P8

Tranformed inputs

Briefly on pro and cons of MPC

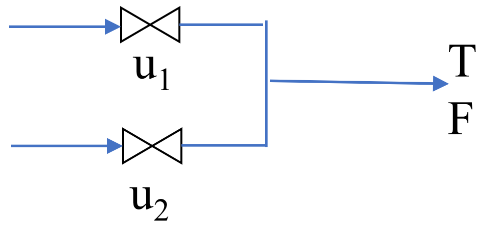
RTO

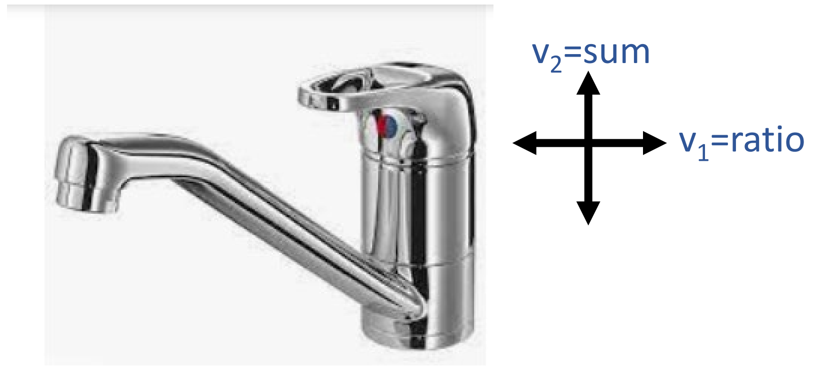
ESC

Nonlinear feedforward, decoupling and linearization

- Transformed inputs: Extremely simple and effective way of achieving feedforward, decoupling and linearization

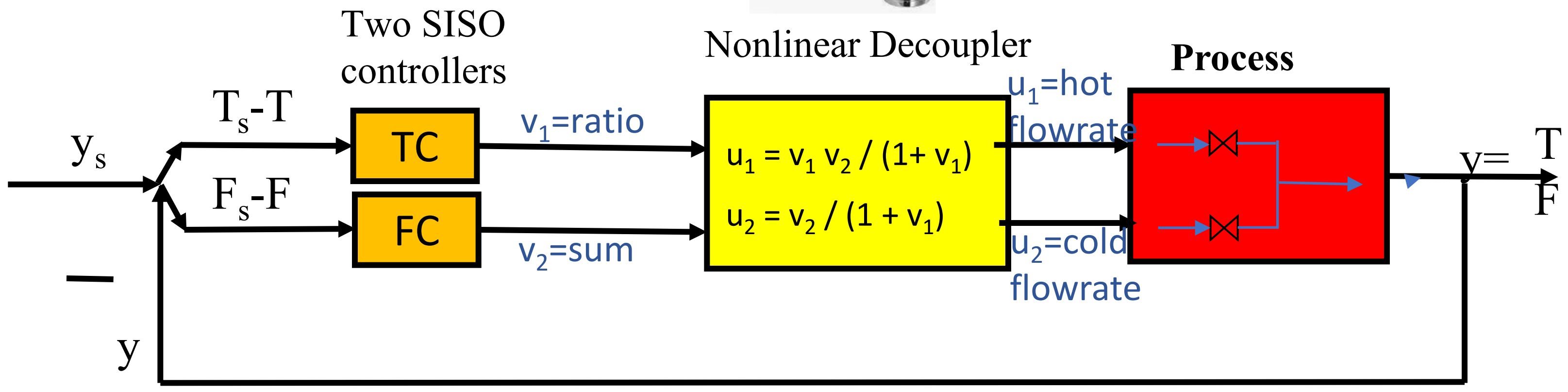
Example decoupling: Mixing of hot (u_1) and cold (u_2) water





- Want to control
  - y_1 = Temperature T
  - y_2 = total flow F
- Inputs, u=flowrates
- May use two SISO PI-controllers
  - TC
  - FC
- Insight: Get decoupled response with transformed inputs
  - TC sets flow ratio, $v_1 = u_1/u_2$
  - FC sets flow sum, $v_2 = u_1 + u_2$
- Decoupler: Need «static calculation block» to solve for inputs

$$u_1 = v_1 v_2 / (1+ v_1)$$
$$u_2 = v_2 / (1 + v_1)$$



Pairings:

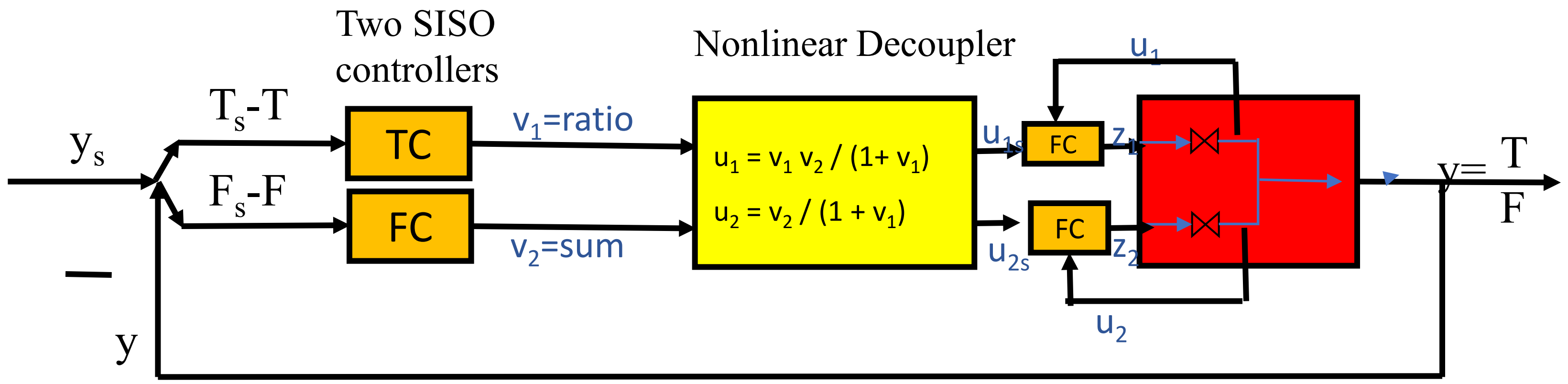
- T – v_1
- F – v_2

No interactions for setpoint change

Note:

- In practice u=valve position (z)
- So must add two flow controllers
    - These generate inverse by feedback

In practice must add two slave flow controllers



v = transformed inputs
u = flowrates
z = valve positions

Decoupler with feedforward:

$$q_h = \frac{v_2(v_1 - T_c)}{T_h - T_c}$$
$$q_c = v_2 - q_h$$

Feedforward (and decoupling) control

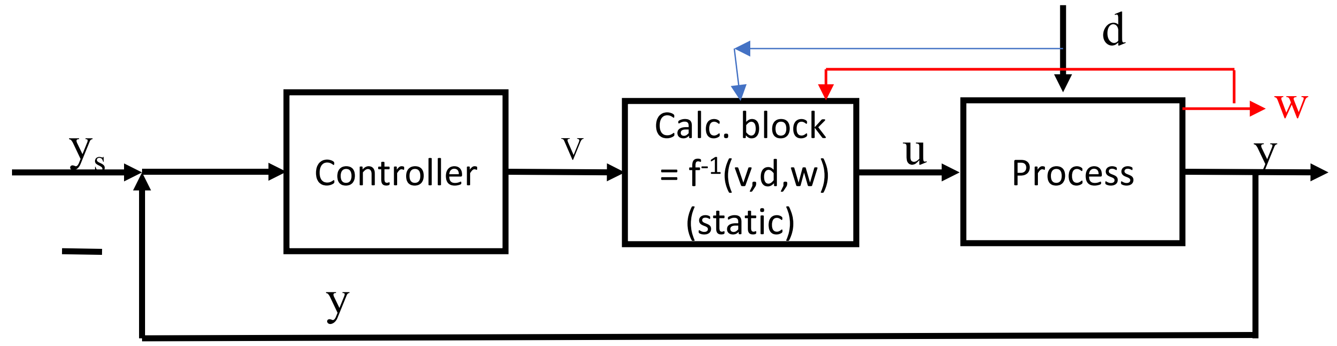
- Feedforward control relies on model
- as opposed to feedback which relies mostly on data

- Feedback control: Linear model is often OK
- Feedforward control: Much less likely that linear model is OK because of process changes and disturbances
- Here: Nonlinear feedforward control using Input transformations based on static process model

Input transformations

General approach: Combined Nonlinear decoupling, feedforward and linearization using Transformed Inputs *

- Generalization: Introduce transformed input v and use Nonlinear calculation block



Genaral Method*:

Steady-state model: $y = f(u,d,w)$

Select transformed input: $v = f(u,d,w)$ («right-hand side» of model)

Calculation block: Invert for given v: $u = f^{-1}(v,d,w)$ (may be replaced by slave v-controller)

w=dependent variable (flow, temperature), but treated as measured disturbance

w-variables may be used to simplify model

Transformed system becomes: $y=I\,v$ («decoupled, linear, indepedent of d»)

Note: To simplify often use only «parts» of f(u,d,w) as v (because of unknown parameters etc.)

*Zotica, Alsop and Skogestad. 2020 IFAC World Congress

Example: Combined nonlinear decoupling and feedforward. Mixing of hot and cold water

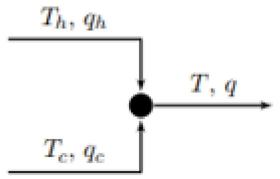
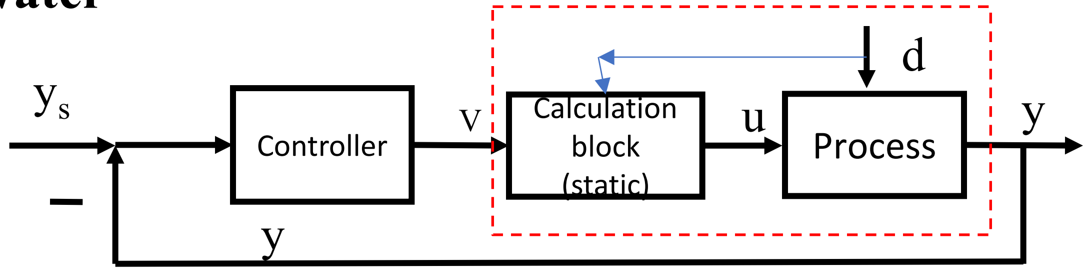


Figure 1: Mixer system



$$u = \begin{pmatrix} q_h \\ q_c \end{pmatrix}$$

$$d = \begin{pmatrix} T_h \\ T_c \end{pmatrix}$$

$$y = \begin{pmatrix} T \\ q \end{pmatrix}$$

Steady-state model written as y=f(u,d):

$$T = \frac{qhTh+qcTc}{qh+qc}$$

$$q = q_c + q_h$$

Select transformed inputs as right hand side, v = f

$$v_1 = \frac{qhTh+qcTc}{qh+qc} \quad (1)$$ Generalized ratio

$$v_2 = q_c + q_h \quad (2)$$

Model from v to y (red box) is then decoupled and with perfect disturbance rejection:

$$T = v_1$$

$$q = v_2$$

- Can then use two single-loop PI controllers for T and q!
  - These controllers are needed to correct for model errors and unmeasured disturbances
- Note that v_1 used to control T is a generalized ratio, but it includes also feedforward from Tc and Th.

Implementation (calculation block) : Solve (1) and (2) with respect to u=(qc qh):

Decoupler with feedforward:

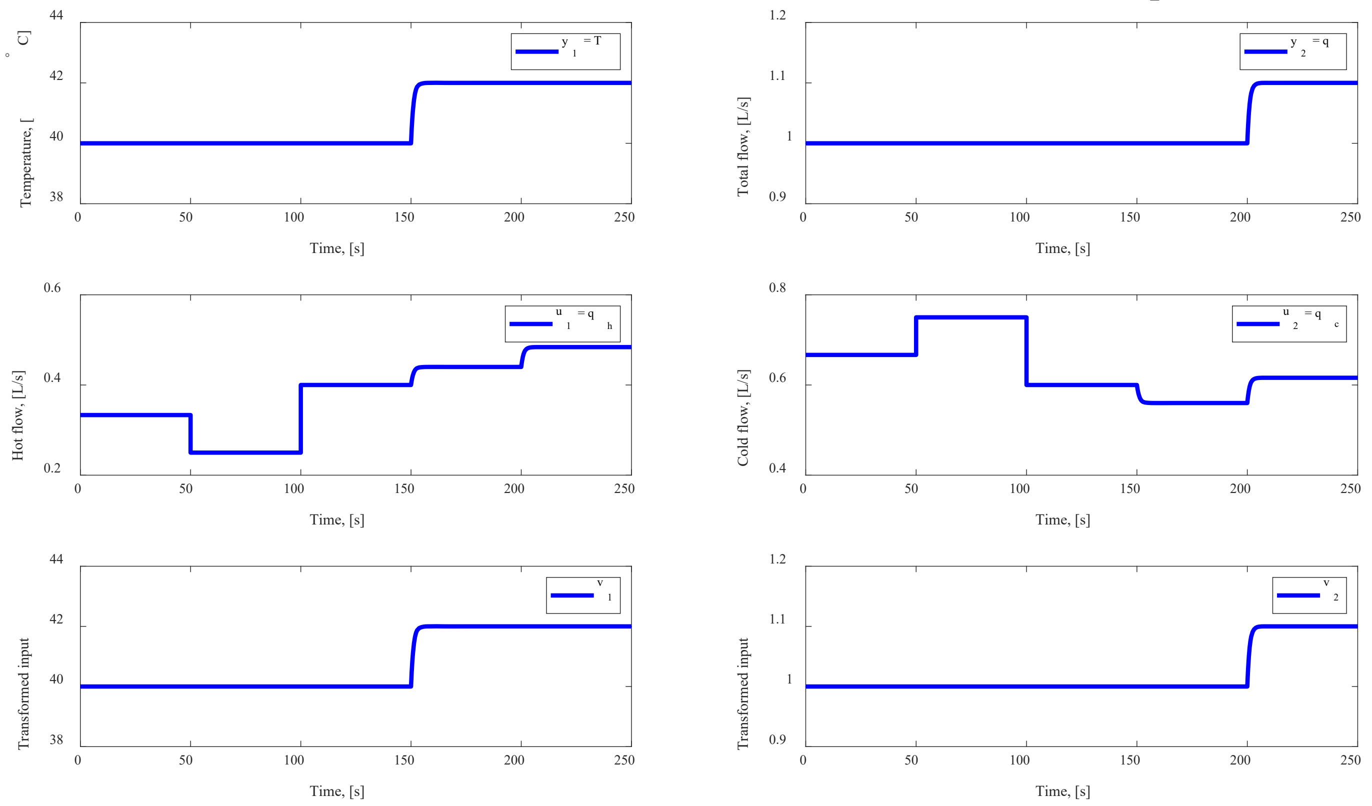
$$q_h = \frac{v_2(v_1 - T_c)}{T_h - T_c}$$

$$q_c = v_2 - q_h$$

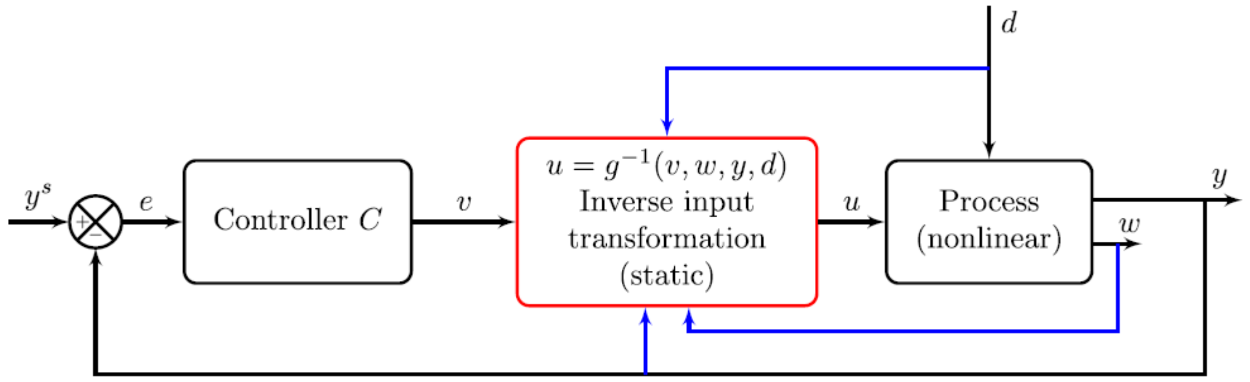
Transformed MVs for decupling, linearization and disturbance rejection
Mixing of hot and cold water (static process)
New system: $T=v_1$ and $q=v_2$

