NTNU – Trondheim
Norwegian University of
Science and Technology

SUBPRO
SUBSEA PRODUCTION AND PROCESSING

Novel Approaches to Online Process Optimization under Uncertainty

Dinesh Krishnamoorthy

Department of Chemical Engineering
Norwegian University of Science and Technology (NTNU)
dinesh.krishnamoorthy@ntnu.no

07 November 2019

Outline

1. Research question - Why? and How?
2. Part 1 - Optimal steady-state operation using transient measurements
3. Part 2 - Dynamic optimization - addressing computation time
4. Summary and future outlook

Outline

1. Research question - Why? and How?
2. Part 1 - Optimal steady-state operation using transient measurements
3. Part 2 - Dynamic optimization - addressing computation time
4. Summary and future outlook

Research Question - Why?

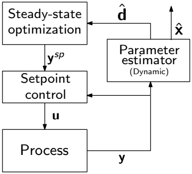
Why is traditional RTO not commonly used in industry?

- **Challenge 1** - Cost of developing the model (offline).
- **Challenge 2** - Model uncertainty, including wrong values of disturbances and parameters (online update).
- **Challenge 3** - Numerical robustness, including computational issues of solving optimization problems.
- **Challenge 4** - Frequent grade changes, which makes steady-state optimization less relevant.
- **Challenge 5** - Dynamic limitations, including infeasibility due to (dynamic) constraint violation.
- **Challenge 6** - Problem formulation - choosing the right formulation for the right problem.

- **Human factors** - Corporate culture and technical competence

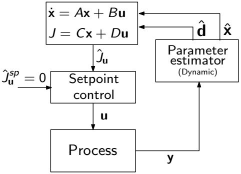
My Main Contributions - How?

Hybrid RTO



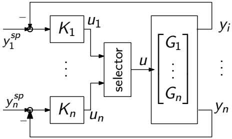
Challenge 2 & 5

Feedback RTO



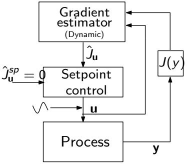
Challenge 2,3 & 5

Optimization using simple controllers



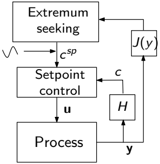
Challenge 1,2,3 & 5

Extremum seeking using transient measurements



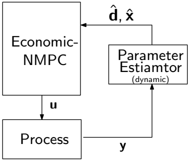
Challenge 1,2,3 & 5

Extremum seeking + Self-optimizing control



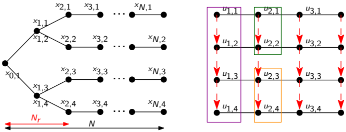
Challenge 1,2,3 & 5

economic MPC



Challenge 3,4& 5

Distributed multistage MPC using primal decomposition



Challenge 3, 4 & 5

Outline

1 Research question - Why? and How?

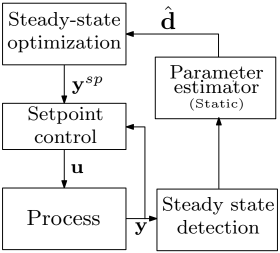
2 Part 1 - Optimal steady-state operation using transient measurements

3 Part 2 - Dynamic optimization - addressing computation time

4 Summary and future outlook

Conventional steady-state RTO

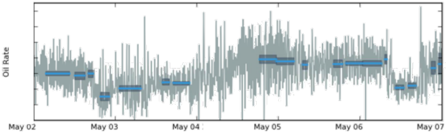
Two-step approach

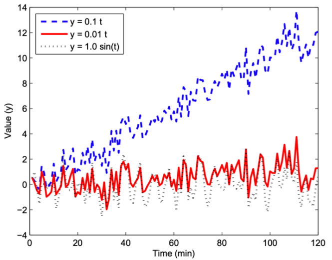


Chen & Joseph, *Ind. Eng. & Chem. Res* (1987)
Darby et al, *J. Proc. Control* (2011)

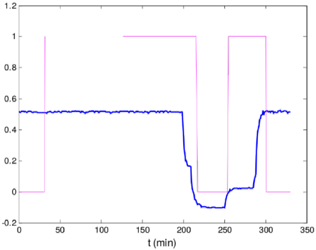
Steady-state wait time

Challenge 2 - Model uncertainty, including wrong values of disturbances and parameters (online update of the model).





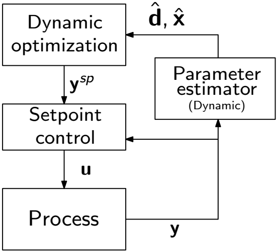
Source: Kelly, J.D. and Hedengren, J.D., 2013. A steady-state detection (SSD) algorithm to detect non-stationary drifts in processes. *Journal of Process Control*, *23*(3), pp.326-331.



Source: Câmara MM, Quelhas AD, Pinto JC. *Performance Evaluation of Real Industrial RTO Systems*. Processes. 2016, 4(4).

How to address steady-state wait time?

Obvious : Dynamic RTO



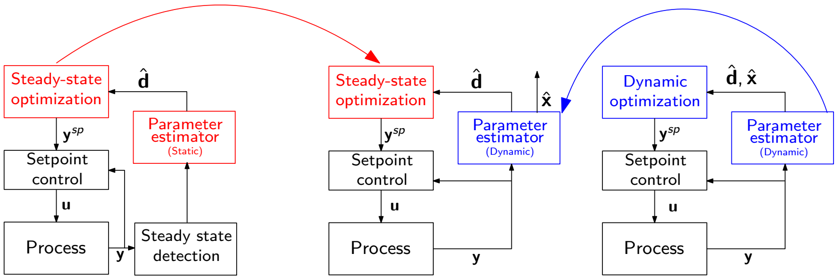
Dynamic RTO has problems - especially the optimization part (Challenge 3!)

Findeisen et al. *IFAC World Congress* (2004)

How to avoid steady-state wait time? - Hybrid RTO

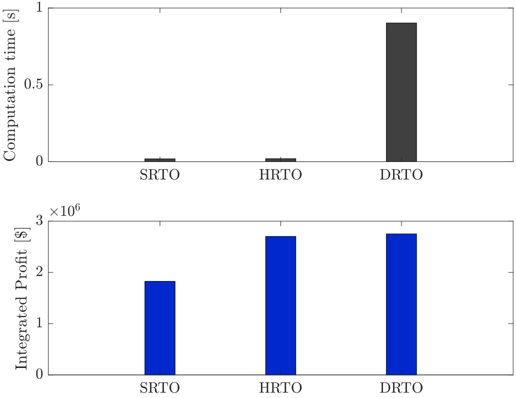
If the focus is steady-state optimization, dynamic terms are needed only for the model update.

- Dynamic models - Online model update
- Steady-state models - Optimization



Krishnamoorthy et al. *Comput. & Chem. Eng.* (2018)

Example: Hybrid RTO for oil production optimization

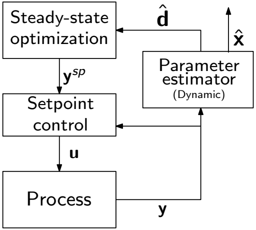


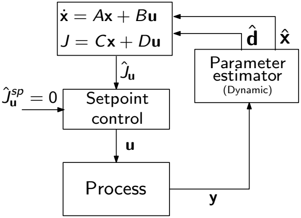
Krishnamoorthy et al. *(Submitted to ACC 2020)* - 6 gas-lifted wells with 20 differential states, 78 algebraic states and 6 inputs
Krishnamoorthy et al. *Comput. & Chem. Eng.* (2018)

Feedback RTO

Challenge 3 - Numerical robustness, including computational issues

Convert the Hybrid RTO problem into a **feedback control** problem





Krishnamoorthy et al. *Ind. Eng. Chem. Res* (2019)

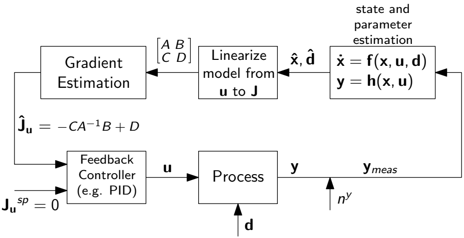
Novel gradient estimation using transient measurements

- **Step 1**- Linearize the dynamic model around $(\hat{x}, \hat{d})$

$$\begin{aligned} \dot{x} &= f(x,u,d) \\ J &= h(x,u,d) \end{aligned} \quad \Rightarrow \quad \begin{aligned} \dot{x} &= Ax + Bu \\ J &= Cx + Du \end{aligned}$$

