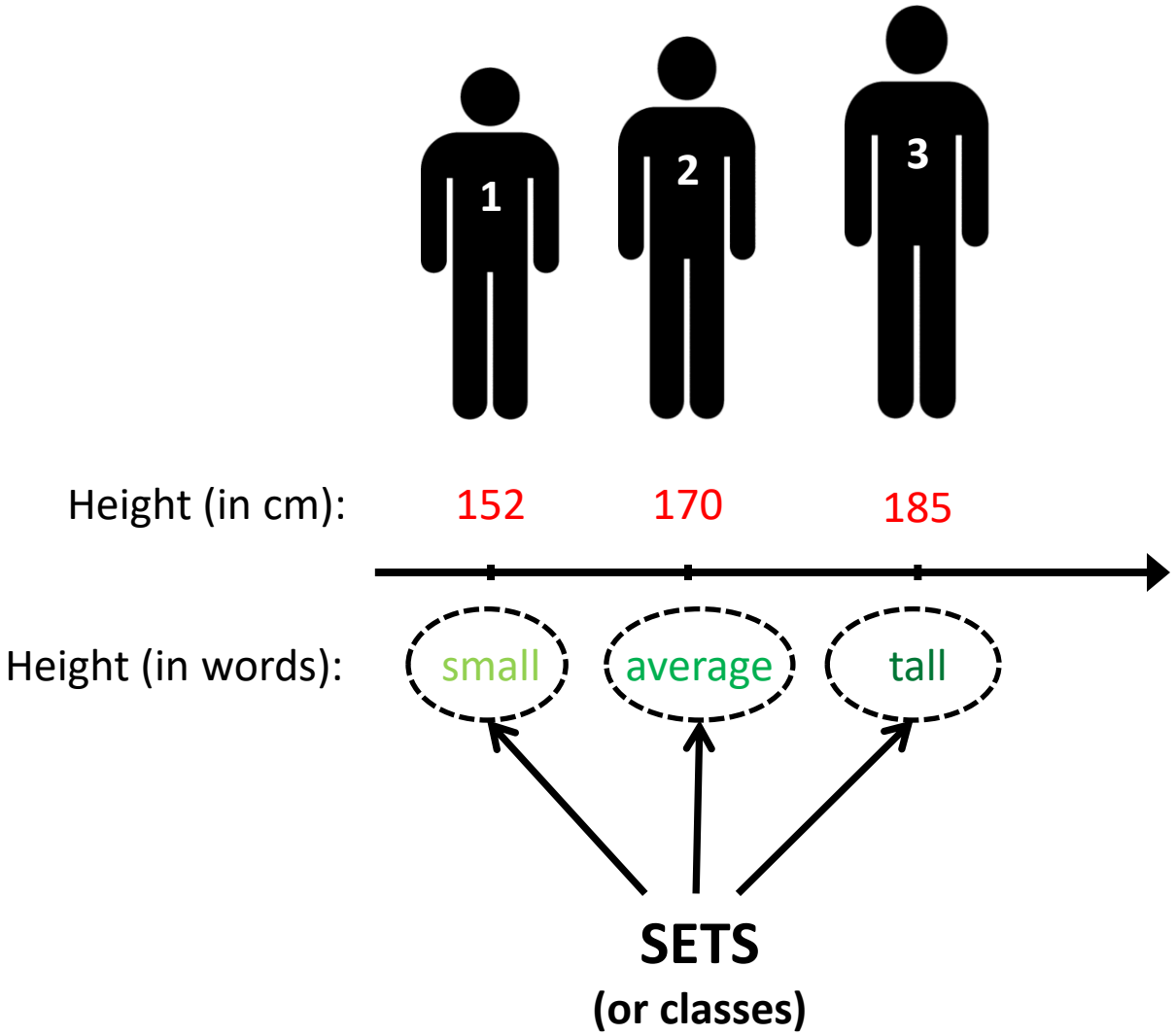
Control Engineering

Fuzzy Logic

MASSEY UNIVERSITY
TE KUNENGA KI PŪREHUROA
UNIVERSITY OF NEW ZEALAND

From numbers to sets

