)
- Categorical variables
- Outliers
- Missing values

Normalization

➢**How to compare measurements on different scales?**

- Data box normalization

$$x'_{jl} = \frac{x_{jl} - \min_j x_{jl}}{\max_j x_{jl} - \min_j x_{jl}}, \quad j = 1,\ldots,N \text{ and } l = 1,\ldots,n$$

- Standard deviation normalization

$$x'_{jl} = \frac{x_{jl} - \bar{x}_l}{\sigma_l}, j = 1,\ldots,N \text{ and } l = 1,\ldots,n$$

- ***Adaptive distance metrics as in Gustafson-Kessel clustering are less sensitive to normalization***

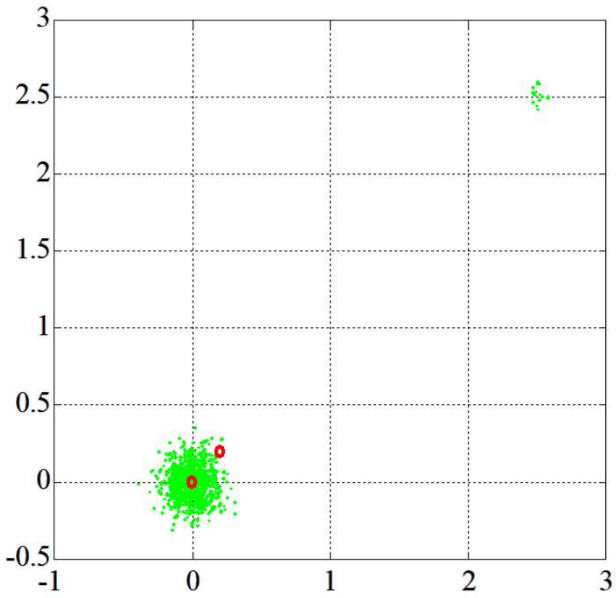
Initialization

How to avoid local minima during the optimization?

- Randomly select a set of cluster prototypes $\mathbf{V}$
- Randomly select a set of data points as cluster centers $\mathbf{V}$
- Randomly initialize the partition matrix $\mathbf{U}$
- Use information (e.g. cluster center locations) from a separate clustering step
- Initialize centers far away from data

Cluster volumes

FCM cluster centres



How large should clusters be?

- Extent of clusters
- Data density and distribution
- Size of cluster prototypes
- ***Cluster volume*** can be a parameter in Gustafson-Kessel clustering

Cluster validity

- How good are the clustering results?
  - Correct number of clusters?
  - Well-separated clusters?
  - Compact clusters?

- **Cluster validity measures** try to quantify the answers to these questions in a formula

- ***Optimal number of clusters*** at a local minimum of the validity measure

Validity measures

- **Gath and Geva index**

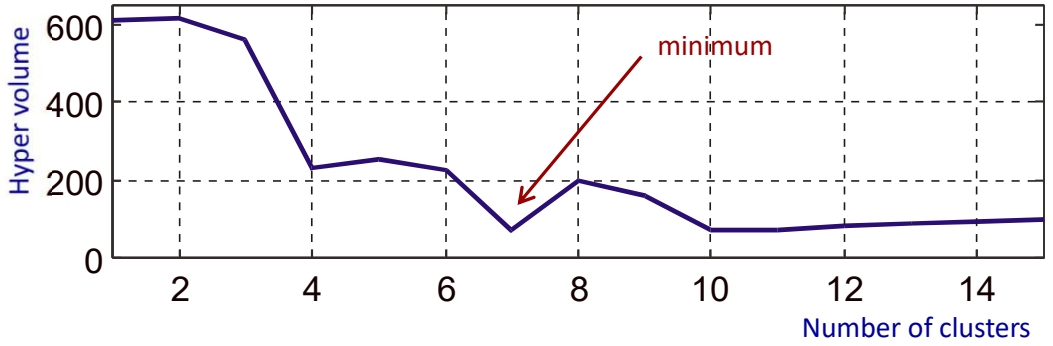
$$S_G = \sum_{i=1}^{C} \sqrt{\left| \frac{\sum_{k=1}^{N} (\mu_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^{N} (\mu_{ik})^m} \right|} + \beta C$$

- **Xie-Beni index**

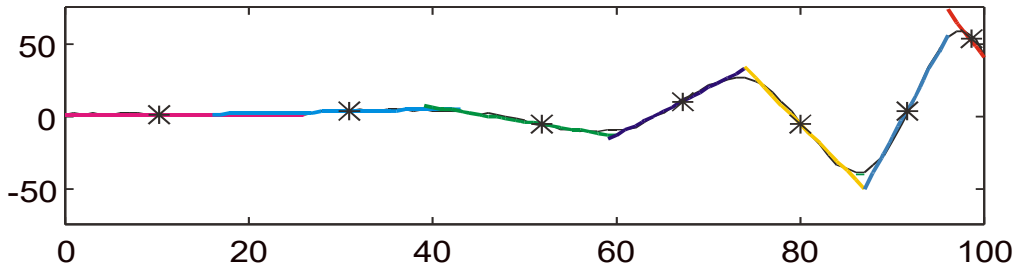
$$S_X = \frac{\sum_{i=1}^{C} \sum_{k=1}^{N} \mu_{ik}^m \, d^2(\mathbf{x}_k, \mathbf{v}_i)}{N \left(\min_{i,j,\, i \neq j} d^2(\mathbf{v}_i - \mathbf{v}_j) \right)}$$

Validity measures - example

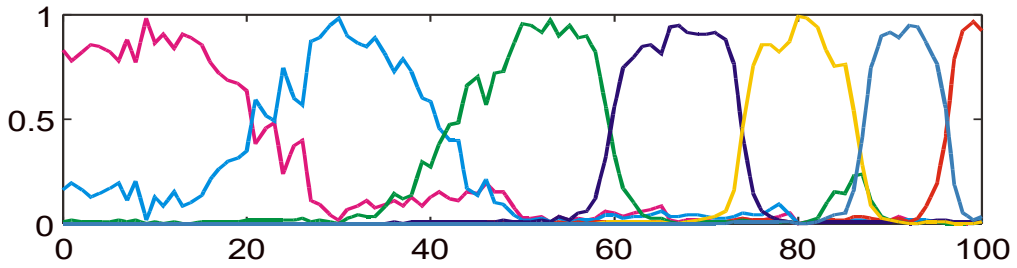
Validity
(Gath and Geva)



Local Models
(Gustafson-Kessel)



Clusters



CLUSTERING FOR IDENTIFICATION

Modeling based on clustering

1. Determine relevant input and output variables/features and collect data
2. Select model structure (Mamdani, Takagi-Sugeno,…)
3. Select number of clusters and clustering algorithm
4. Cluster the data
5. Obtain antecedent membership functions (MF)
6. Obtain consequents (MF or parameters)
7. Simplify the model, if necessary
8. Validate the model