Outer loop: Two I-controllers with $\tau_C = 1$ s

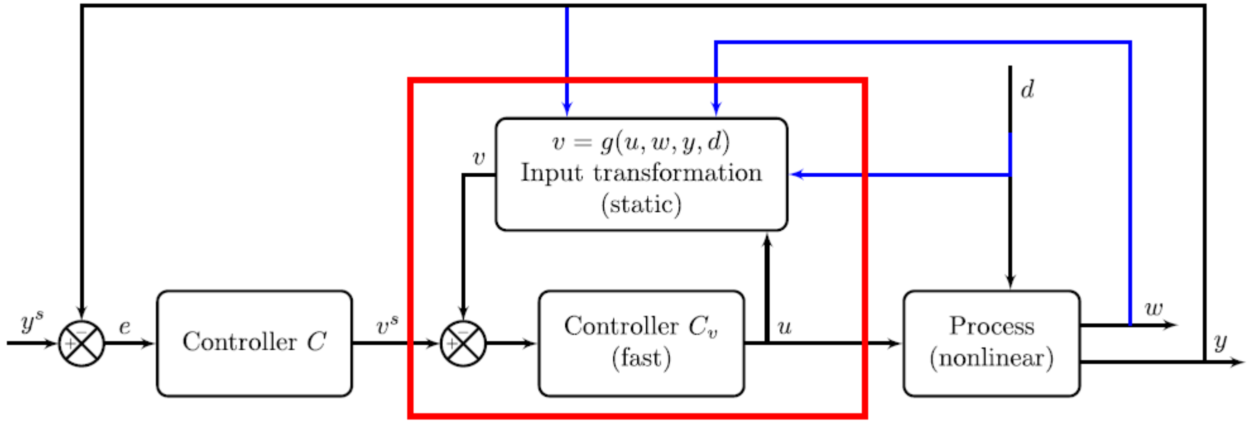
1. T_h: 60->70 °C at t = 50 s
2. T_c: 30->20 °C at t = 100 s
3. T_h^s: 40->42 °C at t = 150 s
4. q^s: 1->1.1 L/s at t = 200 s



Alternative B: Calculation block solved by feedback (using fast slave controller C_v)

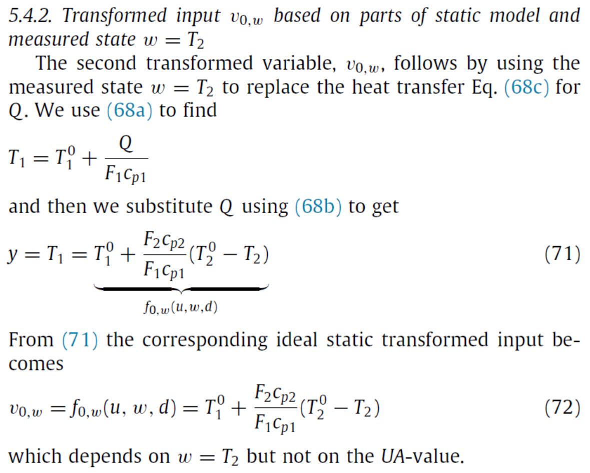


(a) Alternative A. Model-based implementation of transformed input $v = g(u, w, y, d)$. The physical input $u = g^{-1}(v, w, y, d)$ is generated by a static (algebraic) calculation block which inverts the transformed input model equations. The model-based implementation generates the exact inverse for the case with no model error.



(b) Alternative B. Feedback implementation of transformed input $v = g(u, w, y, d)$ using cascade control with a slave v-controller. The computed value of v is driven to its setpoint v_s by the inner (slave) feedback controller C_v which generates the physical input u. This implementation generates an approximate inverse.

Example: Power control



5.4.2. Transformed input $v_{0,w}$ based on parts of static model and measured state $w = T_2$

The second transformed variable, $v_{0,w}$, follows by using the measured state $w = T_2$ to replace the heat transfer Eq. (68c) for Q. We use (68a) to find

$$T_1 = T_1^0 + \frac{Q}{F_1 c_{p1}}$$

and then we substitute Q using (68b) to get

$$y = T_1 = \underbrace{T_1^0 + \frac{F_2 c_{p2}}{F_1 c_{p1}}(T_2^0 - T_2)}_{f_{0,w}(u,w,d)} \qquad (71)$$

From (71) the corresponding ideal static transformed input becomes

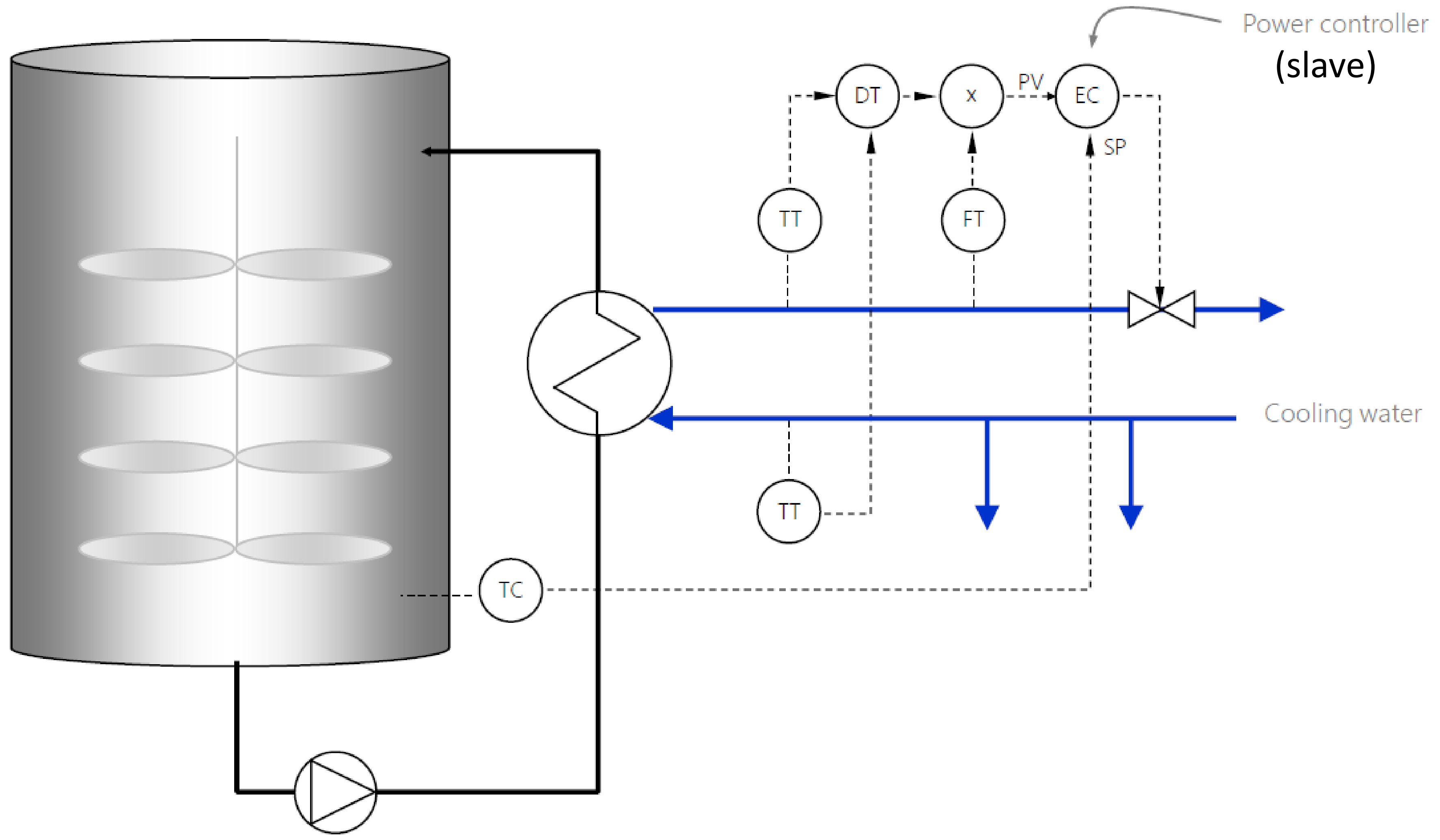
$$v_{0,w} = f_{0,w}(u, w, d) = T_1^0 + \frac{F_2 c_{p2}}{F_1 c_{p1}}(T_2^0 - T_2) \qquad (72)$$

which depends on $w = T_2$ but not on the *UA*-value.

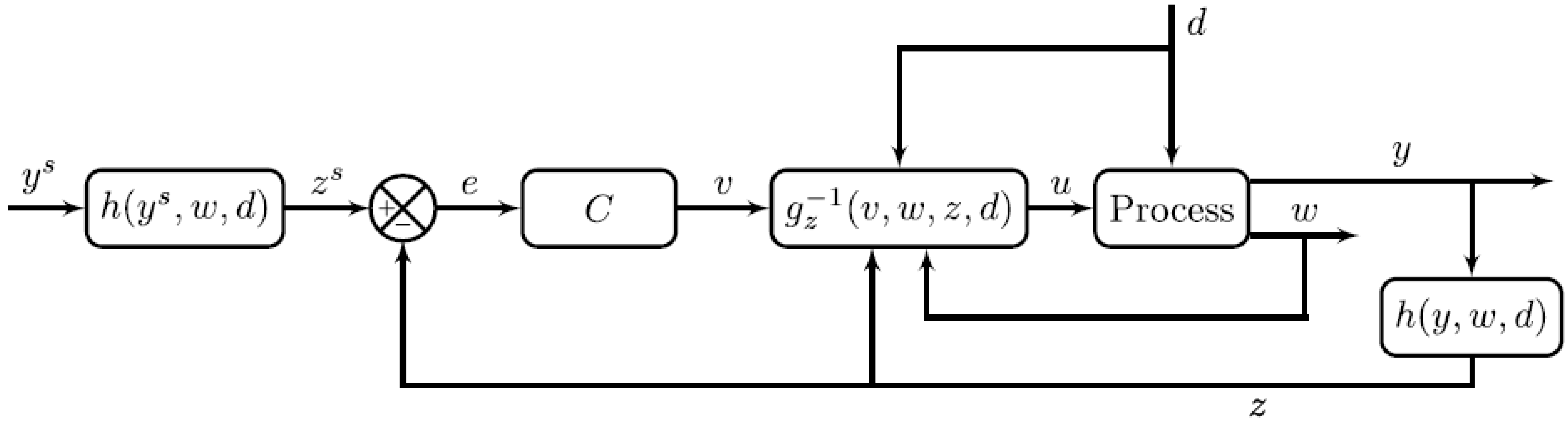
In practice (Perstorp) use only part of this:

$v = F_2(T_2^0 - T_2)$

New control structure: Power control



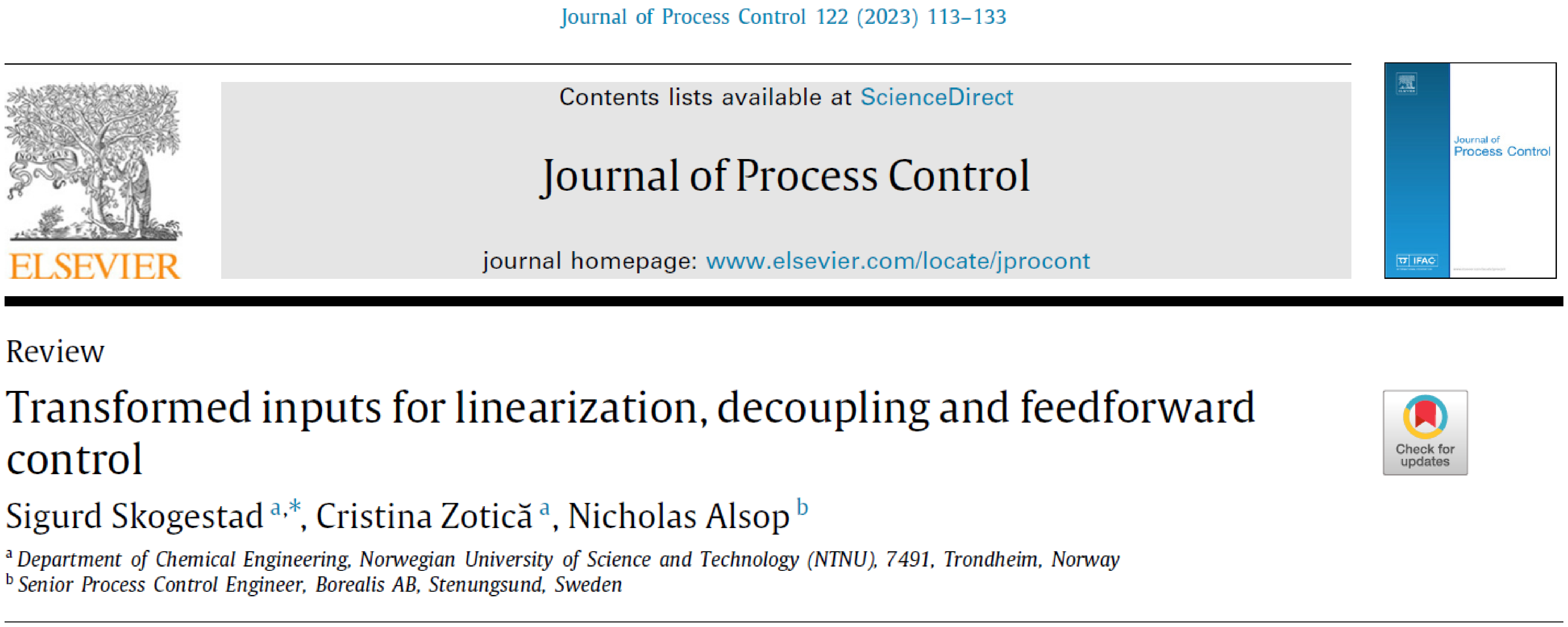
Also: Transformed outputs z



(a) General implementation of transformed output z

- No fundamental advantage, but can simplify input transformation
- For example, y=T, z=H (enthalpy)

More on transformed inputs



Journal of Process Control 122 (2023) 113–133

Contents lists available at ScienceDirect

Journal of Process Control

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

Review

Transformed inputs for linearization, decoupling and feedforward control

Sigurd Skogestad[a,*], Cristina Zotică[a], Nicholas Alsop[b]

[a] *Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), 7491, Trondheim, Norway*
[b] *Senior Process Control Engineer, Borealis AB, Stenungsund, Sweden*

MPC and RTO

What about MPC?

