

INSA de Rennes

IRISA

**Fuzzy Modeling and Optimization:
Interpretability and Accuracy**

Pierre-Yves Glorennec
INSA de Rennes/IRISA
glorenne@irisa.fr

Mécatronique : recherche et application

Beyrouth, les 7, 8 et 9 février 2005

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization.

Outline

Fuzzy Inference Systems

1. General Presentation
 - Motivations
 - Fuzzy Logic in a few words
 - Neuro-Fuzzy ??
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization.

Motivations



Problem

- Explain, in natural language, the largest part of a given input/output relationship.
- Build a knowledge base automatically.

Key-Words

- Knowledge Extraction.
- Interpretable Rules.
- Generic Situations, Prototypes.

Tools

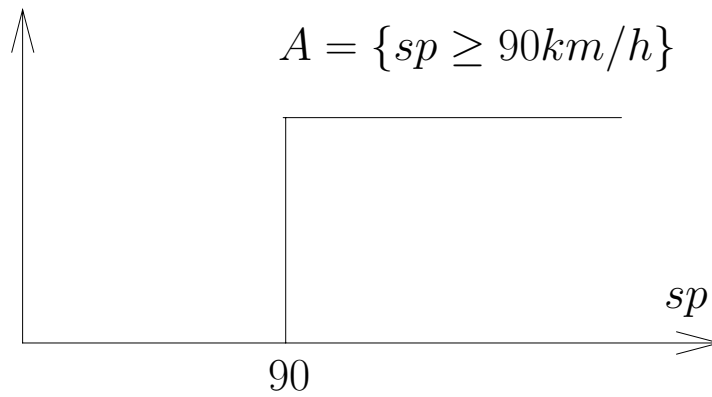
- Fuzzy Inference Systems.
- Fuzzy Regression Trees.

Why Fuzzy Logic ?

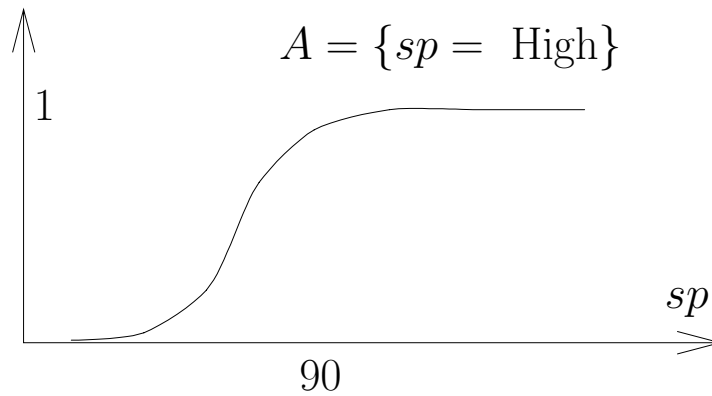
Three main reasons. Fuzzy logic

- is near to the Human Reasoning,
- allows easy knowledge representation,
- avoids the “black box” effects.

Fuzzy sets



$$1_A(sp) = \begin{cases} 0 & \text{if } sp < 90 \\ 1 & \text{otherwise} \end{cases}$$

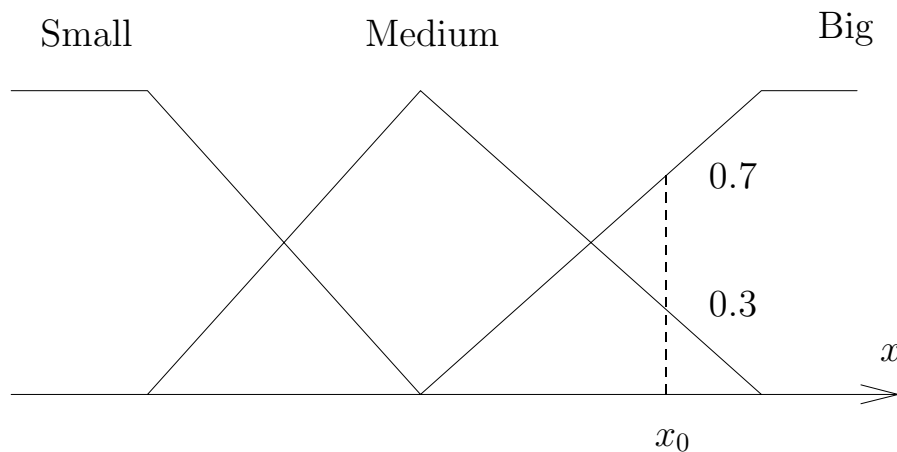


$$\mu : sp \rightarrow \mu_{high}(sp) \in [0, 1]$$

Fuzzy Partitions

Variation domain of a variable divided by a fuzzy partition:

- coarse categorization,
- reducing complexity.



- membership degree $\in [0, 1]$
- gradual change from a category to another.

Fuzzy rules

Knowledge base

A set of rules such that:

if \mathbf{x} is S_i then conclusion

where

- $\mathbf{x} = (x_1, \dots, x_n)^t$ is an input vector,
- S_i is a vector of linguistic labels.
- In this paper, the conclusions have the form:
 $y = \text{XXX}$, where XXX is a real number.

Constrained learning algorithms allow automatic rule extraction from numerical data.

Fuzzy vs Boolean

Fuzzy Logic and continuous data

- avoiding the threshold effects,
- allowing passage from quantitative to qualitative,

examples :

$T < 4.3^{\circ}\text{C}$, yes or no ?

Temperature is cold

at what degree this temperature is cold ?

The real world is not Boolean !!

Knowledge processing

Fuzzy systems:

1. Rule Base:

- Natural language,
- symbolical level,
- interpretability (if constrained learning)

2. Symbolical-Numerical Interface:

- continuous data from sensors,
- membership degrees to fuzzy sets.

3. Inference Engine:

- numerical level,
- the way to process input data using our knowledge.

Benefits of interpretability

- Initialization of parameters from knowledge
 - The initial behavior is reasonably good.
 - Further improvements are possible by learning.
- Detection of aberrant solutions.
- Confidence Degree
 - Zones with insufficient exploration.
 - Particular cases.

Interpretability/Accuracy

Interpretability

- Natural language \Rightarrow fuzzy rules
- Possible *a posteriori* validation by an expert

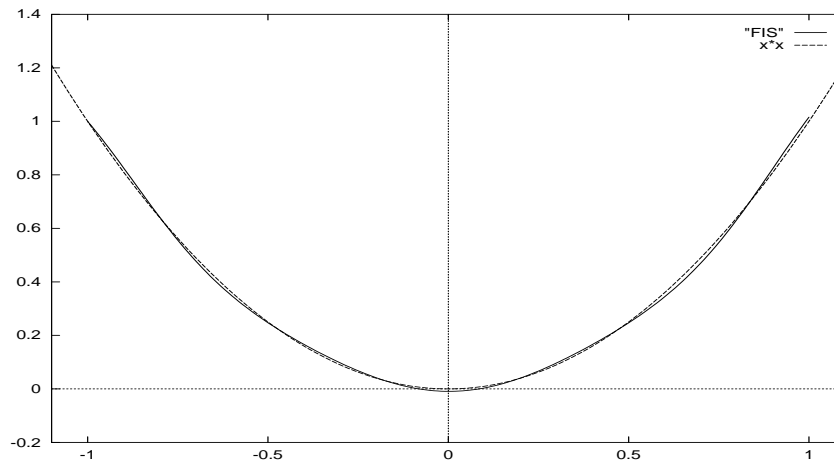
Accuracy

- Parametric Optimization
- Neuro-Fuzzy Models \Leftrightarrow RBF-like NN, but NN are Black-Boxes

“Neuro-fuzzy” ? A wrong way to overcome this Dilemma !!

