MODEL PREDICTIVE CONTROL

HYBRID MPC

Alberto Bemporad

imt.lu/ab
Basic concepts of model predictive control (MPC) and linear MPC

Linear time-varying and nonlinear MPC

Quadratic programming (QP) and explicit MPC

- Hybrid MPC
- Stochastic MPC
- Learning-based MPC

Course page:
http://cse.lab.imtlucca.it/~bemporad/mpc_course.html
Use a hybrid dynamical model of the process to predict its future evolution and choose the “best” control action.
- Finite-horizon optimal control problem (regulation)

\[
\begin{align*}
\min & \quad \sum_{k=0}^{N-1} y_k^T Q y_k + u_k^T R u_k \\
\text{s.t.} & \quad x_{k+1} = A x_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\
& \quad y_k = C x_k + D_1 u_k + D_2 \delta_k + D_3 z_k + D_5 \\
& \quad E_2 \delta_k + E_3 z_k \leq E_4 x_k + E_1 u_k + E_5 \\
& \quad x_0 = x(t)
\end{align*}
\]

\[Q = Q' \succ 0, \quad R = R' \succ 0\]

- Treat \( u_k, \delta_k, z_k \) as free decision variables, \( k = 0, \ldots, N - 1 \)

- Predictions can be constructed exactly as in the linear case

\[
x_k = A^k x_0 + \sum_{j=0}^{k-1} A^j (B_1 u_{k-1-j} + B_2 \delta_{k-1-j} + B_3 z_{k-1-j} + B_5)
\]
After substituting $x_k, y_k$ the resulting optimization problem becomes the following Mixed-Integer Quadratic Programming (MIQP) problem

$$\min_\xi \quad \frac{1}{2} \xi' H \xi + x'(t) F' \xi + \frac{1}{2} x'(t) Y x(t)$$

$$\text{s.t. } G \xi \leq W + S x(t)$$

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

$$u_k \in \mathbb{R}^{m_c} \times \{0, 1\}^{m_b}$$
$$\delta_k \in \{0, 1\}^{r_b}$$
$$z_k \in \mathbb{R}^{r_c}$$

$$\xi \in \mathbb{R}^{N(m_c + r_c)} \times \{0, 1\}^{N(m_b + r_b)}$$
• Consider the more general set-point tracking problem

\[
\min_{\xi} \sum_{k=0}^{N-1} \left( \|y_k - r\|^2_Q + \|u_k - u_r\|^2_R + \sigma \left( \|x_k - x_r\|^2 + \|\delta_k - \delta_r\|^2 + \|z_k - z_r\|^2 \right) \right)
\]

s.t. MLD model equations

\[
x_0 = x(t) \\
x_N = x_r
\]

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

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

\[
x_r = Ax_r + B_1 u_r + B_2 \delta_r + B_3 z_r + B_5 \\
r = Cx_r + D_1 u_r + D_2 \delta_r + D_3 z_r + D_5 \\
E_2 \delta_r + E_3 z_r \leq E_4 x_r + E_1 u_r + E_5
\]
• **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

\[
\begin{align*}
\lim_{t \to \infty} y(t) &= r \\
\lim_{t \to \infty} x(t) &= x_r \\
\lim_{t \to \infty} \delta(t) &= \delta_r \\
\lim_{t \to \infty} z(t) &= z_r
\end{align*}
\]

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)
CONVERGENCE PROOF

- **Main idea:** Use the value function $V^*(x(t))$ as a Lyapunov function
- Let $\xi_t = [u^t_0, \ldots, u^t_{N-1}, \delta^t_0, \ldots, \delta^t_{N-1}, z^t_0, \ldots, z^t_{N-1}]$ be the optimal sequence $\bar{\xi}$
- By construction $\bar{\xi} = [u^1_t, \ldots, u^t_{N-1}, u_r, \delta^t_0, \ldots, \delta^t_{N-1}, \delta_r, z^t_0, \ldots, z^t_{N-1}, z_r]$ is feasible, as it satisfies all MLD constraints + terminal constraint $x_N = x_r$
- The cost of $\bar{\xi}$ is $V^*(x(t)) - \|y(t) - r\|_Q^2 - \|u(t) - u_r\|_R^2$
  $$-\sigma (\|\delta(t) - \delta_r\|_2^2 + \|z(t) - z_r\|_2^2 + \|x(t) - x_r\|_2^2) \geq V^*(x(t+1))$$
- $V^*(x(t))$ is monotonically decreasing and $\geq 0$, so $\exists \lim_{t \to \infty} V^*(x(t)) \in \mathbb{R}$
- Hence $\|y(t) - r\|_Q^2, \|u(t) - u_r\|_R^2, \|\delta(t) - \delta_r\|_2^2, \|z(t) - z_r\|_2^2, \|x(t) - x_r\|_2^2 \to 0$
- Since $R, Q \succ 0$, $\lim_{t \to \infty} y(t) = r$ and all other variables converge.

Global optimum is not needed to prove convergence!
MILP FORMULATION OF HYBRID MPC

(Bemporad, Borrelli, Morari, 2000)

- Finite-horizon optimal control problem using infinity norms

\[
\begin{align*}
\min_{\xi} & \quad \sum_{k=0}^{N-1} \| Q y_k \|_\infty + \| R u_k \|_\infty \\
\text{s.t.} & \quad x_{k+1} = A x_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\
& \quad y_k = C x_k + D_1 u_k + D_2 \delta_k + D_3 z_k + D_5 \\
& \quad E_2 \delta_k + E_3 z_k \leq E_4 x_k + E_1 u_k + E_5 \\
& \quad x_0 = x(t)
\end{align*}
\]

- Introduce additional variables \( \epsilon^y_k, \epsilon^u_k, k = 0, \ldots, N - 1 \)

\[
\begin{align*}
\epsilon^y_k & \geq \| Q y_k \|_\infty \\
\epsilon^u_k & \geq \| R u_k \|_\infty
\end{align*}
\Rightarrow
\[
\begin{align*}
\epsilon^y_k & \geq \pm Q^i y_k \\
\epsilon^u_k & \geq \pm R^i u_k
\end{align*}
\]
After substituting $x_k, y_k$ the resulting optimization problem becomes the following \textbf{Mixed-Integer Linear Programming (MILP)} problem

\[
\begin{align*}
\min_{\xi} & \quad \sum_{k=0}^{N-1} \epsilon_k^y + \epsilon_k^u \\
\text{s.t.} & \quad G\xi \leq W + Sx(t)
\end{align*}
\]

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

$u_k \in \mathbb{R}^{m_c} \times \{0, 1\}^{m_b}$

$\delta_k \in \{0, 1\}^{r_b}$

$z_k \in \mathbb{R}^{r_c}$

$\epsilon_k^y, \epsilon_k^u \in \mathbb{R}$

Same approach applies to any \textbf{convex piecewise affine} stage cost
• **PWA system:**

\[
\begin{align*}
    x(t+1) &= 0.8 \begin{bmatrix} \cos \alpha(t) & -\sin \alpha(t) \\ \sin \alpha(t) & \cos \alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
y(t) &= \begin{bmatrix} 0 & 1 \end{bmatrix} x(t) \\
\alpha(t) &= \begin{cases} \frac{\pi}{3} & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \geq 0 \\ -\frac{\pi}{3} & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) < 0 \end{cases}
\end{align*}
\]

• **Open-loop simulation:**

`go to demo demos/hybrid/bm99sim.m`
/* 2x2 PWA system  - Example from the paper
(C) 2003 by A. Bemporad, 2003 */

SYSTEM pwa { 

INTERFACE { 
  STATE { REAL x1 [-10,10];
           REAL x2 [-10,10];}
  INPUT { REAL u [-1.1,1.1];}
  OUTPUT{ REAL y;}

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

IMPLEMENTATION { 
  AUX { REAL z1,z2;
       BOOL sign; }

  AD { sign = x1<=0; }

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

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

  OUTPUT { y = x2; }
  }
  
}
• Closed-loop MPC results:

\[
\begin{align*}
\min & \quad \sum_{k=1}^{2} |y_k - r(t)| \\
\text{s.t.} & \quad -1 \leq u_k \leq 1, \ i = 0, 1
\end{align*}
\]

• Average CPU time to solve MILP: \(\approx 1\ \text{ms/step}\)

(Macbook Pro 3GHz Intel Core i7 using GLPK)
>> 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.