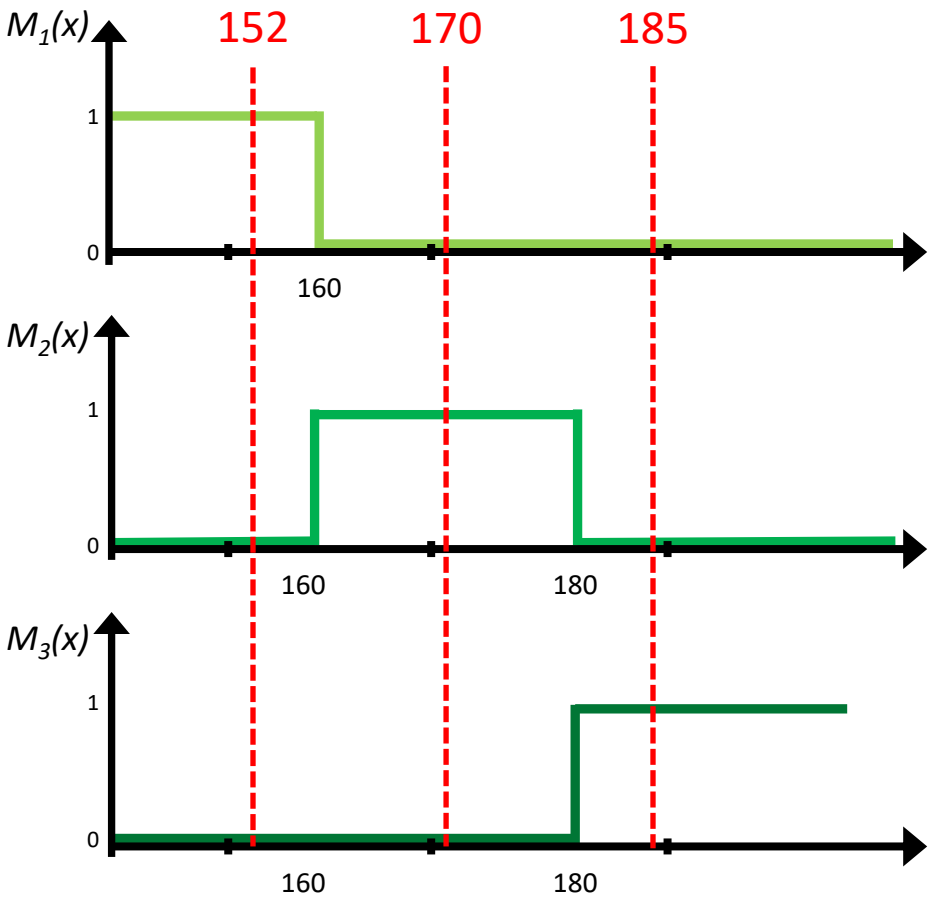


Membership functions

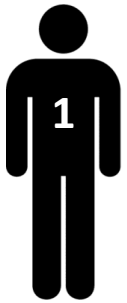
small
(height ≤ 160)

average
(160 < height ≤ 180)

tall
(180 < height)