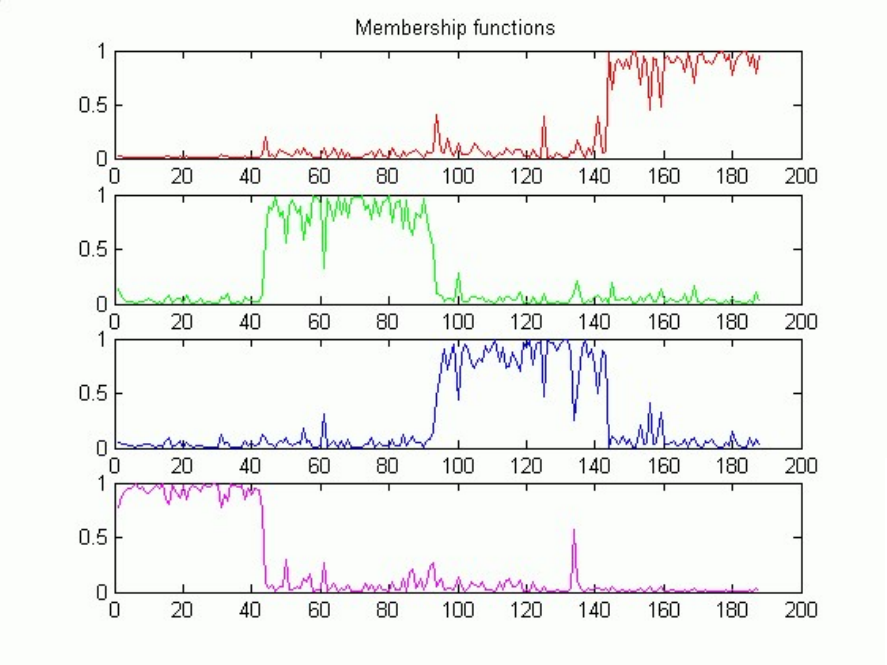
Linguistic models fr. clustering

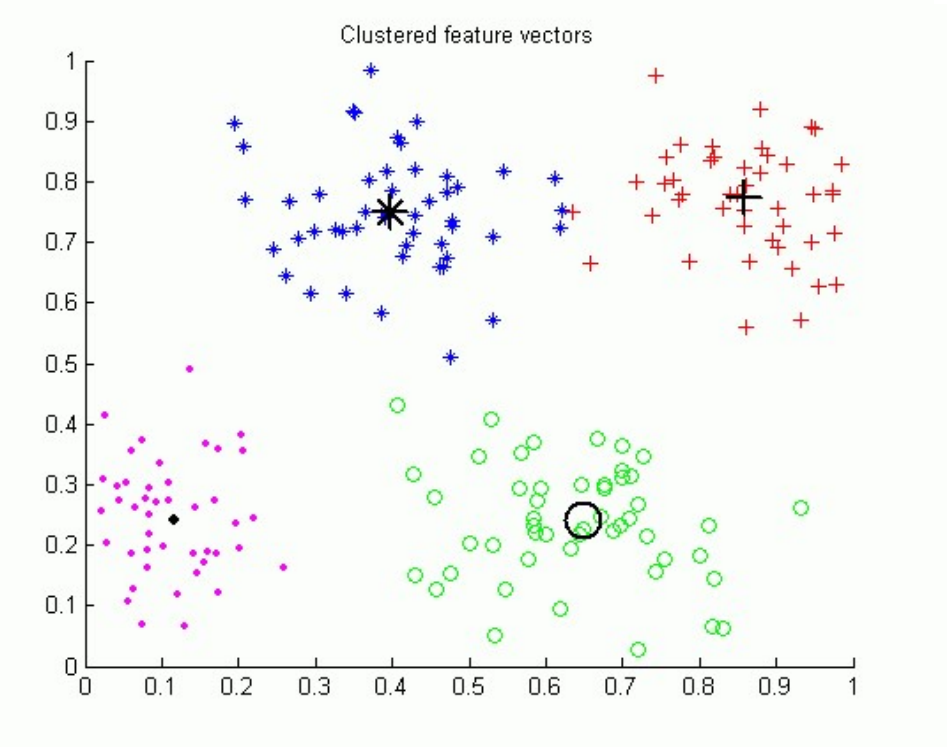
$$R_i: \quad \textbf{If } \mathbf{x} \text{ is } A_i \textbf{ then } \mathbf{y} \text{ is } B_i, \quad i = 1, 2, \ldots, K$$

- Use fuzzy c-means algorithm.
- Cluster data in input–output product space.
- Membership functions obtained by:
  - projection onto variables,
  - membership function parameterization.
- One rule per cluster

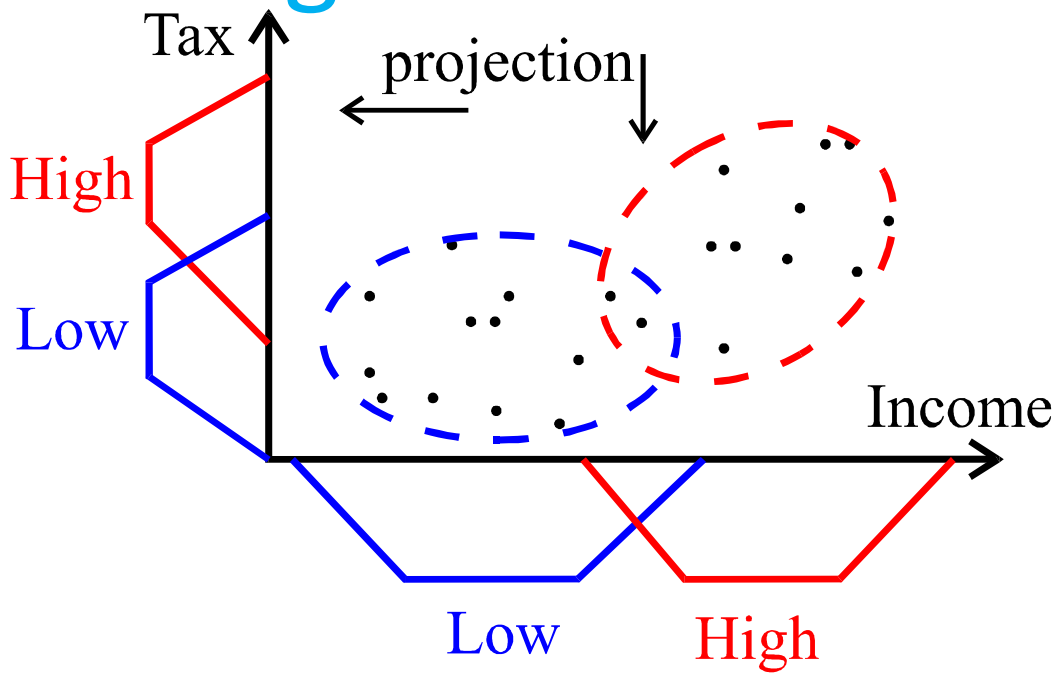
Membership functions

- After clustering, MFs are obtained by projections of partition matrix values.





Example of linguistic model



- **If** *income* is *Low* **then** *tax* is *Low*
- **If** *income* is *High* **then** *tax* is *High*

Data for clustering

- Inputs and output:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \qquad \mathbf{Z} = \begin{bmatrix} \mathbf{X} & \mathbf{y} \end{bmatrix}$$

- Example of data: **what is the structure of the model?**

$$\mathbf{Z} = \begin{bmatrix} y_1(2) & y_2(2) & y_2(1) & u_1(2) & y_1(3) \\ y_1(3) & y_2(3) & y_2(2) & u_1(3) & y_1(4) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ y_1(N-1) & y_2(N-1) & y_2(N-2) & u_1(N-1) & y_1(N) \end{bmatrix}$$

$$y_1(k+1) = f(y_1(k), y_2(k), y_2(k-1), u_1(k))$$

TS models from clustering

$$R_i: \quad \textbf{If } \mathbf{x} \text{ is } A_i \textbf{ then } y_i = f_i(\mathbf{x}), \quad i = 1, 2, \ldots, K$$

- ***Takagi-Sugeno order zero:*** $y_i = b_i$
- ***Takagi-Sugeno order one:*** $y_i = a_i^T \mathbf{x} + b_i$
- Degree of fulfillment $\beta_i \triangleq \mu_{A_i}(\mathbf{x})$
- Model output given by the ***weighted fuzzy-mean***:

$$y = \sum_{k=1}^{K} \gamma_k y_k, \text{with } \gamma_k = \frac{\beta_k}{\sum_{j=1}^{K} \beta_j} \quad (\text{normalized } \beta_k)$$

TS models from clustering

- Use fuzzy clustering algorithm (one rule per cluster)
- Cluster data in input–output product space
- Project clusters onto input variables and fit parametric membership functions to projected clusters
- Estimate consequent parameters

