Fuzzy Inference Systems

Unit 3: Mamdani, Sugeno, Tsukamoto Models, Fuzzy Modelling & ANFIS

Dr. Gopal Chandra Jana

Dept. of CSE, Sharda University
Soft Computing (CSA301) — Unit 3

July 20, 2026

Course Page:
https://www.gcjana.in/courses/shardauniversity/2601/soft-computing/

Outline

Learning Outcomes of Unit 3

After this unit, a student will be able to. . .

1. **Describe** the architecture of a Fuzzy Inference System (FIS) and its five processing steps.
2. **Compare** the Mamdani, Sugeno and Tsukamoto models and choose between them.
3. **Solve** numerical FIS problems for all three models.
4. **Explain** input-space partitioning strategies and their effect on rule count.
5. **Derive** the five-layer **ANFIS** architecture and its hybrid learning rule.
6. **Relate** ANFIS to Radial Basis Function Networks (RBFN) under functional equivalence.

Mapping to the Syllabus

A. FIS; Mamdani & Sugeno models. **B.** Tsukamoto model; input-space partitioning & fuzzy modelling. **C.** Neuro-fuzzy modelling: ANFIS architecture, hybrid learning, ANFIS–RBFN.

What is a Fuzzy Inference System (FIS)?

Definition

A **Fuzzy Inference System** is a computing framework that maps a crisp input vector to a crisp output using **fuzzy set theory**, **IF–THEN rules** and **fuzzy reasoning**.

Also known as

Fuzzy rule-based system, fuzzy expert system, fuzzy model, fuzzy associative memory, or (with a plant in the loop) a *fuzzy logic controller*.

The five functional blocks

1. Fuzzification interface
2. Knowledge base (rules + database of MFs)
3. Decision-making (inference) unit

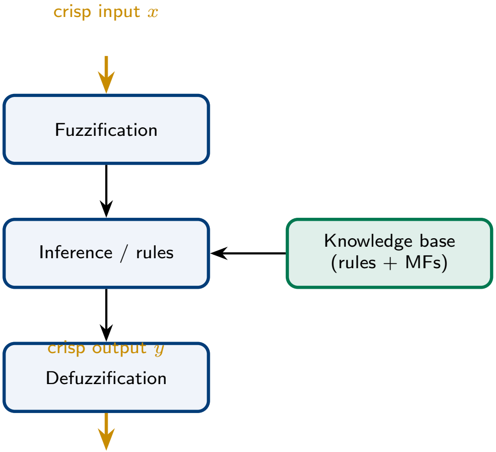
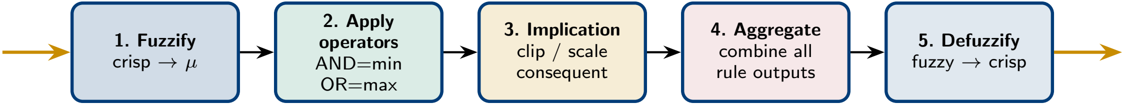


Figure: The generic FIS block diagram.

The Five Steps of Fuzzy Inference



Antecedent processing

Steps 1–2 evaluate the "IF" part and yield a single number per rule — the **firing strength** (degree of fulfilment) w_i.

Consequent processing

Steps 3–5 turn firing strengths into the "THEN" part and finally into one crisp output. *How* steps 3–5 are done is exactly what distinguishes Mamdani, Sugeno and Tsukamoto.

Mamdani Fuzzy Model

Rule form

IF x is A_i AND y is B_i THEN z is C_i

Both antecedent *and* consequent are **fuzzy sets**.
Proposed by Mamdani & Assilian (1975) for a steam-engine controller.

