2E1252 Control Theory and Practice

Lecture 13: Model Predictive Control

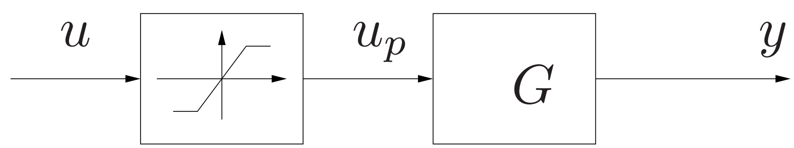
Elling W. Jacobsen

jacobsen@s3.kth.se

S3 - Automatic Control, KTH

Review

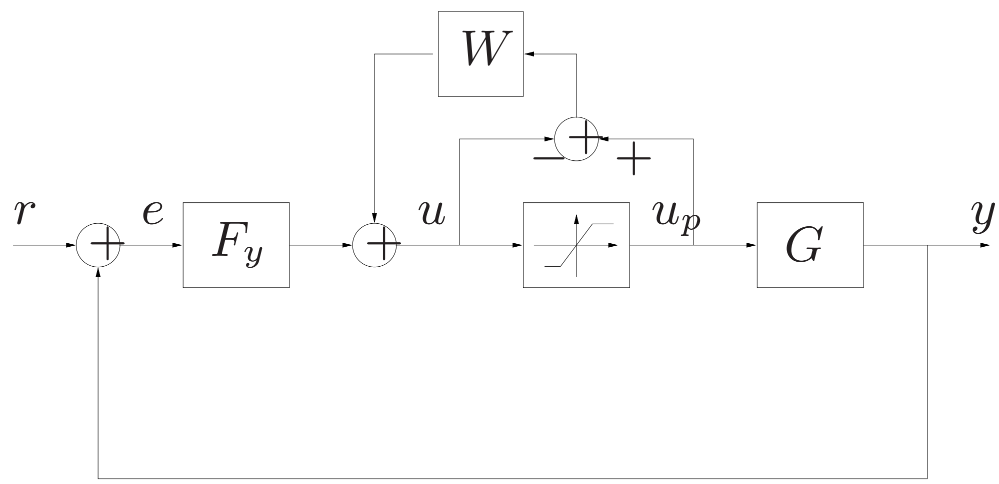
- In practice: control inputs have hard constraints: $u \in [u_{min}, u_{max}]$



- Feedback loop open when input saturated; can imply severe problems if controller contains integrator or is unstable.
- Classic solution: *anti reset windup*
- Better, include constraints in controller calculations:
  Model Predictive Control – MPC

Review: Anti Reset Windup

- If controller can be written on the form *observer + state feedback*: use constrained (real) input in observer.
- Interpretation in block-diagram: feedback from $u_p - u$



