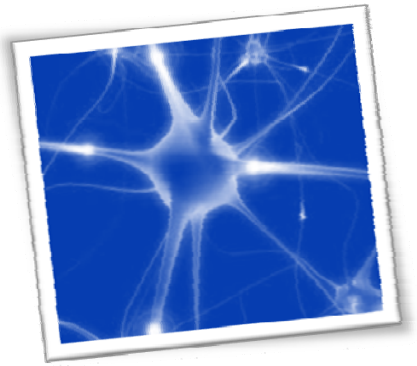
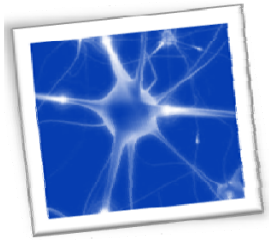
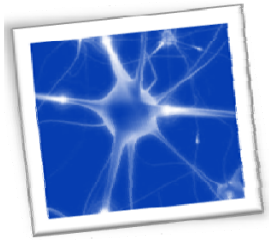
Lecture 17: ANFIS – Adaptive Network-Based Fuzzy Inference System





Outline

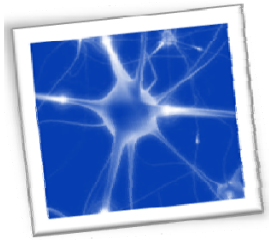
- ANFIS Architecture
- Hybrid Learning Algorithm
- Learning Methods that Cross-Fertilize ANFIS and RBFN
- ANFIS as a universal approximator



What is ANFIS?

- There is a class of adaptive networks that are functionally equivalent to fuzzy inference systems.

- The architecture of these networks is referred to as ANFIS, which stands for **adaptive network-based fuzzy inference system** or semantically equivalently, **adaptive neuro-fuzzy inference system.**

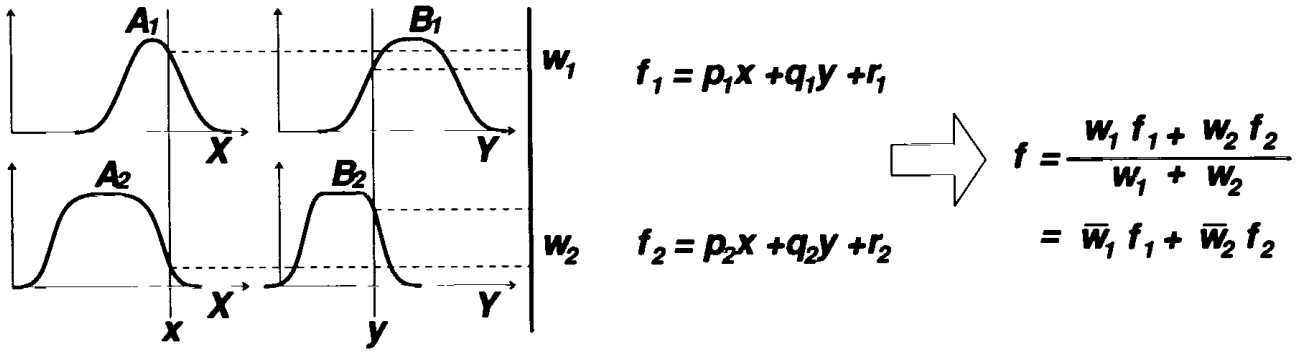


ANFIS Architecture

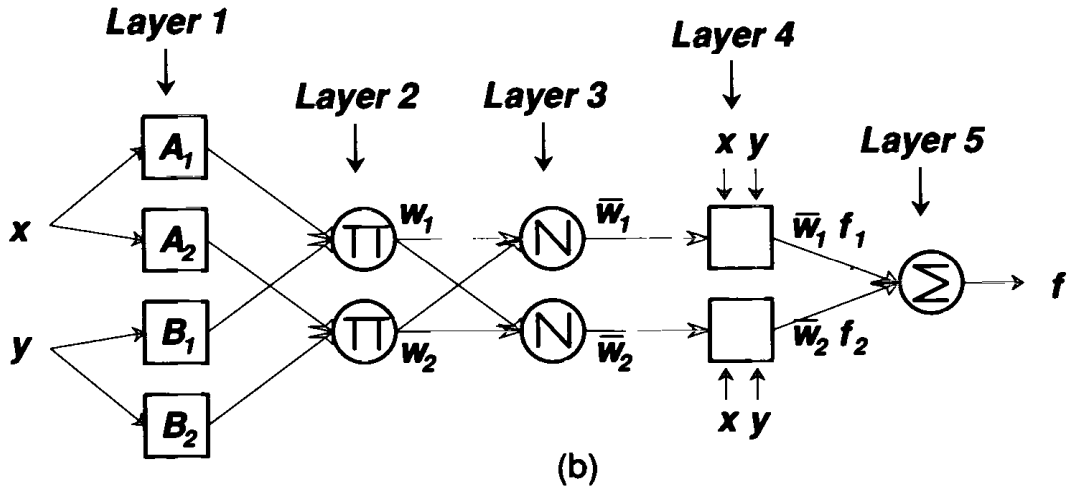
- Assume that the fuzzy inference system under consideration has **two inputs** *x* and *y* and **one output** *z.*
- For a first-order Sugeno fuzzy model with two if-then rules:

$$\text{Rule 1: If } x \text{ is } A_1 \text{ and } y \text{ is } B_1, \text{ then } f_1 = p_1 x + q_1 y + r_1,$$
$$\text{Rule 2: If } x \text{ is } A_2 \text{ and } y \text{ is } B_2, \text{ then } f_2 = p_2 x + q_2 y + r_2.$$

Sugeno model and its corresponding equivalent ANFIS architecture



(a)



(b)

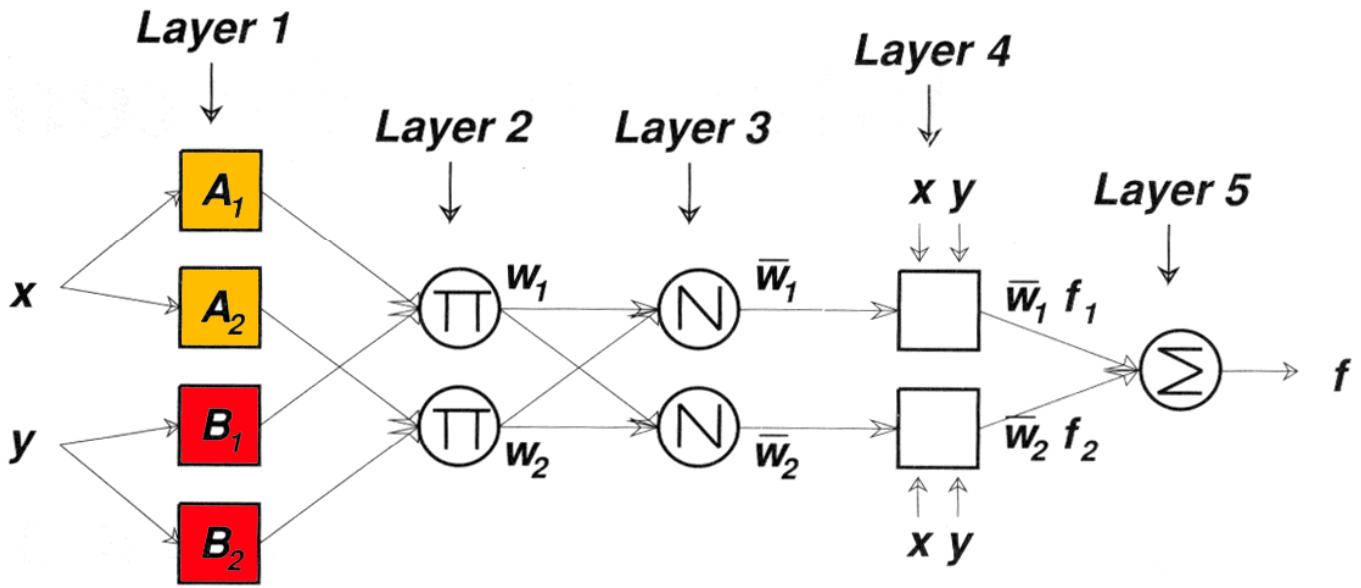
a) A two-input first-order Sugeno fuzzy model with two rules
b) Equivalent ANFIS architecture

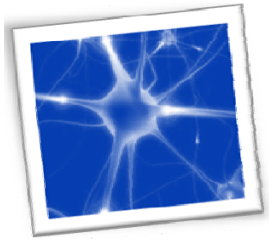
Sugeno ANFIS, Layer 1

- Every node i in this layer is an **adaptive** node with a node function:

$$O_{1,i} = \mu_{A_i}(x), \quad \text{for } i = 1, 2, \text{ or}$$
$$O_{1,i} = \mu_{B_{i-2}}(y), \quad \text{for } i = 3, 4,$$

- In other words, $\mathbf{O_{1,i}}$ is the membership grade of a fuzzy set A_i (or B_i).





Sugeno ANFIS, Layer 1

- Here the membership function can be any appropriate parameterized membership function, such as the generalized bell function:

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

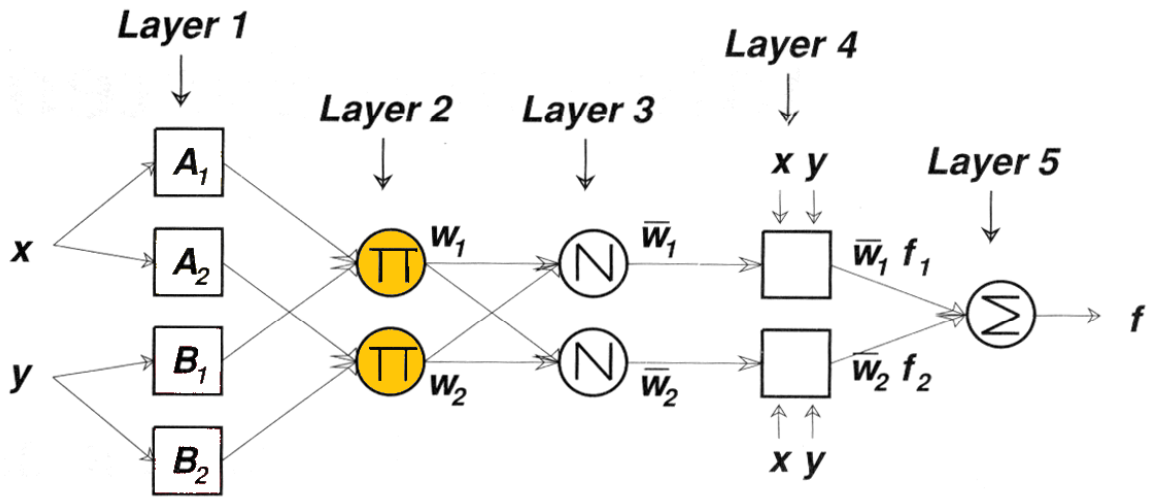
- where $\{a_i, b_i, c_i\}$ is the parameter set.
- Parameters in this layer are referred to as **premise parameters.**

Sugeno ANFIS, Layer 2

- Every node in this layer is a fixed node labeled Π, whose output is the product of all the incoming signals:

$$O_{2,i} = w_i = \mu_{A_i}(x)\mu_{B_i}(y), \; i = 1, 2$$

- Each node's output represents the firing strength of a rule.
- Any other T-norm operators that perform fuzzy AND (e.g. min) can be used as the node function in this layer.

