Knowledge-Based Control Systems

Problems with Solutions

Robert Babuška, Alfredo Núñez, and Jens Kober
Delft Center for Systems and Control
Version 19-07-2017

Delft University of Technology
Delft, the Netherlands

Contents

1 Introduction

This is a draft document. If you find mistakes, or if you find ways to improve the answers here proposed, please send an email to `J.Kober@tudelft.nl`.

Small rewards in the final grade of the course will be granted only to the first student that notifies a mistake, or only to the first that proposes a good suggestion for an answer.

2 Fuzzy Sets and Relations

---

1. What is the difference between the membership function of an ordinary set and of a fuzzy set?

   Answer: The membership function of an ordinary set is a mapping from the universal set to the set $\{0, 1\}$, while in the case of a fuzzy set it is a mapping to the closed *interval* $[0, 1]$.

---

2. Consider fuzzy set C defined by its membership function $\mu_C(x)\colon \mathbb{R} \to [0, 1]$: $\mu_C(x) = 1/(1 + |x|)$. Compute the α-cut of C for $\alpha = 0.5$.

   Answer: Given, set C is a fuzzy set and the membership function will lie in the interval $\{0, 1\}$. The α-cut is the list of elements that belong to the set which obeys (2.1).

$$C_{0.5} = \{x \mid \mu_C(x) \geq 0.5\} \tag{2.1}$$

$$\begin{aligned} \frac{1}{1+|x|} &\geq 0.5 \\ |x| &\leq 1 \end{aligned}$$

$$-1 \leq x \leq 1$$

   This implies that α-cut of C belongs to the interval $[-1, 1]$ for $\alpha = 0.5$.

   Whereas, the *strict* α-cut is given by (2.2).

$$C_{0.5} = \{x \mid \mu_C(x) > 0.5\} \tag{2.2}$$

   Hence, the strict α-cut of C for $\alpha = 0.5$ lies in the interval $(-1, 1)$.

---

3. Consider fuzzy sets A and B such that $\mathrm{core}(A) \cap \mathrm{core}(B) = \emptyset$. Is fuzzy set $C = A \cap B$ normal? What condition must hold for the supports of A and B such that $\mathrm{card}(C) > 0$ always holds?

   Answer: A fuzzy set is said to be normal if at-least one element x in the domain X have membership degree of 1. The core of the fuzzy set is a crisp subset of X consisting of all elements with membership grades equal to one as given by

$$\mathrm{core}(A) = \{x \mid \mu_A(x) = 1, x \in X\}$$

If the core$(A) \cap$ core$(B) = \emptyset$ implies that no elements in A and B have membership degree 1. Therefore, the fuzzy set $C = A \cap B$ is not normal.

The support of a fuzzy set A is the crisp subset of X whose elements all have nonzero membership grades.

$$\text{supp}(A) = \{x \mid \mu_A(x) > 0, x \in X\}$$

The cardinality of a fuzzy set C is the sum of the membership degrees. Hence, it must hold that supp$(A) \cap$ supp$(B) \neq \emptyset$, otherwise the intersection $A \cap B$ would be empty and the cardinality of an empty set is zero.

---

4. Consider fuzzy set A defined in $X \times Y$ with $X = \{x_1, x_2\}$, $Y = \{y_1, y_2\}$:

$$A = \{0.1/(x_1, y_1),\ 0.2/(x_1, y_2),\ 0.7/(x_2, y_1), 0.9/(x_2, y_2)\}$$

Compute the projections of A onto X and Y.

Answer: The projections are:

$$\begin{aligned} \text{proj}_X(A) &= \{\max(0.1, 0.2)/x_1,\ \max(0.7, 0.9)/x_2\} = \{0.2/x_1,\ 0.9/x_2\}, \\ \text{proj}_Y(A) &= \{\max(0.1, 0.7)/y_1,\ \max(0.2, 0.9)/y_2\} = \{0.7/y_1,\ 0.9/y_2\}. \end{aligned}$$

---

5. Compute the cylindrical extension of fuzzy set $A = \{0.3/x_1,\ 0.4/x_2\}$ into the Cartesian product domain $\{x_1, x_2\} \times \{y_1, y_2\}$.

Answer: The cylindrical extension is:

$$\text{ext}(A) = \{0.3/(x_1, y_1),\ 0.3/(x_1, y_2),\ 0.4/(x_2, y_1), 0.4/(x_2, y_2)\}.$$

---

6. For fuzzy sets $A = \{0.1/x_1,\ 0.6/x_2\}$ and $B = \{1/y_1,\ 0.7/y_2\}$ compute the union $A \cup B$ and the intersection $A \cap B$. Use the Zadeh's operators (max, min).

Answer:

a)

$$\begin{aligned} A \cup B &= \{\max(0.1, 1)/(x_1, y_1), \max(0.1, 0.7)/(x_1, y_2), \\ &\quad \max(0.6, 1)/(x_2, y_1), \max(0.6, 0.7)/(x_2, y_2)\} \\ &= \{1/(x_1, y_1),\ 0.7/(x_1, y_2),\ 1/(x_2, y_1),\ 0.7/(x_2, y_2)\} \end{aligned}$$

b)

$$\begin{aligned} A \cap B &= \{\min(0.1, 1)/(x_1, y_1), \min(0.1, 0.7)/(x_1, y_2), \\ &\quad \min(0.6, 1)/(x_2, y_1), \min(0.6, 0.7)/(x_2, y_2)\} \\ &= \{0.1/(x_1, y_1),\ 0.1/(x_1, y_2),\ 0.6/(x_2, y_1),\ 0.6/(x_2, y_2)\} \end{aligned}$$

7. Given is a fuzzy relation $R\colon X \times Y \rightarrow [0,1]$:

$$R \quad = \quad \begin{array}{c|ccc} & y_1 & y_2 & y_3 \\ \hline x_1 & 0.7 & 0.3 & 0.1 \\ x_2 & 0.4 & 0.8 & 0.2 \\ x_3 & 0.1 & 0.2 & 0.9 \end{array}$$

and a fuzzy set $A = \{0.1/x_1,\ 1/x_2,\ 0.4/x_3\}$. Compute fuzzy set $B = A \circ R$, where '$\circ$' is the max-min composition operator.

Answer: Fuzzy subset B of Y can be induced by A through the composition of A and R:

$$B = \operatorname{proj}_Y R \cap \operatorname{ext}_{X\times Y}(A)$$

The composition can be regarded in two phases:

- combination (intersection)
- projection

A is a fuzzy set with membership function $\mu_A(x)$ and R is a fuzzy relation with membership function $\mu_R(x,y)$:

$$\mu_B(y) = \sup_x \left(\min\left(\mu_A(x), \mu_R(x,y)\right)\right)$$

where the cylindrical extension of A into $X \times Y$ is implicit and sup and min represent the projection and combination phase, respectively. Here, it can be re-written as follows,

$$\begin{aligned}
\mu_B(y) &= \max_x \left(\min\left(\mu_A(x), \mu_R(x,y)\right)\right) \\
\mu_B(y) &= \underbrace{\begin{bmatrix} 0.1 \\ 1 \\ 0.4 \end{bmatrix}}_{\mu_A(y)} \circ \underbrace{\begin{bmatrix} 0.7 & 0.3 & 0.1 \\ 0.4 & 0.8 & 0.2 \\ 0.1 & 0.2 & 0.9 \end{bmatrix}}_{\mu_R(x,y)} \\
\mu_B(y) &= \max_x \underbrace{\begin{bmatrix} 0.1 & 0.1 & 0.1 \\ 0.4 & 0.8 & 0.2 \\ 0.1 & 0.2 & 0.4 \end{bmatrix}}_{(\min(\mu_A(y),\mu_R(x,y))} \\
\mu_B(y) &= \underbrace{\begin{bmatrix} 0.4 & 0.8 & 0.4 \end{bmatrix}}_{\max_x(\min(\mu_A(y),\mu_R(x,y))}
\end{aligned}$$

$B = \{0.4/y_1,\ 0.8/y_2,\ 0.4/y_3\}$

8. Prove that the following De Morgan law $\overline{(A \cup B)} = \bar{A} \cap \bar{B}$ is true for fuzzy sets A and B, when using the Zadeh's operators for union, intersection and complement.

