APC-ABB6:
Standard control elements
Constraint switching.

Note: there is no APC-ABB5

QUIZ

What are the three most important inventions of process control?

- Hint 1: According to Sigurd Skogestad
- Hint 2: All were in use around 1940

SOLUTION

1. PID controller, in particular, I-action
2. Cascade control
3. Ratio control

The three main inventions of process control can only indirectly and with effort be implemented with MPC

1. Integral action with MPC: Need to add artificial integrating disturbance in estimator
    - ARC: Just add an integrator in the controller (use PID)

2. Cascade control with MPC: Need model for how u and d affect y_1 and y_2.
    - ARC: Just need to know that control of y_2 indirectly improves control of y_1

3. Ratio control with MPC: Need model for how u and d affect property y
    - ARC: Just need the insight that it is good for control of y to keep the ratio R=u/d constant

Because of this, MPC should be on top of a regulatory control layer with the setpoints for y_2 and R as MVs.

ARC: Standard Advanced control elements

Each element links a subset of inputs with a subset of outputs. Results in simple local design and tuning

First, there are some elements that are used to improve control for cases where simple feedback control is not sufficient:

E1*. Cascade control[2]
E2*. Ratio control
E3*. Valve (input)[3] position control (VPC) on extra MV to improve dynamic response.

Next, there are some control elements used for cases when we reach constraints:

E4*. Selective (limit, override) control (for output switching)
E5*. Split range control (for input switching)
E6*. Separate controllers (with different setpoints) as an alternative to split range control (E5)
E7*. VPC as an alternative to split range control (E5)

All the above seven elements have feedback control as a main feature and are usually based on PID controllers. Ratio control seems to be an exception, but the desired ratio setpoint is usually set by an outer feedback controller. There are also several features that may be added to the standard PID controller, including

E8*. Anti-windup scheme for the integral mode
E9*. Two-degrees of freedom features (e.g., no derivative action on setpoint, setpoint filter)
E10. Gain scheduling (Controller tunings change as a given function of the scheduling variable, e.g., a disturbance, process input, process output, setpoint or control error)

In addition, the following more general model-based elements are in common use:

E11*. Feedforward control
E12*. Decoupling elements (usually designed using feedforward thinking)
E13. Linearization elements
E14*. Calculation blocks (including nonlinear feedforward and decoupling)
E15. Simple static estimators (also known as inferential elements or soft sensors)

Finally, there are a number of simpler standard elements that may be used independently or as part of other elements, such as

E16. Simple nonlinear static elements (like multiplication, division, square root, dead zone, dead band, limiter (saturation element), on/off)
E17*. Simple linear dynamic elements (like lead–lag filter, time delay, etc.)
E18. Standard logic elements

[2] The control elements with an asterisk * are discussed in more detail in this paper.

Sigurd Skogestad, "Advanced control using decomposition and simple elements". Annual Reviews in Control, vol. 56 (2023), Article 100903 (44 pages).

JULY 2023

Annual Reviews in Control 56 (2023) 100903

Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

Review article

Advanced control using decomposition and simple elements

Sigurd Skogestad

Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), Trondheim, Norway

ARTICLE INFO

Keywords:
Control structure design
Feedforward control
Cascade control
PID control
Selective control
Override control
Time scale separation
Decentralized control
Distributed control
Horizontal decomposition
Hierarchical decomposition
Layered decomposition
Vertical decomposition
Network architectures

ABSTRACT

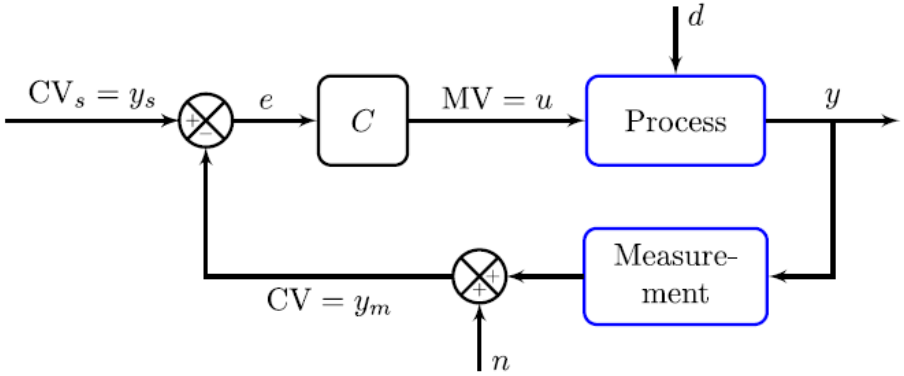
The paper explores the standard advanced control elements commonly used in industry for designing advance control systems. These elements include cascade, ratio, feedforward, decoupling, selectors, split range, ar more, collectively referred to as "advanced regulatory control" (ARC). Numerous examples are provided, wi a particular focus on process control. The paper emphasizes the shortcomings of model-based optimizatic methods, such as model predictive control (MPC), and challenges the view that MPC can solve all contr problems, while ARC solutions are outdated, ad-hoc and difficult to understand. On the contrary, decomposir the control systems into simple ARC elements is very powerful and allows for designing control systems f complex processes with only limited information. With the knowledge of the control elements presented i the paper, readers should be able to understand most industrial ARC solutions and propose alternatives ar improvements. Furthermore, the paper calls for the academic community to enhance the teaching of AR methods and prioritize research efforts in developing theory and improving design method.

Contents

8.2. The harder problem: Control structure synthesis

As a third approach, machine learning may prove to be useful. Machine learning has one of its main strength in pattern recognition, in a similar way to how the human brain works. I have observed over the years that some students, with only two weeks of example-based teaching, are able to suggest good process control solutions with feedback, cascade, and feedforward/ratio control for realistic problems, based on only a flowsheet and some fairly general statements about the control objectives. This is the basis for believing that machine learning (e.g., a tool similar to ChatGPT) may provide a good initial control structure, which may later be improved, either manually or by optimization. It is important that such a tool has a graphical interface, both for presenting the problem and for proposing and improving solutions.

Most basic element: Single-loop PID control (E0)



MV-CV Pairing. Two main pairing rules (supervisory layer*):

1. ***"Pair-close rule" : T****he MV should have a large, fast, and direct effect on the CV.*
2. ***"Input saturation rule":*** *Pair a MV that may saturate with a CV that can be .given up (when the MV saturates).**
   - *Exception: Have extra MV so we use MV-MV switching (e.g., split range control)*

Additional rule for interactive systems:

3. ***" RGA-rule"***
   - *Avoid pairing on negative steady-state RGA-element. Otherwise, the loop gain may change sign (for example, if the input saturates) and we get instability with integral action in the controller.*

*For regulatory (stabilizing) control, we usually want to avoid using <u>any</u> MV that may saturate (so Rule 2 becomes: Avoid using a MV that may saturate), but for the supervisory layer this is not possible

Details on RGA-rule

> **Pairing rule 1** (page 450): *Prefer pairings such that the rearranged system, with the selected pairings along the diagonal, has an RGA matrix close to identity at frequencies around the closed-loop bandwidth.*

However, one should avoid pairings where the sign of the steady-state gain from u_j to y_i may change depending on the control of the other outputs, because this will yield instability with integral action in the loop. Thus, $g_{ij}(0)$ and $\widehat{g}_{11}(0)$ should have the same sign, and we have:

> **Pairing rule 2** (page 450): *Avoid (if possible) pairing on negative steady-state RGA elements.*

The reader is referred to Section 10.6.4 (page 438) for derivation and further discussion of these pairing rules.

S. Skogestad & I. Postlethwaite, *Multivariable feedback control*, 2nd Edition, Wiley, 2005

Most common “Advanced regulatory control” structures

Used when single-loop feedback control (PID) alone is not good enough

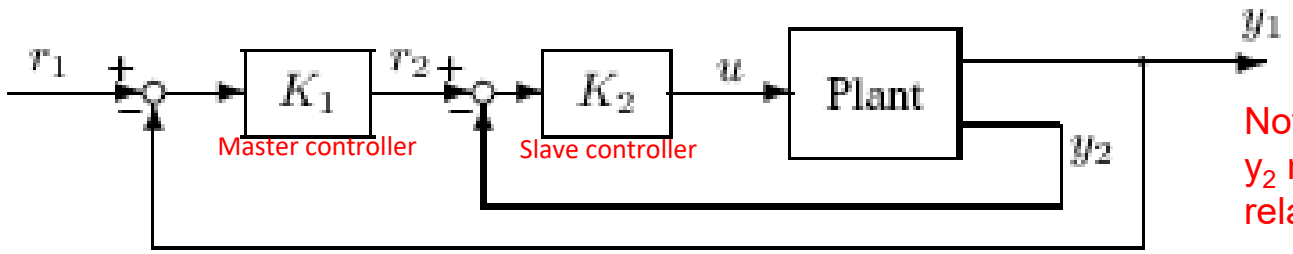
1. Cascade control (measure and control internal variable) E1
2. Ratio control E2
3. Extra MV dynamically: Valve position control (VPC) E3
    - Also known as input resetting or midranging
4. Extra MV steady state (MV-MV switching): 3 alternatives
    1. Split range control E5
    2. Split parallel control E6
    3. Split VPC E7
5. Change in CV (CV-CV switching): Selectors (max, min) E4

All of these are extensively used in practice, but little academic work

E1. Cascade control

- 1 input u
- 1 (main) output y_1
- 1 extra measurement y_2
- Key assumption: Control of y_2 indirectly makes it easier to control y_1
- Solution: Primary controller (master) controls y_1 by setting setpoint $r_2=y_2^{sp}$ to a fast secondary controller (slave) which manipulates u

General case ("parallel cascade")



(a) Extra measurements y_2 (conventional cascade control)

Not always helpful…
y_2 must be closely related to y_1

Special common case ("series cascade")

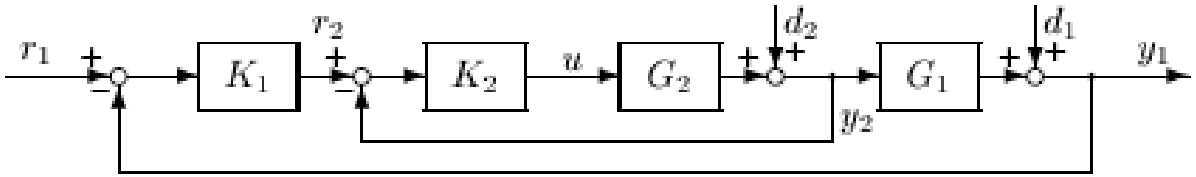
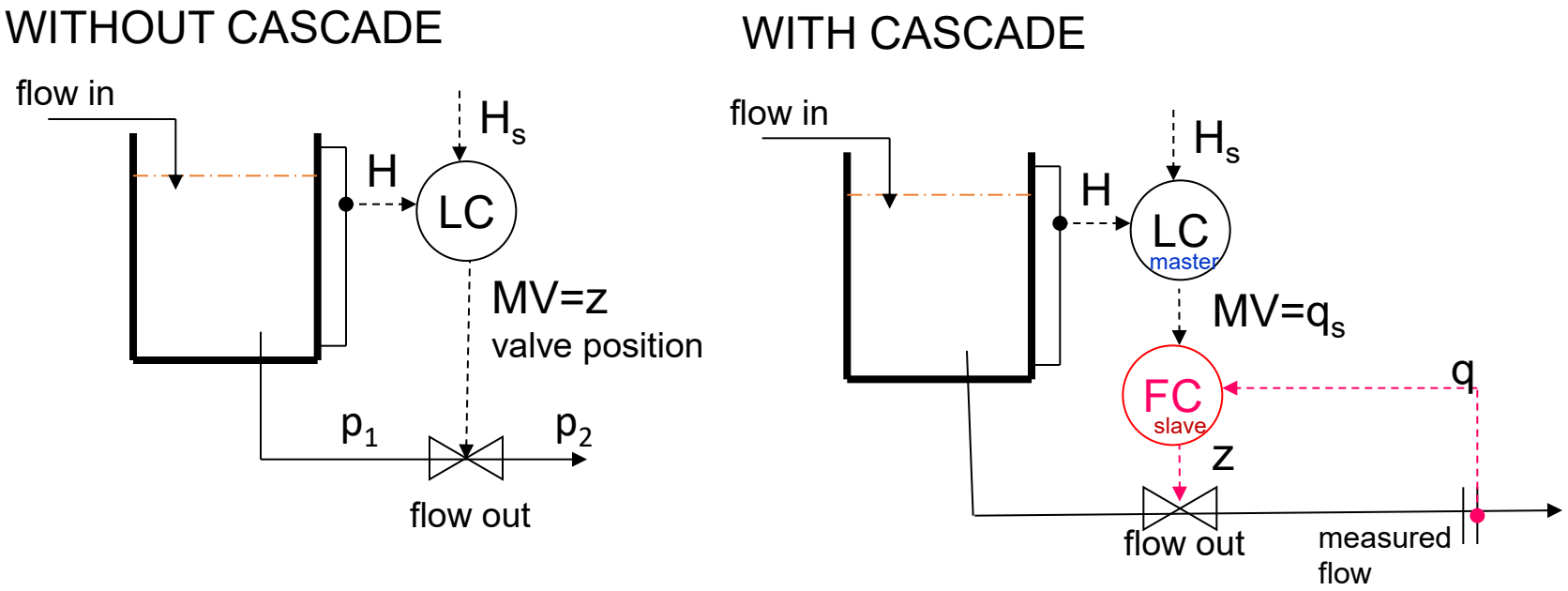


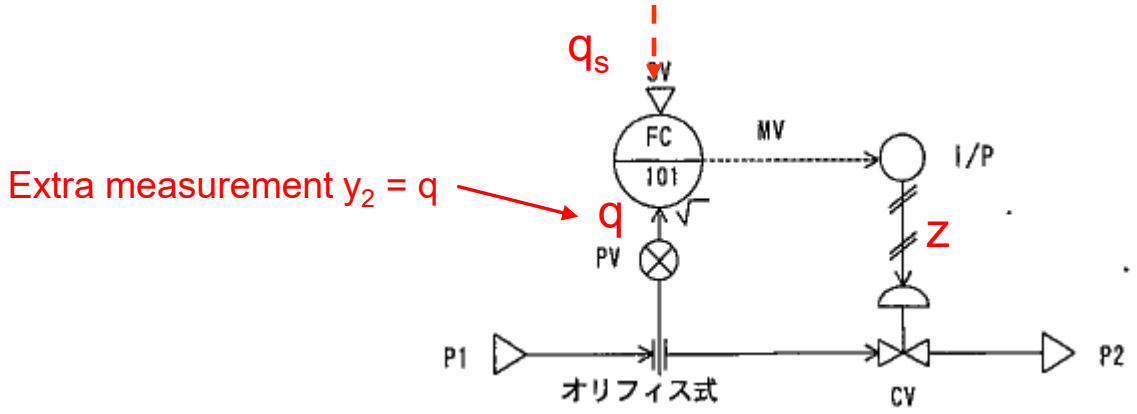
Figure 10.11: Common case of cascade control where the primary output y_1 depends directly on the extra measurement y_2

Example: Flow controller on valve (very common!)

- Helpful to reduce valve nonlinearity and provide local disturbance rejection (d=p_1, p_2)
  - y = level H in tank (or could be temperature etc.)
  - u = valve position (z)
  - y_2 = flowrate q through valve

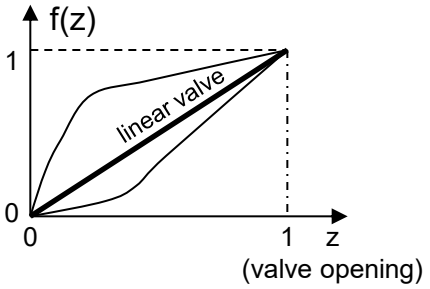


What are the benefits of adding a slave flow controller?



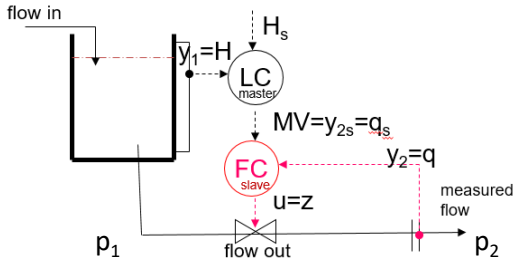
$$\text{Flow rate: } q = C_v f(z)\sqrt{\frac{p_1 - p_2}{\rho}} \quad [\text{m}^3/\text{s}]$$

1. Counteracts nonlinearity in valve, f(z)
   - With a fast flow controller we can assume q = q_s (in spite of nonlinearity in the valve)



2. Eliminates effect of disturbances in p_1 and p_2 (FC reacts faster than outer level loop)

Block diagram flow controller



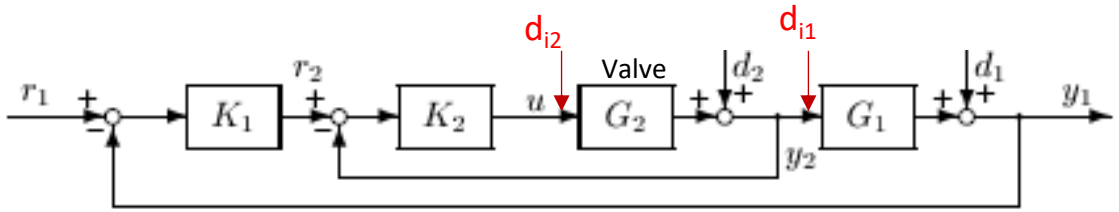
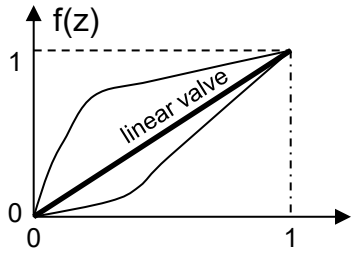


Figure 10.11: Common case of cascade control where the primary output y_1 depends directly on the extra measurement y_2

Example: Level control with slave flow controller:

u = z (valve position, flow out)
y_1 = H
y_2 = q
d_{i1} = flow in (cascade does not help for this one)
d_{i2} = p_1-p_2

Transfer functions:
G_2 = k(z)/(τs+1) where k(z) = dq/dz (nonlinear!)
G_1 = - 1/(As)
K_1 = Level controller (master)
K_2 = Flow controller (slave)



k(z) = slope df/dz

Shinskey (1967)

The principal advantages of cascade control are these:

1. Disturbances arising within the secondary loop are corrected by the secondary controller before they can influence the primary variable.
2. Phase lag existing in the secondary part of the process is reduced measurably by the secondary loop. This improves the speed of response of the primary loop.
3. Gain variations in the secondary part of the process are overcome within its own loop.
4. The secondary loop permits an exact manipulation of the flow of mass or energy by the primary controller.

When use (series) cascade ?

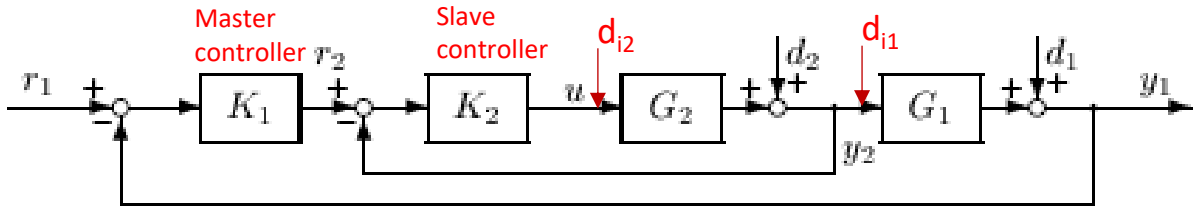


Figure 10.11: Common case of cascade control where the primary output y_1 depends directly on the extra measurement y_2

Use cascade control (with an extra secondary measurement y_2) when one or more of the following occur:

1. **Significant disturbances d_2 and d_{i2} inside slave loop (and y_2 can be controlled faster than y_1)**
2. **The plant G_2 is nonlinear or varies with time or is uncertain.**
3. Measurement delay for y_1
   - **Note: In the flowsheet above, y_1 is the measured output, so any measurement delay is included in G_1**
4. Integrating dynamics (including slow dynamics or unstable) in both G_1 and G_2, (because without cascade a double integrating plant G_1G_2 is difficult to control)

Design / tuning

- First design K_2 ("fast loop") to deal with d_2 and d_{i2} (based on model G_2)
- Then, with K_2 closed, design K_1 to deal with d_1 and d_{i1} (based on model G_1T_2)

Transfer functions and tuning

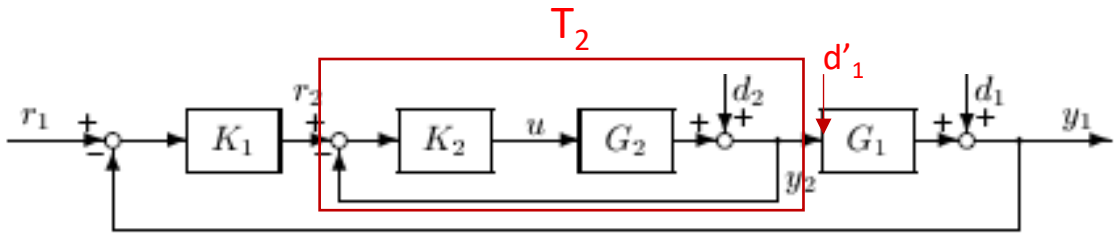


Figure 10.11: Common case of cascade control where the primary output y_1 depends directly on the extra measurement y_2

First tune fast inner controller K_2 ("slave")

Design K_2 based on model G_2

Select τ_{c2} based on effective delay in G_2

Transfer function for inner loop (from y_{2s} to y_2): $T_2 = G_2 K_2/(1+G_2 K_2)$

Because of integral action, T_2 has loop gain = 1 for any G_2.

With SIMC we get: $T_2 \approx e^{-\Theta 2s}/(\tau_{c2}s+1)$

Nonlinearity: Gain variations (in G_2) translate into variations in actual time constant τ_{C2} (see next page)

Then with slave closed, tune slower outer controller K_1 ("master"):

Design K_1 based on model $G_1'=T_2*G_1$

Can often set $T_2=1$ if inner loop is fast!

- Alternatively, $T_2 \approx e^{-\Theta 2s}/(\tau_{c2}s+1) \approx e^{-(\Theta 2+\tau c2)s}$
- Even more accurate: Use actual T_2 (normally not necessary)

Typical choice: $\tau_{c1} = \sigma\ \tau_{c2}$ where time scale separation $\sigma = 4\ to\ 10$.