- First industrial use in the 1970s
- Became common in the refining and petrochemical industry in the 1980s
- In the 1990s a bright future was predicted for MPC in all process industries (chemical, thermal power, …)
- 30 years later: We know that this did not happen
- Why? First, the performance benefits of MPC compared to ARC are often minor (if any)
- In addition, MPC has some limitations
    1. Expensive to obtain model
    2. Does not easily handle integral action, cascade and ratio control
    3. Normally, cannot be used at startup (so need ARC anyway)
    4. Can be difficult to tune. Difficult to incorporate fast control tasks (because of centralized approach)
    5. Computations can be slow
    6. Robustness (e.g., gain margin) handled indirectly
- Advantages of MPC
    1. Very good for interactive multivariable dynamic processes
    2. Coordinates feedforward and feedback
    3. Coordinates use of many inputs
    4. Makes use of information about future disturbances, setpoints and prices (predictive capabilities of MPC)
    5. Can handle nonlinear dynamic processes (nonlinear MPC)
- What about constraints
    - Not really a major advantage with MPC; can be handled well also with ARC

MPC = model predictive control
ARC = advanced regulatory control

7.6.7. Summary of MPC shortcomings

Some shortcomings of MPC are listed below, in the expected order of importance as seen from the user's point of view:

1. MPC requires a "full" dynamic model involving all variables to be used by the controller. Obtaining and maintaining such a model is costly.
2. MPC can handle only indirectly and with significant effort from the control engineer (designer), the three main inventions of process control; namely integral control, ratio control and cascade control (see above).
3. Since a dynamic model is usually not available at the startup of a new process plant, we need initially a simpler control system, typically based on advanced regulatory control elements. MPC will then only be considered if the performance of this initial control system is not satisfactory.
4. It is often difficult to tune MPC (e.g., by choosing weights or sometimes adjusting the model) to give the engineer the desired response. In particular, since the control of all variables is optimized simultaneously, it may be difficult to obtain a solution that combines fast and slow control in the desired way. For example, it may be difficult to tune MPC to have fast feedforward control for disturbances because it may affect negatively the robustness of the feedback part (Pawlowski et al., 2012).
5. The solution of the online optimization problem is complex and time-consuming for large problems.
6. Robustness to model uncertainty is handled in an ad hoc manner, for example, through the use of the input weight R. On the other hand, with the SIMC PID rules, there is a direct relationship between the tuning parameter τ_c and robustness margins, such as the gain, phase and delay margin Grimholt and Skogestad (2012), e.g., see (C.13) for the gain margin.

7.6.8. Summary of MPC advantages

The above limitations of MPC, for example, with respect to integral action, cascade control and ratio control, do not imply that MPC will not be an effective solution in many cases. On the contrary, MPC should definitely be in the toolbox of the control engineer. First, standard ratio and cascade control elements can be put into the fast regulatory layer and the setpoints to these elements become the MVs for MPC. More importantly, MPC is usually better (both in terms of performance and simplicity) than advanced regulatory control (ARC) for:

1. Multivariable processes with (strong) dynamic interactions.
2. Pure feedforward control and coordination of feedforward and feedback control.
3. Cases where we want to dynamically coordinate the use of many inputs (MVs) to control one CV.
4. Cases where future information is available, for example, about future disturbances, setpoint changes, constraints or prices.
5. Nonlinear dynamic processes (nonlinear MPC).

The handling of constraints is often claimed to be a special advantage of MPC, but it can it most cases also be handled well by ARC (using selectors, split-range control solutions, anti-windup, etc.). Actually, for the Tennessee Eastman Challenge Process, Ricker (1996) found that ARC (using decentralized PID control) was better than MPC. Ricker (1996) writes in the abstract: "There appears to be little, if any, advantage to the use of NMPC (nonlinear MPC) in this application. In particular, the decentralized strategy does a better job of handling constraints – an area in which NMPC is reputed to excel". In the discussion section he adds: "The reason is that the TE problem has too many competing goals and special cases to be dealt with in a conventional MPC formulation".

Optimal operation and constraints switching

- We have presented effective decentralized approaches for constraint switching (MV-MV, CV-CV, MV-CV).
  - Optimal in many cases, but not in general
  - For example, may not be able to cover cases with more than one unconstrained region $\Rightarrow$ More than one self-optimizing variable

- An alternative is model-based RTO, usually based on static model

RTO = Real-time optimization

Economic real-time optimization(RTO)

Alternative RTO approaches:

Model-based

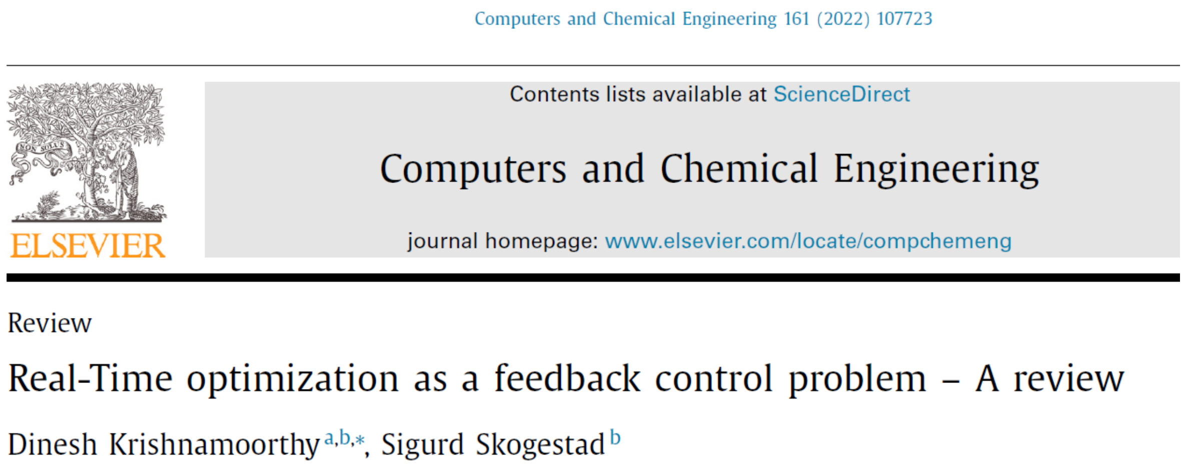
I. Separate RTO layer (online dynamic or steady-state optimization)

II. Feedback-optimizing control (put optimization into control layer)

- Alt.1. (Most general): Based on dual decomposition (iterate on Lagrange multipliers λ)
- Alt.2 (Tighter constraint control): Region-based with reduced gradient

Data-based

III. Hill-climbing methods = Extremum-seeking control (model free. But need to measure cost J)



Computers and Chemical Engineering 161 (2022) 107723

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

ELSEVIER

Review

Real-Time optimization as a feedback control problem – A review

Dinesh Krishnamoorthy[a,b,*], Sigurd Skogestad[b]

I. Conventional (commercial) steady-state RTO

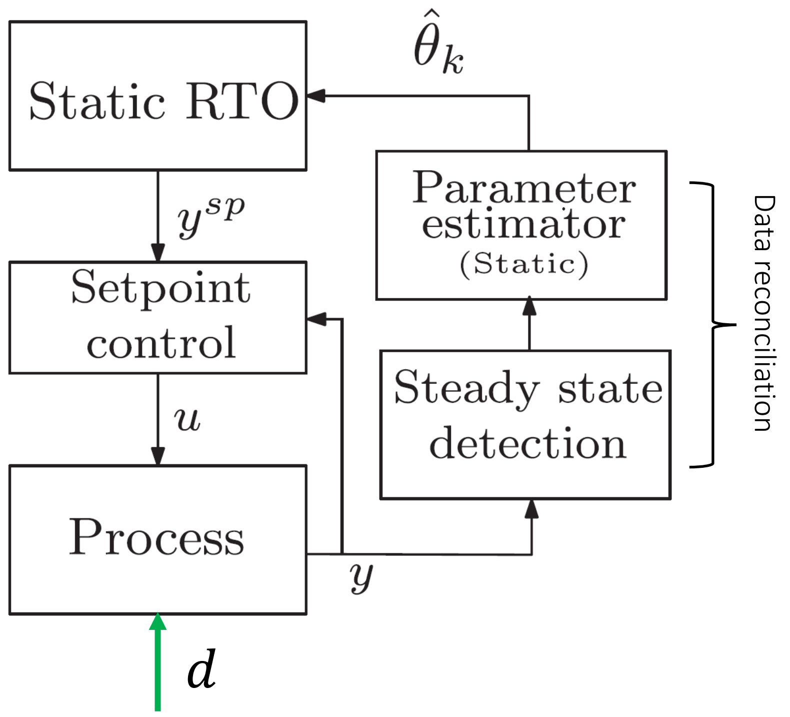
Fairly common in refining and petrochemical industy.

Two-step approach:

Step 1. “Data reconciliation”:

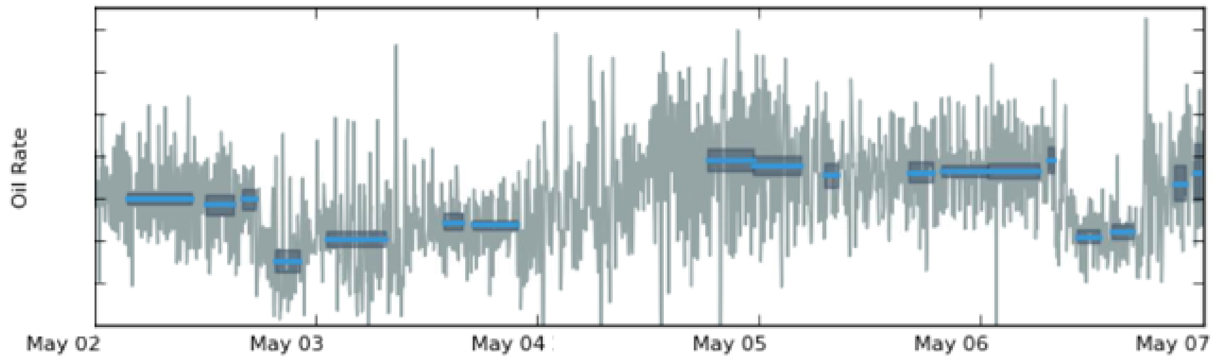
- Steady-state detection
- Update estimate of d: model parameters, disturbances (feed), constraints

Step 2. Re-optimize to find new optimal steady state



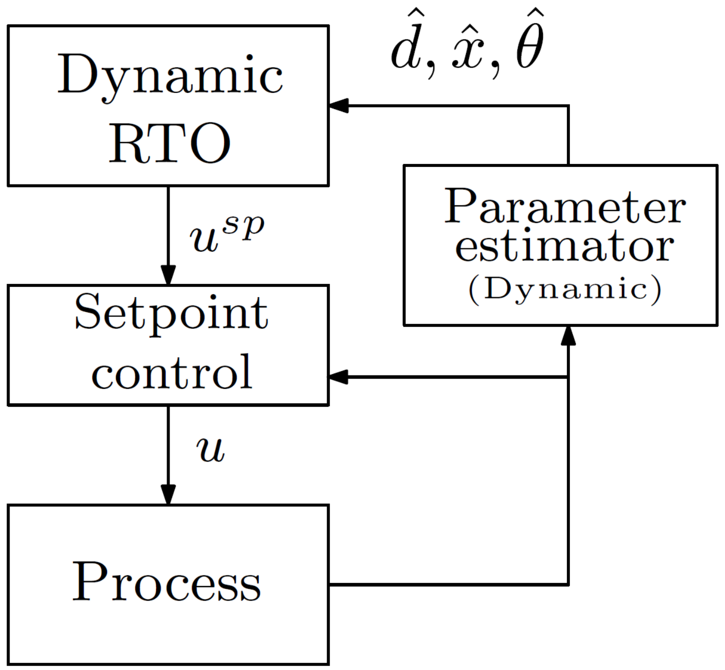
Steady-state wait time

- Transient measurements cannot be used → system must “settle”
- Large chunks of data discarded
- Steady state detection issues
  - Erroneously accept transient data
  - Non-stationary drifts



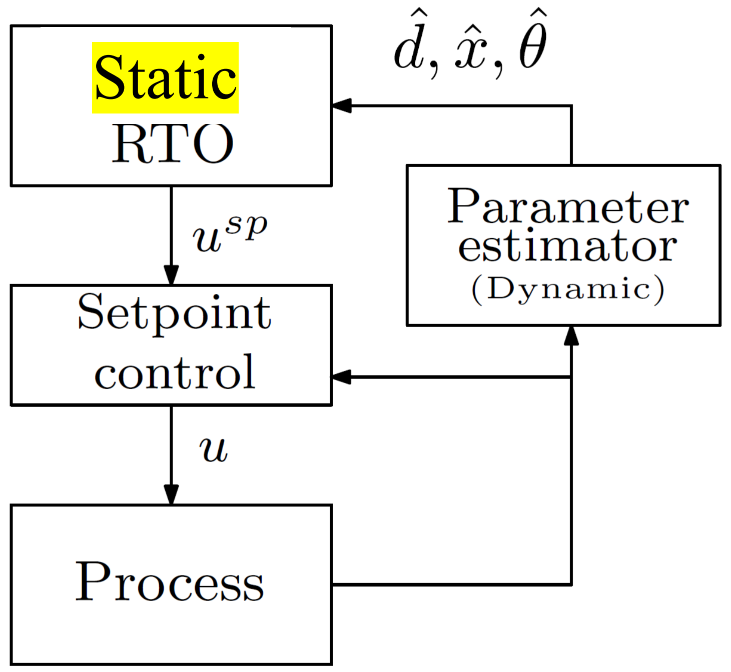
How to avoid steady state wait time?