Mechanism

1. Firing strength $w_i = \min\{\mu_{A_i}(x), \mu_{B_i}(y)\}$
2. **Clip** (min) or **scale** (product) the output MF C_i at w_i
3. **Aggregate** all clipped C_i by max
4. **Defuzzify** (usually centroid)

Best for

Capturing *human/expert* knowledge; interpretable; the

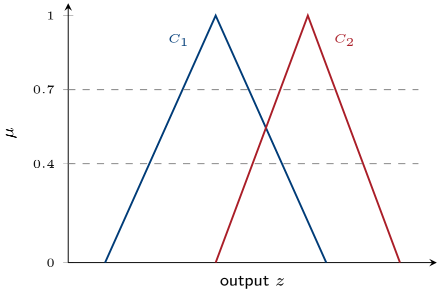


Figure: Two rules clipped at their firing strengths (0.7 and 0.4), then aggregated and defuzzified by centroid.

Numerical Example — Mamdani Centroid

Aggregated output (discrete)

On output universe $z \in \{0, 2, 4, 6, 8, 10\}$ the aggregated membership is

z	0	2	4	6	8	10
$\mu(z)$	0	0.3	0.6	1.0	0.5	0.2

Centroid

$$\sum z_i \mu_i = 0 + 0.6 + 2.4 + 6.0 + 4.0 + 2.0 = 15.0$$

$$\sum \mu_i = 0 + 0.3 + 0.6 + 1.0 + 0.5 + 0.2 = 2.6$$

$$z^* = \frac{15.0}{2.6} \approx \mathbf{5.77}$$

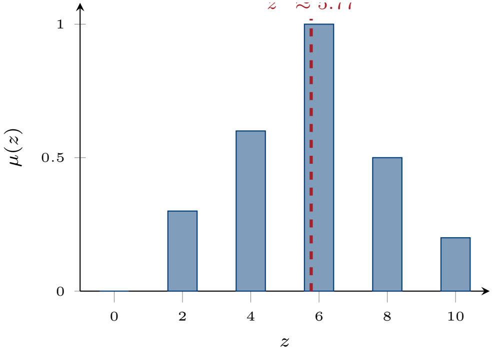


Figure: The centroid balances the aggregated area.

Sugeno (TSK) Fuzzy Model

Rule form (Takagi–Sugeno–Kang, 1985)

IF x is A_i AND y is B_i THEN $z_i = p_i x + q_i y + r_i$

The consequent is a **crisp function** of the inputs — *not* a fuzzy set.

- **Zero-order** Sugeno: $z_i = r_i$ (a constant).
- **First-order** Sugeno: z_i is linear in the inputs.

Output = weighted average

$$z^* = \frac{\sum_i w_i \, z_i}{\sum_i w_i} = \sum_i \bar{w}_i \, z_i, \qquad \bar{w}_i = \frac{w_i}{\sum_j w_j}$$

No defuzzification integral needed — just a weighted average. **Fast & compact.**

Why engineers love Sugeno

- Computationally efficient (no area integral).
- Works seamlessly with **optimisation & adaptive** techniques.
- Consequent parameters can be **learned** (basis of ANFIS).
- Guarantees output-surface continuity.

Trade-off

Less intuitive than Mamdani — a linear consequent is harder for a human to "read" than a linguistic term like *Medium*.

Numerical Example — First-Order Sugeno

Two rules, single input $x = 6$

R1: IF x is A_1 THEN $z_1 = 2x + 1$
R2: IF x is A_2 THEN $z_2 = -x + 20$

Firing strengths from the MFs at $x = 6$:

$$w_1 = 0.6, \qquad w_2 = 0.2$$

Step 1 — consequents

$$z_1 = 2(6) + 1 = 13, \qquad z_2 = -(6) + 20 = 14$$

Step 2 — normalise

$$\bar{w}_1 = \frac{0.6}{0.8} = 0.75, \quad \bar{w}_2 = \frac{0.2}{0.8} = 0.25$$

Step 3 — weighted average

$$\begin{aligned} z^* &= \bar{w}_1 z_1 + \bar{w}_2 z_2 \\ &= 0.75(13) + 0.25(14) \\ &= 9.75 + 3.5 = \mathbf{13.25} \end{aligned}$$