Degree of membership



$M_1(152) = 1$
$M_2(152) = 0$
$M_3(152) = 0$

$M_1(170) = 0$
$M_2(170) = 1$
$M_3(170) = 0$

$M_1(185) = 0$
$M_2(185) = 0$
$M_3(185) = 1$

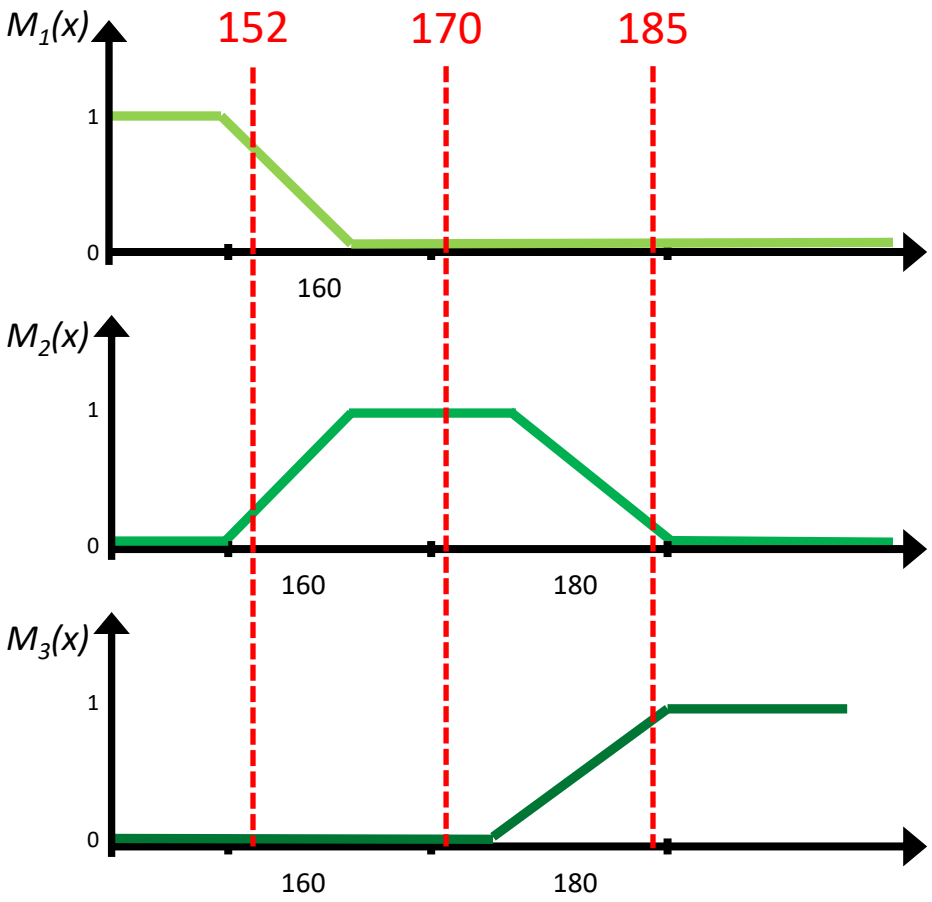
CRISP logic: degree of membership is 0 or 1.

Membership functions

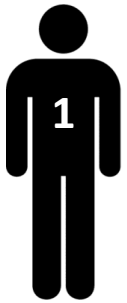
small

average

tall



Degree of membership



$M_1(152) = .8$
$M_2(152) = .2$
$M_3(152) = 0$

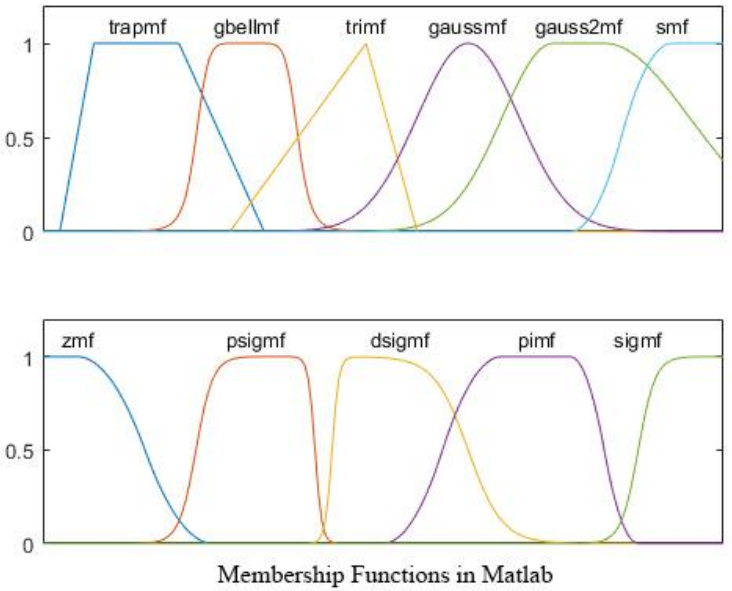
$M_1(170) = 0$
$M_2(170) = 1$
$M_3(170) = 0$