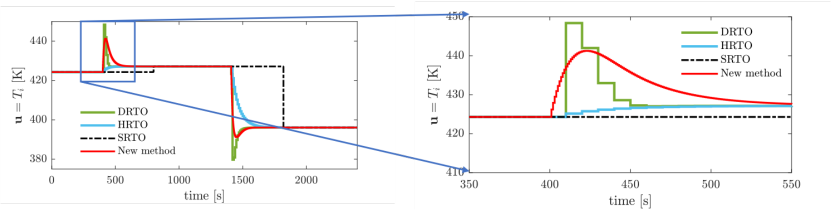
- **Step 2** - Set $\dot{x} = 0$ [†]

$$J = (\underbrace{-CA^{-1}B + D}_{J_u})u$$



Krishnamoorthy et al. *Ind. Eng. Chem. Res* (2019)
[†] Garcia & Morari, *AIChE J.* (1980), Bamberger & Isermann, *Automatica* (1978)

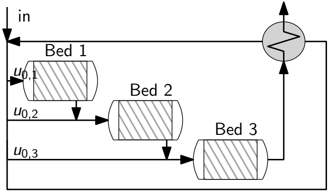
Comparison with other RTO approaches



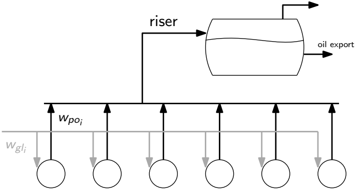
Krishnamoorthy et al. *Ind. Eng. Chem. Res.* (2019)
CSTR process from Economou et. al. (1986)

Other Case Examples

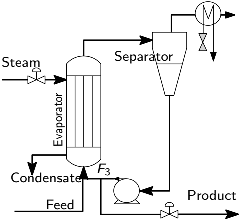
3-bed Ammonia reactor



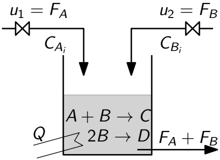
Gas-lift optimization



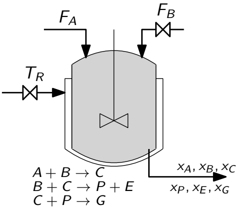
Evaporator process



Isothermal reactor



William-Otto reactor



Krishnamoorthy et al., *IFAC OOGP*, 2018

Krishnamoorthy et al., *Control Engineering Practice*, 2019

Bonnowitz et al.,*Computer Aided Chemical Engineering*, 2018

Krishnamoorthy & Skogestad, *Ind. Eng. Chem. Res*, 2019

Krishnamoorthy et al. *PSE Asia*, 2019

Krishnamoorthy & Skogestad, *Computer-Aided Chemical Engineering*, submitted

Do we always need a model to optimize?

Challenge 1 - Cost of developing the model

Example: Drive from A $\rightarrow$ B in shortest time

- CV: speed $\rightarrow$ 50km/h
- MV: gas pedal

Translate economic objectives into control objectives

What to control?

Control (in this order)

1. Active constraints $\mathbf{g}_{\mathbb{A}} \rightarrow 0$
2. Self-optimizing variable $\mathbf{c} = \mathbf{N}^{\mathsf{T}} \nabla_{\mathbf{u}} J \rightarrow 0$

$\mathbf{N}$ is chosen such that $\mathbf{N}^{\mathsf{T}} \nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}} = 0$

- **Case 1: Fully constrained** - active constraint control
- **Case 2: Fully unconstrained** - control cost gradient to zero ($\mathbf{N} = I$)
- **Case 3: Partially constrained** - active constraint control + control linear gradient combination to zero
- **Case 4: Over constrained** - give up less important constraints

"...the ideal global self-optimizing variable is the gradient J_u."

– Halvorsen & Skogestad (1997)

But gradients are not measured!

Morari et al, *AIChE Journal* (1980)
Skogestad, *J. Proc. Control* (2000)
Krishnamoorthy & Skogestad, *Ind. Eng. Chem. res*, (2019)

Gradient Estimation

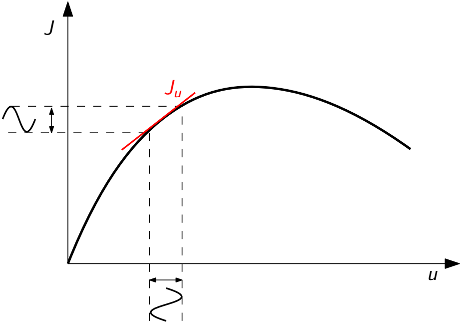
Model-free approaches

- Finite Difference
- Sinusoidal perturbation (Draper & Li, 1951)
- Least sqaures estimation (Hunnekens et al, 2014)
- Linear Identification ARX/ARMAX (Bamberger & Isermann, 1978)
- Nonlinear Identification NARX (Golden & Ydstie, 1989)
- Fixed dynamics (Krishnamoorthy & Skogestad)†
- Kalman filter (Gelbert et al, 2012)
- Polynomial fitted surfaces (Gao & Engell, 2005)
- Guassian Process regression (Ferriera et al, 2018/ Matias & Jaschke 2019)
- Multiple units (Srinivasan, 2007)

Model-based approaches

- Parameter estimation (Chen & Joseph, 1987 / Adetola & Guay 2007)
- Neighboring extremals (Gros et al, 2009)
- Nullspace method (Alstad & Skogestad, 2007)
- Feedback RTO (Krishnamoorthy et al, 2019)

Estimate cost gradient without model - Extremum seeking control



Prohibitively slow convergence!

- Dynamic plant assumed to be static map
- Dither signal $\approx 10\times$ slower than plant dynamics
- Integral gain $\approx 10\times$ slower than perturbations

Draper & Li, *ASME* (1951)
Krstic & Wang, *Automatica* (2000)

ARX-based Extremum seeking control

Identify local linear dynamic model, instead of local linear static model

- **Step 1** - Identify black-box ARX model

$$J(t) + a_1 J(t-1) + \cdots + a_{n_a} J(t-n_a)$$
$$= + b' u(t-1) + \cdots + b'_{n_b} u(t-n_b) + e(t)$$

$$\Rightarrow A_{poly}(q) J = B_{poly}(q) u$$

- **Step 2** - The steady-state part of the locally valid linear model is obtained with $q = 1$

$$J_u = A_{poly}^{-1} B_{poly} \qquad \Leftrightarrow \qquad -CA^{-1}B + D$$

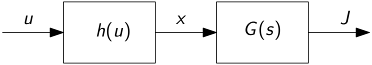
Bamberger & Isermann, *Automatica* (1979)
Garcia & Morari, *AIChE Journal* (1981)

ARX-based extremum seeking control

Robustness Issues with identifying $A_{poly}(q)J = B_{poly}(q)u$

- Too many parameters to estimate : $(n_a + n_b)$
- Neglected dynamics/unmodeled effects

Consider a Hammerstein class of systems



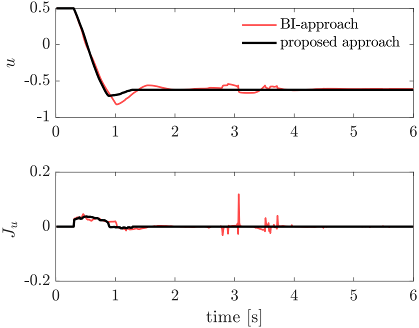
- Fix the nominal plant dynamics, in order to use transient measurements
- Estimate only the steady-state gradient

$$h(u)G_0(q) \approx J_u \underbrace{\left(\frac{b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}}{1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}}\right)}_{\text{fixed nominal dynamics}} u$$