Neuro-Fuzzy Optimization (1)

Approximation of $x \rightarrow x^2$.



Optimized Rule Base:

if x is NB then $y = 1.34$

if x is NM then $y = 0.23$

if x is ZR then $y = -0.11$

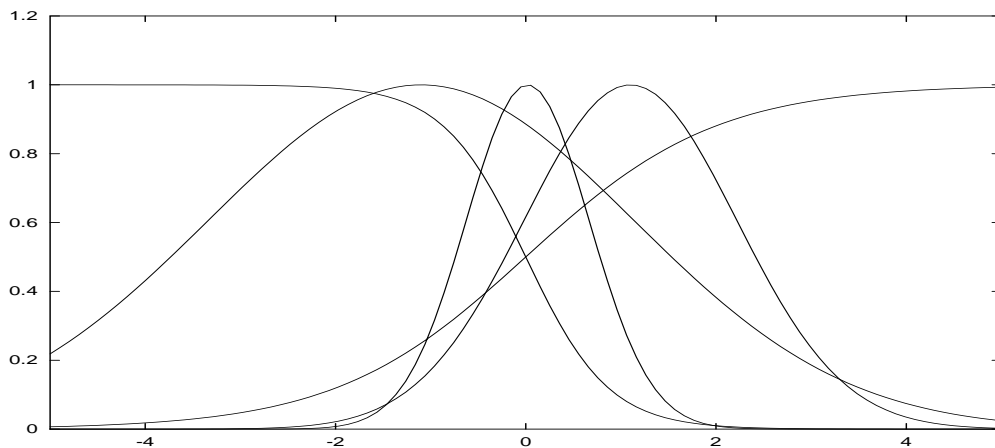
if x is PM then $y = 0.25$

if x is PB then $y = 1.42$

if x is ZR then x^2 is negative ?????

Neuro-Fuzzy Optimization (2)

Intermixed membership functions



On this (*not so caricatural...*) example, the observation “ $x = 0$ ” is:

NB at 50%

NS at 89%

ZR at 100%

PS at 62%

PB at 50%

⇒ the brouhaha overlaps information...

Fuzzy rule Optimization (1)

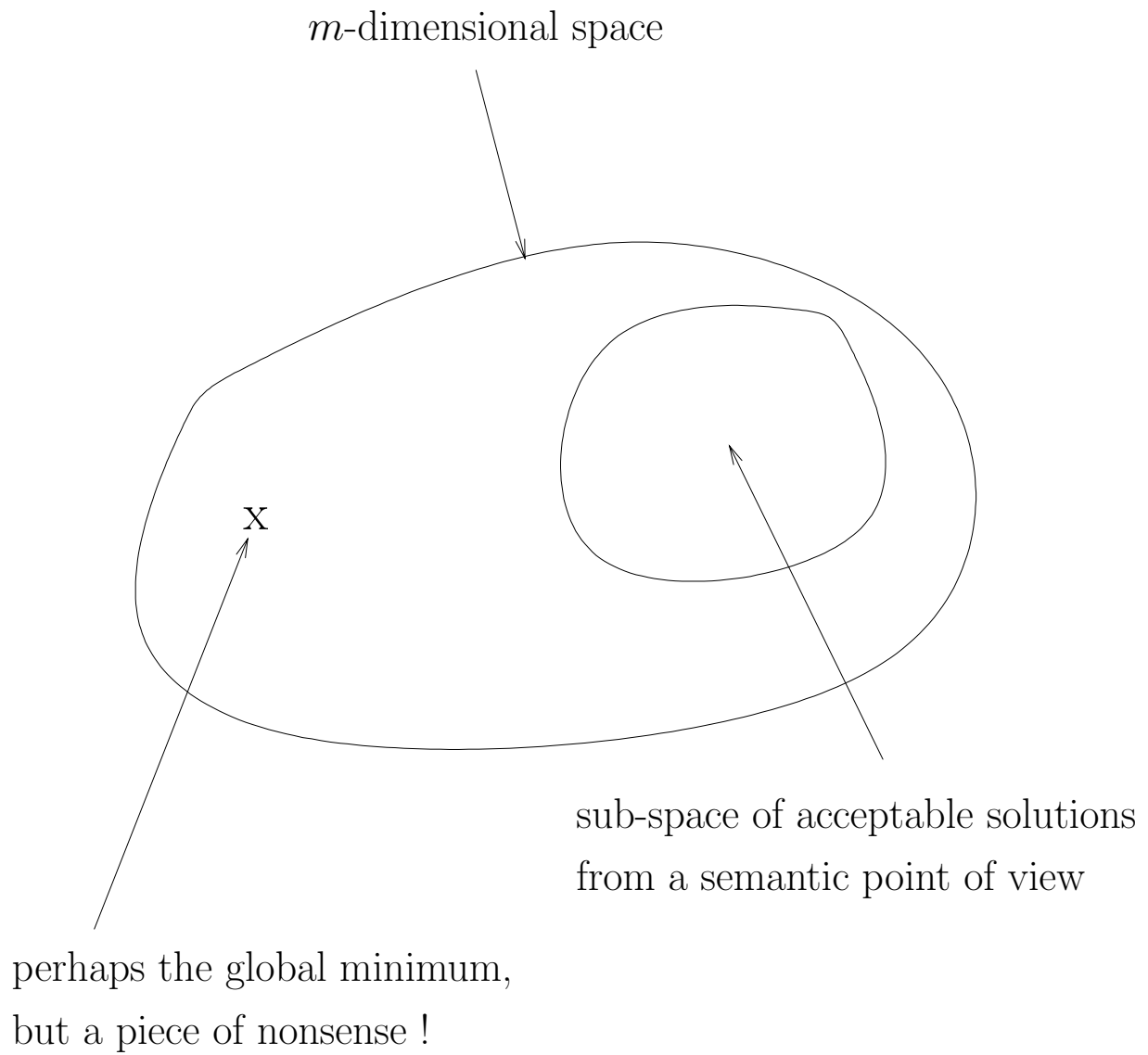
A neural-like approach is not necessary:

- interpretability is not guaranteed by gradient-based methods,
- a fuzzy relation is not always differentiable.