Time scale separation is needed for cascade control to work well

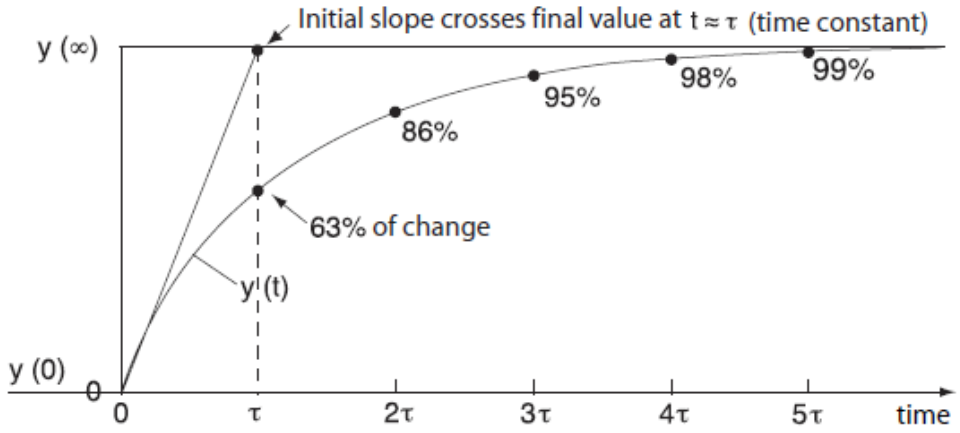
- Inner loop (slave) should be at least 4 times* faster than the outer loop (master)
  - This is to make the two loops (and tuning) independent.
  - Otherwise, the slave and master loops may start interacting
    - The fast slave loop is able to correct for local disturbances, but the outer loop does not «know» this and if it's too fast it may start «fighting» with the slave loop.
- But recommend 10 times faster, $\sigma \equiv \frac{\tau_{c1}}{\tau_{c2}} = 10$.
  - A high σ is robust to gain variations (in both inner and outer loop)
  - The reason for the upper value (σ =10) is to avoid that control gets too slow, especially if we have many layers

* Shinskey (Controlling multivariable processes, ISA, 1981, p.12)

Motivation for factor 4 for time scale separation

Response y(t) (lower layer) to step change in setpoint (coming from upper layer):

$$\Delta y(t) = \left(1 - e^{-t/\tau_{c2}}\right) \Delta y_s$$

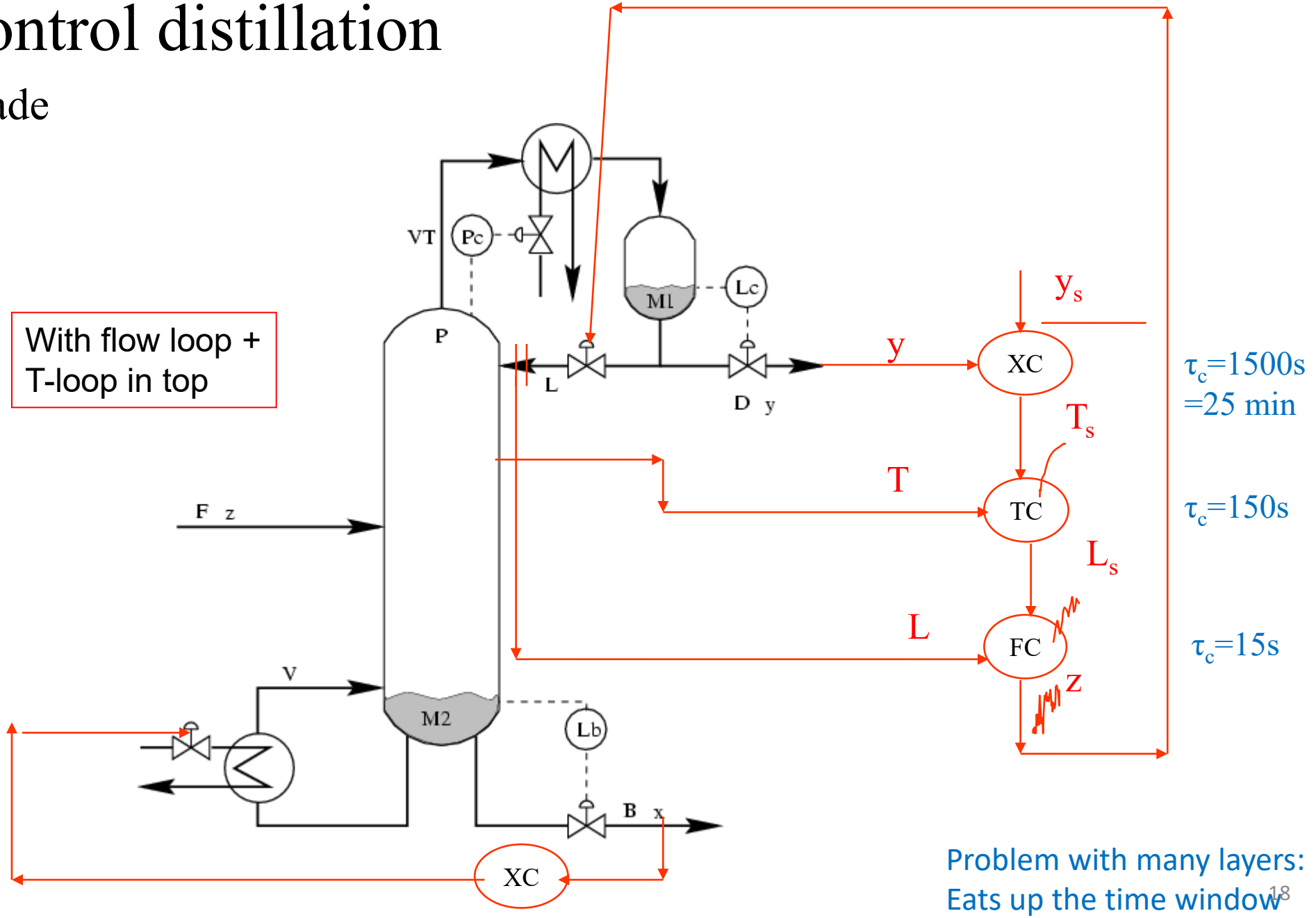


t/τ	$1 - e^{-t/\tau}$	Value	Comment
0	$1 - e^{0} =$	0	
0.1	$1 - e^{-0.1} =$	0.095	
0.5	$1 - e^{-0.5} =$	0.393	
1	$1 - e^{-1} =$	0.632	63% of change is reached after time $t = \tau$
2	$1 - e^{-2} =$	0.865	
3	$1 - e^{-3} =$	0.950	
4	$1 - e^{-4} =$	0.982	98% of change is reached after time $t = 4\tau$
5	$1 - e^{-5} =$	0.993	
∞	$1 - e^{-\infty} =$	1	

Lower layer has for any practical purpose converged (98%) after time 4 τ_{c2}.

Cascade control distillation

3 layers of cascade



Counteracting nonlinearity using cascade control

Example: Consider slave flow controller with u = z (valve position) and y_2 = q (flow)

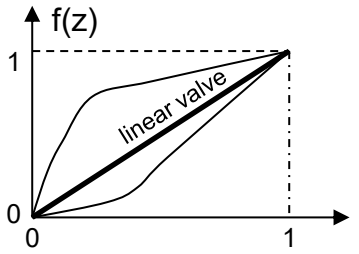
- Nonlinear valve with varying gain k_2: $G_2(s) = k_2(z) / (\tau_2 s+1)$
- Slave (flow) controller K_2: PI-controller with gain K_{c2} and integral time $\tau_I = \tau_2$ (SIMC-rule). Get

$$L_2 = K_2(s)G_2(s) = \frac{K_{c2}k_2}{\tau_2 s}$$

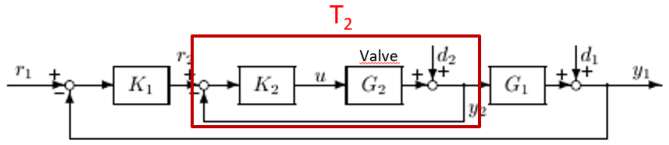
- With slave controller: Transfer function from y_{2s} to y_2 (as seen from outer loop):

$$T_2 = L_2/(1+L_2) = 1/(\tau_{C2}\, s + 1), \text{ where } \tau_{C2} = \tau_2/(k_2\, K_{c2})$$

- Important: Gain for T_2 is always 1 (independent of k_2) because of intergal action in the inner (slave) loop
- But: Gain variation in k_2 (inner loop) translates into variation in actual closed-loop time constant τ_{C2}. This may effect the master loop:
  - The master controller K_1 is designed based on G_1T_2.
  - A smaller process gain k_2 results in a larger τ_{C2} and thus a large effective delay, which mat be bad.
    - Recall $T_2 \approx e^{-\Theta 2 s}/(\tau_{c2}s+1) \approx e^{-(\Theta 2+\tau c2)s}$
  - However, if the time scale separation σ is sufficiently large, the variations in τ_{C2} will not matter



$k_2(z)$ = slope = df/dz



G_1T_2 = «Process» for tuning master controller K_1

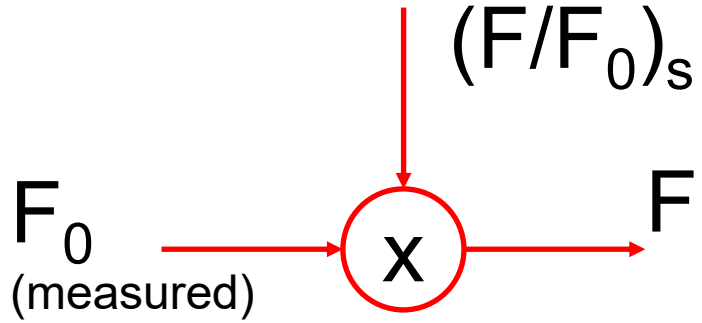
E2. Ratio control

Often viewed as special case of feedforward.

- BUT it doesn't need model
- Based on process insight: Scale all flows by same factor gives constant quality

Example: Process with two feeds F_0 (d) and F (u), where ratio should be constant.

Use* multiplication block**:



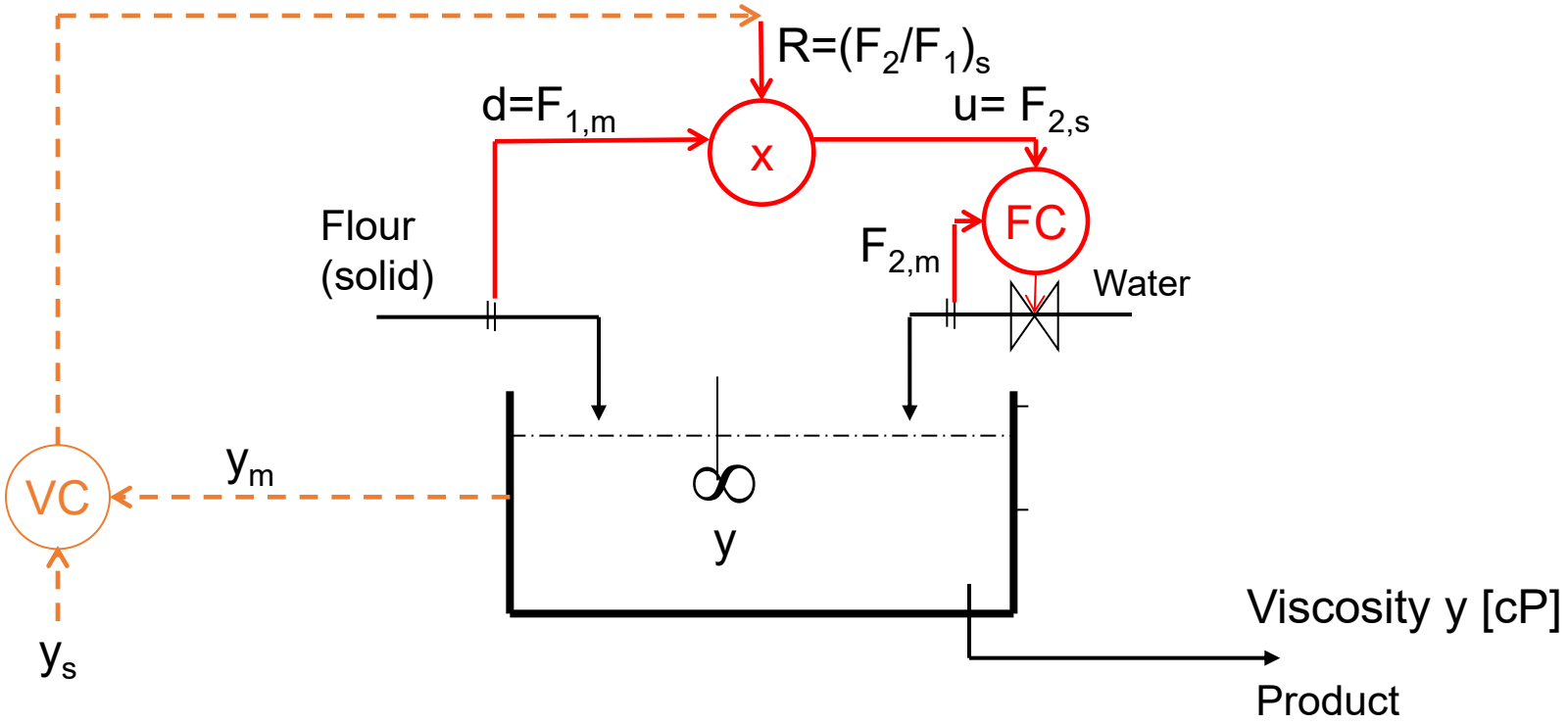
- F is usually setpoint to a flow controller
- Often $(F/F_0)_s$ is adjusted using feedback control in a cascade fashion.

* Don't use division element

** Multiplication block is sometimes called «ratio station» (bad name)

EXAMPLE: CAKE BAKING MIXING PROCESS

RATIO CONTROL with outer feedback trim (to adjust ratio setpoint)



Multiplication element (x) = Recipe (cook book) (feedforward)
VC = feedback correction by tasting

Ratio control

- Keep ratio R (between extensive variables) constant in order to keep property y constant
  - Feedforward: R=u/d
  - Decoupling: $R=u_1/u_2$
    - u,d: extensive variables
  - y: (any!) intensive variable
- Assumes that the «scaling property» holds
  - Based on physical insight
- Don't really need a model (no inverse as in «normal» feedforward!)
  - Setpoint for R may be found by «feedback trim»
- Scaling property holds for mixing and equilibrium processes
    - Rato control is almost always used for mixing of reactants
  - Requires that <u>all</u> extensive variables are scaled by same amount
    - So does <u>not</u> hold for heat exchanger (since area A is constant) or non-equilibrium reactor (since volume V is constant)
    - L/F constant is <u>not</u> good for distillation column with saturated (max) heat input (V)

Theoretical basis of ratio control

Ratio control: Theoretical basis and practical implementation

Sigurd Skogestad

Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), Trondheim, Norway

DRAFT submitted to JPC on May 21, 2025

Abstract

Ratio control is the oldest control approach, dating back thousands of years (think of food recipes), but despite this, there exists no theoretical basis for its use. It is widely used in the process industry, in particular, for mixing processes and chemical rectors. It is sometimes viewed as a special case of feedforward control. However, feedforward control requires an explicit process model, but this is not needed for ratio control. Instead, ratio control is based on the physical insight that scaling all flows to keep constant flow ratios will result in constant product properties, and this scaling assumption is discussed in detail in the paper. Furthermore, the ratio setpoint may be set by an outer feedback loop, again without the need for a process model. The paper also discusses the practical implementation of ratio control, including dual ratio control for the case with saturation and cross-limiting control for keeping one component (typically oxygen) in excess during dynamic transients. Finally, it is shown that the multiplication trick proposed to avoid the limbo-effect for dual ratio control applies more generally to all split-range control solutions.

Keywords: control architecture, control structure design, feedforward control, PID control, advanced regulatory control,

6. Conclusion

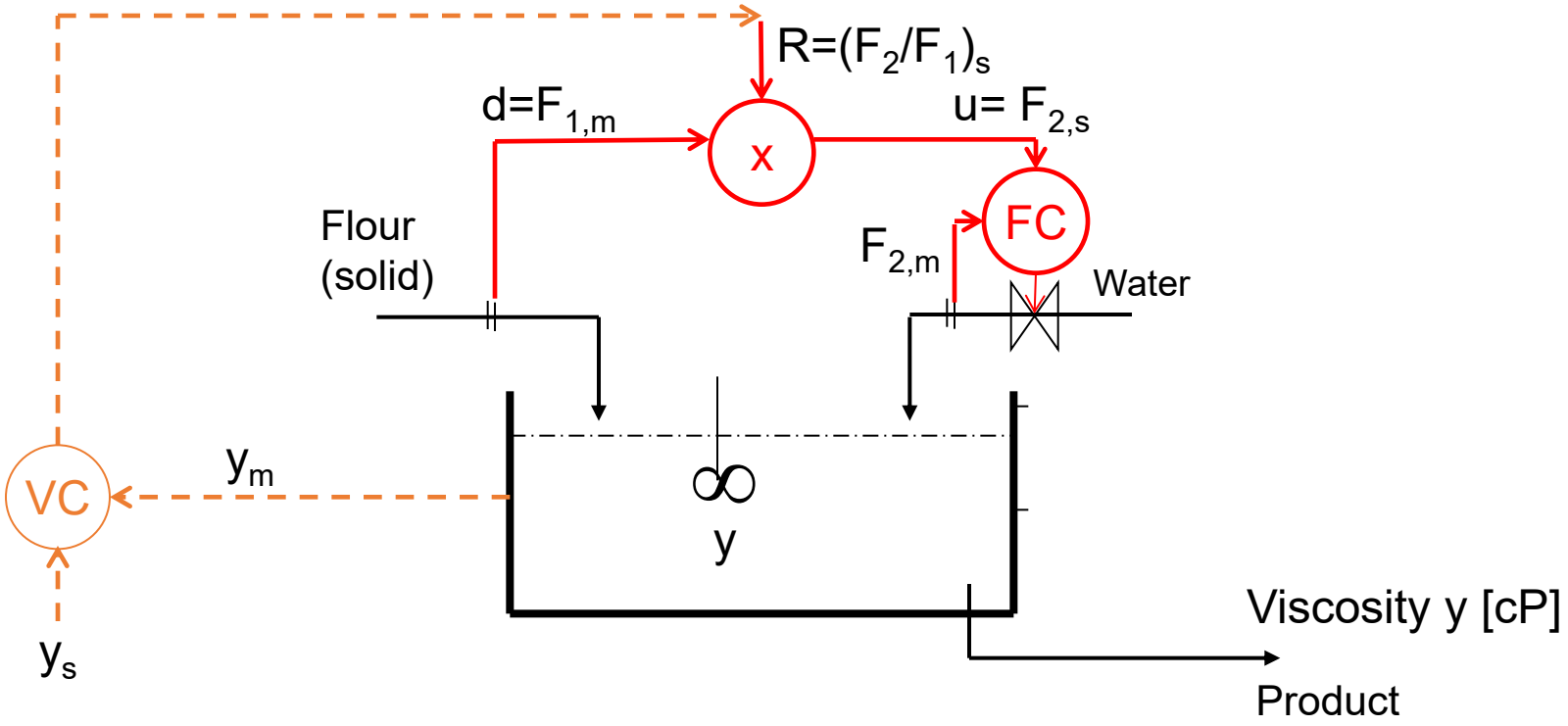
Ratio control is very simple to use and it gives nonlinear feedforward action without needing an explicit process model. It is almost always used for chemical processes to set the ratio of the reactant feed streams. Ratio control is sometimes viewed as a special case of feedforward control, but note that we do not need a model for the controlled property y for ratio control, whereas such a model is needed for feedforward control.

The theoretical basis for ratio control is the scaling assumption which says that we get the same steady-state solution if we increase all extensive variables (flows and heat rates) by the same factor compared to a basis. Similar to the use in thermodynamics, the scaling assumption holds for equilibrium systems with constant efficiencies.

The scaling assumption is formulated mathematically in (2). From this we derived the following rules for the use of ratio control:

- (R1) The controlled variable y is implicitly assumed to be an intensive variable, for example, composition, density, viscosity, taste or temperature.
- (R2) The system must satisfy the scaling assumption (2).
- (R3) Since all extensive variables must be scaled by the same factor k, there can only be one independent extensive disturbance variable. This variable is sometimes called the "basis", "wild variable", "master variable", "flow disturbance" or "throughput manipulator" (TPM).
- (R4) If the system has n independent extensive variables X_i, then from (2) we need to manipulate $n-1$ of these variables to keep the $n-1$ ratios constant (or more generally, $n-1$ dependent intensive variables y_i). For a change (disturbance) in the throughput (basis, wild flow), this will result in keeping *all* dependent intensive variables constant, including the controlled variable(s) y (at steady state).

LINEARITY OF RATIO CONTROL



Note : <u>This way</u> of implementing ratio control makes it easy to tune the outer feedback loop (CC: composition controller) because the gain from MV = R_s to CV=y does not depend on disturbance d=F_1.

Ratio control with feedforward: This implementation may be used if there are measured disturbances in feed quality

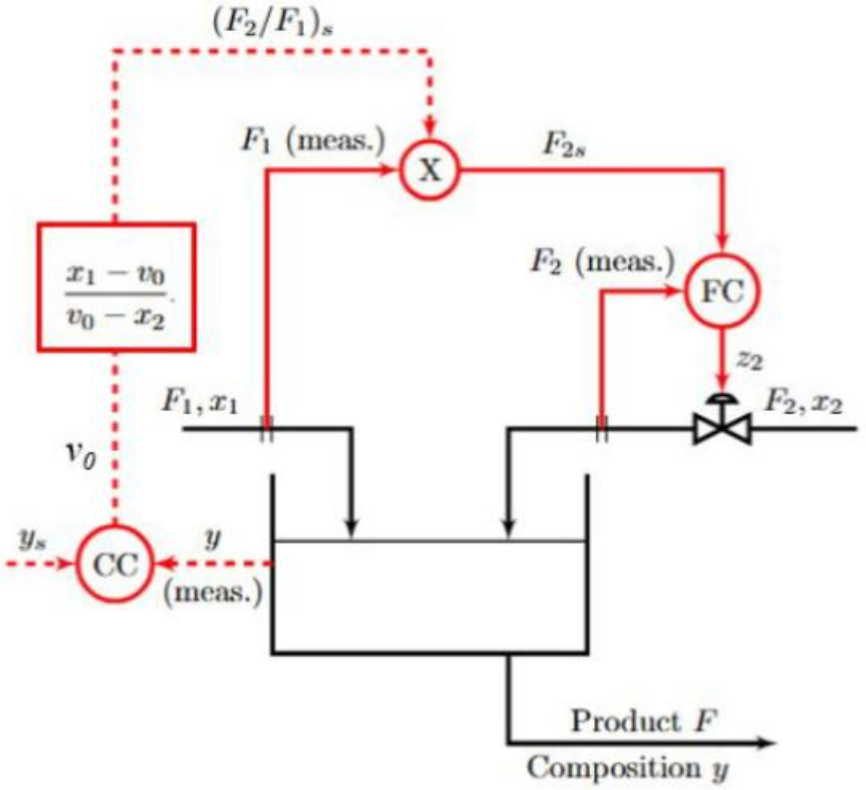


Figure 14: Improved ratio control scheme for mixing process using transformed input v_0. The feed mass fractions x_1 and x_2 that enter the computation block, need to be measured or estimated.

From the steady-state component material balance, we have that y is the weighted average of the feed fractions (recall (4))

$$y = f_0(u, d) = \frac{x_1 F_1 + x_2 F_2}{F_1 + F_2} \tag{12}$$

Note that $u = F_2$. The transformed input is defined as the right-hand side of this equation, $v_0 = f_0(u, d)$. Note that v_0 is the output from the feedback controller. Inverting (12), we find how the input $u = F_2$ depends on the transformed input v_0:

$$u = F_2 = f_0^{-1}(v_0, d) = \frac{x_1 - v_0}{v_0 - x_2} F_1 \tag{13}$$

Theoretical basis: Transformed input $v_0 = f_0(u, d)$ chosen equal to RHS of static model $y = f_0(u, d)$. Resulting model for outer controller (CC): $y = I\, v_0$ (linear, decoupled and perfect disturbance rejection!). Seems too simple to be true, but it works!