>> [XX,UU,DD,ZZ,TT]=sim(C,S,r,x0,
• Average CPU time to solve MILP: $\approx 1 \text{ ms/step}$

(Macbook Pro 3GHz Intel Core i7 using GLPK)
• Binary constraints make Mixed-Integer Programming (MIP) a hard problem (\(NP\)-complete)

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

(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
  - 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
We want to solve the following MIQP

\[
\begin{align*}
\min & \quad V(z) \triangleq \frac{1}{2} z' Q z + c' z \\
\text{st.} & \quad A z \leq b \\
& \quad z_i \in \{0, 1\}, \forall i \in I
\end{align*}
\]

\[z \in \mathbb{R}^n\]
\[Q = Q' \succeq 0\]
\[I \subseteq \{1, \ldots, n\}\]

Branch & Bound (B&B) is the simplest (and most popular) approach to solve the problem to optimality

Key idea:

- for each binary variable \(z_i, i \in I\), either set \(z_i = 0\), or \(z_i = 1\), or \(z_i \in [0, 1]\)
- solve the corresponding **QP relaxation** of the MIQP problem
- use QP result to decide the next combination of fixed/relaxed variables
Branch & Bound Method for MIQP

QP relaxation

\[
\begin{align*}
\text{min} & \quad \frac{1}{2} z'Qz + c'z \\
\text{s.t.} & \quad Az \leq b \\
& \quad 0 \leq z_i \leq 1, \forall i \in I
\end{align*}
\]

\(z_i \in \{0, 1\}, \forall i \in I\)

MIQP solution found (lucky case)

QP infeasible?

yes

no

QP relaxation

MIQP infeasible

MIQP solution found (lucky case)

integer feasible?

yes

no

start branching ...

QP infeasible?

yes

no

QP0

Model Predictive Control" - © A. Bemporad. All rights reserved.
**Branching rule:** pick the index $i$ such that $z_i$ is closest to $\frac{1}{2}$ (max fractional part)  
(Breu, Burdet, 1974)

- Solve two new QP relaxations

\[
\begin{align*}
\min_{z} & \quad \frac{1}{2}z'Qz + c'z \\
\text{s.t.} & \quad Az \leq b \\
& \quad 0 \leq z_i \leq 1, \forall i \in I \\
& \quad z_i = 0 \quad \text{QP}_1 \\
& \quad 0 \leq z_j \leq 1, \forall j \in I, j \neq i \\
\end{align*}
\]

\[
\begin{align*}
\min_{z} & \quad \frac{1}{2}z'Qz + c'z \\
\text{s.t.} & \quad Az \leq b \\
& \quad z_i = 1 \\
& \quad 0 \leq z_j \leq 1, \forall j \in I, j \neq i \\
\end{align*}
\]

- Possibly exploit **warm starting** from QP$_0$ when solving new relaxations QP$_1$ and QP$_2$
Branch & Bound Method for MIQP

**QP 0**

- **QP infeasible?**
  - **no** → update upper bound $V_0 \geq V^*$ on MIQP solution
  - **yes** → stop branching on subtree

- **QP feasible?**
  - **integer?**
    - **yes** → keep branching...
    - **no** → update upper bound $V_0 \geq V^*$ on MIQP solution
The cost $V_0$ of the best integer-feasible solution found so far gives an upper bound $V_0 \geq V^*$ on MIQP solution.

Branch & Bound Method for MIQP

- $QP_0$
- $QP_1$
- $QP_2$

The optimum $\geq V_0$?

- Yes: stop branching (adding further equality constraints can only increase the optimal cost)
- No: keep branching...

"Model Predictive Control" - © A. Bemporad. All rights reserved.
While solving the QP relaxation, if the dual cost is available it gives a lower bound to the solution of the relaxed problem.

The QP solver can be stopped whenever the dual cost $\geq V_0$.

This may save a lot of computations.

When no further branching is possible, either the MIQP problem is recognized infeasible or the optimal solution $z^*$ has been found.
SOLVING MIQP VIA NNLS

(Bemporad, 2015)

• B&B method + QP solver based on **nonnegative least squares** applied to solving the MIQP

\[
\min_z \quad V(z) \triangleq \frac{1}{2} z'Qz + c'z \\
\text{s.t.} \quad \ell \leq Az \leq u \\
Gz = g \\
\bar{A}_i z \in \{\bar{\ell}_i, \bar{u}_i\}, \quad i = 1, \ldots, q
\]

\[Q = Q' \succ 0\]

• Binary constraints on $z$ are a special case: $\bar{\ell}_i = 0, \bar{u}_i = 1, \quad \bar{A}_i = [0 \ldots 0 1 0 \ldots 0]$

• Warm starting from parent node exploited when solving new QP relaxation

• QP solver interrupted when dual cost larger than best known upper-bound
### Worst-case CPU time (ms) on random MIQP problems:

<table>
<thead>
<tr>
<th>n</th>
<th>m</th>
<th>q</th>
<th>NNLS(_{LDL})</th>
<th>NNLS(_{QR})</th>
<th>GUROBI</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>2</td>
<td>2.3</td>
<td>1.2</td>
<td>1.4</td>
<td>8.0</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>2</td>
<td>5.7</td>
<td>3.3</td>
<td>6.1</td>
<td>31.4</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>5</td>
<td>4.2</td>
<td>6.1</td>
<td>14.1</td>
<td>30.1</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>10</td>
<td>68.8</td>
<td>104.4</td>
<td>114.6</td>
<td>294.1</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>2</td>
<td>4.6</td>
<td>10.2</td>
<td>37.2</td>
<td>69.2</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>15</td>
<td>137.5</td>
<td>365.7</td>
<td>259.8</td>
<td>547.8</td>
</tr>
<tr>
<td>150</td>
<td>100</td>
<td>5</td>
<td>15.6</td>
<td>49.2</td>
<td>157.2</td>
<td>260.1</td>
</tr>
<tr>
<td>150</td>
<td>300</td>
<td>20</td>
<td>1174.4</td>
<td>3970.4</td>
<td>1296.1</td>
<td>2123.9</td>
</tr>
</tbody>
</table>

Compiled Embedded MATLAB code (QP solver) + MATLAB code (B&B)
CPU results measured on Macbook Pro 3GHz Intel Core i7

**NNLS-\(LDL\)** = recursive LDL’ factorization used to solve least-square problems in QP solver

**NNLS-\(QR\)** = recursive QR factorization used instead (numerically more robust)
SOLVING MIQP VIA NNLS

- **Worst-case** CPU time (ms) on random purely binary QP problems:

<table>
<thead>
<tr>
<th>n</th>
<th>m</th>
<th>q</th>
<th>NNLS$_{LDL}$</th>
<th>NNLS$_{QR}$</th>
<th>GUROBI</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>2</td>
<td>5.1</td>
<td>4.0</td>
<td>0.7</td>
<td>8.4</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>4</td>
<td>8.9</td>
<td>4.3</td>
<td>4.5</td>
<td>16.7</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>8</td>
<td>19.2</td>
<td>18.0</td>
<td>37.1</td>
<td>14.7</td>
</tr>
<tr>
<td>12</td>
<td>60</td>
<td>12</td>
<td>59.7</td>
<td>57.8</td>
<td>82.3</td>
<td>47.9</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>20</td>
<td>483.5</td>
<td>457.7</td>
<td>566.8</td>
<td>99.6</td>
</tr>
<tr>
<td>25</td>
<td>250</td>
<td>25</td>
<td>110.4</td>
<td>93.3</td>
<td>1054.4</td>
<td>169.4</td>
</tr>
<tr>
<td>30</td>
<td>150</td>
<td>30</td>
<td>1645.4</td>
<td>1415.8</td>
<td>2156.2</td>
<td>184.5</td>
</tr>
</tbody>
</table>