Matrix notation

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \qquad \mathbf{W}^k = \begin{bmatrix} \gamma_{k1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \gamma_{kN} \end{bmatrix}$$

- Consequent parameters:
  - TS order zero: $\theta_k = [b_k], k = 1, \ldots, K$
  - TS order one: $\theta_k = \begin{bmatrix} a_k^T & b_k \end{bmatrix}$
- Extended regression matrix

$$\mathbf{X}_e = [\mathbf{X} \quad \mathbf{1}]$$

Estimation of consequents

- **Global least squares:**

$$\mathbf{W} = \left[\mathbf{W}_1 \mathbf{X}_e \quad \mathbf{W}_2 \mathbf{X}_e \quad \cdots \quad \mathbf{W}_K \mathbf{X}_e \right]$$

$$\boldsymbol{\theta} = \left[\theta_1^T \quad \theta_2^T \quad \cdots \quad \theta_K^T \right]$$

- Resulting least squares problem: $\mathbf{y} = \mathbf{W}\boldsymbol{\theta}$
  - **Solution:**

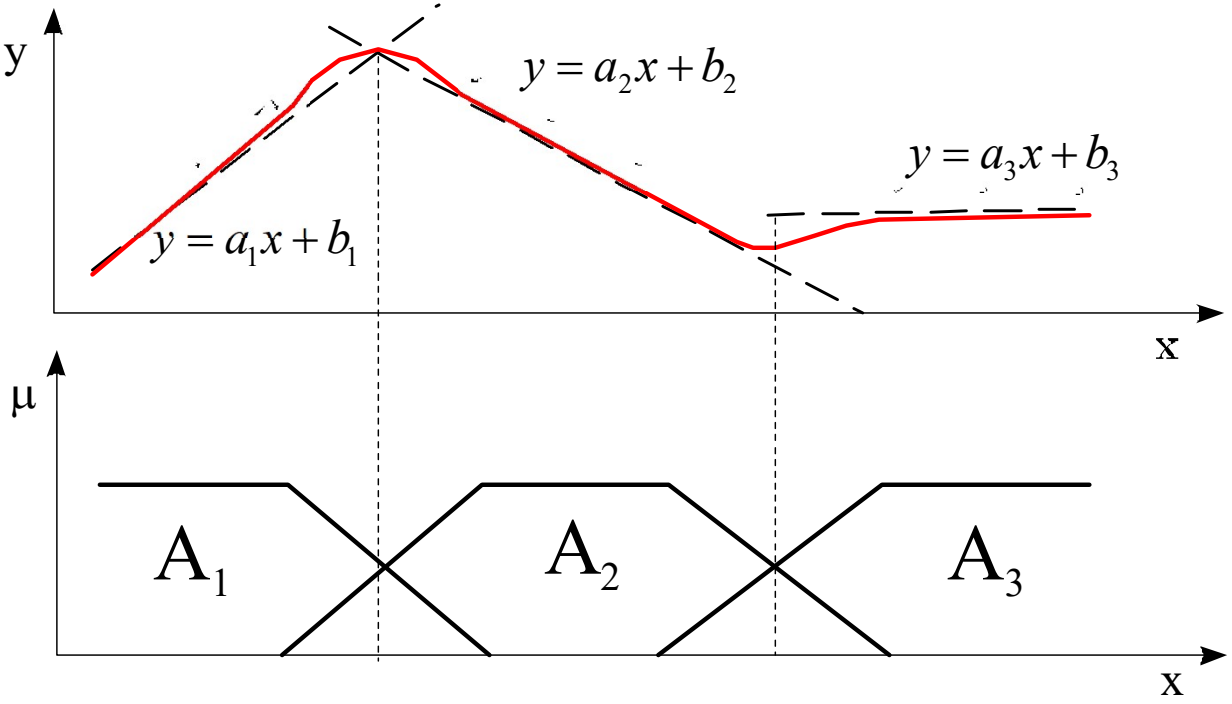
$$\boldsymbol{\theta} = \left(\mathbf{W}^T \mathbf{W} \right)^{-1} \mathbf{W}^T \mathbf{y}$$

- Solution of **local least squares:**

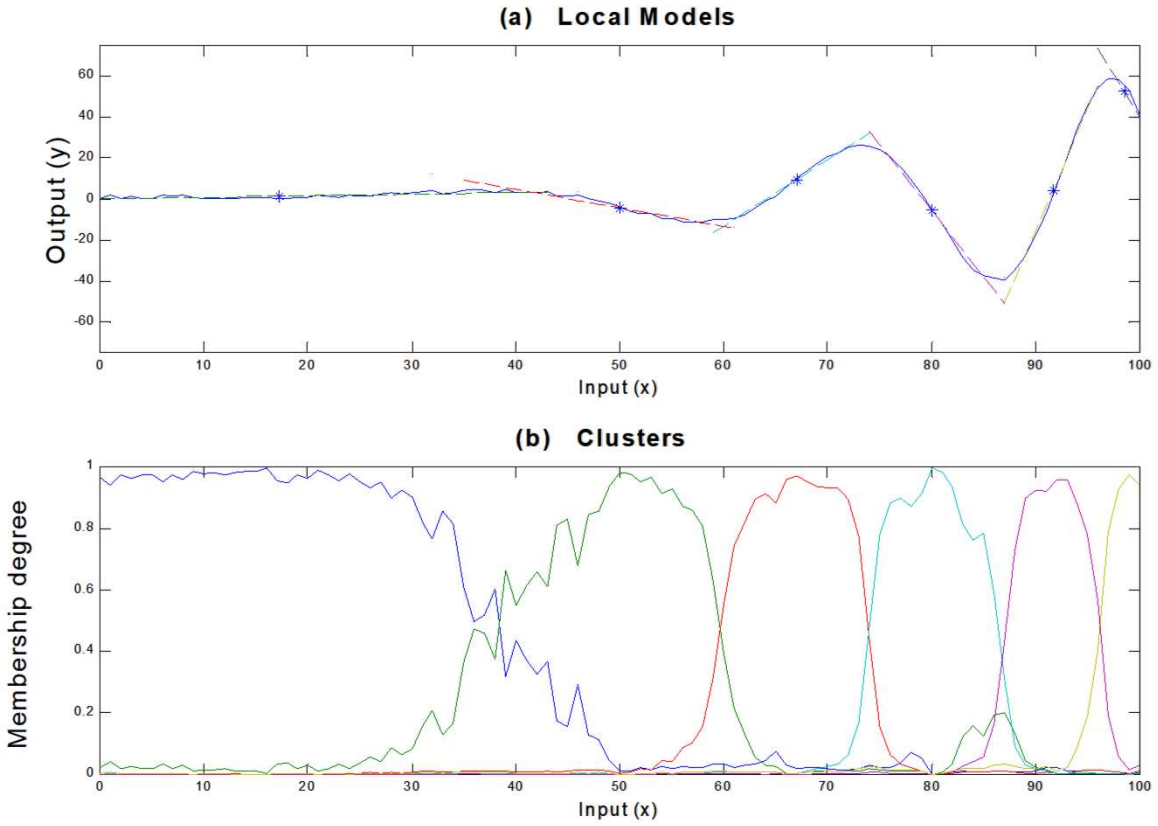
$$\theta_k = \left(\left(\mathbf{X}_e \right)^T \mathbf{W}_k \mathbf{X}_e \right)^{-1} \left(\mathbf{X}_e \right)^T \mathbf{W}_k \mathbf{y}$$

