

Adaptive Neural Fuzzy Inference Systems (ANFIS): Analysis and Applications

Outline

- Objective
- Fuzzy Control
 - Background, Technology & Typology
- ANFIS:
 - as a Type III Fuzzy Control
 - as a fuzzification of CART
 - Characteristics
 - Pros and Cons
 - Opportunities
 - Applications
 - References

ANFIS Objective

- To integrate the best features of Fuzzy Systems and Neural Networks:
 - From FS: Representation of prior knowledge into a set of constraints (network topology) to reduce the optimization search space
 - From NN: Adaptation of backpropagation to structured network to automate FC parametric tuning
- ANFIS application to synthesize:
 - controllers (automated FC tuning)
 - models (to explain past data and predict future behavior)

FC Technology & Typology

- Fuzzy Control
 - A high level representation language with local semantics and an interpreter/compiler to synthesize non-linear (control) surfaces
 - A Universal Functional Approximator
- FC Types
 - Type I: RHS is a monotonic function
 - Type II: RHS is a fuzzy set
 - Type III: RHS is a (linear) function of state

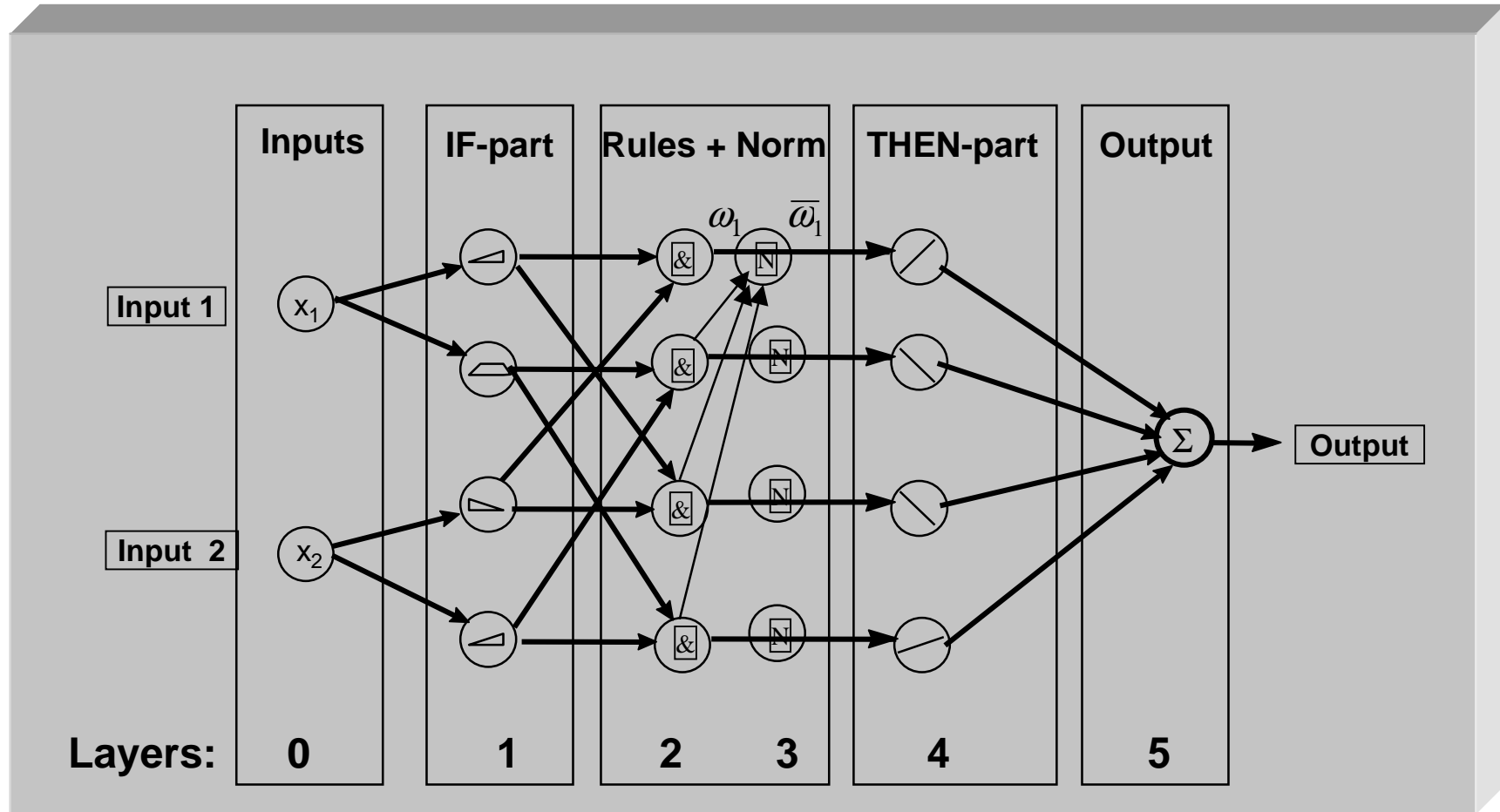
FC Technology (Background)

- Fuzzy KB representation
 - Scaling factors, Termsets, Rules
- Rule inference (generalized *modus ponens*)
- Development & Deployment
 - Interpreters, Tuners, Compilers, Run-time
 - Synthesis of control surface
- FC Types I, II, III

FC of Type II, III, and ANFIS

- Type II Fuzzy Control must be tuned manually
- Type III Fuzzy Control (Takagi-Sugeno type) have an automatic Right Hand Side (RHS) tuning
- ANFIS will provide both:
 - RHS tuning, by implementing the TSK controller as a network
 - and LHS tuning, by using back-propagation

ANFIS Network



ANFIS Neurons: Clarification note

- Note that neurons in ANFIS have different structures:
 - Values [Membership function defined by parameterized soft trapezoids (Generalized Bell Functions)]
 - Rules [Differentiable T-norm - usually product]
 - Normalization [Sum and arithmetic division]
 - Functions [Linear regressions and multiplication with $\bar{\omega}$, i.e., normalized weights ω ,]
 - Output [Algebraic Sum]

ANFIS as a generalization of CART

- Classification and Regression Tree (CART)
 - Algorithm defined by Breiman et al in 1984
 - Creates a binary decision tree to classify the data into one of 2^n linear regression models to minimize the Gini index for the current node c :

$$\text{Gini}(c) = 1 - \sum_j p_j^2$$

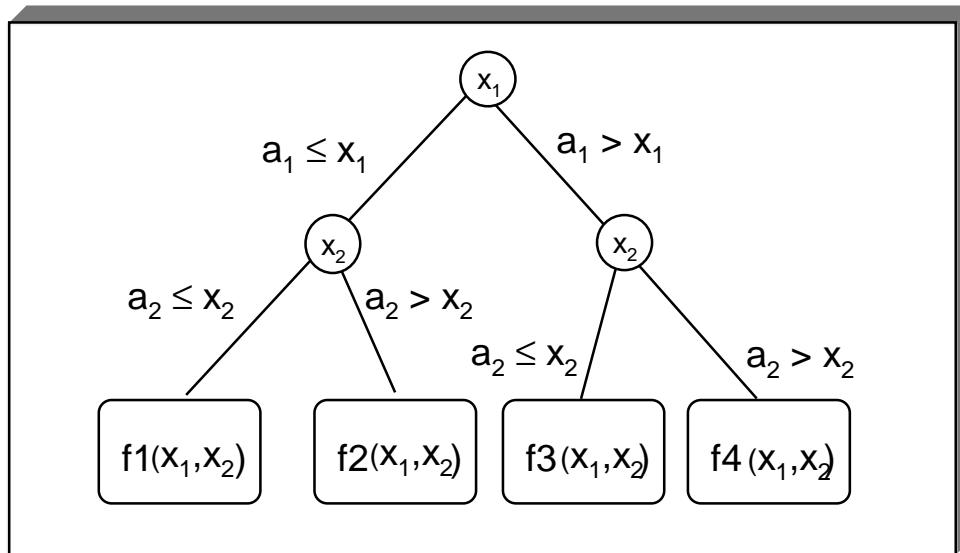
where:

- p_j is the probability of class j in node c
- $\text{Gini}(c)$ measure the amount of “impurity” (incorrect classification) in node c

CART Problems

- Discontinuity
- Lack of locality (sign of coefficients)

CART: Binary Partition Tree and Rule Table Representation

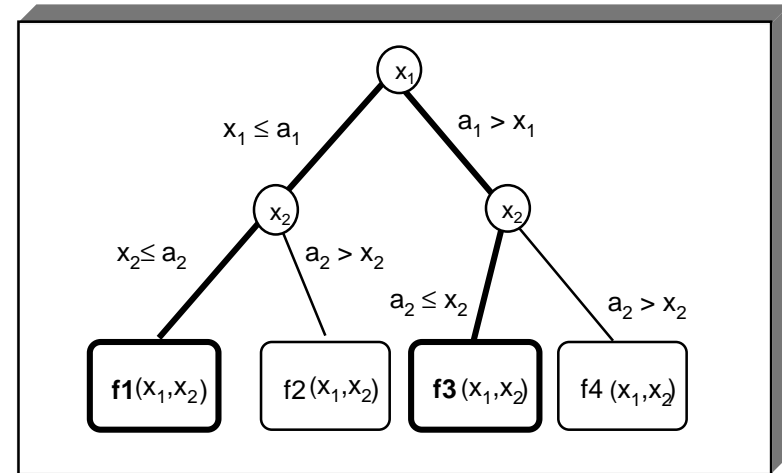
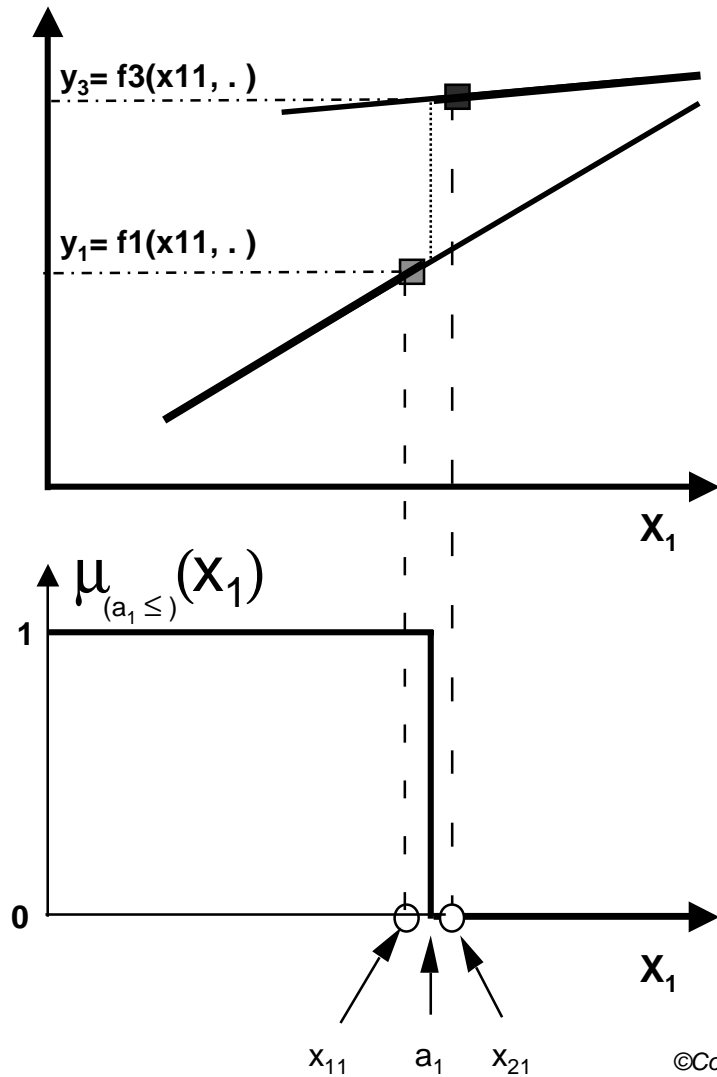


Partition Tree

x_1	x_2	y
$a_1 \leq$	$a_2 \leq$	$f1(x_1, x_2)$
$a_1 \leq$	$> a_2$	$f2(x_1, x_2)$
$a_1 >$	$a_2 \leq$	$f3(x_1, x_2)$
$a_1 >$	$> a_2$	$f4(x_1, x_2)$

Rule Table

Discontinuities Due to Small Input Perturbations



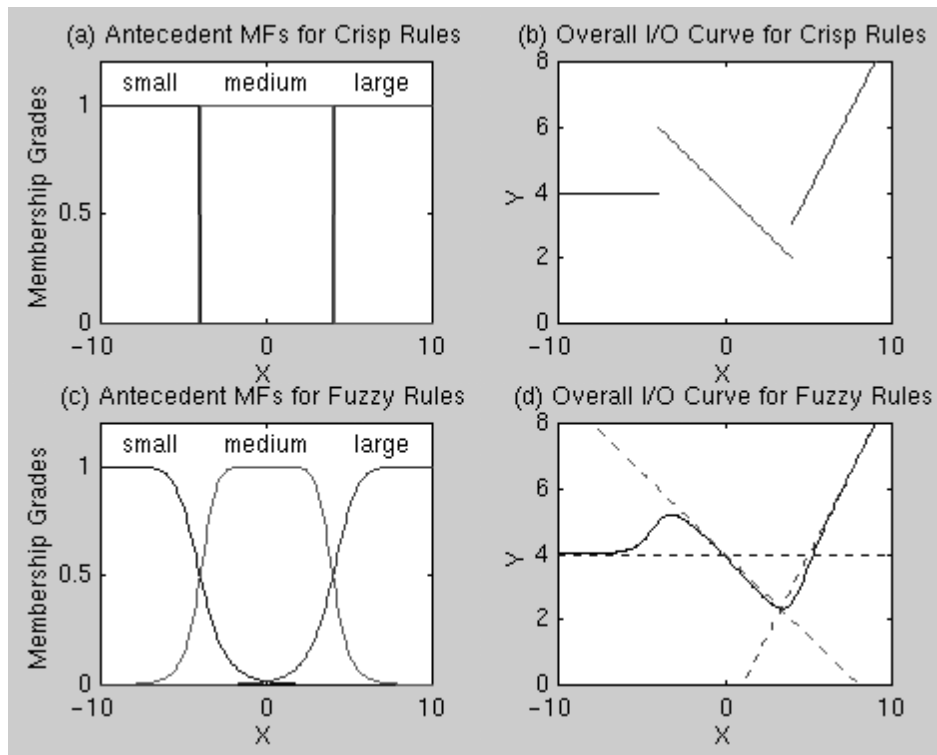
Let's assume two inputs: $I1=(x_{11},x_{12})$, and $I2=(x_{21},x_{22})$ such that:

- $x_{11} = (a_1 - \epsilon)$
- $x_{21} = (a_1 + \epsilon)$
- $x_{12} = x_{22} < a_2$

Then **I1 is assigned $f_1(x_{11},x_{12})$** while **I2 is assigned $f_3(x_1,x_2)$**

Takagi-Sugeno (TS) Model

- Combines fuzzy sets in antecedents with crisp function in output:
- IF (x_1 is A) AND (x_2 is B) THEN $y = f(x_1, x_2)$



IF X is *small*
THEN $Y_1=4$
IF X is *medium*
THEN $Y_2=-0.5X+4$
IF X is *large*
THEN $Y_3=X-1$

$$Y = \frac{\sum_{j=1}^n Y_j w_j}{\sum_{j=1}^n w_j}$$

ANFIS Characteristics

- Adaptive Neural Fuzzy Inference System (ANFIS)
 - Algorithm defined by J.-S. Roger Jang in 1992
 - Creates a fuzzy decision tree to classify the data into one of 2^n (or p^n) linear regression models to minimize the sum of squared errors (SSE):

$$SSE = \sum_j e_j^2$$

where:

- e_j is the error between the desired and the actual output
- p is the number of fuzzy partitions of each variable
- n is the number of input variables

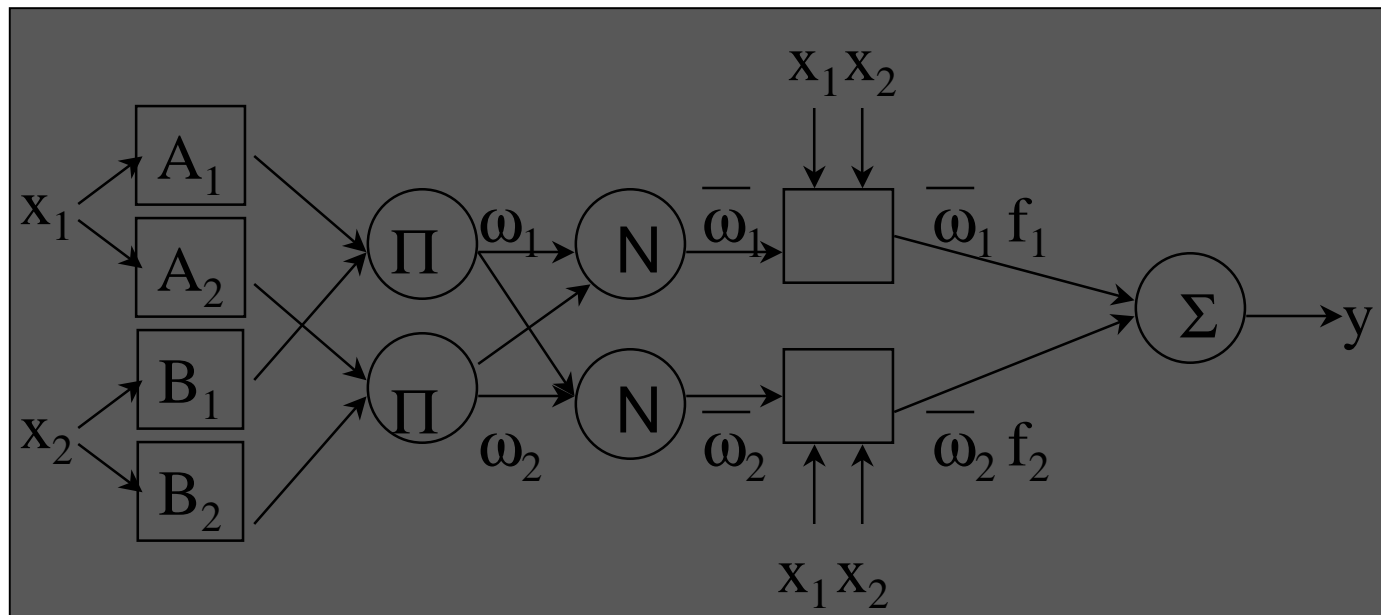
ANFIS as a Type III FC

- L_0 : State variables are nodes in ANFIS *inputs layer*
- L_1 : Termsets of each state variable are nodes in ANFIS *values layer*, computing the membership value
- L_2 : Each rule in FC is a node in ANFIS *rules layer* using soft-min or product to compute the rule matching factor ω_i
- L_3 : Each ω_i is scaled into $\bar{\omega}_i$ in the *normalization layer*
- L_4 : Each $\bar{\omega}_i$ weighs the result of its linear regression f_i in the *function layer*, generating the rule output
- L_5 : Each rule output is added in the *output layer*

ANFIS Architecture

Rule Set:

IF (x_1 is A_1) AND (x_2 is B_1) THEN $f_1 = p_1 x_1 + q_1 x_2 + r_1$
 IF (x_1 is A_2) AND (x_2 is B_2) THEN $f_2 = p_2 x_1 + q_2 x_2 + r_2$
 ...

