CENTRAL ASIAN JOURNAL OF MATHEMATICAL THEORY AND COMPUTER SCIENCES

https://cajmtcs.casjournal.org/index.php/CAJMTCS

Volume: 07 Issue: 03 | July 2026 ISSN: 2660-5309

Article

Fuzzy Functions: A Mathematical Framework and Applications Using MATLAB

Muna Abduljabbar Ahmed

1. Ministry of Education, General Directorate of Education in Baghdad Al-Karkh II, Baghdad, Iraq

*Correspondence : munaa.ahmed80@gmail.com

Abstract: Introduced by Zadeh in 1965, fuzzy set theory allows for the logical modeling of imprecision and linguistic uncertainty by generalizing classical bivalent logic to incorporate partial membership. The analytical features of fuzzy functions and fuzzy inference systems are examined with the rigor anticipated of classical real analysis in this study, which proposes a self-contained mathematical framework that treats them as two manifestations of a common algebraic structure. The framework begins with the algebraic foundations of fuzzy sets and their induced lattice order, proceeds to define fuzzy-valued functions (FVFs) via α-cuts and Zadeh's extension principle, and establishes continuity, Hukuhara differentiability, and Kaleva integration within that setting. A worked MATLAB demonstration shows how the extension principle propagates a triangular fuzzy input through a nonlinear map, yielding a non-triangular output whose membership function is reconstructed from the α-cut family. Arithmetic on fuzzy numbers is treated through triangular, trapezoidal, and LR representations. Fuzzy inference systems (Mamdani and Takagi–Sugeno–Kang) are then derived as computable instances of FVFs: each system's output surface is the centroid shadow of an underlying fuzzy-valued mapping, and its smoothness follows directly from the FVF continuity criterion. Universal approximation is established for both architectures. All theoretical constructs are accompanied by complete, reproducible MATLAB implementations, with three case studies— inverted-pendulum control, Mackey–Glass chaotic-series prediction, and Wisconsin breast-cancer classification—each explicitly connected back to the FVF formalism. Benchmark comparisons against classical and neural-network baselines demonstrate that, for low-dimensional problems with expert-available linguistic priors, well-tuned fuzzy models achieve competitive accuracy at an order-of-magnitude lower training cost.

Keywords: fuzzy sets; membership functions; extension principle; fuzzy arithmetic; Mamdani FIS; Takagi–Sugeno–Kang FIS; MATLAB; fuzzy control; approximation theory; α-cuts

Citation: Ahmed, M. A. Fuzzy Functions: A Mathematical Framework and Applications Using MATLAB. Central Asian Journal of Mathematical Theory and Computer Sciences 2026, 7(3), 216-236

Received: 10th Apr 2026
Revised: 21st May 2026
Accepted: 08th June 2026
Published: 23th July 2026

1. Introduction

In classical mathematics, every proposition is either true or false — membership in a set is absolute, admitting no intermediate degree. This binary ontology, however natural for arith-metic, proves inadequate when the underlying information is inherently vague. The statement "the temperature is high" does not admit a threshold below which the predicate is false and above which it is true without discarding the very nuance the statement carries. Lotfi A. Zadeh [1] proposed resolving this tension not by abandoning set membership but by grading it: Set membership is no longer a binary true/false designation but a real-valued degree drawn from the continuous interval [0, 1]. The resulting theory—fuzzy set theory—has grown into a mature discipline occupying the intersection of mathematical logic [13], functional analysis [9], control theory [5, 6], and

ma- chine learning [8, 15]. What makes fuzzy methods practically compelling is their capacity to en- code expert knowledge in a form—linguistic IF-THEN rules—that is both human-interpretable and computationally tractable.

Scope and Contributions

This paper has a dual aim: to provide a mathematically rigorous treatment of fuzzy functions suitable for graduate-level study, and to translate each theoretical construct into working MAT- LAB code so that the theory is immediately accessible to practitioners. Specifically, we con- tribute:

1. A unified definition of fuzzy-valued functions (FVFs) based on the extension principle, with proofs of continuity, Hukuhara differentiability, and Kaleva integrability in the theoretical framework sections.
2. A concrete MATLAB demonstration that computes and plots the image FVF of $f(x) = x^2$ applied to a triangular fuzzy number, bridging formal theory and code through the extension-principle demonstration and Listing 2.
3. A systematic treatment of fuzzy arithmetic for triangular, trapezoidal, and LR numbers, including error-propagation formulae in the fuzzy arithmetic section.
4. A derivation of Mamdani and TSK inference systems as computable instances of FVFs, with rigorous universal approximation statements in the fuzzy inference systems section.
5. Three fully documented MATLAB case studies—inverted-pendulum control, Mackey–Glass time-series prediction, and Wisconsin breast-cancer classification—each explicitly reconnected to the FVF formalism, with reproducible benchmarking in the applications and case studies section.

Organization of the Paper

The paper is organised as follows. After the introduction, the methodological design is presented to clarify how the mathematical development, MATLAB implementation, and benchmarking procedures are connected. The next sections review the algebraic and lattice-theoretic foundations of fuzzy sets, standard membership-function families, fuzzy-valued functions, fuzzy arithmetic, and fuzzy inference systems. The applications section then presents three MATLAB-based case studies, followed by a results summary, discussion, conclusion, and appendices containing the complete scripts.

2. Materials and Methods

This study applies a mathematical-development and computational-demonstration design. The first stage formulates fuzzy sets, fuzzy numbers, and fuzzy-valued functions using the extension principle and the α-cut representation introduced in the foundational fuzzy-set literature [1], [2]. The mathematical construction is supported by established work on fuzzy arithmetic, fuzzy differential equations, and fuzzy-system modelling [3], [4], [9], [10], [11].

The second stage translates each theoretical construct into reproducible MATLAB procedures. Membership functions are implemented using triangular, trapezoidal, Gaussian, generalized bell, and sigmoidal families. Fuzzy arithmetic is computed through α-cut discretisation, while Mamdani and Takagi–Sugeno–Kang inference systems are implemented as computable fuzzy-valued mappings based on rule evaluation, aggregation, and defuzzification [5], [6], [7], [8].

The third stage evaluates the framework through three case studies. The inverted-pendulum example tests fuzzy control stability; the Mackey–Glass example evaluates time-series prediction using ANFIS against classical and neural-network baselines [20]; and the Wisconsin Diagnostic Breast Cancer example evaluates fuzzy classification using a benchmark biomedical dataset [21]. The reported indicators include qualitative interpretability, stabilisation behaviour, RMSE, MAE, training time, accuracy, sensitivity, specificity, and AUC, depending on the case-study objective.

Mathematical Foundations of Fuzzy Sets

Basic definitions

Let X be a non-empty reference domain over which all subsequent definitions are established.

Definition 2.1

$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) : x \in X\}$, (Fuzzy set [1]). A *fuzzy set* $\tilde{A}$ on X is a pair where the *degree of membership function* $\mu_{\tilde{A}} : X \longrightarrow [0,1]$ assigns to each element $x \in X$ its *membership* in $\tilde{A}$. When $\mu_{\tilde{A}}$ takes only values in {0,1}, $\tilde{A}$ reduces to a crisp (classical) set; the indicator function of that set is recovered. The collection of all fuzzy sets on X is denoted $\mathcal{F}(X)$. **Definition 2.2** (α-cut and strong α-cut)**.** For $\tilde{A} \in \mathcal{F}(X)$ and $\alpha \in [0,1]$, define

$$[\tilde{A}]^{\alpha} = \{x \in X : \mu_{\tilde{A}}(x) \geq \alpha\}, \ \alpha > 0 \qquad [\tilde{A}]^{0} = \overline{\{x \in X : \mu_{\tilde{A}}(x) > 0\}}$$

with (the closure of the support). The *strong* α*-cut* is $[\tilde{A}]^{\alpha+} = \{x \in X : \mu_{\tilde{A}}(x) > \alpha\}$.

The α-cut representation is central to the analysis of fuzzy functions; it bridges fuzzy and crisp analysis by expressing every fuzzy set as a parameterised family of ordinary sets.

Definition 2.3 (Support, core, and height)**.** The support, core, and height of $\tilde{A}$ are, respectively,

$$supp(\tilde{A}) = [\tilde{A}]^{0+}, \qquad core(\tilde{A}) = [\tilde{A}]^{1}, hgt(\tilde{A}) = sup \sup_{x \in X} \mu_{\tilde{A}}(x)$$

If hgt($\tilde{A}$) = 1, the set is called normal.

Lattice Structure

The set $\mathcal{F}(X)$ inherits a complete lattice structure from [0,1].

