Putting optimization into the control layer using the magic of feedback control

Sigurd Skogestad

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

ESCAPE'32. Toulouse, 13 June 2022

Thanks to: Adriana Reyes-Lúa, Cristina Zotică, Dinesh Krishnamoorthy

“The goal of my research is to develop simple yet rigorous methods to solve problems of engineering significance”

http://folk.ntnu.no/skoge/

Startside - innsida.ntnu.no | Sigurd Skogestad

File Edit View Favorites Tools Help

Chemical Engineering

Sigurd Skogestad Professor

Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), N7491 Trondheim, Norway

Start here...

- About me - CV - Lectures - My family - How to reach me - *Email: skoge@chemeng.ntnu.no*
- Teaching: Courses - Master students - Project students
- Research: Process Control Group - Research - Ph.D. students

"We want to find a self-optimizing control structure where acceptable operation under all conditions is achieved with constant setpoints for the controlled variables. More generally, the idea is to use the model off-line to find properties of the optimal solution suited for (simple, model-free) on-line implementation"

"News"...

- PhD position on "Production Optimization" (Deadline: 17 June 2019)
- Two PhD positions on "Process optimization using machine learning" (Deadline: 10 June 2019)
- Special issue of Processes on "Real-time optimization of processes using simple control structures, economic MPC or machine learning." (Deadline: 15 Nov.2019)
- July 2018: PID-paper in JPC that verifies SIMC PI-rules and gives "Improved" SIMC PID-rules for processes with time delay (taud=theta/3)
- June 2018: Video of Sigurd giving lecture at ESCAPE-2018 in Graz on how to use classical advanced control for switching between active constraints
- May 2017: Presentation (slides) on economic plantwide control from AdCONIP conference in Taiwan
- Feb. 2017: Youtube videos of Sigurd giving lectures on PID control and Plantwide control (at University of Salamanca, Spain)
- 06-08 June 2016: IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems (DYCOPS-2016), Trondheim, Norway.
  - Videos and proceedings from DYCOPS-2016
- Aug 2014: Sigurd recieves IFAC Fellow Award in Cape Town
- 2014: Overview papers on "control structure design and "economic plantwide control"
- OLD NEWS

Books...

- Book: S. Skogestad and I. Postlethwaite: MULTIVARIABLE FEEDBACK CONTROL-Analysis and design. Wiley (1996; 2005)
- Book: S. Skogestad: CHEMICAL AND ENERGY PROCESS ENGINEERING CRC Press (Taylor&Francis Group) (Aug. 2008)
- Bok: S. Skogestad: PROSESSTEKNIKK- Masse- og energibalanser Tapir (2000; 2003; 2009).

More information ...

- **Publications** from my Google scholar site
- **Download publications** from my official publication list .... or look HERE if you want to download our most recent and upublished work
- Proceedings from conferences - some of these may be difficult to obtain elsewhere
- PROST - Our activity is part of PROST - Center for Process Systems Engineering at NTNU and SINTEF
- Process control library - We have an extensive library for which Ivar has made a nice on-line search
- Photographs that I have collected from various events (maybe you are included...)
- International conferences - updated with irregular intervals
- SUBPRO (NTNU center on subsea production and processing) [Documents]

Outline

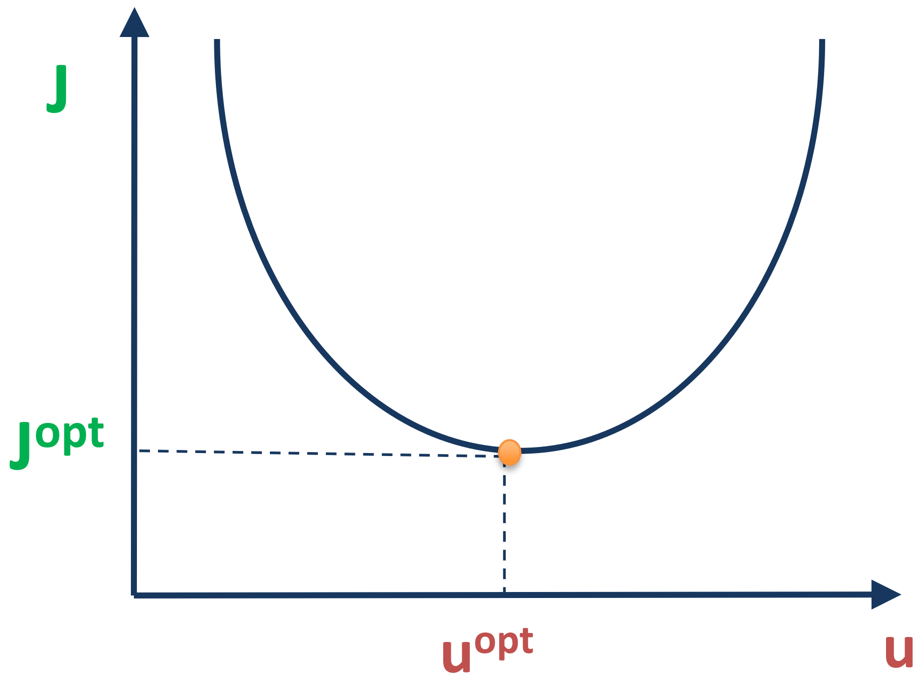
1. Introduction: Optimal economic operation of process plants
2. Control: Implementing optimal operation in practice
    - Model predictive control (MPC)
    - Conventional advanced Process control (APC)
3. Unconstrained Optimization: Self-optimizing variables
4. Constrained Optimization: Using conventional APC to handle changing active constraints
    - Many examples
    - PID-control, Selectors, Split range control
5. Systematic procedure for designing APC
    - More Examples
6. Conclusion

Optimal economic operation

Minimize cost J = J(**u**,**x**,**d**)

Or: Maximize profit P=-J

- **u** = degrees of freedom
- **x** = states (internal variables)
- **d** = disturbances



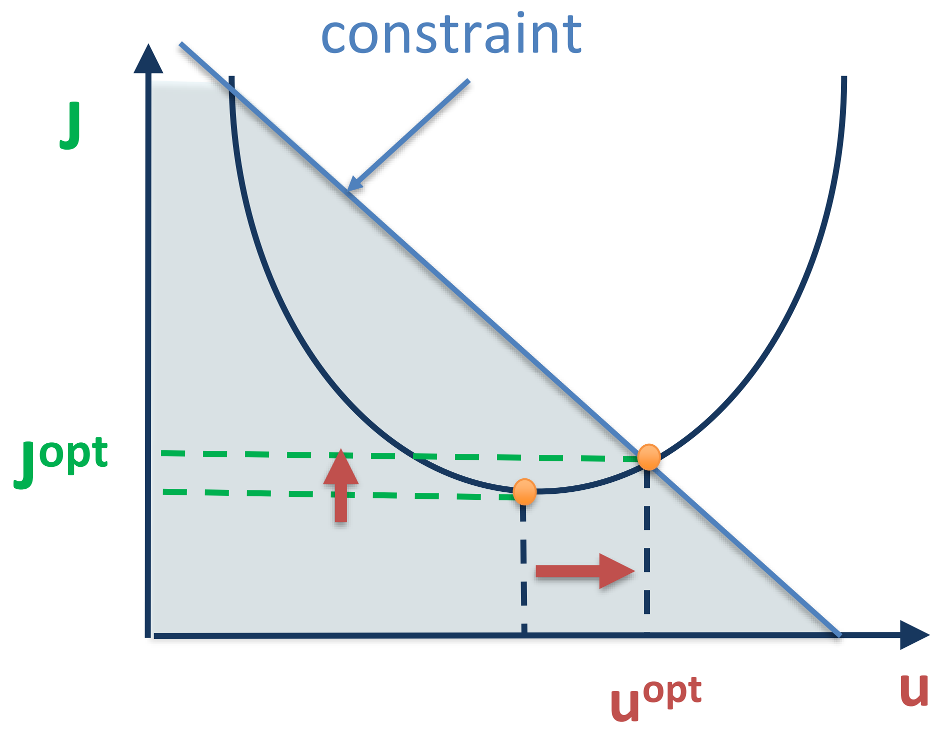
J = cost feed + cost energy – value of products

Optimal economic operation

Minimize cost J = J(**u**,**x**,**d**)

Subject to satisfying constraints

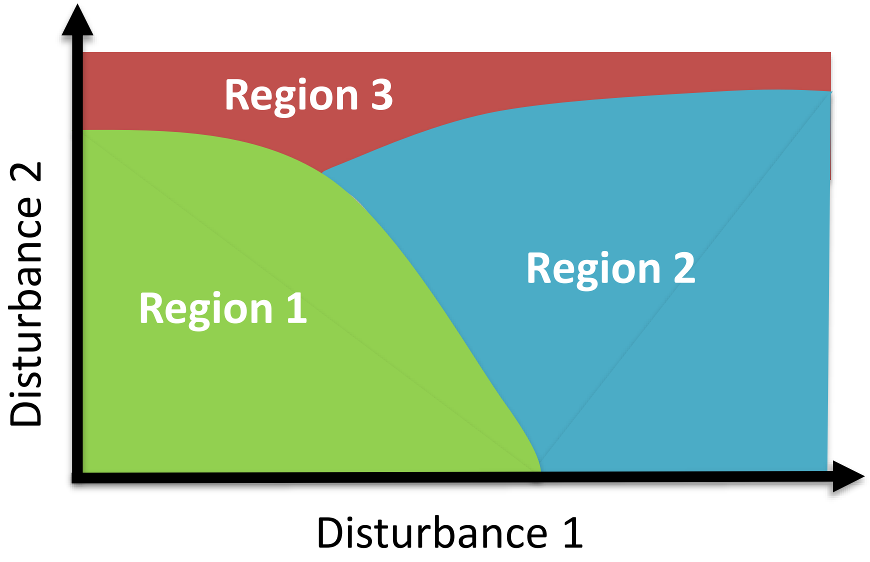
- **u** = degrees of freedom
- **x** = states (internal variables)
- **d** = disturbances



J = cost feed + cost energy – value of products

Active constraints

- **Active constraints:**
  - variables that should optimally be kept at their limiting value.
- **Active constraint region:**
  - region in the disturbance space with fixed active constraints



Optimal operation:
How switch between regions?

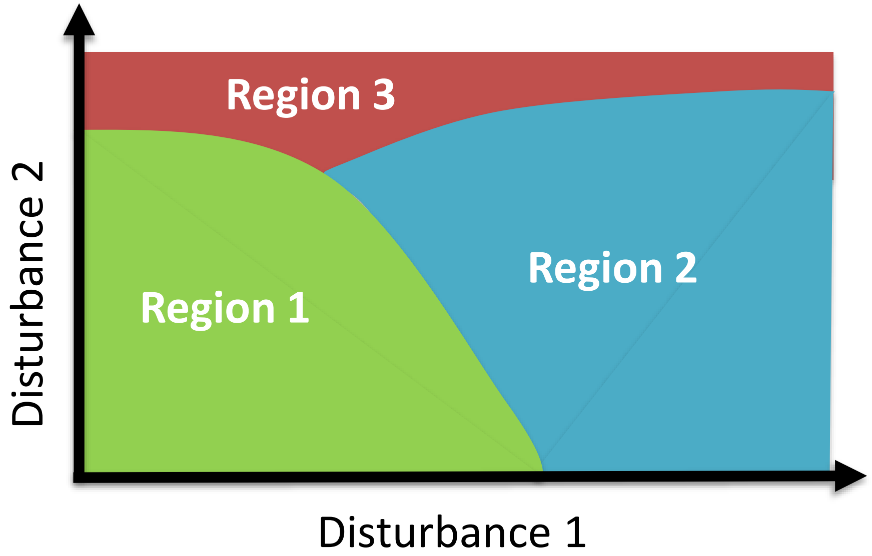
Control is about implementing optimal operation in practice

- Many cases: Solution is fully constrained, but constraints change

→Key is to control the active constraints

- In practice: Don't need to know regions

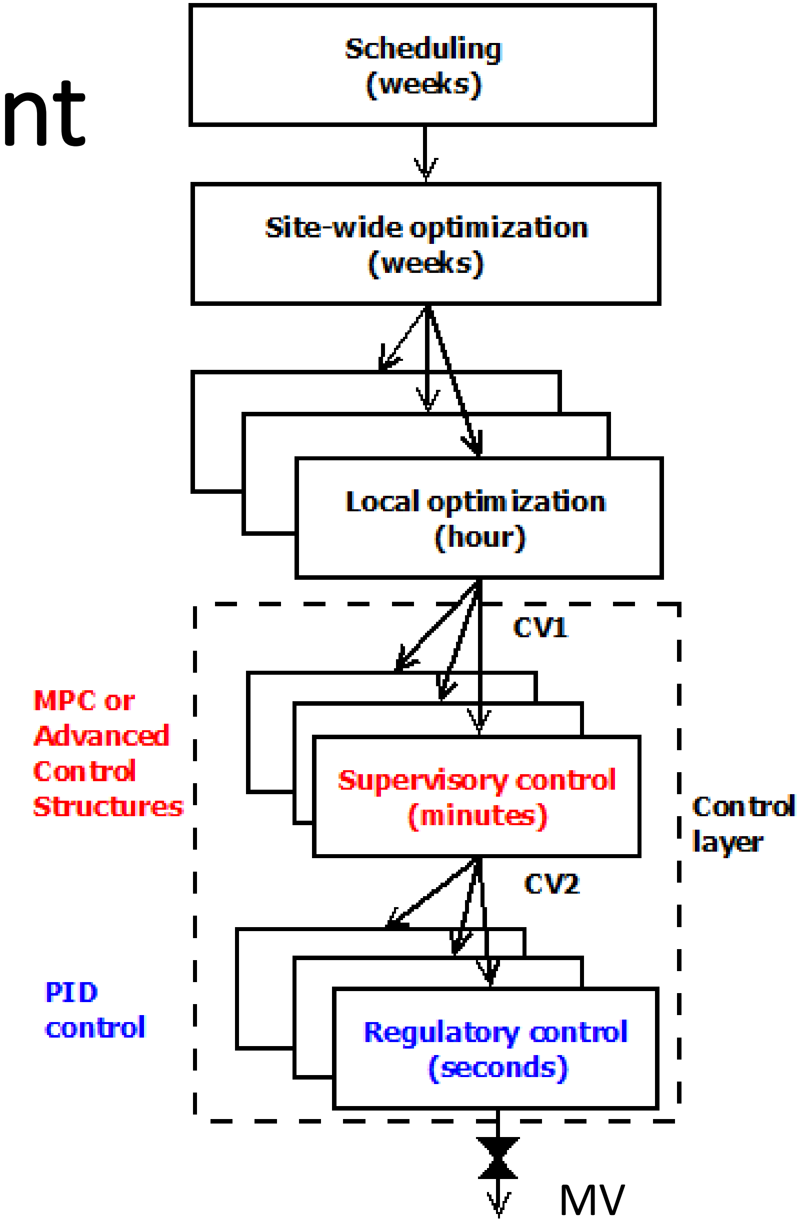
  if we can measure and control the constraints



2. Control hierarchy in a process plant

Key idea: Time scale separation

- **Optimization layer (RTO) (hour)**
  - Minimize economic cost J, satisfying constraints
- **Supervisory layer (APC or MPC) (minutes)**
  - Follow set points (CV1) from optimization layer
  - Switch between active constraints (CV1 change)
  - Look after regulatory layer
- **Regulatory control (PID) (seconds)**
  - Follow setpoints (CV2) from layers above
  - Stabilize: Control drifting variables
- **Key decisions: Select CV1 and CV2**



CV = Controlled variable
MV = Manipulated variable (process input)
RTO = Real-time optimization
APC = Conventional Advanced process control
MPC = Model predictive control
PID = Propertional-Integral-Derivative

Optimal operation of process plants

- Most people think
  - You need a detailed nonlinear model and an on-line optimizer (RTO) if you want to optimize the process
  - You need a dynamic model and model predictive control (MPC) if you want to handle constraints
  - The alternative is Machine Learning

- No! In many cases you just need to measure the constraints and use PID control
  - «Coventional advanced process control (APC)»

- How can this be possible?
  - Because optimal operation is usually at constraints
  - PID-controllers can be used to identify and control the active constraints
  - For unconstrained degrees of freedom, one often have «self-optimizing» variables

- This fact is not well known, even to control professors
  - Because most APC-applications are *ad hoc*
  - Few systematic design methods exists

Example: Optimal operation of runner

- Cost to be minimized, J=T
- One degree of freedom (u=power)
- What should we control?