Illustrative Example

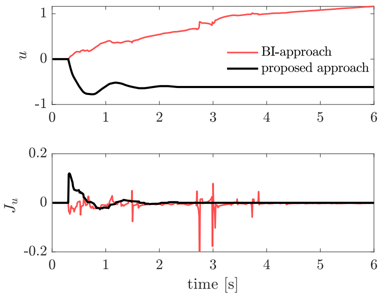
$$h(u) = d + \sum_{i=1}^{3} p_{2i-1}\sin(iu) + p_{2i}\cos(iu)$$

$$G_0(s) = \frac{1}{0.25s + 1}$$

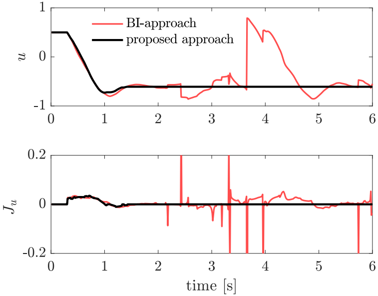


Illustrative Example - Unmodeled dynamics

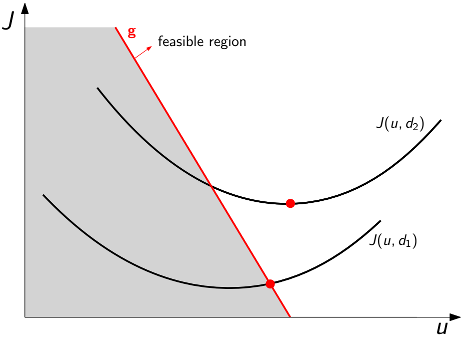
$$G(s) = G_0(s)\frac{-0.05s+1}{0.0001s+1}$$



$$G(s) = G_0(s)e^{-0.01s}$$



Change in active constraint regions



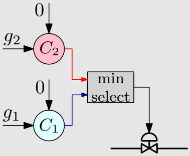
Optimal operation: Need to switch between regions using control system!

Switching between active constraints

- Output-to-output (CV-CV) switching - use selectors
- Input to output (CV-MV) switching - Pair MV that optimally saturates with the CV
- Input-to-input (MV-MV) switching - use split range control / valve position control [1]

Selector block

- Used when one input is used to switch between controlling several outputs.
- Each output has a separate controller and the selectors chooses which output to use.



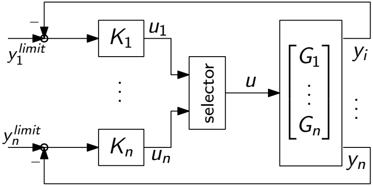
[1] Reyes-Lua et al, *Computer Aided Chemical Engineering*, 2018

CV-CV switching using selectors

For each MV

- at most 1 CV y_0 with setpoint control that may be given up
- n number of CV constraints y_i that may be optimally active

$$u = \min_{i\in[0,n]}(u_i) \quad \text{or} \quad u = \max_{i\in[0.n]}(u_i)$$



When to use min-selector? when to use max-selector? and when is it not feasible?

CV-CV switching using selectors

Introduce logic variable y_i^{lim} for CV constraints

$$y_i^{lim} = \begin{cases} 1, & \text{for max-constraint} \\ -1, & \text{for min-constraint} \end{cases}$$

Theorem

CV-CV switching is feasible only if

$$sgn(G_i)sgn(y_i^{lim}) = sgn(G_j)sgn(y_j^{lim}) \quad \forall i, j \in \{1, \dots, n\}$$

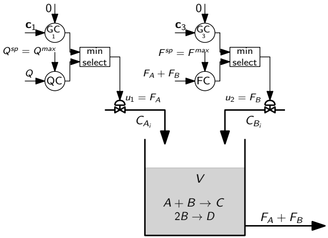
and,

$$sgn(G_i)sgn(y_i^{lim}) = 1 \Rightarrow \text{ min selector}$$
$$sgn(G_i)sgn(y_i^{lim}) = -1 \Rightarrow \text{ max selector}$$

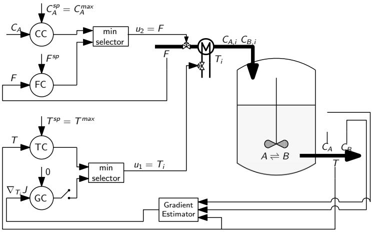
This ensures problem feasibility!

Case examples

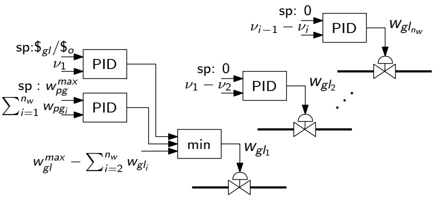
Isothermal CSTR



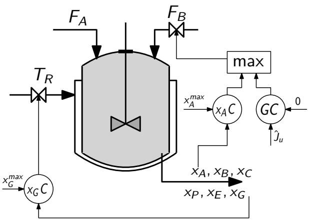
Exothermic Reactor



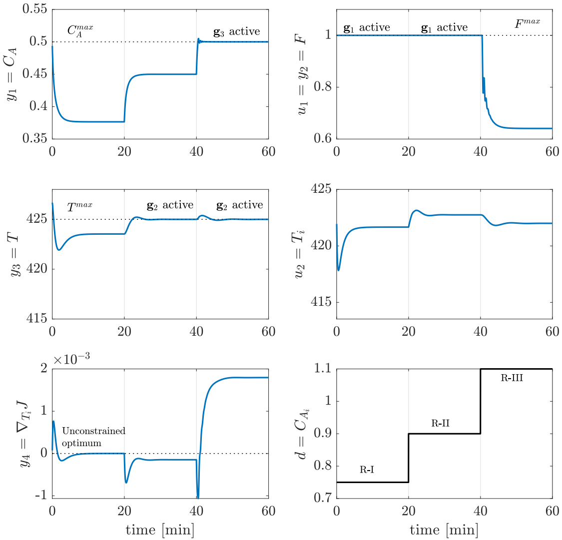
Gas-lift Optimization



William-Otto reactor

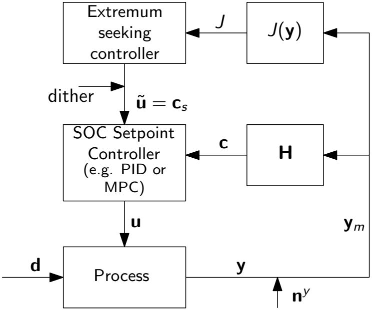


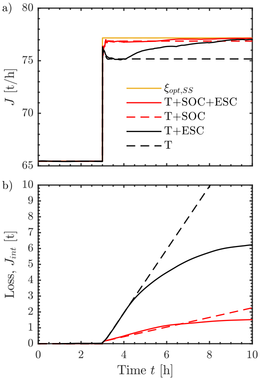
Illustrative example



Hierarchical combination of ESC and SOC

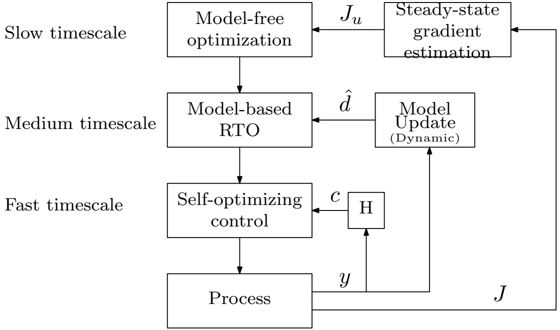
Optimal operation is when the plant gradient $\rightarrow 0$





Krishnamoorthy et al, *AIChE Annual Meeting* (2017)
Straus et al, *J. Proc. Control* (2019)

Hierarchical combination



Outline

1 Research question - Why? and How?

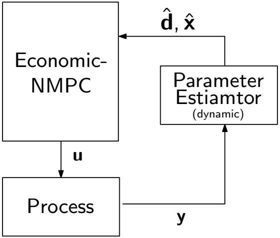
2 Part 1 - Optimal steady-state operation using transient measurements

3 Part 2 - Dynamic optimization - addressing computation time

4 Summary and future outlook

Dynamic RTO / economic MPC

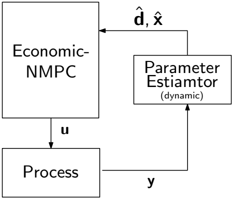
Challenge 4 - Frequent grade changes, which makes steady-state optimization less relevant.



Economic NMPC under uncertainty

Where to handle the uncertainty?

Externally

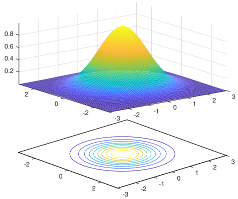


- Use external parameter estimator (observable)

$$\dot{\hat{\mathbf{d}}} = K_e(\mathbf{y} - h(\mathbf{x}, \mathbf{u}, \hat{\mathbf{d}}))$$

- Solve certainty-equivalence MPC with $\hat{\mathbf{d}}$

Internally



- Needs a-priori information (compact set / PDF)

$$\boldsymbol{p} \in \mathcal{P}$$

- Optimize under uncertainty

Economic NMPC under uncertainty

What is a good problem formulation?

