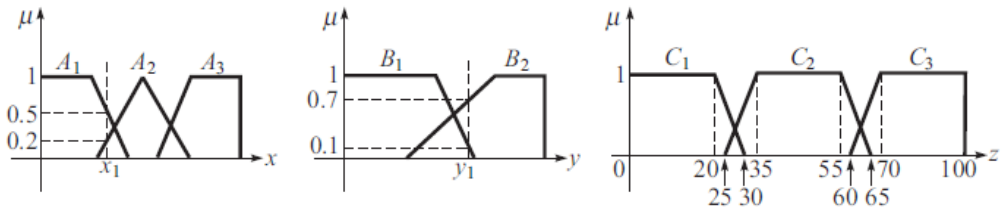
Part 1 – Task 1

Consider the following fuzzy model of a system with inputs *x* and *y* and output *z*:

Rule 1: If *x* is A_3 OR *y* is B_1 THEN *z* is C_1
Rule 2: If *x* is A_2 AND *y* is B_2 THEN *z* is C_2
Rule 3: If *x* is A_1 THEN *z* is C_3

The membership functions of the input and output variables are given in the graphs below:



For this problem, the actual inputs are designated x_1, y_1. By plugging into the membership functions, we are then given their values:

$$\mu(x_1 = A_1) = 0.5 \quad \mu(x_1 = A_2) = 0.2 \qquad \mu(y_1 = B_1) = 0.1 \quad \mu(y_1 = B_2) = 0.7$$

Now we take these fuzzified inputs and apply them to our antecedent rules. For the rules with more than one antecedent, we obtain a single number using the AND/OR rules. If AND is represented by the min function and OR is represented by max, this can also be stated in mathematical notation:

$$\mu_A \cup \mu_B = \max[\mu_A(x), \mu_B(x)] \quad \mu_A \cap \mu_B = \min[\mu_A(x), \mu_B(x)]$$

And we can then plug in our example values for each of our three rules and plot them below:

Rule 1: If $x = A_3$ OR $y = B_1$ THEN $z = C_1$

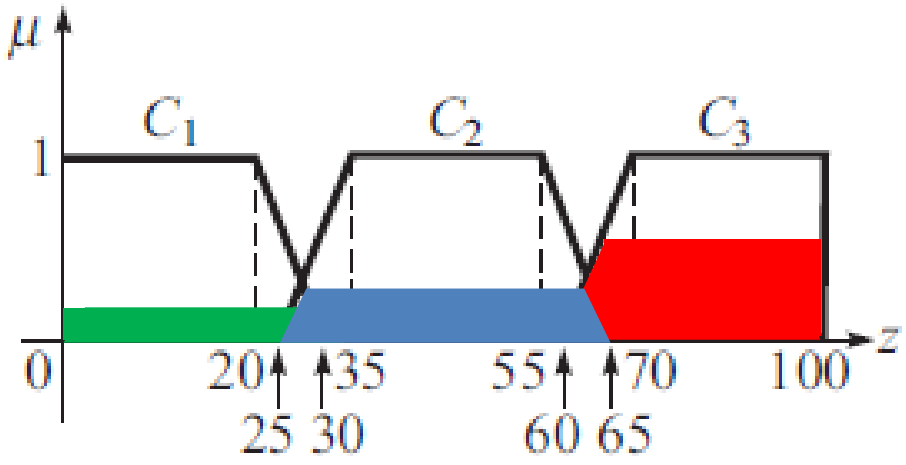
$$\Longrightarrow \max[\mu_{A_3}(x_1), \mu_{B_1}(y_1)] = \max[(0), (0.1)] = 0.1 \qquad \mu_{C_1}(z_1) = \mu_{C_1}(0.1) = 29$$

Rule 2: If $x = A_2$ AND $y = B_2$ THEN $z = C_2$

$$\Longrightarrow \min[\mu_{A_2}(x_1), \mu_{B_2}(y_1)] = \min[(0.2), (0.7)] = 0.2 \qquad \mu_{C_2}(z_2) = \mu_{C_2}(0.2) = [28,63]$$

Rule 3: If $x = A_1$ THEN $z = C_3$

$$\Longrightarrow \qquad \mu_{A_2}(x_1) = 0.5 \qquad \mu_{C_3}(z_3) = \mu_{C_3}(0.5) = 65$$



Task 2.

