Intelligent Systems Engineering

FS-7 Fuzzy Control

Petr Musilek

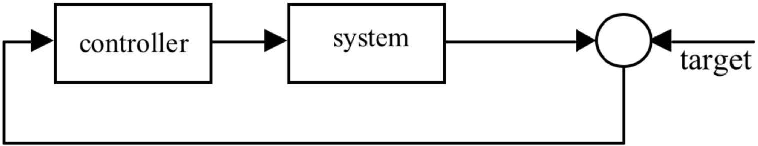
University of Alberta

Outline

These notes are based on [Pedrycz, 2007] and [Keller et al. 2016]

Control Problem

Most typical control scheme is that of *direct control*, where the controller is in the forward path of a feedback control system.



Due to the diversity of systems and control objectives, design of controller is usually nontrivial and includes

(i) identification of the system, i.e. building of mathematical model of system behaviour and its parameters, and
(ii) analytical description of control objectives.

Reality

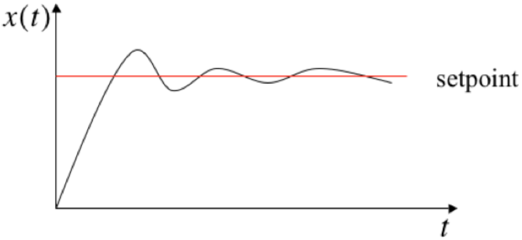
In real situations, mathematical model of the system can be too difficult or too expensive to obtain. This is especially evident in systems that are normally controlled by human, for example: parking a car, driving a car, operating a crane, maintaining comfortable temperature, etc. These situations are characteristic by

- unavailability of mathematical model of the system, and
- no explicitly formulated performance index, but
- availability of experiential knowledge

This last fact (availability of experiential knowledge) is very important because it allows linguistic description of the control and controlled variables (in form of fuzzy sets) and linguistic description of the control strategies (in form of fuzzy rules).

Control problem – example

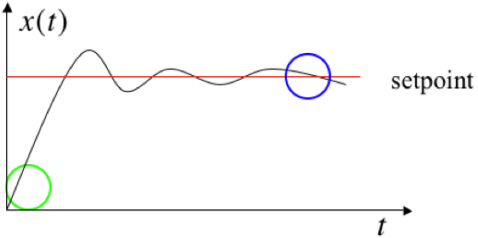
Consider the following simple example of control problem



The objective is to stabilize system very quickly. These requirements are conflicting: we can either approach the setpoint in a short time but with many large oscillations, or avoid oscillations in which case the approach will be very slow.

Control problem – example (cont.)

This problem can be eliminated by considering different strategies in different stages of control, as in the following picture:



Control problem – example (cont.)

By considering variables error = setpoint – x(t),
chage_of_error = error(t)- *error*(t-1), and control,

and defining fuzzy sets *positive large* (*PL*), *positive small* (*PS*), *zero* (*Z*), *negative small* (*NS*), *negative large* (*NL*) on these variables,

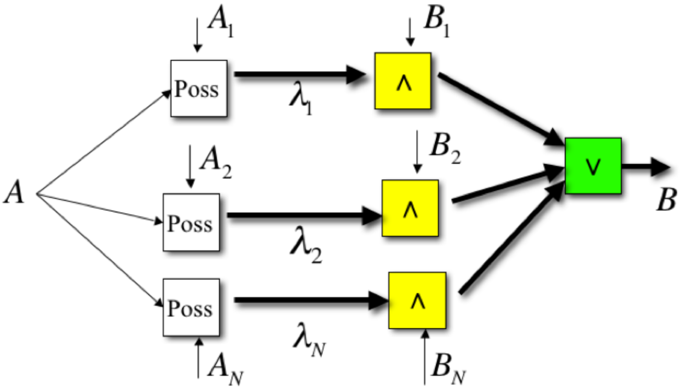
we can formulate the following set of rules:

Control problem – example (cont.)

