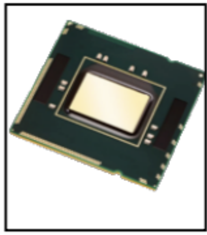
# Lecture 9: Closed loop dynamic optimization – Model Predictive Control (MPC)

- Model Predictive Control (MPC)
- Open loop vs closed loop dynamic optimization
- Feasibility
- Stability

Reference: B&H Ch. 3.3-4.2.2

# MPC: Applications



Computer control — ns

$\mu$s — Power systems

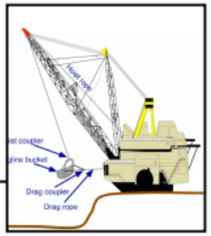
Traction control — ms

Seconds — Buildings

Refineries — Minutes

Hours — Nurse rostering


Train scheduling — Days

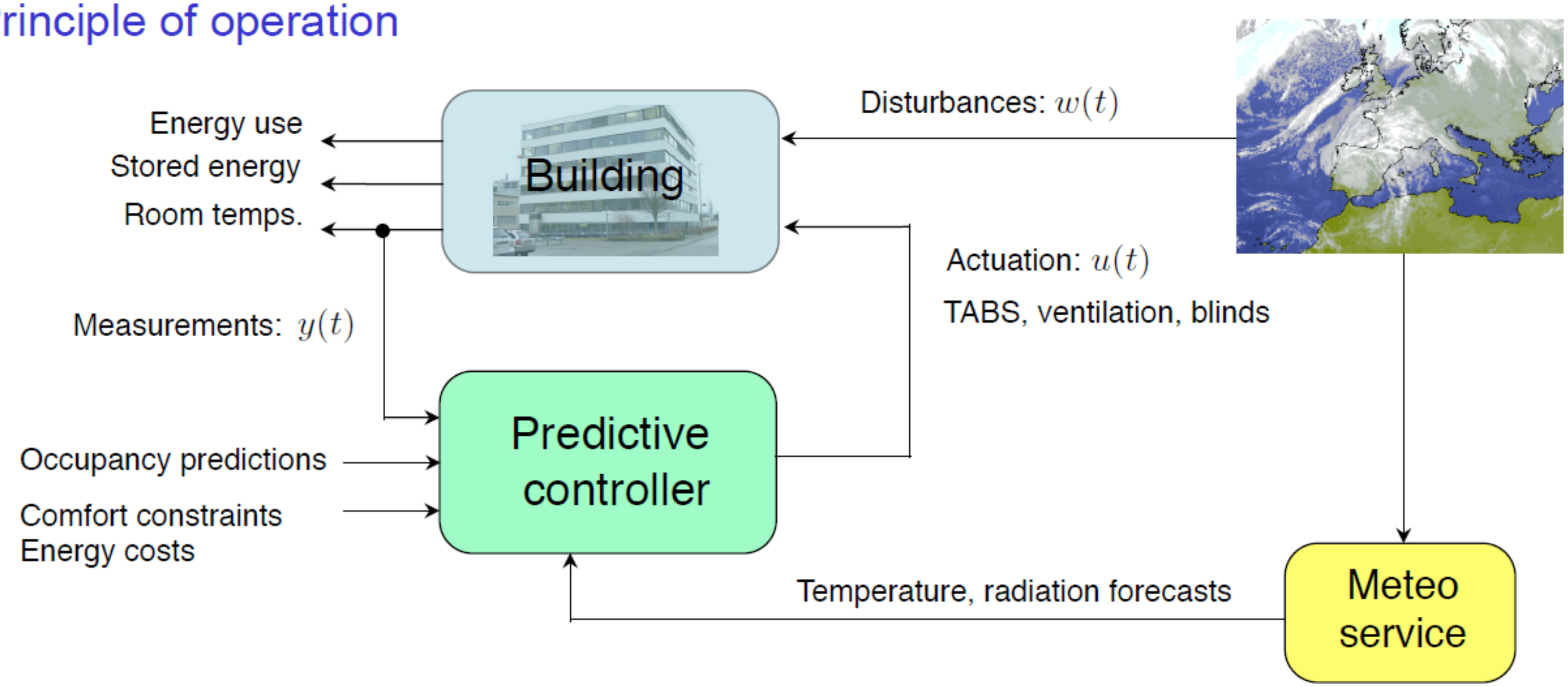



Weeks — Production planning

Zeilinger, Jones, Borrelli, Morari

# Model predictive control (MPC)

## Principle of operation




$$\text{Predicted Cost} \;=\; \underset{u(t)}{\text{minimize}} \;\; \text{Expected}\left(\sum_{t}^{t+N} \text{energy cost(t)}\right)$$ ← Minimize the predicted energy cost

$$\text{subject to} \quad u(t) \in \mathcal{U}$$ ← Actuation within limits

$$x(t) \in \mathcal{X}$$ ← Predicted temperatures within limits

$$x(t+1) = f(x(t), u(t), w(t))$$ ← Predicted dynamics of the building

