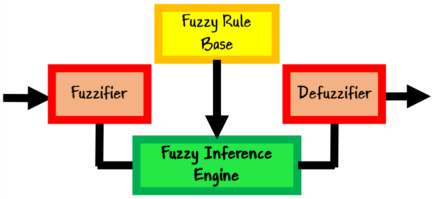
smgrc

Intelligent Control



On Design of Fuzzy Logic Systems

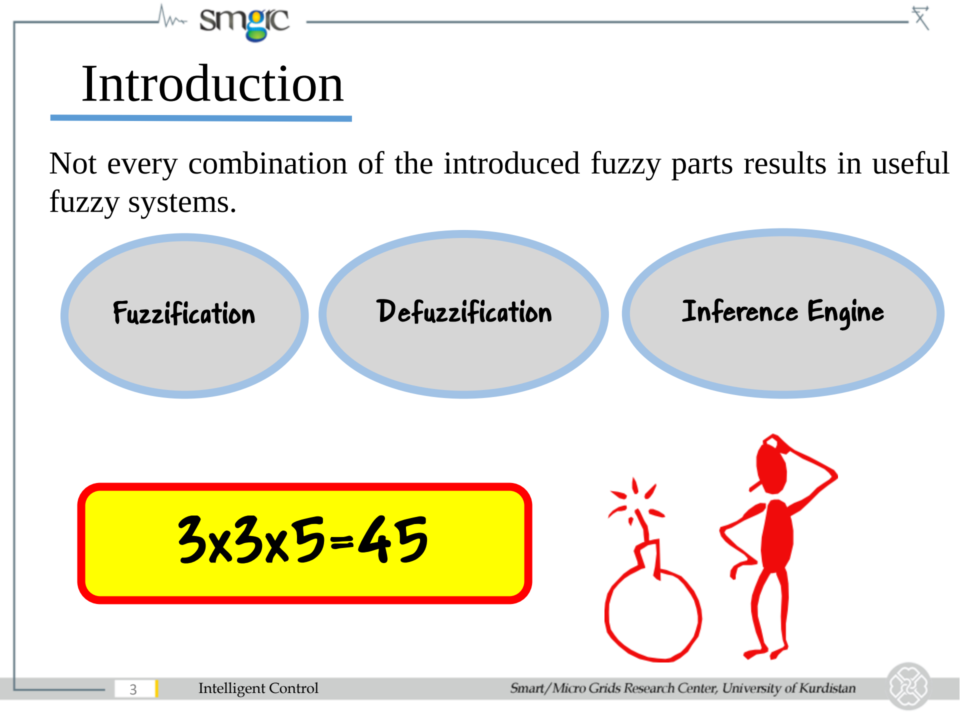
By:
Barmak Baigzadehnoe
b.baigzadeh@uok.ac.ir

Smart/Micro Grids Research Center, University of Kurdistan

smgrc

Content

- ❖ Some Classes of Fuzzy Systems
- ❖ Introduction to Design Fuzzy Systems
- ❖ Look-up Table
- ❖ Gradient Descent Training
- ❖ Recursive Least Squares



Introduction

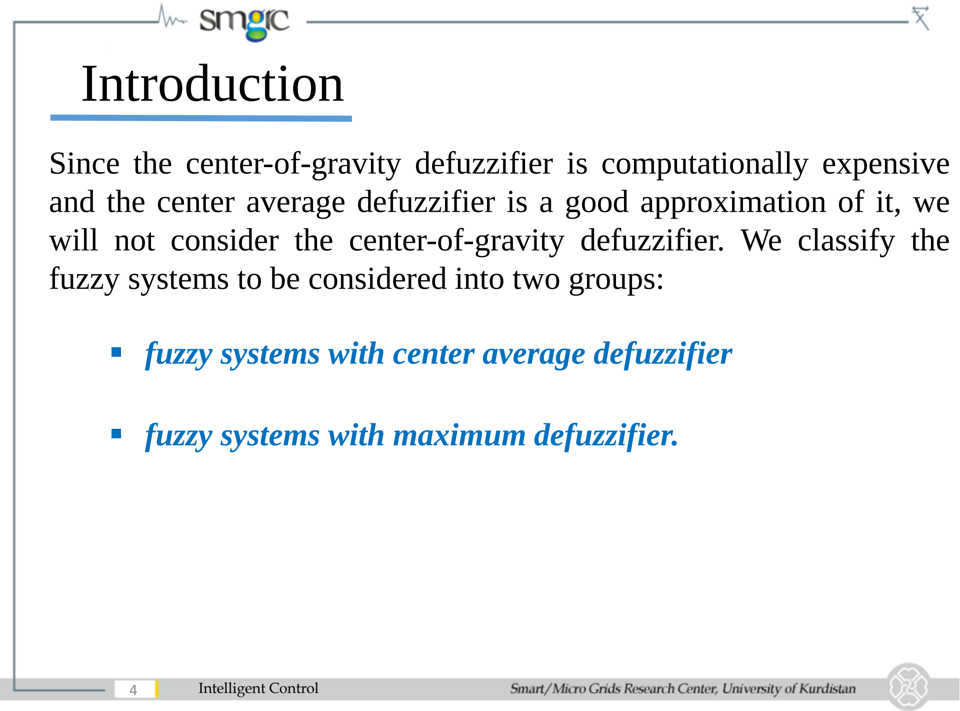
Not every combination of the introduced fuzzy parts results in useful fuzzy systems.

Fuzzification

Defuzzification

Inference Engine

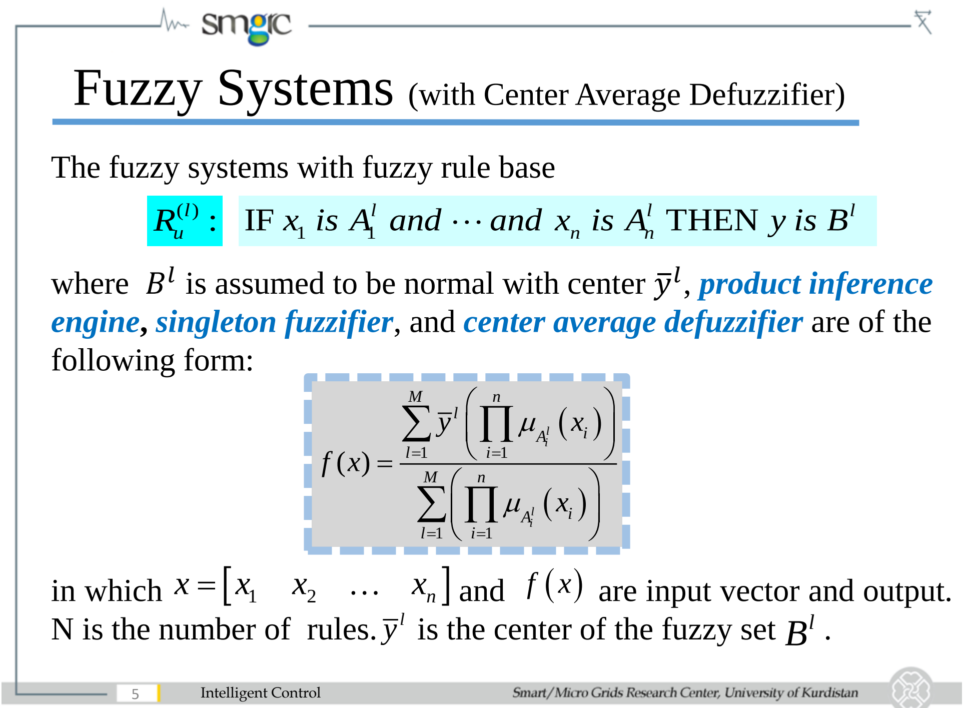
3x3x5=45



Introduction

Since the center-of-gravity defuzzifier is computationally expensive and the center average defuzzifier is a good approximation of it, we will not consider the center-of-gravity defuzzifier. We classify the fuzzy systems to be considered into two groups:

- *fuzzy systems with center average defuzzifier*
- *fuzzy systems with maximum defuzzifier.*



Fuzzy Systems (with Center Average Defuzzifier)

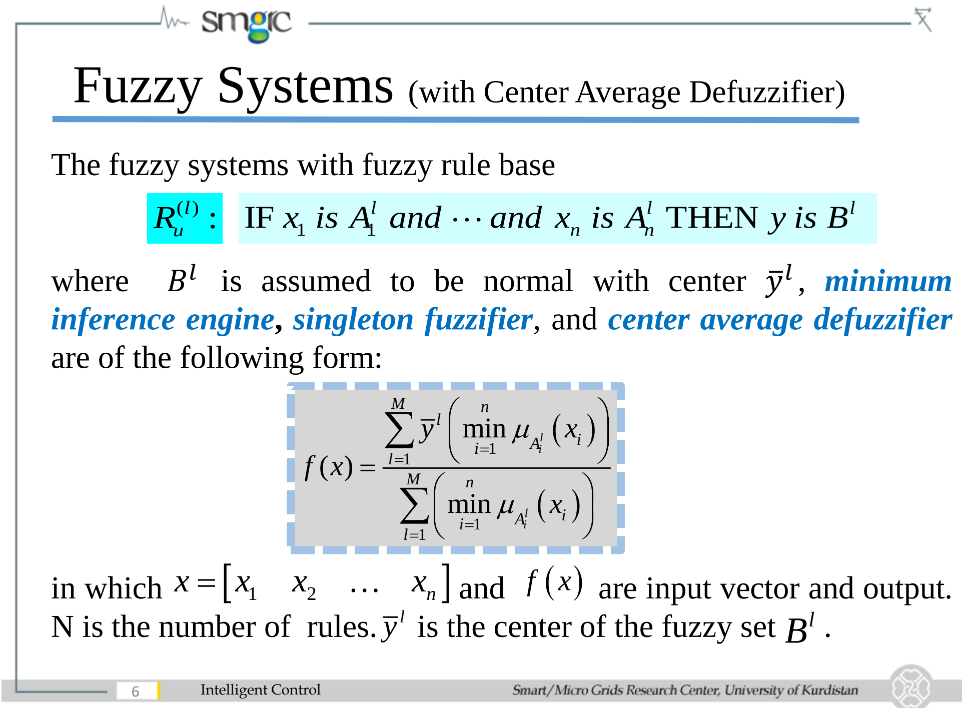
The fuzzy systems with fuzzy rule base

$$R_u^{(l)}: \quad \text{IF } x_1 \textit{ is } A_1^l \textit{ and } \cdots \textit{ and } x_n \textit{ is } A_n^l \text{ THEN } y \textit{ is } B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***product inference engine***, ***singleton fuzzifier***, and ***center average defuzzifier*** are of the following form:

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}{\sum_{l=1}^{M} \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}$$

in which $x = [x_1 \quad x_2 \quad \dots \quad x_n]$ and $f(x)$ are input vector and output. N is the number of rules. $\bar{y}^l$ is the center of the fuzzy set B^l .



