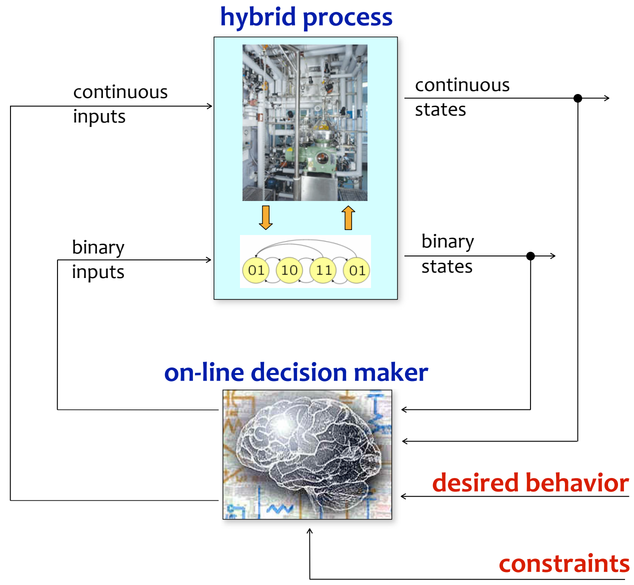
Hybrid Systems: Model Predictive Control

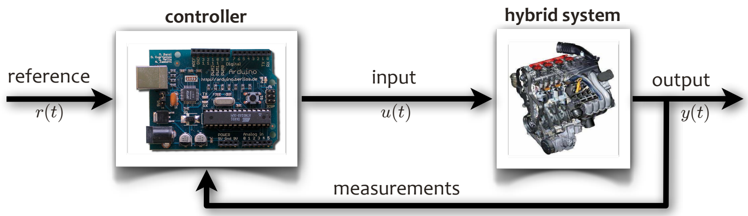
Hybrid control problem



MPC of hybrid systems

MLD model

$$\begin{cases} x_{k+1} = Ax_k + B_1u_k + B_2\delta_k + B_3z_k + B_5 \\ y_k = Cx_k + D_1u_k + D_2\delta_k + D_3z_k + D_5 \\ E_2\delta_k + E_3z_k \leq E_4x_k + E_1u_k + E_5 \end{cases}$$



- **MODEL**: use an MLD (or PWA) model of the plant to predict the future behavior of the hybrid system
- **PREDICTIVE**: optimization is based on the predicted future evolution of the hybrid system
- **CONTROL**: the goal is to control the hybrid system

MPC for hybrid systems

- At time t solve the finite-horizon optimal control problem w.r.t. $U \triangleq \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix}$

$$\begin{aligned} \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \|y_k - r\|_Q^2 + \|u_k - u_r\|_R^2 \\ & +\sigma\left(\|\delta_k - \delta_r\|^2 + \|z_k - z_r\|^2 + \|x_k - x_r\|^2\right) \\ \text{subject to} \quad & \text{MLD equations} \\ & x_0 = x(t) \\ & x_N = x_r \end{aligned}$$

notation: $\|v\|_Q^2 \triangleq v'Qv$

where the equilibrium condition $(x_r, u_r, \delta_r, z_r)$ is obtained by solving the following mixed-integer feasibility problem

$$\begin{cases} x_r = Ax_r + B_1u_r + B_2\delta_r + B_3z_r \\ r = Cx_r + D_1u_r + D_2\delta_r + D_3z_r \\ E_2\delta_r + E_3z_r \leq E_4x_r + E_1u_r + E_5 \end{cases}$$

- Apply only $u(t)=u_0^*$ (discard the remaining optimal inputs)
- At time $t+1$: get new measurements, repeat optimization

MIQP formulation of MPC

(Bemporad, Morari, 1999)

$$
\begin{array}{ll}
\min_\xi & \displaystyle\sum_{k=0}^{N-1} y_k' Q y_k + u_k' R u_k \\
\text{s.t.} & \left\{ \begin{array}{rcl} x_{k+1} & = & Ax_k + B_1 u_k + B_2\delta_k + B_3 z_k + B_5 \\ y_k & = & Cx_k + D_1 u_k + D_2\delta_k + D_3 z_k + D_5 \\ E_2\delta_k + E_3 z_k & \leq & E_4 x_k + E_1 u_k + E_5 \end{array} \right.
\end{array}
$$

- Optimization vector: $\xi = [u_0, \ldots, u_{N-1}, \delta_0, \ldots, \delta_{N-1}, z_0, \ldots, z_{N-1}]$

$$
\Longrightarrow \quad
\begin{array}{ll}
\min_\xi & \frac{1}{2}\xi' H \xi + x'(t) F \xi + \cancel{\frac{1}{2}x'(t) Y x(t)} \\
\text{s.t.} & G\xi \leq W + Sx(t)
\end{array}
$$

Mixed Integer Quadratic Program (MIQP)

$$u \in \mathbb{R}^{n_u}, \;\; \delta \in \{0,1\}^{n_\delta}, \;\; z \in \mathbb{R}^{n_z} \quad \Longrightarrow \quad \xi \in \mathbb{R}^{(n_u+n_z)N} \times \{0,1\}^{n_\delta N}$$

vector ξ has both **real** and **binary** values

Closed-loop convergence

Theorem Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium values corresponding to set point r. Assume $x(0)$ is such that the MPC problem is feasible at time t=0.

Then $\forall Q, R \succ 0$, $\forall \sigma > 0$ the closed-loop hybrid MPC loop converges asymptotically

$$
\begin{array}{rclcrcl}
\lim_{t\to\infty} y(t) & = & r & \quad & \lim_{t\to\infty} x(t) & = & x_r \\
 & & & & \lim_{t\to\infty} \delta(t) & = & \delta_r \\
\lim_{t\to\infty} u(t) & = & u_r & & \lim_{t\to\infty} z(t) & = & z_r
\end{array}
$$

and all constraints are fulfilled at each time t≥0.

(Bemporad, Morari 1999)

Proof: Easily follows from standard Lyapunov arguments

More stability results: see (Lazar, Heemels, Weiland, Bemporad, 2006)

Convergence proof

Main idea: Use **value function** $V^*(x(t))$ as a **Lyapunov** function

- Let $\xi_t = [u_0^t, \ldots, u_{N-1}^t, \delta_0^t, \ldots, \delta_{N-1}^t, z_0^t, \ldots, z_{N-1}^t]$ be the optimal sequence @t
- By construction $\bar{\xi}_{t+1} = [u_1^t, \ldots, u_{N-1}^t, u_r, \delta_1^t, \ldots, \delta_{N-1}^t, \delta_r, z_1^t, \ldots, z_{N-1}^t, z_r]$ is feasible @t+1, as it satisfies all MLD constraints and terminal constraint
- The cost of $\bar{\xi}_{t+1}$ is $V^*(x(t)) - \|y(t) - r\|_Q^2 - \|u - u_r\|_R^2$
$$-\sigma\left(\|\delta(t) - \delta_r\|^2 + \|z(t) - z_r\|^2 + \|x(t) - x_r\|^2\right) \geq V^*(x(t+1))$$
- $V^*(x(t))$ is monotonically decreasing and ≥0, so $\exists \lim_{t\to\infty} V^*(x(t)) \triangleq V_\infty$
- Hence $\|y(t) - r\|_Q^2, \;\|u(t) - u_r\|_R^2$ and $\|\delta(t) - \delta_r\|^2, \;\|z(t) - z_r\|^2, \;\|x(t) - x_r\|^2 \to 0$
- Since R,Q>0, $\lim_{t\to\infty} y(t) = r, \;\; \lim_{t\to\infty} u(t) = u_r$, and all other variables converge. $\square$

Global optimum is not needed to prove convergence !

