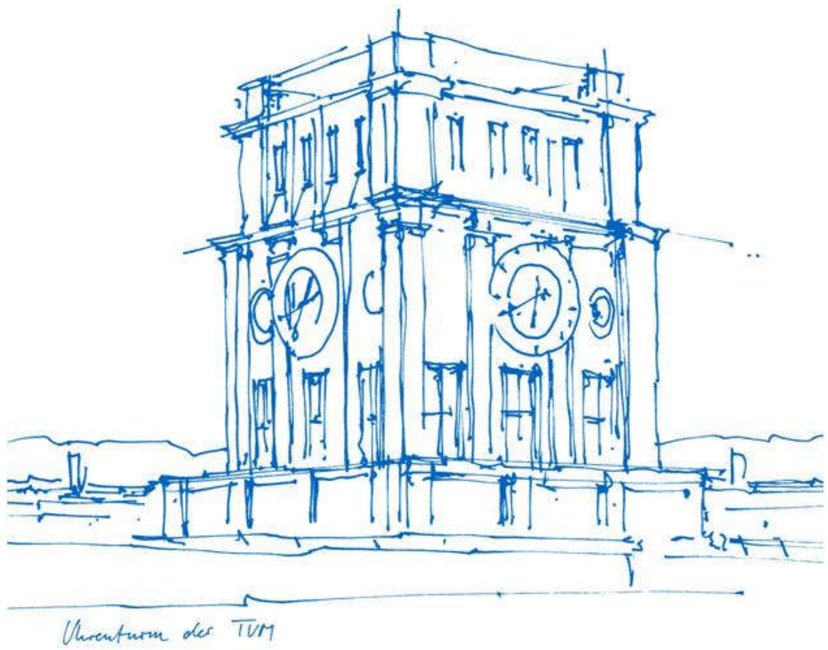
Model Predictive Control (MPC) Toolbox

Ricardo de Castro

Technische Universität München

Lehrstuhl für Elektrische Antriebssysteme



Agenda

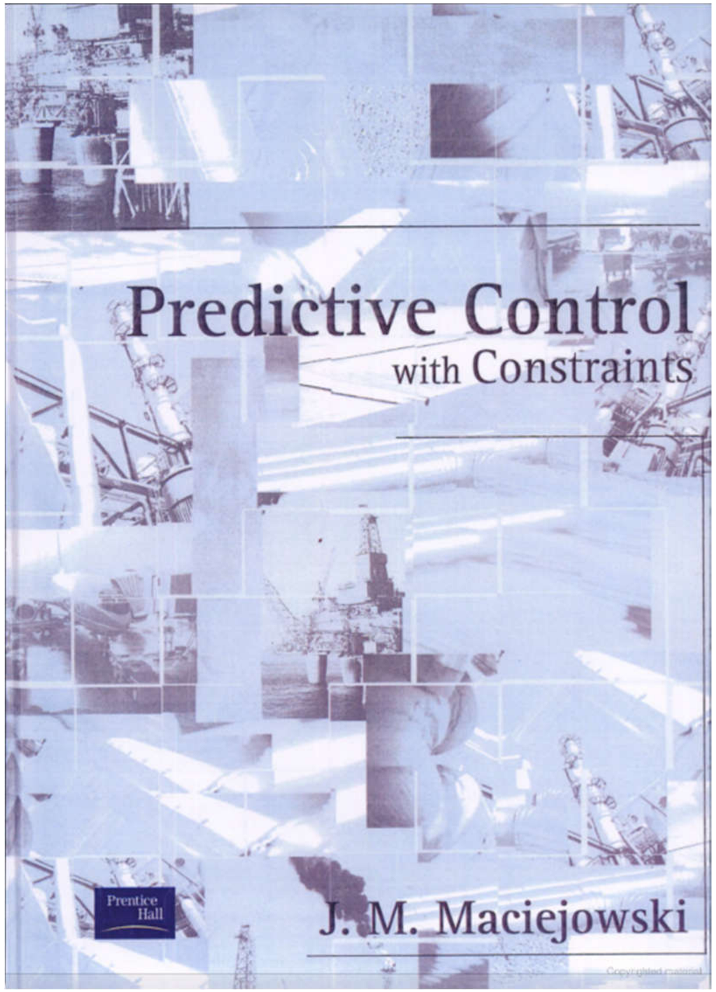
- **Introduction to MPC**
  - Motivation & Preliminaries

- **Unconstrained MPC**
  - Analytical Solution
  - <u>Matlab Example</u>: Double integrator

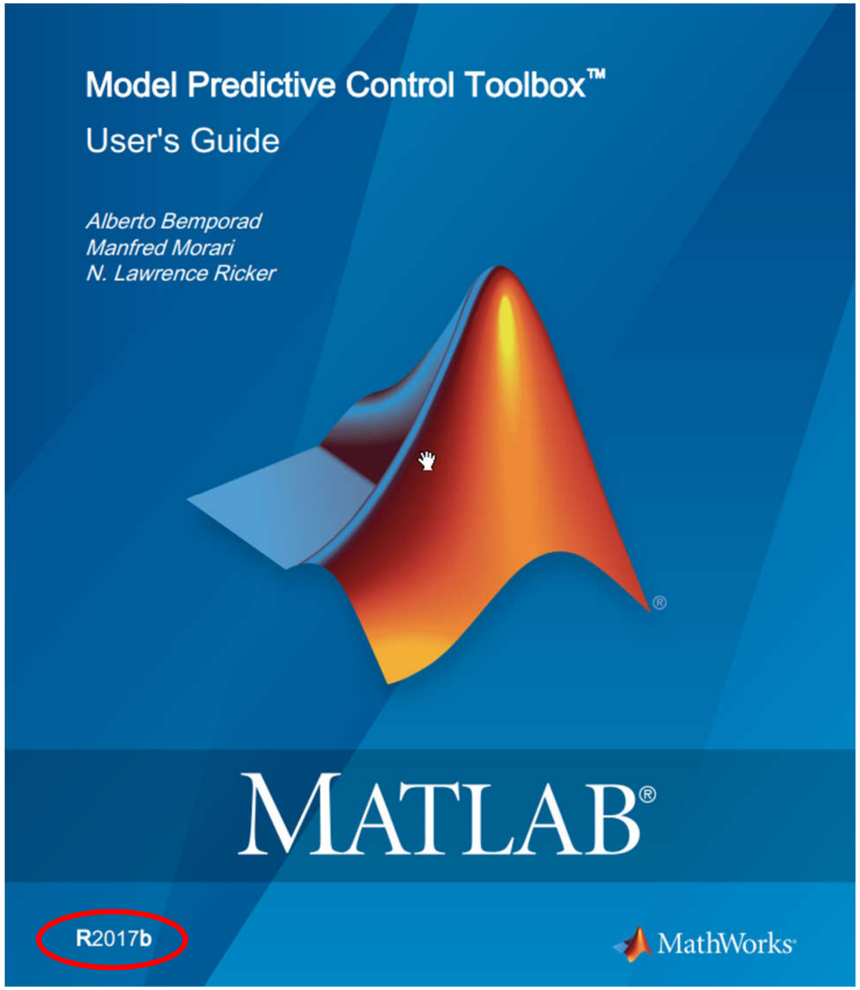
- **Constrained MPC**
  - Solution via *Optimization Toolbox*
  - <u>Matlab Example</u>: Double integrator with input constrains

- ***MPC Toolbox***
  - Command-line API
    - Setup of prediction model, weights & constraints
    - Computing optimal control & closed-loop simulation
  - <u>Matlab Examples</u>:
    - Double integrator with input & state constraints
    - Aircraft pitch & altitude control

References

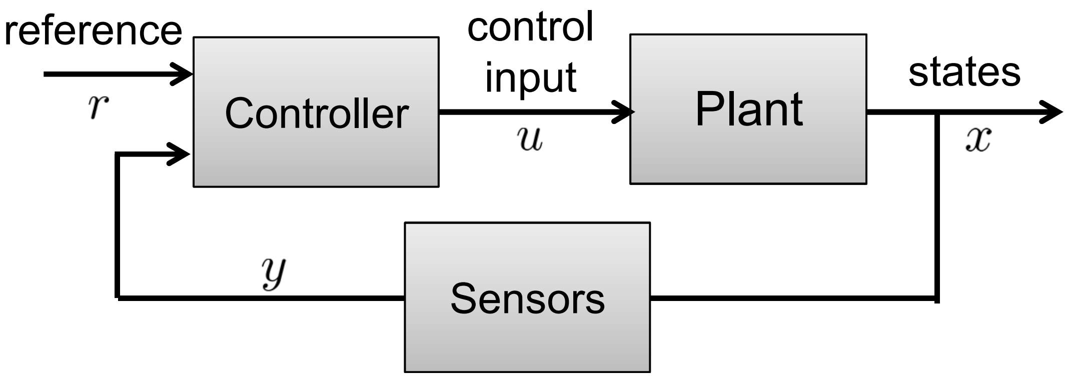


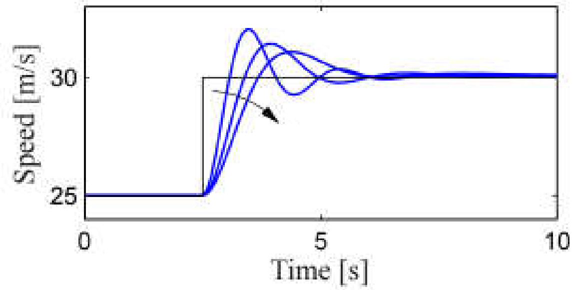
(Chap. 1 & 2)



MPC: Why?

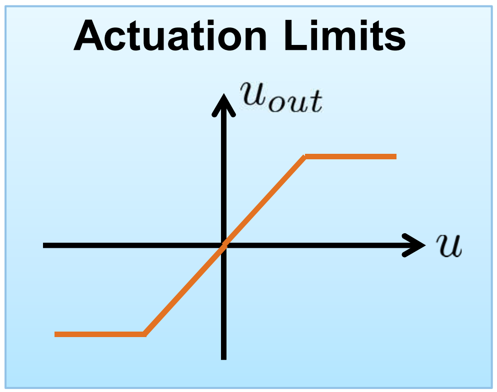
Control System



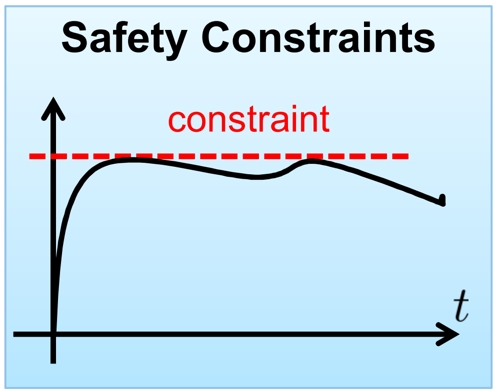


Example: adaptive cruise control

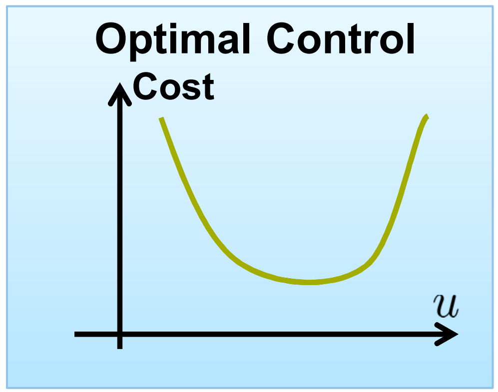
- Control input: motor torque
- States: pos., vel., acc.,
- Reference: desired velocity