From ETH

# Last time: Dynamic open-loop optimization (with linear state-space model)

$$\min_{z \in \mathbb{R}^n} f(z) = \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^\top Q_{t+1} x_{t+1} + d_{xt+1} x_{t+1} + \frac{1}{2} u_t^\top R_t u_t + d_{ut} u_t + \frac{1}{2} \Delta u_t^\top S \Delta u_t$$

subject to

$$\begin{aligned}
x_{t+1} &= A_t x_t + B_t u_t, \quad t = \{0, \dots, N-1\} \\
x^{\text{low}} &\leq x_t \leq x^{\text{high}}, \quad t = \{1, \dots, N\} \\
u^{\text{low}} &\leq u_t \leq u^{\text{high}}, \quad t = \{0, \dots, N-1\} \\
-\Delta u^{\text{high}} &\leq \Delta u_t \leq \Delta u^{\text{high}}, \quad t = \{0, \dots, N-1\} \\
Q_t &\succeq 0 \quad t = \{1, \dots, N\} \\
R_t &\succeq 0 \quad t = \{0, \dots, N-1\}
\end{aligned}$$

where

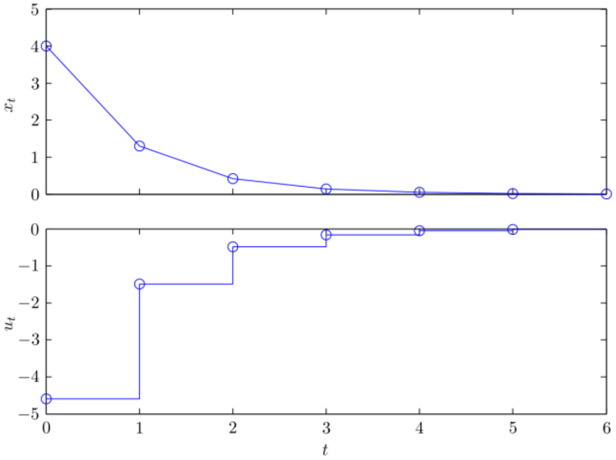
$$\begin{aligned}
x_0 &\text{ and } u_{-1} \text{ is given} \\
\Delta u_t &:= u_t - u_{t-1} \\
z^\top &:= (u_0^\top, x_1^\top, \dots, u_{N-1}^\top, x_N^\top) \\
n &= N \cdot (n_x + n_u)
\end{aligned}$$

