Main control structures for energy systems

- Feedforward compensation
- Cascade control
- Decoupling
- Smith predictor
- Major actuation schemes

↳ some already seen for linear behaviour
+ some logic components involved
(properly state)

Foreword

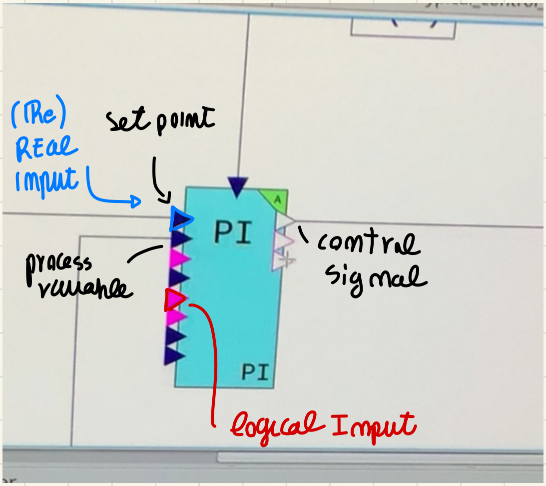
- We shall now review the major control structures of interest for us, i.e.,
  - feedforward compensation,
  - cascade control,
  - multivariable control with decoupling,
  - Smith predictor
- We shall also review some relevant actuation schemes, i.e.,
  - split range,
  - daisy-chaining,
  - time division output.

Foreword

(Modelica) > coursework > typical_control_structures

PI with lots of In/out so setpoint and so on...

- In this part of the course we shall centre our treatise on the modulating side of the matter and stay in an LTI context.
- It is however worth noticing right from now that implementing control structures often requires to set up some logic as well, and in certain cases also a proper management of nonlinear controller features like antiwindup.
- We are dealing with these "advanced" aspects – to a depth compatible with the course – later on, when we shall apply the structures that we are now revising to specific (energy-related) cases.



(The)
REal
input
set point
PI
process
variable
control
signal
PI
logical Input

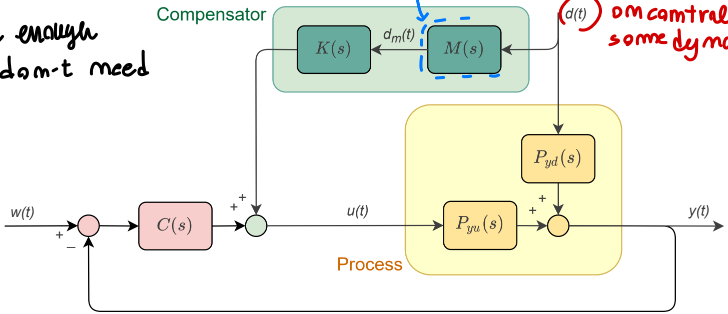
Feedforward compensation

Do you need compensation?

IF feedback is fast enough rejecting d(t) I don't need it...

additional sensor needed → cost!
(compensation IF needed)

disturbance act on control through some dynamic



- Purpose: reduce the influence on the controlled variable $y(t)$ of a measurable disturbance $d(t)$ acting on the loop forward path. → reduce on short run
- How: by computing $K(s)$ so that $Y(s)/D(s)$ be ideally zero, i.e.,

$$\frac{P_{yd}(s)+M(s)K(s)P_{yu}(s)}{1+C(s)P_{yu}(s)}=0, \qquad \Rightarrow K_{ID}(s)=-\frac{P_{yd}(s)}{M(s)P_{yu}(s)}$$

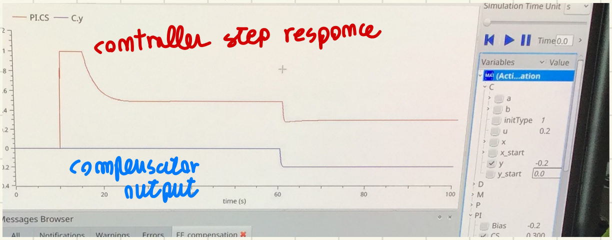
Ideally:

simulating "FF_compensation" with $\tau = 10$ process time const

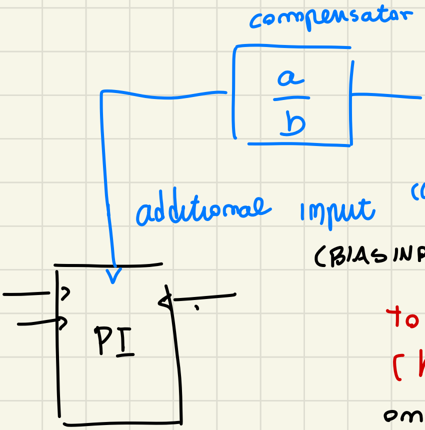
low disturbance

with compensator of d(t)

$\Downarrow$



IF Removing the compensator



additional input "BIAS" added to the control

(BIAS INPUT)

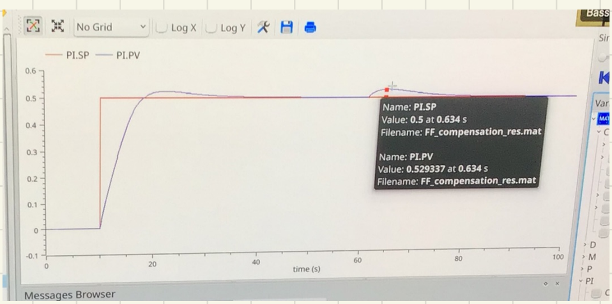
to disable compensator

(hasBias := FALSE)

on PI paramenters menù

$\Downarrow$

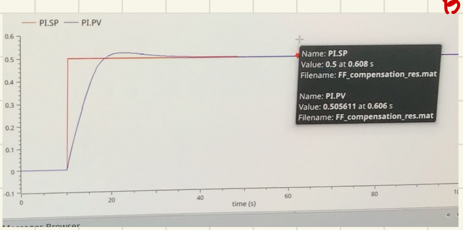
simulation



disturbance killed on long run

BIAS - OFF

BIAS - ON



disturbance Killed on short run!

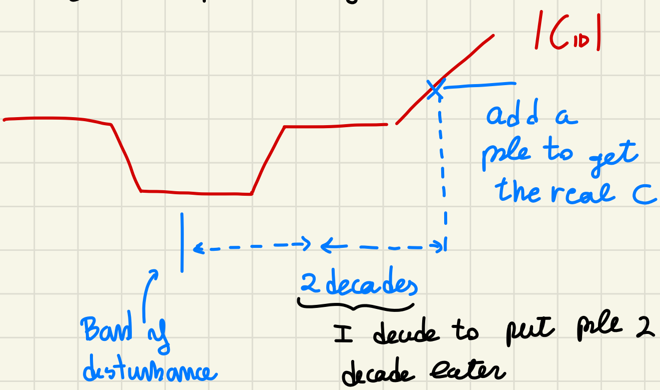
(compensation action)

Feedforward compensation

- The so found compensator is termed "ideal" (ID subscript) as
  - it may have more zeroes than poles (thus not being realisable),
  - and/or have RHP poles (thus producing critical cancellations).
- In such cases one has to obtain from $K_{ID}(s)$ the real compensator $K(s)$
  - omitting zeroes and/or adding poles,
  - and in any case not introducing RHP poles.
- This will yield a compensation "valid up to a certain frequency", namely that for which $K(j\omega)$ starts to differ "significantly" from $K_{ID}(j\omega)$, accounting for *both* magnitude and phase.