e.g. motor max. torque

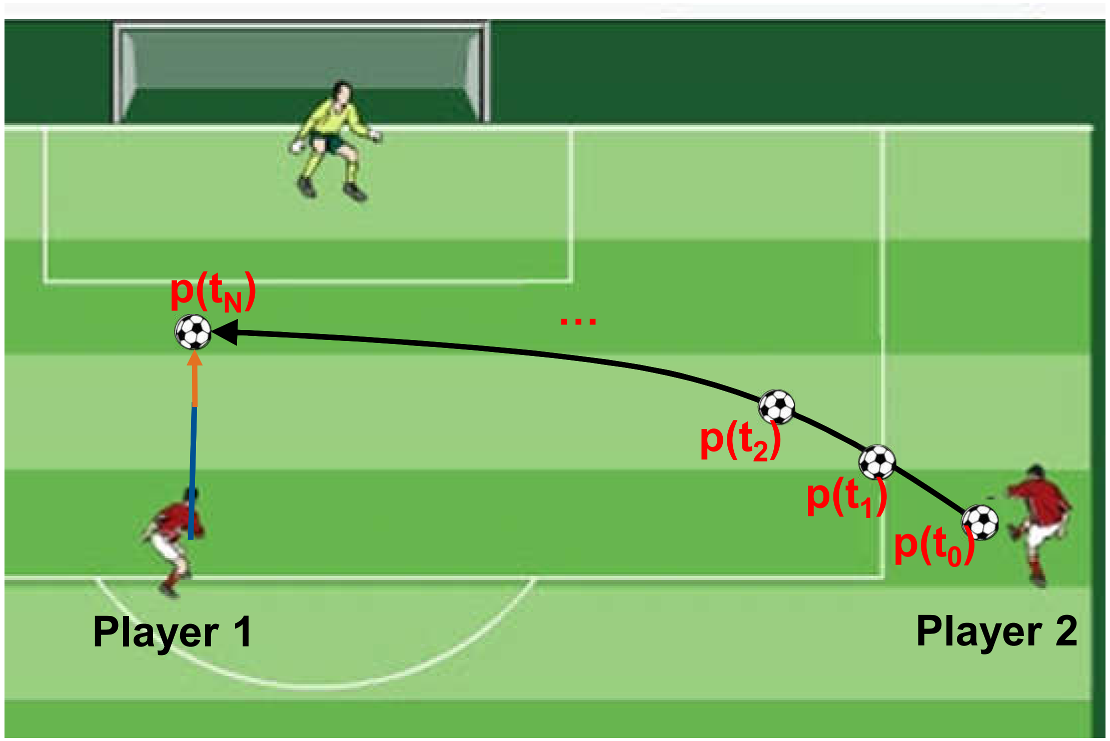


e.g.velocity/temperature/ pressure limits



e.g. minimize actuation energy, control error, …

MPC: A Soccer Analogy



Goal: predict actions of Player 1 in order to hit ball at time t_N

t_0 = start time

t_N = prediction horizon/window

p = position of the ball

1) Predict ball position

$p(t_1)$, $p(t_2)$, … $p(t_N)$

2) Predict actions of player 1

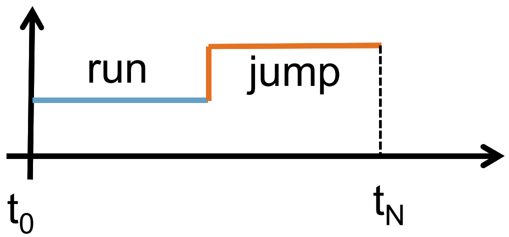


Image source: http://fussballuebung.de/

MPC: Motivation Example

Scalar, linear and discrete-time system:

$$x[k+1] = x[k] + u[k],$$

$$\begin{aligned} \text{initial state: } & x[0] = x_0, \\ \text{control input: } & u \in \mathbb{R}, \\ \text{state: } & x \in \mathbb{R} \end{aligned}$$

Goal: find $u[0], u[1], \ldots, u[N-1]$ that minimizes cost function

$$J = \sum_{k=0}^{N-1} x^2[k] + \rho u^2[k] + x^2[N],$$

parameters defined by the designer:

$$\begin{aligned} \text{prediction horizon: } & N \\ \text{weight: } & \rho > 0 \end{aligned}$$

MPC: Motivation Example

Solution: $(\mathrm{N}=1)$

1) **Predict future states:**

$$x[0] = x_0,$$
$$x[1] = x_0 + u[0],$$

2) **Compute control actions that optimize (predicted) future states:**

$$\min J = x_0^2 + \rho u^2[0] + (x_0 + u[0])^2 \xrightarrow{\nabla J(u[0]) = 0} u^*[0] = -\tfrac{1}{\rho+1} x_0$$

Effect of tuning weight ρ: $x^*[1] = \left(1 - \frac{1}{\rho+1}\right) x_0$

$$\xrightarrow{\rho \gg 1} x^*[1] \approx x_0$$
$$\xrightarrow{\rho \approx 0} x^*[1] \approx 0$$

Questions addressed in this lecture:

- What if $N > 1$?
- What if x is a vector?

→ **Unconstrained MPC** (analytical solutions)

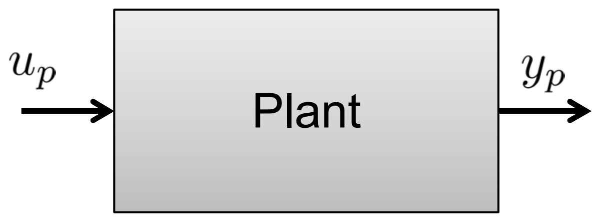
- What if the system has control constraints?
- What if the system has state constraints?

→ **Constrained MPC** (numerical solutions)

MPC: Plant Model & Constraints

Plant model: discrete-time, state-space & linear

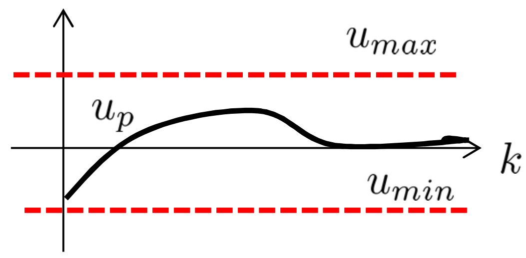
$$x_p[k+1] = Ax_p[k] + Bu_p[k]$$
$$y_p[k] = Cx_p[k]$$



$x_p \in \mathbb{R}^{n_x}$ = state, $\quad u_p \in \mathbb{R}^{n_u}$ = control input

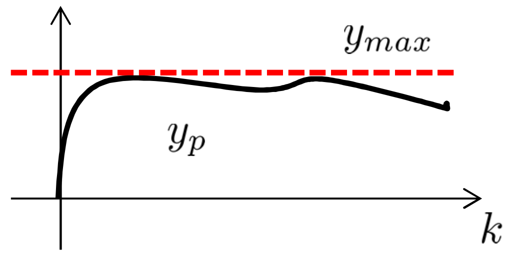
$y_p \in \mathbb{R}^{n_y}$ = measured output

(A, B, C) = state-space matrices



Input constraints: $\mathcal{U} = \{u_p : u_{min} \leq u_p[k] \leq u_{max}\}$

Output constraints: $\mathcal{Y} = \{y : y_{min} \leq y_p[k] \leq y_{max}\}$



Assumptions:

- all states are measured or estimated ($C = I$)
- scalar input ($n_u = 1$)
- regulation goal ($x_p \rightarrow 0$) or tracking goal ($x_p \rightarrow r$)

Unconstrained MPC

Problem Formulation

A) cost function

$$\min\ J(u[k]) = \sum_{k=0}^{N-1} x^T[k]Qx[k] + u^T[k]Ru[k] + x^T[N]Px[N]$$

B) prediction model

$$\text{s.t.}\quad x[0] = x_0$$
$$x[k+1] = Ax[k] + Bu[k], \quad k = 0, \ldots, N-1$$

C) parameters

Q = state penalty (symmetric and positive semi-definite, $Q = Q^T \geq 0$)

R = input penalty (symmetric and positive definite, $R = R^T > 0$)

P = final state penalty (symmetric and positive semi-definite, $P = P^T \geq 0$)

N = prediction horizon, x_0 = initial state

Goal: compute control sequence $u[0], u[1], \ldots, u[N-1]$ that minimize the cost $J(.)$

$$\Updownarrow$$

$$u^*[k] = \arg\min J(u[k])$$

Remark: no input or output constraints ($\mathcal{U} = \mathbb{R}^{n_u}, \mathcal{Y} = \mathbb{R}^{n_y}$)

Unconstrained MPC: Solution (I)

1) Predict future states

1.1) **Group** future control inputs and predicted/future states into vectors

$$U = \begin{bmatrix} u[0] \\ \vdots \\ u[N-1] \end{bmatrix}, \qquad X = \begin{bmatrix} x[1] \\ \vdots \\ x[N] \end{bmatrix}$$

1.2) **Compute** X

$$\boxed{X = S_x x_0 + S_u U,} \qquad S_x = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, \; S_u = \begin{bmatrix} B & 0 & 0 & \dots & 0 \\ AB & B & 0 & \dots & 0 \\ A^2B & AB & B & \dots & 0 \\ \vdots & & & & \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \dots & B \end{bmatrix}$$

1.3) **Express cost function** as function of U, X

$$\min J(U) = x_0^T Q x_0 + X^T \overline{Q} X + U^T \overline{R} U$$

$$\text{s.t.} \quad X = S_x x_0 + S_u U$$

$$\overline{Q} = \text{blockdiag}(Q, \dots, Q, P) \quad \overline{R} = \text{blockdiag}(R, \dots, R)$$

Unconstrained MPC: Solution (II)

2) Compute control actions that optimize (predicted) future states:

2.1) **Substitute** equality constraint into the cost function

$$\boxed{\min J(U) = x_0^T \Gamma x_0 + 2U^T F x_0 + U^T G U}$$

$$\begin{cases} \Gamma = Q + S_x^T \overline{Q} S_x \\ F = S_u^T \overline{Q} S_x \\ G = \overline{R} + S_u^T \overline{Q} S_u \end{cases}$$

2.2) **Minimum obtained** by finding U that yields zero gradient

Analytical solution

$$\nabla J(U) = 0 \longrightarrow \boxed{U^* = -G^{-1} F x_0}$$

Unconstrained MPC: Implementation

Key idea:

- Extract first element of the optimal control vector

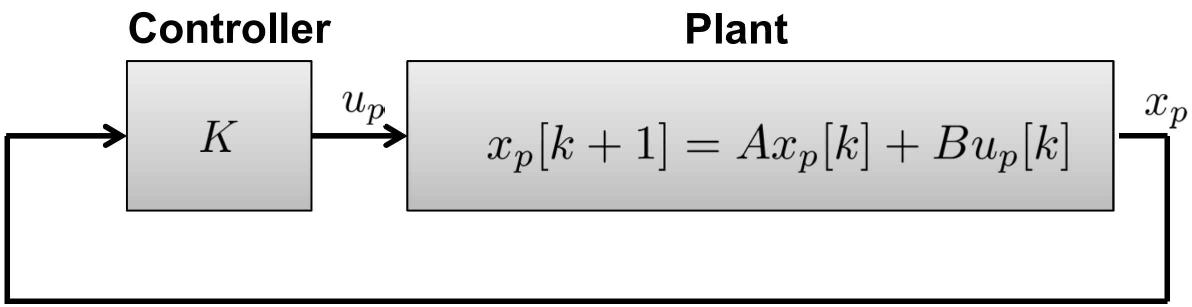
$$U^* = \begin{bmatrix} u^*[0] \\ \vdots \\ u^*[N-1] \end{bmatrix},$$

$$u^*[0] = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} U^* = \underbrace{-\begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} G^{-1} F}_{K} \, x_0$$

