Hybrid Models for Analysis and Control Design

Alberto Bemporad

Dip. di Ingegneria dell’Informazione
Università degli Studi di Siena
bemporad@dii.unisi.it
http://www.dii.unisi.it/~bemporad

Motivation: Embedded Systems

• Consumer electronics
• Home appliances
• Office automation
• Automobiles
• Industrial plants
• ...

Motivating Problem #1

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

Hybrid Systems

\[ \begin{align*}
    x \in \{1, 2, 3, 4, 5\} \\
    u \in \{A, B, C\} \\
    x_{0} \\
    y(t) &= g(x(t), u(t)) \\
    \frac{dx(t)}{dt} &= f(x(t), u(t))
\end{align*} \]
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

\[
\begin{align*}
  x'(k) &= A_i(k)x(k) + B_i(k)u(k) + f_i(k) \\
y(k) &= C_i(k)x(k) + D_i(k)u(k) + g_i(k) \\
i(k) \text{ s.t. } H_i(k)x(k) + J_i(k)u(k) &\leq K_i(k)
\end{align*}
\]

\(x \in \mathcal{X} \subseteq \mathbb{R}^n, \ u \in \mathcal{U} \subseteq \mathbb{R}^m, \ y \in \mathcal{Y} \subseteq \mathbb{R}^l\) 
i(k) \in \{1, \ldots, s\}

- Approximates nonlinear dynamics arbitrarily well

- Suitable for stability analysis, reachability analysis (verification), controller synthesis, ...

---

**Piecewise Affine Systems**

(Sontag 1981)

\(x' \in \mathbb{R}^n, \ y \in \mathbb{R}^m, \ u \in \mathbb{R}^m\) 
i(k) \in \{1, \ldots, s\}

**Discrete Hybrid Automata**

(Torrisi, Bemporad, 2003)

**Discrete Hybrid Automata**

(Torrisi, Bemporad, 2003)

**Switched Affine System**
Discrete Hybrid Automata

Event Generator

$[\delta'_i(k) = 1] \iff [H^t x_r(k) + K^t u_r(k) + J^k y_{\delta'_i}(k) \leq W_i^t]$, $\delta'_i \in \{0, 1\}$

$x_r \in \mathbb{R}^n_r$, $u_r \in \mathbb{R}^m_r$, $\delta'_i \in \{0, 1\}$

Finite State Machine

Discrete dynamics evolving according to a Boolean state update function

$x_{\delta_i}(k + 1) = f_{\delta_i}(x_{\delta_i}(k), u_{\delta_i}(k), \delta_{i}(k));$

$y_{\delta_i}(k) = g_{\delta_i}(x_{\delta_i}(k), u_{\delta_i}(k), \delta_{i}(k));$

$x_{\delta_i} \in \{0, 1\}^n$, $y_{\delta_i} \in \{0, 1\}^p$

Mode Selector

a Boolean function selects the active mode $i(k)$ of the SAS

$i(k) = f_M(x_{\delta_i}(k), u_{\delta_i}(k), \delta_{i}(k)) \in \{0, 1\}^n$

Logic and Inequalities

Glover 1975, Williams 1977

$X_1 \lor X_2$

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

$A \leq B$

$\delta_i + \delta_j \geq 1$

$\delta_i \leq \delta_j$

$1 \leq \sum_{i \in N_i} \sum_{j \in N_j} \delta_i + \sum_{j \in N_j} (1 - \delta_j)$

$\delta_{i}(k) \iff [H^{t} x_{r}(k) \leq W_{i}^{t}]$

$H^{t} x_{r}(k) - W_{i}^{t} \leq M (1 - \delta'_{i})$

$H^{t} x_{r}(k) - W_{i}^{t} > m_{i,0}$

$\delta_{i}(k) \iff [H^{t} x_{r}(k) \leq W_{i}^{t}]$

$z = a_{1} x + b_{1} u + f_{1}$

$z = a_{2} x + b_{2} u + f_{2}$

$z = a_{1} x + b_{1} u + f_{1}$

$z = a_{2} x + b_{2} u + f_{2}$

$z = a_{1} x + b_{1} u + f_{1}$

$z = a_{2} x + b_{2} u + f_{2}$

$z = a_{1} x + b_{1} u + f_{1}$

$z = a_{2} x + b_{2} u + f_{2}$
Logic $\rightarrow$ Inequalities: Symbolic Approach

