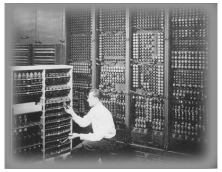
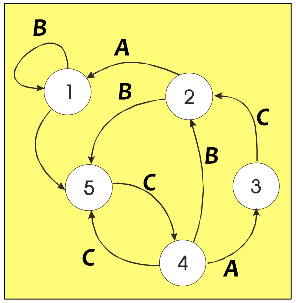
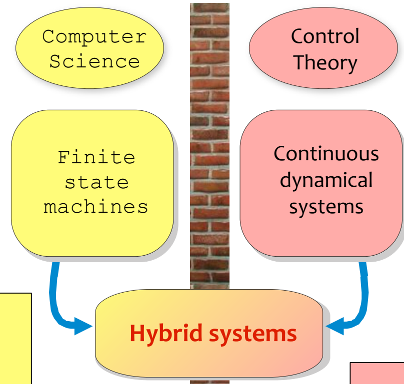
Hybrid Systems: Modeling

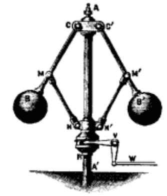
Hybrid systems



$\mathrm{x} \in \{1,2,3,4,5\}$
$\mathrm{u} \in \{A,B,C\}$







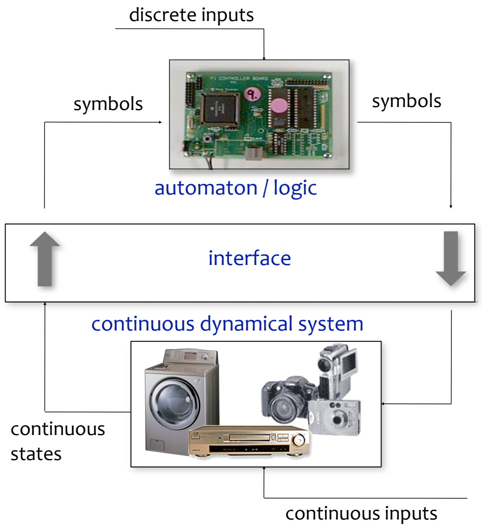
$x \in \mathbb{R}^n$
$u \in \mathbb{R}^m$
$y \in \mathbb{R}^p$

$u(t)$ → system → $y(t)$

$$\begin{cases} \frac{dx(t)}{dt} &= f(x(t),u(t)) \\ y(t) &= g(x(t),u(t)) \end{cases}$$

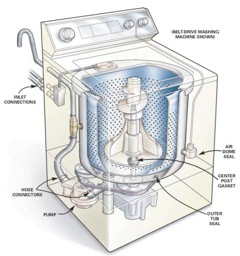
$$\begin{cases} x(k+1) &= f(x(k),u(k)) \\ y(k) &= g(x(k),u(k)) \end{cases}$$

Embedded systems

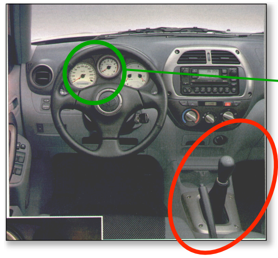


- Automobiles
- Industrial processes
- Consumer electronics
- Home appliances
- …

Example:

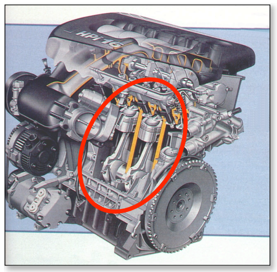


"Intrinsically hybrid" systems



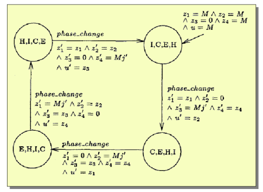
- Transmission

discrete command (R,N,1,2,3,4,5) + continuous dynamical variables (velocities, torques)



- Four-stroke engines

Automaton, dependent on power train motion



Example of hybrid control problem

Cruise control problem

GOAL:

command gear ratio, gas pedal, and brakes to **track** a desired speed and minimize consumptions

CHALLENGES:

- continuous **and** discrete inputs
- dynamics depends on gear
- nonlinear torque/speed maps

Key requirements for hybrid models

- **Descriptive** enough to capture the behavior of the system
  - continuous dynamics (physical laws)
  - logic components (switches, automata, software code)
  - interconnection between logic and dynamics
- **Simple** enough for solving *analysis* and *synthesis* problems

$$\begin{cases} x' = Ax + Bu \\ y = Cx + Du \end{cases}$$

linear systems

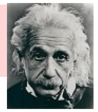
?

linear hybrid systems

$$\begin{cases} x' = f(x,u,t) \\ y = g(x,u,t) \end{cases}$$

nonlinear systems

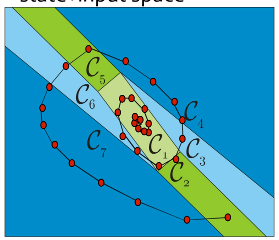
"Make everything as simple as possible, but not simpler."
— Albert Einstein