# The significance of weigths

$$\min \sum_{t=0}^{5} q\, x_{t+1}^2 + r\, u_t^2$$

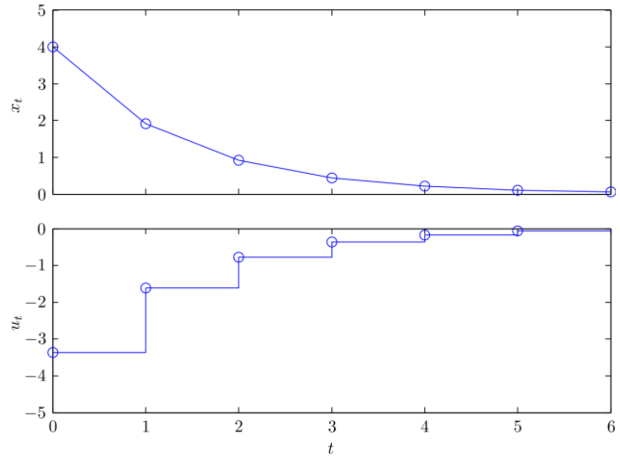
$$\text{s.t.} \qquad x_{t+1} = 0.9 x_t + 0.5 u_t, \quad t = 0, \dots, 4$$

$q = 5,\ r = 1$



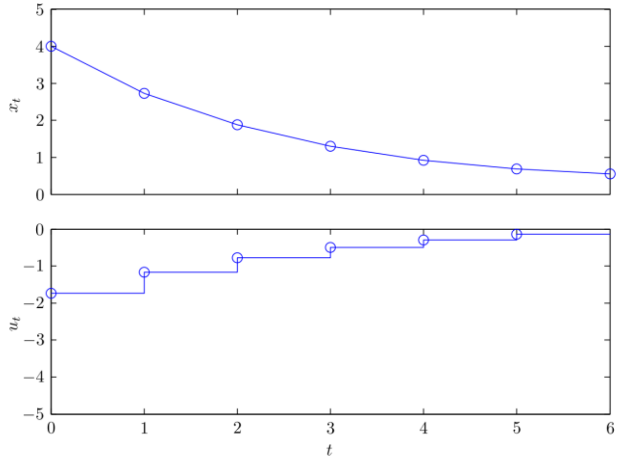

$$\sum_{t=0}^{N-1} x_{t+1}^2 = 1.9, \qquad \sum_{t=0}^{N-1} u_t^2 = 23.6$$

$q = 2,\ r = 1$




$$\sum_{t=0}^{N-1} x_{t+1}^2 = 4.8, \qquad \sum_{t=0}^{N-1} u_t^2 = 14.7$$

$q = 1,\ r = 2$




$$\sum_{t=0}^{N-1} x_{t+1}^2 = 14.3, \qquad \sum_{t=0}^{N-1} u_t^2 = 5.3$$

# Linear quadratic control: Dynamic optimization without constraints

$$\min_{z} \sum_{t=0}^{N-1} x_{t+1}^\top Q x_{t+1} + u_t^\top R u_t$$
$$\text{s.t.} \quad x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \ldots, N-1$$
$$z = (u_0, x_1, u_1, \ldots, u_{N-1}, x_N)^\top$$

Three approaches for solution

- Batch approach v1, “full space” – solve as QP
- Batch approach v2, “reduced space” – solve as QP
- Recursive approach – solve as linear state feedback

# Linear Quadratic Control
# Batch approach v1, "Full space" QP

$$\min_{z} \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^{\top} Q x_{t+1} + \frac{1}{2} u_t^{\top} R u_t$$

$$\text{s.t.} \quad x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \dots, N-1$$

$$z = (u_0, x_1, u_1, \dots, u_{N-1}, x_N)^{\top}$$

- Formulate with model as equality constraints, all inputs and states as optimization variables: EQP!

$$\min_{z} \quad \frac{1}{2} z^{\top} \begin{pmatrix} R & & & & \\ & Q & & & \\ & & R & & \\ & & & \ddots & \\ & & & & Q \end{pmatrix} z$$

$$\text{s.t.} \begin{pmatrix} -B & I & & & & & & \\ & -A & -B & I & & & & \\ & & & -A & -B & I & & \\ & & & & \ddots & \ddots & & \\ & & & & & -A & -B & I \end{pmatrix} z = \begin{pmatrix} A x_0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$z = (u_0, x_1, u_1, \dots, u_{N-1}, x_N)^{\top}$$