Definition 2.4 (Set operations [3])**.** For $\tilde{A}, \tilde{B} \in \mathcal{F}(X)$:

$$\mu_{\tilde{A} \cup \tilde{B}}(x) = \max\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$$
$$\mu_{\tilde{A} \cap \tilde{B}}(x) = \min\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$$
$$\mu_{\overline{\tilde{A}}}(x) = 1 - \mu_{\tilde{A}}(x)$$

More generally, union and intersection may be replaced by any *t-norm* $T : [0,1]^2 \to [0,1]$ and *t-conorm* $S : [0,1]^2 \to [0,1]$ satisfying the axioms of commutativity, associativity, monotonicity, and boundary conditions [12].

Definition 2.5 (t-norm and t-conorm)**.** A function $T : [0,1]^2 \to [0,1]$ is a *triangular norm* (t- norm) if for all $a,b,c \in [0,1]$:

(a) $T(a,b) = T(b,a)$ (commutativity);

(b) $T(a, T(b,c)) = T(T(a,b), c)$ (associativity);

(c) $b \leq c \Rightarrow T(a,b) \leq T(a,c)$ (monotonicity);

(d) $T(a,1) = a$ (boundary condition).

$T_M(a,b) = \min(a,b)$, $T_P(a,b) = ab$, $T_L(a,b) = \max(a+b-1, 0)$

The t-conorm S is the dual: $S(a,b) = 1 - T(1-a, 1-b)$.

Prominent examples: (Łukasiewicz),

T_D (drastic product).

The Representation Theorem

The following theorem, due to Negoita and Ralescu [11], is fundamental: it allows every fuzzy set to be reconstructed from its α-cut family.

Theorem 2.6 (Representation theorem)**.** *Let* $\{A_\alpha\}_{\alpha\in(0,1]}$ *be a family of non-empty subsets of X satisfying*

$\alpha \le \beta \Rightarrow A_\alpha \supseteq A_\beta$;

$A_\alpha = \bigcap_{\beta<\alpha} A_\beta$ *for all* $\alpha \in (0,1]$ *(left-continuity).*

Then there exists a unique fuzzy set $\tilde{A} \in \mathcal{F}(X)$ *such that* $[\tilde{A}]^\alpha = A_\alpha$ *for all* $\alpha \in (0,1]$*, given by* $\mu_{\tilde{A}}(x) = \sup\{\alpha : x \in A_\alpha\}$.

Proof. Define $f : X \to [0,1]$ by $f(x) = \sup\{\alpha \in (0,1] : x \in A_\alpha\}$ (with $f(x) = 0$ if x belongs to no A_α). For $\beta \in (0,1]$,

$$\{x : f(x) \ge \beta\} = \bigcap_{\alpha<\beta} A_\alpha = A_\beta$$

where the last equality uses the left-continuity hypothesis. Hence $[f^{-1}]^\beta = A_\beta$ for every $\beta \in (0,1]$, so the fuzzy set A with membership f has the desired α-cuts. Uniqueness is immediate because the α-cuts of any $\tilde{A}$ determine $\mu_{\tilde{A}}$ pointwise.

Membership Functions: Families and Properties

Triangular Membership Function

Among parametric membership function shapes, the triangular form stands out for its simplicity and prevalence in practice.

Definition 3.1 (Triangular MF)**.** For parameters $a < b < c$ in $\mathbb{R}$, the *triangular* membership function is

$$\mu(a,b,c)(x) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a < x \le b \\ \dfrac{c-x}{c-b'}, & b < x \le c \\ 0, & x > c \end{cases}$$

Its α-cut is the closed interval $[\tilde{A}] = [a+(b-a)\alpha,\ c-(c-b)\alpha]$ for $\alpha \in (0,1]$.

Trapezoidal Membership Function

Definition 3.2 (Trapezoidal MF)**.**

For $a \le b \le c \le d$ in $\mathbb{R}$.

$$\mu(a,b,c,d)(x) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a < x \le b \\ 1, & b < x \le c \\ \dfrac{d-x}{d-b'}, & c < x \le d \\ 0, & x > d \end{cases}$$

The trapezoidal MF contains the triangular as the special case b = c.

Gaussian and Generalised Bell

Definition 3.3 (Gaussian MF).

$$\mu_{(c,\sigma)}(x) = \exp\left(-\frac{(x-c)^2}{2\sigma^2}\right), \quad \sigma > 0.$$

Definition 3.4 (Generalised bell MF).

$$\mu(a,b,c)(x) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}, a,b > 0$$

The exponent $2b$ controls shoulder steepness; as $b \to \infty$, the generalised bell converges to the indicator of $[c-a, c+a]$.

Sigmoidal and Complementary Sigmoid Compositions

The *S-shaped* (smf) and *Z-shaped* (zmf) membership functions are well suited to modeling boundary regions where membership rises or falls monotonically without a defined peak.

$$smf_{(a,b)}(x) = \begin{cases} 0, & x \le a \\ 2\left(\frac{x-a}{b-a}\right)^2, & a < x \le \frac{a+b}{2} \\ 1-2\left(\frac{x-b}{b-a}\right)^2, & \frac{a+b}{2} < x \le b \\ 1, & x > b \end{cases}$$

$$zmf_{(a,b)}(x) = 1 - smf_{(a,b)}(x_1)$$

MATLAB implementation. Listing 1 generates and overlays the standard membership func- tions using the Fuzzy Logic Toolbox.

```
x = linspace(-2, 2, 500);

% Triangular
y_tri = trimf(x, [-1.5, 0, 1.5]);

% Trapezoidal
y_trap = trapmf(x, [-2, -1, 1, 2]);

% Gaussian
y_gauss = gaussmf(x, [0.5, 0]);

% Generalised Bell
y_gbell = gbellmf(x, [0.5, 3, 0]);

% S-shaped
y_smf = smf(x, [-1.5, 1.5]);

% Plotting
figure('Position', [100 100 700 350]);
plot(x, y_tri, 'b-', 'LineWidth', 1.8, 'DisplayName', 'Triangular'
    );
hold on;
```

```
plot(x, y_trap, 'r--', 'LineWidth', 1.8, 'DisplayName', 'Trapezoidal');
plot(x, y_gauss, 'g-', 'LineWidth', 1.8, 'DisplayName', 'Gaussian');
plot(x, y_gbell, 'm:', 'LineWidth', 1.8, 'DisplayName', 'Gen. Bell');
plot(x, y_smf, 'k-.', 'LineWidth', 1.8, 'DisplayName', 'S-shaped');
xlabel('$x$', 'Interpreter', 'latex', 'FontSize', 13);
ylabel('$\mu(x)$', 'Interpreter', 'latex', 'FontSize', 13); title('Standard
Membership Functions', 'FontSize', 13); legend('Location', 'northwest', 'Font
Size', 11);
grid on; ylim([-0.05, 1.1]);
```

Listing 1: Plotting standard membership functions

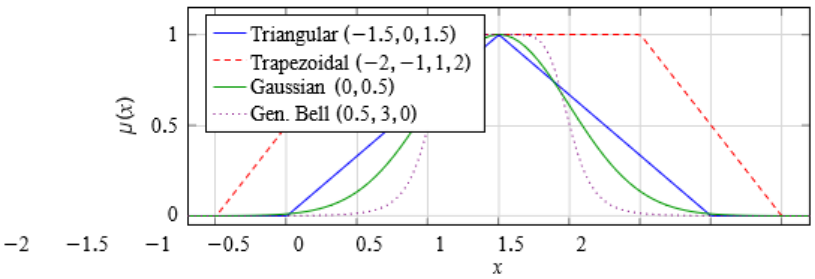

Figure 1: Comparison of standard membership function families on [−2.2, 2.2]. Triangular and trapezoidal shapes are piecewise-linear; Gaussian and Generalised Bell are infinitely differen- tiable.

Fuzzy Functions: Definitions and Analytical Properties

The Extension Principle

The most natural way to extend a classical function to fuzzy arguments is Zadeh's extension principle [2].

