

LINEAR PARAMETER VARYING AND TIME-VARYING MODEL PREDICTIVE CONTROL

Alberto Bemporad - “Model Predictive Control” course - Academic year 2017/18

LINEAR PARAMETER-VARYING (LPV) MPC

LTI prediction
model

$$\begin{cases} x_{k+1} = A(p(t))x_k + B_u(p(t))u_k + B_v(p(t))v_k \\ y_k = C(p(t))x_k + D_v(p(t))v_k \end{cases} \quad x_0 = x(t)$$

Model depends on time t but does not change in prediction

quadratic
performance index

$$\min_U \sum_{k=0}^{N-1} \|W^y(y_k - r(t))\|^2 + \|W^u(u_k - u^{\text{ref}}(t))\|^2$$



$$\begin{aligned} \min_z & \frac{1}{2} z' H(p(t)) z + \theta'(t) F(p(t))' z \\ \text{s.t.} & G(p(t)) z \leq W(p(t)) + S(p(t)) \theta(t) \end{aligned}$$

constraints

$$\begin{cases} u_{\min} \leq u_k \leq u_{\max} \\ y_{\min} \leq y_k \leq y_{\max} \end{cases}$$

All QP matrices are
constructed on line

- LPV models can be obtained from linearization of nonlinear models or from black-box LPV system identification

LINEAR TIME-VARYING (LTV) MPC

LTV prediction model

$$\begin{cases} x_{k+1} = A_k(p(t))x_k + B_{uk}(p(t))u_k + B_{vk}(p(t))v_k & x_0 = x(t) \\ y_k = C_k(p(t))x_k + D_{vk}(p(t))v_k \end{cases}$$

Model depends on time t and prediction step k

quadratic performance index

$$\min_U \sum_{k=0}^{N-1} \|W^y(y_k - r(t))\|^2 + \|W^u(u_k - u^{\text{ref}}(t))\|^2$$



$$\begin{aligned} \min_z \quad & \frac{1}{2} z' H(p(t)) z + \theta'(t) F(p(t))' z \\ \text{s.t.} \quad & G(p(t)) z \leq W(p(t)) + S(p(t)) \theta(t) \end{aligned}$$

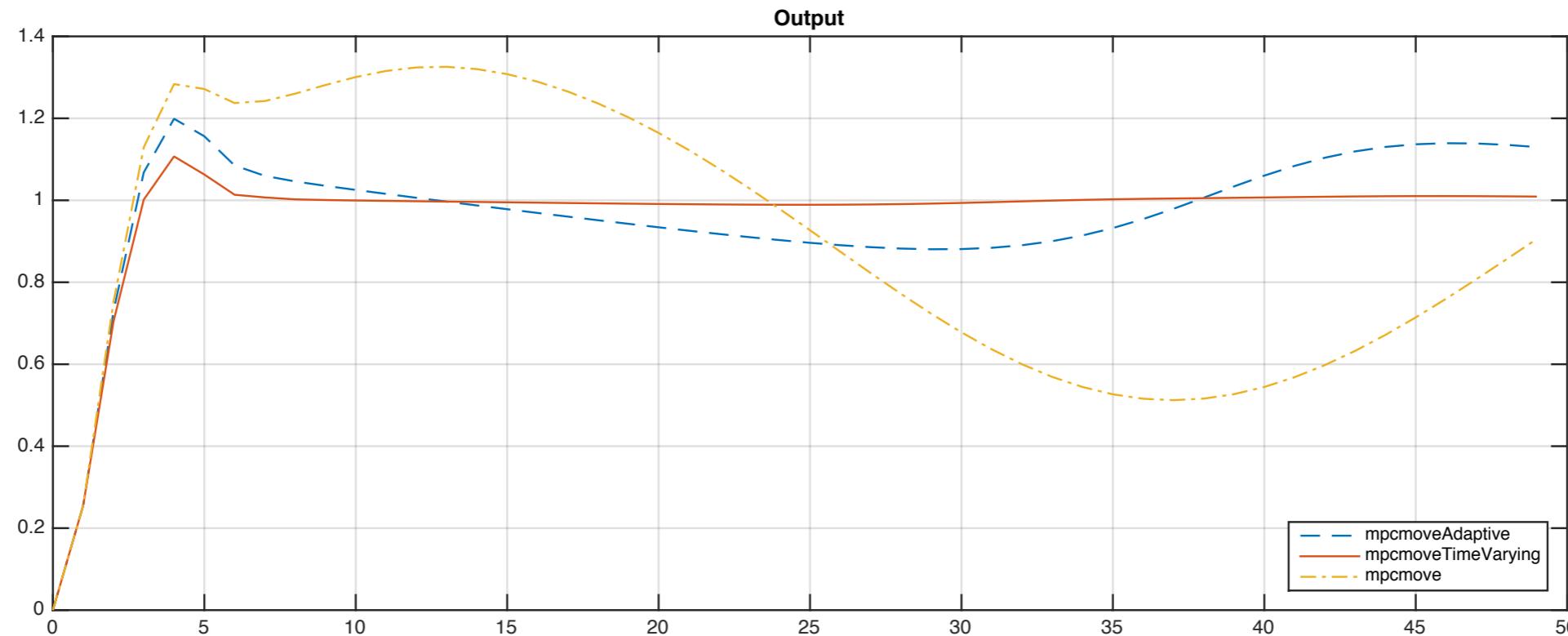
constraints

$$\begin{cases} u_{\min} \leq u_k \leq u_{\max} \\ y_{\min} \leq y_k \leq y_{\max} \end{cases}$$

LTV-MPC still leads to a convex QP

- LTV models can be obtained from linearizing NL models around time-varying reference trajectories (e.g.: previous optimal trajectory)

LTV-MPC EXAMPLE



- Process model is LTV:
- LTV-MPC is quite good
- LPV-MPC tries to catch-up with time-varying model
- LTI-MPC is not good

$$\begin{aligned} \frac{d^3y}{dt^3} + 3\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + (6 + \sin(5t))y \\ = 5\frac{du}{dt} + (5 + 2\cos(\frac{5}{2}t))u \end{aligned}$$

(See demo `TimeVaryingMPCCControlOfATimeVaryingLinearSystemExample`, MPC Toolbox)

LTV-MPC EXAMPLE

- Define LTV model

```
Models = tf; ct = 1;
for t = 0:0.1:10
    Models(:,:,ct) = tf([5 5+2*cos(2.5*t)],[1 3 2 6+sin(5*t)]);
    ct = ct + 1;
end

Ts = 0.1; % sampling time
Models = ss(c2d(Models,Ts));
```

- Design MPC controller

```
sys = ss(c2d(tf([5 5],[1 3 2 6]),Ts)); % average model time
p = 3;                                     % prediction horizon
m = 3;                                     % control horizon
mpcobj = mpc(sys,Ts,p,m);

mpcobj.MV = struct('Min',-2,'Max',2);      % input constraints
mpcobj.Weights = struct('MV',0,'MVRate',0.01,'Output',1);
```

LTV-MPC EXAMPLE

- Simulate LTV system with **LTI** MPC controller

```
for ct = 1:(Tstop/Ts+1)
    real_plant = Models(:,:,ct);      % Get the current plant
    y = real_plant.C*x;
    u = mpcmove(mpcobj,xmpc,y,1);    % Apply LTI MPC
    x = real_plant.A*x + real_plant.B*u;
end
```

- Simulate LTV system with **LPV** MPC controller

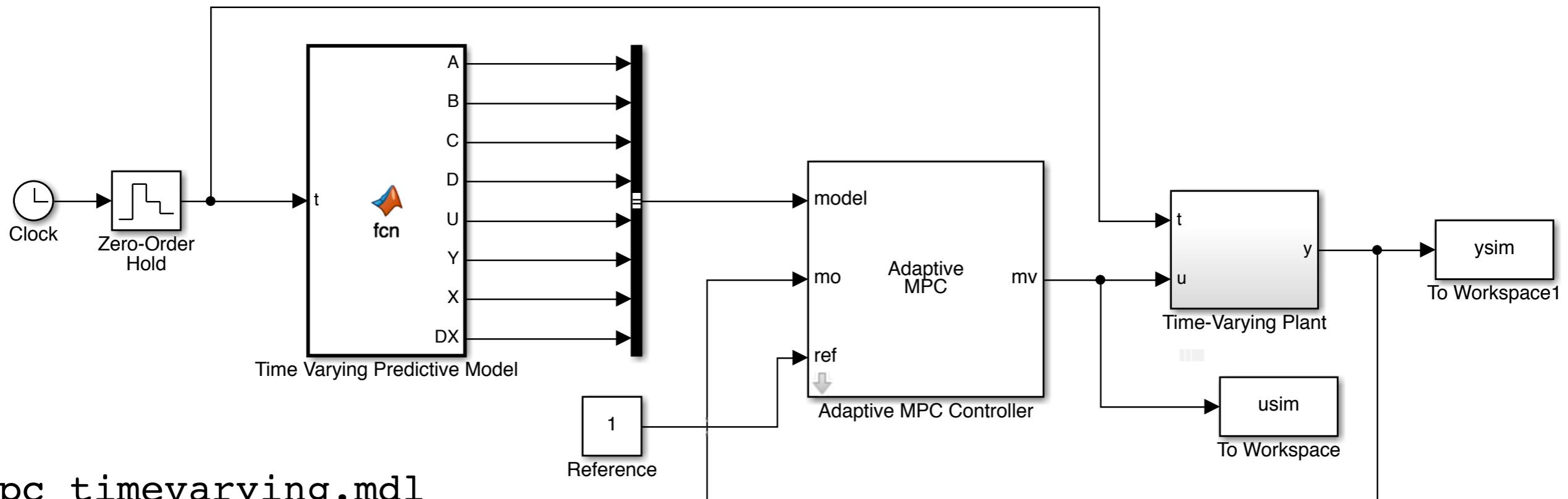
```
for ct = 1:(Tstop/Ts+1)
    real_plant = Models(:,:,ct);      % Get the current plant
    y = real_plant.C*x;
    u = mpcmoveAdaptive(mpcobj,xmpc,real_plant,nominal,y,1);
    x = real_plant.A*x + real_plant.B*u;
end
```

LTV-MPC EXAMPLE

- Simulate LTV system with **LTV** MPC controller

```
for ct = 1:(Tstop/Ts+1)
    real_plant = Models(:,:,ct);           % Get the current plant
    y = real_plant.C*x;
    u = mpcmoveAdaptive(mpcobj,xmpc,Models(:,:,ct:ct+p),...
        Nominals,y,1);
    x = real_plant.A*x + real_plant.B*u;
end
```