For phase 1 (green circle)
IF *error* is *PL* and *change_of_error* is *NL* THEN *control is PL*
For phase 2 (blue circle)
IF *error* is *Z* and *change_of_error* is *Z* THEN *control is Z*
By constructing more rules/relations for additional phases/segments of the control process, a smooth control of the system can be achieved.

Fuzzy Controller

A general structure of fuzzy controller corresponds to the compositional rule of inference developed earlier.



Fuzzy Controller (cont.)

The first column of blocks (Poss) performs matching of fuzzy set A with antecedents A_k of individual rules R_k. The second column ($\wedge$) corresponds to min operation of the inner term,

$$\min\left[\lambda_k, B_k(y)\right].$$

The last block ($\vee$) finally aggregates the results of individual rules

$$B(y) = \max_k \min[\lambda_k, B_k(y)]$$

Fuzzy/crisp transformations

Operation of the fuzzy controller can be summarized as follows: given set of rules IF x is A_k THEN y *is* B_k, and fuzzy set A describing the state of the system, fuzzy controller applies CIR to determine control variable in form of fuzzy set B, or formally

$$\begin{array}{lllll} & x \text{ is } A & & \\ \text{IF} & x \text{ is } A_k & \text{THEN} & y \text{ is } B_k \\ \hline & & & y \text{ is } B \end{array}$$

Real systems do not operate with fuzzy sets $\rightarrow$ need to transform real (crisp) values to fuzzy sets to be processed by the fuzzy controller, and transforming the resulting fuzzy set into a real value that can be used to actually control the system.

These transformations (encoding and decoding)

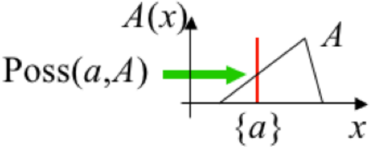
are called fuzzification and dufuzzification, respectively.

Fuzzification

Fuzzification

Encoding of crisp input value is already embedded in the existing structure of the fuzzy controller.

Recall that it is possible to apply possibility measure on a fuzzy singleton $\{a\}$ corresponding to the crisp value of input variable:



In such case, the result of antecedent matching is simply

$$\lambda_k = Poss(a, A_k) = A_k(a)$$

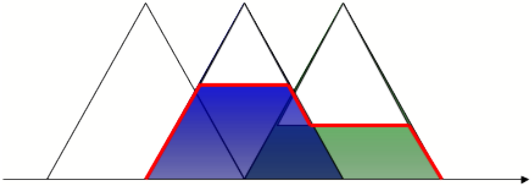
Defuzzification

Defuzzification

Due to the nature of fuzzy rules, its possible that more than one rule applies to a given input value *a*. As a result, the output fuzzy set can be of complicated shape composed of several consequent fuzzy sets B_k.

Defuzzification (cont.)

This situation is illustrated in the following figure for three consequent fuzzy sets B_1, B_2, and B_3, forming the output set B (indicated by red outline).



The goal of defuzzification is to provide a single value that best represents the output fuzzy set. It is important to consider not only shape/location of the fuzzy sets forming B, but also their height carrying information about relative significance of its corresponding fuzzy rule (i.e. how much is given rule activated – how well is its antecedent matching current input).

Defuzzification (cont.)

One possibility is to determine the center of the resulting area B, and project the location of this point to y axis, i.e.

$$b = \mathsf{COG}\left(B(y)\right) = \frac{\int B(y) y dy}{\int B(y) dy}$$

This approach is called Center of Gravity (COG), center of area, or *centroind*.

Defuzzification (cont.)