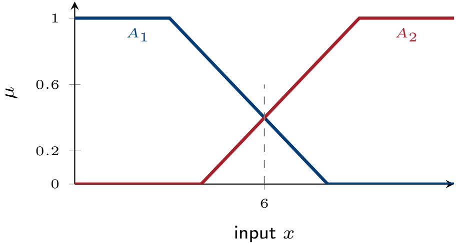


Figure: $w_1 = \mu_{A_1}(6) = 0.6$, $w_2 = \mu_{A_2}(6) = 0.2$.

Mamdani vs. Sugeno — Side by Side

Aspect	**Mamdani**	**Sugeno (TSK)**
Consequent	Fuzzy set (z is C_i)	Crisp function $z_i = p_i x + q_i y + r_i$
Output stage	Aggregation + defuzzification	Weighted average $\sum \bar{w}_i z_i$
Computation	Heavier (area integral)	Light (algebraic)
Interpretability	High — fully linguistic	Lower — linear consequents
Expressive power	Intuitive expert rules	Compact, blends with optimisation
Learning/adaptation	Awkward	**Natural** — underlies ANFIS
Typical use	Manual controller / decision support	Adaptive & model-based control

Rule of thumb

Need *human-readable* rules $\Rightarrow$ Mamdani. Need *speed or learning* $\Rightarrow$ Sugeno. They can even be mixed in one system.

Quick Check — Round 1 (FIS, Mamdani, Sugeno)

[Q1]

Name the five functional blocks of a fuzzy inference system.

[Q2]

In a Sugeno model, what form does the *consequent* of a rule take?

[Q3]

Two Sugeno rules give $z_1 = 8,\ z_2 = 2$ with firing strengths $w_1 = 0.3,\ w_2 = 0.7$. Compute the crisp output.

[Q4] True/False

"A Mamdani model needs defuzzification but a Sugeno model does not."

Think for a minute — answers on the next slide.

Answers — Round 1

[A1]

Fuzzification, **knowledge base** (rules + MF database), **inference/decision unit**, **aggregation**, **defuzzification**.

[A2]

A **crisp function of the inputs**, $z_i = p_i x + q_i y + r_i$ (first-order) or a constant $z_i = r_i$ (zero-order) — not a fuzzy set.

[A3]

$$z^* = \frac{0.3(8) + 0.7(2)}{0.3 + 0.7} = \frac{2.4 + 1.4}{1.0} = \mathbf{3.8}.$$

[A4]

True (essentially). Mamdani aggregates fuzzy sets and defuzzifies; Sugeno replaces that with a simple weighted average, so no separate defuzzification step is needed.

Tsukamoto Fuzzy Model

Rule form

IF x is A_i THEN z is C_i

where each consequent C_i is a **monotonic** (shoulder) membership function.

Mechanism

1. Compute firing strength w_i from the antecedent.
2. Because C_i is monotonic, invert it: the crisp output of rule i is
$$z_i = C_i^{-1}(w_i).$$
3. Combine by weighted average:
$$z^* = \frac{\sum_i w_i z_i}{\sum_i w_i}.$$

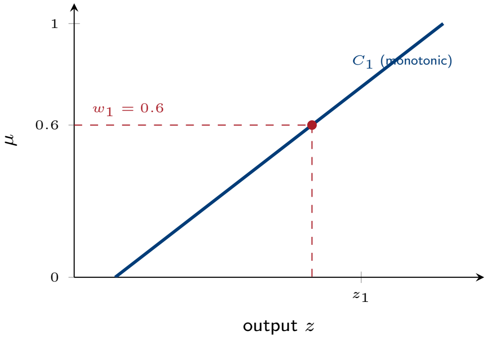


Figure: A monotonic consequent MF lets w_i be read *back* as a unique crisp $z_i = C_i^{-1}(w_i)$.