IF for example ideally

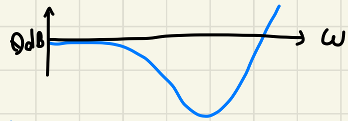
compensator ideal
(NON REALIZ.)



I decide to put pole 2 decade later

on Octave (OR MATLAB)

```
Cid = (1+s)^2/(1+10*s)
bode (Cid)
```



introducing an additional pole to be realizable... IF too near, it affects a lot the phase ∡...

⇐ if only pole one decade later ∡ bad approx on our range

← You are using Cid to counteract d(t) phasor but you do it without killing the disturbance! Bad cancellation

(Rule of thumb) 2 decade

While if POLE 2 decades later instead of 1...
phase approx of C respect Cid is almost equal!

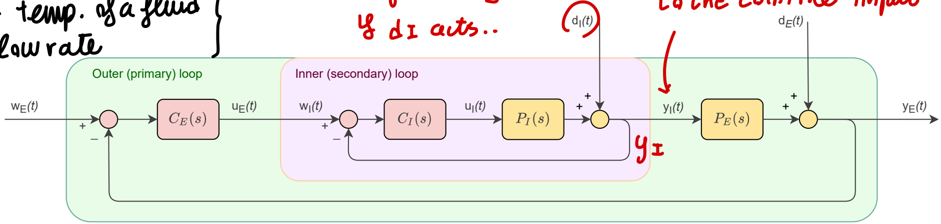
Cascade control

→ from an inner and outer loop

Typical case: control temp. of a fluid by a flow rate

Usefull only if d_I acts..

variable (y_I response faster than y_E) to the control input



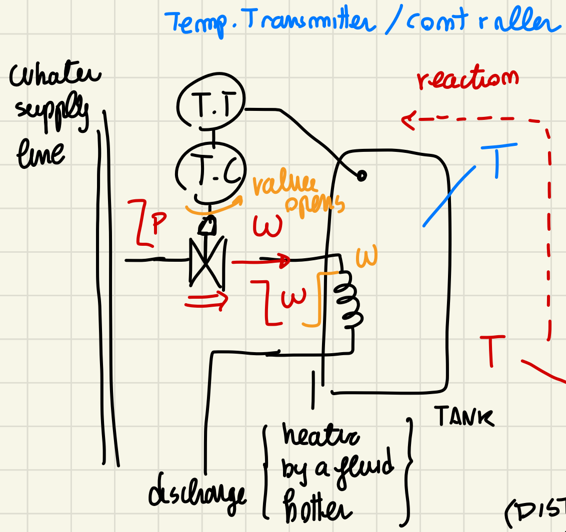
- Purpose: mitigate the effects of a disturbance $d_I(t)$ the effects of which appear on some measurable process variable $y_I(t)$ "before" – in a dynamic sense – they eventually show up on the primary controlled variable $y_E(t)$.
- How: by closing a "fast" inner (secondary) loop so as to hide both the dynamics of P_I and the effects of d_I to the outer (primary) one.
- Two remarks: C If internal loop is not affected by disturb... cascade is useless!
  - the scheme has the inherent cost of measuring $y_I(t)$, → add lots measurements ∀ element
  - and is hardly of any use (i.e., a single-loop control could do the job as well) in the absence of a significant $d_I(t)$.

Cascade control: typical example

control temperature by flow rate

heat a fluid by heat exchanger ...

you have to control fluid temp inside a vessel



pressure goes down $\downarrow P$
(pump failure)

$\Downarrow$

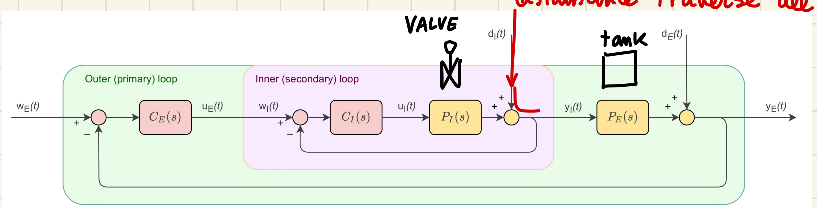
w flow rate goes down

on long run $T \downarrow\downarrow$

reaction of T.T, T.C open valve

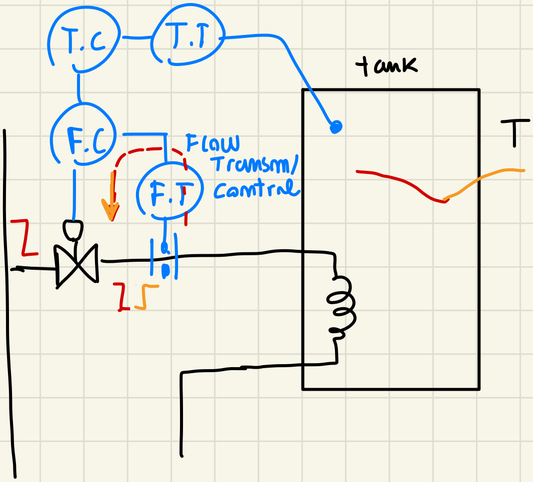
(DISTURBANCE)
enters on d_I
and

disturbance traverse all the dyn of slow tank before outer controller C_E reaction



if flow rate w is measurable $\Rightarrow$ why NOT use Faster controller reacting to d_I immediately

Change the scheme !



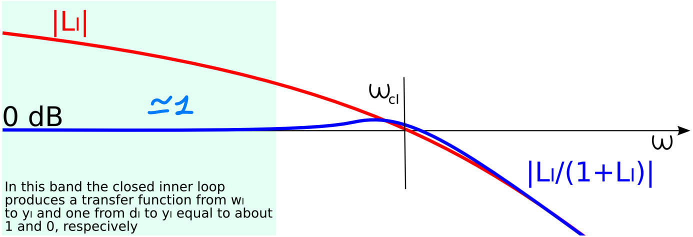
temperature don't even notice that disturbance

Cascade control

Explanation in the frequency domain

- Denote by $L_I(s) = C_I(s)P_I(s)$ the inner loop transfer function:

disturbance killed and set-point of inner loop instantaneusly transferred

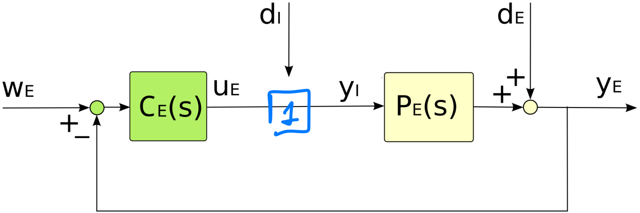


- In practice, indicating by ω_{cI} and ω_{cE} the critical frequencies of the inner and the outer loop, respectively, a minimum bandwidth separation of 0.5–1 decade is advised.
- If this is accomplished, it is possible to (approximately but reliably) compute ω_{cE} as if the outer (open) loop transfer function were just given by $C_E(s)P_E(s)$.