Other defuzzification methods include:

- SCOG (Simplified Center of Gravity) $b = \text{SCOG}\,(B(y)) = \frac{\sum B(y)y}{\sum B(y)}$
- BOA (Bisector of Area) $b = \text{BOA}\,(B(y)) : \int_{\alpha}^{\text{BOA}} B(y)dy = \int_{\text{BOA}}^{\beta} B(y)dy$, where α and β are, respectively, the left and right boundaries of the output fuzzy set
- MOM (Mean of Maximum) $b = \text{MOM}\,(B(y)) = \frac{\alpha^* + \beta^*}{2}$ where α^* and β^* are, respectively, the left and right boundaries of the maximum level of the output fuzzy set.
- SOM (Smallest of Maximum) $b = \text{SOM}\,(B(y)) = \alpha^*$
- LOM (Largest of Maximum) $b = \text{LOM}\,(B(y)) = \beta^*$

Defuzzification in python/scikit-fuzzy

In python/scikit-fuzzy, there is a function `defuzz`, which implements all basic defuzzification methods:

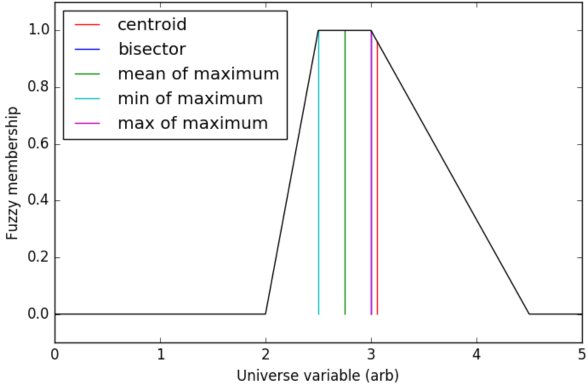
`b = defuzz(x,B,type)`, where type can be one of the following: `centroid`, `bisector`, `mom`, `som`, `lom`

Sample code

```
y = -10:0.1:10;
B = trapmf(y,[-10 -8 -4 7]);
b = defuzz(y, B, 'centroid');
```

Defuzzification in python/scikit-fuzzy (cont.)

```
defuzz = fuzz.defuzz(x, mfx, 'centroid')
{bisector, mom, som, lom}
```



Selection of defuzzification scheme

The choice of defuzzification scheme depends largely on the requirements for speed of processing and implementation constraints

- the centroid (COG, COA) method is most precise, but also most demanding
- using the sum-product composition, calculation of the centroid can be greatly simplified

$$b = \mathsf{COA}\left(B(y)\right) - \frac{\int B(y) y dy}{\int B(y) dy} = \frac{\sum \lambda_i a_i y_i}{\sum \lambda_i a_i}$$

where $a = \int B(y)dy$ and $y_i = \int B(y)ydy / \int B(y)dy$ are the area and centroid of the consequent membership function $B_i(y)$, resp. These values can be easily determined analytically for most standard membership functions.

Numerical characteristic of FLC

Numerical characteristic of FLC

A graph depicting the behavior of a FLC from the point of view of controlled system. It shows functional dependence between input and output variable(s) in their crisp (non-fuzzy) form.

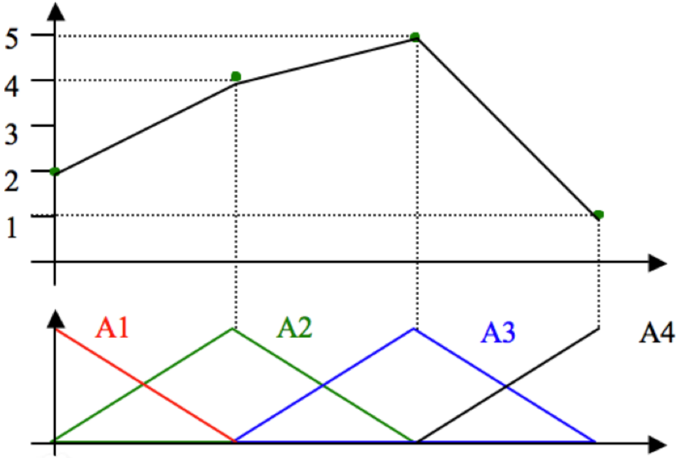
Consider a FLC with input fuzzy sets $A1\{x;0,0,1\}, A2\{x;0,1,2\}, A3\{x;1,2,3\}, A4\{x;2,3,3\}$, output fuzzy singletons $b1=\{1\}, b2=\{2\}, b3=\{4\}, b4=\{5\}$, and the following set of rules:

IF A_1 THEN b_2
IF A_2 THEN b_3
IF A_3 THEN b_4
IF A_4 THEN b_1

Numerical characteristic (cont.)