- Easy to understand → operator confidence
- Low complexity → Maintenance and service by process engineers
- Not overly conservative → Management approval

Feedback!

"...optimize over a sequence of control laws, as done in Dynamic Programming, rather than over a sequence of control actions."

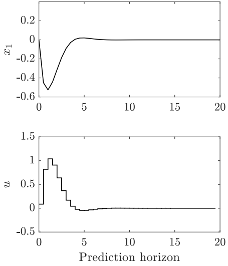
–*Mayne, Automatica (2014)*

Mayne, *Annual Reviews in Control* (2015)

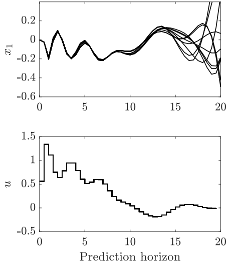
Solution strategies

	Open loop implementation	Closed loop implementation
Open loop optimization	Dynamic optimization	Standard MPC / Receding horizon control
Closed loop optimization	—	Multistage MPC /Feedback min-max MPC

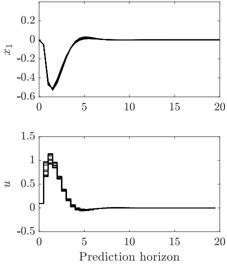
Perfect information



Open-loop optimization

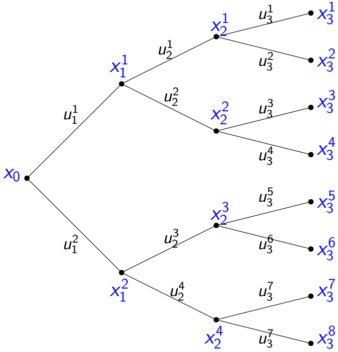


Closed-loop optimization



Multistage scenario MPC

One possible formulation: Multistage scenario MPC / Feedback min-max MPC



- Discrete scenario tree
- Introduces notion of feedback
- Different control trajectories
- Subject to non-anticipativity/causality constraints

Optimization over different control trajectories ≈ Optimization over control policies

(desirable vs. achievable)

Scokaert & Mayne, *IEEE Trans. Autom. Control* (1998)

Lucia et al, *J. Proc. Control* (2013)

Why Multistage MPC?

Certainty equivalence

$$\min_{\mathbf{x}_k,\mathbf{u}_k} \sum_{k=0}^{N-1} \ell(\mathbf{x}_k,\mathbf{u}_k)$$

s.t.

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{f}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{p}_k) && \forall k \in \mathcal{K}\\
\mathbf{g}(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{p}_k) &\leq 0 && \forall k \in \mathcal{K}\\
\mathbf{x}_0 &= \hat{\boldsymbol{x}}_t &&\\
\mathbf{x}_k \in \mathcal{X}, &\quad \mathbf{u}_k \in \mathcal{U} && \forall k \in \mathcal{K}
\end{aligned}$$

Multistage MPC

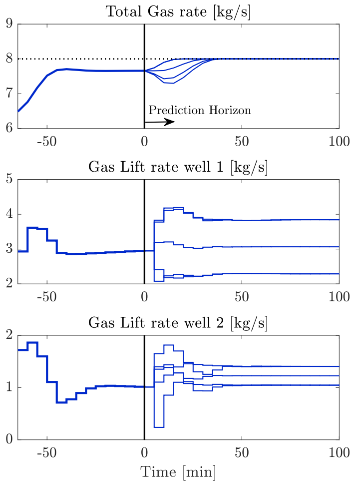
$$\min_{\mathbf{x}_{k,j},\mathbf{u}_{k,j}} \sum_{j=1}^{S} \omega_j \sum_{k=0}^{N-1} \ell(\mathbf{x}_{k,j},\mathbf{u}_{k,j})$$

s.t

$$\begin{aligned}
\mathbf{x}_{k+1,j} &= \mathbf{f}(\mathbf{x}_{k,j},\mathbf{u}_{k,j},\boldsymbol{p}_{k,j}) && \forall j \in \mathcal{S}, \forall k \in \mathcal{K}\\
\mathbf{g}(\mathbf{x}_{k,j},\mathbf{u}_{k,j},\boldsymbol{p}_{k,j}) &\leq 0 && \forall j \in \mathcal{S}, \forall k \in \mathcal{K}\\
\mathbf{x}_{0,j} &= \hat{\boldsymbol{x}}_t && \forall j \in \mathcal{S}\\
\mathbf{x}_{k,j} \in \mathcal{X}, &\quad \mathbf{u}_{k,j} \in \mathcal{U} && \forall j \in \mathcal{S}, \forall k \in \mathcal{K}\\
\sum_{j=1}^{S} \bar{\mathbf{E}}_j \mathbf{u}_j &= \mathbf{0} &&
\end{aligned}$$

Easier to implement and maintain

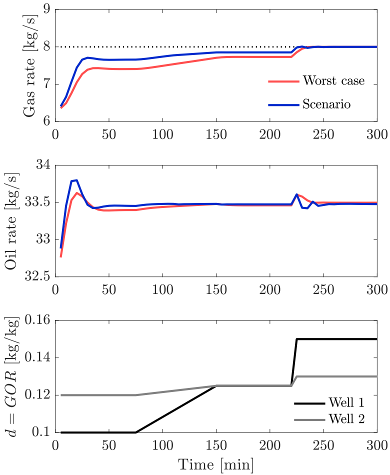
Why Multistage MPC?



Easier to understand the control actions

Krishnamoorthy et al, *Processes* (2016)

Why Multistage MPC?



Less conservative than worst-case MPC

Krishnamoorthy et al, *Processes* (2016)

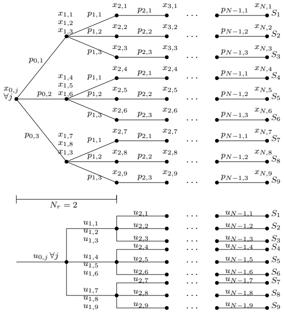
Multistage MPC - Major Drawback

Problem size explodes exponentially $S = M^N$

$$\min_{\mathbf{x}_{k,j},\mathbf{u}_{k,j}} \sum_{j=1}^{S} \omega_j \sum_{k=0}^{N-1} \ell(\mathbf{x}_{k,j},\mathbf{u}_{k,j})$$

s.t

$$\begin{aligned}
\mathbf{x}_{k+1,j} &= \mathbf{f}(\mathbf{x}_{k,j},\mathbf{u}_{k,j},\boldsymbol{p}_{k,j}) && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{g}(\mathbf{x}_{k,j},\mathbf{u}_{k,j},\boldsymbol{p}_{k,j}) &\leq 0 && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{x}_{0,j} &= \hat{\boldsymbol{x}}_t && \forall j \in \mathcal{S} \\
\mathbf{x}_{k,j} \in \mathcal{X}, &\quad \mathbf{u}_{k,j} \in \mathcal{U} && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\sum_{j=1}^{S} \bar{\mathbf{E}}_j \mathbf{u}_j &= \mathbf{0}
\end{aligned}$$

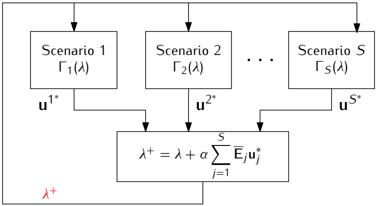


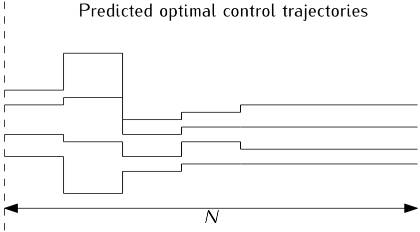
Non-anticipativity constraints enable closed-loop implementation

Lucia et al, *J. Proc. Control* (2013)

Scenario Decomposition using Dual Decomposition

- Relax Non-anticipativity constraints
- Non-anticipativity constraints are may not be feasible - (Duality gap) !