Cascade control

ideal control of inner loop like having a perfect $y_I = u_E$ istantaneus

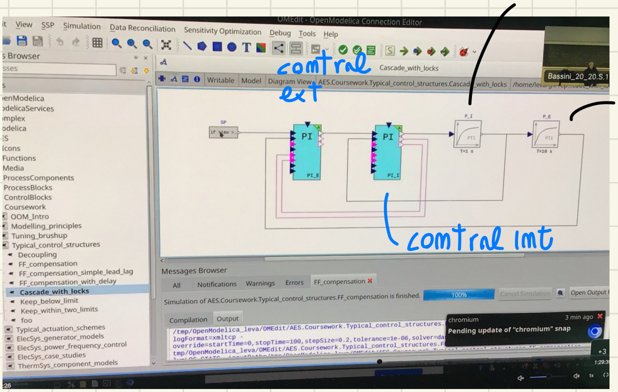
(Ideally):



- For the synthesis, therefore, one can view the outer loop as in the block diagram above.
- Overall, this leads to determining
  - $C_I(s)$ based on $P_I(s)$ only
  - and $C_E(s)$ based on $P_E(s)$ only (preserving however the required band separation).

Real implementation

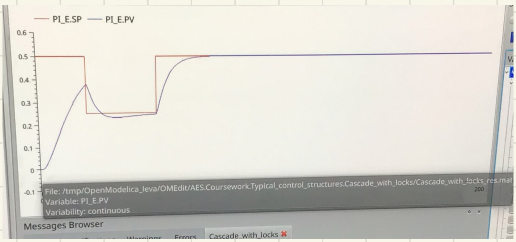
internal process



control ext

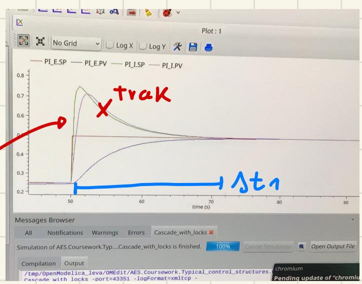
ext process

control int



well tuned cascade controller

↳ zoom in



trak

Δt1

internal loop track set point

internal loop ideally ...

Δt1 time of external loop to reach regime stabilize ...

on time horizon of Ext loop Int one is faster!

⇓

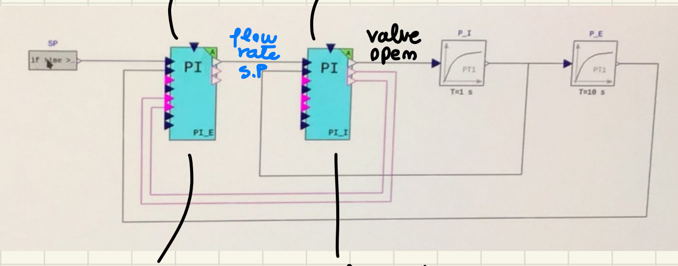
(tuning capability)

NOTICE we have strange connection on control scheme...

↳

Temp control

Flow control



SATURATION PROBLEM

Saturation limit of external controller?

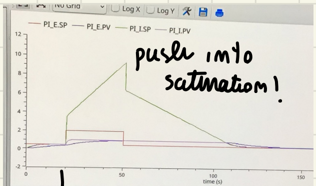
Valve with saturation limit (0,1)

Temp set point such that valve fully open? → depends on lot things

{ it could push to saturation! }

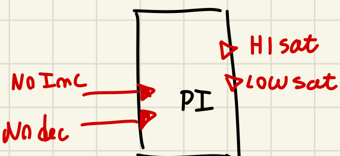
↓

simulation for hasLock := False



SOLUTION :

inner controller can signal
with logical output about HIGH / LOW saturation

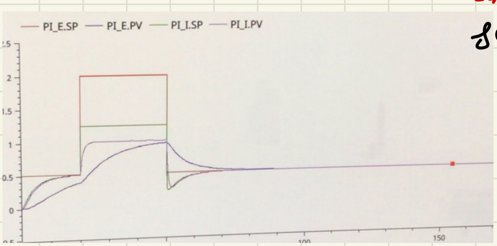


↑ additional logic input preventing incr/dec → with hasLocks := True

avoid saturation

LOGIC on control syst for proper function

{ cascade with inner loop logic for inc/dec }



(2nd problem) if inner controller on manual mode

operate the valve by myself

else outer may stay on manual
(otherwise I cut the loop)

Integral action (NOT A.S)
↳ HITS saturation and on AUTOMATIC mode → bad transient

⇓

MANUAL mode (for maintenance for example)
/ re enable control

you cut some tools... look @ what should I do...

(EXAMPLE) like on electric grids control standard

while Thermal syst control → etengeneus control
↳ hard structure

$\boxed{\text{cascade control}}$ proper design
some logic for proper work

remember logic of cascade control!

Multivariable control with decoupling

- Purpose: address Multiple Input, Multiple Output (MIMO) processes that are
  - square, i.e., with as many control inputs as controlled variables,
  - and interacting, i.e., where a control input influences more than one of the controlled variables.
- How: by taking a two-step approach, namely
  - by first prepending to the process a decoupler MIMO block so that the cascade of the two be diagonal,
  - and then closing one SISO loop per variable, synthesised with the known techniques.
- Remarks:

  relative Gain Array method RGA matrix

  - we treat the 2×2 scheme, generalising to $n\times n$ is straightforward;
  - there exist formal techniques for pairing, i.e. deciding which control input to use for governing which controlled variable; here we do not treat such techniques, and assume that pairing comes from physical considerations. (many times pairing is naturally chosen)

    obvious control var chose sometime

Multivarlable control with decoupling

- A 2 × 2 MIMO process is described in the LTI framework as

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}$$

cross dyn

complete matrix

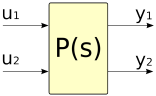
i.e., by the transfer matrix

$$P(s) = \begin{bmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{bmatrix}$$

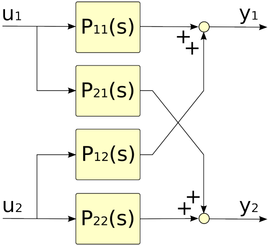
u_1 change y_1 BUT also y_2... interaction

both IN affect both OUT

- In terms of block diagrams this means



$$\begin{cases} Y_1 = P_{11}U_1 + P_{12}U_2 \\ Y_2 = P_{21}U_1 + P_{22}U_2 \end{cases} \Rightarrow$$

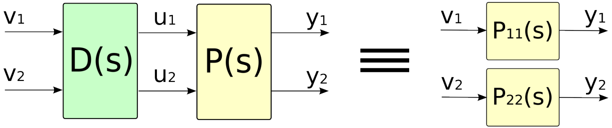


Multivarlable control with decoupling

The decoupling block

Decoupler → create two other control input v_1, v_2 influencing separately y_1, y_2 in cascade

- The purpose of $D(s)$ is to realise the equivalence indicated below:



i.e., setting $Y' = [Y_1\, Y_2]$, $U' = [U_1\, U_2]$ e $V' = [V_1\, V_2]$,

$$Y = PU = PDV = \begin{bmatrix} P_{11}(s) & 0 \\ 0 & P_{22}(s) \end{bmatrix} V$$

- Therefore, $D(s)$ is determined as

$$D(s) = P^{-1}(s) \begin{bmatrix} P_{11}(s) & 0 \\ 0 & P_{22}(s) \end{bmatrix}$$