- **Worst-case** CPU time (ms) on a hybrid MPC problem

  \[ N = \text{prediction horizon} \]

  MIQP regularized to make

  \[ Q \text{ strictly } \succ 0 \]

  (solution difference is negligible)

<table>
<thead>
<tr>
<th>N</th>
<th>NNLS$_{LDL}$</th>
<th>NNLS$_{QR}$</th>
<th>GUROBI</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2.2</td>
<td>2.3</td>
<td>1.2</td>
<td>3.0</td>
</tr>
<tr>
<td>3</td>
<td>3.4</td>
<td>3.9</td>
<td>2.0</td>
<td>6.5</td>
</tr>
<tr>
<td>4</td>
<td>5.0</td>
<td>6.5</td>
<td>2.6</td>
<td>8.1</td>
</tr>
<tr>
<td>5</td>
<td>7.6</td>
<td>9.8</td>
<td>3.7</td>
<td>9.0</td>
</tr>
<tr>
<td>6</td>
<td>12.3</td>
<td>17.7</td>
<td>4.3</td>
<td>11.0</td>
</tr>
<tr>
<td>7</td>
<td>20.5</td>
<td>30.5</td>
<td>5.8</td>
<td>13.1</td>
</tr>
<tr>
<td>8</td>
<td>28.9</td>
<td>47.1</td>
<td>7.3</td>
<td>17.3</td>
</tr>
<tr>
<td>9</td>
<td>38.8</td>
<td>62.5</td>
<td>9.5</td>
<td>18.9</td>
</tr>
<tr>
<td>10</td>
<td>55.4</td>
<td>98.2</td>
<td>10.9</td>
<td>22.4</td>
</tr>
</tbody>
</table>
- **Robustified approach**: use NNLS + proximal-point iterations to solve QP relaxations (Bemporad, 2018)

\[
 z_{k+1} = \arg \min_z \quad \frac{1}{2} z'Qz + c'z + \frac{\epsilon}{2} \|z - z_k\|^2_2 \\
\text{s.t.} \quad \ell \leq Az \leq u \\
Gz = g
\]

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

<table>
<thead>
<tr>
<th>For ( N = 10 ):</th>
<th>( N )</th>
<th>prox-NNLS</th>
<th>prox-NNLS*</th>
<th>GUROBI</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>avg</td>
<td>max</td>
<td>avg</td>
<td>max</td>
</tr>
<tr>
<td>30 real vars</td>
<td>2</td>
<td>2.0</td>
<td>2.6</td>
<td>1.6</td>
<td>2.0</td>
</tr>
<tr>
<td>10 binary vars</td>
<td>4</td>
<td>5.3</td>
<td>8.8</td>
<td>3.1</td>
<td>3.9</td>
</tr>
<tr>
<td>160 inequalities</td>
<td>8</td>
<td>29.7</td>
<td>71.0</td>
<td>8.1</td>
<td>13.2</td>
</tr>
<tr>
<td>prox-NNLS* = warm</td>
<td>10</td>
<td>76.2</td>
<td>146.1</td>
<td>11.1</td>
<td>17.6</td>
</tr>
<tr>
<td>start of binary vars exploited</td>
<td>12</td>
<td>155.8</td>
<td>410.8</td>
<td>14.9</td>
<td>31.2</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>484.2</td>
<td>1242.3</td>
<td>25.9</td>
<td>109.8</td>
</tr>
</tbody>
</table>

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 $Q = Q' \succ 0$

\[
\begin{align*}
\min_z \quad & V(z) \triangleq \frac{1}{2} z' Q z + c' z \\
\text{s.t.} \quad & \ell \leq A z \leq u \\
& G z = g \\
& A_i z \in \{\bar{l}_i, \bar{u}_i\}, \ i = 1, \ldots, p
\end{align*}
\]

\[
\begin{align*}
w^k & = y^k + \beta_k (y^k - y^{k-1}) \\
z^k & = -K w^k - J x \\
s^k & = \ldots \\
y_{i}^{k+1} & = \max \{w_{i}^{k} + s_{i}^{k}, 0\}, \ i \in I_{ineq}
\end{align*}
\]

- Use B&B and \textbf{fast gradient projection} to solve dual of QP relaxation

<table>
<thead>
<tr>
<th>Constraint is relaxed</th>
<th>$A_i z \leq \bar{u}_i$</th>
<th>$y_{i}^{k+1} = \max {w_{i}^{k} + s_{i}^{k}, 0}$</th>
<th>$(y_i \geq 0)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint is fixed</td>
<td>$A_i z = \bar{u}_i$</td>
<td>$y_{i}^{k+1} = w_{i}^{k} + s_{i}^{k}$</td>
<td>$(y_i \leq 0)$</td>
</tr>
<tr>
<td>Constraint is ignored</td>
<td>$A_i z = \bar{l}_i$</td>
<td>$y_{i}^{k+1} = 0$</td>
<td>$(y_i = 0)$</td>
</tr>
</tbody>
</table>
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):

<table>
<thead>
<tr>
<th>n</th>
<th>m</th>
<th>p</th>
<th>q</th>
<th>miqpGPAD</th>
<th>GUROBI</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>15.6</td>
<td>6.56</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>5</td>
<td>3</td>
<td>3.44</td>
<td>8.74</td>
</tr>
<tr>
<td>50</td>
<td>150</td>
<td>10</td>
<td>5</td>
<td>63.22</td>
<td>46.25</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>2</td>
<td>5</td>
<td>6.22</td>
<td>26.24</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>15</td>
<td>5</td>
<td>164.06</td>
<td>188.42</td>
</tr>
<tr>
<td>150</td>
<td>100</td>
<td>5</td>
<td>5</td>
<td>31.26</td>
<td>88.13</td>
</tr>
<tr>
<td>150</td>
<td>200</td>
<td>20</td>
<td>5</td>
<td>258.80</td>
<td>274.06</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>15</td>
<td>6</td>
<td>35.08</td>
<td>144.38</td>
</tr>
</tbody>
</table>

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

n = # variables
m = # inequality constraints
p = # binary constraints
q = # equality constraints
**MIQP AND ADMM**

- **B&B + ADMM**: solve QP relaxations via ADMM
  \[ \min \frac{1}{2} x'Qx + c'x \]
  \[ \text{s.t. } \ell \leq Ax \leq u \]
  \[ A_i x \in \{ \ell_i, u_i \}, i \in I \]
  (Stellato, Naik, Bemporad, Goulart, Boyd, 2018)

- **Simpler heuristic** approach: only perform one set of ADMM iterations
  (Takapoui, Moehle, Boyd, Bemporad, 2017)

  \[
  x^{k+1} = -(Q + \rho A^T A)^{-1} (\rho A^T (y^k - z^k) + c) \\
  z^{k+1} = \min \{ \max \{ Ax^{k+1} + y^k, \ell \}, u \} \\
  z^+_{i} = \begin{cases} 
  \ell_i & \text{if } z^+_{i} < \frac{\ell_i + u_i}{2} \\
  u_i & \text{if } z^+_{i} \geq \frac{\ell_i + u_i}{2}, i \in I 
  \end{cases} \\
  y^{k+1} = y^k + Ax^{k+1} - z^{k+1}
  \]

- Iterations converge to a (local) solution

- Similar heuristic idea also applicable to fast gradient methods
  (Naik, Bemporad, 2017)
HEURISTIC ADMM METHOD FOR (SUBOPTIMAL) MIQP