Piecewise affine systems

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

state+input space

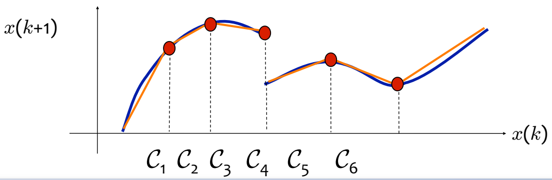


$x \in \mathbb{R}^n,\ u \in \mathbb{R}^m,\ y \in \mathbb{R}^p$

$i(k) \in \{1, \ldots, s\}$

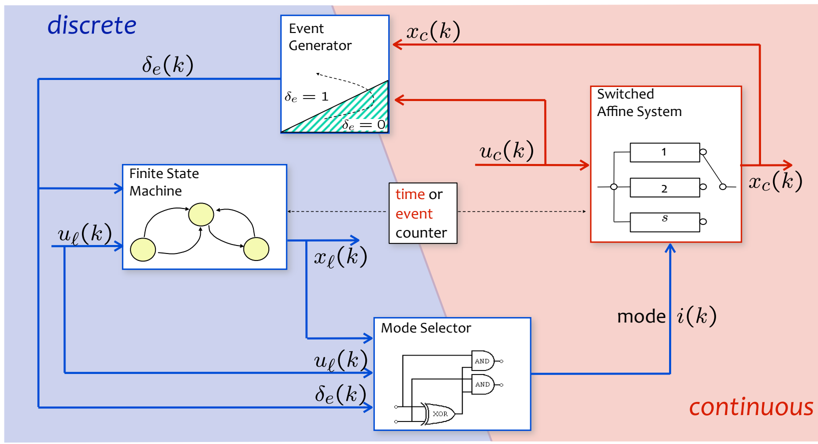
(Sontag 1981)

Can approximate nonlinear and/or discontinuous dynamics arbitrarily well



Discrete Hybrid Automaton (DHA)

(Torrisi, Bemporad, 2004)



$x_\ell \in \{0,1\}^{n_b}$ = binary states

$u_\ell \in \{0,1\}^{m_b}$ = binary inputs

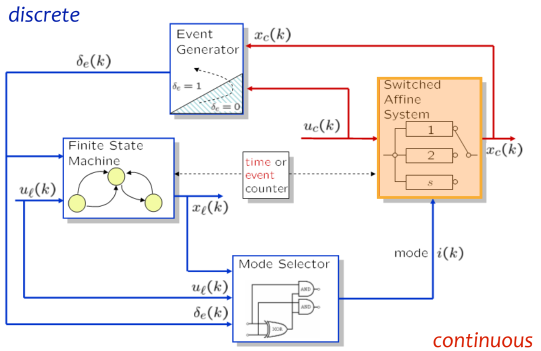
$\delta_e \in \{0,1\}^{n_e}$ = event variables

$x_c \in \mathbb{R}^{n_c}$ = continuous states

$u_c \in \mathbb{R}^{m_c}$ = continuous inputs

$i \in \{1, 2, \ldots, s\}$ = current mode

Switched affine system

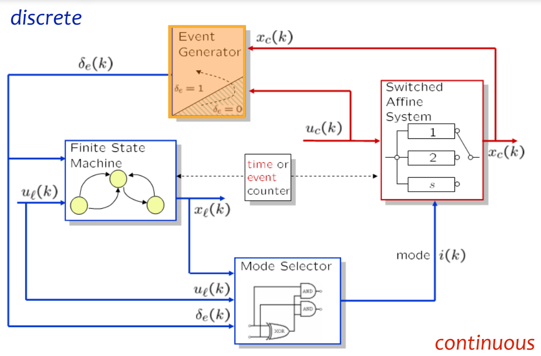


The affine dynamics depend on the current mode $i(k)$:

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

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

Event generator



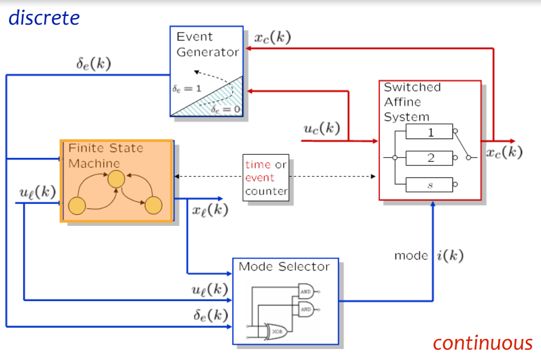
Event variables are generated by linear threshold conditions over continuous states, continuous inputs, and time:

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

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

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

Finite state machine



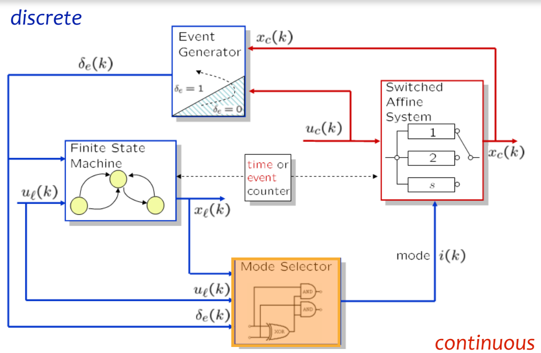
The binary state of the finite state machine evolves according to a Boolean state update function:

$$x_\ell(k+1) = f_{\mathrm{B}}(x_\ell(k), u_\ell(k), \delta_e(k))$$

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

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

Mode selector



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:

$$i(k) = f_{\mathrm{M}}(x_\ell(k), u_\ell(k), \delta_e(k))$$

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

Example:

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

u_ℓ / x_ℓ	0	1
0	$i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$	$i = \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$
1	$i = \left[\begin{smallmatrix}0\\0\end{smallmatrix}\right]$	$i = \left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$

the system has 3 modes

Logic → inequalities

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

(Glover 1975, Williams 1977, Hooker 2000)

0. Given a Boolean statement $F(X_1, X_2, \ldots, X_n) = \text{TRUE}$

1. Convert to Conjunctive Normal Form (CNF):

$$\bigwedge_{j=1}^{m} \left(\bigvee_{i \in P_j} X_i \bigvee_{i \in N_j} \bar{X}_i \right) = \text{TRUE}$$