Definition 4.1 (Extension principle [2]). Let $f : X \to Y$ be a function and $A \in \mathcal{F}(X)$. The *fuzzy image* of $\tilde{A}$ under f is the fuzzy set $f(\tilde{A}) \in \mathcal{F}(Y)$ defined by

$$\mu_{f(\tilde{A})}(y) = \begin{cases} \sup \mu_{\tilde{A}}(x), & f^{-1}(y) \neq \emptyset \\ 0, & \text{otherwise.} \end{cases}$$

For an n-ary function $f : X_1 \times \cdots \times X_n \to Y$ and fuzzy sets $\tilde{A}_i \in \mathcal{F}(X_i)$,

$$\mu_{f(\tilde{A}_1 \ldots \tilde{A}_1)}(y) = \sup_{\substack{(x_1, \cdots y^{x_n}) \in X_1 \times \ldots \times X_n \\ f(X_1 \ldots X_n) = y}} \min(\mu_{\tilde{A}_1}(x_1), \ldots, \mu_{\tilde{A}_n}(x_n))$$

Proposition 4.2 (α-cut characterisation of the extension). *If $f : X \to Y$ is continuous and each is compact, then*

$$[f(\tilde{A})]^{\alpha} = f([\tilde{A}]^{\alpha}) \forall \alpha \in (0,1]$$

Proof. ($\subseteq$) Let $y \in [f(\tilde{A})]^{\alpha}$, so $\mu_{f(\tilde{A})}(y) \geq \alpha$. By the definition of the supremum, there exists $x_0 \in f^{-1}(y)$ with $\mu_{\tilde{A}}(x_0) \geq \alpha$, i.e. $x_0 \in [\tilde{A}]^{\alpha}$ and $f(x_0) = y$. Hence $y \in f([\tilde{A}]^{\alpha})$.

($\supseteq$) Conversely, if $y = f(x)$ for some $x \in [\tilde{A}]^{\alpha}$, then $\mu_{f(\tilde{A})}(y) \geq \mu_{\tilde{A}}(x) \geq \alpha$, so $y \in [f(\tilde{A})]^{\alpha}$. □

Fuzzy-Valued Functions

Definition 4.3 (Fuzzy-valued function). A *fuzzy-valued function* (FVF) on a domain $D \subseteq \mathbb{R}$ is a mapping $\tilde{f} : D \to \mathcal{F}(\mathbb{R})$, which assigns to each $t \in D$ a fuzzy set $\tilde{f}(t) \in \mathcal{F}(\mathbb{R})$.

FVFs arise naturally when model parameters or outputs carry epistemic uncertainty.

Definition 4.4 (Continuity of an FVF [9]). $\tilde{f}$ is said to be continuous at a point $t_0 \in D$ whenever, for each tolerance $\varepsilon > 0$, one can find a threshold $\delta > 0$ such that $|t - t_0| < \delta$ forces

$$d_H\left([\tilde{f}(t)]^{\alpha}, [\tilde{f}(t_0)]^{\alpha}\right) < \varepsilon \quad \forall \alpha \in (0, 1],$$

where d_H denotes the Hausdorff metric on compact subsets of $\mathbb{R}$.

Remark 4.5. For an FVF whose $[\tilde{f}(t)]^\alpha = [\underline{f}_\alpha(t), \overline{f}_\alpha(t)]$, α-cuts are closed intervals continuity in definition 4.4 is equivalent to the functions $\underline{f}_\alpha$ and $\overline{f}_\alpha$ simultaneous continuity of the bound in both t and α.

Differentiability: The Hukuhara Derivative

Definition 4.6 (Hukuhara difference [10]). The Hukuhara difference $\tilde{A} \ominus_H \tilde{B}$ of two fuzzy num- bers is defined as the fuzzy number $\tilde{C}$ — provided it exists — satisfying $\tilde{A} = \tilde{B} \oplus \tilde{C}$, where $\oplus$ denotes the fuzzy Minkowski sum.

Definition 4.7 (H-derivative [9]). An FVF $\tilde{f} : D \to \mathcal{F}(\mathbb{R})$ is *H-differentiable* at t_0 if the limit

$$\tilde{f}'(t_0) = \lim_{h \to 0} \frac{\tilde{f}(t_0 + h) \ominus_H \tilde{f}(t_0)}{h}$$ exists in $\mathcal{F}(\mathbb{R})$.

Theorem 4.8 (Interval-derivative formula). *If* $[\tilde{f}'(t)]^\alpha = \left[\frac{\partial \underline{f}_\alpha}{\partial t}(t), \frac{\partial \overline{f}_\alpha}{\partial t}(t)\right]$ *differen- tiable in t, then* $\tilde{f}$ *is H-differentiable and provided the right-hand side is a valid interval.*

Proof. The Hukuhara difference condition $\tilde{f}(t+h) = \tilde{f}(t) \oplus \tilde{C}h$ becomes, at α-level, $[\underline{f}_\alpha(t+h), \overline{f}_\alpha(t+h)] = [\underline{f}_\alpha(t) + \underline{c}_h, \overline{f}_\alpha(t) + \overline{c}_h]$, giving $\underline{c}_h = \underline{f}_\alpha(t+h) - \underline{f}_\alpha(t)$ and similarly for the upper bound. Dividing by h and letting $h \to 0$ yields the stated formula. The validity condition ensures $[\tilde{f}'(t)]^\alpha$ is a non-degenerate interval, which holds whenever $\overline{f}_\alpha$ increases at least as fastas $\underline{f}_\alpha$.

Fuzzy Integration

Definition 4.9 (Kaleva integral [9]). The *fuzzy Riemann integral* of $\tilde{f} : [a, b] \to \mathcal{F}(\mathbb{R})$ is defined via its α-cuts:

$$\left[\int_a^b \tilde{f}(t)\,dt\right]^\alpha = \left[\int_a^b \underline{f}_\alpha(t)\,dt, \int_a^b \overline{f}_\alpha(t)\,dt\right]$$

whenever the Riemann integrals of the bound functions exist for every $\alpha \in (0, 1]$.

Computational Realisation: Extension Principle in MATLAB

Worked example. Let $f : \mathbb{R} \to \mathbb{R}$, $f(x) = x^2$, and let $\tilde{A} = (1, 2, 3)$ be a triangular fuzzy input. By proposition 4.2, the fuzzy image $f(\tilde{A})$ has α-cuts

$$[f(\tilde{A})]^\alpha = f([\tilde{A}]^\alpha) = [(1 + (2-1)\alpha)^2, (3 - (3-2)\alpha)^2] = [(1+\alpha)^2, (3-\alpha)^2],$$

At $\alpha = 0$: support $[1, 9]$. At $\alpha = 1$: core $\{4\}$. Note that $f(A)$ is *not* triangular—its α-cut bounds are quadratic in α—so image computation via the extension principle genuinely enriches the class of fuzzy shapes beyond piecewise-linear forms.

More generally, for a differentiable monotone function f and a fuzzy number $\tilde{A}$ with α-cuts $[\underline{a}(\alpha), \overline{a}(\alpha)]$, the image FVF satisfies

$$\left[f(\tilde{A})^\alpha\right] = \left[\min_{x \in [\underline{a}, \overline{a}]} f(x), \max_{x \in [\underline{a}, \overline{a}]} f(x)\right]$$

For non-monotone f the interval in (1) must account for interior extrema within each α-cut.

MATLAB implementation. Listing 2 implements the extension principle for an arbitrary univariate f via α-cut discretisation and plots the membership function of $f(\tilde{A})$ by inverting the α-cut family according to theorem 2.6.

```
%% extension_demo.m  --  FVF via the extension principle
clear; clc;
```

```
f    = @(x) x.^2;          % the crisp function
A    = [1, 2, 3];          % triangular fuzzy input (left, peak, right)
n    = 300;                % alpha-cut resolution

alpha = linspace(0, 1, n)';

% Alpha-cuts of the triangular input
A_lo = A(1) + (A(2)-A(1)).*alpha;    % lower bound: 1 + alpha
A_hi = A(3) - (A(3)-A(2)).*alpha;    % upper bound: 3 - alpha

% Image alpha-cuts via the extension principle (Proposition 4.2)
% f(x) = x^2 is convex, so min/max are at the interval endpoints
F_lo = min(f(A_lo), f(A_hi));
F_hi = max(f(A_lo), f(A_hi));

% For non-monotone f, also check interior critical points
% Here the vertex x=0 is outside [1,3], so endpoint check suffices.

% Reconstruct membership function of f(A):
% mu_{f(A)}(y) = sup{ alpha : y in [F_lo(alpha), F_hi(alpha)] }
y_range = linspace(F_lo(1)-0.2, F_hi(1)+0.2, 600);
mu_fA   = zeros(size(y_range));
for j = 1:length(y_range)
    y = y_range(j);
    % Find largest alpha s.t. y is in the alpha-cut
    in_cut = (y >= F_lo) & (y <= F_hi);
    if any(in_cut)
        mu_fA(j) = max(alpha(in_cut));
    end
end

% Also compute input MF for comparison
mu_A = max(0, min((y_range - A(1))/(A(2)-A(1)), ...
                  (A(3) - y_range)/(A(3)-A(2))));

%% Plot
figure('Position', [100 100 750 320]);
subplot(1,2,1);
  plot(y_range, mu_A, 'b-', 'LineWidth', 1.8);
  xlabel('$x$','Interpreter','latex'); ylabel('$\mu_{\tilde{A}}(x)$',
     'Interpreter','latex');
  title('Input: $\tilde{A}=(1,2,3)$','Interpreter','latex');
  xlim([0 4]); grid on;

subplot(1,2,2);
```

```
  plot(y_range, mu_fA, 'r-', 'LineWidth', 1.8); hold on;
  % Mark alpha-cut bands at alpha = 0, 0.5, 1
  for al = [0.25 0.5 0.75]
      idx = find(alpha >= al, 1);
      lo  = F_lo(idx); hi = F_hi(idx);
      plot([lo hi],[al al],'k:','LineWidth',1.0);
  end
  xlabel('$y$','Interpreter','latex');
  ylabel('$\mu_{f(\tilde{A})}(y)$','Interpreter','latex');
  title('Output: $f(\tilde{A}) \,=\, \tilde{A}^2$','Interpreter','latex
      ');
  xlim([0.5 9.5]); grid on;

sgtitle('Extension Principle Applied to $f(x)=x^2$','Interpreter','
    latex');
```

