Fuzzy models

We can use fuzzy logic to build fuzzy models. In this chapter, we examine how this works.

1 Types of fuzzy models

A static/dynamic systems which makes use of fuzzy sets is called a **fuzzy system**. Most common are fuzzy systems defined by if-then rules. These are called **rule-based systems**, also known as **fuzzy models**. An if-then rule generally takes the form of

$$\textbf{If } \textit{antecedent proposition} \textbf{ then } \textit{consequent proposition}. \tag{1.1}$$

The antecedent proposition is always a fuzzy proposition of the type '$\mathbf{x}$ is A', where $\mathbf{x}$ is a **linguistic variable** and A is a **linguistic constant**. (For example, it can be 'if Temperature is high then ....) The structure of the consequent proposition, however, depends on the model we use.

- In a **linguistic fuzzy model**, both the antecedent and the consequent are fuzzy propositions.
- The **fuzzy relational model** is an extension of the linguistic fuzzy model. Now, a fuzzy antecedent can be coupled to multiple fuzzy propositions at the same time.
- In the **Takagi-Sugeno (TS) fuzzy model**, the consequent is a crisp function of the antecedent variables.

2 The linguistic fuzzy model

2.1 Properties of the linguistic model

As we just saw, in a linguistic fuzzy model, relations take the form of

$$\mathcal{R}_i : \textbf{if } \mathbf{x} \text{ is } A_i \textbf{ then } \mathbf{y} \text{ is } B_i. \tag{2.1}$$

A **linguistic variable** L (for example 'Temperature') is defined as a set $L = (\mathbf{x}, \mathcal{A}, X, g, m)$. Here, $\mathbf{x}$ is the **base variable**, having the same name as the linguistic variable. $\mathcal{A}$ is the **set of linguistic terms** (for example 'cold', 'normal' and 'warm'). X is the **domain** of x (for example, $[-273, \infty)$). Finally, g is a **syntactic rule** for generating linguistic terms and m is a **semantic rule** that assigns to each linguistic term its meaning. The latter two are in a way just formalities: we won't consider them here.

It is often required that a linguistic term satisfies properties of coverage and semantic soundness. **Coverage** means that each domain element $\mathbf{x} \in X$ is assigned to at least one fuzzy set A_i. (For example, there isn't a single temperature which is not either 'cold', 'normal' or 'warm'.) A stronger requirement is ϵ-coverage. This demands that each domain element $\mathbf{x} \in X$ is at least assigned to one fuzzy set A_i with $\mu_{A_i}(\mathbf{x}) > \epsilon$. Next to this, **semantic soundness** relates to how well a system can distinguish between different variables $\mathbf{x}$. (For example, if a system can't find the difference between a low temperature of 0° C and a low temperature of 5° C, then it is doesn't have a lot of semantic soundness.)

2.2 Inference in the linguistic model

Inference in fuzzy rule-based systems is the process of deriving a fuzzy output set given the rules and the inputs. Each rule $\mathcal{R}_i$ can be seen as a fuzzy relation $R : (X \times Y) \to [0, 1]$ such that

$$\mu_r(\mathbf{x}, \mathbf{y}) = I(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})). \tag{2.2}$$

The I operator can be either a fuzzy implication or a conjuction operator (t-norm). Fuzzy implication is used when the rule has the form 'A implies B'. Examples of fuzzy implications are the **Lukasiewicz implication** and the **Kleene-Diene implication**, respectively defined as

$$I(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})) = \min(1, 1-\mu_A(\mathbf{x})+\mu_B(\mathbf{y})) \qquad \text{and} \qquad I(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})) = \max(1-\mu_A(\mathbf{x}), \mu_B(\mathbf{y})). \tag{2.3}$$

Alternatively, conjunction is used is when $A \wedge B$. That is, when A and B simultaneously hold. Examples of t-norms are the minimum (also often referred to as the **Mamdani 'implication'** and the **Larsen 'implication'**, respectively defined as

$$I(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})) = \min(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})) \qquad \text{and} \qquad I(\mu_A(\mathbf{x}), \mu_B(\mathbf{y})) = \mu_A(\mathbf{x}) \cdot \mu_B(\mathbf{y}). \tag{2.4}$$