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

2. Transform into inequalities:

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

Any logic proposition can be translated into **integer linear inequalities**

Logic → inequalities: symbolic approach

Example: $F(X_1, X_2, X_3) = [X_3 \leftrightarrow X_1 \wedge X_2]$

1. Convert to Conjunctive Normal Form (CNF):

(see e.g. http://www.oursland.net/aima/propositionApplet.html or just google for "CNF + applet" …)

$$(X_3 \vee \neg X_1 \vee \neg X_2) \wedge (X_1 \vee \neg X_3) \wedge (X_2 \vee \neg X_3)$$

2. Transform into inequalities:

$$\begin{cases} \delta_3 + (1-\delta_1) + (1-\delta_2) \geq 1 \\ \delta_1 + (1-\delta_3) \geq 1 \\ \delta_2 + (1-\delta_3) \geq 1 \end{cases}$$

Logic → inequalities: geometric approach

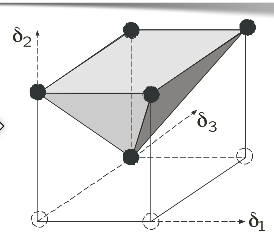
Boolean statement $F(X_1, X_2, \ldots, X_n) = \text{TRUE}$ ⟹ polyhedron $A\delta \leq b, \quad \delta \in \{0,1\}^n$

The polytope $P = \{\delta : \ A\delta \leq b\}$ is the convex hull of the rows of the truth table T associated with formula $F(X_1, \ldots, X_N)$

T:

X_1	X_2	…	X_N
0	0	…	1
0	1	…	0
⋮	⋮	⋮	⋮
1	1	…	0

⟹



Convex hull algorithms: cdd, lrs, qhull, chD, Hull, Porto

CDDMEX package included in the Hybrid Toolbox

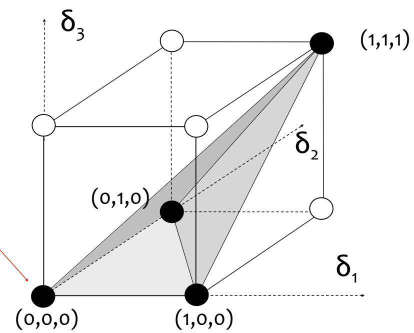
Logic → inequalities: geometric approach

Example: logic "AND"

$F(X_1, X_2, X_3) = [X_3 \leftrightarrow X_1 \wedge X_2]$

T:

X_1	X_2	X_3
0	0	0
0	1	0
1	0	0
1	1	1



Key idea:
White points cannot be in the hull of black points

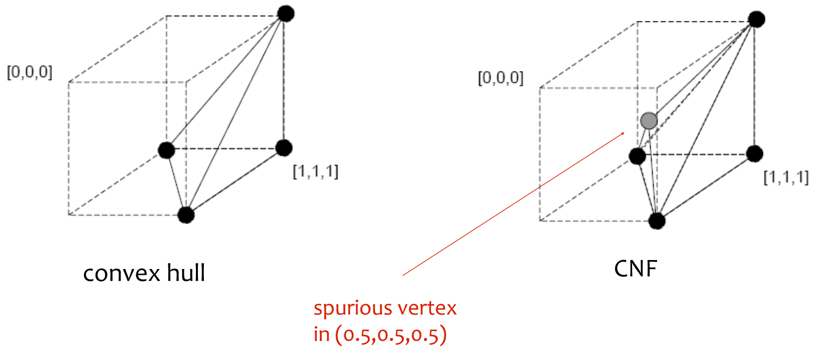
$$\text{conv}\left(\begin{bmatrix} 0\\0\\0 \end{bmatrix}, \begin{bmatrix} 0\\1\\0 \end{bmatrix}, \begin{bmatrix} 1\\0\\0 \end{bmatrix}, \begin{bmatrix} 1\\1\\1 \end{bmatrix} \right) = \left\{ \delta \in \mathbb{R}^3 : \begin{array}{rcl} -\delta_1 + \delta_3 & \leq & 0 \\ -\delta_2 + \delta_3 & \leq & 0 \\ \delta_1 + \delta_2 - \delta_3 & \leq & 1 \end{array} \right\}$$

```
>> V=struct('V',[0 0 0;0 1 0;1 0 0;1 1 1]);
>> H=cddmex('hull',V);A=H.A,b=H.B
```

Geometric vs. symbolic approach

- The polyhedron obtained via convex hull is the smallest one
- The one obtained via CNF may be larger. Example:

$$(X_1 \vee X_2) \wedge (X_1 \vee X_3) \wedge (X_2 \vee X_3) = \text{TRUE}$$



Note: no other example with 3 vars but $(X_1 \vee X_2) \wedge (X_1 \vee X_3) \wedge (X_2 \vee X_3) \wedge (\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3)$

Big-M technique (iff)

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

$$[\delta = 1] \leftrightarrow [a'x_c - b \leq 0]$$

$$x_c \in \mathcal{X} \subset \mathbb{R}^{n_c}$$
$$\delta \in \{0,1\}$$

- Assume $\mathcal{X}$ bounded and let M and m such that

$$M > a'x_c - b, \ \forall x_c \in \mathcal{X}$$
$$m < a'x_c - b, \ \forall x_c \in \mathcal{X}$$

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

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

- To avoid strict inequalities, replace by $a'x_c - b \geq \epsilon + (m - \epsilon)\delta$ where $\epsilon > 0$ is a small number (e.g., machine precision)

Big-M technique (if-then-else)

- Consider the *if-then-else* condition

$$z = \begin{cases} a_1'x_c + f_1 & \text{if } \delta = 1 \\ a_2'x_c + f_2 & \text{otherwise} \end{cases}$$

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

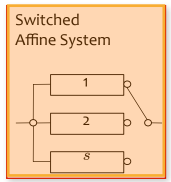
- Assume $\mathcal{X}$ bounded and let M_1, M_2 and m_1, m_2 such that

$$M_1 > a_1'x_c + f_1 > m_1, \ \forall x_c \in \mathcal{X}$$
$$M_2 > a_2'x_c + f_2 > m_2, \ \forall x_c \in \mathcal{X}$$