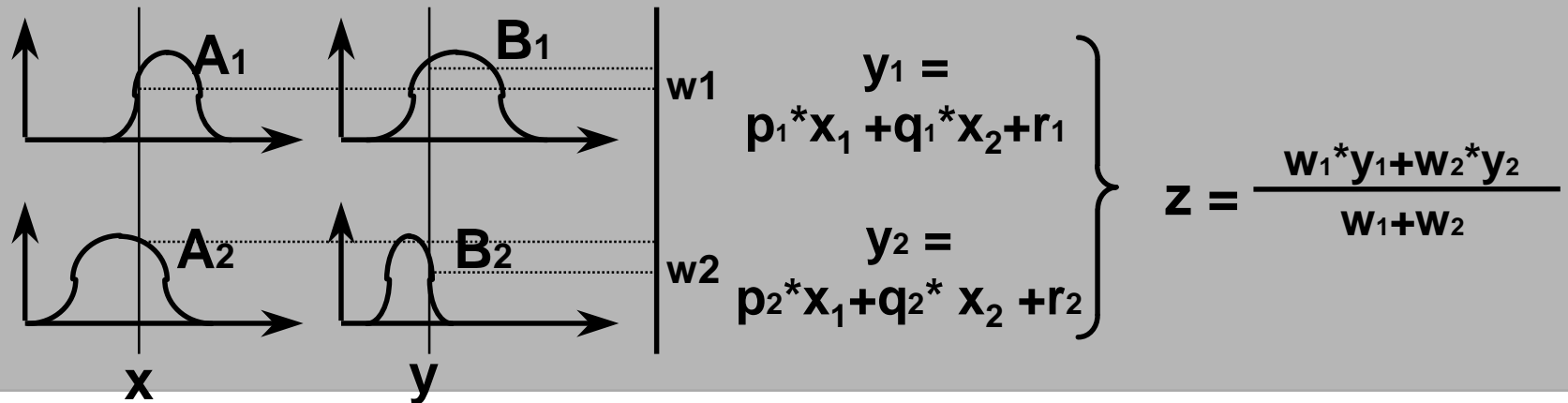


Layers: 0 1 2 3 4 5

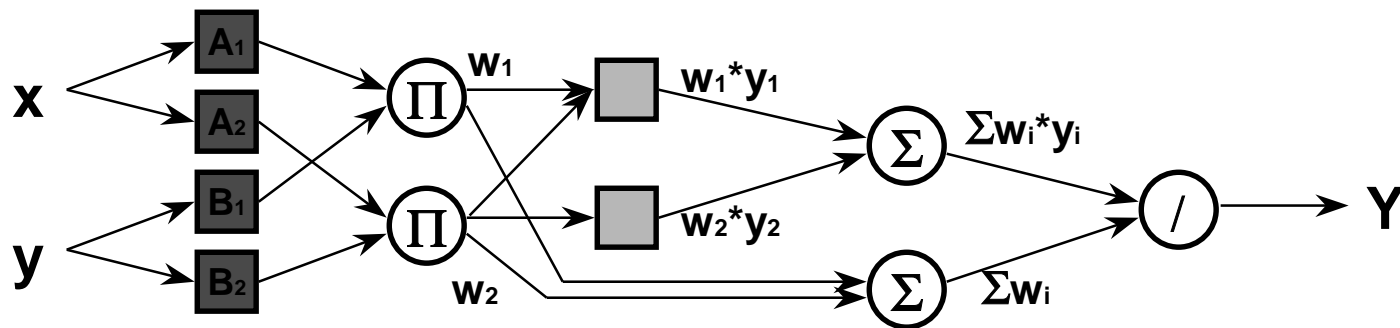
ANFIS-Visualized (Example for n =2)

where A_1 : Medium; A_2 : Small-Medium
 B_1 : Medium; B_2 : Small-Medium

- Fuzzy reasoning



- ANFIS (Adaptive Neuro-Fuzzy Inference System)



Layer 1: Calculate Membership Value for Premise Parameter

- Output $O_{1,i}$ for node $i=1,2$

$$O_{1,i} = \mu_{A_i}(x_1)$$

- Output $O_{1,i}$ for node $i=3,4$

$$O_{1,i} = \mu_{B_{i-2}}(x_2)$$

- where

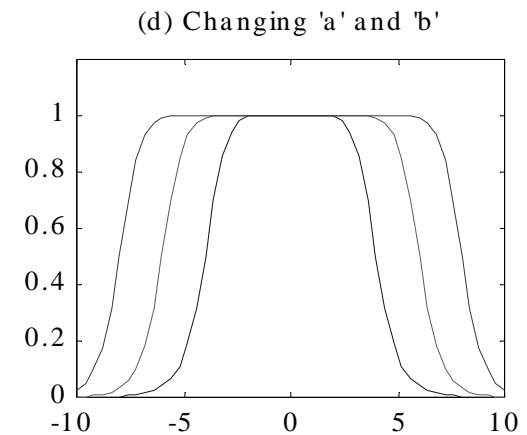
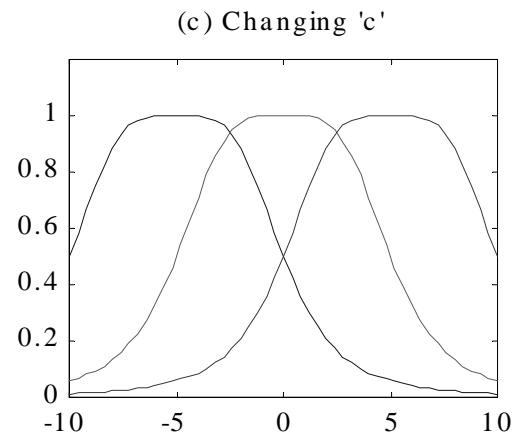
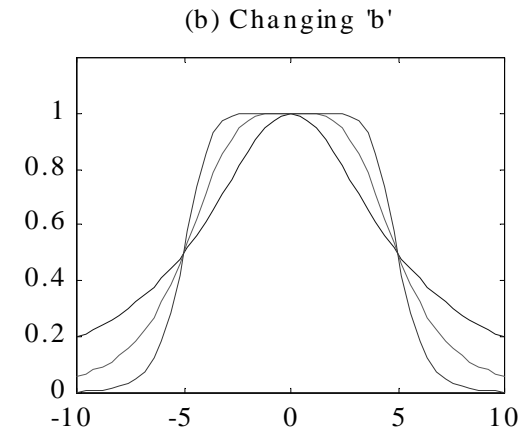
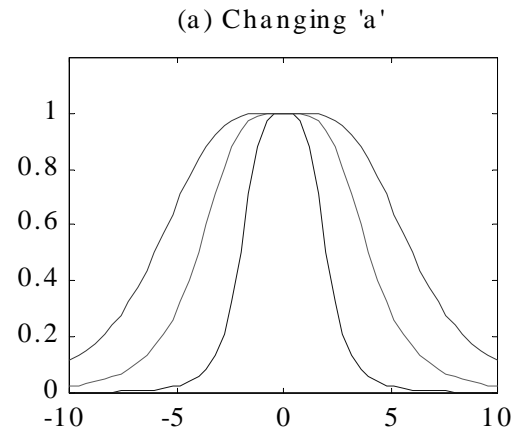
A is a linguistic label (small, large, ...)

$$\mu_A(x_1) = \frac{1}{1 + \left| \frac{x_1 - c_i}{a_i} \right|^{2b}}$$

Node output: *membership value of input*

Layer 1 (cont.): Effect of changing Parameters {a,b,c}

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



Layer 2: Firing Strength of Rule

- Use T-norm (min, product, fuzzy AND, ...)

$$O_{2,i} = w_i = \mu_{A_i}(x_1) \mu_{B_i}(x_2)$$

(for $i=1,2$)

Node output: *firing strength of rule*

Layer 3: Normalize Firing Strength

- Ratio of i^{th} rule's firing strength vs. all rules' firing strength

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}$$

(for $i=1,2$)