Fuzzy Systems (with Center Average Defuzzifier)

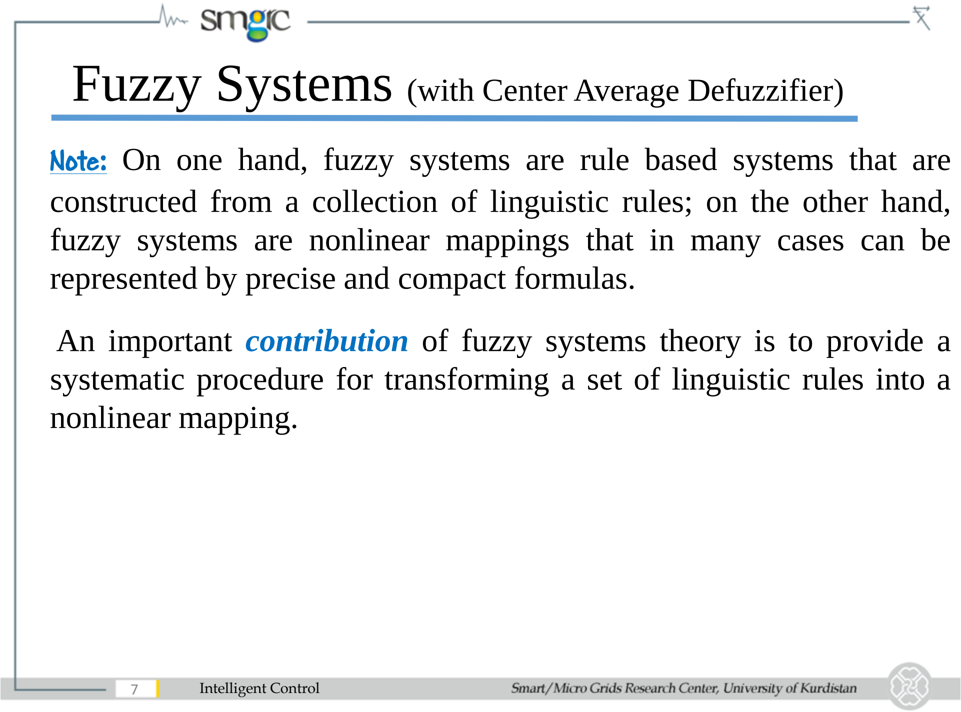
The fuzzy systems with fuzzy rule base

$$R_u^{(l)}: \quad \text{IF } x_1 \textit{ is } A_1^l \textit{ and } \cdots \textit{ and } x_n \textit{ is } A_n^l \text{ THEN } y \textit{ is } B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***minimum inference engine***, ***singleton fuzzifier***, and ***center average defuzzifier*** are of the following form:

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left(\min_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}{\sum_{l=1}^{M} \left(\min_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}$$

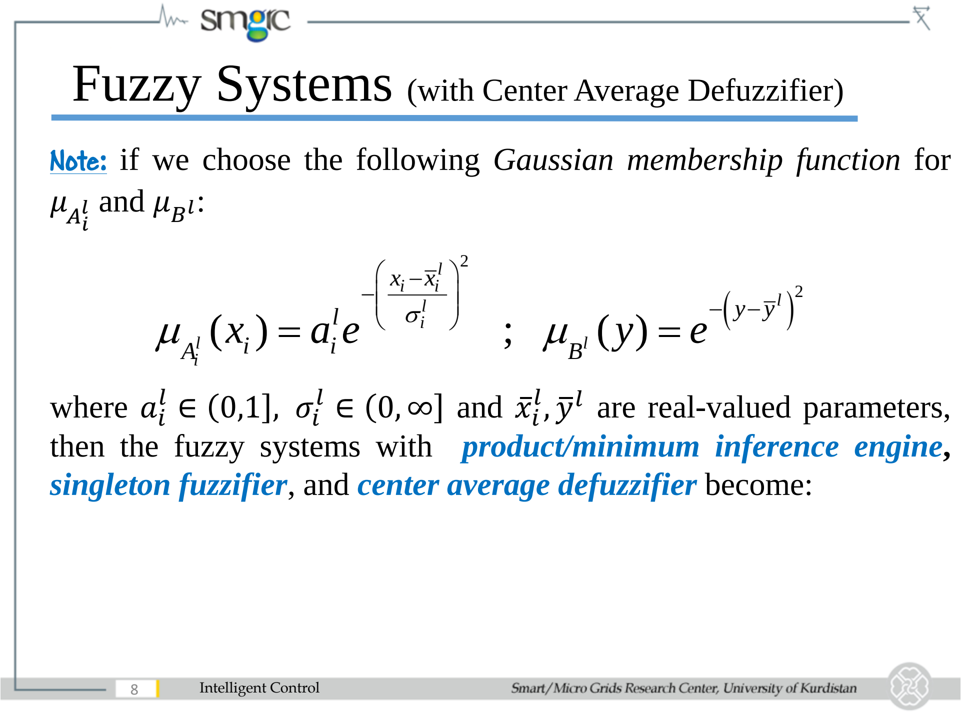
in which $x = [x_1 \quad x_2 \quad \dots \quad x_n]$ and $f(x)$ are input vector and output. N is the number of rules. $\bar{y}^l$ is the center of the fuzzy set B^l .



Fuzzy Systems (with Center Average Defuzzifier)

Note: On one hand, fuzzy systems are rule based systems that are constructed from a collection of linguistic rules; on the other hand, fuzzy systems are nonlinear mappings that in many cases can be represented by precise and compact formulas.

An important ***contribution*** of fuzzy systems theory is to provide a systematic procedure for transforming a set of linguistic rules into a nonlinear mapping.



Fuzzy Systems (with Center Average Defuzzifier)

Note: if we choose the following *Gaussian membership function* for $\mu_{A_i^l}$ and μ_{B^l}:

$$\mu_{A_i^l}(x_i) = a_i^l e^{-\left(\frac{x_i-\bar{x}_i^l}{\sigma_i^l}\right)^2} \quad ; \quad \mu_{B^l}(y) = e^{-\left(y-\bar{y}^l\right)^2}$$

where $a_i^l \in (0,1]$, $\sigma_i^l \in (0,\infty]$ and $\bar{x}_i^l, \bar{y}^l$ are real-valued parameters, then the fuzzy systems with ***product/minimum inference engine***, ***singleton fuzzifier***, and ***center average defuzzifier*** become:

Fuzzy Systems (with Center Average Defuzzifier)

Note:

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^{l}\left(\prod_{i=1}^{n}\mu_{A_i^l}(x_i)\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n}\mu_{A_i^l}(x_i)\right)}$$

Product inference engine

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^{l}\left(\prod_{i=1}^{n}a_i^l e^{-\left(\frac{x_i-\bar{x}_i^l}{\sigma_i^l}\right)^2}\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n}a_i^l e^{-\left(\frac{x_i-\bar{x}_i^l}{\sigma_i^l}\right)^2}\right)}$$

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^{l}\left(\min_{i=1}^{n}\mu_{A_i^l}(x_i)\right)}{\sum_{l=1}^{M}\left(\min_{i=1}^{n}\mu_{A_i^l}(x_i)\right)}$$

Minimum inference engine

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^{l}\left(\min_{i=1}^{n}a_i^l e^{-\left(\frac{x_i-\bar{x}_i^l}{\sigma_i^l}\right)^2}\right)}{\sum_{l=1}^{M}\left(\min_{i=1}^{n}a_i^l e^{-\left(\frac{x_i-\bar{x}_i^l}{\sigma_i^l}\right)^2}\right)}$$

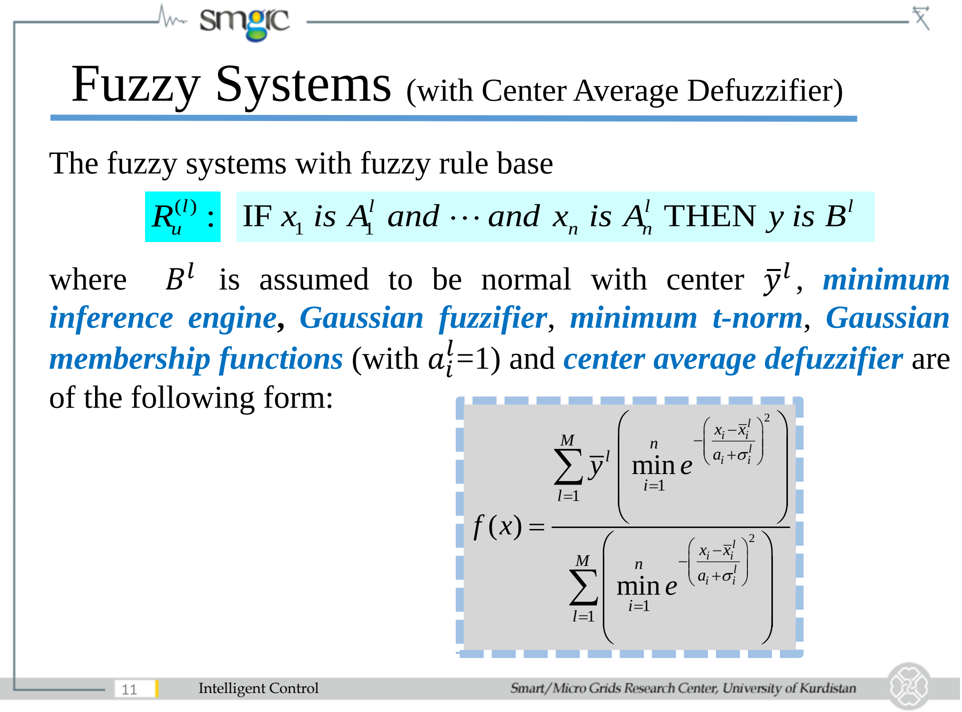
Fuzzy Systems (with Center Average Defuzzifier)

The fuzzy systems with fuzzy rule base

$$R_u^{(l)}:\ \text{IF } x_1 \textit{ is } A_1^l \textit{ and } \cdots \textit{ and } x_n \textit{ is } A_n^l \text{ THEN } y \textit{ is } B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***product inference engine***, ***Gaussian fuzzifier***, ***product t-norm***, ***Gaussian membership functions*** (with a_i^l=1) and ***center average defuzzifier*** are of the following form:

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^{l}\left(\prod_{i=1}^{n}e^{-\left(\frac{x_i-\bar{x}_i^l}{a_i+\sigma_i^l}\right)^2}\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n}e^{-\left(\frac{x_i-\bar{x}_i^l}{a_i+\sigma_i^l}\right)^2}\right)}$$



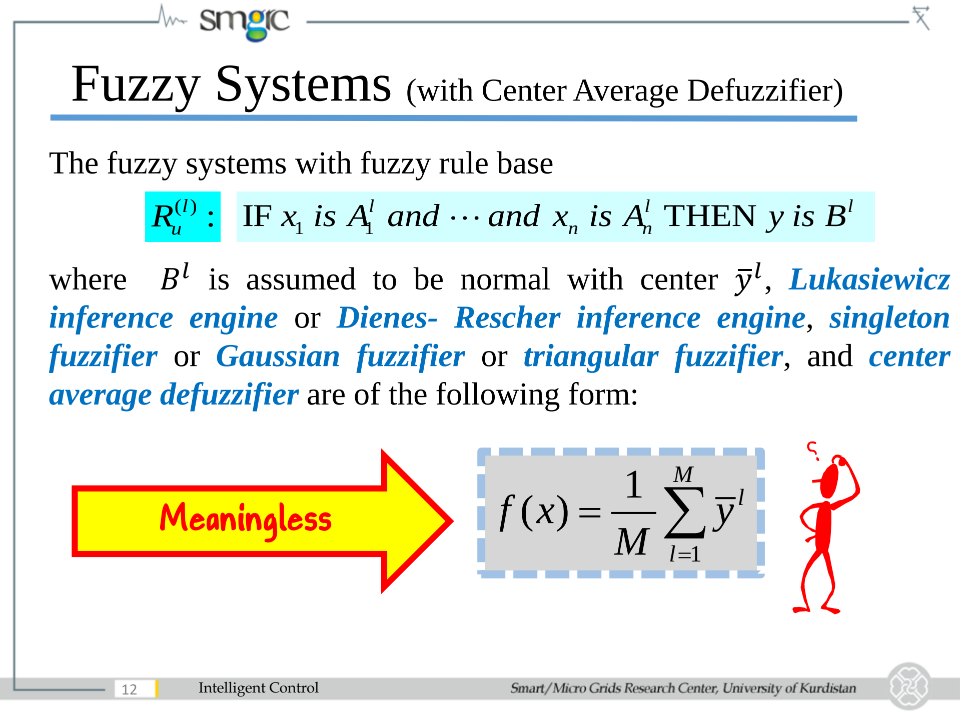
Fuzzy Systems (with Center Average Defuzzifier)