1. Convert to Conjunctive Normal Form (CNF):

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

2. Transform into inequalities:

$A \delta \leq B, \ \delta \in \{0,1\}$

$\sum_{i \in P_1} \delta_i + \sum_{i \in N_1} (1 - \delta_i) \leq 1$

$\sum_{i \in P_m} \delta_i + \sum_{i \in N_m} (1 - \delta_i) \leq 1$

$\Rightarrow$ Every logic proposition can be translated into linear integer inequalities

Logic $\rightarrow$ Inequalities: Geometric Approach

Geometric approach: The polytope $P = \{ \delta : A \delta \leq B \}$ is the convex hull of the rows of the truth table $T$

$T: \begin{array}{cccc}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}$

$\Rightarrow A \delta \leq B, \ \delta \in \{0,1\}^n$

$\Rightarrow$ Every logic proposition can be translated into linear integer inequalities

Mixed Logical Dynamical Systems

$x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) + B_5$

$y(t) = Cx(t) + D_1 u(t) + D_2 \delta(t) + D_3 z(t) + D_5$

$E_2 \delta(t) + E_3 z(t) \leq E_4 x(t) + E_5 u(t)$

(Bemporad, Morari 1999)

Continuous and binary variables:

$x \in \mathbb{R}^n \times \{0,1\}^m, \ u \in \mathbb{R}^m \times \{0,1\}^m, \ \delta \in \{0,1\}^n, \ z \in \mathbb{R}^n$

• Computationally oriented (Mixed Integer Programming)
• Suitable for optimal controller synthesis, verification, ...
A Simple Example

• System:
  \[ x(t+1) = \begin{cases} 
    0.8x(t) + u(t) & \text{if } x(t) \geq 0 \\
    -0.8x(t) + u(t) & \text{if } x(t) < 0 
  \end{cases} \]
  \quad 10 \leq x(t) \leq 10, \quad -1 \leq u(t) \leq 1

• Associate \( \delta(t) = 1 \) \( \iff \) \( x(t) \geq 0 \) and transform
  \[ -m\delta(t) \leq x(t) - m \]
  \[ -(M+\epsilon)\delta(t) \leq -x(t) - \epsilon \quad \epsilon > 0 \quad \text{small} \]

• Then \( x(t+1) = 1.6z(t) - 0.8x(t) + u(t) \)

• Rewrite as a linear equation
  \[ x(t+1) = 1.6z(t) - 0.8x(t) + u(t) \]

Example 1: AD section

\[ s = T \quad \iff \quad [h \geq h_{\text{min}}] \]

SYSTEM tank {
  INTERFACE {
    STATE {
      REAL h [-0.3, 0.3];
    }
    INPUT {
      REAL Q [-10, 10];
    }
    PARAMETER {
      REAL k = 1;
      REAL e = 1e-6;
    }
  }
  IMPLEMENTATION {
    AUX {
      BOOL s;
    }
    AD {
      s = hmax - h \leq 0;
    }
    CONTINUOUS {
      h = h + k \cdot Q;
    }
  }
}

Example 2: DA section

SYSTEM motor {
  INTERFACE {
    STATE {
      REAL ucomp;
    }
    INPUT {
      REAL u [0, 10];
    }
    PARAMETER {
      REAL ut = 1;
      REAL e = 1e-6;
    }
  }
  IMPLEMENTATION {
    AUX {
      REAL unl;
      BOOL th;
    }
    AD {
      th = ut - u \leq 0;
    }
    DA {
      unl = \begin{cases} 
        2.3u - 1.3ut & \text{if } th \text{ THEN } \\
        u & \text{ELSE }
      \end{cases} 
    }
    CONTINUOUS {
      ucomp = unl;
    }
  }
}

HYSDEL

(HYbrid Systems DEscription Language)

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

• Automatically generate MLD models in Matlab

Download:
http://www.dii.unisi.it/~bemporad/tools.html
http://control.ethz.ch/~hybrid/hysdel
Example 3: LOGIC section

SYSTEM train {
    INTERFACE {
        STATE {
            BOOL brake; 
        }
        INPUT {
            BOOL alarm, tunnel, fire; 
        }
    } /* end interface */
    IMPLEMENTATION {
        AUX {
            BOOL decision; 
        }
        LOGIC {
            decision = alarm & (~tunnel | ~fire); 
        }
        AUTOMATA {
            brake = decision; 
        }
        MUST {
            fire -> alarm; 
        } /* end implementation */
    } /* end system */
}