- The *if-then-else* condition is equivalent to

$$\begin{cases} (m_1 - M_2)(1-\delta) + z \leq a_1 x_c + f_1 \\ (m_2 - M_1)(1-\delta) - z \leq -a_1 x_c - f_1 \\ (m_2 - M_1)\delta + z \leq a_2 x_c + f_2 \\ (m_1 - M_2)\delta - z \leq -a_2 x_c - f_2 \end{cases}$$

Switched affine system



The state-update equation can be rewritten as a difference equation + *if-then-else* conditions:

$$z_1(k) = \begin{cases} A_1 x_c(k) + B_1 u_c(k) + f_1, & \text{if } i(k) = 1 \\ 0, & \text{otherwise}, \end{cases}$$
$$\vdots$$
$$z_s(k) = \begin{cases} A_s x_c(k) + B_s u_c(k) + f_s, & \text{if } i(k) = s, \\ 0, & \text{otherwise}, \end{cases}$$

$$x_c(k+1) = \sum_{i=1}^{s} z_i(k)$$

where $z_i(k) \in \mathbb{R}^{n_c}, i = 1, \ldots, s$

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

Logic and inequalities

$X_1 \vee X_2 = \text{TRUE} \longrightarrow \delta_1+\delta_2 \geq 1, \quad \delta_1,\delta_2 \in \{0,1\}$ (Glover 1975, Williams 1977, Hooker 2000)

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

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

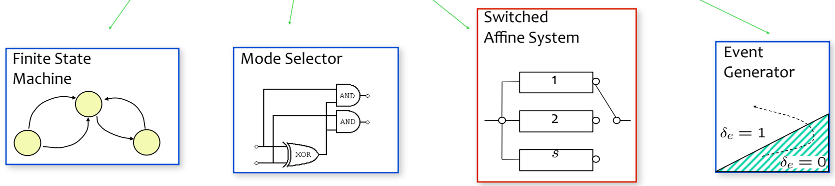
$N_j, P_j \subseteq \{1,\dots,n\}$

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

$$[\delta_e^i(k)=1] \leftrightarrow [H^i x_c(k) \leq W^i] \quad \Rightarrow \quad \begin{cases} H^i x_c(k) - W^i \leq M^i(1-\delta_e^i) \\ H^i x_c(k) - W^i > m^i \delta_e^i \end{cases}$$

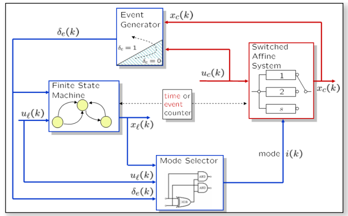
IF $[\delta = 1]$ THEN $z = a_1^T x + b_1^T u + f_1$
ELSE $z = a_2^T x + b_2^T u + f_2$

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



Mixed Logical Dynamical (MLD) systems

Discrete Hybrid Automaton



HYSDEL
(Torrisi, Bemporad, 2004)

Mixed Logical Dynamical (MLD) systems

$$\begin{cases} x_{k+1} = A x_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\ y_k = C x_k + D_1 u_k + D_2 \delta_k + D_3 z_k + D_5 \\ E_2 \delta_k + E_3 z_k \leq E_4 x_k + E_1 u_k + E_5 \end{cases}$$

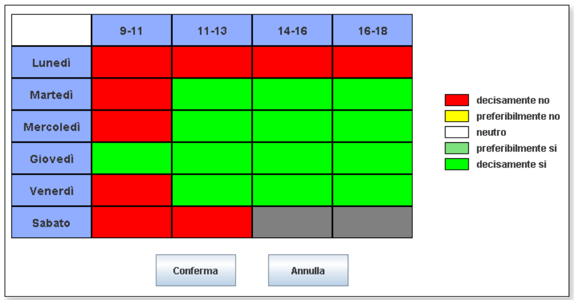
Continuous and binary variables $x \in \mathbb{R}^{n_c} \times \{0,1\}^{n_b}, \; u \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b}$
$y \in \mathbb{R}^{p_c} \times \{0,1\}^{p_b}, \; \delta \in \{0,1\}^{r_b}, \; z \in \mathbb{R}^{r_c}$ (Bemporad, Morari 1999)

- Computationally oriented (mixed-integer programming)
- Suitable for controller synthesis, verification, ...

Mixed-integer models in Operations Research

Translation of logical relations into linear inequalities is heavily used in operations research (OR) for solving complex decision problems by using mixed-integer (linear) programming (MIP)

Example: Timetable generation (for demanding professors ...)



Cost function:
sum of professors' preferences

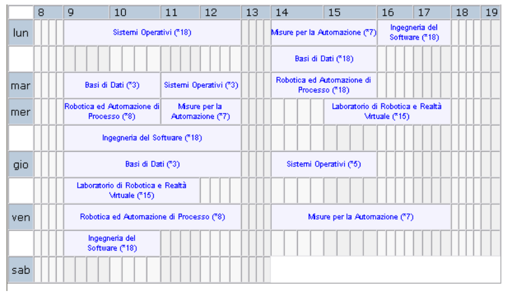
Constraints:
professors [students] cannot teach [take] two courses at the same time, etc.

Mixed-integer models in Operations Research

Translation of logical relations into linear inequalities is heavily used in operations research (OR) for solving complex decision problems by using mixed-integer (linear) programming (MIP)

Example: Timetable generation (for demanding professors ...)

CPU time: 0.2 s



Effort: 5% mathematical problem setup (MILP model)
35% database & web interfaces
60% deal with professors' complaints, complaints, complaints ...