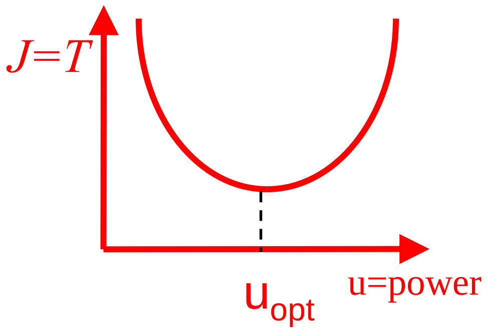
Sigurd Skogestad. «Near-optimal operation by self-optimizing control: From process control to marathon running and business systems», Comp.Chem.Engng. (2004)

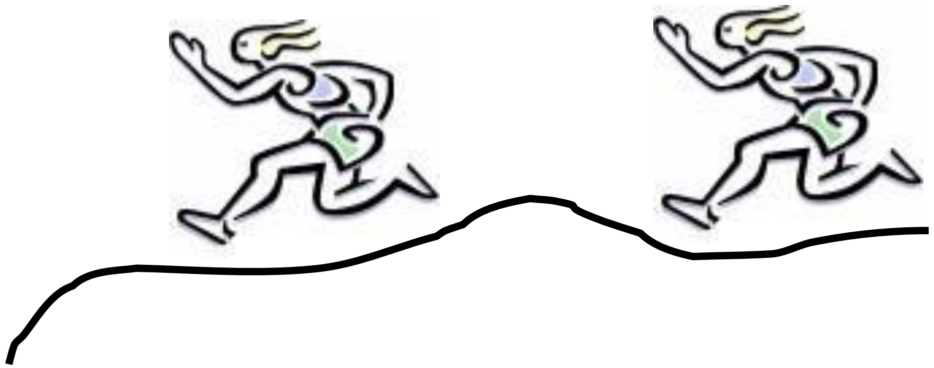
A. Optimal operation of Sprinter

- 100m. J=T
- Active constraint control:
  - Run as fast as you can (”no thinking required”)
  - CV = power (at max)

B. Optimal operation of Marathon runner

- 40 km. J=T
- What should we control? CV=?
- Unconstrained optimum



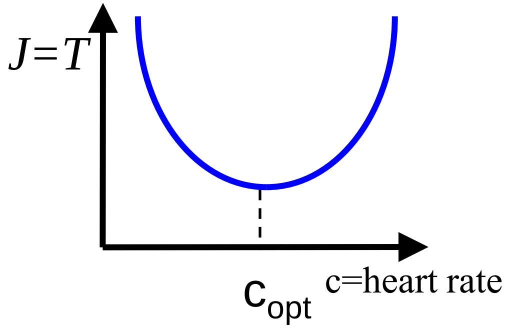


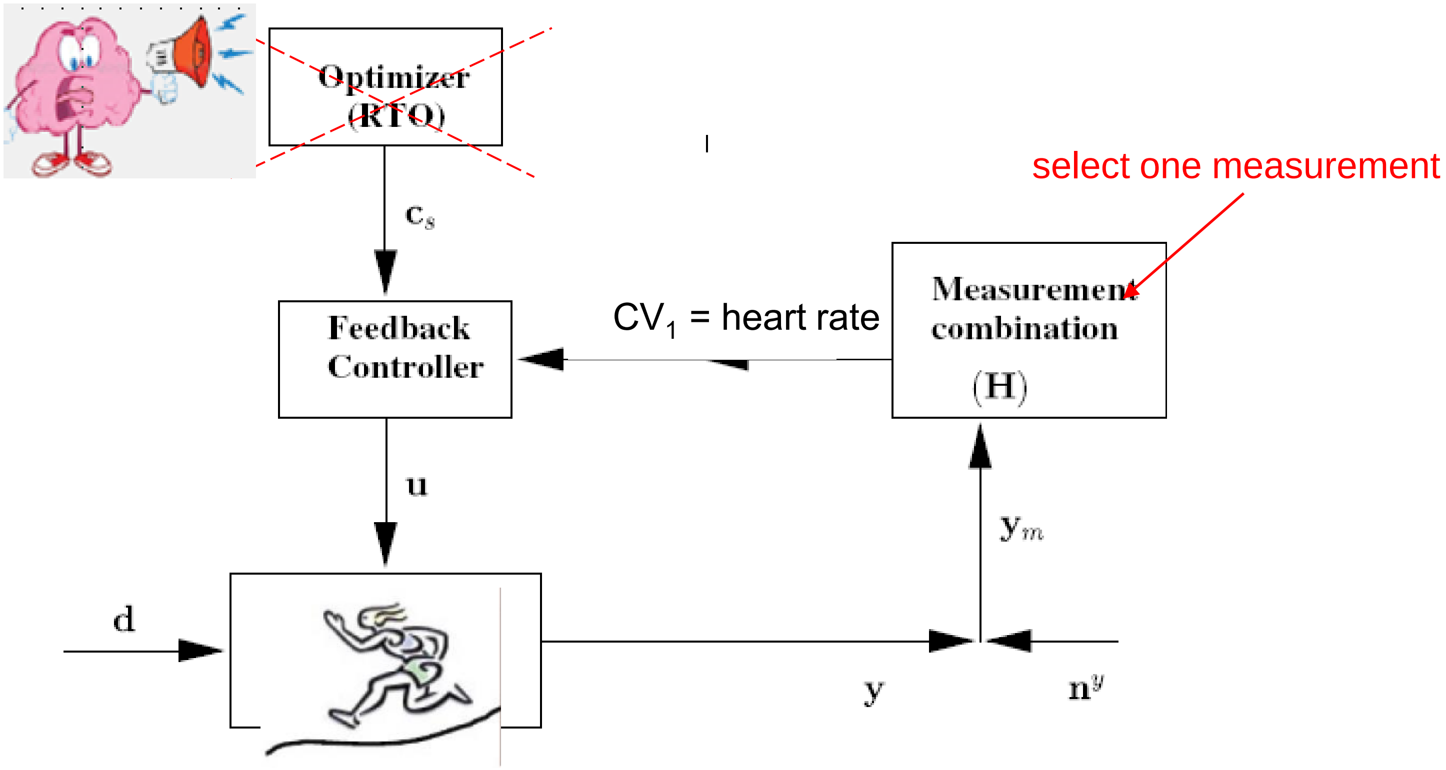
NTNU

Marathon runner (40 km)

- Any self-optimizing variable (to control at constant setpoint)?
  - c_1 = distance to leader of race
  - c_2 = speed
  - **c_3 = heart rate**
  - c_4 = level of lactate in muscles

Conclusion Marathon runner

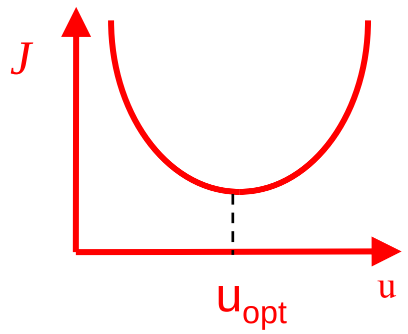




- CV = heart rate is good "self-optimizing" variable
- Simple and robust implementation
- Disturbances (d) are indirectly handled by keeping a constant heart rate
- May have infrequent adjustment of setpoint (c_s)

3. Unconstrained optimization

- Have unconstrained degree of freedom (u)
- Available measurements: y
- What should we control (c=CV1=Hy)?
  - Not at all obvious



Self-optimizing control

***Self-optimizing control** is when we can achieve an acceptable economic loss (between re-optimizations) with constant setpoint values for the controlled variables (c=CV1)*

Self-optimizing control is an old idea (Morari *et al.*, 1980):

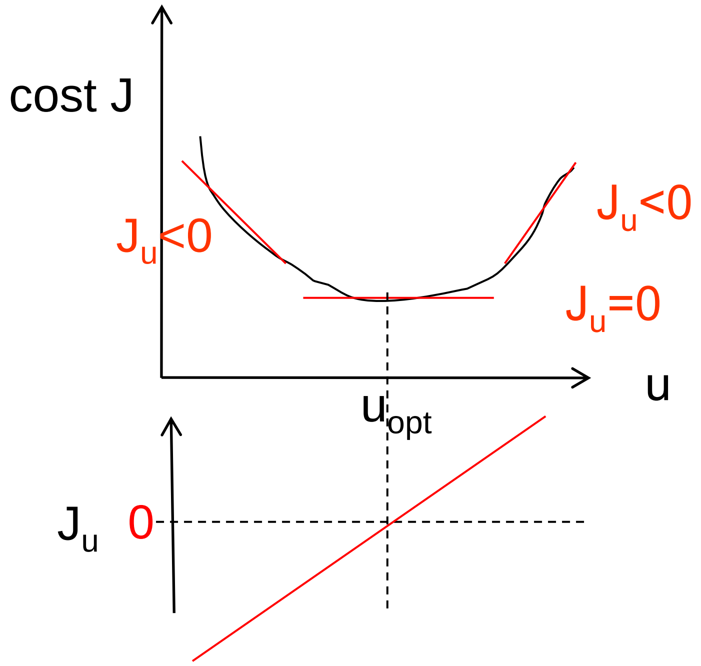
> *"We want to find a function c of the process variables which when held constant, leads automatically to the optimal adjustments of the manipulated variables, and with it, the optimal operating conditions."*

S. Skogestad ``Plantwide control: the search for the self-optimizing control structure'', J. Proc. Control, 2000.

The ideal "self-optimizing" variable is the gradient, J_u

$$c = \partial J/\partial u = J_u$$

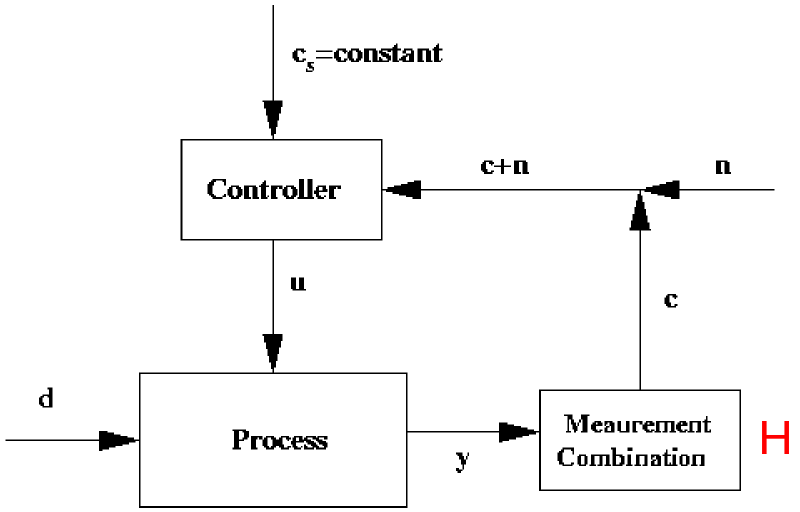
– Keep gradient at zero for all disturbances ($c = J_u = 0$)



Problem: Usually no measurement of gradient

Ideal: $c = J_u$

In practise, use available measurements: $c = H y$. Task: Select H!



- Single measurements:

$$\mathbf{c} = \mathbf{Hy} \qquad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

- Combinations of measurements:

$$\mathbf{c} = \mathbf{Hy} \qquad \mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \end{bmatrix}$$

Self-optimizing variables: Model-based methods for c=Hy

Nullspace method for H

HF=0 where $F = dy_{opt}/dd$

Proof: Want c_{opt} independent of disturbance d

Have. $y_{opt} = F\, d$, so $c_{opt} = H\, y_{opt} = HF\, d$ -> **HF=0**

Exact local method for H

Analytical solution:

$$H = G^{yT}(YY^T)^{-1} \text{ where } Y = [FW_d \quad W_{n^y}]$$

V. Alstad and S. Skogestad, ``Null Space Method for Selecting Optimal Measurement Combinations as Controlled Variables", Ind.Eng.Chem.Res, 46 (3), 846-853 (2007)

V. Alstad, S. Skogestad and E.S. Hori, ``Optimal measurement combinations as controlled variables", Journal of Process Control, Vol.19, 128-148 (2009).

Example. Nullspace Method for Marathon runner

u = power, d = slope [degrees], J=Time

y_1 = hr [beat/min], y_2 = v [m/s]

$c = Hy = h_1 y_1 + h_2 y_2$

From model or data: $F = dy_{opt}/dd = [0.25 \ \ -0.2]'$

HF = 0 **->** $h_1 f_1 + h_2 f_2 = 0.25 h_1 - 0.2 h_2 = 0$

Choose $h_1 = 1$ -> $h_2 = 0.25/0.2 = 1.25$

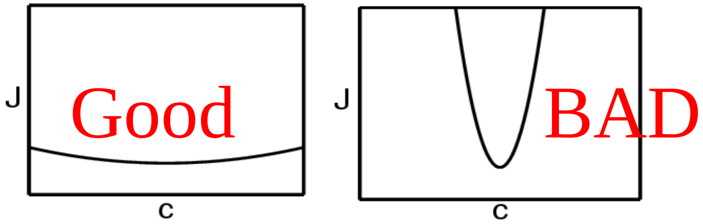
Conclusion: **c = hr + 1.25 v**

Control **c = constant** -> hr increases when v decreases (OK uphill!)

Self-optimizing variables: What should we control?

Engineering insight may be used if we don't have model

1. **The *optimal value* of c should be *insensitive* to disturbances**
   - Small $F^c = HF = dc_{opt}/dd$
2. **The *value* of c should be *sensitive* to the inputs ("maximum gain rule")**
   - Large gain, $G^c = HG^y = dc/du$
   - Equivalent: Want flat optimum



(b) Flat optimum: Implementation easy

(c) Sharp optimum: Sensitive to implementation erros

NEVER try to control a variable that reaches max or min at the optimum

In particular, never try to control directly the cost J

I.J. Halvorsen, S. Skogestad, J.C. Morud and V. Alstad, ``Optimal selection of controlled variables", *Ind. Eng. Chem. Res.*, **42** (14), 3273-3284 (2003)

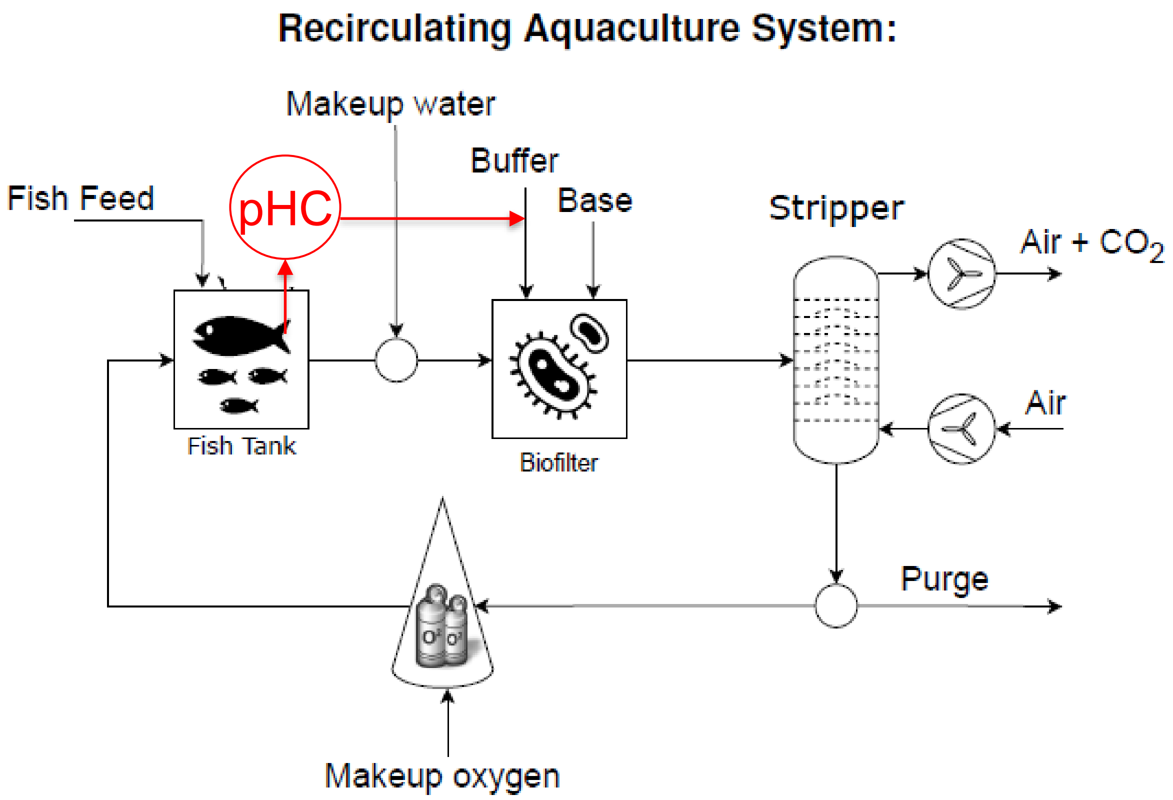
Example: Maximize growth of salmon fish (RAS)

- One unconstrained degree of freedom: Buffer/base addition
- What should we control?
- Self-optimizing variable (CV1): **pH in Fish tank**
  - **Large gain** (sensitive to changes in buffer/base)
  - Small variations in optimal setpoint (7-7.5)

  - Optimize with simple pH-controller

pHC

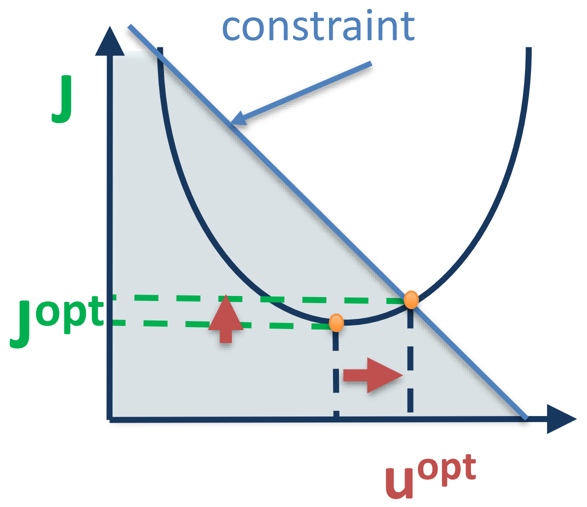
Regulatory layer: pH-control also provides stabilization
- so we have CV2=CV!=pH, which is ideal