- Same idea can be used for any controller $F_y(s)$

S3-Automatic Control

Model Predictive Control - MPC

1. At sample t compute future outputs

$$y(t+k), \quad k = 1, \ldots, M$$

   as a function of future control inputs

$$u(t+k), \quad k = 0, \ldots, N$$

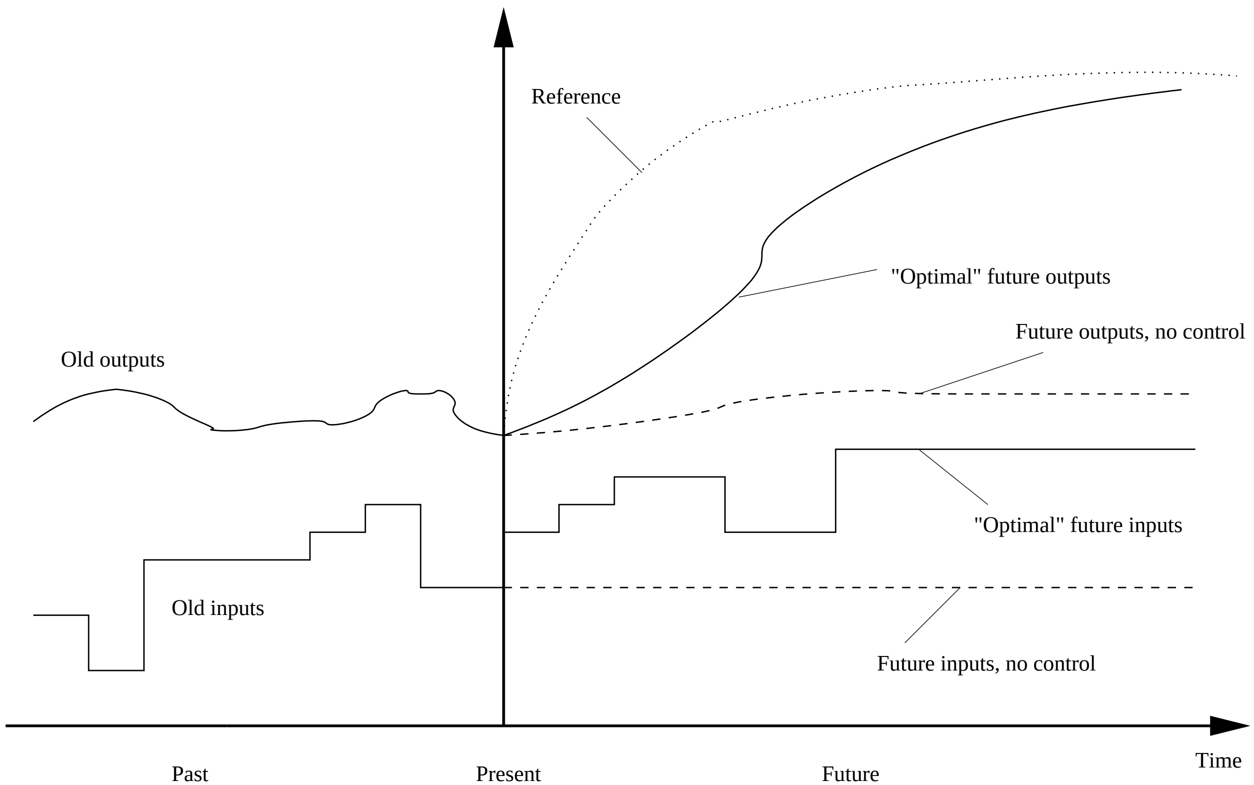
2. Minimize some objective function using $u(t+k),\ k = 0, \ldots, N$

3. Implement $u(t)$ on system

4. At next sample $t+1$ go to 1. ("receding horizon control")

– The main point: include constraints in optimization.

MPC – The Principle Idea



Model: $$x_{k+1} = Ax_k + Bu_k \; ; \quad y_k = Cx_k$$

S3-Automatic Control

This Lecture

- Constrained optimization: Linear (LP) and Quadratic Programming (QP)
- Translating the **MPC problem** into a **QP problem**.
- Examples
- Issues related to performance, stability, feasibility and robustness.

Constrained Optimization Problem

$$\min_u f(x,u)$$

$$\text{subject to} \quad g_1(x,u) \le 0, \quad g_2(x,u) = 0$$

- Solvability depends on form of objective function and constraint
- Efficient solvers exist for

  linear $h^T u$ *and quadratic* $u^T H u + h^T u$ *objective functions*

  combined with *linear constraint* $Lu \le b$

  where H, h, l, A, b are constant matrices/vectors
- Linear objective function: Linear Programming (LP)
- Quadratic objective function: Quadratic Programming (QP)

A General MPC Formulation

$$\min_u f(x,u) = \sum_{i=0}^{N_P-1} [(x_i - x_{ref,i})^T Q_x (x_i - x_{ref,i}) + (u_i - u_{ref,i})^T Q_u (u_i - u_{ref,i})]$$
$$. \qquad + (x_{N_P} - x_{ref,N_P})^T S (x_{N_P} - x_{ref,N_P})$$

subject to

$$x_0 = \texttt{given}$$

$$u_{min} \le u_i \le u_{max}$$

$$y_{min} \le Hx_i \le y_{max}$$

- $Q_x \ge 0$, $Q_u > 0$ and $S > 0$ all symmetric
- $N_p = M = N - 1$ – prediction horizon
- Terminal weight S accounts for different input/output horizons
- Weight outputs $(y - y_{ref})$ with $Q_x = C^T Q_y C$

MPC → QP Problem

- The QP-problem

$$\min_u u^T H u + h^T u \;; \qquad \texttt{s.t.} \quad Lu \leq b$$

  with $H > 0$

- To translate MPC formulation into QP form: translate objectives and constraints for state x into objectives and constraints for input u (and stack everything into a compact matrix form).