Answer: Using membership functions, we have:

$$\begin{aligned}
\overline{(A \cup B)} &= 1 - \max(\mu_A, \mu_B), \\
\bar{A} \cap \bar{B} &= \min(1 - \mu_A, 1 - \mu_B)\,.
\end{aligned}$$

Further, realizing that $\max(\alpha, \beta) = -\min(-\alpha, -\beta)$ and $\min(\alpha + \gamma, \beta + \gamma) = \min(\alpha, \beta) + \gamma$, we obtain $1 - \max(\mu_A, \mu_B) = 1 + \min(-\mu_A, -\mu_B) = \min(1 - \mu_A, 1 - \mu_B)$ which concludes the proof.

3 Fuzzy Systems

1. Give a definition of a linguistic variable. What is the difference between linguistic variables and linguistic terms?

   Answer: A linguistic variable L is defined as the quintuple $L = (\mathbf{x}, \mathcal{A}, X, g, m)$, where $\mathbf{x}$ is the base variable, $\mathcal{A} = \{A_1, A_2, \ldots, A_N\}$ is the set of linguistic terms, X is the domain of $\mathbf{x}$, g is a syntactic rule for generating linguistic terms and m is a semantic rule that assigns to each linguistic term a fuzzy set in X. The difference between linguistic variables and linguistic terms is clear from this definition.

2. The minimum t-norm can be used to represent if-then rules in a similar way as fuzzy implications. It is however not an implication function. Explain why. Give at least one example of a function that is a proper fuzzy implication.

   Answer: Clearly, the truth table of a t-norm (which is a conjunction operator) is different than that of an implication (even for Boolean logic, which can be seen as a special case fuzzy logic):

a	b	$a \wedge b$	$a \to b$
0	0	0	1
0	1	0	1
1	0	0	0
1	1	1	1

   This has consequences for logical inference (using the modus ponens rule, for instance). With the conjunction, when $a \wedge b$ is true and a is true (the last row of the table), one can infer that also b must be true. Conversely, when $a \wedge b$ is true and b is true, also a must be true (again the last row of the table). With the implication, however, the latter case does not hold. When $a \to b$ is true and b is true, nothing can be inferred about a (a can be both true and false, see the second and fourth row of the table).

   One particular fuzzy implication function can be obtained from the definition of the material implication $a \to b = \bar{a} \vee b$ by using the Zadeh's complement and union operators: $a \to b = \max(1 - a, b)$. It is easy to verify that this function satisfies the truth table for implications. Another example is the Łukasiewicz implication, which is obtained in the same way by using the Łukasiewicz (bold) union operator.

3. Consider a rule **If** x is A **then** y is B with fuzzy sets $A = \{0.1/x_1,\ 0.4/x_2,\ 1/x_3\}$ and $B = \{0/y_1,\ 1/y_2,\ 0.2/y_3\}$. Compute the fuzzy relation R that represents the

truth value of this fuzzy rule. Use first the minimum t-norm and then the Łukasiewicz implication. Discuss the difference in the results.

Answer: Using the minimum t-norm, the following relation is obtained:

$$R_m = \begin{bmatrix} 0 & 0.1 & 0.1 \\ 0 & 0.4 & 0.2 \\ 0 & 1 & 0.2 \end{bmatrix}$$

Using the Łukasiewicz implication, the following relation is obtained:

$$R_l = \begin{bmatrix} 0.9 & 1 & 1 \\ 0.6 & 1 & 0.8 \\ 0 & 1 & 0.2 \end{bmatrix}$$

When $\mu_A(x)$ is high or 1, both R_m and R_l are similar to $\mu_B(y)$ as nothing can be inferred using only the information of x (third row of both matrices). When the $\mu_A(x)$ is low or 0, R_m tends to have low values and R_l higher values (first row of both matrices). For the minimum t-norm, if $\mu_A(x)$ is low, it doesn't matter $\mu_B(y)$, the fuzzy relation will have a low value because of the **and** operation. In the case of Łukasiewicz implication, if $\mu_A(x)$ is low, independent of $\mu_B(y)$, the fuzzy relation will have high value as the implication will always be true.

For minimum t-norm, $\mu_A \wedge \mu_B$ does not hold if either μ_A or μ_B or both μ_A and μ_B does not hold, which is why the first column of R_m is 0. For implication function, $\mu_A \rightarrow \mu_B$ holds true if μ_B is true, independent of μ_A, which is why the second column of R_l is 1.

---

4. Explain the steps of the Mamdani (max-min) inference algorithm for a linguistic fuzzy system with one (crisp) input and one (fuzzy) output. Apply these steps to the following rule base:

1) **If** x is A_1 **then** y is B_1,
2) **If** x is A_2 **then** y is B_2,

with

$$\begin{aligned} A_1 &= \{0.1/1,\ 0.6/2,\ 1/3\}, & A_2 &= \{0.9/1,\ 0.4/2,\ 0/3\}, \\ B_1 &= \{1/4,\ 1/5,\ 0.3/6\}, & B_2 &= \{0.1/4,\ 0.9/5,\ 1/6\}, \end{aligned}$$

State the inference in terms of equations. Compute the output fuzzy set B' for $x = 2$.

Answer: a) The Mamdani algorithm for a rule base with K rules:

1) For each rule, compute the degree of fulfillment by: $\beta_i = \mu_{A_i}(x), i = 1, \ldots, K$.
2) For each rule, derive the output fuzzy sets B_i': $\mu_{B_i'}(y) = \beta_i \wedge \mu_{B_i}(y), \quad y \in Y$.

3) Aggregate the output fuzzy sets B_i': $\mu_{B'}(y) = \max_{1\le i\le K} \mu_{B_i'}(y), \quad y \in Y$.

b) Applying to the given rules and fuzzy sets:

1) Degrees of fulfillment:

$$\begin{aligned}\beta_1 &= \mu_{A_1}(2) = 0.6\\ \beta_2 &= \mu_{A_2}(2) = 0.4\end{aligned}$$

2) Individual consequent fuzzy sets:

$$\begin{aligned}B_1' &= \beta_1 \wedge B_1 = \{0.6/4,\ 0.6/5,\ 0.3/6\}\\ B_2' &= \beta_2 \wedge B_2 = \{0.1/4,\ 0.4/5,\ 0.4/6\}\end{aligned}$$

3) Aggregated consequent fuzzy set:

$$B' = B_1' \cup B_2' = \{0.6/4,\ 0.6/5,\ 0.4/6\}$$

---

5. Define the center-of-gravity and the mean-of-maxima defuzzification methods. Apply them to the fuzzy set $B = \{0.1/1,\ 0.2/2,\ 0.7/3,\ 1/4\}$ and compare the numerical results.

Answer: The center-of-gravity method:

$$\operatorname{cog}(B') = \frac{\sum_{j=1}^{F} \mu_{B'}(y_j)\, y_j}{\sum_{j=1}^{F} \mu_{B'}(y_j)}$$

where F is the number of elements y_j in Y (continuous domains must be discretized). The mean-of-maxima method:

$$\operatorname{mom}(B') = \operatorname{cog}\{y \mid \mu_{B'}(y) = \max_{y\in Y} \mu_{B'}(y)\}\,.$$

The numerical results are: $y_{\text{cog}} = (0.1{+}0.4{+}2.1{+}4)/(0.1{+}0.2{+}0.7{+}1) = 6.6/2 = 3.3$ and $y_{\text{mom}} = 4$.