- Simulate in Simulink

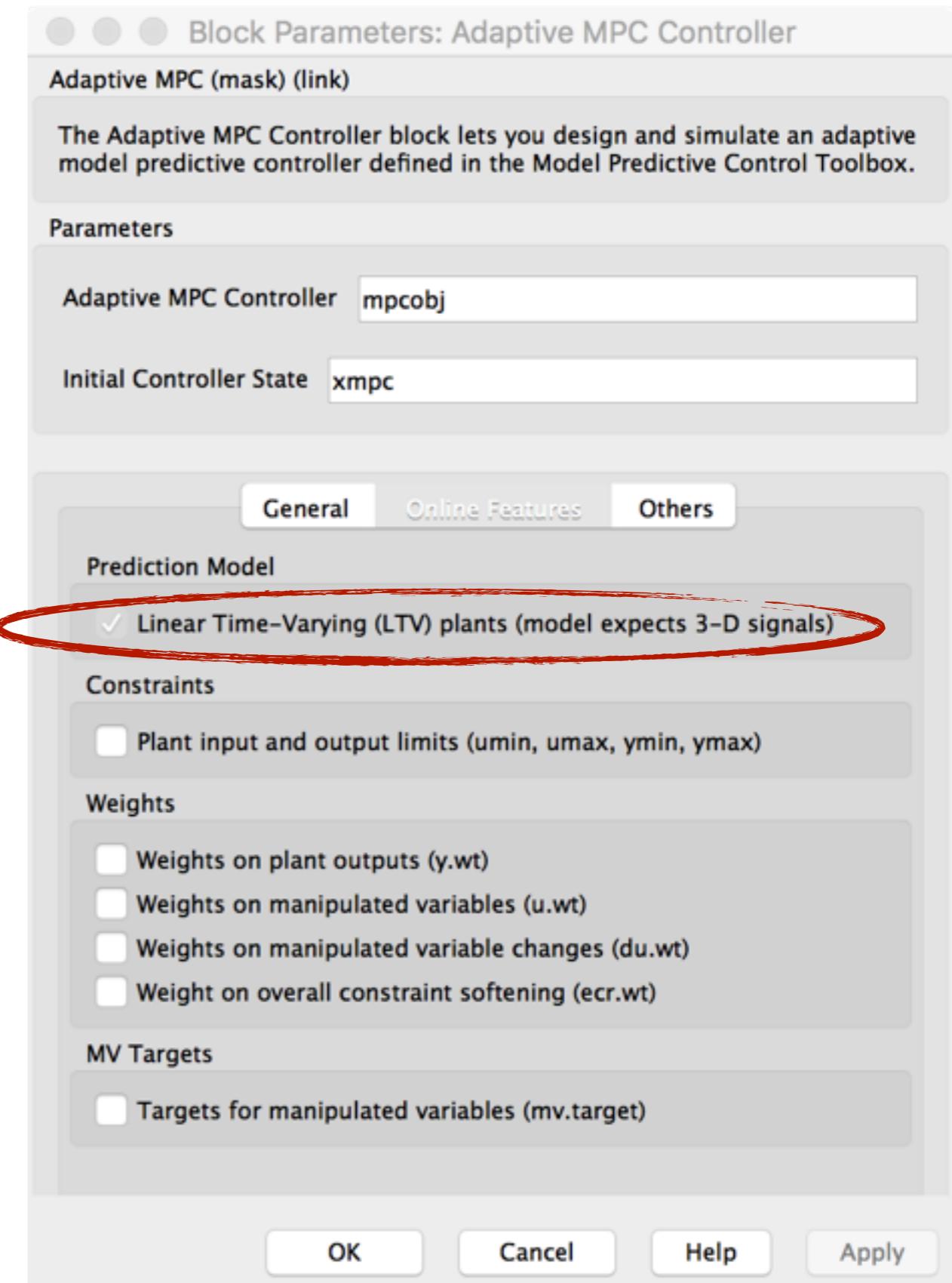


mpc_timevarying.mdl

LTV-MPC EXAMPLE

- Simulink block

need to provide 3D
array of future models



`mpc_timevarying.mdl`

LINEARIZATION AND TIME-DISCRETIZATION

- Assume model is nonlinear and continuous-time

$$\frac{dx}{dt} = f(x(t), u(t))$$

- Linearize around a nominal state $\bar{x}(t)$ and input $\bar{u}(t)$, such as:

- an **equilibrium**
 - a reference **trajectory**
 - the **most updated** value

$$\begin{aligned} \frac{dx}{dt}(t + \tau) &\simeq \left. \frac{\partial f}{\partial x} \right|_{\bar{x}(t), \bar{u}(t)} (x(t + \tau) - \bar{x}(t)) \\ &\quad + \left. \frac{\partial f}{\partial u} \right|_{\bar{x}(t), \bar{u}(t)} (u(t + \tau) - \bar{u}(t)) + f(x(t), u(t)) \end{aligned}$$

- Conversion to discrete-time linear prediction model

discrete-time IPV model

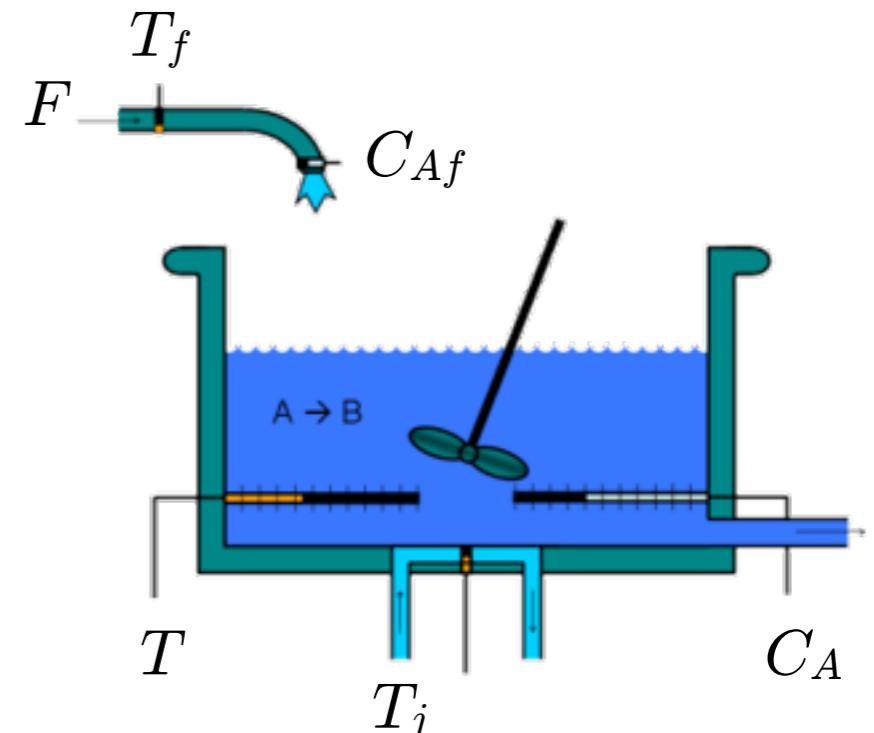
$$x_{k+1} = \left(I + T_s \frac{\partial f}{\partial x} \Big|_{\bar{x}(t), \bar{u}(t)} \right) x_k + \left(T_s \frac{\partial f}{\partial u} \Big|_{\bar{x}(t), \bar{u}(t)} \right) u_k + f_k$$

model matrices
depend on
current time t

EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- MPC control of a diabatic continuous stirred tank reactor (CSTR)
- Process model is nonlinear:

$$\begin{aligned}\frac{dC_A}{dt} &= \frac{F}{V}(C_{Af} - C_A) - C_A k_0 e^{-\frac{\Delta E}{RT}} \\ \frac{dT}{dt} &= \frac{F}{V}(T_f - T) + \frac{UA}{\rho C_p V}(T_j - T) - \frac{\Delta H}{\rho C_p} C_A k_0 e^{-\frac{\Delta E}{RT}}\end{aligned}$$



- T : temperature inside the reactor [K] (state)
- C_A : concentration of the reagent in the reactor [$kgmol/m^3$] (state)
- T_j : jacket temperature [K] (input)
- T_f : feedstream temperature [K] (measured disturbance)
- C_{Af} : feedstream concentration [$kgmol/m^3$] (measured disturbance)

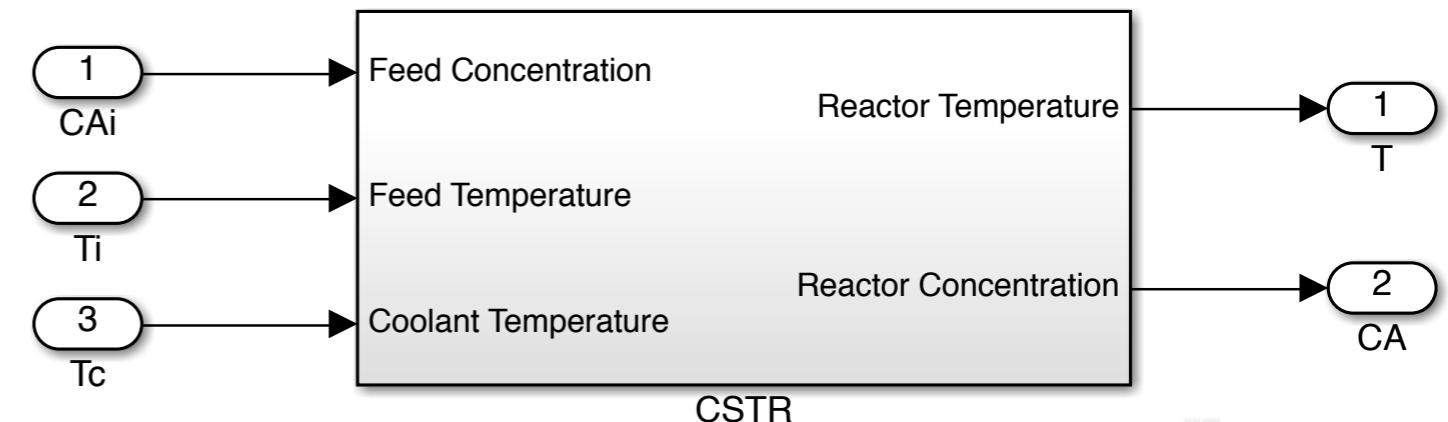
- **Objective:** manipulate T_j to regulate C_A on desired set-point

```
>> edit ampccstr_linearization (MPC Toolbox)
```

EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- Process model

```
>> mpc_cstr_plant
```



```
% Create operating point specification.
plant_mdl = 'mpc_cstr_plant';
op = operspec(plant_mdl);

op.Inputs(1).u = 10; % Feed concentration known @initial condition
op.Inputs(1).Known = true;
op.Inputs(2).u = 298.15; % Feed temperature known @initial condition
op.Inputs(2).Known = true;
op.Inputs(3).u = 298.15; % Coolant temperature known @initial condition
op.Inputs(3).Known = true;

[op_point, op_report] = findop(plant_mdl,op); % Compute initial condition

% Obtain nominal values of x, y and u.
x0 = [op_report.States(1).x;op_report.States(2).x];
y0 = [op_report.Outputs(1).y;op_report.Outputs(2).y];
u0 = [op_report.Inputs(1).u;op_report.Inputs(2).u;op_report.Inputs(3).u];

% Obtain linear plant model at the initial condition.
sys = linearize(plant_mdl, op_point);
sys = sys(:,2:3); % First plant input CAi dropped because not used by MPC
```

EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- MPC design (1/2)

```
% Discretize the plant model
Ts = 0.5; % hours
plant = c2d(sys,Ts);

% Design MPC Controller

% Specify signal types used in MPC
plant.InputGroup.MeasuredDisturbances = 1;
plant.InputGroup.ManipulatedVariables = 2;
plant.OutputGroup.Measured = 1;
plant.OutputGroup.Unmeasured = 2;
plant.InputName = {'Ti','Tc'};
plant.OutputName = {'T','CA'};

% Create MPC controller with default prediction and control horizons
mpcobj = mpc(plant);

% Set nominal values in the controller
mpcobj.Model.Nominal = struct('x', x0, 'u', u0(2:3), 'y', y0, 'dx', [0 0]);
```

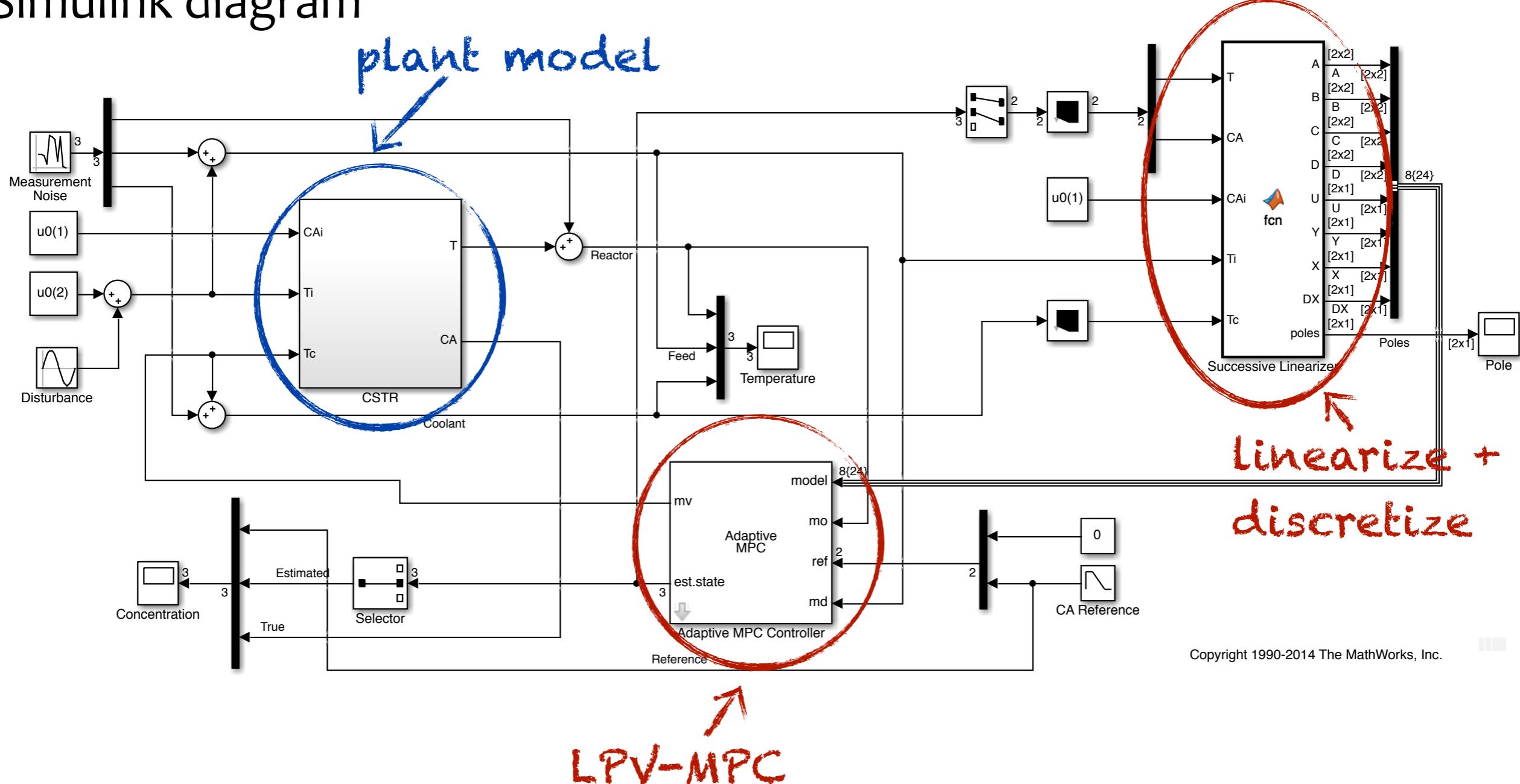
EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- MPC design (2/2)

```
% Set scale factors because plant input and output signals have  
different orders of magnitude  
Uscale = [30 50];  
Yscale = [50 10];  
mpcobj.DV(1).ScaleFactor = Uscale(1);  
mpcobj.MV(1).ScaleFactor = Uscale(2);  
mpcobj.OV(1).ScaleFactor = Yscale(1);  
mpcobj.OV(2).ScaleFactor = Yscale(2);  
  
% Let reactor temperature T float (i.e. with no setpoint tracking  
error penalty), because the objective is to control reactor  
concentration CA and only one manipulated variable (coolant  
temperature Tc) is available.  
mpcobj.Weights.OV = [0 1];  
  
% Due to the physical constraint of coolant jacket, Tc rate of  
change is bounded by degrees per minute.  
mpcobj.MV.RateMin = -2;  
mpcobj.MV.RateMax = 2;
```

EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- Simulink diagram

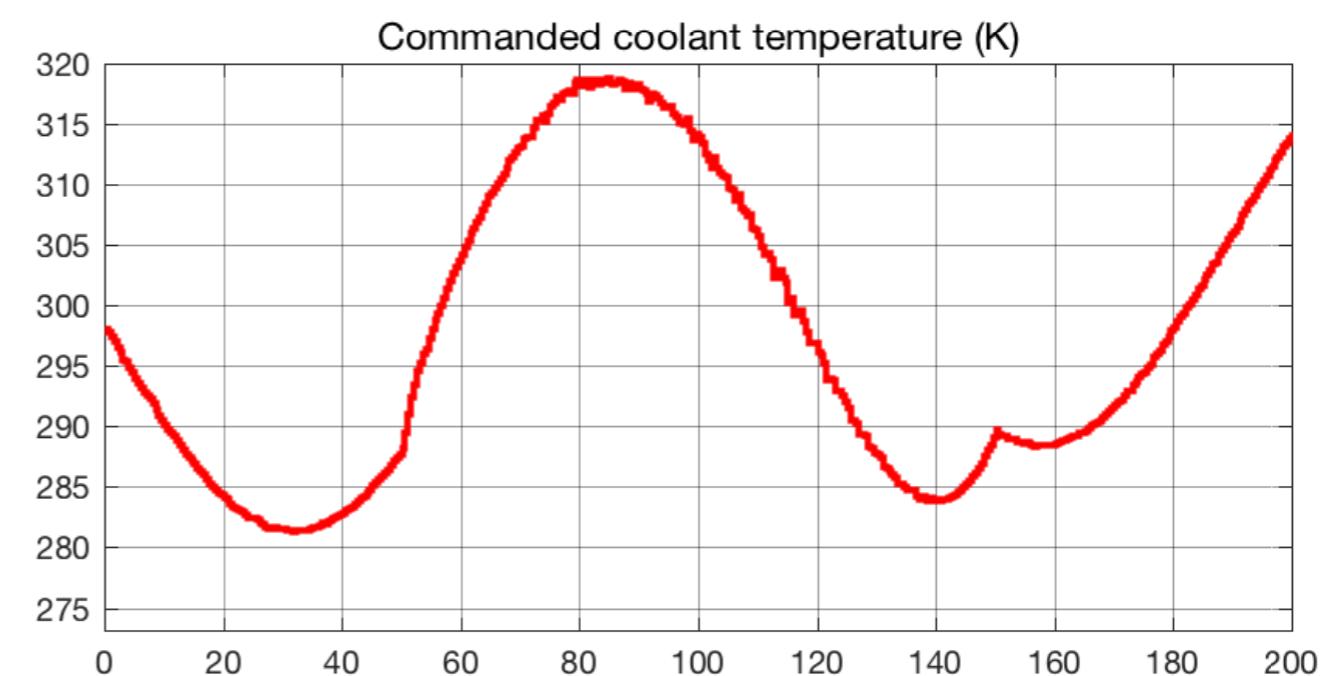
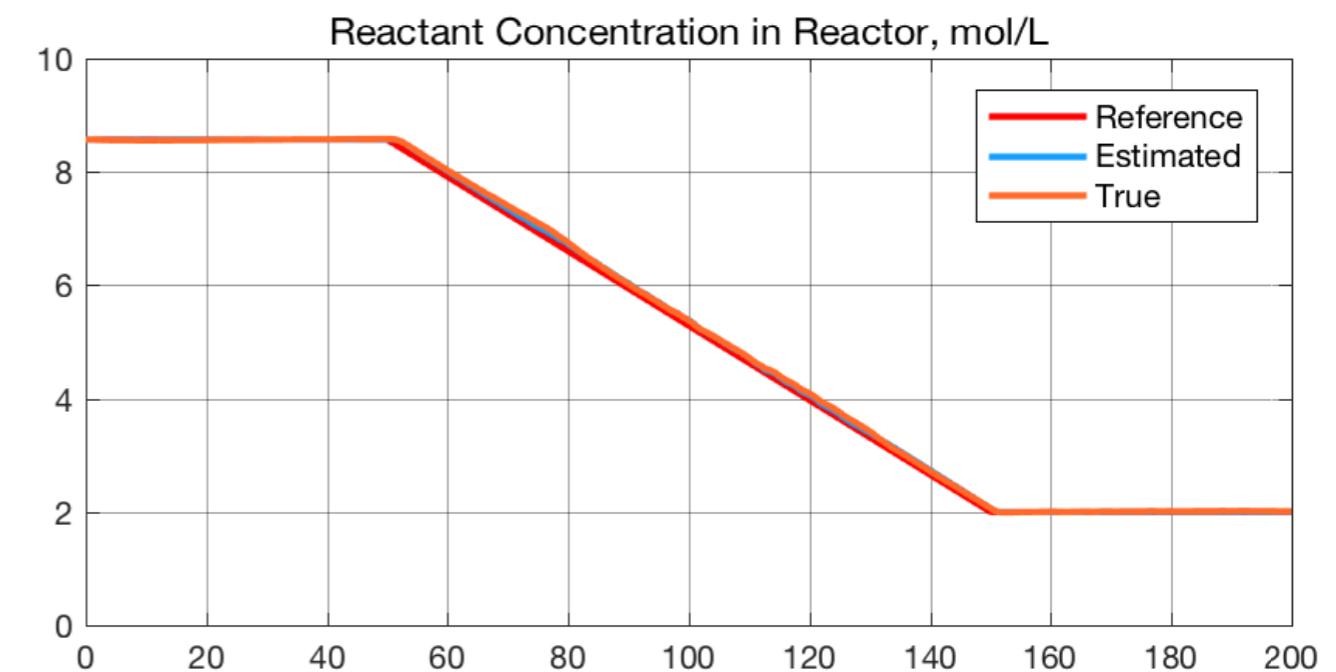
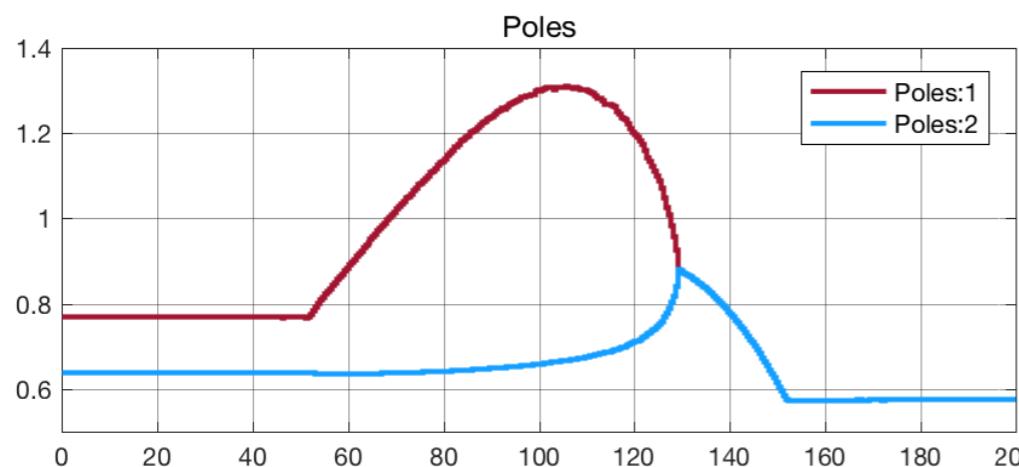
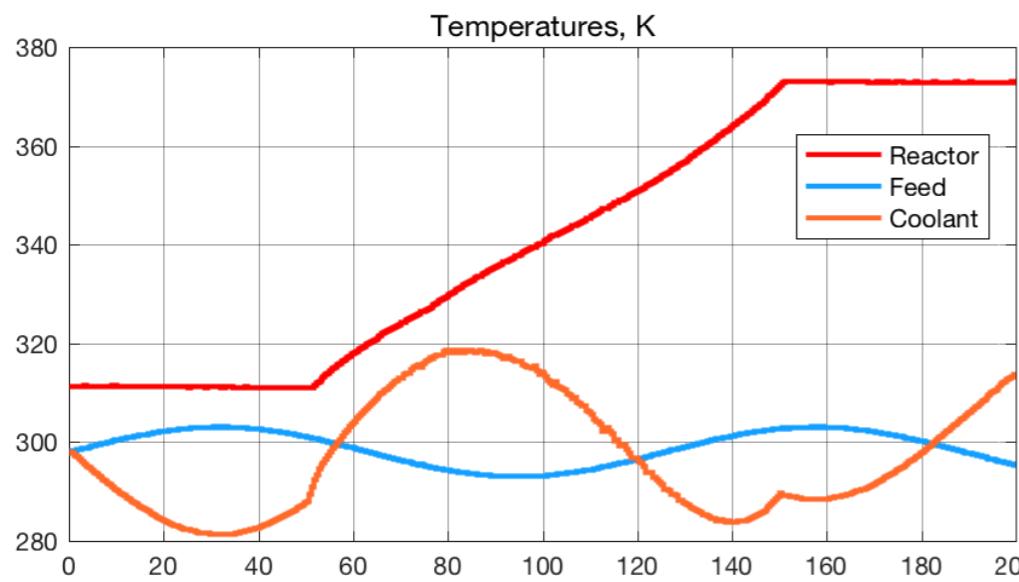