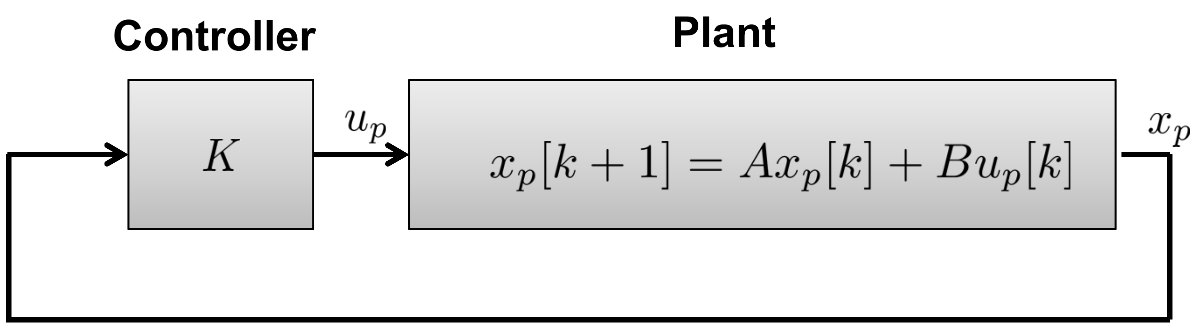
- At every time step i, apply optimal control law

$$x_0 = x_p[i] \qquad \boxed{u_p[i] = K x_p[i]}$$

Unconstrained MPC
= Linear state feedback



Unconstrained MPC: Closed-loop Response



Matlab Code

```
A = …; B=…;      % plant model
Ts = …;          % [s] sample time
n = size(A,2); % number of states
K = …;           % feedback gain

% closed-loop system
A_CL = A+B*K;
B_CL = zeros(n,1);
C_CL = eye(n);
sys_CL = ss( A_CL, B_CL, C_CL,[],Ts);

[y,t,x] = initial(sys_CL , xp0, Tsim);
```

Setup model and feedback gain

Closed-loop system

$$x_p[k+1] = A_{CL}x_p[k] + B_{CL}u_p[k]$$
$$y_p[k] = C_C L x_p[k]$$
$$A_{CL} = A + BK, B_{CL} = 0, C_{CL} = I$$

Response of closed-loop system with initial state (`xp0`) during `Tsim` seconds

Remark: ss(…), initial(…), zeros(..), eye(…), were already introduced in previous lectures

Unconstrained MPC: Matlab Example

Consider the following discrete-time model of a mechanical system

$$x_p[k+1] = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_p + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_p, \;\; y_p = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_p \qquad \left\{ \begin{aligned} & u_p = \text{acceleration } [\text{m/s}^2] \\ & x_p = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T, x_1 = \text{position}[\text{m}] \\ & \qquad\qquad\qquad\; x_2 = \text{velocity } [\text{m/s}] \\ & T_s = 1s \end{aligned} \right.$$

Goal: regulate position and velocity $(x_1, x_2 \to 0)$ using unconstrained MPC

Write a Matlab script that:

a) computes prediction matrices S_x, S_u, assuming

$$N = 2, \quad R = 1/10, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

b) computes cost matrices F, G

c) computes feedback gain K

d) simulates closed-loop response for initial state $x_p[0] = \begin{bmatrix} 10 & 0 \end{bmatrix}^T$ and plot results

e) repeats d) with $R = 6$

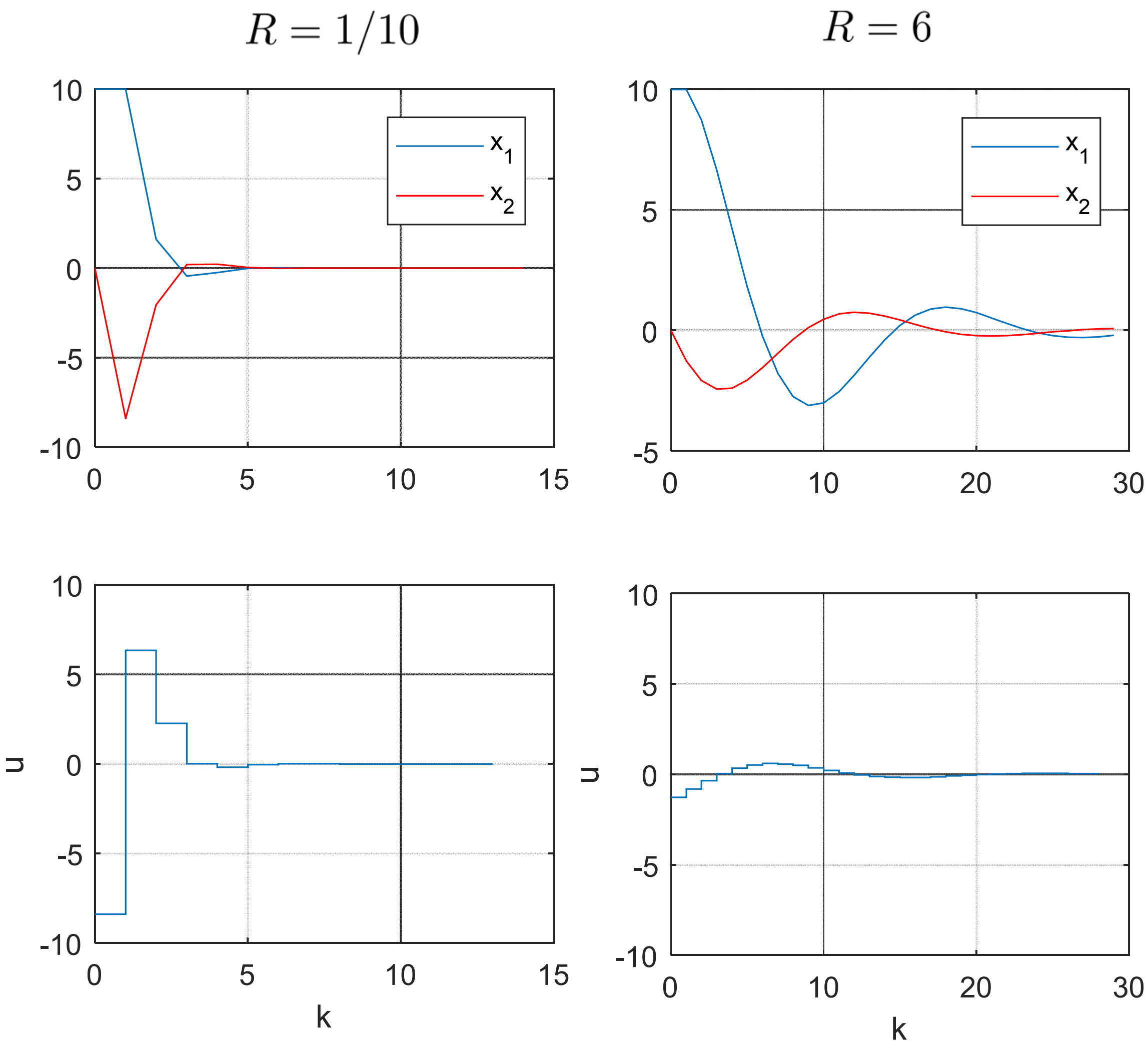
Unconstrained MPC: Example (II)

Solution:

$$S_x = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 2 \\ 0 & 1 \end{bmatrix}, \quad S_u = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix},$$

$$F = \begin{bmatrix} 1 & 3 \\ 0 & 1 \end{bmatrix}, \quad G = \begin{bmatrix} 2.1 & 1.0 \\ 1.0 & 1.1 \end{bmatrix},$$

$$K = \begin{bmatrix} -0.8397 & -1.7557 \end{bmatrix}$$



Constrained MPC

Problem Formulation

A) cost function

$$\min\ J(u[k]) = \sum_{k=0}^{N-1} x^T[k]Qx[k] + u^T[k]Ru[k] + x^T[N]Px[N]$$

B) prediction model

$$\text{s.t.}\quad x[0] = x_0$$
$$x[k+1] = Ax[k] + Bu[k]$$

C) input constraints

$$u_{min} \leq u[k] \leq u_{max}, \quad k = 0, \dots, N-1$$

D) parameters

$Q =$ state penalty (symmetric and positive semi-definite, $Q = Q^T \geq 0$)

$R =$ input penalty (symmetric and positive definite, $R = R^T > 0$)

$P =$ final state penalty (symmetric and positive semi-definite, $P = P^T \geq 0$)

$N =$ prediction horizon $\quad x_0 =$ initial state

Goal: compute control sequence $u[0], u[1], \dots, u[N-1]$ that minimize the cost $J(.)$

$$\Updownarrow$$

$$u^*[k] = \arg\min\ J(u[k])\ \text{ s.t. } u_{min} \leq u[k] \leq u_{max}$$

Remark: no output constraints ($\mathcal{Y} = \mathbb{R}^{n_y}$)

Constrained MPC: Solution

1) **Reformulate** optimization problem using U as decision variable

$$\boxed{\begin{aligned} &\min J(U) = x_0^T \Gamma x_0 + 2U^T F x_0 + U^T G U \\ &\text{s.t.} \quad U_{min} \le U \le U_{max} \end{aligned}}$$

$$U = \begin{bmatrix} u[0] \\ \vdots \\ u[N-1] \end{bmatrix},$$

$$U_{min} = \begin{bmatrix} u_{min} \\ \vdots \\ u_{min} \end{bmatrix}, \quad U_{max} = \begin{bmatrix} u_{max} \\ \vdots \\ u_{max} \end{bmatrix}, \quad \Gamma, F, G \text{ are the same as in the unconstained MPC}$$

Remark: closed-form solution not easy to obtain due to inequality constraints

2) Compute U^* using a **numerical solver**

Constrained MPC: quadprog (I)

- **Optimization Toolbox** provides several functions to solve constrained optimization problems
  - For quadratic programming (QP) problems (i.e. quadratic cost and linear constraints) one can use ***quadprog***
  - (simplified) API:

```
zOpt = quadprog(H,f,E,b)
```

optimal solution

$$z^* = \arg\min_{Ez \leq b} V(z)$$

parameters of the optimization problem

$$\min_z V(z) = \frac{1}{2} z^T H z + f^T z$$
$$\text{s.t.} \quad Ez \leq b$$

$$z \in \mathbb{R}^n \text{ , } n = \text{number of decision variables}$$
$$H \in \mathbb{R}^{n \times n} = \text{positive definite matrix}$$
$$f \in \mathbb{R}^n = \text{linear cost parameter}$$
$$E \in \mathbb{R}^{m \times m}, \quad b \in \mathbb{R}^m = \text{constraints parameters}$$
$$m = \text{number of inequality constraints}$$

Constrained MPC: quadprog (II)

Constrained MPC Problem

$$\min J(U) = x_0^T \Gamma x_0 + 2U^T F x_0 + U^T G U$$
$$\text{s.t.} \quad U_{min} \leq U \leq U_{max}$$

$$\Leftrightarrow$$

quadprog

$$\min_z V(z) = \frac{1}{2} z^T H z + f^T z$$
$$\text{s.t.} \quad Ez \leq b$$

change of variable: $z = U$

cost function: $H = 2G,\ f = 2Fx_0$

constraints: $E = \begin{bmatrix} I \\ -I \end{bmatrix}, \quad b = \begin{bmatrix} U_{max} \\ -U_{min} \end{bmatrix}$

$I = N \times N$ identity matrix

Constrained MPC: Receding Horizon Control

Idea: repeatedly solve MPC problem at each sample time in order to bring feedback action into the controller