Allyne M. dos Santos et al., Soft sensor of key components in recirculation aquaculture systems, using feedforward networks, ESCAPE-32 Toulouse, 2022 **(Poster today)**

4. Constrained optimization

- Obvious what we should control: **Active constraints**
  - Can be measured in most cases and controlled with PID-controller
- Reason for change in active constraints are
  - Disturbances (including changes in parameters and prices)
- Challenge control: Switch between active constraints



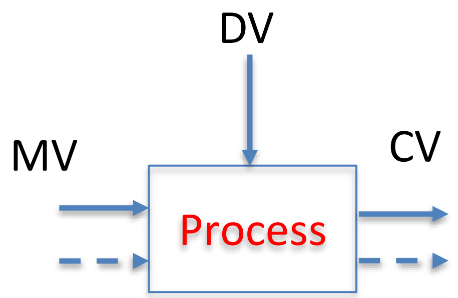
Conventional advanced control structures (ACS)

- Used when single-loop PID is not sufficient.
- Examples:
  - Cascade control
  - Feedforward control / Ratio control
  - Decoupling
  - Selectors
  - Split range control (SRC)
  - Input resetting or valve positioning control (VPC)

Can handle constraint changes

Conventional APC for changing active constraints

- Four cases:
  - MV-MV switching -> Split Range Control + 2 more options
  - CV-CV switching -> Selectors
  - Simple CV-MV switching -> Do nothing
  - Complex CV-MV switching



MV = Manipulated Variable = Input (u)
CV = Controlled Variable = Output (y)
DV = Disturbance Variable (d)

Adriana Reyes-Lua and Sigurd Skogestad, Systematic Design of Active Constraint Switching Using Classical Advanced Control Structures, Ind.Eng.Chem.Res, 2020

This is simple CV-MV switching

Optimization with PI-controller

Example: Minimize heating cost (Norway)

min u

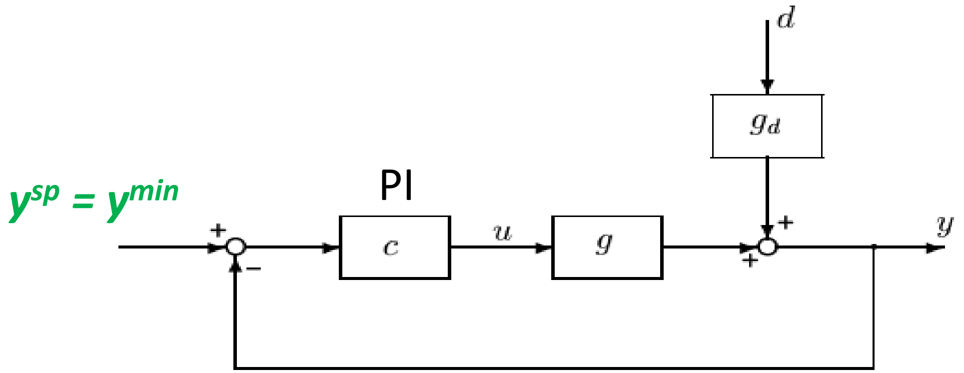
s.t. $y \geq y^{min}$

$u \geq u^{min} = 0$

(u=heating, y=temperature, y^{min}=22 °C)

- **Disturbance (d): Outdoor temperature**
- **Optimal solution has two active constraint regions:**
  1. ***CV=y = y^{min}*** → **minimum temperature (winter)**
  2. ***MV=u = u^{min}*** → **heating off (summer)**

  No unconstrained region
- **Solved with PI-controller («thermostat»)**
  - $y^{sp} = y^{min}$



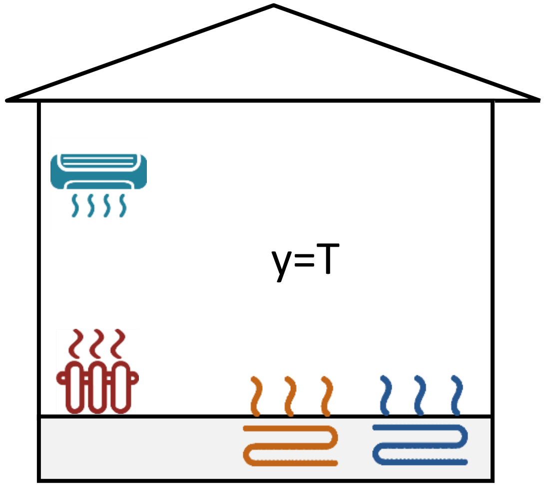
We satisfy the input saturation rule:
«When the MV (u) saturates (at 0), control of the CV (y) can be given up»

u = input = manipulated variable (MV)
y = output = controlled variable (CV)

This is MV-MV switching

Temeperature control with 4 inputs (MVs)

d=T^{amb}



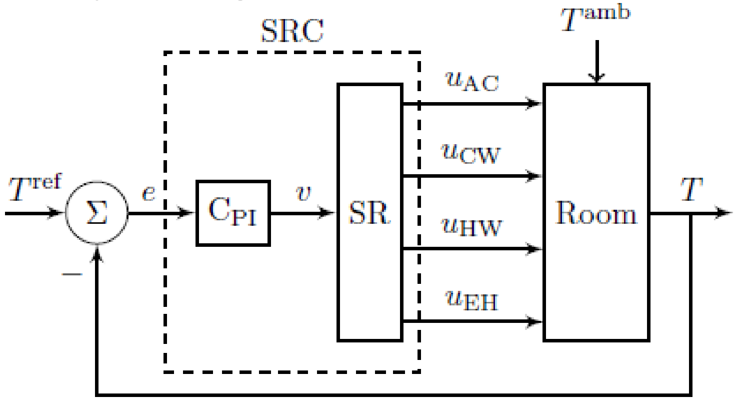
MVs:

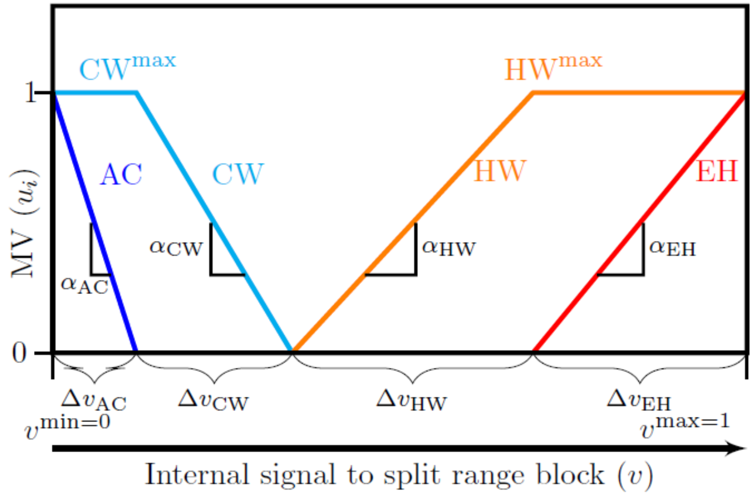
1. AC (expensive cooling)
2. CW (cooling water; cheap)
3. HW (hot water, quite cheap)
4. Electric heat, EH (expensive)

Objective: Minimize cost

- **Use cheap MVs first and use only one MV at the time (difficult with MPC)**

Solution: Split range control (SRC):





C_{PI} – same controller for all inputs (one integral time)

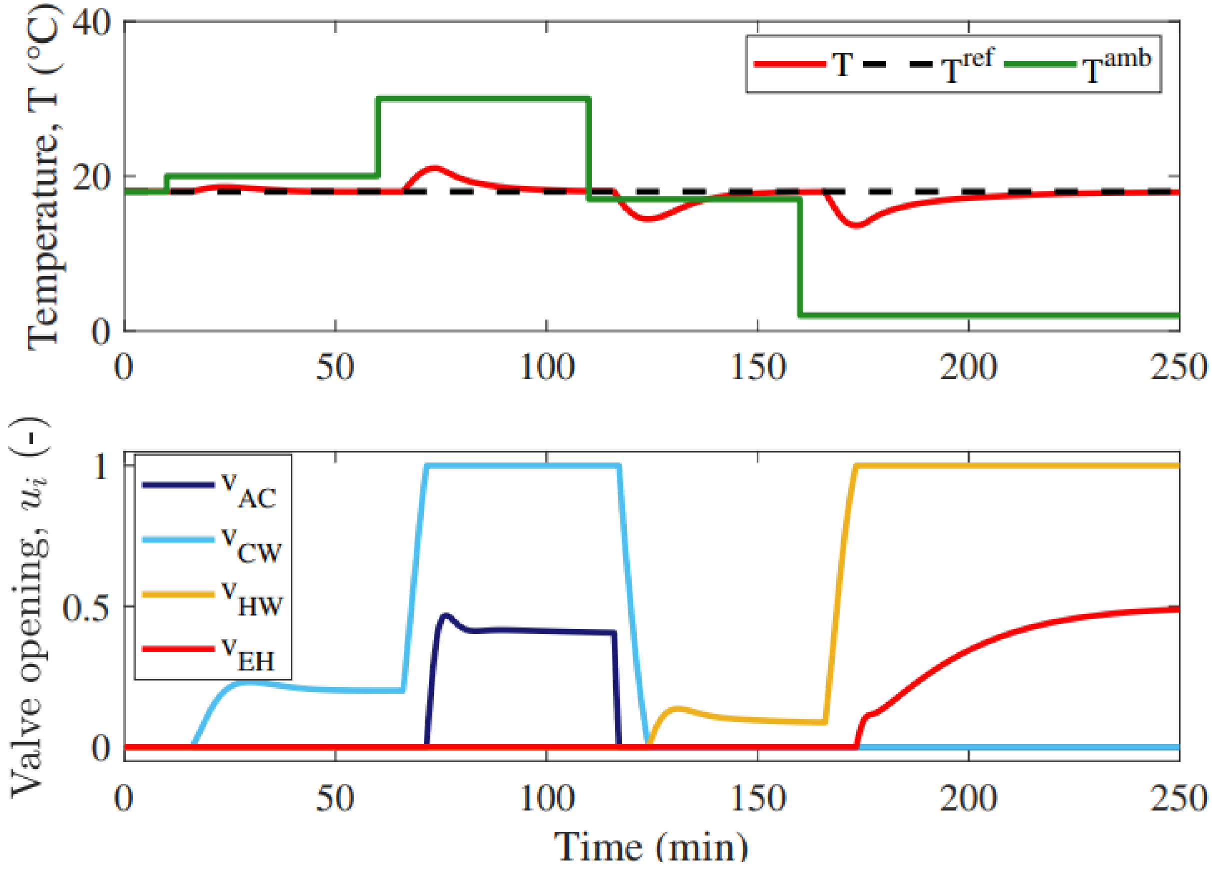
But get different gains by adjusting slopes α in SR-block

NTNU

Split-range control (SRC): Simulation of disturbances in ambient temperature.

y(t), d(t)

u(t)

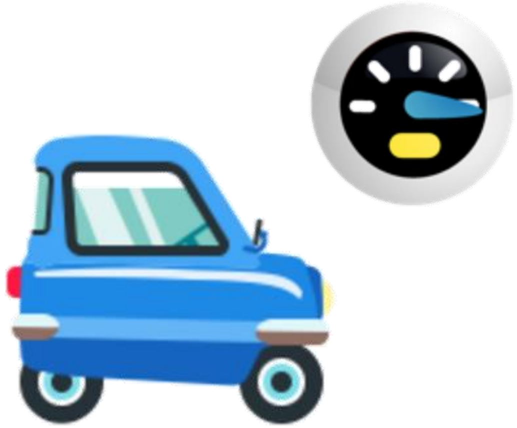


- MPC: Similar output responses (y), BUT different inputs (u). Uses both heating and cooling in some cases
- MPC: Needs dynamic model + more difficult to implement and tune

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)

This is simple CV-MV switching

Optimization with PI-controller



Example: Drive as fast as possible to airport with small car

max y

s.t. $y \leq y^{max}$

$u \leq u^{max}$

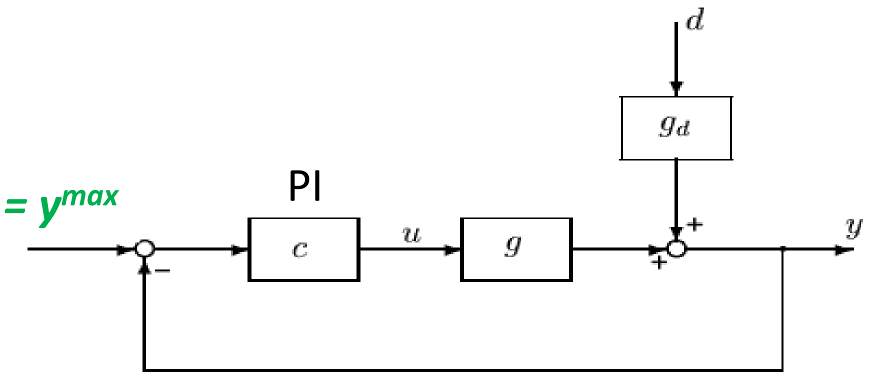
(u=power, y=speed)

Disturbance (d): Slope of road

Optimal solution has two active constraint regions:

1. ***CV=y = y^{max} = 120 km/h*** → **speed limit**
2. ***MV=u = u^{max}*** → **max power (steep hill)**

- **Solved with PI-controller («cruise controller»)**
  - $y^{sp} = y^{max}$
  - Anti-windup: I-action is off when $u=u^{max}$

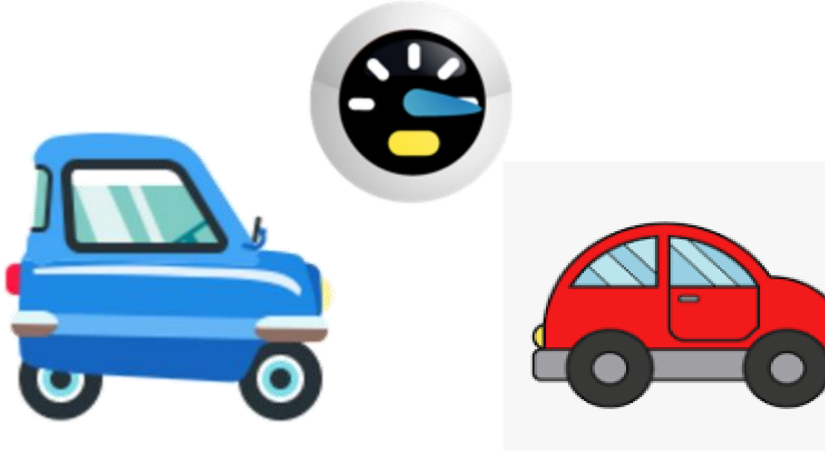


We satisfy the input saturation rule:
«When the MV (u) saturates, control of the CV (y) can be given up»

u = input = manipulated variable (MV)
y = output = controlled variable (CV)

Selector: This is CV-CV switching

Optimization with safety constraint



Example: Drive as fast as possible but safely

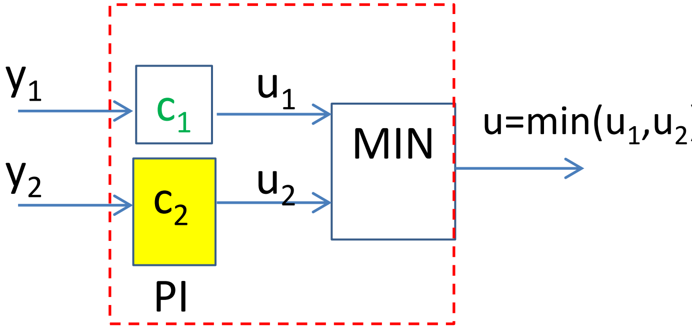
$$\max y$$
$$\text{s.t.} \quad y_1 \le y_1^{max}$$
$$u \le u^{max}$$
$$y_2 \ge y_2^{min}$$

(u=power, y=speed, y_2=distance to car in front)

Disturbances (d): Slope of road, other cars

Optimal solution has three active constraint regions:

1. $CV = y_1 = y_1^{max} = 120$ *km/h* → **speed limit**
2. $MV = u = u^{max}$ → **max power (steep hill)**
3. $CV = y_2 = y_2^{min}$ → **minimum distance (busy road)**