- In principle, straightforward since

$$x_k = A^k x_0 + \sum_{j=0}^{k-1} A^{k-1-j} B u_j$$

- But, translation still require extensive matrix manipulations....

Vector Notation

$$u_{ref} = \begin{pmatrix} u_{ref,0} \\ u_{ref,1} \\ \vdots \\ u_{ref,N_P-1} \end{pmatrix} \; ; \quad x_{ref} = \begin{pmatrix} x_{ref,1} \\ x_{ref,2} \\ \vdots \\ x_{ref,N_P} \end{pmatrix}$$

Introduce $v_i = u_i - u_{ref,i}$ and $\chi_i = x_i - x_{ref,i}$

$$v = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{N_P-1} \end{pmatrix} \; ; \quad \chi = \begin{pmatrix} \chi_1 \\ \chi_2 \\ \vdots \\ \chi_{N_P} \end{pmatrix}$$

Superposition

Split state "error" χ into the part given by reference input u_{ref} (often 0) and the part obtained by input change v computed by optimization

$$\chi = \chi_{dev} + \chi_v$$

where

$$\chi_{dev} = \underbrace{\begin{pmatrix} A \\ A^2 \\ \vdots \\ A^{N_P-1} \\ A^{N_P} \end{pmatrix}}_{\hat{A}} x_0 + \underbrace{\begin{pmatrix} B & 0 & \dots & 0 & 0 \\ AB & B & \dots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ A^{N_P-2}B & A^{N_P-3}B & \dots & B & 0 \\ A^{N_P-1}B & A^{N_P-2}B & \dots & AB & B \end{pmatrix}}_{\hat{B}} u_{ref} - x_{ref}$$

S3-Automatic Control

Thus,

$$\chi = \chi_{dev} + \chi_v$$

where state deviation with u_{ref} given by

$$\chi_{dev} = \hat{A}x_0 + \hat{B}u_{ref} - x_{ref}$$

and contribution from $v = u - u_{ref}$ is

$$\chi_v = \hat{B}v$$

Only χ_v is affected by optimization.

Objective Function on Matrix Form

Introduce

$$\hat{Q}_x = \begin{pmatrix} Q_x & 0 & \dots & 0 & 0 \\ 0 & Q_x & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & Q_x & 0 \\ 0 & 0 & \dots & 0 & S \end{pmatrix} \; ; \quad \hat{Q}_u = \begin{pmatrix} Q_u & 0 & \dots & 0 & 0 \\ 0 & Q_u & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & Q_u & 0 \\ 0 & 0 & \dots & 0 & Q_u \end{pmatrix}$$

The objective function becomes

$$f(x,u) = (x_0 - x_{ref,0})^T Q_x (x_0 - x_{ref,0}) + (\chi_{dev} + \chi_v)^T \hat{Q}_x (\chi_{dev} + \chi_v) + v^T \hat{Q}_u v$$

$$= (x_0 - x_{ref,0})^T Q_x (x_0 - x_{ref,0}) + \chi_{dev}^T \hat{Q}_x \chi_{dev} + 2\chi_{dev}^T \hat{Q}_x \chi_v + \chi_v^T \hat{Q}_x \chi_v + v^T \hat{Q}_u v$$

which is to be minimized using input change v as the free variable

The QP formulation

- The terms $(x_0 - x_{ref,0})^T Q_x (x_0 - x_{ref,0})$ and $\chi_{dev}^T \hat{Q}_x \chi_{dev}$ are not affected by the optimization, i.e., v, and can hence be left out of objective function

$$\tilde{f}(\chi, v) = 2\chi_{dev}^T \hat{Q}_x \chi_v + \chi_v^T \hat{Q}_x \chi_v + v^T \hat{Q}_u v$$