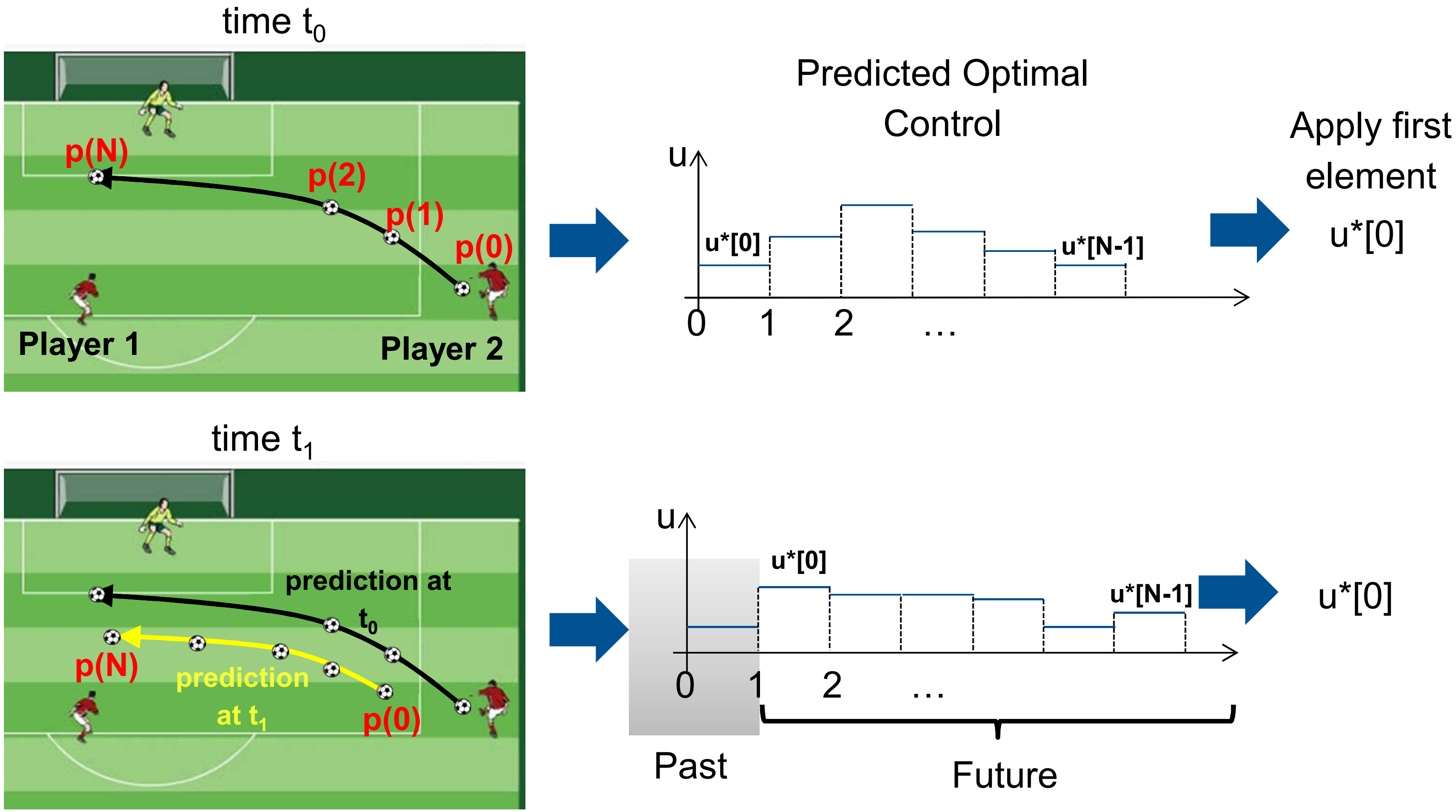
<u>**Algorithm**</u>

1) **measure** the state $x_p[i]$ at time instant *i* ($x_p[i] \rightarrow x_0$)
2) **update** cost vector $f = 2Fx_0$
3) **compute** optimal control U^*
4) **apply** first element $u^*[0]$ of U^* to system
5) **wait** for the new sample time i+1

Source: F. Borrelli, A. Bemporad, M. Morari “Predictive Control for linear and hybrid systems “ (2014)

Constrained MPC: Receding Horizon Control

Idea: repeatedly solve MPC problem at each sample time in order to bring feedback action into the controller



e.g. wind gust deviates ball from the trajectory predicted at time t_0; player 1 needs to re-adjust actions

Constrained MPC: Closed-loop Simulation

Matlab Code

```
A = …; B=…;   % prediction model
H = …; f=…;   % QP parameters (cost)
E = …; b=…; % QP parameters (constr.)
Nsim = 30;    % simulation steps

xp    = [10; 0]; % initial state


for i=1:Nsim-1
      x0 = xp;
      f = 2*(F*x0);
      Uopt= quadprog(H,f,E,b);


      up = U_opt(1);
      xp = A*xp + B*up;
end
```

Setup MPC problem

Algorithm

1) **measure** the state $x_p[i]$ at time instant *i*

2) **update** cost vector $f = 2Fx_0$

3) **compute** optimal control U^*

4) **apply** first element $u^*[0]$ of U^* to system

Constrained MPC: Example

Consider the following discrete-time model of a mechanical system

$$x_p[k+1] = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix} x_p + \begin{bmatrix}0\\ 1\end{bmatrix} u_p, \;\; y_p = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} x_p \quad \begin{cases} u_p = \text{acceleration } [\text{m}/\text{s}^2] \\ x_p = \begin{bmatrix}x_1 & x_2\end{bmatrix}^T, x_1 = \text{position}[\text{m}] \\ \qquad\qquad\qquad\;\; x_2 = \text{velocity } [\text{m}/\text{s}] \\ T_s = 1s \end{cases}$$

Goal: regulate position and velocity $(x_1, x_2 \to 0)$ using MPC

Write a Matlab script that

a) simulates closed-loop response of unconstrained MPC using **`quadprog`**

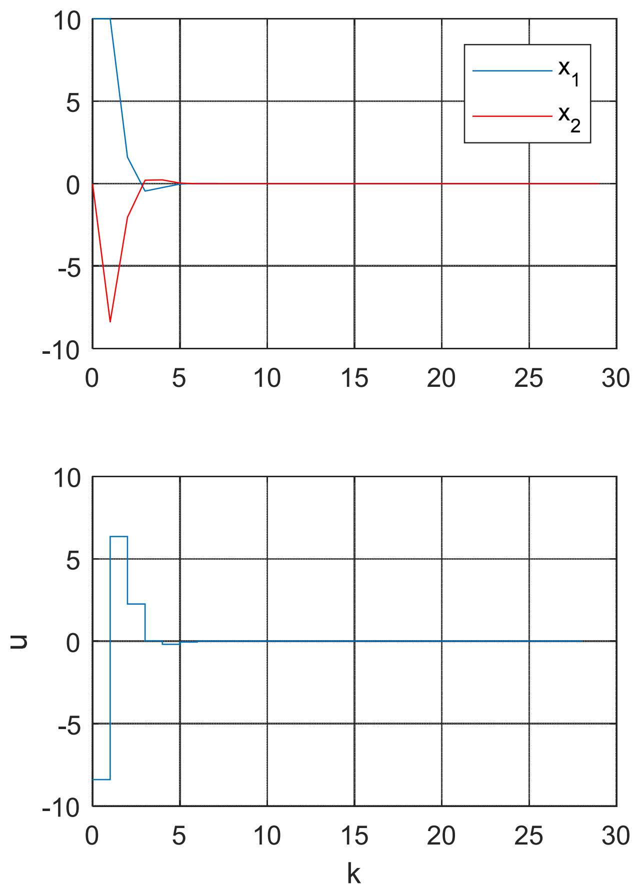
$$N = 2, \quad R = 1/10, \quad Q = \begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}, \quad P = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \qquad x_p[0] = \begin{bmatrix}10 & 0\end{bmatrix}^T$$

b) repeats a) with the input constraint

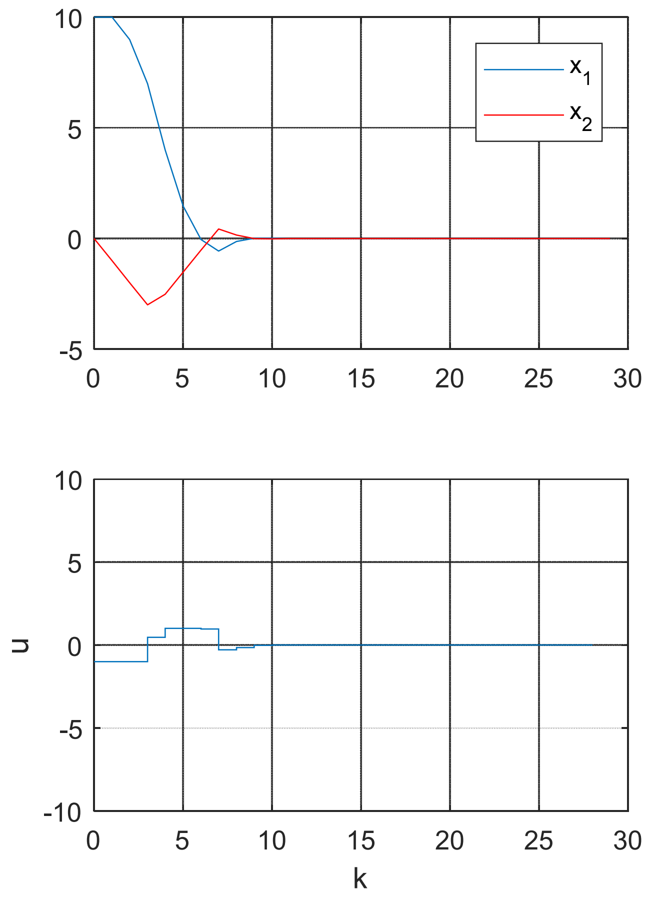
$$-1 \leq u[k] \leq 1$$

Constrained MPC: Example (II)

a) unconstrained MPC



b) constrained MPC



MPC Toolbox

MPC Toolbox: Key Ingredients

prediction model
(A, B, C)
initial state (x_0)

prediction horizon (N)
control horizon (N_u)
weights (Q_k, R_k, E_k)

rate limits $(\Delta u_{min}, \Delta u_{max})$
control limits (u_{min}, u_{max})
output limits (y_{min}, y_{max})

$$\min \sum_{k=1}^{N} (y[k] - r[k])^T Q_k (y[k] - r[k]) + \sum_{k=0}^{N-1} u^T[k] R_k u[k] + \Delta u^T[k] E_k \Delta u[k]$$

s.t.

$$x[0] = x_0$$

$$x[k+1] = Ax[k] + Bu[k], \quad k = 0, \ldots, N-1$$

$$y[k] = Cx[k], \quad k = 0, \ldots, N$$

$$y_{min} \leq y[k] \leq y_{max}, \quad k = 0, \ldots, N-1$$

$$u_{min} \leq u[k] \leq u_{max}, \quad k = 0, \ldots, N_u - 1$$

$$\Delta u_{min} \leq \Delta u[k] \leq \Delta u_{max}$$

$$\Delta u[k] = u[k] - u[k-1], k = 0, \ldots, N-1$$

$$\Delta u[k] = 0, \quad k = N_u \ldots, N-1$$

Remarks:

- reference value (r) employed in tracking applications
- weights (Q_k, R_k, E_k) can vary in time
- $N_u < N$ reduces number of decision variables and computational effort

optimal control, $u^*[0], \ldots, u^*[N_u - 1]$

MPC Toolbox: Overview

	Key Matlab Commands*
1) Setup prediction model	`model=ss(...)% create LTI prediction model`
2) Setup MPC Object	`MPCobj = mpc(model,...); % create MPC controller`
3) Setup Weights and Constraints	`MPCobj.W.Input= …  % modify MPC input weights` `MPCobj.W.Output = …% modify MPC output weights`
4) Compute Optimal Control & Closed-loop Simulation	`mpcmove(MPCobj,...) % computes MPC control action`

* Only a <u>sub-set of the MPC Toolbox is considered</u> (advanced features are not used in this introductory lecture)

MPC Toolbox: Setup Prediction Model (I)

Discrete-time Prediction Model

$$
\begin{aligned}
x[k+1] &= Ax[k] + Bu[k] \\
y[k] &= Cx[k] + Du[k]
\end{aligned}
$$

MPC Problem

$$
\begin{aligned}
\min \quad & J = \ldots \\
\text{s.t.} \quad & x[0] = x_0 \\
& \boxed{\begin{aligned} x[k+1] &= Ax[k] + Bu[k], \quad k = 0, \ldots, N-1 \\ y[k] &= Cx[k], \quad k = 0, \ldots, N \end{aligned}} \\
& \vdots
\end{aligned}
$$

Matlab Code

```
plantDisc = ss(A,B,C,D,Ts);

% (A,B,C,D) = discrete-time SS model
% Ts = sample time [s]
% plantDisc = discrete-time state
              space model
```

NOTE: MPC Toolbox assumes D=0
If D!=0 → create an auxiliary state with delayed output

MPC Toolbox: Setup Prediction Model (II)