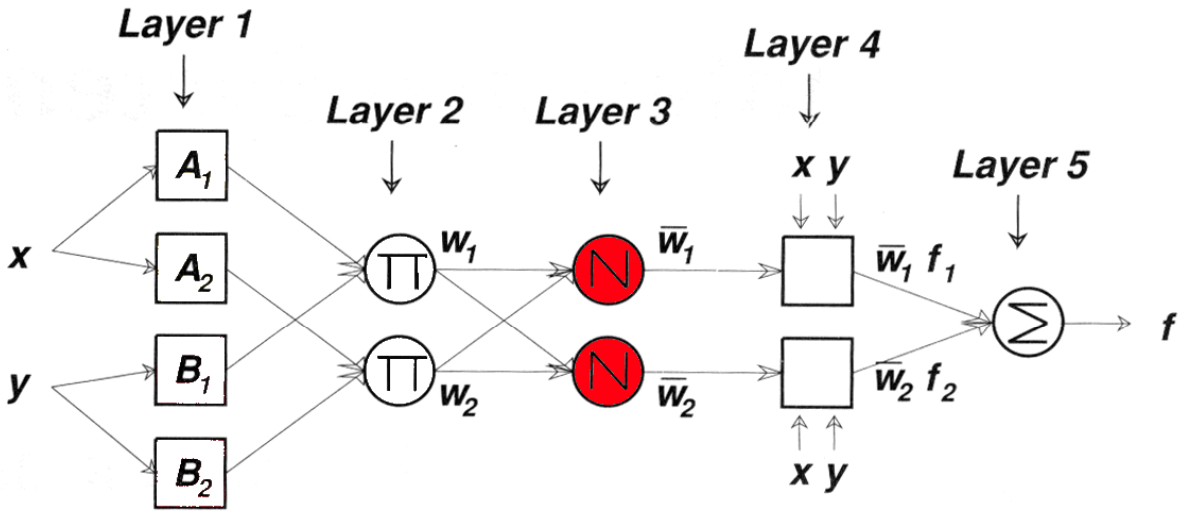


Sugeno ANFIS, Layer 3

- Every node in this layer is a fixed node labeled N. The i-th node calculates the ratio of the i-th rule's firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \; i = 1, 2$$

- For convenience, outputs of this layer are called **normalized firing strengths.**

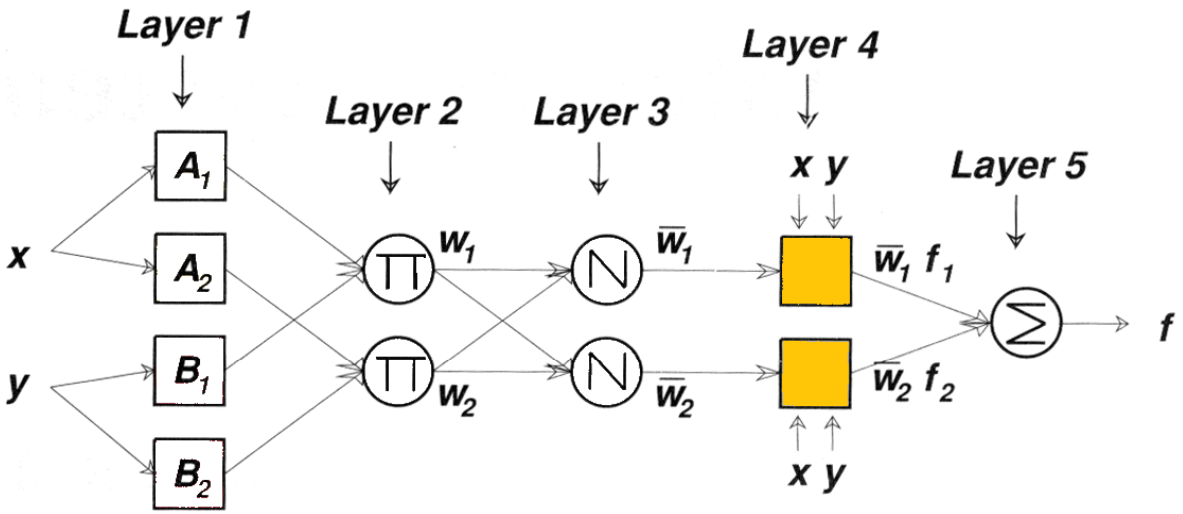


Sugeno ANFIS, Layer 4

- Every node i in this layer is an adaptive node with a node function

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i),$$

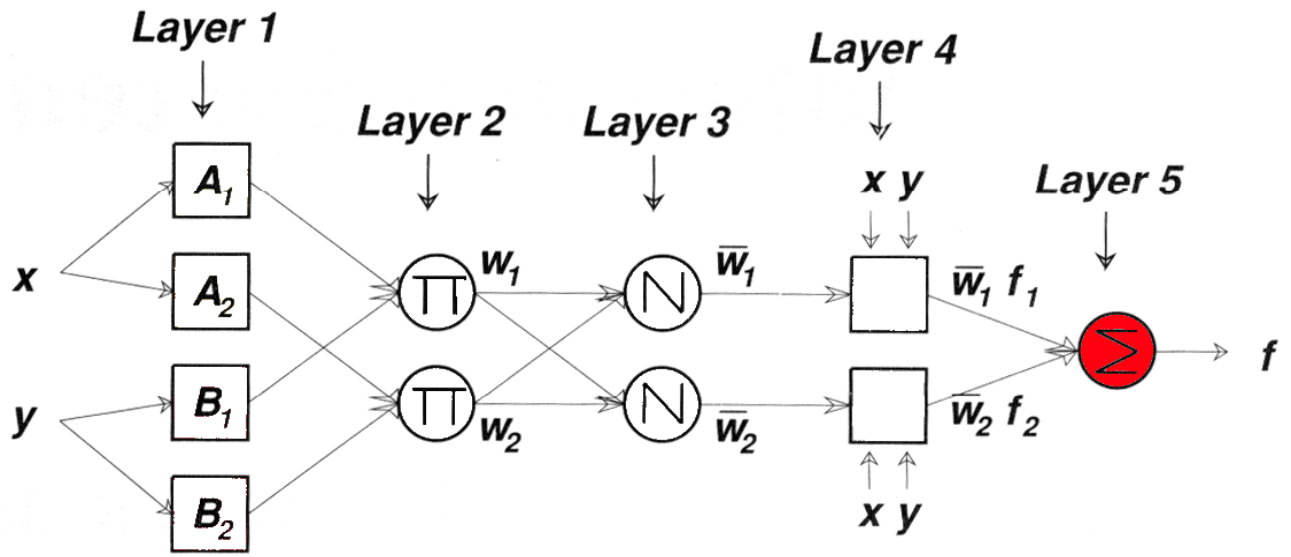
- where w_i is a normalized firing strength from layer 3 and $\{p_i , q_i , r_i\}$ is the parameter set of this node.
- Parameters in this layer are referred to as **consequent parameters.**



Sugeno ANFIS, Layer 5

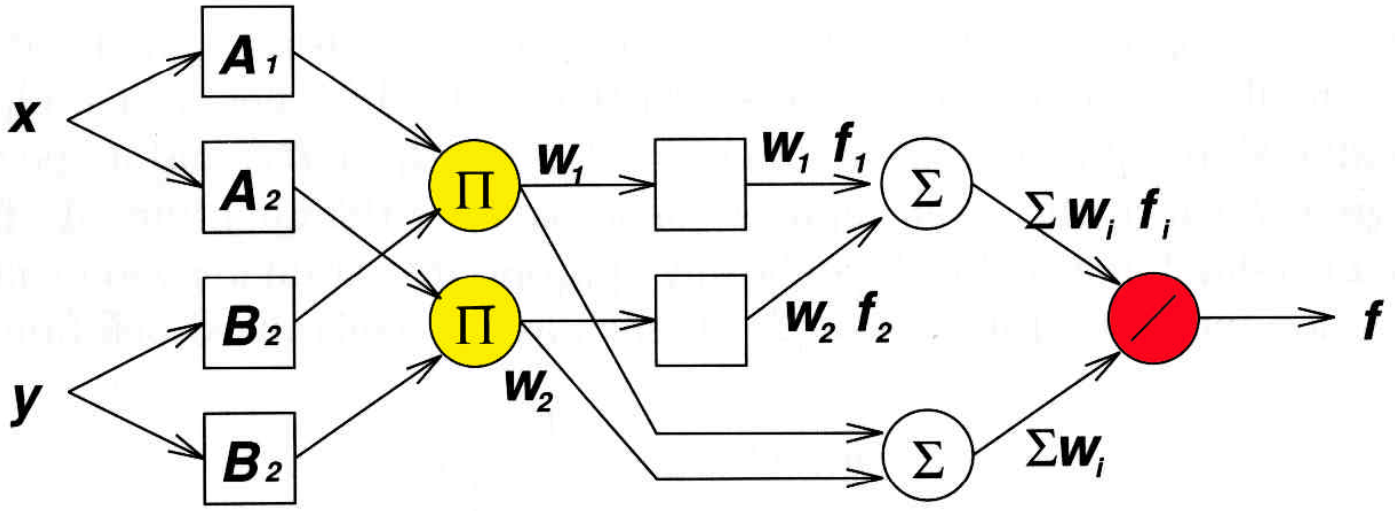
- The single node in this layer is a fixed node labeled ∑, which computes the overall output as the summation of all incoming signals:

$$\text{overall output} = O_{5,1} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$



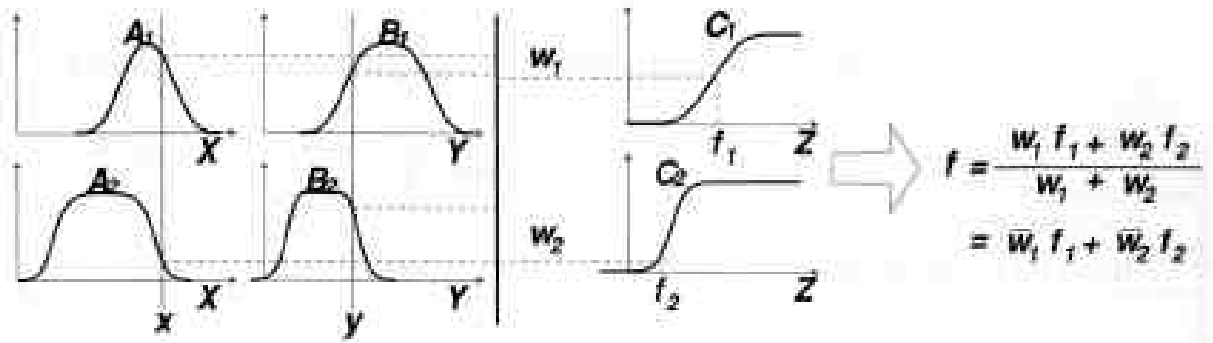
Structure of Sugeno ANFIS

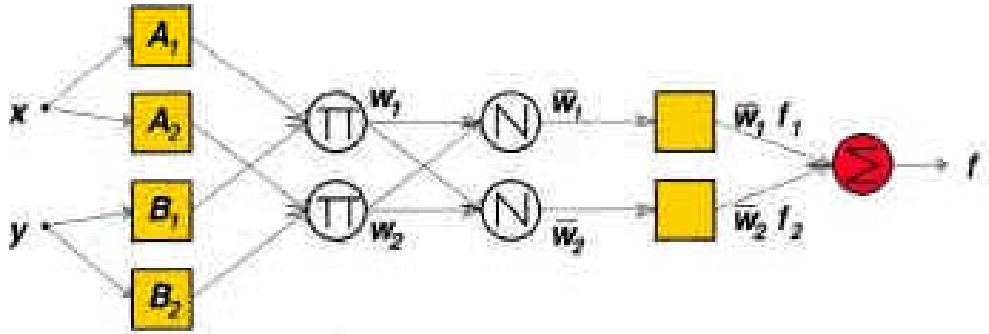
- Note that the structure of this adaptive network is **not** unique.
- For example, we can perform the weight normalization at the last layer.



Extension from Sugeno ANFIS to Tsukamoto ANFIS

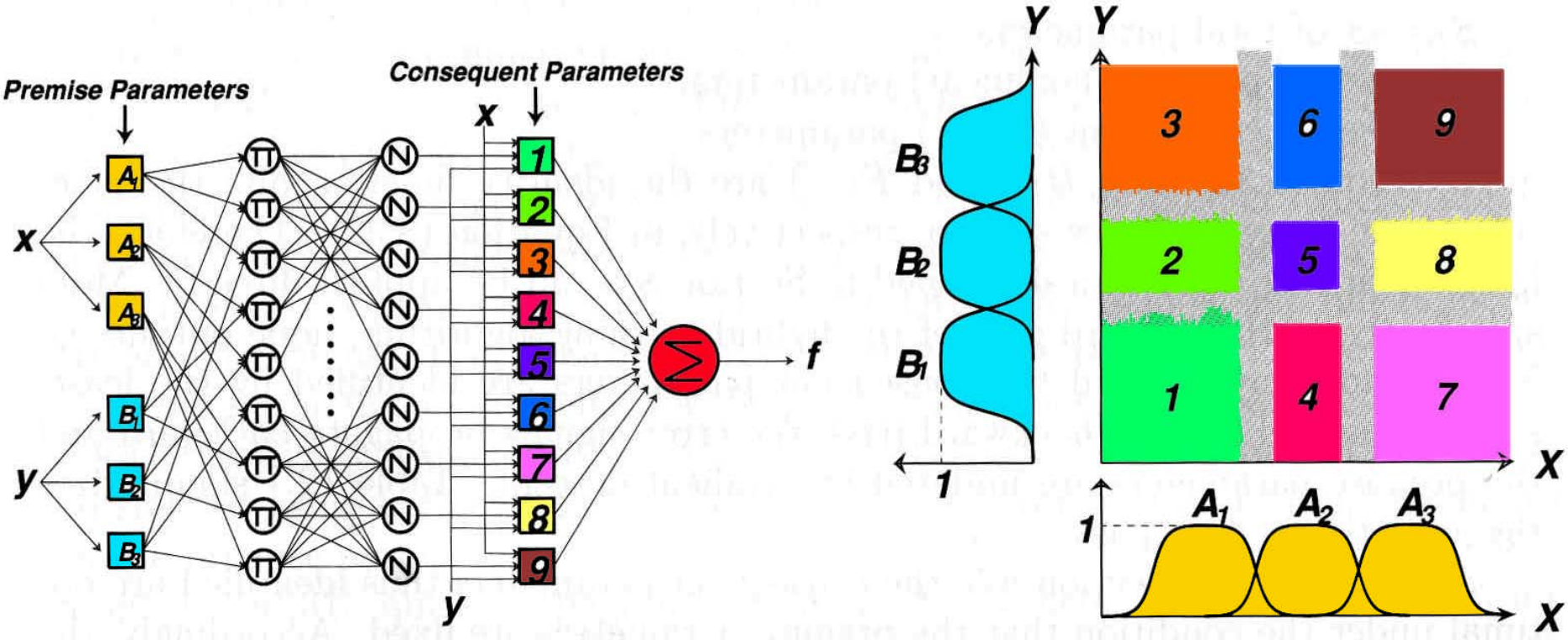
- The output of each rule (f_i) is induced jointly by a consequent membership function and a firing strength.