So how do we use this? Well, let's suppose we have a rule **if** $\mathbf{x}$ is A_i **then** $\mathbf{y}$ is B_i and we also know that $\mathbf{x}$ is A', then we can find the set B' satisfying $\mathbf{y}$ is B' using

$$B' = A' \circ R. \tag{2.5}$$

The question remains, what do we do if we have multiple rules/relations R_i? In that case we have to join them somehow to some joined relation R. When dealing with implications, we do this using an intersection, like

$$R = \bigcap_{i=1}^{K} R_i \qquad \text{meaning that} \qquad \mu_R(\mathbf{x}, \mathbf{y}) = \min_{1 \le i \le K} \mu_{R_i}(\mathbf{x}, \mathbf{y}). \tag{2.6}$$

If, however, we are dealing with conjunction, then the aggregated relation R is the union of the individual relations R_i. So,

$$R = \bigcup_{i=1}^{K} R_i \qquad \text{meaning that} \qquad \mu_R(\mathbf{x}, \mathbf{y}) = \max_{1 \le i \le K} \mu_{R_i}(\mathbf{x}, \mathbf{y}). \tag{2.7}$$

Again, the output set B' is found in the same way, by using $B' = A' \circ R$.

2.3 Max-min Mamdani inference

In the previous method of inference, we had to use relations R. When the domains X and Y get very big, this will become rather complicated. But luckily, the relational calculus can be bypassed by using **max-min (Mamdani) inference**. In Mamdany inference, we can find the output using

$$\mu_{B'}(\mathbf{y}) = \max_{1 \le i \le K} \left(\max_X \left(\mu_{A'}(\mathbf{x}) \wedge \mu_{A_i}(\mathbf{x}) \right) \wedge \mu_{B_i}(\mathbf{y}) \right) = \max_{1 \le i \le K} \left(\beta_i \wedge \mu_{B_i}(\mathbf{y}) \right). \tag{2.8}$$

In this equation, we have defined the **degree of fulfillment** β_i as

$$\beta_i = \max_X \left(\mu_{A'}(\mathbf{x}) \wedge \mu_{A_i}(\mathbf{x}) \right). \tag{2.9}$$

Basically, this number is an indication of how much A' and A_i are alike. A big advantage of the Mamdani method is that it does not require discretization of the domain. It can thus work with analytically defined membership functions.

2.4 Defuzzification

We now know how to find an output fuzzy set B', based on fuzzy rules. But usually, we don't want to know that some parameter $\mathbf{y}$ belongs to a fuzzy set B'. Instead, we want to know a value $\mathbf{y}'$. The process of finding a value $\mathbf{y}'$ from the knowledge that $\mathbf{y}$ is B' is called **defuzzification**.

There are two commonly used defuzzification methods: the center of gravity method and the mean of maxima method. In the **center of gravity (COG) method**, we calculate the **y**-coordinate of the center of gravity of the fuzzy set B'. This is done according to

$$\mathbf{y}' = \text{cog}(B') = \frac{\sum_{j=1}^{F} \mu_{B'}(y_j) y_j}{\sum_{j=1}^{F} \mu_{B'}(y_j)} = \frac{\int_Y \mu_{B'}(y) y \, dy}{\int_Y \mu_{B'}(y)}. \tag{2.10}$$

The first part of the above equation is used for discretized domains Y, whereas the second part is used for continuous domains Y.

In the **mean of maxima (MOM) method**, we find all points where $\mu_{B'}(\mathbf{y})$ is at its maximum. We then take the mean of all these points. In mathematical notation, we then have

$$\mathbf{y}' = \text{mom}(B') = \text{cog}\left(\left\{\mathbf{y} | \mu_{B'}(\mathbf{y}) = \max_{\mathbf{y} \in Y} \mu_{B'}(\mathbf{y})\right\}\right). \tag{2.11}$$

In a way, the MOM method selects the 'most probable' output. It is often used with inference based on fuzzy implications. On the other hand, the COG method is usually used together with Mamdani inference.