(Takapoui, Moehle, Boyd, Bemporad, 2017)

- **Example**: parallel hybrid electric vehicle control problem

engine power

 electrical power

energy stored in battery

engine on/off

optimal solution

ADMM solution
**Example:** power converter control problem

\[
\begin{align*}
\text{minimize} & \quad \sum_{t=0}^{T} (v_{2,t} - v_{\text{des}})^2 + \lambda |u_{t} - u_{t-1}| \\
\text{subject to} & \quad \xi_{t+1} = G\xi_{t} + Hu_{t} \\
& \quad \xi_{0} = \xi_{T} \\
& \quad u_{0} = u_{T} \\
& \quad u_{t} \in \{-1, 0, 1\}
\end{align*}
\]

input voltage sign \(u_t\)

output voltage \(v_2\)

optimal solution

ADMM solution

"Model Predictive Control" - © A. Bemporad. All rights reserved.
A SIMPLE EXAMPLE IN SUPPLY CHAIN MANAGEMENT

manufacturer A

manufacturer B

manufacturer C

inventory 1

inventory 2

retailer 1

U_{A11}(k) → u_{11}(k)

U_{B11}(k) → u_{12}(k)

U_{C11}(k) → x_{11}(k), x_{12}(k)

U_{B12}(k) → u_{22}(k)

U_{C12}(k) → u_{21}(k)

U_{B22}(k) → x_{21}(k), x_{22}(k)

U_{C22}(k) → y_{1}(k)

U_{C22}(k) → y_{2}(k)

go to demo demos/hybrid/supply_chain.m
• **Continuous states:**

\[ x_{ij}(k) = \text{amount of } j \text{ hold in inventory } i \text{ at time } k \ (i = 1, 2, j = 1, 2) \]

• **Continuous outputs:**

\[ y_j(k) = \text{amount of } j \text{ sold at time } k \ (j = 1, 2) \]

• **Continuous inputs:**

\[ u_{ij}(k) = \text{amount of } j \text{ taken from inventory } i \text{ at time } k \ (i = 1, 2, j = 1, 2) \]

• **Binary inputs:**

\[ U_{Xij}(k) = 1 \text{ if manufacturer } X \text{ produces and send } j \text{ to inventory } i \text{ at time } k \]
Supply Chain Management - Constraints

- Max capacity of inventory $i$:
  \[ 0 \leq \sum_{j=1}^{2} x_{ij} \leq x_{Mi} \]

- Max transportation from inventories:
  \[ 0 \leq u_{ij} (k) \leq u_{M} \]

- A product can only be sent to one inventory:
  \[ U_{A11} (k) \text{ and } U_{A21} (k) \text{ cannot be both } = 1 \]
  \[ U_{B11} (k) \text{ and } U_{B21} (k) \text{ cannot be both } = 1 \]
  \[ U_{B12} (k) \text{ and } U_{B22} (k) \text{ cannot be both } = 1 \]
  \[ U_{C12} (k) \text{ and } U_{C22} (k) \text{ cannot be both } = 1 \]

- A manufacturer can only produce one type of product at one time:
  \[ [U_{B11} (k) \text{ or } U_{B21} (k) = 1], [U_{B12} (k) \text{ or } U_{B22} (k) = 1] \text{ cannot be both true} \]
Supply Chain Management - Dynamics

- Let $P_{A1}, P_{B1}, P_{B2}, P_{C2} =$ amount of product of type 1 (2) produced by $A (B, C)$ in one time interval.

- Level of inventories

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

- Retailer: all items requested from inventories are sold

  \[
  \begin{align*}
  y_1 &= u_{11} + u_{21} \\
  y_2 &= u_{12} + u_{22}
  \end{align*}
  \]
SYSTEM supply_chain{
  INTERFACE {
    STATE { REAL x11 [0,10];
             REAL x12 [0,10];
             REAL x21 [0,10];
             REAL x22 [0,10]; }
    INPUT { REAL u11 [0,10];
             REAL u12 [0,10];
             REAL u21 [0,10];
             REAL u22 [0,10];
             BOOL UA11,UA21,UB11,UB12,UB21,UB22,UC12,UC22; }
    OUTPUT { REAL y1,y2; }
  }
  PARAMETER { REAL PA1,PB1,PB2,PC2,xM1,xM2; }
  IMPLEMENTATION {
    AUX { REAL zA11, zB11, zB12, zC12, zA21, zB21, zB22, zC22; }
    DA {
      zA11 = (IF UA11 THEN PA1 ELSE 0);
      zB11 = (IF UB11 THEN PB1 ELSE 0);
      zB12 = (IF UB12 THEN PB2 ELSE 0);
      zC12 = (IF UC12 THEN PC2 ELSE 0);
      zA21 = (IF UA21 THEN PA1 ELSE 0);
      zB21 = (IF UB21 THEN PB1 ELSE 0);
      zB22 = (IF UB22 THEN PB2 ELSE 0);
      zC22 = (IF UC22 THEN PC2 ELSE 0); }
    CONTINUOUS { x11 = x11 + zA11 + zB11 - u11;
                  x12 = x12 + zB12 + zC12 - u12;
                  x21 = x21 + zA21 + zB21 - u21;
                  x22 = x22 + zB22 + zC22 - u22; }
    OUTPUT { y1 = u11 + u21;
              y2 = u12 + u22; }
    MUST { ~(UA11 & UA21);
            ~(UC12 & UC22);
            ~(UB11 & UB21);
            ~(UB11 & UB22);
            ~(UB12 & UB22);
            x11+x12 <= xM1;
            x11+x12 >=0;
            x21+x22 <= xM2;
            x21+x22 >=0; }
  }
}
• Meet customer demand as much as possible:

\[ y_1 \approx r_1, \quad y_2 \approx r_2 \]

• Minimize transportation costs

• Fulfill all constraints
\[
\min \sum_{k=0}^{N-1} \left\{ \begin{array}{l}
\text{penalty on demand tracking error} \\
10(|y_{1,k} - r_1(t)| + |y_{2,k} - r_2(t)|) + \\
\text{shipping cost from inv. 1 to market} \\
4(|u_{11,k}| + |u_{12,k}|) + \\
\text{shipping cost from inv. 2 to market} \\
2(|u_{21,k}| + |u_{22,k}|) + \\
\text{cost from A to inventories} \\
|U_{A11,k}| + |U_{A21,k}| + \\
\text{cost from B to inventories} \\
4(|U_{B11,k}| + |U_{B12,k}| + U_{B21,k} + |U_{B22,k}|) + \\
\text{cost from C to inventories} \\
10(|U_{C12,k}| + |U_{C22,k}|)
\end{array} \right. \]
>> refs.y=[1 2];  % weights output2 #1, #2
>> Q.y=diag([10 10]);  % output weights
...  % infinity norms
>> Q.norm=Inf;
>> N=2;  % optimization horizon
>> limits.umin=umin;  % constraints
>> limits.umax=umax;
>> limits.xmin=xmin;
>> limits.xmax=xmax;  % xij(k)>=0
>> limits.xmax=xmax;  % xij(k)<=xMi (redundant)

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

>> C

Hybrid controller based on MLD model S <supply_chain.hys>

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

44 optimization variable(s) (8 continuous, 12 binary)
176 mixed-integer linear inequalities
sampling time = 1, MILP solver = 'glpk'

Type "struct(C)" for more details.
>>
Supply Chain Management - Simulation Results

% Initial condition

>> x0=[0;0;0;0];

% Reference trajectories

>> r.y=[6+2*sin((0:Tstop-1)'/5)
5+3*cos((0:Tstop-1)'/3)];

% CPU time: ≈ 13 ms/sample (GLPK) or 9 ms (CPLEX) on Macbook Pro 3GHz Intel Core i7
Hybrid MPC of an Inverted Pendulum