- With $\chi_v = \hat{B}v$ we get the corresponding QP-matrices

$$H = \hat{B}^T \hat{Q}_x \hat{B} + \hat{Q}_u$$

$$h^T = \chi_{dev}^T \hat{Q}_x \hat{B}$$

- Objective function:

$$v^T H v + h^T v$$

- Next: put constraints on form $Lv \leq b$

Translating Input Constraints, $0 \leq i \leq N_P - 1$

$$\underbrace{I}_{L_1} v \leq \underbrace{\begin{pmatrix} u_{max} \\ u_{max} \\ \vdots \\ u_{max} \end{pmatrix} - u_{ref}}_{b_1} \; ; \quad \underbrace{-I}_{L_2} v \leq \underbrace{-\begin{pmatrix} u_{min} \\ u_{min} \\ \vdots \\ u_{min} \end{pmatrix} + u_{ref}}_{b_2}$$

Stacked

$$L_u v \leq b_u \; ; \quad L_u = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} \; ; \quad b_u = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Translating State Constraints, $1 < i < N_P$

Upper constraint:

$$\underbrace{\begin{pmatrix} H & 0 & \dots & 0 & 0 \\ 0 & H & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & H & 0 \\ 0 & 0 & \dots & 0 & H \end{pmatrix}}_{\hat{H}} \chi_v \leq \begin{pmatrix} y_{max} \\ y_{max} \\ \vdots \\ y_{max} \end{pmatrix} - \begin{pmatrix} H & 0 & \dots & 0 & 0 \\ 0 & H & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & H & 0 \\ 0 & 0 & \dots & 0 & H \end{pmatrix} (\chi_{dev} + x_{ref})$$

Lower constraint: $-\hat{H}\chi_v \leq - \begin{pmatrix} y_{min} \\ y_{min} \\ \vdots \\ y_{min} \end{pmatrix} + \hat{H}(\chi_{dev} + x_{ref})$

S3-Automatic Control

Inserting $\chi_v = \hat{B}v$

we obtain constraint on v

$$L_x v \le b_x \ ; \quad L_x = \begin{pmatrix} \hat{H}\hat{B} \\ -\hat{H}\hat{B} \end{pmatrix} \ ; \quad b_x = \begin{pmatrix} b_{x1} \\ b_{x2} \end{pmatrix}$$

with

$$b_{x1} = \begin{pmatrix} y_{max} \\ y_{max} \\ \vdots \\ y_{max} \end{pmatrix} - \hat{H}(\chi_{dev} + x_{ref}) \ ; \quad b_{x2} = - \begin{pmatrix} y_{min} \\ y_{min} \\ \vdots \\ y_{min} \end{pmatrix} + \hat{H}(\chi_{dev} + x_{ref})$$

The QP Formulation

At each sample solve problem

$$\min_u v^T H v + h^t v \quad s.t.\ Lv \leq b$$

with

$$H = \hat{B}^T \hat{Q}_x \hat{B} + \hat{Q}_u \ ; \quad h^T = \chi_{dev} \hat{Q}_x \hat{B}$$

and

$$L = \begin{pmatrix} L_u \\ L_x \end{pmatrix} \ ; \quad b = \begin{pmatrix} b_u \\ b_x \end{pmatrix}$$

- Solve e.g., using `quadprog` in Matlab
- Support for setting up all matrices: e.g., MPC Toolbox.

Some Practical Issues

- States usually from observer.
- Need to provide reference x_{ref} for state and u_{ref} for input
- In most cases: $u_{ref,i} = 0 \ \forall i$
- Usually given reference y_{ref} for output y:

$$x_{ref,i} = pinv(c) y_{ref,i}$$

  where $pinv$ denotes pseudoinverse

- Note that constraint is applied on u_0, i.e., input at present time, but not on x_0, i.e., state at present time. This because we can not affect x_0 with v.

The DC Servo – Again....

$$G(s) = \frac{1}{s(s+1)} \; ; \quad T = 0.05$$

$$A = \begin{pmatrix} 1.95 & -0.48 \\ 2 & 0 \end{pmatrix} \; ; \quad B = \begin{pmatrix} 0.031 \\ 0 \end{pmatrix}$$