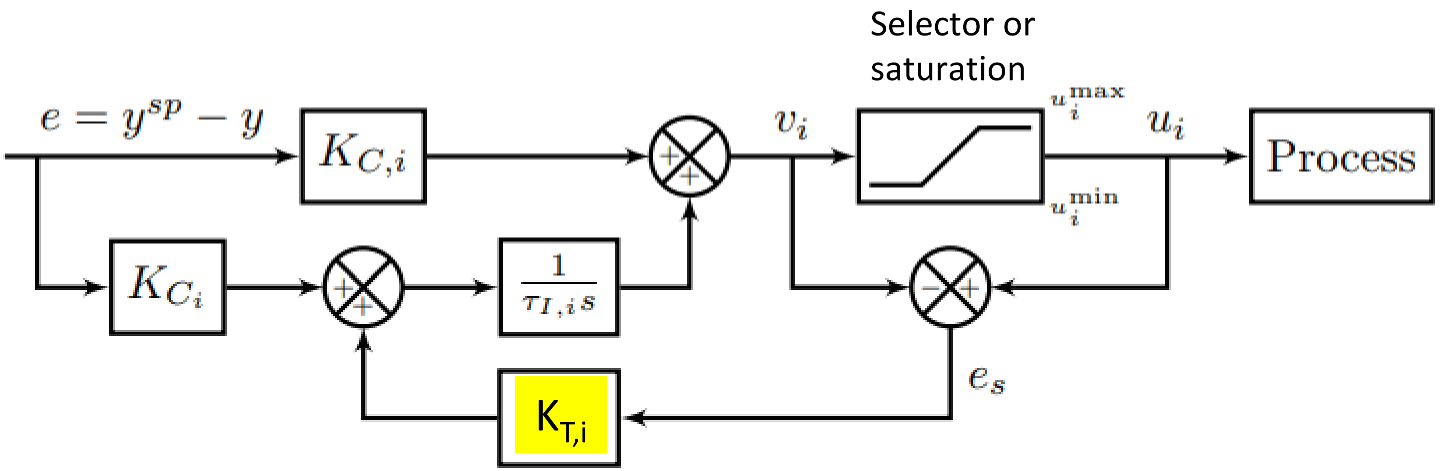
- **Solved with two PI-controllers and min-selector («adaptive cruise control»)**
  - *C_1: Cruise controller with $y_1^{sp} = y_1^{max}$*
  - C_2: Distance controller with $y_2^{sp} = y_2^{min}$
  - *Both* controllers need anti-windup (turn off when inactive)

All three constraints are satisfied with a small u

NTNU

Anti-windup

- All the controllers shown need anti-windup to «stop integration» during periods when the control action (v_i) is not affecting the process:
  - Controller is disconnected (because of selector)
  - Physical MV u_i is saturated



Anti-windup using back-calculation. Typical choice for tracking constant, K_T=1

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
- Order does not matter if problem is feasible
- If infeasible: Put highest priority constraint at the end

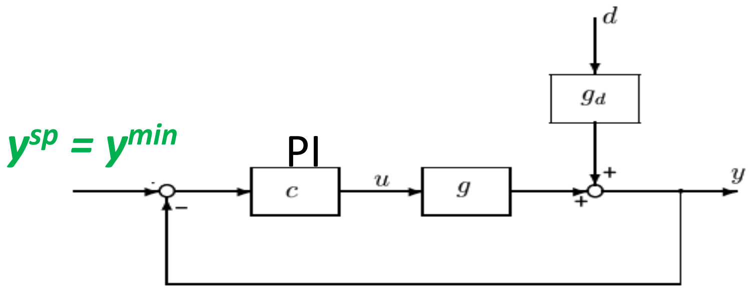
"Systematic design of active constraint switching using selectors." Dinesh Krishnamoorthy , Sigurd Skogestad. Computers & Chemical Engineering, 2020

Valves have “built-in” selectors

- A min-flow (z=0) gives a “built-in” max-selector (to avoid negative flow)
- A max-flow (z=1) gives a “built-in” min-selector
- So it’s not necessary to add these as selector blocks (but it will not be wrong).
  - Both will always be satisfied because physical input constraints can never be violated.
  - There is no danger of infeasibility /inconsistency here because we cannot have both z=0 and z=1 at the same time.

This is simple CV-MV switching

Anti-surge control



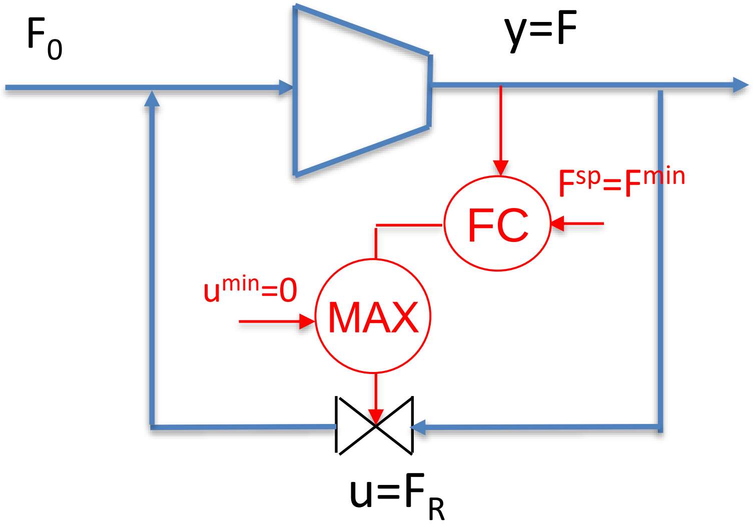
Minimize compression cost but keep safe operation ($F>F^{min}$)

min u

s.t. $y \geq y^{min}$ *(safety constraint)*

$u \geq u^{min} = 0$

(u= F_R=recycle flow, y=F=flow in compressor)



Disturbance (d): Feed flow F_0

Optimal solution has two active constraint regions:

1. $CV=y = y^{min}$ (for small F_0)
2. $MV=u = u^{min} = 0$ (for large F_0)

Solved with PI-controller («anti-surge control»)

- $y^{sp} = y^{min}$
- Anti-windup: I-action is off when $u=u^{min}=0$

MAX-block to avoid negative flow.
Not needed because the input (valve) has «built-in» $u \geq 0$.

We satisfy the input saturation rule:
«When the MV (u) saturates, control of the CV (y) can be given up»

Selector: This is CV-CV switching

Furnace control with safety constraint

Input (MV)

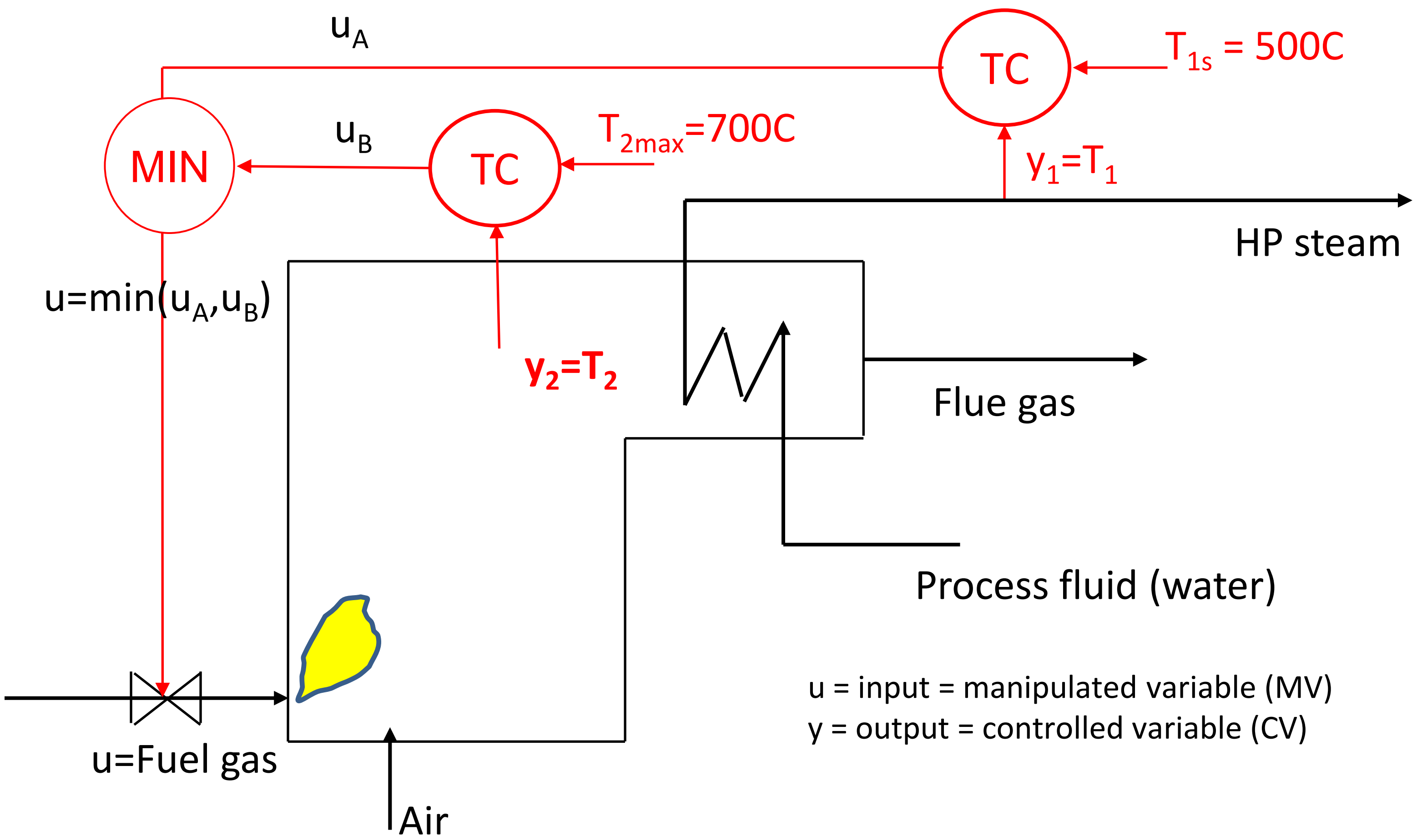
u = Fuel gas flowrate

Output (CV)

y_1 = process temperature T_1 (with desired setpoint)

y_2 = furnace temperature T_2 (max constraint)

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



u = input = manipulated variable (MV)

y = output = controlled variable (CV)

"Systematic design of active constraint switching using selectors." Dinesh Krishnamoorthy , Sigurd Skogestad. Computers & Chemical Engineering, 2020

NTNU

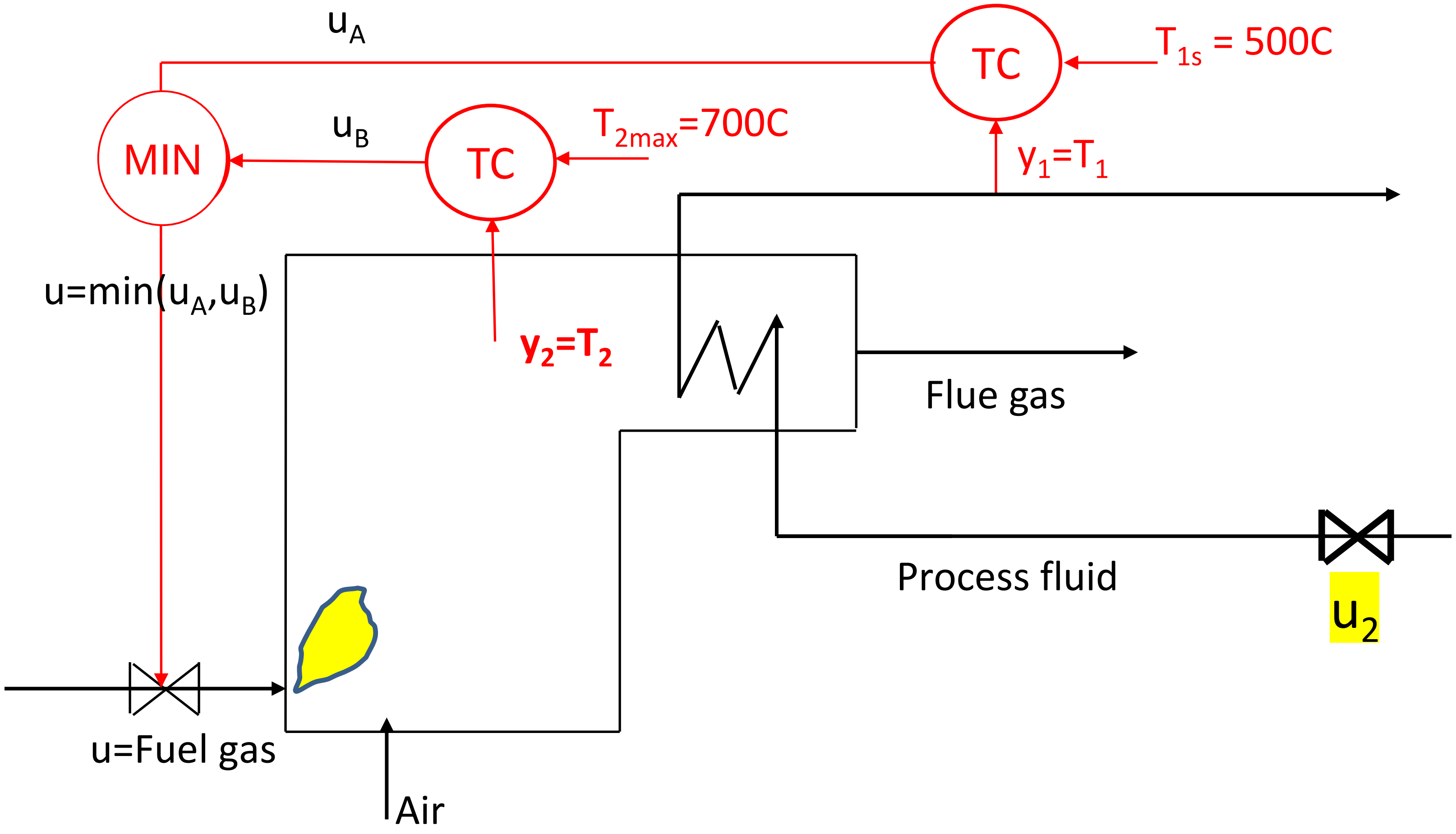
Furnace control : Cannot give up control of y_1=T_1. What to do?

Inputs (MV)

- u = Fuel gas flowrate
- u_2 = Process flowrate

Output (CV)

- y_1 = process temperature T_1 (with desired setpoint)



This is complex CV-MV switching

Cannot give up controlling T_1

Solution: Cut back on process feed (u_2) when T_1 drops too low

Inputs (MV)

- u = Fuel gas flowrate
- u2 = Process flowrate

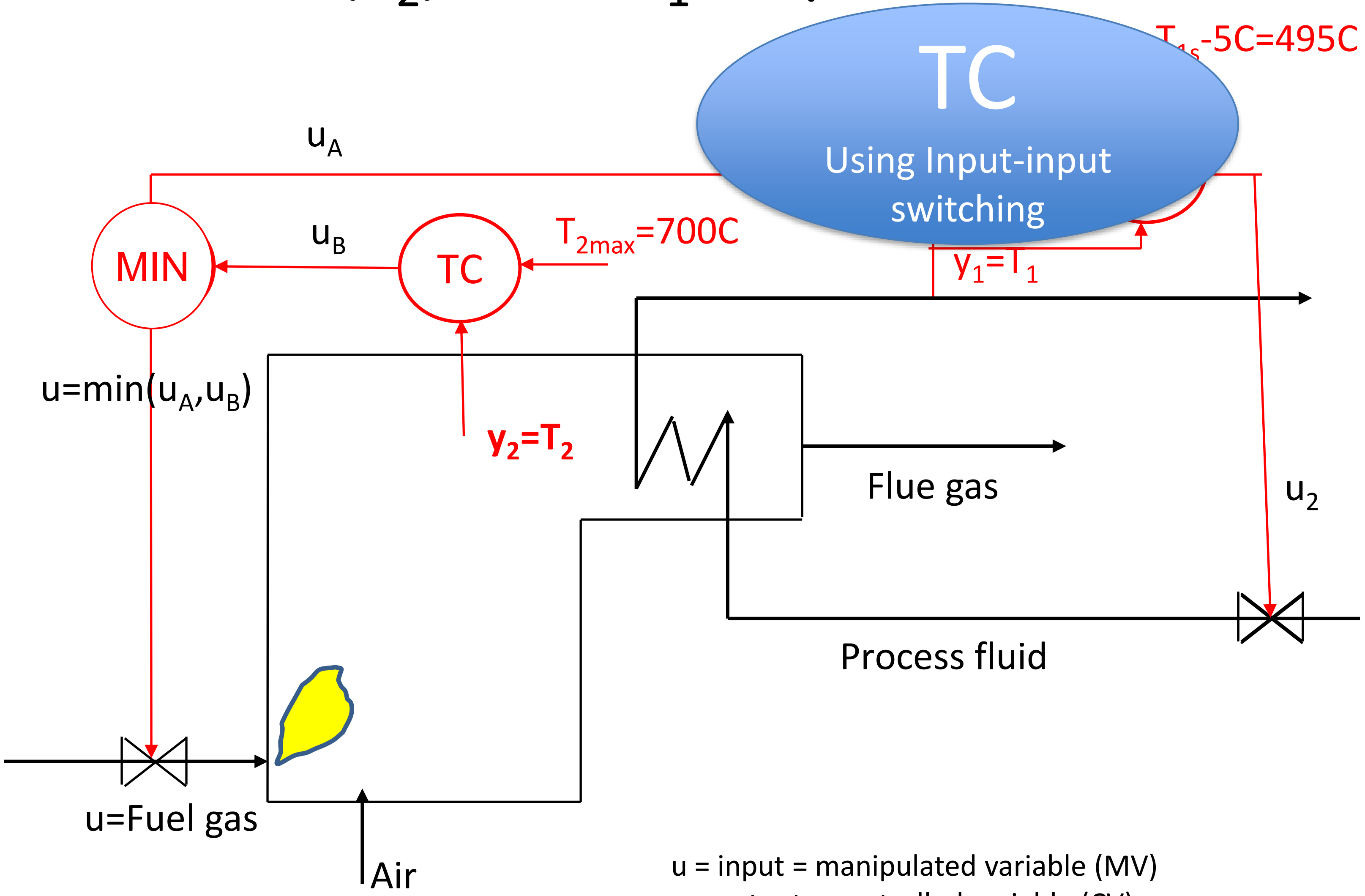
Output (CV)

