WORKSHOP ON

MODEL PREDICTIVE CONTROL

FROM THE BASICS TO REINFORCEMENT LEARNING

Alberto Bemporad alberto.bemporad@imtlucca.it

Mario Zanon mario.zanon@imtlucca.it

IMT SCHOOL FOR ADVANCED STUDIES LUCCA

CDC'19, Nice, France December 10, 2019

WORKSHOP PROGRAM

- Linear MPC: introduction and algorithms (AB)
- Nonlinear and economic MPC (MZ)
- **Hybrid and stochastic MPC (AB)**
- Reinforcement learning and MPC (AB+MZ)
- Concluding remarks (AB+MZ)

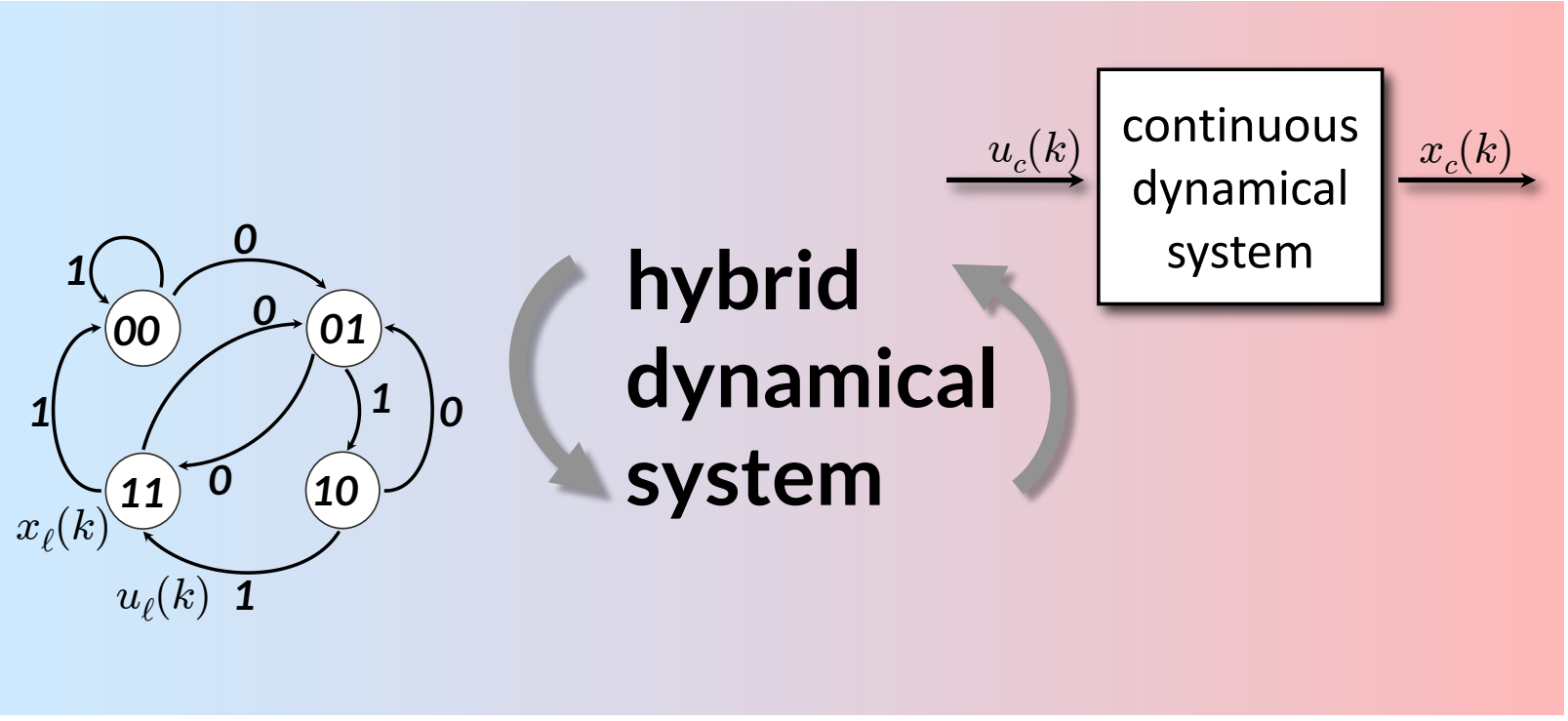
Supplementary material:

`http://cse.lab.imtlucca.it/~bemporad/mpc_course.html`

`https://mariozanon.wordpress.com/teaching/`

HYBRID MODELS

HYBRID DYNAMICAL SYSTEMS

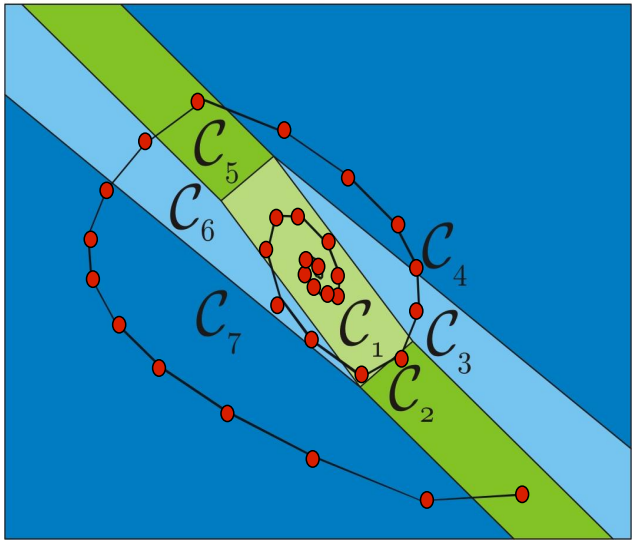


- Variables are **binary-valued**
  $x_\ell \in \{0,1\}^{n_\ell},\ u_\ell \in \{0,1\}^{m_\ell}$
- Dynamics = **finite state machine**
- **Logic constraints**

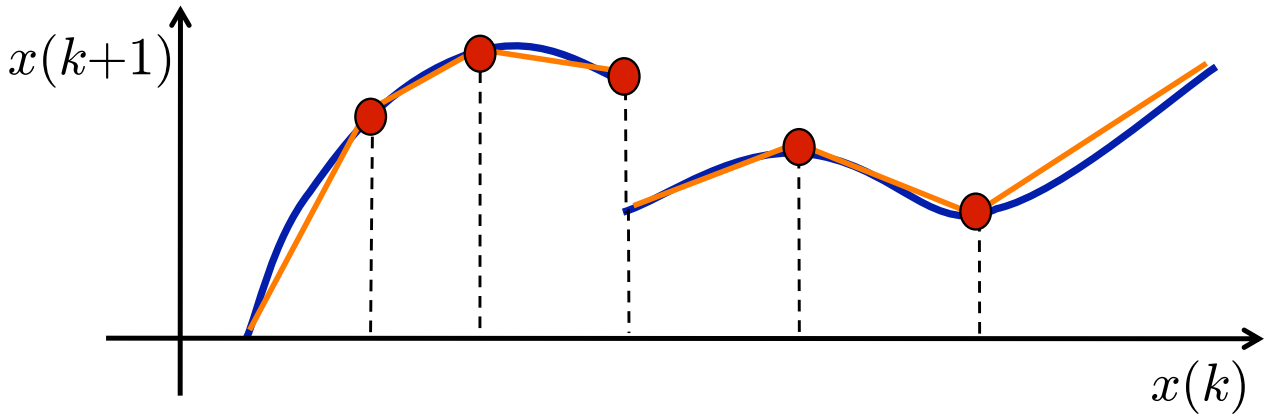
- Variables are **real-valued**
  $x_c \in \mathbb{R}^{n_c},\ u_c \in \mathbb{R}^{m_c}$
- **Difference/differential equations**
- **Linear inequality** constraints

PIECEWISE AFFINE SYSTEMS

$$
\begin{aligned}
x(k+1) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\
y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \\
\\
i(k) \quad &\text{s.t.} \quad H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)}
\end{aligned}
$$

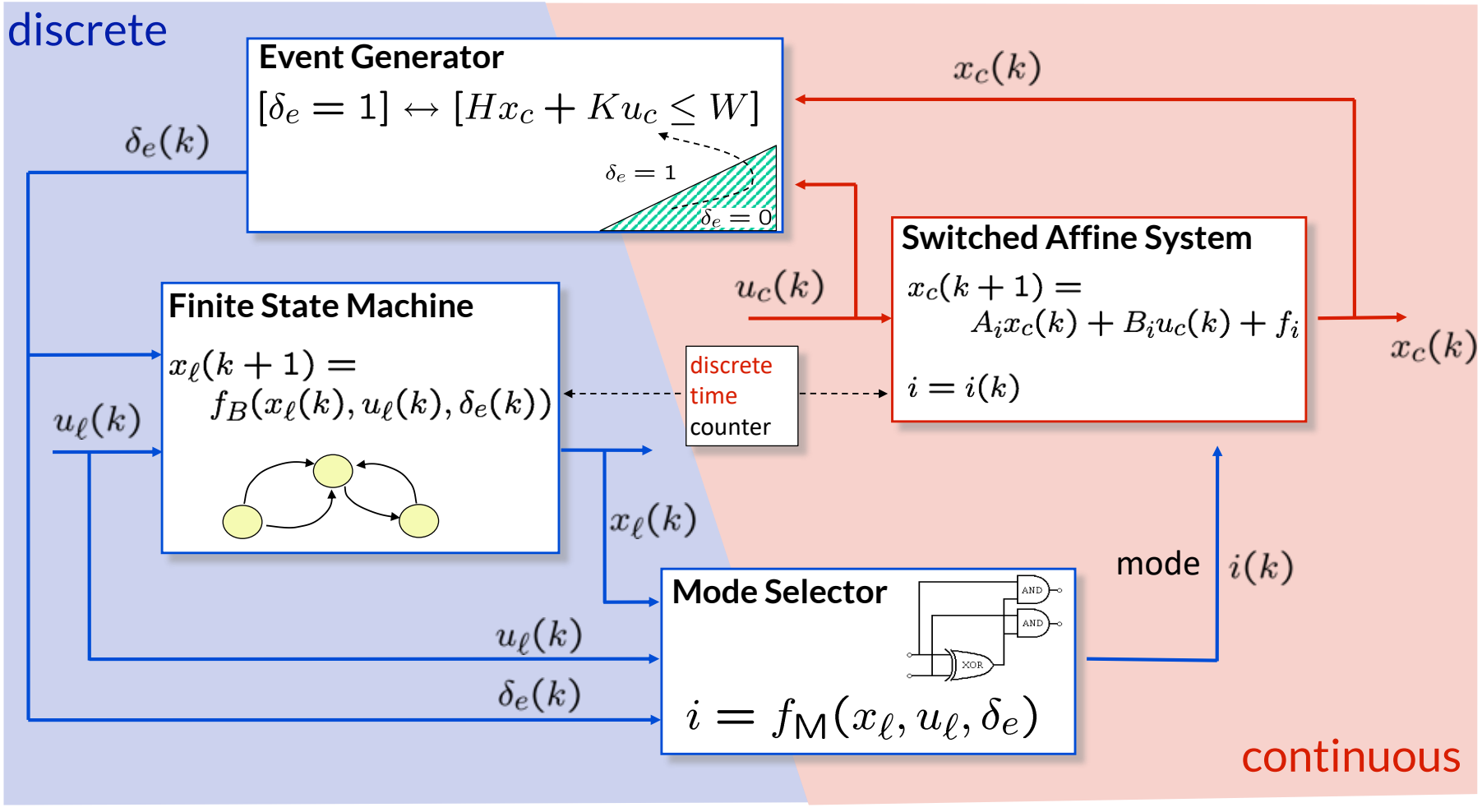


- PWA systems can approximate nonlinear dynamics arbitrarily well (even discontinuous ones)



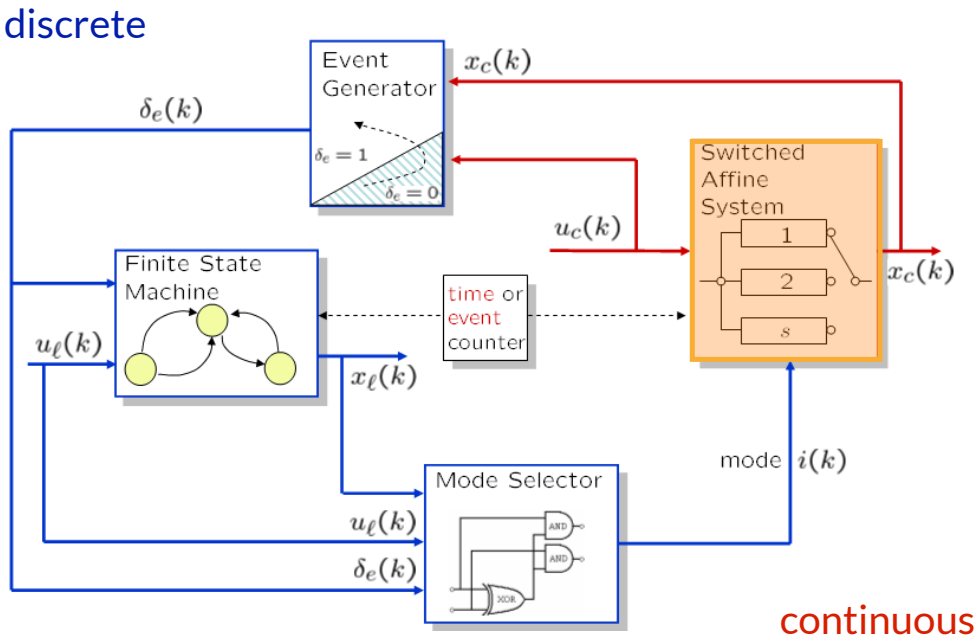
DISCRETE HYBRID AUTOMATON (DHA)

(Torrisi, Bemporad, 2004)



$x_\ell \in \{0,1\}^{n_\ell}$	=	binary state	$x_c \in \mathbb{R}^{n_c}$	=	real-valued state
$u_\ell \in \{0,1\}^{m_\ell}$	=	binary input	$u_c \in \mathbb{R}^{m_c}$	=	real-valued input
$\delta_e \in \{0,1\}^{n_e}$	=	event variable	$i \in \{1,\dots,s\}$	=	current mode

SWITCHED AFFINE SYSTEM

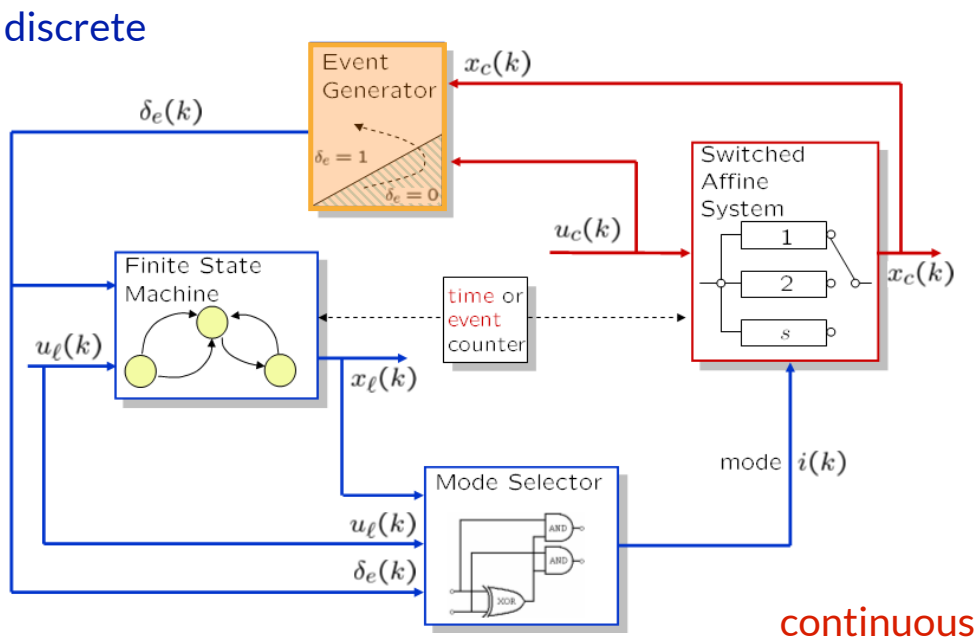


- The **affine dynamics** depend on the current mode $i(k)$:

$$x_c(k+1) = A_{i(k)}x_c(k) + B_{i(k)}u_c(k) + f_{i(k)}$$

$x_c \in \mathbb{R}^{n_c}, u_c \in \mathbb{R}^{m_c}$

EVENT GENERATOR



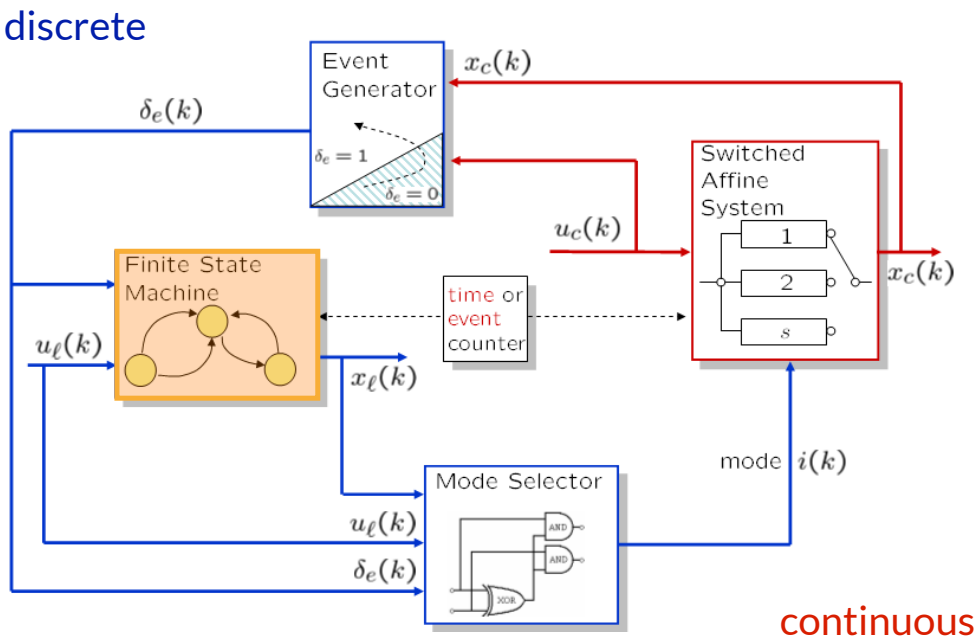
- **Event variables** are generated by **linear threshold conditions** over continuous states, continuous inputs, and time:

$$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_c(k) + K^i u_c(k) \leq W^i]$$

$$x_c \in \mathbb{R}^{n_c}, \quad u_c \in \mathbb{R}^{m_c}$$
$$\delta_e \in \{0,1\}^{n_e}$$

- Example: $[\delta_e(k) = 1] \leftrightarrow [x_c(k) \geq 0]$

FINITE STATE MACHINE



- The binary state of the **finite state machine** evolves according to a Boolean state update function $f_B : \{0,1\}^{n_\ell+m_\ell+n_e} \to \{0,1\}^{n_\ell}$:

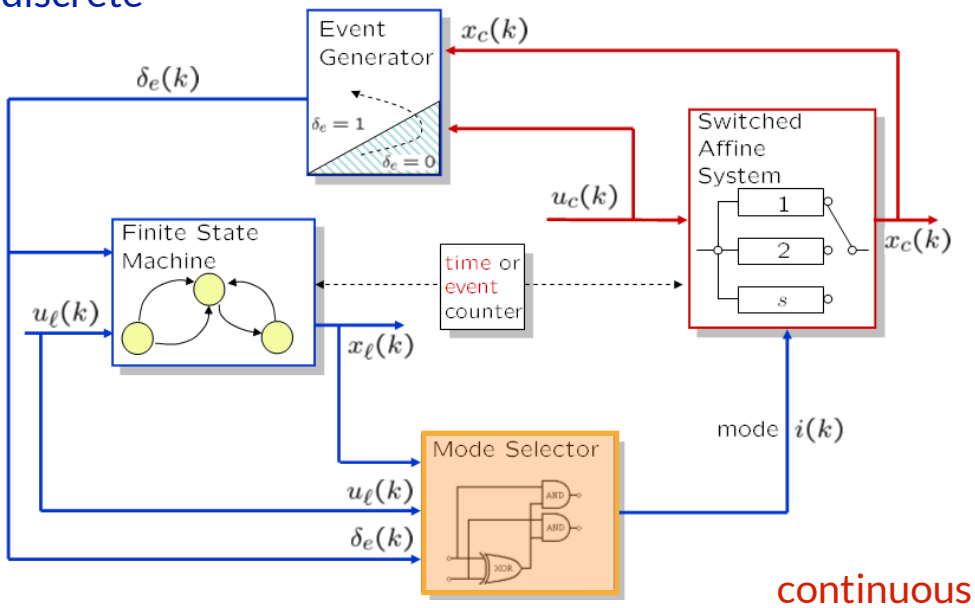
$$x_\ell(k+1) = f_B(x_\ell(k), u_\ell(k), \delta_e(k))$$

$$x_\ell \in \{0,1\}^{n_\ell}, \quad u_\ell \in \{0,1\}^{m_\ell}$$
$$\delta_e \in \{0,1\}^{n_e}$$

- Example: $x_\ell(k+1) = \neg\delta_e(k) \vee (x_\ell(k) \wedge u_\ell(k))$

MODE SELECTOR

discrete



continuous

The mode selector can be seen as the output function of the discrete dynamics

- The active **mode** $i(k)$ is selected by a Boolean function of the current binary states, binary inputs, and event variables:

$$i(k) = f_M(x_\ell(k), u_\ell(k), \delta_e(k))$$

$$x_\ell \in \{0,1\}^{n_\ell}, \quad u_\ell \in \{0,1\}^{m_\ell}$$
$$\delta_e \in \{0,1\}^{n_e}$$

- Example:

$$i(k) = \begin{bmatrix} \neg u_\ell(k) \vee x_\ell(k) \\ u_\ell(k) \wedge x_\ell(k) \end{bmatrix}$$

u_ℓ / x_ℓ	0	1
0	$i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$
1	$i = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$i = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

the system has 3 modes

CONVERSION OF LOGIC FORMULAS TO LINEAR INEQUALITIES

(Glover, 1975) (Williams, 1977) (Hooker, 2000)

- Key observation: $X_1 \vee X_2 = \texttt{true}$ $\Rightarrow$ $\delta_1 + \delta_2 \geq 1, \delta_1, \delta_2 \in \{0,1\}$
- We want to impose the Boolean statement

$$F(X_1, \ldots, X_n) = \texttt{true}$$

- Convert the formula to **Conjunctive Normal Form (CNF)**

$$\bigwedge_{j=1}^{m} \left(\bigvee_{i \in P_j} X_i \bigvee_{i \in N_j} \bar{X}_i \right) = \texttt{true}, \quad P_j \cup N_j \subseteq \{1, \ldots, n\}$$

- Transform the CNF into the equivalent linear inequalities

$$\begin{cases} \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) & \geq & 1 \\ & \vdots & \vdots \\ \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) & \geq & 1 \end{cases} \Rightarrow A\delta \leq b,\ \delta \in \{0,1\}^n$$

polyhedron

Any logic proposition can be translated into integer linear inequalities

BIG-M TECHNIQUE (IFF)

- Consider the **if-and-only-if** condition

$$[\delta = 1] \leftrightarrow [a'x_c - b \leq 0] \qquad x_c \in \mathcal{X}, \quad \delta \in \{0,1\}$$

- Assume $\mathcal{X} \subset \mathbb{R}^{n_c}$ bounded. Let M and m such that $\forall x_c \in \mathcal{X}$

$$\begin{array}{rcl} M & > & a'x_c - b \\ m & < & a'x_c - b \end{array}$$

- The if-and-only-if condition is equivalent to

$$\begin{cases} a'x_c - b & \leq & M(1-\delta) \\ a'x_c - b & > & m\delta \end{cases}$$

- We can replace the second constraint with $a'x_c - b \geq \epsilon + (m-\epsilon)\delta$ to avoid strict inequalities, where $\epsilon > 0$ is a small number (e.g., the machine precision)

BIG-M TECHNIQUE (IF-THEN-ELSE)

- Consider the **if-then-else** condition

$$z = \begin{cases} a_1' x_c - b_1 & \text{if } \delta = 1 \\ a_2' x_c - b_2 & \text{otherwise} \end{cases}$$

$$x_c \in \mathcal{X}$$
$$\delta \in \{0,1\}$$
$$z \in \mathbb{R}$$

- Assume $\mathcal{X} \subset \mathbb{R}^{n_c}$ bounded. Let M_1, M_2 and m_1, m_2 such that $\forall x_c \in \mathcal{X}$

$$\begin{array}{ccccc} M_1 & > & a_1' x_c - b_1 & > & m_1 \\ M_2 & > & a_2' x_c - b_2 & > & m_2 \end{array}$$

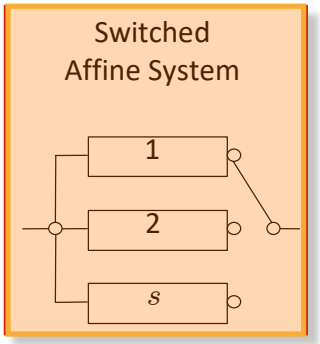
- The if-then-else condition is equivalent to

$$\begin{cases} (m_1 - M_2)(1-\delta) + z & \leq & a_1' x_c - b_1 \\ (m_2 - M_1)(1-\delta) - z & \leq & -(a_1' x_c - b_1) \\ (m_2 - M_1)\delta + z & \leq & a_2' x_c - b_2 \\ (m_1 - M_2)\delta - z & \leq & -(a_2' x_c - b_2) \end{cases}$$

SWITCHED AFFINE SYSTEM

- The state-update equation of a SAS can be rewritten as

$$x_c(k+1) = \sum_{i=1}^{s} z_i(k) \qquad z_i(k) \in \mathbb{R}^{n_c}$$



with

$$z_1(k) = \begin{cases} A_1 x_c(k) + B_1 u_c(k) + f_1 & \text{if } \delta_1(k) = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\vdots$$

$$z_s(k) = \begin{cases} A_s x_c(k) + B_s u_c(k) + f_s & \text{if } \delta_s(k) = 1 \\ 0 & \text{otherwise} \end{cases}$$

and with $\delta_i(k) \in \{0, 1\}$ subject to the **exclusive or** condition

$$\sum_{i=1}^{s} \delta_i(k) = 1 \text{ or equivalently } \begin{cases} \sum_{i=1}^{s} \delta_i(k) \geq 1 \\ \sum_{i=1}^{s} \delta_i(k) \leq 1 \end{cases}$$

- Output eqs $y_c(k) = C_i x_c(k) + D_i u_c(k) + g_i$ admit similar transformation

TRANSFORMATION OF A DHA INTO LINEAR (IN)EQUALITIES

$X_1 \vee X_2 = \text{TRUE} \quad \rightarrow \quad \delta_1 + \delta_2 \geq 1, \qquad \delta_1, \delta_2 \in \{0,1\}$

Any logic statement
$f(X) = \text{TRUE}$

$$\bigwedge_{j=1}^{m} \left(\vee_{i \in P_j} X_i \vee_{i \in N_j} \neg X_i \right) \quad \text{(CNF)}$$

$N_j, P_j \subseteq \{1, \ldots, n\}$

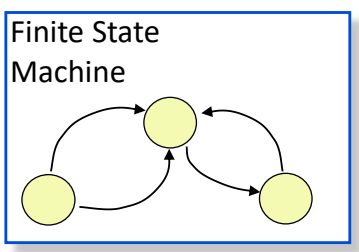
$$\rightarrow \quad \begin{cases} 1 \leq \sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \\ \vdots \\ 1 \leq \sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \end{cases}$$

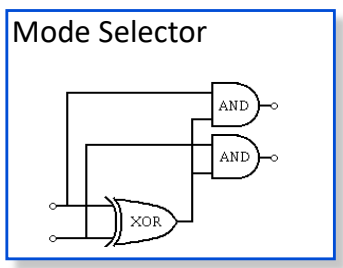
$[\delta_e^i(k) = 1] \leftrightarrow [H^i x_c(k) \leq W^i]$

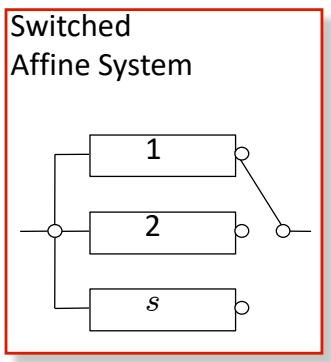
$$\rightarrow \quad \begin{cases} H^i x_c(k) - W^i \leq M^i (1 - \delta_e^i(k)) \\ H^i x_c(k) - W^i > m^i \delta_e^i(k) \end{cases}$$

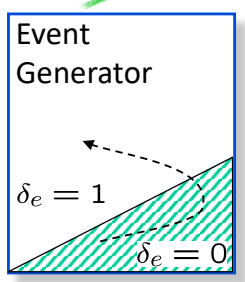
IF $[\delta = 1]$ THEN $z = a_1^T x + b_1^T u + f_1$
ELSE $z = a_2^T x + b_2^T u + f_2$

$$\rightarrow \quad \begin{cases} (m_1 - M_2)(1 - \delta) + z \leq a_1 x + b_1 u + f_1 \\ (m_2 - M_1)(1 - \delta) - z \leq -a_1 x - b_1 u - f_1 \\ (m_2 - M_1)\delta + z \leq a_2 x + b_2 u + f_2 \\ (m_1 - M_2)\delta - z \leq -a_2 x - b_2 u - f_2 \end{cases}$$





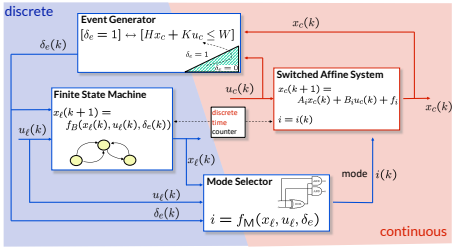




MIXED LOGICAL DYNAMICAL (MLD) SYSTEMS

(Bemporad, Morari, 1999)

- By converting logic relations into mixed-integer linear inequalities a DHA can be rewritten as the **Mixed Logical Dynamical (MLD)** system



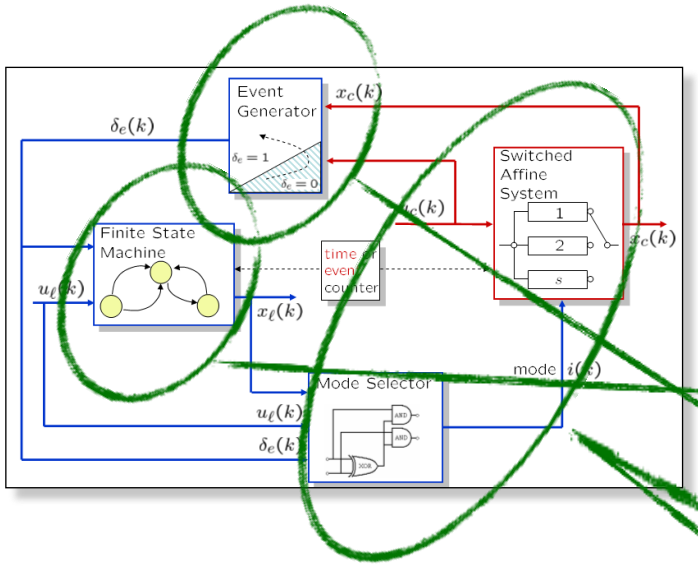
$$\begin{cases} x(k+1) &= Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) + B_5 \\ y(k) &= Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) + D_5 \\ E_2 \delta(k) &+ E_3 z(k) \leq E_4 x(k) + E_1 u(k) + E_5 \end{cases}$$

$$x \in \mathbb{R}^{n_c} \times \{0,1\}^{n_b},\ u \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b}$$
$$y \in \mathbb{R}^{p_c} \times \{0,1\}^{p_b},\ \delta \in \{0,1\}^{r_b},\ z \in \mathbb{R}^{r_c}$$

- The translation from DHA to MLD can be automatized, see e.g. the language **HYSDEL** (HYbrid Systems DEscription Language) (Torrisi, Bemporad, 2004)

- MLD models allow solving MPC, verification, state estimation, and fault detection problems via **mixed-integer programming**

DHA AND HYSDEL MODELS




```
SYSTEM name {
   INTERFACE {
      STATE {
         REAL xc [xmin,xmax];
         BOOL xl; }
      INPUT {
         REAL uc [umin,umax];
         BOOL ul; }
      PARAMETER {
         REAL param1 = 1;}
   } /* end of interface */

   IMPLEMENTATION {
      AUX { BOOL d;
            REAL z; }

      AUTOMATA { xl = xl & ~ul; }

      AD { d = xc - 1 <= 0; }

      DA { z = { IF d THEN 2*xc ELSE -xc }; }

      CONTINUOUS {
         xc = z; }

      MUST {
         xc + uc <= 2;
         ~(xl & ul); }
   } /* end implementation */
} /* end system */
```

Additional relations constraining system's variables

EQUIVALENCE OF HYBRID MODELS

- **MLD** and **PWA** systems are equivalent (Bemporad, Ferrari-Trecate, Morari, 2000)

  Proof: For a given combination (x_ℓ, u_ℓ, δ) of an MLD model, the state and output equation are linear and valid in a polyhedron.

  Conversely, a PWA system can be modeled as MLD system (see next slide)

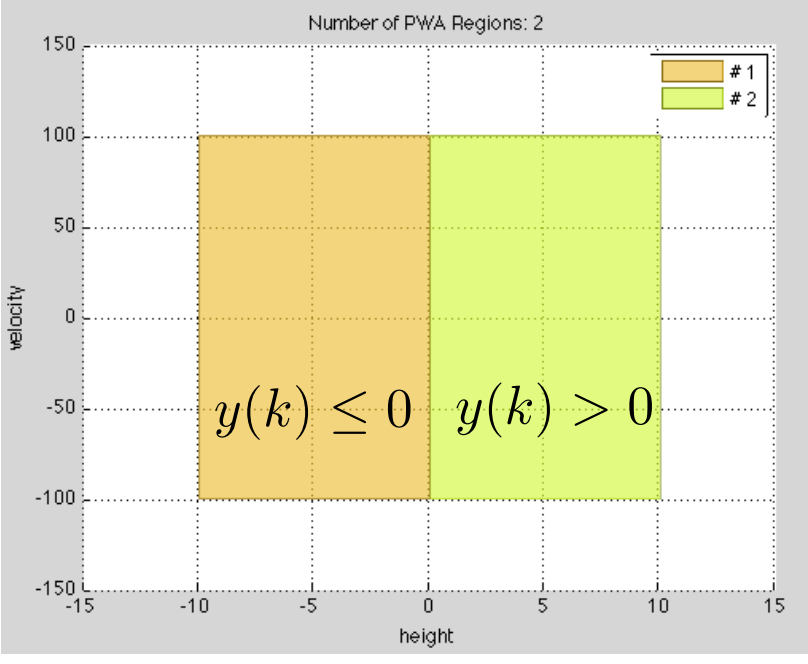
- Efficient **conversion algorithms** from MLD to PWA form exist

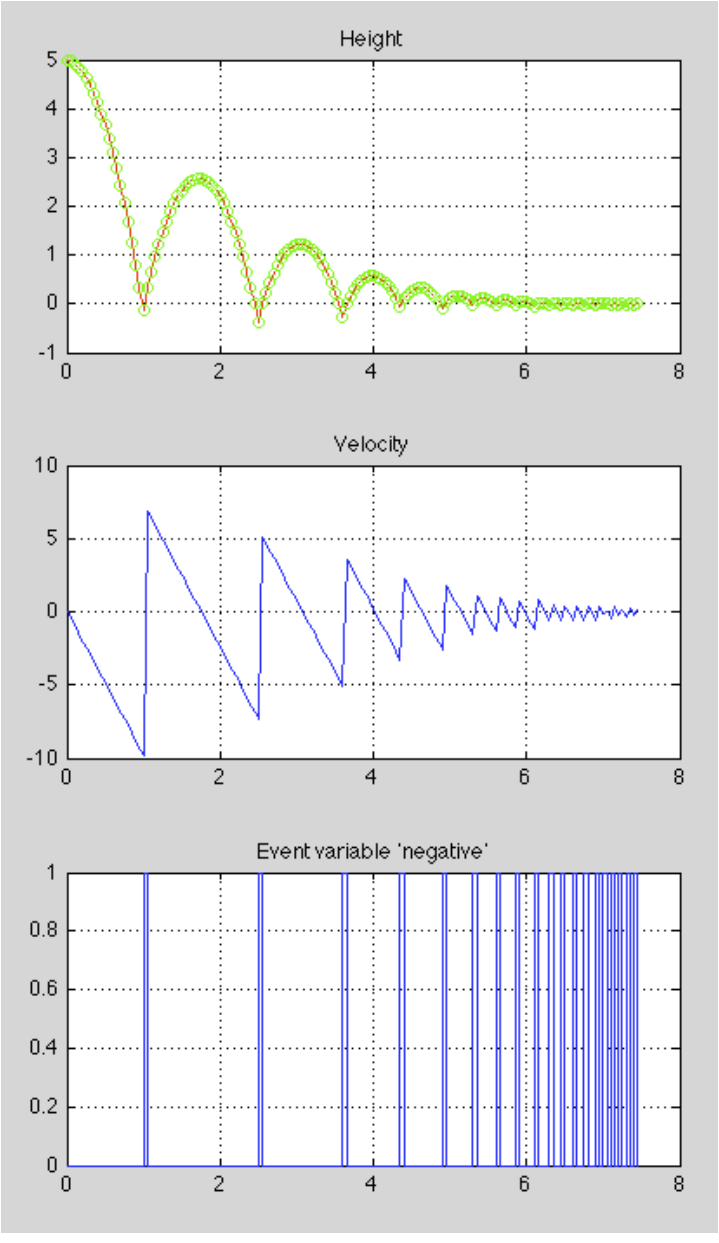
  (Bemporad, 2004) (Geyer, Torrisi, Morari, 2003)

- Further equivalences exist with other classes of hybrid dynamical systems, such as **Linear Complementarity (LC)** systems (Heemels, De Schutter, Bemporad, 2001)