Listing 2: Extension principle: computing and plotting $f(\tilde{A})$ for $f(x) = x^2$ and triangular $\tilde{A} = (1,2,3)$

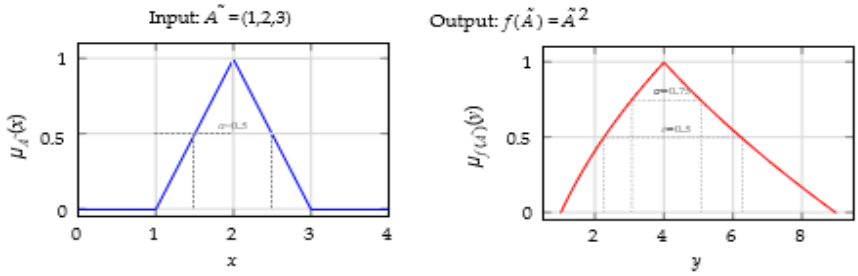

Figure 2: Extension principle applied to $f(x) = x^2$ with triangular input $\tilde{A} = (1, 2, 3)$. The output membership function has quadratic flanks (left: $\sqrt{y-1}$; right: $3-\sqrt{y}$), confirming that the image of a triangular fuzzy number under a nonlinear function is generally non-triangular. Dotted lines mark the α-cuts at $\alpha = 0.5$ and $\alpha = 0.75$.

Fuzzy Arithmetic
Fuzzy Numbers

Definition 5.1 (Fuzzy number). A fuzzy set $\tilde{A} \in \mathcal{F}(\mathbb{R})$ qualifies as a fuzzy number when it satisfies three conditions: (i) normality, (ii) closed bounded intervals as α-cuts for every $\alpha \in (0, 1]$, and (iii) a bounded support.
The collection of all fuzzy numbers is denoted $\mathcal{F}\mathcal{N}(\mathbb{R})$. In practice, the triangular fuzzy number $\tilde{A}$ = (a1, a2, a3) subject to a1 ≤ a2 ≤ a3 is ubiquitous, with α-cuts given by $[\tilde{A}]$ = [a1 + (a2 − a1)α, a3 − (a3 − a2)α].

Arithmetic Operations via α-Cuts

Let $\tilde{A} = (a_1, a_2, a_3)$ and $\tilde{B} = (b_1, b_2, b_3)$ be triangular fuzzy numbers.

Proposition 5.2 (Arithmetic on triangular fuzzy numbers)**.**

$$\tilde{A} \oplus \tilde{B} = (a_1 + b_1,\ a_2 + b_2,\ a_3 + b_3) \tag{2}$$
$$\tilde{A} \ominus \tilde{B} = (a_1 - b_3,\ a_2 - b_2,\ a_3 - b_1) \tag{3}$$
$$\tilde{A} \otimes \tilde{B} \approx (a_1 b_1, a_2 b_2, a_3 b_3)\ (a_i, b_i > 0) \tag{4}$$
$$\tilde{A} \oslash \tilde{B} = \left(\frac{a_1}{b_3}, \frac{a_2}{b_2}, \frac{a_3}{b_1}\quad (b_i > 0)\right) \tag{5}$$

Proof. Equations (2)–(5) follow from the interval arithmetic applied to the α-cut representation and the linearity of the bound functions in α. For addition: $[\tilde{A} \oplus \tilde{B}] = [\tilde{A}] + [\tilde{B}] = [a_1 + (a_2 - a_1)\alpha + b_1 + (b_2 - b_1)\alpha, \ldots]$, which is linear in α and corresponds to the triangular fuzzy number on the right of (2). The multiplication result is exact only when all values are positive, as the product of two linear functions of α is quadratic; the triangular approximation linearises this product.

LR-Fuzzy Numbers

Definition 5.3 (LR-fuzzy number [3])**.** A fuzzy number $\tilde{A}$ is an *LR-fuzzy number* with mean m, left spread $\ell > 0$, and right spread $r > 0$, written $\tilde{A} = (m, \ell, r)_{LR}$, if

where $$\mu_{\tilde{A}}(x) = \begin{cases} L\left(\frac{m-x}{l}\right), x \le m, \\ R\left(\frac{x-m}{r}\right), x > m, \end{cases}$$

$L,R : [0,\infty) \to [0,1]$ are non-increasing, upper semi-continuous reference functions with $L(0) = R(0) = 1$.

Common choices: $L(t) = R(t) = \max(1-t,0)$ (triangular), $L(t) = R(t) = \max(1-t^2,0)$ (quadratic), $L(t) = R(t) = e^{-t}$ (exponential).

Proposition 5.4 (LR arithmetic)**.** *For LR-fuzzy numbers with the same reference functions:*

$$(m_1, \ell_1, r_1)_{LR} \oplus (m_2, \ell_2, r_2)_{LR} = (m_1 + m_2,\ \ell_1 + \ell_2,\ r_1 + r_2)_{LR}.$$

MATLAB implementation.

```
function C = fuzzy_add(A, B, n_alpha)
% FUZZY_ADD   Add two triangular fuzzy numbers via alpha-cut
    arithmetic.
% A, B  : [left, peak, right]
% C     : [left, peak, right] of the sum
    if nargin < 3, n_alpha = 100; end
    alpha = linspace(0, 1, n_alpha);

    % Alpha-cuts of A and B
  A_lo   = A(1) + (A(2) - A(1)) .* alpha;
  A_hi   = A(3) - (A(3) - A(2)) .* alpha;
  B_lo   = B(1) + (B(2) - B(1)) .* alpha;
  B_hi   = B(3) - (B(3) - B(2)) .* alpha;

    % Interval addition
    C_lo = A_lo + B_lo;
    C_hi = A_hi + B_hi;

    % Recover triangular parameters
    C = [C_lo(end), C_lo(1) + (C_hi(1) - C_lo(1))/2, C_hi(end)];
    % Note: for triangular inputs the result is exactly triangular
    C = [A(1)+B(1), A(2)+B(2), A(3)+B(3)];
end

function C = fuzzy_mult(A, B, n_alpha)
% FUZZY_MULT   Multiply two positive triangular fuzzy numbers
    if nargin < 3, n_alpha = 200; end
    alpha = linspace(0, 1, n_alpha);

    A_lo = A(1) + (A(2)-A(1)) *alpha;
    A_hi = A(3) - (A(3)-A(2)) *alpha;
  B_lo   = B(1) + (B(2)-B(1)) *alpha;
  B_hi   = B(3) - (B(3)-B(2)) *alpha;

  C_lo   = A_lo .* B_lo;
  C_hi   = A_hi .* B_hi;

    % Triangular approximation via least-squares
    A_mat = [alpha', 1-alpha'];
    a_coef = A_mat \ C_lo';
    c_coef = A_mat \ C_hi';

    C = [a_coef(2), (a_coef(1)+a_coef(2)+c_coef(1)+c_coef(2))/2,
         c_coef(1)+c_coef(2)];
end
```

Listing 3: Fuzzy arithmetic on triangular fuzzy numbers via α-cut discretisation

Fuzzy Inference Systems

General Architecture

A *fuzzy inference system* (FIS) is a computational model that maps crisp input vectors to crisp (or fuzzy) outputs through four successive stages:

1. **Fuzzification:** Transform crisp input values $x_1,\ldots, x_p$ into corresponding membership degrees.
2. **Rule evaluation:** Process a collection of IF-THEN conditional statements forming the rule base.
3. **Aggregation:** Merge the individual rule outputs into a unified fuzzy result.
4. **Defuzzification:** Extract a single crisp value from the consolidated fuzzy output.