Finally, there is also a third defuzzification, called **fuzzy-mean defuzzification**. It is often used after Mamdani inference, to avoid the integration step from the COG method. When applying this method, first the **consequent fuzzy sets** B_j are found, using

$$\mu_{B_j}(\mathbf{y}) = (\beta_i \wedge \mu_{B_i}(\mathbf{y})). \tag{2.12}$$

Instead of first using $\mu_{B'}(\mathbf{y}) = \max \mu_{B_j}(\mathbf{y})$ and then applying defuzzification, we now first apply defuzzification using $b_j = \text{mom}(B_j)$. Now, a crisp output $\mathbf{y}'$ is obtained by taking the weighted average of b_j. So,

$$\mathbf{y}' = \frac{\sum_{j=1}^{M} \omega_j b_j}{\sum_{j=1}^{M} \omega_j}, \qquad \text{where} \qquad \omega_j = \mu_{B'}(b_j). \tag{2.13}$$

The weight ω_j is thus the maximum of the degrees of fulfilment β_i over all rules $\mathcal{R}_i$ with consequent B_j. In this way, an integration over the domain is avoided.

2.5 Rules with several inputs

Previously, we have considered multivariate membership functions $\mu_A(\mathbf{x})$. Sometimes, it may be convenient to use univariante membership functions $\mu_A(x)$. But what do we do then if we still have multiple variables $x_1, \ldots, x_n$? In this case, we simply add them together in the antecedent. This gives us the **conjunctive form** of the antecedant:

$$\mathcal{R}_i : \textbf{if } x_1 \text{ is } A_{i1} \textbf{ and } \ldots \textbf{ and } x_{in} \text{ is } A_{in} \textbf{ then } \mathbf{y} \text{ is } B_i. \tag{2.14}$$

This is, in fact, a special case of our previous multivariate rules. In fact, if we use $A_i = A_{i1} \times \ldots \times A_{in}$, and insert this into the normal multivariate rule, then we get exactly the same result. As such, the possibilities of the conjunctive form are limited. Also, it is often necessary to define a lot of rules. For every combination of $x_1, \ldots, x_n$, a rule is necessary.

One way in which the number of rules can be reduced, is by using additional **logical connectives**, like 'or' and 'not'. In this way, a rule can be defined like

$$\mathcal{R}_i : \textbf{if } x_1 \text{ is } \textbf{not } A_{i1} \textbf{ or } x_{i2} \text{ is } A_{i2} \textbf{ then } \mathbf{y} \text{ is } B_i. \tag{2.15}$$

In this way, less rules are required than in the conjunctive form. Yet still, this method allows for fewer possibilities than the multivariate rule form. So, the multivariate rule form is the most general form you can use.

3 Other kinds of fuzzy models

3.1 The singleton model

A special case of the linguistic fuzzy model is the **singleton model**. It is obtained when the consequence fuzzy sets B_i are singleton sets. In this case, we can write the rules as

$$\mathcal{R}_i : \textbf{if } \mathbf{x} \text{ is } A_i \textbf{ then } y \text{ is } b_i. \tag{3.1}$$

For the singleton method, defuzzification simply means applying the fuzzy-mean method. So we have

$$y = \frac{\sum_{i=1}^{K} \beta_i b_i}{\sum_{i=1}^{K} \beta_i}. \tag{3.2}$$

We can also generalize the singleton model to a class of functions called the **basis functions expansion**. We now have

$$y = \sum_{i=1}^{K} \phi_i(\mathbf{x}) b_i. \tag{3.3}$$

So, for the singleton model, $\phi_i(\mathbf{x})$ is simply the normalized degree of fulfillment of the rule antecedents.

3.2 The fuzzy relational model

The **fuzzy relational model** is an expansion of the linguistic model. In the linguistic model, y always belonged to a certain linguistic term. (e.g. we had y is Fast.) In the relational model, y can also partly belong to multiple linguistic terms. So, an example of a rule might be

$$\textbf{if } x_1 \text{ is } \textit{Low} \textbf{ and } x_2 \text{ is } \textit{High} \textbf{ then } y \text{ is } \textit{Cold}\ (0.9), y \text{ is } \textit{Normal}\ (0.2), y \text{ is } \textit{Warm}\ (0.0). \tag{3.4}$$

Let's denote the set of linguistic terms of antecedent variable x_j by $\mathcal{A}_j$. The set of all combinations of linguistic variables $x_1, \ldots, x_n$ is now denoted by $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_n$. Similarly, we can denote the set of linguistic terms of consequent variable y by $\mathcal{B}$. The fuzzy relational model can now be seen as a fuzzy relation

$$R : \mathcal{A} \times \mathcal{B} \to [0, 1]. \tag{3.5}$$

The relation R can be represented by a matrix. The elements r_{ij} of this matrix now equal the numbers denoted in parantheses in the rules.