Numerical Example — Tsukamoto Model

Two rules

R1: IF x is A_1 THEN z is C_1 (small, decreasing MF)
R2: IF x is A_2 THEN z is C_2 (large, increasing MF)

At the given input the firing strengths are

$$w_1 = 0.6, \qquad w_2 = 0.2.$$

Invert the monotonic consequents

Reading each C_i^{-1} gives the crisp rule outputs

$$z_1 = C_1^{-1}(0.6) = 7, \qquad z_2 = C_2^{-1}(0.2) = 3.$$

Weighted average

$$\begin{aligned} z^* &= \frac{w_1 z_1 + w_2 z_2}{w_1 + w_2} \\ &= \frac{0.6(7) + 0.2(3)}{0.6 + 0.2} \\ &= \frac{4.2 + 0.6}{0.8} = \frac{4.8}{0.8} = \mathbf{6.0} \end{aligned}$$

Remember

Tsukamoto is mainly of *theoretical / didactic* value; in practice Mamdani (interpretable) and Sugeno (adaptive) dominate. Its restriction to monotonic consequents limits expressiveness.

Three Models at a Glance

	Mamdani	**Sugeno**	**Tsukamoto**
Consequent	Fuzzy set	Crisp linear function	Monotonic MF
Rule output	Fuzzy set	Crisp z_i	Crisp $z_i = C_i^{-1}(w_i)$
Combine by	Aggregate + defuzzify	Weighted average	Weighted average
Cost	High	Low	Low
Interpretability	Highest	Medium	Medium
Main use	Control, expert systems	Adaptive/learning (ANFIS)	Teaching, special cases

Unifying view

All three share *fuzzification* and *firing-strength* computation; they differ only in how the **consequent** is represented and **combined**.

Fuzzy Modelling: The Two Design Tasks

Structure identification

- Which **input variables** matter?
- How to **partition** each input space (how many MFs)?
- How many **rules**, and of what form?

Parameter identification

- **Premise** parameters: MF shapes/positions.
- **Consequent** parameters: the p_i, q_i, r_i (Sugeno).
- Tuned by expert knowledge *or* learned from data.

The core question

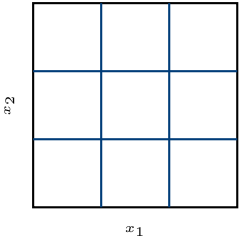
How should we carve up the input space so that a small set of rules covers it well?

Curse of dimensionality

With n inputs and m MFs each, a full grid needs m^n rules.
$n = 3,\ m = 5 \Rightarrow 5^3 =$ **125** rules;
$n = 5,\ m = 5 \Rightarrow 5^5 =$ **3125** rules.
Partitioning strategy is therefore *critical*.

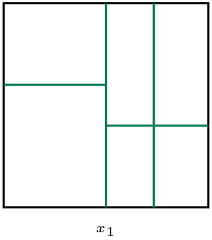
Input-Space Partitioning Strategies

Grid partition



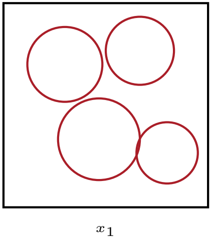
Simple, interpretable, but rules $= m^n$ (explodes).

Tree partition



Fewer regions; flexible; MFs harder to interpret.

Scatter partition



Covers only the region the *data* occupies; from clustering.

Guidance

Grid for few inputs; **tree** to curb rule growth; **scatter** (via subtractive/fuzzy c-means clustering) when only sparse data defines the operating region. ANFIS commonly uses grid (few inputs) or scatter (many inputs).

Quick Check — Round 2 (Tsukamoto & Partitioning)

[Q1]

What special property must a Tsukamoto consequent MF have, and why?

[Q2]

Tsukamoto rules give $z_1 = 10,\ z_2 = 4$ with $w_1 = 0.25,\ w_2 = 0.75.$ Find z^*.