Example: TS order one

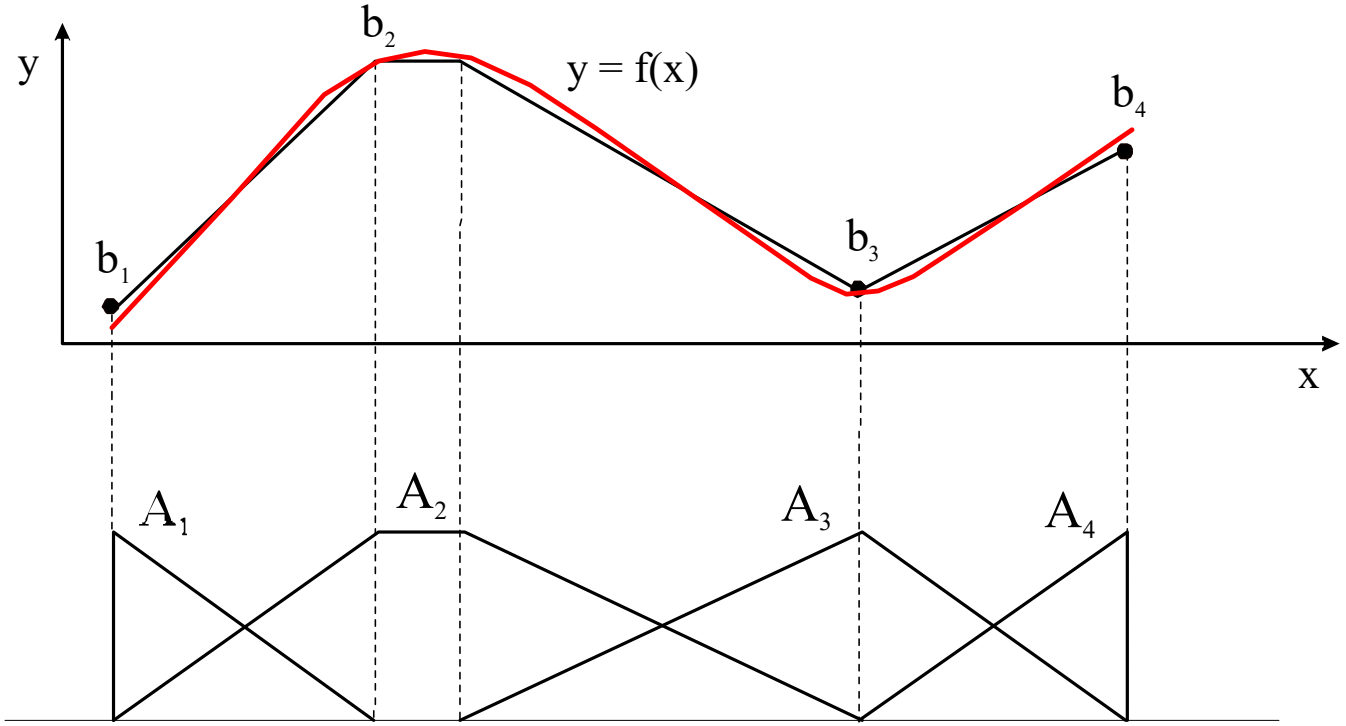
- Consequents can approximate local linear models of the system



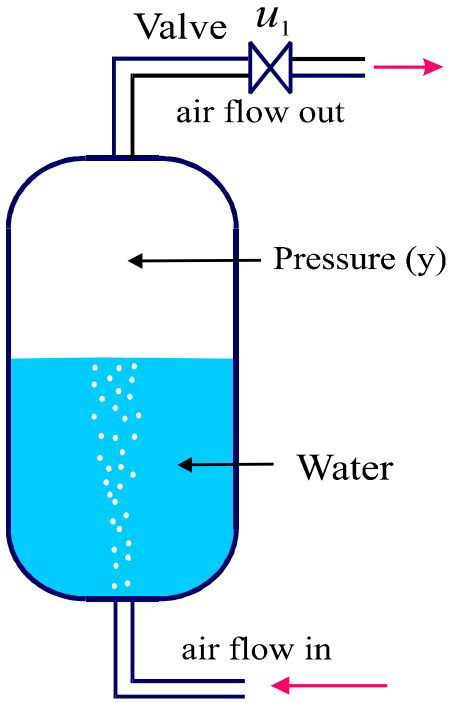
Example: TS model



Example: TS order zero



Example: pressure control

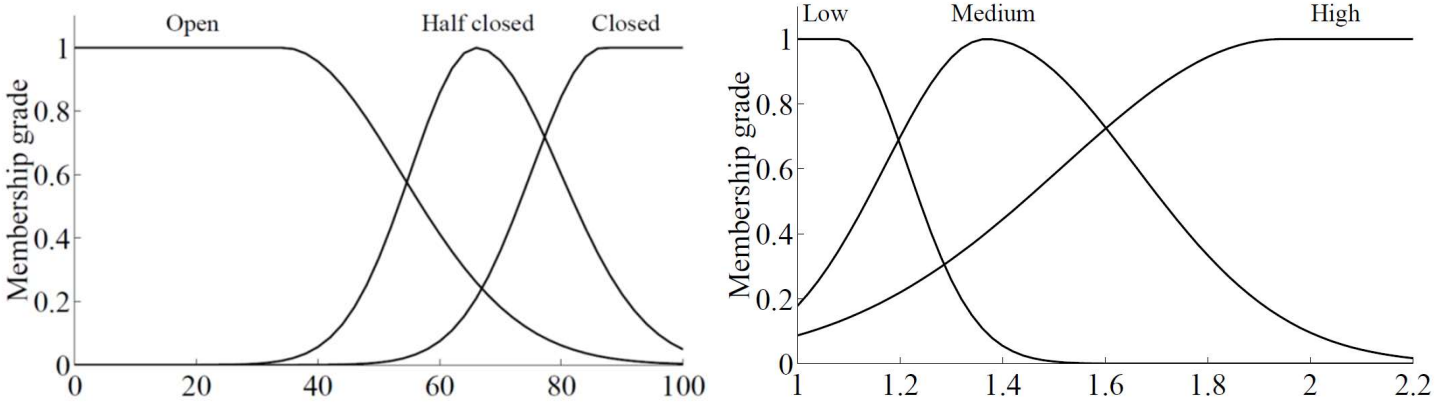


$$\frac{dP}{dt} = \frac{1000RT}{22.4V_h}\left[\Phi_g - (\pi R_H^2)\sqrt{\frac{2P_0}{\rho_0 K_f}\ln\left(\frac{P}{P_0}\right)}\right]$$