Copyright 1990-2014 The MathWorks, Inc.

>> ampc_cstr_linearization

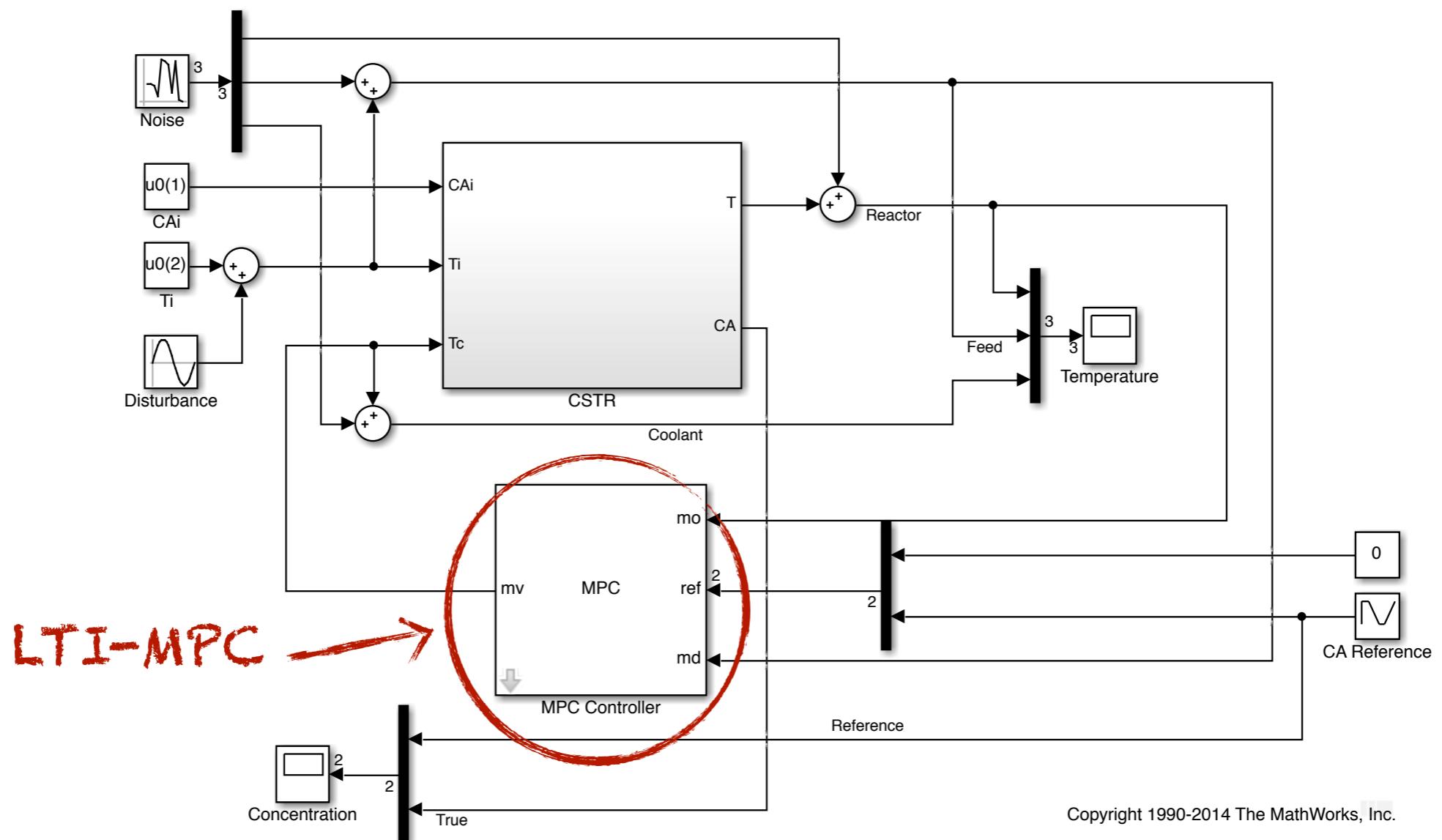
EXAMPLE: LPV-MPC OF A NONLINEAR CSTR SYSTEM