The fuzzy systems with fuzzy rule base

$$R_u^{(l)}: \quad \text{IF } x_1 \ is \ A_1^l \ and \cdots and \ x_n \ is \ A_n^l \ \text{THEN } y \ is \ B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***minimum inference engine***, ***Gaussian fuzzifier***, ***minimum t-norm***, ***Gaussian membership functions*** (with a_i^l=1) and ***center average defuzzifier*** are of the following form:

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left(\min_{i=1}^{n} e^{-\left(\frac{x_i - \bar{x}_i^l}{a_i + \sigma_i^l}\right)^2} \right)}{\sum_{l=1}^{M} \left(\min_{i=1}^{n} e^{-\left(\frac{x_i - \bar{x}_i^l}{a_i + \sigma_i^l}\right)^2} \right)}$$



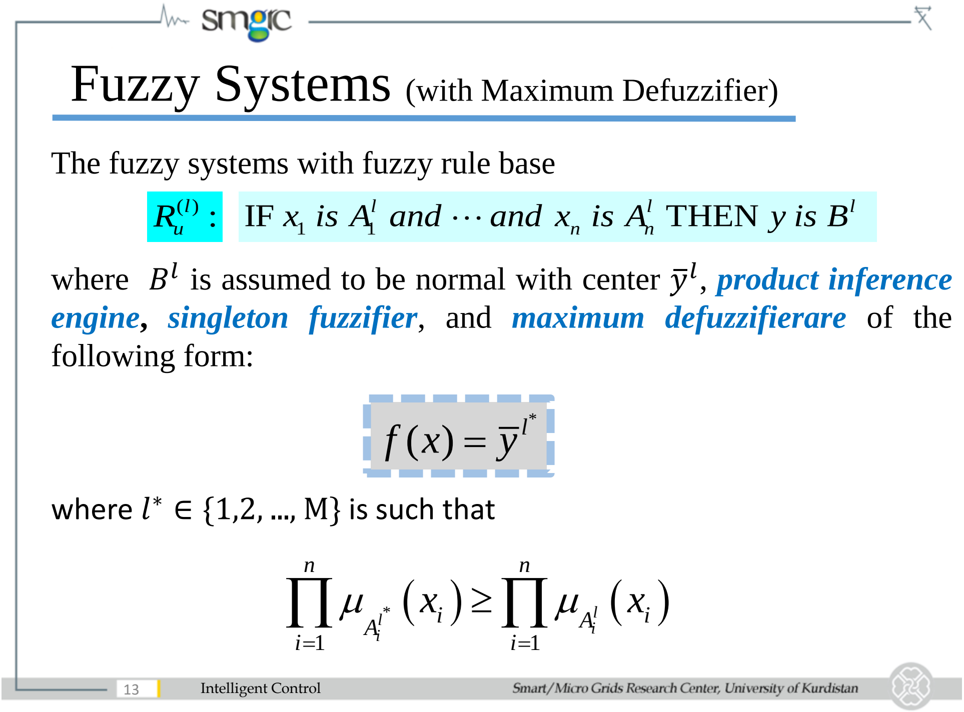
Fuzzy Systems (with Center Average Defuzzifier)

The fuzzy systems with fuzzy rule base

$$R_u^{(l)}: \quad \text{IF } x_1 \ is \ A_1^l \ and \cdots and \ x_n \ is \ A_n^l \ \text{THEN } y \ is \ B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***Lukasiewicz inference engine*** or ***Dienes- Rescher inference engine***, ***singleton fuzzifier*** or ***Gaussian fuzzifier*** or ***triangular fuzzifier***, and ***center average defuzzifier*** are of the following form:

$$f(x) = \frac{1}{M} \sum_{l=1}^{M} \bar{y}^l$$



Fuzzy Systems (with Maximum Defuzzifier)

The fuzzy systems with fuzzy rule base

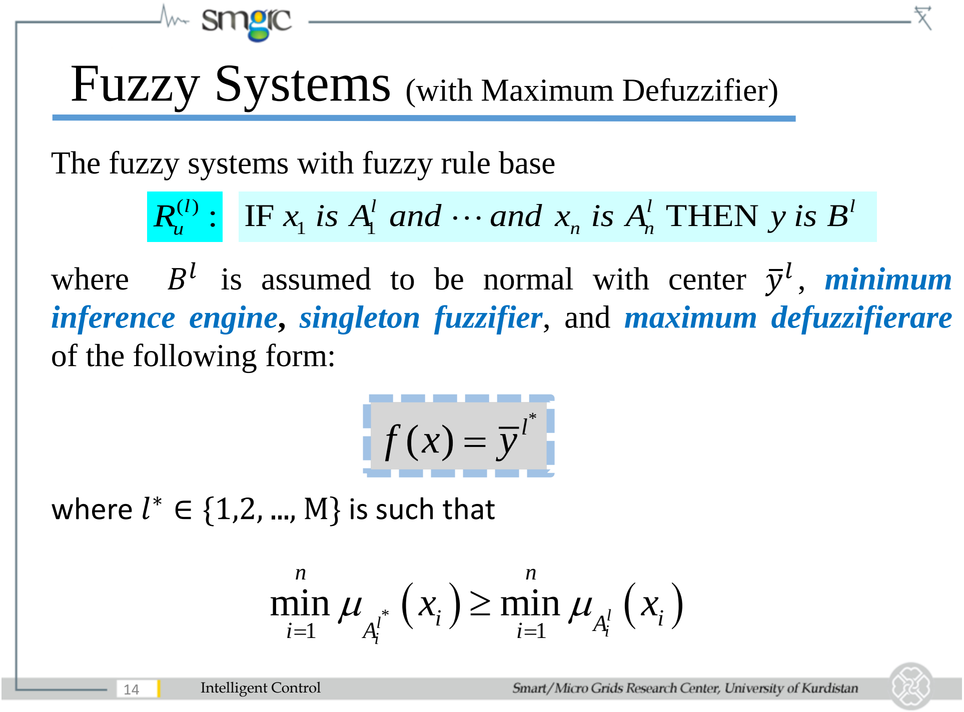
$$R_u^{(l)}: \text{IF } x_1 \textit{ is } A_1^l \textit{ and } \cdots \textit{ and } x_n \textit{ is } A_n^l \text{ THEN } y \textit{ is } B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***product inference engine***, ***singleton fuzzifier***, and ***maximum defuzzifierare*** of the following form:

$$f(x) = \bar{y}^{l^*}$$

where $l^* \in \{1,2, ..., M\}$ is such that

$$\prod_{i=1}^{n} \mu_{A_i^{l^*}}(x_i) \geq \prod_{i=1}^{n} \mu_{A_i^l}(x_i)$$



Fuzzy Systems (with Maximum Defuzzifier)

The fuzzy systems with fuzzy rule base

$$R_u^{(l)}: \text{IF } x_1 \textit{ is } A_1^l \textit{ and } \cdots \textit{ and } x_n \textit{ is } A_n^l \text{ THEN } y \textit{ is } B^l$$

where B^l is assumed to be normal with center $\bar{y}^l$, ***minimum inference engine***, ***singleton fuzzifier***, and ***maximum defuzzifierare*** of the following form:

$$f(x) = \bar{y}^{l^*}$$

where $l^* \in \{1,2, ..., M\}$ is such that

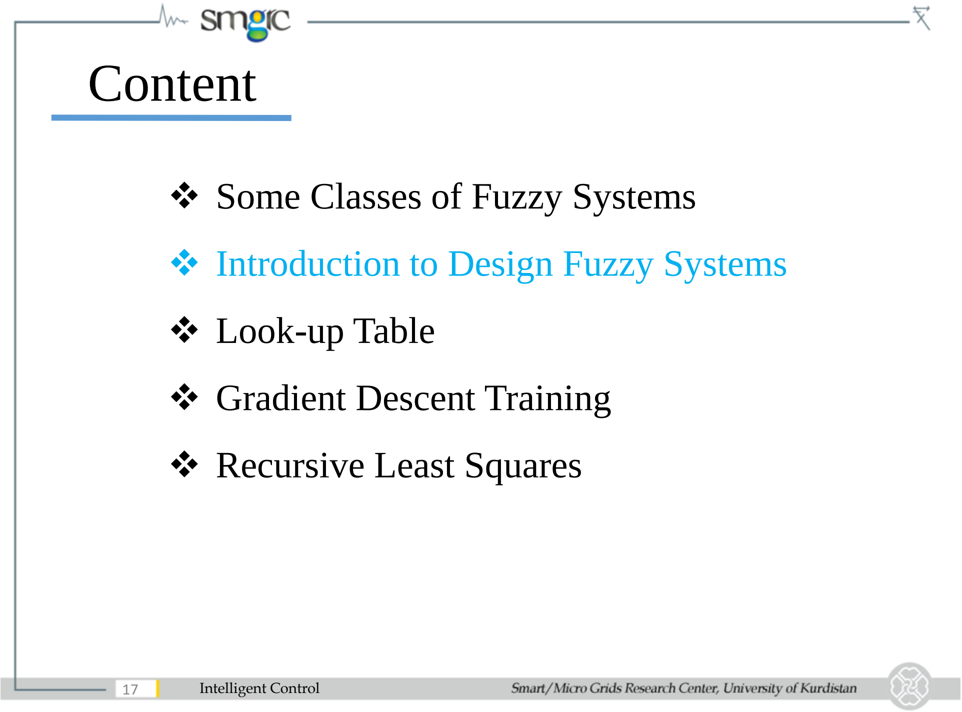
$$\min_{i=1}^{n} \mu_{A_i^{l^*}}(x_i) \geq \min_{i=1}^{n} \mu_{A_i^l}(x_i)$$

Fuzzy Systems (with Maximum Defuzzifier)

Note: These kinds of fuzzy systems are robust to small disturbances in the input and in the membership functions. However, these fuzzy systems are not continuous. As a result, If these fuzzy systems are used in ***decision making*** or other ***open-loop applications***, this kind of abrupt change may be tolerated, but it is usually unacceptable in closed-loop control.