[Q3]

A system has 4 inputs, each with 3 MFs. How many rules does a full *grid* partition need?

[Q4]

Which partitioning strategy is generated by *clustering* the data?

Answers — Round 2

[A1]

It must be **monotonic** (a shoulder function). Monotonicity makes the inverse $C_i^{-1}(w_i)$ *unique*, so each rule yields a single crisp output z_i.

[A2]

$z^* = \frac{0.25(10) + 0.75(4)}{0.25 + 0.75} = \frac{2.5 + 3.0}{1.0} =$ **5.5**.

[A3]

Grid rules $= m^n = 3^4 =$ **81** rules.

[A4]

Scatter partitioning — clusters (e.g. subtractive clustering or fuzzy c-means) become rules, covering only the data-populated region.

Motivation: Why Neuro-Fuzzy?

Fuzzy logic

+ Interpretable IF–THEN rules
+ Uses expert knowledge
– **No learning** — MFs tuned by hand

Neural networks

+ **Learn** from data
+ Adapt, generalise
– Black box — no interpretability

Neuro-fuzzy = best of both

A **neuro-fuzzy** system expresses a fuzzy system as a **layered network** so its parameters can be **learned** by neural methods — keeping the rules readable *and* making them trainable.

The landmark: ANFIS

Adaptive Neuro-Fuzzy Inference System (Jang, 1993): a Sugeno-type FIS realised as a 5-layer feed-forward network with a *hybrid* learning rule.

This directly answers the Unit 2 cliff-hanger: "rules that tune themselves."

ANFIS — The Reference Example (Two Rules)

A first-order Sugeno system with two rules

$$\text{R1: IF } x \text{ is } A_1 \text{ AND } y \text{ is } B_1 \text{ THEN } f_1 = p_1 x + q_1 y + r_1$$
$$\text{R2: IF } x \text{ is } A_2 \text{ AND } y \text{ is } B_2 \text{ THEN } f_2 = p_2 x + q_2 y + r_2$$

Premise (nonlinear) parameters

Shape the input MFs, e.g. generalised bell:

$$\mu_{A_i}(x) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b_i}}$$

Tunable set $\{a_i, b_i, c_i\}$.

Consequent (linear) parameters

The coefficients $\{p_i, q_i, r_i\}$ of each rule's linear output.

Two parameter sets $\Rightarrow$ two learning phases

This split is exactly what the **hybrid** algorithm exploits (next slides).

ANFIS Architecture — Five Layers

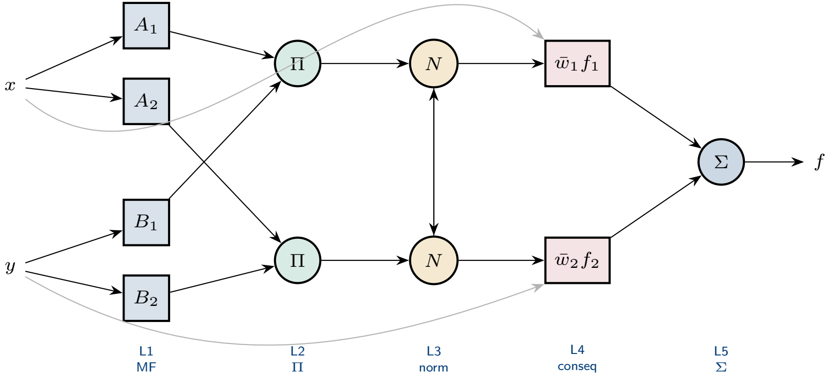


Figure: ANFIS as a 5-layer adaptive network. Square nodes = adaptive (have parameters); circle nodes = fixed.

ANFIS — Layer-by-Layer Mathematics

Layer 1 — Fuzzify (adaptive)

Each node outputs a membership grade:

$$O_i^1 = \mu_{A_i}(x) \quad \text{or} \quad \mu_{B_i}(y).$$

Parameters $\{a_i, b_i, c_i\}$ = **premise params**.