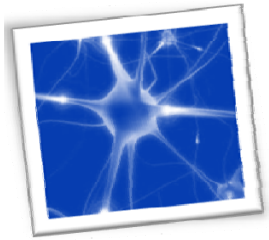




a) A two-input Tsukamoto fuzzy model with two rules
b) Equivalent ANFIS architecture

ANFIS Architecture: two-input Sugeno with nine rules

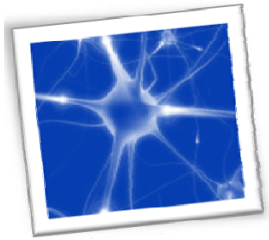




ANFIS for Mamdani FIS

- For the Mamdani fuzzy inference system with max-min composition, a corresponding ANFIS can be constructed if **discrete approximations** are used to replace the integrals in the centroid defuzzification scheme.

- The resulting ANFIS is much more complicated than either **Sugeno ANFIS** or **Tsukamoto ANFIS**.

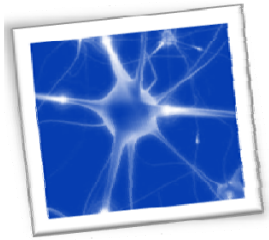


Hybrid Learning Algorithm

- When the values of the **premise parameters** are fixed, the overall output can be expressed as a linear combination of the **consequent parameters**.
- In Symbols, the output f can be written as

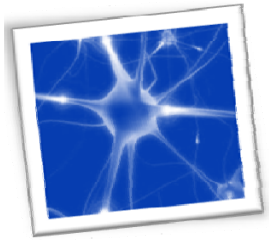
$$
\begin{aligned}
f &= \frac{w_1}{w_1+w_2} f_1 + \frac{w_2}{w_1+w_2} f_2 \\
&= \overline{w}_1(p_1 x + q_1 y + r_1) + \overline{w}_2(p_2 x + q_2 y + r_2) \\
&= (\overline{w}_1 x)p_1 + (\overline{w}_1 y)q_1 + (\overline{w}_1)r_1 + (\overline{w}_2 x)p_2 + (\overline{w}_2 y)q_2 + (\overline{w}_2)r_2,
\end{aligned}
$$

- which is linear in the consequent parameters p_1, q_1, r_1, p_2, q_2, and r_2.



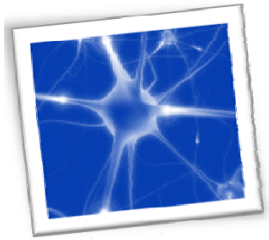
Hybrid Learning Algorithm

- Goal: Calculating **premise parameters** and **consequent parameters** of ANFIS.
- Hybrid learning algorithm consist of two passes:
  - Forward Pass
  - Backward Pass
- Forward Pass:
  - Node outputs go forward until layer 4 and the consequent parameters are identified by the least-squares method (linear learning! how?).
- Backward Pass:
  - The error signals propagate backward and the premise parameters are updated by gradient descent.



Hybrid Learning Algorithm

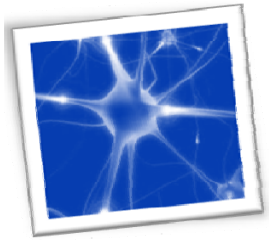
	Forward pass	Backward pass
Premise parameters	Fixed	Gradients descent
Consequent Parameters	Least-squares estimator	Fixed
Signals	Node output	Error signals



Learning Methods that Cross-Fertilize ANFIS and RBFN

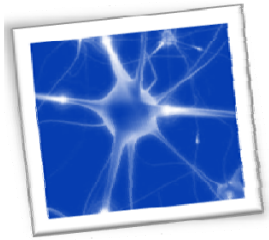
- Under certain minor conditions, an **RBFN** (radial basis function network) is functionally equivalent to a FIS.

- There are a variety of adaptive learning mechanisms that can be used for both adaptive FIS and RBFN.



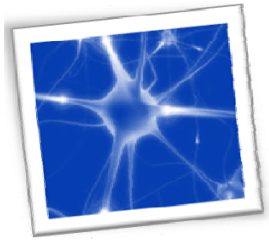
Hybrid Learning Algorithm for ANFIS and RBFN

- An adaptive FIS usually consists of two distinct modifiable parts:
  - The **antecedent** part
  - The **consequent** part
- These two parts can be adapted by different optimization methods, such as hybrid learning procedure combining GD (gradient descent) and LSE (least-squares estimator).
- These learning schemes are equally applicable to RBFNs.



Learning Algorithms for RBFN

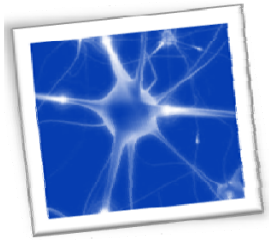
- Conversely, the analysis and learning algorithms for RBFNs are also applicable to adaptive FIS.
- A typical scheme is to fix the **receptive field** (radial basis) functions first and then adjust the **weights** of the output layer.
- There are several schemes proposed to determine the center positions **(μ_i)** of the receptive field functions:
  - Based on standard deviations of training data (Lowe)
  - By means of vector quantization or clustering techniques (Moody and Darken)
- Then, the width parameters $\boldsymbol{\sigma_i}$ are determined by taking the average distance to the first several nearest neighbors of $\mathbf{u_i}$'s.



Learning Algorithms for RBFN

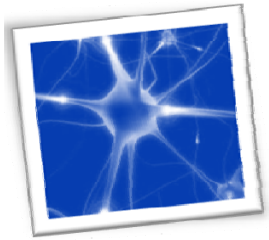
- Once these nonlinear parameters ($\boldsymbol{u_i}$ and $\boldsymbol{\sigma_i}$) are fixed and the receptive fields are frozen, the linear parameters (i.e., the weights of the output layer) can be updated by either the least-squares method or the gradient method.

- Other RBFN analyses, such as **generalization properties**, are all applicable to adaptive FIS.



ANFIS as a Universal Approximator

- When the number of rules is not restricted, a **zero-order Sugeno** model has unlimited approximation power for matching any nonlinear function arbitrarily well on a compact set.
- This fact is intuitively reasonable.
- However, to give a mathematical proof, we need to apply the Stone-Weierstrass theorem (see the book for detailed analysis!).



Example

Rule 1: IF x is small (A1) AND y is small (B1) THEN f1=small
Rule 2: IF x is large (A2) AND y is large (B2) THEN f2=large

A1: $\mu_{A1}(x) = \dfrac{1}{1+\left|\dfrac{x-1}{2}\right|^2}$

B1: $\mu_{B1}(y) = \dfrac{1}{1+\left|\dfrac{y-2}{2}\right|^2}$

$f1 = 0.1x + 0.1y + 0.1$

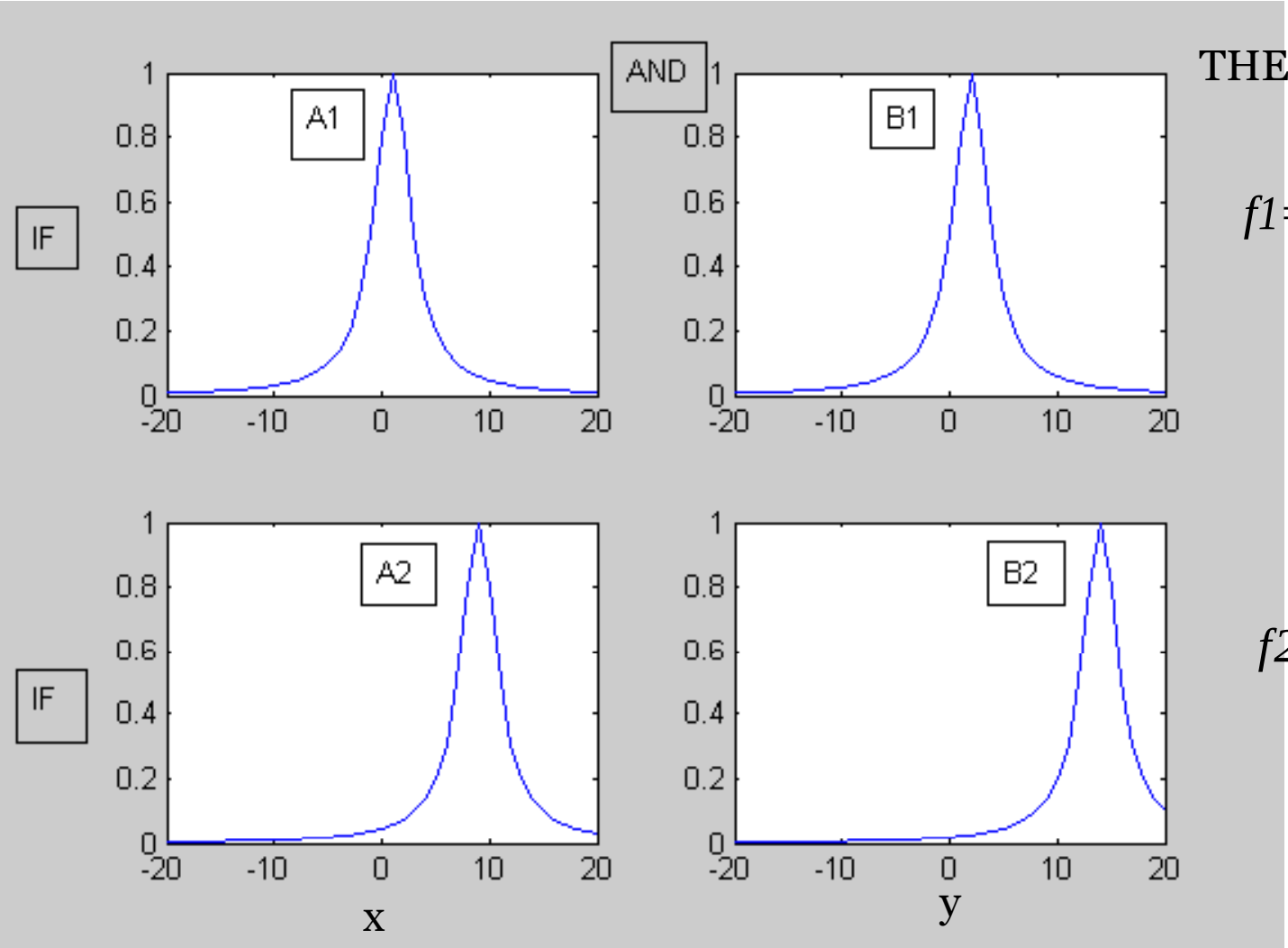
A2: $\mu_{A2}(x) = \dfrac{1}{1+\left|\dfrac{x-9}{2}\right|^2}$

B2: $\mu_{B2}(y) = \dfrac{1}{1+\left|\dfrac{y-14}{2}\right|^2}$

$f2 = 10x + 10y + 10$

Given the trained fuzzy system above and input values of x=3 and y=4, find output of the Sugeno fuzzy system