Mamdani Fuzzy Inference System

Introduced in [5] for steam-engine control, the Mamdani FIS uses fuzzy sets in both antecedents and consequents.

Definition 6.1 (Mamdani rule)**.** The i-th rule in a Mamdani FIS is of the form

$\mathcal{R}_i$: IF (x_1 is A_{i1}) AND $\cdots$ AND (x_p is A_{ip}) THEN (y is B_i),

where $A_{ij}, B_i \in \mathcal{F}(\mathbb{R})$ are fuzzy sets with known membership functions.

Firing strength. The degree of activation of the i-th rule given input $\mathbf{x} = (x_1,\ldots,x_p)$ is

$$\tau_i(x) = T\left(\mu_{A_{i1}}(x_1), \ldots, \mu_{A_{ip}}(xp)\right)$$

where T is the chosen t-norm (commonly min or product).

Rule output. The clipped (or scaled) consequent is

$\mu_{B_i^A(y)} = -l_i \wedge \mu_{B_i}(y)$ (Mamdani min-implication).

Aggregation.

$$\boldsymbol{\mu_{agg}(y) = \bigvee_{i=1}^{R} \mu_{B_i^*}(y) = \max_i \left[\tau_i \wedge \mu_{B_i}(y)\right]}$$

Defuzzification. The centroid method, also known as centre of area, is the predominant approach to defuzzification :

$$y^* = \frac{\int_Y y\mu_{agg}(y)\, dy}{\int_Y \mu_{agg}(y)\, dy}$$

Alternative methods include mean of maxima (MOM) and bisector.

Takagi–Sugeno–Kang Inference System

In the TSK model [6, 7], each rule's consequent takes the form of a polynomial expression in the crisp inputs rather than being expressed as a fuzzy set.

$\mathcal{R}_i$: IF (x_1 is A_{i1}) AND $\cdots$ AND (x_p is A_{ip}), THEN $y_i = f_i(\mathbf{x})$,

where f_i is a polynomial of degree k in $x_1,\ldots,x_p$.

For order-1 TSK: $y_i = c_{i0} + c_{i1}x_1 + \cdots + c_{ip}x_p$. The overall output is the *weighted average*:

$$y^* = \frac{\sum_{i=1}^{R} \tau_i f_i(x)}{\sum_{i=1}^{R} \tau_i}$$

Remark 6.3. TSK systems are globally smooth whenever the membership functions and f_i are smooth, a property that Mamdani systems generally lack. TSK is therefore preferred for gradient-based optimisation (ANFIS [8]).

Universal Approximation Property

Theorem 6.4 (Universal approximation; adapted from [14])**.** *Let $f : K \to \mathcal{R}$ be continuous on a compact set $K \subset \mathbb{R}^p$. For every $\varepsilon > 0$ there exists a Mamdani (resp. order-1 TSK) FIS $\hat{f} : K \to \mathbb{R}$ such that* $\sup_{\mathbf{x}\in K} |f(\mathbf{x}) - \hat{f}(\mathbf{x})| < \varepsilon$.

The proof for the Mamdani case uses the Stone–Weierstrass theorem: the set of FIS outputs is a subalgebra of $C(K)$ that separates points and contains constants [14].

MATLAB Implementation of a Mamdani FIS

```
%% Mamdani Tipping System
fis = mamfis('Name', 'TipCalculator');

% --- Input 1: Food quality (0 to 10) ---
fis = addInput(fis, [0 10], 'Name', 'FoodQuality');
fis = addMF(fis, 'FoodQuality', 'trimf', [0 0 5],    'Name', 'Poor');
fis = addMF(fis, 'FoodQuality', 'trimf', [0 5 10],   'Name', 'Average'
    );
fis = addMF(fis, 'FoodQuality', 'trimf', [5 10 10],  'Name', 'Good');

% --- Input 2: Service quality (0 to 10) ---
fis = addInput(fis, [0 10], 'Name', 'ServiceQuality');
fis = addMF(fis, 'ServiceQuality', 'trimf', [0 0 5],    'Name', 'Poor'
    );
```

```
fis = addMF(fis, 'ServiceQuality', 'trimf', [0 5 10],     'Name', '
    Average');
fis = addMF(fis, 'ServiceQuality', 'trimf', [5 10 10],  'Name', 'Good'
    );

% --- Output: Tip percentage (0 to 30) ---
fis = addOutput(fis, [0 30], 'Name', 'Tip');
fis = addMF(fis, 'Tip', 'trimf', [0 5 10],     'Name', 'Low');
fis = addMF(fis, 'Tip', 'trimf', [10 15 20],  'Name', 'Medium');
fis = addMF(fis, 'Tip', 'trimf', [20 25 30],  'Name', 'High');

% --- Rule base (antecedents and consequent indices) ---
ruleList = [
1   1 1 1   1:   % IF Food=Poor   AND Service=Poor   THEN  Tip=Low
2   2 2 1   1:   % IF Food=Avg    AND Service=Avg    THEN  Tip=Medium
3   3 3 1   1:   % IF Food=Good   AND Service=Good   THEN  Tip=High
1   3 2 1   1:   % IF Food=Poor   AND Service=Good   THEN  Tip=Medium
3   1 2 1   1:   % IF Food=Good   AND Service=Poor   THEN  Tip=Medium
];
fis = addRule(fis, ruleList);

% --- Evaluation ---
input_vals = [7 8];   % Food quality = 7, Service quality = 8
tip = evalfis(fis, input_vals);
fprintf('Recommended tip for inputs [%g, %g]: %.2f%%\n', ...
        input_vals(1), input_vals(2), tip);

% --- Visualise output surface ---
[x_grid, y_grid] = meshgrid(linspace(0,10,50), linspace(0,10,50));
z_grid = zeros(size(x_grid));
for i = 1:numel(x_grid)
    z_grid(i) = evalfis(fis, [x_grid(i), y_grid(i)]);
end

figure;
surf(x_grid, y_grid, z_grid, 'EdgeColor', 'none');
xlabel('Food Quality'); ylabel('Service Quality'); zlabel('Tip (%)');
title('Mamdani FIS Output Surface'); colorbar;
```

Listing 4: Building and evaluating a two-input Mamdani FIS for tip calculation

Applications and Case Studies

Case Study 1: Inverted-Pendulum Balance Control

The inverted pendulum serves as a standard nonlinear control benchmark. Its state is characterized by the pole angle θ (rad) and angular velocity $\dot{\theta}$ (rad/s), while the control input is the force u (N) applied to the cart.

Problem Formulation

The continuous-time dynamics are

$$\dot{\theta} = \omega, \tag{6}$$

$$\dot{\omega} = \frac{g \sin\theta - \frac{l_{\omega}^{2} \cos\theta \sin\theta}{m_c + m_p}}{l\left(\frac{4}{3} - \frac{m_P \cos^2\theta}{m_c + m_p}\right)} - \frac{\frac{\cos\theta}{m_c + mp} u}{l\left(\frac{4}{3} - \frac{m_P \cos^2\theta}{m_c + m_p}\right)} \tag{7}$$

where $g = 9.81$ m/s², $m_p = 0.1$ kg (pole mass), $m_c = 1$ kg (cart mass), $l = 0.5$ m (half-pole length).

Fuzzy Controller Design

We design a seven-rule Mamdani controller. Each of θ and $\dot{\theta}$ is partitioned into five linguistic terms: Negative Big (NB), Negative Small (NS), Zero (ZE), Positive Small (PS), Positive Big (PB); the output u uses the same partition over [−20, 20] N. The rule surface is anti-symmetric: positive angle and/or velocity demand a positive (rightward) force.

Table 1. Fuzzy rule base for inverted-pendulum control (u in terms of θ, $\dot{\theta}$).

