Explicit Model Predictive Control

KKT Conditions for Optimality

$$\begin{array}{ll} \min\limits_{U} & f(U) \\ \text{s.t.} & g_i(U) \le 0, \quad \forall i = 1,\ldots,m \\ & h_j(U) = 0, \quad \forall j = 1,\ldots,p \end{array}$$

KKT Conditions (necessary) Let U^* be a feasible solution, let $I = \{i : g_i(U^*) = 0\}$. Suppose f and g_i, h_j differentiable at U^*, Suppose $\nabla g_i(U^*)$, $\nabla h_j(U^*)$ linearly independent for $i \in I$.

Then, if U^* is optimal, there exists vector of **Lagrange multipliers** $\lambda \in \mathbb{R}^m$, $\nu \in \mathbb{R}^p$ such that

$$\nabla f(U^*) + \sum_{i=1}^{m} \lambda_i \nabla g_i(U^*) + \sum_{j=1}^{p} \nu_j \nabla h_j(U^*) = 0 \qquad \text{(1a)}$$

$$\lambda_i g_i(U^*) = 0, \quad \forall i = 1,\ldots,m \qquad \text{(1b)}$$

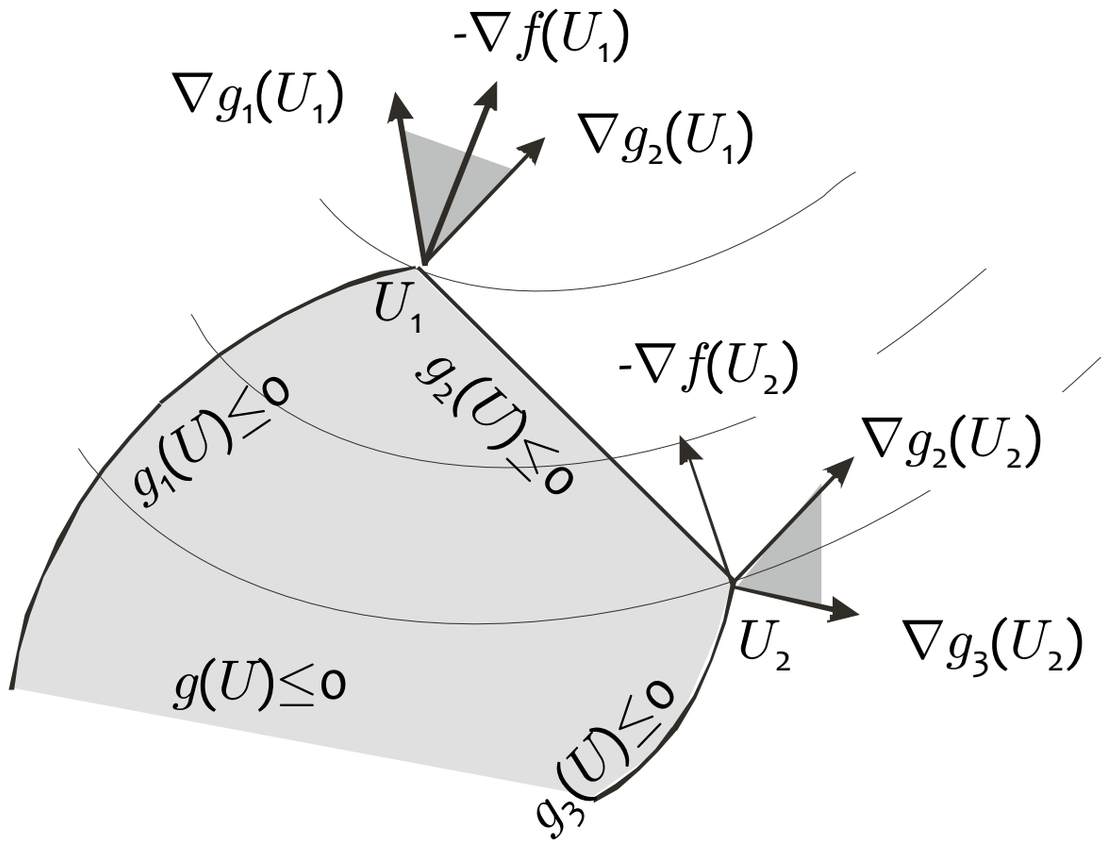
$$\lambda_i \ge 0 \qquad \text{(1c)}$$

$$g_i(U^*) \le 0 \qquad \text{(1d)}$$

$$h_j(U^*) = 0, \quad \forall j = 1,\ldots,p \qquad \text{(1e)}$$

When f, g_i are convex functions and h_j are linear, the condition is also sufficient

KKT - Geometric Interpretation



- $-\nabla f(U) = \sum_{i\in I} \lambda_i \nabla g_i(U),\ \lambda_i \geq 0$, i.e. $-\nabla f$ (the direction of maximum decrease of f) belongs to the convex cone spanned by ∇g_i's, where g_i is a binding constraint

- f can only decrease if some constraint is violated, Therefore U must be optimal (see point U_1)

- Conversely, if some $\lambda_i < 0$ one can move within the set of feasible points $g(U) \leq 0$ and decrease f, which implies that U is not optimal (see point U_2).