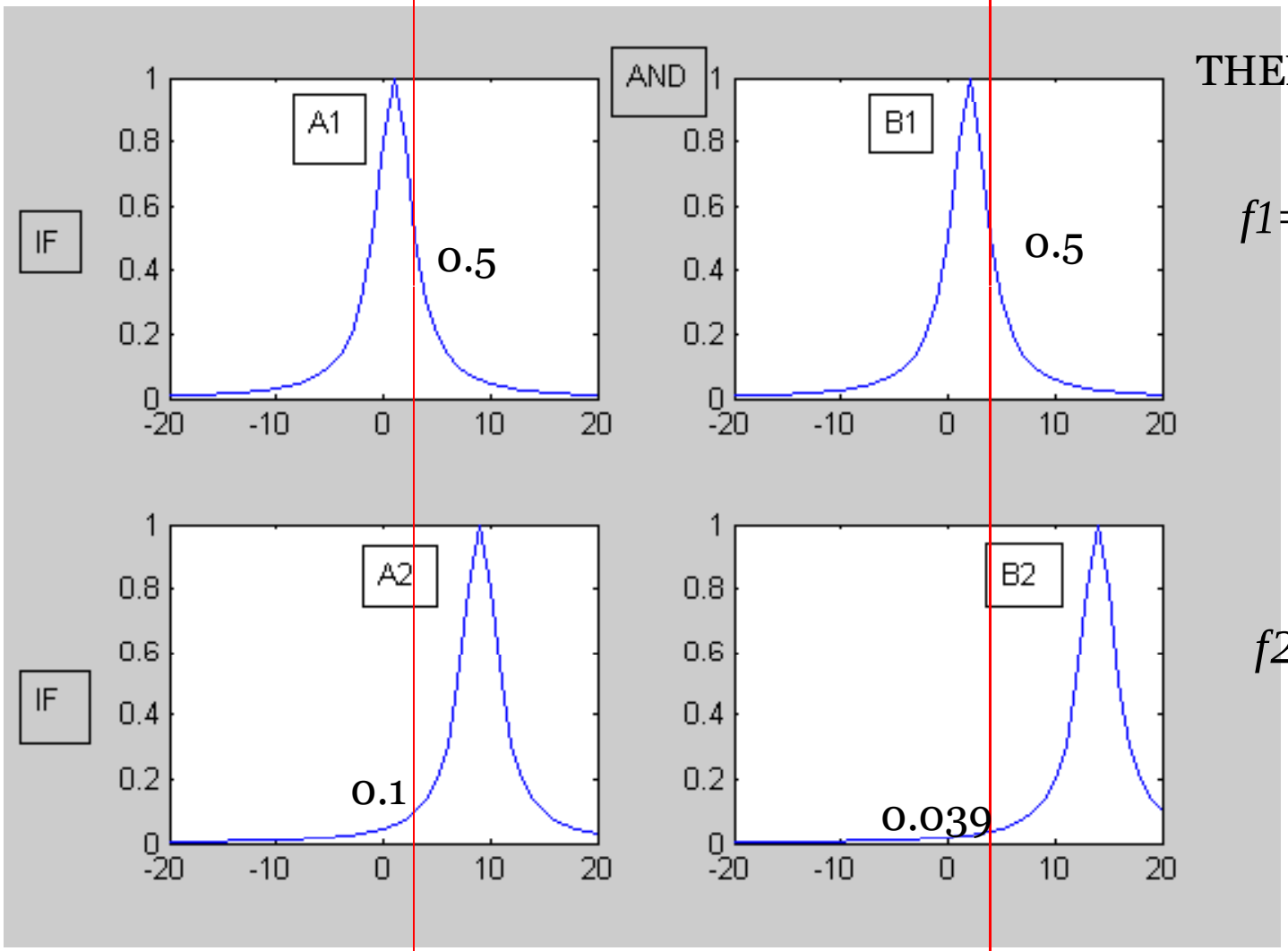
Example



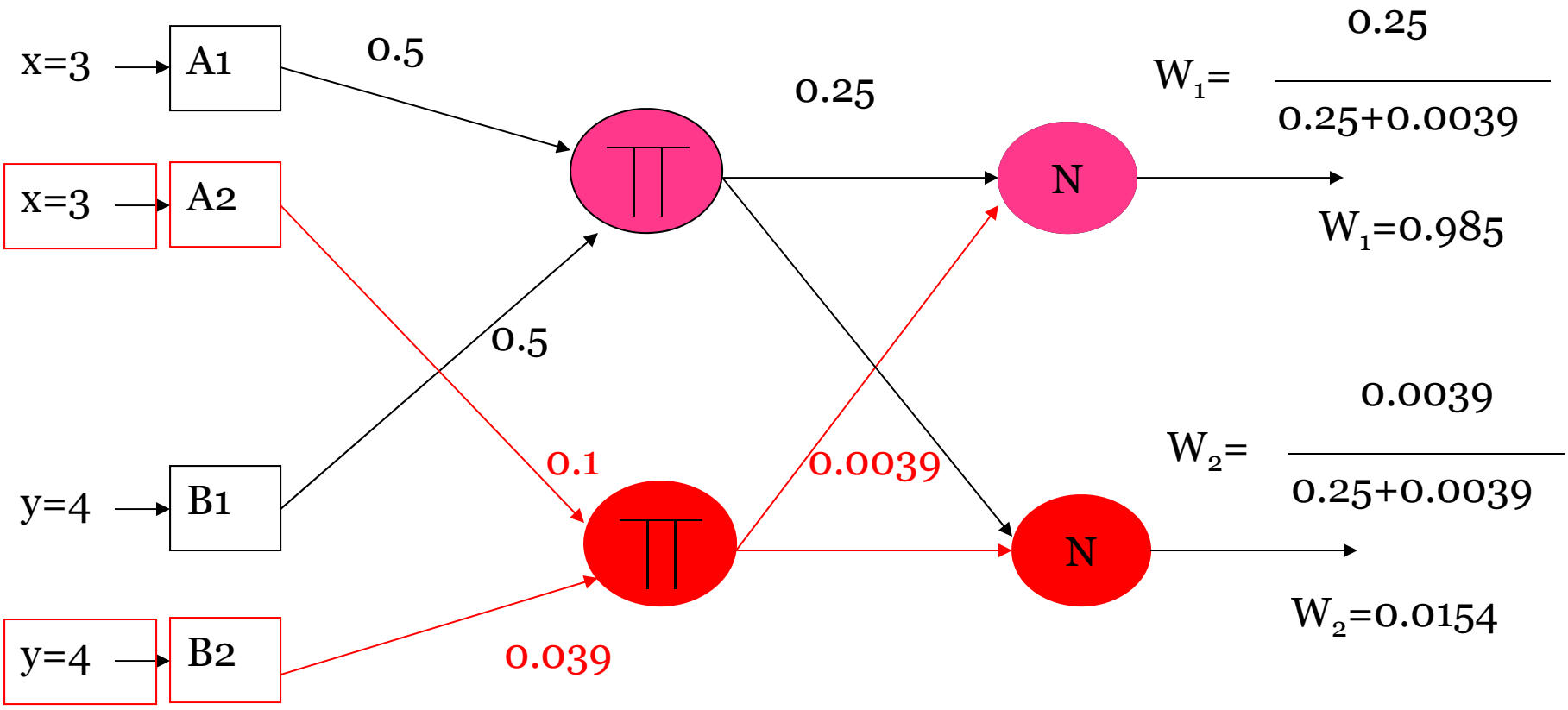
$f1 = 0.1x + 0.1y + 0.1$

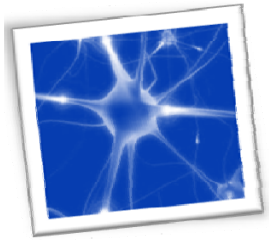
$f2 = 10x + 10y + 10$

Example

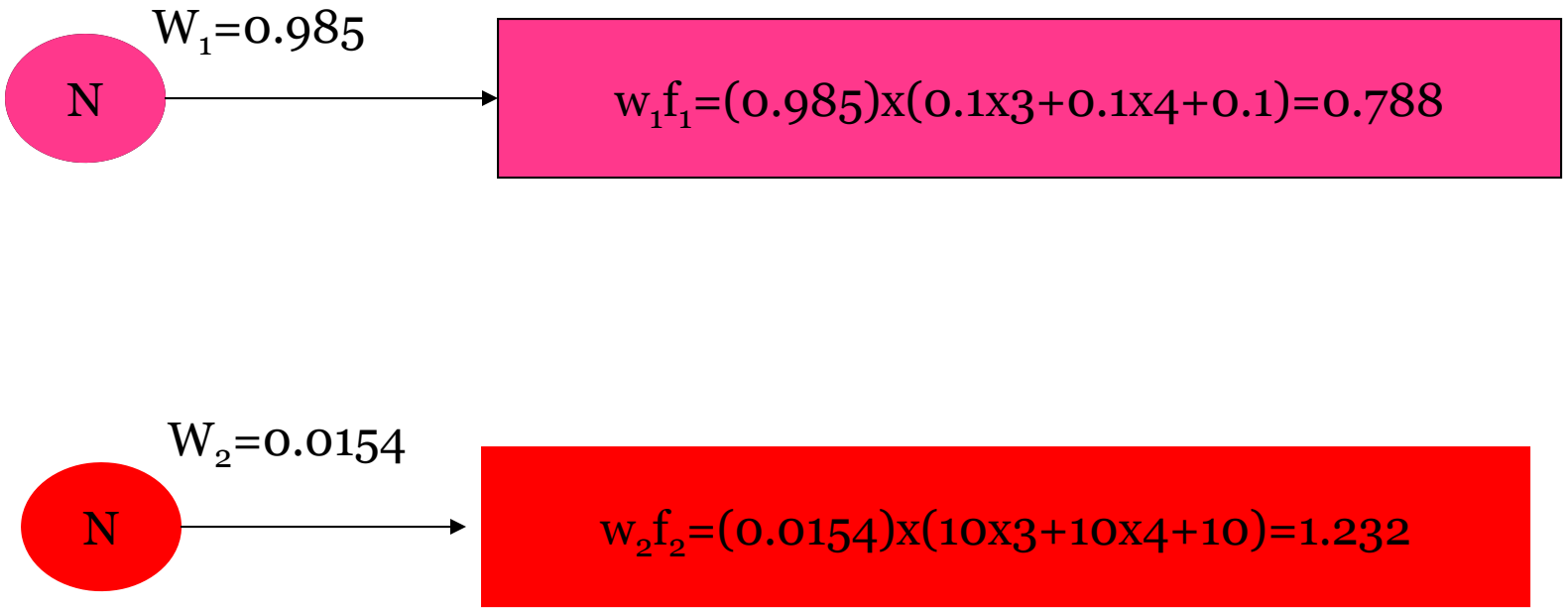

IF
A1
0.5
AND
B1
0.5
THEN
f1=0.1x+0.1y+0.1
IF
A2
0.1
B2
0.039
f2=10x+10y+10
x=3
y=4

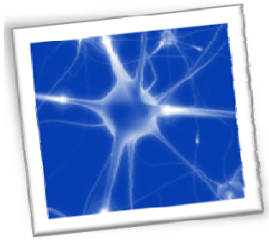
Example


x=3
A1
x=3
A2
y=4
B1
y=4
B2
0.5
0.5
0.1
0.039
0.25
0.0039
N
N
W_1= 0.25 / 0.25+0.0039
W_1=0.985
W_2= 0.0039 / 0.25+0.0039
W_2=0.0154