- y_1 = process temperature (with desired setpoint)

Note: Standard Split Range Control is not good here.
Could be two reasons for too little fuel

- Fuel is cut back by override (safety)
- Fuel at max,

So don't know limit for MV1 to use in SRC-block.



u = input = manipulated variable (MV)
y = output = controlled variable (CV)

NTNU

This is complex CV-MV switching

Cannot give up controlling T_1

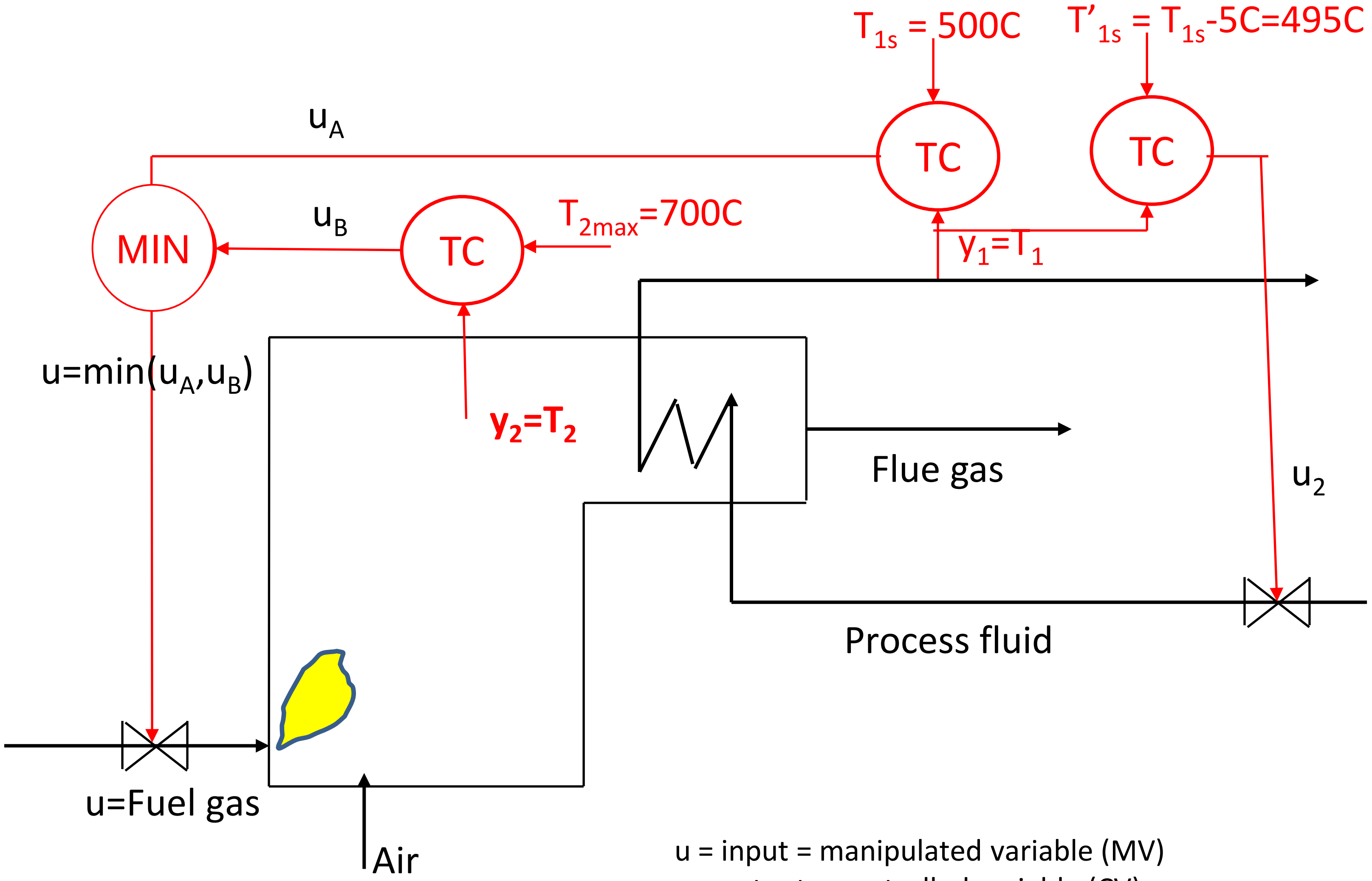
Solution: Cut back on process feed (u_2) when T_1 drops too low

Inputs (MV)

- u = Fuel gas flowrate
- u2 = Process flowrate

Output (CV)

- y_1 = process temperature (with desired setpoint)



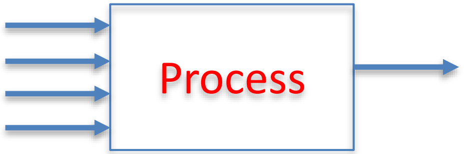
- Solution: Two controllers with different setpoints

u = input = manipulated variable (MV)

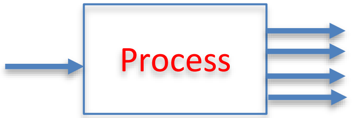
y = output = controlled variable (CV)

Summary constraint switching:
Only three different cases (or maybe four)

1. MV-MV switching
   - Need many MVs to cover whole steady-state range
   - Use only one MV at a time
   - Three options: 1. Split range control, 2. Different setpoints, 3. Valve position control (VPC)



2. CV-CV switching («override»)
   - Must select between CVs
   - Only one option: Many controllers with Max-or min-selector



3. CV-MV switching (because MV saturates)

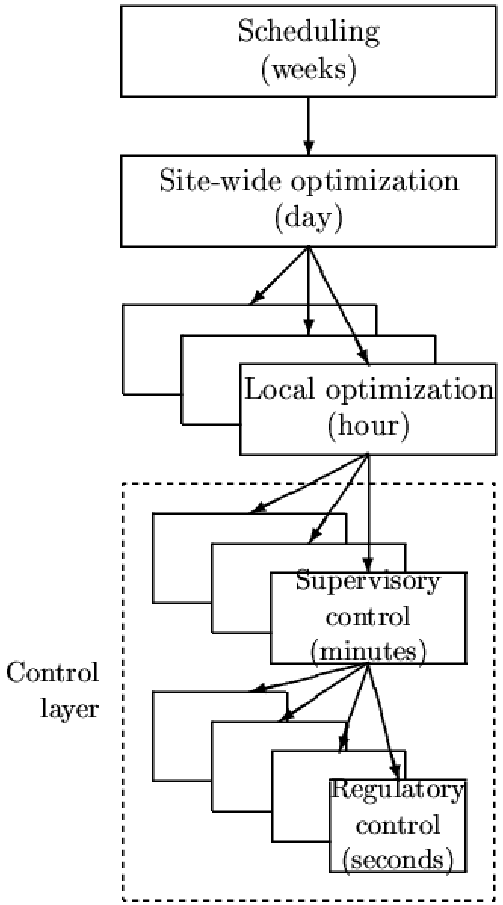
3A. Simple: CV can be given up (follow «input saturation rule»)
   - Don’t need to do anything (except anti-windup in controller)

3B. Complex: CV cannot be given up
   - Combine MV-MC switching (three options) with CV-CV switching (selector)

Oops…out of time

- Because the timr for the plenary was reduced from 60 to 40 minutes because of delays, I only got to this point during my presentation in Toulouse
- But I think it was enough to give the audience the message:
  - Put optimization into the control layer whenever feasible
  - It's a complement and not alternative to online model-based optimization

5. Systematic procedure for designing control system that achieves optimal operation

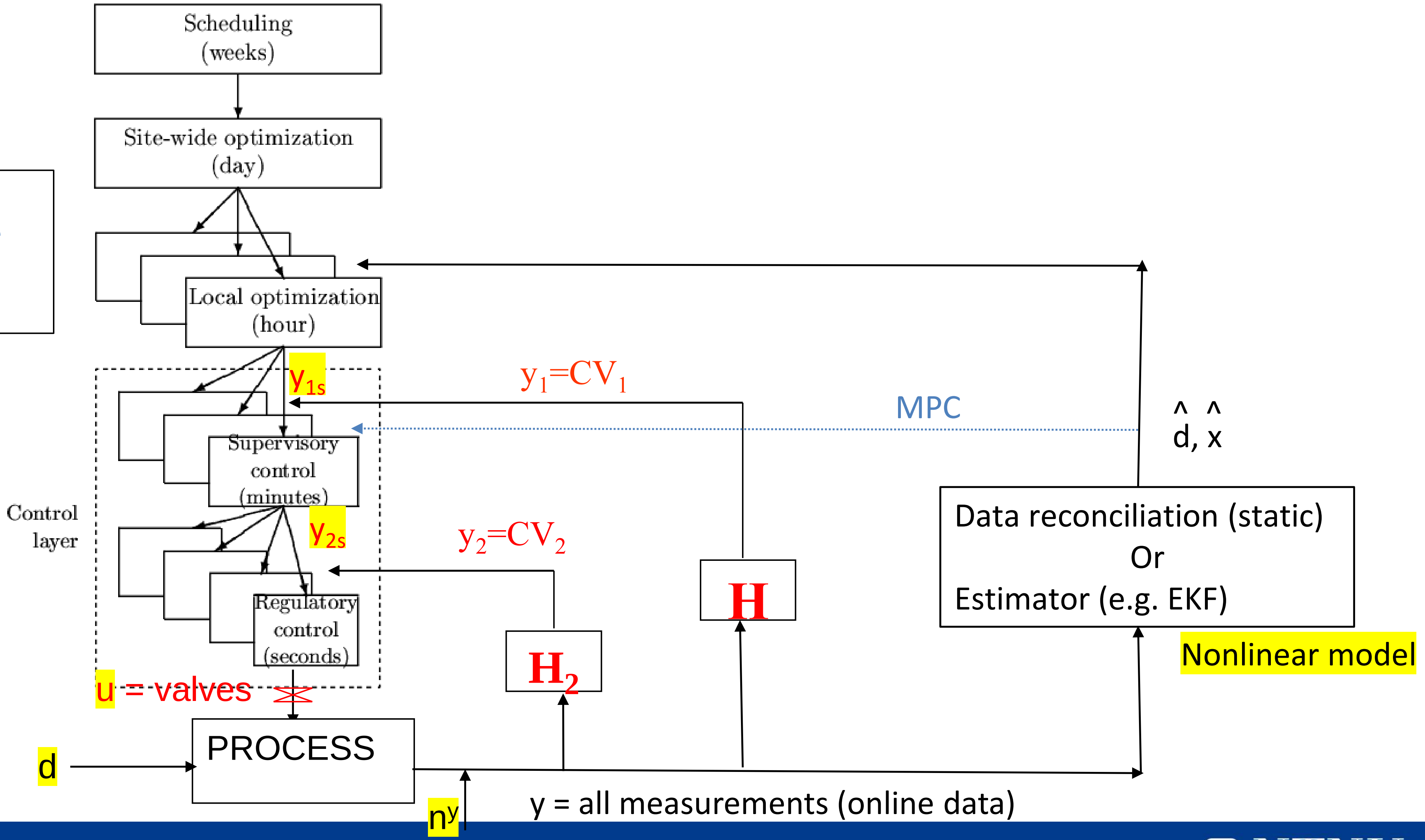


Use of models and data

RTO layer:
- Nonlinear model of whole process
- usually physical and static

MPC layer:
- Multivariable dynamic linear model for each unit
- usually from data

PID-layer:
- Dynamic linear model for each loop
- usually from data.



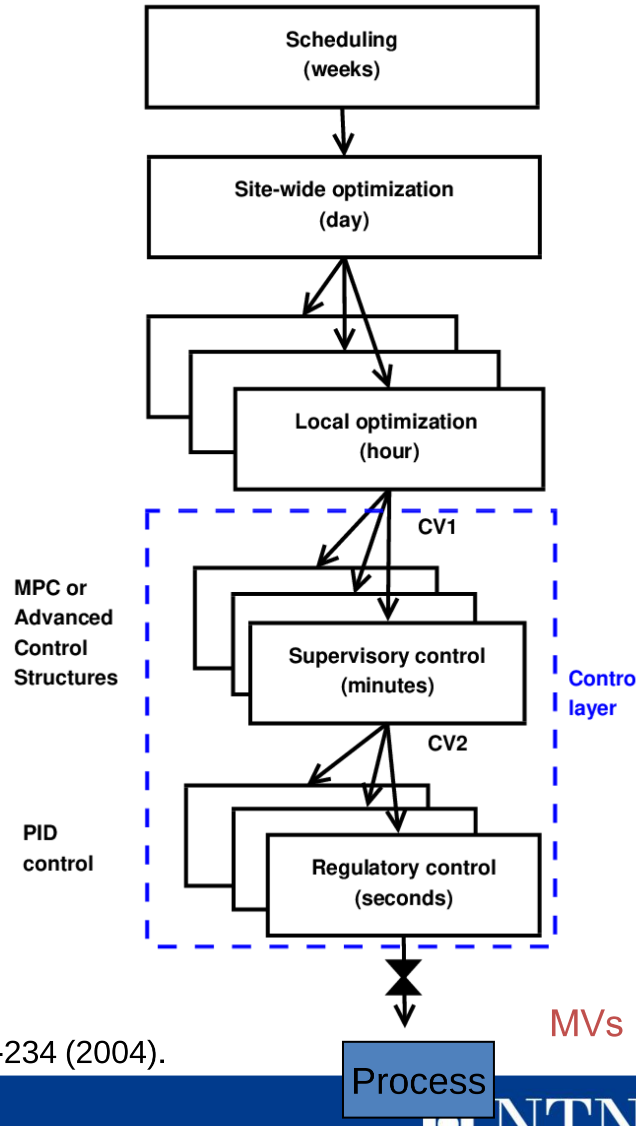
Systematic procedure for designing plantwide control system

Start "top-down" with economics:

- Step 1: Define operational objectives and constraints
- Step 2: Optimize steady-state operation
- Step 3: Decide what to control (CV1 and CV2)
- Step 4: Choose TPM location

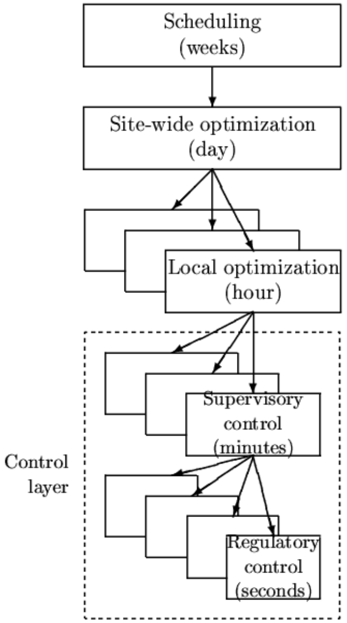
Then design control system bottom-up:

- Step 5: Regulatory control
- Step 6: "Advanced/supervisory control" system
- Step 7: Real-time optimization (Do we need it?)



S. Skogestad, ``Control structure design for complete chemical plants'', *Computers and Chemical Engineering*, **28** (1-2), 219-234 (2004).