MILP formulation of MPC

(Bemporad, Borrelli, Morari, 2000)

$$
\begin{array}{ll}
\min_\xi & \displaystyle\sum_{k=0}^{N-1} \|Qy_k\|_\infty + \|Ru_k\|_\infty \\
\text{s.t.} & \left\{ \begin{array}{rcl} x_{k+1} & = & Ax_k + B_1 u_k + B_2\delta_k + B_3 z_k + B_5 \\ y_k & = & Cx_k + D_1 u_k + D_2\delta_k + D_3 z_k + D_5 \\ E_2\delta_k + E_3 z_k & \leq & E_4 x_k + E_1 u_k + E_5 \end{array} \right.
\end{array}
\qquad
\begin{array}{l} Q \in \mathbb{R}^{m_y \times n_y} \\ R \in \mathbb{R}^{m_u \times n_u} \end{array}
$$

- Introduce slack variables: $\min_x |x| \quad \Longrightarrow \quad \begin{array}{ll} \min_{x,\epsilon} & \epsilon \\ \text{s.t.} & \epsilon \geq x, \;\; \epsilon \geq -x \end{array}$

$$
\left\{ \begin{array}{rcl} \epsilon_k^y & \geq & \|Qy_k\|_\infty \\ \epsilon_k^u & \geq & \|Ru_k\|_\infty \end{array} \right.
\Longrightarrow
\left\{ \begin{array}{rcllll} \epsilon_k^y & \geq & \pm Q^i y_k & i = 1, \ldots, m_y & k = 0, \ldots, N-1 \\ \epsilon_k^u & \geq & \pm R^i u_k & i = 1, \ldots, m_u & k = 0, \ldots, N-1 \end{array} \right.
$$

Q^i = ith row of matrix Q

- Optimization vector:
$$\xi = [\epsilon_0^y, \ldots, \epsilon_{N-1}^y, \epsilon_0^u, \ldots, \epsilon_{N-1}^u, u_0, \ldots, u_{N-1}, \delta_0, \ldots, \delta_{N-1}, z_0, \ldots, z_{N-1}]$$

$$
\Longrightarrow \quad
\begin{array}{ll}
\min_\xi & \displaystyle\sum_{k=0}^{N-1} \epsilon_k^y + \epsilon_k^u \\
\text{s.t.} & G\xi \leq W + Sx(t)
\end{array}
$$

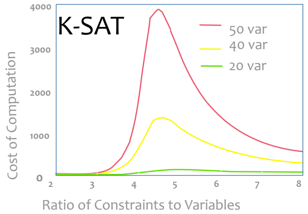
Mixed Integer Linear Program (MILP)

vector ξ has both **real** and **binary** values

Mixed-Integer Program (MIP) solvers

- Mixed-Integer Programming is *NP*-complete

Phase transitions have been found in computationally hard problems



(Monasson et al., *Nature*, 1999)

BUT

- General purpose branch & bound / branch & cut solvers available for MILP and MIQP (CPLEX, GLPK, Xpress-MP, CBC, Gurobi, ...)

  More solvers and benchmarks: http://plato.la.asu.edu/bench.html

- No need to reach global optimum (see proof of the theorem), although performance deteriorates

Hybrid MPC example

PWA system:

$$x(t+1) = 0.8\begin{bmatrix}\cos\alpha(t) & -\sin\alpha(t)\\ \sin\alpha(t) & \cos\alpha(t)\end{bmatrix}x(t) + \begin{bmatrix}0\\1\end{bmatrix}u(t)$$

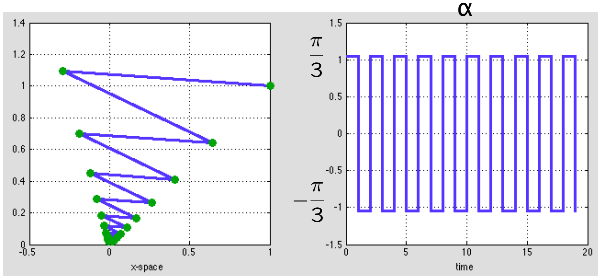
$$y(t) = x_2(t)$$

$$\alpha(t) = \begin{cases}\frac{\pi}{3} & \text{if } x_1(t) > 0\\ -\frac{\pi}{3} & \text{if } x_1(t) \le 0\end{cases}$$

constraint:

$$-1 \le u(t) \le 1$$

open-loop simulation



go to demo `/demos/hybrid/bm99sim.m`

Hybrid MPC example

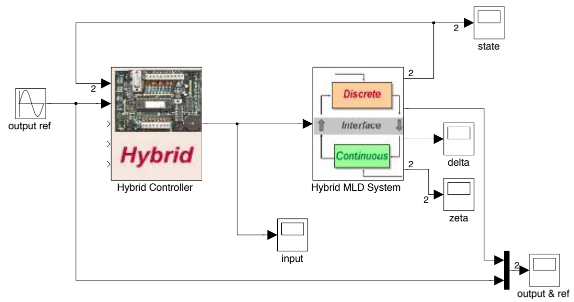
HYSDEL model

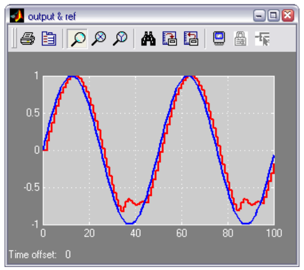
```
/* 2x2 PWA system  - Example from the paper
   A. Bemporad and M. Morari, ``Control of systems integrating logic, dynamics,
   and constraints,'' Automatica, vol. 35, no. 3, pp. 407-427, 1999.
   (C) 2003 by A. Bemporad, 2003 */

SYSTEM pwa {

INTERFACE {
            STATE { REAL x1 [-10,10];
                    REAL x2 [-10,10];}

            INPUT { REAL u [-1.1,1.1];}

            OUTPUT{ REAL y;}

            PARAMETER {
              REAL alpha = 1.0472; /* 60 deg in radiants */
              REAL C = cos(alpha);
              REAL S = sin(alpha);}
            }

IMPLEMENTATION {
            AUX { REAL z1,z2;
                  BOOL sign; }
            AD  { sign = x1<=0; }

            DA  { z1 = {IF sign THEN 0.8*(C*x1+S*x2)
                        ELSE 0.8*(C*x1-S*x2) };
                  z2 = {IF sign THEN 0.8*(-S*x1+C*x2)
                        ELSE 0.8*(S*x1+C*x2) };  }

            CONTINUOUS {x1 = z1;
                        x2 = z2+u; }

            OUTPUT { y = x2;  }
              }
}
```