Multivarlable control with decoupling

The decoupling block

- Interpretation:

$$D = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}^{-1} \begin{bmatrix} P_{11} & 0 \\ 0 & P_{22} \end{bmatrix}$$

hence

$$\begin{aligned} D^{-1} &= \left(\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}^{-1} \begin{bmatrix} P_{11} & 0 \\ 0 & P_{22} \end{bmatrix} \right)^{-1} \\ &= \begin{bmatrix} P_{11} & 0 \\ 0 & P_{22} \end{bmatrix}^{-1} \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \\ &= \frac{1}{P_{11}P_{22}} \begin{bmatrix} P_{22} & 0 \\ 0 & P_{11} \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \\ &= \frac{1}{P_{11}P_{22}} \begin{bmatrix} P_{11}P_{22} & P_{12}P_{22} \\ P_{11}P_{21} & P_{11}P_{22} \end{bmatrix} = \begin{bmatrix} 1 & P_{12}/P_{11} \\ P_{21}/P_{22} & 1 \end{bmatrix} \end{aligned}$$

Multivarlable control with decoupling

The decoupling block

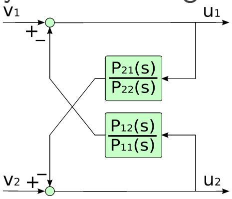
- Carrying on, $U = DV \Rightarrow V = D^{-1}U$, thus

$$\begin{bmatrix} V_1(s) \\ V_2(s) \end{bmatrix} = \begin{bmatrix} 1 & \frac{P_{12}}{P_{11}} \\ \frac{P_{21}}{P_{22}} & 1 \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \quad \Rightarrow \quad \begin{cases} V_1 & = & U_1 + \frac{P_{12}}{P_{11}} U_2 \\ V_2 & = & \frac{P_{21}}{P_{22}} U_1 + U_2 \end{cases}$$

- In synthesis, then

$$\begin{cases} U_1 & = & V_1 - \frac{P_{12}}{P_{11}} U_2 \\ U_2 & = & V_2 - \frac{P_{21}}{P_{22}} U_1 \end{cases}$$

and the decoupler is described by the block diagram



whence the frequently encountered name “backward decoupling”.

Multivarlable control with decoupling

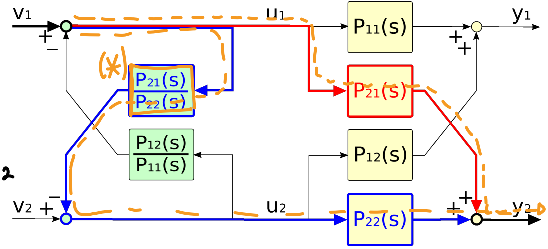
(No interest on FOREWARD decoupler)!

when you open one loop the dyn seen by other change ↳ (set Act MANUAL)

The decoupling block

↳ scheme

(only backward decoupling:) decoupler purpose is to ZERO the effect of v_1 on y_2 and v_2 on y_1...



← while NOT in backward (compensation)!

effect of v_1 on y_2, must be 0 by proper choice of (*)

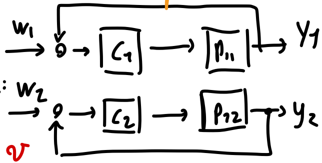
- The scheme shows the backward decoupler operation by evidencing how it zeroes the net signal path from v_1 to y_2 (sum of read and blue); of course the same holds for the symmetric path.
- The same feasibility/stability issues of feedforward compensation may arise, requiring to use approximated decoupling blocks and thus limiting the band where decoupling is effective.

Multivarlable control with decoupling

The overall scheme

closing the loop, thanks to decoupler is like:

(decoupled)

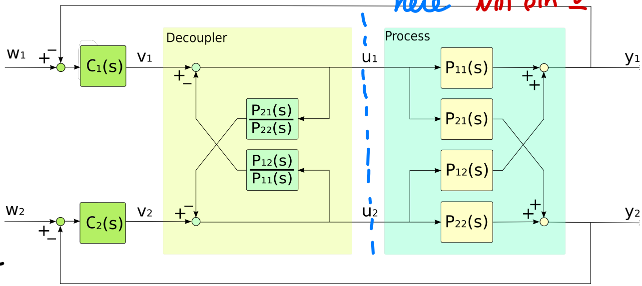


however...

PROBLEM
IF C1, C2 has
anti wind-up
functionalities
↓
manage saturation
of control var

$C_1 \to v_1$ but
$C_2 \to v_2$ saturation
on u_1, u_2

Saturation here NOT on v



phisical variables are subjected to SATURATION

- The two controllers $C_1(s)$ and $C_2(s)$ are designed with known SISO techniques, as if dealing with two independent processes having transfer functions $P_{11}(s)$ and $P_{22}(s)$, respectively.

Multivariable control with decoupling

The overall scheme

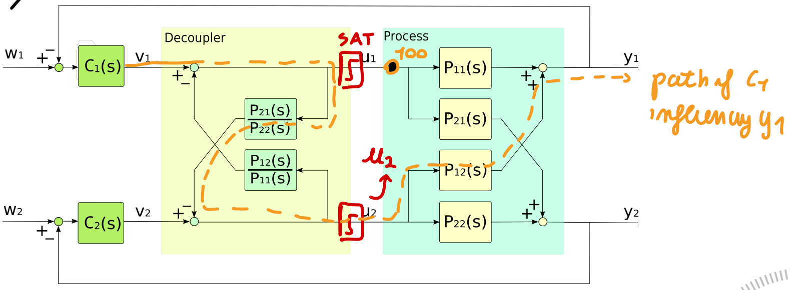
IF u_1 SAT. $\bar{u}_1 = 100$ ... BUT C_1 still influence y_1! because C_2 has a path. ⇒ (C_1, C_2) controllers compete for one control var u_2

C_1, C_2 acts on $\boxed{u_2}$

↓

u_2 take order from two CONTROL...

both loops gad bad performance



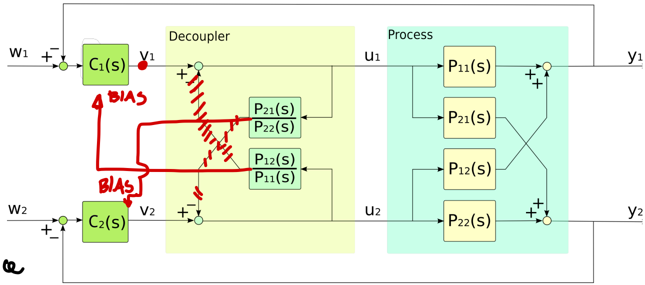
- The two controllers $C_1(s)$ and $C_2(s)$ are designed with known SISO techniques, as if dealing with two independent processes having transfer functions $P_{11}(s)$ and $P_{22}(s)$, respectively.