Example: Bicycle riding
Note: Design starts from the bottom



- *Regulatory control (step 5)*:
  - First need to learn to stabilize the bicycle
    - CV2 = y_2 = tilt of bike
    - MV = body position

- *Supervisory control (step 6)*:
  - Then need to follow the road.
    - CV1 = y_1 = distance from right hand side
    - MV=$CV2_s$
  - Usually a constant setpoint policy is OK, e.g. y_{1s}=0.5 m

- *Optimization layer (step 7)*:
  - Which road to follow?
  - RTO = GPS

Systematic design of simple advanced controllers (APC)

- First design simple control system for nominal operation
  - With single-loop PID control we need to make pairing between inputs (MVs) and outputs (CVs):
  - Should try to follow two rules
    1. «Pair close rule» (for dynamics). Pair such that we have small effective delay and large gain
       - This is to get fast control and avoid instability
    2. «Input saturation rule»: «Pair MV that may saturate with CV that can be given up (at least when the MV constraint is reached)".
       - This avoids loss of control
       - Gives simple CV-MV switching

Systematic design of simple advanced controllers (APC)

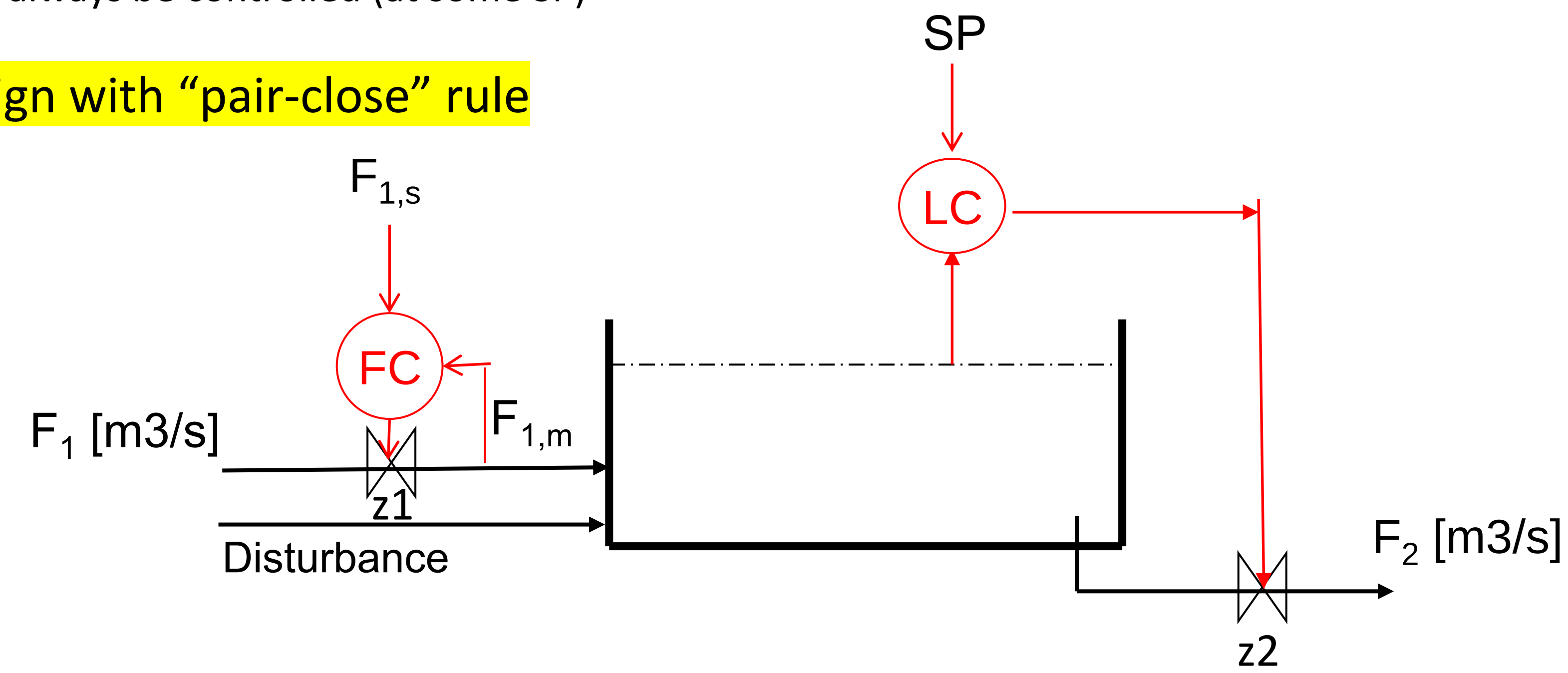
- First design simple control system for nominal operation
  - With single-loop PID control we need to make pairing between inputs and outputs:
  - Should try to follow two rules
    1. «Pair close rule». Pair such that we have fast reponse and large gain
       - This is to get fast control and avoid instability
    2. «Input saturation rule»: «Pair MV that may saturate with CV that can be given up".
       - This avoids loss of control
       - Gives simple CV-MV switching

- Then make a list of possible new contraints that may be encountered (because of disturbances, parameter changes, price changes)
- Reach constraint on new CV
  - Simplest: Find an unused input (simple CV-MV switching)
  - Otherwise: CV-CV switching using selector
- Reach constraint on MV (which is used to control CV)
  - Simplest (If we followed input saturation rule):
    - Can give ip controlling CV (Simple CV-MV switching)
    - Don't ned to do anything
  - Otherwise (if we cannot give up controlling CV)
    - Simplest: Find an unused input
      - MV-MV switching
    - Otherwise: Pair with a MV that already controls another CV
      - Complex CV-MV switching
      - Must combine MV-MV and CV-CV switching
- Is this always possible? No, pairing inputs and outputs may be impossible with many constraints.
- May then use MPC instead

Example : Level control

u1 = z1 (inflow valve position)
u2 = z2 (outflow valve position) (likely to saturate)
y1 = F1 (inflow): Should be controlled at setpoint $F_{1,s}$ (if possible)
y2 = level: must always be controlled (at some SP)

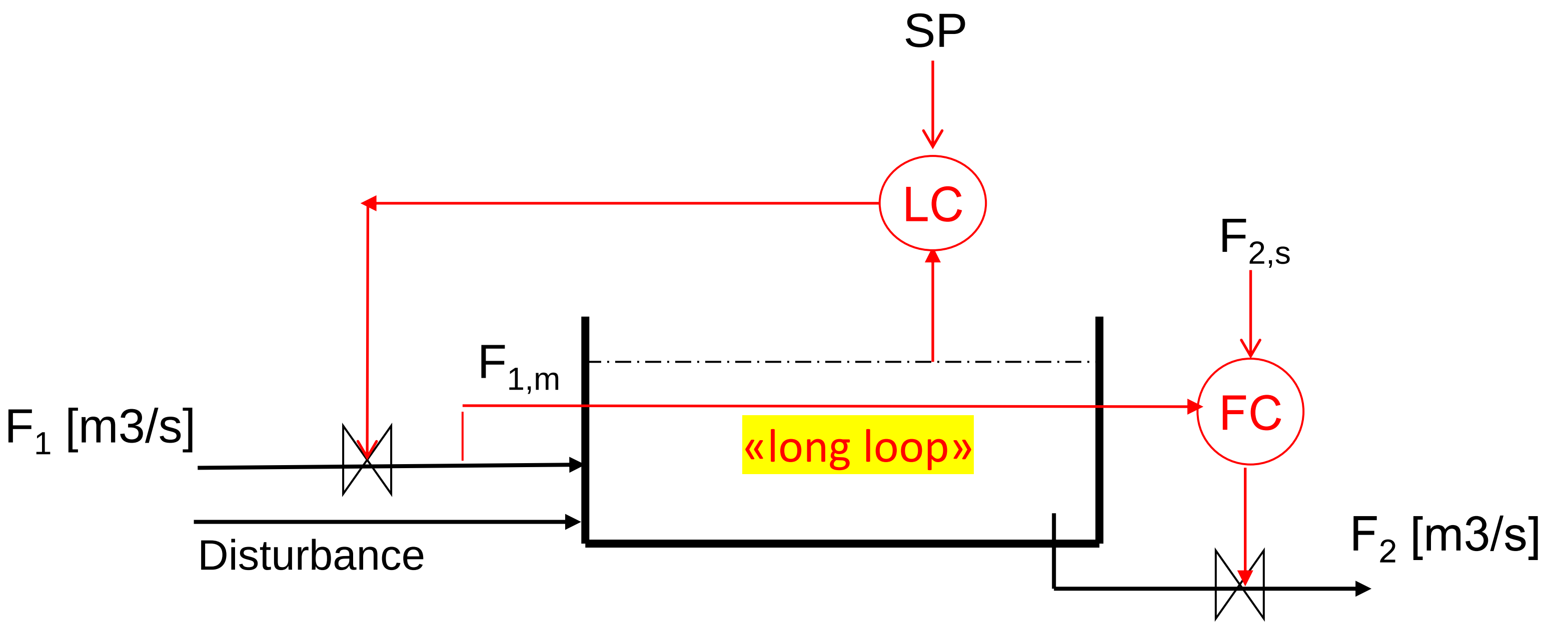
Nomimal design with “pair-close” rule



Problem: outflow-valve may saturate at fully open (z2=1) and then we lose level control
Note: We did not following the “input saturation rule” which says:
Pair MV that may saturate (z2) with CV that can be given up (F1)

This gives simple CV-MV switching (if z2 saturates at fully open)

Nominal design with Reverse pairing (follows "input saturation rule"):



BUT with Reverse pairing: Get "long loop" for flow control
In addition: loose control of y2= level if z1 (F1-valve) saturates

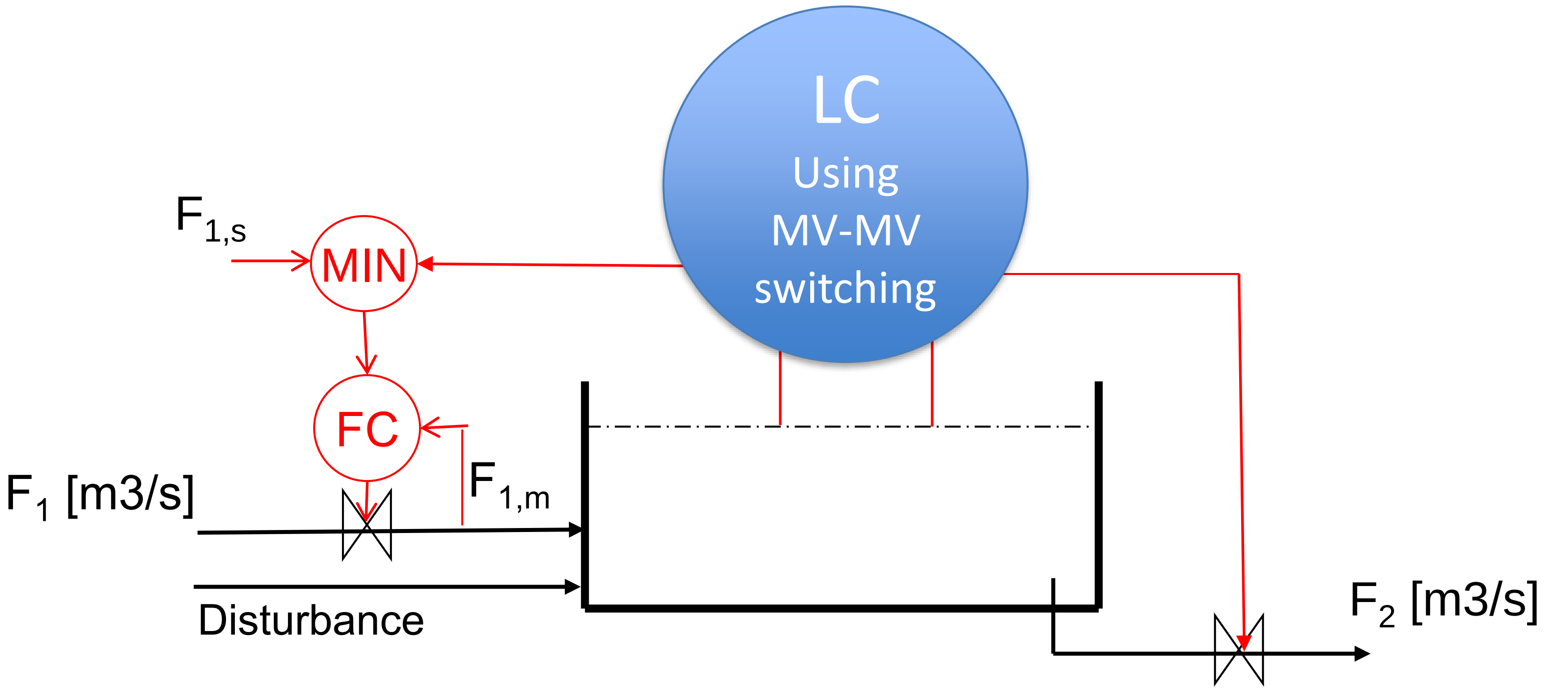
«Long loop» = Works through other loops

This is complex CV-MV switching

Alternative solution: Follow “Pair close”-rule and use Complex CV-MV switching.

When z2 saturates at max, use the other MV (z1) for level control and give up controlling F1

Get: “Bidirectional inventory control”



- Avoid long loop for control of F1
- Works both when F1-valve or F2-valve saturate at open

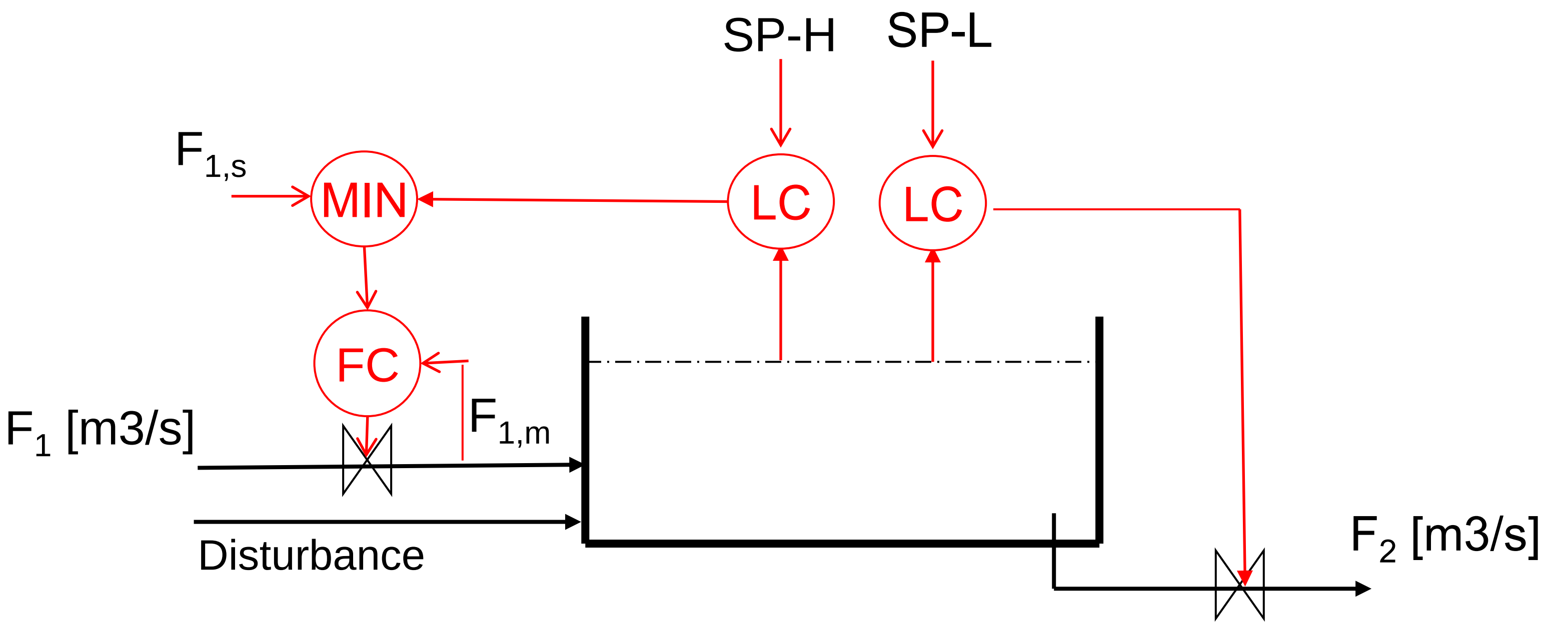
Overall: seems to be the best solution

This is complex CV-MV switching

Alternative solution: Follow "Pair close"-rule and use Complex CV-MV switching.

When z2 saturates at max, use the other MV (z1) for level control and give up controlling F1

Get: "Bidirectional inventory control"



Recommended: Two controllers
SP-L = low level setpoint
SP-H = high level setpoint

Use of two setpoints is good for using buffer dynamically!!

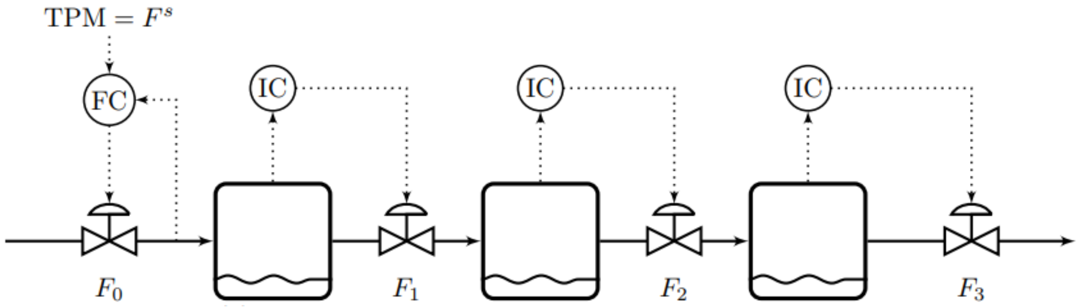
Generalization of bidirectional inventory control

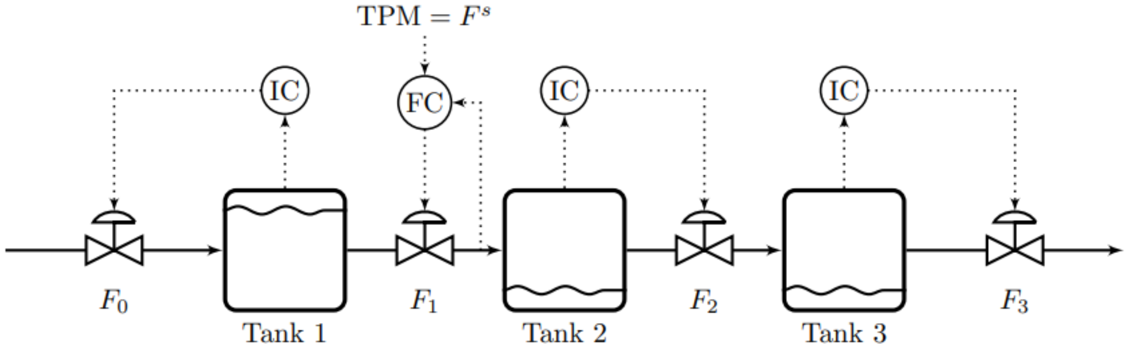
Radiation rule for Inventory control

(Georgakis)

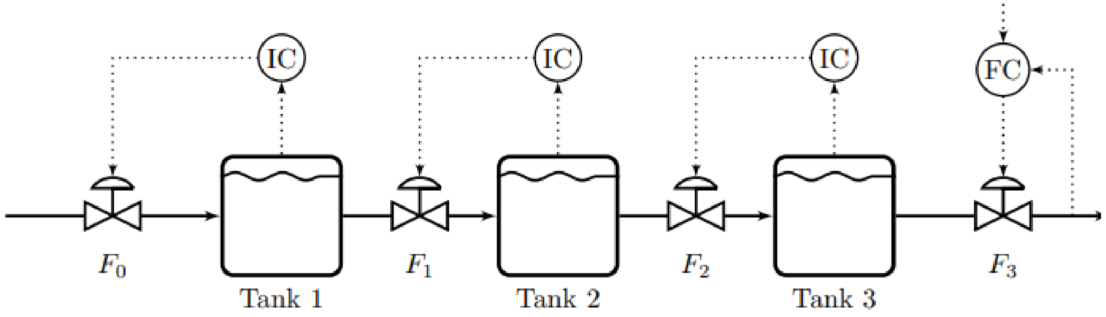
«Inventory loops are radiating around given flow (TPM)»

- Follows «pair-close» rule
- Avoids «long loops» for inventory control





(b) TPM at F_1. Inventory control radiating around the TPM.