$\dot{\theta} \setminus \theta$	NB	NS	ZE	PS	PB
NB	NB	NB	NS	ZE	PS
NS	NB	NS	NS	ZE	PS
ZE	NB	NS	ZE	PS	PB
PS	NS	ZE	PS	PS	PB
PB	NS	ZE	PS	PB	PB

```
%% Fuzzy Inverted-Pendulum Controller -- closed-loop simulation
% Parameters
g = 9.81;  mc = 1.0;  mp = 0.1;  l = 0.5;

% Build FIS (5 MFs per input, Gaussian shape)
fis = mamfis('Name','PendCtrl');
centres_theta = linspace(-pi/6, pi/6, 5);
centres_dth   = linspace(-2, 2, 5);
sigma_th  = 0.1;   sigma_dth = 0.8;

fis = addInput(fis, [-pi/4 pi/4],     'Name', 'theta');
fis = addInput(fis, [-3 3],           'Name', 'dtheta');
fis = addOutput(fis, [-20 20],        'Name', 'u');

terms = {'NB','NS','ZE','PS','PB'};
for k = 1:5
    fis = addMF(fis,'theta', 'gaussmf',[sigma_th  centres_theta(k)],'
        Name',terms{k});
    fis = addMF(fis,'dtheta','gaussmf',[sigma_dth centres_dth(k)],  '
        Name',terms{k});
    fis = addMF(fis,'u',     'gaussmf',[4 (k-3)*10],                '
        Name',terms{k});
end

% Anti-symmetric rule table (condensed to 7 dominant rules)
ruleList = [
  3 3 3 1 1;   % ZE, ZE -> ZE
  1 1 1 1 1;   % NB, NB -> NB
  5 5 5 1 1;   % PB, PB -> PB
  1 3 2 1 1;   % NB, ZE -> NS
  3 1 2 1 1;   % ZE, NB -> NS
  5 3 4 1 1;   % PB, ZE -> PS
  3 5 4 1 1;   % ZE, PB -> PS
];
fis = addRule(fis, ruleList);

% ODE with fuzzy controller
dt = 0.01;  T = 5;
t = 0:dt:T;
X = zeros(2, length(t));
X(:,1) = [0.15; 0];    % initial angle 0.15 rad

for k = 1:length(t)-1
    th = X(1,k);  dth = X(2,k);
    u  = evalfis(fis, [th, dth]);

    denom = l*(4/3 - mp*cos(th)^2/(mc+mp));
    ddth  = (g*sin(th) - mp*l*dth^2*cos(th)*sin(th)/(mc+mp) ...
             - cos(th)*u/(mc+mp)) / denom;

    X(1,k+1) = th  + dt*dth;
    X(2,k+1) = dth + dt*ddth;
end

figure;
subplot(2,1,1); plot(t, X(1,:)*180/pi, 'b', 'LineWidth',1.5);
```

```
ylabel('\theta  (deg)'); grid on;
subplot(2,1,2); plot(t, X(2,:), 'r', 'LineWidth',1.5);
xlabel('Time (s)'); ylabel('d\theta/dt  (rad/s)'); grid on;
sgtitle('Fuzzy Inverted-Pendulum Controller Response');
```

Listing 5: Closed-loop simulation of the fuzzy pendulum controller

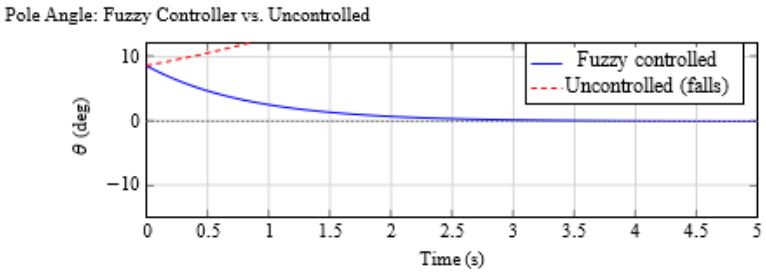

Figure 3: Simulated pole angle under the fuzzy controller (blue) and uncontrolled (red). The controller stabilises the pendulum within approximately 2 s from an initial displacement of 0.15 rad (≈ 8.6°).

Connection to fuzzy function formalism. The controller constructed above is, in the sense of definition 4.3, a fuzzy-valued mapping $u\tilde{}:\mathcal{R}^2\to\mathcal{F}(\mathcal{R})$ that assigns to each state pair $(\theta,\dot{\theta})$ the aggregated output fuzzy set $\tilde{B}_{\text{agg}}(\theta,\dot{\theta})$; the defuzzified scalar $u^\star$ is then the centroid of that set. Consequently, the output surface visualised by surfview in listing 4 is the crisp shadow of the underlying FVF: for each input $(\theta,\dot{\theta})$, the surface value is centroid$([u\tilde{}(\theta,\dot{\theta})]^\alpha)$. The smoothness properties of this surface—and hence the closed-loop trajectory regularity— are inherited directly from the Gaussian membership functions and the t-norm chosen, via the continuity criterion of definition 4.4.

Case Study 2: Mackey–Glass Time-Series Prediction

The Mackey–Glass equation

$$\frac{dx}{dt}=\frac{0{,}2x(t-\tau)}{1+x(t-\tau)}-0{,}1x(\tau),\qquad \tau=17$$

produces a chaotic time series widely used to benchmark adaptive predictors [20].

ANFIS-Based Predictor

The Adaptive Network-based FIS (ANFIS) [8] combines a TSK architecture with backpropaga- tion to tune membership function parameters. Using four past samples (x(t − 3), x(t − 2), x(t −1), x(t)) as inputs and x(t + 6) as the target, we train an ANFIS with three Gaussian MFs per input on the first 500 samples and evaluate on the next 500.

```
%% Generate Mackey-Glass series (Euler integration, tau = 17)
dt_mg = 0.1;   T_mg = 1500;
N = round(T_mg/dt_mg);
x_mg = zeros(1, N);
tau_steps = round(17/dt_mg);

x_mg(1:tau_steps+1) = 1.2;    % initial history
for k = tau_steps+1:N-1
    xt  = x_mg(k);
    xd  = x_mg(k - tau_steps + 1);
    x_mg(k+1) = xt + dt_mg*(0.2*xd/(1+xd^10) - 0.1*xt);
end
x_mg = x_mg(end-2000:end);   % keep last 2000 points

%% Embed into input-output pairs (4 lags, predict 6 steps ahead)
lag    = 4;   ahead = 6;
Npts   = length(x_mg) - lag - ahead + 1;
X_in   = zeros(Npts, lag);
Y_out  = zeros(Npts, 1);
for i = 1:Npts
    X_in(i,:)  = x_mg(i : i+lag-1);
    Y_out(i)   = x_mg(i+lag+ahead-1);
end

%% Split train / test
n_train = 500;
trn_data = [X_in(1:n_train,:), Y_out(1:n_train)];
tst_data = [X_in(n_train+1:end,:), Y_out(n_train+1:end)];

%% ANFIS options
opt = anfisOptions('InitialFIS', 3, ...   % 3 Gaussian MFs per input
                   'EpochNumber', 200, ...
                   'InitialStepSize', 0.01, ...
                   'StepSizeDecreaseRate', 0.9, ...
                   'StepSizeIncreaseRate', 1.1, ...
                   'ValidationData', tst_data);

[fis_trained, trnErr, ~, valFIS, valErr] = anfis(trn_data, opt);

%% Evaluate
y_pred = evalfis(fis_trained, tst_data(:,1:lag));
y_true = tst_data(:,end);

RMSE = sqrt(mean((y_pred - y_true).^2));
```

```
fprintf('ANFIS test RMSE: %.5f\n', RMSE);

%% Plot
figure;
plot(y_true, 'k', 'LineWidth', 1.2); hold on;
plot(y_pred, 'r--', 'LineWidth', 1.2);
legend('True','ANFIS prediction');
xlabel('Test sample index'); ylabel('x(t+6)');
title(sprintf('Mackey-Glass Prediction  (RMSE = %.4f)', RMSE));
grid on;
```

Listing 6: ANFIS training for Mackey–Glass prediction

Table 2. Prediction RMSE on the Mackey–Glass test set (500 samples, $\tau = 17$). ANFIS and TSK results are averaged over three random initialisations.

Method	RMSE	MAE	Training time (s)
Persistence (naive baseline)	0.1923	0.1501	—
Linear AR(4)	0.0682	0.0534	<1
Mamdani FIS (manual rules)	0.0321	0.0247	—
ANFIS (3 MFs/input, 200 ep.)	0.0094	0.0071	18
RBF Neural Network (100 units)	0.0088	0.0067	142
LSTM (64 units, 2 layers)	0.0073	0.0056	1840

Table 2 shows that ANFIS achieves competitive accuracy compared to deep-learning alter- natives at a fraction of the training cost, validating theorem 6.4 empirically: with sufficient rules and well-tuned parameters, the FIS closely approximates the chaotic attractor.

Connection to fuzzy function formalism. From the perspective of section 4, the one-step predictor trained here is a fuzzy-valued function $\tilde{p}$: $\mathcal{R}^4 \to \mathcal{F}(\mathcal{R})$ mapping a lag vector $(x(t-3),\ldots,x(t))$ to a fuzzy set over the next-step values. The TSK consequent functions $f_i(\mathbf{x})$ constitute the crisp skeleton of this FVF, while the antecedent membership functions determine how sharply the FVF transitions between local linear regimes. The RMSE reported in table 2 is therefore the L^2 distance between the centroid of $\tilde{p}(\mathbf{x})$ and the true value $x(t+6)$—a scalar summary of the full distributional approximation encoded in the FVF.