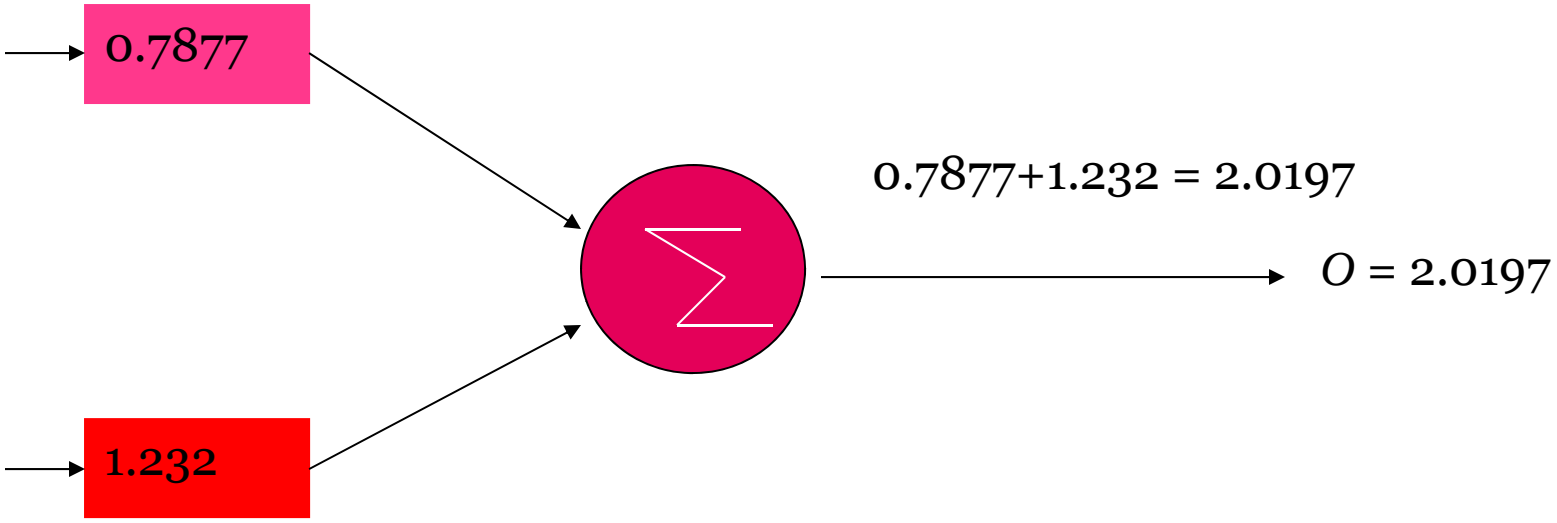


Example

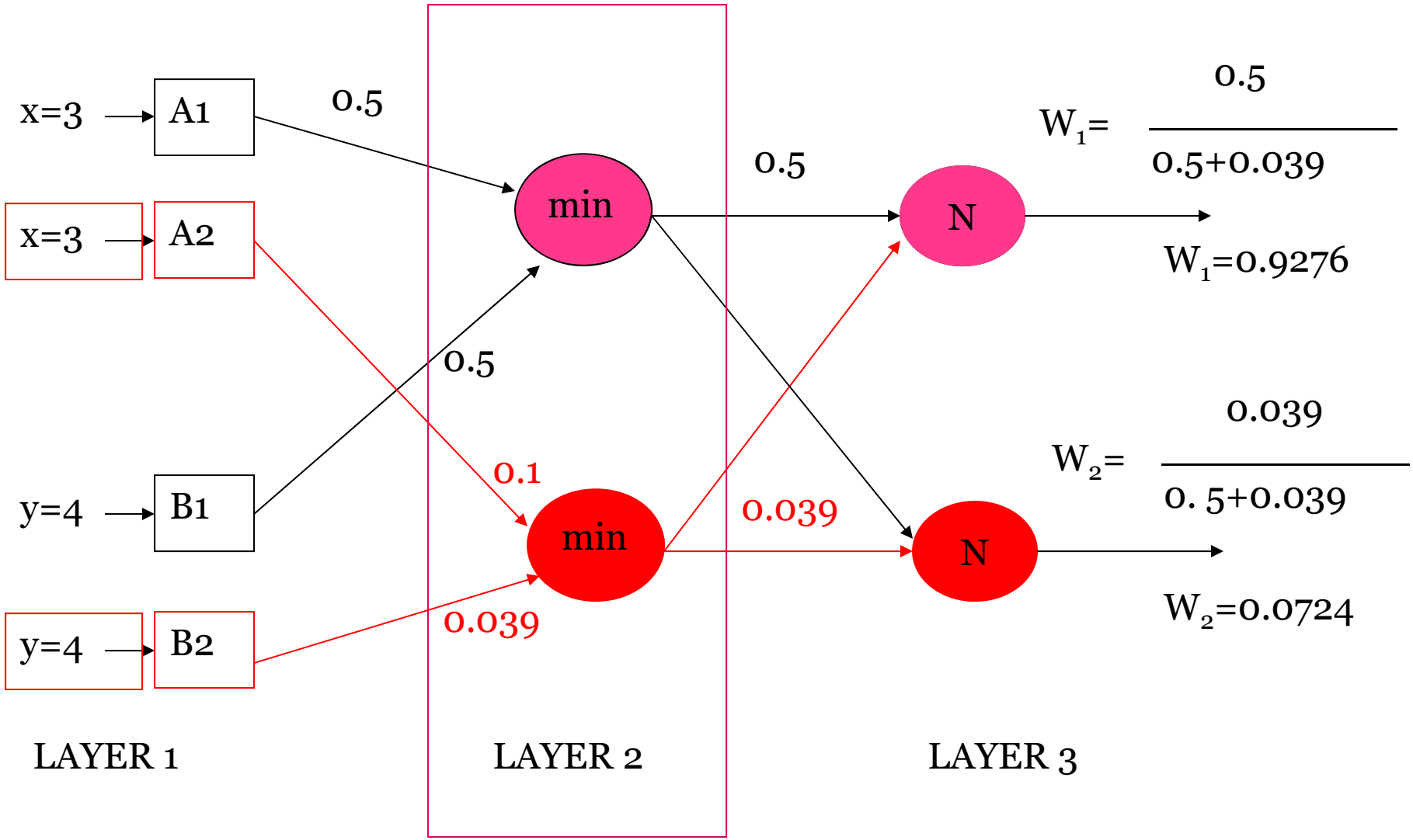

W_1=0.985
N
w_1f_1=(0.985)x(0.1x3+0.1x4+0.1)=0.788
W_2=0.0154
N
w_2f_2=(0.0154)x(10x3+10x4+10)=1.232



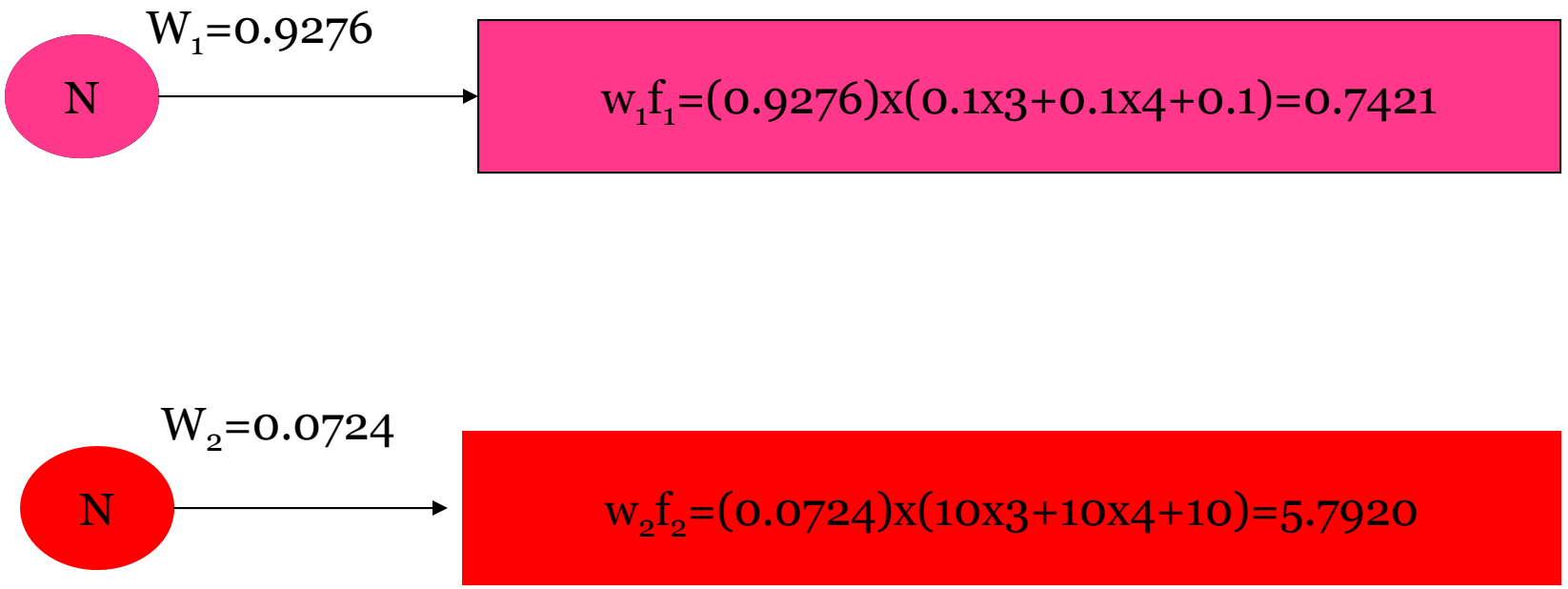
Example



What if "T-norm" is taken at layer 2 to perform AND operation



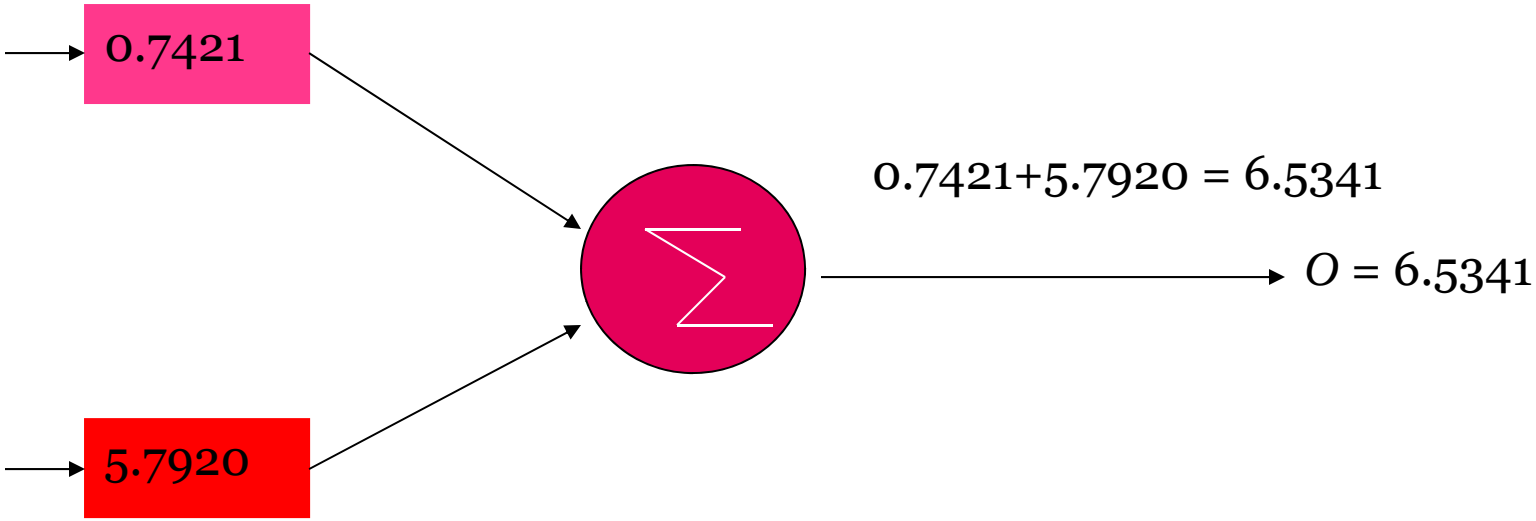
What if “T-norm” is taken at layer 2 to perform AND operation



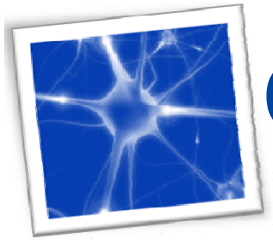
LAYER 3

LAYER 4

What if “T-norm” is taken at layer 2 to perform AND operation



LAYER 4 LAYER 5



Gradient Descent learning for ANFIS

$$F = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

i=1,2,3, …*R* # of rules

F is the calculated/estimated output value (by ANFIS)

Error = e = $(d - F)^2$

d = Actual/Real Output

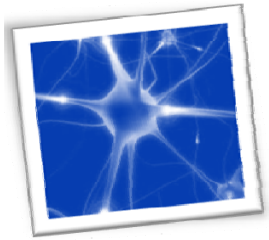
$$\frac{\partial e}{\partial (x, y, ...)}$$

Gradient of ANFIS's output: Making ANFIS's output (O) closer to actual output (AO)

$$a(n+1) = a(n) - \eta \frac{\partial e}{\partial a}$$

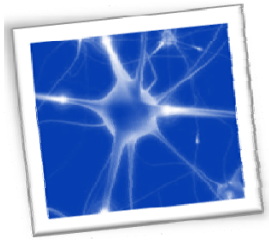
This can be done by updating values of the parameters (e.g., a, c,…) over n (iteration/step)

η: learning rate



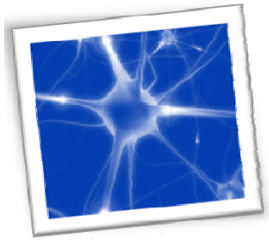
Practical considerations

- In a conventional FIS, the number of rules is determined by an expert familiar with the target
- In ANFIS, it is chosen empirically
  - By plotting the data sets and examining them visually (for less than three inputs)
  - Or by trail and error
- The initial values of premise parameters are set in such a way that the centers of MFs are equally spaced along the range of each input variable



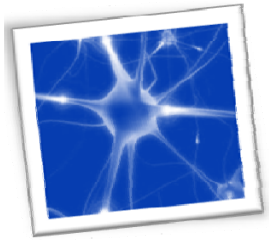
Practical considerations

- These MFs satisfy the condition of ε-completeness with ε = 0.5
  - Given a value x of one of the inputs, we can always find a linguistic label A that $\mu_A \geq \varepsilon$
- This helps the smoothness
- ε-completeness can also be maintained during the optimization process



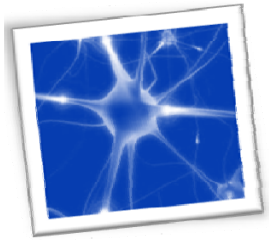
Extensions of ANFIS

- The MFs can be of any parameterized forms introduced previously
- Any T-norm operator can be replaced the algebraic product
- A learning rule can decide the best T-norm operations should be used
- The rules can be realized using OR operators instead of AND



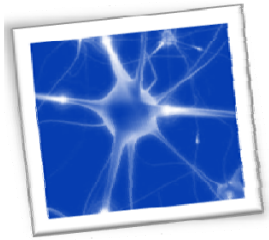
More on ANFIS

- ε-completeness ensures that for any given value of an input variable, there is at least a MF with a grade greater or equal to ε
- This guarantees that the whole input space is covered properly if ε is greater than zero
- ε-completeness can be maintained by the constrained gradient descent



More on ANFIS

- Moderate fuzzyness: within most regions of the input space, there should be a dominant fuzzy if-then rule with a firing strength close to unity that accounts for the final output
- This preserve MFs from too much overlap and make the rule set more informative
- This eliminate the cases where one MFs goes under the other one

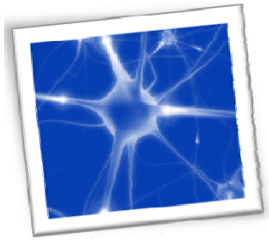


More on ANFIS

- A simple way is to use a modified error measure

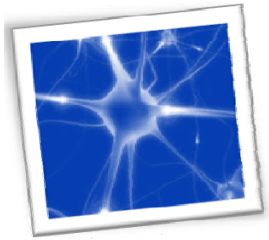
$$E' = E + \beta \sum_{i=1}^{P} [-\overline{w}_i \ln(\overline{w}_i)]$$