- Constructed by sweep of the values of input variable x over the universe $[0, 4]$, and determining numerical value of the output.
- Determining the activity level of individual rules λ_k for each value of x and defuzzification of the output set B obtained as the union of the output subsets B_k combined with their activities λ_k.
- In our case – this is simplified:
  - fuzzy singletons used in place of output fuzzy subsets
  - input space is partitioned evenly.

Numerical characteristic (cont.)



Numerical characteristic (cont.)

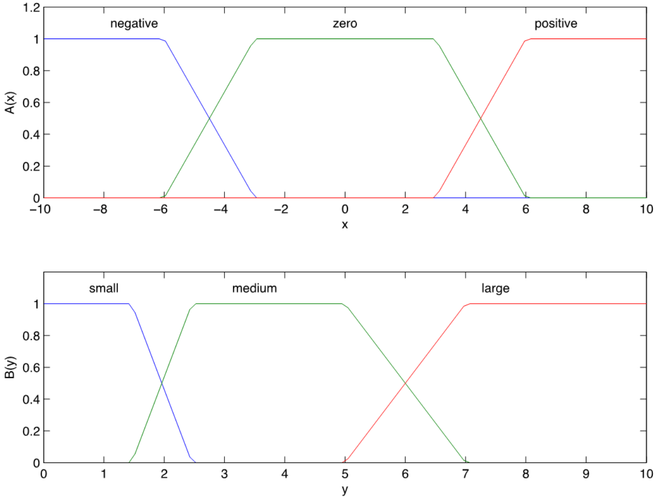
- As two subsequent input fuzzy sets compensate activities of their corresponding rules (i.e. as λ_k decreases λ_{k+1} increases totalling to $\sum kA_k(x) = 1$ for any value of x), these points corresponding to the modal values can be connected by straight lines.
- This example illustrates that using linear concepts (triangular fuzzy sets, even partition) can yield a non-linear behaviour of fuzzy controller (due to nonlinearity of rules).

Compiled fuzzy controller

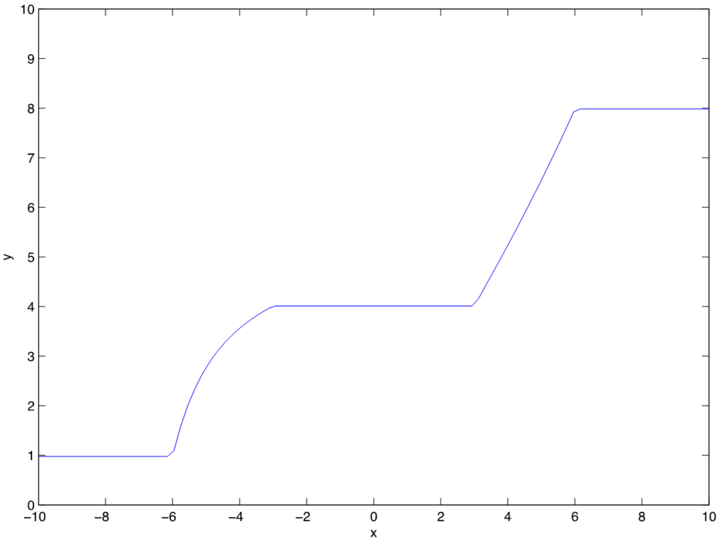
The concept of numerical characteristic can be used to build a compiled version of fuzzy controller stored as a look-up table on particular hardware (ROM, FPGA, etc.).