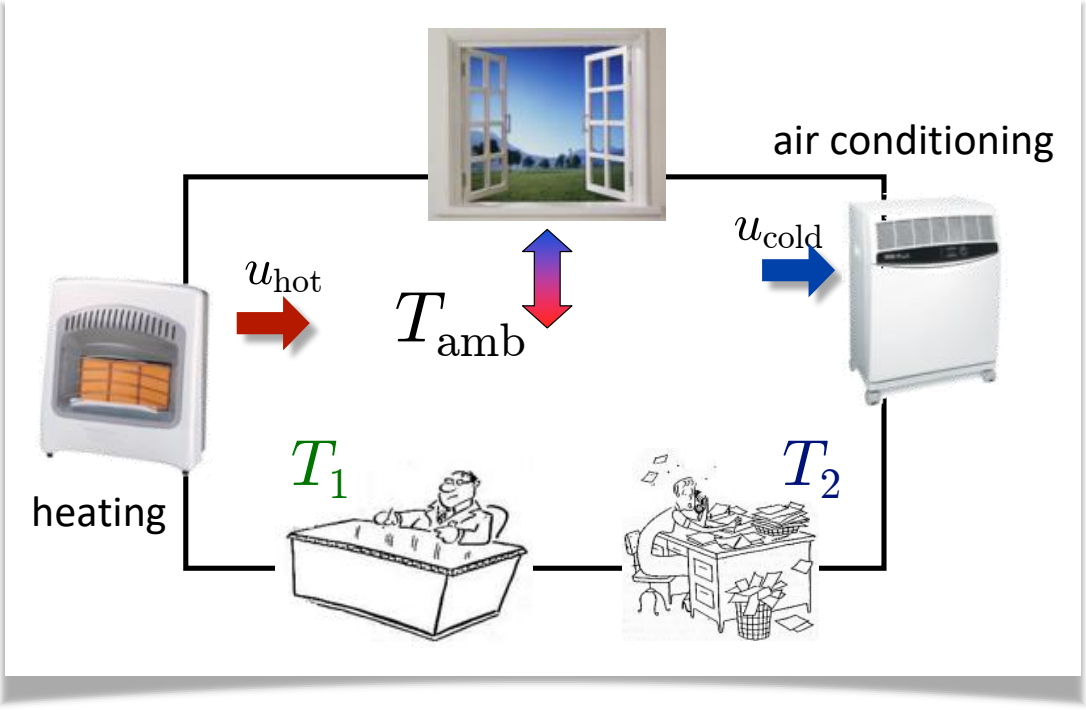
BOUNCING BALL - PWA EQUIVALENT

```
>> P=pwa(S);
>> plot(P)

>> [X,T,I]=sim(P,x0,U);
```





EXAMPLE: ROOM TEMPERATURE CONTROL



discrete dynamics

- #1 = cold $\rightarrow$ heater = on
- #2 = cold $\rightarrow$ heater = on **unless** #1 hot
- A/C activation has similar rules

continuous dynamics

$$\frac{dT_i}{dt} = -\alpha_i(T_i - T_{\text{amb}}) + k_i(u_{\text{hot}} - u_{\text{cold}})$$

$$i = 1, 2$$

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

EXAMPLE: ROOM TEMPERATURE CONTROL

```
SYSTEM heatcool {

INTERFACE {
    STATE { REAL T1 [-10,50];
            REAL T2 [-10,50];
          }
    INPUT { REAL Tamb [-10,50];
          }
    PARAMETER {
        REAL Ts, alpha1, alpha2, k1, k2;
        REAL Thot1, Tcold1, Thot2, Tcold2, Uc, Uh;
        }
}
IMPLEMENTATION {
        AUX { REAL uhot, ucold;
              BOOL hot1, hot2, cold1, cold2;
        }
        AD  { hot1 = T1>=Thot1;
              hot2 = T2>=Thot2;
              cold1 = T1<=Tcold1;
              cold2 = T2<=Tcold2;
        }
        DA  { uhot = (IF cold1 | (cold2 & ~hot1) THEN Uh ELSE 0);
              ucold = (IF hot1 | (hot2 & ~cold1) THEN Uc ELSE 0);
        }
        CONTINUOUS { T1 = T1+Ts*(-alpha1*(T1-Tamb)+k1*(uhot-ucold));
                     T2 = T2+Ts*(-alpha2*(T2-Tamb)+k2*(uhot-ucold));
        }
    }
}
```

```
>> S=mld('heatcoolmodel',Ts);
```

get the MLD model in MATLAB

```
>> [XX,TT]=sim(S,x0,U);
```

simulate the MLD model

EXAMPLE: ROOM TEMPERATURE CONTROL

- MLD model of the room temperature system

$$\left\{\begin{array}{rcl} x(k+1) & = & Ax(k)+B_1u(k)+B_2\delta(k)+B_3z(k)+B_5 \\ y(k) & = & Cx(k)+D_1u(k)+D_2\delta(k)+D_3z(k)+D_5 \\ E_2\delta(k) & + & E_3z(k) \leq E_4x(k)+E_1u(k)+E_5 \end{array}\right.$$

  - 2 continuous states (temperature T_1, T_2)
  - 1 continuous input (room temperature $T_{\rm amb}$)
  - 2 auxiliary continuous vars (power flows $u_{\rm hot}, u_{\rm cold}$)
  - 6 auxiliary binary vars (4 threshold events + 2 for the OR condition)
  - 20 mixed-integer inequalities

- In principle we have $2^6 = 64$ possible combinations of integer variables

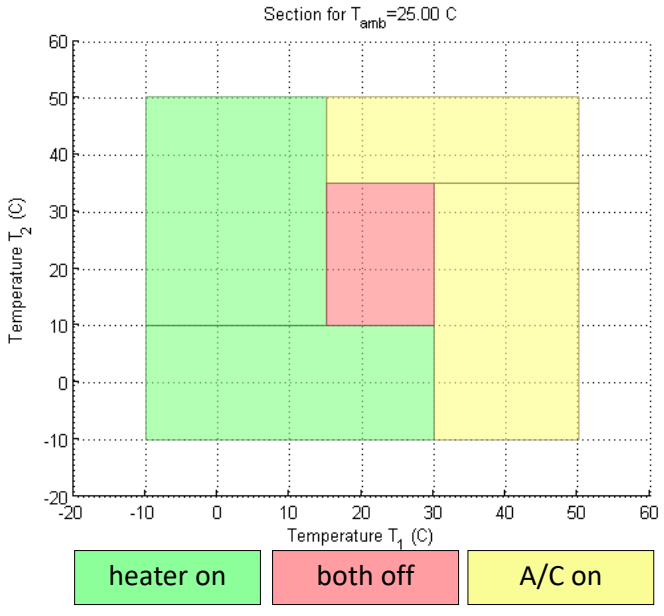
EXAMPLE: ROOM TEMPERATURE CONTROL

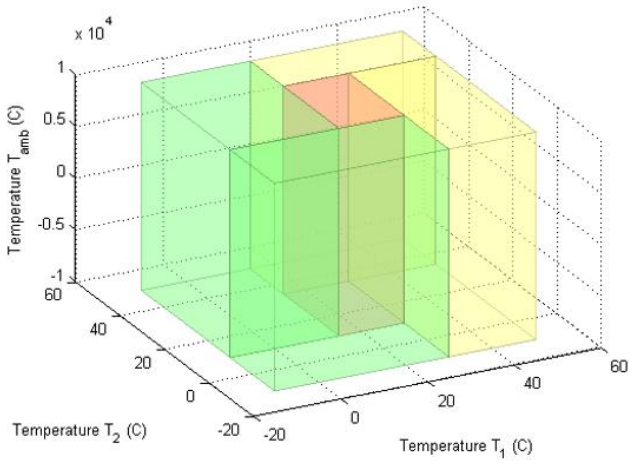
- PWA model of the room temperature system

$$\begin{aligned} x(k+1) &= A_{i(k)}x(k) + B_{i(k)}u(k) + f_{i(k)} \\ y(k) &= C_{i(k)}x(k) + D_{i(k)}u(k) + g_{i(k)} \end{aligned}$$

$$i(k) \quad \text{s.t.} \quad H_{i(k)}x(k) + J_{i(k)}u(k) \leq K_{i(k)}$$

```
>> P=pwa(S);
```





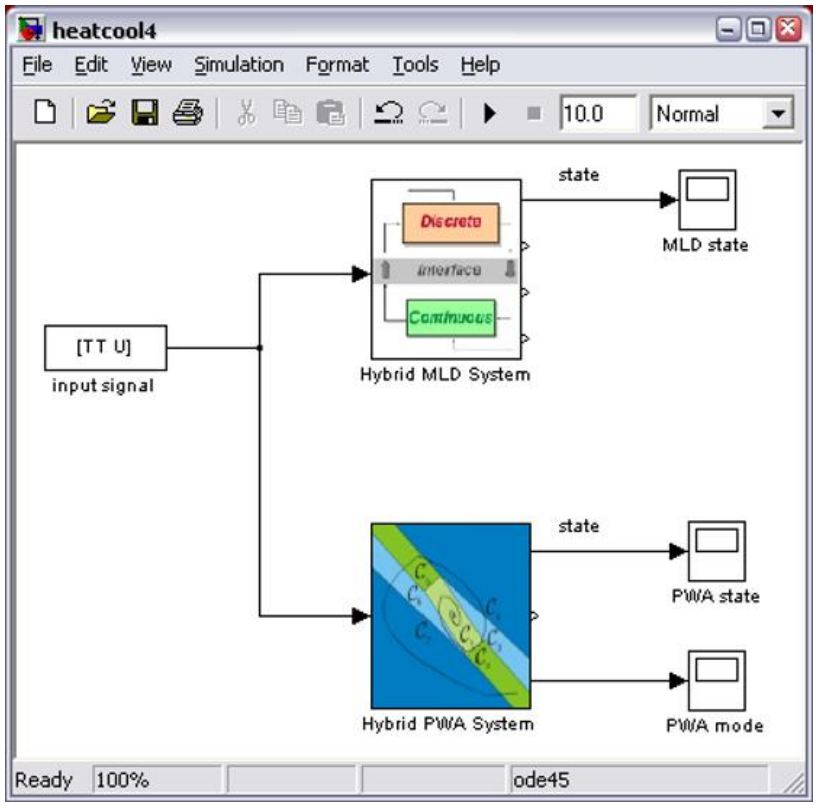
5 polyhedral regions

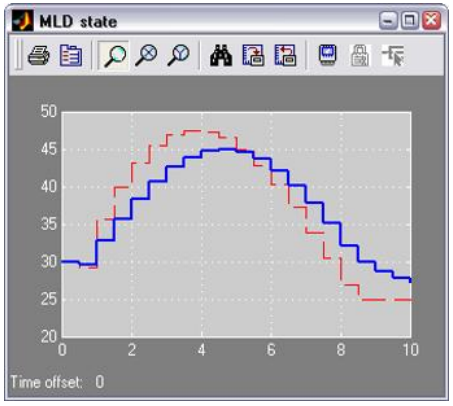
(partition does not depend on input)

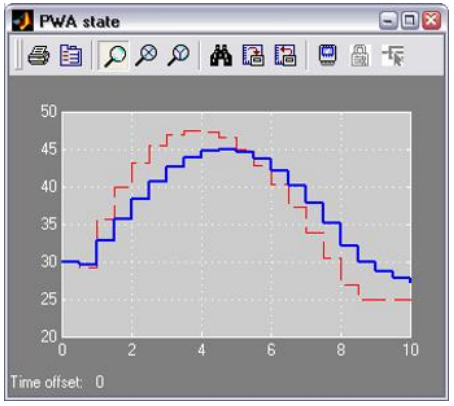
2 continuous states (T_1, T_2)

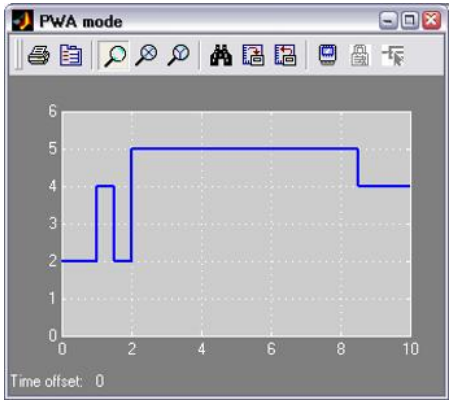
1 continuous input (T_{amb})

EXAMPLE: ROOM TEMPERATURE CONTROL



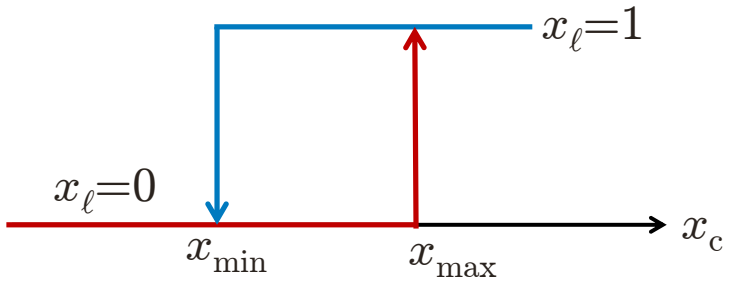






- MLD and PWA models are equivalent, hence simulated states are the same

MODELING HYSTERESIS



- Hysteresis between $x_{\min} \leq x_c(k) \leq x_{\max}$
- Introduce two binary variables

$$[\delta_{\min}(k) = 1] \quad \leftrightarrow \quad [x_c(k) \leq x_{\min}]$$
$$[\delta_{\max}(k) = 1] \quad \leftrightarrow \quad [x_c(k) \geq x_{\max}]$$

- Introduce logic state $x_\ell \in \{0, 1\}$ with dynamics

$$x_\ell(k+1) = (x_\ell(k) \wedge \neg\delta_{\min}(k)) \vee (\neg x_\ell(k) \wedge \delta_{\max}(k))$$

CHOICE CONSTRAINTS

- **Logic constraint**: make one or more **choices** out of a set of alternatives:
  - make **at most one** choice: $\delta_1 + \delta_2 + \delta_3 \leq 1$
  - make **at least two** choices: $\delta_1 + \delta_2 + \delta_3 \geq 2$
  - **exclusive or** constraint: $\delta_1 + \delta_2 + \delta_3 = 1$

- More generally:

$$\sum_{i=1}^{N} \delta_i \leq m \qquad \text{choose at most } m \text{ items out of } N$$

$$\sum_{i=1}^{N} \delta_i = m \qquad \text{choose exactly } m \text{ items out of } N$$

$$\sum_{i=1}^{N} \delta_i \geq m \qquad \text{choose at least } m \text{ items out of } N$$

"NO-GOOD" CONSTRAINTS

- Given a binary vector $\bar{\delta} \in \{0,1\}^n$ we want to impose the constraint

$$\delta \neq \bar{\delta}$$

- This may be useful for example to extract different solutions from an MIP that has multiple optima
- The **"no-good"** condition can be expressed equivalently as

$$\sum_{i \in T} \delta_i - \sum_{i \in F} \delta_i \leq -1 + \sum_{i=1}^{n} \bar{\delta}_i \qquad \begin{aligned} F &= \{i : \bar{\delta}_i = 0\} \\ T &= \{i : \bar{\delta}_i = 1\} \end{aligned}$$

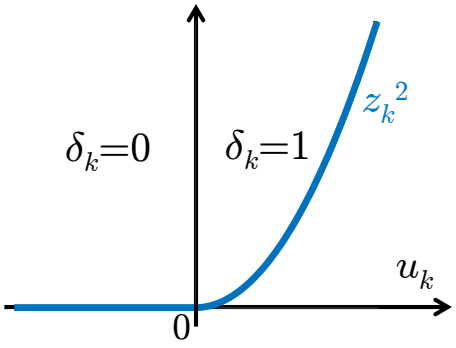
or

$$\sum_{i=1}^{n} (2\bar{\delta}_i - 1)\delta_i \leq \sum_{i=1}^{n} \bar{\delta}_i - 1$$

ASYMMETRIC WEIGHTS

- **Asymmetric weight**: only weight a variable u_k if $u_k \geq 0$
- We can introduce a binary variable $[\delta_k = 1] \leftrightarrow [u_k \geq 0]$ and

$$z_k = \max\{u_k, 0\} = \begin{cases} u_k & \text{if } \delta_k = 1 \\ 0 & \text{otherwise} \end{cases}$$



then weight z_k instead of u_k

- **Better solution**: only introduce auxiliary variable z_k and optimize

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

- Similar approach when $\|\cdot\|_\infty$ or $\|\cdot\|_1$ are used as penalties
- 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

IDENTIFICATION OF HYBRID SYSTEMS

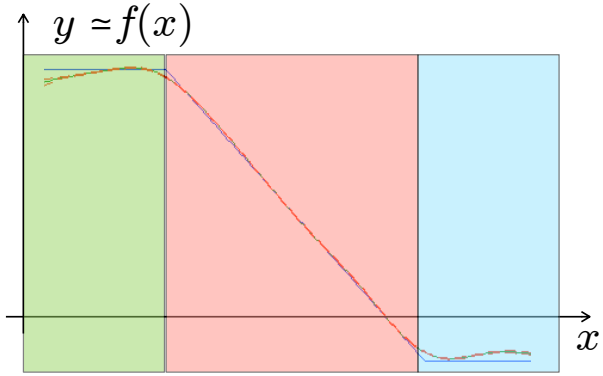
PWA REGRESSION PROBLEM

- **Problem**: Given input/output pairs $\{x(k), y(k)\}, k = 1, \ldots, N$ and number s of models, compute a **piecewise affine** (PWA) approximation $y \approx f(x)$

$$f(x) = \begin{cases} F_1 x + g_1 & \text{if } H_1 x \leq K_1 \\ \vdots & \\ F_s x + g_s & \text{if } H_s x \leq K_s \end{cases}$$

- Need to learn **both** the parameters $\{F_i,\ g_i\}$ of the affine submodels **and** the partition $\{H_i,\ K_i\}$ of the PWA map from data (**off-line learning**)

- Possibly update model and partition as new data become available (**on-line learning**)



APPROACHES TO PWA IDENTIFICATION

- Mixed-integer linear or quadratic programming (Roll, Bemporad, Ljung, 2004)
- Partition of infeasible set of inequalities (Bemporad, Garulli, Paoletti, Vicino, 2005)
- K-means clustering in a feature space (Ferrari-Trecate, Muselli, Liberati, Morari, 2003)
- Bayesian approach (Juloski, Wieland, Heemels, 2004)
- Kernel-based approaches (Pillonetto, 2016)
- Hyperplane clustering in data space (Münz, Krebs, 2002)
- **Recursive multiple least squares & PWL separation** (Breschi, Piga, Bemporad, 2016)

PWA REGRESSION ALGORITHM

(Breschi, Piga, Bemporad, 2016)

1. Estimate models $\{F_i,\ g_i\}$ **recursively**. Let $e_i(k) = y(k) - F_i x(k) - g_i$ and only update model $i(k)$ such that

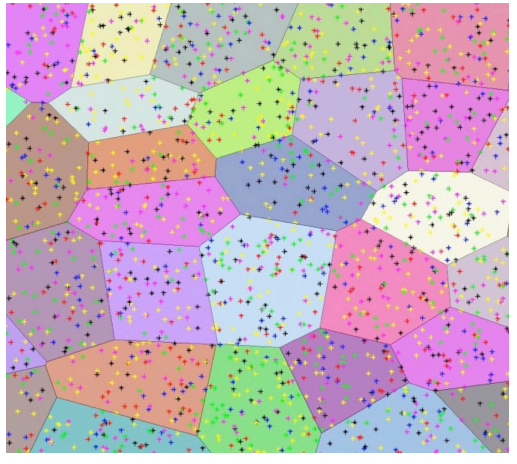
$$i(k) \leftarrow \arg\min_{i=1,\dots,s} \underbrace{e_i(k)' \Lambda_e^{-1} e_i(k)}_{\text{one-step prediction error of model \#i}} + \underbrace{(x(k)-c_i)' R_i^{-1} (x(k)-c_i)}_{\text{proximity to centroid of cluster \#i}}$$

using **recursive LS** and **inverse QR decomposition** (Alexander, Ghirnikar, 1993)

This also splits the data points $x(k)$ in **clusters** $C_i = \{x(k) :\ i(k) = i\}$

2. Compute a polyhedral partition $\{H_i,\ K_i\}$ of the regressor space via **multi-category linear separation**

$$\phi(x) = \max_{i=1,\dots,s} \{w_i' x - \gamma_i\}$$



PWA REGRESSION EXAMPLES

(Breschi, Piga, Bemporad, 2016)