---

6. Consider the following Takagi–Sugeno rules:

1) **If** x is A_1 **and** y is B_1 **then** $z_1 = x + y + 1$
2) **If** x is A_2 **and** y is B_1 **then** $z_2 = 2x + y + 1$
3) **If** x is A_1 **and** y is B_2 **then** $z_3 = 2x + 3y$
4) **If** x is A_2 **and** y is B_2 **then** $z_4 = 2x + 5$

Give the formula to compute the output z and compute the value of z for $x = 1$, $y = 4$ and the antecedent fuzzy sets

$$\begin{aligned}A_1 &= \{0.1/1,\ 0.6/2,\ 1/3\}, & A_2 &= \{0.9/1,\ 0.4/2,\ 0/3\},\\ B_1 &= \{1/4,\ 1/5,\ 0.3/6\}, & B_2 &= \{0.1/4,\ 0.9/5,\ 1/6\}\,.\end{aligned}$$

Answer: a) The output is given by

$$z = \frac{\beta_1(x+y+1) + \beta_2(2x+y+1) + \beta_3(2x+3y) + \beta_4(2x+5)}{\beta_1+\beta_2+\beta_3+\beta_4}$$

with

$$\begin{aligned} \beta_1 &= \mu_{A_1}(x) \wedge \mu_{B_1}(y), & \beta_2 &= \mu_{A_2}(x) \wedge \mu_{B_1}(y) \\ \beta_3 &= \mu_{A_1}(x) \wedge \mu_{B_2}(y), & \beta_4 &= \mu_{A_2}(x) \wedge \mu_{B_2}(y) \end{aligned}$$

Other t-norms can be used as well.

b) Fill out $x = 1$ and $y = 4$ in the antecedent fuzzy sets to obtain the required membership functions (by taking the μ_{Ai} at /1 and μ_{Bi} at /4) for calculation of the degrees of fulfillment (β_i). The degrees of fulfillment are (using the minimum operator):

$$\beta_1 = 0.1, \quad \beta_2 = 0.9 \quad \beta_3 = 0.1, \quad \beta_4 = 0.1$$

which gives

$$z = \frac{0.6+6.3+1.4+0.7}{0.1+0.9+0.1+0.1} = \frac{9}{1.2} = 7.5$$

---

7. Consider an unknown dynamic system $y(k+1) = f\big(y(k), u(k)\big)$. Give an example of a singleton fuzzy model that can be used to approximate this system. What are the free parameters in this model?

Answer: a) A possible rule base is:

1) **If** $y(k)$ is A_1 **and** $u(k)$ is B_1 **then** $y(k+1) = c_1$
2) **If** $y(k)$ is A_2 **and** $u(k)$ is B_1 **then** $y(k+1) = c_2$
3) **If** $y(k)$ is A_1 **and** $u(k)$ is B_2 **then** $y(k+1) = c_3$
4) **If** $y(k)$ is A_2 **and** $u(k)$ is B_2 **then** $y(k+1) = c_4$

You need to also define the operators **and** and **then** .

b) The free parameters are the membership functions A_1, A_2, B_1, B_2, and the consequent parameters c_1 to c_4. The membership functions are usually chosen as some functions (exponential, triangular, etc.) parameterized by a small number of parameters (centers, spreads, etc.).

4 Fuzzy Clustering

1. State the definitions and discuss the differences of fuzzy and non-fuzzy (hard) partitions. Give an example of a fuzzy and non-fuzzy partition matrix. What are the advantages of fuzzy clustering over hard clustering?

   Answer: With N being the number of data samples and c the number of clusters, the hard partition is defined by:

$$\mu_{ik} \in \{0,1\}, \qquad 1 \le i \le c, \quad 1 \le k \le N,$$
$$\text{with} \quad \sum_{i=1}^{c} \mu_{ik} = 1, \qquad 1 \le k \le N, \qquad \text{and} \qquad 0 < \sum_{k=1}^{N} \mu_{ik} < N, \quad 1 \le i \le c\,.$$

   The fuzzy partition is defined by:

$$\mu_{ik} \in [0,1], \qquad 1 \le i \le c, \quad 1 \le k \le N,$$
$$\text{with} \quad \sum_{i=1}^{c} \mu_{ik} = 1, \qquad 1 \le k \le N, \qquad \text{and} \qquad 0 < \sum_{k=1}^{N} \mu_{ik} < N, \quad 1 \le i \le c\,.$$

   One example of fuzzy partition matrix is:

$$\mathbf{U}_f = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 0.8 & 0.5 & 0.5 & 0.2 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.2 & 0.5 & 0.5 & 0.8 & 1.0 & 1.0 & 1.0 \end{bmatrix}.$$

   One example of non-fuzzy partition matrix is:

$$\mathbf{U}_n = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 1.0 & 1.0 & 1.0 \end{bmatrix}.$$

   In hard partitioning, each data point is assigned to one cluster; the data points cannot be part of more than one cluster. In fuzzy clustering, data points get a membership value assigned to all clusters. This means every data point is part of every cluster (though it may be infinitesimally small in most clusters). In the example above, hard partitioning ($\mathbf{U}_n$) groups z_1 to z_6 together, as well as z_7 to z_10. Since the data points must either be part of subsets $\mathbf{A}_1$ or $\mathbf{A}_2$, some information is lost in the process of clustering (a data point that is mostly part of $\mathbf{A}_1$ is now seen as completely part of $\mathbf{A}_1$) . Looking at fuzzy clustering ($\mathbf{U}_f$ above), one can see that point z_4 to z_7 are not completely part of either set, but rather somewhat part of both sets. More information is preserved in this form of clustering, which is exactly the advantage of fuzzy clustering over hard partitioning.

Another limitation of hard partitioning is when a data point (z) is equally part of two classes $\mathbf{A}_1$ and $\mathbf{A}_2$, and therefore cannot be fully assigned to either one. Fuzzy partitioning amounts for this limitation as well by assigning the membership value 0.5 to both classes.

Defining a data point as partly in one or multiple clusters can let you maintain extra information about the data point which can be used afterwards. For instance when classifying objects seen by an intelligent vehicles, pedestrians are important to find as these can start moving into the path of the vehicle and you most certainly do not want the vehicle to hit the pedestrian. By applying fuzzy partitioning, the degree of an object being part of the pedestrian cluster can represent some kind of 'degree of probability' of the object being a pedestrian and this can put flags around objects that have for instance an 40% chance of being a pedestrian. These objects can than be monitored more closely by the car and would otherwise be discarded as not being a pedestrian (while 40% of the times, it is).

(Draft answer).

---

2. State mathematically at least two different distance norms used in fuzzy clustering. Explain the differences between them.

Answer: Distance norms have the following form:

$$D_{ik}^2 = (\mathbf{z}_k - \mathbf{v}_i)^T \mathbf{A} (\mathbf{z}_k - \mathbf{v}_i).$$

A common choice is the standard Euclidean norm:

$$D_{ik}^2 = (\mathbf{z}_k - \mathbf{v}_i)^T (\mathbf{z}_k - \mathbf{v}_i).$$

Another choice is using as $\mathbf{A}$ a diagonal matrix that accounts for different variances in the directions of the coordinate axes of $\mathbf{Z}$:

$$\mathbf{A} = \begin{bmatrix} (1/\sigma_1)^2 & 0 & \cdots & 0 \\ 0 & (1/\sigma_2)^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (1/\sigma_n)^2 \end{bmatrix}.$$

This matrix induces a diagonal norm on $\mathbb{R}^n$.

Finally, $\mathbf{A}$ can be defined as the inverse of the covariance matrix of $\mathbf{Z}$: $\mathbf{A} = \mathbf{R}^{-1}$, with

$$\mathbf{R} = \frac{1}{N} \sum_{k=1}^{N} (\mathbf{z}_k - \bar{\mathbf{z}})(\mathbf{z}_k - \bar{\mathbf{z}})^T .$$

Here $\bar{\mathbf{z}}$ denotes the mean of the data. In this case, $\mathbf{A}$ induces the Mahalanobis norm on $\mathbb{R}^n$.

The norm influences the clustering criterion by changing the measure of dissimilarity. The Euclidean norm induces hyperspherical clusters (surfaces of constant membership are hyperspheres). Both the diagonal and the Mahalanobis norm generate hyperellipsoidal clusters. With the diagonal norm, the axes of the hyperellipsoids are parallel to the coordinate axes, while with the Mahalanobis norm the orientation of the hyperellipsoid is arbitrary.