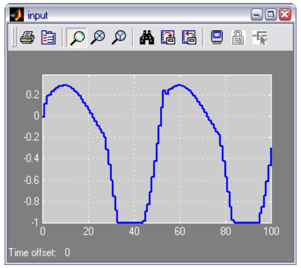
`/demos/hybrid/bm99.hys`

Hybrid MPC example

closed-loop:







performance index: $\min \sum_{k=1}^{2} |y_k - r(t)|$

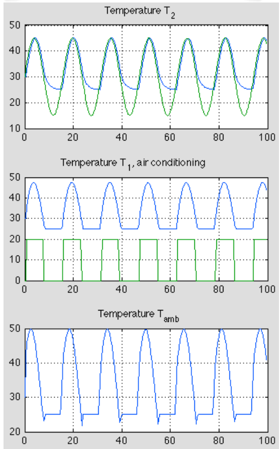
Hybrid MPC – Temperature control

```
>>refs.x=2;    % just weight state #2
>>Q.x=1;       % unit weight on state #2
>>Q.rho=Inf;   % hard constraints
>>Q.norm=Inf;  % infinity norms
>>N=2;         % prediction horizon
>>limits.xmin=[25;-Inf];
```

$$\min \sum_{k=0}^{2} \|x_{2k} - r(t)\|_\infty$$
$$\text{s.t.} \begin{cases} x_{1k} \geq 25, \; k = 1,2 \\ \text{MLD model} \end{cases}$$

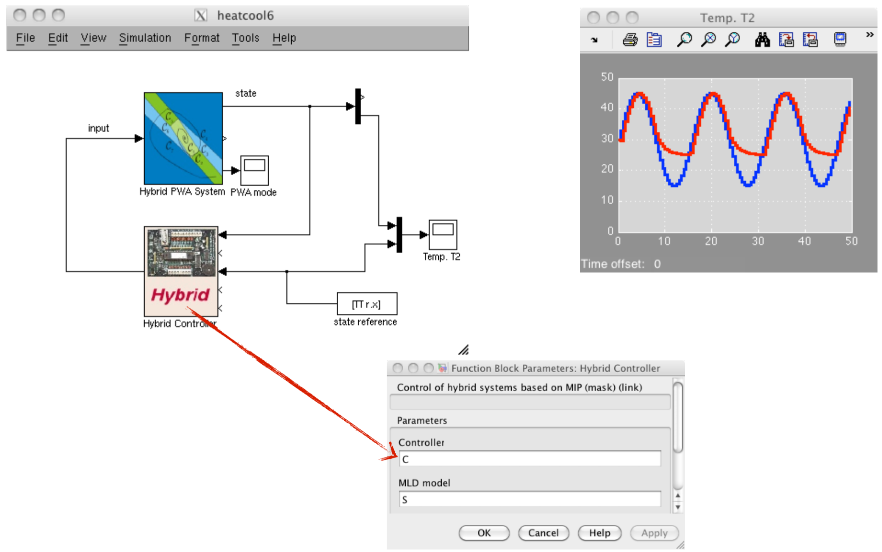
```
>>C=hybcon(S,Q,N,limits,refs);
```

```
>> C

Hybrid controller based on MLD model S <heatcoolmodel.hys> [Inf-norm]

  2 state measurement(s)
  0 output reference(s)
  0 input reference(s)
  1 state reference(s)
  0 reference(s) on auxiliary continuous z-variables

 20 optimization variable(s) (8 continuous, 12 binary)
 46 mixed-integer linear inequalities
sampling time = 0.5, MILP solver = 'glpk'

Type "struct(C)" for more details.
>>
```

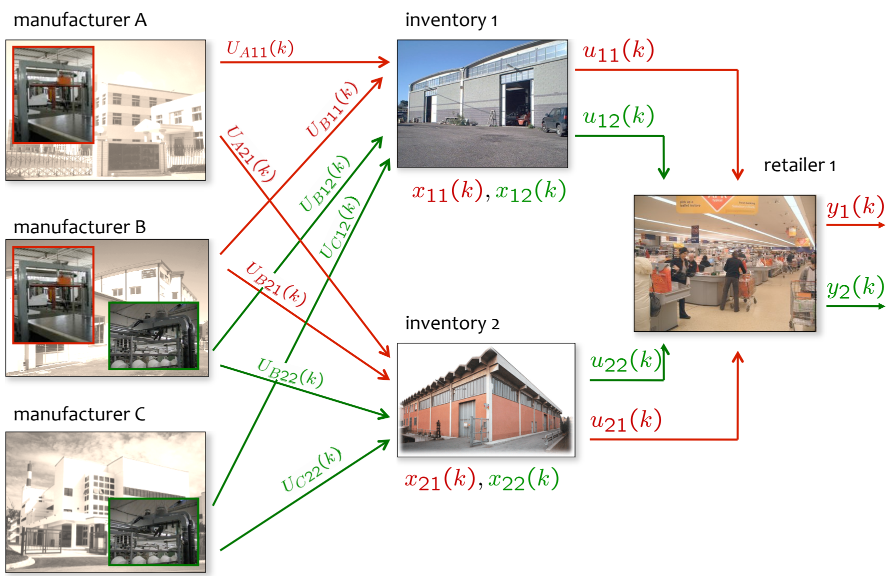



```
>>[XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tstop);
```


Hybrid MPC – Temperature control

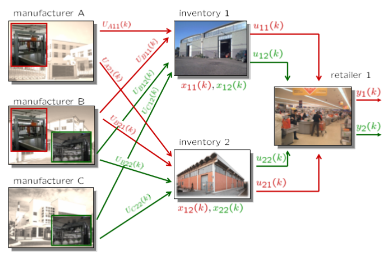


A simple example in supply chain management



System variables

- continuous states:
  $x_{ij}(k)$ = amount of j hold in inventory i at time k (i=1,2, j=1,2)
- continuous outputs:
  $y_j(k)$ = amount of j sold at time k (j=1,2)
- continuous inputs:
  $u_{ij}(k)$ = amount of j taken from inventory i at time k (i=1,2, j=1,2)
- binary inputs:
  $U_{Xij}(k)$ = 1 if manufacturer X produces and send j to inventory i at time k

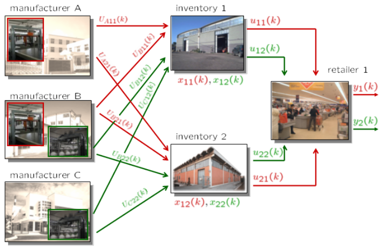


Constraints

- Max capacity of inventory i:

$0 \leq \sum_j x_{ij}(k) \leq x_{Mi}$

Numerical values:
x_{M_1}=10, x_{M_2}=10



- Max transportation from inventories:

$0 \leq u_{ij}(k) \leq u_M$

- A product can only be sent to one inventory:

  $UA_{11}(k)$ and $UA_{21}(k)$ cannot be both $=1$
  $UB_{11}(k)$ and $UB_{21}(k)$ cannot be both $=1$
  $UB_{12}(k)$ and $UB_{22}(k)$ cannot be both $=1$
  $UC_{12}(k)$ and $UC_{22}(k)$ cannot be both $=1$