Node output: *Normalized firing strengths*

Layer 4: Consequent Parameters

- Takagi-Sugeno type output

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x_1 + q_i x_2 + r_i)$$

- Consequent parameters $\{p_i, q_i, r_i\}$

Node output: *Evaluation of Right Hand Side Polynomials*

Layer 5: Overall Output

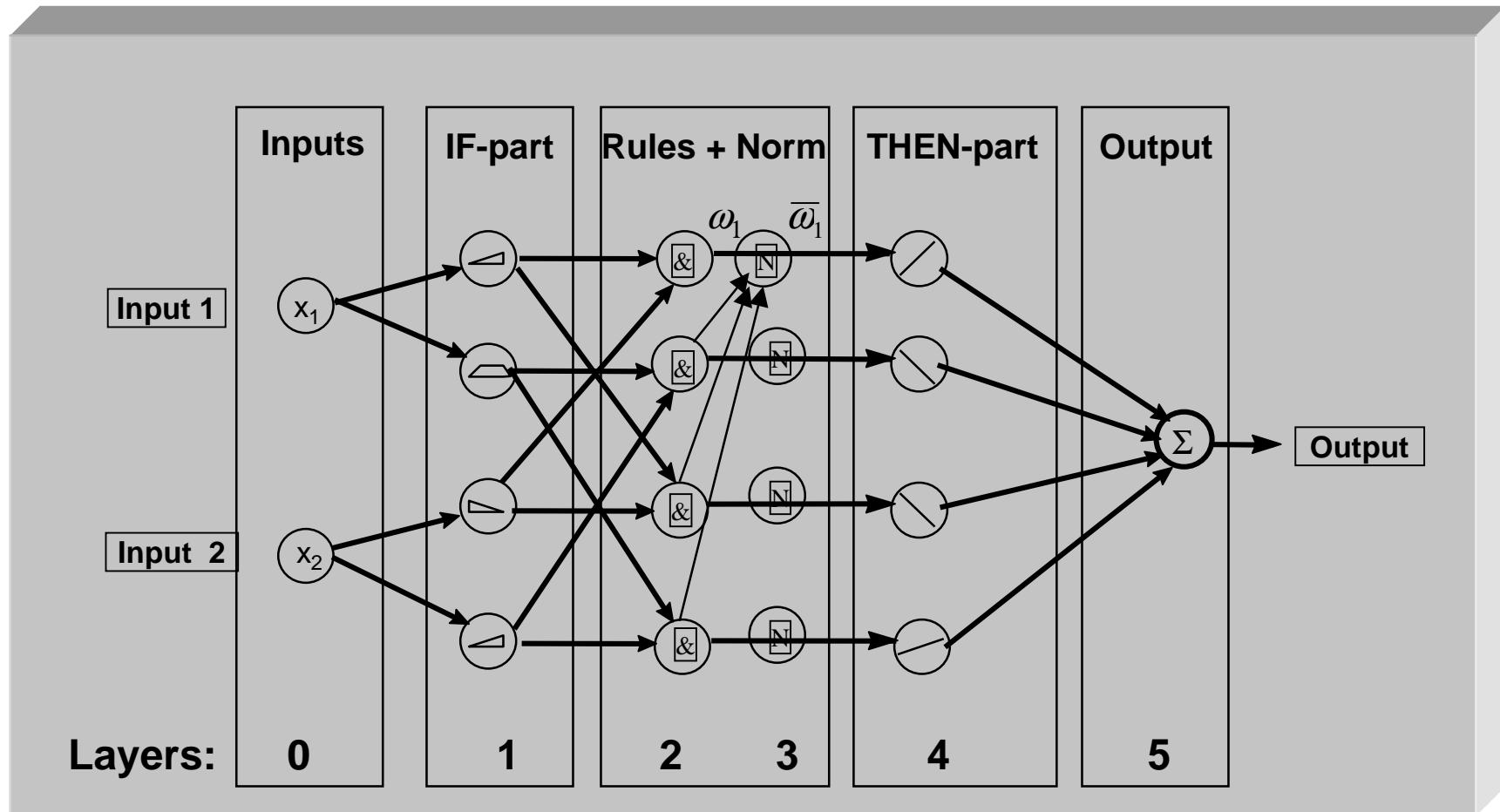
$$O_{5,1} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

- Note:
 - Output is linear in consequent parameters p, q, r .

$$\begin{aligned} &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \bar{w}_1 (p_1 x_1 + q_1 x_2 + r_1) + \bar{w}_2 (p_2 x_1 + q_2 x_2 + r_2) \\ &= (\bar{w}_1 x_1) p_1 + (\bar{w}_1 x_2) q_1 + (\bar{w}_1) r_1 + (\bar{w}_2 x_1) p_2 + (\bar{w}_2 x_2) q_2 + (\bar{w}_2) r_2 \end{aligned}$$

Node output: *Weighted Evaluation of RHS Polynomials*

ANFIS Network



ANFIS Computational Complexity

<u>Layer #</u>	<u>L-Type</u>	<u># Nodes</u>	<u># Param</u>
L ₀	Inputs	n	0
L ₁	Values	(p•n)	3•(p•n)= S1
L ₂	Rules	p ⁿ	0
L ₃	Normalize	p ⁿ	0
L ₄	Lin. Funct.	p ⁿ	(n+1)•p ⁿ = S2
L ₅	Sum	1	0

ANFIS Parametric Representation

- ANFIS uses two sets of parameters: S1 and S2
 - S1 represents the fuzzy partitions used in the rules LHS

$$S1 = \left\{ \left\{ a_{11}, b_{11}, c_{11} \right\}, \left\{ a_{12}, b_{12}, c_{12} \right\}, \dots, \left\{ a_{1p}, b_{1p}, c_{1p} \right\}, \dots, \left\{ a_{np}, b_{np}, c_{np} \right\} \right\}$$

- S2 represents the coefficients of the linear functions in the rules RHS

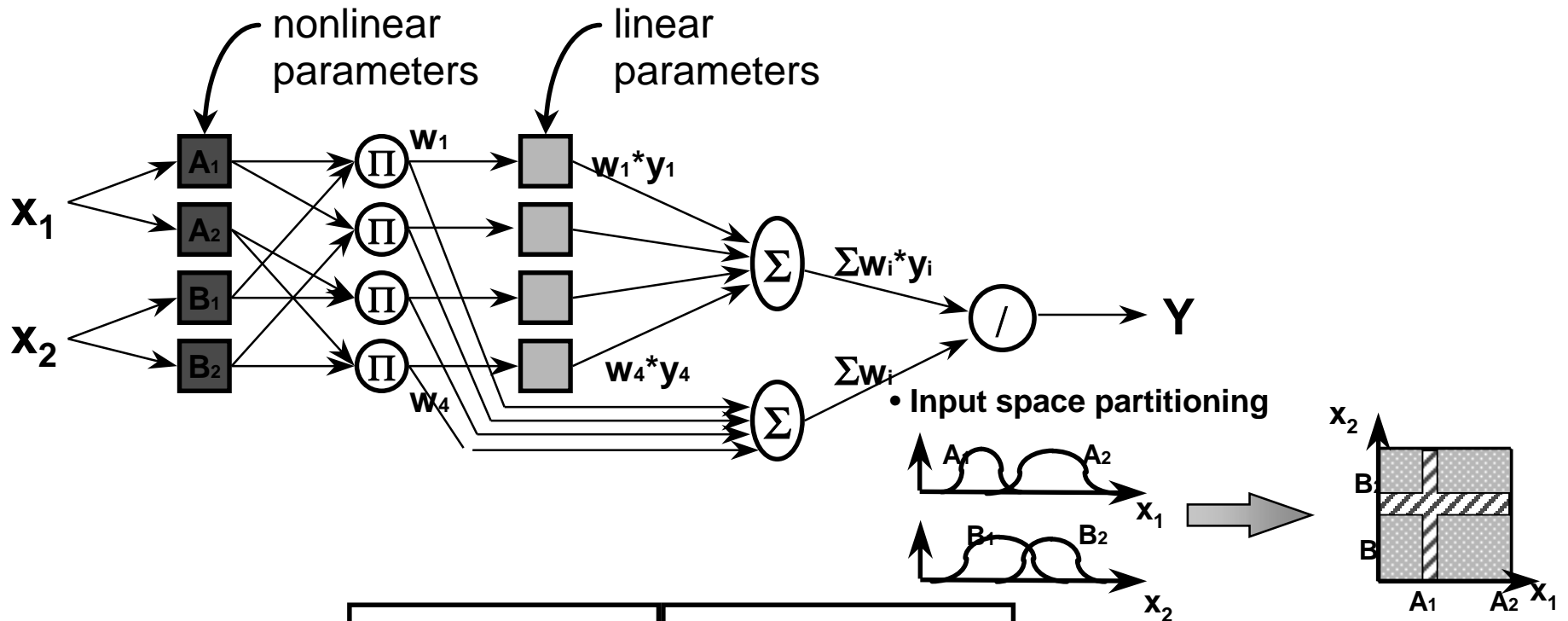
$$S2 = \left\{ \left\{ c_{10}, c_{11}, \dots, c_{1n} \right\}, \dots, \left\{ c_{p^0}, c_{p^1}, \dots, c_{p^n} \right\} \right\}$$