Mixed-integer models in Operations Research

Translation of logical relations into linear inequalities is heavily used in operations research (OR) for solving complex decision problems by using mixed-integer (linear) programming (MIP)

Example: Optimal multi-period investments for maintenance and upgrade of electrical energy distribution networks

(Bemporad, Muñoz, Piazzesi, 2006)

Enel

A simple example

- System: $x(k+1) = \begin{cases} 0.8x(k) + u(k) & \text{if } x(k) \geq 0 \\ -0.8x(k) + u(k) & \text{if } x(k) < 0 \end{cases}$

$$-10 \leq x(k) \leq 10$$

- Associate $[\delta(k) = 1] \leftrightarrow [x(k) \geq 0]$ and transform

$$x(k) \geq m(1 - \delta(k)) \qquad M = -m = 10$$
$$x(k) \leq -\epsilon + (M + \epsilon)\delta(k) \qquad \epsilon > 0 \text{ "small"}$$

- Then $x(k+1) = 1.6\delta(k)x(k) - 0.8x(k) + u(k)$

$z(k) = \delta(k)x(k)$ →

$$z(k) \leq M\delta(k) \qquad \delta(k) \in \{0, 1\}$$
$$z(k) \geq m\delta(k)$$
$$z(k) \leq x(k) - m(1 - \delta(k))$$
$$z(k) \geq x(k) - M(1 - \delta(k))$$

- Rewrite as a linear equation

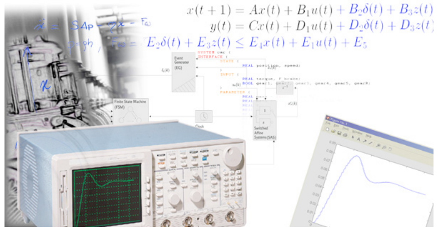
$$x(k+1) = 1.6z(k) - 0.8x(k) + u(k)$$

(**Note:** this is the nonlinear system $x(k+1) = 0.8|x(k)| + u(k)$)

HYSDEL

(HYbrid Systems DEscription Language)

- Describe *hybrid systems:*
  - Automata
  - Logic
  - Lin. Dynamics
  - Interfaces
  - Constraints



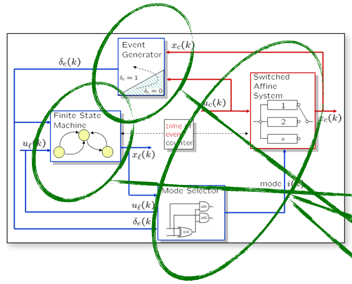
(Torrisi, Bemporad, 2004)

- Automatically generate MLD models in MATLAB

Download: http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox

Reference: http://control.ethz.ch/~hybrid/hysdel

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; }

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

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

      CONTINUOUS {
         xc = z; }

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

Additional relations constraining system's variables

Example 1: Definition of event variables

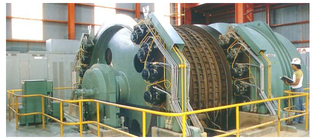
$$[\delta = 1] \leftrightarrow [h \geq h_{\text{max}}]$$

$\delta \in \{0, 1\}$
$h \in \mathbb{R}$

```
AD { delta = hmax - h <= 0; }
```


Example 2: Nonlinear (PWA) functions

Nonlinear amplification unit

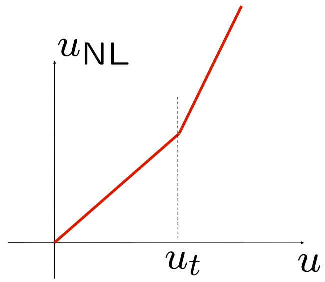


$$u_{\text{NL}}(k) = \begin{cases} u(k) & \text{if } u(k) < u_t \\ 2.3u(k) - 1.3u_t & \text{if } u(k) \geq u_t \end{cases}$$

```
DA { unl = { IF th THEN 2.3*u - 1.3*ut
                   ELSE u }; }

AD { th = ut - u <= 0; }
```

$$[t_h = 1] \leftrightarrow [u \geq u_t]$$



Example 3: Logical relations

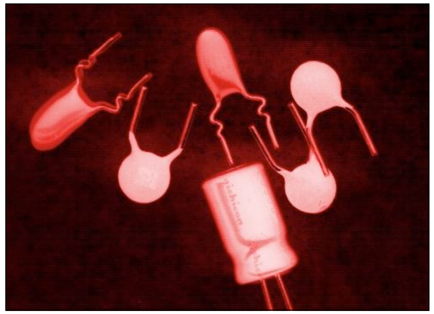
Rule: brake if there is an alarm signal, but only if the train is not on fire in a tunnel

$\delta_{\text{brake}}, \delta_{\text{alarm}}, \delta_{\text{tunnel}}, \delta_{\text{fire}} \in \{0, 1\}$

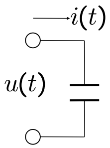
$$\delta_{\text{brake}} = \delta_{\text{alarm}} \wedge \neg(\delta_{\text{tunnel}} \wedge \delta_{\text{fire}})$$

```
LOGIC {
   brake =  alarm & ~(tunnel & fire);
}
```


Example 4: Continuous dynamics



$$i(t) = C\frac{du(t)}{dt}$$



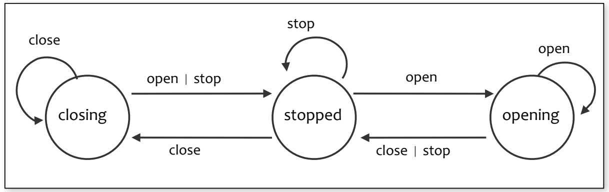
Apply forward difference rule:

$$u((k+1)T_s) = u(kT_s) + \frac{T}{C}i(kT_s)$$

```
CONTINUOUS {
        u = u + Ts*iC*i;
}
```