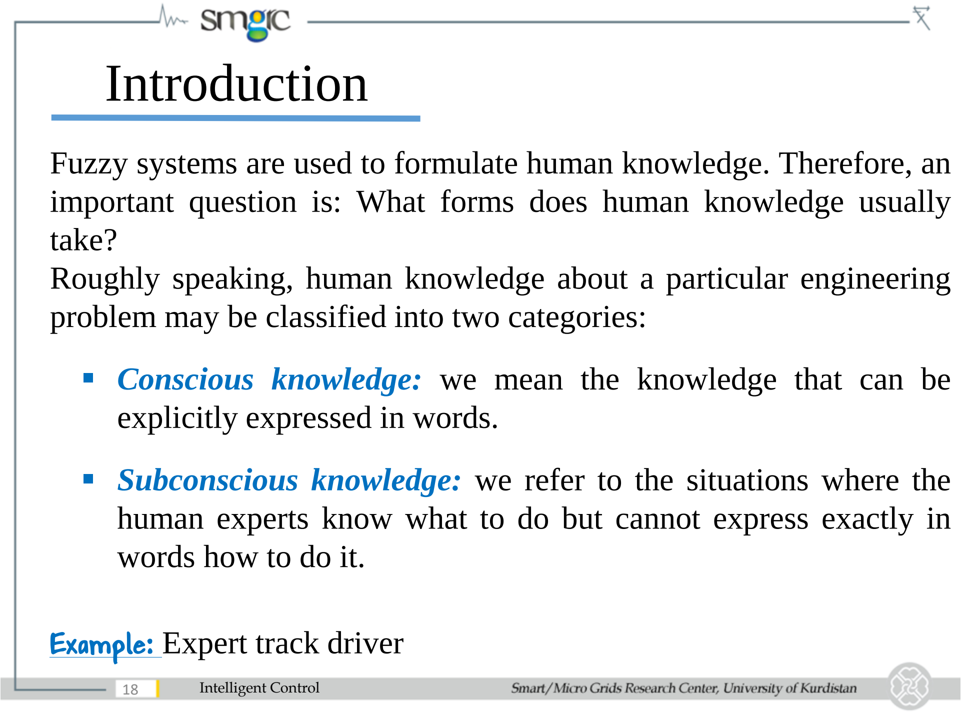
Fuzzy Systems

Software Example:



Content

- Some Classes of Fuzzy Systems
- Introduction to Design Fuzzy Systems
- Look-up Table
- Gradient Descent Training
- Recursive Least Squares



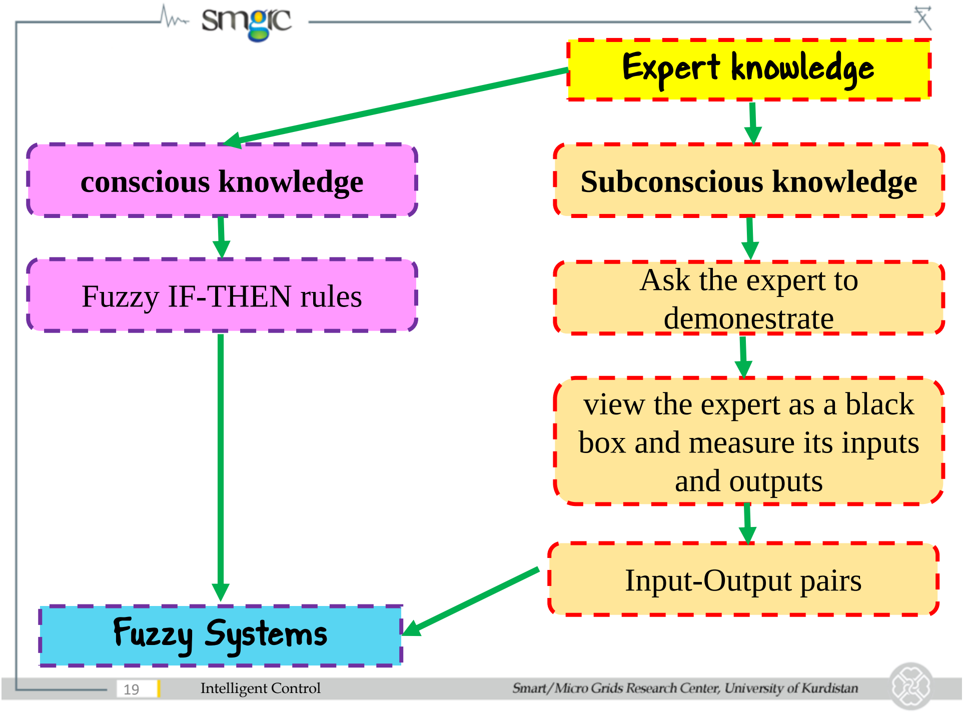
Introduction

Fuzzy systems are used to formulate human knowledge. Therefore, an important question is: What forms does human knowledge usually take?

Roughly speaking, human knowledge about a particular engineering problem may be classified into two categories:

- ***Conscious knowledge:*** we mean the knowledge that can be explicitly expressed in words.
- ***Subconscious knowledge:*** we refer to the situations where the human experts know what to do but cannot express exactly in words how to do it.

Example: Expert track driver



Expert knowledge

conscious knowledge

Subconscious knowledge

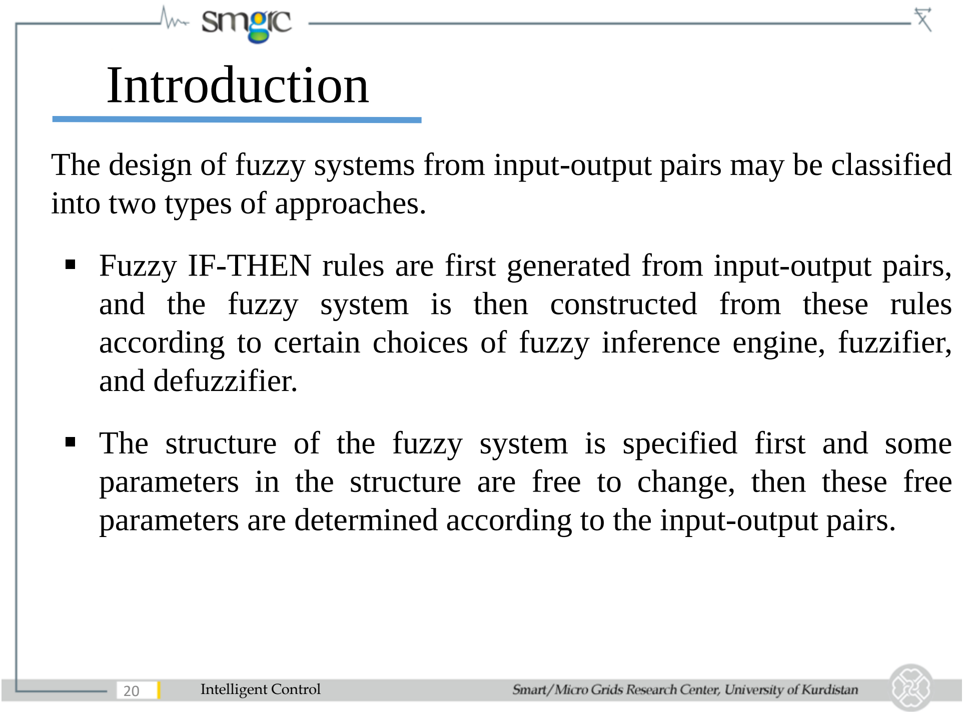
Fuzzy IF-THEN rules

Ask the expert to demonstrate

view the expert as a black box and measure its inputs and outputs

Input-Output pairs

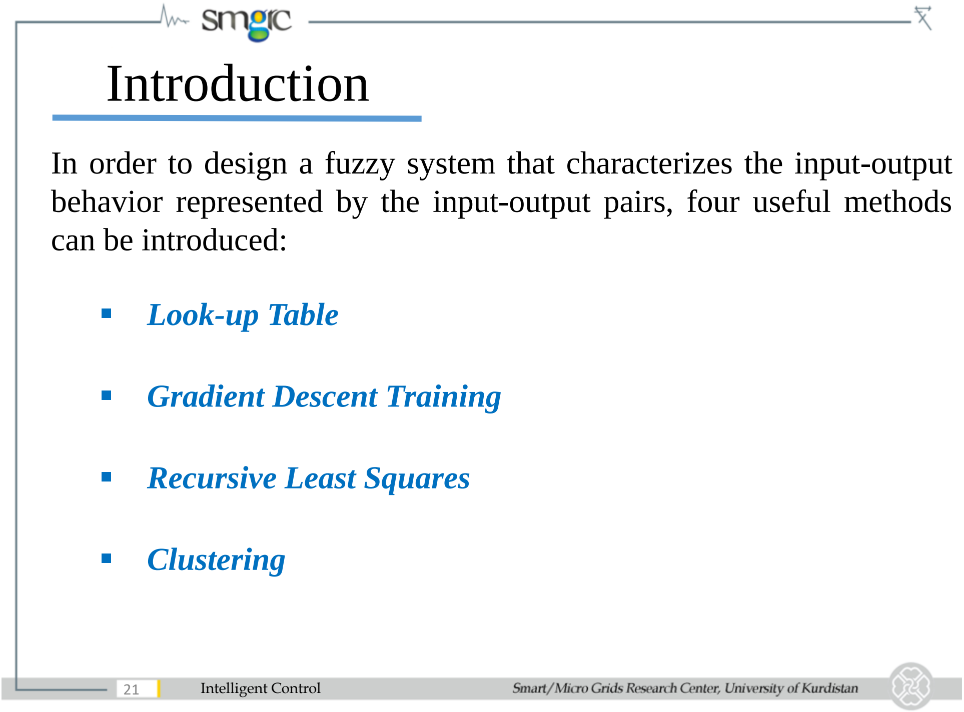
Fuzzy Systems



Introduction

The design of fuzzy systems from input-output pairs may be classified into two types of approaches.

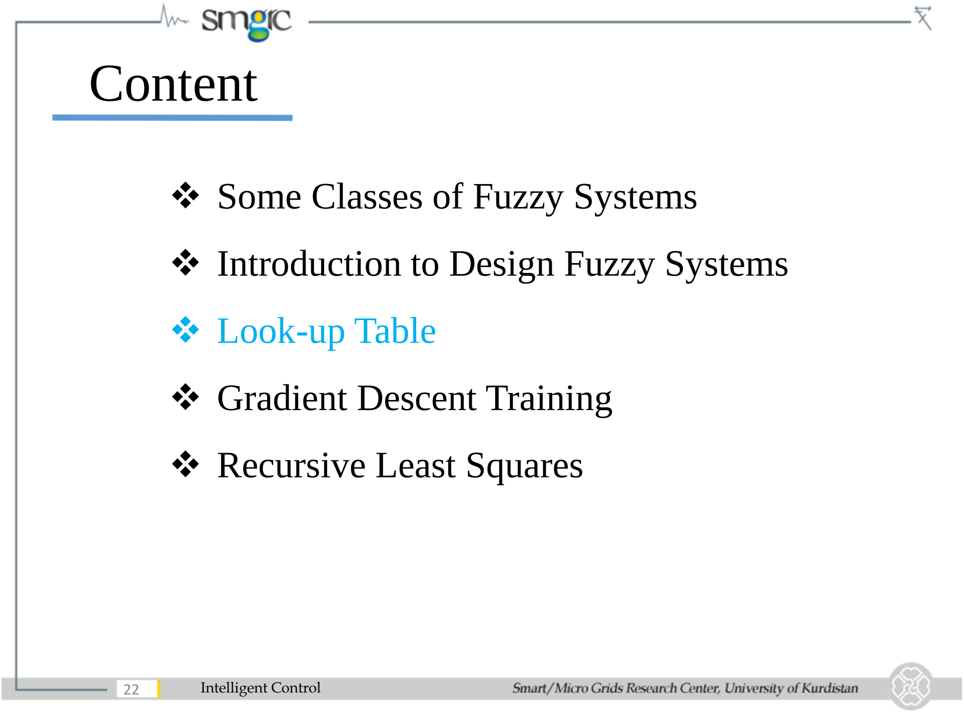
- Fuzzy IF-THEN rules are first generated from input-output pairs, and the fuzzy system is then constructed from these rules according to certain choices of fuzzy inference engine, fuzzifier, and defuzzifier.
- The structure of the fuzzy system is specified first and some parameters in the structure are free to change, then these free parameters are determined according to the input-output pairs.