It is important to note the difference between the matrix R for the linguistic model and the matrix R for the relational model. In the linguistic model, R denoted the degree of association between elements from X and Y (that is, from the input and the output space). However, in the relational model R denotes the association between the linguistic terms of the input and the output.

So how does inference work in the relational model? Well, we first compute the degree of fulfillment of the rules. This still goes according to

$$\beta_i = \mu_{A_{i1}}(x_1) \wedge \ldots \wedge \mu_{A_{in}}(x_n). \tag{3.6}$$

Now, we can find the degree ω_j to which y belongs to class B_j. This is done using $\omega = \beta \circ R$ or, equivalently,

$$\omega_j = \max_{1 \leq i \leq K} \left(\beta_i \wedge r_{ij} \right). \tag{3.7}$$

Finally, we need to find the defuzzified output y. For that, we simply take the weighted mean of the classes B_j. So,

$$y = \frac{\sum_{j=1}^{M} \omega_j b_j}{\sum_{j=1}^{M} \omega_j}. \tag{3.8}$$

Here, $b_j = \text{cog}(B_j)$ is the centroid of B_j.

The main advantage of the relational model is that the input-output model can be fine-tuned without changing the consequent fuzzy sets. Instead, you can simply adjust the values of r_{ij} in the rules of the fuzzy system.

3.3 The Takagi-Sugeno model

The **Takagi-Sugeno (TS) model** uses crisp functions as consequents. Basically, a rule has the form

$$\mathcal{R}_i : \textbf{if } \mathbf{x} \text{ is } A_i \textbf{ then } \mathbf{y} = \mathbf{f_i}(\mathbf{x}). \tag{3.9}$$

If the function $\mathbf{f_i}(\mathbf{x})$ has an affine form (so $\mathbf{f_i}(\mathbf{x}) = \mathbf{a_i}^T\mathbf{x} + \mathbf{b_i}$), then the model is called an **affine TS model**. To apply inference with the Takagi-Sugeno model, we simply use the fulfillment degrees. So,

$$\mathbf{y} = \frac{\sum_{i=1}^K \beta_i \mathbf{y_i}}{\sum_{i=1}^K \beta_i} = \frac{\sum_{i=1}^K \beta_i \mathbf{f_i}(\mathbf{x})}{\sum_{i=1}^K \beta_i} = \frac{\sum_{i=1}^K \beta_i \left(\mathbf{a_i}^T\mathbf{x} + \mathbf{b_i}\right)}{\sum_{i=1}^K \beta_i}. \tag{3.10}$$

3.4 Dynamic fuzzy systems

Let's examine a time-invariant system. We can model such a system using

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)), \tag{3.11}$$

where $\mathbf{x}(k)$ is the **state**, $\mathbf{u}(k)$ is the **input** and $\mathbf{f}$ is the **state transition function**. We can use a fuzzy model to approximate $\mathbf{f}$. However, it is usually hard to do this, since we can't always measure the state $\mathbf{x}$. So instead, we usually use a fuzzy model to approximate the **output y** of the system. In the **dynamic TS model** this is done according to rules of the form

$$\textbf{if } \mathbf{y}(k) \text{ is } A_{i1} \textbf{ and } \mathbf{y}(k-1) \text{ is } A_{i2} \textbf{ and } \ldots \textbf{ and } \mathbf{y}(k-n_y+1) \text{ is } A_{in_y} \tag{3.12}$$

$$\textbf{and } \mathbf{u}(k) \text{ is } B_{i1} \textbf{ and } \mathbf{u}(k-1) \text{ is } B_{i2} \textbf{ and } \ldots \textbf{ and } \mathbf{u}(k-n_u+1) \text{ is } B_{in_u} \tag{3.13}$$

$$\textbf{then } \mathbf{y}(k+1) = \sum_{j=1}^{n_y} a_{ij} y(k-j+1) + \sum_{j=1}^{n_u} b_{ij} u(k-j+1) + c_i. \tag{3.14}$$

The values of n_u and n_y (i.e. how far we look back in time for the input/output) depend on the order of the dynamic system.