Continuous-time Prediction Model

$$\dot{x}(t) = A_c x(t) + B_c u(t)$$
$$y(t) = C_c x(t) + D_c u(t)$$

Discrete-time Prediction Model

$$x[k+1] = Ax[k] + Bu[k]$$
$$y[k] = Cx[k] + Du[k]$$

MPC Problem

$$\min \quad J = \dots$$

$$\text{s.t.} \quad x[0] = x_0$$

$$x[k+1] = Ax[k] + Bu[k], \quad k = 0, \dots, N-1$$
$$y[k] = Cx[k], \quad k = 0, \dots, N$$

$$\vdots$$

Matlab Code

```
plantCont = ss(Ac,Bc,Cc,Dc);
% (Ac,Bc,Cc,Dc) = continuous-time SS
                  model

plantDisc = c2d(plantCont, Ts);
% plantCont = continuous-time SS model
% Ts = sample time [s]
% plantDisc = discrete-time SS model
```

MPC Toolbox: Setup MPC Object

MPC Problem

$$\min \sum_{k=1}^{\boxed{N}} (y[k]-r[k])^T Q_k (y[k]-r[k]) + \sum_{k=0}^{\boxed{N}-1} u^T[k] R_k u[k]$$

$$\text{s.t.} \quad x[0] = x_0$$

$$x[k+1] = Ax[k] + Bu[k], \quad k = 0, \ldots, \boxed{N}-1$$

$$y[k] = Cx[k], \quad k = 0, \ldots, \boxed{N}$$

$$y_{min} \leq y[k] \leq y_{max}, \quad k = 0, \ldots, \boxed{N}-1$$

$$u_{min} \leq u[k] \leq u_{max}, \quad k = 0, \ldots, \boxed{N_u}-1$$

$$\vdots$$

Matlab Code

```
MPCobj = mpc(plant, Ts, N, Nu);

% INPUTS
% plant= prediction model
% Ts = sample time
% N = prediction horizon
% Nu = control horizon (1<=Nu<=N)

% OUTPUT
% MPCobj = MPC controller object
```

MPC Toolbox: Setup Weights (I)

```
MPCobj=mpc(…);
```

Field	Description	Symbol	Size	Default value
`MPCobj.W.Output`	Output weight over prediction horizon	q	(1-to-N) by ny array	ones(1,nu)
`MPCobj.W.Input`	Input weights over prediction horizon	r_u	(1-to-N) by nu array	zeros(1,nu)
`MPCobj.W.InputRate`	Weight of Input increments over prediction horizon	r_e	(1-to-N) by nu array	0.1*ones(1,nu)

nu=number of control inputs, ny=number of outputs

Cost Function: $$\min \sum_{k=1}^{N} (y[k]-r[k])^T \boxed{Q_k} (y[k]-r[k]) + \sum_{k=0}^{N-1} u^T[k] \boxed{R_k} u[k] + \Delta u^T[k] \boxed{E_k} \Delta u[k]$$

Constant Weights $(q, r_u, r_e \text{ are row vectors})$

$$Q_1 = Q_2 = \ldots = Q_N = \mathrm{diag}(q)^2$$
$$R_0 = R_1 = \ldots = R_{N-1} = \mathrm{diag}(r_u)^2$$
$$E_0 = E_1 = \ldots = E_{N-1} = \mathrm{diag}(r_e)^2$$

Time-varying weights $(q, r_u, r_e \text{ are arrays})$

$$Q_1 = \mathrm{diag}(q(1,:))^2, \ldots, Q_N = \mathrm{diag}(q(N,:))^2$$
$$R_0 = \mathrm{diag}(r_u(1,:))^2, \ldots, R_{N-1} = \mathrm{diag}(r_u(N-1,:))^2$$
$$E_0 = \mathrm{diag}(r_e(1,:))^2, \ldots, R_{N-1} = \mathrm{diag}(r_e(N-1,:))^2$$

Remark: only diagonal elements of weight matrices (Q_k, R_k, E_k) are specified in the MPC controller object

MPC Toolbox: Setup Weights (II)

MPC Problem	Matlab Code

$$\min \sum_{k=1}^{N} (y[k]-r[k])^T \boxed{Q_k} (y[k]-r[k]) + \sum_{k=0}^{N-1} u^T[k] \boxed{R_k} u[k] + \Delta u^T[k] \boxed{E_k} \Delta u[k]$$

<u>2D output (ny=2), constant weights</u>

$$Q_1 = Q_2 = \ldots = Q_N = \mathrm{diag}(q)^2$$
$$q = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

```
q = [1 2];
MPCobj.W.Output = q;
```

<u>2D output (ny=2), time-varying weights (N=3)</u>

$$Q_1 = \mathrm{diag}(\begin{bmatrix} 1 & 2 \end{bmatrix})^2$$
$$Q_2 = \mathrm{diag}(\begin{bmatrix} 1 & 1 \end{bmatrix})^2$$
$$Q_3 = \mathrm{diag}(\begin{bmatrix} 1 & 0.1 \end{bmatrix})^2$$

```
MPCobj.W.Output = [1 2;
                   1 1;
                   1 0.1];
```

<u>Scalar input (nu=1), constant weights</u>

$$R_0 = R_1 = \ldots = R_{N-1} = c^2$$
$$E_0 = E_1 = \ldots = E_{N-1} = d^2$$
$$c = 10, d = 5$$

```
c = 10;
MPCobj.W.Input=c;
d = 5;
MPCobj.W.InputRate=d;
```

MPC Toolbox: Setup Output Constraints

```
MPCobj=mpc(…);
```

Field	Description	Symbol	Default value
`MPCobj.OV(i).Min`	Lower-bound of output *i*	$y_{min,i}$	-Inf
`MPCobj.OV(i).Max`	Upper-bound of output *i*	$y_{max,i}$	+Inf

Note: $i = 1, \ldots, n_y$; OV= Output Variable

Output Constraints (vector)

$$y_{min} \le y[k] \le y_{max}$$

$\Leftrightarrow$

Output Constraints (elementwise)

$$\begin{bmatrix} y_{min,1} \\ \vdots \\ y_{min,n_y} \end{bmatrix} \le \begin{bmatrix} y_1[k] \\ \vdots \\ y_{n_y}[k] \end{bmatrix} \le \begin{bmatrix} y_{max,1} \\ \vdots \\ y_{max,n_y} \end{bmatrix}$$

MPC Problem

example, ny=3

$$\begin{bmatrix} -1 \\ 0 \end{bmatrix} \le \begin{bmatrix} y_1[k] \\ y_3[k] \end{bmatrix} \le \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Matlab Code

```
MPCobj.OV(1).Min = -1;
MPCobj.OV(1).Max =  1;
MPCobj.OV(3).Min =  0;
MPCobj.OV(3).Max =  2;
```

MPC Toolbox: Setup Input Constraints

`MPCobj=mpc(…);`

Field	Description	Symbol	Default value
`MPCobj.MV(i).Min`	Lower bound of control i	$u_{min,i}$	-Inf
`MPCobj.MV(i).Max`	Upper bound of control i	$u_{max,i}$	+Inf
`MPCobj.MV(i).RateMin`	Lower rate-limit bound of control i	$\Delta u_{min,i}$	-Inf
`MPCobj.MV(i).RateMax`	Upper rate-limit bound of control i	$\Delta u_{max,i}$	+Inf

Note: $i = 1, \ldots, n_u$; MV= Manipulated variable

Input Constraints (vector)

$$u_{min} \leq u[k] \leq u_{max}$$
$$\Delta u_{min} \leq \Delta u[k] \leq \Delta u_{max}$$

$\Leftrightarrow$

Input Constraints (elementwise)

$$\begin{bmatrix} u_{min,1} \\ \vdots \\ u_{min,n_u} \end{bmatrix} \leq \begin{bmatrix} u_1[k] \\ \vdots \\ u_{n_u}[k] \end{bmatrix} \leq \begin{bmatrix} u_{max,1} \\ \vdots \\ u_{max,n_u} \end{bmatrix}, \quad \begin{bmatrix} \Delta u_{min,1} \\ \vdots \\ \Delta u_{min,n_u} \end{bmatrix} \leq \begin{bmatrix} \Delta u_1[k] \\ \vdots \\ \Delta u_{n_u}[k] \end{bmatrix} \leq \begin{bmatrix} \Delta u_{max,1} \\ \vdots \\ \Delta u_{max,n_u} \end{bmatrix}$$

MPC Problem

<u>example, nu=1</u>

$$-10 \leq u[k] \leq 10$$
$$-100 \leq \Delta u[k] \leq 100$$

Matlab Code

```
MPCobj.MV(1).Min = -10;
MPCobj.MV(1).Max =  10;
MPCobj.MV(1).RateMin =  -100;
MPCobj.MV(1).RateMax =   100;
```

MPC Toolbox: Compute Optimal Control

1) Definition of MPC controller state

```
MPCstate=mpcstate(MPCobj);
```

Field	Description	Symbol
`MPCstate.plant`	Current state of the plant	x_0
`MPCstate.LastMove`	Control input used in the last control interval	$u[-1]$

2) Computational of MPC's optimal control action

```
uOpt = mpcmove(MPCobj, MPCstate, ym, r);
    % INPUTS
    % MPCobj = MPC controller object
    % MPCstate = current MPC controller state
    % ym = [1 by ny] current output measurement
    % r  = [(1-to-N)by ny] reference value over prediction
           horizon
    % OUTPUT
    % uOpt = optimal control action (u*[0])
```

Notes:

- `mpcmove` also updates `MPCstate.plant` with expected state after application of `u*[0]`
- if `r` is a [1 by ny] row vector → constant reference used throughout the prediction

MPC Toolbox: Closed-loop Simulation

Example with scalar input (nu=1) and 2D output (ny=2) using mpcmove(.)

Matlab Code

```
MPCstate = mpcstate(MPCobj);
MPCstate.Plant = [5, 0];
r = [0,0];

Nsim = 30;
for i=1:Nsim
      y=C* MPCstate.Plant;
      u(i)=mpcmove(MPCobj, MPCstate,y,r)
end

stairs(u);
```

Create and initialize MPC controller state at $x_p[0] = \begin{bmatrix} 5 & 0 \end{bmatrix}^T$

Compute output

1) Compute optimal control action
2) Computes next state after application of optimal control action & update `MPCstate.Plant`

plot optimal control actions with stairstep graph

MPC Toolbox: Matlab Example

Consider the following discrete-time model of a mechanical system

$$x[k+1] = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \qquad y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x$$

$$\left\{ \begin{array}{ll} u = \text{acceleration } [\text{m/s}^2] & \\ x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T, & x_1 = \text{position}[\text{m}] \\ & x_2 = \text{velocity } [\text{m/s}] \\ -1 \leq u \leq 1, & T_s = 1s \end{array} \right.$$

Goal: regulate position and velocity $(x_1, x_2 \to 0)$

Write a Matlab script that:

a) creates a discrete-time, state-space prediction model

b) constructs a MPC object with

$$N = N_u = 2 \qquad R_k = 1/10 \quad Q_1 = \text{diag}(1,0) \quad Q_2 = \text{diag}(1,1) \quad E_k = 0$$