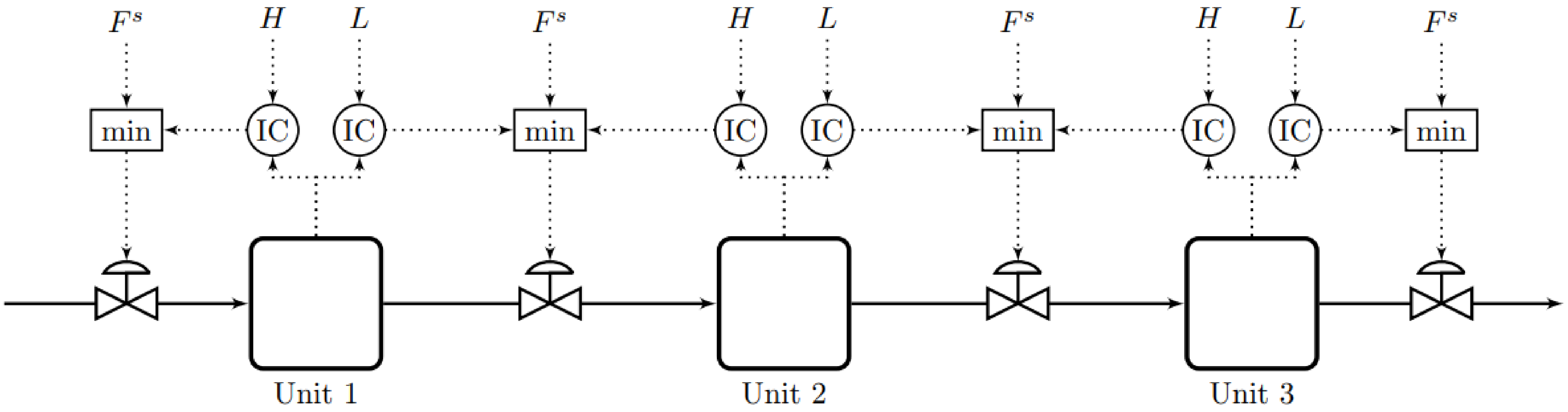
(d) TPM at F_3. Inventory control in direction opposite of flow.

TPM = throughput manipulator
(located at bottleneck = flow constraint)

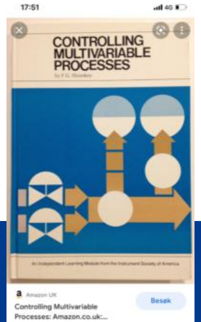
Very smart selector strategy: **Bidirectional inventory control**

Reconfigures automatically with optimal buffer management!!

Max flow:
$F^s=\infty$



F.G. Shinskey, «Controlling multivariable processes», ISA, 1981



Cristina Zotica, Krister Forsman, Sigurd Skogestad ,»Bidirectional inventory control with optimal use of intermediate storage», Computers and chemical engineering, 2022

Example: Optimal control of a cooler

Main control objective:

$y_1 = T_H = T_H^{sp}$

Secondary objective (can be given up)

$y_2 = F_H = F_H^{sp}$

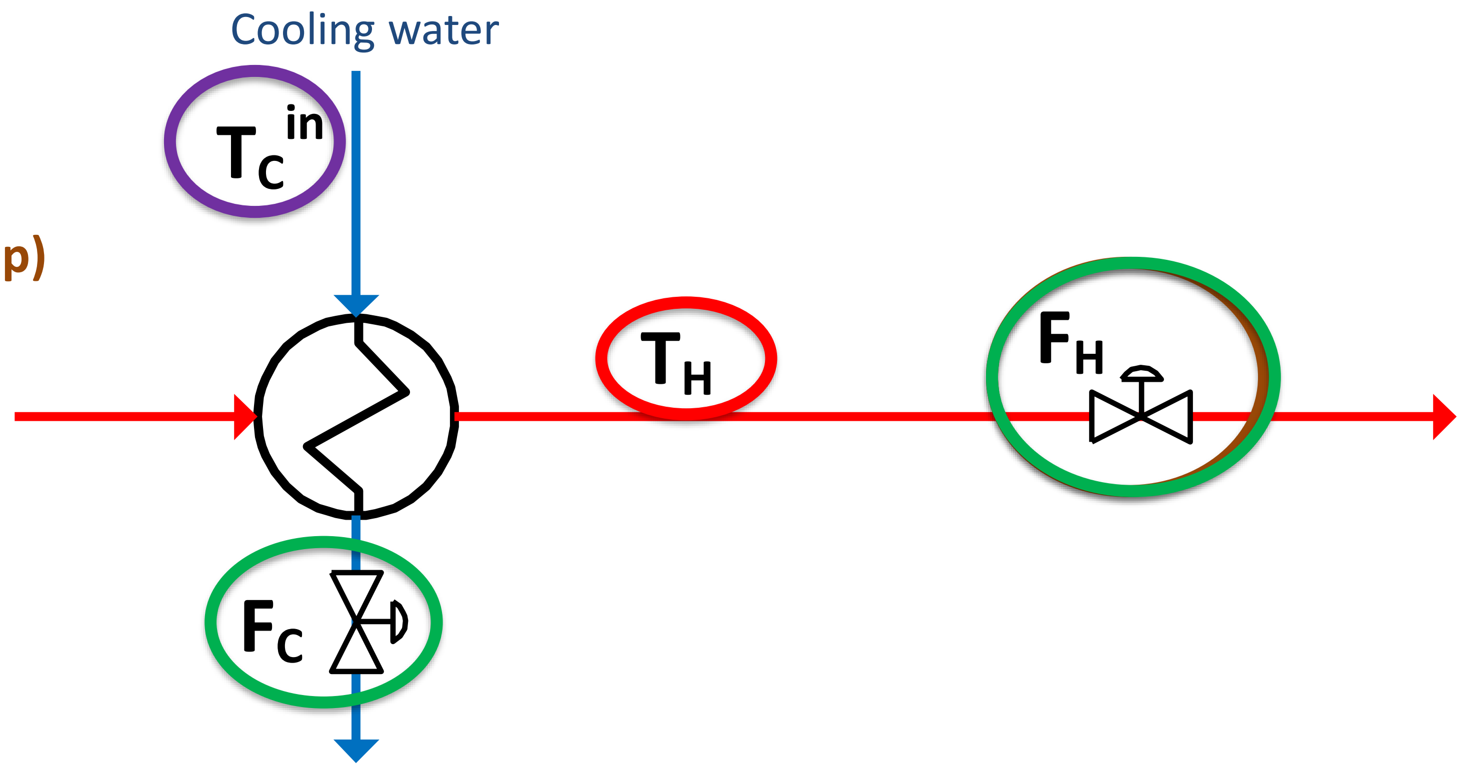
Manipulated Variables:

$u_1 = z_C$, $u_2 = z_H$

Both valves may satúrate at max

Disturbance:

T_C^{in}



Optimization of Cooler

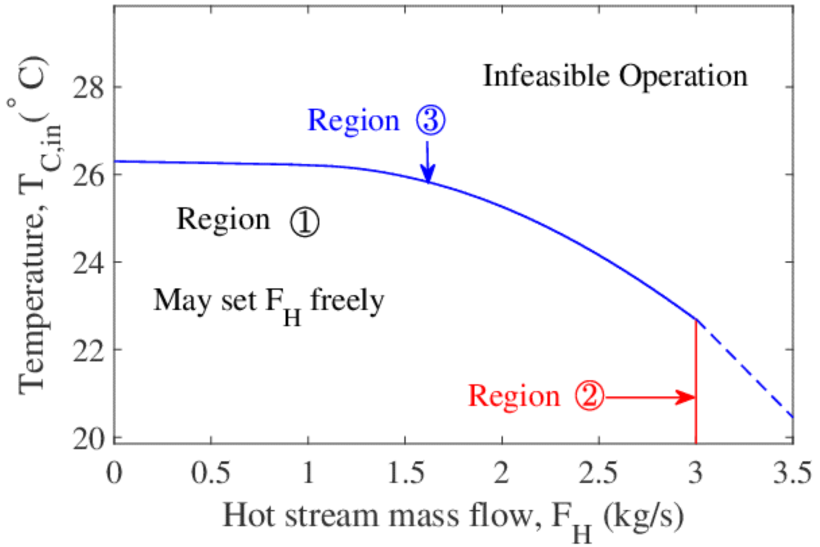
max y_2 (throughput)

s.t. $y_1 = y_1^{sp}$ ← temperature

$u_1 \leq u_1^{max}$

$u_2 \leq u_2^{max}$ ← max. throughput

$y_2 \leq y_2^{sp}$ ← desired throughput



Active constraint regions:

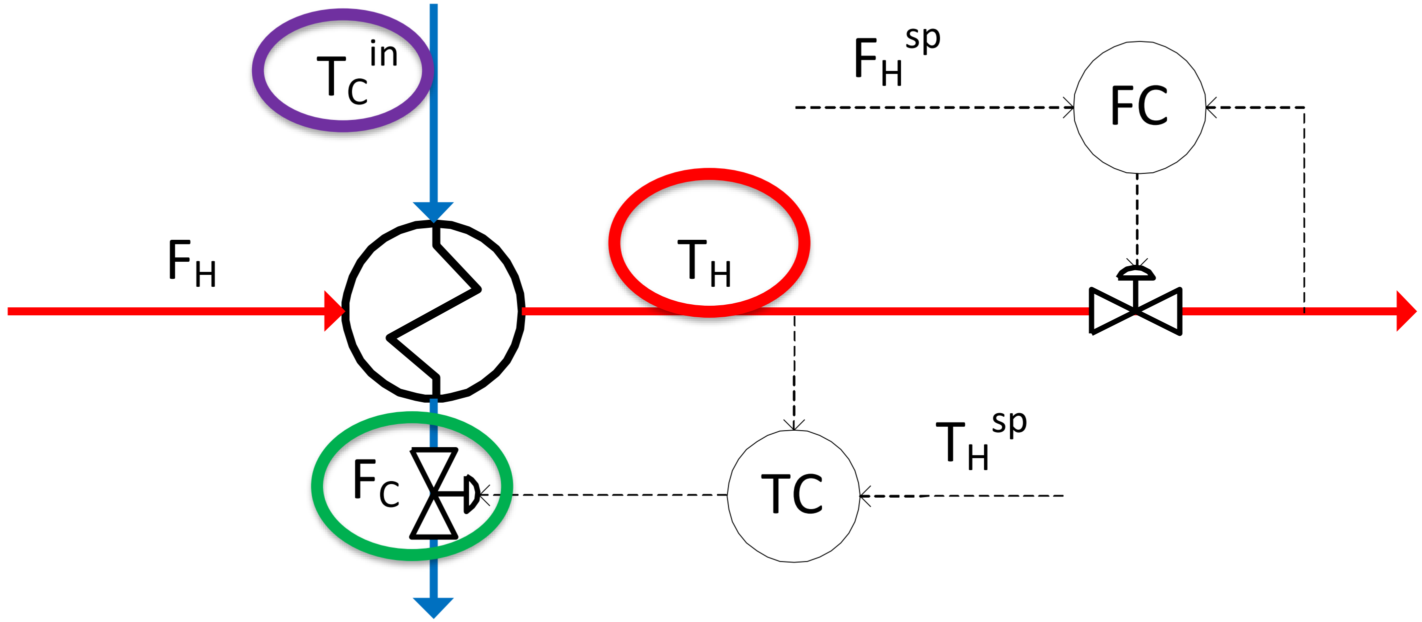
1. **$y_1 = y_1^{sp}$, $y_2 = y_2^{sp}$ ← Nomimal = unconstrained**
2. **$y_1 = y_1^{sp}$, $u_2 = u_2^{max}$**
3. **$y_1 = y_1^{sp}$, $u_1 = u_1^{max}$**

Input saturation pairing rule: It's **not possible** to follow this rule since both MVs may saturate…

- Will pair y_1 with u_1 for dynamic reasons («pair close rule»)
- And use «complex» CV-MV switching logic when u_1 saturates

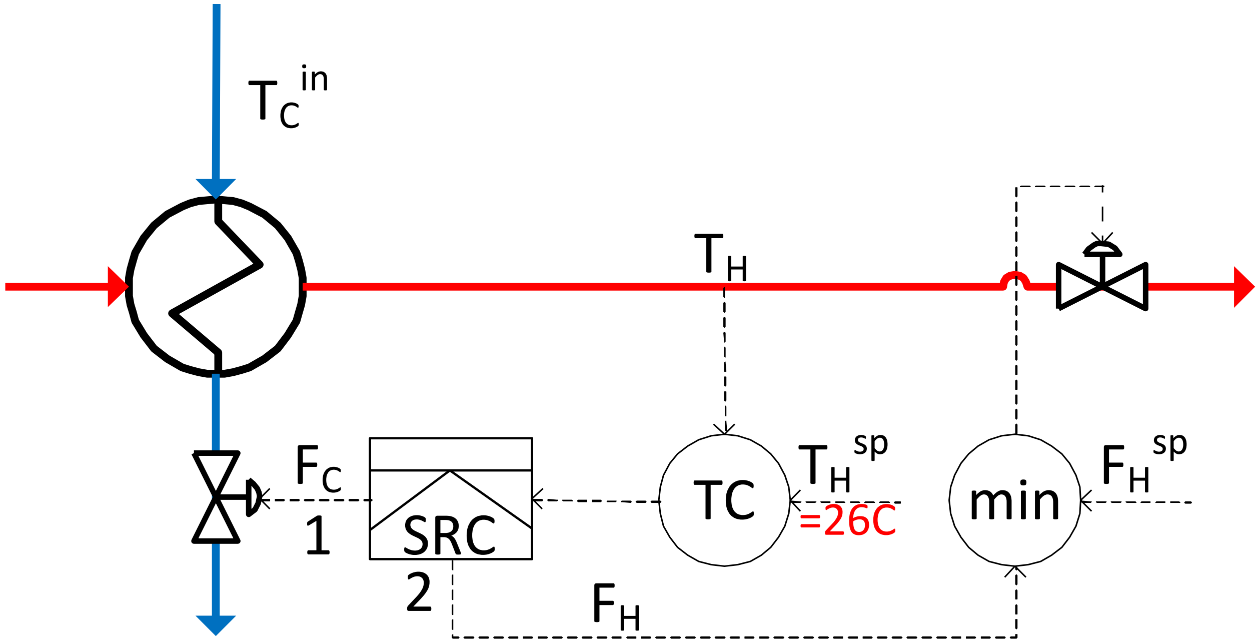
NTNU

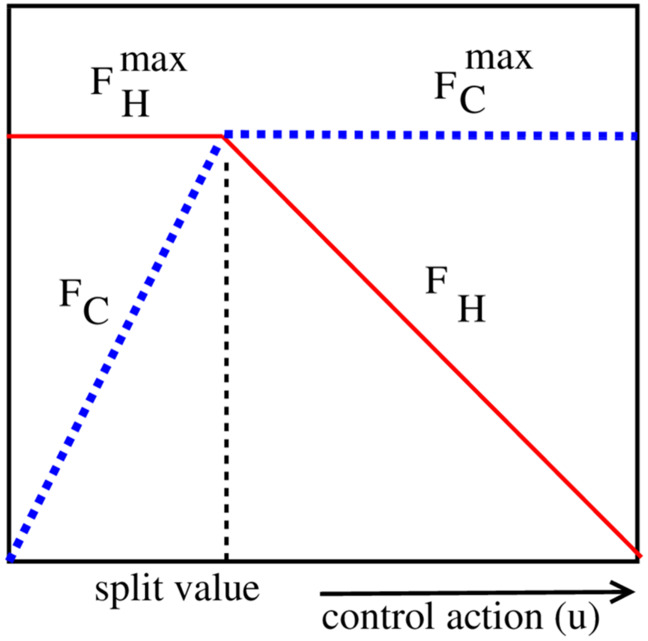
Pairings at nominal «unconstrained» operating point



Use F_C to control T_H ⇒ F_C may saturate for a large disturbance (T_C^{in})