- Identification of **piecewise-affine ARX** model

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} -0.83 & 0.20 \\ 0.30 & -0.52 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} -0.34 & 0.45 \\ -0.30 & 0.24 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix}$$
$$+ \begin{bmatrix} 0.20 \\ 0.15 \end{bmatrix} + \max \left\{ \begin{bmatrix} 0.20 & -0.90 \\ 0.10 & -0.42 \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} \right.$$
$$\left. + \begin{bmatrix} 0.42 & 0.20 \\ 0.50 & 0.64 \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + \begin{bmatrix} 0.40 \\ 0.30 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\} + e_o(k),$$

- **Quality of fit**: best fit rate (BFR) = $\max \left\{ 1 - \frac{\|y_{o,i} - \hat{y}_i\|_2}{\|y_{o,i} - \bar{y}_{o,i}\|_2}, 0 \right\}, i = 1, 2$

		$N = 4000$	$N = 20000$	$N = 100000$
y_1	(Off-line) RLP	96.0 %	96.5 %	99.0 %
	(Off-line) RPSN	96.2 %	96.4 %	98.9 %
	(On-line) ASGD	86.7 %	95.0 %	96.7 %
y_2	(Off-line) RLP	96.2 %	96.9 %	99.0 %
	(Off-line) RPSN	96.3 %	96.8 %	99.0 %
	(On-line) ASGD	87.4 %	95.2 %	96.4 %

RLP = Robust linear programming
(Bennett, Mangasarian, 1994)

RPSN = Piecewise-smooth Newton method
(Bemporad, Bernardini, Patrinos, 2015)

ASGD = Averaged stochastic gradient descent
(Bottou, 2012)

- **CPU time for computing the partition**:

	$N = 4000$	$N = 20000$	$N = 100000$
(Off-line) RLP	0.308 s	3.227 s	112.435 s
(Off-line) RPSN	0.016 s	0.086 s	0.365 s
(On-line) ASGD	0.013 s	0.023 s	0.067 s

PWA REGRESSION EXAMPLES

(Breschi, Piga, Bemporad, 2016)

- Identification of **linear parameter varying ARX** model

$$\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} = \begin{bmatrix} \bar{a}_{1,1}(p(k)) & \bar{a}_{1,2}(p(k)) \\ \bar{a}_{2,1}(p(k)) & \bar{a}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} y_1(k-1) \\ y_2(k-1) \end{bmatrix} + \begin{bmatrix} \bar{b}_{1,1}(p(k)) & \bar{b}_{1,2}(p(k)) \\ \bar{b}_{2,1}(p(k)) & \bar{b}_{2,2}(p(k)) \end{bmatrix} \begin{bmatrix} u_1(k-1) \\ u_2(k-1) \end{bmatrix} + e_o(k)$$

$\bar{a}(p)$ = PWA function of p

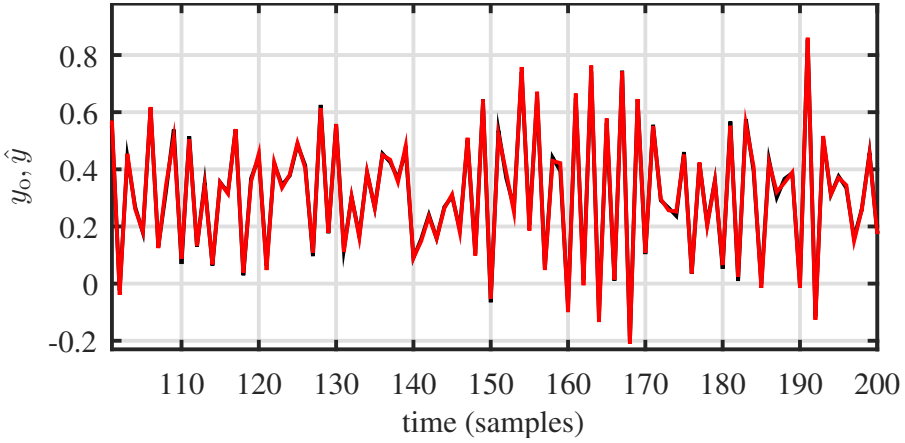
$\bar{b}(p)$ has **quadratic** and **sin** terms

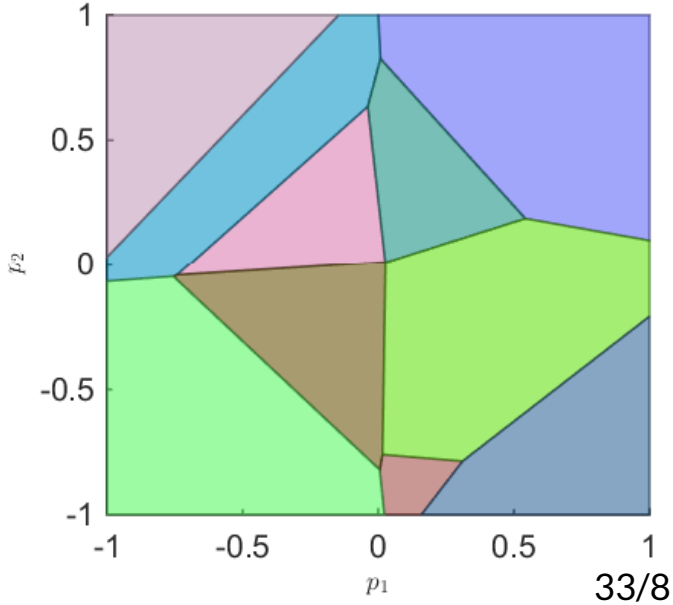
- **Quality of fit (BFR)**:

	y_1	y_2
PWA regression	87 %	84 %
parametric LPV*	80 %	70 %

* (Bamieh, Giarré, 2002)

- **Validation data (open-loop)**:

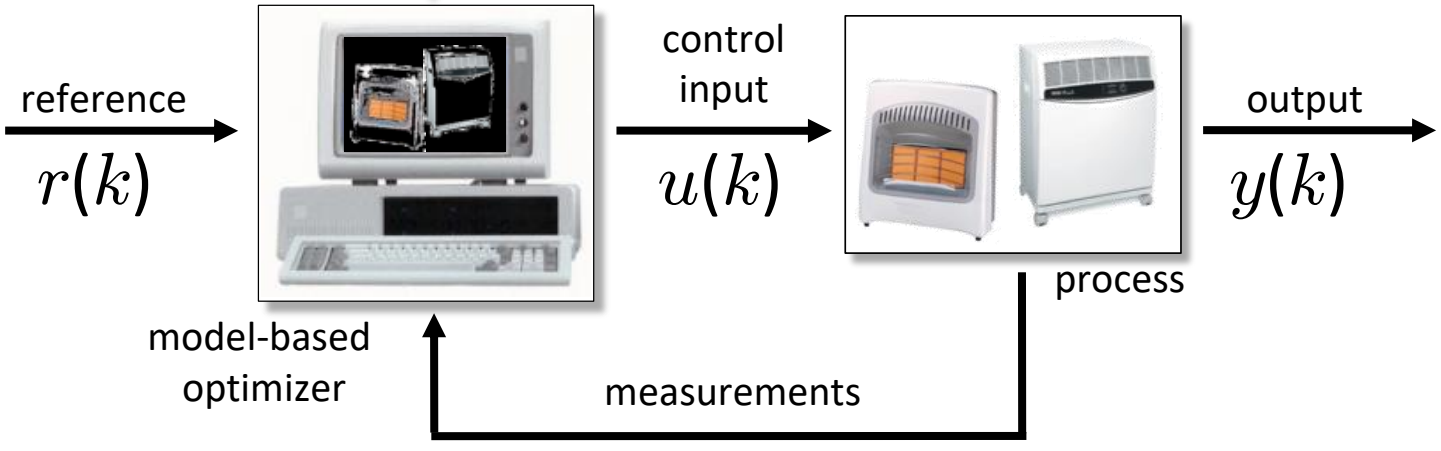




HYBRID MPC

HYBRID MODEL PREDICTIVE CONTROL

$$\begin{cases} x(k+1) &= Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) + B_5 \\ y(k) &= Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) + D_5 \\ E_2 \delta(k) &+ E_3 z(k) \leq E_4 x(k) + E_1 u(k) + E_5 \end{cases}$$



Use a hybrid dynamical model of the process to predict its future evolution and choose the "best" control action

MIQP FORMULATION OF HYBRID MPC

(Bemporad, Morari, 1999)

- Finite-horizon optimal control problem (regulation)

$$\min \quad \sum_{k=0}^{N-1} y_k' Q y_k + u_k' R u_k$$

$$\text{s.t.} \quad \begin{cases} x_{k+1} = Ax_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\ y_k = Cx_k + D_1 u_k + D_2 \delta_k + D_3 z_k + D_5 \\ E_2 \delta_k + E_3 z_k \leq E_4 x_k + E_1 u_k + E_5 \\ x_0 = x(t) \end{cases}$$

$Q = Q' \succ 0, R = R' \succ 0$

- Treat u_k, δ_k, z_k as free decision variables, $k = 0, \ldots, N-1$

- Predictions can be constructed **exactly as in the linear case**

$$x_k = A^k x_0 + \sum_{j=0}^{k-1} A^j (B_1 u_{k-1-j} + B_2 \delta_{k-1-j} + B_3 z_{k-1-j} + B_5)$$

MIQP FORMULATION OF HYBRID MPC

(Bemporad, Morari, 1999)

- After substituting x_k, y_k the resulting optimization problem becomes the following **Mixed-Integer Quadratic Programming (MIQP)** problem

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

- The optimization vector $\xi = [u_0, \ldots, u_{N-1}, \delta_0, \ldots, \delta_{N-1}, z_0, \ldots, z_{N-1}]$ has **mixed real and binary** components

$$\begin{aligned} & u_k \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b} \\ & \delta_k \in \{0,1\}^{r_b} \\ & z_k \in \mathbb{R}^{r_c} \end{aligned} \quad \Rightarrow \quad \xi \in \mathbb{R}^{N(m_c+r_c)} \times \{0,1\}^{N(m_b+r_b)}$$

HYBRID MPC FOR REFERENCE TRACKING

- Consider the more general set-point tracking problem

$$\begin{aligned}\min_{\xi} \quad & \sum_{k=0}^{N-1} \|y_k - r\|_Q^2 + \|u_k - u_r\|_R^2 \\ & +\sigma \left(\|x_k - x_r\|_2^2 + \|\delta_k - \delta_r\|_2^2 + \|z_k - z_r\|_2^2 \right) \\ \text{s.t.} \quad & \text{MLD model equations} \\ & x_0 = x(t) \\ & x_N = x_r\end{aligned}$$

with $\sigma > 0$ and $\|v\|_Q^2 = v'Qv$

- The equilibrium $(x_r, u_r, \delta_r, z_r)$ corresponding to r can be obtained by solving the following mixed-integer feasibility problem

$$\begin{aligned} x_r &= Ax_r + B_1 u_r + B_2 \delta_r + B_3 z_r + B_5 \\ r &= Cx_r + D_1 u_r + D_2 \delta_r + D_3 z_r + D_5 \\ E_2 \delta_r &+ E_3 z_r \leq E_4 x_r + E_1 u_r + E_5 \end{aligned}$$

CLOSED-LOOP CONVERGENCE

(Bemporad, Morari, 1999)

- **Theorem.** Let $(x_r, u_r, \delta_r, z_r)$ be the equilibrium corresponding to r. Assume $x(0)$ such that the MIQP problem **is feasible at time** $t = 0$. Then $\forall Q, R \succ 0, \sigma > 0$ the hybrid MPC closed-loop **converges asymptotically**

$$\lim_{t\to\infty} y(t) = r \qquad \lim_{t\to\infty} x(t) = x_r$$
$$\lim_{t\to\infty} \delta(t) = \delta_r$$
$$\lim_{t\to\infty} u(t) = u_r \qquad \lim_{t\to\infty} z(t) = z_r$$

  and **all constraints are fulfilled** at each time $t \geq 0$.

- The proof easily follows from standard Lyapunov arguments (see next slide)

- **Lyapunov asymptotic stability** and **exponential stability** follows if proper terminal cost and constraints are imposed (Lazar, Heemels, Weiland, Bemporad, 2006)

MILP FORMULATION OF HYBRID MPC

(Bemporad, Borrelli, Morari, 2000)

- Finite-horizon optimal control problem using infinity norms

$$\min_{\xi} \quad \sum_{k=0}^{N-1} \|Qy_k\|_\infty + \|Ru_k\|_\infty$$

$$\text{s.t.} \quad \begin{cases} x_{k+1} = Ax_k + B_1u_k + B_2\delta_k + B_3z_k + B_5 \\ y_k = Cx_k + D_1u_k + D_2\delta_k + D_3z_k + D_5 \\ E_2\delta_k + E_3z_k \le E_4x_k + E_1u_k + E_5 \\ x_0 = x(t) \end{cases}$$

$$Q \in \mathbb{R}^{m_y \times n_y}$$
$$R \in \mathbb{R}^{m_u \times n_u}$$

- Introduce additional variables $\epsilon_k^y, \epsilon_k^u, k = 0, \ldots, N-1$

$$\begin{cases} \epsilon_k^y \ge \|Qy_k\|_\infty \\ \epsilon_k^u \ge \|Ru_k\|_\infty \end{cases} \quad \Rightarrow \quad \begin{cases} \epsilon_k^y \ge \pm Q^i y_k \\ \epsilon_k^u \ge \pm R^i u_k \end{cases}$$

Q^i = ith row of Q

MILP FORMULATION OF HYBRID MPC

(Bemporad, Borrelli, Morari, 2000)

- After substituting x_k, y_k the resulting optimization problem becomes the following **Mixed-Integer Linear Programming (MILP)** problem

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

- $\xi = [u_0, \ldots, u_{N-1}, \delta_0, \ldots, \delta_{N-1}, z_0, \ldots, z_{N-1}, \epsilon_0^y, \epsilon_0^u, \ldots, \epsilon_{N-1}^y, \epsilon_{N-1}^u]$ is the optimization vector, with **mixed real and binary** components

$$\begin{aligned} & u_k \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b} \\ & \delta_k \in \{0,1\}^{r_b} \\ & z_k \in \mathbb{R}^{r_c} \\ & \epsilon_k^y, \epsilon_k^u \in \mathbb{R} \end{aligned} \quad \Rightarrow \quad \xi \in \mathbb{R}^{N(m_c+r_c+2)} \times \{0,1\}^{N(m_b+r_b)}$$

- Same approach applies to any **convex piecewise affine** stage cost

HYBRID MPC – TEMPERATURE CONTROL

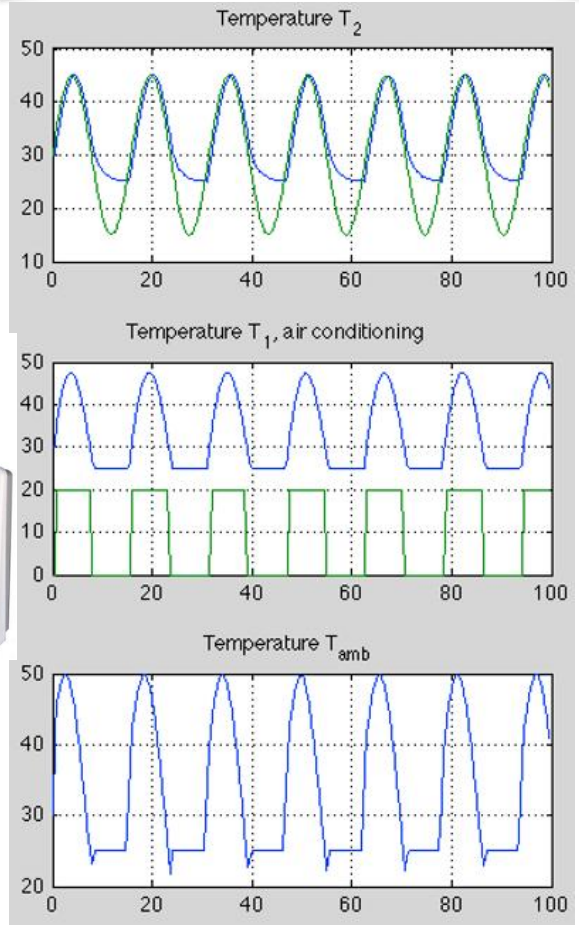
```
>> refs.x=2;          % just weight state #2
>> Q.x=1;          % unit weight on state #2
>> Q.rho=Inf;             % hard constraints
>> Q.norm=Inf;              % infinity norms
>> N=2;                  % prediction horizon
>> limits.xmin=[25;-Inf];
```

```
>> 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.
>>
```

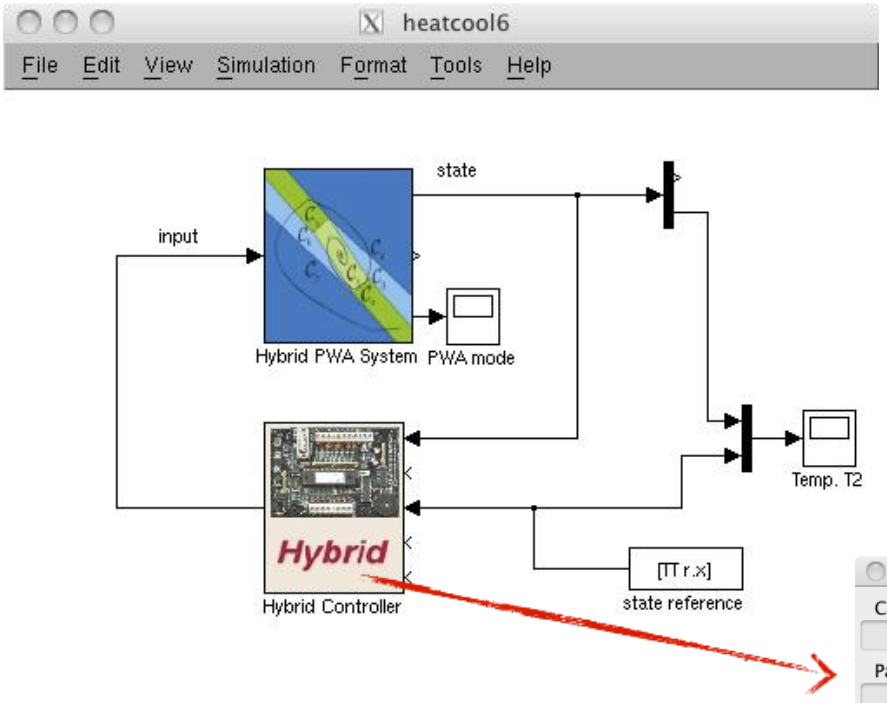
$$\min \quad \sum_{k=1}^{2} \|x_{2k} - r(t)\|_\infty$$

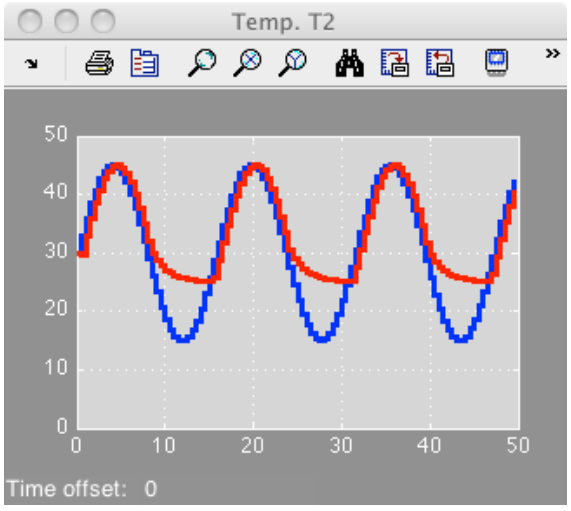
$$\text{s.t.} \quad \begin{cases} x_{1k} \geq 25, \; k = 1, 2 \\ \text{MLD model} \end{cases}$$