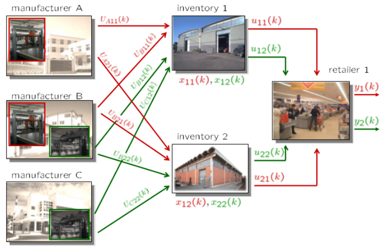
- A manufacturer can only produce one type of product at one time:

  $[UB_{11}(k)$ or $UB_{21}(k)=1]$, $[UB_{12}(k)$ or $UB_{22}(k)=1]$ cannot be both true

Dynamics

P_{A_1}, P_{B_1}, P_{B_2}, P_{C_2} = amount of product of type 1 (2) produced by A (B,C) in one time interval

Numerical values:
P_{A_1}=4, P_{B_1}=6, P_{B_2}=7, P_{C_2}=3



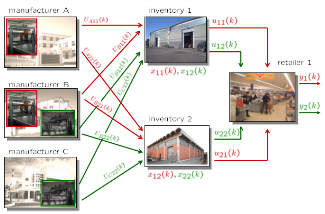
- Level of inventories:

$$\begin{cases} x_{11}(k+1) &= x_{11}(k) + P_{A1}U_{A11}(k) + P_{B1}U_{B11}(k) - u_{11}(k) \\ x_{12}(k+1) &= x_{12}(k) + P_{B2}U_{B12}(k) + P_{C2}U_{C12}(k) - u_{12}(k) \\ x_{21}(k+1) &= x_{21}(k) + P_{A1}U_{A21}(k) + P_{B1}U_{B21}(k) - u_{21}(k) \\ x_{22}(k+1) &= x_{22}(k) + P_{B2}U_{B22}(k) + P_{C2}U_{C22}(k) - u_{22}(k) \end{cases}$$

- Retailer: $\begin{cases} y_1 &= u_{11} + u_{21} \\ y_2 &= u_{12} + u_{22} \end{cases}$

Hybrid dynamical model

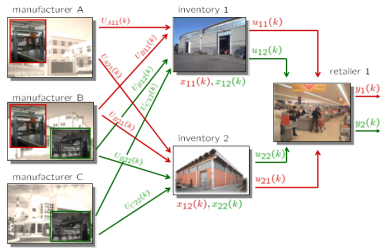
```
SYSTEM supply_chain{
INTERFACE {
        STATE { REAL x11   [0,10];
                REAL x12   [0,10];
                REAL x21   [0,10];
                REAL x22   [0,10]; }

        INPUT { REAL u11 [0,10];
                REAL u12 [0,10];
                REAL u21 [0,10];
                REAL u22 [0,10];
                BOOL UA11,UA21,UB11,UB12,UB21,UB22,UC12,UC22; }

        OUTPUT {REAL y1,y2;}

        PARAMETER { REAL PA1,PB1,PB2,PC2,xM1,xM2;}
}
IMPLEMENTATION {

        AUX { REAL zA11, zB11, zB12, zC12, zA21, zB21, zB22, zC22;}


        DA { zA11 = {IF UA11 THEN PA1 ELSE 0};
             zB11 = {IF UB11 THEN PB1 ELSE 0};
             zB12 = {IF UB12 THEN PB2 ELSE 0};
             zC12 = {IF UC12 THEN PC2 ELSE 0};
             zA21 = {IF UA21 THEN PA1 ELSE 0};
             zB21 = {IF UB21 THEN PB1 ELSE 0};
             zB22 = {IF UB22 THEN PB2 ELSE 0};
             zC22 = {IF UC22 THEN PC2 ELSE 0}; }

        CONTINUOUS {x11 = x11 + zA11 + zB11 - u11;
                    x12 = x12 + zB12 + zC12 - u12;
                    x21 = x21 + zA21 + zB21 - u21;
                    x22 = x22 + zB22 + zC22 - u22;  }

        OUTPUT {    y1 = u11 + u21;
                    y2 = u12 + u22; }

        MUST {  ~(UA11 & UA21);
            ~(UC12 & UC22);
            ~((UB11 | UB21) & (UB12 | UB22));
             ~(UB11 & UB21);
            ~(UB12 & UB22);
            x11+x12 <= xM1;
            x11+x12 >=0;
            x21+x22 <= xM2;
            x21+x22 >=0; }
} }
```



/demos/hybrid/supply_chain.m

Objectives

- Meet customer demand as much as possible: $y_1 \approx r_1$, $y_2 \approx r_2$



- Minimize transportation costs

- Fulfill all constraints

Performance specs

$$\min \sum_{k=0}^{N-1} 10(|y_{1,k} - r_1(t)| + |y_{2,k} - r_2(t)| + 4(|u_{11,k}| + |u_{12,k}|) + 2(|u_{21,k}| + |u_{22,k}|) + 1(|U_{A11,k}| + |U_{A21,k}|) + 4(|U_{B11,k}| + |U_{B12,k}| + U_{B21,k}| + |U_{B22,k}|) + 10(|U_{C12,k}| + |U_{C22,k}|)$$

penalty on demand tracking error

cost for shipping from inv.#1 to market

cost for shipping from inv.#2 to market

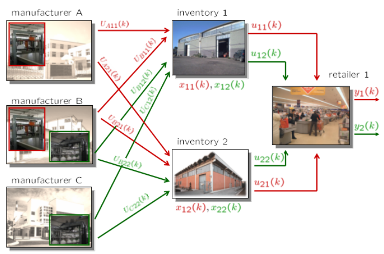
cost from A to inventories

cost from B to inventories

cost from C to inventories

Simulation setup

```
>>refs.y=[1 2];        % weights output2 #1,#2
>>Q.y=diag([10 10]);  % output weights
…
>>Q.norm=Inf;          % infinity norms
>>N=2;                 % optimization horizon
>>limits.umin=umin;    % constraints
>>limits.umax=umax;
>>limits.xmin=xmin;
>>limits.xmax=xmax;
```



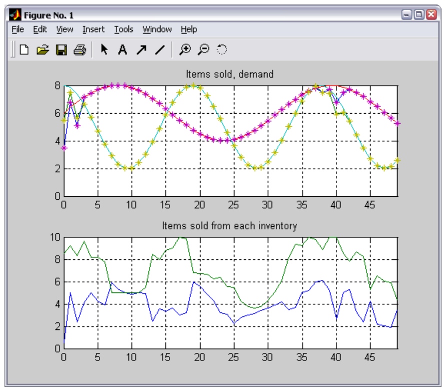

```
>>C=hybcon(S,Q,N,limits,refs);
```

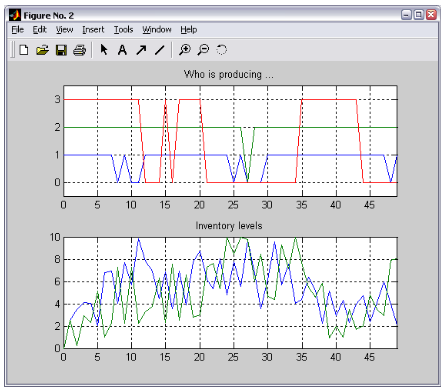
```
>> C

Hybrid controller based on MLD model S <supply_chain.hys> [Inf-norm]

  4 state measurement(s)
  2 output reference(s)
 12 input reference(s)
  0 state reference(s)
  0 reference(s) on auxiliary continuous z-variables

 44 optimization variable(s) (28 continuous, 16 binary)
176 mixed-integer linear inequalities
sampling time = 1, MILP solver = 'glpk'

Type "struct(C)" for more details.

>>
```