→ SOLUTION: correction produced by v_1 could enter v_2 even during saturation → you summ decoupling to BIAS INPUT ⇒

Multivarlable control with decoupling

The overall scheme

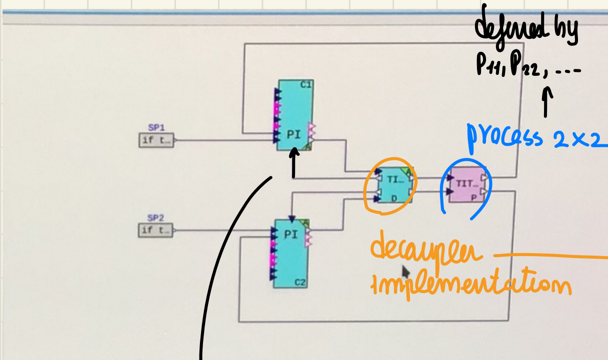
summing BIAS as decoupling action NOW v_1, v_2 are the signals SATURATING
↓
so C_1 does NOT influence y_1 through decoupling block



{Tuning scheme}
↓
to compute K, Ti...
LINEAR design
↓
don't care about SATURATION
↓
with interacting blocks, on Realization there is NO summation, sum through BIAS on CONTROLLER

- The two controllers $C_1(s)$ and $C_2(s)$ are designed with known SISO techniques, as if dealing with two independent processes having transfer functions $P_{11}(s)$ and $P_{22}(s)$, respectively.

on MODELICA "decoupling" (same system modelled)



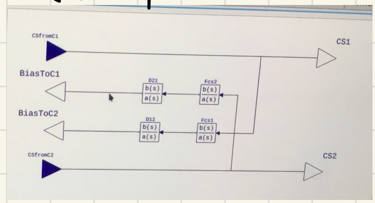
defined by $P_{11}, P_{22}, \ldots$

process 2x2

parametrized process 2I/2O

decoupler implementation

(Feedback Decoupler)



decoupler output to bias input of controllers

Decoupler takes the output of controllers and output signal to BIAS of controllers C_1, C_2

{ REALIZATION scheme }

the BIAS act before the saturation

as a DIGITAL block...

prevents to lose properties

PI block in INDUSTRIAL P.O.V you see lots of I/O → controller (BIAS..) (use properly input!)

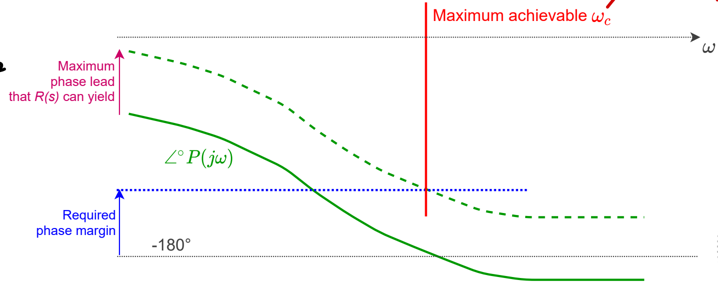
Smith predictor

large delay ? ⇒

- Purpose: address cases where the process has so large a delay that obtaining a certain stability degree (e.g., a desired phase margin) requires to reduce performance (e.g., response speed in terms of ω_c) unacceptably.
- Let us interpret this graphically:

limit !

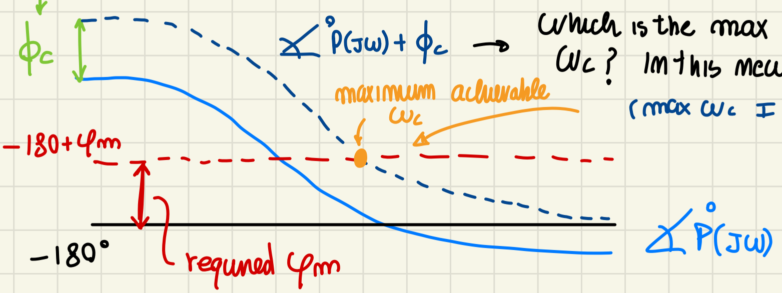
If you know process phase Bode plot ... and max phase limit



- This is how one *meaningfully* says that a delay is "too large".

on BODE hypotesis
⇓ suppose a certain $\measuredangle$ of the process

max $\measuredangle$ $\mathring{C}(j\omega)$ (exist on a general controller structure)
ω



→ which is the max ammissible ω_c? In this new plot (max ω_c I can achieve)

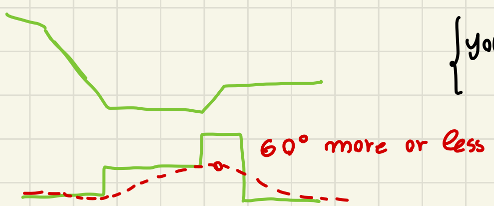
If I want an higher ω_c → I can have $(-180+\varphi_m)$ given $\measuredangle P$ + Regulator structure + φ_m and I can tell you the max ω_c achievable → this is why you may need a SMITH PREDICTOR!

I want $\varphi_m = 60°$ ⇒ so you have a certain $\omega_{c\,MAX}$
↓
smith predictor to have $\omega_{c\,MAX}$ ↑↑

IF $\omega_{c\,MAX}$ too slow
NOT only feedback

"delay is long" → when a process as a delay so large that obtaining a certain φ_m require ω_c unaccetable!

{I'm a PID CONTROLLER}



{you can have lot LF zeros..} dangerous...
↓
don't put freq spread singularities

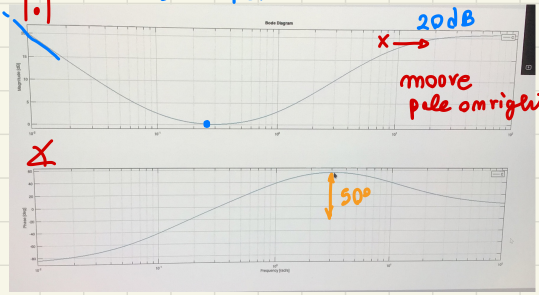
plotting Bode om Octave

↓

$K = 1, \quad T_i = 10, \quad T_d = 1, \quad N = 10$

$$C = K^* \left(1 + \frac{1}{s/T_i} + \frac{s^* T_d}{1 + s^* T_d / N} \right) \rightarrow$$

Bode plot

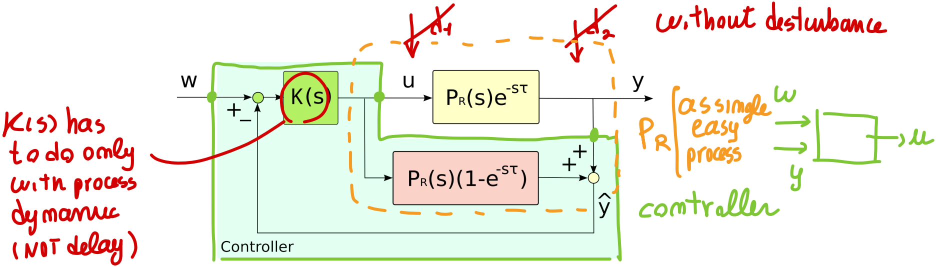


there is a
certain max ω_c you can get!

to be more sensitive to noise

Smith predictor

to deal with high delay:



- How: by observing that in the scheme above

$$\frac{\widehat{Y}(s)}{U(s)} = P_R(s)$$

where the transfer function $P_R(s)$ is assumed rational.

⇒ computing $\frac{\hat{Y}}{W}$ (T.F) →

- Block $K(s)$ can thus be synthesised with known methods, accounting only for the rational dynamics of the process.