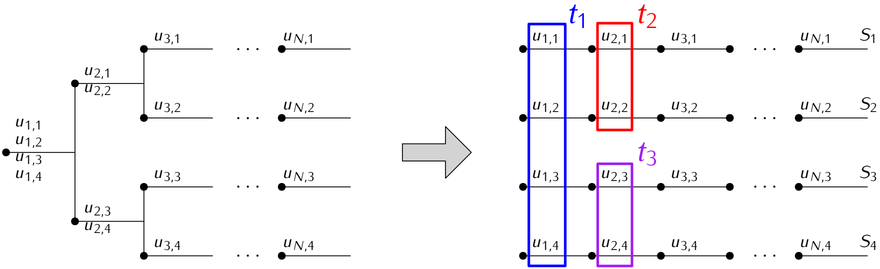




Closed-loop implementation fails!!

Marti et al, *Comput. & Chem. Eng* (2015)

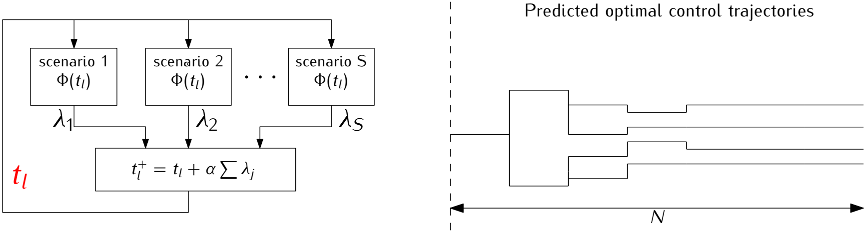
Scenario decomposition using Primal decomposition



Krishnamoorthy et al, *IFAC ADCHEM* (2018)
Krishnamoorthy et al, *J. Proc. Control* (2019)

Scenario Decomposition using Primal Decomposition

Non-anticipativity constraints are always feasible !



Closed-loop implementation OK!!

"Approximate solution now is better than accurate solution tomorrow!"

Krishnamoorthy et al, *J. Proc. Control* (2019)

Scenario Decomposition using Parametric Sensitivities

Reduce the number of NLPs needed to solve the distributed scenario MPC problem

$$\mathbf{w}_0^* = \arg\min_{\mathbf{w}} \quad \Phi(\mathbf{w}, \mathbf{p}_0)$$

s.t.

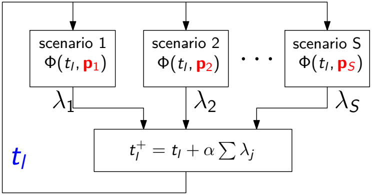
$$\mathbf{g}(\mathbf{w}, \mathbf{p}_0) \leq 0$$

$$\mathbf{w}_0^* + \Delta\mathbf{w}^* = \arg\min_{\mathbf{w}} \quad \Phi(\mathbf{w}, \mathbf{p}_0 + \Delta\mathbf{p})$$

s.t.

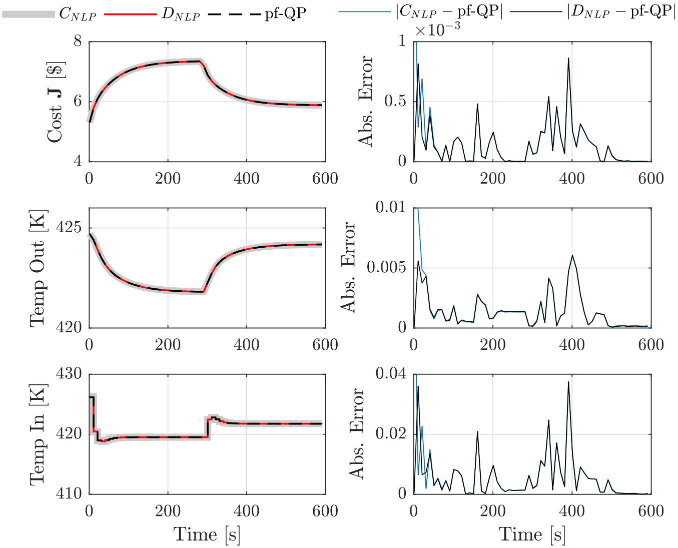
$$\mathbf{g}(\mathbf{w}, \mathbf{p}_0 + \Delta\mathbf{p}) \leq 0$$

Compute $\Delta\mathbf{w}^*$ using path-following predictor-corrector QP.



Krishnamoorthy et al, *IEEE Contr. Syst. Lett.* (2018)

Illustrative example



Outline

1 Research question - Why? and How?

2 Part 1 - Optimal steady-state operation using transient measurements

3 Part 2 - Dynamic optimization - addressing computation time

4 Summary and future outlook

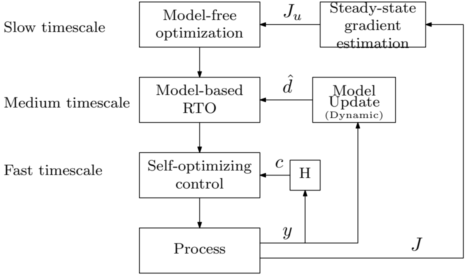
Future Outlook

Understand the uncertainty - Handle only those uncertainties that have an impact.

In the context of multistage NMPC,

- How to select the scenarios - use of data analytic tools
- Shrink the uncertainty set using data - Online scenario tree update

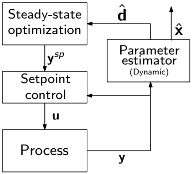
Different methods work in different timescales, can handle different kinds of uncertainty, and have different advantages and disadvantages!



Krishnamoorthy et al. *IFAC NMPC* (2018)
Krishnamoorthy et al. *ECC* (2019)

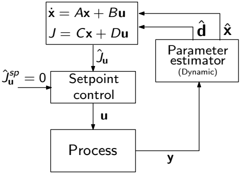
Summary - The Big Picture

Hybrid RTO



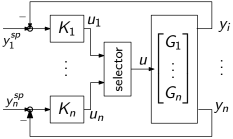
Challenge 2 & 5

Feedback RTO



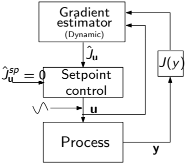
Challenge 2,3 & 5

Optimization using simple controllers



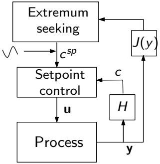
Challenge 1,2,3 & 5

Extremum seeking using transient measurements



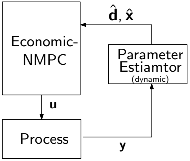
Challenge 1,2,3 & 5

Extremum seeking + Self-optimizing control



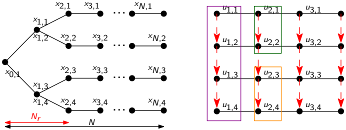
Challenge 1,2,3 & 5

economic MPC



Challenge 3,4& 5

Distributed multistage MPC using primal decomposition



Challenge 3, 4 & 5

Thank you!

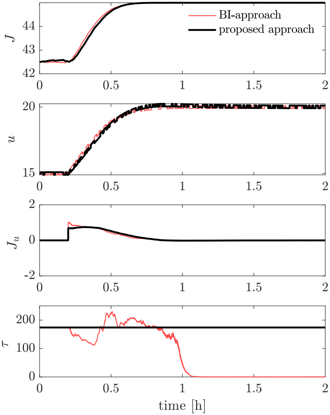
ARX-based extremum seeking control

$$G(s) = \frac{1}{(\tau s + 1)}$$

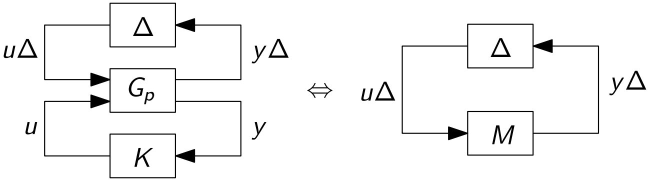
ARX model with $n_a = 1$ and $n_b = 1$

$$J(t) = -a_1 J(t-1) + b_1 u(t-1)$$

$$J_u = \frac{b_1}{1 + a_1}, \qquad \tau = \frac{-T_s}{ln(-a_1)}$$



Dynamic Extremum seeking control



$$P := \begin{bmatrix} 0 & 1 \\ w_I G_0 & G_0 \end{bmatrix}, \qquad M := w_I G_0 K (1 + G_0 K)^{-1}$$

$$w_I(j\omega) = \left| \frac{G_p(j\omega) - G_0(j\omega)}{G_0(j\omega)} \right| = \left| 1 - \frac{(-5j\omega + 1)}{(10j\omega + 1)} \right|$$