Simulation results

```
>>x0=[0;0;0;0];                           % Initial condition
>>r.y=[6+2*sin((0:Tstop-1)'/5)            % Reference trajectories
       5+3*cos((0:Tstop-1)'/3)];
```

```
>>[XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tstop);
```





CPU time: $\approx$ 13 ms per time step using GLPK (9 ms using CPLEX) on this machine

Explicit Hybrid MPC

Explicit hybrid MPC (MLD formulation)

$$\min_\xi J(\xi, x(t)) = \sum_{k=0}^{N-1} \|Qy_k\|_\infty + \|Ru_k\|_\infty$$

$$\text{subject to} \quad \left\{ \begin{array}{rcl} x_{k+1} & = & Ax_k + B_1u_k + B_2\delta_k + B_3z_k + B_5 \\ y_k & = & Cx_k + D_1u_k + D_2\delta_k + D_3z_k + D_5 \\ E_2\delta_k + E_3z_k & \leq & E_4x_k + E_1u_k + E_5 \\ x_0 & = & x(t) \end{array} \right.$$

- On-line optimization: solve the problem for a given state $x(t)$

Mixed-Integer Linear Program (MILP)

- Off-line optimization: solve the MILP in advance **for all** $x(t)$

$$\begin{array}{ll} \min_\xi & \sum_{k=0}^{N-1} \epsilon_k^y + \epsilon_k^u \\ \text{s.t.} & G\xi \leq W + Sx(t) \end{array}$$

multi-parametric Mixed Integer Linear Program (mp-MILP)

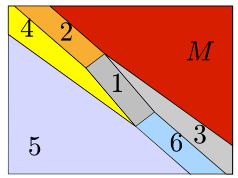
Multiparametric MILP

$$\begin{array}{ll} \min_{\xi_c, \xi_d} & f_c'\xi_c + f_d'\xi_d \\ \text{s.t.} & G_c\xi_c + G_d\xi_d \leq W + Sx \end{array} \qquad \begin{array}{l} \xi_c \in \mathbb{R}^n \\ \xi_d \in \{0,1\}^m \end{array}$$

- mp-MILP can be solved (by alternating MILPs and mp-LPs) (Dua, Pistikopoulos, 1999)
- **Theorem:** The multiparametric solution $\xi^*(x)$ is **piecewise affine** (but possibly discontinuous)
- **The MPC controller is piecewise affine in** $x = x(t)$

$$u(x) = \left\{ \begin{array}{lll} F_1x + g_1 & \text{if} & H_1x \leq K_1 \\ \vdots & & \vdots \\ F_Mx + g_M & \text{if} & H_Mx \leq K_M \end{array} \right.$$

x-space



(more generally, the MPC controller is a PWA function of x and of the reference signals)

Alternative: use PWA formulation

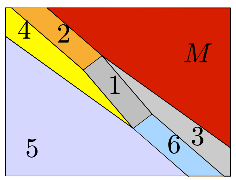
$$\min_\xi J(\xi, x(t)) = \sum_{k=0}^{N-1} \|Qy_k\|_\infty + \|Ru_k\|_\infty$$

$$\text{subject to} \quad \left\{ \begin{array}{rcl} x_{k+1} & = & A_{i(k)}x_k + B_{i(k)}u_k + f_{i(k)} \\ y_k & = & C_{i(k)}x_k + D_{i(k)}u_k + g_{i(k)} \\ & & i(k) \text{ such that } H_{i(k)}x_k + W_{i(k)}u_k \leq K_{i(k)} \\ x_0 & = & x(t) \end{array} \right.$$

- **The MPC controller is piecewise affine in** $x = x(t)$

$$u(x) = \left\{ \begin{array}{lll} F_1x + g_1 & \text{if} & H_1x \leq K_1 \\ \vdots & & \vdots \\ F_Mx + g_M & \text{if} & H_Mx \leq K_M \end{array} \right.$$

x-space



(more generally, the MPC controller is a PWA function of x and of the reference signals)

Alternative: use PWA formulation

Method #1 (Borrelli, Baotic, Bemporad, Morari, *Automatica*, 2005)

Use a combination of DP (dynamic programming) and mpLP (1-norm, ∞-norm), or mpQP (quadratic forms)

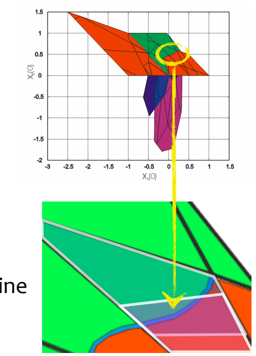
Method #2 (Bemporad, *Hybrid Toolbox*, 2003) (Alessio, Bemporad, ADHS 2006) (Mayne, ECC 2001) (Mayne, Rakovic, 2002)

1 - Use backwards (=DP) reachability analysis for enumerating all feasible mode sequences $I = \{i(0), i(1), \ldots, i(N)\}$

2 - For each fixed sequence I, solve the explicit finite-time optimal control problem for the corresponding linear time-varying system (mpQP or mpLP)

3a - Case 1/∞-norm: Compare value functions and split regions

3b - Case quadratic costs: the partition may not be fully polyhedral ⇒ better keep overlapping polyhedra and compare on-line quadratic cost functions (when overlaps are detected)



Hybrid control examples (revisited)

Hybrid control example

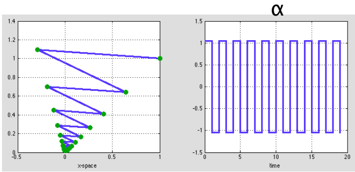
PWA system:

$$
\begin{aligned}
x(t+1) &= 0.8\begin{bmatrix}\cos\alpha(t) & -\sin\alpha(t)\\ \sin\alpha(t) & \cos\alpha(t)\end{bmatrix}x(t)+\begin{bmatrix}0\\1\end{bmatrix}u(t)\\
y(t) &= x_2(t)\\
\alpha(t) &= \begin{cases}\frac{\pi}{3} & \text{if } x_1(t)\geq 0\\ -\frac{\pi}{3} & \text{if } x_1(t)<0\end{cases}
\end{aligned}
$$

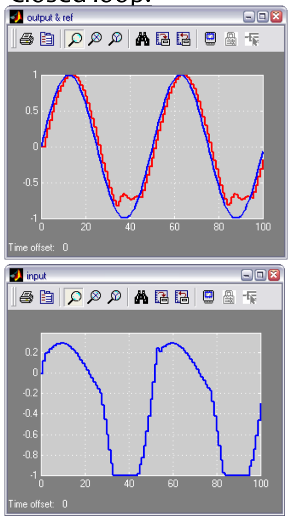
Constraints: $-1\leq u(t)\leq 1$

Objective: $\min \sum_{k=1}^{2} |y_k - r(t)|$

Open loop behavior:



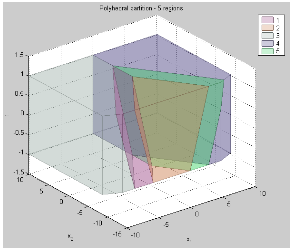
Closed loop:



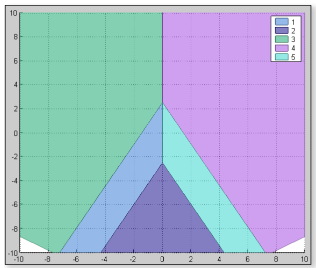
HybTbx: /**demos/hybrid/bm99sim.m**

Explicit PWA controller

$$
u(x,r)=\begin{cases}
\begin{bmatrix}0.6928 & -0.4 & 1\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix} & \text{if } \begin{bmatrix}0.6928 & -0.4 & 1\\ -0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\\ -0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix}\leq\begin{bmatrix}1\\10\\10\\1\\1\\1\\1e{-}006\end{bmatrix} \text{ (Region \#1)}\\
1 & \text{if } \begin{bmatrix}-0.6928 & 0.4 & -1\\ 0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ 0 & -1 & 0\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix}\leq\begin{bmatrix}-1\\-1\\1\\1\\10\end{bmatrix} \text{ (Region \#2)}\\
-1 & \text{if } \begin{bmatrix}-0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 1 & 0 & 0\\ 0 & 1 & 0\\ 0.6928 & -0.4 & 1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ -1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix}\leq\begin{bmatrix}10\\10\\1e{-}006\\10\\-1\\1\\1\\10\end{bmatrix} \text{ (Region \#3)}\\
-1 & \text{if } \begin{bmatrix}-1 & 0 & 0\\ 0 & 1 & 0\\ 0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ -0.6928 & -0.4 & 1\\ 0 & 0 & 1\\ 0 & 0 & -1\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix}\leq\begin{bmatrix}0\\10\\10\\10\\-1\\1\\1\\10\end{bmatrix} \text{ (Region \#4)}\\
\begin{bmatrix}-0.6928 & -0.4 & 1\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix} & \text{if } \begin{bmatrix}-0.6928 & -0.4 & 1\\ 0.4 & -0.6928 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\\ 0.6928 & 0.4 & -1\\ 0 & 0 & 1\\ -1 & 0 & 0\end{bmatrix}\begin{bmatrix}x\\r\end{bmatrix}\leq\begin{bmatrix}1\\10\\10\\1\\1\\1\\0\end{bmatrix} \text{ (Region \#5)}
\end{cases}
$$



Section with r=0



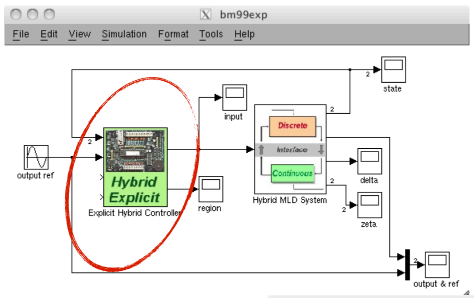
HybTbx: /**demos/hybrid/bm99sim.m**

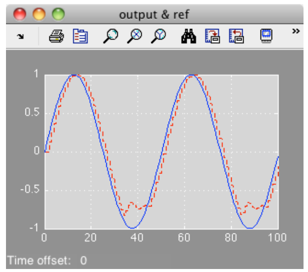
(CPU time = 1.51 s, this machine)

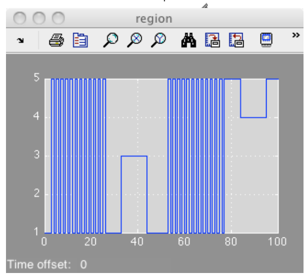
PWA law ≡ MPC law !

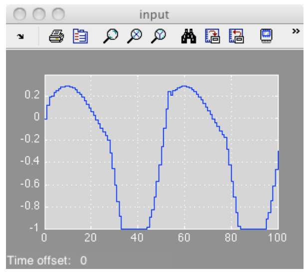
Hybrid control example

Closed loop:









Explicit PWA regulator

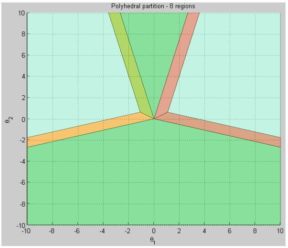
MPC problem:

$$\min \; 10\|x_N\|_\infty + \sum_{k=0}^{N-1} 10\|x_k\|_\infty + \|u_k\|_\infty$$

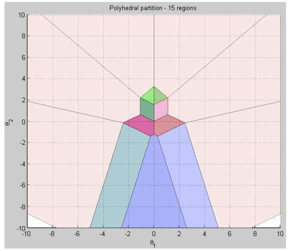
$$\text{s.t.} \; \begin{cases} -1 \le u_k \le 1, & k=0,\ldots,N-1 \\ -10 \le x_k \le 10, & k=1,\ldots,N \end{cases}$$

$$\left(\begin{array}{l} Q = \left[\begin{smallmatrix} 10 & 0 \\ 0 & 10 \end{smallmatrix}\right] \\ R = 1 \end{array} \right)$$

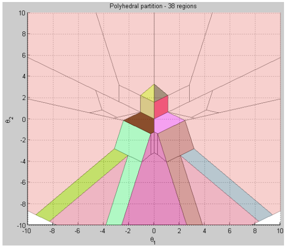
Prediction horizon N=1



Prediction horizon N=2



Prediction horizon N=3

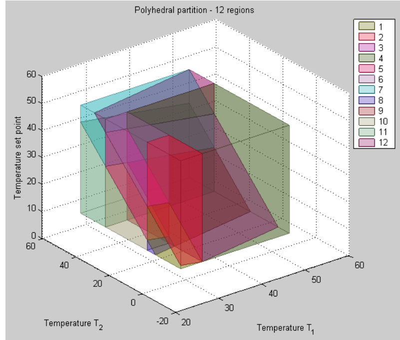


HybTbx: /**demos/hybrid/bm99benchmark.m**

Explicit MPC – Temperature control

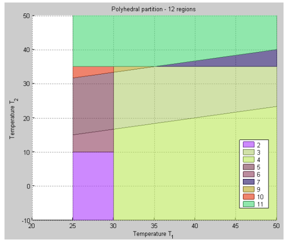
```
>>E=expcon(C,range,options);
```

```
>> E

Explicit controller (based on hybrid controller C)
  3 parameter(s)
  1 input(s)
 12 partition(s)
sampling time = 0.5

The controller is for hybrid systems (tracking)
This is a state-feedback controller.

Type "struct(E)" for more details.
>>
```



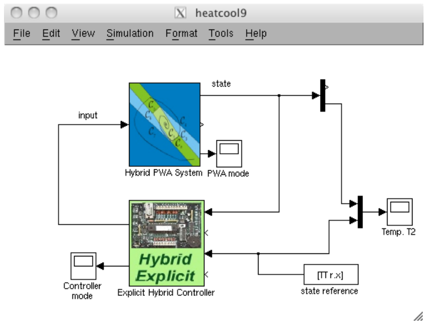
$$\min \sum_{k=0}^{2} \|x_{2k} - r(t)\|_\infty$$