- Closed-loop results



EXAMPLE: LTI-MPC OF A NONLINEAR CSTR SYSTEM

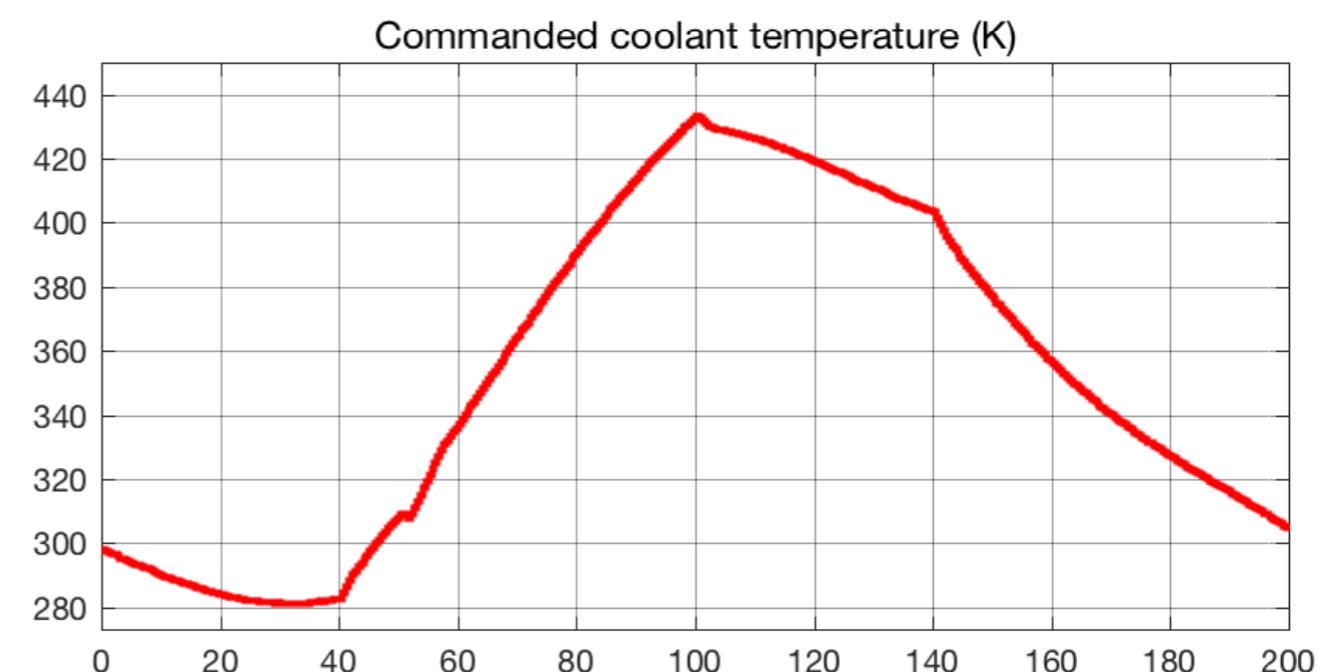
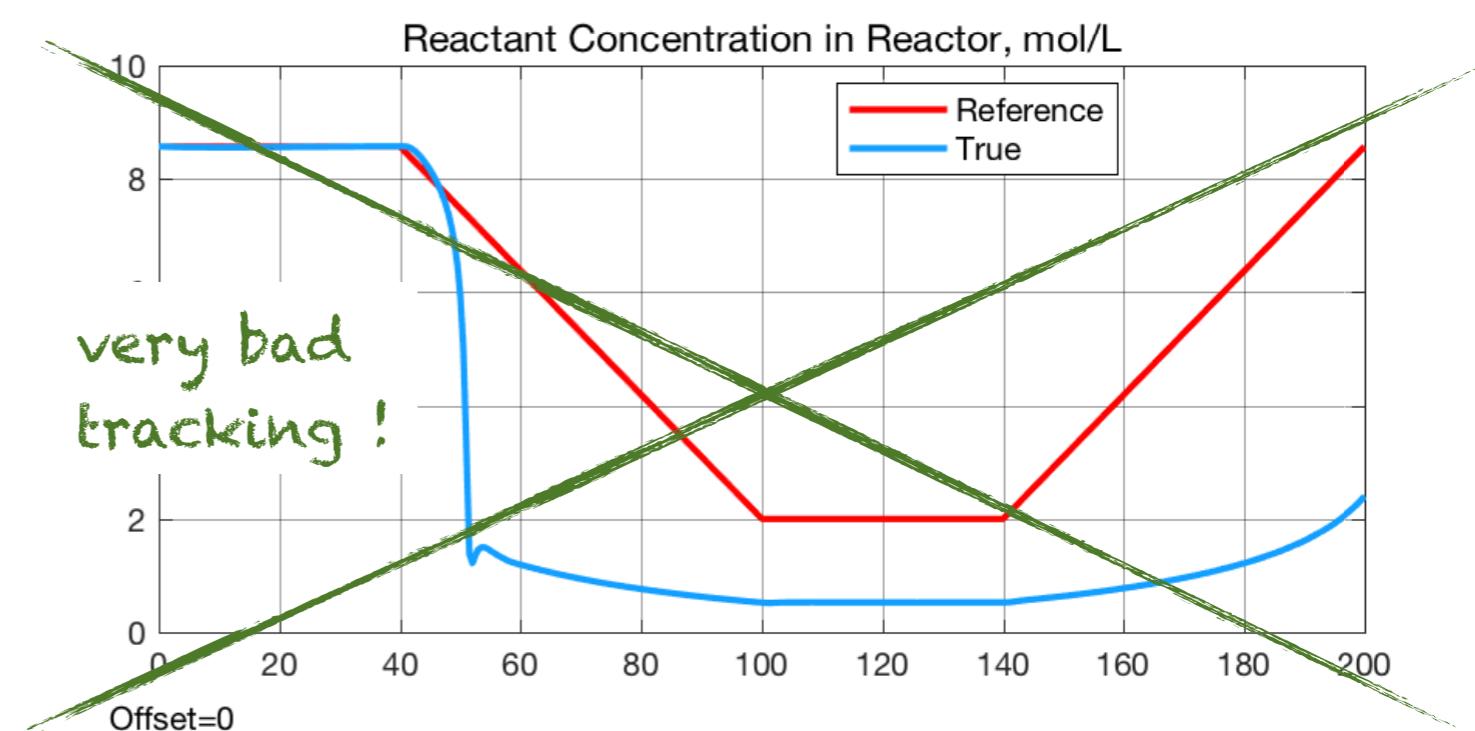
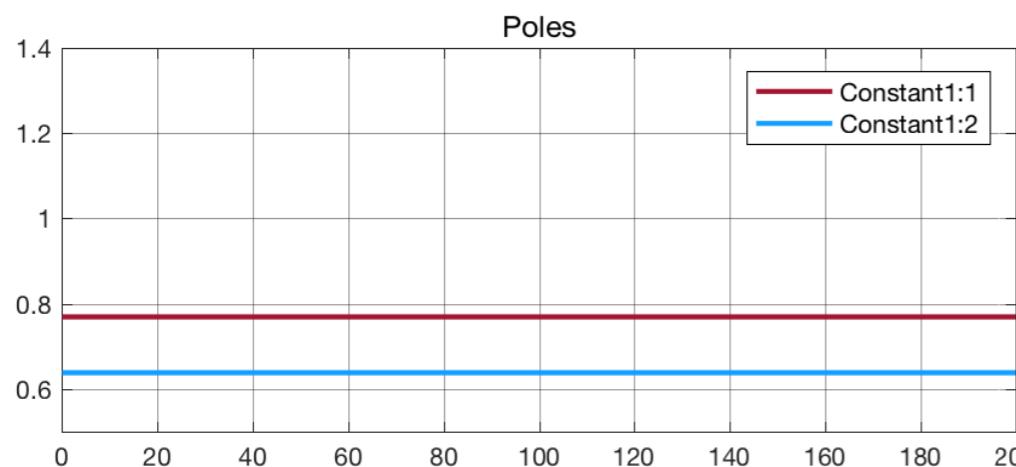
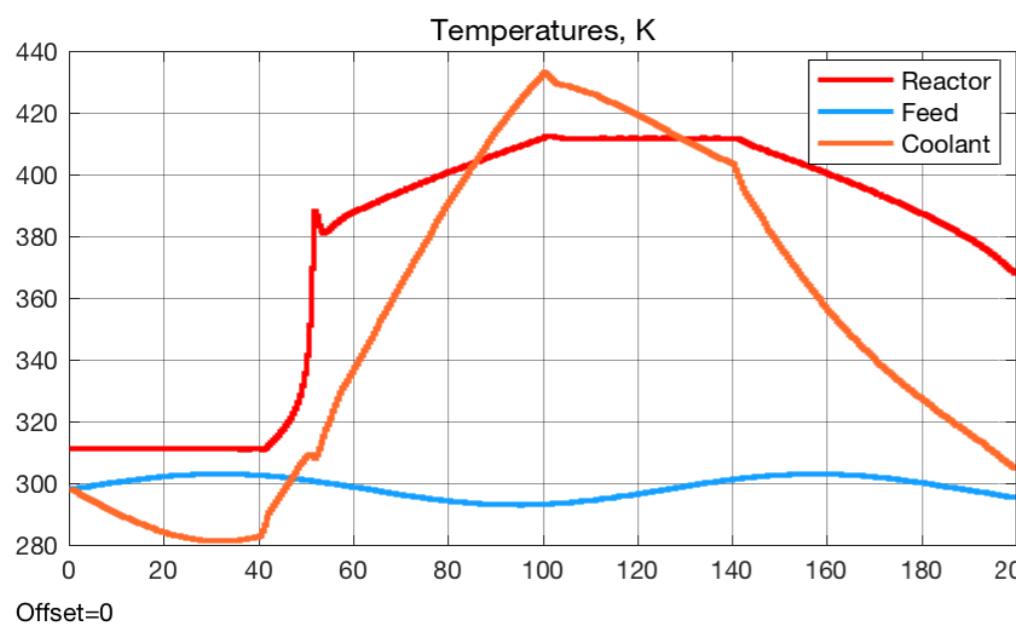
- Let us try with a fixed (LTI) MPC controller



- Same tuning of MPC parameters

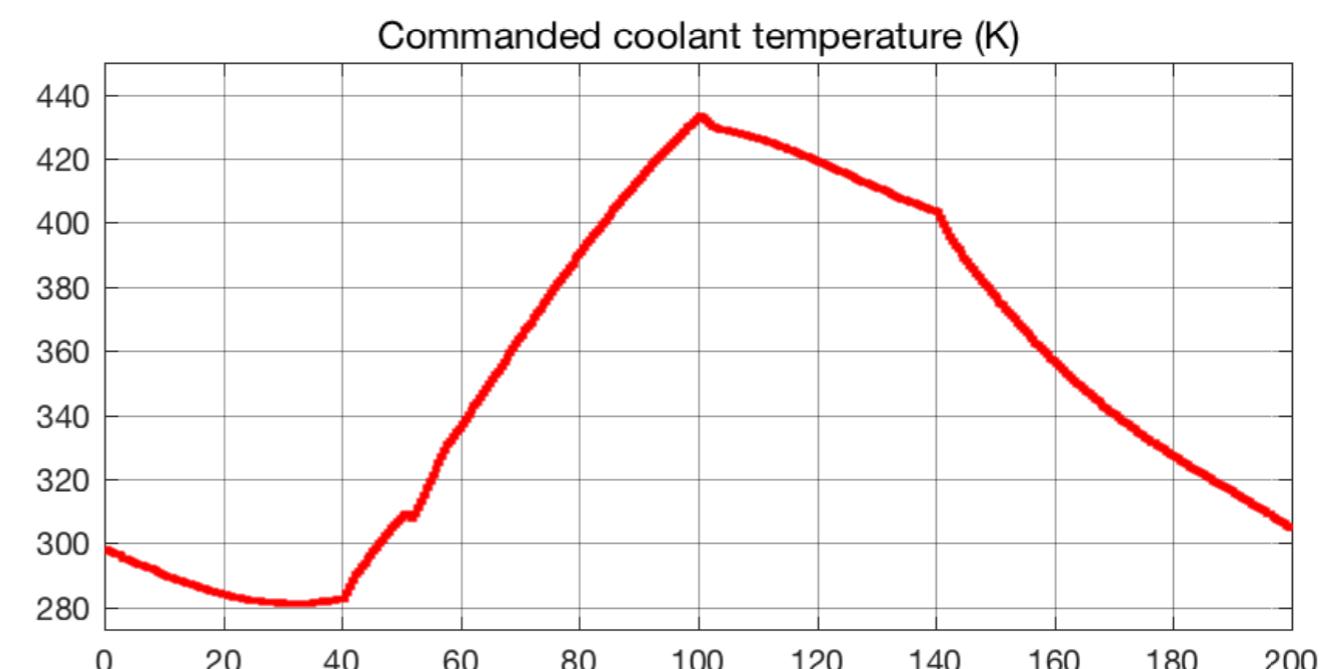
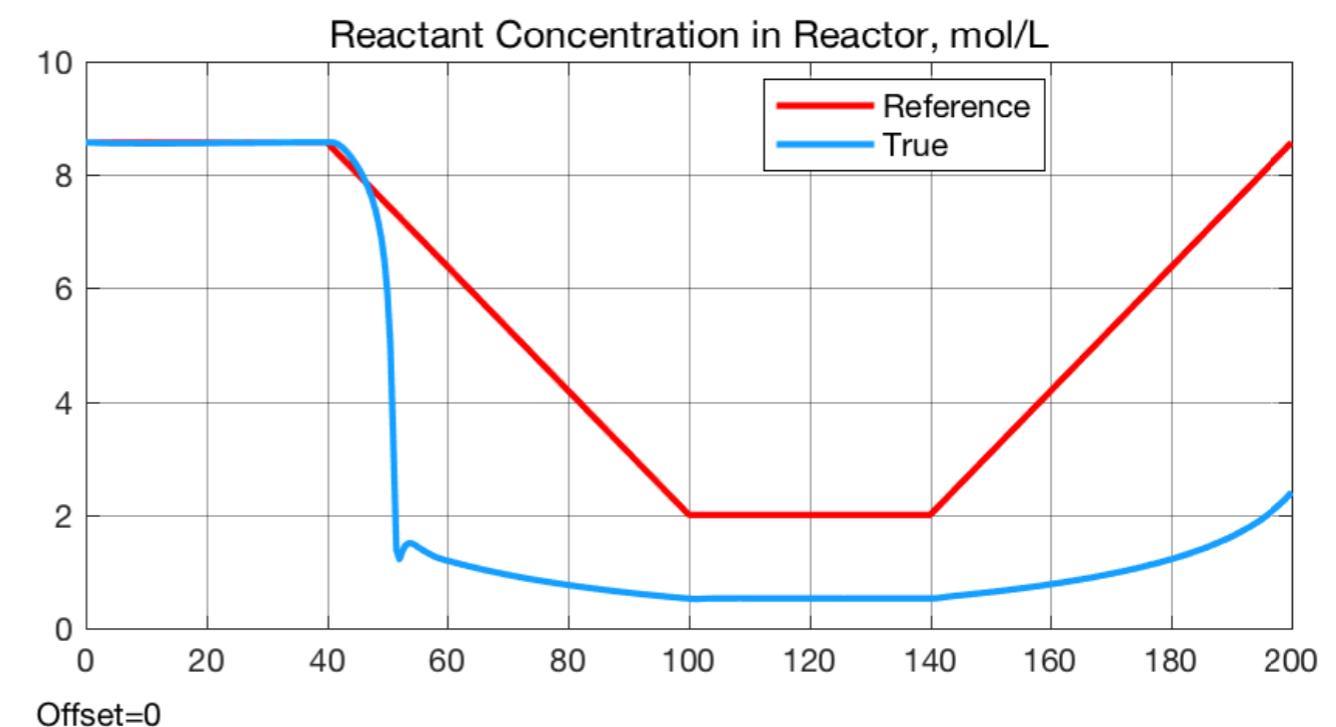
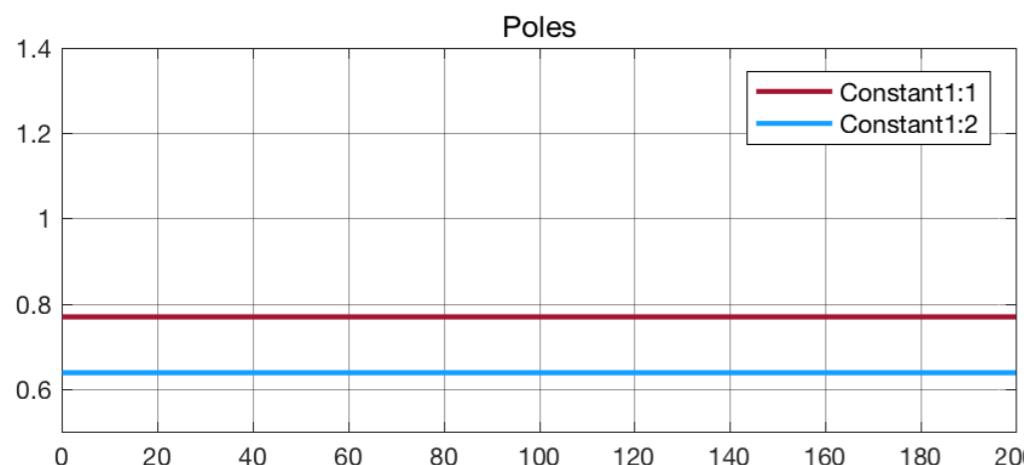
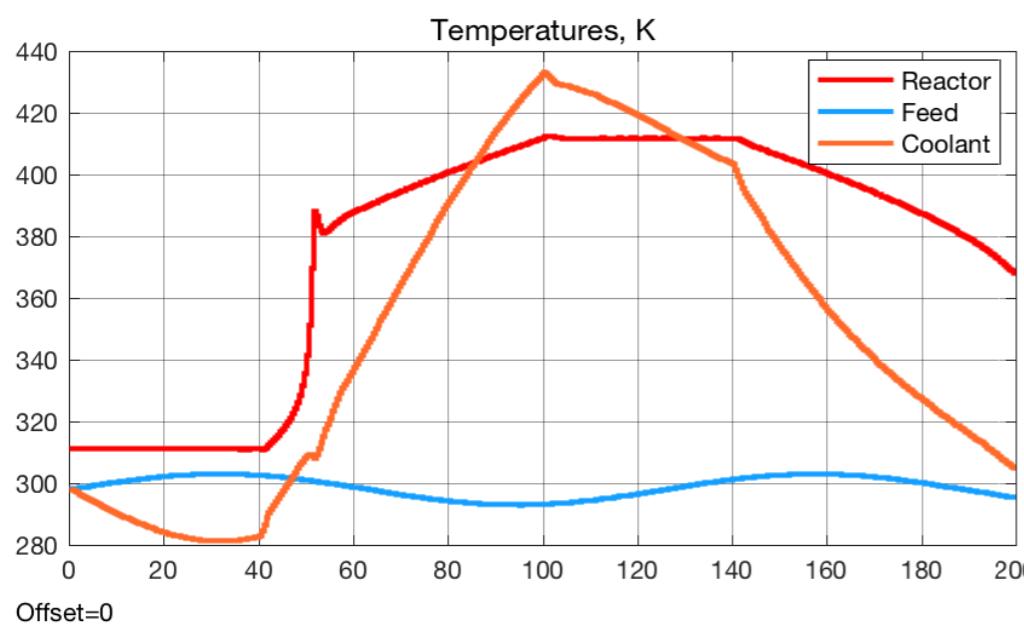
EXAMPLE: LTI-MPC OF A NONLINEAR CSTR SYSTEM