1. Dynamic RTO = EMPC


Dynamic RTO
$\hat{d}, \hat{x}, \hat{\theta}$
u^{sp}
Parameter estimator (Dynamic)
Setpoint control
u
Process

How to avoid steady state wait time?

2. Hybrid RTO


Static
RTO
$\hat{d}, \hat{x}, \hat{\theta}$
u^{sp}
Parameter estimator (Dynamic)
Setpoint control
u
Process

RTO problem

Steady-state RTO (used in Hybrid RTO):

$$\min_{x,u} J(x,d,u)$$

s.t.:

$$\begin{gathered} 0 = F(x,d,u) \\ 0 = h(x,d,u) \\ g(x,d,u) \leq 0 \end{gathered}$$

Dynamic RTO ≡ (Economic) nonlinear MPC **:**

$$\min_{x(t),u(t)} \int_{t_0}^{t_f} J\big(x(t),d(t),u(t)\big)\,dt$$

s.t.:

$$\begin{gathered} \dot{x}(t) = F\big(x(t),d(t),u(t)\big) \\ 0 = h\big(x(t),d(t),u(t)\big) \\ g\big(x(t),d(t),u(t)\big) \leq 0 \\ x(t_0) = \hat{x}_0 \end{gathered}$$

Now we calculate not only an optimal point, but an <u>optimal trajectory</u>!'

BUT Much more complex that static RTO, and may not give much economic benefit

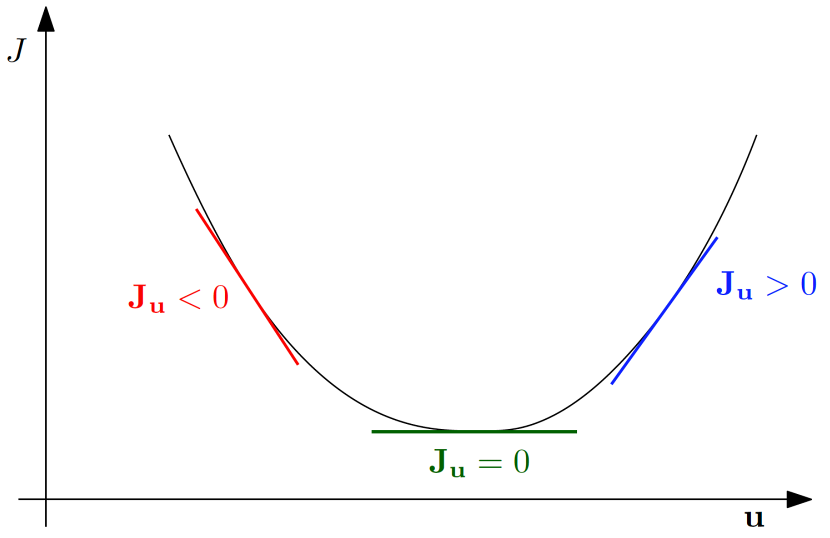
II. Feedback RTO (unconstrained case)

«Solving RTO-problem using PI control»

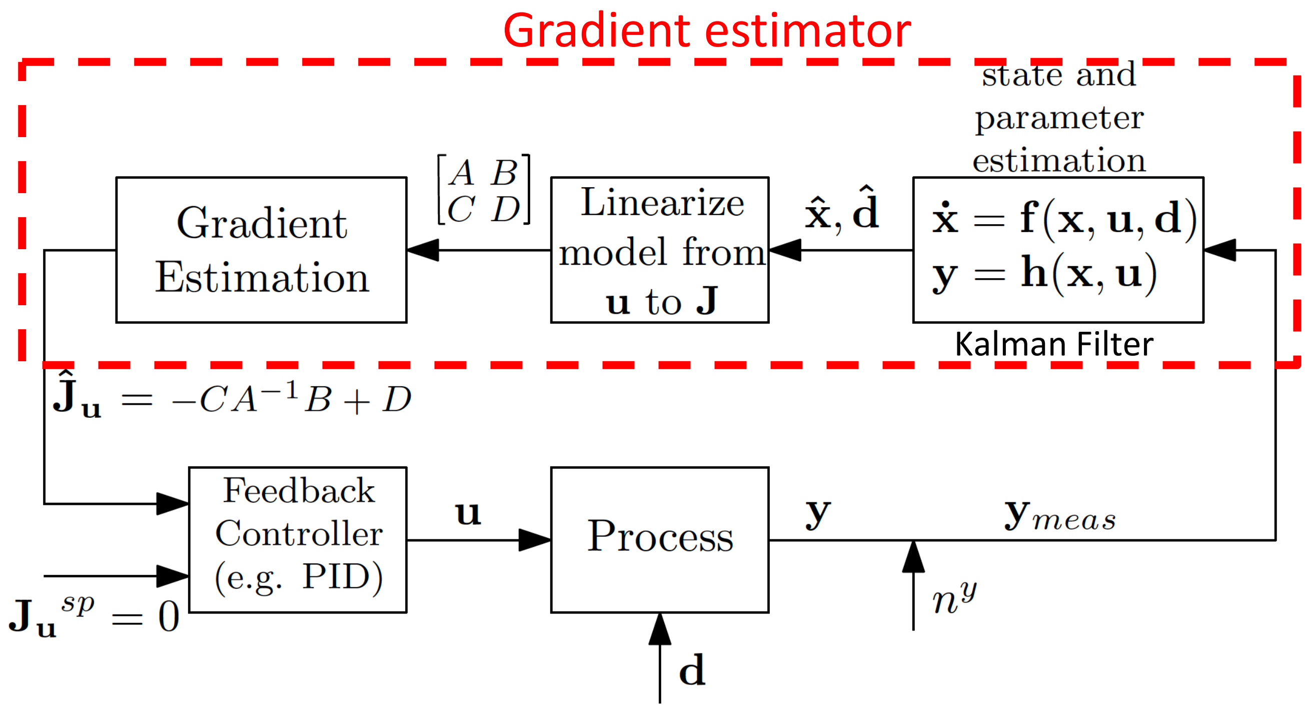
Unconstrained optimization.

Necessary condition of optimality (NCO):

- Gradient of cost function = 0
- $J_u \equiv \frac{\partial J}{\partial u} \equiv \nabla_u J = 0$



IIA. Feedback RTO (unconstrained case)


Gradient estimator
state and parameter estimation
Gradient Estimation
$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$
Linearize model from u to J
$\hat{\mathbf{x}}, \hat{\mathbf{d}}$
$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d})$
$\mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{u})$
Kalman Filter
$\hat{\mathbf{J}}_{\mathbf{u}} = -CA^{-1}B + D$
Feedback Controller (e.g. PID)
u
Process
y
$\mathbf{y}_{meas}$
n^y
$\mathbf{J_u}^{sp} = 0$
d

Linearize the dynamic model

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d}) \\ J &= \mathbf{g}(\mathbf{x}, \mathbf{u}) \end{aligned} \Rightarrow \begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} \\ J &= C\mathbf{x} + D\mathbf{u} \end{aligned}$$

$$A = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad B = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}$$

$$C = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}} \quad D = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{u}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}$$

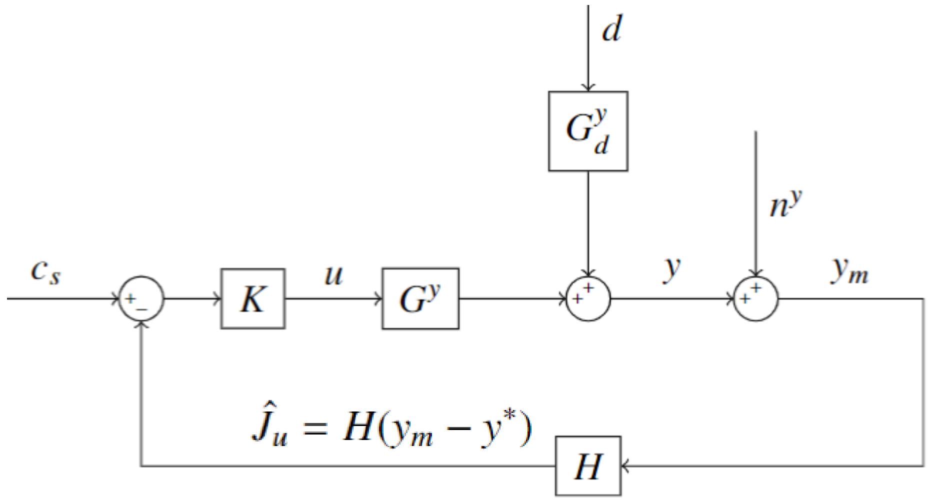
Trick, set $\dot{x} = 0, to\ get\ estimate\ of\ static\ gradient:$

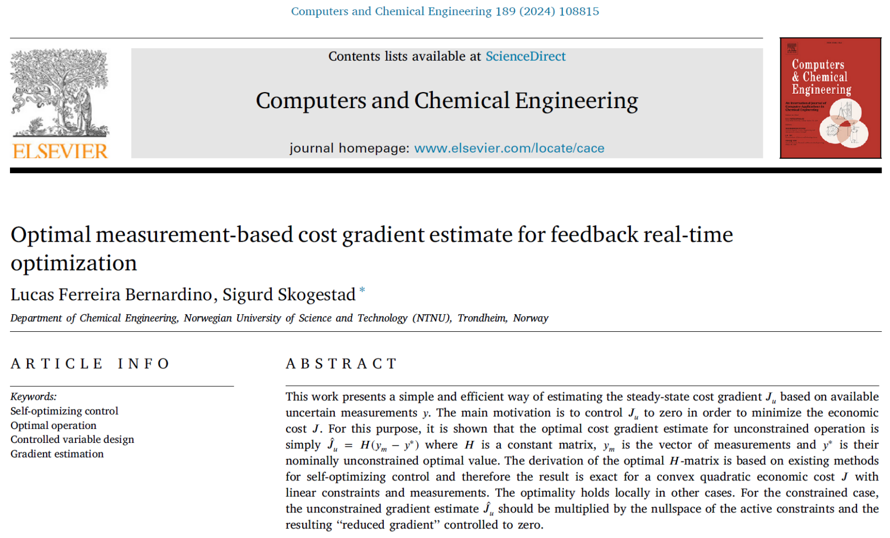
$$J = \underbrace{\left(-CA^{-1}B + D\right)}_{\hat{\mathbf{J}}_{\mathbf{u}}} \mathbf{u}$$

Note: This is one simple way of doing the gradient estimation, but needfs dynamic model (Kalman Filter)

Here is another Static gradient estimation:

Based on self-optimizing control. Very simple and works well!





Computers and Chemical Engineering 189 (2024) 108815

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

Optimal measurement-based cost gradient estimate for feedback real-time optimization

Lucas Ferreira Bernardino, Sigurd Skogestad *

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

ARTICLE INFO

Keywords:
Self-optimizing control
Optimal operation
Controlled variable design
Gradient estimation

ABSTRACT

This work presents a simple and efficient way of estimating the steady-state cost gradient J_u based on available uncertain measurements y. The main motivation is to control J_u to zero in order to minimize the economic cost J. For this purpose, it is shown that the optimal cost gradient estimate for unconstrained operation is simply $\hat{J}_u = H(y_m - y^*)$ where H is a constant matrix, y_m is the vector of measurements and y^* is their nominally unconstrained optimal value. The derivation of the optimal H-matrix is based on existing methods for self-optimizing control and therefore the result is exact for a convex quadratic economic cost J with linear constraints and measurements. The optimality holds locally in other cases. For the constrained case, the unconstrained gradient estimate $\hat{J}_u$ should be multiplied by the nullspace of the active constraints and the resulting "reduced gradient" controlled to zero.

From «exact local method» of self-optimizing control:

$$H^J = J_{uu}\left[G^{yT}\left(\tilde{F}\tilde{F}^T\right)^{-1}G^y\right]^{-1}G^{yT}\left(\tilde{F}\tilde{F}^T\right)^{-1}$$

$$\text{where } \tilde{F} = [FW_d \quad W_{n^y}] \text{ and } F = \frac{dy^{opt}}{dd} = G_d^y - G^y J_{uu}^{-1} J_{ud}.$$

- Bernardino and Skogestad, Optimal measurement-based cost gradient estimate for real-time optimization, Comp. Chem. Engng., 2024

With constraints

Constrained optimization problem

$$\min_{\mathbf{u}} \quad J(\mathbf{u}, \mathbf{y}, \mathbf{d})$$
$$\text{s.t.} \quad \mathbf{g}(\mathbf{u}, \mathbf{y}, \mathbf{d}) \leq 0$$

Solution: Turn into unconstrained optimization problem using Lagrange multipliers

$$\mathcal{L}(\mathbf{u}, \mathbf{y}, \mathbf{d}, \boldsymbol{\lambda}) = J(\mathbf{u}, \mathbf{y}, \mathbf{d}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{u}, \mathbf{y})$$

$\min_{u,\lambda} L$

u = primal variables = inputs

λ ≥ 0 = dual variables = Lagrange multipliers = shadow prices

Necessary conditions of optimality (KKT-conditions)

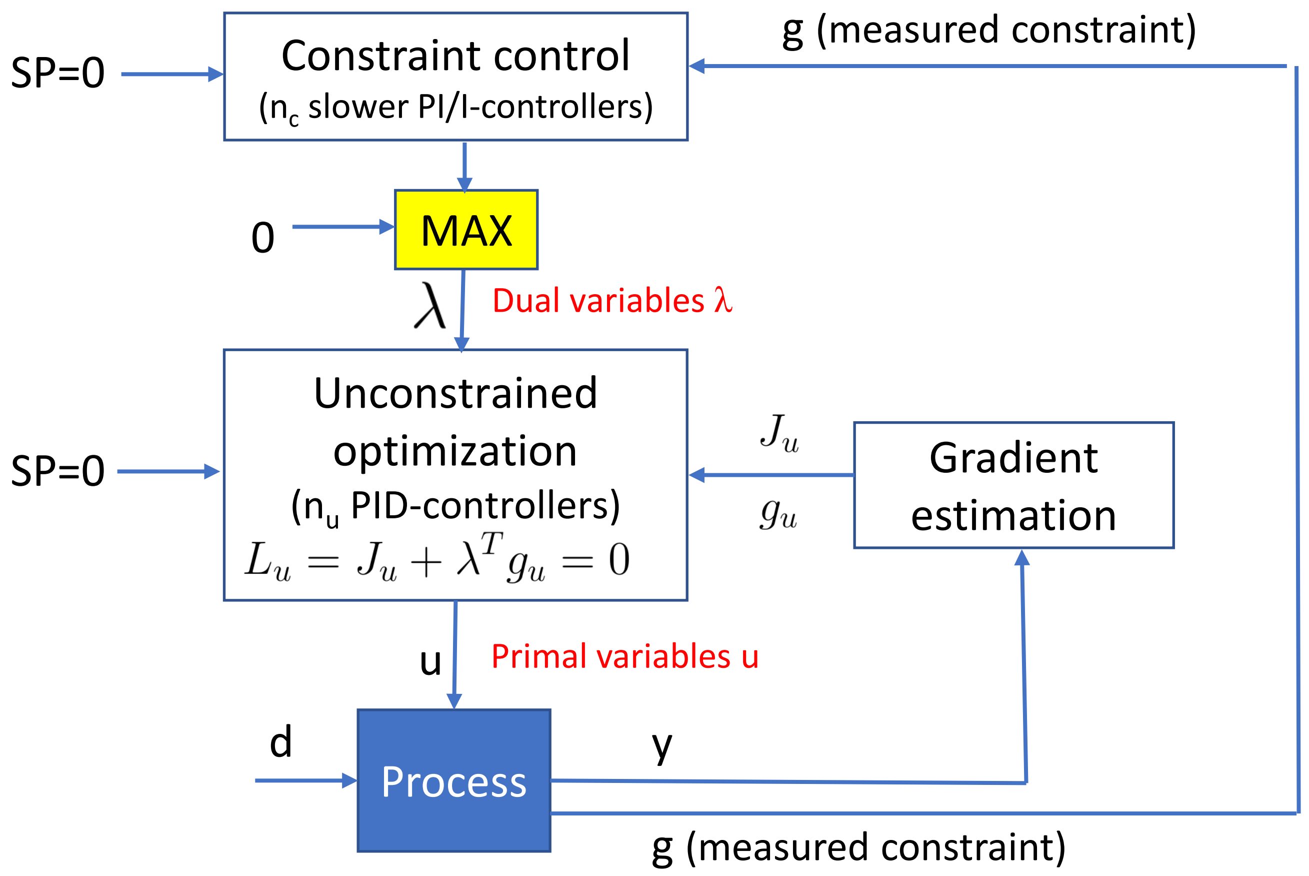
$$\nabla_u \mathcal{L} = 0, \quad \lambda \geq 0, \quad g \cdot \lambda = 0$$