Valve position control (VPC)

- One CV
- Extra MV (input)

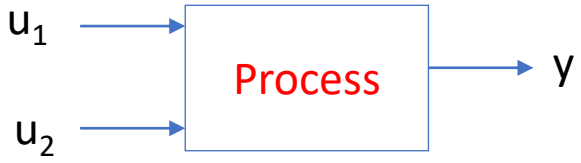
Two different cases of VPC:

- E3. “Conventional VPC”.
  - Use extra MV to improve dynamics
  - Both MVs are used all the time
-
- E7. ”Split VPC”
  - Use extra MV only when “primary” input approaches saturation (static)

E3. "Conventional" VPC

u_2 = primary input for steady-state control of CV
(but u_2 is poor for directly controlling y
- e.g. time delay or u_2 is on/off)

u_1 = extra dynamic input for fast control of y



3.4. Input (valve) position control (VPC) to improve the dynamic response (E3)

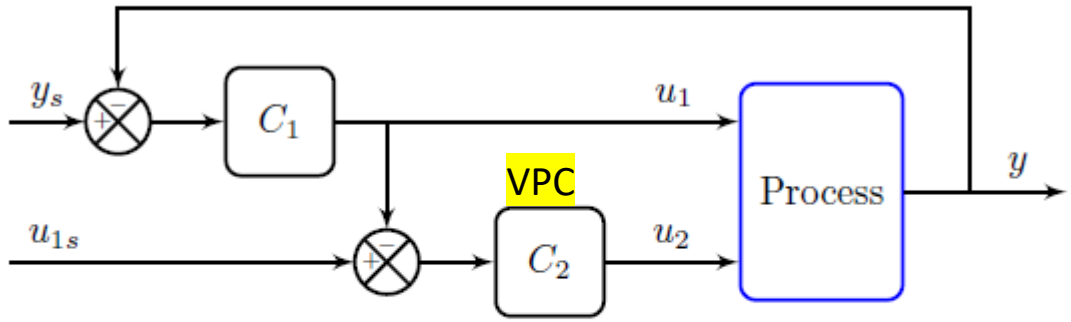
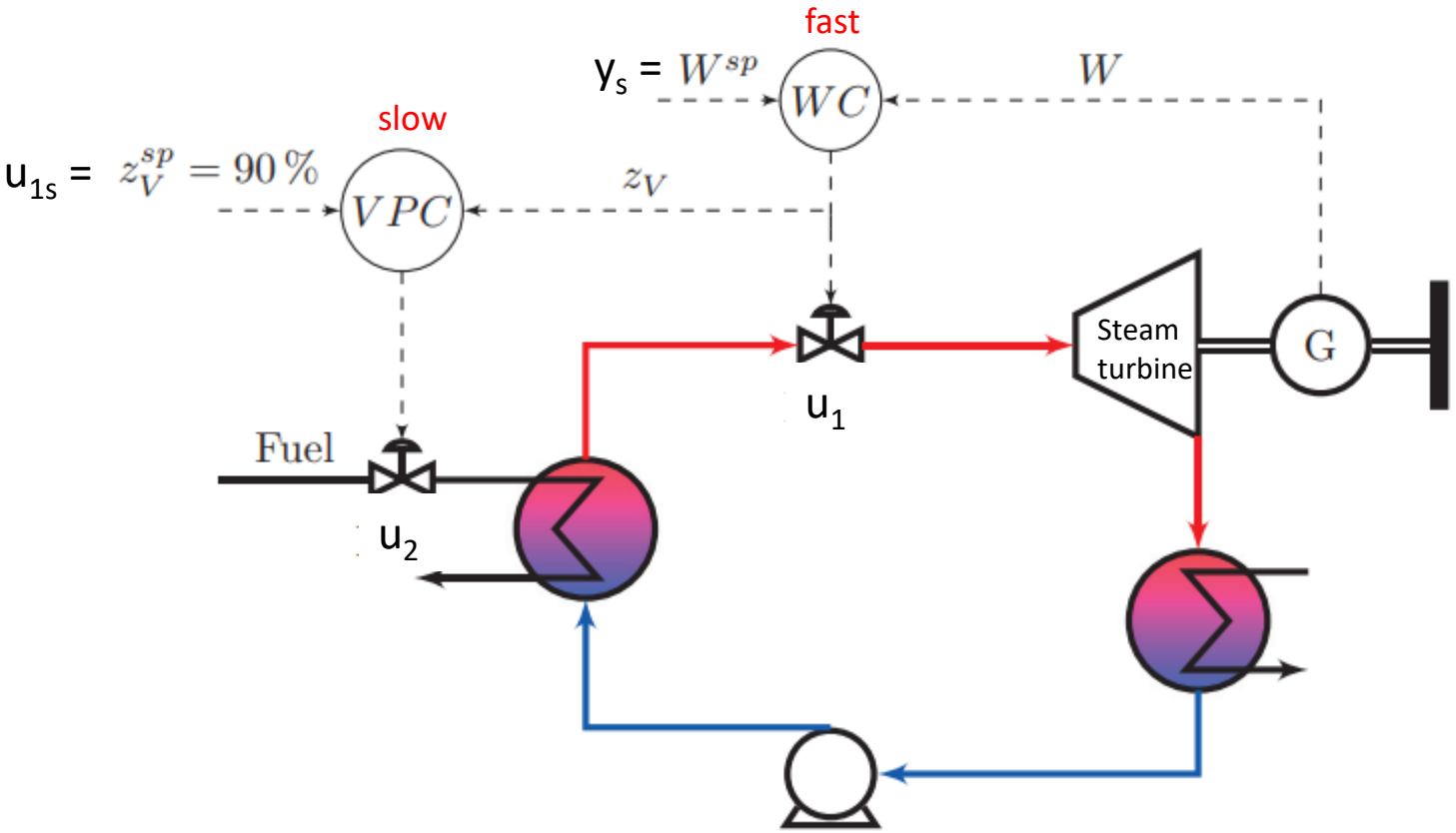


Figure 12: Valve (input) position control (VPC) for the case when an "extra" MV (u_1) is used to improve the dynamic response. A typical example is when u_1 is a small fast valve and u_2 is a large slower valve.
C_1 = fast controller for y using u_1.
C_2 = slow valve position controller for u_1 using u_2 (always operating).
u_{1s} = steady-state resting value for u_1 (typically in mid range. e.g. 50%).

Alternative term for dynamic VPC:
- Mid-ranging control (Sweden)

Example 1: Large (u_2) and small valve (u_1) (in parallell) for controlling total flowrate (y=F)
- The large valve (u_2) has a lot of stiction which gives oscillations if used alone for flow control
- The small valve (u_1) has less stiction and gives good flow control, but it's too small to use alone

Example 2 VPC: Power plant control



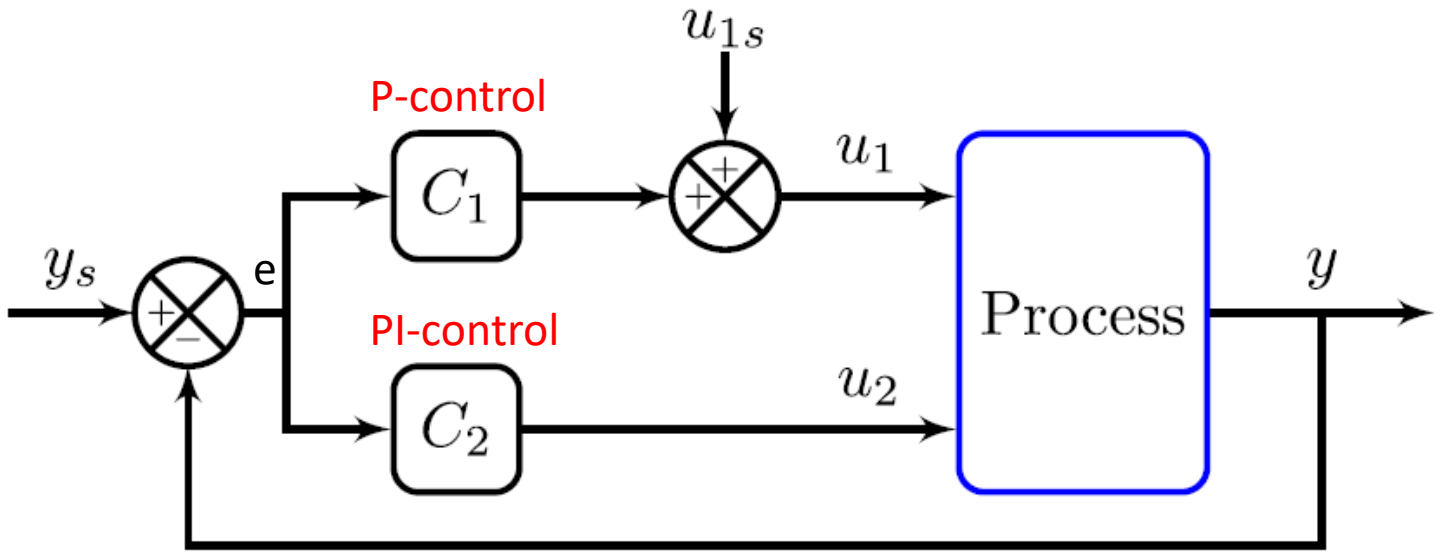
y = power W [MW]
u_1 = steam valve z (fast transient effect)
u_2 = Fuel (slow static effect),

VPC: Uses u_2 to control u_1 (u_{1s} = 90%)

- u_1 returns to u_{1s}=90% because of VPC
- Both controllers may be PI
- Need time scale separation: VPC slow

Optimal operation and control of heat to power cycles: A new perspective from a systematic plantwide control approach
Cristina Zotica , Lars O. Nord, Jenö Kovács , Sigurd Skogestad. Computers & Chemical Engineering,. Volume 141, 4 October 2020, 106995

Alternative to VPC: Parallell control

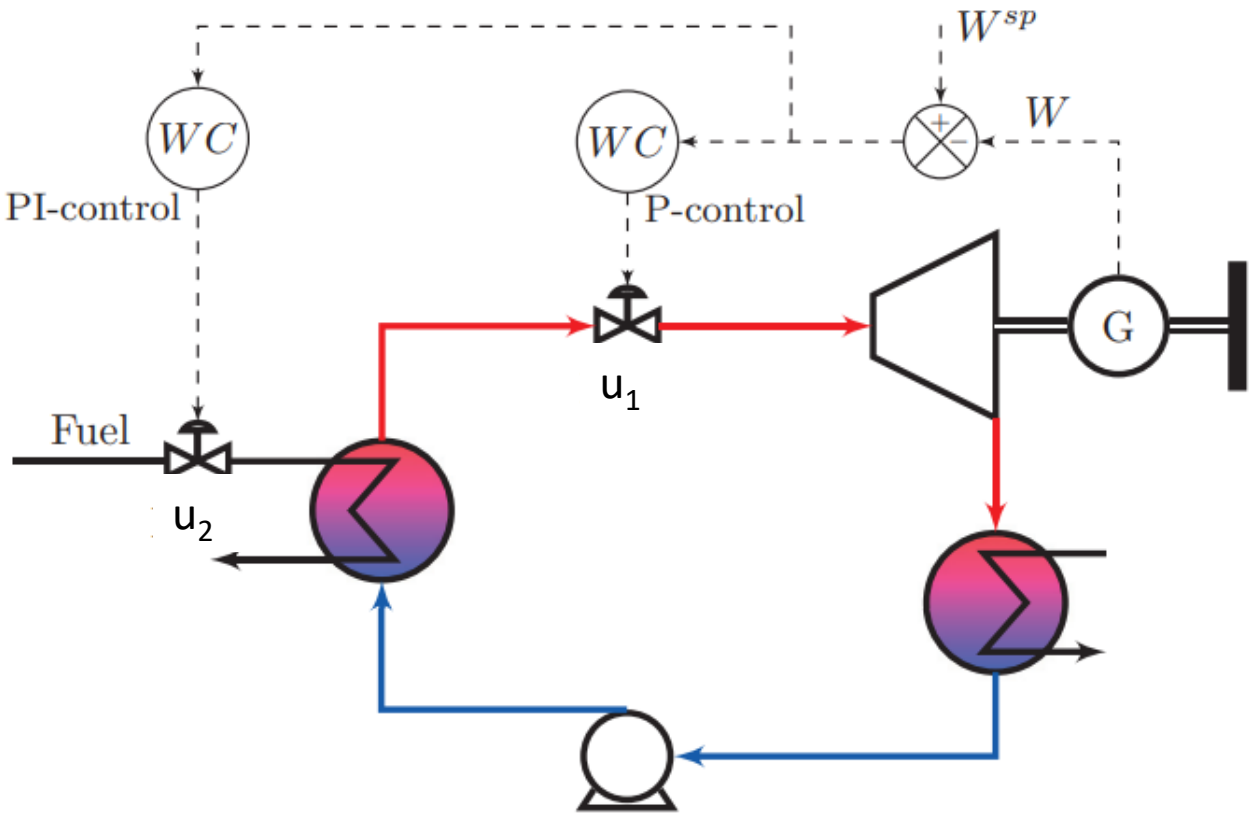


At steady state, e=0 because of the I-action in C_2, so the output from C_1 (P-control) is zero and u_1 returns to $u_1=u_{1s}$.

Fig. 13. Parallel control to improve dynamic response – as an alternative to the VPC solution in Fig. 12.
The "extra" MV (u_1) is used to improve the dynamic response, but at steady-state it is reset to u_{1s}. The loop with C_2 has more integral action and wins a steady state.

The advantage with valve position control compared to parallel control is that the two controllers in Figure 12 can be tuned independently (but C_1 must be tuned first) and that both controllers can have integral action. On the other hand, with some tuning effort, it may be easier to get good control performance for y with parallel control.

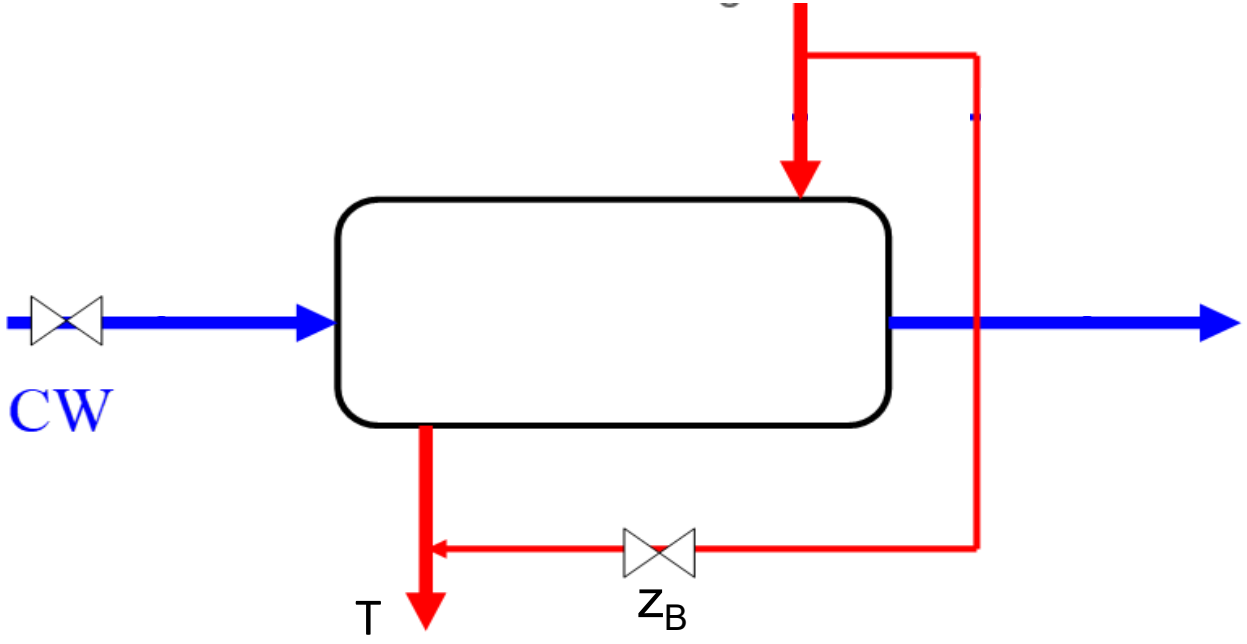
Alternative to VPC: Parallel control



y = power W [MW]
u_1 = steam valve z (Fast P-control, $u_1=u_{10} + K_c\, e$)
u_2 = Fuel (Slow PI-control)

- u_1 returns to bias u_{01}=90% because PI controller gives e = W_s-W = 0 at steady state

Example 3: Heat exchanger with bypass

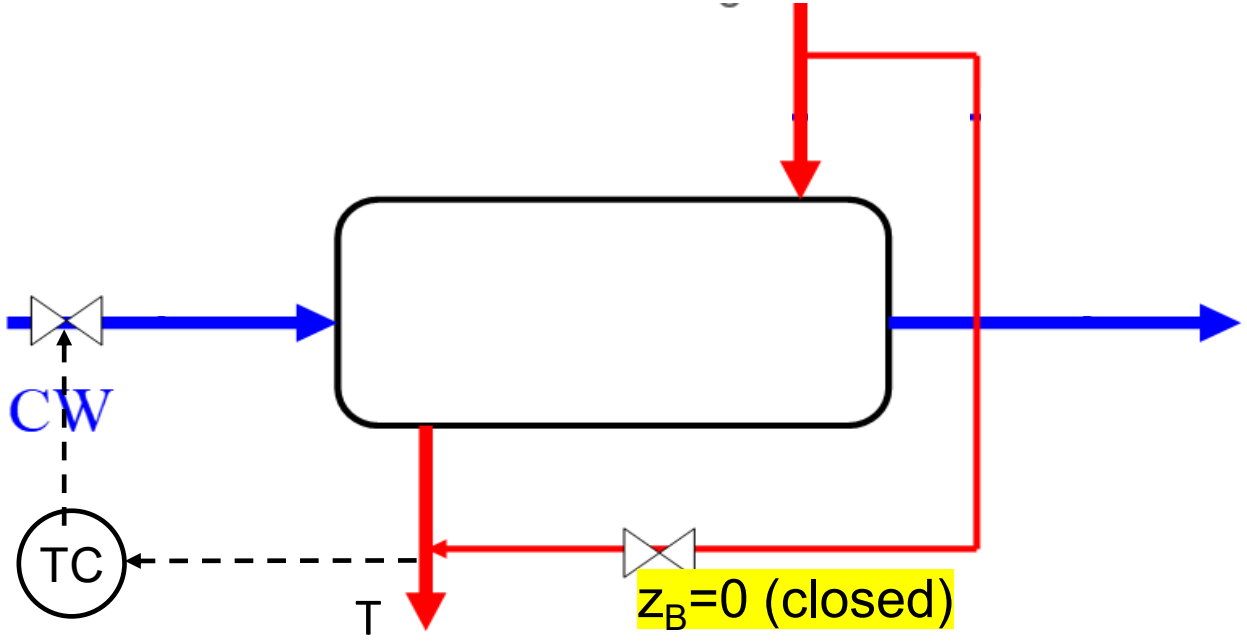


Want tight control of y=T.

- $u_1=z_B$ (bypass)
- u_2=CW

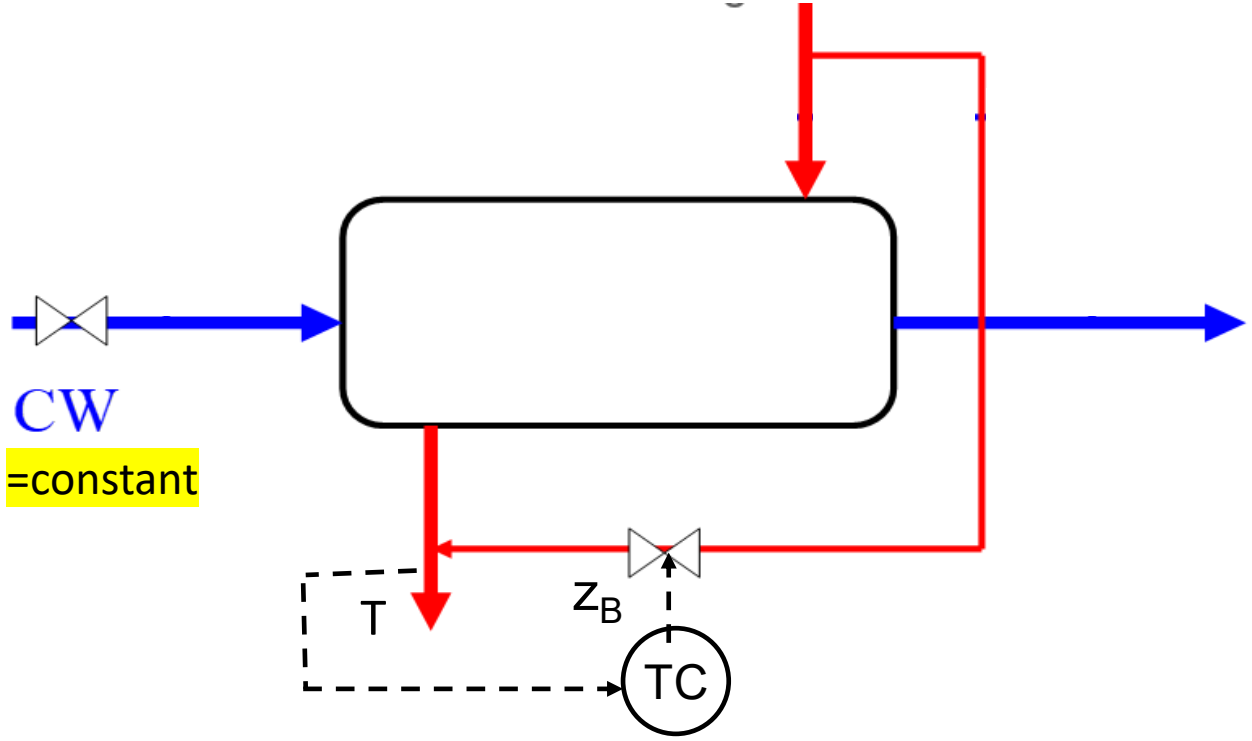
Proposed control structure?