Dynamic Extremum seeking control - SISO case

SISO ARX model:

$$J(t) + a_1 J(t-1) + \cdots + a_{n_a} J(t-n_a) = J_u \left[b_1 u(t-1) + b_2 u(t-2) + \cdots + b_{n_b} u(t-n_b) \right]$$

$$\hat{\theta} = arg \ \min_{\theta} \|\psi - \Phi\theta\|_2^2$$

$$\theta = J_u$$

$$\psi = \begin{bmatrix} J(N) + a_1 J(N-1) + \cdots + a_{n_a} J(N-n_a) \\ J(N-1) + a_1 J(N-2) + \cdots + a_{n_a} J(N-1-n_a) \\ \vdots \\ J(n+1) + a_1 J(n+2) + \cdots + a_{n_a} J(n+1-n_a) \end{bmatrix}$$

$$\Phi = \begin{bmatrix} b_1 u(N-1) + \cdots + b_{n_b} u(N-n_b) \\ b_1 u(N-2) + \cdots + b_{n_b} u(N-1-n_b) \\ \vdots \\ b_1 u(n) + \cdots + b_{n_b} u(n-1-n_b) \end{bmatrix}$$

Dynamic Extremum seeking control - Multivariable case

MISO ARX model with n inputs

$$J(t) + a_1 J(t-1) + \cdots + a_{n_a} J(t-n_a) = \mathbf{B}_1 \mathbf{u}(t-1) + \cdots + \mathbf{B}_{n_b} \mathbf{u}(t-n_b)$$

with $\mathbf{B}_i \in \mathbb{R}^{1\times n}$

$$\begin{aligned} J(t) &+ a_1 J(t-1) + \cdots + a_{n_a} J(t-n_a) \\ &= \sum_{i=1}^{n} J_{u_i} \left[b_{1,i} u_i(t-1) + b_{2,i} u_i(t-2) + \cdots + b_{n_b,i} u_i(t-n_b) \right] \end{aligned}$$

$$\hat{\theta} = arg \ \min_{\theta} \|\psi - \Phi\theta\|_2^2$$

$$\theta = [J_{u_1}, J_{u_2}, \ldots J_{u_n}]^T, \ \psi \text{ - same as before}$$

$$\Phi = [\Phi_1, \Phi_2, \ldots \Phi_n]$$

Zhu, *Multivariable System Identification for Process Control*.

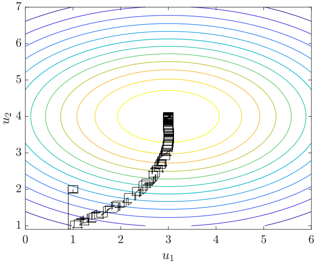
Lyung, *System Identification: Theory for the user*.

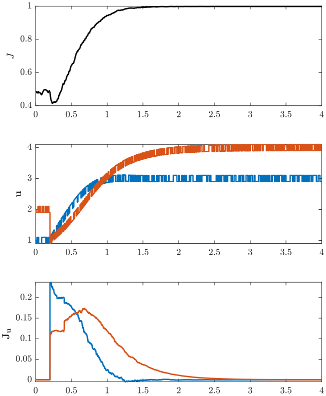
Dynamic Extremum seeking control - Multivariable case

Illustrative example - 2D Gaussian function

$$J(u_1, u_2) = exp\left(-\left(\frac{(u_1-3)^2}{8} + \frac{(u_2-4)^2}{18}\right)\right)$$

$$G(s) = \frac{1}{120s+1}$$





Linear gradient combination satisfies KKT conditions

For a steady-state optimization problem with $n_a < n_u$ active constraints $\mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d})$

$$\begin{aligned} \nabla_{\mathbf{u}} \mathcal{L}(\mathbf{u}, \mathbf{d}) =& \nabla_{\mathbf{u}} J(\mathbf{u}, \mathbf{d}) + \lambda_{\mathbb{A}}^T \nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d}) = 0 \\ \Rightarrow& \nabla_{\mathbf{u}} J(\mathbf{u}, \mathbf{d}) = -\lambda_{\mathbb{A}}^T \nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d}) \end{aligned}$$

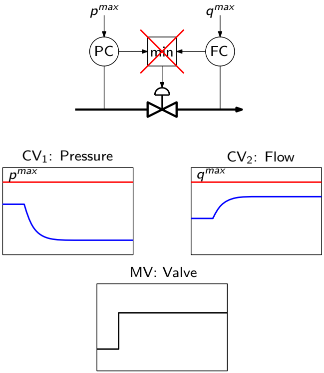
Pre-multiplying by $\mathbf{N}^\mathsf{T}$ gives

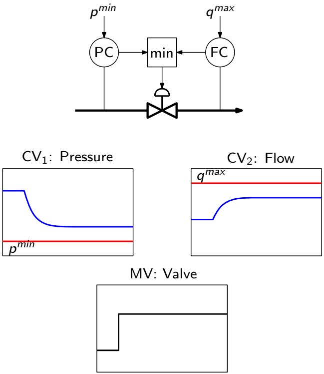
$$\mathbf{N}^\mathsf{T} \nabla_{\mathbf{u}} J(\mathbf{u}, \mathbf{d}) = -\mathbf{N}^\mathsf{T} \nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d})^\mathsf{T} \lambda$$

Since $\mathbf{N}^\mathsf{T} \nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d})^\mathsf{T} = 0$, $\Rightarrow$ $\mathbf{N}^\mathsf{T} \nabla_{\mathbf{u}} J(\mathbf{u}, \mathbf{d}) = 0$

Since $n_a < n_u$, $\nabla_{\mathbf{u}} \mathbf{g}_{\mathbb{A}}(\mathbf{u}, \mathbf{d})$ has full row rank and $\mathbf{N}$ is well defined.

CV-CV switching using selectors - Illustrative Example





When a CV constraint is active, all other CVs must be within its limit. To remain feasible,

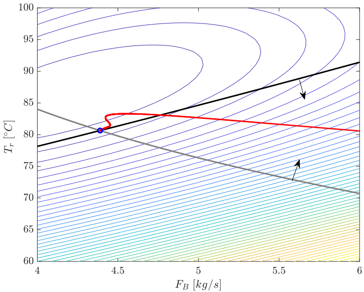
- when the MV changes, all the CVs must change in the same direction relative to their constraints
- if $sgn(G_i)sgn(y_i^{lim}) = 1$, then increasing u moves output y_i closer to its constraint.

$$u = \min(u_i) \Rightarrow \text{feasibility}$$

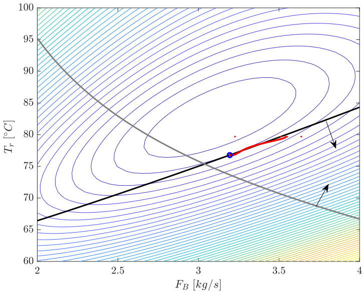
CV-CV switching: William-Otto reactor

2 CV constraints; $x_G \leq 0.08$kg/kg and $x_A \leq 0.12$kg/kg

$F_A > 1.5$

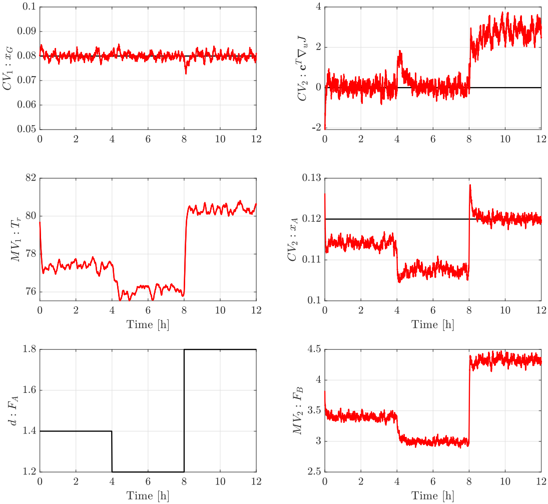


$F_A < 1.5$



CV-CV switching: William-Otto reactor

Use max-selector to switch between x_A^{max} and c



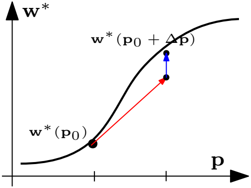
Parametric Optimization

$$\mathbf{w}_0^* = \arg\min_{\mathbf{w}} \quad \Phi(\mathbf{w}, \mathbf{p}_0)$$
s.t.
$$\mathbf{g}(\mathbf{w}, \mathbf{p}_0) \leq 0$$

$$\mathbf{w}_0^* + \Delta\mathbf{w}^* = \arg\min_{\mathbf{w}} \quad \Phi(\mathbf{w}, \mathbf{p}_0 + \Delta\mathbf{p})$$
s.t.
$$\mathbf{g}(\mathbf{w}, \mathbf{p}_0 + \Delta\mathbf{p}) \leq 0$$