(complementary condition)

$$\boxed{L_u = J_u + \lambda^T g_u = 0}$$

Feedback RTO with constyraints

A. Primal-dual control based on KKT conditions: Feedback solution that automatically tracks active constraints by adjusting Lagrange multipliers (= shadow prices = dual variables) λ



KKT: $L_u = J_u + \lambda^T g_u = 0$

Inequality constraints: $\lambda \geq 0$

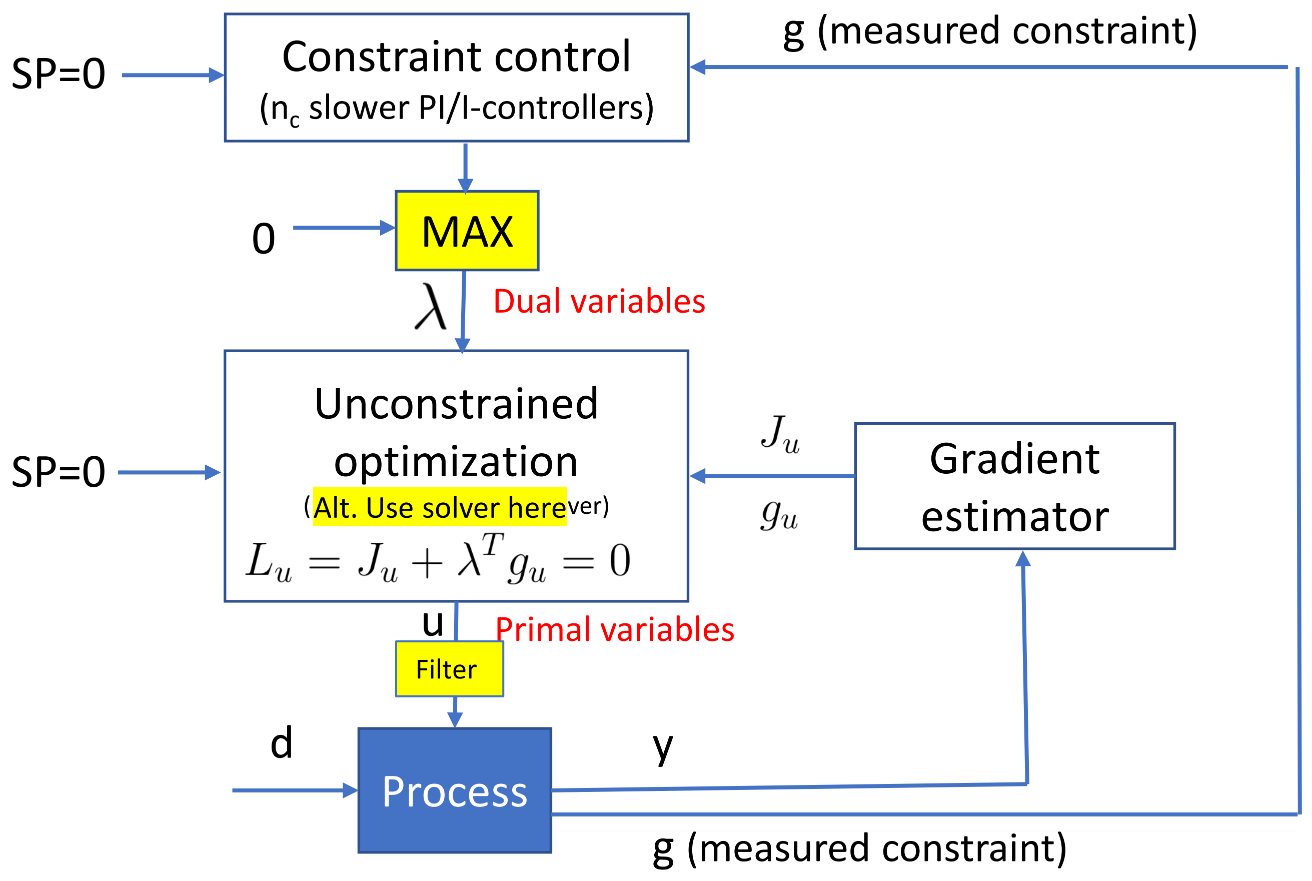
Primal-dual feedback control.

- Makes use of «dual decomposition» of KKT conditions
- Selector on dual variables λ
- Problem: Constraint control using dual variables is on slow time scale (upper layer)
  - Can be fixed using override at bottom of hiearchy (Dirza)
- Problem 2: Single-loop PID control in lower layer (L_u=0) may not be possible for coupled processes so may need to use Solver.

- D. Krishnamoorthy, A distributed feedback-based online process optimization framework for optimal resource sharing, J. Process Co[illegible]
- R. Dirza and S. Skogestad . Primal–dual feedback-optimizing control with override for real-time optimization. J. Process Control, Vol. 138 (2024), 103208.

Alternative: Dual composition with optimization/solver for computing u (primal variables)

- May need to add filter to avoid instability



Alternative: Direct control of constraints

KKT: $L_u = J_u + \lambda^T g_u = 0$

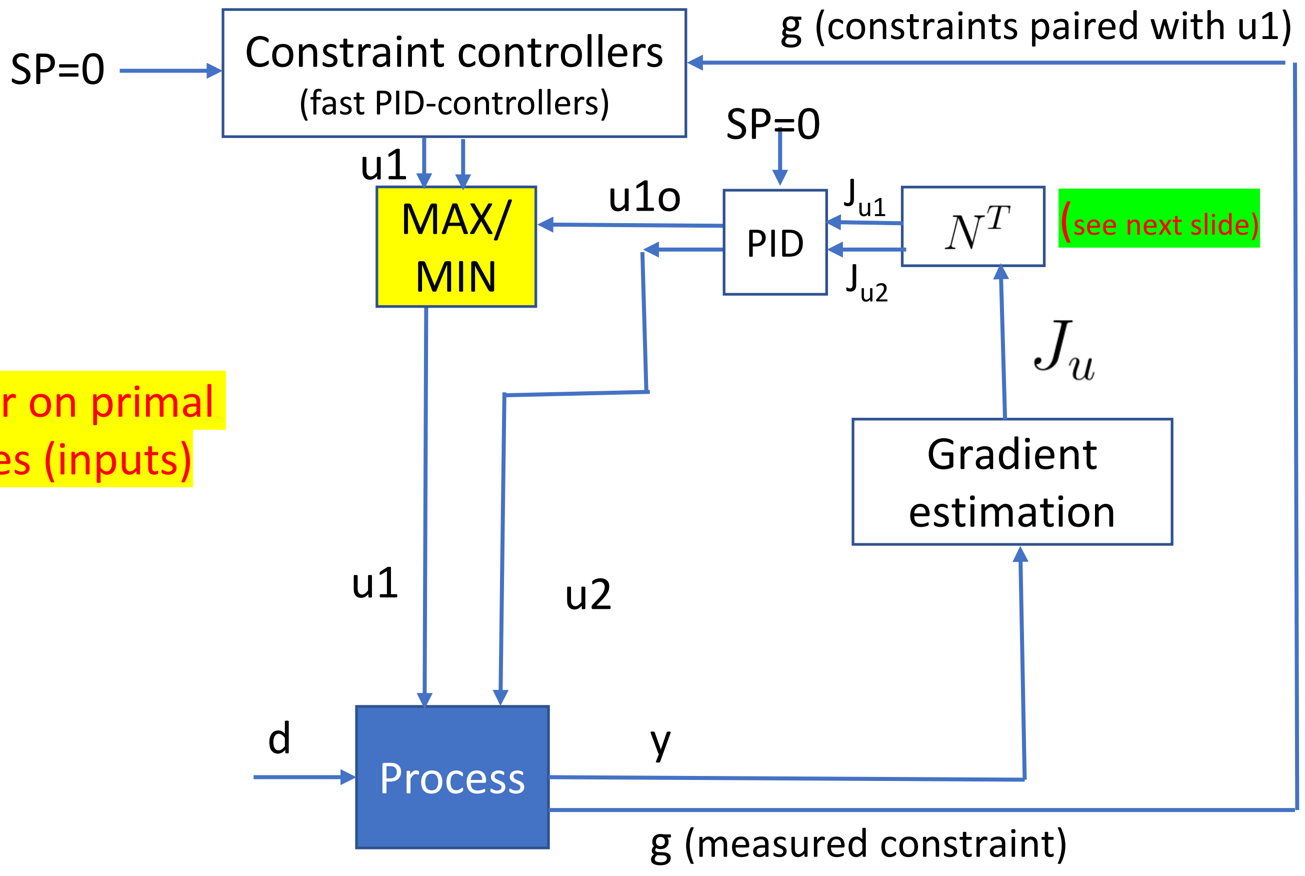
Introduce N: $N^T g_u = 0$

Control
1. Active constraints $g_A = 0$.
2. Reduced gradient $N_A{}^T J_u = 0$
   - for the remaining inbconstrained degrees of freedom
   - «self-optimizing variables»

Seems easy. But how do we handle changes in constraints?
- Because g_A and N_A varies
- Originally, I thought we need a new control structure (with pairings) in each region

- Jaschke and Skogestad, «Optimal controlled variables for¨ polynomial systems». S., J. Process Control, 2012
- D. Krishnamoorthy and S. Skogestad, «Online Process Optimization with Active Constraint Set Changes using Simple Control Structure», I&EC Res., 2019

B. Region-based feedback solution with «direct» constraint control



KKT: $L_u = J_u + \lambda^T g_u = 0$

Introduce N: $N^T g_u = 0$

- Selector on primal variables (inputs)

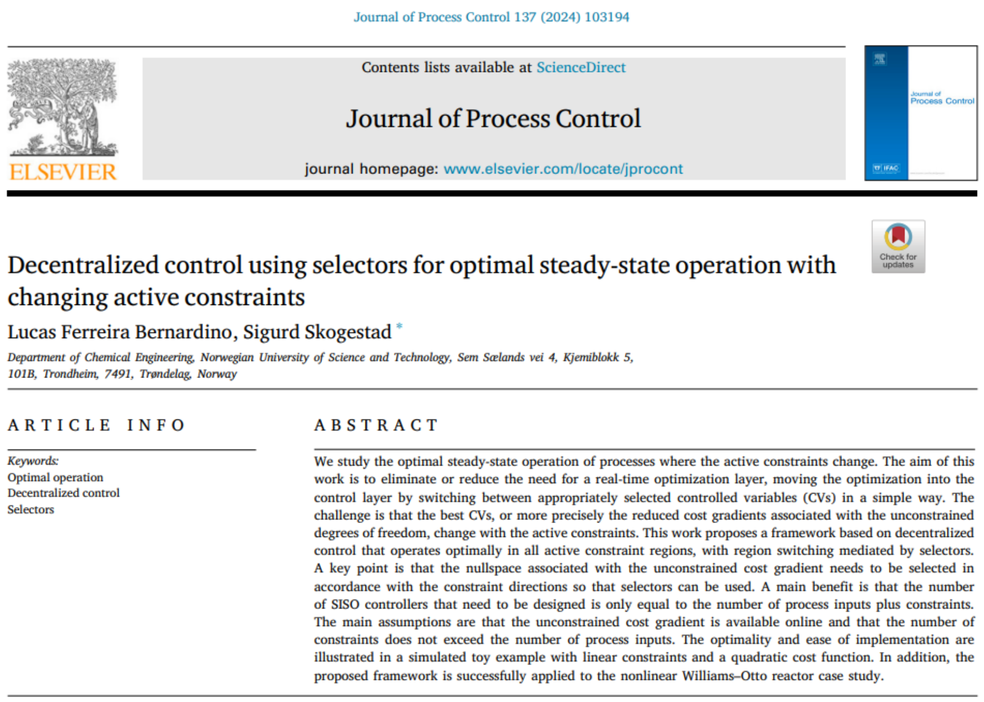
- Selector on primal variables (inputs)
- Similar to selectors in ARC
- Limitation: need to pair each constraint with an input u, may not work if many constraints

- Jaschke and Skogestad, «Optimal controlled variables for¨ polynomial systems». S., J. Process Control, 2012
- D. Krishnamoorthy and S. Skogestad, «Online Process Optimization with Active Constraint Set Changes using Simple Control Structure», I&EC Res., 2019
- L. Bernadino and S. Skogestad, Decentralized control using selectors for optimal steady-state operation with active constraints, J. Proc. Control, 2024

Feedback RTO with constraints

Assume: Have at least as many inputs as constraints

Can them have fixed pairings between constraints and unconstrained CVs! (with N is fixed)



Journal of Process Control 137 (2024) 103194

Contents lists available at ScienceDirect

Journal of Process Control

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

Decentralized control using selectors for optimal steady-state operation with changing active constraints

Lucas Ferreira Bernardino, Sigurd Skogestad *

Department of Chemical Engineering, Norwegian University of Science and Technology, Sem Sælands vei 4, Kjemiblokk 5, 101B, Trondheim, 7491, Trøndelag, Norway

ARTICLE INFO

Keywords:
Optimal operation
Decentralized control
Selectors

ABSTRACT

We study the optimal steady-state operation of processes where the active constraints change. The aim of this work is to eliminate or reduce the need for a real-time optimization layer, moving the optimization into the control layer by switching between appropriately selected controlled variables (CVs) in a simple way. The challenge is that the best CVs, or more precisely the reduced cost gradients associated with the unconstrained degrees of freedom, change with the active constraints. This work proposes a framework based on decentralized control that operates optimally in all active constraint regions, with region switching mediated by selectors. A key point is that the nullspace associated with the unconstrained cost gradient needs to be selected in accordance with the constraint directions so that selectors can be used. A main benefit is that the number of SISO controllers that need to be designed is only equal to the number of process inputs plus constraints. The main assumptions are that the unconstrained cost gradient is available online and that the number of constraints does not exceed the number of process inputs. The optimality and ease of implementation are illustrated in a simulated toy example with linear constraints and a quadratic cost function. In addition, the proposed framework is successfully applied to the nonlinear Williams–Otto reactor case study.