- where E is the original squared error, β is a weighting constant, and $\overline{w}_i$ is the normalized firing strength
- The second term is indeed **Shannon's information entropy**
- This error measure is not based on data fitting alone, and trained ANFIS has better generalizability



More on ANFIS

- The easy way of maintaining reasonably shaped MFs is to parameterize the MF correctly



Example

ANFIS is used to model a two-dimensional *sinc* equation defined by

$$z = \sin c(x, y) = \frac{\sin(x)\sin(y)}{xy}$$

x and *y* are in the range [-10,10]

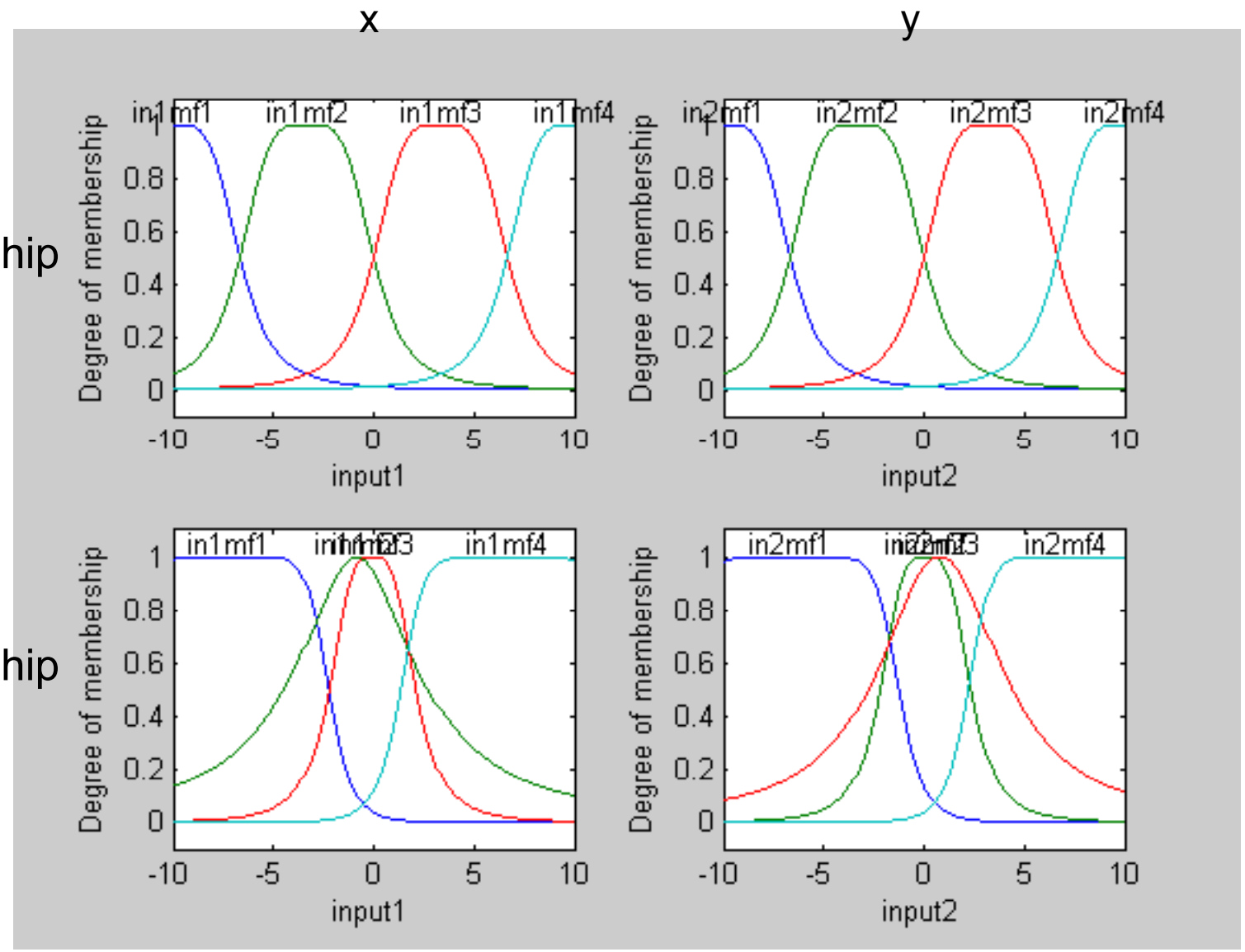
Number of membership functions for each input: 4

Number of rules: 16

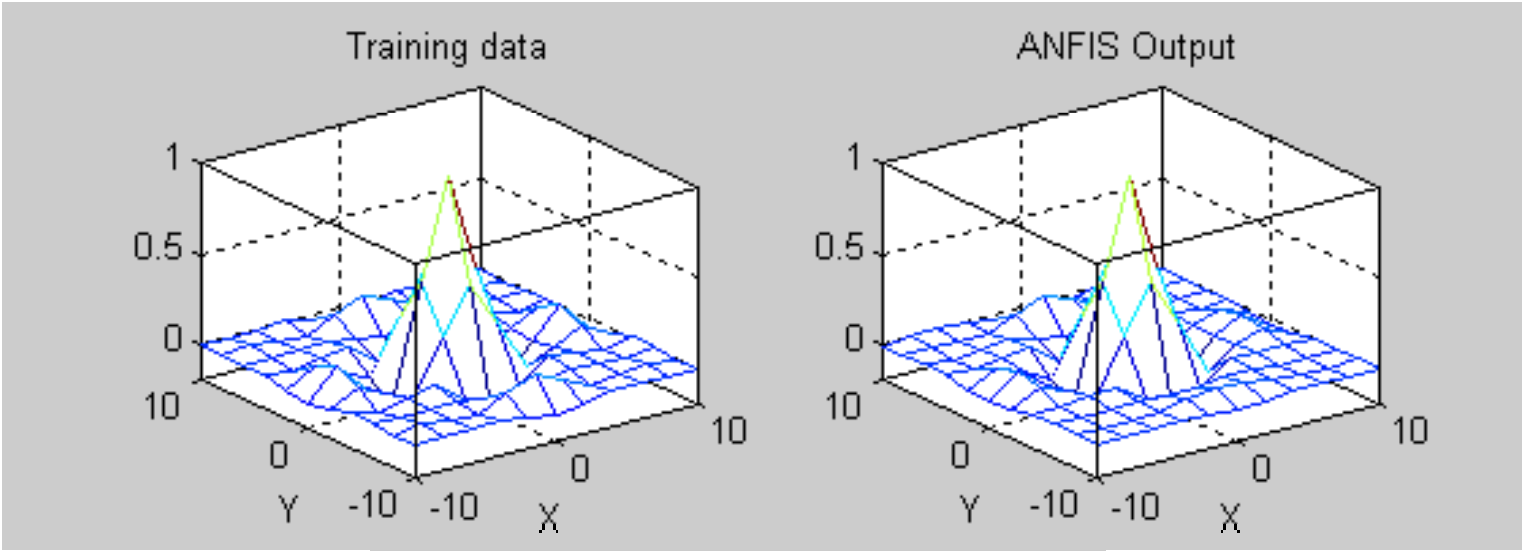
Example

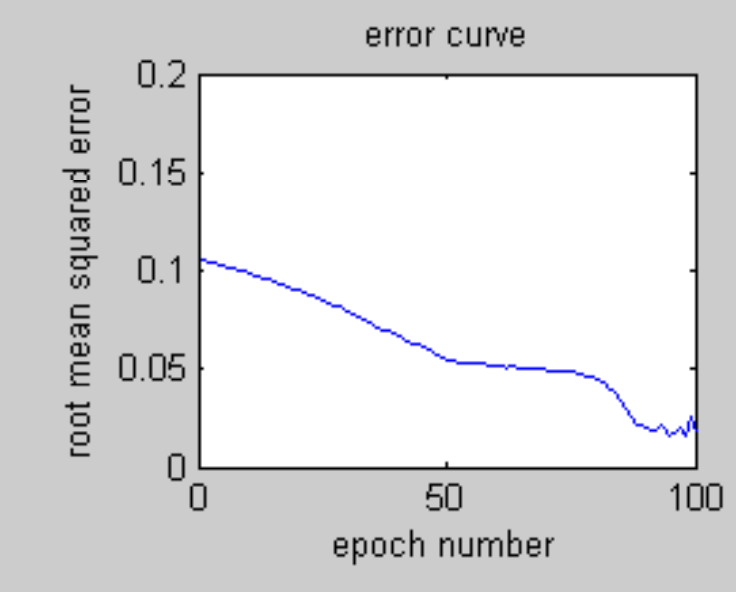
Initial membership functions

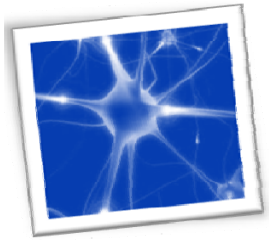
Final (trained) membership functions after 100 epochs



Example


Training data
ANFIS Output
1
0.5
0
10
0
Y
-10
-10
0
10
X


error curve
root mean squared error
0.2
0.15
0.1
0.05
0
0
50
100
epoch number



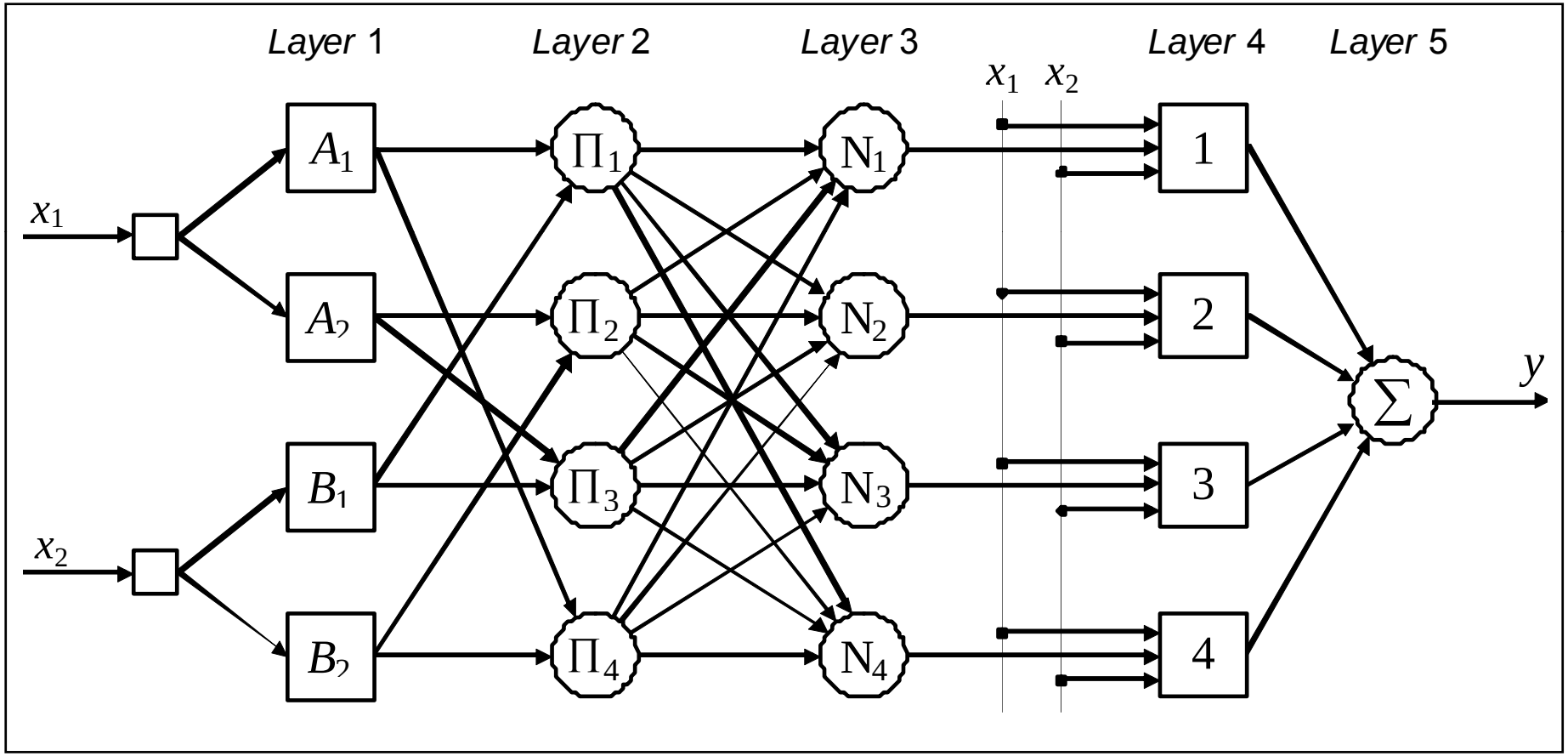
Function approximation using ANFIS

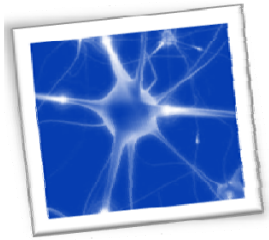
- In this example, an ANFIS is used to follow a trajectory of the non-linear function defined by the equation

$$y = \frac{\cos(2\,x_1)}{e^{x_2}}$$

- First, we choose an appropriate architecture for the ANFIS. An ANFIS must have two inputs – x_1 and x_2 – and one output – y.
- Thus, in our example, the ANFIS is defined by four rules, and has the structure shown below.