Smith predictor

- The scheme immediately yields

$$\frac{\widehat{Y}(s)}{W(s)} = \frac{K(s)P_R(s)}{1+C(s)P_R(s)}$$

- Additionally, the "controller" in the same scheme is equivalent to a feedback one with transfer function

$$C(s) = \frac{K(s)}{1+K(s)P_R(s)(1-e^{-s\tau})}$$

thus

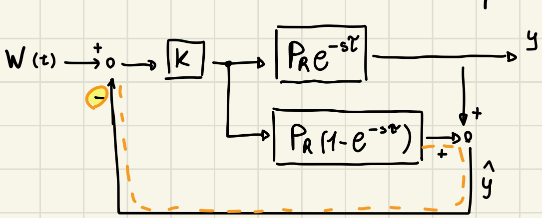
$$\frac{Y(s)}{W(s)} = \frac{C(s)P_R(s)e^{-s\tau}}{1+C(s)P_R(s)e^{-s\tau}}$$

$$= \frac{\frac{K(s)P_R(s)e^{-s\tau}}{1+K(s)P_R(s)(1-e^{-s\tau})}}{1+\frac{K(s)P_R(s)e^{-s\tau}}{1+K(s)P_R(s)(1-e^{-s\tau})}} = \ldots = \boxed{\frac{K(s)P_R(s)e^{-s\tau}}{1+K(s)P_R(s)}}$$

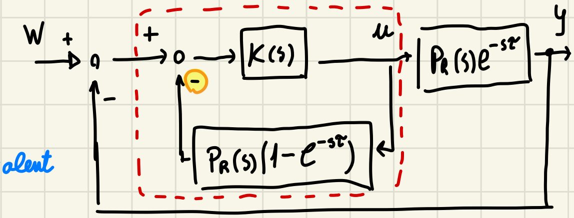
delayed

Smith predictor

→ operate well if $\hat{y}$ Good estimation of y



≡ equivalent



$$C(s) = \frac{K(s)}{1+K(s)P_R(s)\,(1-e^{-s\tau})}$$

↑ NOT Rational... you need to realize delay ecc...

IF $\hat{y}$ Bad estimation

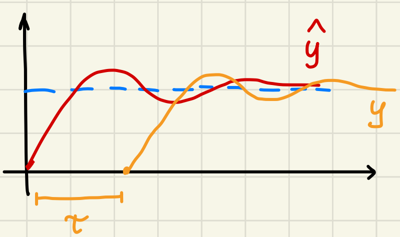
↓

POOR approx with delay

$$\frac{Y(s)}{W(s)} = \frac{C}{1+CP_R e^{-s\tau}} = \frac{K}{1+KP_R(1-e^{-s\tau})} \quad \frac{1}{1+\dfrac{P_R e^{-s\tau}}{1+CP_R(1-e^{-s\tau})}} =$$

$$= \frac{K P_R e^{-s\tau}}{1+CP_R(1-e^{-s\tau})+P_R e^{-s\tau}} = \dots$$

computing the controller



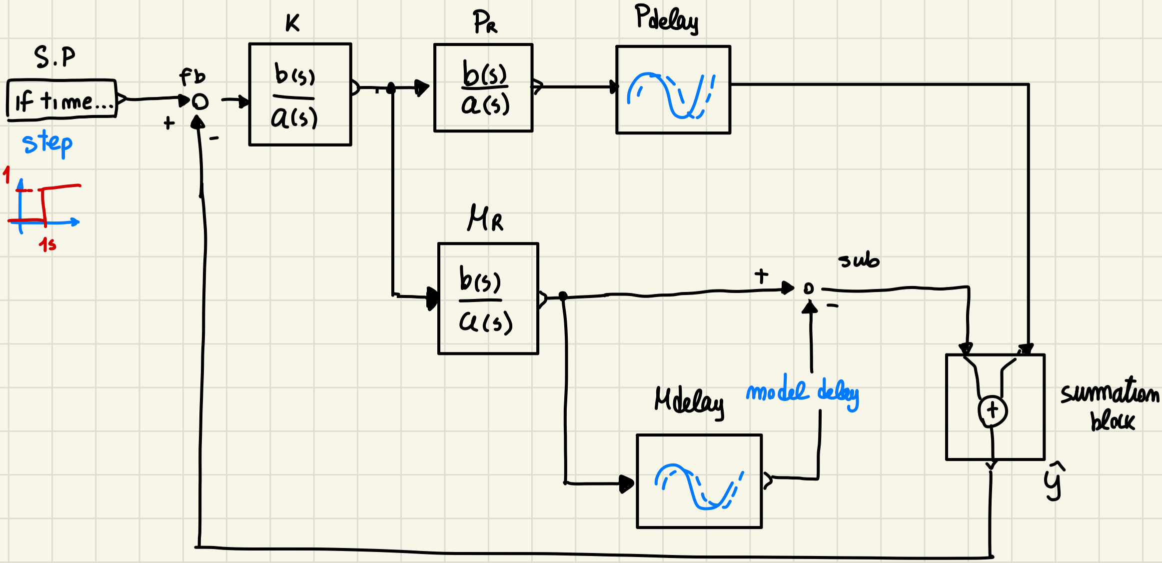
Smith predictor relays on having a Good model of the process

take desired td set time and delay τ...
Knowing this I set up the CONTROLLER

simtesize of controller

Modelica: build smith predictor

With linear block:
(neglect non linearity)



(Data of the system)

$\tau = 5s$

$P_R(s) = \frac{1}{(1+s)^2}$ rational dynamic

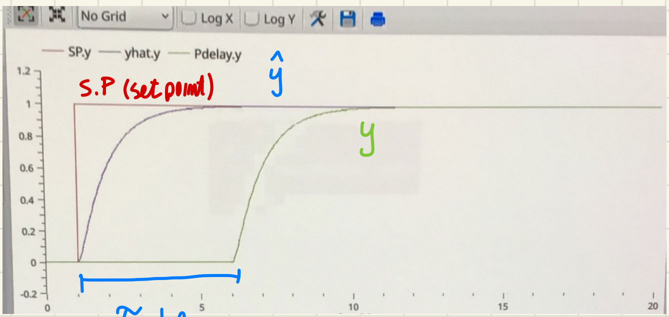
$K(s) = \frac{(1+s)^2}{s(1+0.1s)}$ $\Rightarrow L_R(s) = K(s) P_R(s) = \frac{1}{s(1+0.1s)}$

+5s delay: it should setup $\simeq 5s$

↓ we get this response:

⇒ SIMULATE for 20s

$\hat{y}$ settles in $\simeq 5s$
y in $5 + \tau = 10s$
to the set-point



τ delay

Was the smith predictor necessary?

y settles in 10s ↴ equivalent cut-off freq

$\Rightarrow$ equivalent $\omega_c = \dfrac{1}{10/5} = 0{,}5$ rad/s

process phase? @ $\omega = 0.5$ rad/s

$\Downarrow$

without smith predictor

$$\angle^{\circ} P(j0.5) = \overbrace{-2\,\mathrm{atan}^{\circ}(0.5)}^{\text{rational dyn}} \overbrace{- 0.5 \cdot 5 \cdot \tfrac{180}{\pi}}^{e^{-5s}\text{ delay}} =$$

$$= -3.4273 \text{ rad} = -196.37^{\circ}$$

↓ yes & No

BUT

suppose $\tau = 20$ with same P_R

⟶ in this case

$$\angle^{\circ} P(j0.5) = -2\,\mathrm{atan}^{\circ}(0.5) - 20 \cdot 0.5 \cdot \frac{180}{\pi} =$$