# Linear Quadratic Control
# Batch approach v2, “Reduced space” QP

$$\min_z \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^\top Q x_{t+1} + \frac{1}{2} u_t^\top R u_t$$
$$\text{s.t.} \quad x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \dots, N-1$$
$$z = (u_0, x_1, u_1, \dots, u_{N-1}, x_N)^\top$$

- Use model to eliminate states as variables
  - Future states as function of inputs and initial state

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} = \begin{pmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{pmatrix} x_0 + \begin{pmatrix} B & & & & \\ AB & B & & & \\ A^2 & AB & B & & \\ \vdots & \vdots & \vdots & \ddots & \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \dots & B \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{pmatrix} = S^x x_0 + S^u U$$

  - Insert into objective (no constraints!)

$$\min_U \frac{1}{2} (S^x x_0 + S^u U)^\top \mathbf{Q} (S^x x_0 + S^u U) + \frac{1}{2} U^\top \mathbf{R} U \qquad \mathbf{Q} = \begin{pmatrix} Q & & \\ & Q & \\ & & \ddots \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} R & & \\ & R & \\ & & \ddots \end{pmatrix}$$

  - Solution found by setting gradient equal to zero:

$$U = \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{pmatrix} = -\left((S^u)^\top \mathbf{Q} S^u + \mathbf{R}\right)^{-1} (S^u)^\top \mathbf{Q} S^x x_0 = -F x_0$$

# Linear Quadratic Control
# Recursive approach

$$
\min_{z} \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^\top Q x_{t+1} + \frac{1}{2} u_t^\top R u_t
$$

$$
\text{s.t.} \quad x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \dots, N-1
$$

$$
z = (u_0, x_1, u_1, \dots, u_{N-1}, x_N)^\top
$$

- By writing up the KKT-conditions, we can show (we will do this later) that the solution can be formulated as:

$$
u_t = -K_t x_t
$$

where the feedback gain matrix is derived by

$$
\begin{aligned}
K_t &= R^{-1} B^\top P_{t+1} (I + B R^{-1} B^\top P_{t+1})^{-1} A, && t = 0, \dots, N-1 \\
P_t &= Q + A^\top P_{t+1} (I + B R^{-1} B^\top P_{t+1})^{-1} A, && t = 0, \dots, N-1 \\
P_N &= Q
\end{aligned}
$$