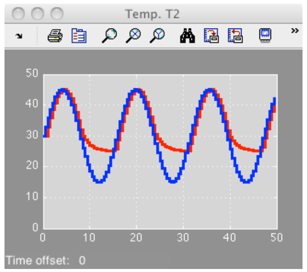
$$\text{s.t.} \; \begin{cases} x_{1k} \ge 25, \; k=1,2 \\ \text{hybrid model} \end{cases}$$



Section in the (T_1,T_2)-space for T_{ref} = 30

Explicit MPC – Temperature control

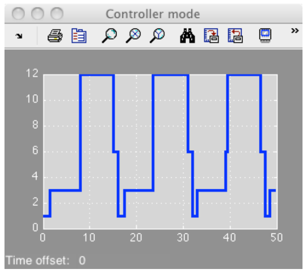




generated C-code

utils/expcon.h

```c
#define EXPCON_REG 12
#define EXPCON_NTH 3
#define EXPCON_NYM 2
#define EXPCON_NH 72
#define EXPCON_NF 12
static double EXPCON_F[]={
    -1,0,0,0,-1,0,
    -1,-1,-1,-1,-1,-1,0,-3,-3,
    -3,0,-3,0,0,0,0,0,
    0,0,4,4,4,0,4,0,0,
    0,0,0,0};

static double EXPCON_G[]={
    101.6,1.6,1.6,-1.6,98.4001,0,100,51.6,
    101.6,51.6,48.4,50};

static double EXPCON_H[]={
    0,0,0,-0.00999999,0,-0.0333333,
    0.02,0.00999999,-0.02,0,0,-0.0333333,0.02,0.00999999,
    0,0,-0.02,0.02,0,-1,0.00999999,0,
```



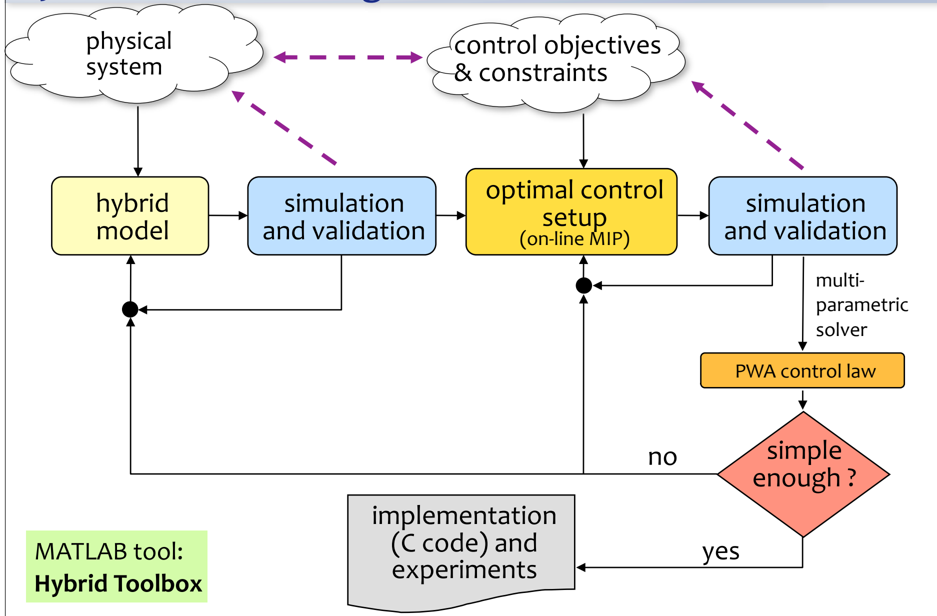
Implementation aspects of hybrid MPC

- **Alternatives:** (1) solve MIP on-line
  (2) evaluate a PWA function

- **Small problems** (short horizon N=1,2, one or two inputs): explicit PWA control law preferable
  - time to evaluate the control law is shorter than MIP
  - control code is simpler (no complex solver must be included in the control software !)
  - more insight in controller's behavior

- **Medium/large problems** (longer horizon, many inputs and binary variables): MIP preferable

Hybrid control design flow

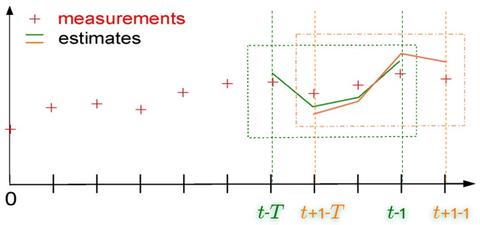


MATLAB tool:
Hybrid Toolbox

Moving Horizon Estimation
Fault Detection & Isolation

State estimation / fault detection

- Problem: given past output measurements and inputs, estimate the current states and faults
- Solution: Use Moving Horizon Estimation for MLD systems (dual of MPC)



Augment the MLD model with:

- Input disturbances $\xi \in \mathbb{R}^n$
- Output disturbances $\zeta \in \mathbb{R}^p$

At each time t solve the problem:

$$\min \sum_{k=1}^{T} \|\hat{y}(t-k|t) - y(t-k)\|^2 + \dots$$

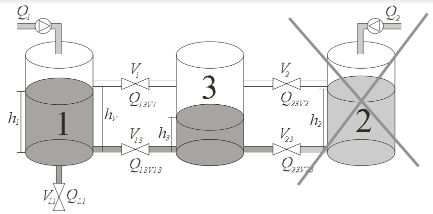
and get estimate $\hat{x}(t)$

MHE optimization = MIQP (Bemporad, Mignone, Morari, ACC 1999)

Convergence can be guaranteed (Ferrari-T., Mignone, Morari, 2002)

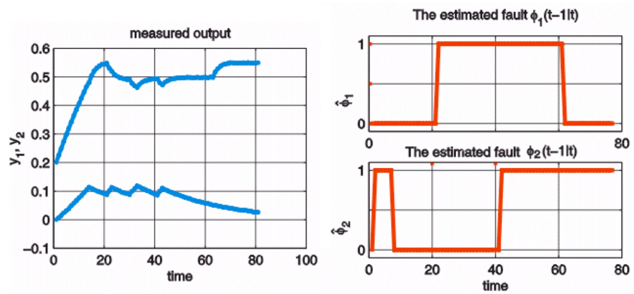
Fault detection: augment MLD with unknown **binary** distubances $\phi \in \{0,1\}^{n_f}$

Example: three tank system

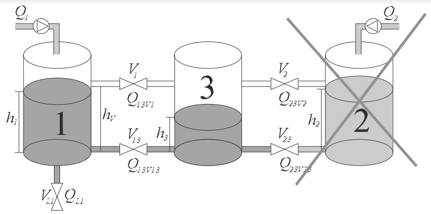


COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for 20s $\leq t \leq$ 60s
- ϕ_2 : valve V_1 blocked for $t \geq$ 40s