Example: Single-input Mamdani fuzzy model

IF x is “negative”	THEN y is “small”
IF x is “zero”	THEN y is “medium”
IF x is “positive”	THEN y is “large”



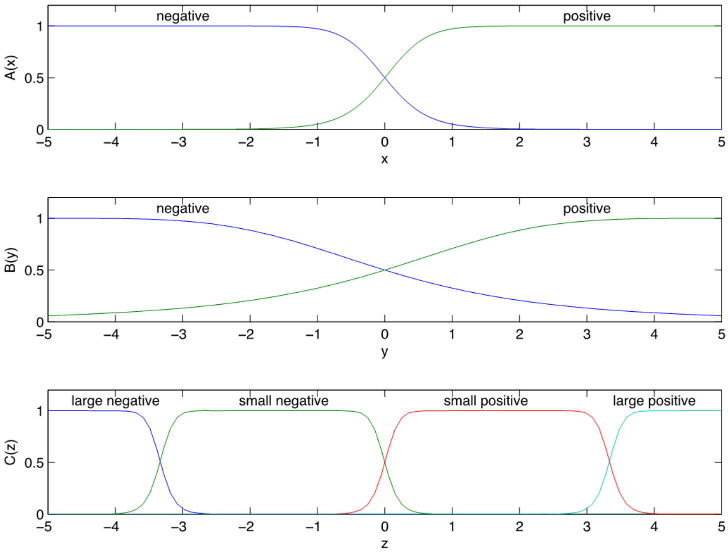
negative
zero
positive
A(x)
x
-10
-8
-6
-4
-2
0
2
4
6
8
10
0
0.2
0.4
0.6
0.8
1
1.2
small
medium
large
B(y)
y
0
1
2
3
4
5
6
7
8
9
10

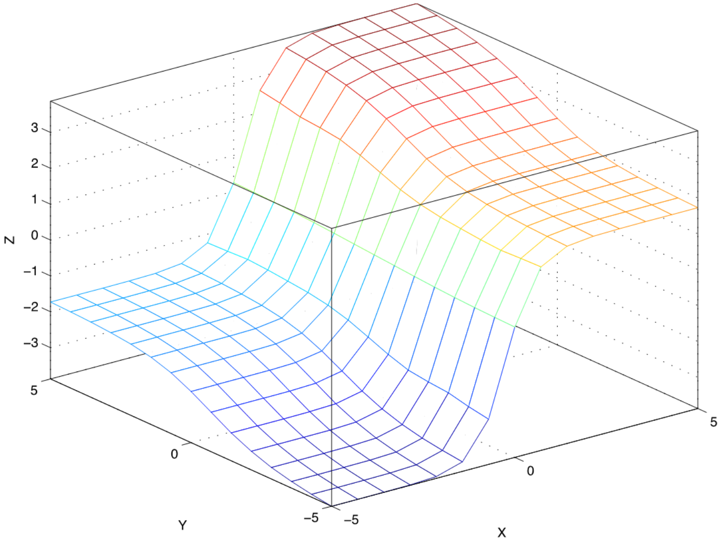

10
9
8
7
6
5
4
3
2
1
0
y
−10
−8
−6
−4
−2
0
2
4
6
8
10
x

Compiled fuzzy controller (cont.)

Example: Two-input Mamdani fuzzy model

IF x is "negative"	AND y is "negative"	THEN z is "negative large"
IF x is "negative"	AND y is "positive"	THEN z is "negative small"
IF x is "positive"	AND y is "negative"	THEN z is "positive smal"
IF x is "positive"	AND y is "positive"	THEN z is "positive large"


negative
positive
A(x)
1
0.5
0
–5
–4
–3
–2
–1
0
1
2
3
4
5
x
negative
positive
B(y)
y
large negative
small negative
small positive
large positive
C(z)
z


Z
3
2
1
0
−1
−2
−3
5
0
−5
Y
−5
0
5
X