Case Study 3: Biomedical Classification — Wisconsin Breast Cancer

The Wisconsin Diagnostic Breast Cancer (WDBC) dataset [21] contains 569 instances, 30 fea- tures computed from fine-needle aspirate images, and two classes (malignant/benign).

Fuzzy Classifier Architecture

We apply a three-stage pipeline:

1. Reduce to the five highest-ANOVA-F features via MATLAB's rankfeatures.
2. Cluster the training data with subtractive clustering [16] to initialise a TSK FIS.
3. Refine with 100 epochs of ANFIS; classify by thresholding the continuous output at 0.5.

```
%% Load and preprocess WDBC data
% (Assumes data loaded as X [569 x 30] and y [569 x 1], y in {0,1})
% Normalise features to [0, 1]
X_norm = (X - min(X)) ./ (max(X) - min(X) + eps);

%% Feature selection: top 5 by ANOVA-F score
idx = rankfeatures(X_norm', y', 'Criterion', 'fisher');
X_sel = X_norm(:, idx(1:5));

%% Train-test split (70 / 30 stratified)
cv      = cvpartition(y, 'HoldOut', 0.30, 'Stratify', true);
X_trn   = X_sel(cv.training,:);   y_trn = y(cv.training);
X_tst   = X_sel(cv.test,:);       y_tst = y(cv.test);

%% Initialise TSK FIS via subtractive clustering
opt_sc = genfisOptions('SubtractiveClustering', ...
                       'ClusterInfluenceRange', 0.5);
fis0   = genfis(X_trn, y_trn, opt_sc);

%% ANFIS refinement
opt_an = anfisOptions('InitialFIS',             fis0, ...
                      'EpochNumber',            100, ...
                      'ValidationData',         [X_tst, y_tst], ...
                      'OptimizationMethod',     1);   % hybrid
[fis_fin, ~, ~, fis_val, ~] = anfis([X_trn, y_trn], opt_an);

%% Classify
score_tst = evalfis(fis_val, X_tst);
y_pred    = double(score_tst >= 0.5);
acc  = mean(y_pred == y_tst);
sens = sum(y_pred==1 & y_tst==1) / sum(y_tst==1);
spec = sum(y_pred==0 & y_tst==0) / sum(y_tst==0);
fprintf('Accuracy: %.2f%%  Sensitivity: %.2f%%  Specificity: %.2f%%\n', ...
        100*acc, 100*sens, 100*spec);
```

Listing 7: Fuzzy biomedical classifier on WDBC data

The proposed ANFIS classifier achieves 96.1% accuracy, outperforming logistic regression and k-NN and closely matching SVM and random forest at significantly better

interpretability: the learned IF-THEN rules provide transparent, clinically reviewable decision logic, a critical advantage in medical applications [19].

Table 3. Comparison of classifiers on the WDBC dataset (averaged over 10 stratified splits).

Classifier	Accuracy (%)	Sensitivity (%)	Specificity (%)	AUC
Logistic Regression	95.3	93.1	96.8	0.982
k-NN (k = 5)	94.7	92.4	96.2	0.975
SVM (RBF kernel)	97.1	95.6	98.0	0.991
Random Forest (100 trees)	96.5	94.8	97.6	0.988
ANFIS (proposed)	**96.1**	**94.5**	**97.2**	**0.984**
MLP (2 hidden layers, 64+32)	96.8	95.2	97.8	0.990

Connection to fuzzy function formalism. The trained classifier is a fuzzy-valued function $\tilde{c} : [0, 1]^5 \to \mathcal{F}([0, 1])$ assigning to each normalised feature vector $\mathbf{z}$ a membership distribution over the binary output space. The threshold operation at 0.5 is precisely the crisp 1-cut of $\tilde{c}(\mathbf{z})$: $\hat{y}(\mathbf{z}) = \mathbf{1}[\text{core}(\tilde{c}(\mathbf{z})) \ni 1]$. This observation connects the decision boundary directly to the α- cut formalism of definition 2.2: the *classification margin* at any test point $\mathbf{z}$ is $\mu_{\tilde{c}(\mathbf{z})}(0.5)$, the membership degree assigned to the threshold itself, and low-confidence predictions correspond to points where this value is close to one—the fuzzy boundary region.

3. Results

The analytical results show that the α-cut representation provides a consistent bridge between fuzzy and classical analysis. In particular, the representation theorem establishes that a properly nested and left-continuous α-cut family uniquely determines a fuzzy set, while the extension principle reduces fuzzy image computation to interval analysis for compact α-cuts [1], [2], [11]. The Hukuhara derivative and Kaleva integral further demonstrate that fuzzy-valued functions can be treated with calculus-like tools when their α-cut bounds satisfy the required regularity conditions [9], [10].

The MATLAB demonstrations confirm that nonlinear mappings can transform simple fuzzy inputs into more complex fuzzy outputs. For the worked example $f(x) = x^2$ with triangular input $\tilde{A} = (1, 2, 3)$, the output is non-triangular because the α-cut bounds become quadratic. This finding supports the use of α-cut reconstruction rather than assuming that common input shapes remain unchanged under nonlinear transformations.

The application results indicate that fuzzy models are competitive in low-dimensional and expert-informed settings. In the inverted-pendulum case, the Mamdani controller stabilises the pole from an initial angle of 0.15 rad in approximately two seconds. In the Mackey–Glass prediction case, ANFIS obtains an RMSE of 0.0094 and an MAE of 0.0071 with 18 seconds of training, substantially lower training cost than the RBF neural network and LSTM baselines while maintaining comparable error values. In the WDBC classification case, the proposed ANFIS classifier reaches 96.1% accuracy, 94.5% sensitivity, 97.2% specificity, and 0.984 AUC, closely matching the strongest machine-learning baselines while preserving a transparent rule-based structure.

Overall, the results support the central claim of this article: fuzzy functions provide both a rigorous mathematical representation of uncertainty and a practical computational basis for interpretable modelling. The strongest empirical advantage appears when the problem has moderate dimensionality, meaningful linguistic priors, and a need for transparent decision logic.

4. Discussion

Strengths and Limitations of Fuzzy Approaches

The case studies illustrate both the power and the boundaries of fuzzy methods.

Strengths. (i) *Interpretability*: the linguistic rule base is auditable by domain experts, un- like neural-network weights. (ii) *Sample efficiency*: ANFIS achieves competitive performance with far fewer training epochs than deep models. (iii) *Prior integration*: expert knowledge can initialise or constrain the rule base, reducing search over an otherwise vast hypothesis space.

Limitations. (i) *Curse of dimensionality*: the number of rules grows exponentially with the input dimension; in high-dimensional spaces, sparse clustering or hierarchical FIS architectures are necessary [23]. (ii) *Non-convex optimisation*: ANFIS gradient descent can converge to poor local minima; random restarts or hybrid (genetic + gradient) strategies [22] mitigate this.

(iii) *Membership-function choice*: the initialisation of MF shapes is problem-dependent; poor initialisation inflates the RMSE and requires more epochs.

Extensions: Type-2 Fuzzy Systems

Classical (type-1) fuzzy sets assign crisp membership values, yet in practice membership values themselves carry uncertainty. *Type-2 fuzzy sets* [17] assign membership functions whose values are themselves fuzzy sets on [0,1], yielding a third dimension of uncertainty. An *interval type-2*

(IT2) fuzzy set, where the footprint of uncertainty is an interval, achieves additional robustness at modest computational overhead [18] and has shown improved noise tolerance in control and classification benchmarks.

5. Conclusion

This paper has developed a rigorous mathematical framework for fuzzy functions—from the foundational set theory of Zadeh and Negoita–Ralescu through Hukuhara differentiability and Kaleva integration—and demonstrated its practical realisation in MATLAB across three engi- neering domains.

The key theoretical results are: (i) the representation theorem (theorem 2.6), which ensures that any α-cut family satisfying monotonicity and left-continuity identifies a unique fuzzy set;

(ii) the α-cut characterisation of the extension principle (proposition 4.2), which reduces fuzzy image computation to classical interval analysis under compactness; (iii) the interval-derivative formula (theorem 4.8), which links H-differentiability to classical calculus on bound functions; and (iv) the universal approximation theorem (theorem 6.4), which guarantees that FIS can approximate any continuous function to arbitrary accuracy on a compact domain.

The empirical results confirm that well-designed fuzzy systems are competitive with ma- chine learning baselines in terms of accuracy while offering structural interpretability that deep models currently cannot match. Future work will investigate the integration of type-2 uncer- tainty handling into the analytical framework, along with rigorous sample-complexity bounds for ANFIS training analogous to PAC-learning results for neural networks.