Fermenter: parameters

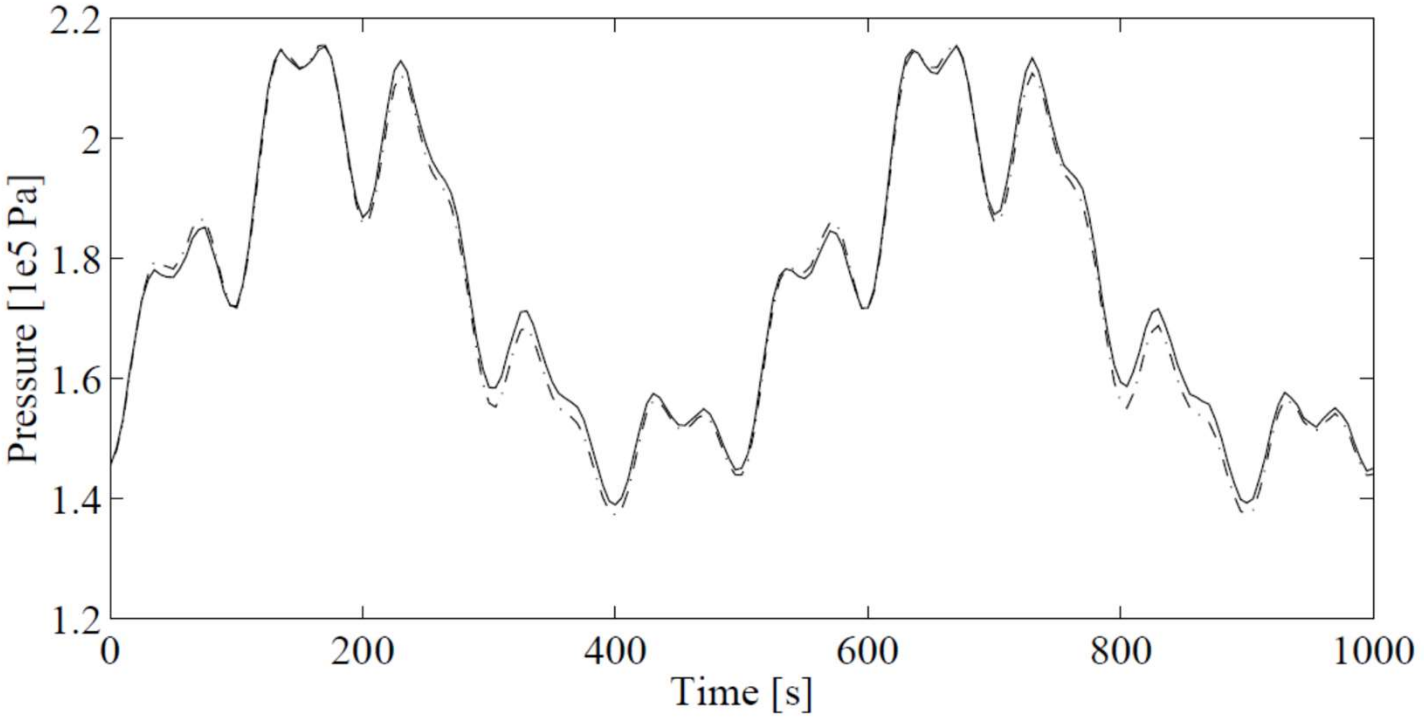
- R gas constant (8.134 J mol $^{-1}$ K^{-1})
- T temperature (305 K)
- V_h gas volume (0.015 m^3)
- Φ_g gas flow-rate (3.75×10 $^{-4}$ m^3 s^{-1})
- R_H radius of the outlet pipe (0.0178 m)
- P_0 reference pressure (1.013 ×10^5 N m^{-2})
- ρ_0 outside air density (1.2 Kg m^{-3})
- P pressure in the tank (N m^{-2})
- K_f valve friction factor (J mol^{-1})

Takagi-Sugeno fuzzy model



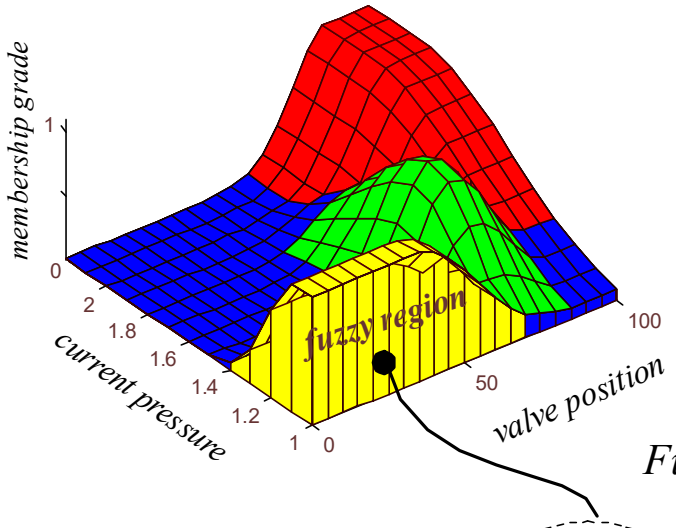
1. **If** $y(k)$ is LOW **and** $u(k)$ is OPEN
   then $y(k+1) = 0.67y(k) + 0.0007u(k) + 0.35$
2. **If** $y(k)$ is MEDIUM **and** $u(k)$ is HALF CLOSED
   then $y(k+1) = 0.80y(k) + 0.0028u(k) + 0.07$
3. **If** $y(k)$ is HIGH **and** $u(k)$ is CLOSED
   then $y(k+1) = 0.90y(k) + 0.0071u(k) - 0.39$

Validation

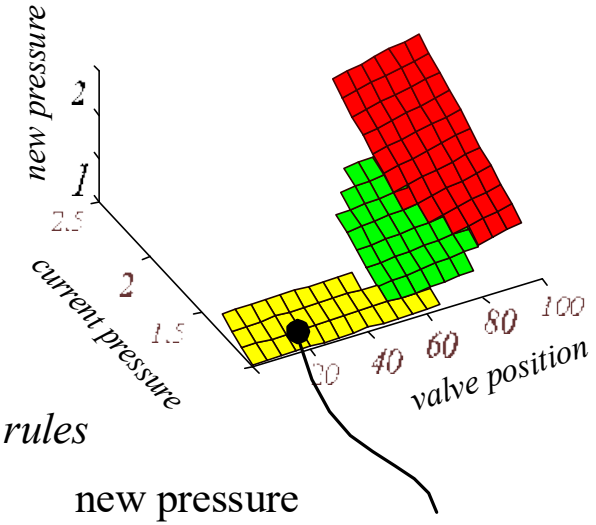


Example: 3-D representation

Fuzzy partition of the input-state product space



Corresponding local linear models



Fuzzy-linear rules

If current pressure $y(k)$ **is** ***LOW***
and valve $u(k)$ **is** ***OPEN***

then new pressure $y(k+1) = a_1 y(k) + b_1 u(k) + c_1$

Interpretability in fuzzy models

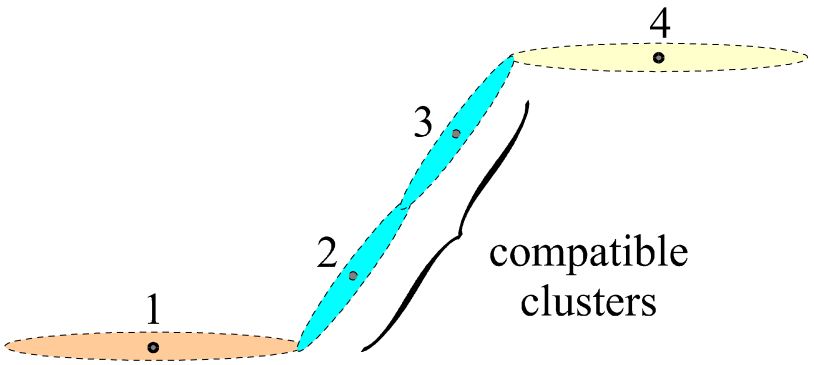
- **Interpretability** is not obtained automatically
  - Knowledge acquisition ensures **transparency**.
  - Based on data: some **redundancy** is unavoidable.
- **Redundancy** manifests itself in two ways:
  - High number of rules. Trade-off between:
    - model accuracy / model complexity
    - generalization capability / approximating data
  - Very similar fuzzy sets
    - similarity between fuzzy sets
    - similarity to the universal fuzzy set

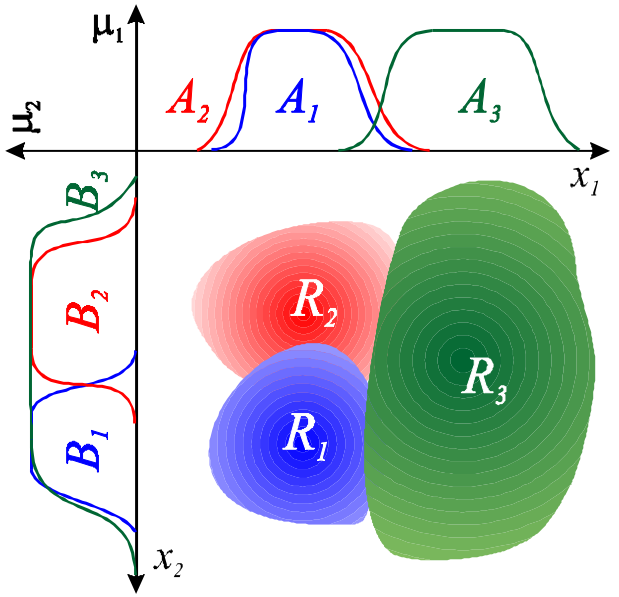
Redundancy in fuzzy models

- **High number of rules**
- **Overlapping similar fuzzy sets**
  - Unnecessary complexity
  - Difficult to assign linguistic labels
  - Less transparency and generality