Introduction

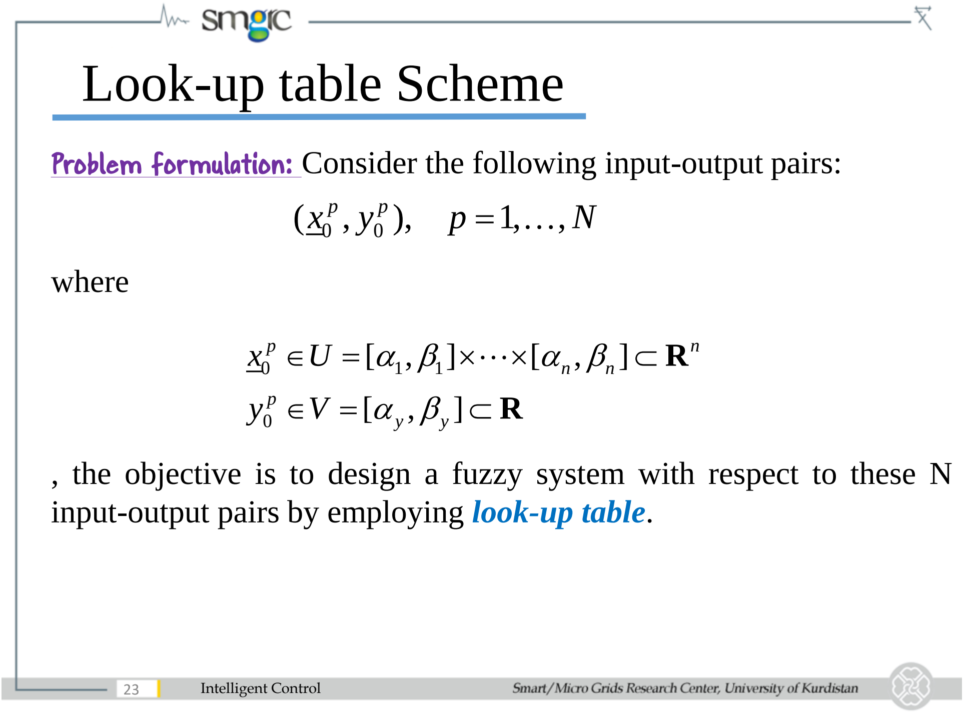
In order to design a fuzzy system that characterizes the input-output behavior represented by the input-output pairs, four useful methods can be introduced:

- ***Look-up Table***
- ***Gradient Descent Training***
- ***Recursive Least Squares***
- ***Clustering***



Content

- ❖ Some Classes of Fuzzy Systems
- ❖ Introduction to Design Fuzzy Systems
- ❖ Look-up Table
- ❖ Gradient Descent Training
- ❖ Recursive Least Squares



Look-up table Scheme

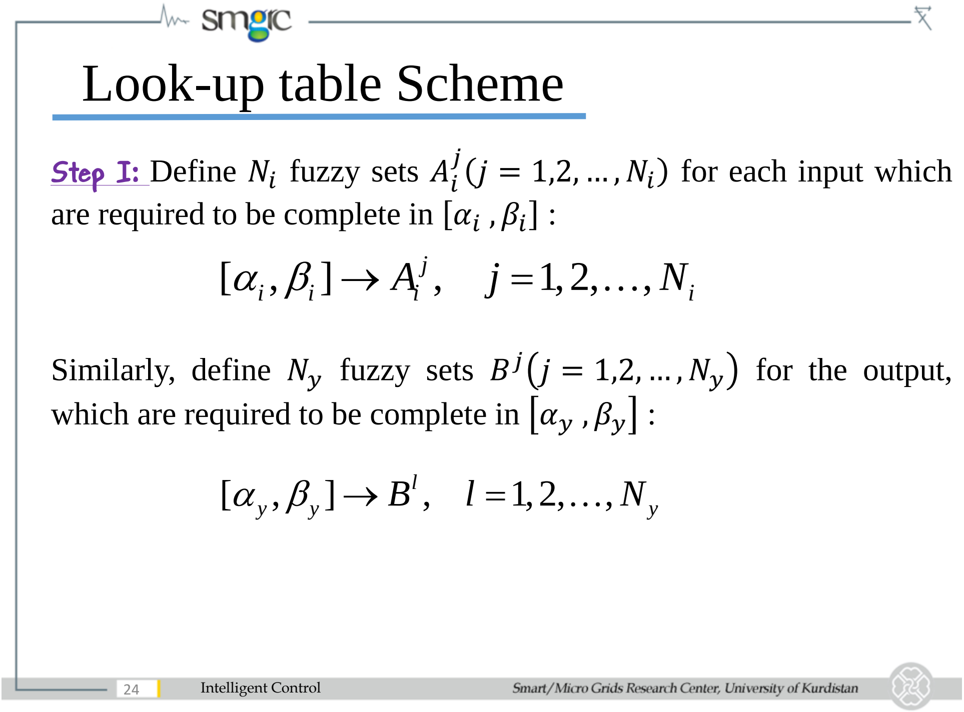
Problem formulation: Consider the following input-output pairs:

$$(\underline{x}_0^p, y_0^p), \quad p = 1, \ldots, N$$

where

$$\underline{x}_0^p \in U = [\alpha_1, \beta_1] \times \cdots \times [\alpha_n, \beta_n] \subset \mathbf{R}^n$$
$$y_0^p \in V = [\alpha_y, \beta_y] \subset \mathbf{R}$$

, the objective is to design a fuzzy system with respect to these N input-output pairs by employing ***look-up table***.



Look-up table Scheme

Step I: Define N_i fuzzy sets $A_i^j (j = 1,2, \ldots, N_i)$ for each input which are required to be complete in $[\alpha_i, \beta_i]$:

$$[\alpha_i, \beta_i] \rightarrow A_i^j, \quad j = 1, 2, \ldots, N_i$$

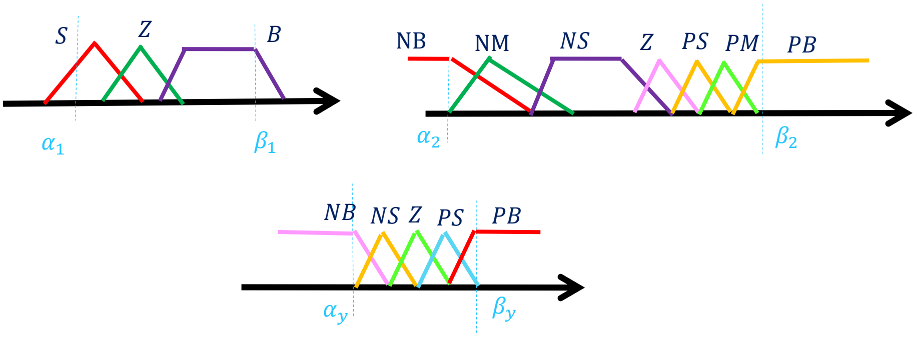
Similarly, define N_y fuzzy sets $B^j \left(j = 1,2, \ldots, N_y\right)$ for the output, which are required to be complete in $\left[\alpha_y, \beta_y\right]$:

$$[\alpha_y, \beta_y] \rightarrow B^l, \quad l = 1, 2, \ldots, N_y$$

smgc

Look-up table Scheme

Example: Consider a system with two inputs and one output. If it is assumed $N_1 = 3, N_2 = 7$ and N_y=5, then the following membership function can be chosen:



smgc

Look-up table Scheme

Step II: For each input-output pair $(x_{01}^p, x_{02}^p, \dots x_{0n}^p, y_0^p)$, Calculate the following membership functions:

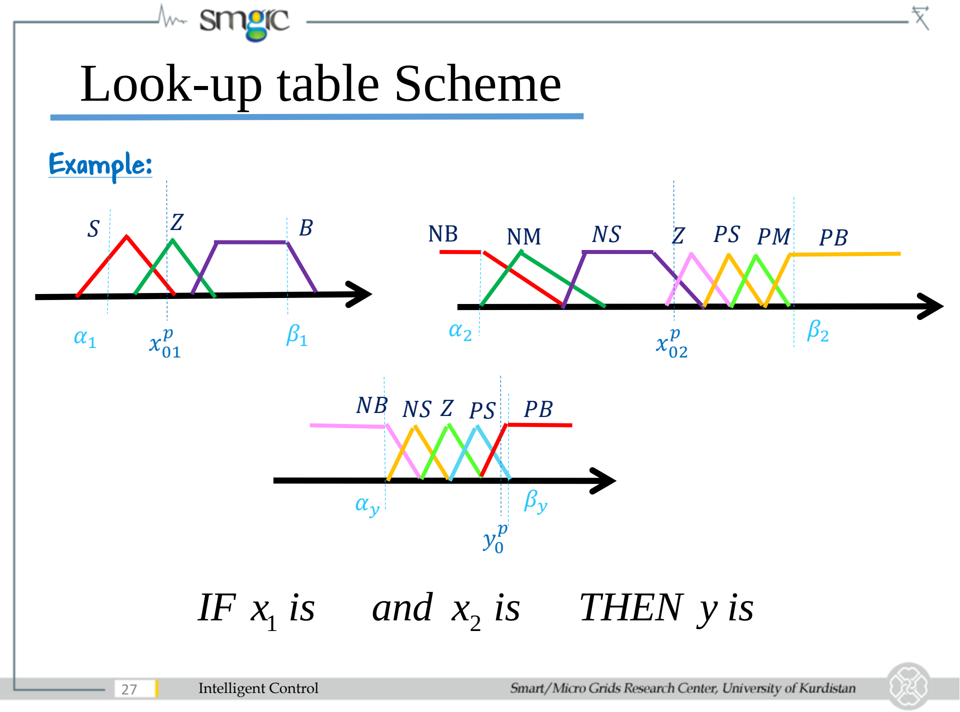
- $\mu_{A_i^j}(x_{0i}^p)$ for $(i = 1,2, \dots, n\,,\ j = 1,2, \dots, N_i)$
- $\mu_{B^j}(y_0^p)$ for $(j = 1,2, \dots, N_y)$

Then, for each input and output variable, Determine the fuzzy set with largest membership function value:

$$\mu_{A_i^{j*}}(x_{0i}^p) \geq \mu_{A_i^j}(x_{0i}^p) \qquad \mu_{B^{l*}}(y_0^p) \geq \mu_{B^l}(y_0^p)$$