$M_1(185) = 0$
$M_2(185) = .1$
$M_3(185) = .9$

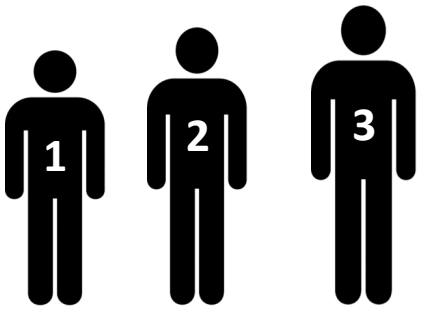
FUZZY logic: degree of membership is *anything between* 0 and 1.

Membership functions

- A membership function (MF) is a curve that defines how each point in the input space is mapped to a membership value.
- It can be of pretty much any kind, as long as it is between 0 and 1.
- Measuring the degree of membership of an input signal to different sets is called 'fuzzification'.



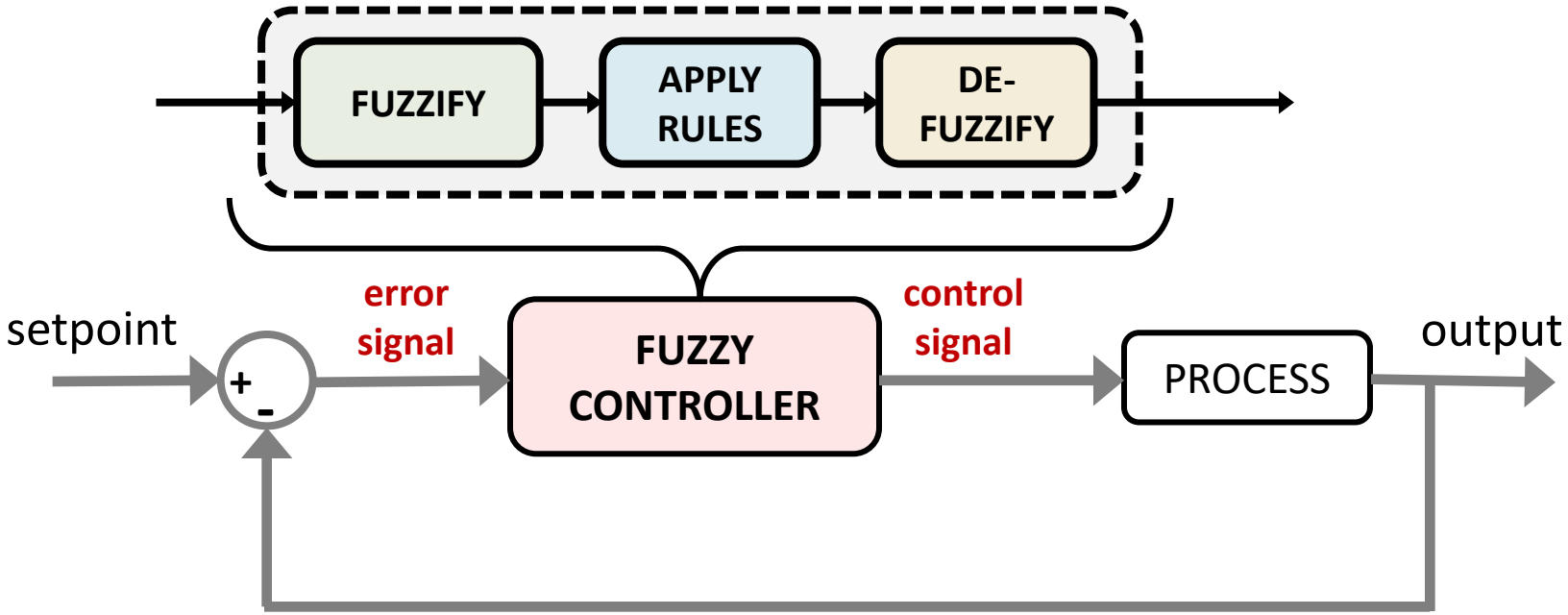
Membership Functions in Matlab



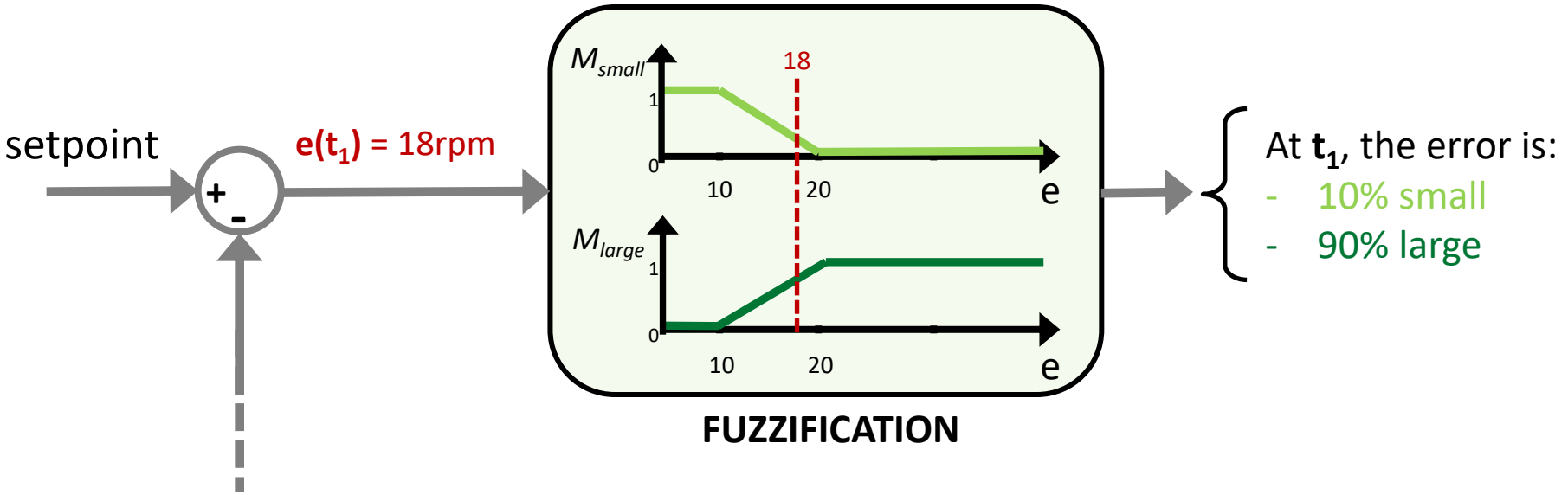
Instead of saying:
"Person 3's height is 185cm.",
we can now say:
"Person 3 is 90% tall and 10% average".

Talking in terms of membership is usually **much more intuitive,** especially within a program.

Fuzzy logic-based control



Fuzzification



(in this example, assume the error can only be positive)

Rules

(the most intuitive part of your program)

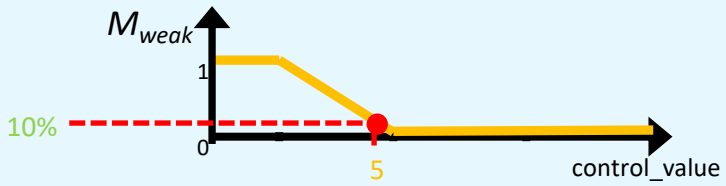
IF error **IS** small **THEN** control_value **IS** weak

IF error **IS** large **THEN** control_value **IS** strong

At $\mathbf{t_1}$, the error is:

- 10% small
- 90% large

Interpretation of 1st rule: "small maps to weak"



APPLY RULES

(map inputs to outputs)

Rules

(the most intuitive part of your program)