Redundancy in fuzzy clustering

- Number of clusters
- Projection of clusters onto antecedent variables





Methods to solve redundancy

- **Cluster merging**
  - Method to determine the ‘best’ number of clusters
  - Merge clusters that are compatible
  - Cluster again and continue merging until there are no compatible clusters
- **Similarity based simplification**
  - Merge similar antecedent fuzzy sets
  - Remove sets similar to universal set
  - Combine / merge similar consequents
  - Combine rules with equal antecedents

Cluster merging

- Select number of clusters larger than needed and do clustering
- Merge clusters that are compatible
- Cluster again and continue merging until there are no compatible clusters
- Cluster compatibility measured by
  - Compatibility criteria
    How close are clusters?
    How similar are their characteristics? Etc.
  - Similarity measures

Compatible cluster merging

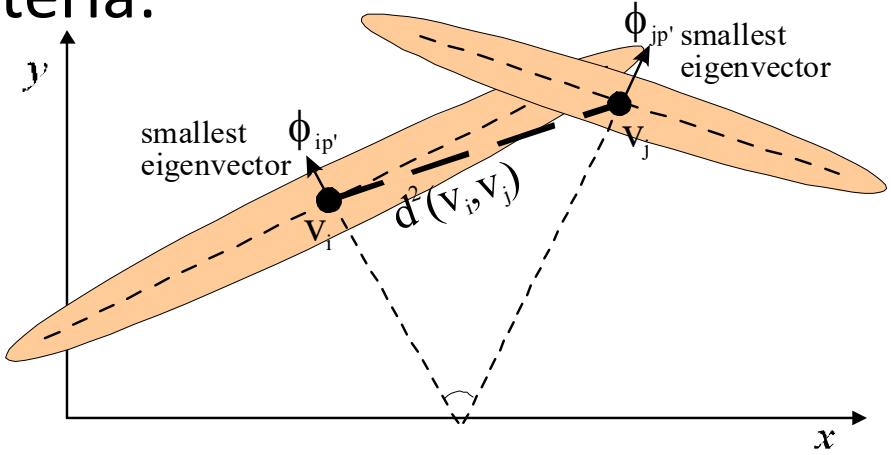
- Select *K* number of clusters larger than needed.

Repeat

1. Perform clustering
2. Evaluate compatibility criteria:

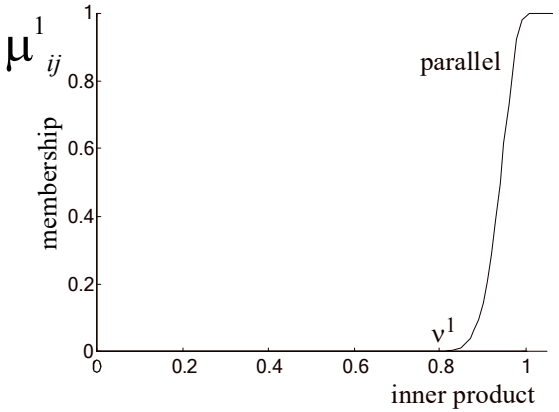
$$\left|\phi_{ip'} \cdot \phi_{jp'}\right| \geq k_1, \;\; k_1 \text{ close to } 1$$

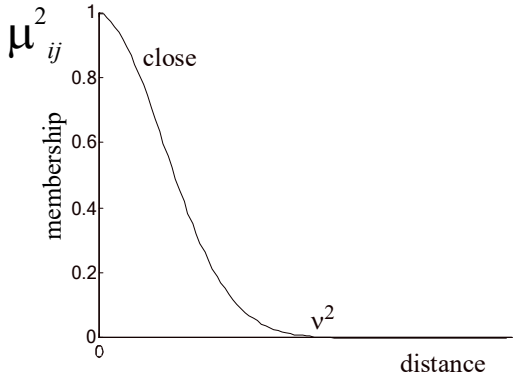
$$\left\|\mathbf{v}_i - \mathbf{v}_j\right\| \leq k_2, \;\; k_2 \text{ close to } 0$$



Compatible cluster merging

3. Compute parallelism and closeness μ_{ij}^1 and μ_{ij}^2





4. Compute compatibility s_{ij}

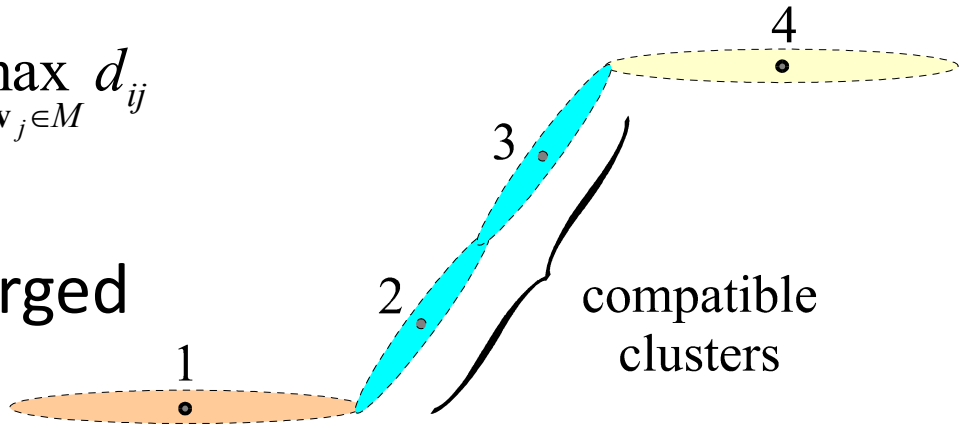
$$s_{ij} = \left(\frac{\left(\mu_{ij}^1\right)^\gamma + \left(\mu_{ij}^2\right)^\gamma}{2} \right)^{1/\gamma}$$

Compatible cluster merging

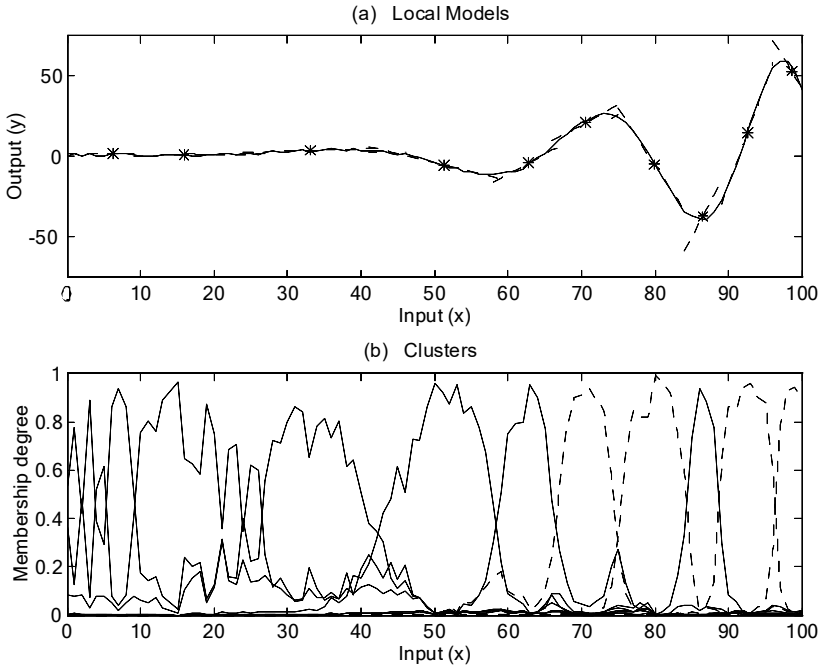
5. Compute transitive closure of compatibility matrix $\mathbf{S}$ and threshold with s^*.
6. Merge clusters if

$$\min \max_{\substack{\mathbf{v}_i \in M \\ \mathbf{v}_k \notin M}} d_{ik} > \max_{\mathbf{v}_i, \mathbf{v}_j \in M} d_{ij}$$