Layer 2 — Firing strength (fixed Π)

$$O_i^2 = w_i = \mu_{A_i}(x)\,\mu_{B_i}(y).$$

Layer 3 — Normalise (fixed N)

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2}.$$

Layer 4 — Consequent (adaptive)

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i\,(p_i x + q_i y + r_i).$$

Parameters $\{p_i, q_i, r_i\}$ = **consequent params**.

Layer 5 — Output (fixed Σ)

$$O^5 = f = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}.$$

Crucial observation

For *fixed* premise parameters, f is **linear** in the consequent parameters $\{p_i, q_i, r_i\}$. This is the key that makes hybrid learning work.

The Output is Linear in the Consequent Parameters

Expand the ANFIS output

$$f = \bar{w}_1 f_1 + \bar{w}_2 f_2 = \bar{w}_1(p_1 x + q_1 y + r_1) + \bar{w}_2(p_2 x + q_2 y + r_2)$$

Grouping the unknowns:

$$f = (\bar{w}_1 x)p_1 + (\bar{w}_1 y)q_1 + (\bar{w}_1)r_1 + (\bar{w}_2 x)p_2 + (\bar{w}_2 y)q_2 + (\bar{w}_2)r_2.$$

Why this matters

The coefficients in brackets are *known* once the inputs and premise parameters are fixed. Hence f is a **linear combination** of the six consequent parameters — solvable by **least squares**.

Matrix form

For P training pairs,

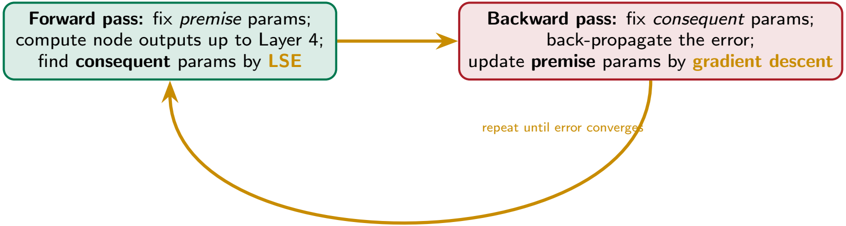
$$\mathbf{A}\,\boldsymbol{\theta} = \mathbf{y},$$

$\boldsymbol{\theta} = [p_1, q_1, r_1, p_2, q_2, r_2]^\top$; solve $\hat{\boldsymbol{\theta}} = (\mathbf{A}^\top \mathbf{A})^{-1}\mathbf{A}^\top \mathbf{y}$ (LSE).

ANFIS Hybrid Learning Algorithm

Two passes per epoch

Each training epoch has a **forward pass** and a **backward pass**.



Why hybrid beats pure gradient descent

The consequent params are solved *optimally* (LSE) each epoch, so only the premise params are left to gradient descent — **faster, more stable convergence**.

Summary table

	Premise	Consequent
Forward	fixed	**LSE**

ANFIS in Action — Function Approximation

What ANFIS learns

Given input–output data, ANFIS **adjusts** both the MF shapes (premise) and the linear consequents so its output surface fits the data.

Typical applications

- Non-linear **function approximation**
- Time-series **prediction** (e.g. chaotic Mackey–Glass)
- Adaptive **control** & system identification
- Channel equalisation, signal processing

As epochs increase, the fitted curve hugs the target and the training error falls monotonically.

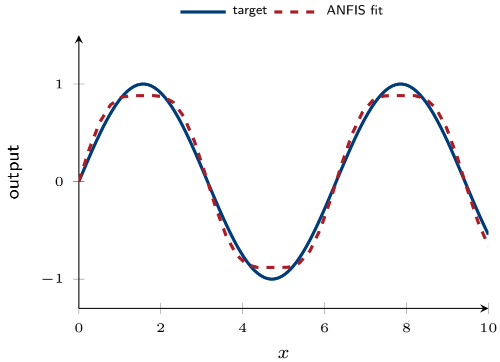


Figure: After training, the ANFIS output (dashed) closely tracks the target function.