An ANFIS model with four rules

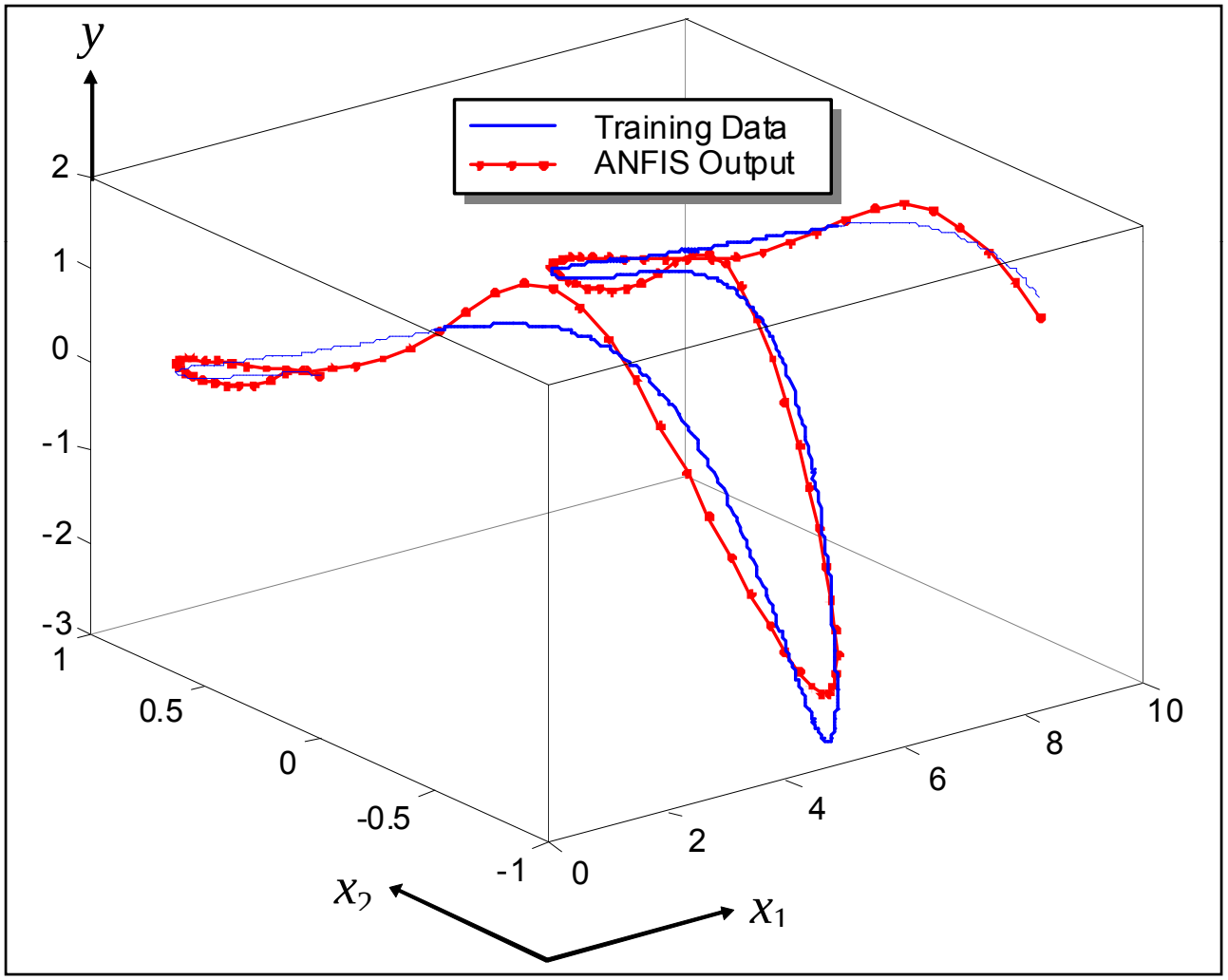




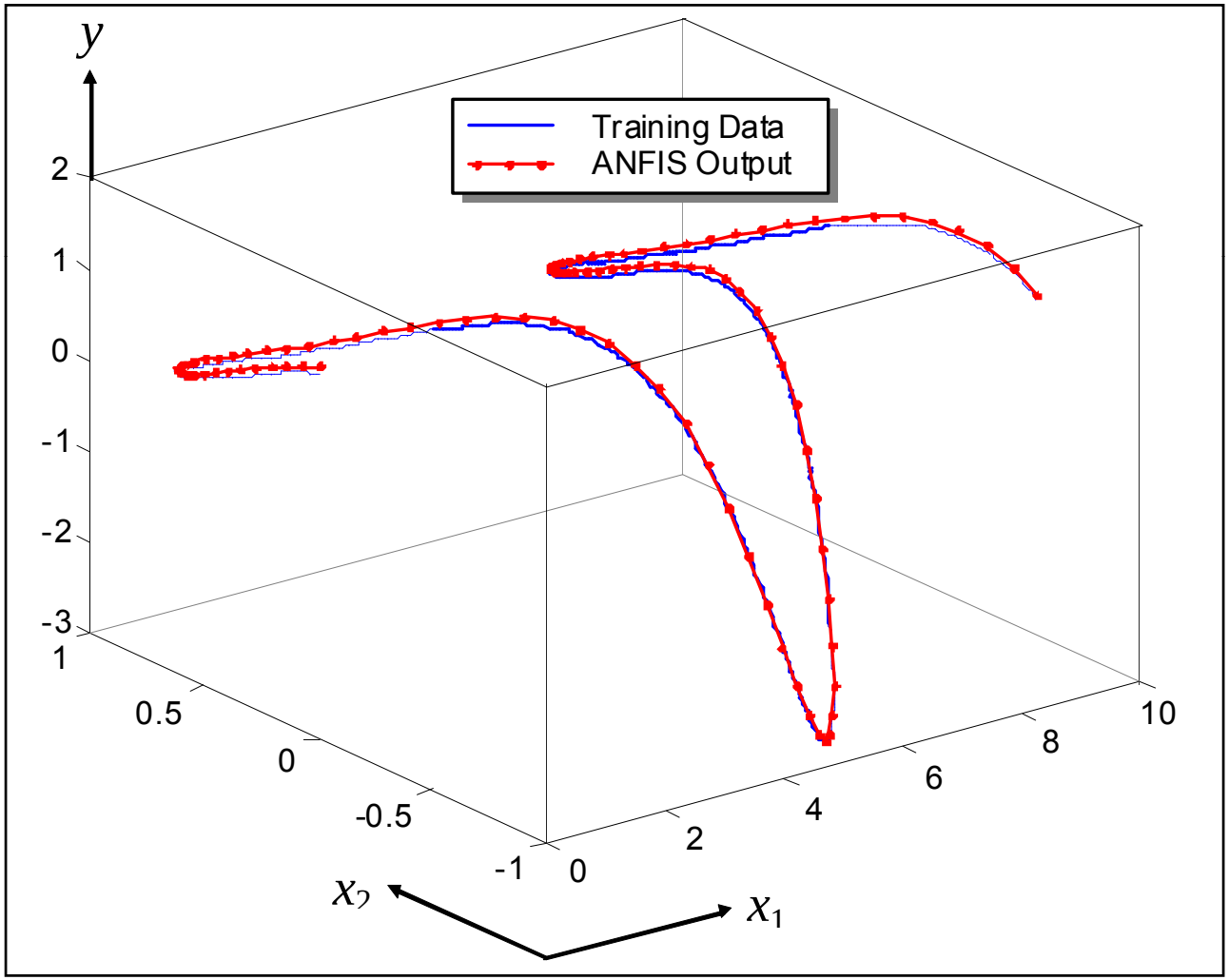
Function approximation using ANFIS

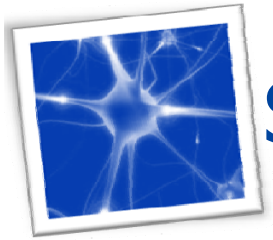
- The ANFIS training data includes 101 training samples. They are represented by a 101×3 matrix $[\mathbf{x}_1\ \mathbf{x}_2\ \mathbf{y}_d]$, where $\mathbf{x}_1$ and $\mathbf{x}_2$ are input vectors, and $\mathbf{y}_d$ is a desired output vector.
- The first input vector, $\mathbf{x}_1$, starts at 0, increments by 0.1 and ends at 10.
- The second input vector, $\mathbf{x}_2$, is created by taking sin from each element of vector $\mathbf{x}_1$, with elements of the desired output vector, $\mathbf{y}_d$, determined by the function equation.

Learning in an ANFIS with two MFs assigned to each input (1 epoch)



Learning in an ANFIS with two MFs assigned to each input (100 epochs)