- **Goal**: swing the pendulum up

- **Non-convex** input constraint

  \[ u \in [-\tau_{\text{max}}, -\tau_{\text{min}}] \cup \{0\} \cup [\tau_{\text{min}}, \tau_{\text{max}}] \]

- **Nonlinear** dynamical model

  \[ l^2 M \ddot{\theta} = M g l \sin \theta - \beta \dot{\theta} + u \]
• Approximate $\sin(\theta)$ as the piecewise linear function

\[
\sin \theta \approx s \triangleq \begin{cases} 
-\alpha \theta - \gamma & \text{if } \theta \leq -\frac{\pi}{2} \\
\alpha \theta & \text{if } |\theta| \leq \frac{\pi}{2} \\
-\alpha \theta + \gamma & \text{if } \theta \geq \frac{\pi}{2}
\end{cases}
\]

• Get optimal values for $\alpha$ and $\gamma$ by minimizing fit error

\[
\min_{\alpha} \int_0^{\pi/2} (\alpha \theta - \sin(\theta))^2 d\theta = \frac{\theta}{2} - \frac{1}{2} \cos \theta \sin \theta - 2\alpha \sin \theta + \frac{1}{3} \alpha^2 \theta^3 + 2\alpha \theta \cos \theta \bigg|_{0}^{\pi/2} = \frac{1}{24} \pi^3 \alpha^2 - 2\alpha + \frac{\pi}{4}
\]

• Zeroing the derivative with respect to $\alpha$ gives $\alpha = \frac{24}{\pi^3}$

• Requiring $s = 0$ for $\theta = \pi$ gives $\gamma = \frac{24}{\pi^2}$
• Introduce the event variables

\[ [\delta_3 = 1] \iff [\theta \leq -\frac{\pi}{2}] \]

\[ [\delta_4 = 1] \iff [\theta \geq \frac{\pi}{2}] \]

along with the logic constraint

\[ [\delta_4 = 1] \rightarrow [\delta_3 = 0] \]

• Set \( s = \alpha \theta + s_3 + s_4 \) with

\[
s_3 = \begin{cases} 
-2\alpha \theta - \gamma & \text{if } \delta_3 = 1 \\
0 & \text{otherwise}
\end{cases}
\]

\[
s_4 = \begin{cases} 
-2\alpha \theta + \gamma & \text{if } \delta_4 = 1 \\
0 & \text{otherwise}
\end{cases}
\]
• To model the constraint \( u \in [-\tau_{\text{max}}, -\tau_{\text{min}}] \cup \{0\} \cup [\tau_{\text{min}}, \tau_{\text{max}}] \) introduce the auxiliary variable

\[
\tau_A = \begin{cases} 
  u & \text{if } -\tau_{\text{min}} \leq u \leq \tau_{\text{min}} \\
  0 & \text{otherwise}
\end{cases}
\]

and let \( u - \tau_A \) be the torque acting on the pendulum, with

\[ u \in [-\tau_{\text{max}}, \tau_{\text{max}}] \]

• The input \( u \) has no effect on the dynamics for \( u \in [-\tau_{\text{min}}, \tau_{\text{min}}] \). Hence, the solver will not choose values in that range if \( u \) is penalized in the MPC cost.
Inverted Pendulum: Non-Convex Constraint

- Introduce new event variables

\[
\begin{align*}
\delta_1 = 1 & \leftrightarrow [u \leq \tau_{\text{min}}] \\
\delta_2 = 1 & \leftrightarrow [u \geq -\tau_{\text{min}}]
\end{align*}
\]

along with the logic constraint \([\delta_1 = 0] \rightarrow [\delta_2 = 1]\) and set

\[
\tau_A = \begin{cases} 
  u & \text{if } [\delta_1 = 1] \land [\delta_2 = 1] \\
  0 & \text{otherwise}
\end{cases}
\]

so that \(u - \tau_A\) is zero in for \(u \in [-\tau_{\text{min}}, \tau_{\text{min}}]\)
• Set \( x \triangleq \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \), \( y \triangleq \theta \) and transform into linear model

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{g}{l} \alpha & -\frac{\beta}{l^2 M} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{g}{l} & \frac{1}{l^2 M} \end{bmatrix} \begin{bmatrix} s_3 + s_4 \\ u - \tau_A \end{bmatrix}
\]

\[
y = \begin{bmatrix} 1 & 0 \end{bmatrix} x
\]

• Discretize in time with sample time \( T_s = 50 \text{ ms} \)

\[
\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + B \begin{bmatrix} s_3(k) + s_4(k) \\ u(k) - \tau_A(k) \end{bmatrix}
\]

\[
y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k)
\]

\[
A \triangleq e^{T_s A_c}, \quad B \triangleq \int_0^{T_s} e^{t A_c} B_c dt
\]
/* Hybrid model of a pendulum
(C) 2012 by A. Bemporad, April 2012 */

SYSTEM hyb_pendulum {

INTERFACE {
STATE {
    REAL th [-2*pi,2*pi];
    REAL thdot [-20,20];
}
INPUT {
    REAL u [-11,11];
}
OUTPUT {
    REAL y;
}
PARAMETER {
    REAL tau_min,alpha,gamma;
    REAL a11,a12,a21,a22,b11,b12,b21,b22;
}
}

IMPLEMENTATION {

AUX {
    REAL tauA,s3,s4;
    BOOL d1,d2,d3,d4;
}

AD {
    d1 = u<=tau_min;
    d2 = u>=-tau_min;
    d3 = th <= -0.5*pi;
    d4 = th >= 0.5*pi;
}

DA {
    tauA = {IF d1 & d2 THEN u ELSE 0};
    s3 = {IF d3 THEN -2*alpha*th-gamma ELSE 0};
    s4 = {IF d4 THEN -2*alpha*th+gamma ELSE 0};
}

CONTINUOUS {
    th = a11*th+a12*thdot+b11*(s3+s4)+b12*(u-tauA);
    thdot = a21*th+a22*thdot+b21*(s3+s4)+b22*(u-tauA);
}

OUTPUT {
    y = th;
}

MUST {
    d4->~d3;
    ~d1->d2;
}
}

}
Inverted Pendulum: Model Validation

- Open-loop simulation from initial condition $\theta(0) = 0, \dot{\theta}(0) = 0$
- Input torque excitation

$$u(t) = \begin{cases} 
2 \text{Nm} & \text{if } 0 \leq t \leq 10 \text{ s} \\
0 & \text{otherwise}
\end{cases}$$

```
>> u0=2;
>> U=[2*ones(200,1);zeros(200,1)];
>> x0=[0;0];

>> [X,T,D,Z,Y]=sim(S,x0,U);
```
• MPC cost function

\[ \sum_{k=0}^{4} |y_k - r(t)| + |0.01u_k| \]

• MPC constraints \( u \in [-\tau_{\text{max}}, \tau_{\text{max}}] \)

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

```
>> C

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

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

55 optimization variable(s) (30 continuous, 25 binary)
155 mixed-integer linear inequalities
sampling time = 0.05, MILP solver = 'gurobi'

Type "struct(C)" for more details.
>>
```
INVERTED PENDULUM: CLOSED-LOOP RESULTS

- Nominal simulation
  
  \[ \{X, U, D, Z, T, Y\} = \text{sim}(C, S, r, x0, 4); \]

- Nonlinear simulation
  
  CPU time:
  
  - 51 ms per time step (GLPK)
  - 22 ms per time step (CPLEX)
  - 25 ms (GUROBI)
  (Macbook Pro 3GHz Intel Core i7)
EXPLICIT HYBRID MPC
Explicit Hybrid MPC (MLD formulation)

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

subject to \[
\begin{align*}
  x_{k+1} &= Ax_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\
  y_k &= Cx_k + D_1 u_k + D_2 \delta_k + D_3 z_k + D_5 \\
  E_2 \delta_k + E_3 z_k &\leq E_4 x_k + E_1 u_k + E_5 \\
  x_0 &= x(t)
\end{align*}
\]