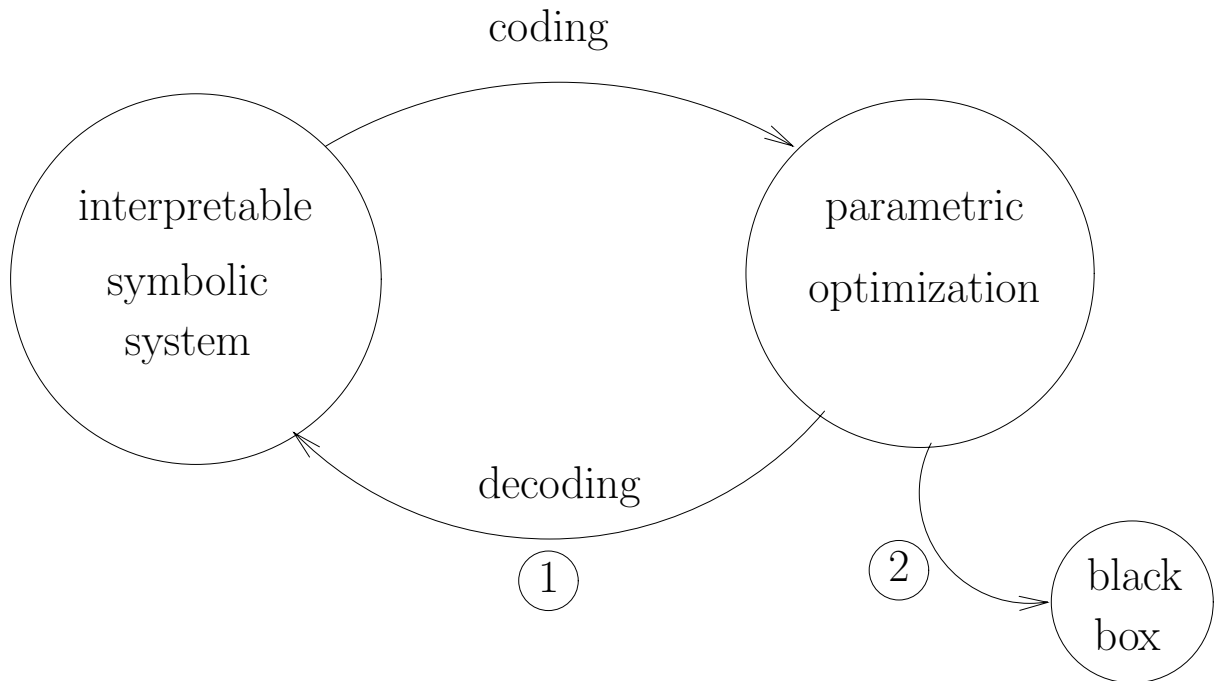
A fuzzy rule base has some specificities:

- a fuzzy rule represents a piece of knowledge,
- interpretability is necessary *before* and *after* optimization.

Fuzzy rule Optimization (2)



Solving the dilemma



Is it possible to increase accuracy while preserving interpretability ?

\Rightarrow *Constrained Optimization*

Outline

Fuzzy Inference Systems

1. General Presentation.
2. **Constrained Optimization**
 - Takagi-Sugeno Models
 - Constraints to introduce
 - Divide and conquer
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization.

Fuzzy Inference

Order-0 Takagi-Sugeno Model

linguistic expression

N rules of type :

$$\left\{ \begin{array}{l} \text{if } x_1 \text{ is } A_1^{i(1)} \text{ and ... and } x_n \text{ is } A_n^{i(n)} \\ \text{then } y = c_i \end{array} \right.$$

analytic expression

$$y(\mathbf{x}) = \frac{\sum_{i=1}^N \alpha_i(\mathbf{x}) \times c_i}{\sum_j \alpha_j(\mathbf{x})}$$

$\alpha_i(\mathbf{x}) =$ truth value of rule i given input \mathbf{x}

Comments

From the linguistic form

⇒ knowledge representation

From the analytic form

⇒ a lot of optimization methods,

but remember the linguistic point of view !!

Learning methods

Supervised Learning

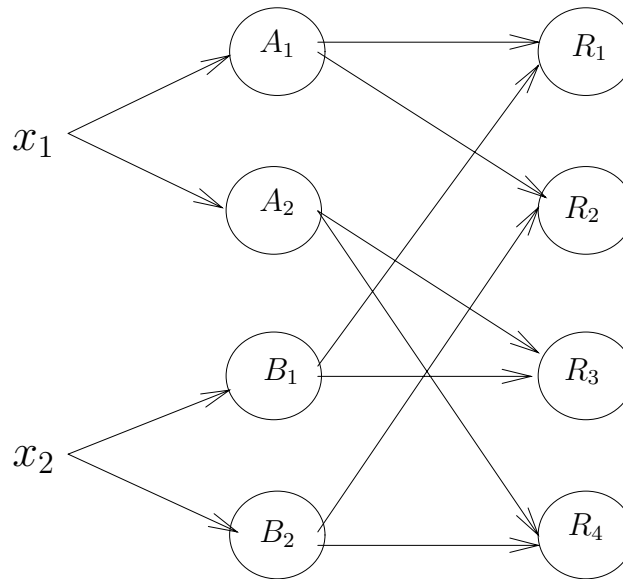
- Structural and parametric optimization.
- On-line or off-line algorithms.
- Input/output reference pairs:
 $\mathcal{L} = \{\mathbf{x}, d(\mathbf{x})\}$.

Reinforcement Learning

- Only optimization of the conclusion parts.
- On-line algorithms.
- Model-free optimization is possible.

Constraints to introduce

- Shared membership functions:



- Strong Fuzzy Partitions:

$$\forall x \in \mathcal{D}, \sum_i \mu_{A_i}(x) = 1$$

- Physically possible conclusions:

$$\forall i, c_i \in [\min_{\mathbf{x}} d(\mathbf{x}), \max_{\mathbf{x}} d(\mathbf{x})]$$

Property

Proposition

For all rule, i , we can find an input vector, \mathbf{x}^i , such that:

$$\alpha_i(\mathbf{x}^i) = 1$$

$$(\forall j \neq i)(\alpha_j(\mathbf{x}^i) = 0)$$

Moreover, if y is the inferred output:

$$y(\mathbf{x}^i) = c_i$$

Each rule can be examined separately

Definition

Such a vector, \mathbf{x}^i , is called a *Prototype* for rule i .

Two sets of parameters (1)

Output of a Takagi-Sugeno FIS:

$$y(\mathbf{x}) = \frac{\sum_{i=1}^N \alpha^i(\mathbf{x}) \times c_i}{\sum_{j=1}^N \alpha^j(\mathbf{x})} = \sum_{i=1}^N \frac{\alpha^i(\mathbf{x})}{\sum_{j=1}^N \alpha^j(\mathbf{x})} \times c_i$$

If the input membership functions are given,

then, the conclusions $(c_i)_{i=1}^N$ are solution of a set of linear equations

Two sets of parameters (2)

⇒ Two sets of parameters :

- parameters of input membership functions: set M ,
- rule conclusions, for a given set of membership functions: set C_M .

⇒ *For all modification of membership functions, we need to recompute the conclusions.*

Benefits

1. Reduction of the search space.
2. Specific algorithms for each set of parameters:
 - $C_M \rightarrow$ two initialization algorithms
(+ gradient descent)
 - $M \rightarrow$ evolutionary strategy:
 - differentiability is not necessary,
 - easy introduction of constraints.
3. convergence theorems
4. The semantics can be guaranteed.

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. **Parameter Optimization**
 - General algorithm
 - Conclusion part of fuzzy rules
 - Optimal positioning of memb. functions
 - Analysis of the learning stage
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization.