ANFIS Learning Algorithms

- ANFIS uses a two-pass learning cycle
 - Forward pass:
 - S1 is fixed and S2 is computed using a Least Squared Error (LSE) algorithm (Off-line Learning)
 - Backward pass:
 - S2 is fixed and S1 is computed using a gradient descent algorithm (usually Back-propagation)

Structure ID & Parameter ID

Hybrid training method



	Forward stroke	Backward stroke
MF param. (nonlinear)	fixed	steepest descent
Coef. param. (linear)	least-squares	fixed

ANFIS Least Squares (LSE) Batch Algorithm

- LSE used in Forward Stroke:
 - Parameter Set: $S = \{S1 \cup S2\}$, and $\{S1 \cap S2 = \emptyset\}$
Output = $F(\bar{I}, S)$, where \bar{I} is the input vector
 $H(\text{Output}) = H \circ F(\bar{I}, S)$, where $H \circ F$ is linear in $S2$
 - For given values of $S1$, using K training data, we can transform the above equation into **$B=AX$** , where X contains the elements of $S2$
 - This is solved by: **$(A^T A)^{-1} A^T B = X^*$** where **$(A^T A)^{-1} A^T$** is the pseudo-inverse of A (if **$A^T A$** is nonsingular)
 - The LSE minimizes the error $\|AX-B\|^2$ by approximating X with X^*

ANFIS LSE Batch Algorithm (cont.)

- Rather than solving directly $(A^T A)^{-1} A^T B = X^*$, we resolve it iteratively (from numerical methods):

$$\left. \begin{aligned}
 S_{i+1} &= S_i - \frac{S_i a_{(i+1)} a_{(i+1)}^T S_i}{1 + a_{(i+1)}^T S_i a_{(i+1)}}, \\
 X_{i+1} &= X_i + S_{(i+1)} a_{(i+1)} (b_{(i+1)}^T - a_{(i+1)}^T X_i)
 \end{aligned} \right\} \text{for } i = 0, 1, \dots, K-1$$

where:

$$\begin{aligned}
 X_0 &= \bar{0}, \\
 S_0 &= \mathcal{M}, (\text{where } \gamma \text{ is a large number}) \\
 a_i^T &= i^{\text{th}} \text{ line of matrix } A \\
 b_i^T &= i^{\text{th}} \text{ element of vector } B \\
 X^* &= X_k
 \end{aligned}$$

ANFIS Back-propagation

- Error measure E_k
(for the k^{th} ($1 \leq k \leq K$) entry of the training data)

$$E_k = \sum_{i=1}^{N(L)} (d_i - x_{L,i})^2$$

where :

$N(L)$ = number nodes in layer L

d_i = i^{th} component of *desired* output vector

$x_{L,i}$ = i^{th} component of *actual* output vector

- Overall error measure E :

$$E = \sum_{k=1}^K E_k$$

ANFIS Back-propagation (cont.)

- For each parameter α_i the update formula is:

$$\Delta \alpha_i = -\eta \frac{\partial^+ E}{\partial \alpha_i}$$

where :

$$\eta = \frac{\kappa}{\sqrt{\sum_i \left(\frac{\partial E}{\partial \alpha_i} \right)^2}} \quad \text{is the learning rate}$$

κ is the step size

$\frac{\partial^+ E}{\partial \alpha_i}$ is the ordered derivative

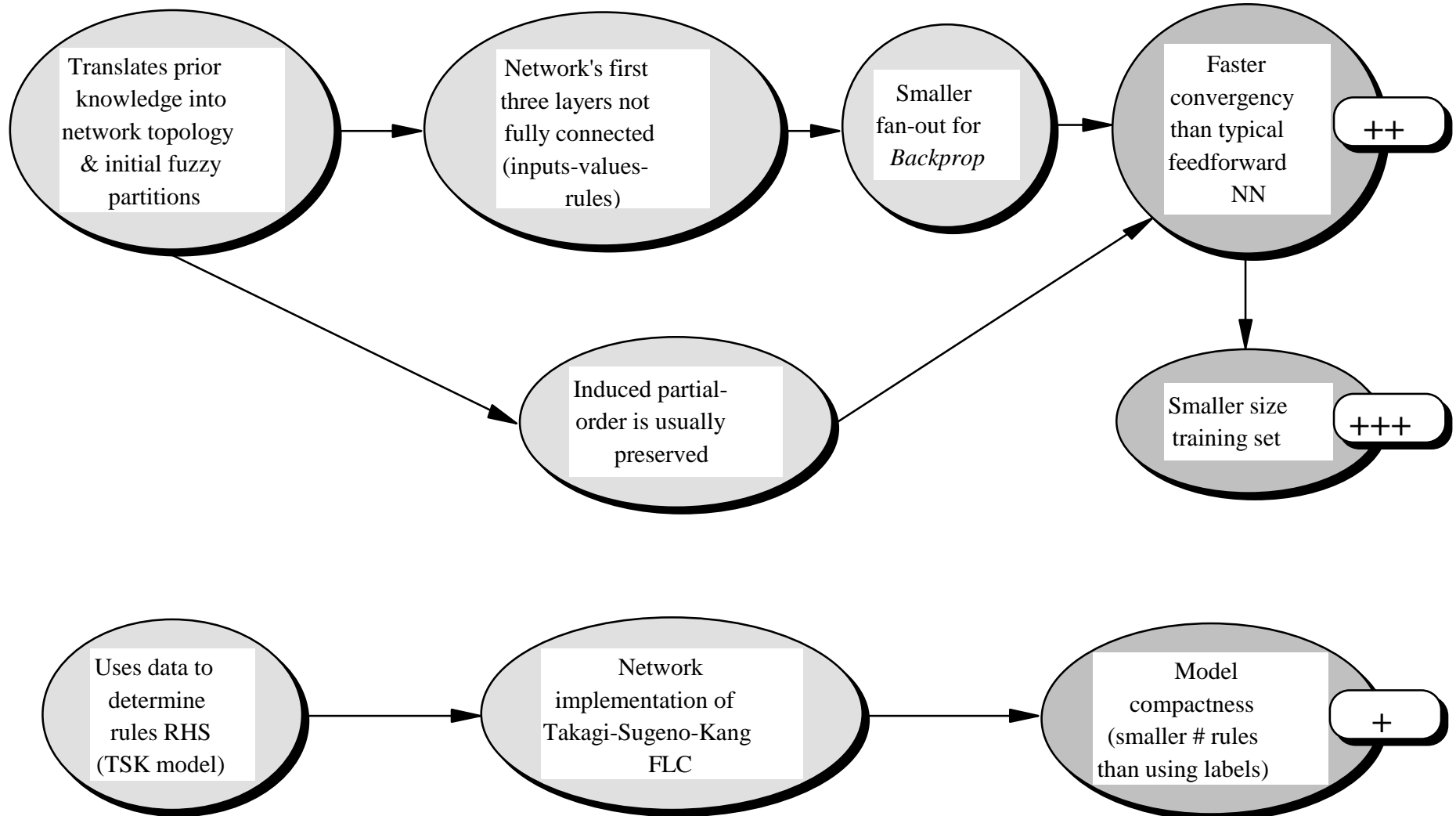
ANFIS Pros and Cons

- ANFIS is one of the best tradeoff between neural and fuzzy systems, providing:
 - *smoothness*, due to the FC interpolation
 - *adaptability*, due to the NN Backpropagation
- ANFIS however has strong computational complexity restrictions