Attempt 1. Use u_2=cooling water: TOO SLOW



Attempt 2. Use $u_1=z_B$=bypass. SATURATES

(at z_B=0=closed if CW too small)

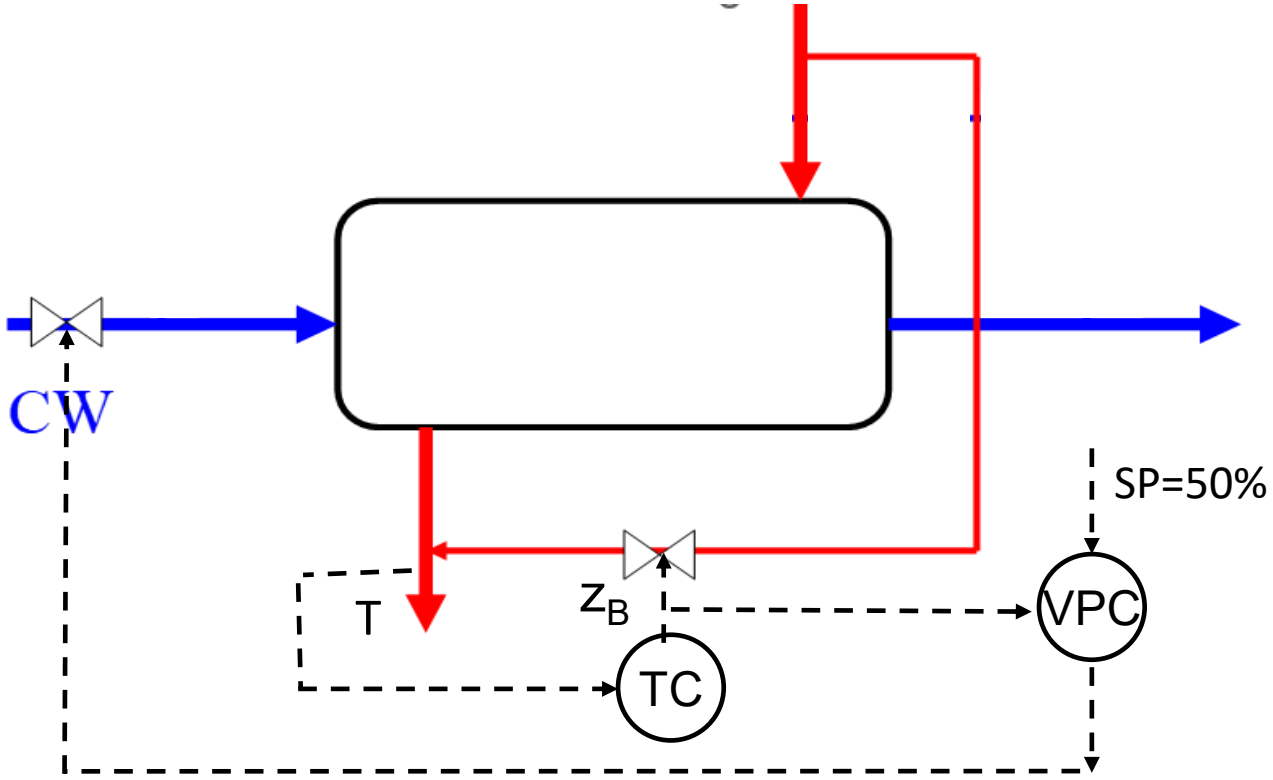


Advantage: Very fast response (no delay)

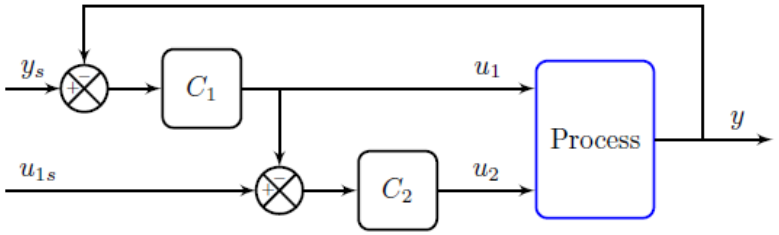
Problem: z_B is too small to cover whole range

+ not optimal to fix at large bypass (waste of CW)

Attempt 3 (recommended): VPC



- Fast control of y: $u_1 = z_B$
- Main control (VPC): u_2=CW (slow loop)
- Need time scale separation between the two loops

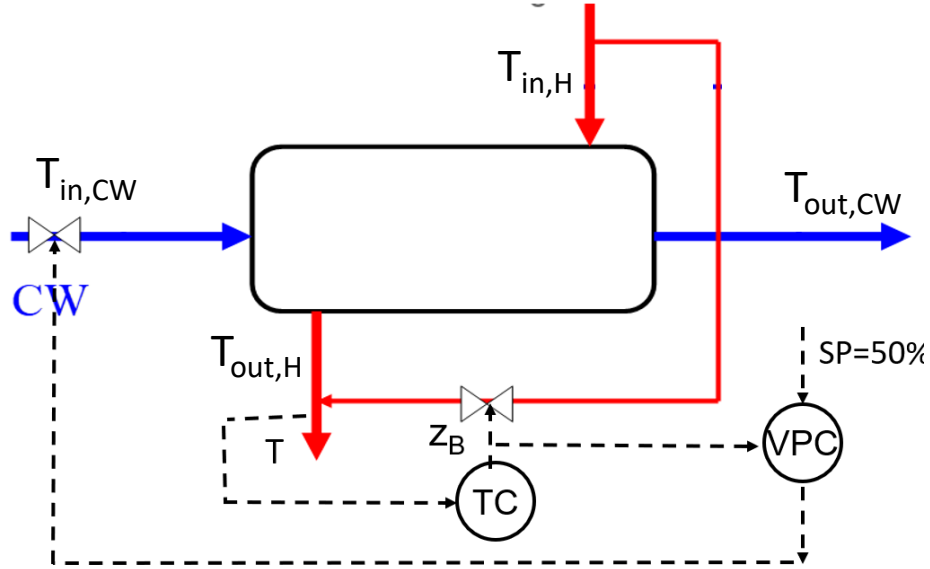


Comment on heat exchanger example

- The above example assumes that the flows on the two sides are «balanced» (mc_P for cooling water (CW) and hot flow (H) are not too different) such that both the bypass flow (u1) and CW flow (u2) have an effect on T (CV)

- There are two «unbalanced» cases, which is when we have «pinch» in the heat exchanger ends:
  - If CW flow is small, then $T_{out,CW}$ will always approach $T_{in,H}$, so from a total energy balance, the bypass will have almost zero *steady-state* effect on T (but it has a dynamic effect so TC+VPC may still work)
  - If CW flow is large, then $T_{out,H}$ (before bypass mixing point) will always approach $T_{in,CW}$, so CW will have almost zero effect on T (*both* steady state and dynamically) (in this case there is no point in using VPC)

- This illustrates that heat exchanger may behave very nonlinearly, and a good control structure for one heat exchanger case, may not work well for another case



VPC with one MV:

Anti-slug Stabilizing control with resetting of MV

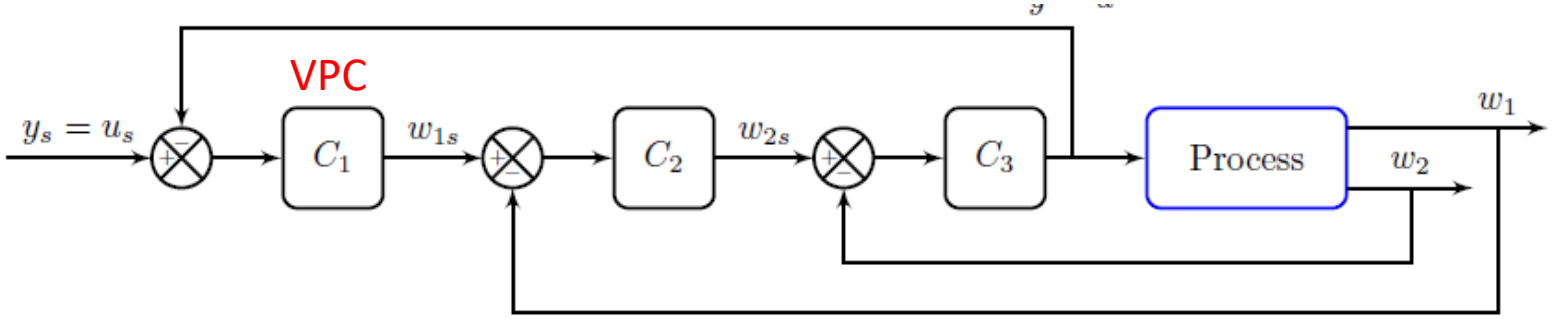
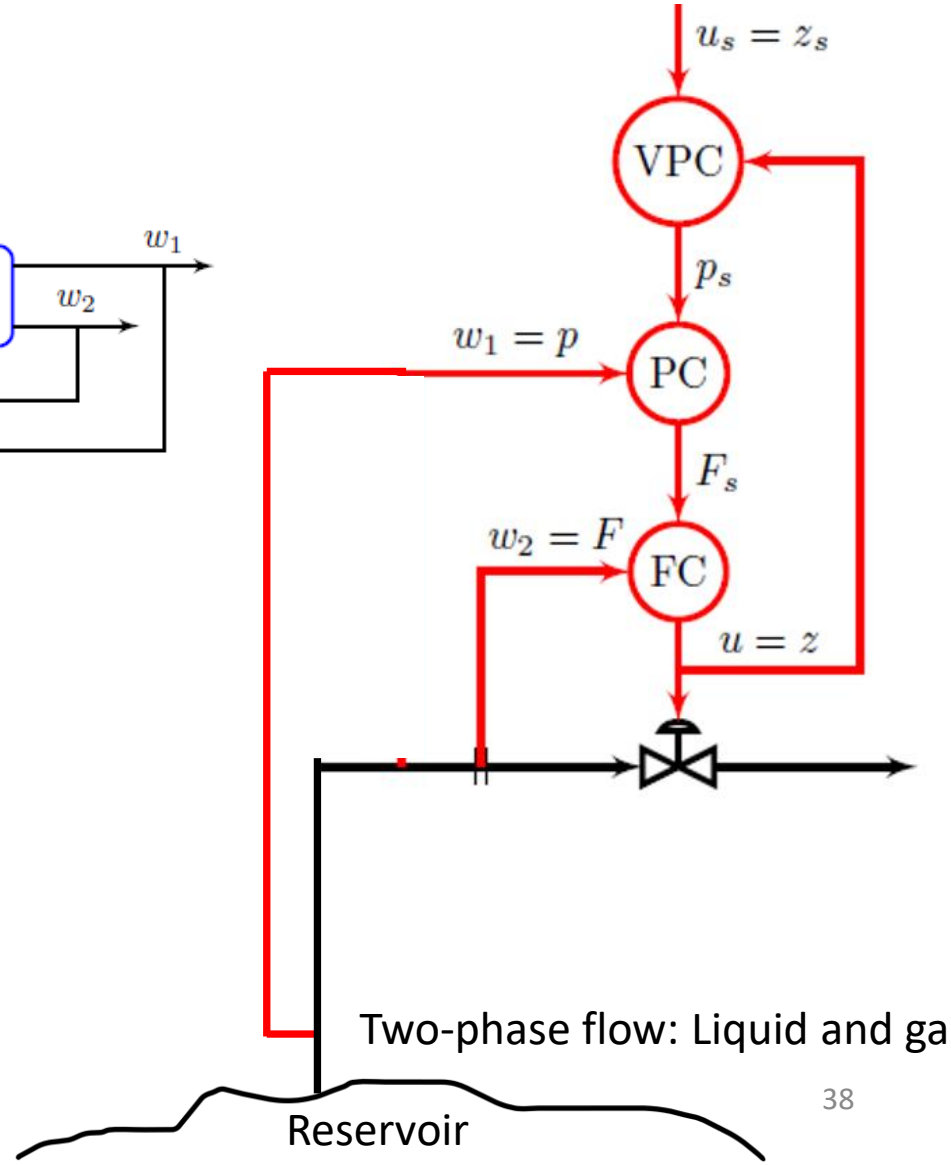
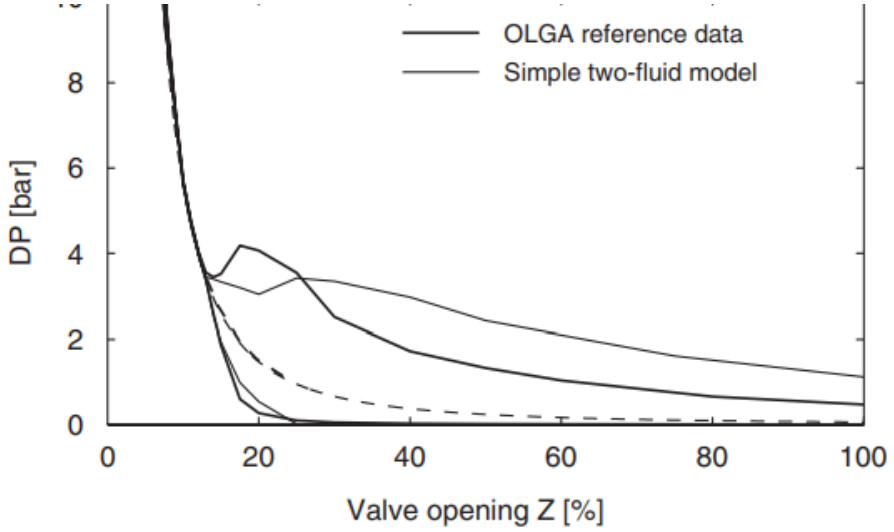


Figure 14: Stabilizing control of variable w_1 combined with valve position control (VPC) for u (=valve position) and inner flow controller ($w_2 = F$).
It corresponds to the flowsheet in Figure 15 with $w_1 = p$ (pressure), C_1 = outer VPC (slow), C_2 = stabilizing controller (fast), C_3 = inner flow controller (very fast).
Note that the process variables (w_1, w_2) have no fixed setpoint, so they are "floating".

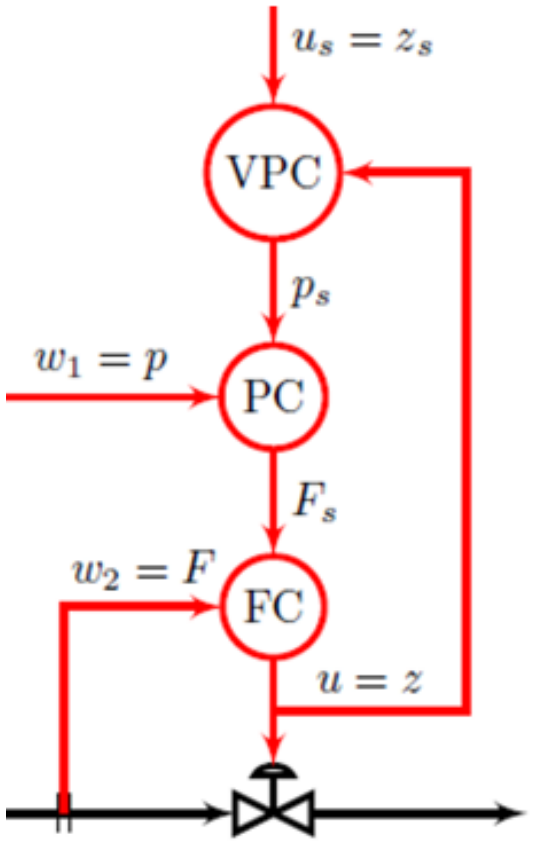
Note: u is both an MV and a CV



More on Anti-slug control cascade



E. Storkaas and S. Skogestad, "Controllability analysis of two-phase pipeline-riser systems at riser slugging conditions", *Control Engineering Practice*, **15**, 567-581 (2007)



The objective is to stabilize the system in a nominally unstable non-slugging operation point.

VPC (slow): Want valve as open as possible to maximize production. BUT may have problems stabilizing flow if it is too open (Storkaas et al., 2001)

PC (anti-slug controller): Place pressure sensor at sea bottom for vertical riser. Get inverse response if close to valve, and this makes stabilization difficult (Storkaas et al., 2001)

FC (fast): Linearizes system (both valve characteristic and effect of changing DP) and removes «fast» slugs

All controllers are PI. Need time scale separation

E.Storkaas, S. Skogestad and V. Alstad, "Stabilizing of desired flow regimes in pipelines " AIChE Annual meeting, Paper 287d, Reno, Nevada, November 5-9, 2001

Inside the limit cycle there exists an unstable stationary operating point. The gain of the system approaches zero as the valve opening is increased, at the same time the unstable poles move further into the right half plane. Because of this, stabilizing the system with large valve openings is not practically possible. However, the losses in terms of pressure drop are small even if one operates at quite low valve openings.

The pressure sensor used as measurement for control should be placed in the lower part of the system. With the pressure sensor located in the riser, RHP-zeros close to the imaginary axis limits the bandwidth of the control system, making stabilization of the system difficult.

Stabilizing the system consists of two tasks. First the limit cycle has to be broken and the system brought to the desired operating point. Nonlinear aspects will probably be important for achieving this. Then the system has to be stabilized and kept in this (unstable) state. A linear controller should be sufficient for this.

VPC to reset input u

- If the underlying process is unstable, then the instability will result in an inverse response when attempting to reset u.
- Proof: G(s) has unstable pole at s=p.
  - Then transfer function KS from u_d (at input) to u has unstable zero (=inverse response) at s=p.
  - This is because G(p)=infinity so $S(p)=(I+G(p)K(p))^{-1} = 0$.

Example: Stabilize bycycle

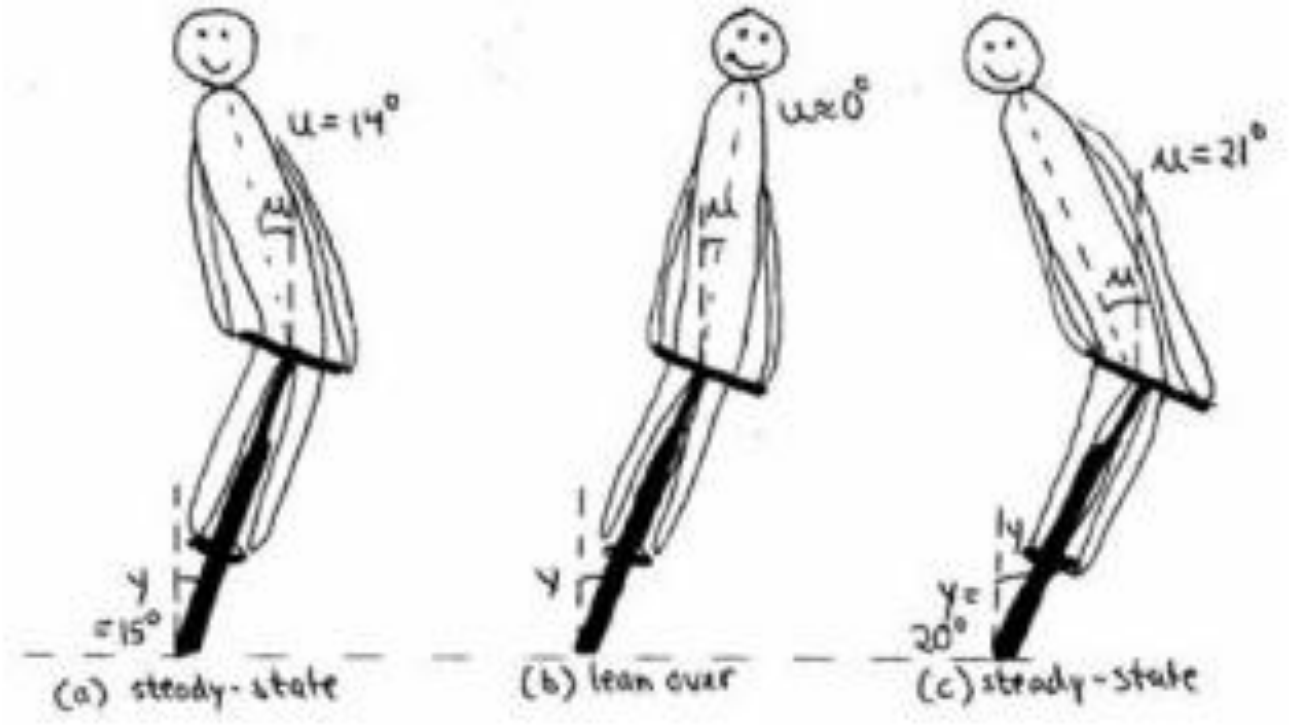


Fig. 2. Inverse response for a bicycle caused by an underlying instability

Consider Figure 2 where the aim is to tilt the bike from an initial angle $y = 15°$ (Fig. 2a) using your body (u) to an angle $y = 20°$ (Fig. 2c). Because of the inverse response, you first have to tilt your body in the direction of the tilt to start the movement (Fig. 2b). Eventually, you will have to move your body back to restore balance. This inverse response will be slower the greater the angle y, changing the angle while keeping balanced gets progressively slower as the tilting angle is increased.

Comment: Another example is a motorcycle where tilting is required for making turns.

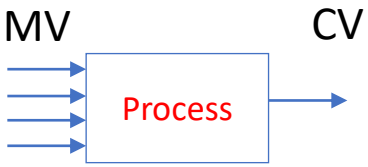
Espen Storkaas and Sigurd Skogestad, "Cascade Control of unstable systems with application to stabilization of slug flow", IFAC-symposium Adchem'2003, Hong Kong, Presented: Jan. 2004 (original conference date: June 2003.)

Switching of MVs and CVs

Constraint switching (because it is optimal at steady state)

A. MV-MV switching

- Use one MV at a time
- Three alternatives



B. CV-CV switching

- Control one CV at a time
- Always selector (or similar)

• MV-CV switching

- MV saturates so must give up CV

C. Simple («do nothing»)

D. Complex (repairing of loops)

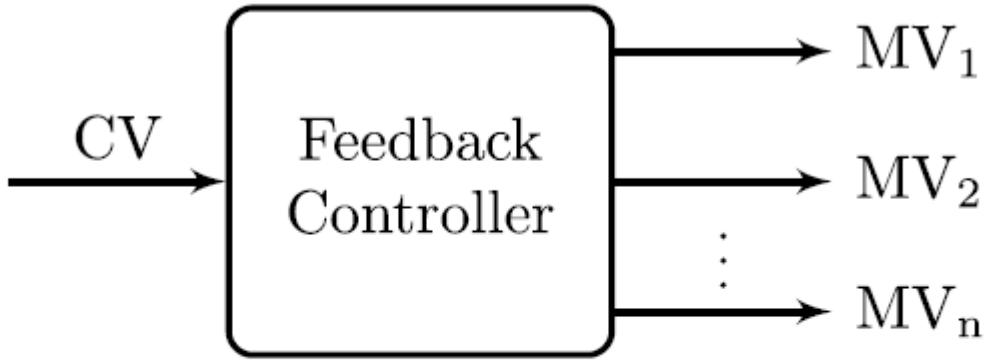


Fig. 5. MV-MV switching is used when we have multiple MVs to control one CV, but only one MV should be used at a time. The block "feedback controller" usually consists of several elements, for example, a controller and a split range block.

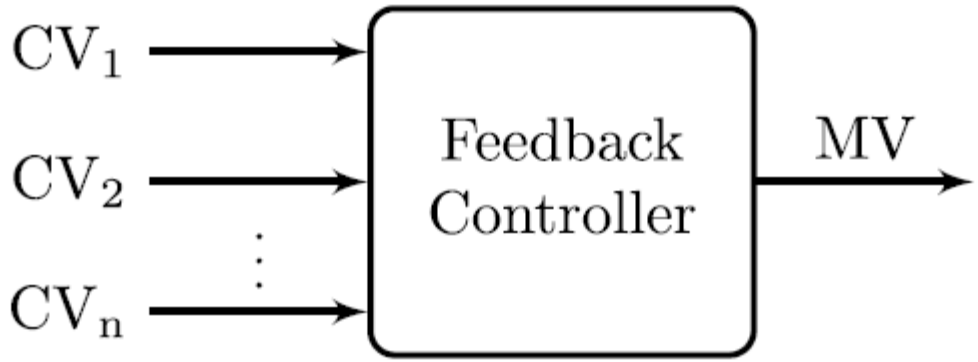


Fig. 6. CV-CV switching is used when we have one MV to control multiple CVs, but the MV should control only one CV at a time. The block "feedback controller" usually consists of several elements, typically several PID-controllers and a selector.