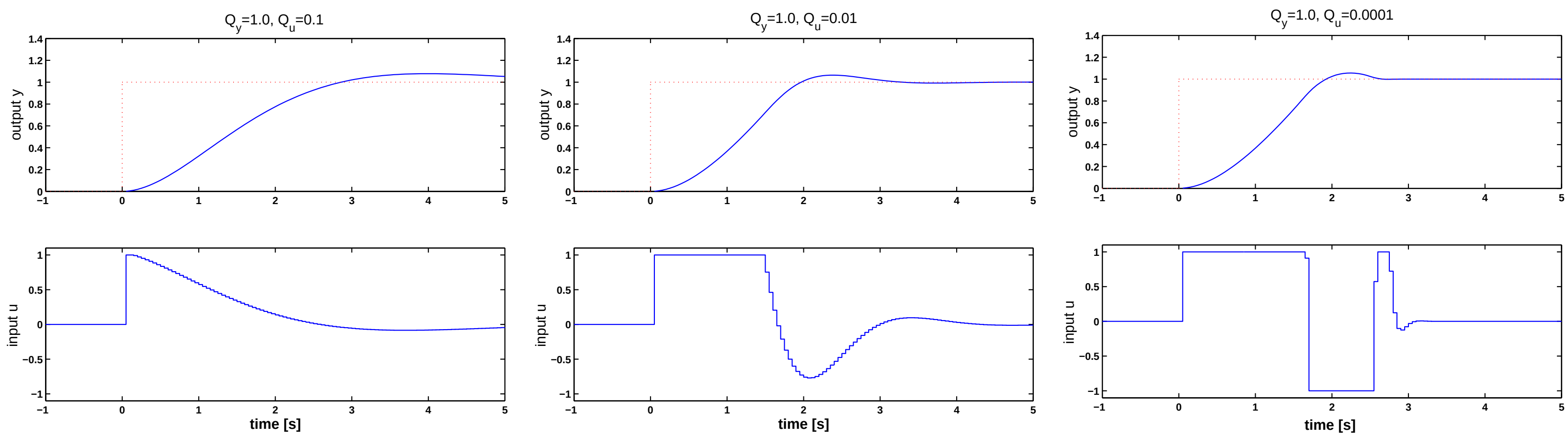
$$C = \begin{pmatrix} 0.039 & 0.019 \end{pmatrix}$$

- Constrained input voltage: $-1 < u < 1$
- Constrained position: $y_{min} < y < y_{max}$

S3-Automatic Control

Impact of weightings Q_u/Q_y (no output constraints)