General learning algorithm

initialization

choose the membership functions, M

compute conclusions, C_M

termination condition = FALSE

while \neg (termination condition) **do**

modify the membership functions

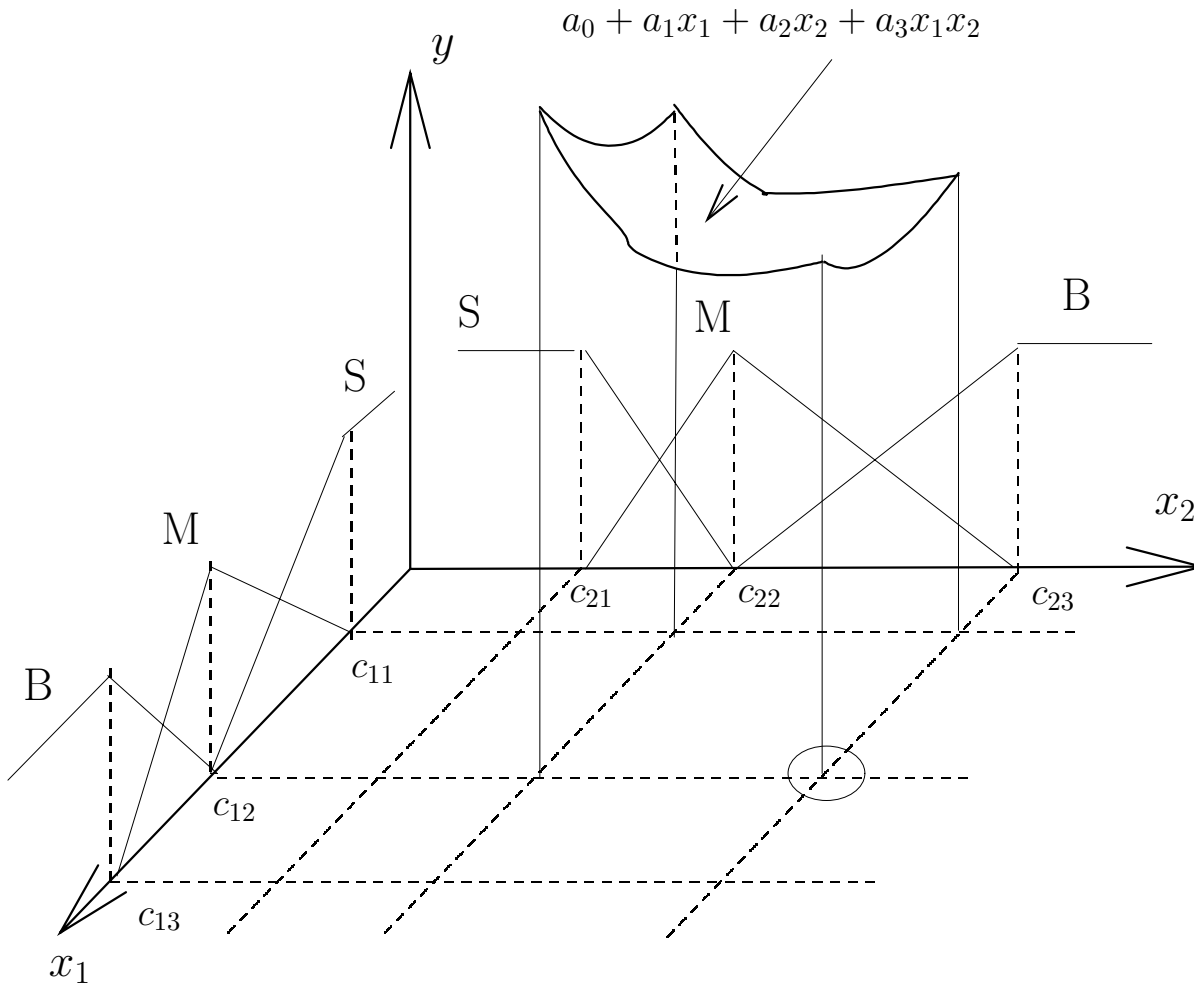
compute the corresponding concl.

accept or reject the changes

compute the termination condition

done

Off-line initialization of conclusions (1)



$(x_1, x_2) \rightarrow y = FIS(x_1, x_2)$ defines a surface in \mathcal{R}^3 .

- The grid points correspond to the prototypes.
- Let $(\mathbf{x}, d(\mathbf{x}))$ be a training pair, then $d(\mathbf{x})$ belongs to the surface.

Off-line initialization of conclusions (2)

Let $\mathcal{L} = \{(\mathbf{x}_p, d_p)_{p \in P}, \mathbf{x}_p \in \mathcal{R}^n, d_p \in \mathcal{R}\}$ be a learning set.

Principle : the prototype for rule i is approximated by the example closest to the corresponding grid point.

Algorithm

for each rule i **do**
 search (\mathbf{x}^*, d^*) in \mathcal{L} such that :
 $\alpha_i(\mathbf{x}^*) = \max_{\mathbf{x} \in \mathcal{L}} \alpha_i(\mathbf{x})$
 $c_i = d^*$
done

Comment

Some sensibility to noise on learning data.

Off-line initialization of conclusions (3)

Let $S_k(i)$ be the set of the k training vectors with the greatest activation for rule i , ($k \leq 3$ usually):

$$S_k(i) = \{ \mathbf{x}_{i(1)} \dots \mathbf{x}_{i(k)} \in \mathcal{L} / (\forall j \in [1, k]) \\ (\forall \mathbf{x} \notin S_k(i)) (\alpha_i(\mathbf{x}_{i(j)}) \geq \alpha_i(\mathbf{x})) \}$$

The conclusion is approximated by:

$$c_i = \frac{\sum_{\mathbf{x} \in S_k(i)} \alpha_i(\mathbf{x}) \times d(\mathbf{x})}{\sum_{\mathbf{x} \in S_k(i)} \alpha_i(\mathbf{x})}$$

Properties

$$(\forall i) (\min_{\mathbf{x} \in \mathcal{L}} d(\mathbf{x}) \leq c_i \leq \max_{\mathbf{x} \in \mathcal{L}} d(\mathbf{x}))$$

$$val(i) = \max_{\mathbf{x} \in S_k(i)} \alpha_i(\mathbf{x}) = \alpha_i^{\max}$$

On-line initialization of conclusions (1)

Let \mathcal{O}_i^K be the set of observed training pairs for rule i , from 0 to K :

$$\mathcal{O}_i^K = \{(\mathbf{x}_t, d(\mathbf{x}_t)) / \alpha_i(\mathbf{x}_t) > 0, 0 \leq t \leq K\}$$

and:

$$\alpha_i^{\max}(K) = \max_{\mathbf{x} \in \mathcal{O}_i^K} \alpha_i(\mathbf{x})$$

Algorithm

1. $\alpha_i^{\max}(= c_i) = 0$, for all i .

2. For each pair $(\mathbf{x}, d(\mathbf{x}))$

- compute $FIS(\mathbf{x})$

- if $\alpha_i(\mathbf{x}) > \alpha_i^{\max}$ or if $\alpha_i(\mathbf{x}) > 0.9$,

$$c_i \leftarrow \frac{\alpha_i^{\max} \times c_i + \alpha_i(\mathbf{x}) \times d(\mathbf{x})}{\alpha_i^{\max} + \alpha_i(\mathbf{x})}$$

- update α_i^{\max}

3. go to step 2 or end.

On-line initialization of conclusions (2)

Properties

1. For each conclusion:

$$c_i \in \left[\min_{\mathbf{x} \in \mathcal{O}_i^K} d(\mathbf{x}), \max_{\mathbf{x} \in \mathcal{O}_i^K} d(\mathbf{x}) \right]$$

2. Validity of rule i :

$$val(i, K) = \alpha_i^{\max}(K)$$

Embedding initial knowledge

$$c_i = \text{appropriate value}$$

$$\alpha_i^{\max}(0) = \text{from 0 to 1, according to the initial knowledge}$$

Main characteristics

For both on-line and off-line updating of conclusions of the rules:

1. the training data are used **only once**: it is a direct method;
2. **the errors** (difference between actual and desired outputs) **are not taken into account !!**

Optimization of memb. functions (1)