Note: `iC` = $1/C$ is used due to a bug in HYSDEL, that cannot handle division by a scalar parameter

Example 5: Automaton



Flow control through a dam

binary inputs: $u_{\text{open}}, u_{\text{close}}, u_{\text{stop}} \in \{0,1\}$

binary states: $x_{\text{opening}}, x_{\text{closing}}, x_{\text{stopped}} \in \{0,1\}$

```
AUTOMATA {
    xclosing = (uclose & xclosing) | (uclose & xstopped);
    xstopped = ustop | (uopen & xclosing) | (uclose & xopening);
    xopening = (uopen & xstopped) | (uopen & xopening);
}
```

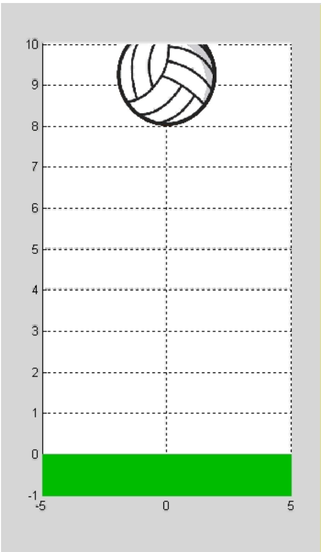

Example 6: Impose a constraint

$$0 \leq h(k) \leq h_{\text{max}}$$

```
MUST {
      h - hmax <= 0;
      -h       <= 0;
}
```

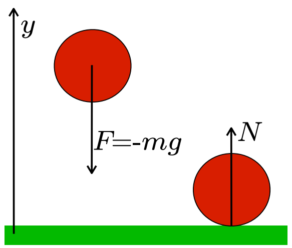
Arbitrary logic constraints are also supported

Bouncing ball example



$$\ddot{y} = -g$$
$$y \leq 0 \;\Rightarrow\; \dot{y}(t^+) = -(1-\alpha)\dot{y}(t^-)$$

$\alpha \in [0,1]$

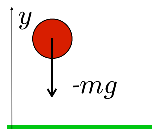


How to model the bouncing ball as a discrete-time hybrid system ?

Bouncing ball – Time discretization

$y(t) > 0$

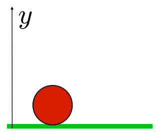
$$v(t) \approx \frac{y(t) - y(t-1)}{T_s} \qquad -g = \ddot{y}(t) \approx \frac{v(t) - v(t-1)}{T_s}$$



$$\begin{cases} v(t+1) &= v(t) - T_s g \\ y(t+1) &= y(t) + T_s v(t+1) \\ &= y(t) + T_s v(t) - T_s^2 g \end{cases}$$

$y(t) \leq 0$

$$v(t) = -(1-\alpha)v(t-1) \qquad \begin{aligned} y(t+1) &= y(t-1) \\ &= y(t) - T_s v(t) \end{aligned}$$



$$\begin{cases} v(t+1) &= -(1-\alpha)v(t) \\ y(t+1) &= y(t) - T_s v(t) \end{cases}$$

go to demo **/demos/hybrid/bball.m**

Bouncing ball - HYSDEL model

```
SYSTEM bouncing_ball {
INTERFACE {
/* Description of variables and constants */
        STATE { REAL height [-10,10];
                REAL velocity [-100,100];    }


        PARAMETER {
                REAL g;
                REAL alpha;   /* 0=elastic, 1=completely anelastic */
                REAL Ts; }
}
IMPLEMENTATION {
        AUX {   BOOL negative;
                REAL hnext;
                REAL vnext;    }


        AD {    negative = height <= 0; }


        DA {    hnext = {  IF negative THEN height-Ts*velocity
                        ELSE height+Ts*velocity-Ts*Ts*g};
                vnext = {  IF negative THEN -(1-alpha)*velocity
                        ELSE velocity-Ts*g};  }
        CONTINUOUS {
                height   = hnext;
                velocity = vnext;}
}}
```

go to demo **/demos/hybrid/bball.m**

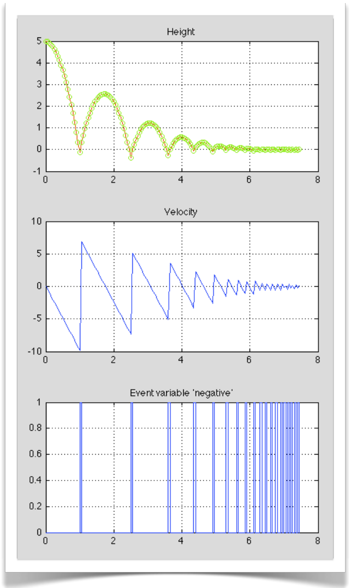
Bouncing ball - Simulation

```
>>Ts=0.05;
>>g=9.8;
>>alpha=0.3;

>>S=mld('bouncing_ball',Ts);

>>N=150;
>>U=zeros(N,0);
>>x0=[5 0]';

>>[X,T,D]=sim(S,x0,U);
```

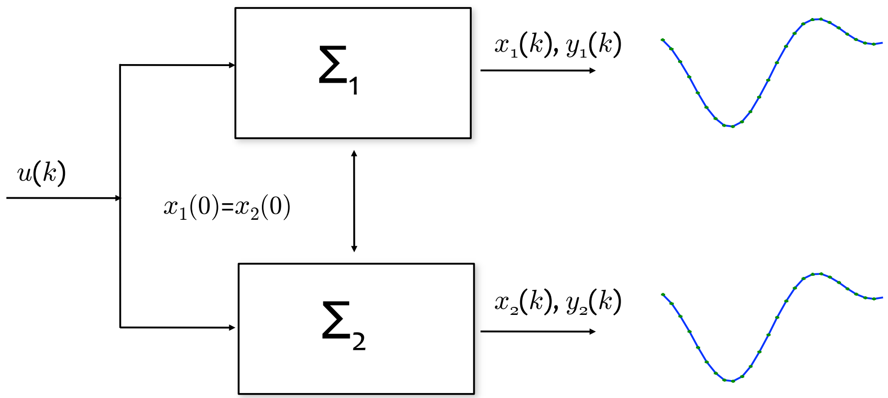


Note: no Zeno effect in discrete time !