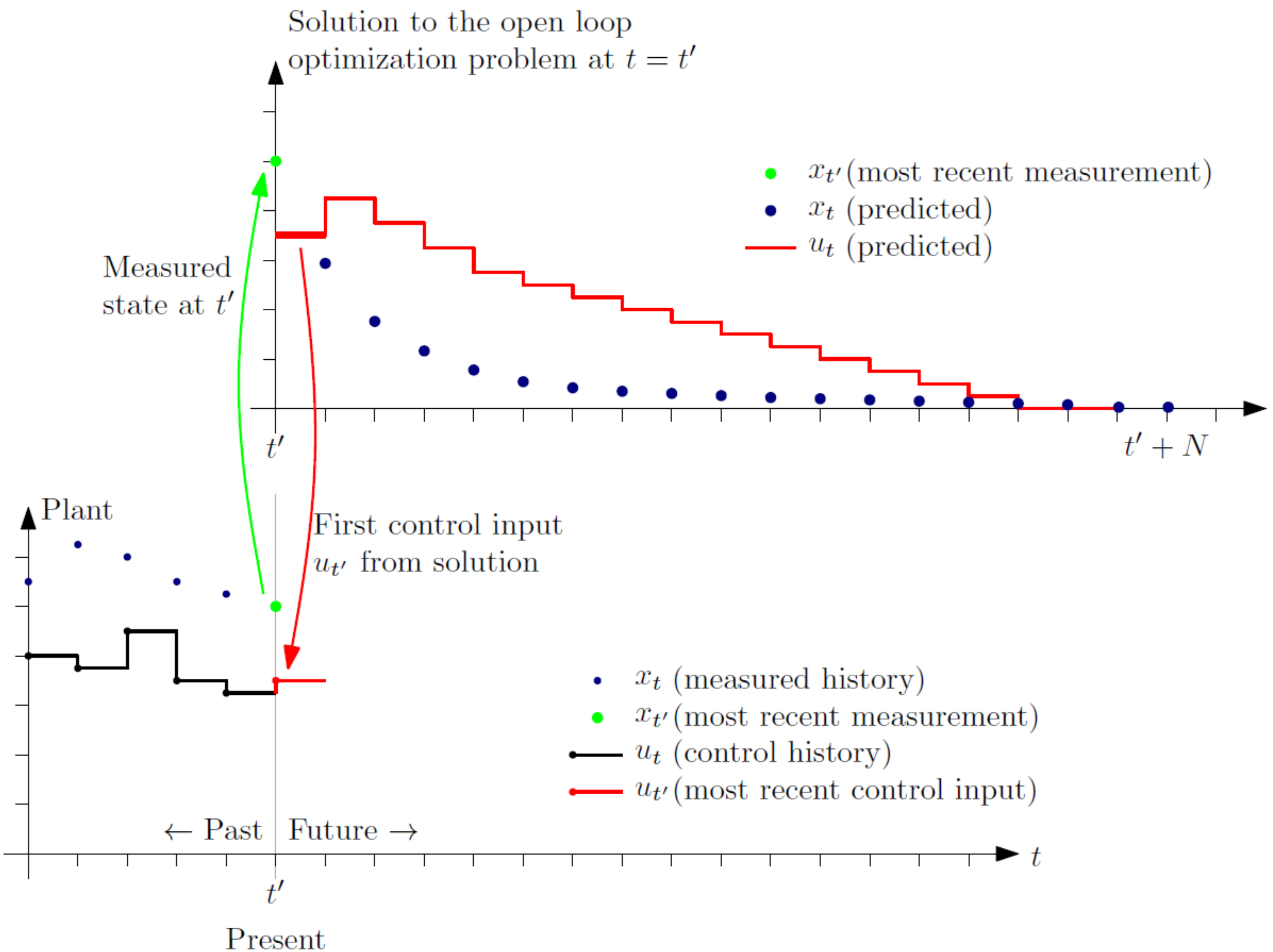
# Comments to the three solution approaches

- All give same numerical solution
  - If problem is strictly convex (Q psd, R pd), solution is unique
- The batch approaches give an open-loop solution, while the recursive approach give a closed-loop solution
  - Implies the recursive solution is more robust in implementation

$$\begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{pmatrix} = -Fx_0 \qquad \text{vs} \qquad u_t = -K_t x_t$$

- What about constraints:
  - Straightforward to add constraints to batch approaches (both becomes convex QPs)
  - Much more difficult to add constraints to the recursive approach
- Can we handle constraints (use batch approaches) <u>and</u> have feedback (and thereby robustness)?
  - Model predictive control!