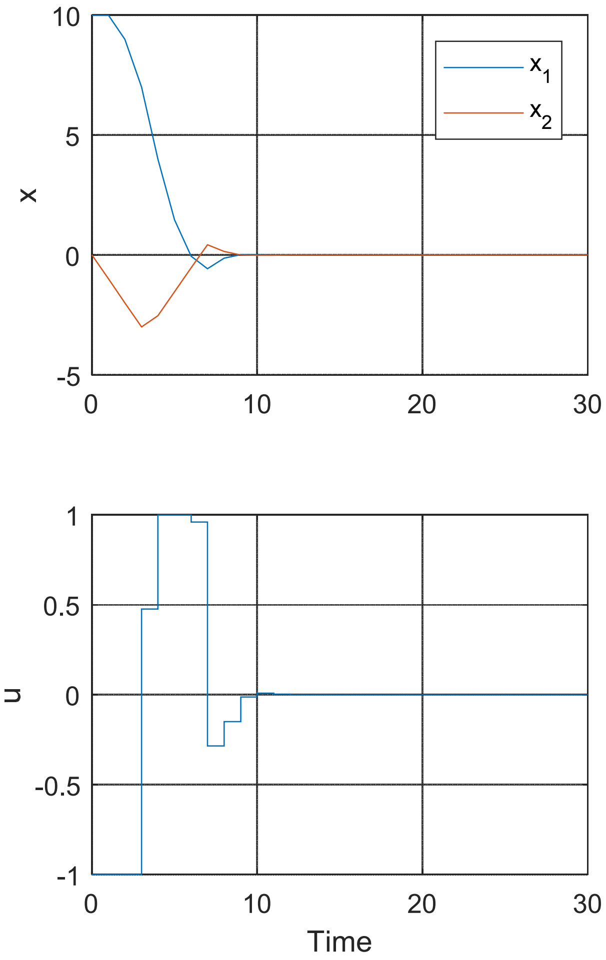
c) simulates closed-loop response during 30s with $x = \begin{bmatrix} 10 & 0 \end{bmatrix}^T$ and plots results

d) repeats c) with the following velocity constraint

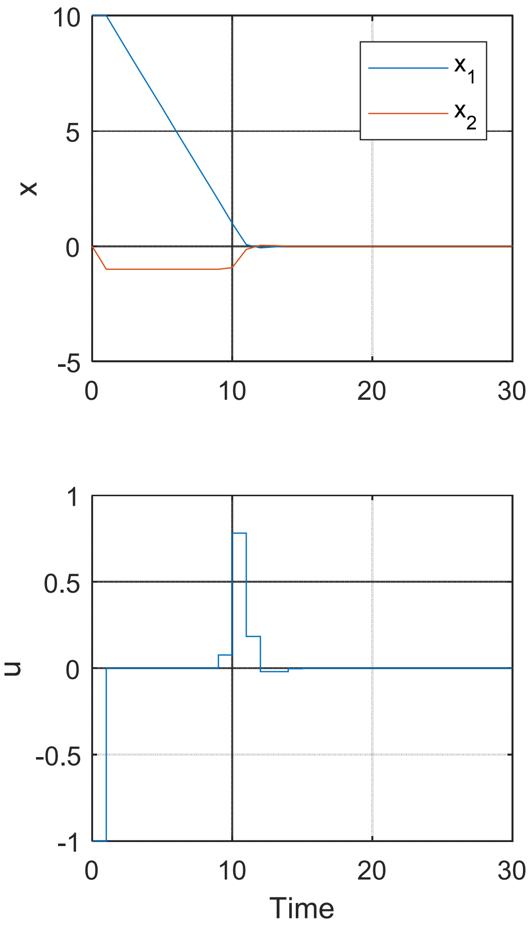
$$-1 \leq x_2[k] \leq 1$$

MPC Toolbox: Example (II)

c) results with input constraints



d) results with input & state constraints



MPC Toolbox: Closed-loop Simulation (II)

Simulation of closed-loop response of MPC with sim(.)

Matlab Code

```
simOptions = mpcsimopt(MPCobj)
    % MPCobj = MPC controller object
    % simOptions = MPC simulation options object


simOptions.PlantInitialState = x0; % initial plant state

[y,t,u]=sim(MPCobj,T,r, simOptions)
     % INPUTS
     % T = Number of simulation steps,
     % r  = [(1-to-T)by ny] reference value over simulation steps
     % simOptions =  mpcsimopt object with simulation settings

     % OUTPUT
     % y = [T by ny] = controlled plant outputs
     % t = [T by 1] = time sequence
     % u = [T by nu] = control inputs by the MPC controller
```

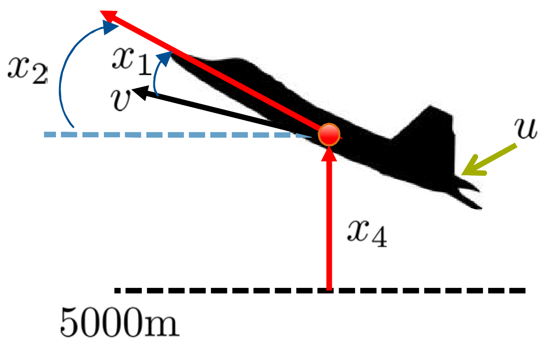
Example: Cessna Citation Aircraft

Consider the linearized model of an aircraft at altitude of 5000m and speed of 128.2 m/s:

$$\dot{x} = \begin{bmatrix} -1.28 & 0 & 0.98 & 0 & \\ & 0 & 0 & 1 & 0 \\ -5.43 & 0 & -1.84 & 0 & \\ -128.2 & 128.2 & 0 & 0 & \end{bmatrix} x + \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u, \qquad y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x$$

where

$$x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^T \qquad \begin{aligned} x_1 &= \text{angle of attack [rad]} \\ x_2 &= \text{pitch angle [rad]} \\ x_3 &= \text{pitch rate [rad/s]} \\ x_4 &= \text{altitude [m]} \end{aligned}$$



$$u = \text{elevator angle [rad]}$$

actuation constraints: $|u| \leq 0.262 \text{ rad} \quad (\pm 15 \deg)$

$$|\dot{u}| \leq 0.542 \text{ rad/s} \quad (\pm 30 \deg / s)$$

Goal: regulate pitch angle and altitude $(x_2, x_4 \rightarrow 0)$

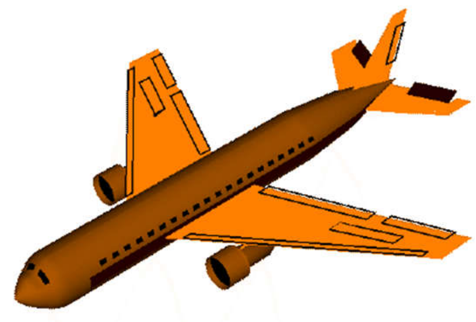


Image source: wikipedia.org

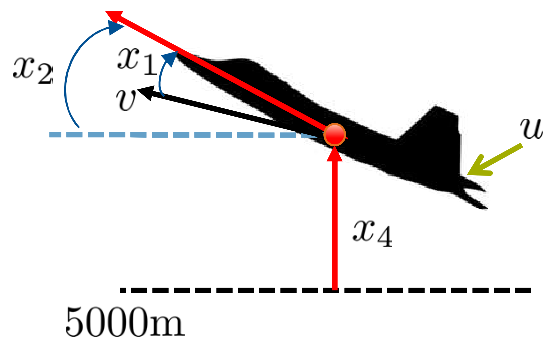
Source: Maciejowski (Example 2.7)

Example: Cessna Citation Aircraft (II)

Consider the linearized model of an aircraft at altitude of 5000m and speed of 128.2 m/s:

$$\dot{x} = \begin{bmatrix} -1.28 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.43 & 0 & -1.84 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u, \qquad y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x$$

$$|u| \leq 0.262 \text{ rad} \quad (\pm 15\,\text{deg})$$
$$|\dot{u}| \leq 0.542 \text{ rad/s} \quad (\pm 30\,\text{deg}\,/s)$$



Write a Matlab script that:

a) creates a discrete-time, state-space prediction model (sample time =0.25s)

b) constructs a MPC object with
$$N = N_u = 10 \quad Q_k = \text{diag}(0,1,0,1) \quad R_k = 10 \quad E_k = 0$$

c) simulates closed-loop response during 10s with $x(0) = \begin{bmatrix} 0 & 0 & 0 & 10 \end{bmatrix}^T$ and plot results

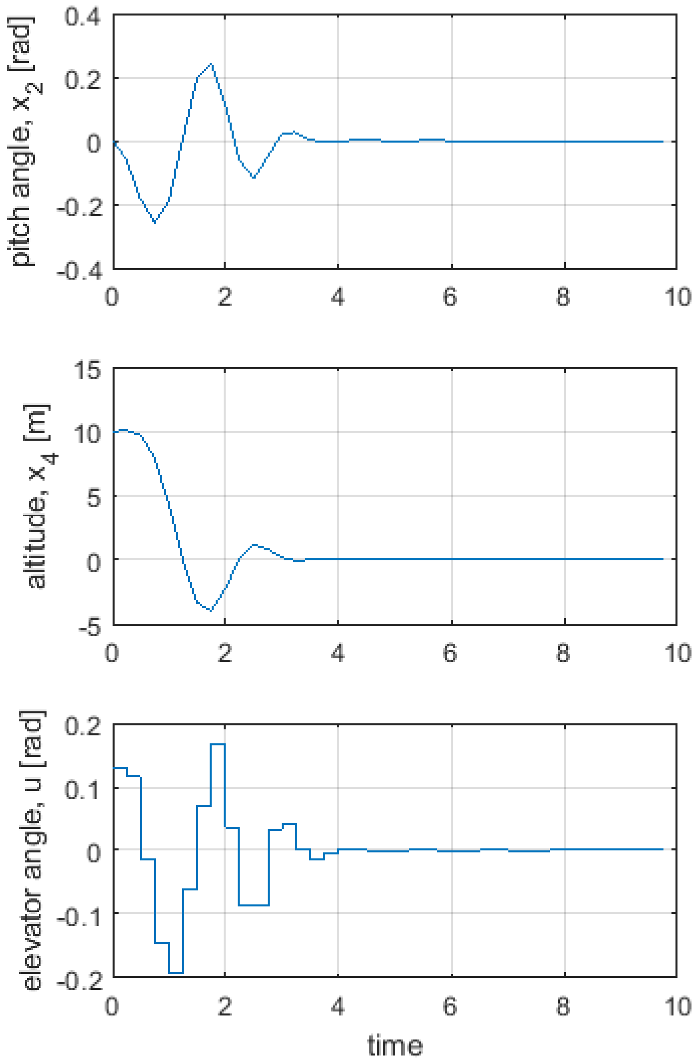
d) repeats c) with $R = 100000$

e) [optional] repeats c) with an altitude setpoint of 5100 m,
$$r = \begin{bmatrix} 0 & 0 & 0 & 100 \end{bmatrix}^T, \quad R = 100000$$

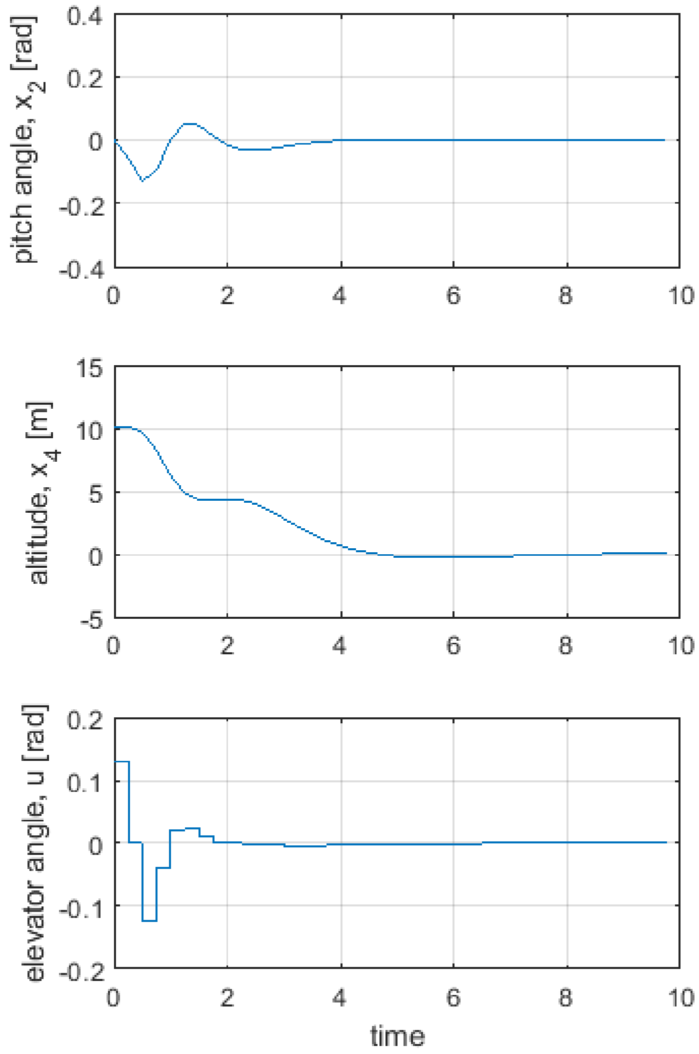
MATLAB

Example: Cessna Citation Aircraft (III)