Realization and Transformations (State-Space Hybrid Models)

Equivalences of hybrid models

Definition 1 *Two hybrid systems* Σ_1, Σ_2 *are* equivalent *if for all initial conditions* $x_1(0) = x_2(0)$ *and input* $\{u_1(k)\}_{k\in\mathbb{Z}_+} = \{u_2(k)\}_{k\in\mathbb{Z}_+}$ *then* $x_1(k) = x_2(k)$ *and* $y_1(k) = y_2(k)$, *for all* $k \in \mathbb{Z}_+$.



MLD and PWA Systems

Theorem MLD systems and PWA systems are equivalent

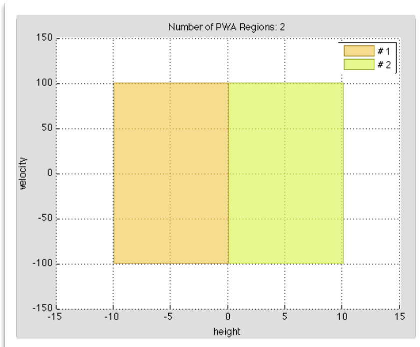
(Bemporad, Ferrari-Trecate, Morari, *IEEE TAC,2000*)

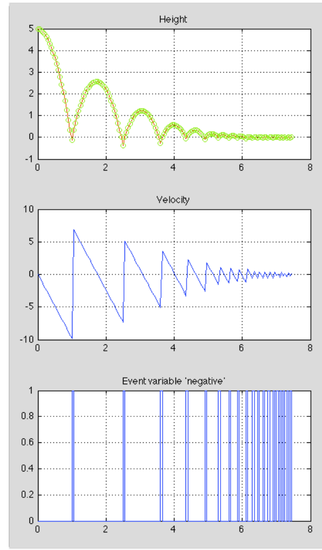
- Proof is constructive: given an MLD system it returns its equivalent PWA form
- Drawback: it needs the enumeration of all possible combinations of binary states, binary inputs, and δ variables
- Most of such combinations lead to empty regions
- Efficient algorithms are available for converting MLD models into PWA models avoiding such an enumeration:
  - A. Bemporad, "*Efficient Algorithms for Converting Mixed Logical Dynamical Systems into an Equivalent Piecewise Affine Form*", IEEE Trans. Autom. Contr., 2004.
  - T. Geyer, F.D. Torrisi and M. Morari, "*Efficient Mode Enumeration of Compositional Hybrid Models*", HSCC'03

Bouncing ball - PWA equivalent

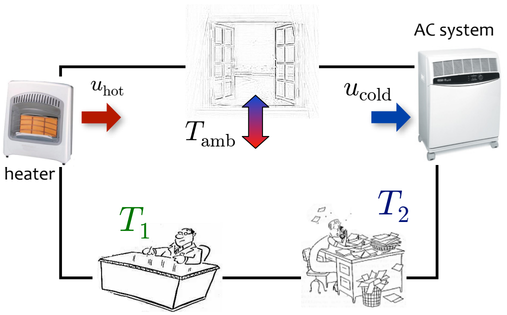
```
>>P=pwa(S);
>>plot(P)

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





Example: Room temperature



Hybrid dynamics

- #1 turns the heater (A/C) on whenever he is cold (hot)
- If #2 is cold he turns the heater on, unless #1 is hot
- If #2 is hot he turns A/C on, unless #2 is cold
- Otherwise, heater and A/C are off

- $\dot{T}_1 = -\alpha_1(T_1 - T_{amb}) + k_1(u_{hot} - u_{cold})$ (body temperature dynamics of #1)
- $\dot{T}_2 = -\alpha_2(T_2 - T_{amb}) + k_2(u_{hot} - u_{cold})$ (body temperature dynamics of #2)

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

HYSDEL model

```
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

Hybrid MLD model

- MLD model

$$\begin{aligned} x(k+1) &= Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) \\ y(k) &= Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) \\ E_2\delta(k) + E_3z(k) &\leq E_1u(k) + E_4x(k) + E_5 \end{aligned}$$

- 2 continuous states: (temperatures T_1,T_2)
- 1 continuous input: (room temperature T_{amb})
- 2 auxiliary continuous vars: (power flows u_{hot}, u_{cold})
- 6 auxiliary binary vars: (4 thresholds + 2 for OR condition)
- 20 mixed-integer inequalities

Possible combination of integer variables: $2^6 = 64$

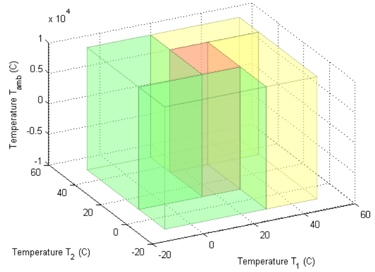
Hybrid PWA model

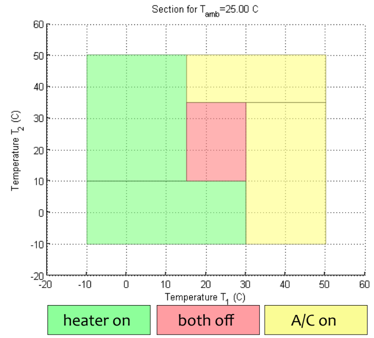
- PWA model

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

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

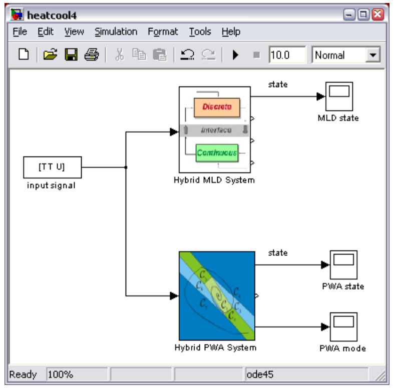
- 2 continuous states: (temperatures T_1,T_2)
- 1 continuous input: (room temperature T_{amb})