Example 4: CONTINUOUS section

SYSTEM capacitorD {
    INTERFACE {
        STATE {
            REAL u; 
        }
        PARAMETER {
            REAL R = 1e4; REAL C = 1e-4; REAL T = 1e-1; 
        }
    } /* end interface */
    IMPLEMENTATION {
        CONTINUOUS {
            u = u - T/C*R*i; 
        } /* end implementation */
    } /* end system */
}

Example 5: AUTOMATA section

SYSTEM outflow {
    INTERFACE {
        STATE {
            BOOL closing, stop, opening; 
        }
        INPUT {
            BOOL uclose, uopen, ustop; 
        }
    } /* end interface */
    IMPLEMENTATION {
        AUTOMATA {
            closing = (uclose & closing) | (uclose & stop); 
            stop    = ustop | (uopen & closing) | (uclose & opening); 
            opening = (uopen & stop) | (uopen & opening); 
        }
        MUST {
            ~(uclose & uopen); 
            ~(uclose & ustop); 
            ~(uopen & ustop); 
        } /* end implementation */
    } /* end system */
}

Example 6: MUST section

SYSTEM watertank {
    INTERFACE {
        STATE {
            REAL h; 
        }
        INPUT {
            REAL Q; 
        }
        PARAMETER {
            REAL hmax = 0.3; REAL k = 1; 
        }
    } /* end interface */
    IMPLEMENTATION {
        CONTINUOUS {
            h = h + k*Q; 
        }
        MUST {
            h <= hmax; 
            h <= Q; 
        } /* end implementation */
    } /* end system */
Modeling Flow

\[ x(t + 1) = A_1 x(t) + B_1 u(t) + f_1 \]

\[ x(t + 1) = A_2 x(t) + B_2 u(t) + f_2 \]

\[ x(t + 1) = A_3 x(t) + B_3 u(t) + f_3 \]

\[ g_12 x(t) + h_12 u(t) < k_{12} \]

\[ g_32 x(t) + h_32 u(t) < k_{32} \]

\[ g_23 x(t) + h_23 u(t) < k_{23} \]

\[ g_13 x(t) + h_13 u(t) < k_{13} \]

Example: Bouncing Ball

\[ \dot{x} = -g \]

\[ x \leq 0 \Rightarrow \dot{x}(t^+) = -(1 - \alpha)\dot{x}(t^-) \]

\[ \alpha \in [0, 1] \]

How to model this system in MLD form?