Finally, obtain a fuzzy IF-THEN rule as

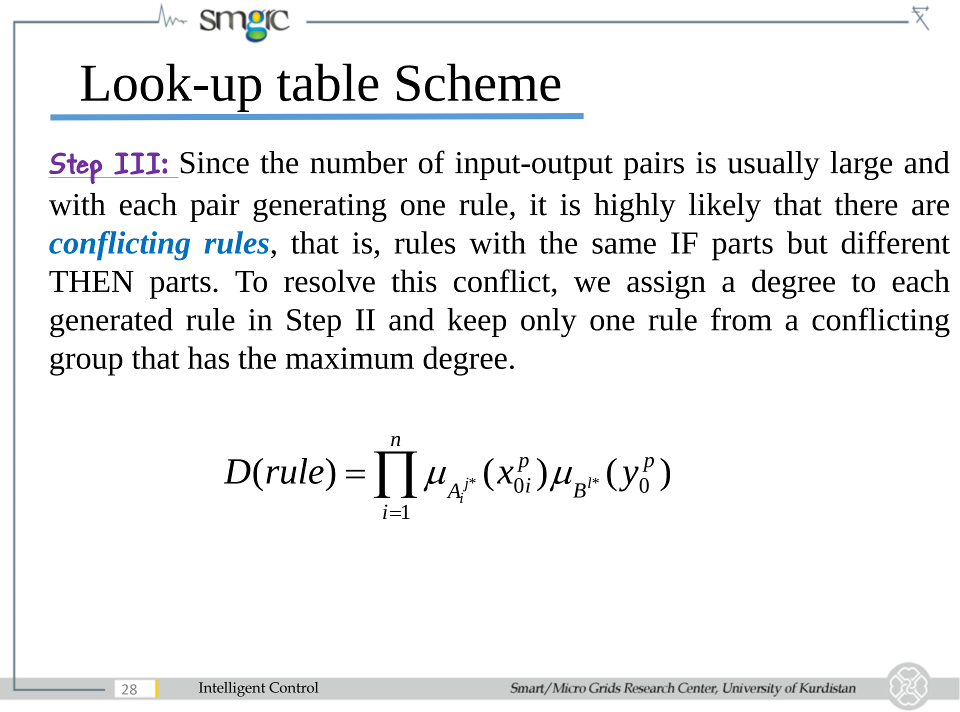
$$IF\ x_1\ is\ A_i^{j*}\ and \cdots and\ x_n\ is\ A_n^{j*}\ THEN\ y\ is\ B^{l*}$$



Look-up table Scheme

Example:

IF x_1 *is* *and* x_2 *is* *THEN* y *is*



Look-up table Scheme

Step III: Since the number of input-output pairs is usually large and with each pair generating one rule, it is highly likely that there are ***conflicting rules***, that is, rules with the same IF parts but different THEN parts. To resolve this conflict, we assign a degree to each generated rule in Step II and keep only one rule from a conflicting group that has the maximum degree.

$$D(rule)=\prod_{i=1}^{n}\mu_{A_i^{j*}}(x_{0i}^{p})\mu_{B^{l*}}(y_0^{p})$$

Look-up table Scheme

Step IV: The ***fuzzy rule base*** consists of the following three sets of rules:

- The rules generated in Step II that do ***not conflict*** with any other rules.
- The rule from a ***conflicting group*** that has the maximum degree
- ***Linguistic rules*** from human experts (due to conscious knowledge)

Look-up table Scheme

Step IV: The ***fuzzy rule base*** consists of the following three sets of rules:

- The rules generated in Step II that do ***not conflict*** with any other rules.
- The rule from a ***conflicting group*** that has the maximum degree
- ***Linguistic rules*** from human experts (due to conscious knowledge)

Note: It should be noted that a fuzzy rule base can be illustrated as a ***look-up table*** smgc

Look-up table Scheme

Example:

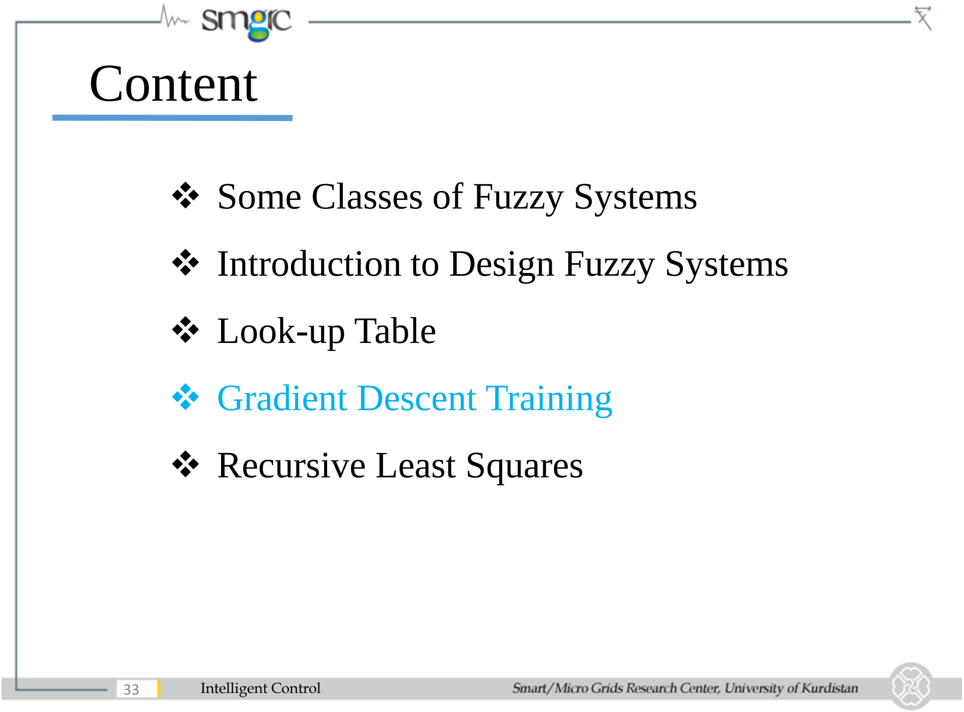
	PB			
	PM			
	PS			
x_2	Z			
	NS			
	NM			
	NB			
		S	Z	B
			x_1	

smgc

Look-up table Scheme

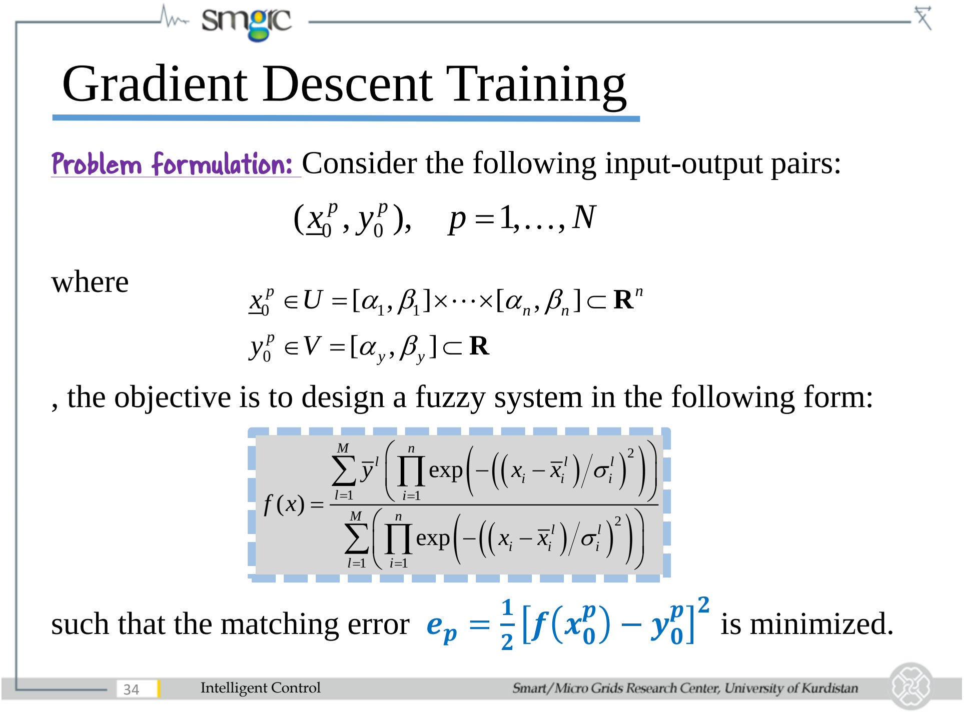
Step V: We can use any Methods explained previously to construct the fuzzy system based on the fuzzy rule base created in Step IV.

Note that the look-up table method can be employed for many applicable problems such as washing machine control, automobile control, time-series prediction and so on.



Content

- ❖ Some Classes of Fuzzy Systems
- ❖ Introduction to Design Fuzzy Systems
- ❖ Look-up Table
- ❖ Gradient Descent Training
- ❖ Recursive Least Squares



Gradient Descent Training

Problem formulation: Consider the following input-output pairs:

$$(\underline{x}_0^p, y_0^p), \quad p = 1, \ldots, N$$

where

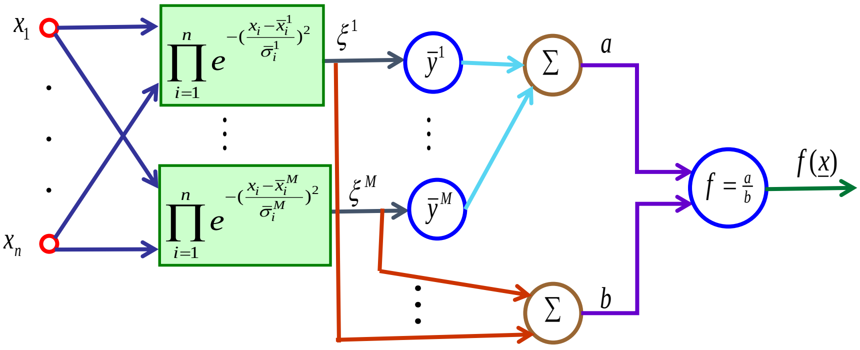
$$\underline{x}_0^p \in U = [\alpha_1, \beta_1] \times \cdots \times [\alpha_n, \beta_n] \subset \mathbf{R}^n$$
$$y_0^p \in V = [\alpha_y, \beta_y] \subset \mathbf{R}$$

, the objective is to design a fuzzy system in the following form:

$$\boxed{f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left(\prod_{i=1}^{n} \exp\left(-\left(\left(x_i - \bar{x}_i^l \right) / \sigma_i^l \right)^2 \right) \right)}{\sum_{l=1}^{M} \left(\prod_{i=1}^{n} \exp\left(-\left(\left(x_i - \bar{x}_i^l \right) / \sigma_i^l \right)^2 \right) \right)}}$$

such that the matching error $e_p = \frac{1}{2}\left[f\left(x_0^p\right) - y_0^p\right]^2$ is minimized.