- **Pure Predictor QP**

$$\Delta\mathbf{w}^* = \arg\min_{\Delta\mathbf{w}} \frac{1}{2}\Delta\mathbf{w}^T \nabla^2_{\mathbf{ww}}\mathcal{L}\Delta\mathbf{w} + \Delta\mathbf{w}^T \nabla_{\mathbf{wp}}\mathcal{L}\Delta\mathbf{p}$$
s.t
$$\nabla_{\mathbf{w}}\mathbf{g}^T\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^T\Delta\mathbf{p} \leq 0$$

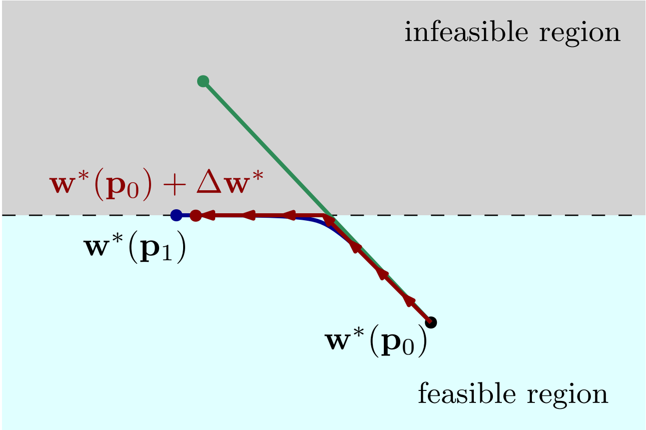


-**Predictor Corrector QP**

$$\Delta\mathbf{w}^* = \arg\min_{\Delta\mathbf{w}} \frac{1}{2}\Delta\mathbf{w}^T \nabla^2_{\mathbf{ww}}\mathcal{L}\Delta\mathbf{w} + \Delta\mathbf{w}^T \nabla_{\mathbf{wp}}\mathcal{L}\Delta\mathbf{p} + \nabla_{\mathbf{w}}\Phi^T\Delta\mathbf{w}$$
s.t
$$\nabla_{\mathbf{w}}\mathbf{g}^T\Delta\mathbf{w} + \nabla_{\mathbf{p}}\mathbf{g}^T\Delta\mathbf{p} \leq 0$$

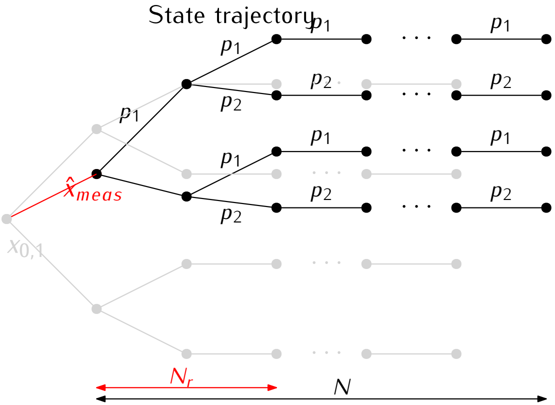
Suwartadi et al, *Processes* (2017)

Path-following predictor corrector QP



Suwartadi et al, *Processes* (2017)

Online update of the scenario tree



Feedback versus open-loop optimization

feedback optimization $\equiv$ open-loop optimization with closed-loop implementation

$$x_{k+1} = x_k + u_k + w_k$$

Open-loop optimization with $w_k = 0$

$$\mathbf{u}^0(x_0) = [-(8/13)x_0, -(3/13)x_0, -(1/13)x_0]$$
$$\mathbf{x}^0(x_0) = [x_0, (5/13)x_0, (2/13)x_0, (1/13)x_0]$$

Dynamic Programming

$$\boldsymbol{\mu} := (\mu_0(\cdot), \mu_1(\cdot), \mu_2(\cdot))$$

$$\mu_0 = -(8/13)x_0 \Rightarrow x^0(0) = x_0 \qquad u^0(0) = -(8/13)x_0$$
$$\mu_1 = -(3/5)x_1 \Rightarrow x^0(1) = (5/13)x_0 \qquad u^0(1) = -(3/13)x_0$$
$$\mu_2 = -(1/2)x_2 \Rightarrow x^0(2) = (2/13)x_0 \qquad u^0(2) = -(1/13)x_0$$
$$\Rightarrow x^0(3) = (1/13)x_0$$

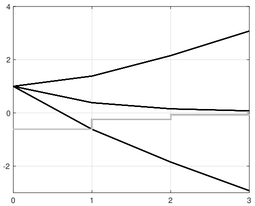
Section3.1.2 - Rawlings, Mayne, Diehl, *Model Predictive Control*, 2nd ed.

Feedback versus open-loop optimization

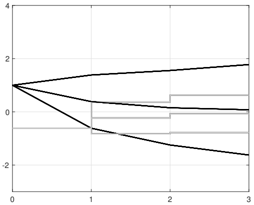
feedback optimization ≈ multistage optimization with closed-loop implementation

$$x_{k+1} = x_k + u_k + w_k \qquad w_k \in [-1, 1]$$

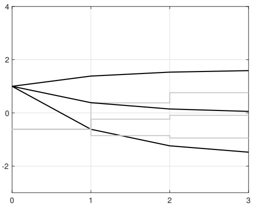
Open-loop



Feedback - DP



Feedback - multistage

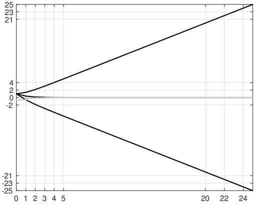


Section3.1.2 - Rawlings, Mayne, Diehl, *Model Predictive Control*, 2nd ed.

Feedback versus open-loop optimization

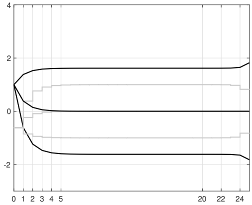
feedback optimization $\approx$ multistage optimization with closed-loop implementation

Open-loop



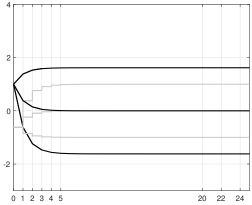
$|x_{w=-1}(N) - x_{w=1}(N)| = 2N$

Feedback - DP



$|x_{w=-1}(N) - x_{w=1}(N)| \to 3.24$
$N \to \infty$

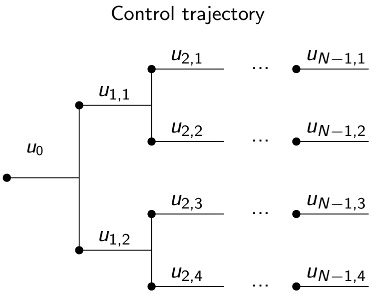
Feedback - multistage



$|x_{w=-1}(N) - x_{w=1}(N)| \to 3.24$
$N \to \infty$

Section3.1.2 - Rawlings, Mayne, Diehl, *Model Predictive Control*, 2nd ed.