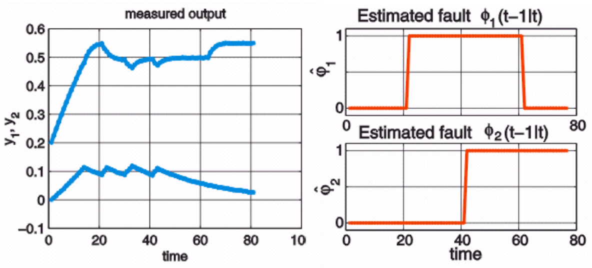


Example: Three Tank System



COSY Benchmark problem, ESF

- ϕ_1 : leak in tank 1 for 20s $\leq t \leq$ 60s
- ϕ_2 : valve V_1 blocked for $t \geq$ 40s
- Add logic constraint $[h_1 \leq h_v] \Rightarrow \phi_2 = 0$



A Few Hybrid MPC Tricks …

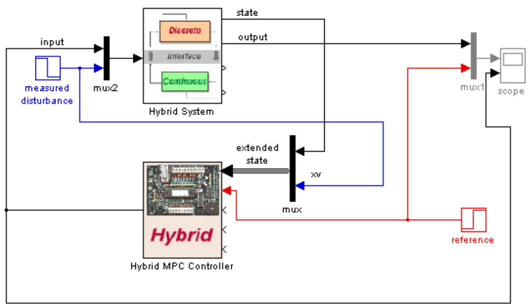
Measured disturbances

- Disturbance $v(t)$ can be measured at time t
- Augment the hybrid prediction model with a constant state

$$x_v(t+1) = x_v(t)$$

- In HYSDEL:

```
INTERFACE{
      STATE{
          REAL x      [-1e3, 1e3];
          REAL xv     [-1e3, 1e3];
      }
      ...
   }
IMPLEMENTATION{
      CONTINUOUS{
          x = A*x + B*u + Bv*xv
          xv= xv;
          ...
      }
}
```



/demos/hybrid/hyb_meas_dist.m

Note: same trick applies to linear MPC

Hybrid MPC - Tracking

- Optimal control problem (quadratic performance index):

$$\min_{\Delta U} \sum_{k=0}^{N-1} \|W^y(y_{k+1} - r(t))\|^2 + \|W^{\Delta u}\Delta u_k\|^2$$

$$[\Delta u_k \triangleq u_k - u_{k-1}],\ u_{-1} = u(t-1)$$

subj. to hybrid dynamics

$$u_{\min} \leq u_k \leq u_{\max},\ k = 0, \dots, N-1$$
$$y_{\min} \leq y_k \leq y_{\max},\ k = 1, \dots, N$$
$$\Delta u_{\min} \leq \Delta u_k \leq \Delta u_{\max},\ k = 0, \dots, N-1$$

- Note: $\|Wz\|^2 = (Wz)'(Wz) = z'(W'W)z = z'Qz$

  ⇒ same formulation as before (W = Cholesky factor of weight matrix Q)

- Optimization problem:

Mixed-Integer Quadratic Program (MIQP)

$$\min_{\Delta U} J(\Delta U, x(t)) = \tfrac{1}{2}\Delta U' H \Delta U + [x'(t)\ r'(t)\ u'(t-1)]F\Delta U$$

$$\text{s.t.}\quad G\Delta U \leq W + S \begin{bmatrix} x(t) \\ r(t) \\ u(t-1) \end{bmatrix}$$

Note: same trick as in linear MPC

Integral action in hybrid MPC

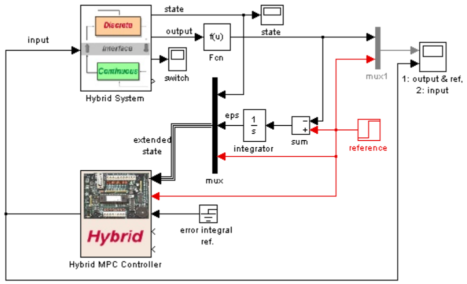
- Augment the hybrid prediction model with integrators of output errors as additional states:

$$\epsilon(k+1) = \epsilon(k) + T_s \cdot (r(k) - y(k))$$

T_s = sampling time

- Treat $r(k)$ as a measured disturbance (=additional constant state)
- Add weight on $\epsilon(k)$ in cost function to make $\epsilon(k) \to 0$
- In HYSDEL:

```
INTERFACE{
      STATE{
         REAL x          [-100,100];
         ...
         REAL epsilon    [-1e3, 1e3];
         REAL r          [0,    100]; }
      OUTPUT {
         REAL y; }
   ... }
IMPLEMENTATION{
      CONTINUOUS{
         epsilon=epsilon+Ts*(r-(c*x));
         r=r;
         ...   }
      OUTPUT{
         y=c*x; }    }
```



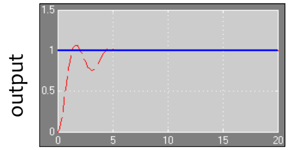
/demos/hybrid/hyb_integral_action.m

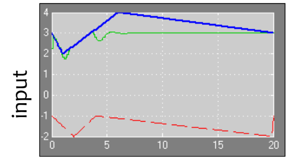
Note: same trick applies to linear MPC

Variable constraints

Problem: change upper (and/or lower) bounds on line

$$u(t) \leq u_{\max}(t)$$





1. Add a constant state and a new output in the prediction model:

$$\begin{cases} x_u(k+1) &= x_u(k) \\ y_u(k) &= x_u(k) - u(k) \end{cases}$$

2. Add output constraint

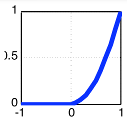
$$y_u(k) \geq 0, \quad k = 0, 1, \ldots, N$$

3. On-line implementation: feed the state back to the controller $x_u(t) = u_{\max}(t)$

Note: same trick applies to linear MPC

/demos/linear/varbounds.m

Asymmetric weights



- Say you want to weight a variable $u(k)$ only if $u(k) \geq 0$
- One way is to introduce a binary variable $[\delta=1] \leftrightarrow [u \geq 0]$, a continuous variable $z=u$ if δ=1, z=0 otherwise, and weight z
- Better solution: avoid δ and set:
- In HYSDEL:

```
INTERFACE{
      INPUT{
         REAL u   [-100,100];
         REAL z   [-1, 1e3];
      ... }
IMPLEMENTATION{
      MUST{
         z >= u;
         z >= 0;
}    }
```

$$\begin{cases} \min & (\ldots) + \sum_{k=0}^{N-1} z_k^2 \\ \text{s.t.} & z_k \geq u_k \\ & z_k \geq 0 \end{cases}$$

- When ∞-norms are used, one can do the same trick:

$$\begin{cases} \min & (\ldots) + \sum_{k=0}^{N-1} |z_k| \\ \text{s.t.} & z_k \geq u_k \\ & z_k \geq 0 \end{cases}$$

(better way: if the MILP problem constructor can be accessed, avoid introducing z_u and just remove the constraint $\epsilon_u(k) \geq -[R]^i u(k)$ used to minimize $|Ru(k)|$, with constraint $\epsilon_u(k) \geq 0$)

Note: same trick applies to linear MPC

General remarks about MIP modeling

The complexity of solving a mixed-integer program largely depends on the number of integer (binary) variables involved in the problem

Hence, when creating a hybrid model one has to

Be thrifty with binary variables !

Adding logical constraints usually helps ...

Generally speaking:

Modeling is an art

(a unifying general theory does not exist)