Cross-Fertilisation: ANFIS and RBFN

Radial Basis Function Network

An RBFN computes

$$f(\mathbf{x}) = \frac{\sum_i w_i \, R_i(\mathbf{x})}{\sum_i R_i(\mathbf{x})}, \quad R_i(\mathbf{x}) = \exp\Big(-\frac{\|\mathbf{x}-\mathbf{c}_i\|^2}{2\sigma_i^2}\Big)$$

i.e. a weighted sum of localised **radial basis functions**.

Functional equivalence (Jang & Sun, 1993)

A normalised ANFIS and an RBFN are **functionally equivalent** when:

- both use the same number of basis / rules,
- the receptive field of each RBF matches a rule's firing strength,
- the output combination is identical (normalised weighted sum).

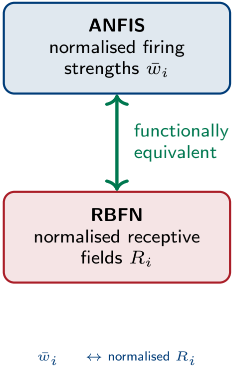


Figure: The two models are two views of one computation.

Why the Equivalence is Useful

Learning transfers both ways

- RBFN **fast training** tricks (e.g. clustering for centres, LSE for weights) carry over to ANFIS.
- ANFIS **interpretability** (linguistic rules) lends meaning to otherwise opaque RBF centres.
- Results proven for one model **apply** to the other.

Design consequences

- Initialise MF centres by **k-means / subtractive clustering** (an RBFN idea).
- Use **LSE + gradient descent** hybrid training (shared by both).
- Choose the model by *convenience*: same expressive power.

Take-away

Neuro-fuzzy and RBF communities solved the *same* problem in different languages; recognising this lets us reuse the best of both.

Advantages & Limitations of ANFIS

Advantages

- Learns MFs & rules **from data** — no hand tuning.
- Keeps **interpretable** Sugeno rules.
- **Fast, stable** hybrid convergence (LSE + GD).
- Excellent **non-linear** approximation & prediction.
- Bridges fuzzy, neural and RBF paradigms.

Limitations

- **Curse of dimensionality**: rules $= m^n$ (grid).
- Usually **single output**; MISO by design.
- Only **Sugeno-type** (not Mamdani).
- Needs enough **representative data**.
- MF **type/number** chosen empirically.
- Can **over-fit** without validation.

Mitigations: scatter partitioning + clustering to limit rules; cross-validation to control over-fitting.

Quick Check — Round 3 (ANFIS)

[Q1]

How many layers does ANFIS have, and which layers are *adaptive*?

[Q2]

In the hybrid rule, which optimisation updates the *consequent* parameters, and which updates the *premise* parameters?

[Q3]

Why can the consequent parameters be found by least squares?

[Q4] MCQ

ANFIS is functionally equivalent to which network under matching conditions?
(a) CNN (b) RBFN (c) LSTM (d) SOM

Answers — Round 3

[A1]

Five layers. The *adaptive* ones are **Layer 1** (premise MF parameters) and **Layer 4** (consequent parameters); Layers 2, 3, 5 are fixed.

[A2]

Forward pass: consequent params by **least-squares estimation (LSE)**; *backward pass*: premise params by **gradient descent** (back-propagation).

[A3]

Because, with premise parameters fixed, the ANFIS output $f = \sum_i \bar{w}_i f_i$ is a **linear combination** of the consequent parameters $\{p_i, q_i, r_i\}$ — a linear least-squares problem.

[A4]

(b) RBFN — a Radial Basis Function Network (under matching basis count and normalised combination).