HYSDEL - Bouncing Ball

```system
SYSTEM bouncing_ball {
    INTERFACE {
        /* Description of variables and constants */
        STATE { REAL height [-10,10];
                 REAL velocity [-100,100]; }
    }
    PARAMETER {
        REAL g=9.8;
        REAL dissipation=.4; /* 0=elastic, 1=completely anelastic */
        REAL Ts=.05; }
    }
    IMPLEMENTATION {
        AUX { REAL z1;
                 REAL z2;
                 BOOL negative; }
        AD { negative = height <= 0; }
        DA { z1 = { IF negative THEN height-Ts*velocity
                    ELSE height+Ts*velocity-Ts*Ts*g};
             z2 = { IF negative THEN -(1-dissipation)*velocity
                    ELSE velocity-Ts*g};
        }
        CONTINUOUS {
            height = z1;
            velocity=z2; }
    }
}```

System Theory for Hybrid Systems

- Analysis
  - Well-posedness
  - Realization & Transformation
  - Reachability (=Verification)
  - Observability
  - Stability

- Synthesis
  - Control
  - State estimation
  - Identification
  - Modeling language
Well-posedness

Are state and output trajectories defined? Uniquely defined? Persistently defined?

- MLD well-posedness:

\[
\delta(t) = F(x(t), u(t))
\]

\[
z(t) = G(x(t), u(t))
\]

\[
\{x(t), u(t)\} \rightarrow \{x(t+1)\}
\]

\[
\{x(t), u(t)\} \rightarrow \{y(t)\}
\]

are single valued

**Definition 1** Let \( \Omega \subseteq \mathbb{R}^n \times \mathbb{R}^m \) be a set of input-state pairs. A hybrid MLD system is called well-posed on \( \Omega \), if for all pairs \((x(t), u(t)) \in \Omega \) there exists a solution \( x(t+1), y(t), \delta(t), z(t) \) and moreover, \( x(t+1), y(t) \) are uniquely determined.

Numerical test based on mixed-integer programming available

(Bemporad, Morari, Automatica, 1999)

---

**Other Existing Hybrid Models**

- **Linear complementarity (LC) systems** (Heemels, 1999)

\[
x(t+1) = Ax(t) + Bu(t) + B_2w(t)
\]

\[
y(t) = Cx(t) + Dw(t) + D_2w(t)
\]

\[
v(t) = E_1x(t) + E_2u(t) + E_3w(t) + e
\]

\[
0 \leq v(t), w(t) \geq 0
\]

Ex: mechanical systems, circuits with diodes etc.

---

- **Extended linear complementarity (ELC) systems**

(De Schutter, De Moor, 2000)

- **Min-max-plus-scaling (MMPS) systems**

(De Schutter, Van den Boom, 2000)

\[
x(t+1) = M_1x(t), u(t), d(t))
\]

\[
y(t) = M_2x(t), u(t), d(t))
\]

\[
0 \geq M_3x(t), u(t), d(t))
\]

**MMPS function:** defined by the grammar

\[
M : = x \mid 0 \mid \max(M_1, M_2) \mid \min(M_1, M_2) \mid M_1 + M_2 \mid \beta(M_1)
\]

Example: \( x(t+1) = 2 \max(x(t), 0) + \min(-\frac{1}{2}x(t), 1) \)

Used for modeling discrete-event systems (t=event counter)

---

**Realization and Transformation**

(State-Space Hybrid Models)

---

**Equivalences of Hybrid Models**

**Definition 1** Two hybrid systems \( \Sigma_1, \Sigma_2 \) 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 \( k \in \mathbb{Z}_+ \)
**Equivalence Results**

**Theorem 1** All the above six classes of discrete-time hybrid models are equivalent (possibly under some additional assumptions, such as boundedness of input and state variables)

(Heemels, De Schutter, Bemporad, Automatica, 2001)

(Torrisi, Bemporad, IEEE CST, 2003)

(Bemporad and Morari, Automatica, 1999)

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

Theoretical properties and analysis/synthesis tools can be transferred from one class to another

---

**MLD and PWA Systems**

**Theorem** MLD systems and PWA systems are equivalent

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

- MLD:
  \[ x(t+1) = Ax(t) + Bu(t) + B_2\delta(t) + B_3z(t) \]
  \[ y(t) = Cx(t) + Du(t) + D_2\delta(t) + D_3z(t) \]
  
  \[ E_2\delta(t) + E_3z(t) \leq E_1u(t) + E_4x(t) + E_5 \]

- By well-posedness hypothesis on \( z(t), \delta(t) \) + linearity of MLD constraints
  \[ z = K_4x + K_1u + K_3 \quad \forall (x, u) : F(x, u) = \delta \]

- **PWA form**
  \[ x(t+1) = Ax(t) + Bu(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \]
  \[ y(t) = Cx(t) + Du(t) + D_1\delta(t) + D_2z(t) \]
  
  \[ E_1\delta(t) + E_2z(t) \leq E_1u(t) + E_4x(t) + E_5 \]

- Efficient algorithms are available for converting MLD models into PWA models avoiding such an enumeration:

---

**Efficient MLD to PWA Conversion**

- 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 \( \delta \) variables
- Most of such combinations lead to empty regions
MLD to PWA Conversion Algorithm (Bemporad, 2002)