Gradient Descent Training


x_1
x_n
$\prod_{i=1}^{n} e^{-(\frac{x_i-\bar{x}_i^1}{\sigma_i^1})^2}$
$\prod_{i=1}^{n} e^{-(\frac{x_i-\bar{x}_i^M}{\sigma_i^M})^2}$
ξ^1
ξ^M
$\bar{y}^1$
$\bar{y}^M$
Σ
a
Σ
b
$f=\frac{a}{b}$
$f(x)$

$$f(x)=\frac{\sum_{l=1}^{M}\bar{y}^l\left(\prod_{i=1}^{n}\exp\left(-\left(\left(x_i-\bar{x}_i^l\right)/\sigma_i^l\right)^2\right)\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n}\exp\left(-\left(\left(x_i-\bar{x}_i^l\right)/\sigma_i^l\right)^2\right)\right)}$$

Gradient Descent Training

In the gradient descent method, a coefficient of the derivative of the error relative to the designed parameter is subtracted from the old value of this parameter. For this purpose, the designed parameters of our problem are written as follows

$$\bar{y}^l(k+1)=\bar{y}^l(k)+\Delta\bar{y}^l(k)$$
$$\bar{x}_i^l(k+1)=\bar{x}_i^l(k)+\Delta\bar{x}_i^l(k)$$
$$\bar{\sigma}_i^l(k+1)=\bar{\sigma}_i^l(k)+\Delta\bar{\sigma}_i^l(k)$$

Gradient Descent Training

where

$$\Delta\bar{y}^l(k) = -\alpha\frac{\partial e(k)}{\partial\bar{y}^l(k)}$$

$$\Delta\bar{x}_i^l(k) = -\beta\frac{\partial e(k)}{\partial\bar{x}_i^l(k)}$$

$$\Delta\bar{\sigma}_i^l(k) = -\gamma\frac{\partial e(k)}{\partial\bar{\sigma}_i^l(k)}$$

with α, β and γ are minimized

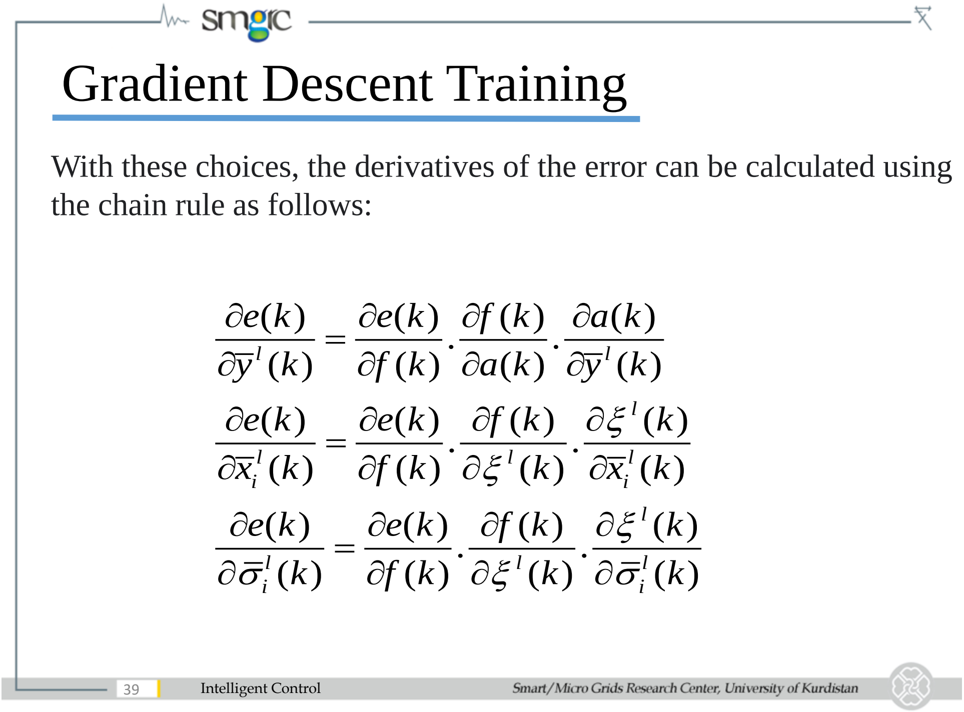
Gradient Descent Training

In order to calculate the derivatives of the error with respect to the designed parameters and according to the fuzzy system structure, we first rewrite the fuzzy system as follows:

$$f(x) = \frac{\sum_{l=1}^{M}\bar{y}^l\left(\prod_{i=1}^{n}\exp\left(-\left(\left(x_i-\bar{x}_i^l\right)/\sigma_i^l\right)^2\right)\right)}{\sum_{l=1}^{M}\left(\prod_{i=1}^{n}\exp\left(-\left(\left(x_i-\bar{x}_i^l\right)/\sigma_i^l\right)^2\right)\right)} \Rightarrow \boxed{f(k) = \frac{a(k)}{b(k)}}$$

and define

$$\boxed{\xi^l(k) = \prod_{i=1}^{n} e^{-(\frac{x_i-\bar{x}_i^l(k)}{\bar{\sigma}_i^l(k)})^2}}$$



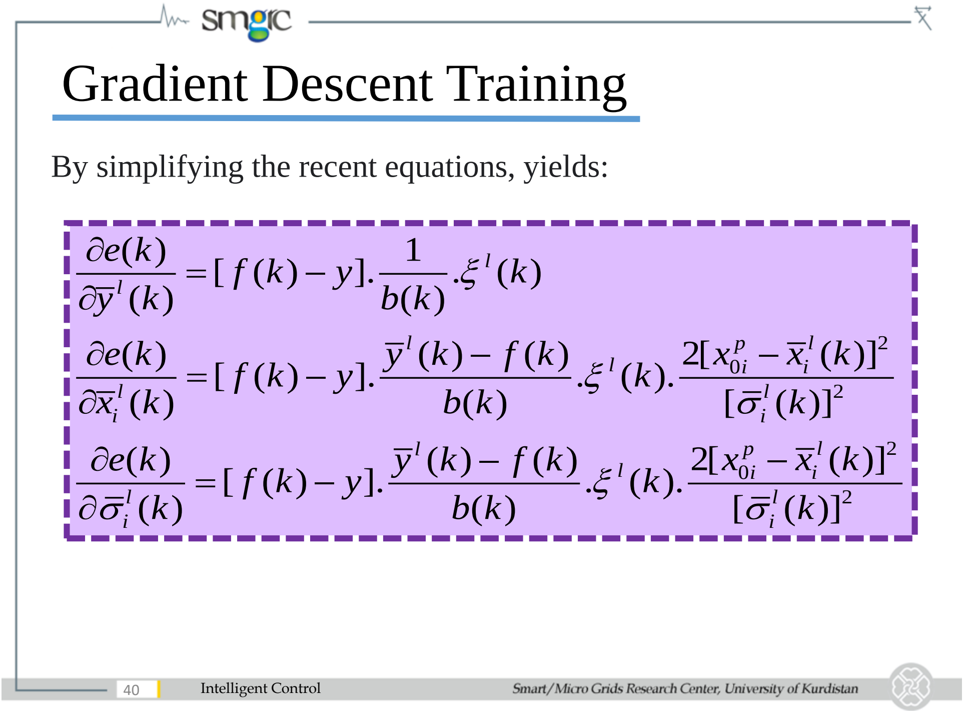
Gradient Descent Training

With these choices, the derivatives of the error can be calculated using the chain rule as follows:

$$\frac{\partial e(k)}{\partial \bar{y}^{l}(k)}=\frac{\partial e(k)}{\partial f(k)}\cdot\frac{\partial f(k)}{\partial a(k)}\cdot\frac{\partial a(k)}{\partial \bar{y}^{l}(k)}$$

$$\frac{\partial e(k)}{\partial \bar{x}_{i}^{l}(k)}=\frac{\partial e(k)}{\partial f(k)}\cdot\frac{\partial f(k)}{\partial \xi^{l}(k)}\cdot\frac{\partial \xi^{l}(k)}{\partial \bar{x}_{i}^{l}(k)}$$

$$\frac{\partial e(k)}{\partial \bar{\sigma}_{i}^{l}(k)}=\frac{\partial e(k)}{\partial f(k)}\cdot\frac{\partial f(k)}{\partial \xi^{l}(k)}\cdot\frac{\partial \xi^{l}(k)}{\partial \bar{\sigma}_{i}^{l}(k)}$$



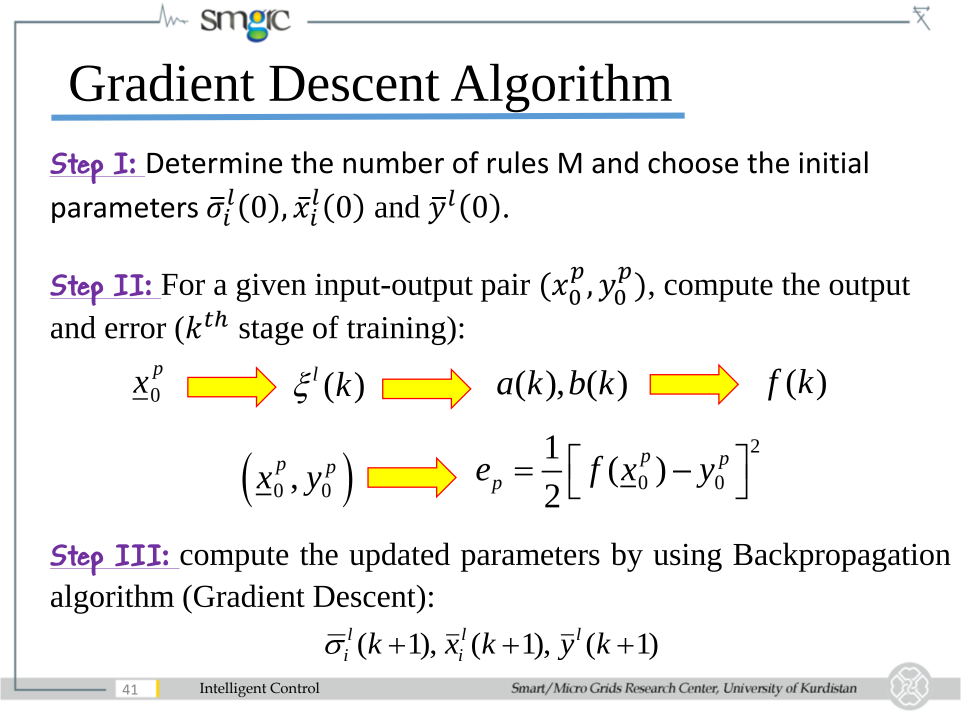
Gradient Descent Training

By simplifying the recent equations, yields:

$$\frac{\partial e(k)}{\partial \bar{y}^{l}(k)}=[f(k)-y].\frac{1}{b(k)}.\xi^{l}(k)$$

$$\frac{\partial e(k)}{\partial \bar{x}_{i}^{l}(k)}=[f(k)-y].\frac{\bar{y}^{l}(k)-f(k)}{b(k)}.\xi^{l}(k).\frac{2[x_{0i}^{p}-\bar{x}_{i}^{l}(k)]^{2}}{[\bar{\sigma}_{i}^{l}(k)]^{2}}$$

$$\frac{\partial e(k)}{\partial \bar{\sigma}_{i}^{l}(k)}=[f(k)-y].\frac{\bar{y}^{l}(k)-f(k)}{b(k)}.\xi^{l}(k).\frac{2[x_{0i}^{p}-\bar{x}_{i}^{l}(k)]^{2}}{[\bar{\sigma}_{i}^{l}(k)]^{2}}$$



Gradient Descent Algorithm

Step I: Determine the number of rules M and choose the initial parameters $\bar{\sigma}_i^l(0), \bar{x}_i^l(0)$ and $\bar{y}^l(0)$.