MV-MV switching

- One CV, many MVs (to cover whole steady-state range because primary MV may saturate)*
- Use one MV at a time

Three alternatives:

Alt.1 Split-range control (SRC)

- One controller

Alt.2 Split-parallel control:

- Several controllers (one for each MV) with different CV setpoints

Alt.3 Split valve position control (VPC)

- Use VPC when necessary to avoid saturation

Which is best? It depends on the case!

***Optimal Operation with Changing Active Constraint Regions using Classical Advanced Control, Adriana Reyes-Lua Cristina Zotica, Sigurd Skogestad, Adchem Conference, Shenyang, China. July 2018 ,**

Example MV-MV switching

- Break and gas pedal in a car
- Use only one at a time,
- «manual split range control»

MV-MV switching

E5. Split-range control (SRC)

[7]Note the blue saturation elements for the inputs in Figure 21 and other block diagrams. Saturation can occur for any physical input, but they are explicitly shown for cases where the saturation is either the reason for or part of the control logic. For example, in Figure 21, the reason for using u_2 is that u_1 may saturate.

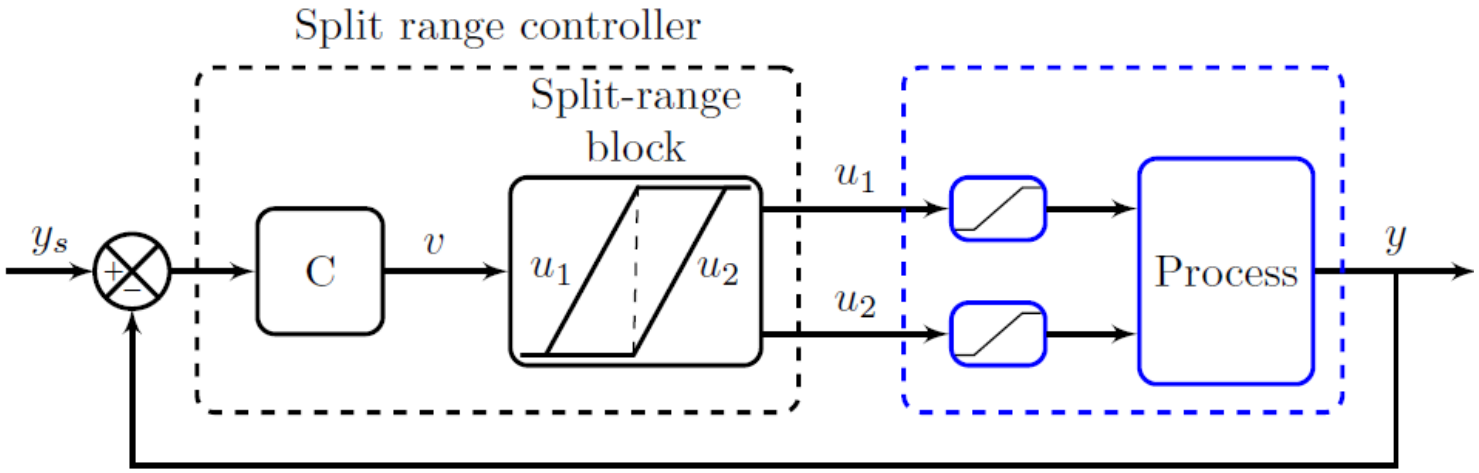


Figure 21: Split range control for MV-MV switching.

For MVs (u) that have same effect (same sign) on the output (y) (Fig. 21), we need to define the order in which the MVs will be used. This is done by the order in in the SR-block.

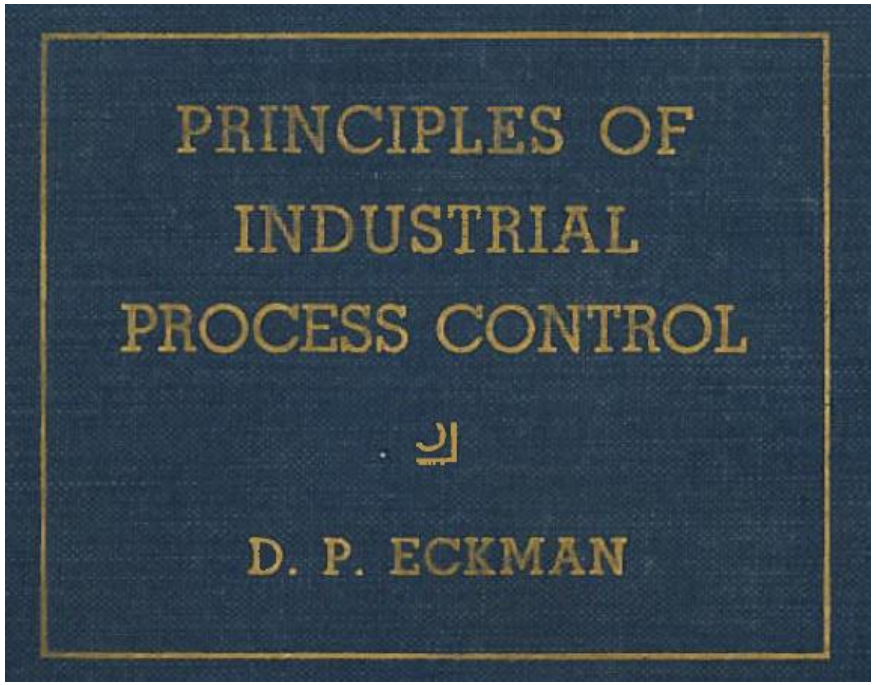
Example: With two heating sources, we need to decide which to use first (see next Example)

Advantage: SRC is easy to understand and implement!

Disadvantages:

1. Only one controller C ⇒ Same integral time for all inputs u_i (MVs)
   - Controller gains can be adjusted with slopes in SR-block!
2. Does not work well for cases where constraint values for u_i change

Split range control:
Donald Eckman (1945)



The temperature of plating tanks is controlled by means of dual control agents. The temperature of the circulating water is controlled by admitting steam when the temperature is low, or cold water when it is high. Figure 10–12 illustrates a system where pneumatic proportional control and diaphragm valves with split ranges are used. The steam valve is closed at 8.5 lb per sq in. pressure from the controller, and fully open at 14.5 lb per sq in. pressure. The cold water valve is closed at 8 lb per sq in. air pressure and fully open at 2 lb per sq in. air pressure.

If more accurate valve settings are required, pneumatic valve positioners will accomplish the same function. The zero, action, and range adjustments of valve positioners are set so that both the steam and cold water valves are closed at 8 lb per sq in. controller output pressure. The advantages gained with valve positioners are that

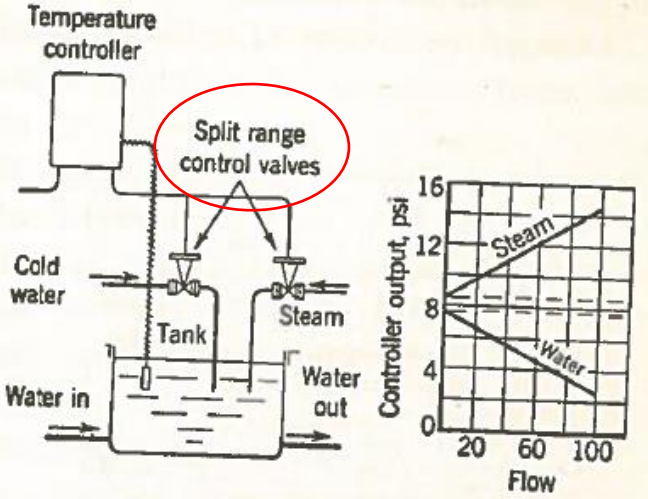
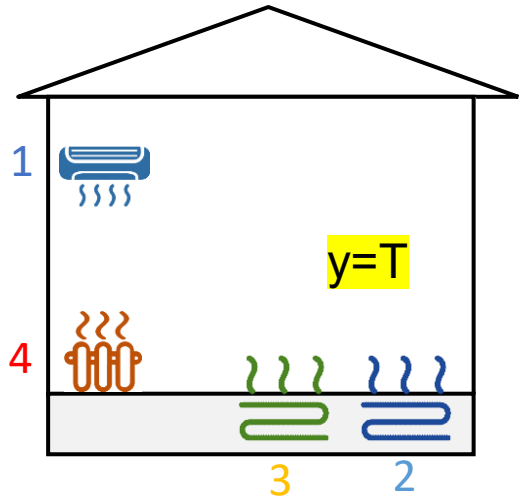


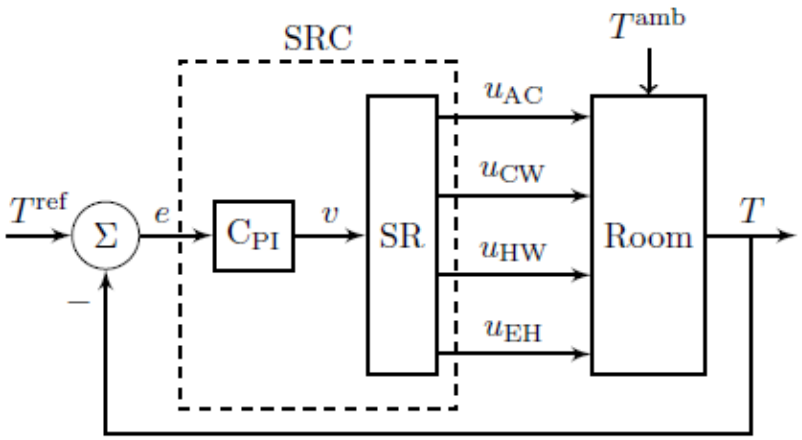
FIG. 10–12. Dual-Agent Control System for Adjusting Heating and Cooling of Bath.

Example split range control: Room temperature with 4 MVs

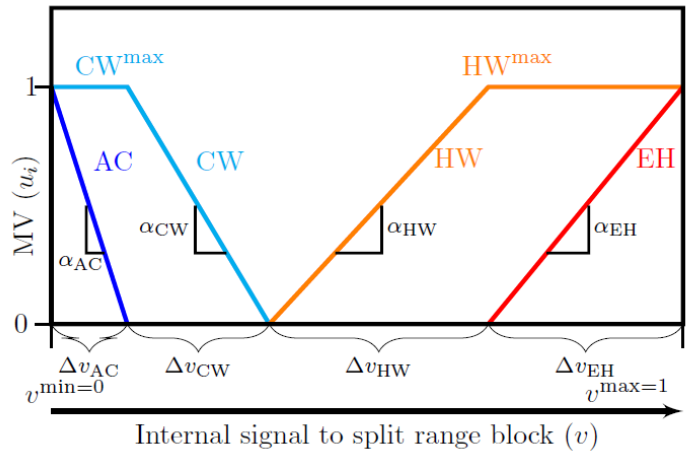


MVs (two for summer and two for winter):

1. AC (expensive cooling)
2. CW (cooling water, cheap)
3. HW (hot water, quite cheap)
4. Electric heat, EH (expensive)

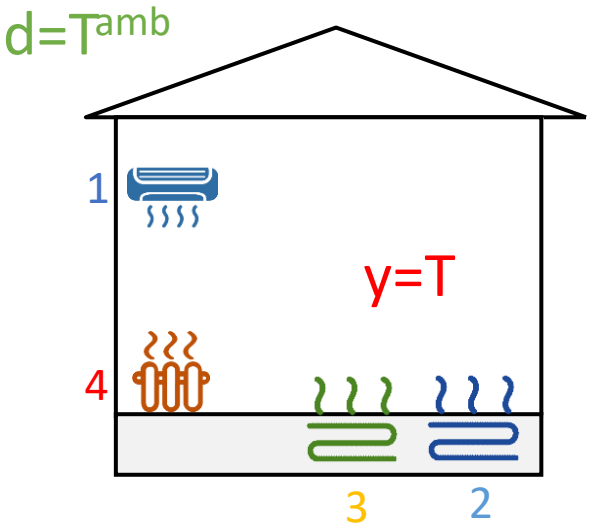


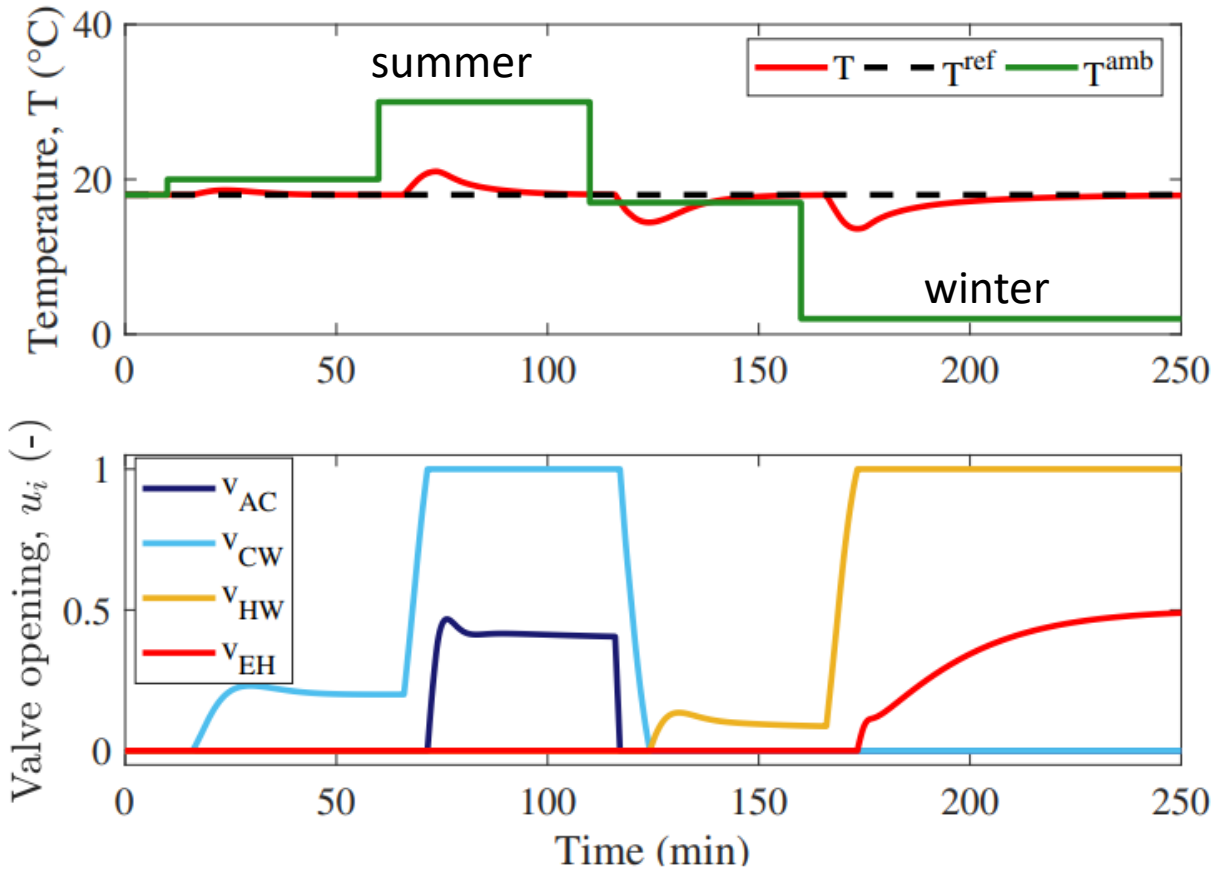
SR-block:



C_{PI} – same controller for all inputs (one integral time)
But get different gains by adjusting slopes α in SR-block

Simulation Split-range control (SRC)





E6. Split parallel control: Separate controllers with different setpoints

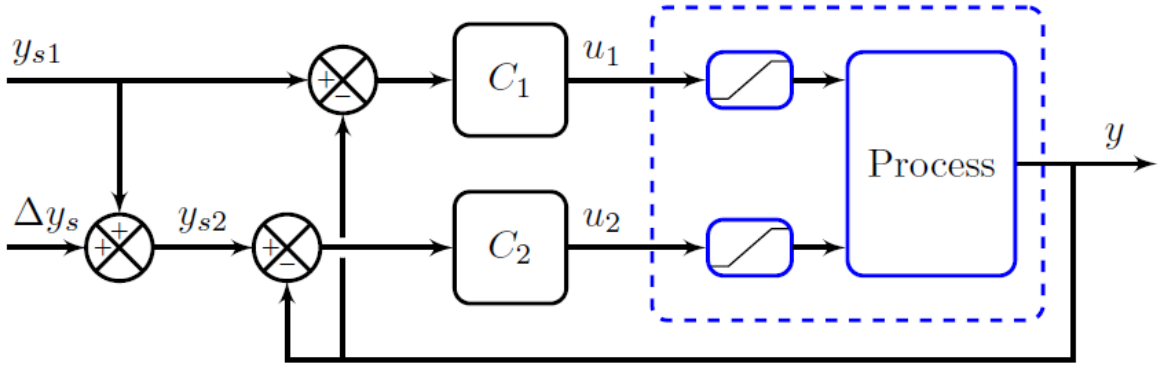


Figure 22: Separate controllers with different setpoints for MV-MV switching.

Comment on the blue blocks: Saturation can occur for any physical input, but they are explicitly shown for cases where the saturation is either the reason for or part of the control logic.

The setpoints $(y_{s1}, y_{s2}, \dots)$ should in the same order as we want to use the MVs. The setpoint differences (e.g., $\Delta y_s = y_{s2} - y_{s1}$ in Fig. 22) should be large enough so that, in spite of disturbances and measurement noise for y, only one controller (and its associated MV) is active at a given time (with the other MVs at their relevant limits).

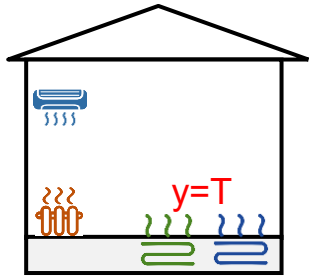
Advantages E6 (compared to split range control, E5):

1. Simple to implement (no logic)
2. Controllers can be tuned independently (different integral times)
3. Switching by feedback: Do not need to know constraint values
   - Big advantage when switching point varies (complex MV-CV switching)

Disadvantages:

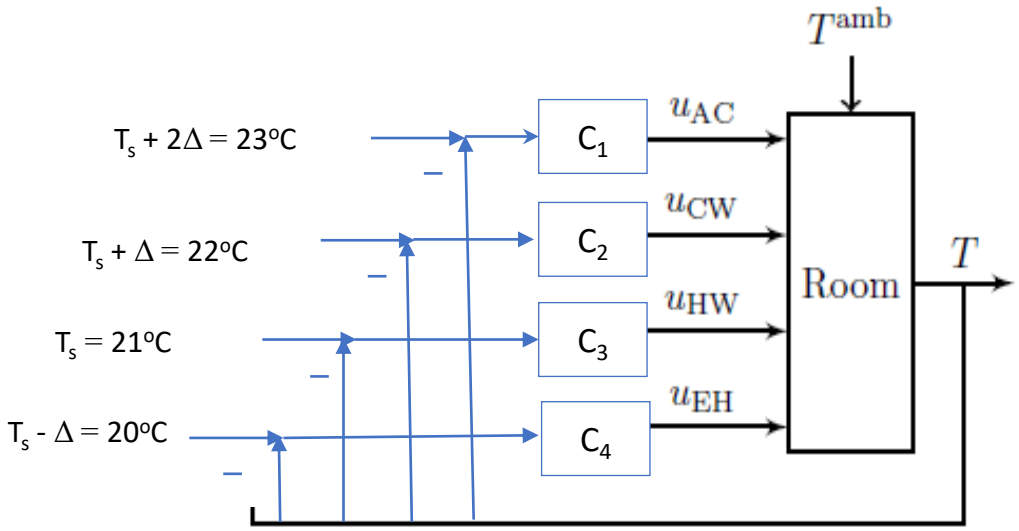
1. Temporary loose control during switching
2. Need reasonably large setpoint separation, so setpoint will vary
   - Can be an advantage (for example, may give energy savings for room heating)
   - Can "fix" with slower outer loop

Example: Room heating with one CV (T) and 4 MVs



MVs (two for summer and two for winter):
1. AC (expensive cooling)
2. CW (cooling water, cheap)
3. HW (hot water, quite cheap)
4. Electric heat, EH (expensive)

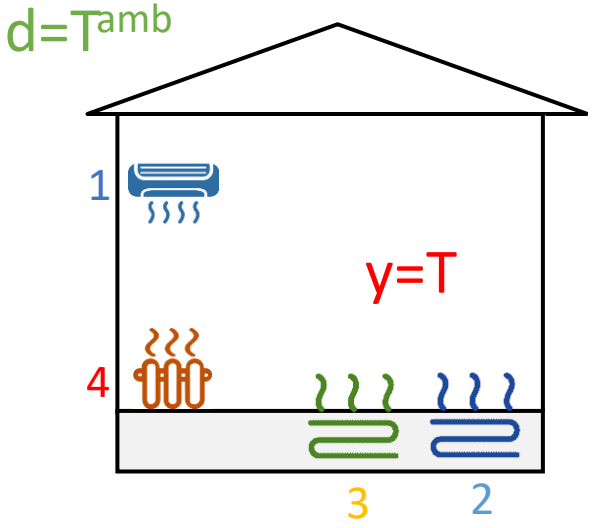
Alt. A2 for MV-MV switching. Multiple controllers with different setpoints

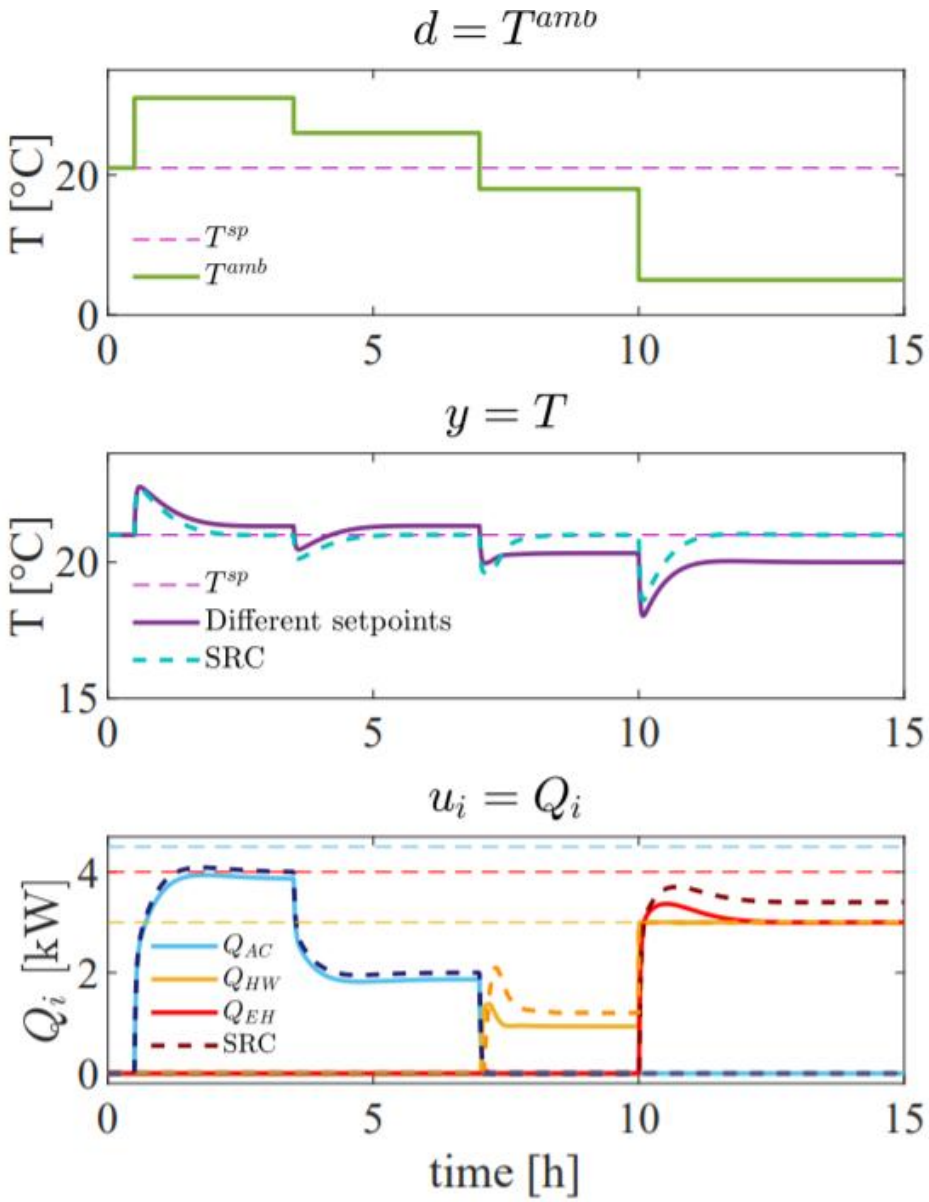


Disadvantage (comfort):
- Different setpoints

Advantage (economics) :
- Different setpoints (energy savings)