A common limitation of clustering algorithms based on a fixed distance norm is that such a norm forces the objective function to prefer clusters of a certain shape even if they are not present in the data and inclusion of a pre-defined volume per cluster (ρ_i). Without any prior knowledge, it is simply fixed at 1 for each cluster. A drawback is that in that case the GK algorithm is only able to find clusters of approximately equal volumes. (Draft answer)

---

3. Name two fuzzy clustering algorithms and explain how they differ from each other.

Answer: Fuzzy C-means clustering and Gustafson-Kessel algorithm.

From the membership level curves, the FCM algorithm imposes a circular shape. If the clusters have different shape, it is of no help to use another fixed $\mathbf{A}$. Generally, different matrices $\mathbf{A}_i$ are required to capture the different shapes of the clusters, but usually there is no guideline as to how to choose them a priori. In Gustafson-Kessel algorithm, these matrices can be adapted by using estimates of the data covariance. (Draft answer).

---

4. State the fuzzy c-mean functional and explain all symbols.

Answer: The stationary points of the objective function in the FCM can be found by adjoining the constraints to J by means of Lagrange multipliers:

$$\bar{J}(\mathbf{Z}; \mathbf{U}, \mathbf{V}, \boldsymbol{\lambda}) = \sum_{i=1}^{c} \sum_{k=1}^{N} (\mu_{ik})^m D_{ik\mathbf{A}}^2 + \sum_{k=1}^{N} \lambda_k \left[\sum_{i=1}^{c} \mu_{ik} - 1 \right],$$

the number of clusters is c, the 'fuzziness' exponent is m, the cluster index is i, k is the sample index.

The data matrix $\mathbf{Z}$ contains all the samples (columns). The number of columns of $\mathbf{Z}$ is the number of samples N, and the number of rows corresponds to the dimensionality of a single sample.

The fuzzy partition matrix $\mathbf{U}$ contains the membership degrees to each fuzzy subset (row) of each sample (column), μ_{ik} is the membership degrees of sample k to cluster i.

The prototype matrix $\mathbf{V}$ contains a parametrization for each cluster. A cluster is defined by its center (mean of the cluster's samples). It is represented in the same space as a the samples in $\mathbf{Z}$ (dimension n).

The Lagrange multiplier λ_k ensures that the constraint (between square brackets) on the sum of the memberships is fulfilled (for each sample) when minimizing J.

Finally, $D_{ik\mathbf{A}}^2$ is the squared inner-product distance norm (based on the cluster prototypes $\mathbf{v}_i$, observations $\mathbf{z}_k$, and norm-inducing matrix $\mathbf{A}$) such as the Euclidean or Mahalonobis distance e.g., defined as

$$D^2_{ik\mathbf{A}} = \|\mathbf{z}_k - \mathbf{v}_i\|^2_{\mathbf{A}} = (\mathbf{z}_k - \mathbf{v}_i)^T \mathbf{A} (\mathbf{z}_k - \mathbf{v}_i).$$

(Draft answer).

---

5. Outline the steps required in the initialization and execution of the fuzzy c-means algorithm. What is the role and the effect of the user-defined parameters in this algorithm?

Answer: Initialization:

Before using the FCM algorithm, the following parameters must be specified: the number of clusters, c, the 'fuzziness' exponent, m, the termination tolerance, ϵ, and the norm-inducing matrix, $\mathbf{A}$. Moreover, the fuzzy partition matrix, $\mathbf{U}$, must be initialized. The choices for these parameters are now described one by one.

The number of clusters c is the most important parameter, in the sense that the remaining parameters have less influence on the resulting partition. When clustering real data without any a priori information about the structures in the data, one usually has to make assumptions about the number of underlying clusters.

The weighting exponent m is a rather important parameter as well, because it significantly influences the fuzziness of the resulting partition. As m approaches one from above, the partition becomes hard ($\mu_{ik} \in \{0, 1\}$) and $\mathbf{v}_i$ are ordinary means of the clusters. As $m \to \infty$, the partition becomes completely fuzzy ($\mu_{ik} = 1/c$) and the cluster means are all equal to the mean of $\mathbf{Z}$. These limit properties are independent of the optimization method used. Usually, $m = 2$ is initially chosen.

The FCM algorithm stops iterating when the norm of the difference between $\mathbf{U}$ in two successive iterations is smaller than the termination parameter ϵ. For the maximum norm $\max_{ik}(|\mu^{(l)}_{ik} - \mu^{(l-1)}_{ik}|)$, the usual choice is $\epsilon = 0.001$, even though $\epsilon = 0.01$ works well in most cases, while drastically reducing the computing times.

The shape of the clusters is determined by the choice of the matrix $\mathbf{A}$ in the distance measure. A common choice is $\mathbf{A} = \mathbf{I}$, which gives the standard Euclidean norm.

Execution: Algorithm 4.1

(Draft answer)

Algorithm 4.1 Fuzzy c-means (FCM).

Given the data set $\mathbf{Z}$, choose the number of clusters $1 < c < N$, the weighting exponent $m > 1$, the termination tolerance $\epsilon > 0$ and the norm-inducing matrix $\mathbf{A}$. Initialize the partition matrix randomly, such that $\mathbf{U}^{(0)} \in M_{fc}$.

Repeat for $l = 1, 2, \ldots$

Step 1: Compute the cluster prototypes (means):

$$\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m \mathbf{z}_k}{\sum_{k=1}^{N} \left(\mu_{ik}^{(l-1)}\right)^m}, \quad 1 \le i \le c\,.$$

Step 2: Compute the distances:

$$D_{ik\mathbf{A}}^2 = (\mathbf{z}_k - \mathbf{v}_i^{(l)})^T \mathbf{A} (\mathbf{z}_k - \mathbf{v}_i^{(l)}), \quad 1 \le i \le c, \quad 1 \le k \le N\,.$$

Step 3: Update the partition matrix:
for $1 \le k \le N$
if $D_{ik\mathbf{A}} > 0$ for all $i = 1, 2, \ldots, c$

$$\mu_{ik}^{(l)} = \frac{1}{\sum_{j=1}^{c} (D_{ik\mathbf{A}}/D_{jk\mathbf{A}})^{2/(m-1)}},$$

otherwise

$$\mu_{ik}^{(l)} = 0 \text{ if } D_{ik\mathbf{A}} > 0, \quad \text{and} \quad \mu_{ik}^{(l)} \in [0, 1] \text{ with } \sum_{i=1}^{c} \mu_{ik}^{(l)} = 1\,.$$

until $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \epsilon$.

---

5 Construction Techniques for Fuzzy Systems

---

1. Explain the steps one should follow when designing a knowledge-based fuzzy model. One of the strengths of fuzzy systems is their ability to integrate prior knowledge and data. Explain how this can be done.

   Answer: In order to design a knowledge-based fuzzy model, one must:

   - Select the input and output variables, the structure of the rules, and the inference and defuzzification methods.
   - Decide on the number of linguistic terms for each variable and define the corresponding membership functions.
   - Formulate the available knowledge in terms of fuzzy if-then rules.
   - Validate the model. If the model does not meet the expected performance, iterate on the above design steps.

   Integrating of prior knowledge brings together the partly knowledge (data driven) and partly unknown parts. Including the prior knowledge could be done by means of the least-squares estimation of consequents or of template-based modeling. (Draft answer).

---

2. Consider a singleton fuzzy model $y = f(x)$ with the following two rules:

$$1)\ \textbf{If } x \text{ is } \textit{Small} \textbf{ then } y = b_1, \quad 2)\ \textbf{If } x \text{ is } \textit{Large} \textbf{ then } y = b_2\,.$$

   and membership functions as given in Figure 5.1.

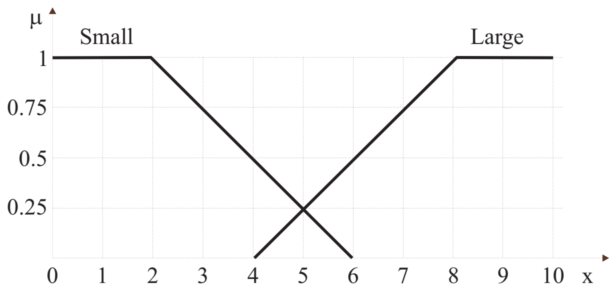


Figure 5.1: Membership functions.