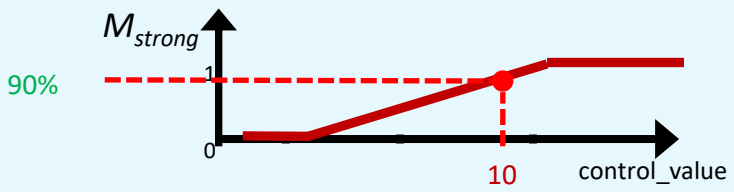
IF error **IS** small **THEN** control_value **IS** weak

IF error **IS** large **THEN** control_value **IS** strong

At $\mathbf{t_1}$, the error is:

- 10% small
- 90% large

Interpretation of 2nd rule: "large maps to strong"



APPLY RULES

(map inputs to outputs)

Rules

(the most intuitive part of your program)

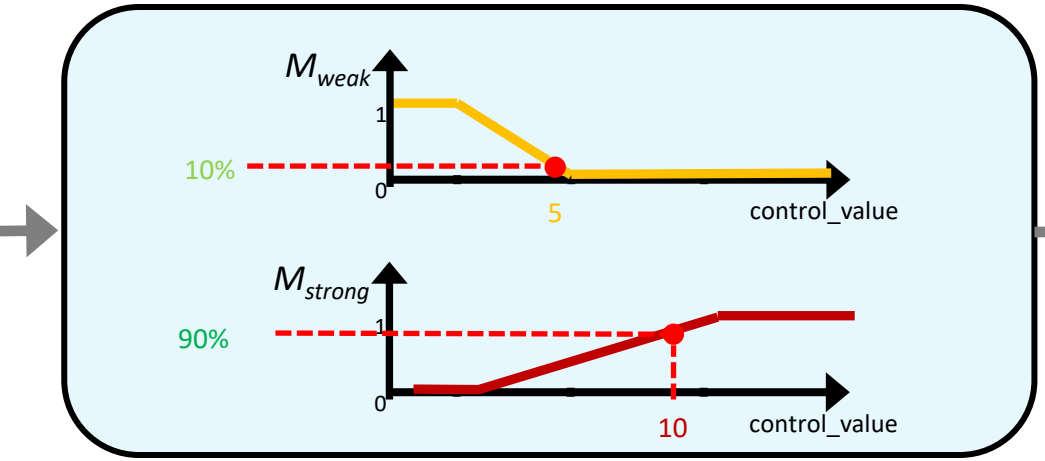
Note that rules can involve several membership functions at once.

They are combined with logical operators.

IF error **IS** small **OR** error **IS** large **THEN** control_value **IS** weak

IF error **IS** small **AND** rate of change of error **IS** positive **THEN** control_value **IS** strong

De-fuzzification



Output of first rule:
control_value1 = 5

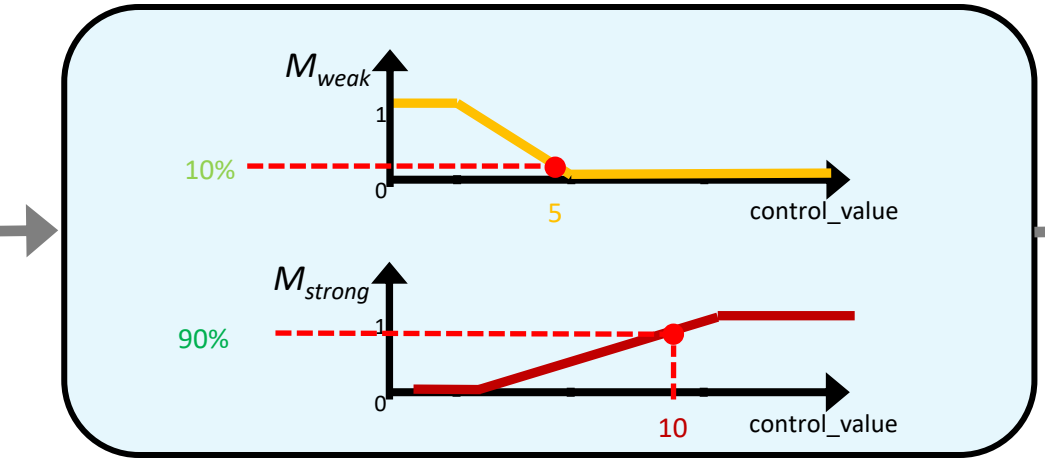
Output of second rule:
control_value2 = 10

These outputs need to be **aggregated** to obtain the final control value.
There are different ways to do this.