L. Bernadino and S. Skogestad, Decentralized control using selectors for optimal steady-state operation with active constraints, J. Proc. Control, 2024

Journal of Process Control 137 (2024) 103194

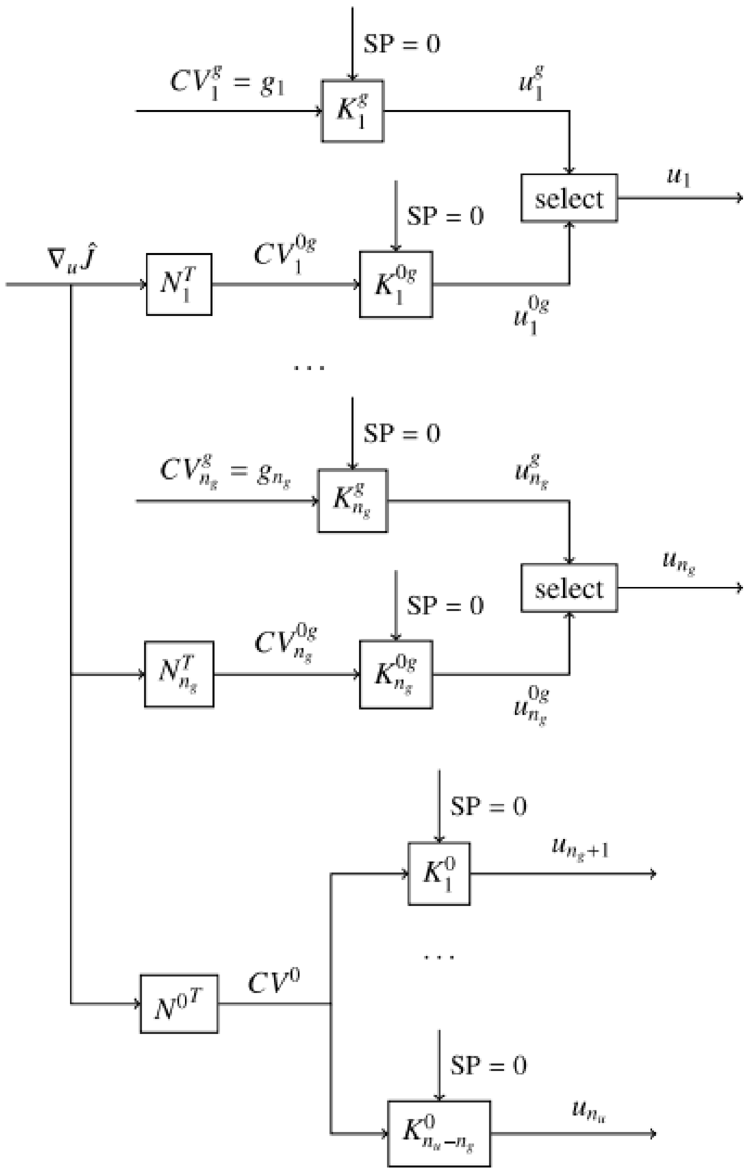


Fig. 3. Decentralized control structure for optimal operation according to Theorem 2. The "select" blocks are usually max or min selectors (see Theorem 3).

Feedback RTO WITH CONSTRAINTS

C. Region-based MPC with switching of cost function (for general case)

Standard MPC with fixed CVs: Not optimal

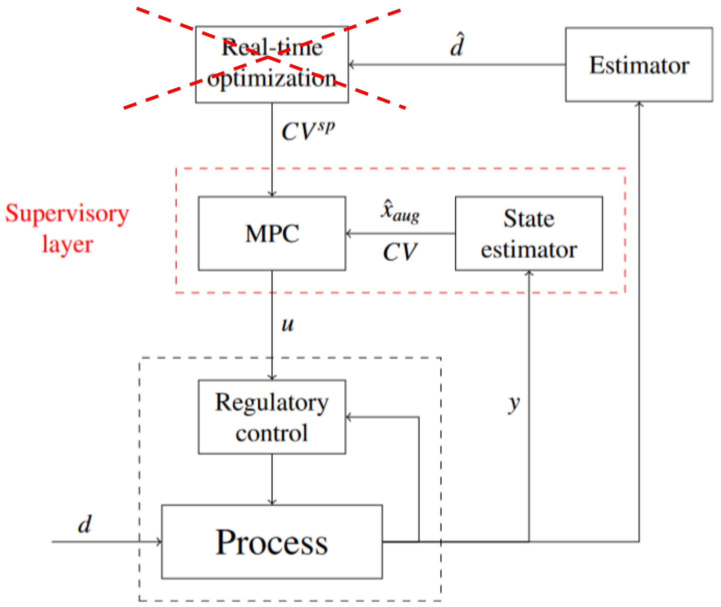


Figure 1: Typical hierarchical control structure with standard setpoint-tracking MPC in the supervisory layer. The cost function for the RTO layer is J^{ec} and the cost function for the MPC layer is J^{MPC}. With no RTO layer (and thus constant setpoints CV^{sp}), this structure is not economically optimal when there are changes in the active constraints. For smaller applications, the state estimator may be used also as the RTO estimator.

$$J^{MPC} = \sum_{k=1}^{N} \|CV_k - CV^{sp}\|_Q^2 + \|\Delta u_k\|_R^2$$

Proposed: With changing cost (switched CVs)

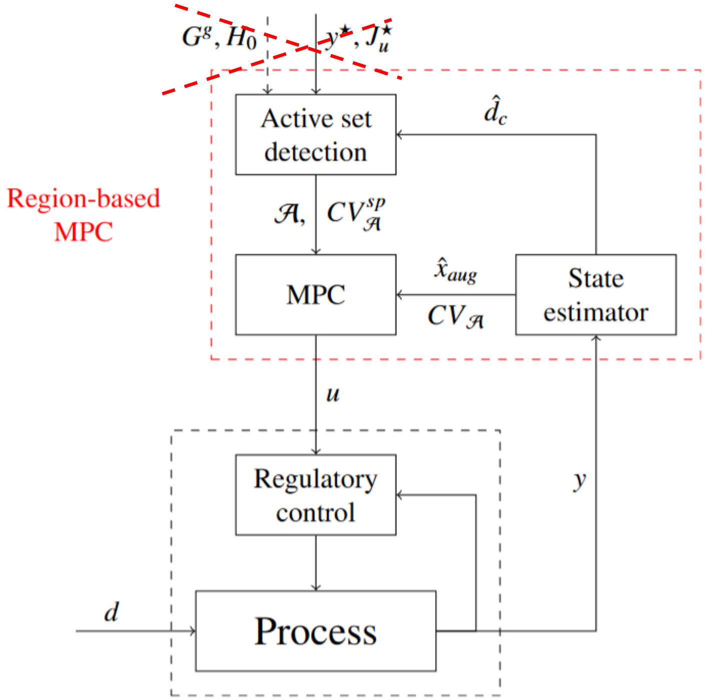


Figure 2: Proposed region-based MPC structure with active set detection and change in controlled variables. The possible updates from an upper RTO layer (y^*, J_u^* etc.) are not considered in the present work. Even with no RTO layer (and thus with constant setpoints $CV_{\mathcal{A}}^{sp}$, see (14) and (15), in each active constraint region), this structure is potentially economically optimal when there are changes in the active constraints.

$$J_{\mathcal{A}}^{MPC} = \sum_{k=1}^{N} \|CV_{\mathcal{A}} - CV_{\mathcal{A}}^{sp}\|_{Q_{\mathcal{A}}}^2 + \|\Delta u_k\|_{R_{\mathcal{A}}}^2$$

$$CV_{\mathcal{A}} = \begin{bmatrix} g_{\mathcal{A}} \\ c_{\mathcal{A}} \end{bmatrix} = \begin{bmatrix} g_{\mathcal{A}} \\ N_{\mathcal{A}}^T H_0 y \end{bmatrix} \tag{14}$$

$$H_0 = \begin{bmatrix} J_{uu} & J_{ud} \end{bmatrix} \begin{bmatrix} G^y & G_d^y \end{bmatrix}^\dagger$$

$$H^J = J_{uu} \left[G^{yT} \left(\tilde{F}\tilde{F}^T \right)^{-1} G^y \right]^{-1} G^{yT} \left(\tilde{F}\tilde{F}^T \right)^{-1}$$

- Bernardino and Skogestad, Optimal switching of MPC cost function for changing active constraints. J. Proc. Control, 2024

Model-free optimization: Extremum Seeking Control (ESC) based on measuring cost J

$$\min_{u} J(u,d)$$

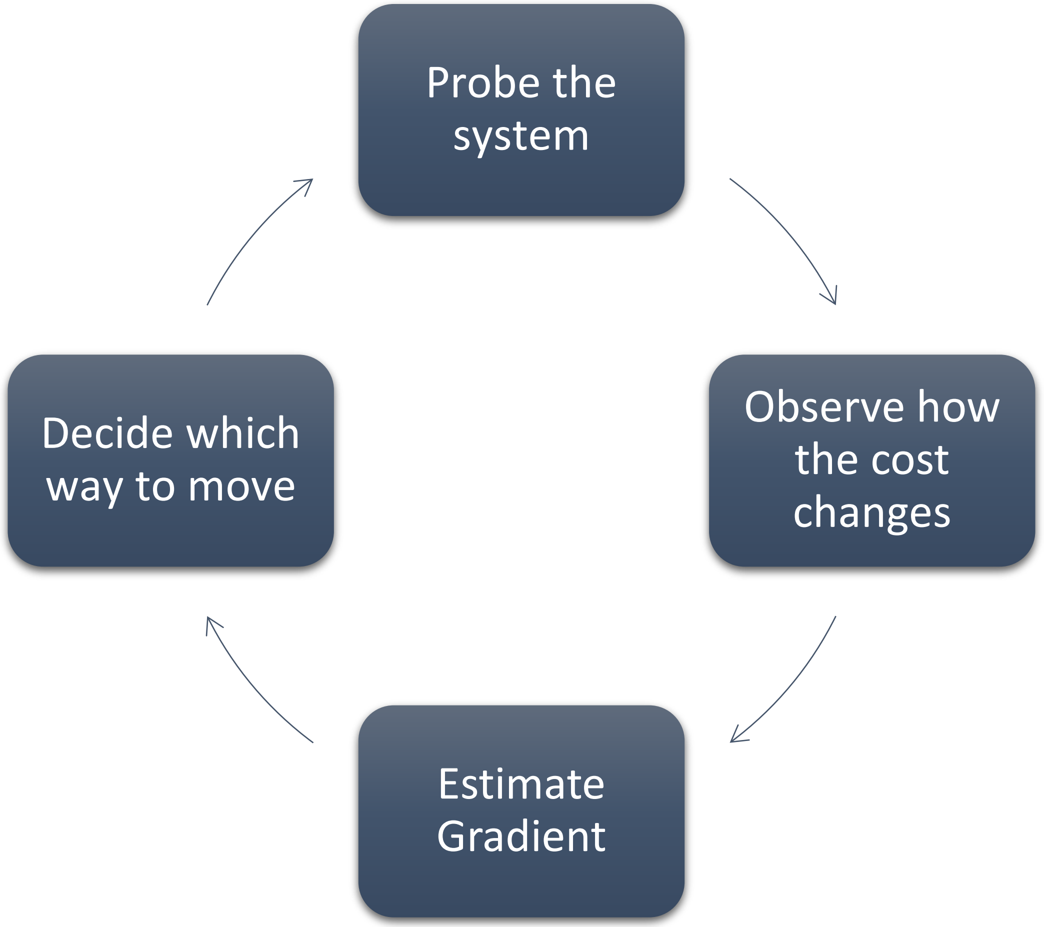
Why ESC?

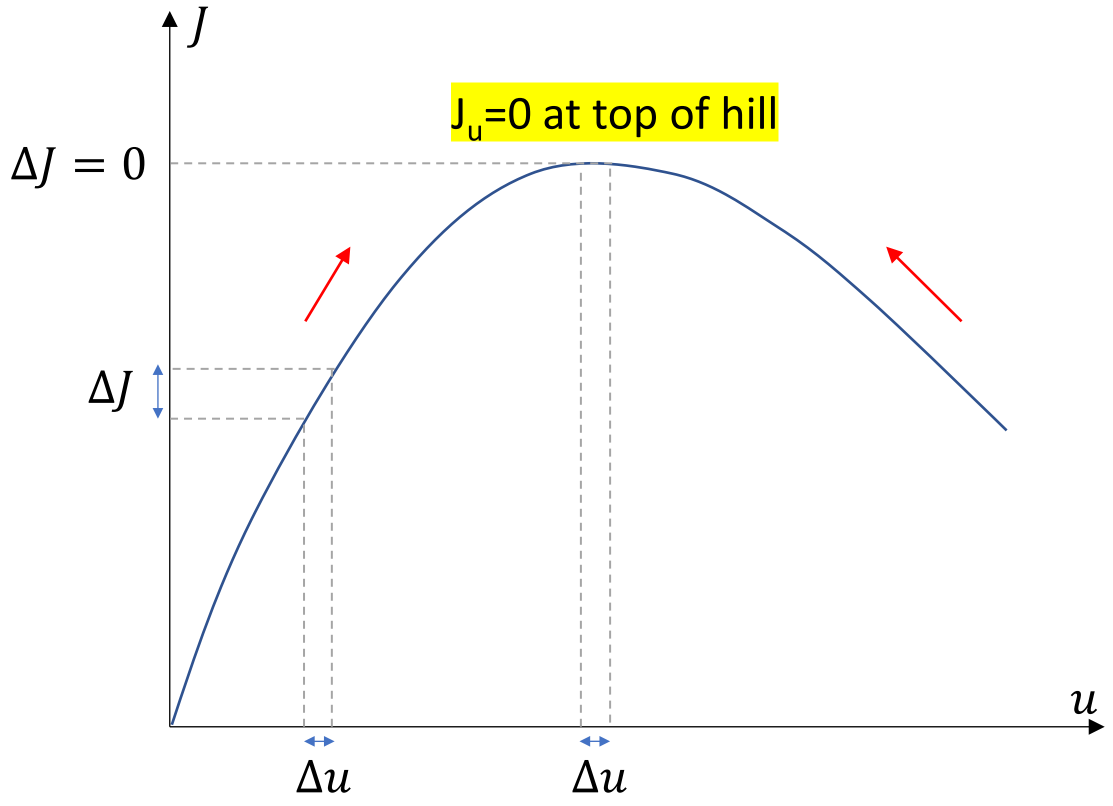
- Expensive to obtain model (for J): Use data-based ESC instead of model-based RTO
- May also be used on top of RTO
  - «Adapt» setpoint for J_u (to a nonzero bias value) to correct for model error
  - Aka «modifier adaptation»