- Closed-loop results



EXAMPLE: LTI-MPC OF A NONLINEAR CSTR SYSTEM

- Closed-loop results



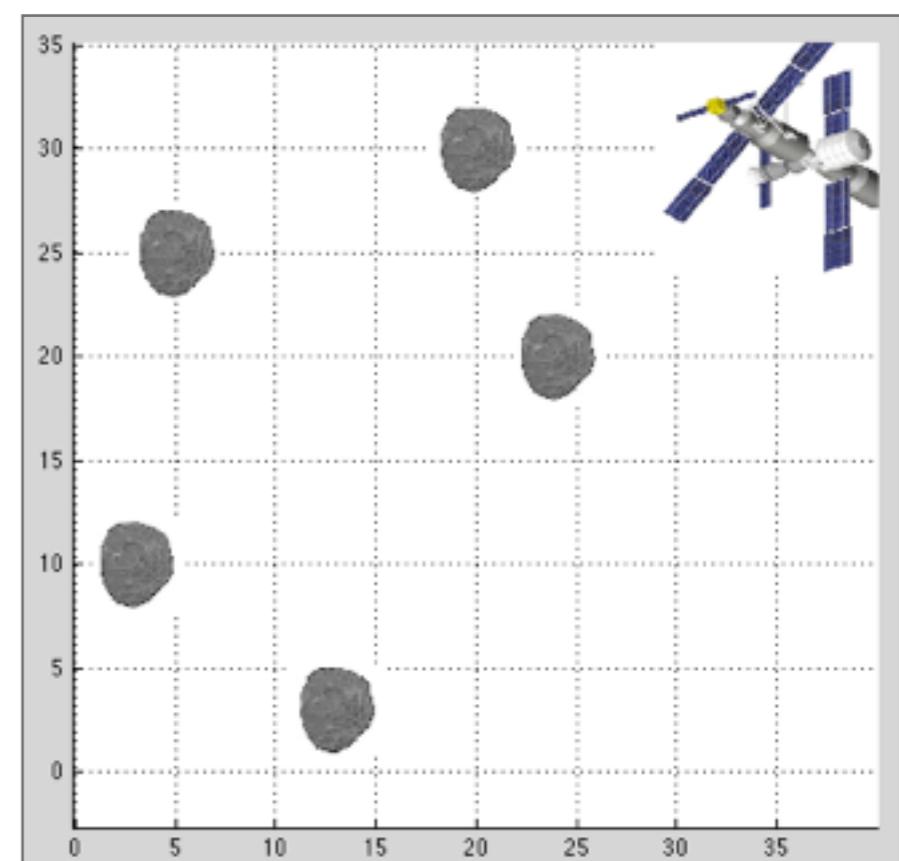
EXAMPLE: LTV-MPC FOR UAV NAVIGATION

- Goal: navigate two planar vehicles among obstacles to a target position
- The dynamical model of each vehicle is described by

$$p(t) = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} p_c(t)$$

$p = [x, y]$ vehicle position
 p_c = commanded position

- Vehicle dynamics converted to discrete-time by exact sampling
- Five square obstacles placed in workspace



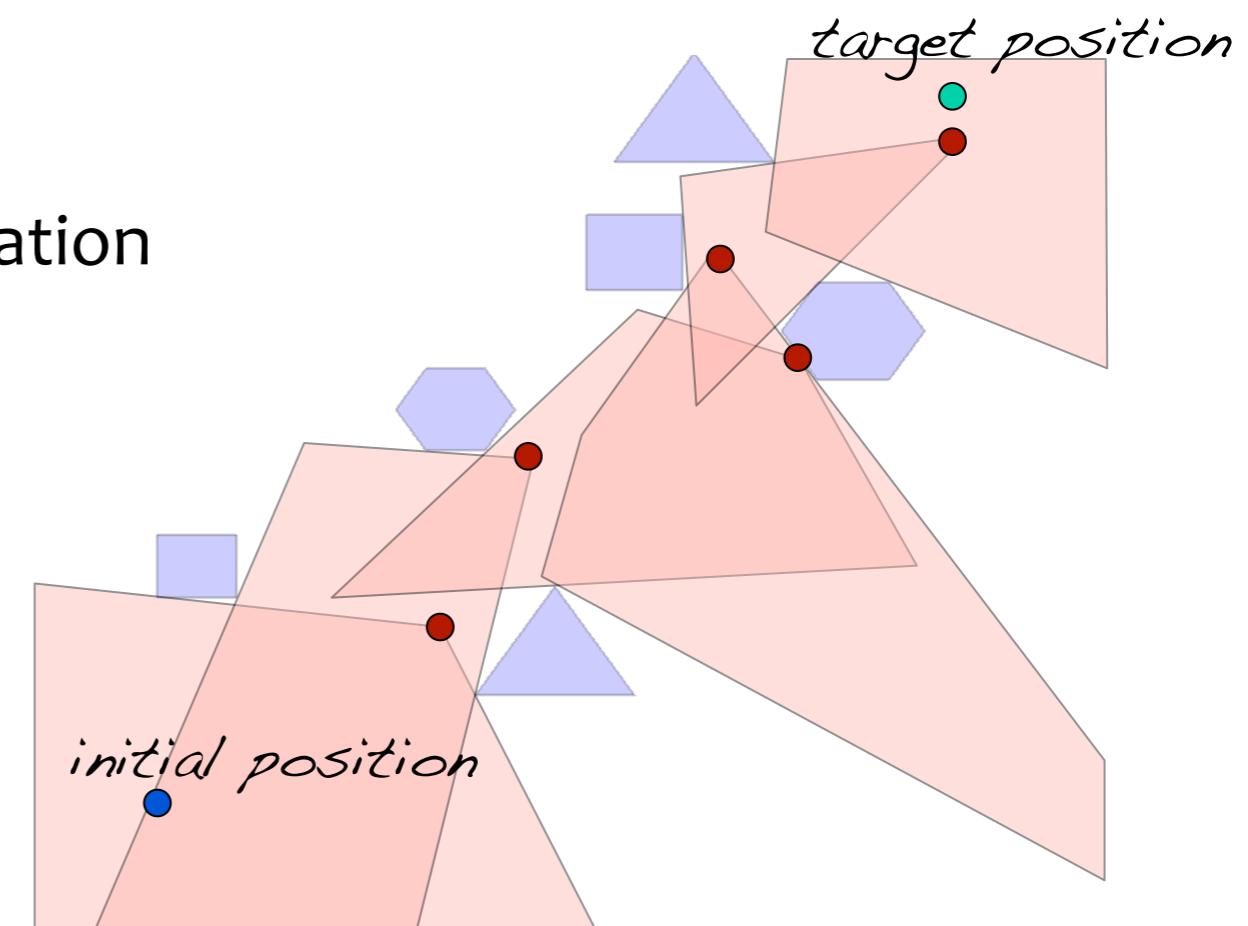
LTV-MPC FOR OBSTACLE AVOIDANCE

- Goal: Use **linear time-varying (LTV) MPC** to generate desired positions to stabilizing controller in **real-time** to avoid **obstacles** and other UAVs

- Problem: feasible space is non-convex !