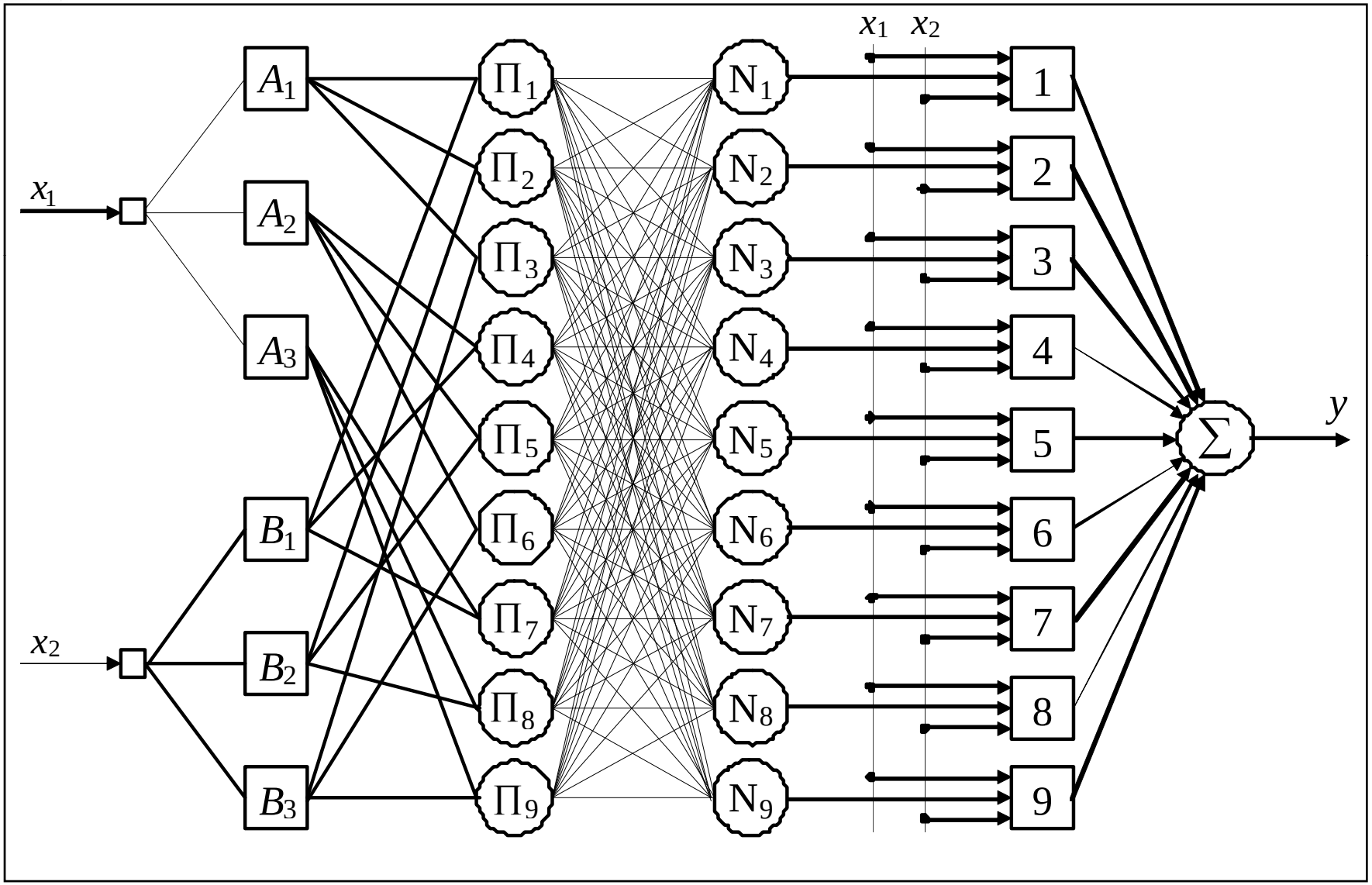




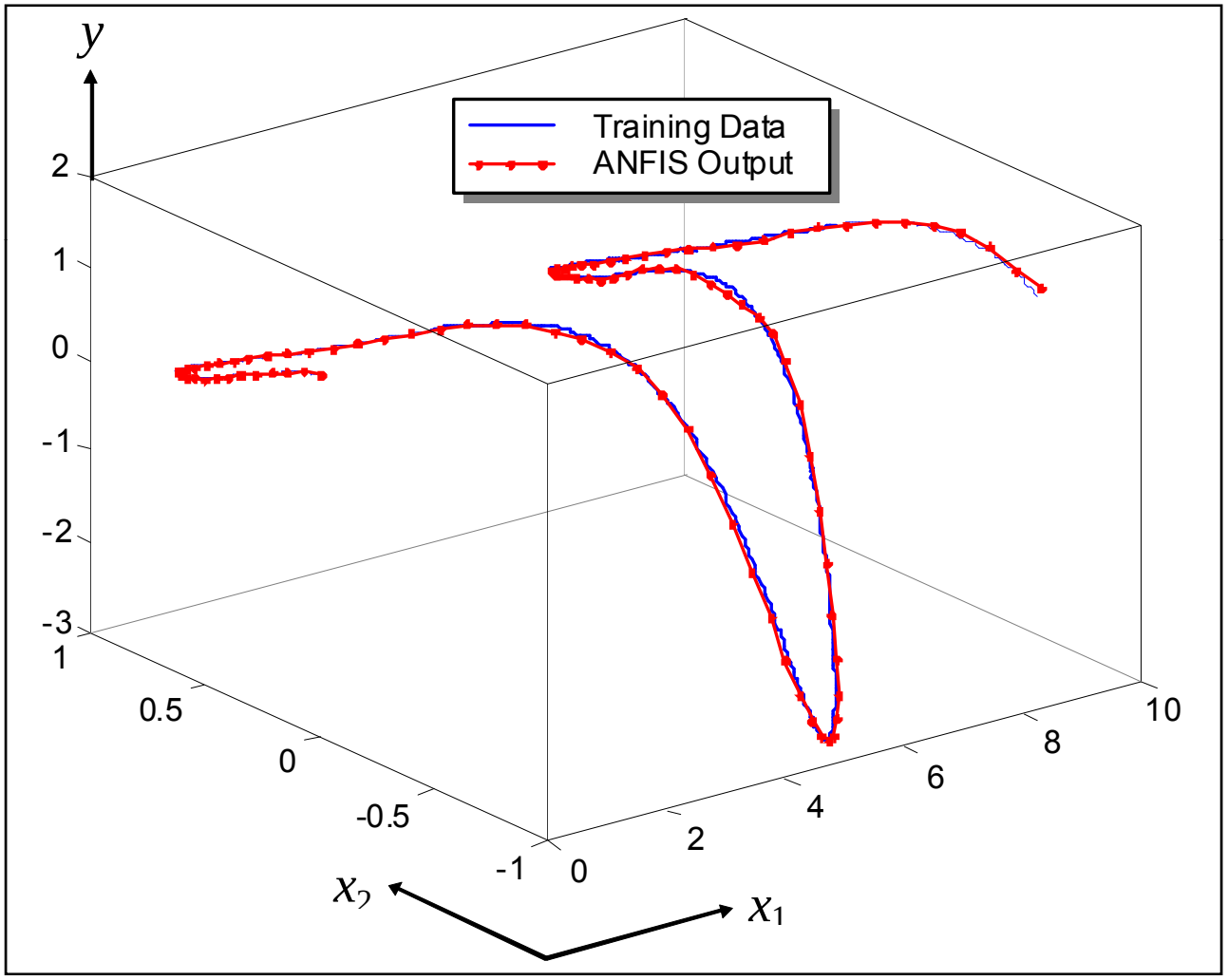
So, ...

We can achieve some improvement, but much better results are obtained when we assign three membership functions to each input variable. In this case, the ANFIS model will have nine rules, as shown in figure below.

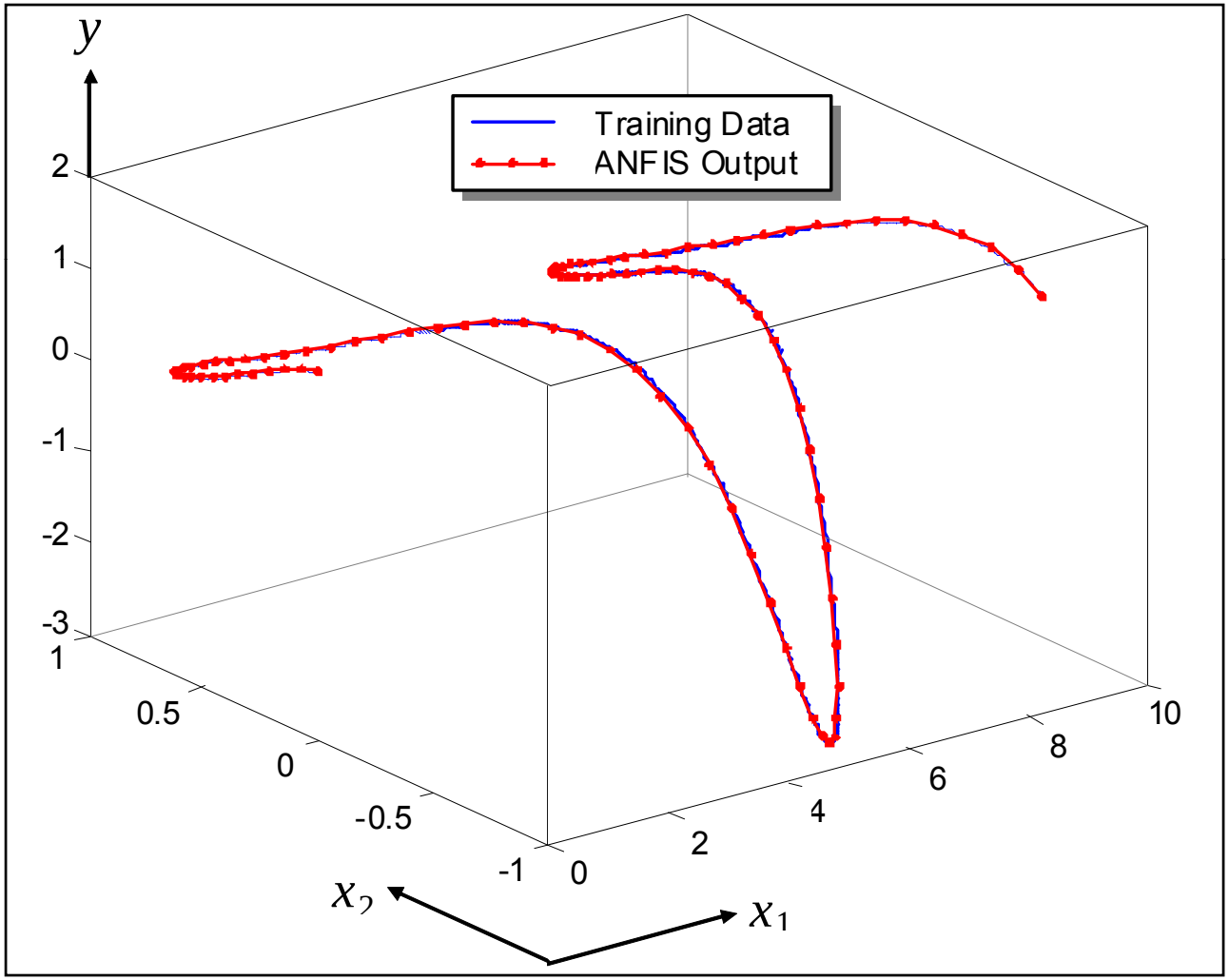
An ANFIS model with nine rules



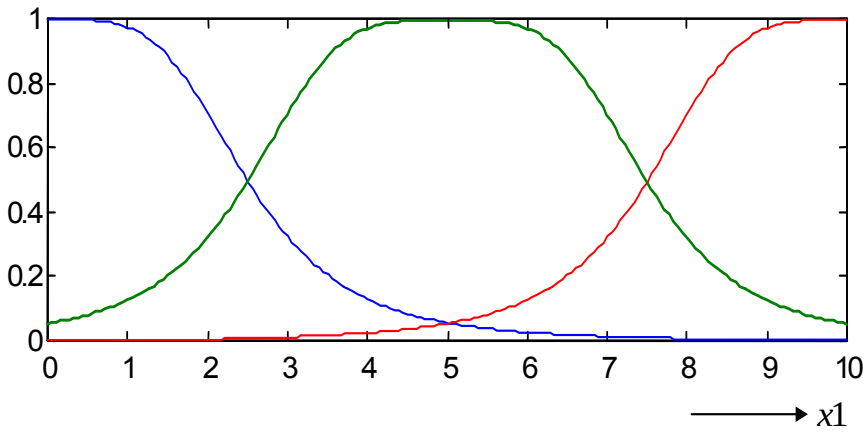
Learning in an ANFIS with three MFs assigned to each input (1 epoch)

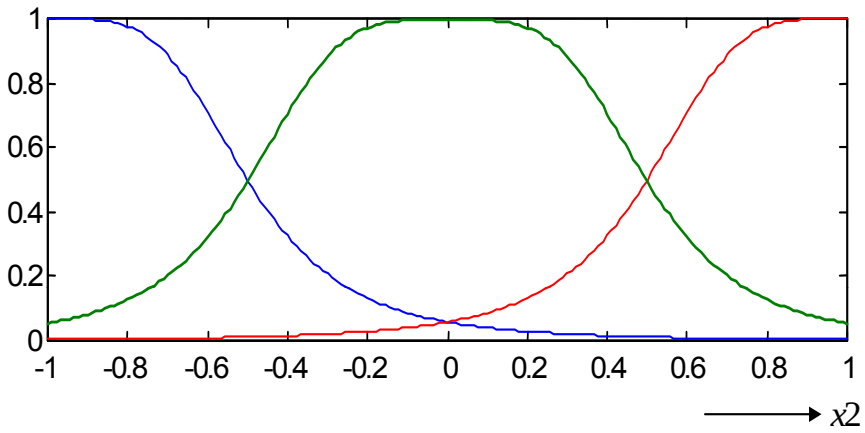


Learning in an ANFIS with two MFs assigned to each input (100 epochs)

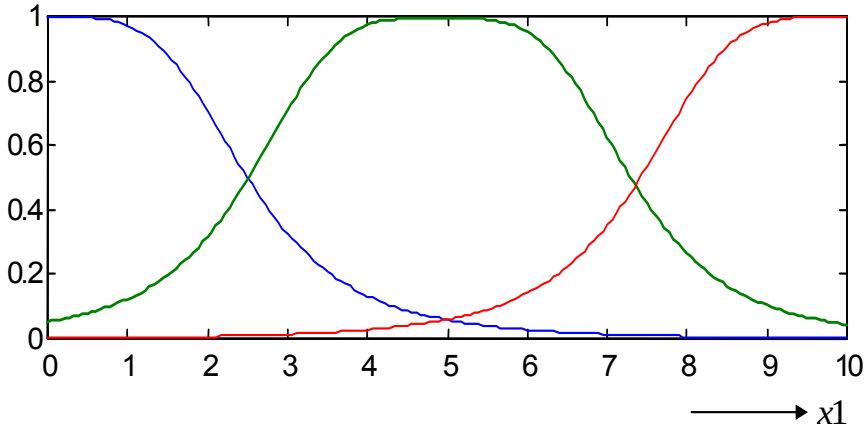


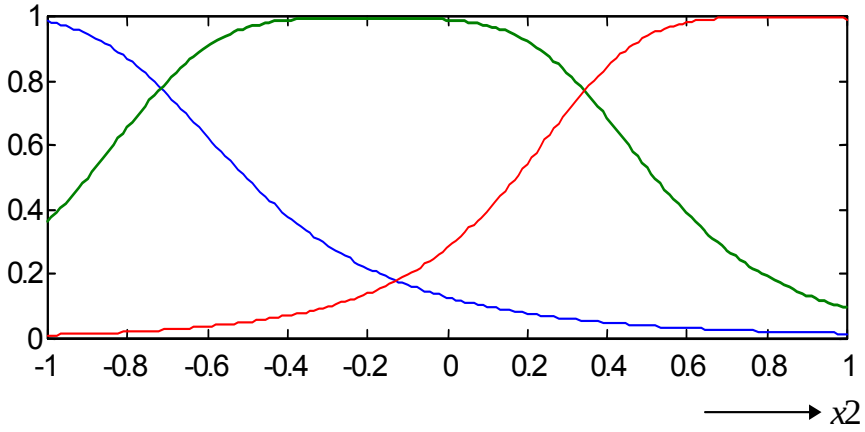
Initial and final membership functions of the ANFIS





(*a*) Initial membership functions.





(*b*) Membership functions after 100 epochs of training.

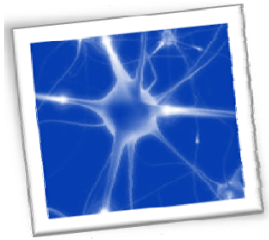
Key Features of Different Modeling Tools

Technique	Model free	Can resist outliers	Explains output	Suits small data sets	Can be adjusted for new data	Reasoning process is visible	Suits complex models	Include known facts
Least squares regression								
Neural networks								
Fuzzy Systems								
ANFIS								

Yes	
No	
Partially	

An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.

Adapted from Gray and MacDonell, 1997



Reading

- J-S R Jang and C-T Sun, Neuro-Fuzzy and Soft Computing, Prentice Hall, 1997 (Chapter 12).