   Furthermore, the following data set is given:

$x_1 = 1, \qquad y_1 = 3$
$x_2 = 5, \qquad y_2 = 4.5$

Compute the consequent parameters b_1 and b_2 such that the model gives the least summed squared error on the above data. What is the value of this summed squared error?

Answer: The degrees of fulfillment are:

$$\beta_1(x_1) = 1, \qquad \beta_2(x_1) = 0 \qquad \beta_1(x_2) = 0.25, \qquad \beta_2(x_2) = 0.25$$

For each sample in the data set, we can write one equation with two unknown parameters:

$$y_1 = \frac{\beta_1(x_1) \cdot b_1 + \beta_2(x_1) \cdot b_2}{\beta_1(x_1) + \beta_2(x_1)}, \quad y_2 = \frac{\beta_1(x_2) \cdot b_1 + \beta_2(x_2) \cdot b_2}{\beta_1(x_2) + \beta_2(x_2)}$$

After filling the data in, we get: $b_1 = 3$ and $b_2 = 6$.

$$\begin{aligned} 3 &= \frac{1b_1 + 0b_2}{1+0} = \frac{b_1}{1} \\ b_1 &= 3 \end{aligned}$$

$$\begin{aligned} 4.5 &= \frac{0.25b_1 + 0.25b_2}{0.25 + 0.25} = \frac{0.25 \cdot 3 + 0.25b_2}{0.5} = \frac{0.75 + 0.25b_2}{0.5} \\ 2.25 &= 0.75 + 0.25b_2 \\ 1.5 &= 0.25b_2 \\ b_2 &= 6 \end{aligned}$$

Obviously, the summed squared error is zero (two parameters, two independent data samples available).

---

3. Consider the following fuzzy rules with singleton consequents:

1) **If** x is A_1 **and** y is B_1 **then** $z = c_1$, 3) **If** x is A_1 **and** y is B_2 **then** $z = c_3$,
2) **If** x is A_2 **and** y is B_1 **then** $z = c_2$, 4) **If** x is A_2 **and** y is B_2 **then** $z = c_4$.

Draw a scheme of the corresponding neuro-fuzzy network. What are the free (adjustable parameters in this network? What methods can be used to optimize these parameters by using input–output data?

Answer: The nodes in the first layer compute the membership degree of the inputs in the antecedent fuzzy sets. The product nodes Π in the second layer represent the antecedent conjunction operator. The normalization node N and the summation node Σ realize the fuzzy-mean operator

$$z = \frac{\sum_{i=1}^{K} \beta_i c_i}{\sum_{i=1}^{K} \beta_i}.$$

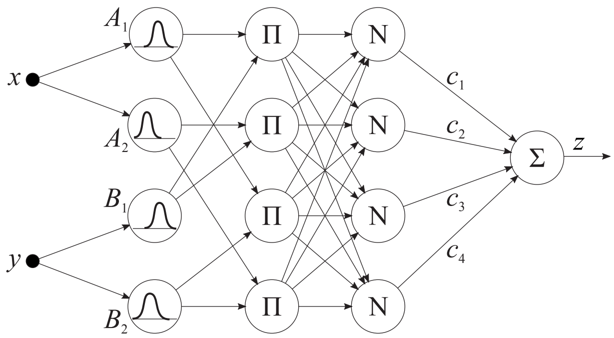


Figure 5.2: A neuro-fuzzy network with four rules.

By using smooth (e.g., Gaussian) antecedent membership functions

$$\mu_{A_{ij}}(x_j; c_{ij}, \sigma_{ij}) = \exp\left(-\left(\frac{x_j - c_{ij}}{2\sigma_{ij}}\right)^2\right),$$

the c_{ij} and σ_{ij} parameters can be adjusted by gradient-descent learning algorithms, such as back-propagation.

(Draft answer)

---

4. Give a general equation for a NARX (Nonlinear AutoRegressive with eXogenous input) model. Explain all symbols. Give an example of a some NARX model of your choice.

Answer: The general equation for a NARX model is defined as:

$$y(k+1) = f(y(k), y(k-1), \ldots, y(k-n_y+1), u(k), u(k-1), \ldots, u(k-n_u+1)).$$

Here $y(k), \ldots, y(k-n_y+1)$ and $u(k), \ldots, u(k-n_u+1)$ denote the past model outputs and inputs respectively and n_y, n_u are integers related to the order of the dynamic system. For example, a singleton fuzzy model of a dynamic system may consist of rules of the following form:

$$\begin{aligned}\mathcal{R}_i: \quad & \textbf{If } y(k) \text{ is } A_{i1} \textbf{ and } y(k-1) \text{ is } A_{i2} \textbf{ and} \ldots y(k-n+1) \text{ is } A_{in} \\ & \textbf{and } u(k) \text{ is } B_{i1} \textbf{ and } u(k-1) \text{ is } B_{i2} \textbf{ and} \ldots u(k-m+1) \text{ is } B_{im} \\ & \textbf{then } y(k+1) \text{ is } c_i\,.\end{aligned}$$

---

5. Explain the term semi-mechanistic (hybrid) modeling. What do you understand under the terms “structure selection” and “parameter estimation” in case of such a model?

Answer: Semi-mechanistic modeling is a type of modeling that uses a priori knowledge of the system (or the working of its parts). With information available about a system, the blackbox that encompasses the nonlinearities and unknowns will get less complex. The main advantage of semi-mechanistic modeling is that there will be no time wasted on estimation of the parts of the system that are already known.

Structure Selection: The structure determines the flexibility of the model in the approximation of (unknown) mappings. Normally, a model with a rich structure approximates complicated functions well, but is not very robust (i.e. it will not perform well on another data set from the same structure). Now, if there is some prior knowledge of the internal system, there will be less unknown data, which means the unknown mapping can be approximated better. Thus, the use of semi-mechanistic modeling will keep the richness of the structure while enhancing the generalization properties.

Parameter Estimation: With the training data, the parameters are estimated and tuned. The available knowledge will increase the granularity of the model. This is logical, if the entire system would be known, the detail would be infinitely high. The more you know, the more detailed your model will be. In short, there is a better parameter estimation when using semi-mechanistic modeling compared to normal modeling.

(Draft answer)

6 Knowledge-Based Fuzzy Control

---

1. There are various ways to parameterize nonlinear models and controllers. Name at least three different parameterizations and explain how they differ from each other.

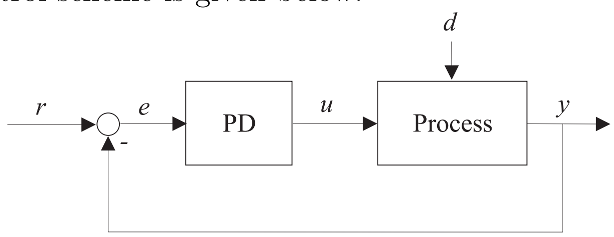
   Answer:

   - For instance fuzzy systems, multi-layer neural networks, radial basis function networks, polynomials, look-up tables.
   - The differ in their approximation power (number of parameters versus approximation accuracy), the way they are constructed (purely from data, based on prior knowledge, combination of data and prior knowledge), memory and CPU time requirements, etc.

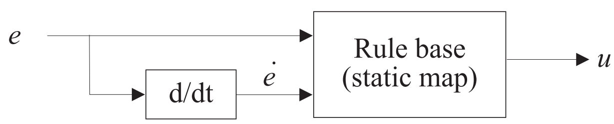
---

2. Draw a control scheme with a fuzzy PD (proportional-derivative) controller, including the process. Explain the internal structure of the fuzzy PD controller, including the dynamic filter(s), rule base, etc.