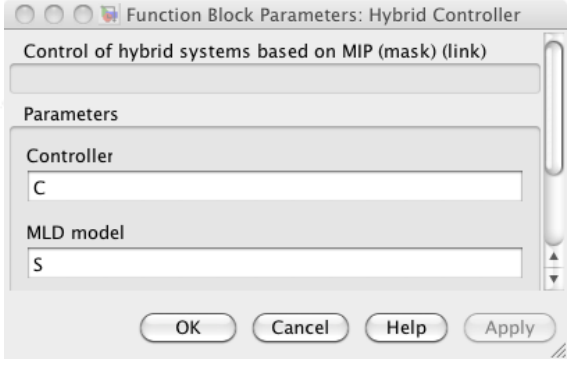



```
>> [XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,Tstop);
```

HYBRID MPC – TEMPERATURE CONTROL







- Average CPU time to solve MILP: $\approx$ 1 ms/step

  (Macbook Pro 3GHz Intel Core i7 using GLPK)

MIXED-INTEGER PROGRAMMING SOLVERS

- Binary constraints make Mixed-Integer Programming (MIP) a hard problem ($\mathcal{NP}$-complete)

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

  (more solvers/benchmarks: see `http://plato.la.asu.edu/bench.html`)

- MIQP approaches tailored to embedded hybrid MPC applications:
  - B&B + (dual) active set methods for QP

    (Leyffer, Fletcher, 1998) (Axehill, Hansson, 2006) (Bemporad, 2015) (Bemporad, Naik, 2018)
  - B&B + interior point methods: (Frick, Domahidi, Morari, 2015)
  - B&B + fast gradient projection: (Naik, Bemporad, 2017)
  - B&B + ADMM: (Stellato, Naik, Bemporad, Goulart, Boyd, 2018)

- No need to reach global optimum (see convergence proof), although performance may deteriorate

SOLVING MIQP VIA NNLS AND PROXIMAL-POINT ITERATIONS

(Bemporad, Naik, 2018)

- **Robustified approach**: use **NNLS + proximal-point iterations** to solve QP relaxations (Bemporad, 2018)

$$\begin{aligned} z_{k+1} = \arg\min_z \quad & \tfrac{1}{2} z'Qz + c'z + \tfrac{\epsilon}{2}\|z - z_k\|_2^2 \\ \text{s.t.} \quad & \ell \le Az \le u \\ & Gz = g \end{aligned}$$

- CPU time (ms) on **MIQP** coming from hybrid MPC (`bm99` demo):

For $N = 10$:
30 real vars
10 binary vars
160 inequalities

prox-NNLS* = warm start of binary vars exploited

N	prox-NNLS		prox-NNLS*		GUROBI		CPLEX	
	avg	max	avg	max	avg	max	avg	max
2	2.0	2.6	2.0	2.6	1.6	2.0	3.1	6.0
4	5.3	8.8	3.1	6.9	3.1	3.9	8.9	15.7
8	29.7	71.0	8.1	43.4	7.2	13.2	15.5	80.2
10	76.2	146.1	14.4	103.2	11.1	17.6	35.1	95.3
12	155.8	410.8	26.9	263.4	14.9	31.2	61.7	103.7
15	484.2	1242.3	61.7	766.9	25.9	109.8	89.9	181.1

CPU time measured on Intel Core i7-4700MQ CPU 2.40 GHz

FAST GRADIENT PROJECTION FOR MIQP

(Naik, Bemporad, 2017)

- Consider again the MIQP problem with Hessian $Q = Q' \succ 0$

$$\begin{aligned} \min_z \quad & V(z) \triangleq \frac{1}{2} z'Qz + c'z \\ \text{s.t.} \quad & \ell \le Az \le u \\ & Gz = g \\ & \bar{A}_i z \in \{\bar{\ell}_i, \bar{u}_i\},\ i = 1, \dots, p \end{aligned}$$

$$\begin{aligned} w^k &= y^k + \beta_k (y^k - y^{k-1}) \\ z^k &= -Kw^k - Jx \\ s^k &= \tfrac{1}{L} Gz^k - \tfrac{1}{L}(W + Sx) \\ y^{k+1} &= \max\{w^k + s^k, 0\} \end{aligned}$$

- Use B&B and **fast gradient projection** to solve dual of QP relaxation

constraint is relaxed	$\bar{A}_i z \le \bar{u}_i$	$\rightarrow$	$y_i^{k+1} = \max\{y_i^k + s_i^k, 0\}$	$(y_i \ge 0)$
constraint is fixed	$\bar{A}_i z = \bar{u}_i$	$\rightarrow$	$y_i^{k+1} = y_i^k + s_i^k$	$(y_i \lesseqgtr 0)$
constraint is ignored	$\bar{A}_i z = \bar{\ell}_i$	$\rightarrow$	$y_i^{k+1} = 0$	$(y_i = 0)$

FAST GRADIENT PROJECTION FOR MIQP

(Naik, Bemporad, 2017)

- **Same dual QP matrices** at each node, **preconditioning** computed only once
- **Warm-start** exploited, **dual cost** used to stop QP relaxations earlier
- Criterion based on Farkas lemma to detect **QP infeasibility**
- Numerical results (time in ms):

n	m	p	q	miqpGPAD	GUROBI
10	100	2	2	15.6	6.56
50	25	5	3	3.44	8.74
50	150	10	5	63.22	46.25
100	50	2	5	6.22	26.24
100	200	15	5	164.06	188.42
150	100	5	5	31.26	88.13
150	200	20	5	258.80	274.06
200	50	15	6	35.08	144.38

CPU time measured on Intel Core i7-4700MQ CPU 2.40 GHz

HEURISTIC ADMM METHOD FOR (SUBOPTIMAL) MIQP

(Takapoui, Moehle, Boyd, Bemporad, 2017)

- Consider again MIQP problem

$$\begin{aligned} \min \quad & \tfrac{1}{2}x'Qx + q'x \\ \text{s.t.} \quad & \ell \le Ax \le u \\ & A_i x \in \{\ell_i, u_i\},\ i \in I \end{aligned}$$

- ADMM iterations:

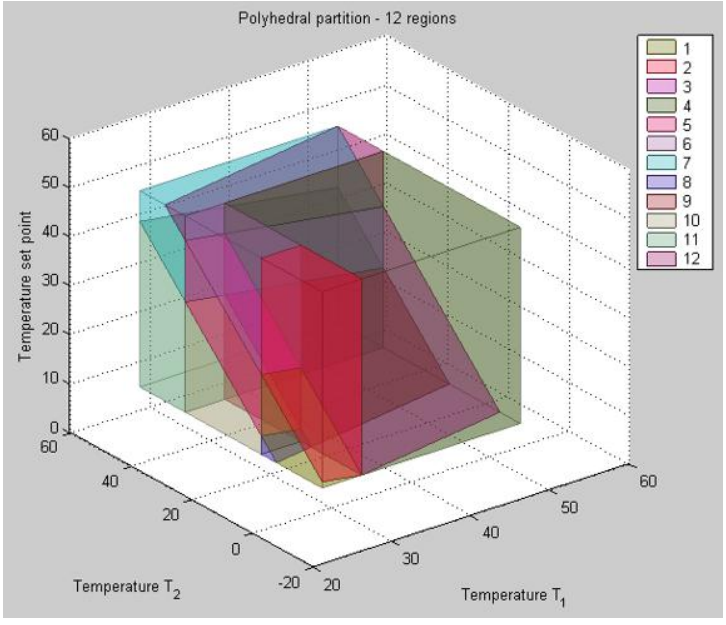
quantization step

$$\begin{aligned} x^{k+1} &= -(Q+\rho A^TA)^{-1}(\rho A^T(y^k - z^k) + q) \\ z^{k+1} &= \min\{\max\{Ax^{k+1} + y^k, \ell\}, u\} \\ z_i^{k+1} &= \begin{cases} \ell_i & \text{if} \quad z_i^{k+1} < \frac{\ell_i+u_i}{2} \\ u_i & \text{if} \quad z_i^{k+1} \ge \frac{\ell_i+u_i}{2} \end{cases},\ i \in I \\ y^{k+1} &= y^k + Ax^{k+1} - z^{k+1} \end{aligned}$$

- Iterations converge to a (local) solution
- Similar idea also applicable to fast gradient methods (Naik, Bemporad, 2017)

EXPLICIT HYBRID MPC

- It is possible to write hybrid MPC laws in explicit form too !
- The explicit MPC law is still piecewise affine on polyhedra



(Bemporad, Borrelli, Morari, 2000)

(Mayne, ECC 2001)

(Mayne, Rakovic, 2002)

(Bemporad, Hybrid Toolbox, 2003)

(Borrelli, Baotic, Bemporad, Morari, 2005)

(Alessio, Bemporad, 2006)

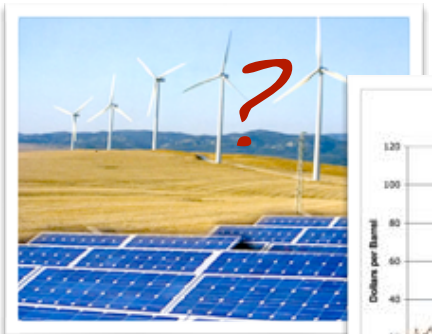
- The control law may be discontinuous, polyhedra may overlap
- Comparison of quadratic costs can be avoided by lifting the parameter space

(Fuchs, Axehill, Morari, 2015)

STOCHASTIC MODEL PREDICTIVE CONTROL

OPTIMIZE DECISIONS UNDER UNCERTAINTY

- In many control problems decisions must be taken under **uncertainty**



renewable power

prices

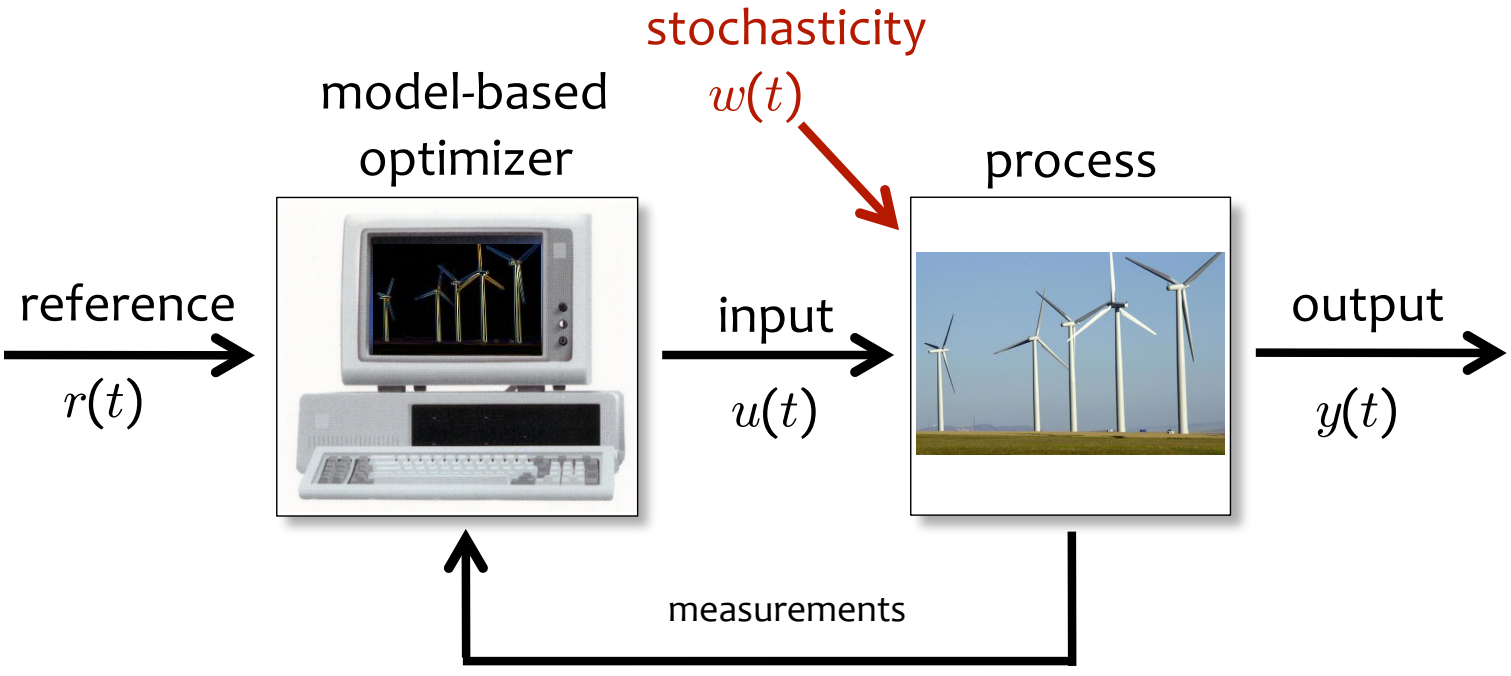
water

demand

human (inter)action

- **Robust** control approaches do not model uncertainty (only assume that is bounded) and pessimistically consider the worst case
- **Stochastic models** provide instead additional information about uncertainty
- **Optimality** often sought (ex: minimize expected economic cost)

STOCHASTIC MODEL PREDICTIVE CONTROL (SMPC)

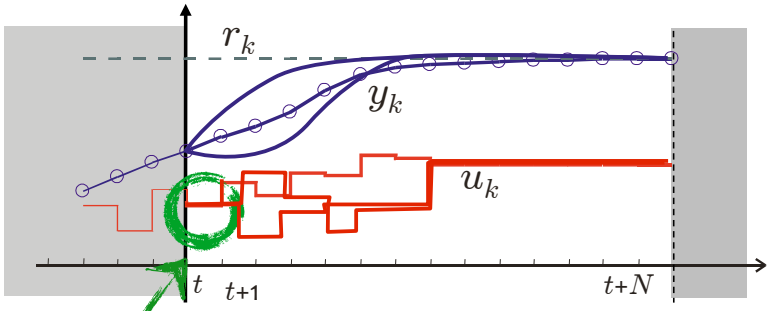


*Use a **stochastic** dynamical **model** of the process to **predict** its possible future evolutions and choose the "best" **control** action*

STOCHASTIC MODEL PREDICTIVE CONTROL

- At time t: solve a **stochastic optimal control** problem over a finite future horizon of N steps:

$$\begin{aligned}
\min_{u} \quad & E_w\left[\sum_{k=0}^{N-1} \ell(y_k, u_k, w_k)\right] \\
\text{s.t.} \quad & x_{k+1} = A(w_k)x_k + B(w_k)u_k + f(w_k) \\
& y_k = C(w_k)x_k + D(w_k)u_k + g(w_k) \\
& u_{\min} \le u_k \le u_{\max} \\
& y_{\min} \le y_k \le y_{\max}, \forall w \quad \text{robustness} \\
& x_0 = x(t) \quad \text{feedback}
\end{aligned}$$



$x(t)$ = process state

$u(t)$ = manipulated vars

$y(t)$ = controlled output

$w(t)$ = **stochastic disturbances**

- Solve stochastic optimal control problem w.r.t. future input sequence
- Apply the first optimal move $u(t)=u_0^*$, throw the rest of the sequence away
- At time $t+1$: Get new measurements, repeat the optimization. And so on …

LINEAR STOCHASTIC MODEL W/ DISCRETE DISTURBANCE

- **Linear stochastic** prediction model

$$\begin{cases} x_{k+1} &= A(w_k)x_k + B(w_k)u_k + f(w_k) \\ y_k &= C(w_k)x_k + g(w_k) \end{cases}$$

- **Discrete disturbance**

$$w_k \in \{w^1, \dots, w^s\}$$

with discrete probabilities $p_j = \Pr\left[w_k = w^j\right], p_j \geq 0, \sum_{j=1}^{s} p_j = 1$

- (A, B, C) can be sparse matrices (ex: network of interacting subsystems), while often w_k is low-dimensional (ex: electricity price, weather, etc.)

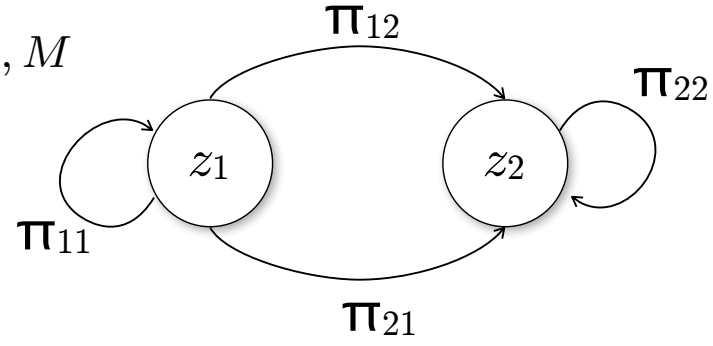
LINEAR STOCHASTIC MODEL W/ DISCRETE DISTURBANCE

- Probabilities p_j can be time varying, $p_j(t)$, and have their own dynamics

  Example: Markov chain

$$\pi_{ih} = \mathbf{Pr}[z(t+1) = z_h \mid z(t) = z_i], \;\; i,h = 1,\dots,M$$

$$p_j(t) = \begin{cases} e_{1j} & \text{if} \;\; z(t) = z_1 \\ \vdots & \vdots \\ e_{Mj} & \text{if} \;\; z(t) = z_M \end{cases}$$

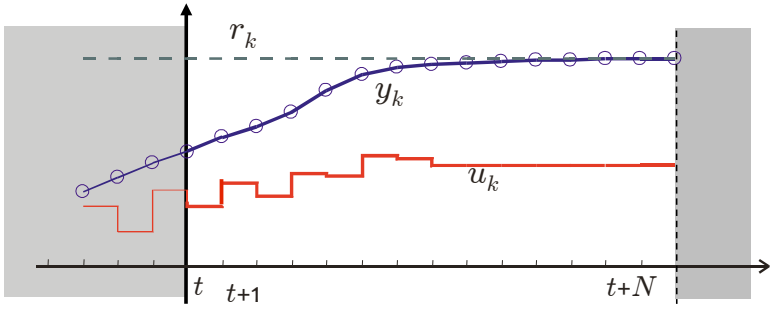


- Discrete distributions can be **estimated from historical data** (and adapted on-line)

COST FUNCTIONS FOR SMPC TO MINIMIZE

- **Expected performance**

$$\min_u \sum_{k=0}^{N-1} E_w\left[(y_k - r_k)^2\right]$$



- Tradeoff between **expectation & risk**

$$\min_u \sum_{k=0}^{N-1} (E_w\left[y_k - r_k\right])^2 + \alpha \mathrm{Var}_w\left[y_k - r_k\right] \qquad \alpha \geq 0$$

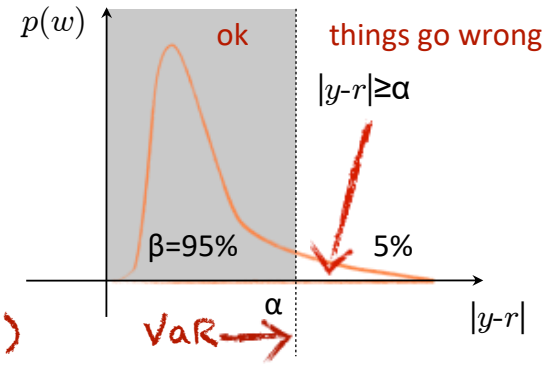
- Note that they coincide for α=1, since

$$\mathrm{Var}_w\left[y_k - r_k\right] = E_w\left[(y_k - r_k)^2\right] - (E_w\left[y_k - r_k\right])^2$$

COST FUNCTIONS FOR SMPC TO MINIMIZE

- Conditional Value-at-Risk (**CVaR**) (Rockafellar, Uryasev, 2000)

$$\min_{u,\alpha} \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} E_w[\max\{|y_k - r_k| - \alpha_k, 0\}]$$

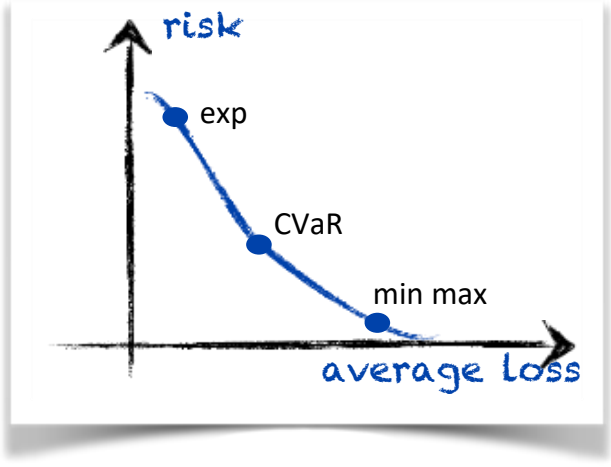


= minimize expected loss when things go wrong (convex !)