• **Online optimization**: solve the problem for a given state \(x(t)\) as the MILP

\[
\min_{\xi} \sum_{k=0}^{N-1} \epsilon_k^y + \epsilon_k^u \\
\text{s.t. } G\xi \leq W + S x(t)
\]

• **Offline optimization**: solve the MILP in advance for all states \(x(t)\)

**Multiparametric Mixed-Integer Linear Program (mp-MILP)**

"Model Predictive Control" - © A. Bemporad. All rights reserved.
Consider the mp-MILP

\[
\begin{align*}
\min_{\xi_c, \xi_d} & \quad f'_c \xi_c + f'_d \xi_d \\
\text{s.t.} & \quad G_c \xi_c + G_d \xi_d \leq W + S x \\
\end{align*}
\]

\[
\xi_c \in \mathbb{R}^{n_c} \\
\xi_d \in \{0, 1\}^{n_d} \\
x \in \mathbb{R}^m
\]

A mp-MILP can be solved by alternating MILPs and mp-LPs (Dua, Pistikopoulos, 1999)

The multiparametric solution \(\xi^*(x)\) is PWA (but possibly discontinuous)

The MPC controller is piecewise affine in \(x = x(t)\)

\[
u(x) = \begin{cases} 
F_1 x + g_1 & \text{if } H_1 x \leq K_1 \\
\vdots & \vdots \\
F_M x + g_M & \text{if } H_M x \leq K_M 
\end{cases}
\]

(More generally, the parameter vector \(x\) includes states and reference signals)
• Consider the MPC formulation using a PWA prediction model

\[
\min_{\xi} J(\xi, x(t)) = \sum_{k=0}^{N-1} \| Q y_k \|_{\infty} + \| R u_k \|_{\infty}
\]

subject to

\[
\begin{align*}
x_{k+1} &= A_{i(k)} x_k + B_{i(k)} u_k + f_{i(k)} \\
y_k &= C_{i(k)} x_k + D_{i(k)} u_k + g_{i(k)} \\
i(k) & \text{ such that } H_{i(k)} x_k + W_{i(k)} u_k \leq K_{i(k)} \\
x_0 &= x(t)
\end{align*}
\]

• **Method #1**: The explicit solution can be obtained by using a combination of **dynamic programming (DP)** and **mpLP** (Borrelli, Baotic, Bemporad, Morari, 2005)

• Clearly the explicit hybrid MPC law is again piecewise affine, as PWA systems \(\equiv\) MLD systems
- **Method #2:** (Bemporad, Hybrid Toolbox, 2003)

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

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

3a **Case of \( 1 / \infty \)-norms or convex PWA costs:** Compare value functions and **split regions**

3b **Case of quadratic costs:** the partition may not be fully polyhedral, better **keep overlapping polyhedra** and compare online quadratic cost functions when overlaps are detected

- **Comparison of quadratic costs** can be avoided by lifting the parameter space (Fuchs, Axehill, Morari, 2015)
HYBRID MPC EXAMPLE - EXPLICIT VERSION

- PWA system:

\[
\begin{align*}
  x(t+1) &= 0.8 \begin{bmatrix} \cos \alpha(t) & -\sin \alpha(t) \\ \sin \alpha(t) & \cos \alpha(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\
  y(t) &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} x(t) \\
  \alpha(t) &= \begin{cases} 
    \frac{\pi}{3} & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \geq 0 \\
    -\frac{\pi}{3} & \text{if } \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) < 0
  \end{cases}
\end{align*}
\]

subject to \(-1 \leq u(t) \leq 1\)

- MPC objective:

\[
\min \sum_{k=1}^{2} |y_k - r(t)|
\]

- Open-loop behavior:

"Model Predictive Control" - © A. Bemporad. All rights reserved.
Hybrid MPC example - explicit version

\[ u(x,r) = \begin{cases} 
\left[ 0.6928 -0.4 \ 1 \right] \left[ \begin{array}{c} x \\ r \end{array} \right] & \text{if } \left[ \begin{array}{ccc} 0.6928 & -0.4 & 1 \\ -0.4 & -0.6928 & 0 \\ 0 & -1 & 0 \\ -0.6928 & 0.4 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{array} \right] \left[ \begin{array}{c} x \\ r \end{array} \right] \leq \left[ \begin{array}{c} 1 \\ 10 \\ 1 \\ 1 \\ 1e-006 \end{array} \right] \\
1 & \text{if } \left[ \begin{array}{ccc} 0.6928 & 0.4 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{array} \right] \left[ \begin{array}{c} x \\ r \end{array} \right] \leq \left[ \begin{array}{c} -1 \\ 1 \\ 1 \\ 10 \end{array} \right] \\
-1 & \text{if } \left[ \begin{array}{ccc} -0.4 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0.6928 & 0.4 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{array} \right] \left[ \begin{array}{c} x \\ r \end{array} \right] \leq \left[ \begin{array}{c} 10 \\ 10 \\ 10 \\ 1e-006 \\ 1 \\ 10 \\ 10 \\ 10 \end{array} \right] \\
-1 & \text{if } \left[ \begin{array}{ccc} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0.4 & 0 & 0 \\ -0.6928 & 0.4 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{array} \right] \left[ \begin{array}{c} x \\ r \end{array} \right] \leq \left[ \begin{array}{c} 0 \\ 10 \\ 10 \\ 10 \\ 1 \\ 10 \\ 10 \\ 10 \end{array} \right] \\
\left[ -0.6928 & -0.4 \ 1 \right] \left[ \begin{array}{c} x \\ r \end{array} \right] & \text{if } \left[ \begin{array}{ccc} 0.6928 & -0.4 & 1 \\ 0.4 & -0.6928 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & -1 \\ 0.6928 & 0.4 & -1 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{array} \right] \left[ \begin{array}{c} x \\ r \end{array} \right] \leq \left[ \begin{array}{c} 1 \\ 10 \\ 10 \\ 10 \\ 1 \\ 1 \\ 1 \end{array} \right] \\
\end{cases} 
\]

Go to /demos/hybrid/bm99sim.m

Offline CPU time = 1.51 s (Macbook Pro 3GHz Intel Core i7)

PWA law \equiv MPC law!
• Closed-loop explicit MPC
• MPC problem:

\[
\text{min} \quad 10\|x_N\|_\infty + \sum_{k=0}^{N-1} 10\|x_k\|_\infty + \|u_k\|_\infty
\]

s.t. \[
\begin{align*}
-1 & \leq u_k \leq 1, \ k = 0, \ldots, N - 1 \\
-10 & \leq x_k \leq 10, \ k = 1, \ldots, N
\end{align*}
\]

\[
Q = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \quad R = 1
\]

prediction horizon \( N = 1 \)

prediction horizon \( N = 2 \)

prediction horizon \( N = 3 \)

go to demos/hybrid/bm99benchmark.m
**Explicit Hybrid MPC — Temperature Control**

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

```
E

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

The controller is for hybrid systems (tracking)
This is a state-feedback controller.
Type "struct(E)" for more details.
```

384 numbers to store in memory

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

"Model Predictive Control" - © A. Bemporad. All rights reserved.
explicit hybrid MPC — temperature control

defined in
utils/expcon.h

controller mode

Model Predictive Control © A. Bemporad. All rights reserved.
Alternatives:

1. solve MIP online
2. evaluate a PWA function (explicit solution)

Small problems (short horizon $N = 1, 2$, one or two inputs, 4-6 binary vars): explicit PWA control law is preferable

- CPU time to evaluate the control law is shorter than by MIP
- control code is simpler (no complex solver must be included in the control software!)
- more insight in controller behavior

Medium/large problems (longer horizon, many inputs and binary variables): online MIP is preferable
• **Goal:** estimate the state of a hybrid system from past I/O measurements