E6: MV-MV switching





A Reyes-Lua, S Skogestad. Multiple-Input Single-Output Control for Extending the Steady-State Operating Range - Use of Controllers with Different Setpoints. Processes 7 (12), 941

MV-MV switching, Alt. 2

Fix Split-parallel control: Outer cascade to avoid different setpoints

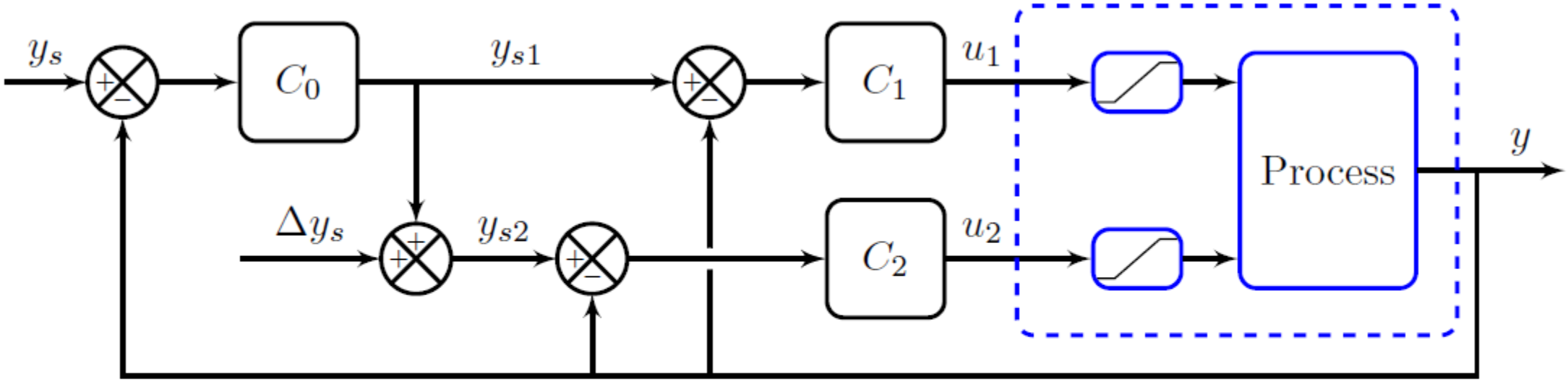


Figure 23: Separate controllers for MV-MV switching with outer resetting of setpoint. This is an extension of the scheme in Figure 22, with a slower outer controller C_0 that resets y_{1s} to keep a fixed setpoint $y = y_s$ at steady state.

E7. Split VPC

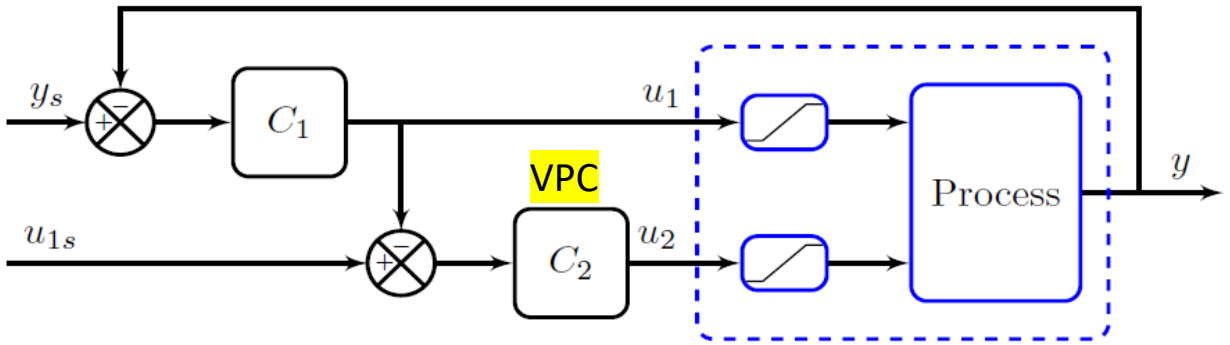


Figure 24: Valve (input) position control for MV-MV switching. A typical example is when u_2 is needed only in fairly rare cases to avoid that u_1 saturates.

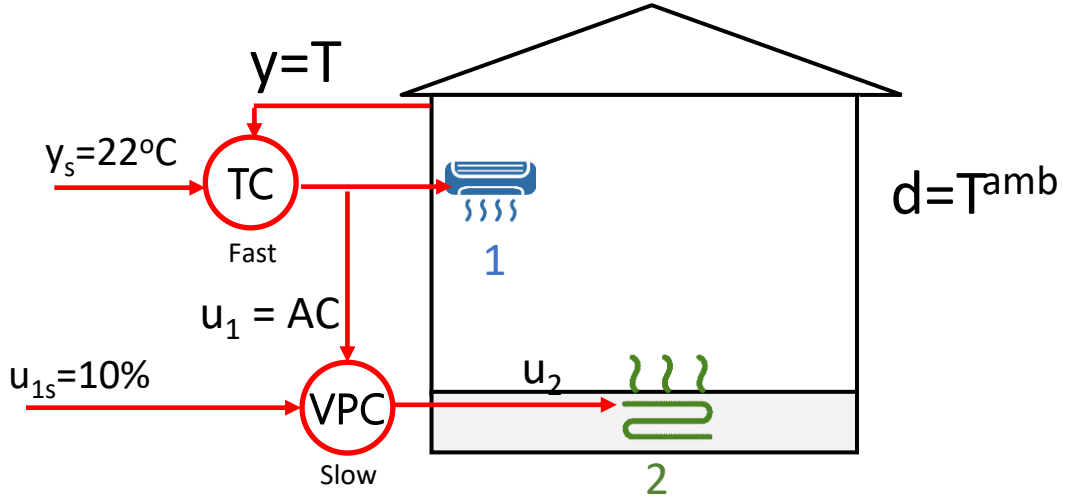
Advantages E7 (for MV-MV switching): Always use u_1 to control y

- For example, u_2 may only allow discrete changes (e.g., u_2=0,1,2,3)
- or dynamics for u_2 may be very slow

Disadvantages E7:

1. We cannot let u_1 become fully saturated because then control of y is lost
   - This means that we cannot use the full range for u_1 (potential economic loss)
2. Related: When u_2 is used, we need to keep using a ‘‘little’’ of u_1.
   - Example temperature control: In summer, use both heating (u1) and cooling (u2) at the same time.

Split VPC for MV-MV switching
Example: Room heating with fast cooling (AC) and slow heating



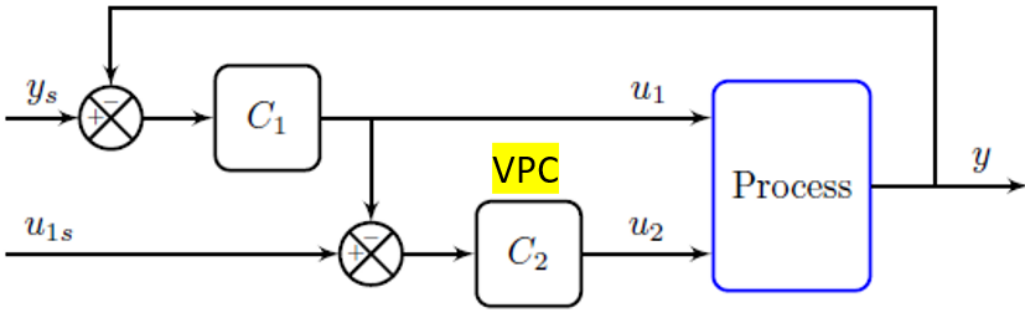
MVs:

1. u_1= AC (cooling with fan, fast)
2. u_2 = HW (hot water in floor, slow)

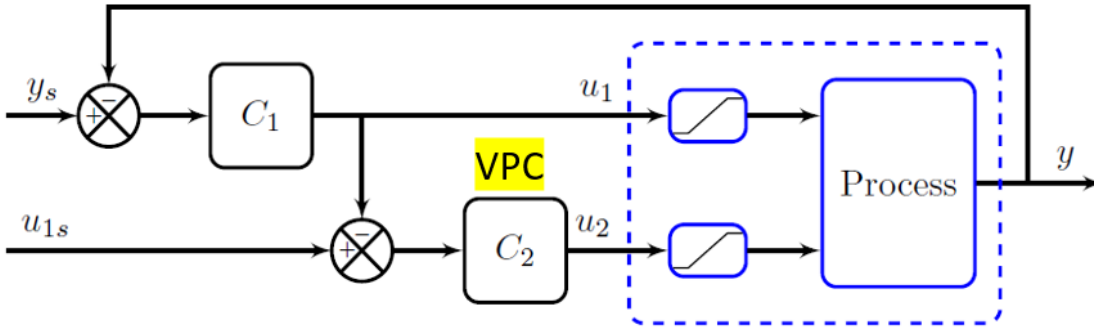
u_2= Hot water (VPC) is only used in winter (in the summer u_2=0%).

Advantage: Temperature is always controlled by fast cooling (u_1=AC)
Economic disadvantage: Cooling u_1 is used also in winter (about 10% load)

Beware: Two different applications of VPC (E3 and E7)



Normal VPC. E3 (Fig. 12)



Split VPC. E7 (Fig. 24)

The VPC schemes in Figure 12 (E3 - VPC on dynamic input) and Figure 24 (E7) seem to be the same

- In fact, they are the same - except for the blue saturation elements - which tells that in Figure 24 (E7) the saturation has to be there for the structure to work as expected

But their behavior is different!

- Normal VPC (E3) both inputs are used all the time
  - u_1 is used to improve the dynamic response
  - u2 is the main steady-state input (and used all the time)
  - u_{1s} is typically 50% (mid-range)
- Split VPC (E7)
  - u_1 is the main input (and used all the time)
  - u_2 is only used when u_1 approaches saturation (for MV-MV switching)
  - The setpoint u_{1s} is typically close to the expected saturation constraint (10% or 90%)

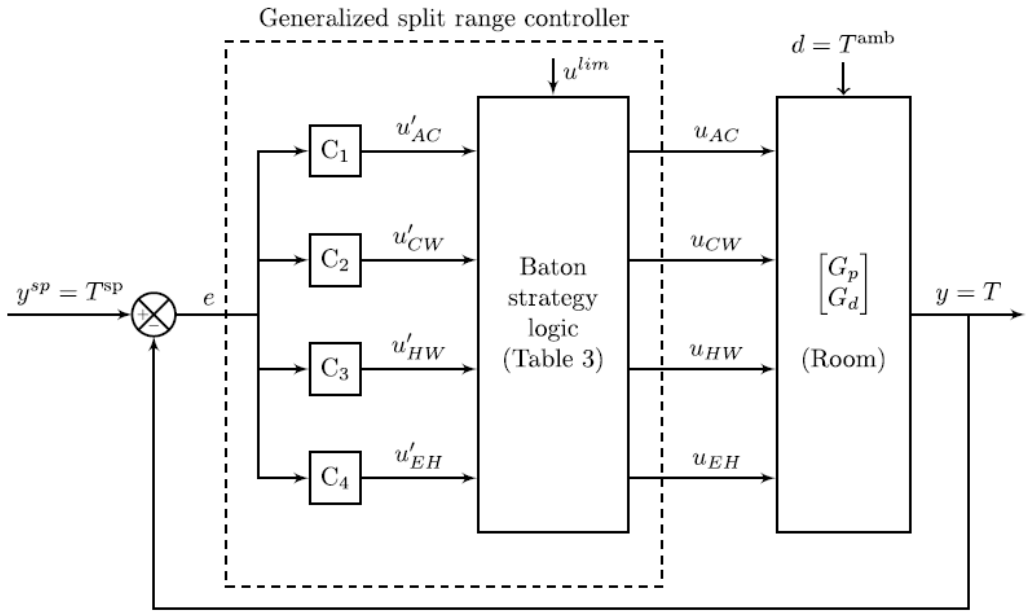
I frequently see people confuse E3 and E7 - which is very understandable!

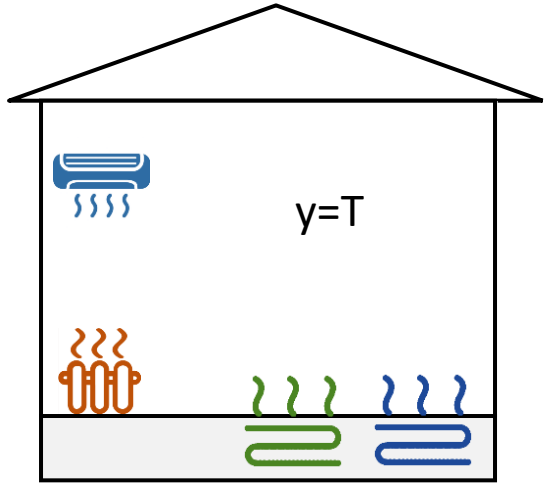
MV-MV switching

Disadvantages Standard Split-range control (SRC):

1. Must use same integral/derivative time for all MVs
2. Does not work well when constraint values change (SR-block problem)

Alternative: Generalized SRC (Baton strategy: multiple independent controllers)





d=T^{amb}

All four controllers need anti-windup

Table 3
Baton strategy logic for case study.

Value of u'_k	Active input (input with *baton*, u_k)			
	$u_1 = u_{AC}$	$u_2 = u_{CW}$	$u_3 = u_{HW}$	$u_4 = u_{EH}$
$u_k^{min} < u'_k < u_k^{max}$	Keep u_1 active $u_1 \leftarrow u'_1$ $u_2 \leftarrow u_2^{max}$ $u_3 \leftarrow u_3^{min}$ $u_4 \leftarrow u_4^{min}$	Keep u_2 active $u_1 \leftarrow u_1^{min}$ $u_2 \leftarrow u'_2$ $u_3 \leftarrow u_3^{min}$ $u_4 \leftarrow u_4^{min}$	Keep u_3 active $u_1 \leftarrow u_1^{min}$ $u_2 \leftarrow u_2^{min}$ $u_3 \leftarrow u'_3$ $u_4 \leftarrow u_4^{min}$	Keep u_4 active $u_1 \leftarrow u_2^{min}$ $u_2 \leftarrow u_1^{min}$ $u_3 \leftarrow u_3^{max}$ $u_4 \leftarrow u'_4$
$u'_k \geq u_k^{max}$	Keep u_1 active (max. cooling)	Baton to u_1 $u_1^0 = u_1^{min}$	Baton to u_4 $u_4^0 = u_4^{min}$	Keep u_4 active (max. heating)
$u'_k \leq u_k^{min}$	Baton to u_2 $u_2^0 = u_2^{max}$	Baton to u_3 $u_3^0 = u_3^{min}$	Baton to u_2 $u_2^0 = u_2^{min}$	Baton to u_3 $u_3^0 = u_3^{max}$

A. Reyes-Lúa and S. Skogestad. "Multi-input single-output control for extending the operating range: Generalized split range control using the baton strategy". Journal of Process Control 91 (2020)

MV-MV switching

Alternative: Generalized SRC (Baton strategy: multiple independent controllers)

Table 3
Baton strategy logic for case study.

Value of u'_k	Active input (input with *baton*, u_k)			
	$u_1 = u_{AC}$	$u_2 = u_{CW}$	$u_3 = u_{HW}$	$u_4 = u_{EH}$
$u_k^{min} < u'_k < u_k^{max}$	Keep u_1 active $u_1 \leftarrow u'_1$ $u_2 \leftarrow u_2^{max}$ $u_3 \leftarrow u_3^{min}$ $u_4 \leftarrow u_4^{min}$	Keep u_2 active $u_1 \leftarrow u_1^{min}$ $u_2 \leftarrow u'_2$ $u_3 \leftarrow u_3^{min}$ $u_4 \leftarrow u_4^{min}$	Keep u_3 active $u_1 \leftarrow u_1^{min}$ $u_2 \leftarrow u_2^{min}$ $u_3 \leftarrow u'_3$ $u_4 \leftarrow u_4^{min}$	Keep u_4 active $u_1 \leftarrow u_2^{min}$ $u_2 \leftarrow u_1^{min}$ $u_3 \leftarrow u_3^{max}$ $u_4 \leftarrow u'_4$
$u'_k \geq u_k^{max}$	Keep u_1 active (max. cooling)	Baton to u_1 $u_1^0 = u_1^{min}$	Baton to u_4 $u_4^0 = u_4^{min}$	Keep u_4 active (max. heating)
$u'_k \leq u_k^{min}$	Baton to u_2 $u_2^0 = u_2^{max}$	Baton to u_3 $u_3^0 = u_3^{min}$	Baton to u_2 $u_2^0 = u_2^{min}$	Baton to u_3 $u_3^0 = u_3^{max}$

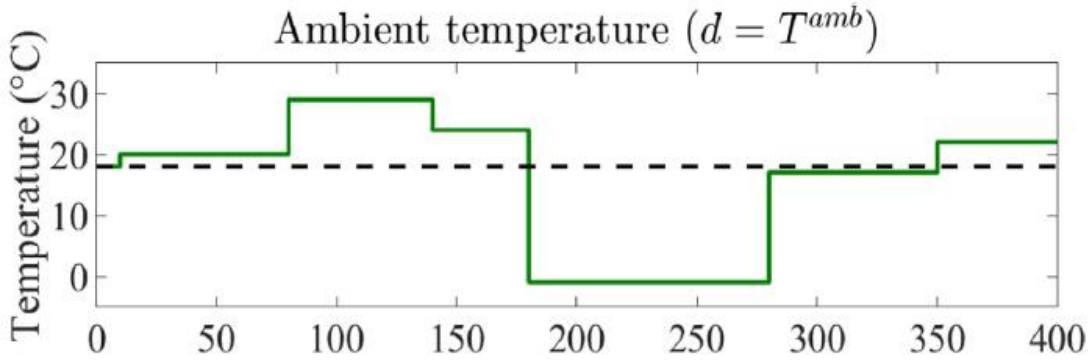
All four controllers need anti-windup

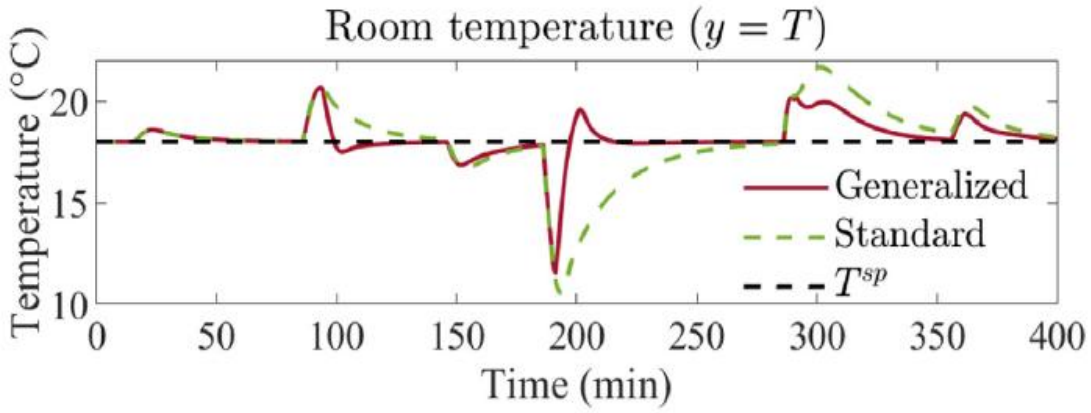
A. Reyes-Lúa and S. Skogestad. "Multi-input single-output control for extending the operating range: Generalized split range control using the baton strategy". Journal of Process Control 91 (2020)

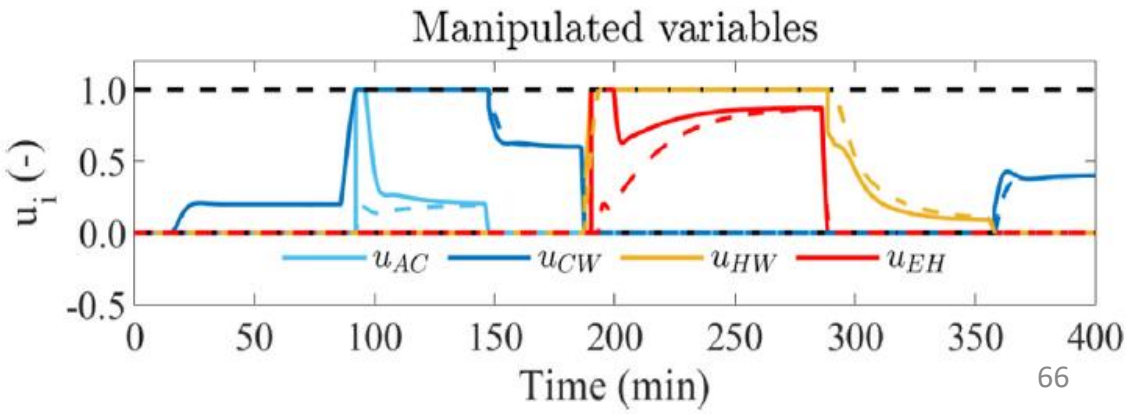
Comparison of standard and generalized SRC

Generalized split range control:

- Different (smaller) integral times for each input
- Gives faster settling for most inputs







A. Reyes-Lúa and S. Skogestad. "Multi-input single-output control for extending the operating range: Generalized split range control using the baton strategy". Journal of Process Control 91 (2020)

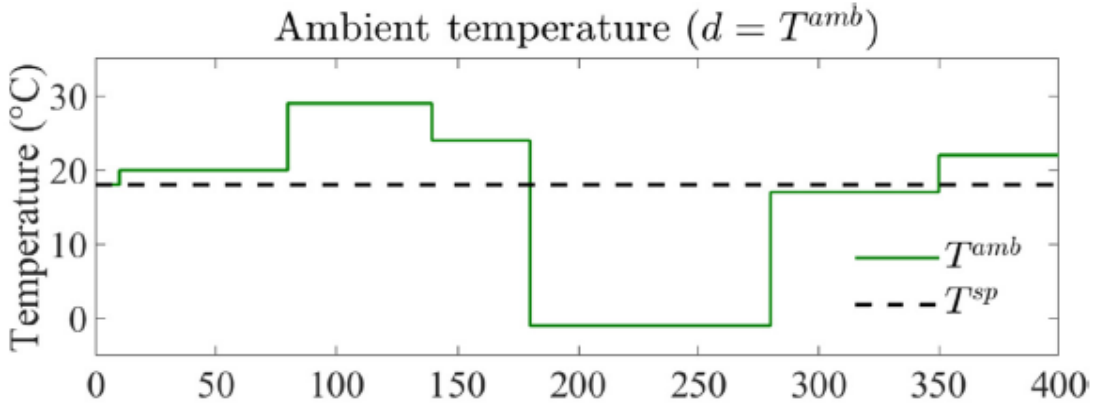
What about Model Predictive control (MPC)?