= expected shortfall

- **Min-max** = minimize worst case performance

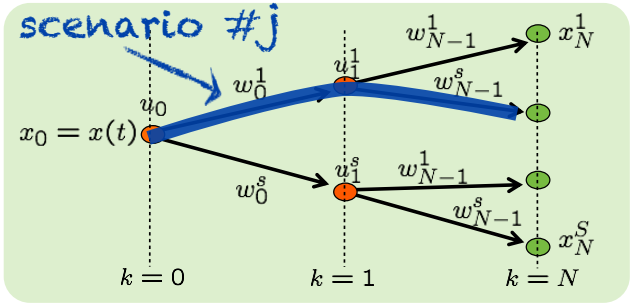
$$\min_{u} \sum_{k=0}^{N-1} \max_{w} |y_k - r_k|$$



STOCHASTIC OPTIMAL CONTROL PROBLEM

- Enumerate all possible scenarios $\{w_0^j, w_1^j, \ldots, w_{N-1}^j\}, j = 1, \ldots, S$

- Scenario = path on the tree



- Number S of scenarios = number of leaf nodes

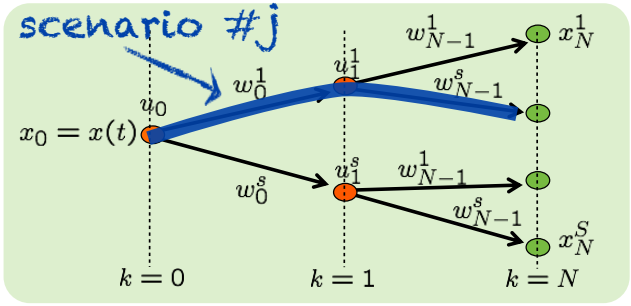
- Each scenario has probability $p_j = \prod_{k=0}^{N-1} \mathbf{Pr}[w_k = w^j]$

STOCHASTIC OPTIMAL CONTROL PROBLEM

- Each scenario has its own evolution

$$x_{k+1}^j = A(w_k^j)x_k^j + B(w_k^j)u_k^j + f(w_k^j)$$

(=linear time-varying system)



- Expectations become simple sums!

Example:

$$\min E_w \left[x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k \right]$$

$$\min \sum_{j=1}^{S} p^j \left((x_N^j)' P x_N^j + \sum_{k=0}^{N-1} (x_k^j)' Q x_k^j + (u_k^j)' R u_k^j \right)$$

Expectations of quadratic costs remain quadratic costs

STOCHASTIC OPTIMAL CONTROL PROBLEM

- CVaR optimization (Rockafellar, Uryasev, 2000)

$$\min_{u,\alpha} \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} E_w \left[\max\{|y_k - r_k| - \alpha_k, 0\}\right]$$

$$\begin{aligned} \min_{u,z,\alpha} \quad & \sum_{k=0}^{N-1} \alpha_k + \frac{1}{1-\beta} \sum_{j=1}^{S} p^j z_k^j \\ \text{s.t.} \quad & z_k^j \geq y_k^j - r_k^j - \alpha_k \\ & z_k^j \geq r_k^j - y_k^j - \alpha_k \\ & z_k^j \geq 0 \end{aligned}$$

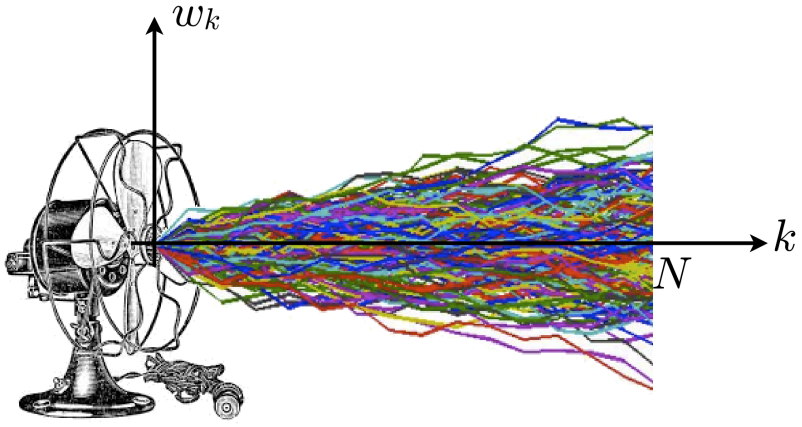
CVaR optimization becomes a linear programming problem

SCENARIO TREE GENERATION FROM DATA

- Scenario trees can be generated by **clustering** sample paths

- Paths can be obtained by **Monte Carlo simulation** of (estimated) models, or from **historical data**

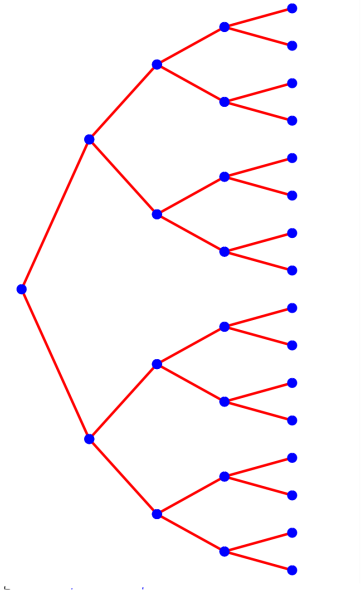
- The **number of nodes** can be decided a priori



scenario "fan" (collection of sample paths)

Heuristic Multilevel Clustering

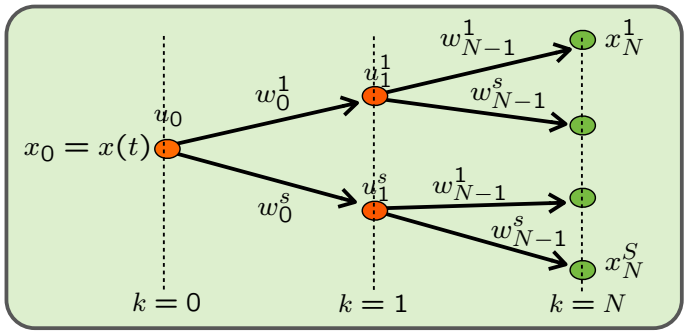
(Heitsch, Römisch, 2009)



scenario tree

- Alternative (simpler/less accurate) approach: **k-means** clustering

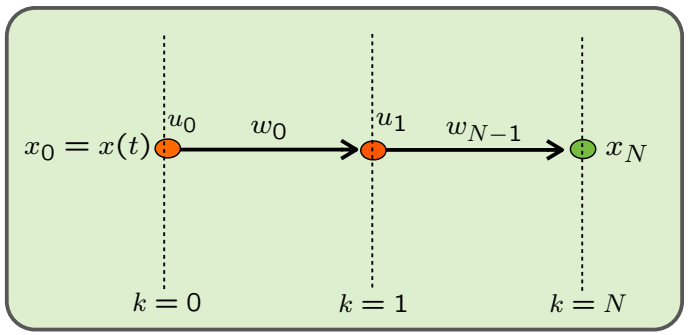
FREE CONTROL VARIABLES



Stochastic optimal control

Causality constraint: $u_k^j = u_k^h$ when scenarios j and h share the same node at prediction time k (for example: $u_0^j \equiv u_0^h$ at root node k=0)

Decision u_k only depends on past disturbance realizations $\{w_0, w_1, \ldots, w_{k-1}\}$



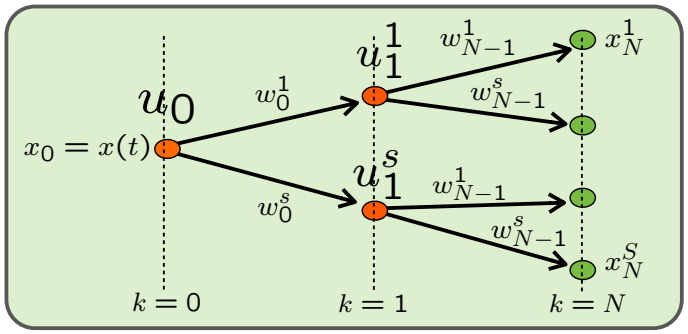
Deterministic control

Only a sequence of disturbances is considered

- frozen-time: $w_k \equiv w(t),\ \forall k$ (causal prediction)
- prescient control: $w_k \equiv w(t+k)$ (non-causal)
- certainty equivalence: $w_k = E[w(t+k)|t]$ (causal)

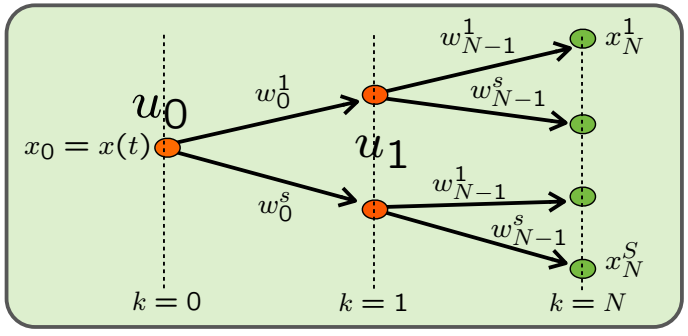
We can trade off between **complexity** of optimization problem (=number of nodes) and **performance** (=accuracy of stochastic modeling)

OPEN-LOOP VS CLOSED-LOOP PREDICTION



closed-loop prediction

A proper move u is optimized to counteract each possible outcome of the disturbance w



open-loop prediction

Only a sequence of inputs $\{u_0, u_1, \ldots, u_{N-1}\}$ is optimized, the same u must be good for all possible disturbance w

- Intuitively: OL prediction is more conservative than CL in handling constraints
- OL problem = CL problem + additional constraints $u^j \equiv u, \ \forall j = 1, \ldots, S$ (=less degrees of freedom)

LINEAR STOCHASTIC MPC FORMULATION

- A rich literature on stochastic MPC is available

  (Schwarme, Nikolaou, 1999) (Munoz de la Pena, Bemporad, Alamo, 2005) (Primbs, 2007)

  (Oldewurtel, Jones, Morari, 2008) (Wendt, Wozny, 2000) (Couchman, Cannon, Kouvaritakis, 2006)

  (Ono, Williams, 2008) (Batina, Stoorvogel, Weiland, 2002) (van Hessem, Bosgra 2002)

  (Bemporad, Di Cairano, 2005) (Bernardini, Bemporad, 2012)

  See also recent survey (Mesbah, 2016)

- Performance index: $\min E_w \left[x'_N P x_N + \sum_{k=0}^{N-1} x'_k Q x_k + u'_k R u_k \right]$

- **Goal**: ensure **mean-square convergence** $\lim_{t\to\infty} E[x'(t)x(t)] = 0 \quad (f(w(t)) = 0)$

- Mean-square stability ensured by **stochastic Lyapunov function** $V(x) = x'Px$

$$E_{w(t)}\left[V(x(t+1))\right] - V(x(t) \leq -x'(t)Lx(t),\ \forall t \geq 0$$

$$P = P' \succ 0$$
$$L = L' \succ 0$$

STABILIZING STOCHASTIC MPC

(Bernardini, Bemporad, 2012)

- Impose stochastic stability constraint in SMPC problem (=quadratic constraint w.r.t. u_0)

$$\begin{aligned}\min_{u} \quad & E_w\left[\sum_{k=0}^{N-1} \ell(x_k,u_k)\right] \\ \text{s.t.} \quad & x_{k+1} = A(w_k)x_k + B(w_k)u_k \\ & E\left[V(A(w_0)x_0 + B(w_0)u_0)\right] \le x_0'(Q^{-1} - L)x_0 \\ & x_0 = x(t)\end{aligned}$$

performance and stability are decoupled

- SMPC approach:
  1. Solve LMI problem off-line to find stochastic Lyapunov fcn $V(x) = x'Q^{-1}x$
  2. Optimize stochastic performance based on scenario tree

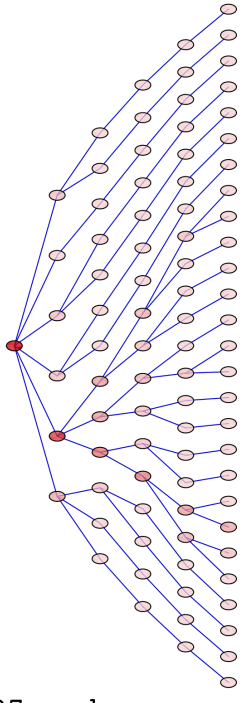
Theorem: The closed-loop system is as. stable in the mean-square sense

- SMPC can be generalized to handle **input and state constraints**

Note: *recursive feasibility* guaranteed by backup solution $u(k) = Kx(k)$

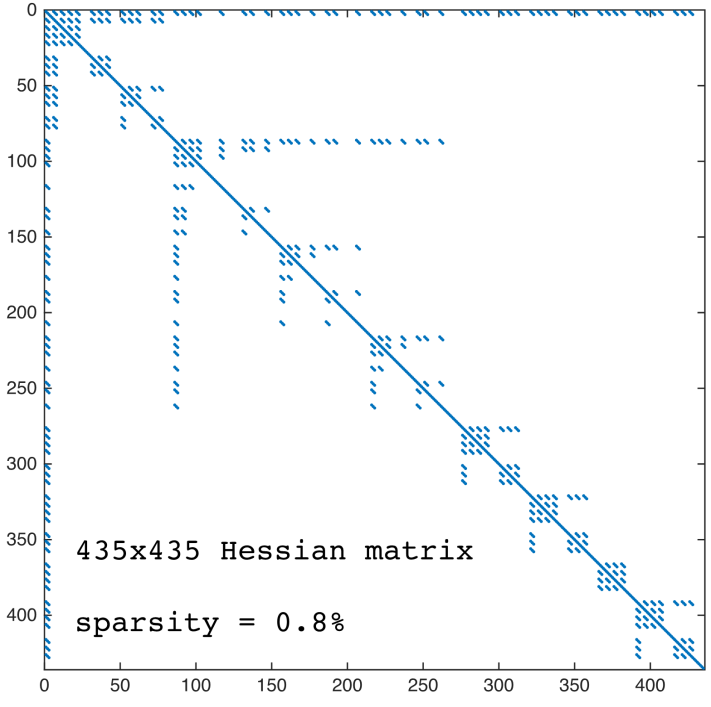
COMPLEXITY OF STOCHASTIC OPTIMIZATION PROBLEM

- #optimization variables = #nodes x #inputs (in condensed version)
- Problems are **very sparse** (well exploited by **interior point methods**)
- Example: SMPC with quadratic cost and linear constraints

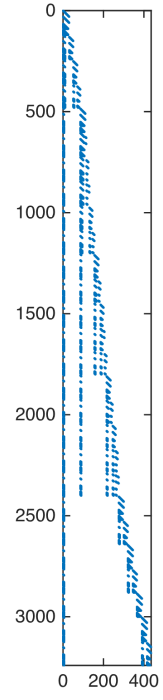


Tree=87 nodes

Branching factor M=[6 3 2 2 2]



435 free variables (5 inputs x node)



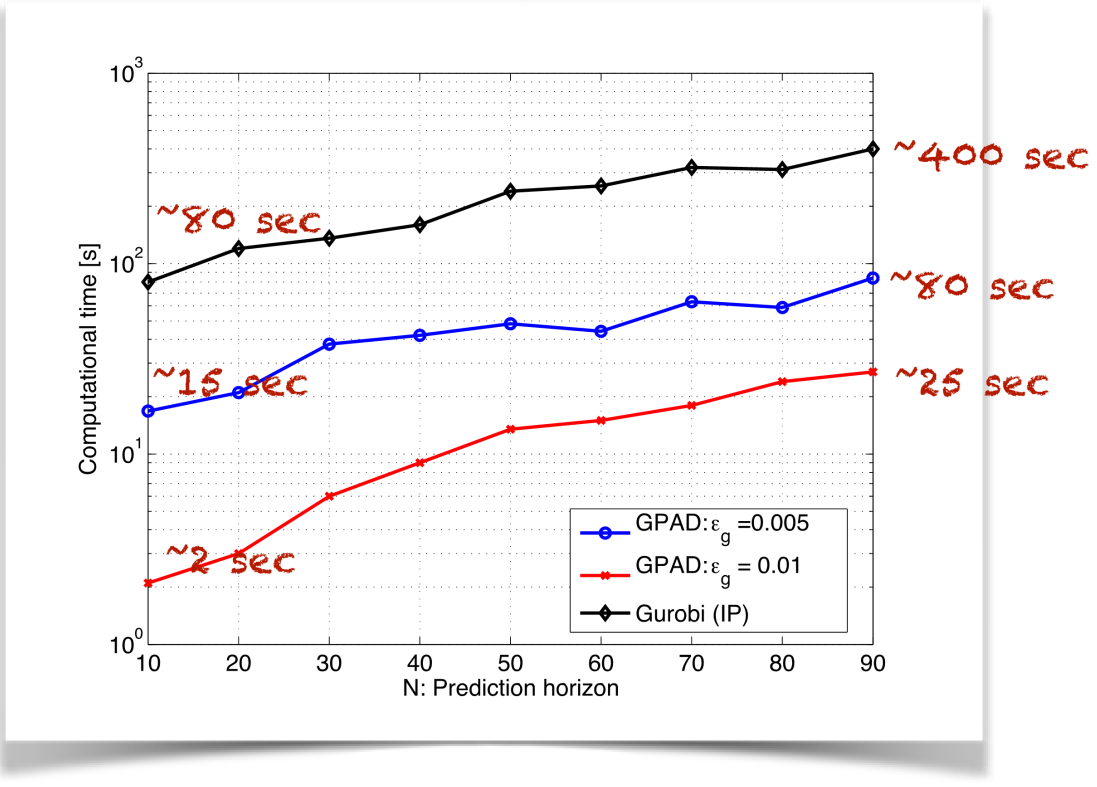
3240x435 constraint matrix
sparsity = 1.1%

DISTRIBUTED GPAD FOR STOCHASTIC MPC

- A distributed (parallelized) variant of the **A**ccelerated **G**radient **P**rojection applied to **D**ual (GPAD) for solving SMPC problems is available

(Sampathirao, Sopasakis, Bemporad, 2014)