# Model predictive control principle



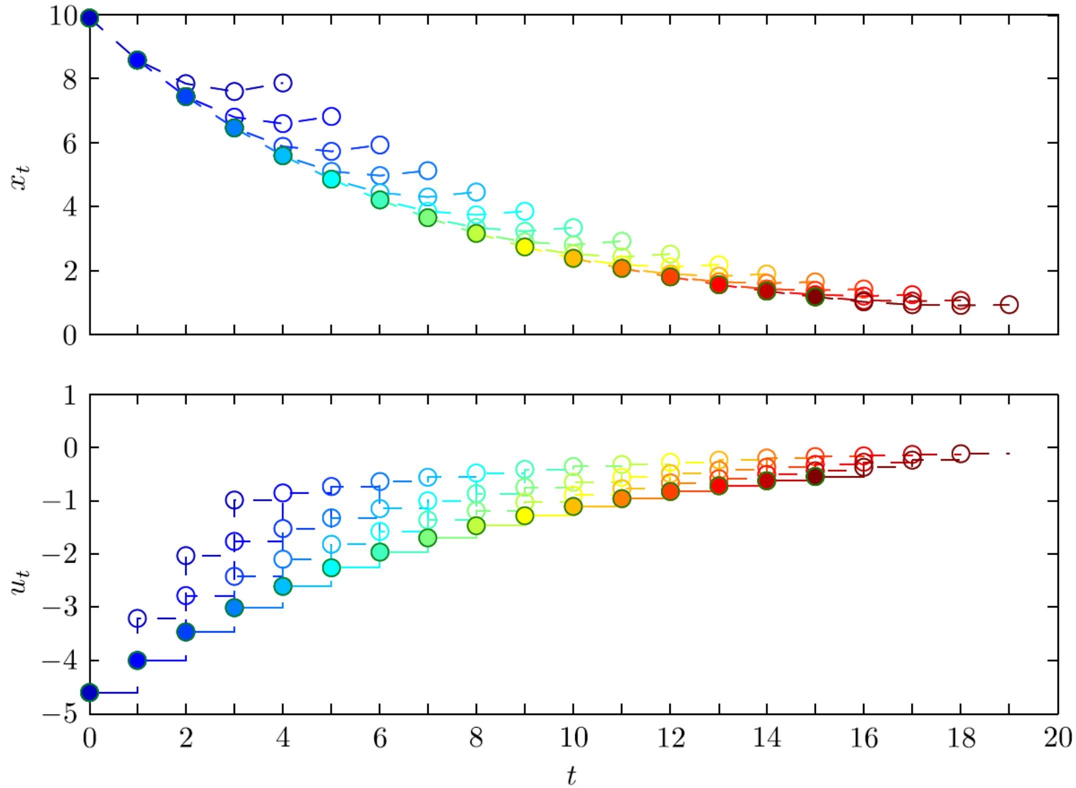
# MPC illustration

- Illustration

# Open-loop vs closed-loop trajectories

$$\min \sum_{t=0}^{4} x_{t+1}^2 + 4\,u_t^2$$

$$\text{s.t.} \qquad x_{t+1} = 1.2x_t + 0.5u_t, \quad t = 0, \ldots, 4$$




- Closed-loop trajectories different from open-loop (optimized) trajectories!
- It is the closed-loop trajectories that must analyzed for feasibility and stability.

# MPC optimality implies stability?

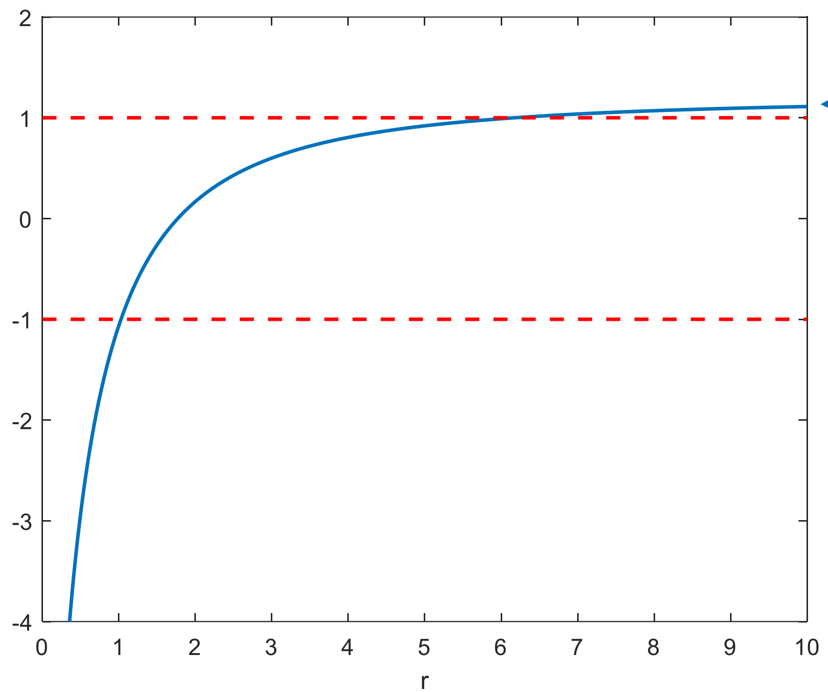
$$\min \sum_{t=0}^{1} x_{t+1}^2 + r\, u_t^2$$
$$\text{s.t.} \quad x_{t+1} = 1.2x_t + u_t, \quad t = 0, 1$$