$Q = \mathrm{diag}(0,1,0,1) \qquad R = 10$



$Q = \mathrm{diag}(0,1,0,1) \quad R = 100000$



Summary

- **Introduction to MPC**
  - Advanced control method to handle constraints & provide optimal performance
- **Unconstrained MPC:** analytical solution
- **MPC with input constraints**
  - Solution via *Optimization Toolbox*
- ***MPC with input&state constraints***
  - Solution via *MPC Toolbox*
    - Setup of prediction model, weights & constraints
    - Computing optimal control & closed-loop simulation
- **Examples**
  - Double integrator
  - Aircraft

Matlab API

operations with matrices,
`ss(), initial()`

QP Solver: `quadprog()`

Create MPC object: `mpc()`
Modify MPC Settings:

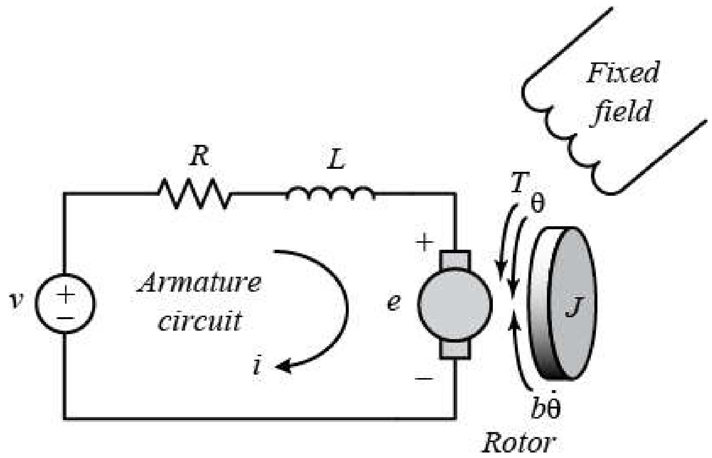
```
Mpcobj.W.Input = …
Mpcobj.MV(1).min =
…
```

Closed-loop simulation

```
mpcmove(), sim()
```

Homework: Velocity control of a DC Motor

Consider the following model of a DC-motor



Parameters:

J= moment of inertia of the rotor = 0.01 kg.m^2
b= motor viscous friction constant = 0.01 N.m.s
K_e= electromotive force constant = 0.01 V/rad/sec
K_t= motor torque constant = 0.01 N.m/Amp
R = electric resistance = 1 Ohm
L= electric inductance= 0.5 H
Note: $K_e = K_t = K$

State-space model

$$\frac{d}{dt}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} = \begin{bmatrix}-\frac{b}{J} & \frac{K}{J}\\ -\frac{K}{L} & -\frac{R}{L}\end{bmatrix}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{L}\end{bmatrix} v$$

Control input:

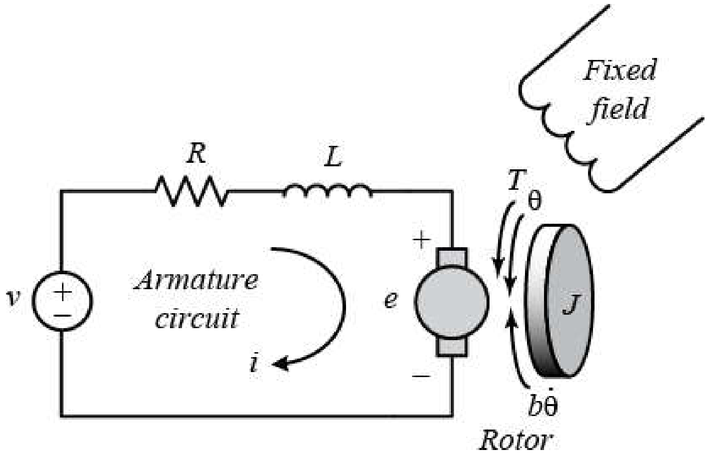
v = motor voltage [V]

States:

$\dot{\theta}$ = motor velocity, [rad/s]
i = motor current [A]

Homework: Velocity control of a DC Motor (II)

Consider the following model of a DC-motor,



$$\frac{d}{dt}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} = \begin{bmatrix}-\frac{b}{J} & \frac{K}{J}\\ -\frac{K}{L} & -\frac{R}{L}\end{bmatrix}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{L}\end{bmatrix} v$$

a) creates a discrete-time, state-space prediction model for the DC-motor (sample time =0.05s, $x = \begin{bmatrix}\dot{\theta} & i\end{bmatrix}^T$, $y = x$)

b) Simulate closed-loop response with **MPC toolbox,** reference output $r[k] = [10, 0]^T$ and

$$N = N_u{=}20 \quad Q_k = \text{diag}(10^3, 1) \quad R_k = 0.1 \quad E_k = 0 \quad x(0) = \begin{bmatrix}0 & 0\end{bmatrix}^T$$

c) plot states and control input

d) add the following constraints to the MPC: $-50\text{V} \leq v \leq 50\text{V},\ -20\text{A} \leq i \leq 20\text{A}$ and plot closed-loop response

To Probe further…

Recall motivation example:

Scalar, linear and discrete-time system: $x[k+1] = x[k] + u[k],$

Goal: $\text{find } u[0], u[1], \ldots, u[N-1] \text{ that minimizes cost } J = \sum_{k=0}^{N-1} x^2[k] + \rho u^2[k] + x^2[N],$

Questions addressed in this lecture:

- What if $N > 1$?
- What if x is a vector?

→ **Unconstrained MPC** (analytical solutions)

- What if the system has constraints? → **Constrained MPC** (numerical solutions)

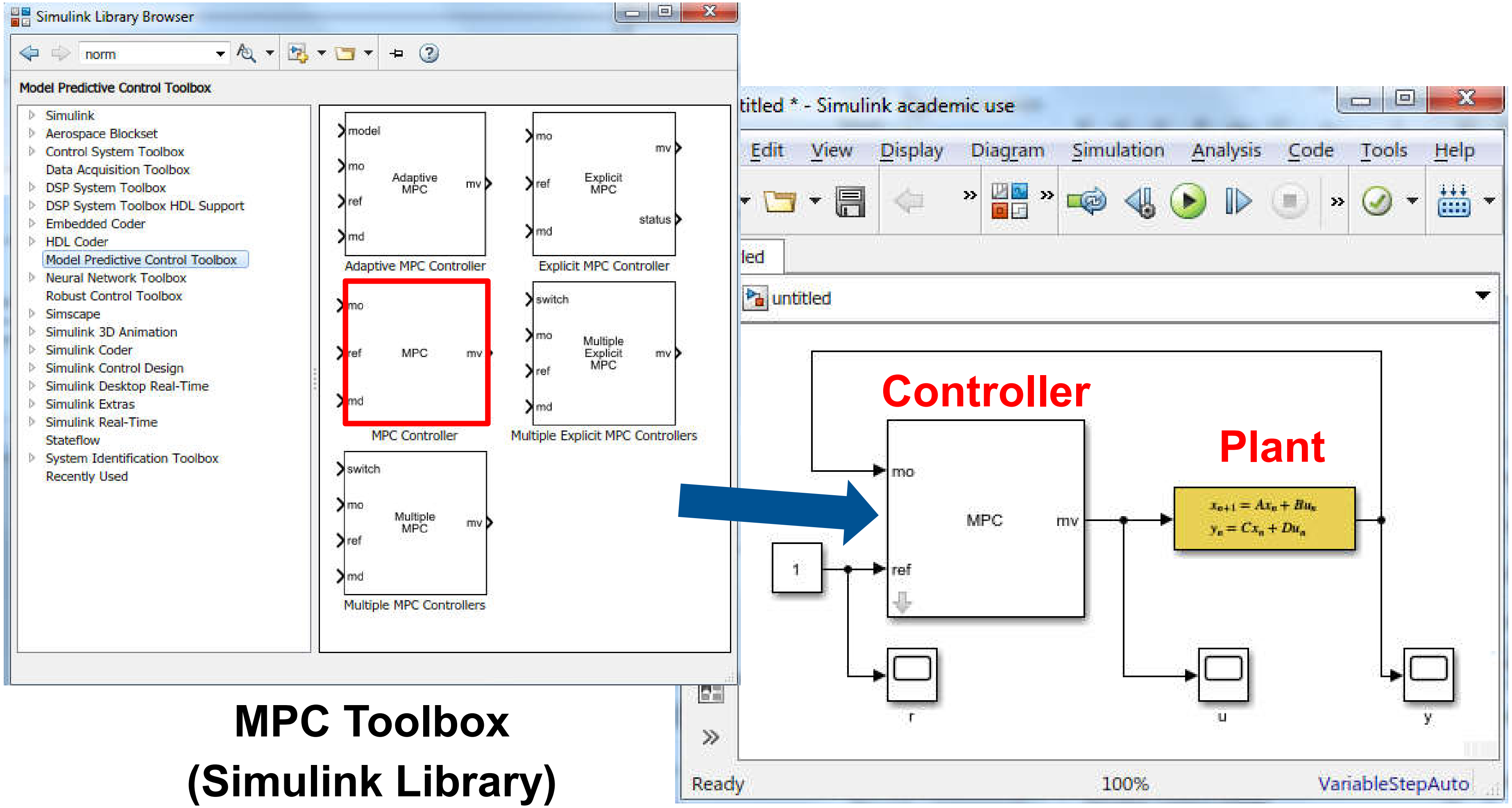
Advanced topics & Applications:

- What if plant model and/or cost function are nonlinear? **Nonlinear MPC…**
- What if plant model != prediction model? **Robust MPC …**
- What if plant model is stochastic? **Stochastic MPC …**
- What if I want to run MPC in an embedded system? **Real-time numerical optimization…**
- Where can I apply MPC? **Automotive, Power Electronics, UAVs, Buildings…**
- ….