Summary of Unit 3

What we covered

1. **FIS** — the five blocks and the antecedent→consequent flow.
2. **Mamdani** — fuzzy consequents, aggregation + centroid defuzzification.
3. **Sugeno (TSK)** — crisp linear consequents, weighted-average output.
4. **Tsukamoto** — monotonic consequents, crisp $z_i = C_i^{-1}(w_i)$.
5. **Fuzzy modelling** — structure & parameter identification; grid, tree and scatter partitioning; the curse of dimensionality.
6. **ANFIS** — 5-layer Sugeno network; premise & consequent parameters.
7. **Hybrid learning** — LSE (forward) + gradient descent (backward).
8. **ANFIS $\equiv$ RBFN** — functional equivalence and shared training ideas.

ANFIS turns hand-crafted fuzzy rules into rules that learn.

Practice Questions (Exam Oriented)

Short answer (2–5 marks)

1. Draw the block diagram of a fuzzy inference system and name each block.
2. Differentiate Mamdani and Sugeno models on any four points.
3. Why must a Tsukamoto consequent MF be monotonic?
4. Define premise and consequent parameters in ANFIS.
5. State the curse of dimensionality for grid partitioning with a formula.

Long answer (8–10 marks) — include numerics

6. For a 2-rule first-order Sugeno system with $w_1=0.4, w_2=0.6$, $z_1=3x+1, z_2=x+5$ and $x=2$, compute the crisp output.
7. Derive the ANFIS output and show it is linear in the consequent parameters.
8. Explain the hybrid learning algorithm (forward + backward passes) in detail.
9. Solve a Tsukamoto example given firing strengths and inverted consequents.
10. Explain ANFIS–RBFN functional equivalence and why it is useful.

Think & Explore (Take-Home)

Numeric drill

Build a 2-input, 2-MF-per-input *grid* Sugeno system on paper:

- Write all $2^2 = 4$ rules.
- Pick bell MFs, evaluate firing strengths at one test point.
- Compute the weighted-average output by hand.

Reading & reflection

- Jang, Sun & Mizutani, *Neuro-Fuzzy and Soft Computing* — ANFIS chapter.
- Jang (1993), "ANFIS: Adaptive-Network-based Fuzzy Inference System," *IEEE Trans. SMC*.
- Prepare for **Unit 4**: Genetic Algorithms and swarm optimisation.

Coding (self-learning)

Train an ANFIS on the Mackey–Glass or $\sin(x)$ data using Python (`anfis`/`scikit-fuzzy` or `PyTorch`); plot RMSE vs. epochs.

Reminder

Assessment-2 covers Units 3 & 4; Assignment 2 covers Units 3, 4 & 5 — handwritten, step-by-step, single PDF via the course page.

References

Textbooks (as per the course page)

1. J.-S. R. Jang, C.-T. Sun and E. Mizutani, *Neuro-Fuzzy and Soft Computing*, Prentice Hall. **[Main text for Unit 3]**
2. G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall.
3. T. J. Ross, *Fuzzy Logic with Engineering Applications*, McGraw Hill.

Seminal papers

4. J.-S. R. Jang, "ANFIS: Adaptive-Network-based Fuzzy Inference System," *IEEE Trans. Systems, Man & Cybernetics*, 23(3):665–685, 1993.
5. T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. SMC*, 1985.
6. J.-S. R. Jang and C.-T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems," *IEEE Trans. Neural Networks*, 1993.

NPTEL: *Soft Computing*, IIT Kharagpur · Course page:
https://www.gcjana.in/courses/shardauniversity/2601/soft-computing/

Thank You

Any Questions?

"A fuzzy system with a learning algorithm is a neural network;
a neural network with linguistic rules is a fuzzy system."

— the spirit of neuro-fuzzy computing

Post your doubts on the course page →
https://www.gcjana.in/courses/shardauniversity/2601/soft-computing/#Post-doubts