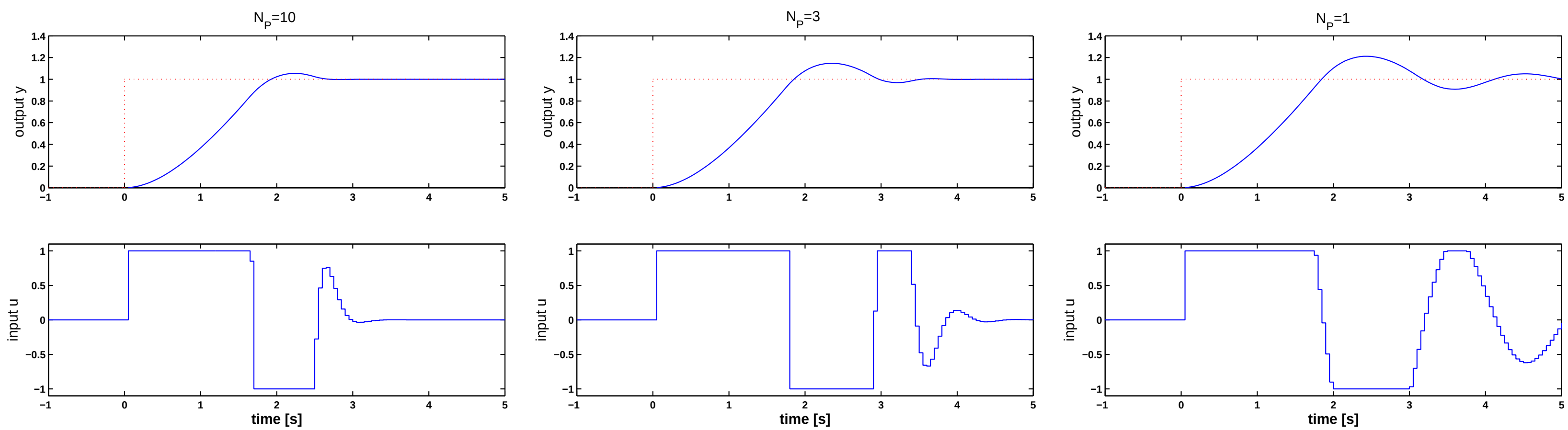
Prediction horizon $N_P = 10$



S3-Automatic Control

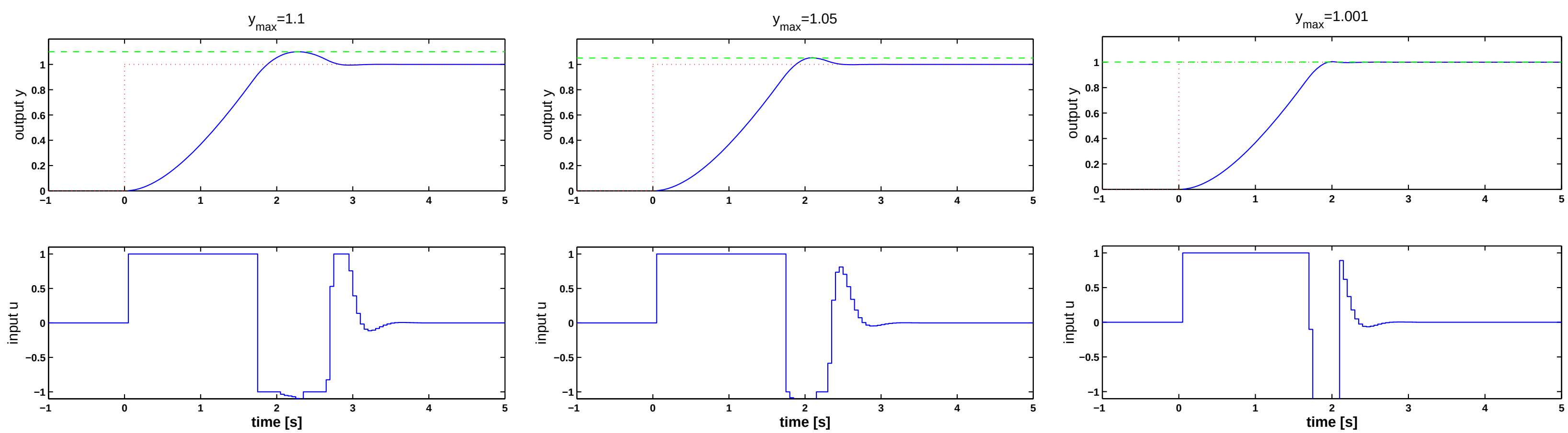
Impact of horizon N_P (no output constraints)

Weightings $Q_y = 1.0, Q_u = 0.0001$



Adding constraint on output

$$Q_y = 1, Q_u = 0.0001, N_P = 5, 0.0 < y < y_{max}$$



S3-Automatic Control

Including Integral Action

- Integral action often included by using *changes in free variables* $\Delta u_i = u_i - u_{i-1}$ as free variables in the optimization.
- Actual inputs then obtained by integrating changes in the input
- Use framework as above, but with extended model

$$\begin{pmatrix} x_{k+1} \\ u_k \end{pmatrix} = \underbrace{\begin{pmatrix} A & B \\ 0 & I \end{pmatrix}}_{\tilde{A}} \begin{pmatrix} x_k \\ u_{k-1} \end{pmatrix} + \underbrace{\begin{pmatrix} B \\ I \end{pmatrix}}_{\tilde{B}} \Delta u_k$$

$$y_k = \underbrace{\begin{pmatrix} C & 0 \end{pmatrix}}_{\tilde{C}} \begin{pmatrix} x_k \\ u_{k-1} \end{pmatrix}$$