- Main idea: get a convex inner approximation of the feasible space that is

- **Polyhedral** = linear constraints on predicted states in LTV-MPC problem
- **Very simple** to compute on-line
- Can handle polytopic obstacles of **arbitrary shape** and **dimension**



- Alternative: non-convex feasible space modeled by mixed-integer constraints, guidance problem solved by (time-invariant) **hybrid MPC**

(Bemporad, Rocchi, IFAC 2011)

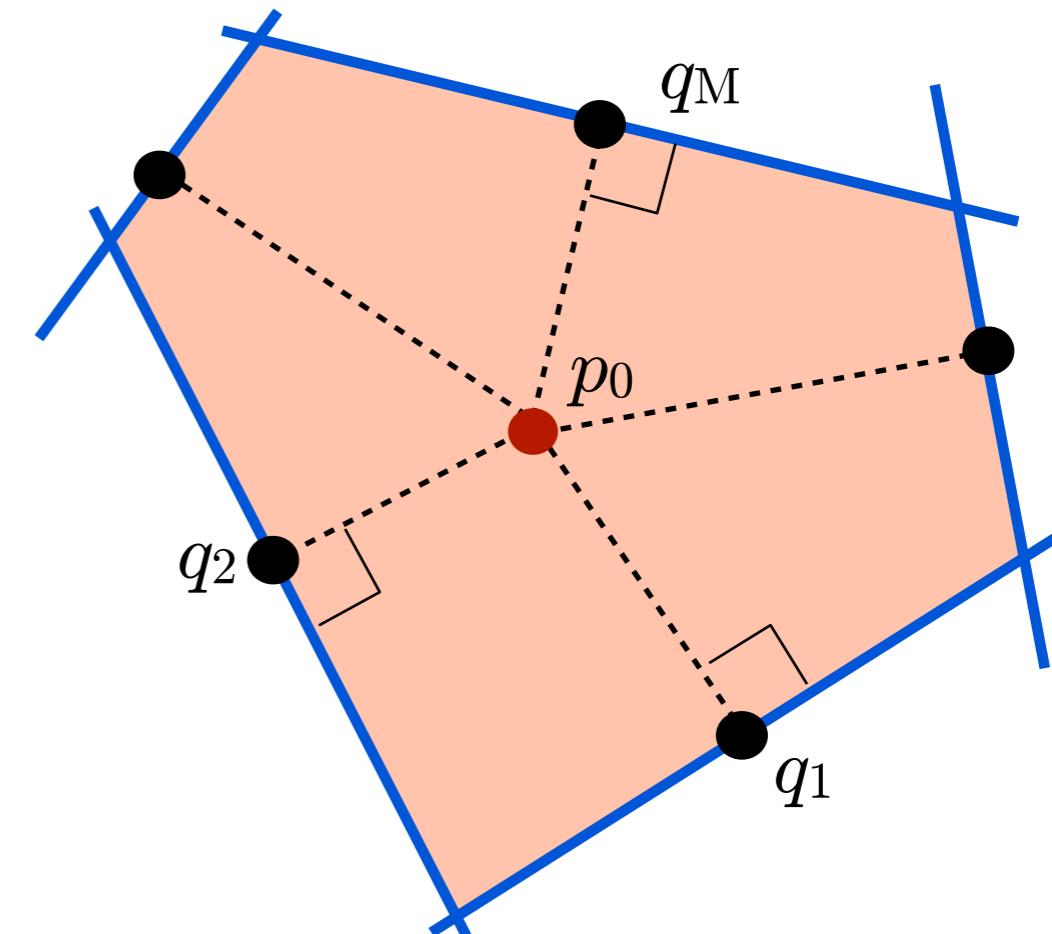
FAST GREEDY APPROXIMATION ALGORITHM (1/3)

(Bemporad, Rocchi, CDC 2011)

- Assume (for the moment) that vehicle and obstacles are points in space

$p_0 \in \mathbb{R}^d$ = current vehicle position

$q_1, q_2, \dots, q_M \in \mathbb{R}^d$ = obstacle positions



- Polyhedral approximation:

$$P = \left\{ p \in \mathbb{R}^d : \begin{bmatrix} (q_1 - p_0)' \\ \vdots \\ (q_M - p_0)' \end{bmatrix} p \leq \begin{bmatrix} (q_1 - p_0)' q_1 \\ \vdots \\ (q_M - p_0)' q_M \end{bmatrix} \right\} \quad p_0 \neq q_i, \forall i = 1, \dots, M$$

- P contains p_0 and does not contain any of the obstacles q_1, q_2, \dots, q_M in its interior

FAST GREEDY APPROXIMATION ALGORITHM (2/3)

- Assume now **polyhedral obstacles** with shapes W_1, W_2, \dots, W_M
- For each obstacle j define the scalar g^j

$$g^j = \min_{\substack{w \in \mathbb{R}^d \\ \text{s.t.}}} (q_j - p_0)' w$$

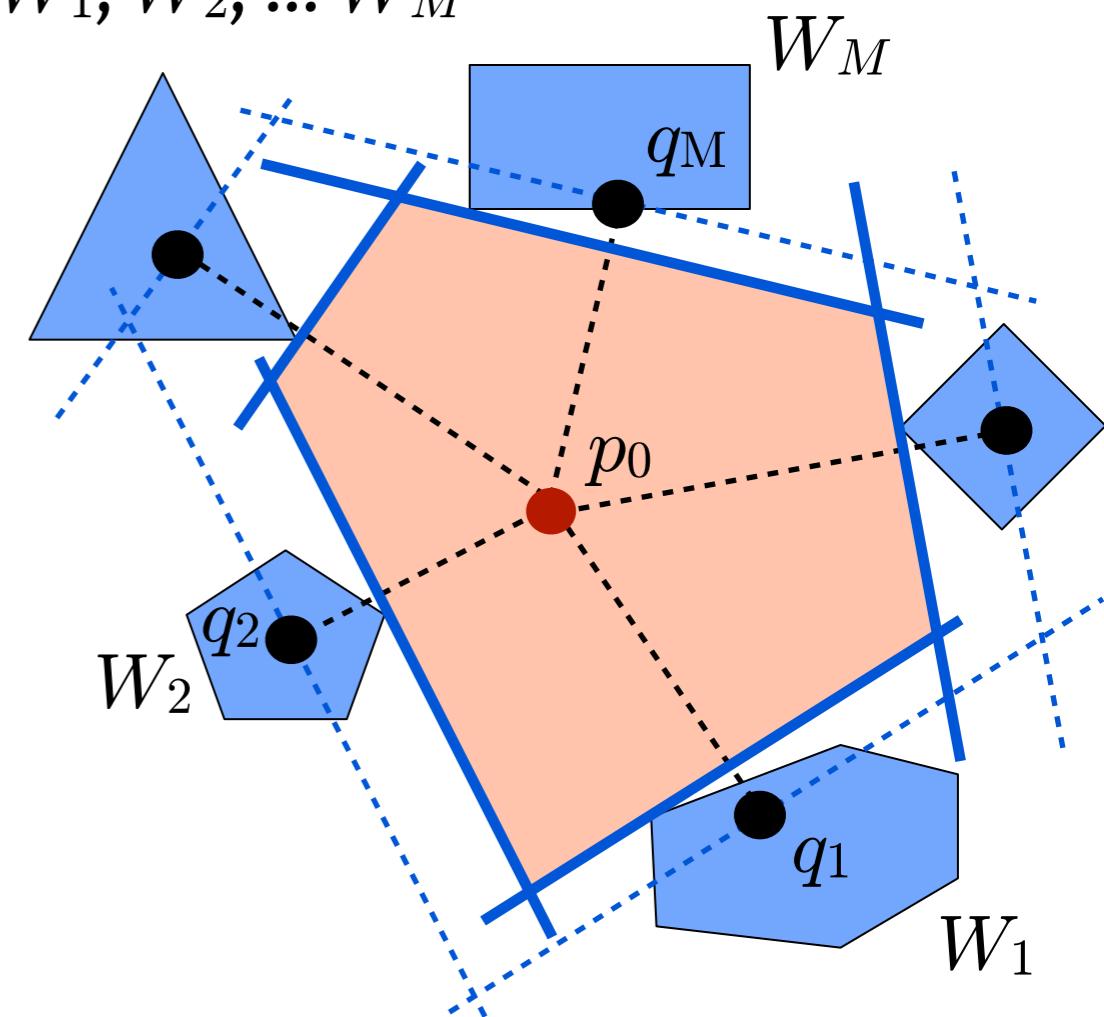
$$w \in W_j$$

- If the vertices of W_M are available, simply

$$g^j = \min_{h=1, \dots, s_j} A_c^j w_{jh} \quad (w_{jh} = \text{vertex of } W_j)$$

- **Polyhedral approximation:**

$$P = \left\{ p \in \mathbb{R}^d : \begin{bmatrix} (q_1 - p_0)' \\ \vdots \\ (q_M - p_0)' \end{bmatrix} p \leq \begin{bmatrix} (q_1 - p_0)' q_1 + g^1 \\ \vdots \\ (q_M - p_0)' q_M + g^M \end{bmatrix} \right\}$$