   Answer: a) The control scheme is given below:



   b) The internal structure of the controller is:



   c) This is an example of a possible rule base:

		$\dot{e}$ NB	$\dot{e}$ NS	$\dot{e}$ ZE	$\dot{e}$ PS	$\dot{e}$ PB
	NB	NB	NB	NS	NS	ZE
	NS	NB	NS	NS	ZE	PS
e	ZE	NS	NS	ZE	PS	PS
	PS	NS	ZE	PS	PS	PB
	PB	ZE	PS	PS	PB	PB

d) The design parameters are the rules, the membership functions and possibly also scaling factors in the inputs and in the output of the static map given by the inference system.

---

3. Give an example of a rule base and the corresponding membership functions for a fuzzy PI (proportional-integral) controller. What are the design parameters of this controller and how can you determine them?

Answer: An example of a rule base for a fuzzy PI controller is:

$$\begin{aligned} \mathcal{R}_1 &: \quad \textbf{If } e \text{ is } \textit{Low} \textbf{ then } P = P_{\text{Low}} \textbf{ and } I = I_{\text{Low}} \\ \mathcal{R}_2 &: \quad \textbf{If } e \text{ is } \textit{High} \textbf{ then } P = P_{\text{High}} \textbf{ and } I = I_{\text{High}} \end{aligned}$$

or in the Takagi–Sugeno fuzzy way

$$\begin{aligned} \mathcal{R}_1 &: \quad \textbf{If } e \text{ is } \textit{Low} \textbf{ then } u_1 = P_{\text{Low}} e + I_{\text{Low}} \int e \, \mathrm{d}t \\ \mathcal{R}_2 &: \quad \textbf{If } e \text{ is } \textit{High} \textbf{ then } u_2 = P_{\text{High}} e + I_{\text{High}} \int e \, \mathrm{d}t \end{aligned}$$

Possible membership functions can be seen in Figure 6.1.

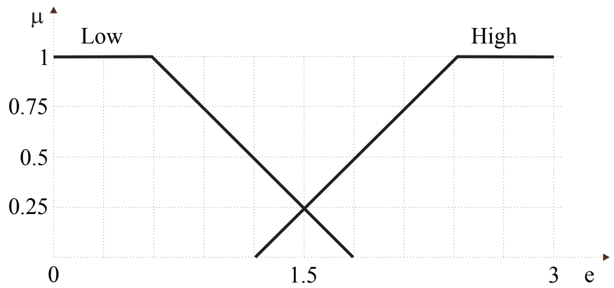


Figure 6.1: Membership functions *Low* and *High*

The design parameters are the membership functions *Low* and *High* and the consequent parameters P_{Low}, I_{Low} and P_{High}, I_{High}. They can be determined by linearizing the process at different operating points and use a Bode-, Nyquist- or root locus plot to tune the PI controller at the different operating points. The control values found during tuning have to be incorporated in the membership functions and consequent parameters. Fine tuning of these functions and parameters will in the end give an optimal solution.

(Draft answer)

---

4. State in your own words a definition of a fuzzy controller. How do fuzzy controllers differ from linear controllers, such as PID or state-feedback control? For what kinds of processes have fuzzy controllers the potential of providing better performance than linear controllers?

Answer:

a) A *fuzzy controller* is a controller that contains a (nonlinear) mapping that has been defined by using fuzzy if-then rules. Fuzzy controllers include if-then logic as to enable the system to be controlled in a smooth way.

b) Fuzzy controllers are nonlinear and are typically designed by using knowledge in the form of if–then rules. Conventional controllers are linear and are usually designed by means of mathematical methods (pole placement, root locus, for instance). Fuzzy controllers enable the inclusion of expert knowledge and are therefore very suitable as supervisory controller/in case of nonlinearities, changing environment, etc.

c) Fuzzy controllers have the potential of providing better performance than conventional controllers for nonlinear processes or when the controller is tuned by a non-expert in control. Linear has mathematical methods (root locus, pole placement).

---

5. a) Give an example of several rules of a Takagi–Sugeno fuzzy controller. b) What are the design parameters of this controller? c) Give an example of a process to which you would apply this controller.

Answer:

a) An example of a rule base is:

$$\begin{aligned}
\mathcal{R}_1 \quad &: \quad \textbf{If } r \text{ is } \textit{Low} \textbf{ then } u_1 = P_{\text{Low}} e + I_{\text{Low}} \int e \, \mathrm{d}t \\
\mathcal{R}_2 \quad &: \quad \textbf{If } r \text{ is } \textit{High} \textbf{ then } u_2 = P_{\text{High}} e + I_{\text{High}} \int e \, \mathrm{d}t
\end{aligned}$$

b) The design parameters are the membership functions *Low* and *High* and the consequent parameters P_{Low}, I_{Low} and P_{High}, I_{High}.

c) This controller would typically be applied to a process whose parameters change with the operating point (reference).

6. Is special fuzzy-logic hardware always needed to implement a fuzzy controller? Explain your answer.

   Answer: No, special hardware is not necessarily needed. One can use an available software package and generate $\mathcal{C}$ or machine code that can be run on a variety of general-purpose micro-controllers, PCs, etc.

7 Artificial Neural Networks

1. What has been the original motivation behind artificial neural networks? Give at least two examples of control engineering applications of artificial neural networks.

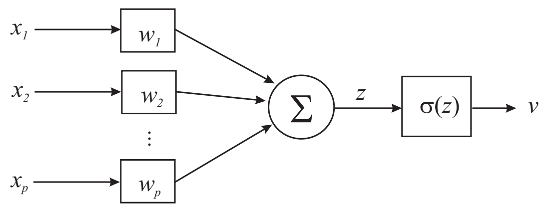
   Answer: The motivation has been to imitate biological neural networks, namely their learning and adaptation capabilities and parallel information processing. To imitate human learning, one: where no prior knowledge is present (new environment) and two: to overcome complex problems (pattern recognition and image processing).

   From a mathematical point of view, artificial neural nets are effective function approximators. As such, they are being used in a variety of engineering applications, including systems identification, adaptive control, or classification.

---

2. Draw a block diagram and give the formulas for an artificial neuron. Explain all terms and symbols.

   Answer: An artificial neuron is schematically depicted in the figure below:



   The inputs x_i, $i = 1, 2, \ldots, p$ are first multiplied by weights w_i and summed to obtain the activation z:

$$z = \sum_{i=1}^{p} w_i x_i = \mathbf{w}^T \mathbf{x}\,.$$

   The output v of the neuron then obtained by passing z through the activation function σ, which can be, for instance, the tangent hyperbolic function:

$$v = \sigma(z) = \frac{1 - \exp(-2z)}{1 + \exp(-2z)}$$

---

3. Give at least three examples of activation functions.

   Answer: Examples of activation functions are: threshold function (hard limiter), saturated linear function, sigmoidal function, tangent hyperbolic, etc. An important property of the activation function is that it maps the activation $z \in (-\infty, \infty)$ onto the interval $[0, 1]$ or $[-1, 1]$. Therefore, it is also called a squeezing function.

4. Explain the term "training" of a neural network.

Answer: Training adjusts weights as to minimize an error cost function. Training is the adaptation of weights in a network such that a certain goal is achieved. In a multi-layer neural networks, for instance, the goal is to minimize the error between the desired output and the actual network output. By training, the network learns to approximate a certain function from examples of input–output data pairs.

5. What are the steps of the backpropagation algorithm? With what neural network architecture is this algorithm used?

Answer: The backpropagation training proceeds in two steps:

a) Feedforward computation: given the network inputs x_i, the network outputs are computed, proceeding through the hidden layer(s).

b) Weight adaptation: the output of the network is compared to the desired output. The difference of these two values, the error e, is used to adjust the weights in the network. The weights are adjusted such that the following sum-squared error cost function decreases:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{j=1}^{l} e_j^2$$

where l is the number of output neurons and $\mathbf{w}$ are the weights. The weight update rule is derived from the general gradient-descent formula:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \alpha(n)\nabla J(\mathbf{w}(n)),$$

where $\mathbf{w}(n)$ is the weights vector at iteration n, $\alpha(n)$ is a (variable) learning rate and $\nabla J(\mathbf{w})$ is the Jacobian:

$$\nabla J(\mathbf{w}) = \left[\frac{\partial J(\mathbf{w})}{\partial w_1}, \frac{\partial J(\mathbf{w})}{\partial w_2}, \ldots, \frac{\partial J(\mathbf{w})}{\partial w_M}\right]^T .$$

The backpropagation algorithm is used with a multi-layer neural network. The algorithm uses the first-order gradient to adapt the weight. There is no feedback involved in the neural network, so backpropagation algorithms are used in feedforward networks.