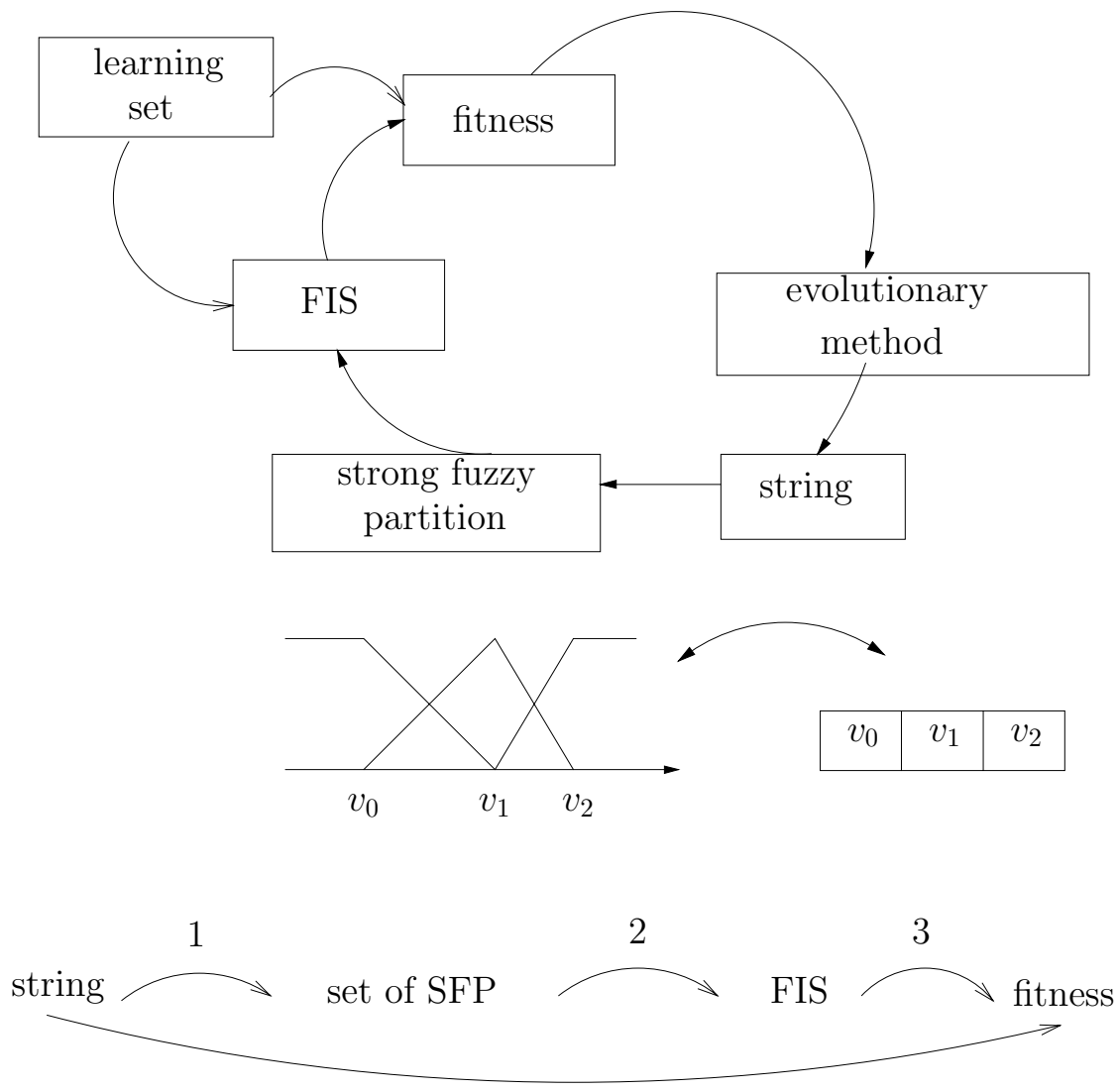
- The parameters of input membership functions are coded by a string.
- With triangular fuzzy partitions, we have only to code the modal values with the constraints

$$(\forall i)(\forall k)(c_k^i < c_{k+1}^i)$$

c_1^1	\dots	$c_{n_1}^1$	c_1^2	\dots	$c_{n_2}^2$	\dots	c_1^m	\dots	$c_{n_m}^m$
---------	---------	-------------	---------	---------	-------------	---------	---------	---------	-------------

To each string is associated a FIS that has to learn its conclusion parts.

Optimization of memb. functions (2)



1. Decoding the string.
2. off-line optimization of conclusions.
3. MSE.

Optimization of memb. functions (3)

Solis and Wetts's algorithm

1. Choose $M^{(0)}$. Let $B^{(0)} = 0$ and $k = 0$.
2. Calculate $E(M^{(k)})$
3. Generate a Gaussian vector, $g^{(k)}$,
with $g^{(k)} = B^{(k)} + \mathcal{N}(0, \sigma)$.
4. **if** $E(M^{(k)} + g^{(k)}) < E(M^{(k)})$
then $M^{(k+1)} = M^{(k)} + g^{(k)}$
 $B^{(k+1)} = 0.4g^{(k)} + 0.2B^{(k)}$
else if $E(M^{(k)} - g^{(k)}) < E(M^{(k)})$
then $M^{(k+1)} = M^{(k)} - g^{(k)}$
 $B^{(k+1)} = B^{(k)} - 0.4g^{(k)}$
else $M^{(k+1)} = M^{(k)}$
 $B^{(k+1)} = 0.5B^{(k)}$
5. **if** $k > \text{MaxIter}$ or error $<$ threshold
then stop
else $k = k + 1$; go to 3.

Stochastic gradient descent (1)

Complementary optimization of conclusions

- after optimal positioning of membership functions,
- and updating the corresponding conclusions by the off-line initialization algorithm.

But gradient descent algorithms don't guarantee the constraint:

$$\forall i, c_i \in [\min_{\mathbf{x}} d(\mathbf{x}), \max_{\mathbf{x}} d(\mathbf{x})]$$

\Rightarrow we have to introduce it

$$\left(\begin{array}{l} \text{Remember the stupid rule:} \\ \text{if } x \text{ is Zero then } x^2 = -0.11 \end{array} \right)$$

Stochastic gradient descent (2)

For each example (\mathbf{x}_p, d_p) , the conclusion of each rule is modified according to:

$$\Delta c_i = -\epsilon \frac{\partial E}{\partial c_i}$$

where $E = \frac{1}{2}(y(\mathbf{x}_p) - d_p)^2$ and ϵ is the learning rate.

Applying this method to the optimization of the rule conclusions, one obtains:

$$\Delta c_i = -\epsilon (y(\mathbf{x}_p) - d_p) \frac{\alpha_i(\mathbf{x}_p)}{\sum_{i=1}^N \alpha_i(\mathbf{x}_p)}$$

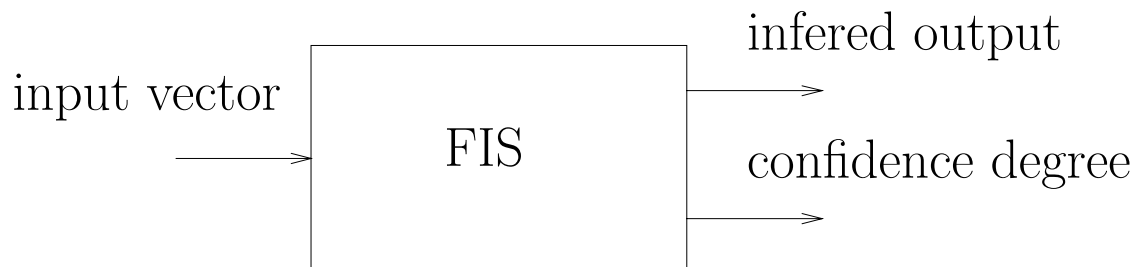
with the constraints

$$\begin{aligned} \text{if } c_i^{new} > \max_{\mathbf{x}} d(\mathbf{x}) & \text{ then } c_i^{new} = \max_{\mathbf{x}} d(\mathbf{x}) \\ \text{if } c_i^{new} < \min_{\mathbf{x}} d(\mathbf{x}) & \text{ then } c_i^{new} = \min_{\mathbf{x}} d(\mathbf{x}) \end{aligned}$$

Analysis of the learning stage

For each input vector, the FIS computes:

- the corresponding inferred output,
- **and** its confidence degree,



giving informations about both

- the quality of the learned rules,
- the quality of the learning set.