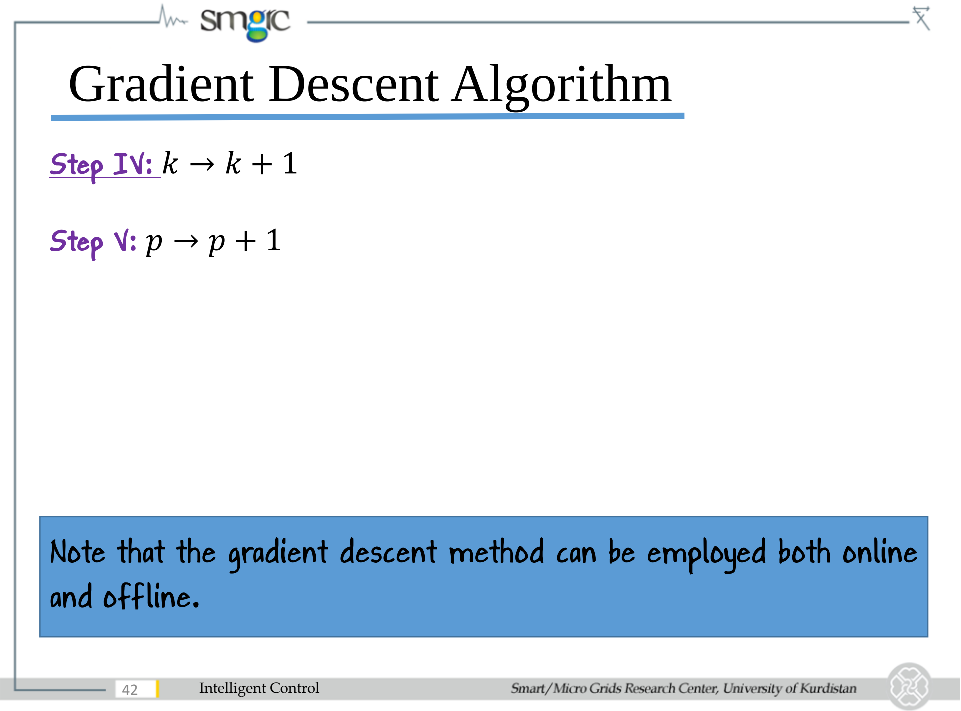
Step II: For a given input-output pair (x_0^p, y_0^p), compute the output and error (k^{th} stage of training):

$$\underline{x}_0^p \Rightarrow \xi^l(k) \Rightarrow a(k), b(k) \Rightarrow f(k)$$

$$\left(\underline{x}_0^p, y_0^p\right) \Rightarrow e_p = \frac{1}{2}\left[f(\underline{x}_0^p) - y_0^p\right]^2$$

Step III: compute the updated parameters by using Backpropagation algorithm (Gradient Descent):

$$\bar{\sigma}_i^l(k+1), \bar{x}_i^l(k+1), \bar{y}^l(k+1)$$



Gradient Descent Algorithm

Step IV: $k \rightarrow k+1$

Step V: $p \rightarrow p+1$

Note that the gradient descent method can be employed both online and offline.

Content

- ❖ Some Classes of Fuzzy Systems
- ❖ Introduction to Design Fuzzy Systems
- ❖ Look-up Table
- ❖ Gradient Descent Training
- ❖ Recursive Least Squares

Recursive Least Squares

Problem formulation: Consider the following input-output pairs:

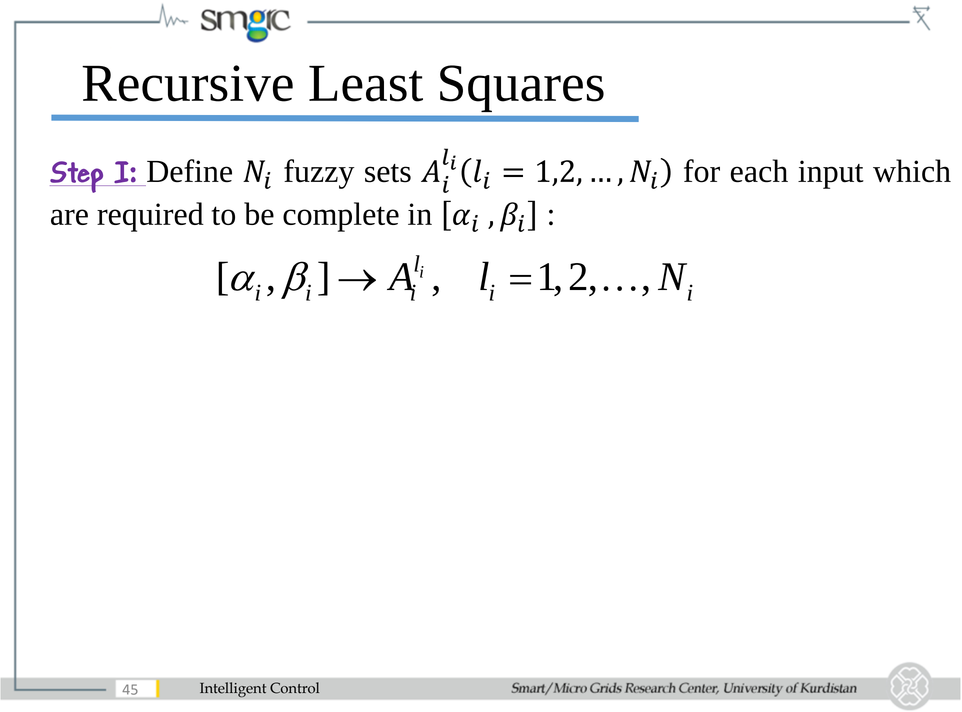
$$(\underline{x}_0^p, y_0^p), \quad p = 1, \ldots, N$$

where $\underline{x}_0^p \in U = [\alpha_1, \beta_1] \times \cdots \times [\alpha_n, \beta_n] \subset \mathbf{R}^n \;;\; y_0^p \in V = [\alpha_y, \beta_y] \subset \mathbf{R}$

, the objective is to recursively; design a fuzzy system in the following form:

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)}{\sum_{l=1}^{M} \left(\prod_{i=1}^{n} \mu_{A_i^l}(x_i) \right)} \Rightarrow \boxed{f(x) = \frac{\sum_{l_1=1}^{N_1} \cdots \sum_{l_n=1}^{N_n} \bar{y}^{l_1 \cdots l_n} \left[\prod_{i=1}^{n} \mu_{A_i^{l_i}}(x_i) \right]}{\sum_{l_1=1}^{N_1} \cdots \sum_{l_n=1}^{N_n} \left[\prod_{i=1}^{n} \mu_{A_i^{l_i}}(x_i) \right]}}$$

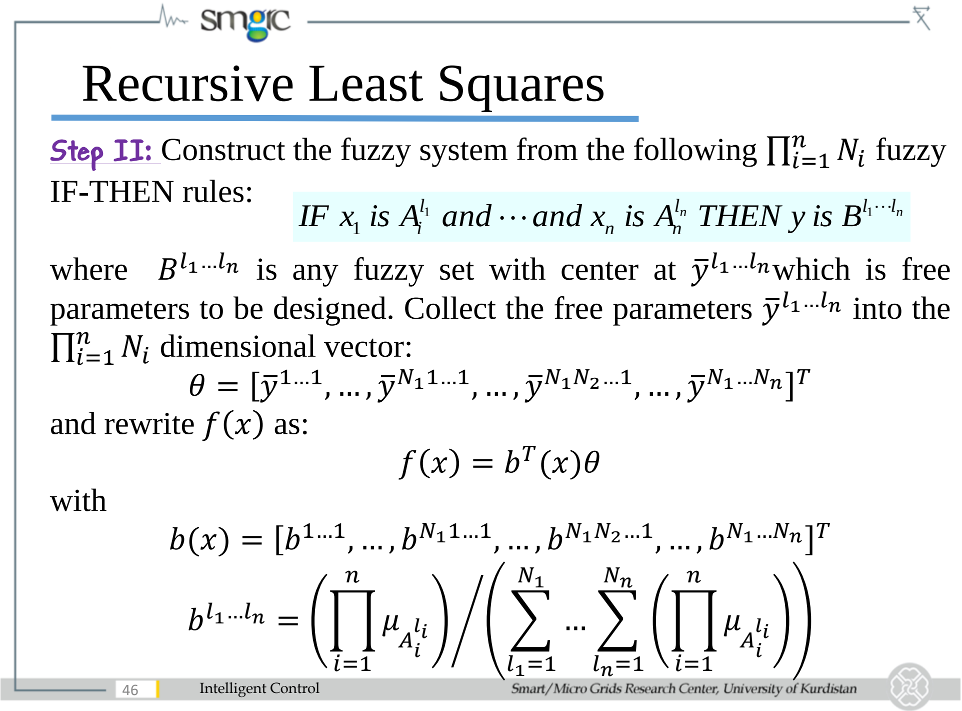
such that the matching error $J_p = \sum_{j=1}^{p} \left[f\left(x_0^j\right) - y_0^j \right]^2$ is minimized.



Recursive Least Squares

Step I: Define N_i fuzzy sets $A_i^{l_i}(l_i = 1,2,\dots,N_i)$ for each input which are required to be complete in $[\alpha_i\ ,\beta_i]$:

$$[\alpha_i,\beta_i] \to A_i^{l_i}, \quad l_i = 1,2,\ldots,N_i$$



Recursive Least Squares

Step II: Construct the fuzzy system from the following $\prod_{i=1}^{n} N_i$ fuzzy IF-THEN rules:

$$IF\ x_1\ is\ A_i^{l_1}\ and \cdots and\ x_n\ is\ A_n^{l_n}\ THEN\ y\ is\ B^{l_1\cdots l_n}$$

where $B^{l_1\dots l_n}$ is any fuzzy set with center at $\bar{y}^{l_1\dots l_n}$which is free parameters to be designed. Collect the free parameters $\bar{y}^{l_1\dots l_n}$ into the $\prod_{i=1}^{n} N_i$ dimensional vector:

$$\theta = [\bar{y}^{1\dots1},\dots,\bar{y}^{N_1 1\dots 1},\dots,\bar{y}^{N_1 N_2\dots 1},\dots,\bar{y}^{N_1\dots N_n}]^T$$

and rewrite $f(x)$ as:

$$f(x) = b^T(x)\theta$$

with

$$b(x) = [b^{1\dots1},\dots,b^{N_1 1\dots 1},\dots,b^{N_1 N_2\dots 1},\dots,b^{N_1\dots N_n}]^T$$

$$b^{l_1\dots l_n} = \left(\prod_{i=1}^{n}\mu_{A_i^{l_i}}\right) \Bigg/ \left(\sum_{l_1=1}^{N_1}\cdots\sum_{l_n=1}^{N_n}\left(\prod_{i=1}^{n}\mu_{A_i^{l_i}}\right)\right)$$

smgrc

Recursive Least Squares

Step III: Choose the initial parameters θ(0) :

- Use linguistic rules
- select arbitrarily in the output space

Step IV: For p = 1,2, … , N , compute the parameters θ using the following recursive least squares algorithm:

$$\theta(p)=\theta(p-1)+k(p)[y_0^p-\underline{b}^T[x_0^p]\theta(p-1)]$$

$$k(p)=Q(p-1)\underline{b}^T(x_0^p)[\underline{b}^T(x_0^p)Q(p-1)\underline{b}^T(x_0^p)+1]^{-1}$$

$$Q(p)=Q(p-1)-Q(p-1)\underline{b}(x_0^p)[\underline{b}^T(x_0^p)Q(p-1)\underline{b}^T(x_0^p)+1]^{-1}$$
$$.\underline{b}^T(x_0^p)Q(p-1)$$

with Q(0) = σI (σ is a large constant)

smgrc

Thanks