6. Explain the difference between first-order and second-order gradient optimization.

Answer: First-order methods use the first term of the Taylor series expansion of the objective function (the gradient or Jacobian). Second-order gradient methods use the second term (the curvature) as well:

$$J(\mathbf{w}) \approx J(\mathbf{w}_0) + \nabla J(\mathbf{w}_0)^T(\mathbf{w}-\mathbf{w}_0) + \frac{1}{2}(\mathbf{w}-\mathbf{w}_0)^T\mathbf{H}(\mathbf{w}_0)(\mathbf{w}-\mathbf{w}_0),$$

The matrix $\mathbf{H}(\mathbf{w}_0)$ is called the Hessian and it's inverse appears in the weight update formula as the step size of the gradient descent. Therefore, second-order methods generally require fewer iterations (take larger steps on flat cost function surfaces).

---

7. Derive the backpropagation rule for an output neuron with a sigmoidal activation function.

Answer: Consider the output neuron

$$y = \sigma(z) \quad \text{with} \quad z = \sum_{j=1}^{p} w_j v_j$$

and the cost function (for one data point):

$$J = \frac{1}{2}e^2, \quad \text{with } e = d - y$$

The Jacobian is:

$$\frac{\partial J}{\partial w_j} = \frac{\partial J}{\partial e} \cdot \frac{\partial e}{\partial y} \cdot \frac{\partial y}{\partial z} \cdot \frac{\partial z}{\partial w_j}$$

computing the partial derivatives:

$$\frac{\partial J}{\partial e} = e, \quad \frac{\partial e}{\partial y} = -1, \quad \frac{\partial y}{\partial z} = \sigma(z)', \quad \frac{\partial z}{\partial w_j} = v_j$$

we obtain:

$$\frac{\partial J}{\partial w^o_{jl}} = -v_j e \sigma(z)' \, .$$

The backpropagation update law for the output weights directly follows:

$$w_j(n+1) = w_j(n) + \alpha(n) v_j e \sigma(z)',$$

where $\alpha(n)$ is the learning rate (step size).

---

8. What are the differences between a multi-layer feedforward neural network and the radial basis function network?

Answer: The differences are: the activation function (radial basis function vs. a squeezing function), in an RBF network, the only adjustable weights are the output ones, the input weights are unity (the spreads and centers of the basis functions can be adjusted instead), an RBF network only has one layer of neurons.

---

9. Consider a dynamic system $y(k+1) = f\big(y(k), y(k-1), u(k), u(k-1)\big)$, where the function f is unknown. Suppose, we wish to approximate f by a neural network trained by using a sequence of N input–output data samples measured on the unknown system, $\{(u(k), y(k)) | k = 0, 1, \ldots, N\}$.

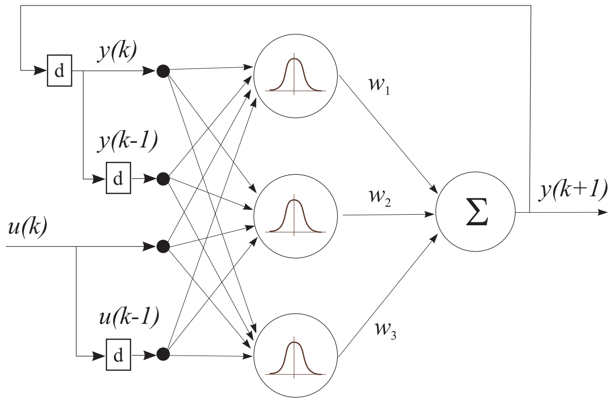
a) Choose a neural network architecture, draw a scheme of the network and define its inputs and outputs.

b) What are the free parameters that must be trained (optimized) such that the network fits the data well?

c) Define a cost function for the training (by a formula) and name examples of two methods you could use to train the network's parameters.

Answer:

a) One possibility is to choose a radial basis function network:



Another possibility would be a multilayer neural net. In the above scheme, the 'd' block denotes the delay operator. The output of the network is the predicted output of the dynamic system $y(k+1)$. The inputs of the network are $y(k)$, $y(k-1)$ (obtained by delaying the output) and $u(k)$, $u(k-1)$, where $u(k)$ is the given input to the dynamic system and $u(k-1)$ is the one step delayed value of this input.

b) The weights w_i and the parameters of the radial basis functions (centers and widths).

c) The sum of squared errors $J = \sum_{i=1}^{N}\left[y_d(k+1) - y(k+1)\right]^2$, where $y_d(k+1)$ are the measured data and $y(k+1)$ is the network's output. For the output weights, one can use the least squares method and for the parameters of the basis functions a first-order or second-order gradient-descent method (back-propagation or Levenberg-Marquardt, for instance). Fuzzy clustering techniques can be used to initialize the centers and spreads.

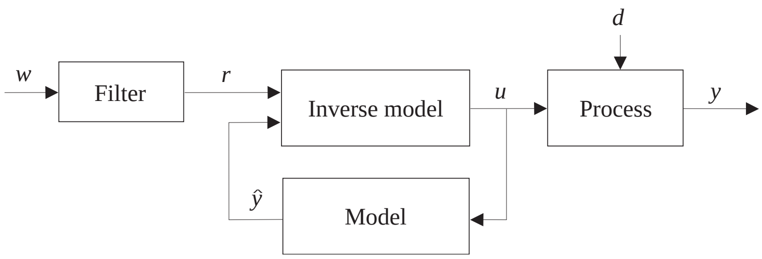
8 Control Based on Fuzzy and Neural Models

1. Draw a general scheme of a feedforward control scheme where the controller is based on an inverse model of the dynamic process. Describe the blocks and signals in the scheme.

   Answer: The objective of inverse control is to compute for the current state the control input, such that the system's output at the next sampling instant is equal to the desired (reference) output. This can be achieved if the process model can be inverted according to:

   $$u(k) = f^{-1}(\mathbf{x}(k), r(k+1))$$

   where the state $\mathbf{x}(k) = [\hat{y}(k), \dots, u(k-1), \dots]^T$ can be updated by using the output of the model as shown in the figure below.



   The reference-shaping filter generates the desired dynamics in order to avoid peaks in the control action for step-wise references.

2. Consider a first-order affine Takagi–Sugeno model:

   $$R_i \qquad \textbf{If } y(k) \text{ is } A_i \textbf{ then } y(k+1) = a_i y(k) + b_i u(k) + c_i$$

   Derive the formula for the controller based on the inverse of this model, i.e., $u(k) = f(r(k+1), y(k))$, where r is the reference to be followed.