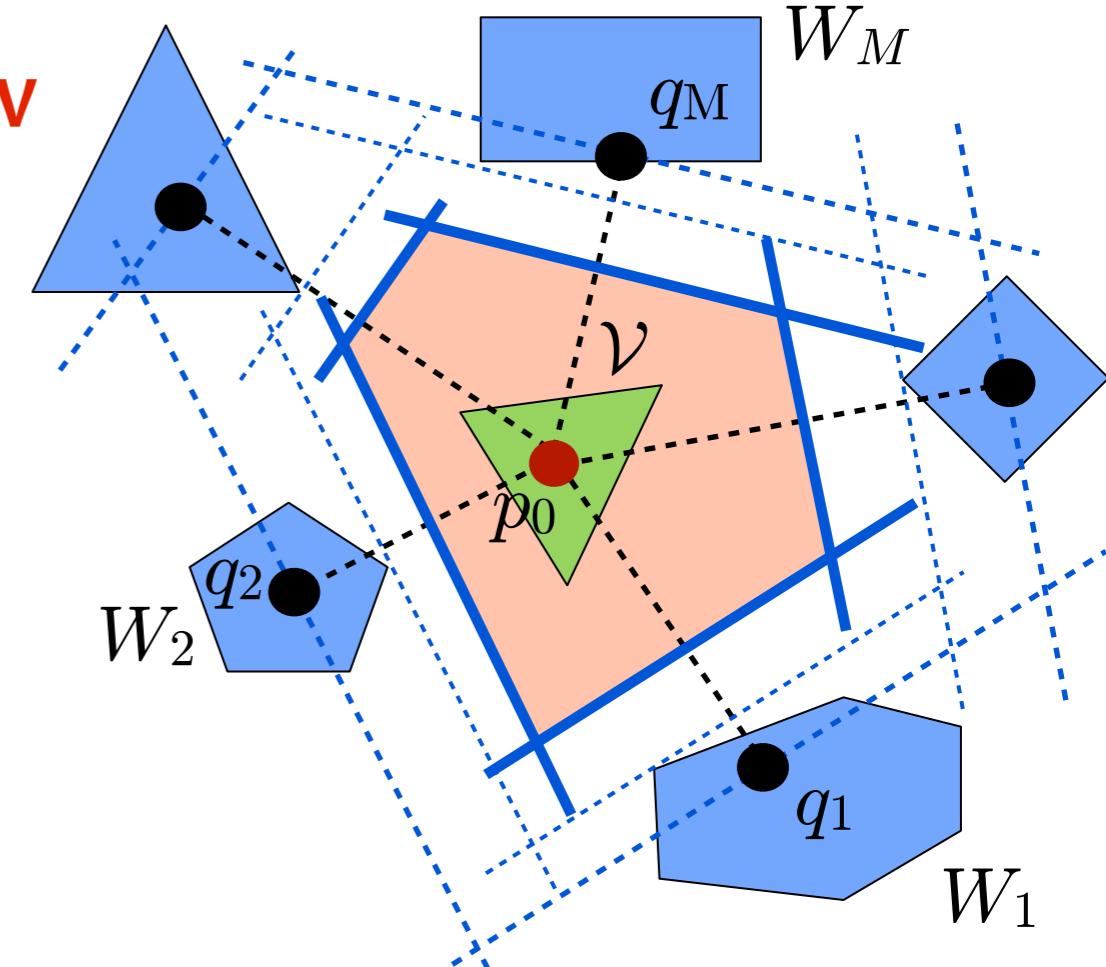


If P is nonempty, then P contains p_0 and does not contain any of the obstacles $B_j = \{q_j\} \oplus W_j$ in its interior

FAST GREEDY APPROXIMATION ALGORITHM (3/3)

- Assume polyhedral obstacles and **polytopic UAV**

$$\mathcal{V} \triangleq \text{conv}(p_0 + d_1, \dots, p_0 + d_r)$$



- Polyhedral approximation:**

$$P = \left\{ p \in \mathbb{R}^d : \begin{bmatrix} (q_1 - p_0)' \\ \vdots \\ (q_M - p_0)' \end{bmatrix} p \leq \begin{bmatrix} (q_1 - p_0)'(q_1 - \mathbf{d}_i) + g^1 \\ \vdots \\ (q_M - p_0)'(q_M - \mathbf{d}_i) + g^M \end{bmatrix}, i = 1, \dots, r \right\}$$

If P is nonempty, then P contains \mathcal{V} and does not contain any of the obstacles
 $B_j = \{q_j\} \oplus W_j$ in its interior

LTV-MPC GUIDANCE ALGORITHM: PREDICTION MODEL

- Linear approximation of stabilized dynamics of UAV # i

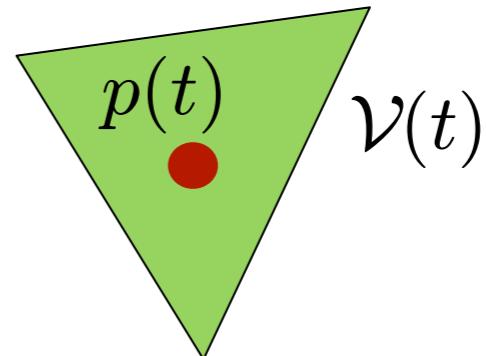
$$p_i(t+1) = A_i p_i(t) + B_i p_{ci}(t)$$

$$p_i(t) = \begin{bmatrix} x_i(t) \\ y_i(t) \\ z_i(t) \end{bmatrix} \quad \begin{array}{l} \text{position} \\ \text{of vehicle } \#i \\ \text{at time } t \end{array}$$

$$p_{ci}(t) = \begin{bmatrix} x_{ci}(t) \\ y_{ci}(t) \\ z_{ci}(t) \end{bmatrix} \quad \begin{array}{l} \text{position} \\ \text{commanded} \\ \text{at time } t \end{array}$$

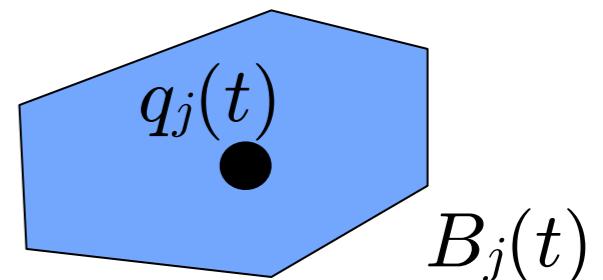
- Polytopic shape of UAV # i at time t

$$\mathcal{V}(t) = \{p(t)\} + \text{conv}(d_1(t), \dots, d_r(t))$$



- Polyhedral shape of obstacle # j at time t

$$B_j(t) = \{q_j(t)\} \oplus W_j(t)$$



- Note: other vehicles in formation are treated as polytopic obstacles ($B_j = \mathcal{V}$)

LTV-MPC GUIDANCE ALGORITHM: OPTIMIZATION MODEL

- At time t , select the desired position $p_c(t)$ for the UAV by solving the optimal control problem (=quadratic programming)

prediction horizon

$$\min \quad \rho\epsilon^2 + \sum_{k=0}^{N-1} \|W^y(p_k - p_d(t))\|^2 + \|W^{\Delta u}(p_{c,k} - p_{c,k-1})\|^2$$

desired position

s.t. $p_{k+1} = Ap_k + Bp_{c,k}, \quad k = 0, \dots, N-1$ ← UAV dynamics

bounds on input $\Delta_{\min} \leq p_{c,k} - p_{c,k-1} \leq \Delta_{\max}, \quad k = 0, \dots, N_u - 1$

increments $A_{c,k}p_k \leq b_{c,k} + g_k - A_{c,k}d_{h,k} + \mathbb{I}\epsilon \quad \leftarrow$ (soft) obstacle avoidance constraints

$k = 0, \dots, N-1, \quad h = 1, \dots, r_i$

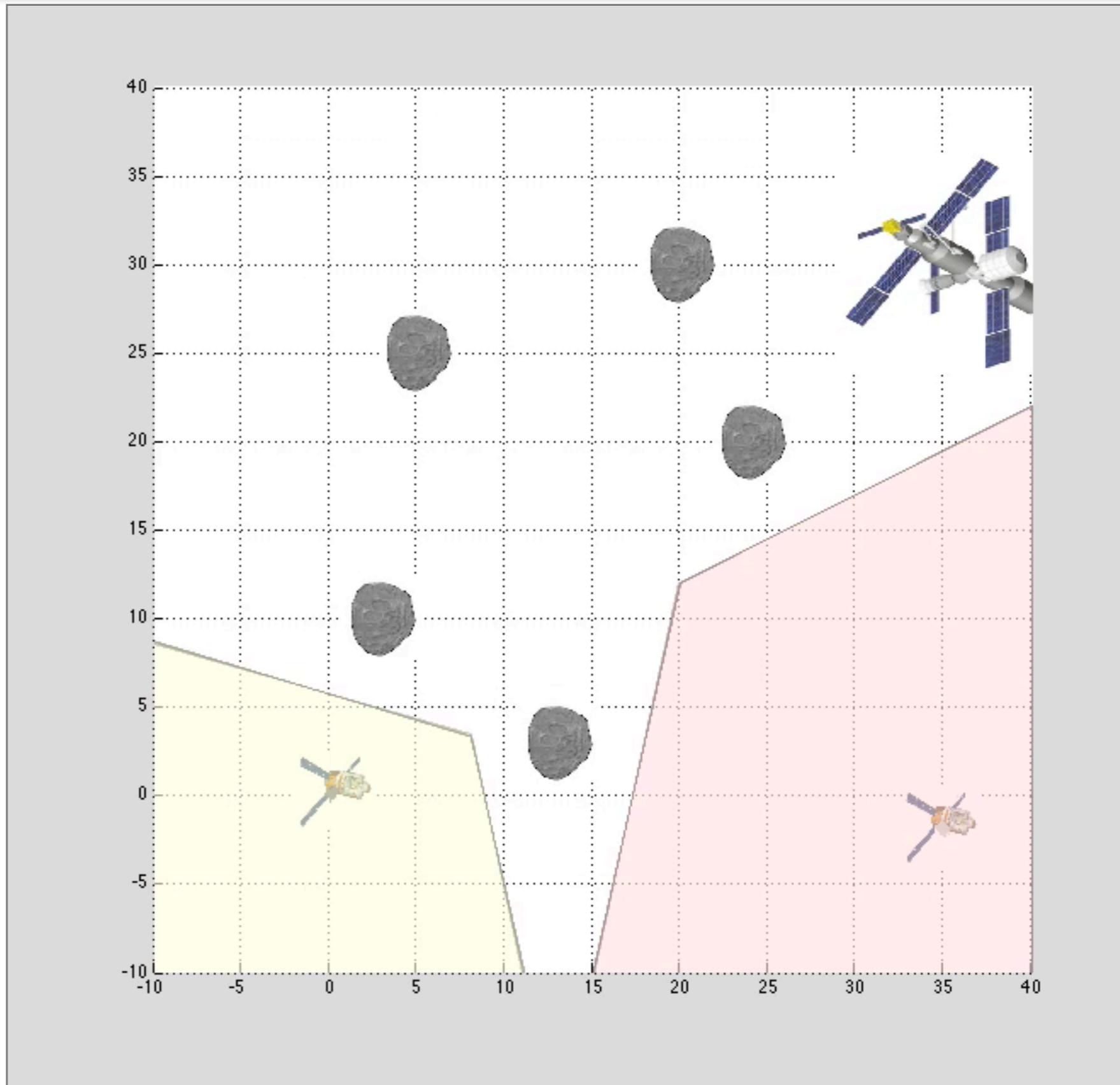
$p_{c,k} = p_{c,N_u-1}, \quad k = N_u, \dots, N \quad \leftarrow$ blocked moves

$$A_c = \begin{bmatrix} (q_1 - p_0)' \\ \vdots \\ (q_M - p_0)' \end{bmatrix}, \quad b_c = \begin{bmatrix} (q_1 - p_0)' q_1 \\ \vdots \\ (q_M - p_0)' q_M \end{bmatrix}$$

EXAMPLE: NAVIGATION DEMO

- Initial position #1 = $(0,0)$
- Target position #1 = $(35,30)$

- Initial position #2 = $(35,-3)$
- Target position #2 = $(0,20)$



ROBMPC project
Robust Model Predictive
Control (MPC) for Space
Constrained Systems

MPCSoft Toolbox
for MATLAB

(Bemporad, 2010-2011)

EXAMPLE: NAVIGATION DEMO (COOPERATIVE MPC)

- Two vehicles avoiding each other and obstacles towards their targets