• **Moving horizon estimation** based on MLD models solves the problem

  MLD model augmented by
  - state disturbance $\xi \in \mathbb{R}^n$
  - output disturbance $\zeta \in \mathbb{R}^p$

• At each time $t$ get the estimate $\hat{x}(t)$ by solving the **MIQP**

  $\min_{\hat{x}(t-T|t)} \sum_{k=0}^{T} \|\hat{y}(t-k|t) - y(t-k)\|^2_2 + \ldots$

  s.t. constraints on $\hat{x}(t-T+k|t), \hat{y}(t-T+k|t)$

• For **fault detection** also include unknown binary disturbances $\phi \in \{0, 1\}^{n_f}$
- Can only measure tank levels $h_1, h_2$

- The system has two faults:
  - $\phi_1$: leak in tank 1
    between $20 \leq t \leq 60$ s
  - $\phi_2$: valve $V_1$ blocked
    for $t \geq 40$ s

- Add logic constraint
  
  \[ h_1 \leq h_v \rightarrow \phi_2 = 0 \]
MHE EXAMPLE - THREE TANK SYSTEM

- Can only measure tank levels $h_1, h_2$

- The system has two faults:
  - $\phi_1$: leak in tank 1 between $20 \leq t \leq 60$ s
  - $\phi_2$: valve $V_1$ blocked for $t \geq 40$ s

- Add logic constraint $[h_1 \leq h_v] \rightarrow \phi_2 = 0$
A FEW (HYBRID) MPC TRICKS
A measured disturbance $v(t)$ enters the hybrid system.

Augment the hybrid prediction model with the constant state:

$$x^v_{k+1} = x^v_k,$$
$$x^0_v = v(t)$$

HYSDEL model:

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

Same trick applies to linear MPC.

Go to demo demos/hybrid/hyb_meas_dist.m
• Hybrid MPC formulation for reference tracking

\[
\min \sum_{k=0}^{N-1} \| W^y (y_{k+1} - r(t)) \|_2^2 + \| W^\Delta u \Delta u_k \|_2^2
\]

s.t. hybrid dynamics

\[
\Delta u_k = u_k - u_{k-1}, \ k = 0, \ldots, N-1, \ u_{-1} = u(t-1)
\]

\[
u_{\min} \leq u_k \leq u_{\max}, \ k = 0, \ldots, N-1
\]

\[
y_{\min} \leq y_k \leq y_{\max}, \ k = 1, \ldots, N
\]

\[
\Delta u_{\min} \leq \Delta u_k \leq \Delta u_{\max}, \ k = 0, \ldots, N-1
\]

• The resulting optimization problem is the MIQP

\[
\min_{\xi} \quad J(\xi, x(t)) = \frac{1}{2} \xi' H \xi + [x'(t) r'(t) u'(t-1)] F \xi
\]

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

\[
\xi = \begin{bmatrix} \Delta u_0 \\ \delta_0 \\ z_0 \\ \vdots \\ \Delta u_{N-1} \\ \delta_{N-1} \\ z_{N-1} \end{bmatrix}
\]

• Same trick as in linear MPC
- Augment hybrid prediction model with **integrals of output tracking errors**

\[
\epsilon_{k+1} = \epsilon_k + T_s (r(t) - y_k)
\]

- Treat set point \( r(t) \) as a measured disturbance (= constant state)
- Add weight on \( \epsilon_k \) in cost function
- HYSDEL model:

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

- Same trick applies to linear MPC

**go to demo demos/hybrid/hyb_integral_action.m**
• Consider the **time-varying constraint**

\[ u(t) \leq u_{\text{max}}(t) \]

• Augment the hybrid prediction model with the constant state

\[
\begin{align*}
    x_{k+1}^u &= x_k^u \\
    x_0^u &= u_{\text{max}}(t)
\end{align*}
\]

and output \( y_k^u = x^u(k) - u_k \), subject to the constraint \( y_k^u \geq 0, k = 0, 1, \ldots, N \)

• Same trick applies to linear MPC

  go to demo demos/linear/varbounds.m

• **Alternative**: in HYSDEL simply impose \texttt{MUST \{u <= xu;\}}
• Measured disturbance $v(t)$ is known $M$ steps in advance

• Augment the model with the following buffer dynamics

$$\begin{align*}
x_{M-1}^{k+1} &= x_k^{M-2} \\
x_{M-2}^{k+1} &= x_k^{M-3} \\
\vdots \\
x_1^{k+1} &= x_k^0 \\
x_0^{k+1} &= x_k^0
\end{align*}$$

with initial condition

$$\begin{align*}
x_0^{M-1} &= v(t) \\
x_0^{M-2} &= v(t + 1) \\
\vdots &= \vdots \\
x_0^1 &= v(t + M - 2) \\
x_0^0 &= v(t + M - 1)
\end{align*}$$

• The predicted state $x_{M-1}^k$ of the buffer is

$$x_{M-1}^k = \begin{cases} v(t + k) & k = 0, \ldots, M - 1 \\
v(t + M - 1) & k = M, \ldots, N - 1 \end{cases}$$

• Preview of reference signal $r(t + k)$ can be dealt with in a similar way

• Same trick applies to linear MPC
**DELAYS - METHOD #1**

- Hybrid model with **delays**

\[
x(t + 1) = Ax(t) + B_1 u(t - \tau) + B_2 \delta(t) + B_3 z(t) + B_5 E_2 \delta(t) + E_3 z(t) \leq E_1 u(t - \tau) + E_4 x(t) + E_5
\]

- Map delays to poles in \(z = 0\):

\[
x_k(t) \triangleq u(t - k) \quad \Rightarrow \quad x_k(t + 1) = x_{k-1}(t), \quad k = 1, \ldots, \tau
\]

\[
\begin{bmatrix}
  x(t+1) \\
  x_\tau(t+1) \\
  x_{\tau-1}(t+1) \\
  \vdots \\
  x_1(t+1)
\end{bmatrix}
= 
\begin{bmatrix}
  A & B_1 & 0 & 0 & \cdots & 0 \\
  0 & 0 & I_m & 0 & \cdots & 0 \\
  0 & 0 & 0 & I_m & \cdots & 0 \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & 0 & 0 & \cdots & 0
\end{bmatrix}
\begin{bmatrix}
  x(t) \\
  x_\tau(t) \\
  x_{\tau-1}(t) \\
  \vdots \\
  x_1(t)
\end{bmatrix}
+ 
\begin{bmatrix}
  0 \\
  0 \\
  0 \\
  \vdots \\
  I_m
\end{bmatrix}
\begin{bmatrix}
  u(t) \\
  \delta(t) \\
  z(t)
\end{bmatrix}
+ 
\begin{bmatrix}
  B_2 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\begin{bmatrix}
  B_3 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
+ 
\begin{bmatrix}
  B_5 \\
  0 \\
  0 \\
  \vdots \\
  0
\end{bmatrix}
\]

- Apply MPC to the extended MLD system

- Same trick as in linear MPC
• **Delay-free** model:

\[
\bar{x}(t) \triangleq x(t + \tau) \Rightarrow \begin{cases} 
\bar{x}(t + 1) = A\bar{x}(t) + B_1 u(t) + B_2 \bar{\delta}(t) + B_3 \bar{z}(t) + B_5 \\
E_2 \bar{\delta}(t) + E_3 \bar{z}(t) \leq E_1 u(t) + E_4 \bar{x}(t) + E_5 
\end{cases}
\]

• Design MPC for delay-free model, \( u(t) = f_{\text{MPC}}(\bar{x}(t)) \)

• Compute the predicted state

\[
\bar{x}(t) = \hat{x}(t+\tau) = A^\tau x(t) + \sum_{j=1}^{\tau-1} A^j (B_1 u(t - 1 - j) + B_2 \bar{\delta}(t+j) + B_3 \bar{z}(t+j) + B_5)
\]

where \( \bar{\delta}(t + j), \bar{z}(t + j) \) are obtained from MLD inequalities or by simulation