To probe further: S. V. Raković, W. S. Levine (Ed.) *Handbook of Model Predictive Control, 2019*

Optional Topic:
MPC Toolbox in Simulink

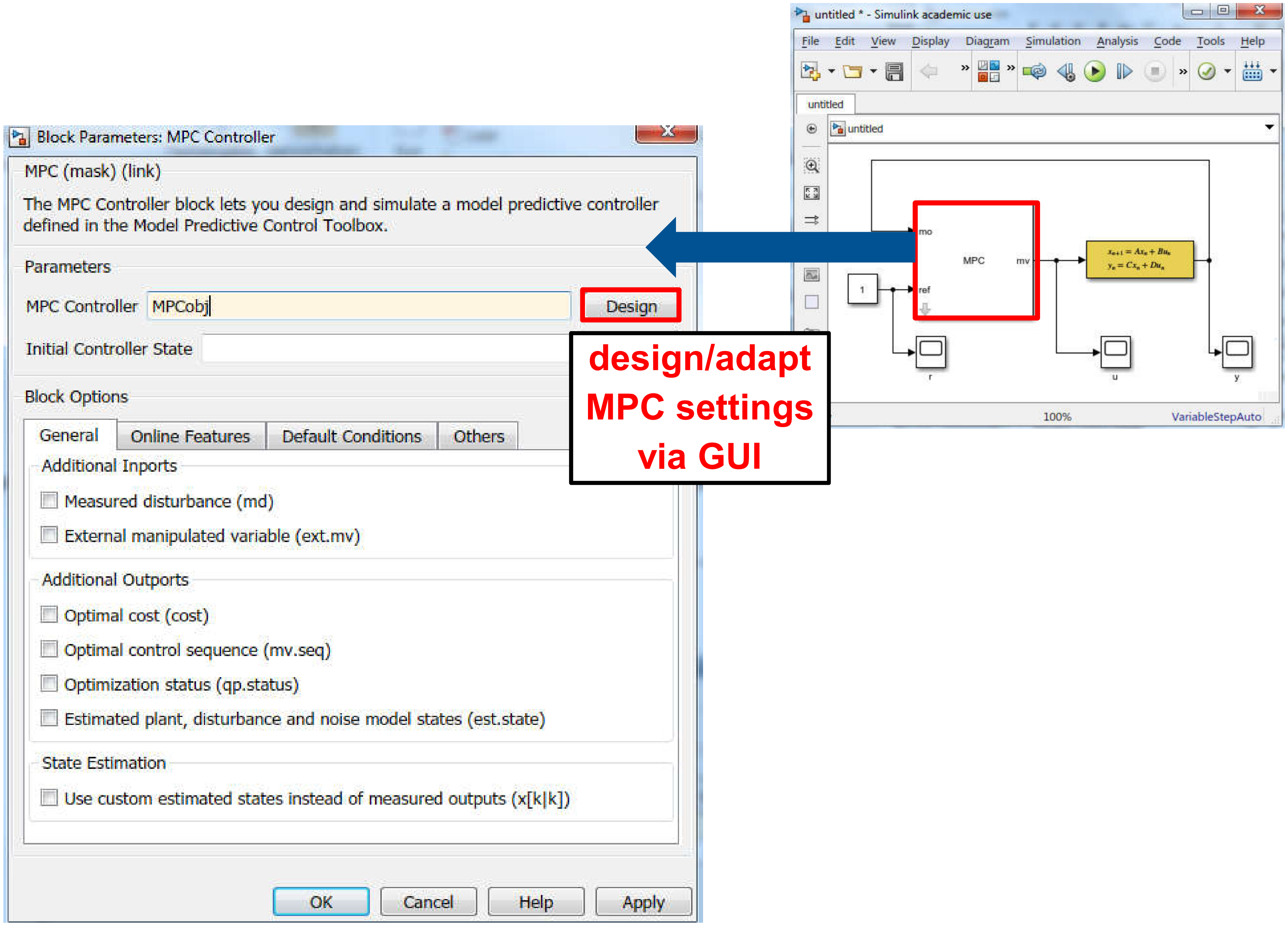
MPC Toolbox in Simulink



MPC Toolbox (Simulink Library)

Closed-Loop Simulation

MPC Toolbox in Simulink (II)

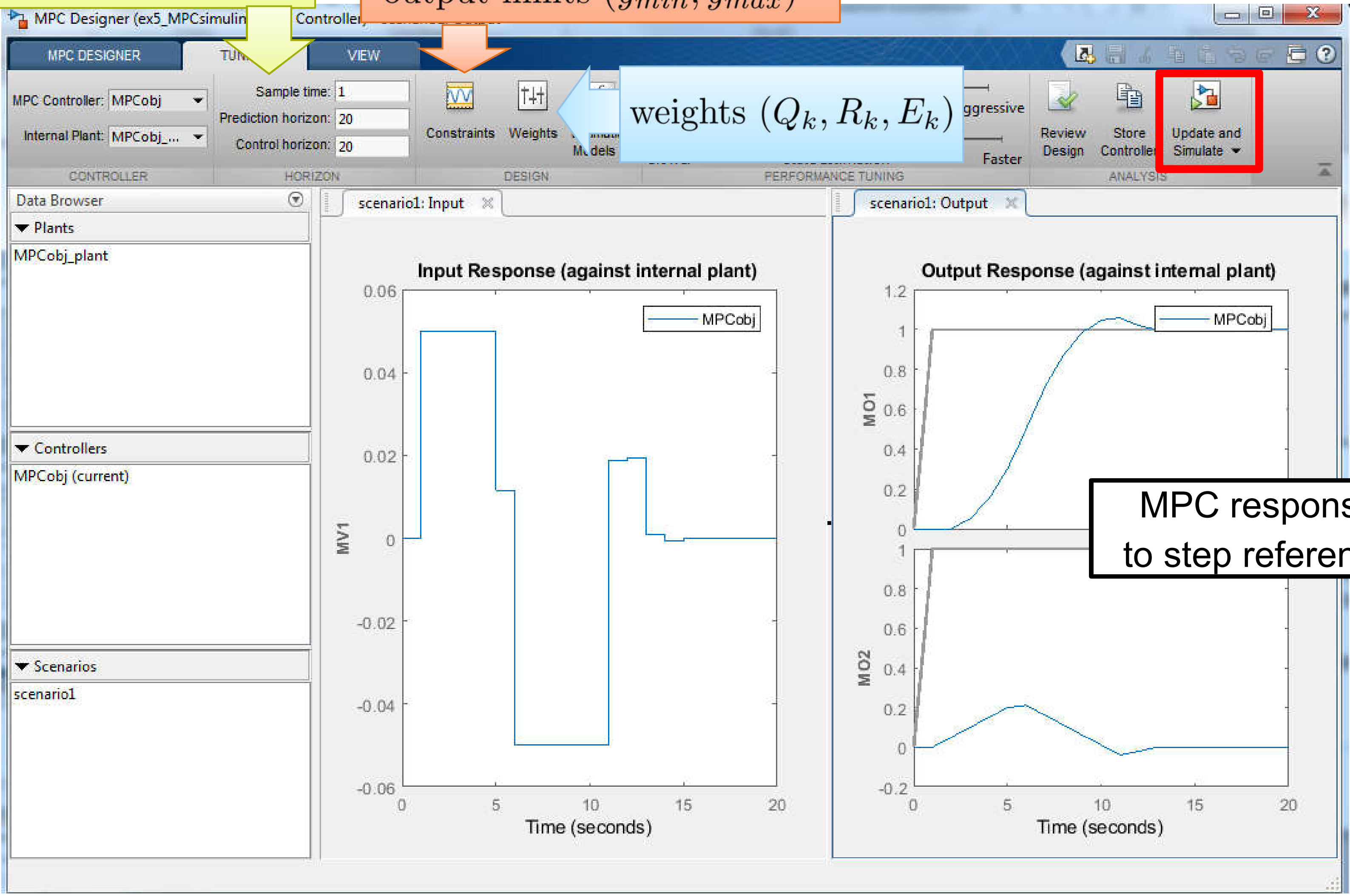


MPC Toolbox in Simulink (III)

prediction horizon (N)
control horizon (N_u)

rate limits $(\Delta u_{min}, \Delta u_{max})$
control limits (u_{min}, u_{max})
output limits (y_{min}, y_{max})

weights (Q_k, R_k, E_k)



MPC response to step reference

MPC Toolbox: Simulink Example

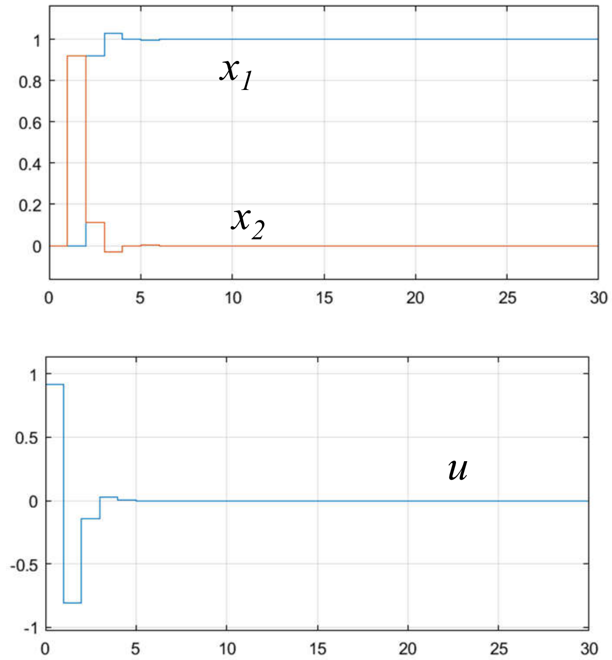
Consider the following discrete-time model of a mechanical system

$$x[k+1] = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \qquad y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x \qquad \begin{cases} u = \text{acceleration } [\mathrm{m/s^2}] \\ x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T, \; x_1 = \text{position}[\mathrm{m}] \\ \qquad\qquad\qquad x_2 = \text{velocity } [\mathrm{m/s}] \\ T_s = 1s \end{cases}$$

a) Write a Matlab script that creates
   a) discrete-time, state-space prediction model
   b) MPC object with $N = N_u$=20 (and default weights/constraints)
b) Simulate the closed-loop response of the system in **Simulink**
   - Simulation duration: 30s
   - Initial state: $x(0) = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$
   - Reference value: $r = [1, 0]^T$
c) Using **MPC Designer GUI**
   1. add input constraint $-0.05 \leq u \leq 0.05$, and simulate closed-loop response
   2. change the MPC weights $Q_k = \mathrm{diag}(1,0) \quad R_k = 20^2 \quad E_k = 0$
      and simulate closed-loop response

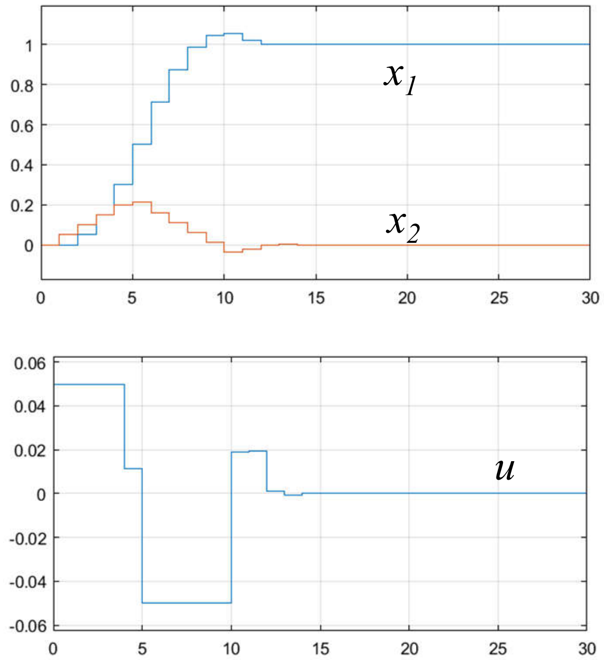
MPC Toolbox: Simulink Example (II)

Default settings (b)



time

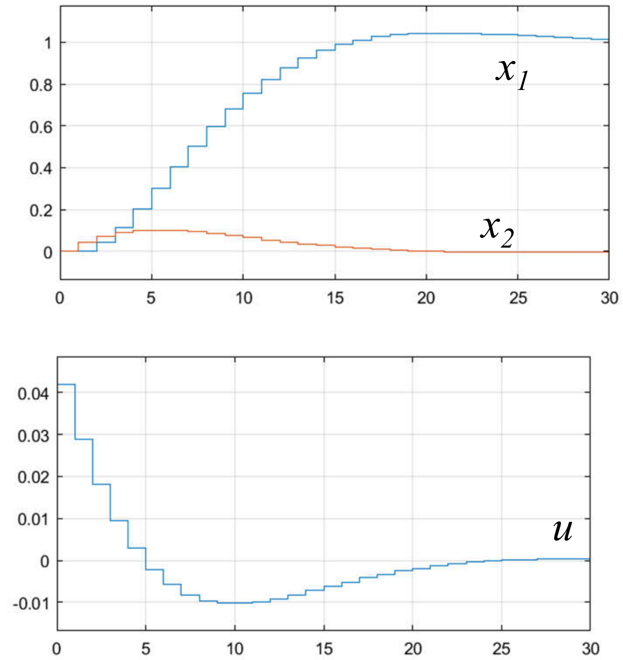
Default Settings
+ input constraints (c1)



time

$$-0.05 \leq u \leq 0.05,$$

Custom Settings
+ input constraints (c2)



time

$$-0.05 \leq u \leq 0.05,$$
$$R_k = 20^2$$
$$Q_k = \mathrm{diag}(1, 0)$$
$$E_k = 0$$