MPC solution

$$u_t = -\frac{1.2 + 2.64r}{1 + 3.2r + r^2} x_t$$

MPC closed loop

$$x_{t+1} = \underbrace{\left(1.2 - \frac{1.2 + 2.64r}{1 + 3.2r + r^2}\right)}_{} x_t$$



# MPC and stability

Nominal vs robust stability

- “nominal stability”: The model used in optimization is correct (no “model-plant mismatch”, no disturbances)
- “robust stability” is stability under “model-plant mismatch” and/or disturbances (more difficult to analyze)

Requirements for stability:

- Stabilizability ( $(A,B)$ stabilizable )
- Detectability ( $(A,D)$ detectable )
  - $D$ is a matrix such that $Q = D^T D$ (that is, “$D$ is matrix square root of $Q$”)
  - Detectability: No modes can grow to infinity without being “visible” through $Q$

# How to achieve nominal stability?

$$
\begin{aligned}
\min_{z} \sum_{t=0}^{N-1} & \frac{1}{2} x_{t+1}^\top Q x_{t+1} + \frac{1}{2} u_t^\top R u_t \\
\text{s.t.} \quad & x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \dots, N-1 \\
& x^{\text{low}} \leq x_t \leq x^{\text{high}}, \quad t = 1, \dots, N \\
& u^{\text{low}} \leq u_t \leq u^{\text{high}}, \quad t = 0, \dots, N-1
\end{aligned}
$$

- Choose prediction horizon equal to infinity ($N = \infty$)
  - Usually not possible
- For given $N$, design $Q$ and $R$ such that MPC is stable (cf. example)
  - Difficult, and not always possible!
- Change the optimization problem such that
  - The new problem gives a finite upper bound of infinite horizon problem cost
  - The constraints is guaranteed to hold after the prediction horizon

$$
\begin{aligned}
\min_{z} \sum_{t=0}^{N-1} & \left( \frac{1}{2} x_t^\top Q x_t + \frac{1}{2} u_t^\top R u_t \right) + \frac{1}{2} x_N^\top P x_N \quad \longleftarrow \text{Terminal cost} \\
\text{s.t.} \quad & x_{t+1} = A x_t + B u_t, \quad t = 0, 1, \dots, N-1 \\
& x^{\text{low}} \leq x_t \leq x^{\text{high}}, \quad t = 1, \dots, N \\
& u^{\text{low}} \leq u_t \leq u^{\text{high}}, \quad t = 0, \dots, N-1 \\
& x_N \in \mathcal{S} \quad \longleftarrow \text{Terminal constraint}
\end{aligned}
$$

  - Fairly straightforward to do in theory, but can be “clumsy” in practice
- Typically, in practice: Choose $N$ “large”
  - Stability guaranteed for $N$ large enough, but difficult/conservative to compute this limit
  - Shorter $N$ often OK
  - So what is “large enough” in practice? Rule of thumb: longer than dominating dynamics