Alt.1: Split range control with min-selector





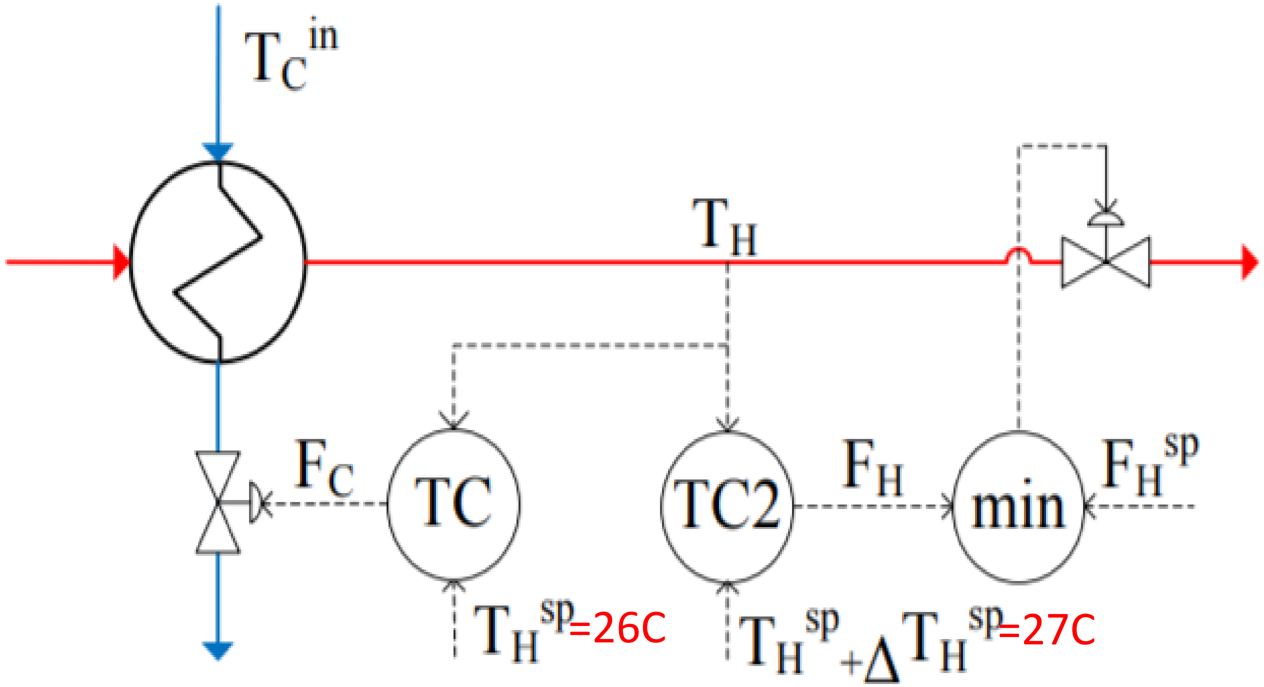
Tuning of TC using SIMC rule:

τ_c = 2θ = 88 s

Kc = -0.55

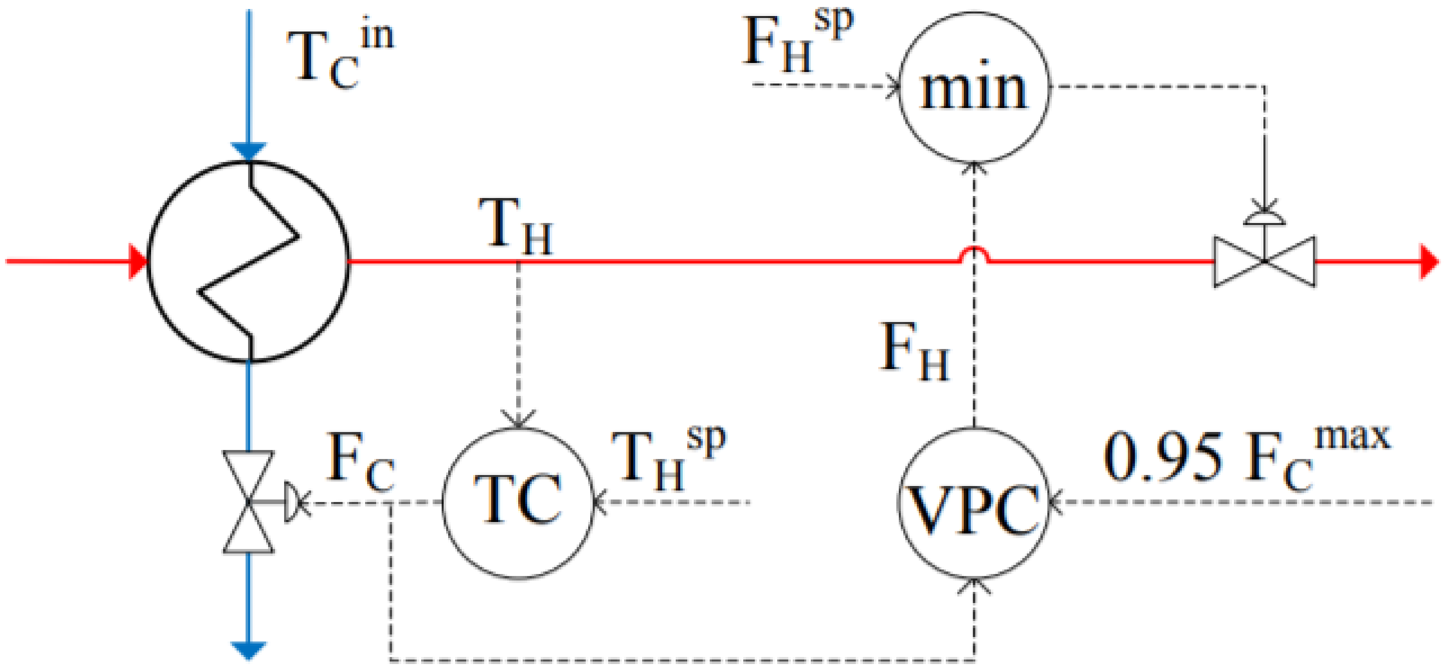
τ_I = 74 s

Alt.2 . Two controllers/setpoints and min-selector



(c) Two controllers with different setpoints.

Alt. 3 VPC with min-selector



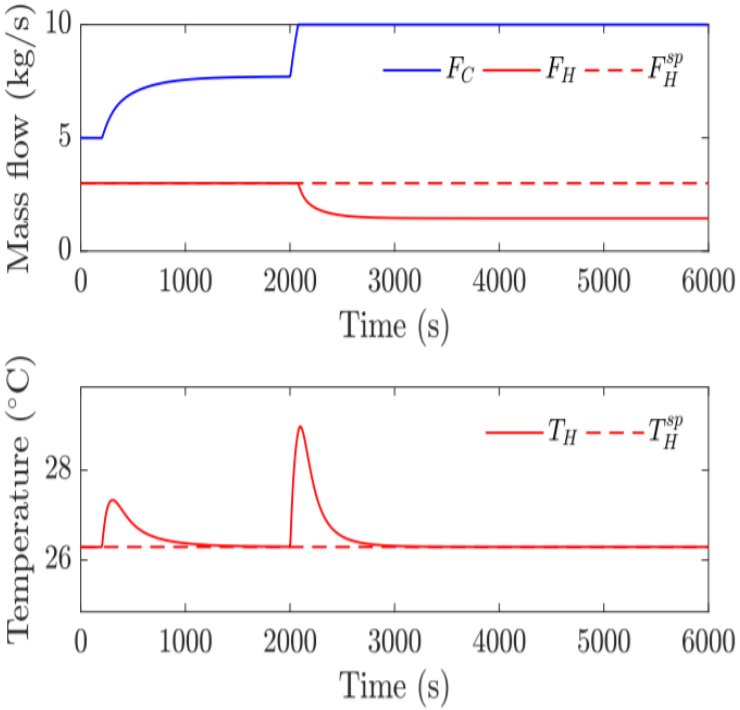
(b) Valve position control (VPC).

Complex CV-MV switching

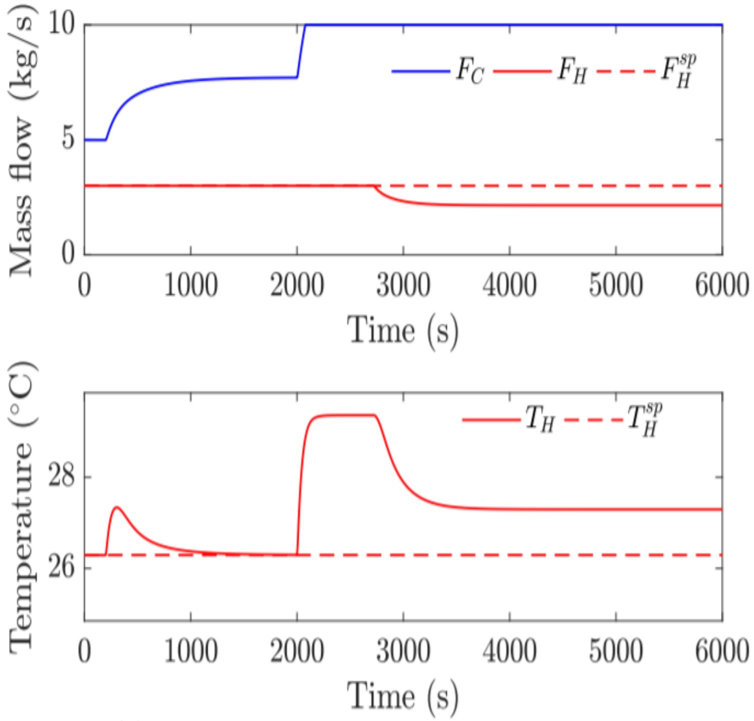
Alt. 1 Split range control

Alt. 2 Two controllers/setpoints

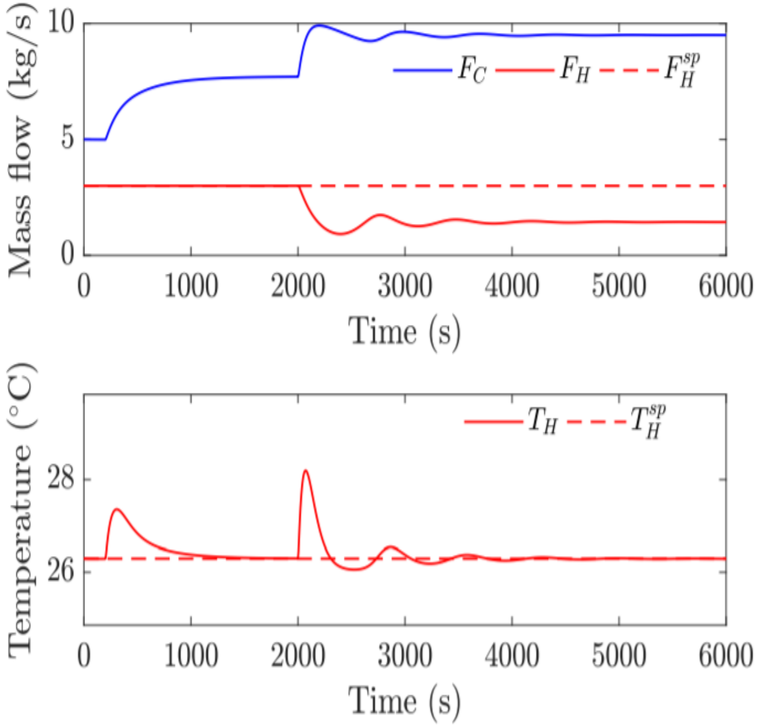
Alt. 3 Valve position control



(a) Split Range Control (SRC).



(c) Two controllers with different setpoints.



(b) Valve position control (VPC).

Disturbances: Tcin +2◦C at t = 200 s, Tcin additional +4 ◦C at t = 2000 s.

MPC for cooler

Tuning ← trial and error

$$\min \sum_{k=1}^{N} \left(\omega_1 \| (T_{H_k} - T_H^{sp}) \|^2 + \omega_2 \| (F_{H_k}^{max} - F_{H_k}) \|^2 \right)$$

← Objective function (CV constraints)

s.t.

Model →

MV constraints →

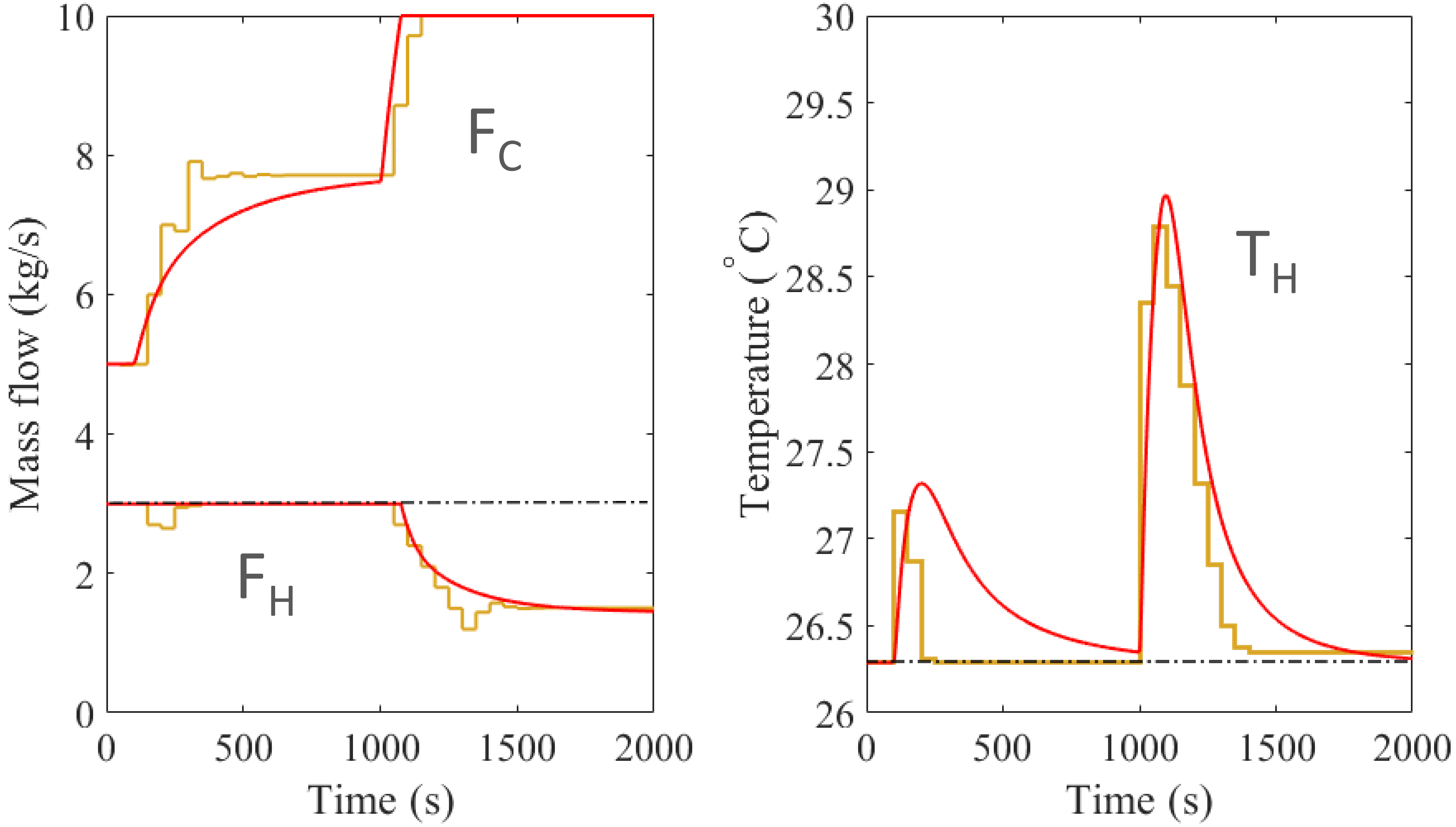
$$\left.\begin{aligned} &T_{k,i} = f(T_{H_k,i}, T_{H_k,i-1}, T_{C_k,i}, T_{C_k,i+1}, F_{H_k}, F_{C_k}) \\ &0 \le F_{H_k} \le F_H^{max} \\ &0 \le F_{C_k} \le F_C^{max} \end{aligned}\right\} \quad \forall k \in \{1, \ldots, N\}$$

$$\left.\begin{aligned} &0 \le \Delta F_{H_k} \le 0.1 F_H^{max} \\ &0 \le \Delta F_{C_k} \le 0.1 F_C^{max} \end{aligned}\right\} \quad \forall k \in \{1, \ldots, N-1\}$$

$$\Delta F_k = F_k - F_{k-1}, \forall k \in \{1, \ldots, N-1\}.$$

For $k = 1$, F_{k-1} represents the flow at the nominal point.

MPC vs Split range Control (PI)



Disturbance
(T_C^{in})

t = 10 s; + 2°C

t = 1000 s; + 4°C

Red: Split Range Control (PI)

Yellow: MPC:

Δt = 50 s

$\omega_1 = 3$

$\omega_2 = 0.1$

Many people think they need to use MPC if they encounter constraints

- True only for more complicated multivariable cases
- In most cases PI(D)-control is simpler and equallly good
  - Need anti-windup on the controller

=26C

6. Conclusion

- Put optimization into the control layer
  - It’s much faster and more effective
- Conventional APC works very well in many cases
  - Optimization by feedback
    - Self-optimizing control
    - Active constraint switching
  - Need to pair input and output.
    - Advantage: The engineer can specify directly the solution
    - Problem: May not be possible for complex cases
  - Need model only for parts of the process (for tuning)
  - Challenge: Need better teaching and design methods

HighEFF

sfi Centre for Research-based Innovation
The Research Council of Norway

SUBPRO

NTNU