   Answer: Denote $\lambda_i(y(k))$ is the normalized degree of fulfillment of the rules:

   $$\lambda_i(y(k)) = \frac{\mu_{A_i}(y(k))}{\sum_{j=1}^{K} \mu_{A_j}(y(k))} .$$

   The output of the model is:

   $$y(k+1) = \left(\sum_{i=1}^{K} \lambda_i(y(k)) a_i \right) y(k) + \left(\sum_{i=1}^{K} \lambda_i(y(k)) b_i \right) u(k) + \sum_{i=1}^{K} \lambda_i(y(k)) c_i .$$

After substituting $r(k+1)$ for $y(k+1)$

$$
\begin{aligned}
y(k+1) &= a_i y(k) + b_i u(k) + c_i \\
r(k+1) &= a_i y(k) + b_i u(k) + c_i \\
b_i u(k) &= r(k+1) - a_i(y) - c_i \\
u(k) &= \frac{r(k+1) - a_i(y) - c_i}{b_i}
\end{aligned}
$$

the inverse model directly follows:

$$
u(k) = \frac{r(k+1) - \sum_{i=1}^{K} \lambda_i(y(k)) a_i y(k) - \sum_{i=1}^{K} \lambda_i(y(k)) c_i}{\sum_{i=1}^{K} \lambda_i(y(k)) b_i} .
$$

---

3. Explain the concept of predictive control. Give a formula for a typical cost function and explain all the symbols.

   Answer: In predictive control a model is used to predict the process output at future discrete time instants over some prediction horizon. The sequence of future control actions is computed by minimizing an objective function which penalized the difference between the predicted output and the desired output (the reference). Only the first control action of the obtained sequence is applied to the process, the horizon is moved one step forward and the optimization is repeated. Usually, the following quadratic cost function is applied:

$$
J = \sum_{i=1}^{H_p} \|(\mathbf{r}(k+i) - \hat{\mathbf{y}}(k+i))\|^2_{\mathbf{P}_i} + \sum_{i=1}^{H_c} \|(\Delta \mathbf{u}(k+i-1))\|^2_{\mathbf{Q}_i} .
$$

   The first term penalizes the variations of the process output around the reference, while the second is a penalty on the control effort. $\mathbf{P}_i$ and $\mathbf{Q}_i$ are positive definite weighting matrices that specify the relative importance of two terms.

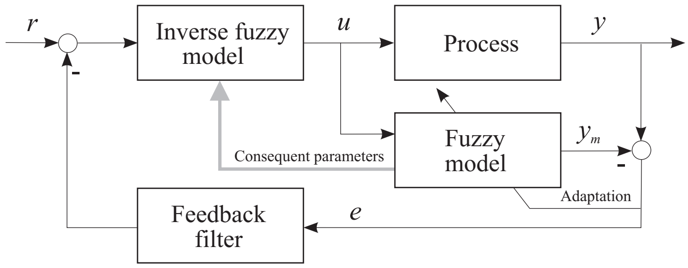
---

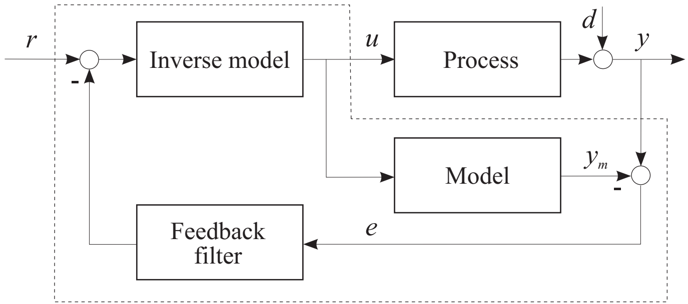
4. What is the principle of indirect adaptive control? Draw a block diagram of a typical indirect control scheme and explain the functions of all the blocks.

   Answer: In indirect adaptive control, a process model is adapted on-line and the controller's parameters are derived from the parameters of the model. The figure below depicts a control scheme in which the consequent parameters of a fuzzy model are adapted on-line (by means of recursive least squares). The controller is obtained by inverting the fuzzy model.

---

5. Explain the idea of internal model control (IMC).





Answer: The IMC scheme is depicted in the figure below. It consists of a controller, which is the inverse of the process model, the model itself, and a feedback filter.

The purpose of the process model working in parallel with the process is to subtract the effect of the control action u from the process output. If the predicted output y_m and the measured output y are equal, the error e is zero and the controller works in an open-loop configuration. If a disturbance d acts on the process output, the feedback signal e is equal to that disturbance and is not affected by the effects of the control action. This signal is subtracted from the reference r. With a perfect process model, the IMC scheme is hence able to cancel the effect of unmeasured output-additive disturbances. The feedback filter is introduced in order to filter out the measurement noise.

9 Reinforcement Learning

1. Prove mathematically that the discounted sum of immediate rewards

$$R_k = \sum_{n=0}^{\infty} \gamma^n r_{k+n+1}$$

is indeed finite when $0 \leq \gamma < 1$. Assume that the agent always receives an immediate reward of -1. Compute R_k when $\gamma = 0.2$ and $\gamma = 0.95$. Compare your results and give a brief interpretation of this difference.

Answer: The limit is computed using the sum of a geometric series:

$$R_k = \sum_{n=0}^{\infty} \gamma^n r_{k+n+1} = r\frac{1}{1-\gamma}$$

This sum is finite for $|\gamma| < 1$, so it is finite as well for $0 \leq \gamma < 1$. With $r = -1$, we find that for $\gamma = 0.2$, $R_k = -1.25$ and for $\gamma = 0.95$, $R_k = -20$. The agent thus expects a much larger negative reward when γ is closer to 1. One can say that the agent is much more concerned with the long-term rewards when the discount rate is close to 1.

2. Explain the difference between the V-value function and the Q-value function and give an advantage of both.

Answer: The V-value function and Q-value function are defined for following a policy π as follows:

$$\begin{aligned} V^\pi(s) &= E_\pi\{R_k | s_k = s\} \\ Q^\pi(s,a) &= E_\pi\{R_k | s_k = s, a_k = a\} \end{aligned}$$

Note that a policy is a mapping from states to actions, $\pi(S) \rightarrow A$, with S the state space and A the action space. The greedy policy π can easily be derived from the Q-value function by

$$\pi(s) = \arg\max_a Q(s,a),$$

because it stores values associated with state-action pairs. A V-value function stores only values associated with states. The advantage of V-value functions is thus that it consumes much less memory than Q-value functions. On the other hand, to derive the greedy policy from a V-value function is much more tedious than it is for Q-value functions. The greedy policy is derived by first rewriting it into a Q-value function:

$$Q(s,a) = E\{r_{k+1} + \gamma V(s_{k+1})|s_k = s, a_k = a\}$$

and then applying the equation of the greedy policy π derived from the Q-value function. For practical applications this comes down to directly using Q-value functions.

---

3. Explain the difference between on-policy and off-policy learning.

   Answer: In on-policy learning, the update of the parameters (V-, or Q-values) is done by taking into account the actual action that is applied to the process. In off-policy this is not necessarily the case. E.g. in Q-learning, the update is performed by considering the greedy action, which is due to exploration not necessarily the same as the action that is applied to the process.

---

4. Is actor-critic reinforcement learning an on-policy or off-policy method? Explain you answer.

   Answer: The update rule for the actor is given as

$$\begin{aligned}\varphi_{k+1} &= \varphi_k + \alpha_a \Delta_k [a_k - a_k^\pi] \frac{\partial \hat{\pi}(s_k, \varphi_k)}{\partial \varphi_k} \\ &= \varphi_k + \alpha_a \Delta_k \tilde{a}_k \frac{\partial \hat{\pi}(s_k, \varphi_k)}{\partial \varphi_k},\end{aligned}$$

   where φ_k is a vector of adjustable parameters, α_a the learning rate, Δ_k the temporal difference error, a_k^π the action according to the current policy π, and a_k the applied control action including exploration $\tilde{a}_k$. The update rule thus includes the applied control action and can thus be considered to be an on-policy learning method.

---

5. Describe the tabular algorithm for $Q(\lambda)$-learning, using replacing traces to update the eligibility trace.

   Answer: The algorithm is as follows:

   Algorithm 9.1 *$Q(\lambda)$-learning*

a) Initialize $Q(s,a) = 0$ and $e(s,a) = 0$, for all s, a.

b) Repeat (for each episode):

   i. Initialize s, a.

   ii. Repeat (for each step of the episode):

      Apply action a to the process. Observe the immediate reward r and the new state s'.

      Determine the next action a^* according to the policy by

$$a^* \leftarrow \arg\max_b Q(s', b) \text{ (break ties randomly)}$$

      Modify the action according to your exploration strategy to a'.

      Compute:

$$\begin{aligned} \delta &\leftarrow r + \gamma Q(s', a^*) - Q(s,a) \\ e(s,a) &\leftarrow 1 \end{aligned}$$

      For all s, a:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \delta e(s,a)$$

      If $a' = a^*$ then $e(s,a) \leftarrow \gamma\lambda e(s,a)$ else $e(s,a) \leftarrow 0$.

      $s \leftarrow s'$, $a \leftarrow a'$