WEIGHTED AVERAGE-BASED DE-FUZZIFICATION

*control_value = 0.1*control_value1 + 0.9*control_value2 =* ***9.5***

De-fuzzification



Output of first rule:
control_value1 = 5

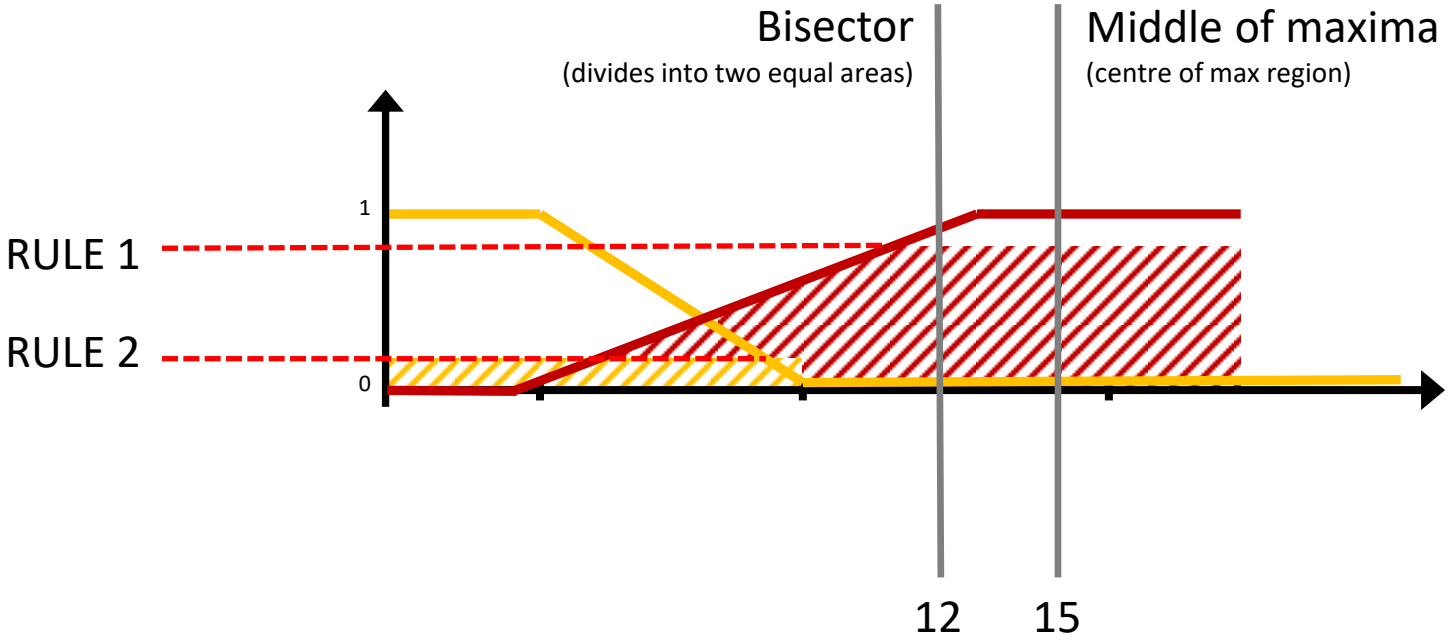
Output of second rule:
control_value2 = 10

These outputs need to be **aggregated** to obtain the final control value.
There are different ways to do this.