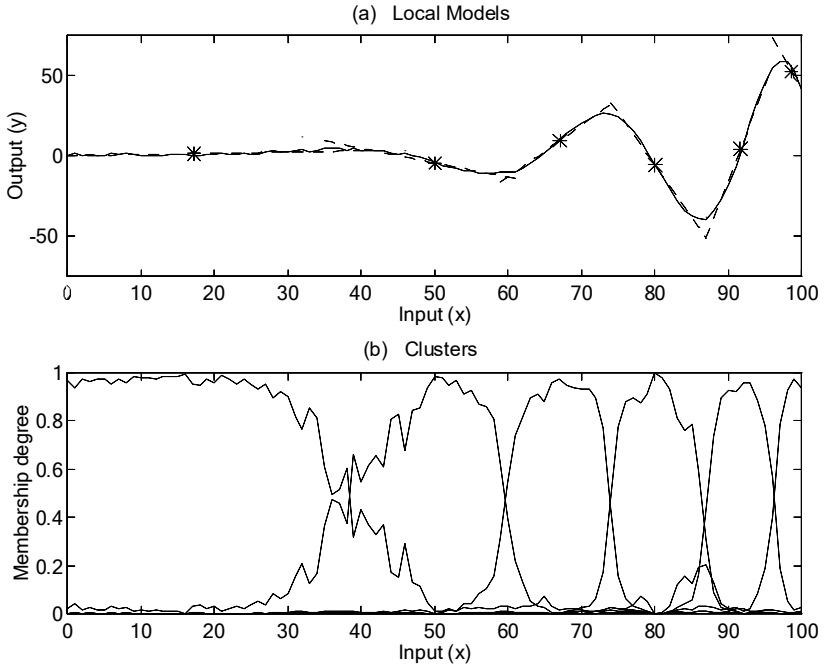
Until no clusters can be merged



Example

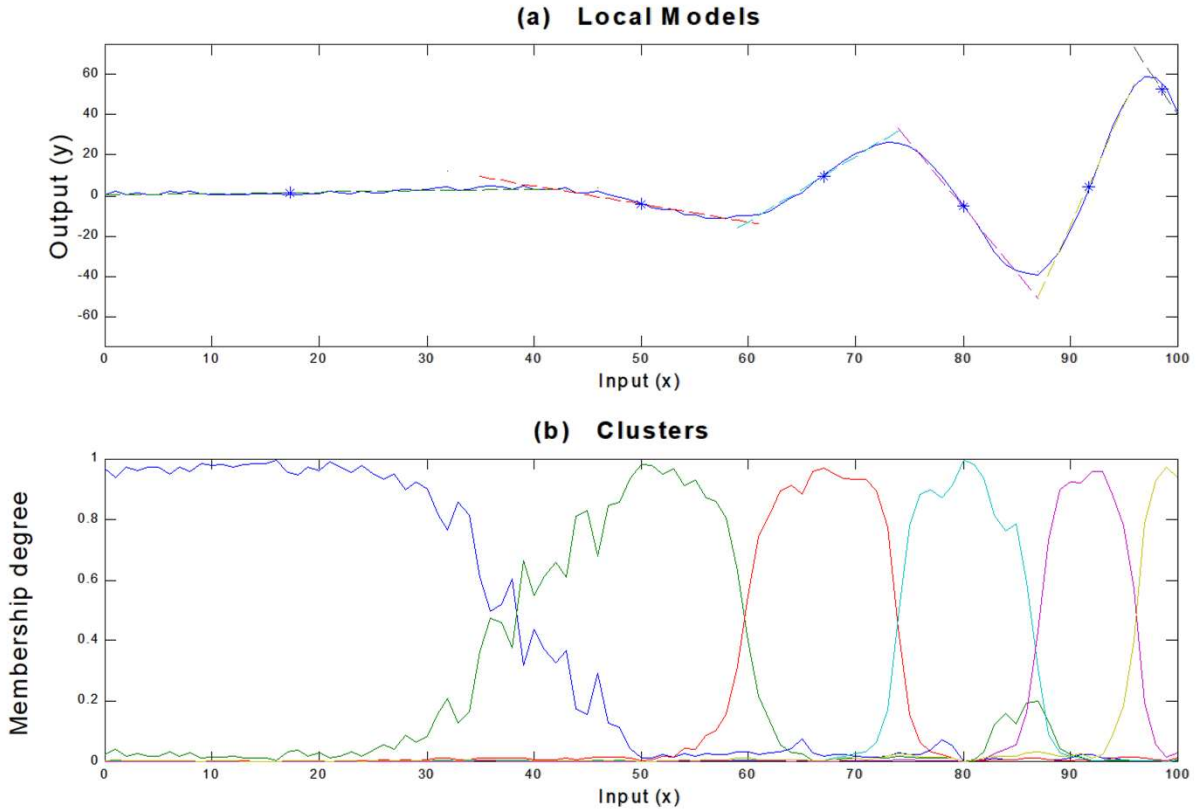


Initial (10 clusters)



After CCM (6 clusters)

Example


(a) Local Models
Output (y)
Input (x)
(b) Clusters
Membership degree
Input (x)

Similarity based simplification

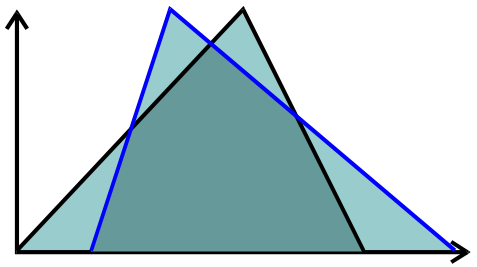
Reduce redundancy in rule and term set

- **Merge similar antecedent fuzzy sets**
  - create generalized concepts
  - reduce the number of terms
- **Remove sets similar to universal set** (always fires)
  - reduce number of terms
- **Combine/merge similar consequents**
  - reduce the number of consequent values
- **Combine rules with equal antecedents**
  - reduce number of rules