• Compute the MPC control move \( u(t) = f_{\text{MPC}}(\hat{x}(t + \tau)) \)
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:
  \[
  \sum_{i=1}^{N} \delta_i \leq m \quad \text{choose at most } m \text{ items out of } N
  \]
  \[
  \sum_{i=1}^{N} \delta_i = m \quad \text{choose exactly } m \text{ items out of } N
  \]
  \[
  \sum_{i=1}^{N} \delta_i \geq m \quad \text{choose at least } m \text{ items out of } N
  \]
Given a binary vector $\vec{\delta} \in \{0, 1\}^n$ we want to impose the constraint

$$\delta \neq \vec{\delta}$$

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

$$\sum_{i \in T} \delta_i - \sum_{i \in F} \delta_i \leq -1 + \sum_{i=1}^{n} \vec{\delta}_i$$

$$F = \{i : \vec{\delta}_i = 0\}$$

$$T = \{i : \vec{\delta}_i = 1\}$$

or

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

$$\min \ (\ldots) + \sum_{k=0}^{N-1} z_k^2$$

s.t. $z_k \geq u_k$

$z_k \geq 0$

• 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
VERIFICATION (REACHABILITY ANALYSIS)
Hybrid verification problem

hybrid process

continuous disturbances

continuous states

binary disturbances

binary states

verification algorithm

safety query

set of possible initial states

set of possible disturbances

answer

counter-example

always safe!
**Query**: Is the target set $X_f$ reachable after $N$ steps from some initial state $x_0 \in X_0$ for some input $u_0, \ldots, u_{N-1} \in U$?

The query can be answered by solving the **mixed-integer feasibility test**

\[
\min_{\xi} \quad 0 \\
\text{s.t.} \quad x_{k+1} = Ax_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5 \\
E_2 \delta_k + E_3 z_k \leq E_4 x_k + E_1 u_k + E_5 \\
S_u u_k \leq T_u \quad (u_k \in U), \quad k = 0, 1, \ldots, N - 1 \\
S_0 x_0 \leq T_0 \quad (x_0 \in X_0) \\
S_f x_N \leq T_f \quad (x_N \in X_f)
\]

with respect to $\xi = [x_0, \ldots, x_N, u_0, \ldots, u_{N-1}, \delta_0, \ldots, \delta_{N-1}, z_0, \ldots, z_{N-1}]$

**Other approaches:**

- Exploit structure and use polyhedral computation (Torrisi, 2003)
- Use abstractions (LPs) + SAT solvers (Giorgetti, Pappas, Bemporad)
• MLD model: room temperature control system

• Set of unsafe states:
  \[ X_f = \{ \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 10 \leq T_1, T_2 \leq 15 \} \]

• Set of initial states:
  \[ X_0 = \{ \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} : 35 \leq T_1, T_2 \leq 40 \} \]

• Set of possible inputs:
  \[ U = \{ T_{\text{amb}} : 10 \leq T_{\text{amb}} \leq 30 \} \]

• Time horizon: \( N = 10 \) steps
>> umin=20;
>> reach(S,N,Xf,X0,umin,umax);

Hybrid Toolbox v.1.4.2 [February 2, 2020]
Elapsed time is 0.023282 seconds.

Xf is not reachable from X0

\[
U = \{T_{\text{amb}} : 10 \leq T_{\text{amb}} \leq 30\} \quad \text{and} \quad U = \{T_{\text{amb}} : 20 \leq T_{\text{amb}} \leq 30\}
\]
**Query:** Is the target set $X_f$ reachable **within** $N$ steps from some initial state $x_0 \in X_0$ for some input $u_0, \ldots, u_{N-1} \in U$?

**Augment the MLD system to register the entrance of the target (unsafe) set $X_f = \{x : A_f x \leq b_f\}$:**

- Add a new variable $\delta^f_k$, with $[\delta^f_k = 1] \rightarrow [A_f x_{k+1} \leq b_f]$

\[
\begin{align*}
A_f (Ax_k + B_1 u_k + B_2 \delta_k + B_3 z_k + B_5) &\leq b_f + M (1 - \delta^f_k) \\
\text{big-M}
\end{align*}
\]

- Add the constraint $\sum_{k=0}^{N-1} \delta^f_k \geq 1$ (i.e., $x_k \in X_f$ for at least one $k$)

- Solve MILP feasibility test
• States $x_1, x_2, x_3 \in \mathbb{R}, x_4, x_5 \in \{0, 1\}$, inputs $u_1, u_2 \in \mathbb{R}, u_3 \in \{0, 1\}$

\[
\begin{align*}
[\delta_1 = 1] & \leftrightarrow [x_1 \leq 0] \\
[\delta_2 = 1] & \leftrightarrow [x_2 \geq 1] \\
[\delta_3 = 1] & \leftrightarrow [x_3 - x_2 \leq 1]
\end{align*}
\]

• Events:

\[
\begin{align*}
[\delta_1 = 1] & \leftrightarrow [x_1 \leq 0] \\
[\delta_2 = 1] & \leftrightarrow [x_2 \geq 1] \\
[\delta_3 = 1] & \leftrightarrow [x_3 - x_2 \leq 1]
\end{align*}
\]

• Switched dynamics

\[
\begin{align*}
x_1(k+1) &= \begin{cases} 
0.1x_1(k) + 0.5x_2(k) & \text{if } (\delta_1(k) \land \delta_2(k)) \lor x_4(k) \text{ true} \\
-0.3x_3(k) - x_1(k) + u_1(k) & \text{otherwise}
\end{cases} \\
x_2(k+1) &= \begin{cases} 
-0.8x_1(k) + 0.7x_3(k) - u_1(k) - u_2(k) & \text{if } \delta_3(k) \lor x_5(k) \text{ true} \\
-0.7x_1(k) - 2x_2(k) & \text{otherwise}
\end{cases} \\
x_3(k+1) &= \begin{cases} 
-0.1x_3(k) + u_2(k) & \text{if } (\delta_3(k) \land x_5(k)) \lor (\delta_1(k) \land x_4(k)) \text{ true} \\
x_3(k) - 0.5x_1(k) - 2u_1(k) & \text{otherwise}
\end{cases}
\end{align*}
\]

• Automaton

\[
\begin{align*}
x_4(k+1) &= \delta_1(k) \land x_4(k) \\
x_5(k+1) &= ((x_4(k) \lor x_5(k)) \land (\delta_1(k) \lor \delta_2(k)) \lor (\delta_3(k) \land u_3(k))
\end{align*}
\]
A MORE COMPLEX VERIFICATION EXAMPLE

- **Query**: Verify if it possible that, starting from the set $X_0$

  $$X_0 = \{ x : -0.1 \leq x_1, x_3 \leq 0.1, x_2 = 0.1, x_4, x_5 \in \{0, 1\} \}$$

  the state $x(k) \in X_f$

  $$X_f = \{ x : -1 \leq x_1, x_3 \leq 1, 0.5 \leq x_2 \leq 1, x_4, x_5 \in \{0, 1\} \}$$

  at some $k \leq N, N = 5$, under the restriction that $\forall k \leq N$

  $$x_3(k) + x_2(k) \leq 0$$
  $$\delta_1(k) \vee \delta_2(k) \vee x_5(k) = \text{true}$$
  $$\neg x_4(k) \vee x_5(k) = \text{true}$$

  >> [flag,x0,U,xf,X,T,D,Z,Y,reachtime]=reach(S,[1 N],Xf,X0);

  go to demo demos/hybrid/reachtest.m
>> reachtest

Hybrid Toolbox v.1.4.2 [February 2, 2020]
Elapsed time is 0.038049 seconds.

>> reachtime

reachtime =

3
4

>>

The set $X_f$ is reached by $x(k)$ at time steps $k = 3, 4$