MAX-BASED DE-FUZZIFICATION

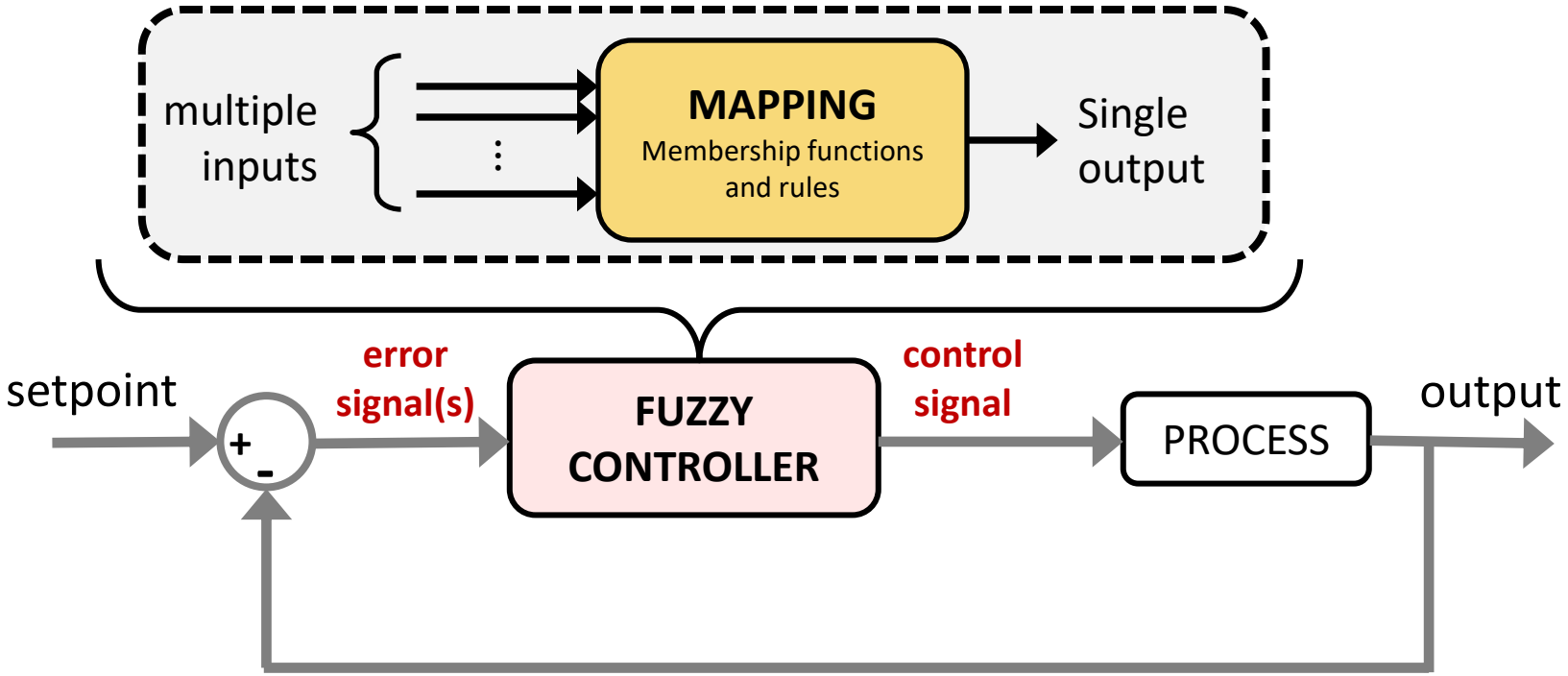
control_value = MAX(control_value1, control_value2) = ***10***

De-fuzzification



Fuzzy Inference

- Fuzzy inference is the overall process of designing the input-output **mapping**.



Exercise

Design a fuzzy PD controller with the following variables:

- **Error** (possible states: *Large Positive, Medium Positive ,Very Small, Medium Negative, Large Negative*).
- **Rate of change of error** (possible states: *Positive, Zero, Negative*).
- **Control value** (possible states: *LP, MP, SP, SN, MN, LN, Zero*)

1) Infer a maximum of 9 rules.
2) Choose membership functions for each input and output.
3) Show how your model would work for a set of input variables all being equal to 0.6. You may use any method of defuzzification.