Confidence degree (1)

$val(i)$ (or $val(i, K)$) is a measure of the validity of c_i . For an input vector, \mathbf{x} , we can extend this validity measure to the FIS output by defining a confidence degree:

$$conf(\mathbf{x}) = \frac{\sum_{i=1}^N \alpha_i(\mathbf{x}) \times \alpha_i^{\max}}{\sum_j \alpha_j(\mathbf{x})} \in [0, 1]$$

The input space can be partially or totally covered by the training pairs:

$conf(\mathbf{x})$ is high $\Rightarrow \mathbf{x} \in$ a known part
 $conf(\mathbf{x})$ is low $\Rightarrow \mathbf{x} \in$ an unexplored part

Confidence degree (2)

	<i>conf</i>		
		low	high
error			
	low	1	3
	high	2	4

1. You are lucky !!
2. Normal situation due to a lack of information in this part of the input space.
3. Normal situation after learning.
4. Perhaps the sign of a missing variable ? Noise on input data ?

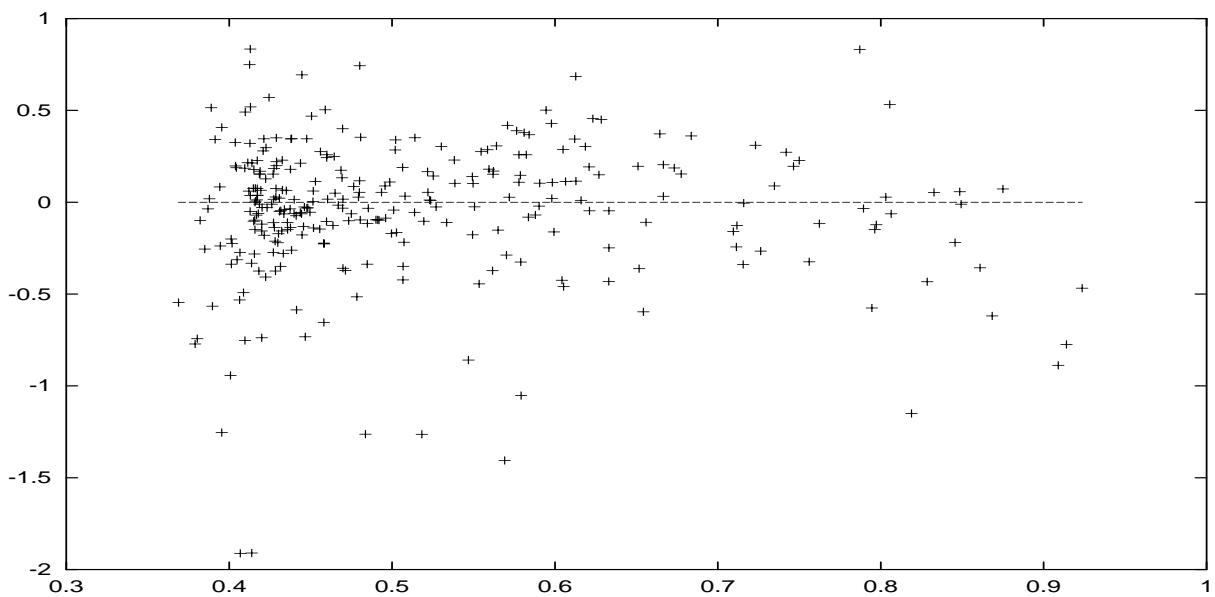
In case 2,

- impossible to increase the accuracy of the model without new available informations,
- the model may be excellent in other parts of the input space.

Confidence degree (3)

Plot of:

$$\text{error} = f(\text{confidence degree})$$



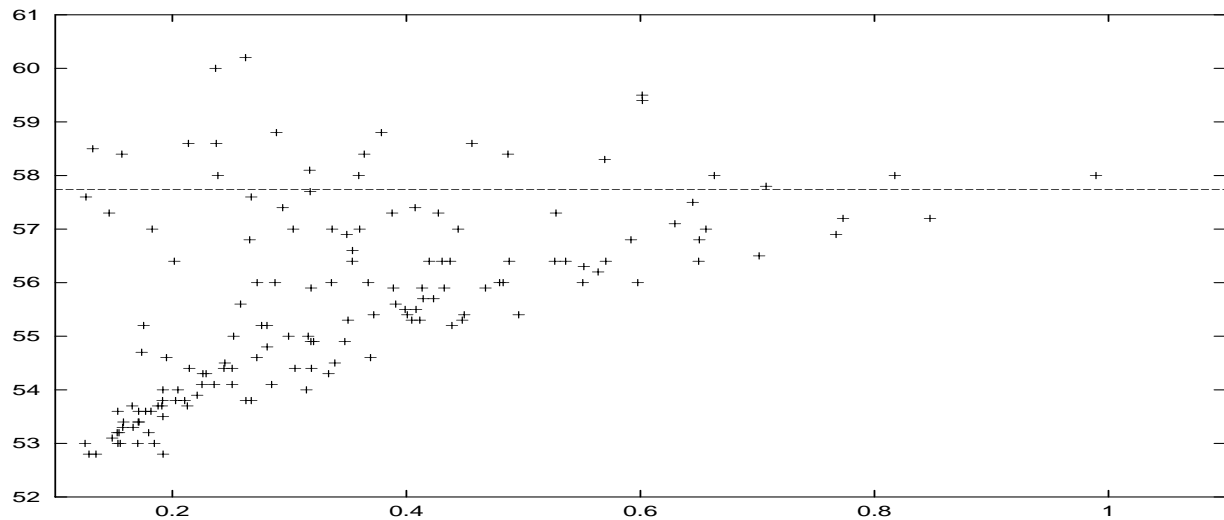
As we can see, the learning set contains questionable data:

- confidence degree more than 0.8
- and errors more than 0.5 (about 2%).

Rule by rule Analysis

Plot of:

$$d(\mathbf{x}) = f(\alpha_i(\mathbf{x}))$$



According to the **Theorem**:

$$\lim_{\alpha_i(\mathbf{x}) \rightarrow 1} d(\mathbf{x}) = c_i$$

where c_i is the conclusion of rule i and $(\mathbf{x}, d(\mathbf{x}))$ are training pairs.

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization
 - A simple heuristics

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization.