Example: stochastic MPC with **60 states, 25 inputs, 256 scenarios**

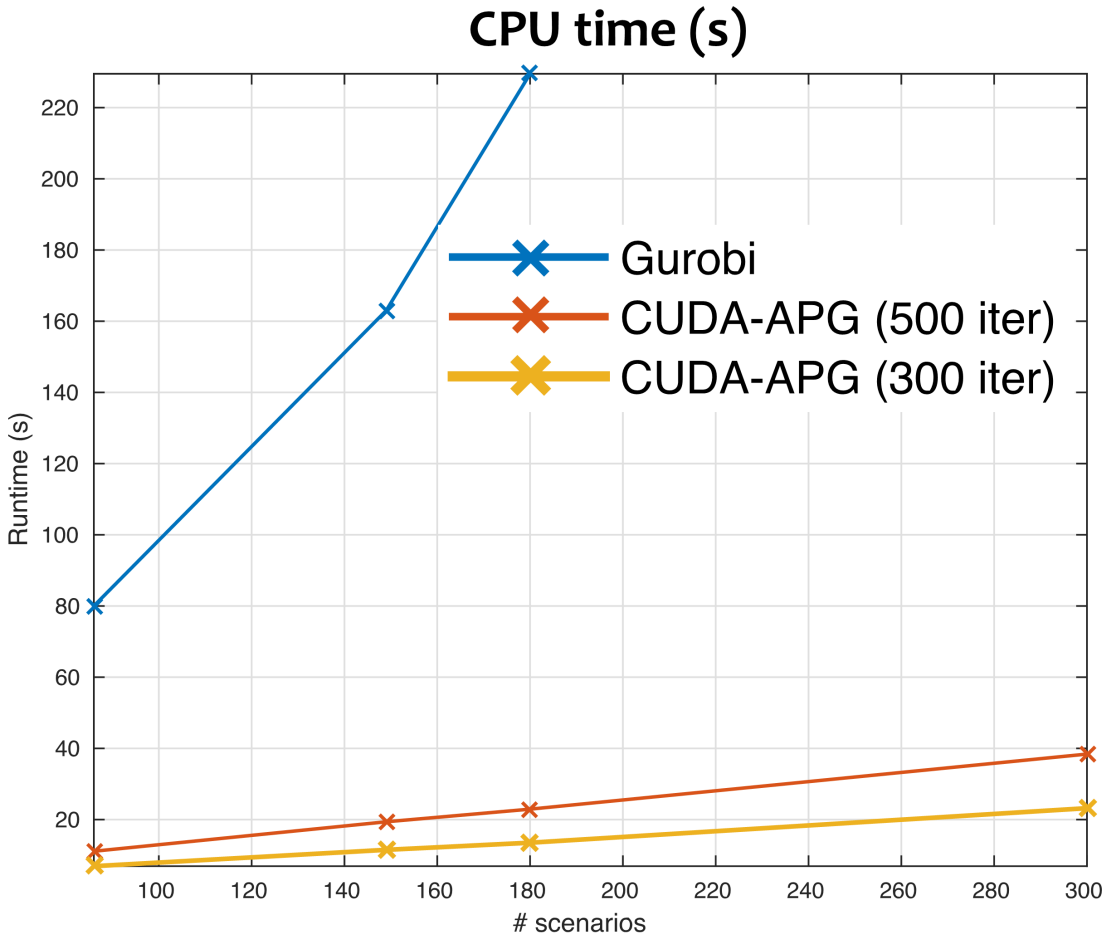


16x ÷ 40x faster than commercial state-of-the-art Interior-Point method

Remark: For larger problems (*e.g.*, *50* states, *30* inputs, *9036* nodes) GUROBI gets stuck on a 4GB 4-core PC, while dGPAD can solve the problem

DISTRIBUTED GPAD FOR STOCHASTIC MPC

(Sampathirao, Sopasakis, Bemporad, 2015)



APG = Accelerated Proximal Gradient, parallel implemented on NVIDIA Tesla 2075 CUDA platform

SMPC FOR REAL-TIME MARKET-BASED POWER DISPATCH

(Patrinos, Trimboli, Bemporad, 2011)

- We are a legal entity (BRP) trading on the energy (PX) and ancillary service (AS) markets

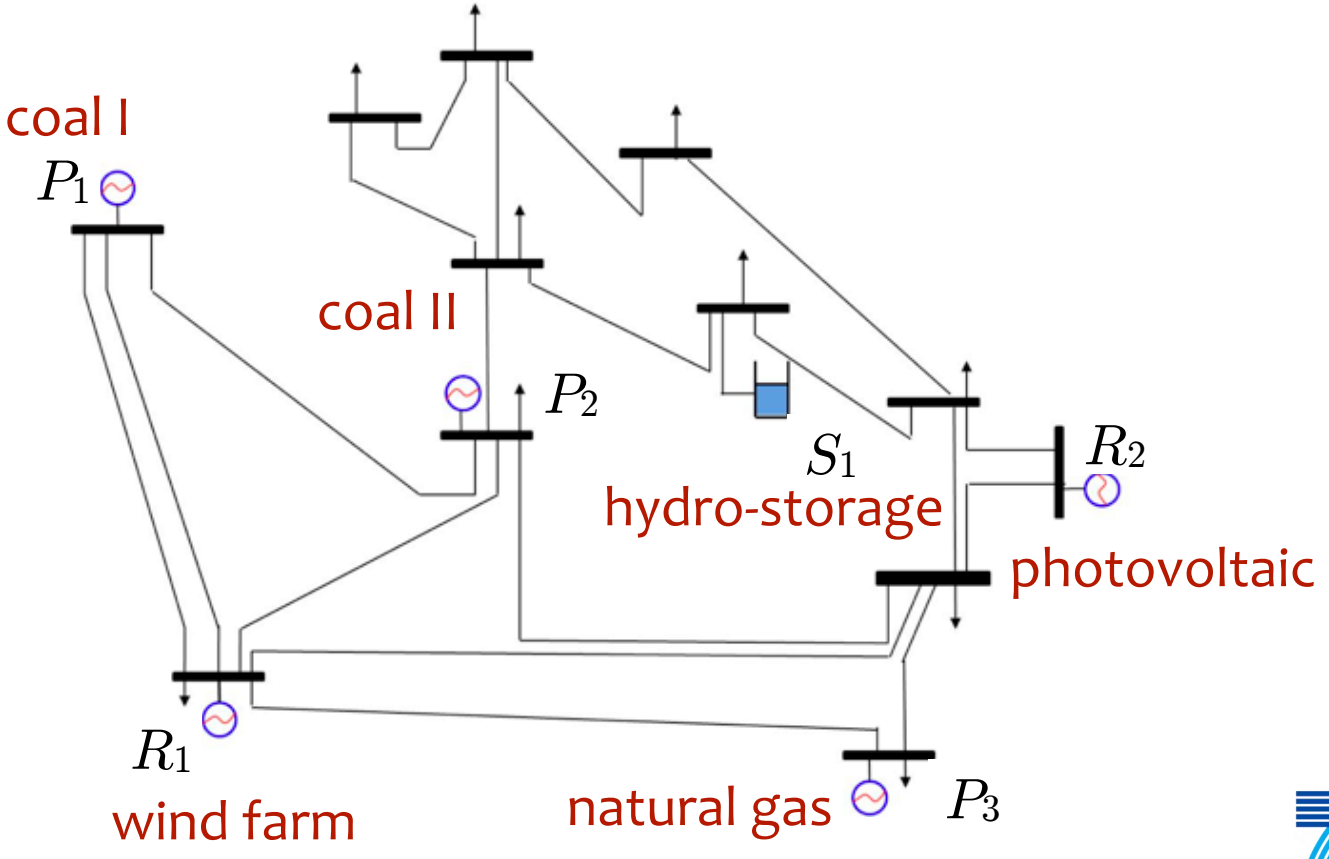
- **Objective:** Minimize costs via efficient use of intermittent resources, and maximize profits by trading on electricity (PX, AS) markets

- **Constraints**: Grid capacity, rate limits, load balancing, AS balancing

POWER DISPATCH MODEL

(Patrinos, Trimboli, Bemporad, 2011)

- Microgrid with three conventional power generators (P1,P2,P3), two renewables (R1,R2), one storage system (S1), satisfying local load and exchanging power with the grid



http://www.e-price-project.eu/

POWER DISPATCH MODEL

- Conventional generator model (i=1,2,3)

power generated at next time unit

$$P_{i,k+1} = P_{i,k} + \Delta P_{i,k}$$

constraints on generated power: $P_{i,\min} \le P_{i,k} \le P_{i,\max}$

constraints on power variation: $\Delta P_{i,\min} \le \Delta P_{i,k} \le \Delta P_{i,\max}$

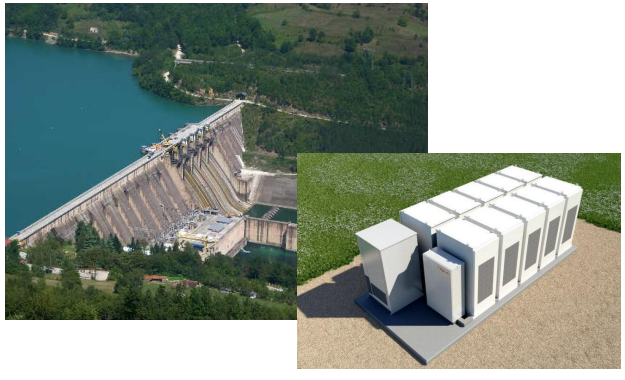
- Storage model

$$S_{k+1} = \alpha S_k + \alpha_c u_{c,k} - \frac{1}{\alpha_d} u_{d,k}$$

α = self discharge loss

α_c = charge efficiency

α_d = discharge efficiency



constraints on charge/discharge: $S_{\min} \le S_k \le S_{\max}$

constraints on charge/discharge rate: $\Delta_{S,\min} \le S_{k+1} - S_k \le \Delta_{S,\max}$

constraints on power flows: $0 \le u_{c,k} \le u_{c,\max}, \;\; 0 \le u_{d,k} \le u_{d,\max}$

POWER DISPATCH MODEL

- Power exchanged with the rest of the grid (=balance)

$$P_{\text{ex},k} = P_{1,k} + P_{2,k} + P_{3,k} - u_{c,k} + u_{d,k} + (r_k - l_k)$$

r-l is STOCHASTIC !

conventional power

storage charge/ discharge

renewable power

load

- Overall linear model and constraints

$$x = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ S \end{bmatrix} \qquad u = \begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \Delta P_3 \\ u_c \\ u_d \\ r - l \end{bmatrix} \qquad y = \begin{bmatrix} P_{\text{ex}} \\ P_1 \\ P_2 \\ P_3 \\ S \\ \Delta S \end{bmatrix}$$

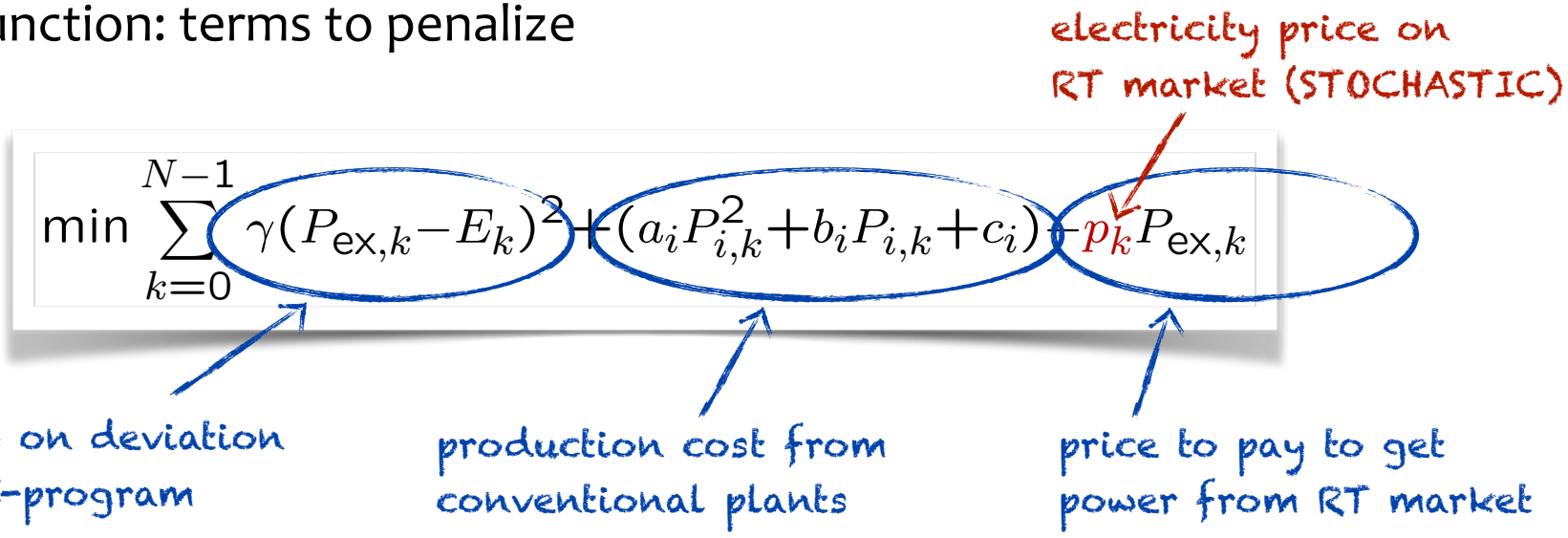
uncontrolled input

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \alpha \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_c & -\frac{1}{\alpha_d} & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \alpha - 1 \end{bmatrix} \qquad D = \begin{bmatrix} 0 & 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_c & -\frac{1}{\alpha_d} & 0 \end{bmatrix}$$

POWER DISPATCH COST FUNCTION

- Cost function: terms to penalize



$E_k = 0$, $\gamma = 0$ if no E-Program is agreed on the day-ahead market

- The overall linear MPC problem maps into a QP:

$$\min_z \quad \frac{1}{2}z'Hz + \left(F\begin{bmatrix} x_0 \\ r-l \\ E \end{bmatrix} + c\right)z + d$$

$$\text{s.t.} \quad Gz \le W + S\begin{bmatrix} x_0 \\ r-l \\ E \end{bmatrix}$$

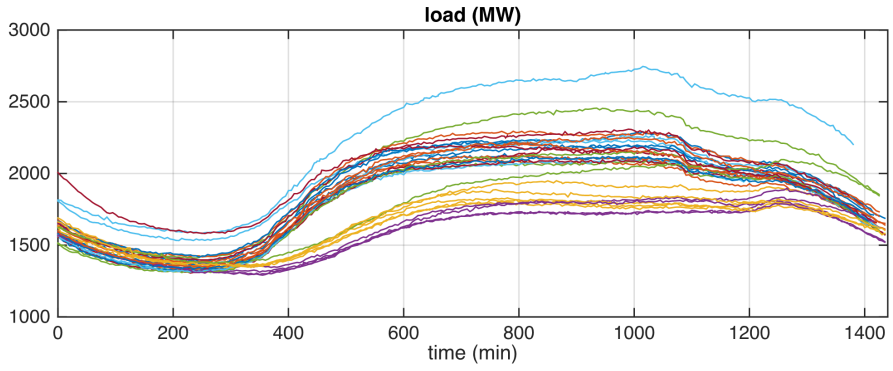
x_0 = current state
$r-l$ = predicted renewable power - load
E = E-program

$$z = \left\{\Delta P_{i,k}, u_{c,k}, u_{d,k}\right\}_{k=0}^{N-1}$$

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of load (MW)

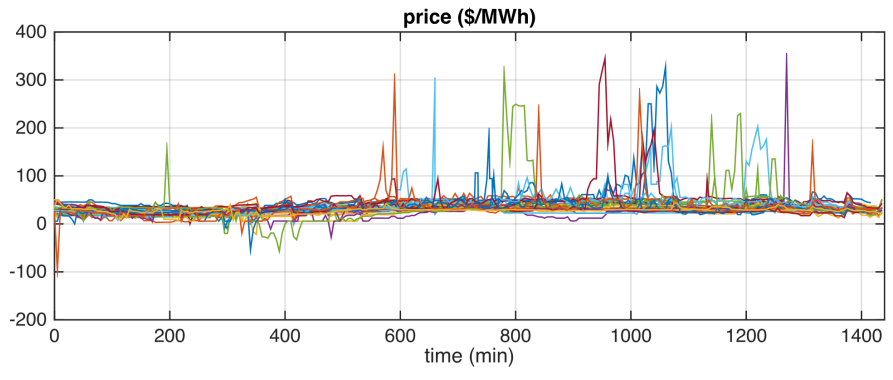
load = 1/3 load of N.Y.C. district
(daily data of 1-31 May 2014,
sampling time = 5 min)



http://www.nyiso.com/public/markets_operations/market_data/load_data/index.jsp

- Historical data of price (MW)

electricity price of N.Y.C. district
(daily data of 1-31 May 2014,
sampling time = 5 min)

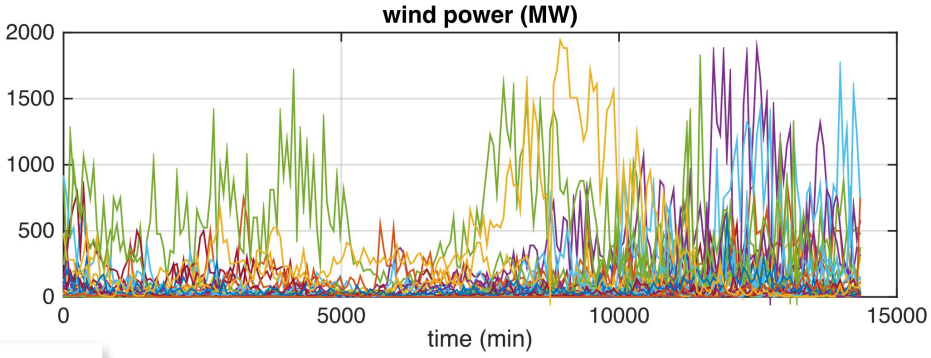


http://www.nyiso.com/public/markets_operations/market_data/pricing_data/index.jsp

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of wind speed (m/s)

Station BGNN4 (NY)
(daily data of 1-31 May 2014,
sampling time = 6 min)



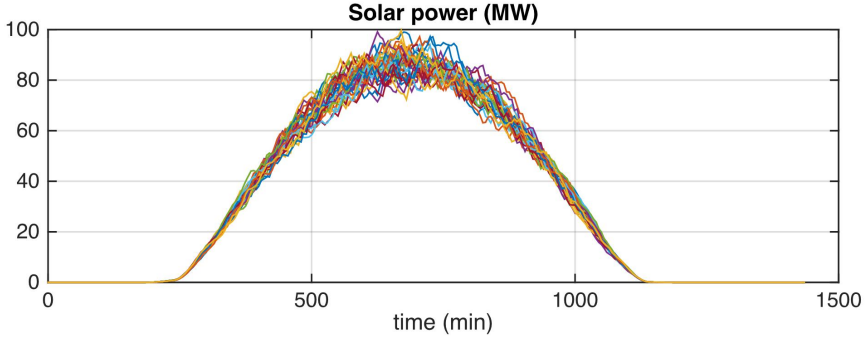
wind power proportional to cubic wind velocity → $P_w = k v_w^3$

http://www.ndbc.noaa.gov/station_history.php?station=bgnn4

- Historical data of solar irradiation (W/m²)