ANFIS Pros

Characteristics

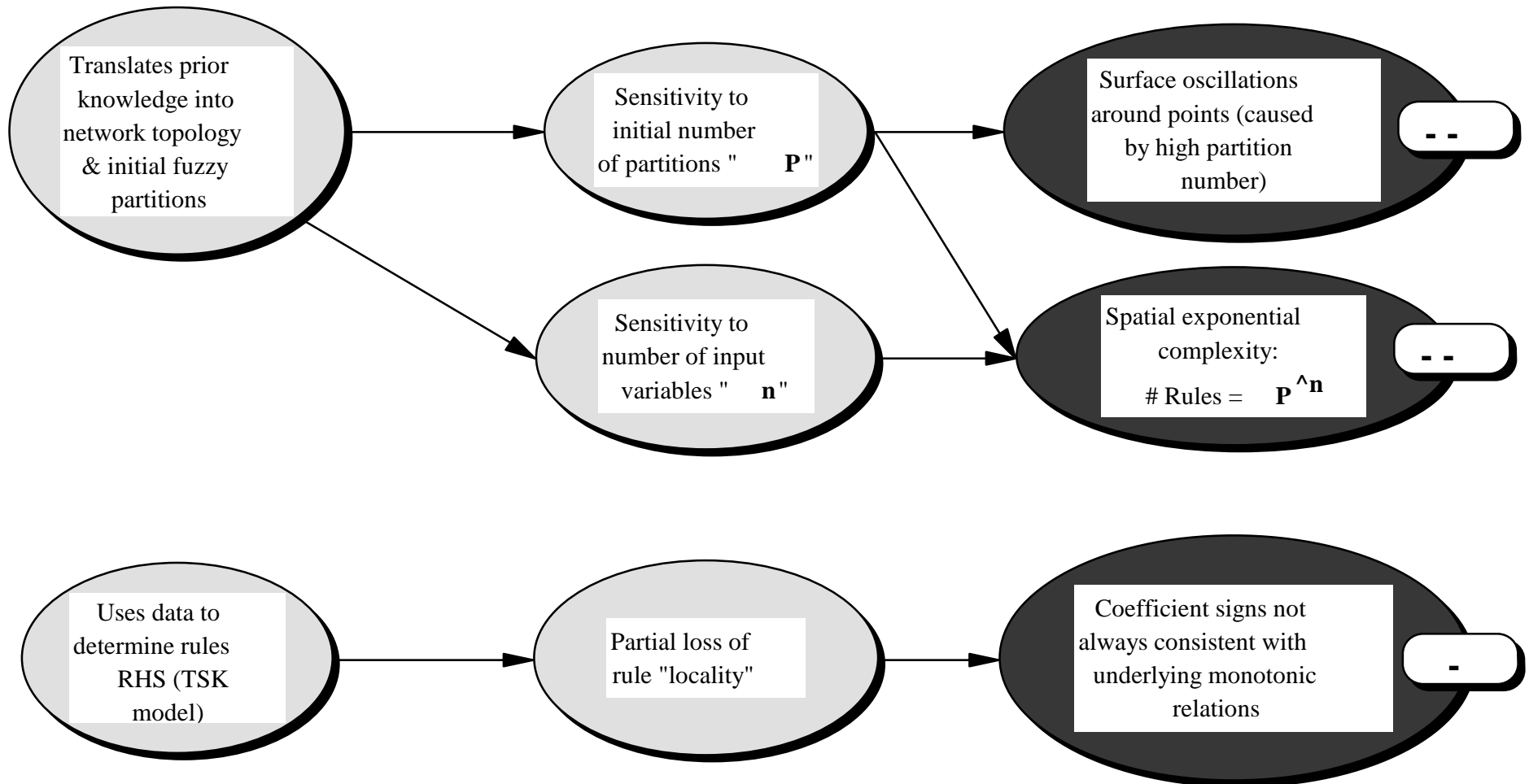
Advantages



ANFIS Cons

Characteristics

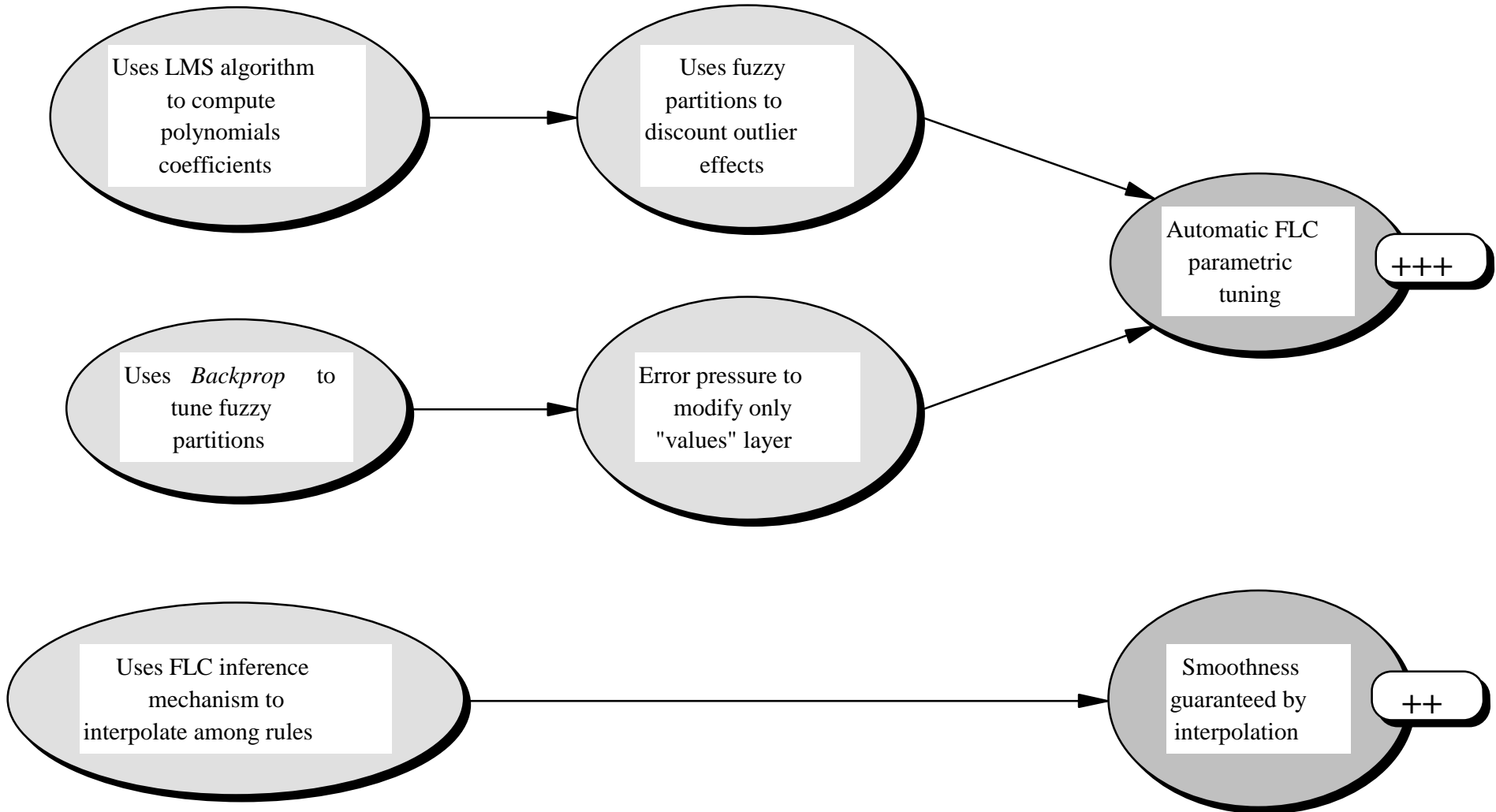
Disadvantages



ANFIS Pros (cont.)

Characteristics

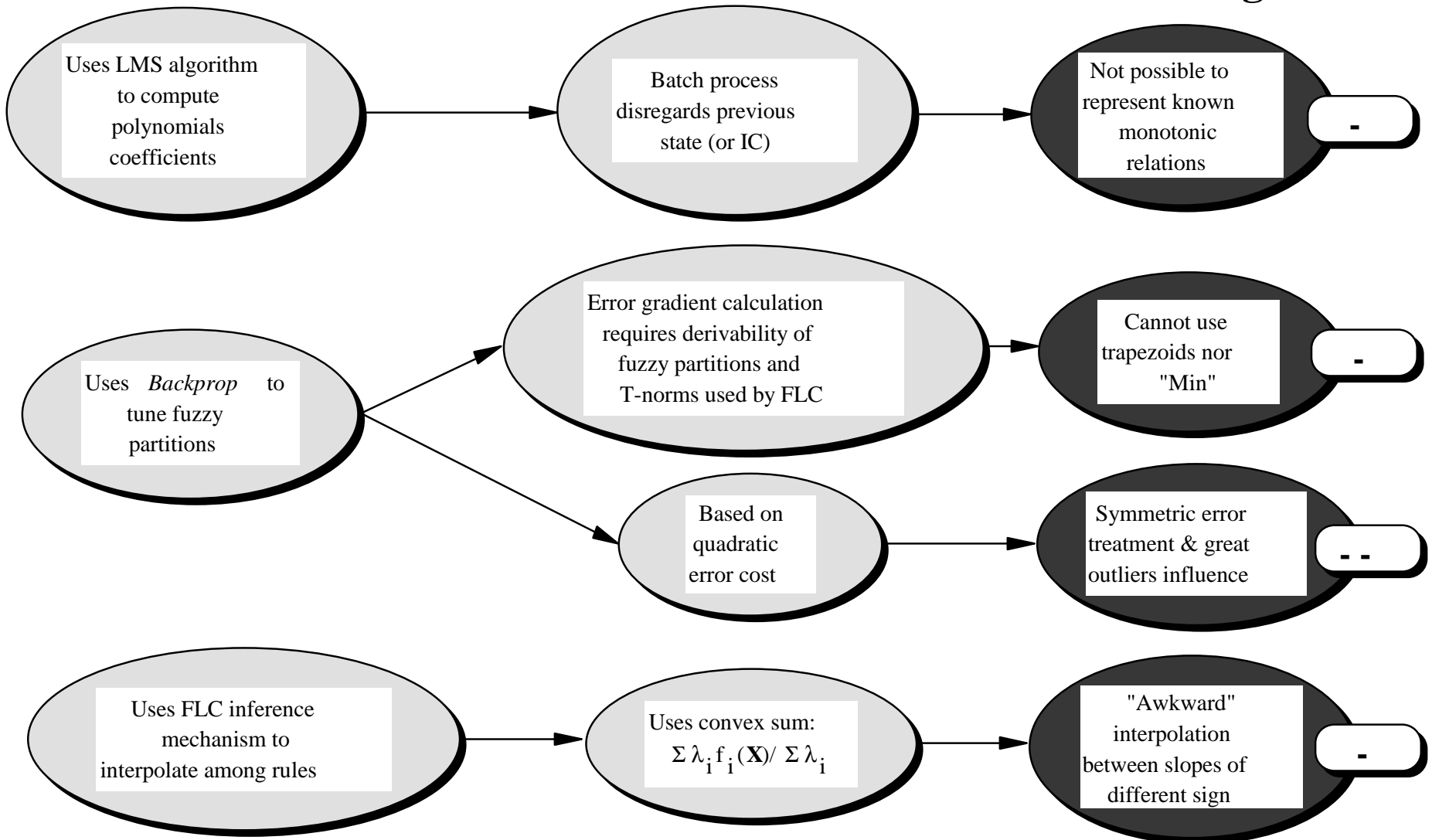
Advantages



ANFIS Cons (cont.)

Characteristics

Disadvantages



ANFIS Opportunities

- Changes to decrease ANFIS *complexity*
 - Use “don’t care” values in rules (no connection between any node of value layer and rule layer)
 - Use reduced subset of state vector in partition tree while evaluating linear functions on complete state

$$\bar{X} = (\bar{X}_r \cup \bar{X}_{(n-r)})$$

- Use heterogeneous partition granularity (different partitions p_i for each state variable, instead of “p”)

$$\# \text{ RULES} = \prod_{i=1}^n p_i$$

ANFIS Opportunities (cont.)

- Changes to extend ANFIS *applicability*
 - Use other cost function (rather than SSE) to represent the user's utility values of the error (error asymmetry, saturation effects of outliers, etc.)