Multistage MPC approximation of DP



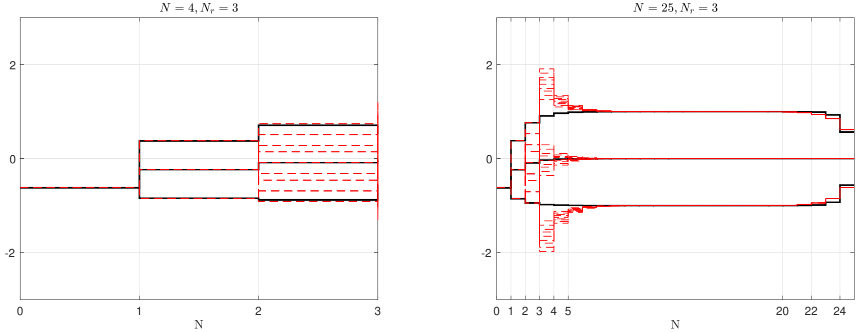
Ideal: Optimize over control policies

$$\boldsymbol{\mu} = (\mu_0, \mu_1, \mu_2 \ldots, \mu_{N-1})$$

Sampled control policy

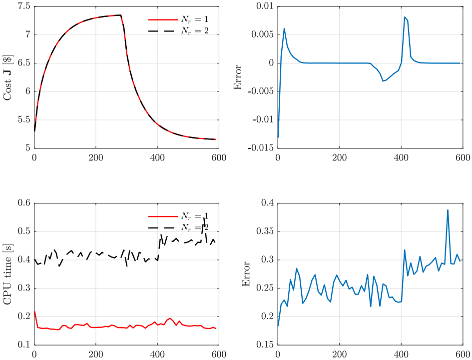
$$\begin{aligned}
\mu_0 &= \{u_0\} \\
\mu_1 &= \{u_{1,1}, u_{1,2}\} \\
\mu_2 &= \{u_{2,1}, u_{2,2}, u_{2,3}, u_{2,4}\} \\
&\vdots \\
\mu_{N-1} &= \{u_{N-1,1}, u_{N-1,2}, u_{N-1,3}, u_{N-1,4}\}
\end{aligned}$$

Multistage MPC approximation of DP



Price of robust horizon - what am I getting for the additional CPU time?

Simulation results from Chapter 9.



Partially deterministic approximation (Berstekas, 2019)

"When the problem is stochastic, one may consider an approximation to the ℓ-step lookahead, which is based on deterministic computations. This is a hybrid, partially deterministic approach, whereby at state x_k we allow for a stochastic disturbance at the current stage, but fix the future disturbances... up to the end of the lookahead horizon, to some typical values."

Stability properties of AHeNMPC

- Lyapunov stability framework

$$\ell(\mathbf{x}_0, \mathbf{u}_0) + \Delta\mathbf{x}_{\tilde{N}}^{\mathsf{T}} P \Delta\mathbf{x}_{\tilde{N}} - \ell(\mathbf{x}_{\tilde{N}}, \mathbf{u}_{\tilde{N}}) - \mathbf{f}(\mathbf{x}_{\tilde{N}}, \mathbf{u}_{\tilde{N}})^{\mathsf{T}} P \mathbf{f}(\mathbf{x}_{\tilde{N}}, \mathbf{u}_{\tilde{N}}) \leq 0$$

- For descent property, ℓ must be strictly dissipative $\Rightarrow$ $(\mathbf{x}_f, \mathbf{u}_f)$ unique global minima[2]
- if stage cost and dynamic model form strong duality $\Rightarrow$ dissipativity [3]
- Steady-state optimization problem has a strongly convex Lagrange function $\Rightarrow$ strong duality[4]
- Regularization of the stage cost $\Rightarrow$ strong convexity

$$\psi_{reg}(\mathbf{x}, \mathbf{u}) := \ell(\mathbf{x}, \mathbf{u}) + \frac{1}{2}\|(\mathbf{x} - \mathbf{x}_f, \mathbf{u} - \mathbf{u}_f)\|_{\hat{Q}}^2 + \lambda^{\mathsf{T}}(\mathbf{x} - \mathbf{f}(\mathbf{x}, \mathbf{u}))$$

- With $\hat{Q}$: $eig(\nabla^2\psi + \hat{Q}) > 0 \forall (x, u) \in \mathcal{X} \times \mathcal{U}$

[2] Faulwasser et al *Foundations and Trends in System and Control* (2018)
[3] Angeli et al, *IEEE Trans. Autom. Control*(2011)
[4] Huang et al, *J. Proc. Control* (2018)

Scenario-based MPC

OL Scenario-based MPC / Sampled average approximation/ Randomized MPC

$$\min_{\mathbf{x}_{k,j},\mathbf{u}_k} \sum_{j=1}^{S} \omega_j \sum_{k=0}^{N-1} \ell(\mathbf{x}_{k,j}, \mathbf{u}_k)$$

s.t

$$\begin{aligned}
\mathbf{x}_{k+1,j} &= \mathbf{f}(\mathbf{x}_{k,j}, \mathbf{u}_k, \boldsymbol{p}_j) && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{g}(\mathbf{x}_{k,j}, \mathbf{u}_k, \boldsymbol{p}_j) &\leq 0 && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{x}_{0,j} &= \hat{\boldsymbol{x}}_t && \forall j \in \mathcal{S} \\
\mathbf{x}_{k,j} \in \mathcal{X}, &\quad \mathbf{u}_k \in \mathcal{U} && \forall j \in \mathcal{S}, \forall k \in \mathcal{K}
\end{aligned}$$

Multistage scenario-based MPC / Feedback min-max MPC

$$\min_{\mathbf{x}_{k,j},\mathbf{u}_{k,j}} \sum_{j=1}^{S} \omega_j \sum_{k=0}^{N-1} \ell(\mathbf{x}_{k,j}, \mathbf{u}_{k,j})$$

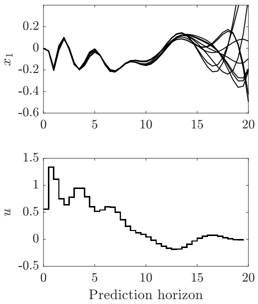
s.t

$$\begin{aligned}
\mathbf{x}_{k+1,j} &= \mathbf{f}(\mathbf{x}_{k,j}, \mathbf{u}_{k,j}, \boldsymbol{p}_{k,j}) && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{g}(\mathbf{x}_{k,j}, \mathbf{u}_{k,j}, \boldsymbol{p}_{k,j}) &\leq 0 && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\mathbf{x}_{0,j} &= \hat{\boldsymbol{x}}_t && \forall j \in \mathcal{S} \\
\mathbf{x}_{k,j} \in \mathcal{X}, &\quad \mathbf{u}_{k,j} \in \mathcal{U} && \forall j \in \mathcal{S}, \forall k \in \mathcal{K} \\
\sum_{j=1}^{S} \bar{\mathbf{E}}_j \mathbf{u}_j &= \mathbf{0}
\end{aligned}$$

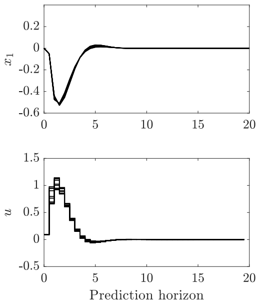
Schildbach et al, *Automatica* 2014
Fagiano et al, *IEEE Trans. Autom. Control* 2014
Campi & Garatti, *IEEE Trans. Autom. Control* (2012)

Scenario-based MPC

OL Scenario-based MPC / Sampled average approximation/ Randomized MPC



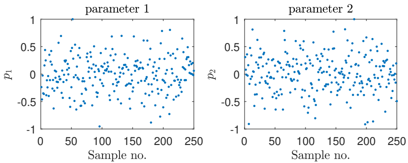
Multistage scenario-based MPC / Feedback min-max MPC

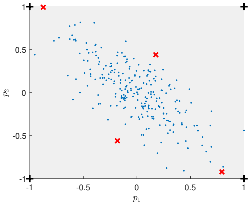


Data analytic tools to select the discrete scenarios

How to select the different scenarios?

- Aim: Seek robustness to variability
- Use data analytic tools to explain the variability





Uncover hidden data structures to select scenarios judiciously !

Krishnamoorthy et al, *IFAC NMPC* (2018)

Robust-adaptive framework - updating the uncertainty set

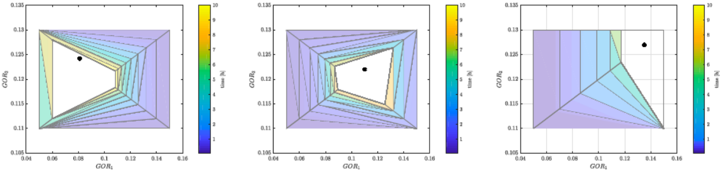
For time-invariant parameters, shrink the uncertainty set by updating the scenarios based on recursive Bayesian probability.

Recursive Bayes Theorem

Conditional probability for the j^{th} scenario being the true realization,

$$P_{k,j} = \frac{e^{-0.5\epsilon_{k,j}^T K \epsilon_{k,j}} P_{k-1,j}}{\sum_{m=1}^{S} e^{-0.5\epsilon_{k,m}^T K \epsilon_{k,m}} P_{k-1,m}}$$

$$W_{k,j} = \frac{P_{k,j}}{\sum_{m=1}^{S} P_{k,m}}$$



Krishnamoorthy et al, *ECC* (2019)

Real time optimization