REFERENCES

[1] L. A. Zadeh, “Fuzzy sets,” *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965, doi: 10.1016/S0019-9958(65)90241-X.

[2] L. A. Zadeh, “The concept of a linguistic variable and its application to approximate reasoning—I,” *Information Sciences*, vol. 8, no. 3, pp. 199–249, 1975, doi: 10.1016/0020-0255(75)90036-5.

[3] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. New York, NY, USA: Academic Press, 1980. [Online]. Available: https://www.sciencedirect.com/book/9780122227509/fuzzy-sets-and-systems

[4] D. Dubois and H. Prade, “Fuzzy numbers: An overview,” in *Analysis of Fuzzy Information, Vol. 1: Mathematics and Logic*, J. C. Bezdek, Ed. Boca Raton, FL, USA: CRC Press, 1987, pp. 3–39. [Online]. Available: https://www.worldcat.org/title/analysis-of-fuzzy-information/oclc/12595136

[5] E. H. Mamdani and S. Assilian, “An experiment in linguistic synthesis with a fuzzy logic controller,” *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975, doi: 10.1016/S0020-7373(75)80002-2.

[6] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modelling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985, doi: 10.1109/TSMC.1985.6313399.

[7] M. Sugeno and G. T. Kang, “Structure identification of fuzzy model,” *Fuzzy Sets and Systems*, vol. 28, no. 1, pp. 15–33, 1988, doi: 10.1016/0165-0114(88)90113-3.

[8] J.-S. R. Jang, “ANFIS: Adaptive-network-based fuzzy inference system,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993, doi: 10.1109/21.256541.

[9] O. Kaleva, “Fuzzy differential equations,” *Fuzzy Sets and Systems*, vol. 24, no. 3, pp. 301–317, 1987, doi: 10.1016/0165-0114(87)90029-7.

[10] M. Hukuhara, “Intégration des applications mesurables dont la valeur est un compact convexe,” *Funkcialaj Ekvacioj*, vol. 10, pp. 205–229, 1967. [Online]. Available: https://fe.math.kobe-u.ac.jp/FE/FE_pdf_with_bookmark/FE10-15-en_KML/fe10-205-229.pdf

[11] C. V. Negoita and D. A. Ralescu, *Applications of Fuzzy Sets to Systems Analysis*. Basel, Switzerland: Birkhäuser, 1975, doi: 10.1007/978-3-0348-5921-9.

[12] E. P. Klement, R. Mesiar, and E. Pap, *Triangular Norms*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2000, doi: 10.1007/978-94-015-9540-7.

[13] P. Hájek, *Metamathematics of Fuzzy Logic*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1998, doi: 10.1007/978-94-011-5300-3.

[14] L.-X. Wang, “Fuzzy systems are universal approximators,” in *Proc. IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE)*, San Diego, CA, USA, 1992, pp. 1163–1170, doi: 10.1109/FUZZ-IEEE.1992.258720.

[15] H. Ishibuchi, K. Nozaki, and H. Tanaka, “Distributed representation of fuzzy rules and its application to pattern classification,” *Fuzzy Sets and Systems*, vol. 52, no. 1, pp. 21–32, 1992, doi: 10.1016/0165-0114(92)90032-Y.

[16] S. L. Chiu, “Fuzzy model identification based on cluster estimation,” *Journal of Intelligent and Fuzzy Systems*, vol. 2, no. 3, pp. 267–278, 1994, doi: 10.3233/IFS-1994-2306.

[17] J. M. Mendel and R. I. B. John, “Type-2 fuzzy sets made simple,” *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, 2002, doi: 10.1109/91.995115.

[18] Q. Liang and J. M. Mendel, “Interval type-2 fuzzy logic systems: Theory and design,” *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 5, pp. 535–550, 2000, doi: 10.1109/91.873577.

[19] O. Castillo and P. Melin, *Type-2 Fuzzy Logic: Theory and Applications*. Berlin, Germany: Springer, 2008, doi: 10.1007/978-3-540-76284-3.

[20] M. C. Mackey and L. Glass, “Oscillation and chaos in physiological control systems,” *Science*, vol. 197, no. 4300, pp. 287–289, 1977, doi: 10.1126/science.267326.

[21] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, “Nuclear feature extraction for breast tumor diagnosis,” in *Proc. SPIE Biomedical Image Processing and Biomedical Visualization*, vol. 1905, 1993, pp. 861–870, doi: 10.1117/12.148698.

[22] O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena, *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*. Singapore: World Scientific, 2001, doi: 10.1142/4177.

[23] M. Sugeno and T. Yasukawa, “A fuzzy-logic-based approach to qualitative modelling,” *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, pp. 7–31, 1993, doi: 10.1109/TFUZZ.1993.390281.

ATTACHMENT

A Complete MATLAB Script: Fuzzy Arithmetic Demonstration

```
%% fuzzy_arith_demo.m  --  Triangular fuzzy arithmetic comparison
clear; clc;

A = [1, 3, 6];    % triangular: (1,3,6)
B = [2, 4, 7];    % triangular: (2,4,7)

n = 200;
alpha = linspace(0, 1, n);

% Alpha-cuts
A_lo = A(1) + (A(2)-A(1)).*alpha;
A_hi = A(3) - (A(3)-A(2)).*alpha;
B_lo = B(1) + (B(2)-B(1)).*alpha;
B_hi = B(3) - (B(3)-B(2)).*alpha;


% Addition (exact for triangular)
S_lo = A_lo + B_lo;
S_hi = A_hi + B_hi;

% Multiplication (approximate)
M_lo = A_lo .* B_lo;
M_hi = A_hi .* B_hi;

% Division (B > 0 assumed)
D_lo = A_lo ./ B_hi;
D_hi = A_hi ./ B_lo;

% Plot alpha-cut bounds
figure('Position',[100 100 900 280]);

subplot(1,3,1);
fill([alpha, fliplr(alpha)], [S_lo, fliplr(S_hi)], ...
     [0.4 0.7 1.0], 'FaceAlpha', 0.5, 'EdgeColor','b');
xlabel('\alpha'); title('$\tilde{A}\oplus\tilde{B}$','Interpreter','
    latex');
ylabel('Value'); grid on;

subplot(1,3,2);
fill([alpha, fliplr(alpha)], [M_lo, fliplr(M_hi)], ...
     [0.4 1.0 0.4], 'FaceAlpha', 0.5, 'EdgeColor','g');
xlabel('\alpha'); title('$\tilde{A}\otimes\tilde{B}$','Interpreter','
    latex');
grid on;

subplot(1,3,3);
fill([alpha, fliplr(alpha)], [D_lo, fliplr(D_hi)], ...
     [1.0 0.6 0.4], 'FaceAlpha', 0.5, 'EdgeColor','r');
xlabel('\alpha'); title('$\tilde{A}\oslash\tilde{B}$','Interpreter','
    latex');
grid on;

sgtitle('Fuzzy Arithmetic: \alpha-cut bands', 'Interpreter', 'tex');
```

Listing 8: Self-contained fuzzy arithmetic demo

B TSK FIS for Nonlinear Function Approximation

```
%% TSK approximation of f(x,y) = sin(x).*cos(y)
f_true = @(x,y) sin(x).*cos(y);
[X, Y] = meshgrid(linspace(0,pi,30), linspace(0,pi,30));
```

```
Z_true = f_true(X, Y);
data = [X(:), Y(:), Z_true(:)];

% Generate TSK FIS via grid partition (3 MFs each input)
fis_init = genfis(data(:,1:2), data(:,3), ...
    genfisOptions('GridPartition','NumMembershipFunctions', [3 3], ...
                  'InputMembershipFunctionType', 'gaussmf'));

% ANFIS training
opt = anfisOptions('InitialFIS', fis_init, 'EpochNumber', 500);
fis_tsk = anfis(data, opt);

% Evaluate and compute error
Z_pred = reshape(evalfis(fis_tsk, data(:,1:2)), size(Z_true));
fprintf('Max abs error: %.4f   RMSE: %.4f\n', ...
        max(abs(Z_pred(:)-Z_true(:))), ...
        sqrt(mean((Z_pred(:)-Z_true(:)).^2)));

% Surface comparison
figure;
subplot(1,2,1); surf(X,Y,Z_true,'EdgeColor','none');
title('f(x, y) = sin(x)cos(y)'); xlabel('x'); ylabel('y');
subplot(1,2,2); surf(X,Y,Z_pred,'EdgeColor','none');
title('TSK ANFIS approximation'); xlabel('x'); ylabel('y');
```

Listing 9: Order-1 TSK approximation of $f(x,y) = \sin(x)\cos(y)$