Main problems with ESC:

- Cost function J often not measured
  - For chemical process $J=p_F F - p_P P - p_Q Q$
  - need model (!) to estimate flows F, P and utility Q
- Very slow. Typically 100 times slower than process dynamics

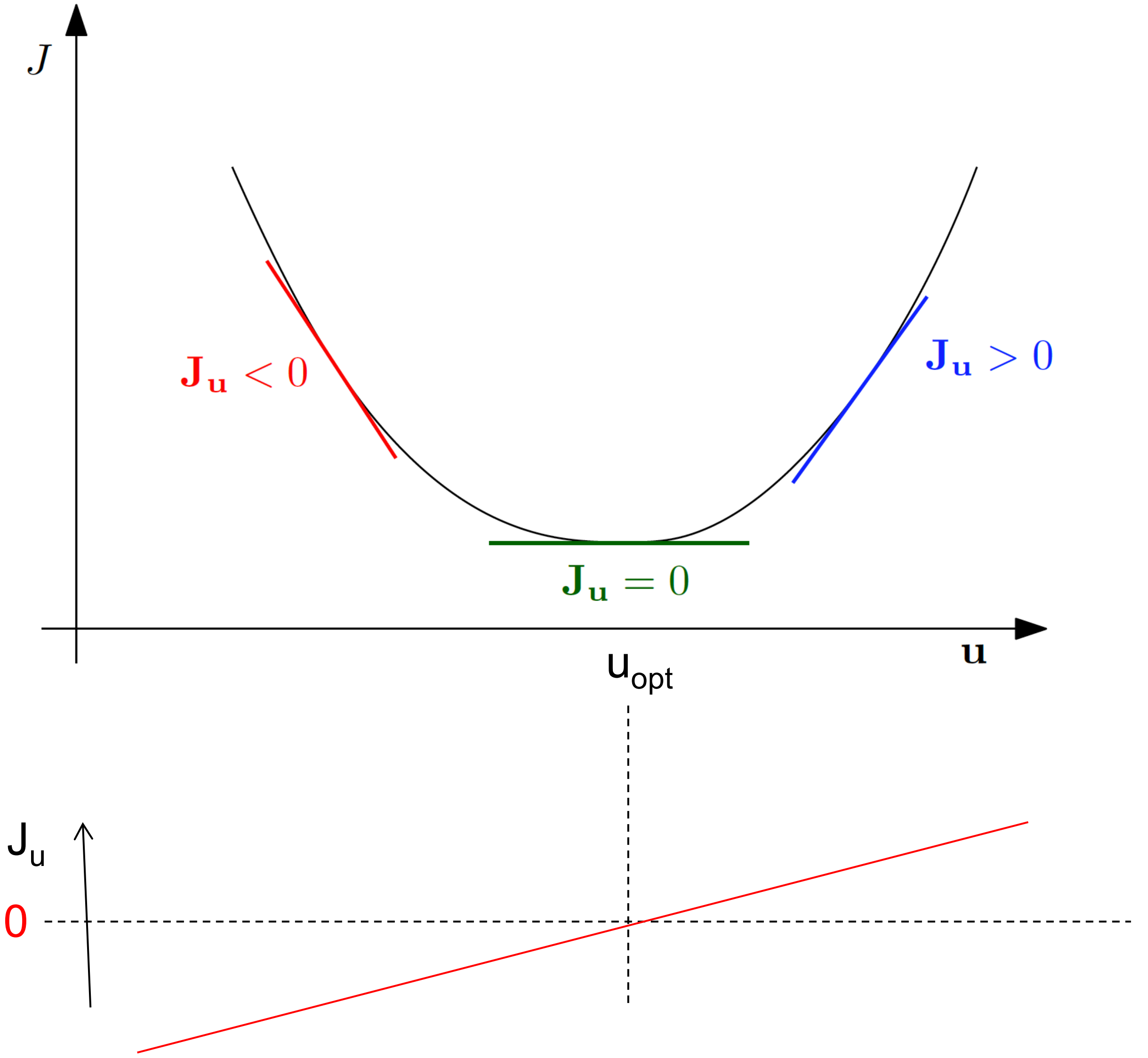
Data-based optimization: "Hill-climbing" / "Extremum seeking control" Drive gradient $J_u = dJ/du$ to zero.





NTNU

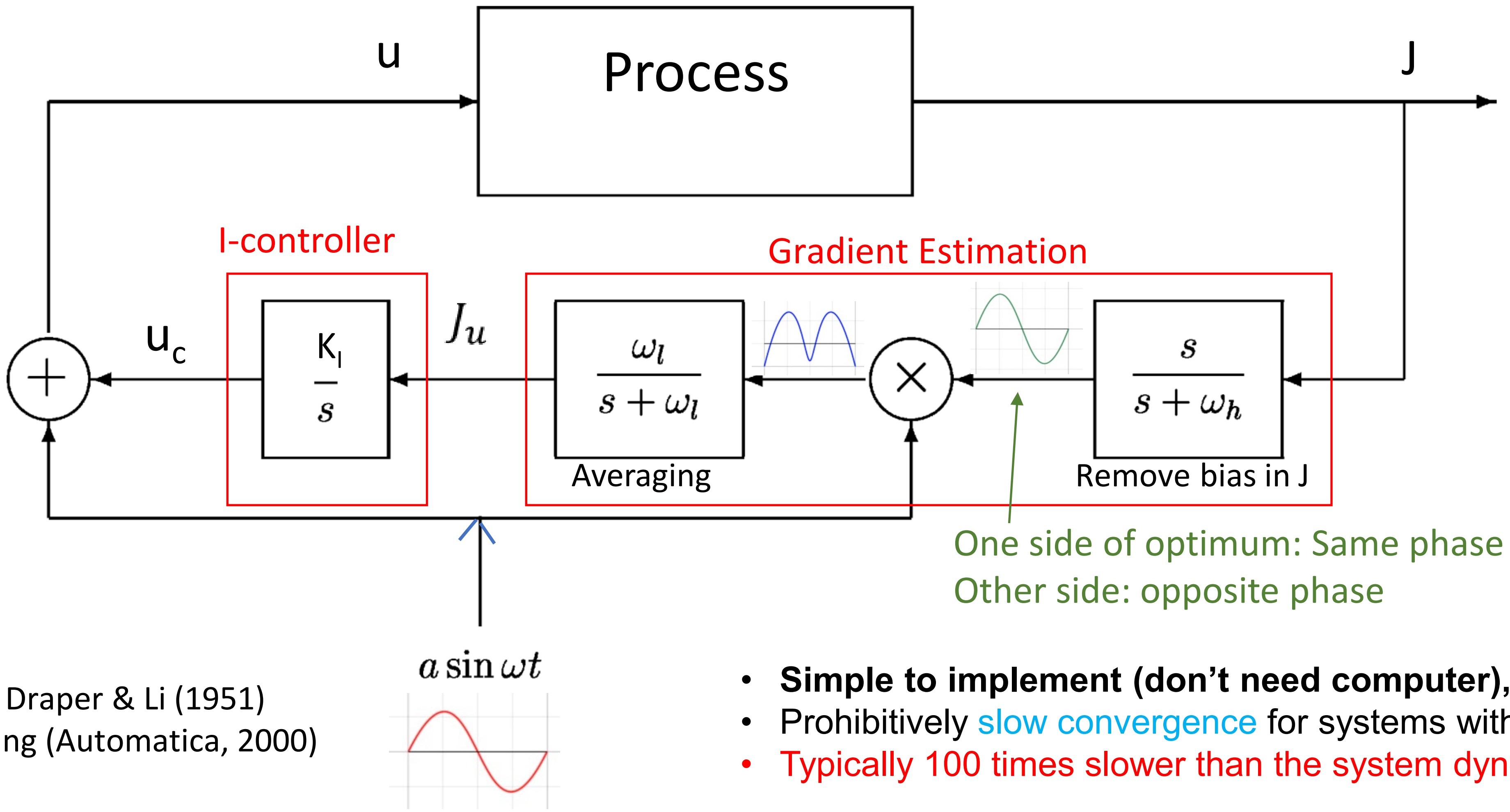
Equivalent: Minimize cost J (go to bottom of valley)



- Optimal setpoint: $J_u=0$
- If Hessian J_{uu} is constant:
    - J_u as a function of u is a straight line with slope J_{uu}
- Nice properties for feedback control of J_u
- No dynamics: Pure I-controller optimal
    - SIMC-rule: $K_I = 1/(J_{uu}\ \tau_c)$

NTNU

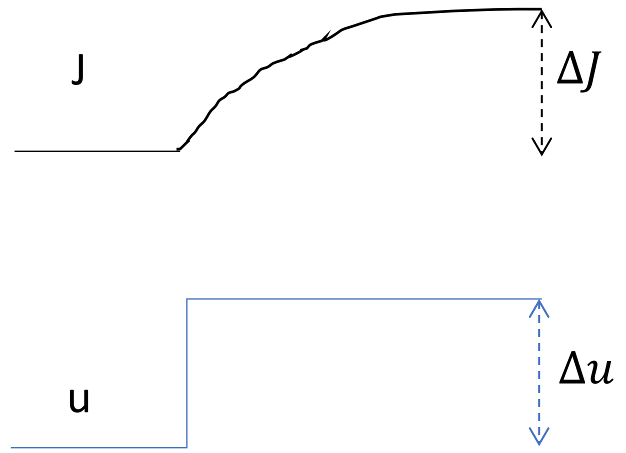
Classical Extremum seeking control using sinusoids



Multiplication trick: Draper & Li (1951)
Theory: Krstic & Wang (Automatica, 2000)

- **Simple to implement (don't need computer),** but
- Prohibitively slow convergence for systems with slow dynamics
- Typically 100 times slower than the system dynamics !

More common today: Estimate Steady-state gradient using discrete perturbations (steps)



$$J_u = \frac{\Delta J}{\Delta u}$$

Usually only one input. Simplest: step change in u:

- Hill climbing control (Shinskey, 1967)
- Evolutionary operation (EVOP) (1960's)
- NCO tracking (Francois & Bonvin, 2007)
- "Peturb and observe" = Maximum power point tracking (MPPT) (2010's).

More advanced variants which may also be applied to multivariable systems

- **Least squares estimation**
- Fast Fourier transform (Dinesh Krishnamoorthy)

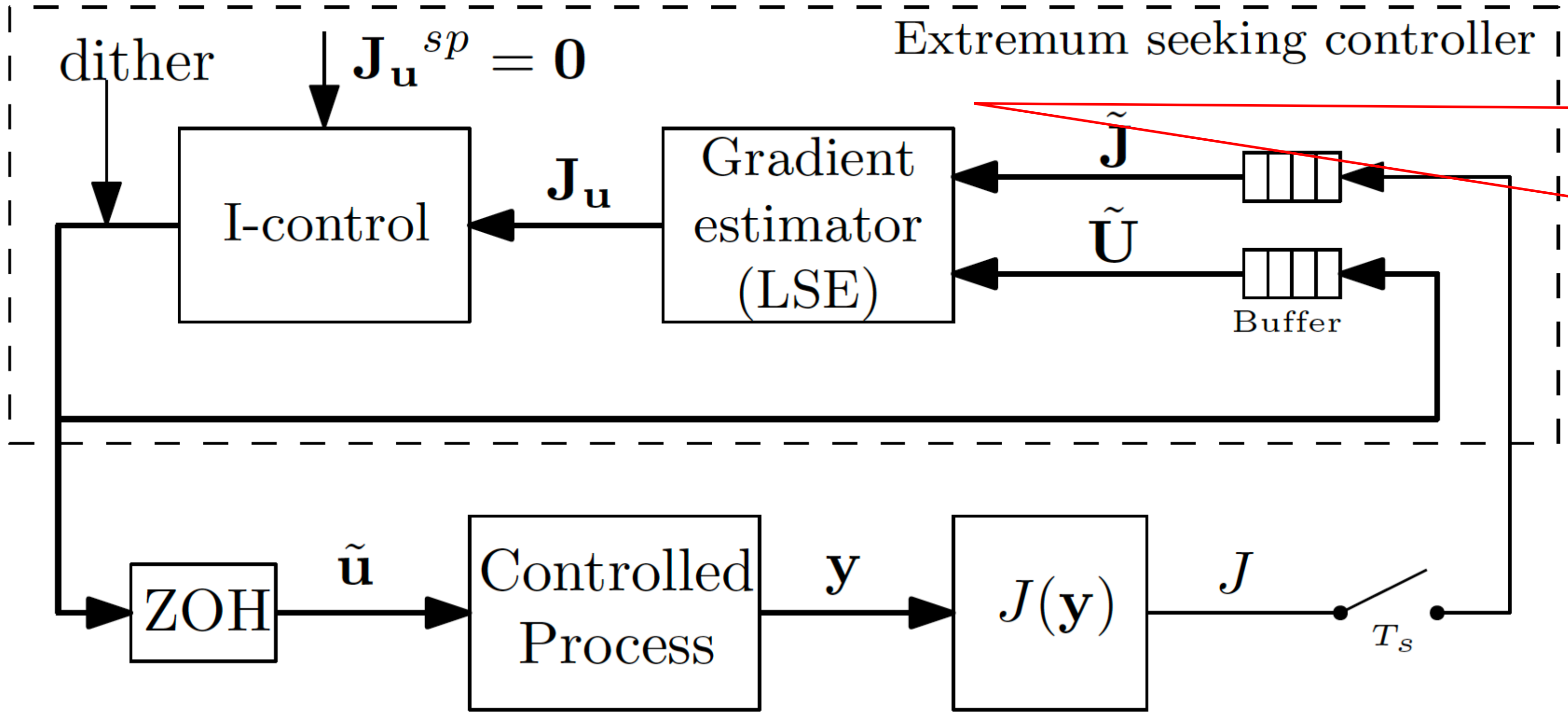
To avoid waiting for steady state

- Fitting of data to ARX model (difficult to make robust)

Note: Assumes steady state -> samling (step) time > 3-10 time process time constant