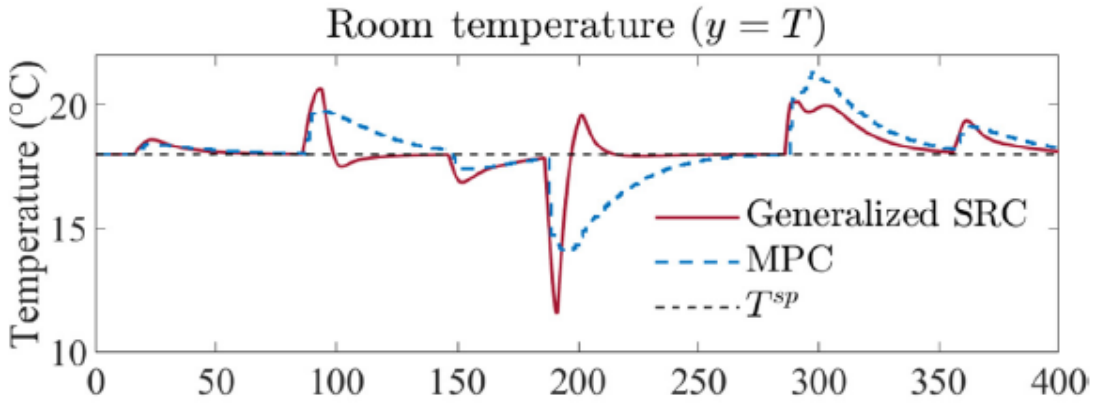
Comparison of Generalized SRC and MPC

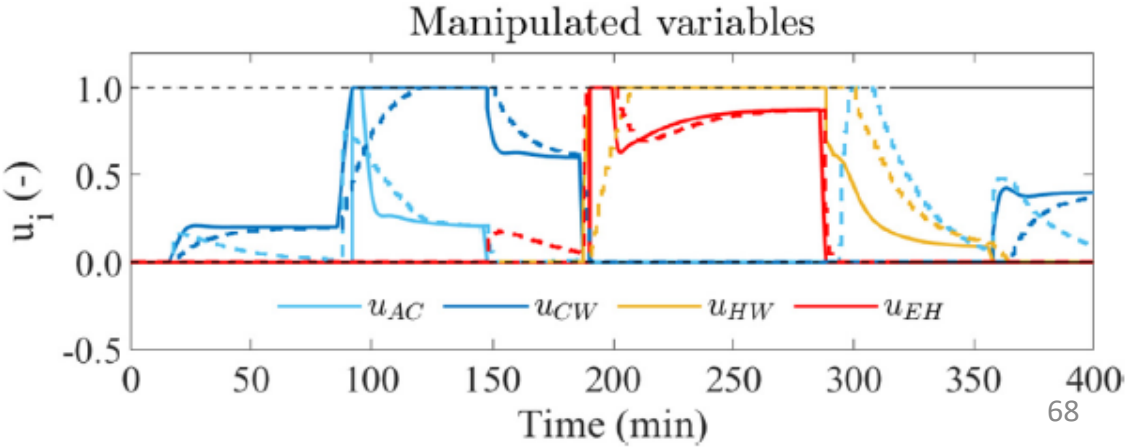
Responses
MPC: Similar response to standard SRC
MPC: Faster initially, uses several input simultanously
MPC: Slower settling

Disadvantage MPC:

- Complex: Requires full dynamic model
- Does not use on input at a time

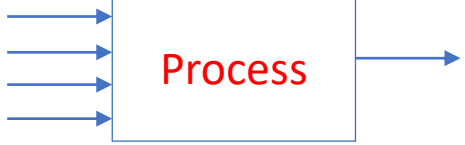






A. Reyes-Lúa and S. Skogestad. "Multi-input single-output control for extending the operating range: Generalized split range control using the baton strategy". Journal of Process Control 91 (2020)

Summary MV-MV switching



Split-range control (E5)

Advantage: SRC is easy to understand and implement!

Disadvantages:

1. Only one controller C ⇒ Same integral time for all inputs u_i (MVs)
   - Controller gains can be adjusted with slopes in SR-block!
2. Does not work well for cases where constraint values for u_i change

Split parallel control (with different setpoints) (E6)

Advantages several controllerd (compared to split range control, E5):

1. Simple to implement (no logic)
2. Controllers can be tuned independently (different integral times)
3. Switching by feedback: Do not need to know constraint values
   - Big advantage when switching point varies (complex MV-CV switching)

Disadvantages:

1. Temporary loose control during switching
2. Setpoint separation (setpoint not constant)
   - Can be an advantage (for example, may give energy savings for room heating)

Split VPC (E7)

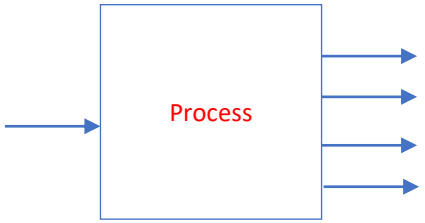
Advantage : Always use u_1 to control y

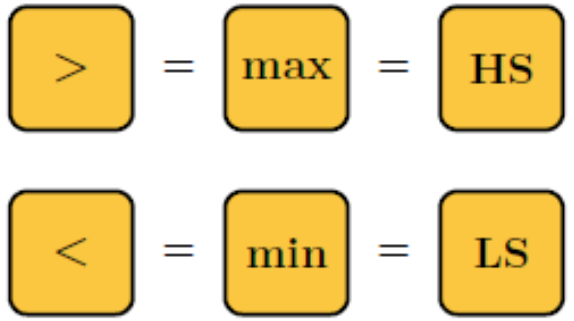
Disadvantages:

1. We cannot let u_1 become fully saturated because then control of y is lost
   - potential economic loss
2. Related: When use u_2, need keep using a ‘‘little’’ of u_1.
   - Example: May use both heating and cooling at the same time

B. CV-CV switching*

- One MV
- Many CVs, but control only one at a time
- Always use Selector**
  - + One controller for each CV
- Selector is generally on MV (compare output from many controllers):





=
max
=
HS
<
=
min
=
LS

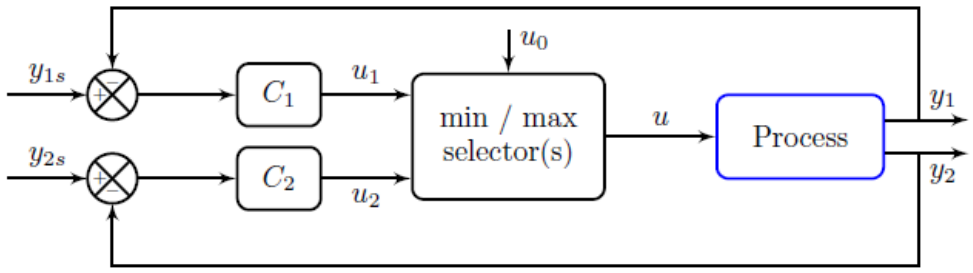


Figure 17: CV-CV switching with selector on MV (input u).

* It is sometimes called **«override».** But I prefer to use «override» for undesirable temporary (dynamic) switches, for example, to avoid overflowing a tank dynamically. Otherwise, it's (desired) CV-CV switching

**Well… A Selector is a logic element and may be implemented using "if-then-else" logic. Also, in some cases it may implemented using a saturation element.

Implementation selector

Alt. I (General). Several controllers (different CVs)

- Selector on MVs*
  - Must have anti windup for C_1 and C_2 !

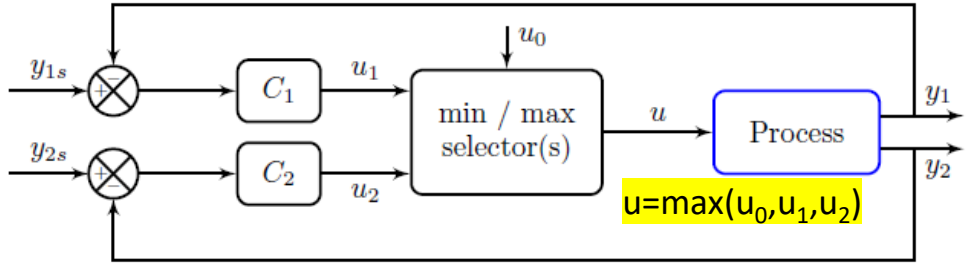
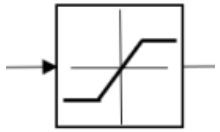


Figure 17: CV-CV switching with selector on MV (input u).

Alt. II (Less general) Controllers in cascade

- Selector on CV setpoint
- Good alternative if CVs (y_1 and y_2) are related so that cascade is good
- In this case: Selector may be replaced by saturation element
  (with y_{2s} as the max or min)



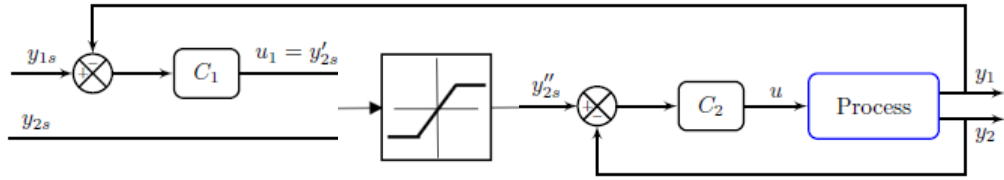
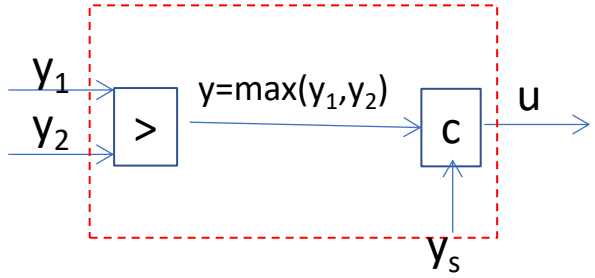


Figure 19: Alternative cascade CV-CV switching implementation with selector on the setpoint. In many cases, y_{1s} and y_{2s} are constraint limits.

Alt. III (For special case where all CVs have same bound). One controller

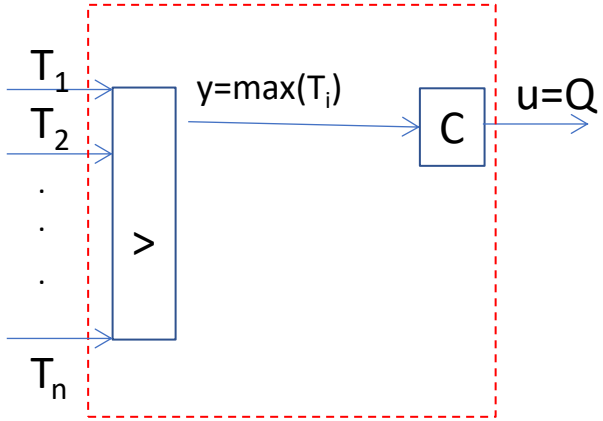
- Selector is on CVs (Auctioneering)
- Also assumes that dynamics from u to y_1 and y_2 are similar; otherwise use Alt.I
- Example: Control hot-spot in reactor or furnace.



*It may seem surprising that the selection is on the MV for a CV-CV switch, but this turns out to be the most general and most effecftive.

Example Alt. III

- Hot-spot control in reactor or furnace



- Comment: Could use General Alternative I (many controllers) for hot-spot control, with each temperature controller (c_1, c_2,…) computing the heat input ($u_1=Q_1$, $u_2=Q_2$, ….) and then select u = **min**(u_1, u_2, …), but it is more complicated.

Furnace control with safety constraint (Alt. I)

Input (MV)

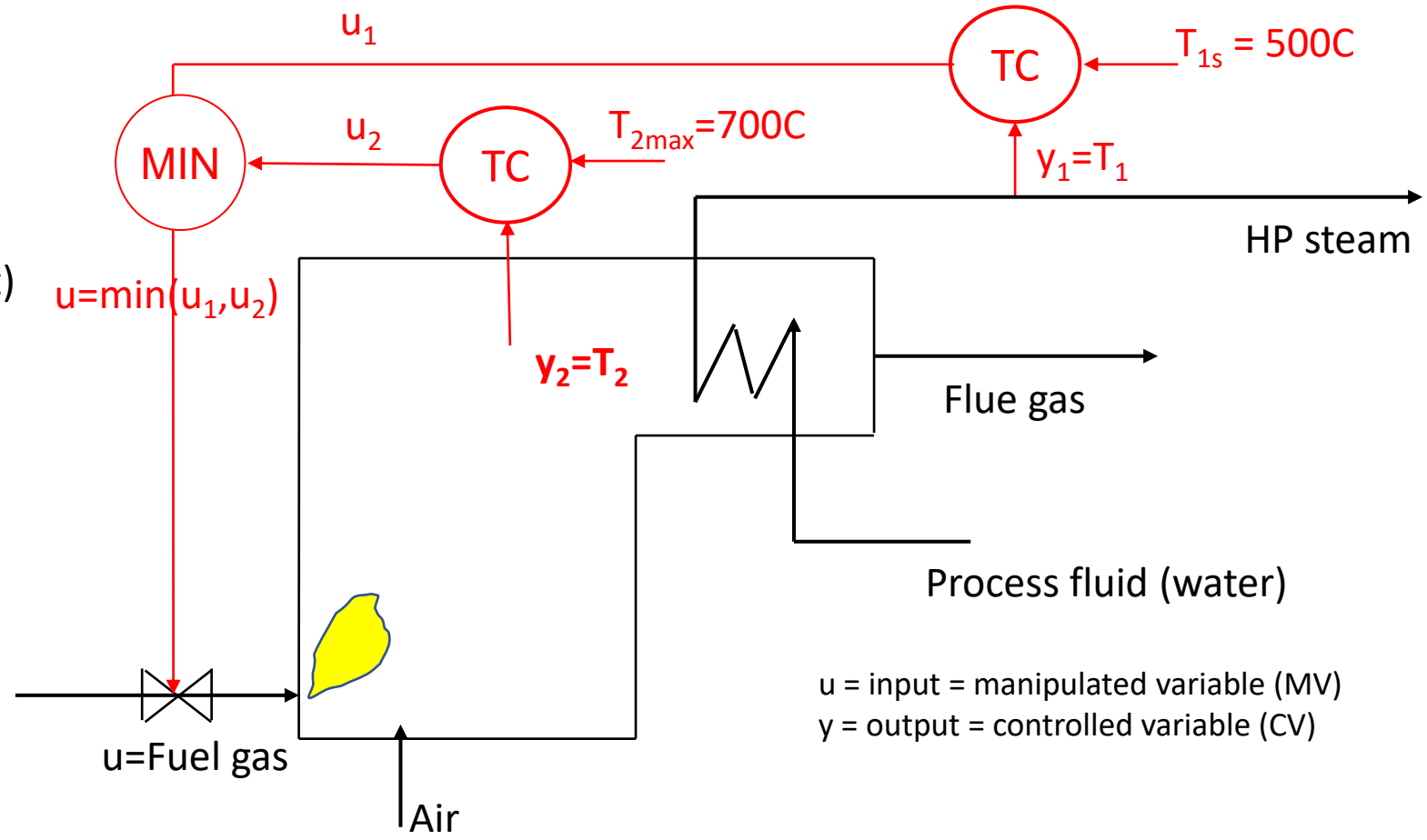
u = Fuel gas flowrate

Output (CV)

y_1 = process temperature T_1
(desired setpoint or max constraint)

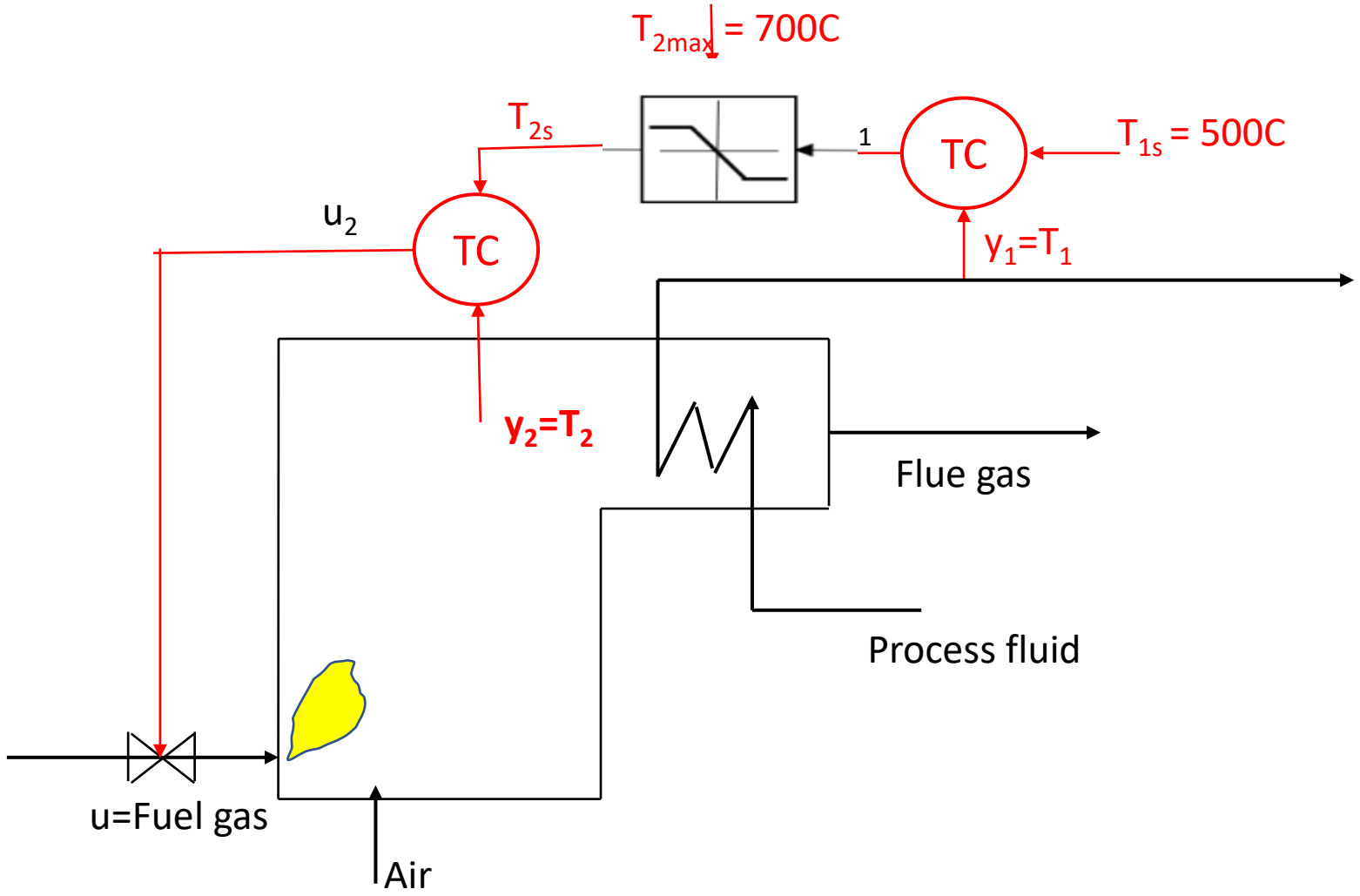
y_2 = furnace temperature T_2 (max constraint)

Rule: Use min-selector for constraints that are satisfied with a small input



Furnace control with cascade (Alt. II, selector on CV-sp)

Comparison
The cascade solution is less general but it may be better in this case.
Why better? Inner T2-loop is fast and always active and may improve control of T1.

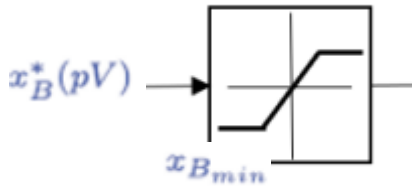


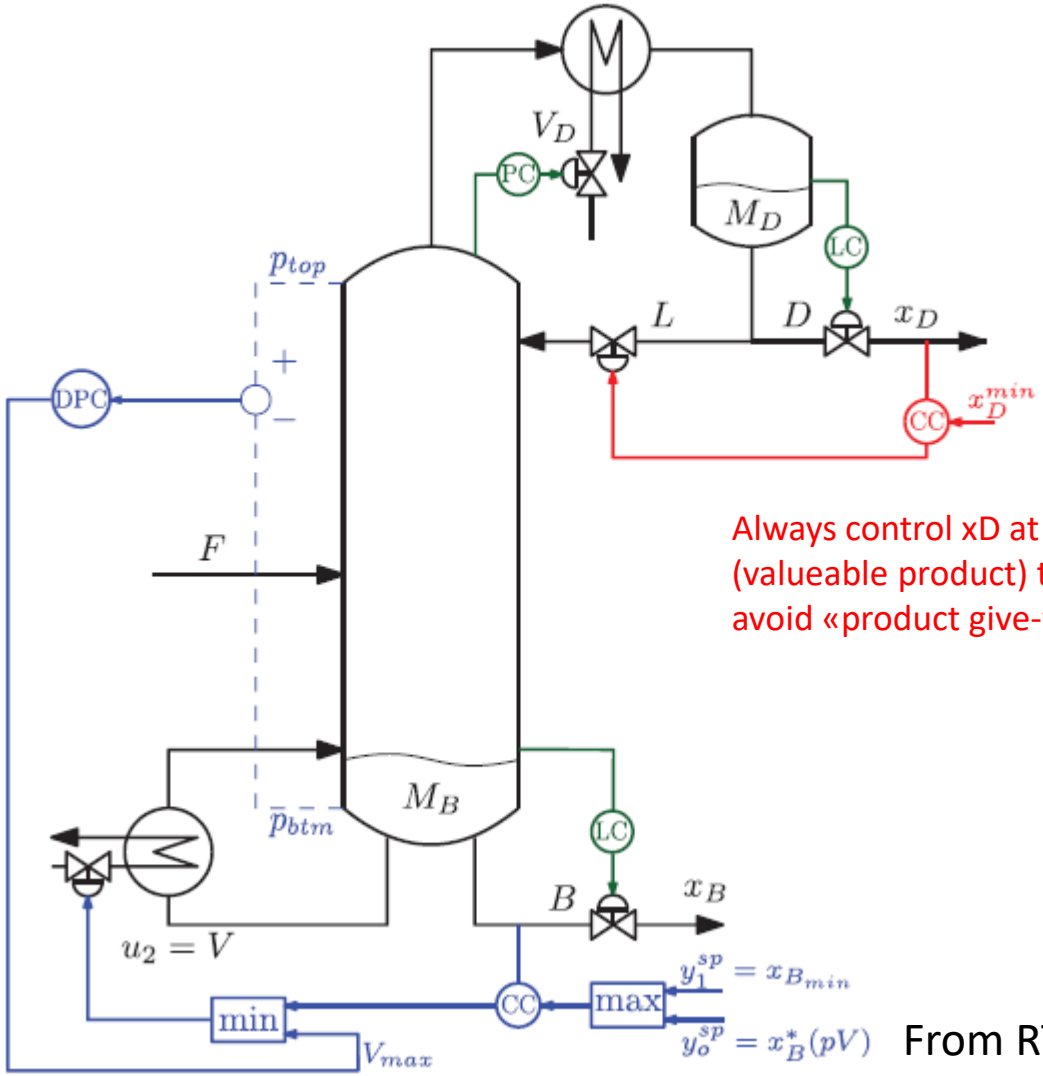
Comment: For both Alt. I and II, we loose control of T1 (it drops below T1s=500C) when T2=T2max. If this is not acceptable then we need to something- More on this later!