The article by Cococcioni et al. (2008) addresses the problem of detecting and determining concentrations of visual biological quantities in bodies of water from remote sensors of reflected sunlight. One method of accomplishing this task involves the Takagi-Sugeno (TS) fuzzy model, since the problem of classification involves optical input features that are not strictly categorical. In this case, the antecedents of the fuzzy rules are determined by primarily applying a fuzzy clustering algorithm, then by projecting the resulting clusters onto each input variable. The number of rules is equal to the number of clusters determined by the clustering algorithm. The resulting parameter coefficients are then computed through least squares estimation. The fuzzy rules follow the simple pattern displayed in Task 1 above, wherein they are defined as a set of n "IF-THEN" statements for the membership functions of the fuzzified inputs. The THEN statements can be more generally defined in the form of $y_i = f_i(X_1, \dots, X_p)$ where p is the number of parameters. The consequent parameters are defined linearly in the paper but can extend to more general cases. In the specific context of the article, the approach is used to classify the solution domination in chromosome coding. An example is provided in their solution section, which shows a combination of IF-AND statements for the classes Low, Medium, High (reflectance). It also demonstrates that each rule only uses a subset of the parameters, and the consequent is a linear combination of these inputs.

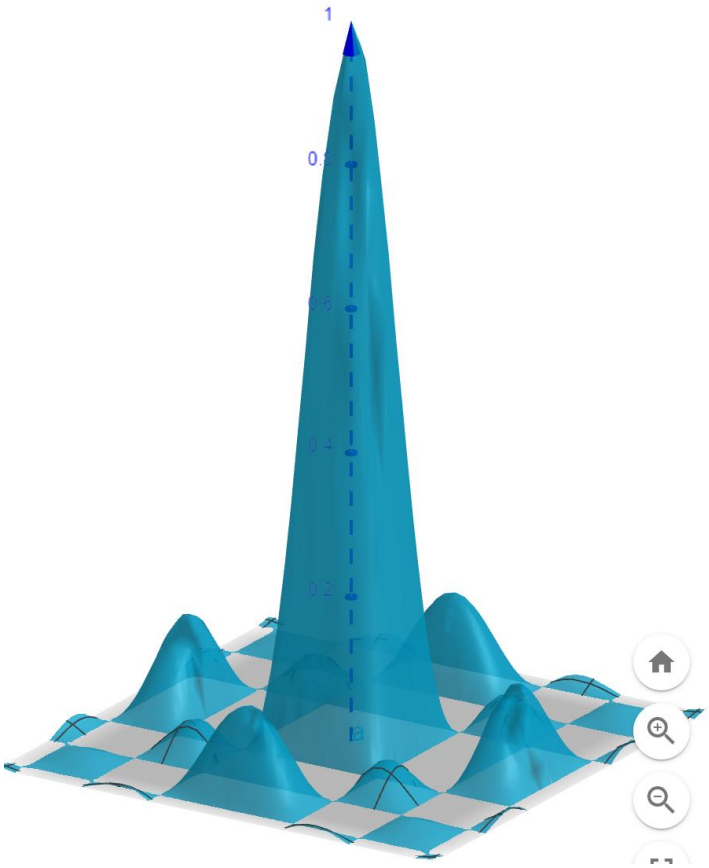
The paper also demonstrates a similar algorithm, namely the Adaptive-Network Based Fuzzy Inference System (ANFIS). This is a neuro-fuzzy model, which uses the above TS to implement an artificial neural network and then using it to update the parameter estimations. The authors note the classical implementation of ANFIS is very messy as it forces large amounts of rules that serve no purpose (skip over rule activation if membership function is equal to 1 everywhere). Thus they introduce a second classification of ANFIS, which is sparsely connected. This sharply reduces the complexity of the model and the authors note that they observed only negligible depreciations in accuracy due to the lowered capabilities. They also note that setting a maximum number of rules allows the program to be greatly sped in terms of computing time. The learning phase of ANFIS simultaneously tunes the antecedent and consequent parameters by adopting gradient descent for the antecedents, then using recursive least squares to determine the consequents. This procedure is performed recursively, and the number of iterations is user-defined. The application in the work is for chromosome solution estimation, wherein the program iteratively tries to find solutions by applying mutation operators and applying ANFIS until one is not dominated by the other. This process continues to try to find the solution in the archive residing in the region with the highest crowding degree.