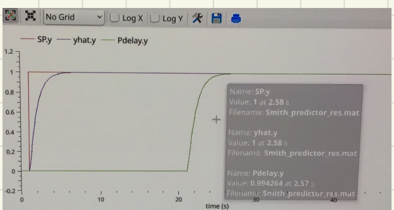
$$= -10 \text{ rad} \simeq \underset{\sim\sim\sim}{-600^{\circ}} \;!$$

IMPOSSIBLE to RECOVER with a sensible NON predictor controller

While with SMITH PREDICTOR for $\tau = 20$ ⟵

SIMULATING on Modelica:

for 20s simulate



same shape BUT

⟶ settling time for y will be

$$20 + \underbrace{5}_{\text{settling time for } \hat{y}} = 25$$

↓

$$\omega_c \text{ equivalent} = \frac{1}{25/5} = 0.2$$

phase of P: $\angle^{\circ} P(j0.2) = -\mathrm{atan}(0.2) - 20 \cdot 0.2 \cdot \dfrac{180}{\pi} =$

$$= -4.39 \text{ rad} \simeq \underbrace{-251^{\circ}} \ll 600^{\circ}$$

to get $\varphi_m = 60^{\circ}$ a lead of

$250 - 120 = 130^{\circ}$ could be required

$\Downarrow$

virtually impossible

... While with SMITH PREDICTOR

↳

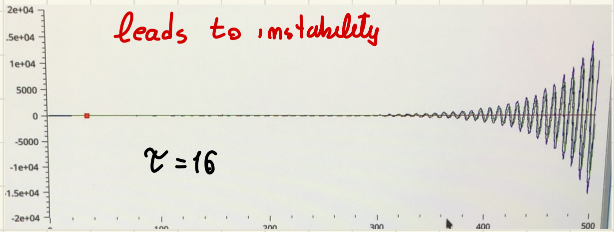
With smith predictor ⇒ longer delay BUT same shape

you need
a PRECISE model!

in a τ mistake,
running SIMULATION all
becomes different → NO reliable

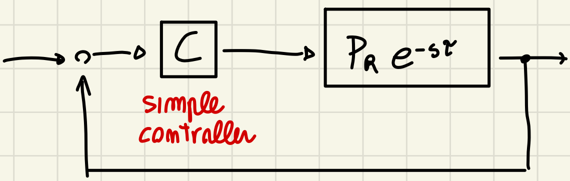
[mistake on
τ is dangerous!]

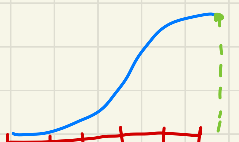
both if $\tau \gtrless \tau_{estimated}$
↓
don't work...
good prediction
needed



(Note) how to compute ω_c

to compute ω_c equivalent





I want $\bar{\omega}_c$ and a certain $\bar{\varphi}_m$

$\tau = \frac{1}{5} t_R$
↓
$\omega_c = \frac{1}{\tau} = 5/t_R$

$\measuredangle P(j\bar{\omega}_c) = \bar{\varphi}_P$
↓
I need $\bar{\varphi}_P + \measuredangle C(j\bar{\omega}_c) = \bar{\varphi}_m - 180°$ <0
↓
$\measuredangle C(j\bar{\omega}_c) = \bar{\varphi}_m - 180° - \bar{\varphi}_P$

above 70, 80° is hard to
obtain by a single loop !

Smith predictor

- Summing up, therefore, we get

$$\frac{Y(s)}{W(s)} = \frac{\widehat{Y}(s)}{W(s)} e^{-s\tau}$$

which means that synthesising $K(s)$ based on $P_R(s)$ and using the Smith predictor scheme, the obtained behaviour of y is the same as that of $\widehat{y}$, just delayed by τ.

- Caveat:
  - the model has to be "more precise" than is needed for mere feedback control
  - and disturbances need not to be too significant (ore be *very well* compensated for in the prediction path)

otherwise $\widehat{y}$ ceases to be a good prediction of y, to the detriment of the scheme operation.

Actuation schemes

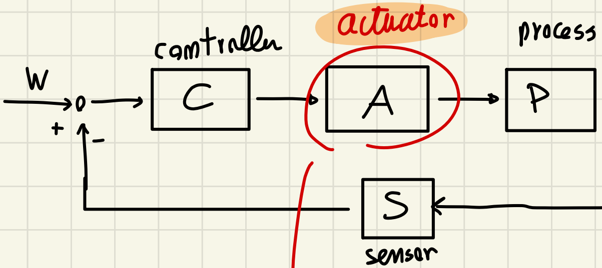
companion to control structures

General idea

- In general we think of schemes in which a controller drives one actuator.
- In several cases, this is not really true.
- Sometimes an action as decided by a controller has to be exerted through more articulated physical machineries... more complex machine
- ...that however *at the control synthesis level* need representing in such a way to accept one "command" from the controller.
- Dealing with this part of a control system means talking about actuation schemes.
- We are going to see a few ones of interest for us.

→ see how it impacts the CONTROLLER not always as a single actuator ! sometimes have to face

In general when drawing control scheme...

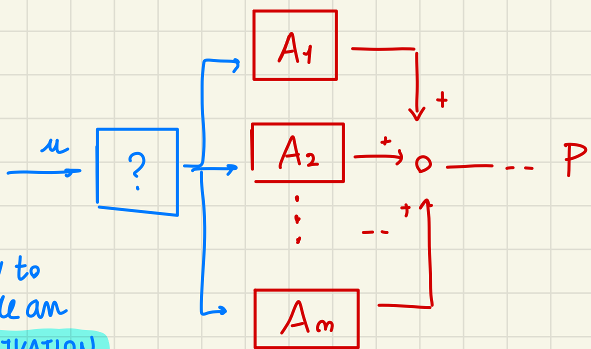


"text book" loop

sometimes this is more complex!

⇓

you may have



We wanna find an equivalent single command system

how to make an ACTIVATION SCHEME with one single equivalent control

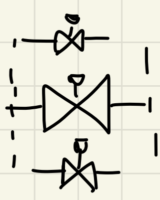
- for example on THERMAL system { heat / cool } machine are different syst

electric resistor

water cooling syst

- OR for govern flow rate using a VALVE which when is almost fully open/closed NOT ideal (Imperfection issues)!

⇓ you prefer to use multiple valve in parallel or sequence...



(ISSUE) from one control signal $u \rightarrow A_1, A_2, \vdots, A_M$ commanding more actuators

how can I manage it?

Split range

ask for a Thermal controller with 2 outputs
heat/cool

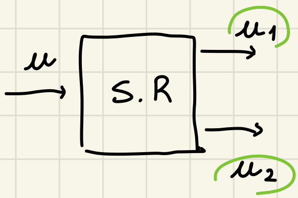
or more

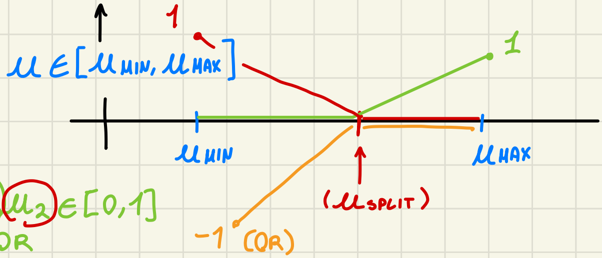
- Purpose: make two actuators behave like a single one by having each of them act in a different range of the control variable (whence the name).
- Typical example: a temperature controller with one actuator for heating ~~for heating~~ and one for cooling.
- How: denoting by $u_1 \in [0,1]$ and $u_2 \in [0,1]$ the two actuators and supposing – without loss of generality – that the transition happens for $u = 0$, where $u \in [-1,1]$ is the controller output, by simply setting

$$u_1 = \begin{cases} u & u \geq 0 \\ 0 & u < 0 \end{cases} \qquad u_2 = \begin{cases} 0 & 0 \geq 0 \\ -u & u < 0 \end{cases}$$

- Note: sometimes a *dead zone* is introduced around $u = 0$ to avoid switching in and out the two actuators too frequently.