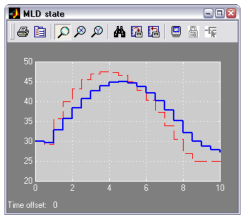


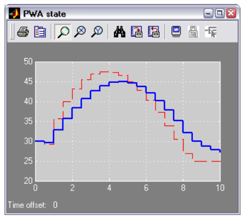


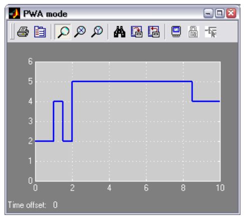
- 5 polyhedral regions *(partition does not depend on input)*

Simulation in Simulink









MLD and PWA models are equivalent

Using MLD to PWA for Model Checking

- Assume plant + controller can be modeled as DHA:
  - **Plant** = PWA approximation (e.g.: nonlinear switched model)
  - **Controller** = switched linear controller (e.g: a combination of threshold conditions, logic, linear feedback laws, …)
- Write HYSDEL model, convert to MLD, then to PWA
- The resulting PWA map tells how the closed-loop system behaves in different regions of the state-space

Identification of Hybrid Systems

Hybrid system identification

- Sometimes a *hybrid model* of the process (or of a part of it) cannot be derived manually from available knowledge.
- Therefore, a model must be either
  - Estimated from data (model unknown)
  - or *hybridized* before it can be used for control/analysis (model known but nonlinear)
- If a linear model is enough, no problem: several algorithms are available (e.g.: use Ljung's ID TBX)
- If switching modes are known and data can be generated for each mode, no problem: we identify one linear model per mode (e.g.: use Ljung's ID TBX)
- If modes & dynamics must be identified together, we need

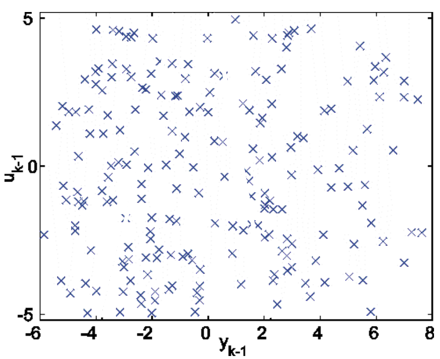
hybrid system identification

PWA identification problem

Estimate from data **both** the parameters of the affine submodels **and** the partition of the PWA map

Example Let the data be generated by the PWARX system

$$y_k = \begin{cases} \begin{bmatrix} -0.4 & 1 & 1.5 \end{bmatrix}\varphi_k + \varepsilon_k & \text{if } \begin{bmatrix} 4 & -1 & 10 \end{bmatrix}\varphi_k < 0 \\ \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix}\varphi_k + \varepsilon_k & \text{if } \begin{bmatrix} -4 & 1 & -10 \\ 5 & 1 & -6 \end{bmatrix}\varphi_k \leq 0 \\ \begin{bmatrix} -0.3 & 0.5 & -1.7 \end{bmatrix}\varphi_k + \varepsilon_k & \text{if } \begin{bmatrix} -5 & -1 & 6 \end{bmatrix}\varphi_k < 0 \end{cases}$$



with $\varphi_k = [y_{k-1}\ u_{k-1}\ 1]'$, $|u_k| \leq 5$ and $|\varepsilon_k| \leq 0.1$

PWA identification problem

$$\min_{\theta_j, H_j} \frac{1}{2N} \sum_{t=1}^{N} \left(\sum_{j=1}^{s} \|y_t - \varphi_t'\theta_j\| \mathcal{J}_j(\varphi_t) \right)$$

$$\text{subj. to } \mathcal{J}_j(\varphi_t) = \begin{cases} 1 & \text{if } H_j\varphi_t \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$+ \text{ linear bounds over } \theta_j,\ H_j$$

$\|v\|_2^2$	$=$	$v'v$	Euclidean norm
$\|v\|_\infty$	$=$	$\max \|v_i\|$	∞-norm
$\|v\|_1$	$=$	$\sum \|v_i\|$	1-norm

A. Known guardlines (partition H_j known, θ_j unknown): ordinary least-squares problem (or linear/quadratic program if linear bounds over θ_j are given)
EASY PROBLEM ...

B. Unknown guardlines (partition H_j and θ_j unknown): generally non-convex, local minima **HARD PROBLEM!**

Approaches to PWA Identification

- Mixed-integer linear or quadratic programming

  J. Roll, A. Bemporad and L. Ljung, "*Identification of hybrid systems via mixed-integer programming*", Automatica, 2004

- Bounded error & partition of infeasible set of inequalities

  A. Bemporad, A. Garulli, S. Paoletti and A. Vicino, "*A Greedy Approach to Identification of Piecewise Affine Models*", HSCC'03 / IEEE TAC 2005

- K-means clustering in a feature space

  G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, "*A clustering technique for the identification of piecewise affine systems*", Automatica, 2003

- Bayesian approach

  Juloski A, Wieland S, Heemels WPMH, "A Bayesian approach to identification of hybrid systems. CDC 2004

- Other approaches:

  - Polynomial factorization (algebraic approach) (R. Vidal, S. Soatto, S. Sastry, 2003)

  - Hyperplane clustering in data space (E. Münz, V. Krebs, *IFAC* 2002)