Part 2 – Fuzzy Models

Problem 1. Consider a two-dimensional *sinc* equation defined by:

$$y = \text{sinc}(x_1, x_2) = \frac{\sin(x_1)\sin(x_2)}{x_1 x_2}$$

Training data are sampled uniformly from the input range [–10, 10] × [–10, 10]. We can view the theoretical plot of this function on the right, which is defined in the prescribed ranges for x_1 and x_2, and from [0,1] in y. As is clear, the center of the graph is where the largest y values lie, so we construct two triangular membership functions for each of the in the corresponding Jupyter notebook, which map each variable to classes of *small*, *medium*, and *large*.



A Takagi-Sugeno fuzzy model with linear static mapping as rule consequents defines rules for the system and how the inputs map to the output. In other words, we can therefore can define rules that show how x_1 and x_2 affect what y will be classified as.

In general, a set of R rules for n variables can be defined as:

Rule r: If $x_1 = A_{1,k_1}$ AND ... AND $x_n = A_{n,k_n}$ THEN $y^r = a_0^r + a_1^r x_1 + \cdots + a_n^r x_n \ : \ r = 1,2,\ldots R$

Therefore, the consequent part is a linear function of the input variables. The aim of this function is to describe the local behavior of the system. Each *r* gives rise to a local linear model. The set of selected rules (*R*) are required to approximate the function that theoretically underlines the system most consistently, which in this case is the two-variable sinc function. The set of IF-THEN rules dictates how the inputs (x_1,x_2) are mapped to the output (y) in the form of defined fuzzy regions. The rules below and the Takagi-Sugeno FIS are carried out in the Jupyter Notebook:

Rule 1: If $x_1 = A_1$ AND $x_2 = B_1$ THEN $y = C_1$
Rule 2: If $x_1 = A_2$ AND $x_2 = B_2$ THEN $y = C_2$
Rule 3: If $x_1 = A_1$ AND $x_2 = B_2$ THEN $y = C_3$

Problem 2. To identify the non-linear system

$$y = (1 + (x_1)^{0.5} + (x_2)^{-1} + (x_3)^{-1.5})^2$$

Training and testing data are sampled uniformly from the input ranges:

Training data: [1, 6] × [1, 6] × [1, 6]
Testing data: [1.5, 5.5] × [1.5, 5.5] × [1.5, 5.5]

We need to assign two membership functions to each input variable. This is accomplished by finding cluster means in the skfuzzy package from the input data and how it is mapped to y. This allows us to define fuzzy sets for each of x_1,x_2,x_3 and we can therefore find the rules for the Takagi-Sugeno method that could be used in an ANFIS model.

Recall that the membership function A_{ij} is given by:

$$A_{ij}(X_j) = \exp[-\frac{1}{2}\left\{\frac{X_j - x_{ij}^*}{\sigma_{ij}}\right\}^2]$$

Where X_j is the jth input variable, x_{ij}^* is the jth element of cluster center x_i^*, and σ_{ij} is a constant related to the radius of the cluster. The degree of fulfillment of each rule is computed by using multiplication as the AND operator, and, because the fuzzy system performs classification, we simply select the consequent of the rule with the highest degree of fulfillment to be the output of the fuzzy system.