S3-Automatic Control

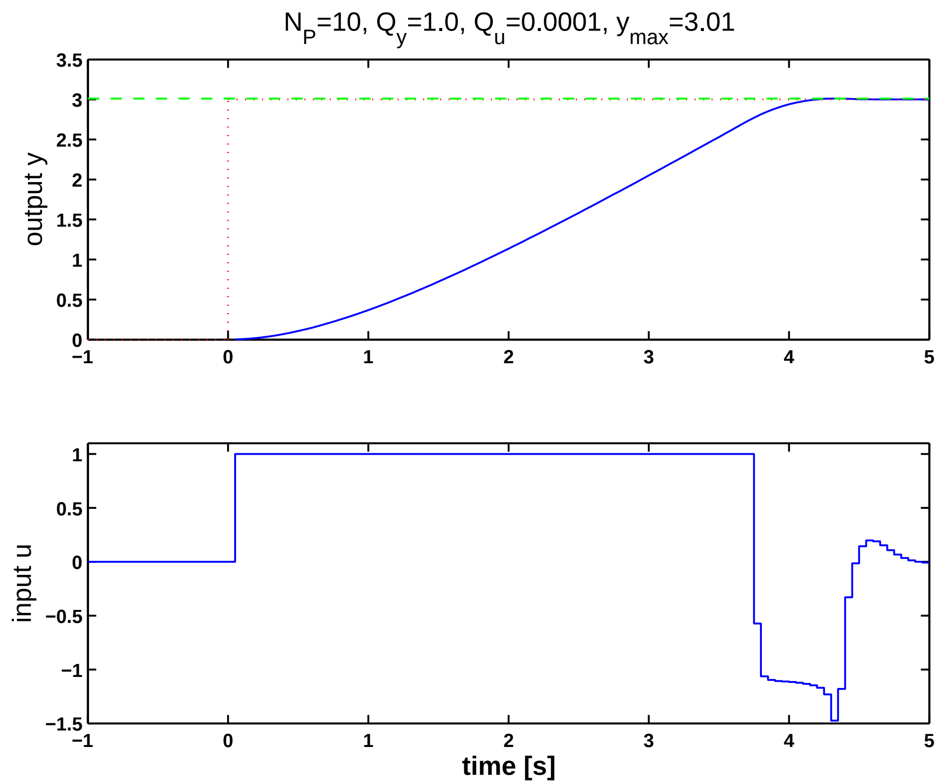
Input and Output Constraints for $i \geq N_P$

- Often constraints imposed also on states beyond optimization horizon N_P
- Requires assumption on u_i for $i \geq N_P$. Typical
  - $u_i = u_{ref,i}$
  - $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$ where K is a stabilizing state feedback, e.g., from LQR.
- In the latter case, also input constraints can be included for $i \geq N_P$ by translating into state constraints.

Stability?

- Objective function closely resembles discrete time LQ optimal control.
- Infinite horizon LQ stable for stabilizable and detectable systems.
- Finite horizon LQ stable provided weight S on "terminal" state x_{NP} sufficiently large.
- With constraints: no stability guarantee.

Infeasible Solution – DC Servo



Ensuring Feasibility – Soft Constraints

- Constrained optimization problems may become **infeasible**, i.e., no solution exists that satisfies all constraints.
- In that case, computation of optimal solution breaks down.
- Input constraints are typically inherently hard, e.g., max power
- But, state and output constraints usually imposed, e.g., safety limit
- Typical solution to ensure feasibility: introduce penalty functions in the optimization problem, e.g.,

$$y \leq y_{max} + f_y$$

  Add f_y in objective function and in the free variables for optimization.

  Relaxed constraints are often termed *soft constraints*.

Robustness of MPC

- Consider unconstrained optimization problem

$$\min_v f(v) = 0.5v^T(\hat{B}\hat{Q}_x\hat{B} + Q_u)v + \chi_{dev}^T\hat{Q}_x\hat{B}v$$

- Without constraints, analytical solution becomes

$$v = -(\hat{B}^T\hat{Q}_x\hat{B} + \hat{Q}_u)^{-1}\hat{B}^T\hat{Q}_x\chi_{dev}$$

- Model uncertainty, i.e., uncertain A and B in state-space model, implies uncertain $\hat{B}$
- If *Hessian* $\hat{B}^T\hat{Q}_x\hat{B} + \hat{Q}_u$ is ill-conditioned ($\bar{\sigma}/\underline{\sigma} >> 1$), then inversion highly sensitive to uncertainty.
- Thus, MPC potentially displays poor robustness.

- Improving robustness, i.e., reducing condition number of $Hessian$ $\hat{B}^T\hat{Q}_x\hat{B} + \hat{Q}_u$

  1. Scaling inputs and outputs in model, so as to change $\hat{B}$
  2. Modifying weighting matrices $\hat{Q}_x$ and $\hat{Q}_u$

Get a Feel for Tuning MPC

Try it out!

Files for setting up problem and simulating on homepage.

See also Computer Exercise 5 (download from homepage next week)