CV-CV switching

Distillation example

Note: A selector where one input is a constant (like the max-block for xB) may be replaced by a saturation element





Always control xD at constraint (valueable product) to avoid «product give-way»

From RTO

Avoid flooding using constraint on DP (Alt. I: selector in input)

May overpurity bottom to get more of the valuable product (Alt. II: Selector on setpoint)

Design of selector structure

Rule 1 (max or min selector)

- Use max-selector for constraints that are satisfied with a large input
- Use min-selector for constraints that are satisfied with a small input

Rule 2 (order of max and min selectors):

- If need both max and min selector: Potential infeasibility (conflict)
- Order does not matter if problem is feasible
- If infeasible: Put highest priority constraint at the end

"Systematic design of active constraint switching using selectors." Dinesh Krishnamoorthy , Sigurd Skogestad. Computers & Chemical Engineering, Volume 143, (2020)
"Advanced control using decomposition and simple elements". Sigurd Skogestad. Annual Reviews in Control, Volume 56, 100903 (2023)

Rule 2 (order of selectors)

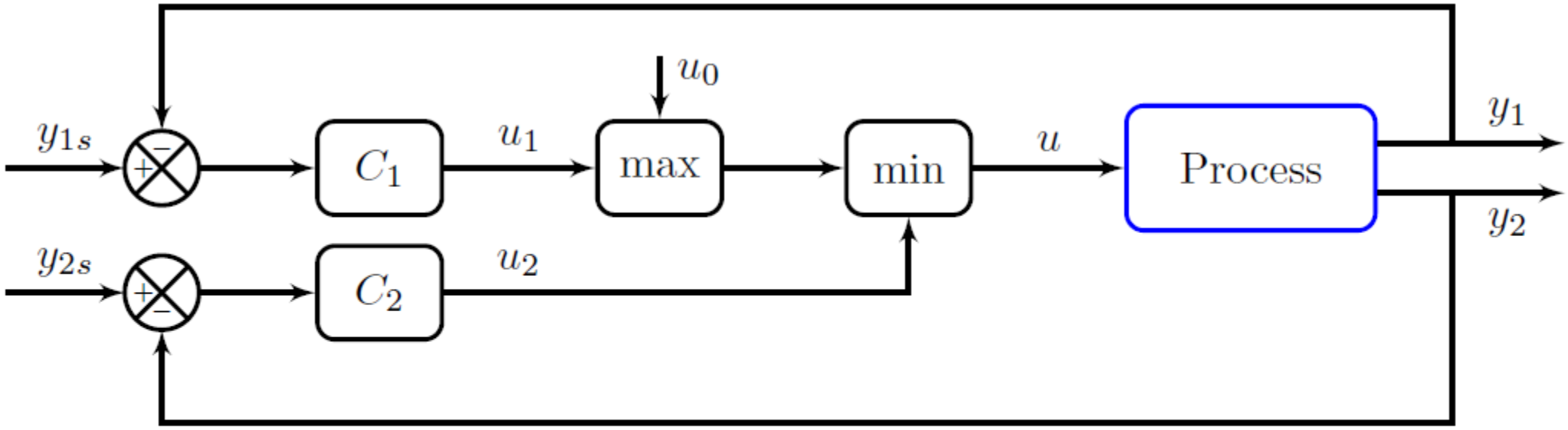


Figure 18: CV-CV switching for case with possibly conflicting constraints. In this case, constraint y_{1s} requires a max-selector and y_{2s}) requires a min-selector. The selector block corresponding to the most important constraint (here y_{2s}) should be at the end (Rule 2).

To understand the logic with selectors in series, start reading from the first selector. In this case, this is the max-selector: The constraint on y_1 is satisfied by a large value for u which requires a max-selector (Rule 1). u_0 is the desired input for cases when no constraints are encountered, but if y_1 reaches its constraint y_{1s}, then one gives up u_0. Next comes the min-selector: The constraint on y_2 is satisfied by a small value for u which requires a min-selector (Rule 1). If y_2 reaches its constraint y_{2s}, then one gives up controlling all previous variables (u_0 and y_1) since this selector is at the end (Rule 2). However, note that there is also a "hidden" max- and min- selector (Rule 3) at the end because of the possible saturation of u, so if the MV (input) saturates, then all variables (u_0, y_1, y_2) will be given up.

CV-CV switching

Example. Maximize flow with pressure constraints

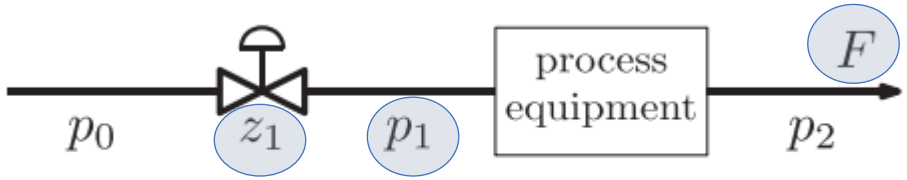


Fig. 6. Example 2: Flow through a pipe with one MV ($u = z_1$).

Input u = z_1
Want to maximize flow, J=-F:

Optimization problem is:

$$\max_{z_1} F$$

s.t.

$$\begin{aligned} F &\leqslant F_{max} \\ p_1 &\leqslant p_{1,max} \\ p_1 &\geqslant p_{1,min} \\ z_1 &\leqslant z_{1,max} \end{aligned} \qquad (15)$$

Satisfied by

Small u

Small u

Large u

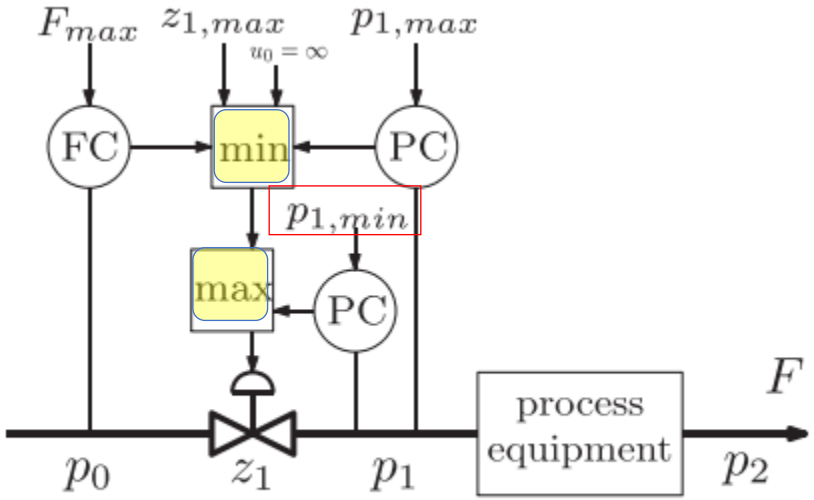
Possible conflict

-

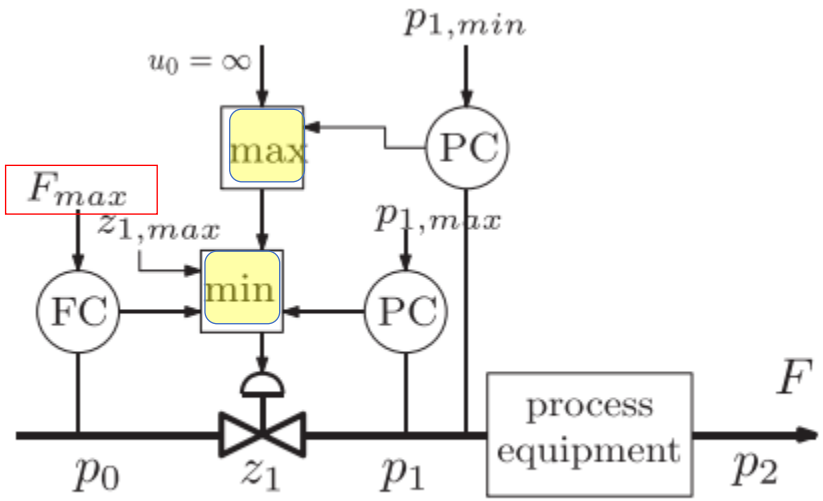
where F_{max} =10 kg/s, $z_{1,max} = 1$, $p_{1,max} = 2.5$ bar, and $p_{1,min} = 1.5$ bar. Note that there are both max and min- constraints on p_1. De-

The two p1-constraints are not conflicting, because they are on the same variable. However the Fmax-constraint and p1min-constraint may be conflicting: Must choose which is most important.

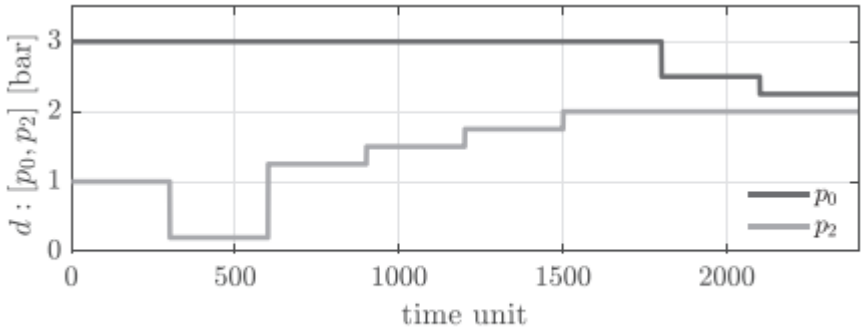
CV-CV switching



(a)

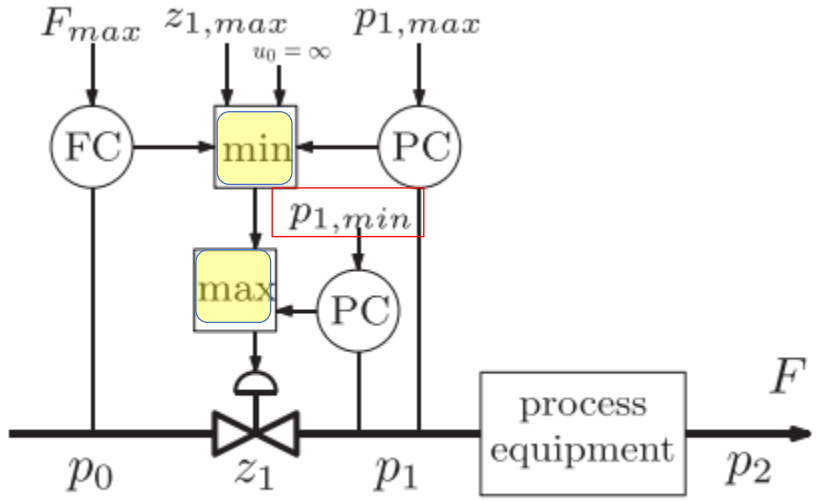


(b)

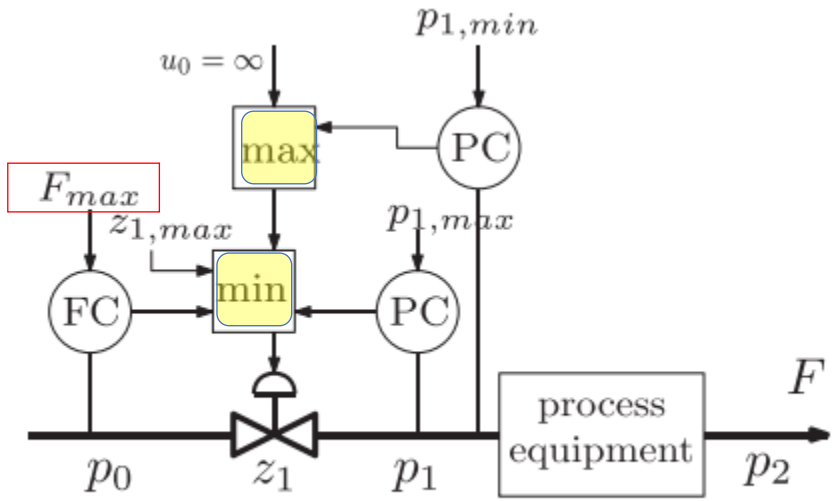


Disturbances in p_0 and p_2 (unmeasured)

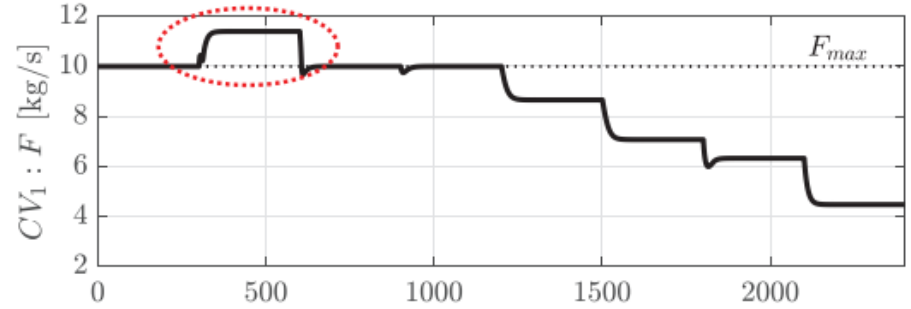
CV-CV switching

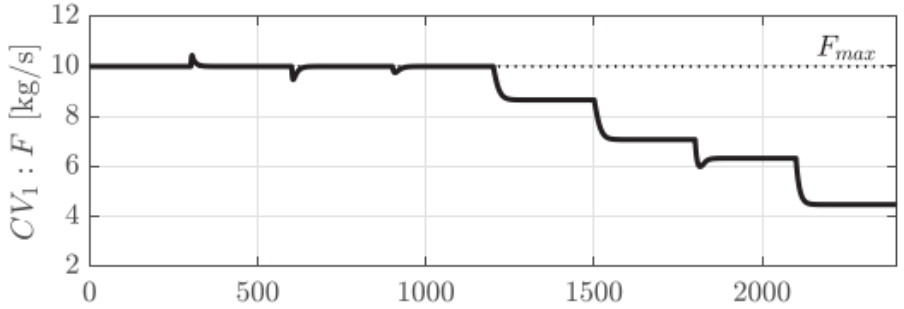


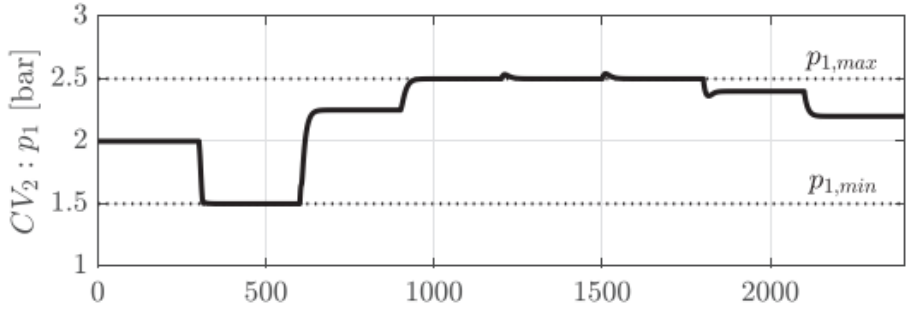
(a)

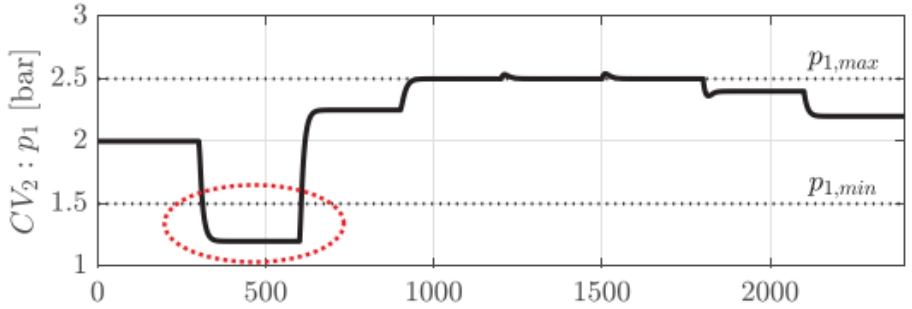


(b)







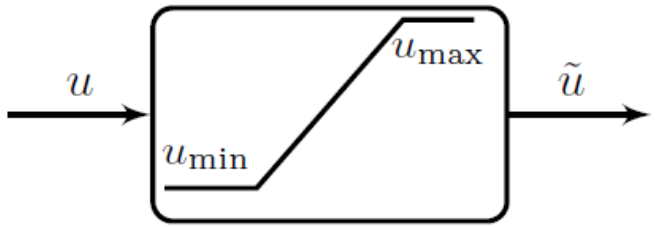


t>1800: u=zmax=1

Valves have “built-in” selectors

Rule 3 (maybe a bit opposite of what you may guess)

- A closed valve (u_{min}=0) gives a “built-in” max-selector (to avoid negative flow)
- An open valve (u_{max}=1) gives a “built-in” min-selector
  - So: Not necessary to add these as selector blocks (but it will not be wrong).
  - The “built-in” selectors are never conflicting because cannot have closed and open at the same time
  - Another way to see this is to note that a valve works as a saturation element



Saturation element may be implemented in three ways (equivalent because never conflict)

1. Min-selector followed by max-selector
2. Max-selector followed by min-selector
3. Mid-selector

$$\tilde{u} = \max(u_{min}, \min(u_{max}, u)) = \min(u_{max}, \max(u_{min}, u)) = \mathrm{mid}(u_{min}, u, u_{max})$$

Quiz. Is this OK?

Cristina: I am looking at a control solution using selectors for keeping the pressure within constraints, while maximizing the valve opening. See figure This is for the valve before a steam turbine. This should work OK, right? Of course, if pmax is reached while the valve is fully open, the turbine bypass will have to open.

Answer:
Rule 1. Yes, the rule is to use a max-selector for a constraint which is satisfied with a large input. And since the pressure is measured upstream, the pressure will get lower if we increase the valve opening, making it easier to satisfy the pmax-constraint. So yes, this is OK.

Rule 1. Similar for the min-block with pmin.

Rule 2. Since you have two constraints on the same variable, you cannot have infeasibility so the order of the min. and max-blocks doesn't matter for pmin and pmax.

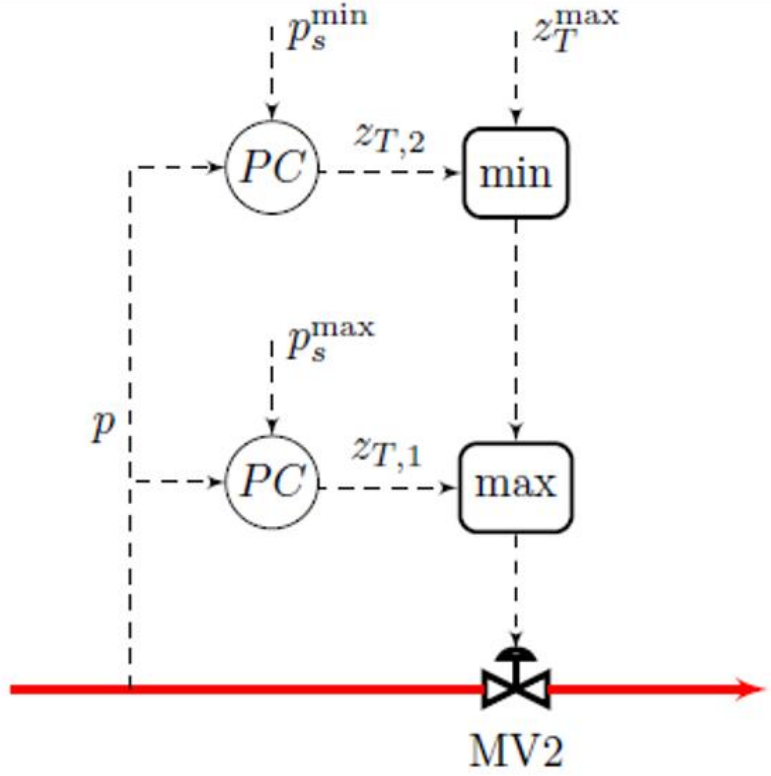
Rule 2. Yes, the desired value uo=zmax should always enter the first block.

Conclusion: yes, it works.

BUT….
Comment 1: But note that there is also a "hidden" min-selector just before the valve because of the valve which has zmax. And also a "hidden" max-selector because of zmin (a fully closed valve). These constraints may be inconsistent with the pressure constraints.

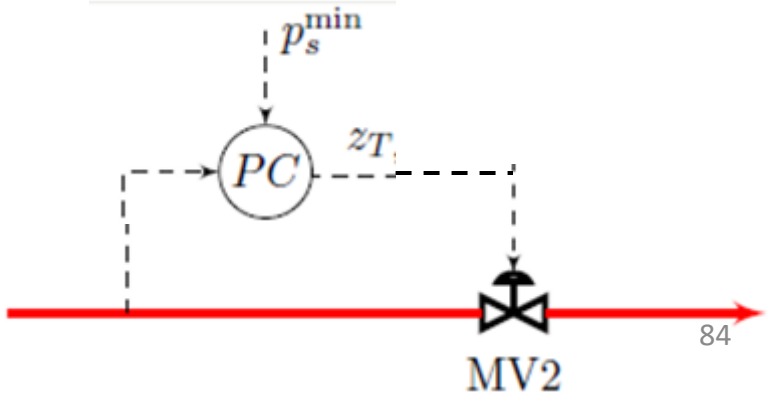
Comment 2: Since the order of the two selectors does not matter in this case, one may instead use the "equivalent" alternative with the max-block first. But we then see clearly that *the constraint on pmax will never be activated*, because ztmax is large. I guess this makes sense since you want to have the valve as open as possible, so then the you will always be at the pmin-constraint or have a fully open valve. *So you can cut the pmax-constraint (and thus the max-selector)* as you anyway want to open the valve as much as possible.

In addition, you can also cut the min-selector because there is already a "hidden" min-selector with zmax. (On the other hand, it will not be wrong to keep them.)



Final conclusion: Yes, it works, but it's much too complicated.

- **All what is shown can be replaced by a pressure controller (PC) with setpoint p^{min}.**



Challenges selectors

- Standard approach requires pairing of each active constraint with a single input
  - May not be possible in complex cases
  - See RTO/feedback-based RTO
- Stability analysis of switched systems is still an open problem
  - Undesired switching may be avoided in many ways:
    - Filtering of measurement
    - Tuning of anti-windup scheme
    - Minimum time between switching
    - Minimum input change