 - Use other type of aggregation function (rather than convex sum) to better handle slopes of different signs.

ANFIS Applications at GE

- Margoil Oil Thickness Estimator
- Voltage Instability Predictor (Smart Relay)
- Collateral Evaluation for Mortgage Approval
- Prediction of Time-to-Break for Paper Web

ANFIS References

- “ANFIS: Adaptive-Network-Based Fuzzy Inference System”, J.S.R. Jang, *IEEE Trans. Systems, Man, Cybernetics*, 23(5/6):665-685, 1993.
- “Neuro-Fuzzy Modeling and Control”, J.S.R. Jang and C.-T. Sun, *Proceedings of the IEEE*, 83(3):378-406
- “Industrial Applications of Fuzzy Logic at General Electric”, Bonissone, Badami, Chiang, Khedkar, Marcelle, Schutten, *Proceedings of the IEEE*, 83(3):450-465
- *The Fuzzy Logic Toolbox for use with MATLAB*, J.S.R. Jang and N. Gulley, Natick, MA: The MathWorks Inc., 1995
- *Machine Learning, Neural and Statistical Classification* Michie, Spiegelhart & Taylor (Eds.), NY: Ellis Horwood 1994
- *Classification and Regression Trees*, Breiman, Friedman, Olshen & Stone, Monterey, CA: Wadsworth and Brooks, 1985