Because of the squared term within the definition of A_{ij}, this implies that the membership function for each data point is defined based on the Euclidean distance away from each cluster center. The clusters can then be converted to fuzzy rules which are then defined generally as

Rule r: If $x_1 = A_{1,k_1}$ AND ... AND $x_n = A_{n,k_n}$THEN $y^r = a_0^r + a_1^r x_1 + \cdots + a_n^r x_n \ : \ r = 1,2, \ldots R$

And as an example as:

Rule 1: If $x_1 = A_1$ AND $x_2 = B_1$ AND $x_3 = C_1$THEN $y^{(1)} = a_0^{(1)} + a_1^{(1)} x_1 + a_2^{(1)} x_2 + a_3^{(1)} x_3$

The centers are computed in the Jupyter Notebook. Since there are 3 variables and we have 2 classes of each, there will be 8 total rules:

Rule 2: If $x_1 = A_1$ AND $x_2 = B_1$ AND $x_3 = C_2$THEN $y^{(2)} = a_0^{(2)} + a_1^{(2)} x_1 + a_2^{(2)} x_2 + a_3^{(2)} x_3$
Rule 3: If $x_1 = A_1$ AND $x_2 = B_2$ AND $x_3 = C_1$THEN $y^{(3)} = a_0^{(3)} + a_1^{(3)} x_1 + a_2^{(3)} x_2 + a_3^{(3)} x_3$
Rule 4: If $x_1 = A_2$ AND $x_2 = B_1$ AND $x_3 = C_1$THEN $y^{(4)} = a_0^{(4)} + a_1^{(4)} x_1 + a_2^{(4)} x_2 + a_3^{(4)} x_3$
Rule 5: If $x_1 = A_1$ AND $x_2 = B_2$ AND $x_3 = C_2$THEN $y^{(5)} = a_0^{(5)} + a_1^{(5)} x_1 + a_2^{(5)} x_2 + a_3^{(5)} x_3$
Rule 6: If $x_1 = A_2$ AND $x_2 = B_1$ AND $x_3 = C_2$THEN $y^{(6)} = a_0^{(6)} + a_1^{(6)} x_1 + a_2^{(6)} x_2 + a_3^{(6)} x_3$
Rule 7: If $x_1 = A_2$ AND $x_2 = B_2$ AND $x_3 = C_1$THEN $y^{(7)} = a_0^{(7)} + a_1^{(7)} x_1 + a_2^{(7)} x_2 + a_3^{(7)} x_3$
Rule 8: If $x_1 = A_2$ AND $x_2 = B_2$ AND $x_3 = C_2$THEN $y^{(8)} = a_0^{(8)} + a_1^{(8)} x_1 + a_2^{(8)} x_2 + a_3^{(8)} x_3$

Notice that in y, the only thing that changes is the parameters associated with the rule. These are related to the base function, which raises each of x_1,x_2,x_3 to a decimal power, then adds 1 and squares the result. The fuzzy sets are all defined in the corresponding notebook. These can be extracted and employed in an ANFIS model. This is one that combines both a normal FIS, such as the Takagi-Sugeno, and a neural network that enables both learning and explanation of how it arrived at its particular result. It works by taking these rules, then outputs products of all incoming signals. It then calculates the firing strength of each node, and finds the weighted consequent parameters. These consequents are then summed to produce the final node.

References:

Chen-Chia Chuang, Shun-Feng Su, & Song-Shyong Chen. (2001). Robust TSK fuzzy modeling for function approximation with outliers. *IEEE Transactions on Fuzzy Systems*, *9*(6), 810–821. https://doi.org/10.1109/91.971730

Cococcioni, M., Corsini, G., Lazzerini, B., & Marcelloni, F. (2009). Solving the ocean color inverse problem by using evolutionary multi-objective optimization of neuro-fuzzy systems. *International Journal of Knowledge-Based and Intelligent Engineering Systems*, *12*(5–6), 339–355. https://doi.org/10.3233/kes-2008-125-604

Gopal, M. (2020). *Applied Machine Learning* (1st ed.) [E-book]. McGraw-Hill Education. https://www.gcumedia.com/digital-resources/mcgraw-hill/2019/applied-machine-learning_1e.php

Spolaor, S., Fuchs, C., Cazzaniga, P., Kaymak, U., Besozzi, D., & Nobile, M. S. (2020). Simpful: A User-Friendly Python Library for Fuzzy Logic. *International Journal of Computational Intelligence Systems*, *13*(1), 1687. https://doi.org/10.2991/ijcis.d.201012.002