NY Central Park, daily data of
1-31 May 1991-2005, sampling time = 1 h

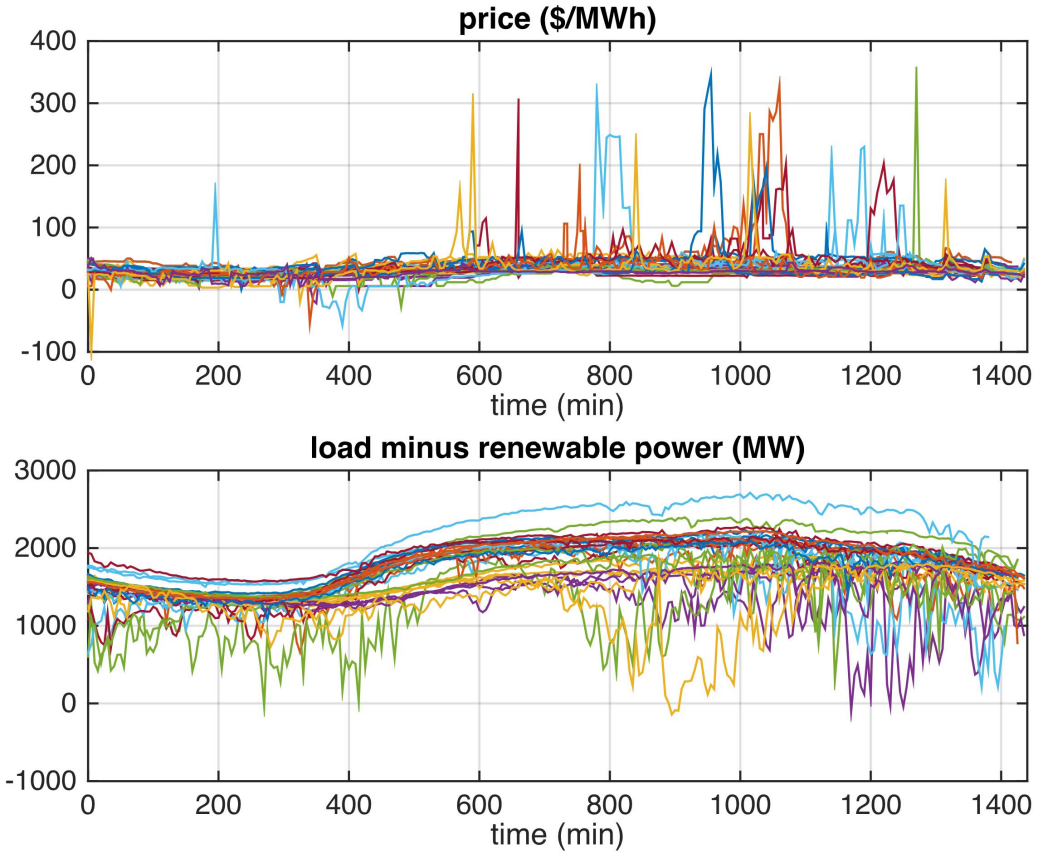
Data perturbed by noise to mimic
account cloud coefficient (unavailable)



http://en.openei.org/datasets/files/39/pub/725033.tar.gz

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Historical data of overall uncertainty



SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Data used for scenario generation (31 days):

value at time t+k in scenario #j

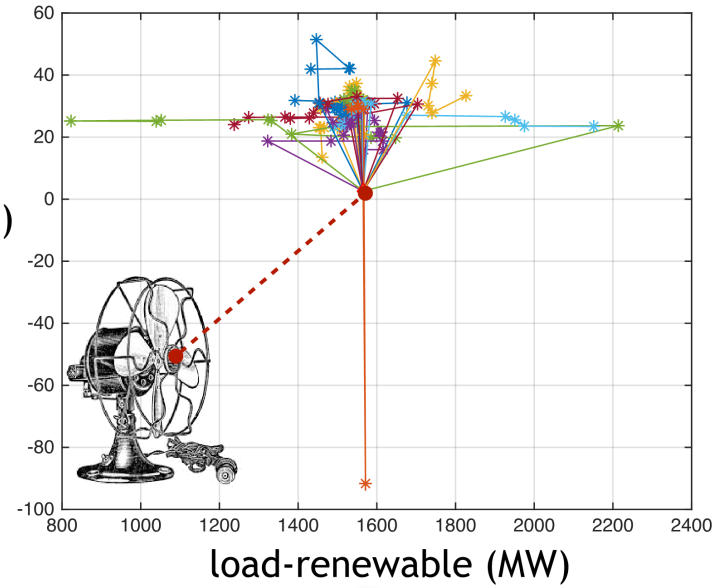
initial value at time t in scenario #j

$$w^j(t+k) = v^j(t+k) - v^j(t) + v(t)$$

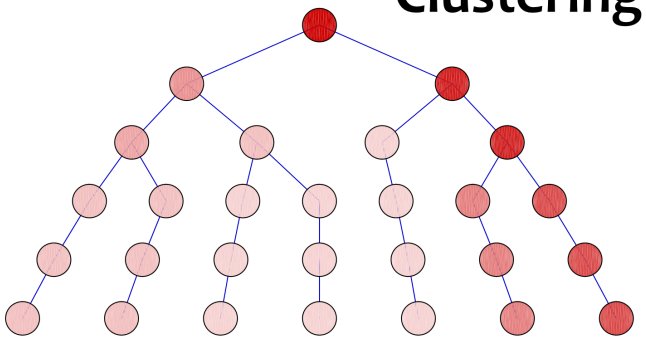
$$w^j(t) = v(t)$$

stochastic vector on scenario #j

actual value at time t



Heuristic Multilevel Clustering



28 nodes

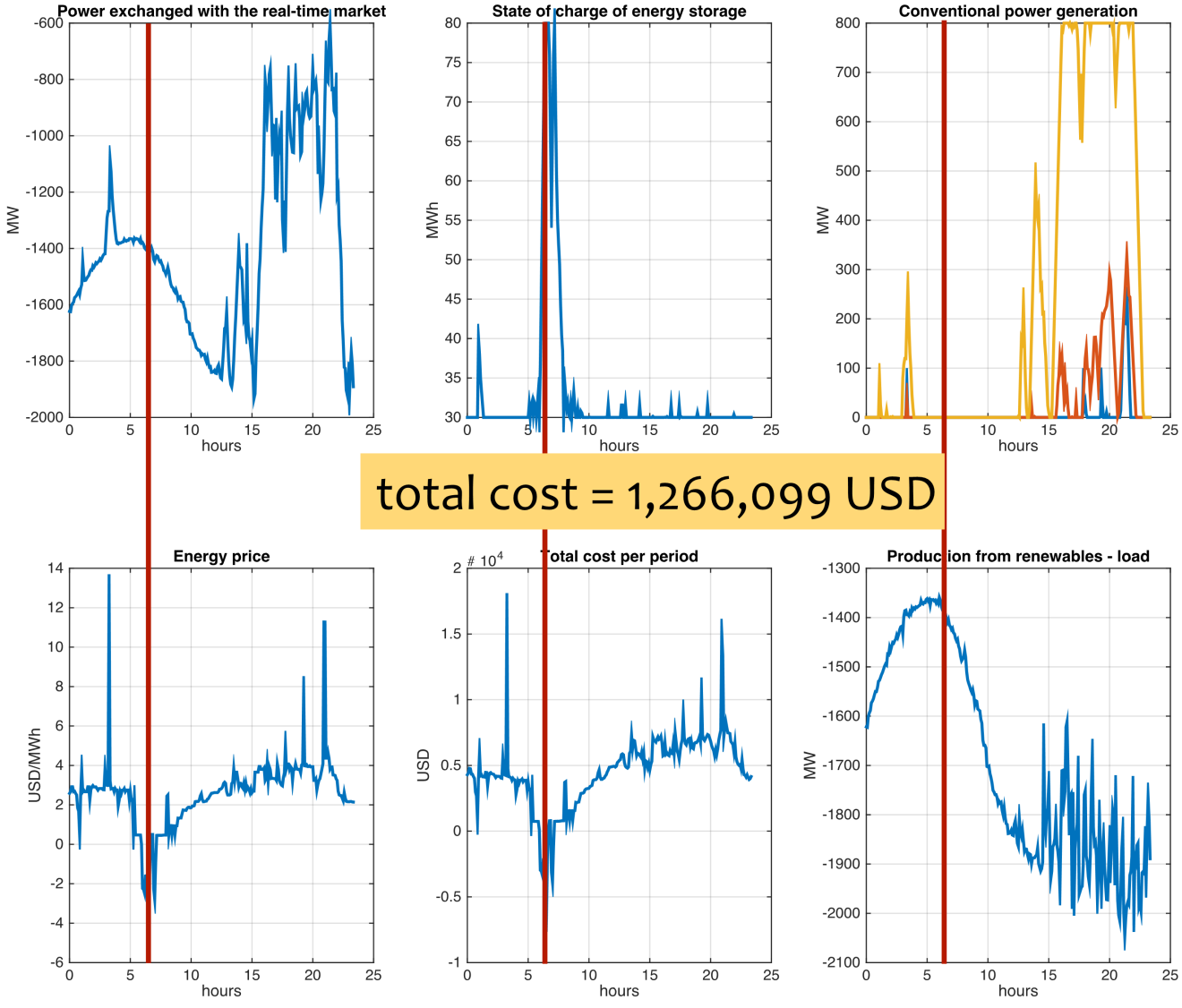
- Tree obtained from 31 scenarios (branching factor M=[2 2 2 1 1])

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- MPC setup:
  - Sampling time: T_s=5 min
  - Prediction horizon: N=6 steps (=1/2 hour ahead)
  - Three controller options:
    - **Stochastic MPC**, with branching factor M (ex: M=[4 3 2 2 1])
    - **Average MPC**, that is deterministic MPC based on the **expected** (price, load-renewable) realization
    - **Prescient MPC**, that is deterministic MPC based on the **exact** future (price, load-renewable) realization

SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Simulation results using SMPC, M=[2,2,2,1,1] (1 day, May 26, 2014)



SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Compare simulation results wrt different tree complexity, prescient, and deterministic (1 day, May 26, 2014)

exact knowledge of future uncertainty

stochastic formulation

```
Prescient:    Total cost=   1,247,909 [USD],                          nvar=  30, CPUTIME =  14 [ms]
Stochastic:   Total cost=   1,266,099 [USD], M=[2,2,2,1,1], nvar= 105, CPUTIME =  43 [ms]
Stochastic:   Total cost=   1,266,123 [USD], M=[3,3,1,1,1], nvar= 140, CPUTIME =  50 [ms]
Stochastic:   Total cost=   1,266,214 [USD], M=[2,2,1,1,1], nvar=  95, CPUTIME =  30 [ms]
Stochastic:   Total cost=   1,266,701 [USD], M=[3,1,1,1,1], nvar=  80, CPUTIME =  27 [ms]
Stochastic:   Total cost=   1,267,069 [USD], M=[2,1,1,1,1], nvar=  55, CPUTIME =  22 [ms]
Average:      Total cost=   1,267,113 [USD], M=[1,1,1,1,1]  nvar=  30, CPUTIME =  14 [ms]
Frozen-time:  Total cost=   1,267,401 [USD],                          nvar=  30, CPUTIME =  14 [ms]
```

deterministic: assume future disturbance = average of historical data

deterministic: assume future disturbance = current disturbance

nvar = number of variables in QP problem = 5*(# nodes), **CPUTIME** = time to build tree,build QP matrices, and solve

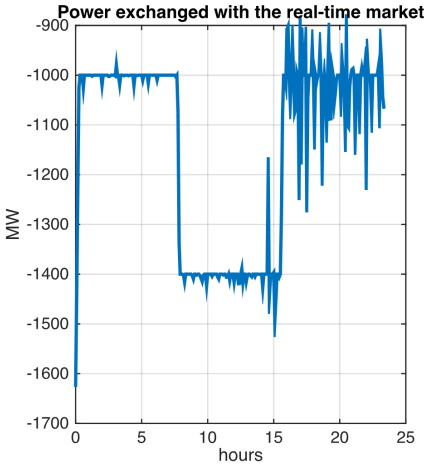
SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

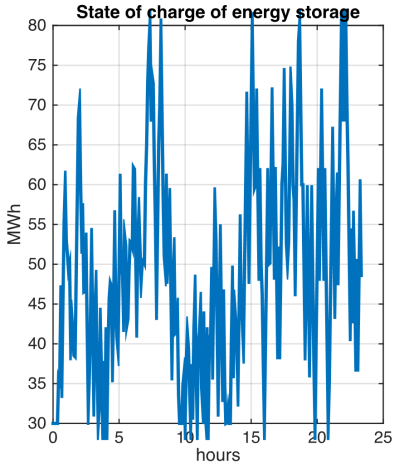
- Tracking an E-Program

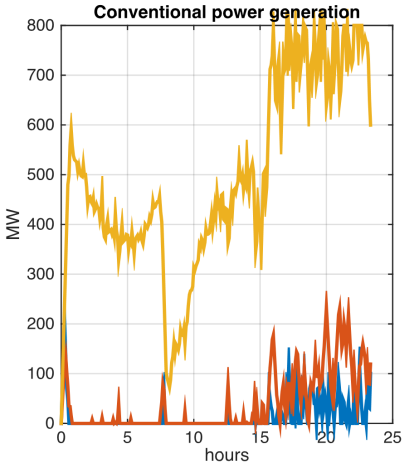
$$\min \sum_{k=0}^{N-1} \gamma(P_{\text{ex},k} - E_k)^2 + (a_i P_{i,k}^2 + b_i P_{i,k} + c_i) - p_k[P_{\text{ex},k} - E_k]$$

$\gamma = 10^3$

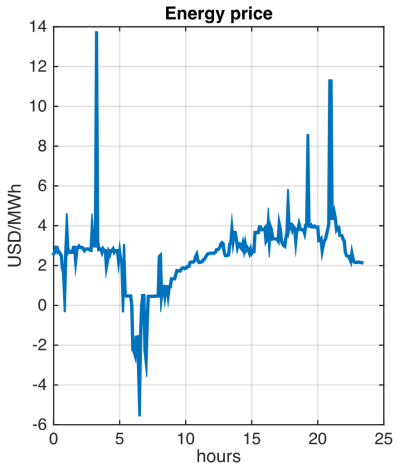
SMPC with branching factor M=[2 2 2 1 1]

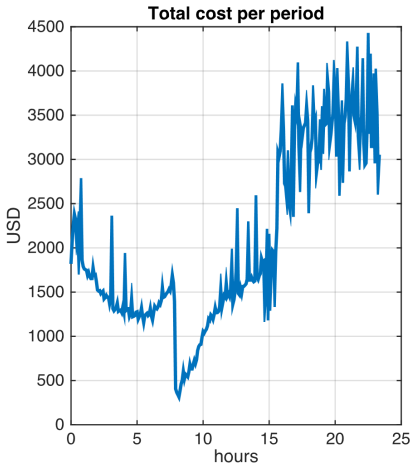


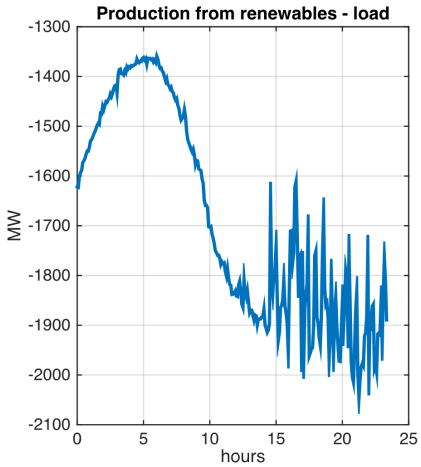




total cost = 574,388 USD





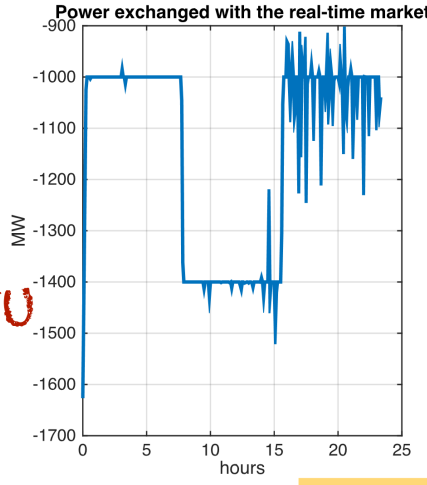


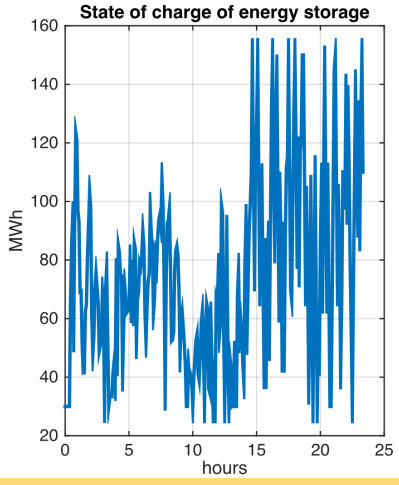
Revenues from day-ahead market are not counted

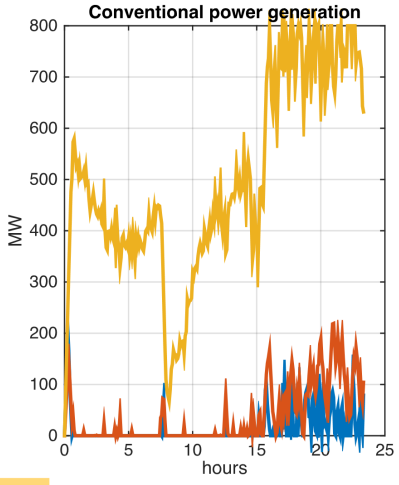
SMPC FOR MARKET-BASED OPTIMAL POWER DISPATCH

- Change storage type: $S_{\min} \le S_k \le S_{\max}$ $\Delta_{S,\min} \le S_{k+1} - S_k \le \Delta_{S,\max}$

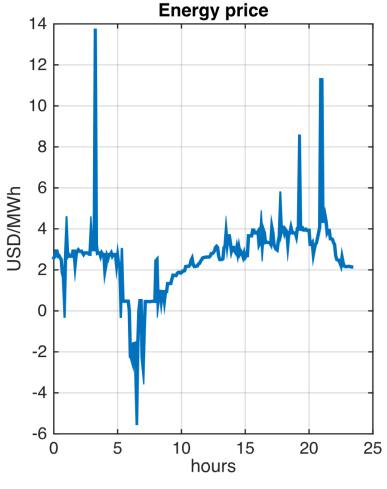
$S_{\min} = 30$
$S_{\max} = \cancel{80}$ 150 MWh
$\Delta_{S,\min} = \cancel{-10}$
$\Delta_{S,\max} = \cancel{10}$ ±40 MWh/PTU

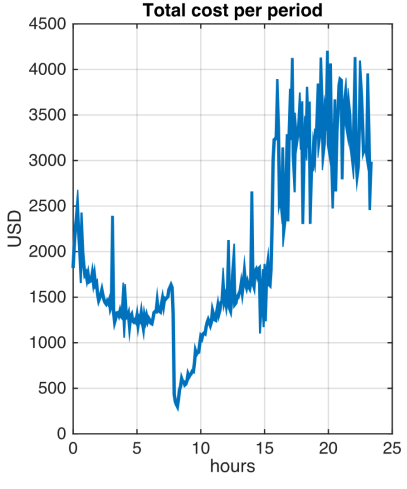


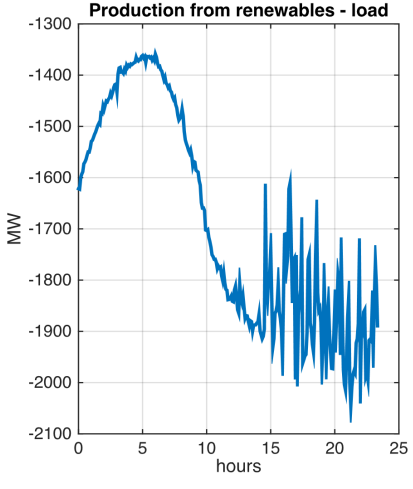




total cost = 563,283 USD







Revenues from day-ahead market are not counted

WORKSHOP PROGRAM

- Linear MPC: introduction and algorithms (AB)
- Nonlinear and economic MPC (MZ)
- Hybrid and stochastic MPC (AB)
- **Reinforcement learning and MPC (AB+MZ)**
- Concluding remarks (AB+MZ)

Supplementary material:

`http://cse.lab.imtlucca.it/~bemporad/mpc_course.html`

`https://mariozanon.wordpress.com/teaching/`