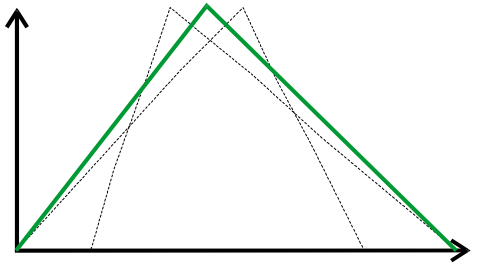
Similarity measures

- **Similarity measure:**

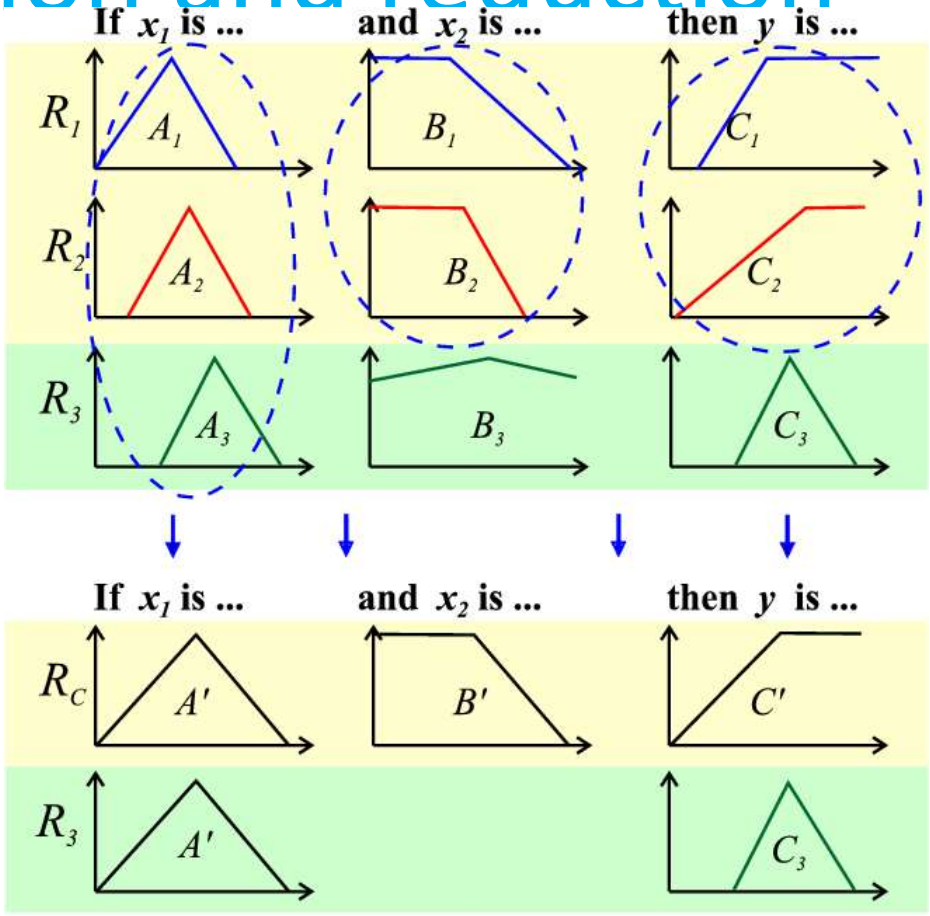
$$S(A,B) = \frac{|A \cap B|}{|A \cup B|}$$



- **Merging of similar sets**
  - Gives generalizing term
  - Merging by aggregating the parameters of the individual sets



Simplification and reduction



Example: algae growth

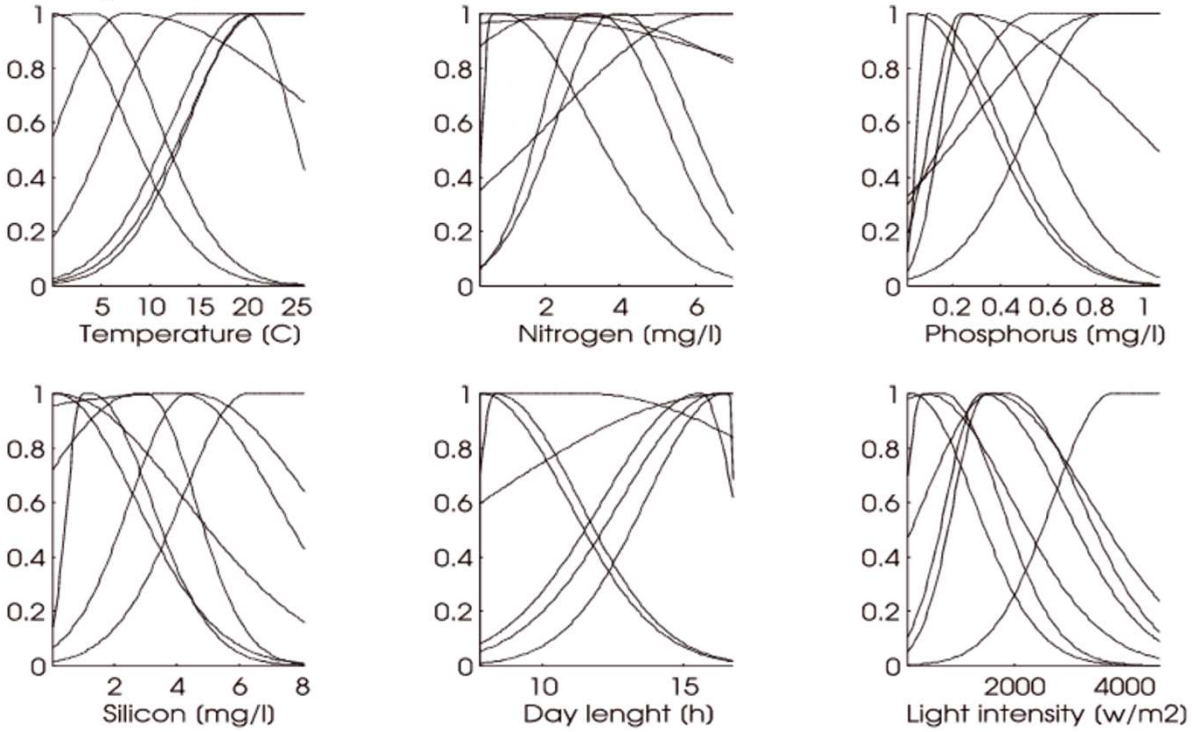
- Prediction of chlorophyll-a concentration in lake ecosystems.
- 998 observations from nine different lakes.
- **Inputs:** temperature, N,P,Si, day length, light intensity
- **Output:** chlorophyll-a concentration
- **Takagi-Sugeno model:**

  If T is A_T, N is A_N, P is A_P, Si is A_{Si}, D is A_D and I is A_I

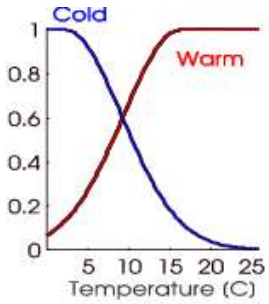
  then $\text{Chl} = p_0 + p_1 T + p_2 N + p_3 P + p_4 Si + p_5 D + p_6 I$
- **Method:** fuzzy clustering and similarity analysis

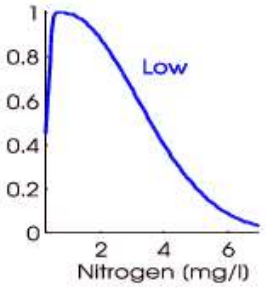
Initial rule base

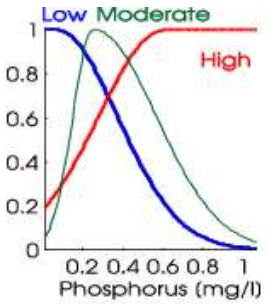
- Seven rules with a total of 42 fuzzy sets; Difficult to assign linguistic labels, inspection is virtually impossible

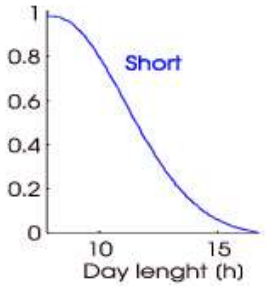


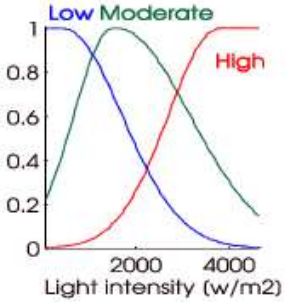
Simplified rule base











			Temp	N	P	Day	Light	
(Summer)	R_1	If	*Warm*	-	*High*	-	*Mod.*	Then ...
(Winter)	R_2	If	*Cold*	-	*Low*	*Short*	*Low*	Then ...
(Exception)	R_3	If	*Warm*	*Low*	*Low*	-	*High*	Then ...
(Summer)	R_4	If	*Warm*	-	*High*	-	*Low*	Then ...
(Winter)	R_5	If	*Cold*	-	*Mod.*	*Short*	*Low*	Then ...

TÉCNICO LISBOA