# Why MPC over PI?

## Advantages of MPC

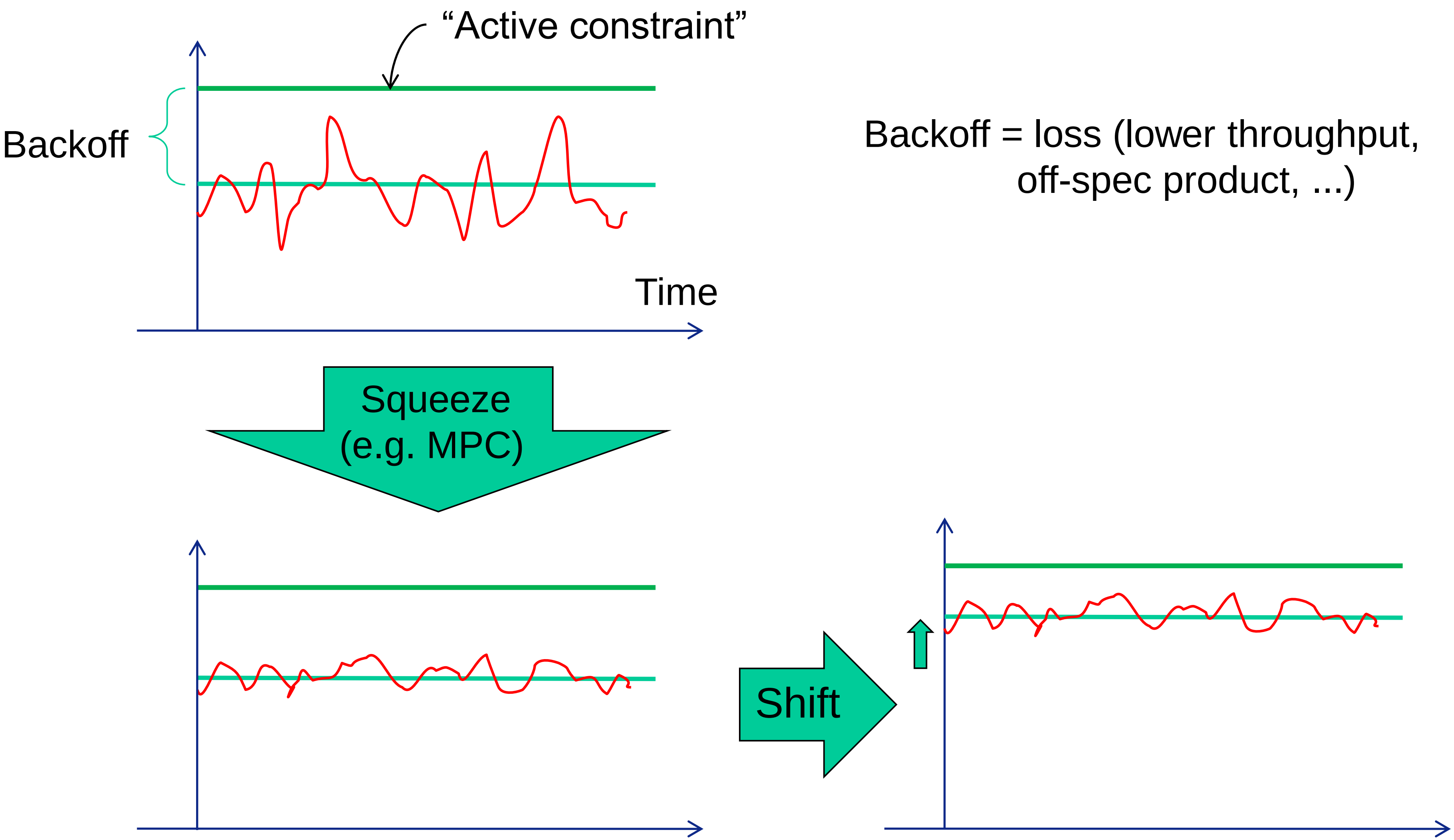
- MPC handles constraints in a transparent way
  - Physical constraints (actuator limits), performance constraints, safety limits, …
- MPC is by design multivariable (MIMO)
- MPC gives “optimal” performance

## Disadvantage with MPC

- Online complexity
- Requires models! Increased commisioning cost?
- Difficult to maintain?

# “Squeeze and shift”

How MPC (or better control in general) improves profitability




Backoff = loss (lower throughput,
off-spec product, ...)