Structural optimization (1)

Tree-based heuristics

A node is a FIS characterized by

- the number of membership functions per input, m_i ,
- a performance index, Q (*e.g.* MSE).

m_1, \dots, m_n
$Q(m_1, \dots, m_n)$

Structural optimization (2)

Algorithm for a FIS with n inputs

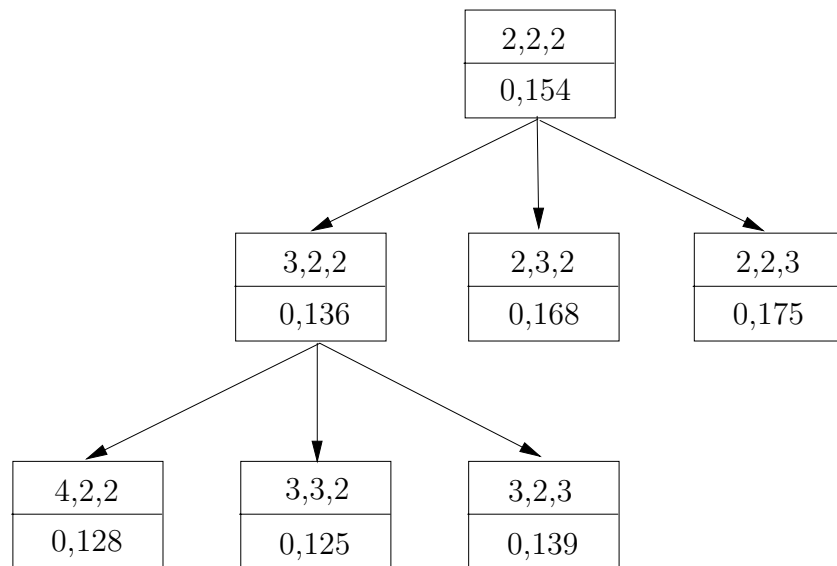
1. Begin with the minimal structure ($m_i = 2$) and compute $Q(m_1, \dots, m_n)$.
2. Build n nodes by adding only one membership function for each input and compute $Q(m_1 + 1, m_2, \dots, m_n), \dots$
 $\dots, Q(m_1, \dots, m_{n-1}, m_n + 1)$.
3. Select the node with the best Q value, ignore the others and go to step 2 or end.

Complexity

$$n \times \text{depth} + 1$$

Structure optimization (3)

Example using the gas furnace data
(Box and Jenkins)



Conclusion

We have proposed different algorithms

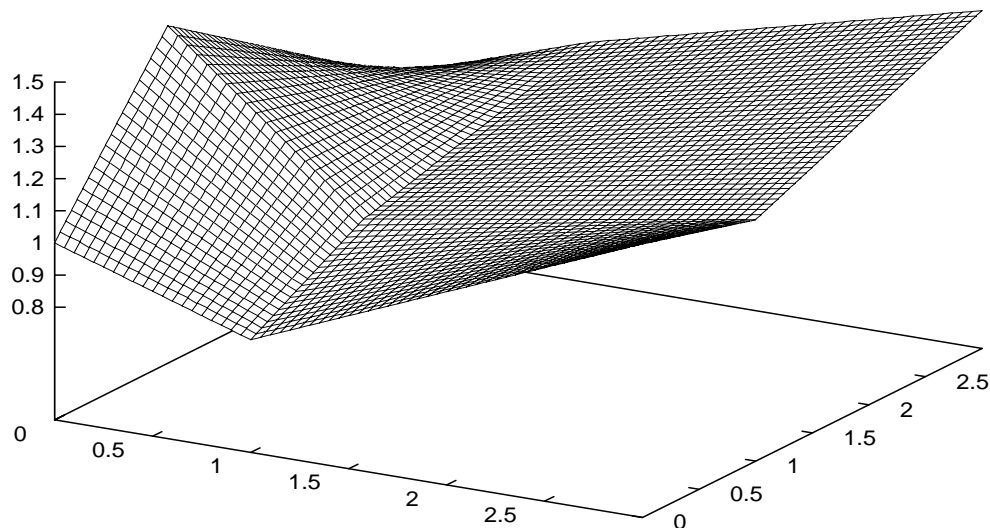
- very simple and efficient,
- allowing both interpretability and accuracy,
- with easy programming on μ -controllers.

Small is beautiful

Small is beautiful !! (1)

A triangle is as good as a Gaussian function...

- Very simple computation of membership degrees.
- Easy coding on μ -controller.
- Reasonably smooth output surfaces



Small is beautiful !! (2)

What is the optimization problem:

- rule conclusions ?
- positioning membership functions ?
- structure ?

⇒ *Three sub-problems*

⇒ *Choose the right algorithms !!*

Small is beautiful !! (3)

Example of C code for on-line learning without any initial knowledge.

Data Structures

trVal[j] : truth value for rule j
Concl[j] : conclusion of rule j (init = 0)
trValMax[j] : maximum truth value (init = 0)
des : FIS desired output for a given input vector

Algorithm

```
for(j=0;j<NbRule;j++)
  if(trVal[j] > 0.0)
  {
    if((trVal[j] > trValMax[j])||(trVal[j] > 0.9))
      Concl[j] = (trValMax[j]*Concl[j] + trVal[j]*des)
                /(trValMax[j] + trVal[j]);
    if(trVal[j] > trValMax[j])
      trValMax[j] = trVal[j];
  }
```

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations
 - Crisp and Fuzzy Trees
 - FIS and Fuzzy Trees
2. Building a FRT.
3. FRT Optimization.

Decision Tree Advantages

- Great similarity with Human reasoning.
- Immediate interpretability and readability.
- Large dimensional input spaces:
 - up to 100 input variables,
 - but automatic selection of the more informative ones.
- Short computational time, for both building and optimization.

Crisp Decision Trees (1)

Three linked problems when dealing with continuous variables.

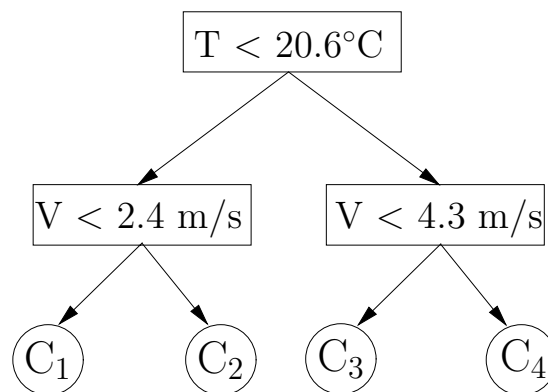
- Lack of genericity of Boolean tests: $x < a$, where a is a given number.
- Instability (due to the threshold effects).
- Sensitiveness to learning data.

Crisp Decision Trees (2)

Small difference on inputs

⇒ possible strong influence on the result !!

example



	T	
V \	20.5°C	20.7°C
2.2 m/s	C ₁	C ₃
2.4 m/s	C ₂	C ₃

Decision and Regression Trees

- Decision Tree
 - Each leaf gives a class number,
 - tool for classification problems.
- Regression Tree
 - Continuous value at each leaf,
 - generalization of decision trees,
 - Tool for function approximation.
- Fuzzy Regression Trees (FRT): Extension of advantages of both
 - crisp decision trees,
 - Fuzzy Inference Systems.

FRT and FIS

FIS

- Very easy to use and optimize,
- but combinational explosion with too many inputs.

⇒ better for less than 5 or 6 inputs.

FRT

- hierarchization of inputs variables,
- information gain,
- incomplete rules.

⇒ larger input states.

FRT Advantages

Combination of advantages.

FIS

- Interpretable fuzzy rules,
- re-use of most optimization algorithms.

Crisp Decision Trees

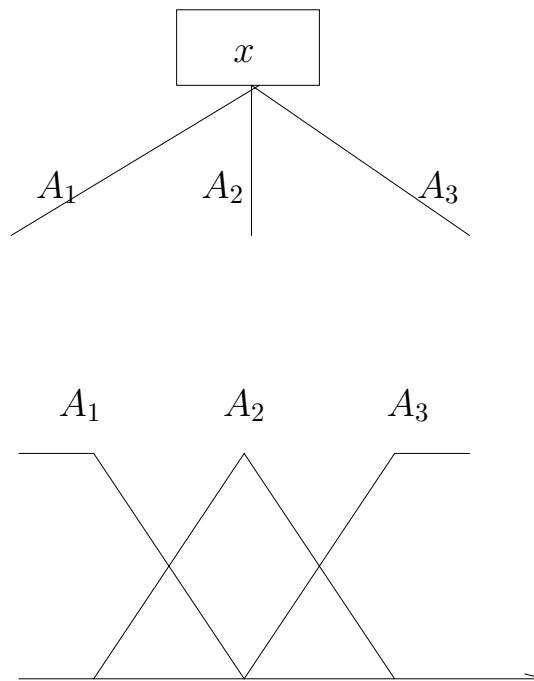
- Better management of continuous variables,
- re-use of the tree-building process.