SPLIT RANGE (S.R)





$u_1, u_2 \in [0,1]$

OR

$u_1 \in [0,1]$

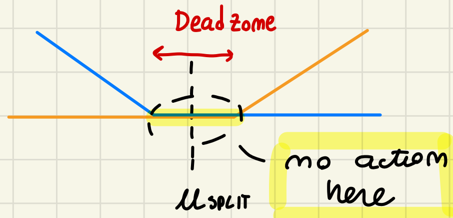
$u_2 \in [-1,0]$

OR

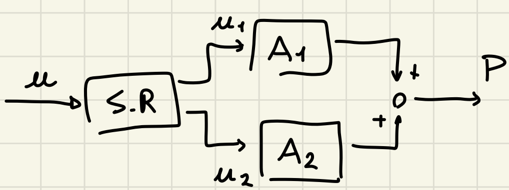
... other combination

sometimes → DEAD ZONE is added around u_{SPLIT}

to prevent → a continuous oscillating switch ON/OFF



PROBLEM (addressed later on..)



these dynamics can be different making the controller see either A_1P or A_2R as the controlled process depending on the value of u

↑ different power requirement !

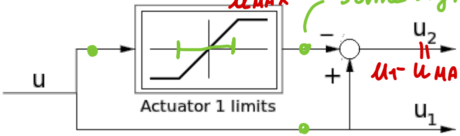
Daisy chaining

"Generalization" of SPLIT RANGE

each one mooves to its max

- Purpose: have several actuators activated in sequence (the $i+1$-th starting to operate when the i-th has reached its maximum).
- Most typical motivation: start with the most energy-efficient actuator (e.g., a heat pump) and have a less efficient one (e.g., an electric heater) intervene only if the first one is not sufficient. → bring in more actuator if need
- How: by using the diagram

for 2 actuators: ↳



u_{MAX} — same signals → $u_2 = 0$

IF $u > u_{MAX}$: $u_2 = u_1 - u_{MAX}$, $u_1 = u_{MAX}$ ...

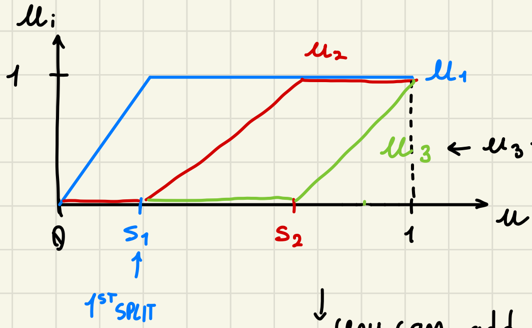
that easily generalises to an arbitrary number of actuators.

- Note:here too dead zones are sometimes introduced, for the same reason mentioned above.

for more actuators DAISY CHAIN

Generalization

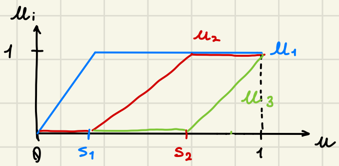
⇓ easy to (de-normalize)

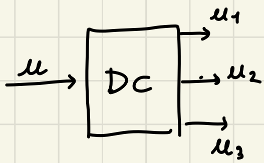


← u_3 take care of final Rang

↓ you can add dead-zone...

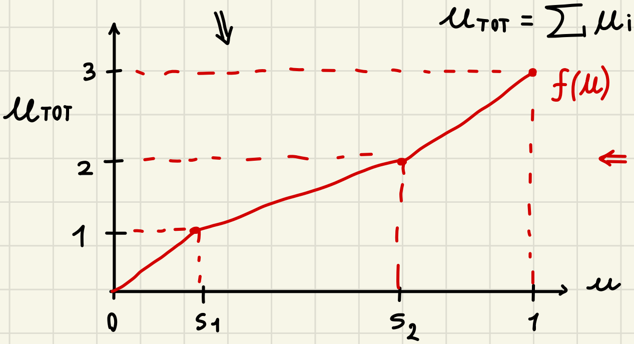
- and.. This is seen by the controller as a non linear characteristic



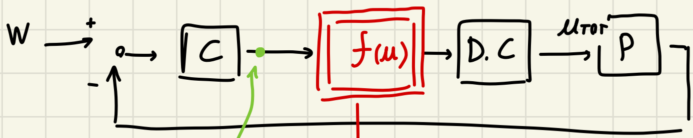


⇓

$u_{TOT} = \sum u_i$ sent to the process



⇐ What see the controller



hard tuning of the controller!

(Compensating by the inverse) ⇐ { IF I know f I could put f^{-1} here to invert dynamic }

{ non linear characteristics } to account for

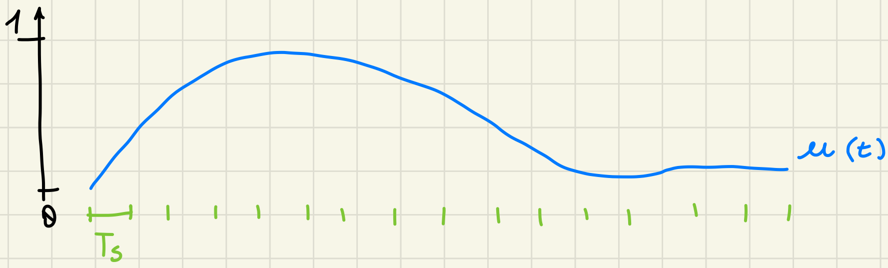
↑ to properly mitigate

Time Division Output

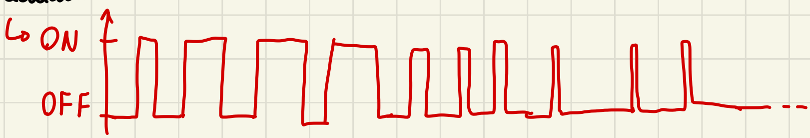
with high power actuator, modulation is hard to do...
you can lose efficiency!

- Purpose: make an on/off actuator behave like a modulating one.
- Most typical motivations:
  - modulating high-power actuators may be impractical or even impossible;
  - even in the absence of the above problem, operating the actuator not at 100% may reduce its efficiency.
- How: by deciding an *actuation period* T_a, small w.r.t. the process dynamics' time scale, taking this as the sampling time for the (digital) controller, and having the control signal $u \in [0,1]$ provide the actuator activation's duty cycle.
- Example: with $T_a = 10\,s$, $u = 0.6$ means that in the sampling period the actuator will be on for $6\,s$ and then off for $4\,s$.

TIME DIVISION OUTPUT



Commant to actuator



depending on the percentage $0 \div 1$ of $u(t)$ → setting the Δt on which stay ON during T_s

↓

you see the average effect NOT ON–OFF frequent change

↓ depending on application also