**GOAL**: split a given set $B$ of states+inputs into polyhedral cells and find the affine dynamics in each cell, therefore determining the PWA system which is equivalent to the given MLD system

Note: the cells $\Omega_i$ are embedded in $\mathbb{R}^{n+m}$ by treating integer states and integer inputs as continuous vars:

$x_i, u_i \in [0,1] \rightarrow x_i, u_i \in [-1/2,1/2) \cup [1/2,3/2]$

**MLD to PWA Conversion**

Now fix $\delta = \delta_1$. To find the polyhedral cell $\Omega_i$ and dynamics $(A_i, B_i, f_i)$:

1. From MLD constraints, compute
   
   $$z(k) = K_u z_e(k) + K_i u_i(k) + K_u$$
   $$\forall z_e(k), u(k) : \begin{bmatrix} x_e(k) \\ u(k) \\ F(z(k), u(k)) \end{bmatrix} = \begin{bmatrix} x_e(0) \\ u(0) \\ \delta \end{bmatrix}$$

2. Substitute $z(k)$ in the MLD dynamics
   
   $$x_e(k+1) = (A_e + B_e z_e(k)) x_e(k) + (B_e u_e + B_i z_e(k)) u_i(k) + A_e x_e(k) + B_i u_i(k) + A_e z_e(k) + B_i u_i(k)$$
   $$u_e(k+1) = \text{similiar}$$
   $$y(k) = (C_e + D_e z_e(k)) x_e(k) + (D_e u_e + D_i z_e(k)) u_i(k) + D_e x_e(k) + D_i u_i(k) + C_e x_e(k) + D_i u_i(k)$$

3. Find polyhedral cell $\Omega_i$

$$\Omega_i = \{ z \in \mathbb{R}^n : (E_0 K_u - E_i K_u - E_0 u_i \leq (E_i x_e(k) - E_0 x_e(k) - E_0 u_i) \leq (E_0 x_e(k) - E_0 x_e(k) - E_0 u_i) \times \{ x_e(0) \times \{ u_i \} \}$$

**MLD to PWA Conversion - Start**

- Let $(x^*, u^*)$ be a given point in $B$
  (e.g.: $(x^*, u^*)$ is the Chebychev center of $B$, computable via LP)

- Problem: $(x^*, u^*)$ may not be 0/1 valued

- Find $(x_i, u_i)$ which is closest to $(x^*, u^*)$, is integer feasible, and satisfies the MLD constraints:

$$\arg \min_{x, u, \delta} \| (x - x^*) \|_{\infty}$$
subject to

$$E_{z} z + E_{x} x \leq E_{b} b + E_{d} d + E_{e}$$
$$\| u \|_{1} \in B$$
$$x_e \in [0,1]^{n}$, $u_i \in \{0,1\}^{m_i}$
$$\delta \in [0,1]^{r}, z \in \mathbb{R}^{c}$

**MLD to PWA Conversion - Partition**

Now partition the rest of the space $B \setminus \Omega_i$

$$\Omega_i = \{ z \in \mathbb{R}^n : A \{ z \} \leq B \}$$

$$R_i = \{ z \in \mathbb{R}^n : A' \{ z \} > B', A' \{ z \} > B', \forall j < i \}$$

**Theorem**: $\{ \Omega_i, R_3, \ldots, R_N \}$ is a partition of $B$

Proceed iteratively: for each region $R_i$, repeat the whole procedure with $B \setminus R_i$.

Note: similar to multiparametric Quadratic Programming algorithm (Bemporad et al., 2002)
MLD to PWA Conversion - Union

Regions where the affine dynamics is the same can be joined (when their union is convex).

(Bemporad, Fukuda, Torrisi, 2001)

Example: Heat and Cool

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid HYSDEL Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Example: Heat and Cool

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Example: Heat and Cool

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)

Hybrid MLD Model

Rules of the game:
- #1 turns the heater (air conditioning) 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 the air conditioning on, unless #2 is cold.
- Otherwise, heater and air conditioning are off.

\[
\begin{align*}
T_1 &= -\alpha_1 (T_1 - T_{amb}) + k_1 (u_{hot} - u_{cold}) \\
T_2 &= -\alpha_2 (T_2 - T_{amb}) + k_2 (u_{hot} - u_{cold})
\end{align*}
\]

(body temperature dynamics of #1)

(body temperature dynamics of #2)
Hybrid PWA Model

• PWA model

\[ x(k+1) = A_i(k)x(k) + B_i(k)u(k) + f_i(k) \]
\[ y(k) = C_i(k)x(k) + D_i(k)u(k) + g_i(k) \]
\[ i(k) \quad \text{s.t.} \quad H_i(k)x(k) + J_i(k)u(k) \leq K_i(k) \]

• 2 continuous states:
  (temperatures \( T_1, T_2 \))

• 1 continuous input:
  (room temperature \( T_{\text{amb}} \))

• 5 polyhedral regions
  (partition does not depend on input)

Why are we interested in getting MLD and PWA models?

Hybrid Models

Each model is most advantageous depending on task:

<table>
<thead>
<tr>
<th>Task</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modeling</td>
<td>DHA</td>
</tr>
<tr>
<td>Simulation</td>
<td>DHA</td>
</tr>
<tr>
<td>Control</td>
<td>MLD,PWA,MMPS</td>
</tr>
<tr>
<td>Stability</td>
<td>PWA</td>
</tr>
<tr>
<td>Verification</td>
<td>PWA</td>
</tr>
<tr>
<td>Identification</td>
<td>PWA</td>
</tr>
<tr>
<td>Fault Detection</td>
<td>MLD</td>
</tr>
<tr>
<td>Estimation</td>
<td>MLD</td>
</tr>
</tbody>
</table>

Major Advantage of MLD Framework

Many problems of analysis:
- Stability
- Safety
- Controllability
- Observability

Many problems of synthesis:
- Controller design
- Filter design / Fault detection & state estimation

These problems can be expressed as (mixed integer) mathematical programming problems for which many algorithms and software tools exist.

(However, all these problems are NP-hard!)