NTNU

Least square Extremum seeking control



LSE: Fit a linear model

$$J = \mathbf{J}_{\tilde{\mathbf{u}}}{}^{\mathsf{T}} \tilde{\mathbf{u}} + m$$

Using least squares fit

$$\mathbf{Y} = [J_k, J_{k-1}, \ldots, J_{k-N+1}]^T$$
$$\mathbf{U} = [\mathbf{u}_k, \ldots, \mathbf{u}_{k-N+1}]^T$$
$$\theta = [\hat{\mathbf{J}}_{\mathbf{u}}^T, m]^T$$

$$\hat{\theta} = \arg\min_{\theta} \|\mathbf{Y} - \Phi^T\theta\|_2^2$$

to which the analytical solution is given by

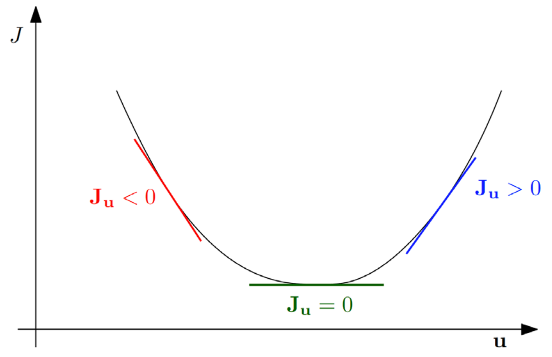
$$\hat{\theta} = [\Phi^T\Phi]^{-1}\Phi^T\mathbf{Y}$$

Note: Assumes no dynamics -> samling time > 3-10 time process constant

Hunnekens et al. (2011, 2014)

NTNU

Summary extremum seeking control



Idea: Estimate the cost gradient J_u from data and drive it to zero

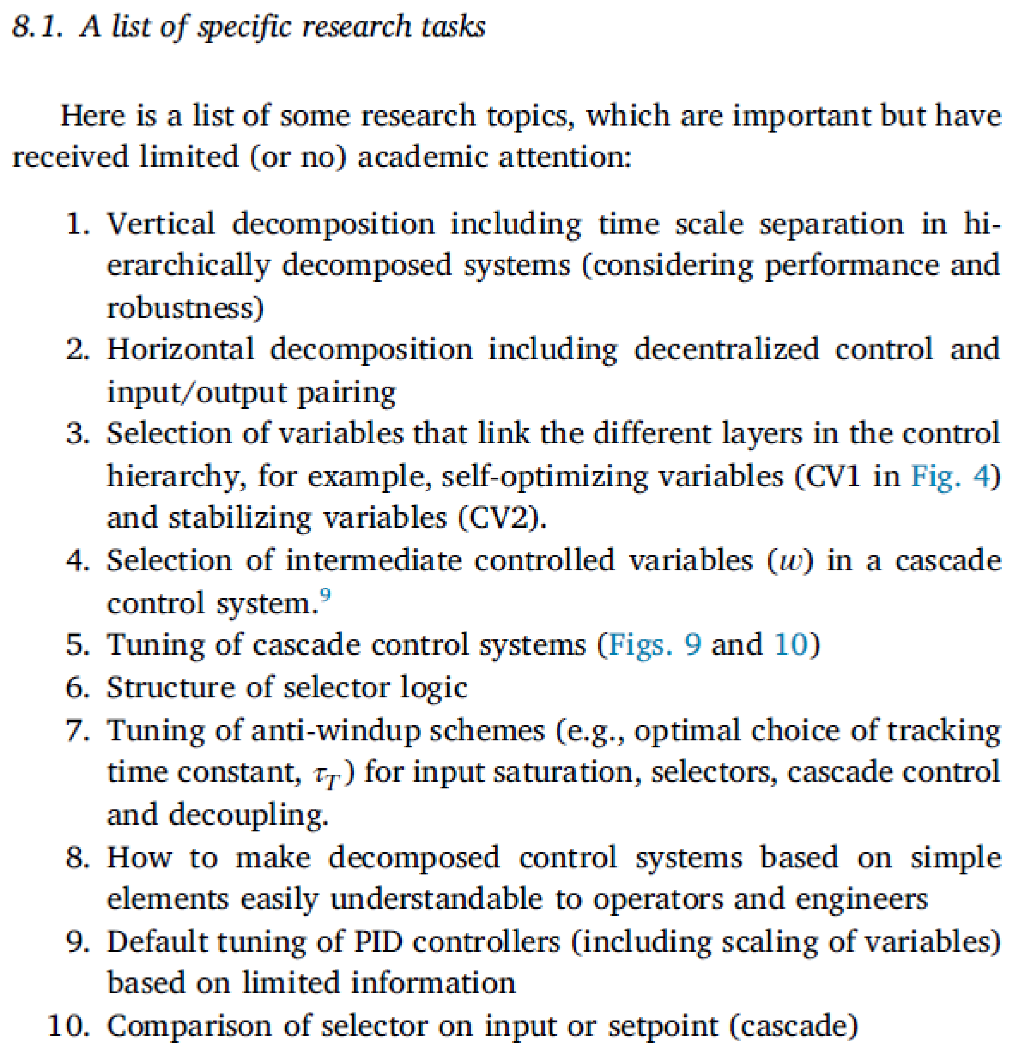
- Common to all methods:
  - Need measurement of cost J
  - Must wait for steady state (except ARX method which fails frequently)
  - Must assume no «fast» disturbances (while optimizing)

Algorithm needs two layers on top of process:

1. Optimization layer (slowest): Drive J_u to zero (may use I-controller)
2. Lower estimation layer: Estimate the local gradient J_u using data
   - Must wait for the process to reach steady state

- Need time scale separation between layers.
  - At best this means that the optimization needs to be 10 times slower than the process.
  - Often it needs to be 100 times slower.
- **Useful for fast processes with settling time a few seconds**
- Not useful for many chemical processes where time constant typically are several minutes
  - 10 minutes * 100 = 1000 minutes = 16 hours
  - Unllikely with 16 hours without disturbance

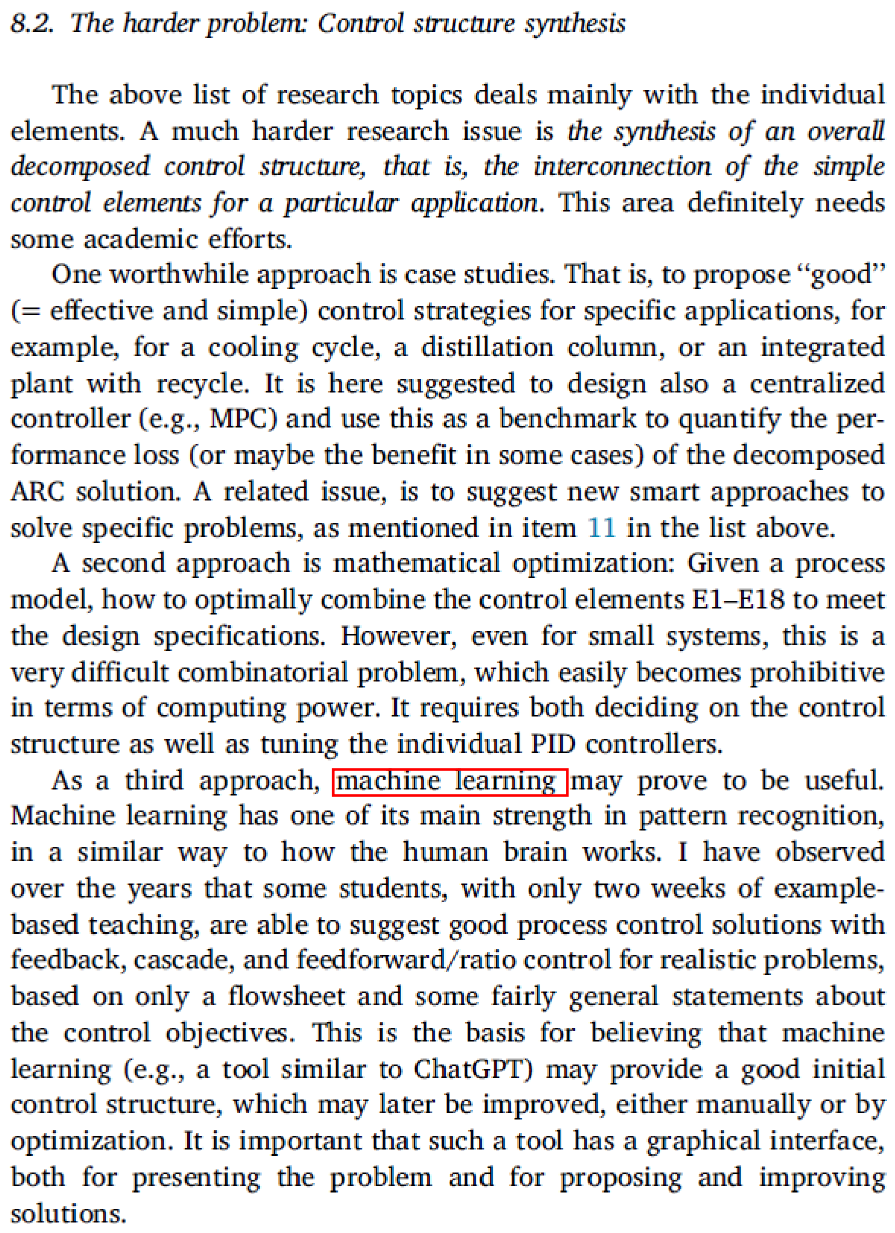
ARC: Research tasks



8.1. A list of specific research tasks

Here is a list of some research topics, which are important but have received limited (or no) academic attention:

1. Vertical decomposition including time scale separation in hierarchically decomposed systems (considering performance and robustness)
2. Horizontal decomposition including decentralized control and input/output pairing
3. Selection of variables that link the different layers in the control hierarchy, for example, self-optimizing variables (CV1 in Fig. 4) and stabilizing variables (CV2).
4. Selection of intermediate controlled variables (w) in a cascade control system.[9]
5. Tuning of cascade control systems (Figs. 9 and 10)
6. Structure of selector logic
7. Tuning of anti-windup schemes (e.g., optimal choice of tracking time constant, τ_T) for input saturation, selectors, cascade control and decoupling.
8. How to make decomposed control systems based on simple elements easily understandable to operators and engineers
9. Default tuning of PID controllers (including scaling of variables) based on limited information
10. Comparison of selector on input or setpoint (cascade)



8.2. The harder problem: Control structure synthesis

The above list of research topics deals mainly with the individual elements. A much harder research issue is *the synthesis of an overall decomposed control structure, that is, the interconnection of the simple control elements for a particular application*. This area definitely needs some academic efforts.

One worthwhile approach is case studies. That is, to propose "good" (= effective and simple) control strategies for specific applications, for example, for a cooling cycle, a distillation column, or an integrated plant with recycle. It is here suggested to design also a centralized controller (e.g., MPC) and use this as a benchmark to quantify the performance loss (or maybe the benefit in some cases) of the decomposed ARC solution. A related issue, is to suggest new smart approaches to solve specific problems, as mentioned in item 11 in the list above.

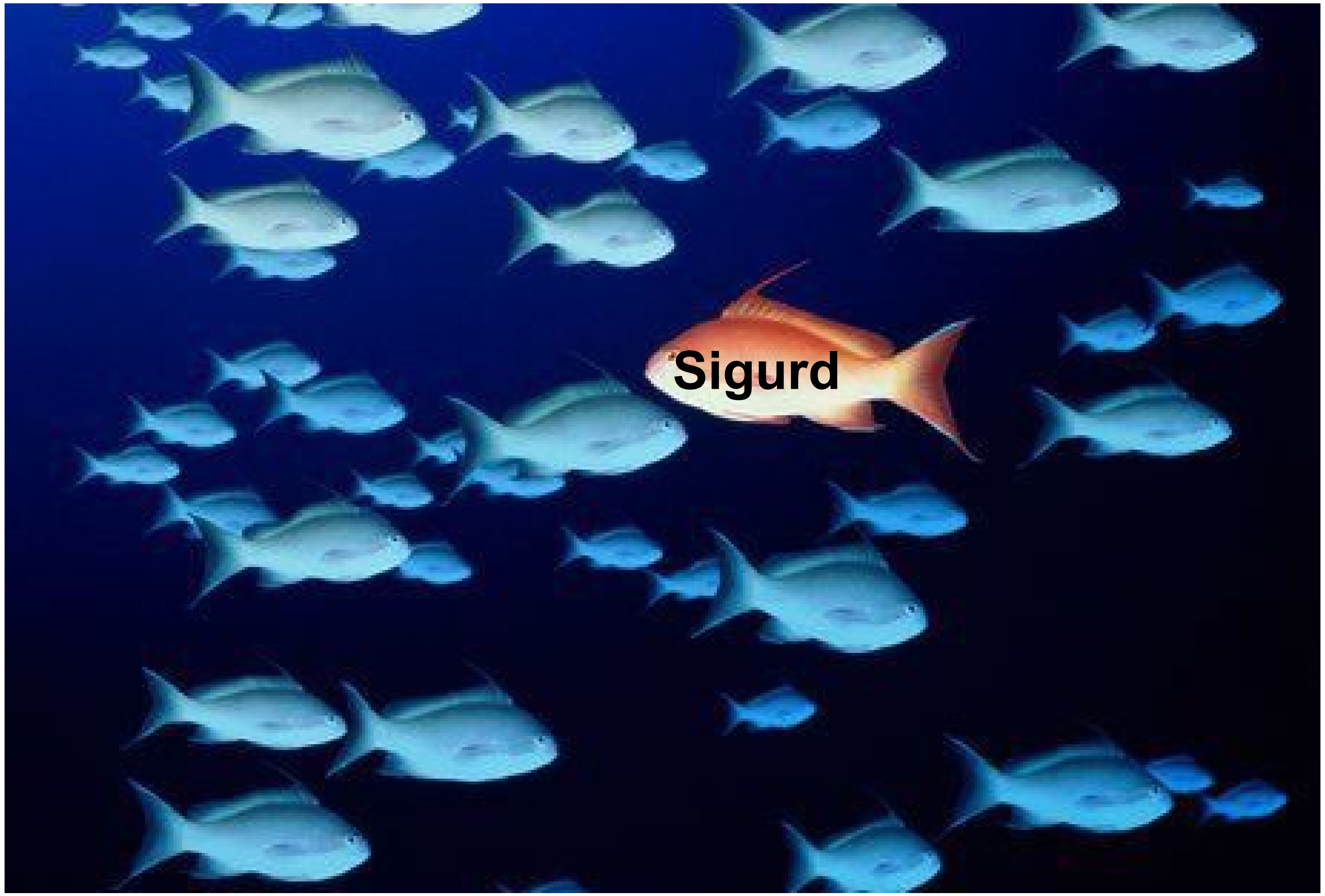
A second approach is mathematical optimization: Given a process model, how to optimally combine the control elements E1–E18 to meet the design specifications. However, even for small systems, this is a very difficult combinatorial problem, which easily becomes prohibitive in terms of computing power. It requires both deciding on the control structure as well as tuning the individual PID controllers.

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.

Present Academic control community fish pond

Complex optimal centralized Solution (**EMPC, FL**)

Simple solutions that work (ARC, PID)



FL = feedback linearization

Future Academic control community fish pond

Complex optimal centralized Solution (**EMPC, FL**)

Simple solutions that work (SRC,PID)