Chosen structure

Extension of ID3

- One variable is evaluated at each node.
- Number of childs = number of fuzzy labels.

⇒ *interpretation and readability very easy*



Inference

We have chosen a Takagi-Sugeno reasoning scheme.
Let:

- $\mathbf{x} = (x_1, \dots, x_n)^t$: input vector.
- $\alpha_L(\mathbf{x})$: truth value of leaf L for a given \mathbf{x} .
- C_L : value at leaf L .

Inferred output:

$$FRT(\mathbf{x}) = \frac{\sum_L \alpha_L(\mathbf{x}) \times C_L}{\sum_L \alpha_L(\mathbf{x})}$$

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT
 - Information gain
 - Induction from data
3. FRT Optimization.

Notations (1)

- $\mathbf{x}_i = (x_{i1}, \dots, x_{iM})$: input vector, each component, x_{ij} , having fuzzy modalities.
- $\hat{\mu}_k(\mathbf{x}_i)$: membership degree of \mathbf{x}_i to (crisp or fuzzy) k class.
- $\mu_A(x_{ij})$: membership degree of x_{ij} fuzzy set A (sometimes written $\mu_A(\mathbf{x}_i)$ for simplicity).
- $\alpha_N(\mathbf{x}_i)$: membership degree of \mathbf{x}_i to node N .

Notations (2)

Learning set

$$E = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_P, y_P)\}$$

where \mathbf{x}_i is an input vector and y_i the corresponding desired value.

Classification problem :

y_i corresponds to the class number.

Regression problem :

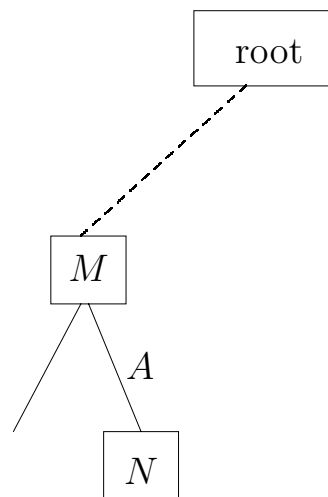
A strong fuzzy partition with K modalities is defined on the output space; we define:

$$\hat{\mu}_k(\mathbf{x}_i) = \mu_k(y_i), \quad k = 1 \text{ to } K$$

Membership to a node

Membership of input vector \mathbf{x} to node N

$$\alpha_N(\mathbf{x}) = \begin{cases} 1 & \text{if } N \text{ is the root} \\ \alpha_M(\mathbf{x}) \wedge \mu_A(\mathbf{x}) & \text{else} \end{cases}$$



Information gain (1)

Representation of class k at node N for modality A_j of the considered variable.

$$r(k, j, N) = \sum_{i=1}^P \hat{\mu}_k(\mathbf{x}_i) \wedge \mu_{A_j}(\mathbf{x}_i) \wedge \alpha_N(\mathbf{x}_i)$$

Parameters p_k et w_j used for information gain computation:

$$p_k = \frac{\sum_j r(k, j, N)}{\sum_k \sum_j r(k, j, N)} = \frac{\sum_j r(k, j, N)}{\sum_{i=1}^P \alpha_N(\mathbf{x}_i)}$$

$$w_j = \frac{\sum_k r(k, j, N)}{\sum_k \sum_j r(k, j, N)} = \frac{\sum_k r(k, j, N)}{\sum_{i=1}^P \alpha_N(\mathbf{x}_i)}$$

Information gain (2)

Gain of X at node N

$$\begin{aligned} G(X, N) &= I(N) - \text{Info}(X, N) \\ &= -\sum_k p_k \log(p_k) - \sum_j w_j I(X_j) \end{aligned}$$

where

$$I(X_j) = -\sum_k \frac{r(k, j, N)}{\sum_k r(k, j, N)} \log\left(\frac{r(k, j, N)}{\sum_k r(k, j, N)}\right)$$

Extracting a FRT from data

1. At each node:

- choose the variable with the best information gain,
- expand the node, according to the number of fuzzy sets,
- compute the leaves.

2. Stopping criteria:

- the MSE on the test set does not decrease,
- the leaves are (almost) pure.

Value at a leaf (1)

The same initialization method that for FIS !

Algorithm

```
pour for each leaf  $L$  do  
    find  $(\mathbf{x}^*, y^*)$  in  $E$  such that :  
         $\alpha_L(\mathbf{x}^*) = \max_{\mathbf{x} \in E} \alpha_L(\mathbf{x})$   
     $c_L \leftarrow y^*$   
done
```

Value at a leaf (2)

Value at a leaf:

$$c_F = \frac{\sum_{j \in J} \alpha_F(\mathbf{x}_j) y_j}{\sum_{j \in J} \alpha_F(\mathbf{x}_j)}$$

J : set of training pairs (\mathbf{x}_j, y_j) with truth value more than a given threshold.

Outline

Fuzzy Inference Systems

1. General Presentation.
2. Constrained Optimization.
3. Parameter Optimization.
4. Structural Optimization.

Fuzzy Regression/Decision Trees

1. Motivations.
2. Building a FRT.
3. FRT Optimization

Interpretability

FRT and FIS: the same constraints for interpretability

- Fuzzy sets shared by the rules.
- Prototypes: each rule can be examined separately.
- Linguistic Order Relation all along the optimization process.

⇒ A sufficient condition: the strong fuzzy partitions.

FRT Optimisation (1)

Structure

Number of membership functions by input variable.

Parameters

- Placement of membership functions on each input domain,
- values at the leaves (rule conclusions).

⇒ These choices have a strong influence on the tree performances.

FRT Optimisation (2)

Proposed Method

Principle

Optimization of fuzzy partitions in order to maximize the **information gain** for each variable.

Advantages

- Variable by variable optimization.
- Reduced search spaces.

Implementation (1)

The user has to fix:

1. the variation domain of each input variable (min and max),
2. the initial number of fuzzy labels
⇒ variation domain divided into equal parts
3. a learning and a test set,
4. **and it is (almost) all...**

Implementation (2)

Structural optimization:

1. user driven induction, according to:
 - the information gains,
 - the purity degree at a leaf,
 - the MSE for learning and test sets.
2. automatic moving of membership functions using Solis and Wetts's algorithm,
 - with the initial number of m.f. (given by the user): m
 - with one supplementary m.f. per input: $m + 1$.
3. the user makes the final choice of m.f. (m or $m + 1$), according to the corresponding variations of information gains.

WOMBAT

Wombat: a toolbox to build FRT.

1. Scilab environment:

- a **free** scientific software package,
- `http://scilabsoft.inria.fr`.

2. Graphical user interface

- Definition of fuzzy partitions
- FRT induction.

3. Export

- C code of the induced tree
- Figures.

General Conclusion

Automatic rule extraction for FIS and FRT

1. Interpretability is guaranteed.
2. On-line and off-line supervised learning.
3. Appropriate algorithms.
4. Confidence degree.

⇒ function approximation, prediction, diagnosis, classification...

⇒ **All the domains where human expertise and knowledge can be useful and/or needed !!**