KKT Conditions for QP

$$\begin{array}{ll} \min\limits_{U} & f(U) \triangleq \frac{1}{2}U'HU + c'U \\ \text{s.t.} & AU \leq b \end{array}$$

$U \in \mathbb{R}^n,\ H \succeq 0 \in \mathbb{R}^{n \times n}$
$A \in \mathbb{R}^{m \times n}.$

$$\nabla f(U) = HU + c$$
$$g_i(U) = A_i'U - b_i \ (A_i' \text{ is the } i\text{-th row of } A)$$
$$\nabla g_i(U) = A_i$$

Harold W. Kuhn
(1925 -)

$$\begin{aligned} HU + c + A'\lambda &= 0 \\ \lambda_i(A_i'U - b_i) &= 0 \\ \lambda &\geq 0 \\ AU - b &\leq 0 \end{aligned}$$

Albert W. Tucker
(1905 - 1995)

William Karush
(1917 - 1997)

QP Algorithms

- **Method of feasible directions**. Given a feasible point U_k and a direction d_k, find a scalar α such that (*i*) $U_k + \alpha d_k$ is feasible, and (*ii*) $f(U_k + \alpha d_k) < f(U_k)$. The method proceeds iteratively by letting $U_{k+1} = U_k + \alpha^* d_k$, where α^* is the optimal α.

- **Active set strategy**. A combinatorial approach to iteratively determine the set of binding constraints at optimality.

- **Linear complementary problem**. The KKT conditions are reformulated as a linear complementary problem, for which efficient simplex-like methods (similar to the LP case) are available.

- **Interior point methods**. Constraints are replaced by barrier functions, and a sequence of unconstrained optimization problems is solved.

MPC Computations

- The on-line optimization problem is a Quadratic Program (QP)

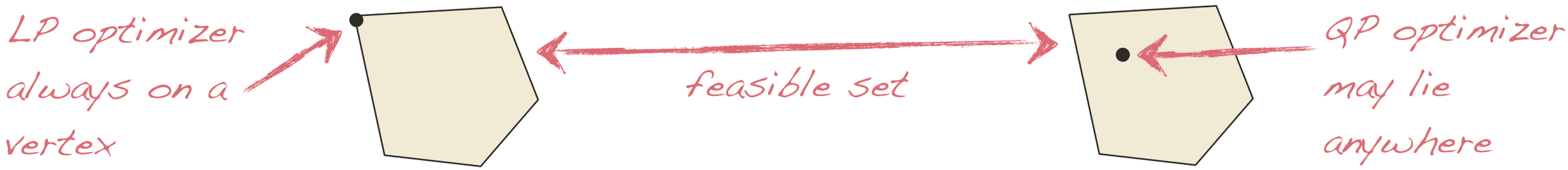
  (or Linear Program, LP)

- No need to reach global optimum (see theorem proof)
- Algorithms:
  - Active set methods (small/medium size)
  - Interior point methods (large size)

- Benchmarks on commercial/public domain QP/LP solvers:

  http://plato.la.asu.edu/bench.html

- Note: using Linear Programming (LP) for 1- or ∞-norms, control action may be less smooth than with QP



Matlab Optimization Toolbox - LP

linprog

Purpose

Solve a linear programming problem

$$\min_x \; f^T x \quad \text{such that} \quad \begin{aligned} A \cdot x &\le b \\ Aeq \cdot x &= beq \\ lb \le x &\le ub \end{aligned}$$

where f, x, b, beq, lb, and ub are vectors and A and Aeq are matrices.

Syntax

```
x = linprog(f,A,b,Aeq,beq)
x = linprog(f,A,b,Aeq,beq,lb,ub)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options)
[x,fval] = linprog(...)
[x,fval,exitflag] = linprog(...)
[x,fval,exitflag,output] = linprog(...)
[x,fval,exitflag,output,lambda] = linprog(...)
```

Description

linprog solves linear programming problems.

Much more efficient LP solvers are available !

(GLPK, CDD, CPLEX, NAG, and many others ...)

Hybrid Toolbox

LPSOL

Matlab Optimization Toolbox QP

quadprog

Purpose

Solve the quadratic programming problem

$$\min_x \ \frac{1}{2}x^T Hx + f^T x \quad \text{such that} \quad \begin{aligned} A \cdot x &\le b \\ Aeq \cdot x &= beq \\ lb \le x &\le ub \end{aligned}$$

where H, A, and Aeq are matrices, and f, b, beq, lb, ub, and x are vectors.

Syntax

```
x = quadprog(H,f,A,b)
x = quadprog(H,f,A,b,Aeq,beq)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0)
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
[x,fval] = quadprog(...)
[x,fval,exitflag] = quadprog(...)
[x,fval,exitflag,output] = quadprog(...)
[x,fval,exitflag,output,lambda] = quadprog(...)
```

Description

`x = quadprog(H,f,A,b)` returns a vector x that minimizes `1/2*x'*H*x + f'*x` subject to `A*x <= b`.

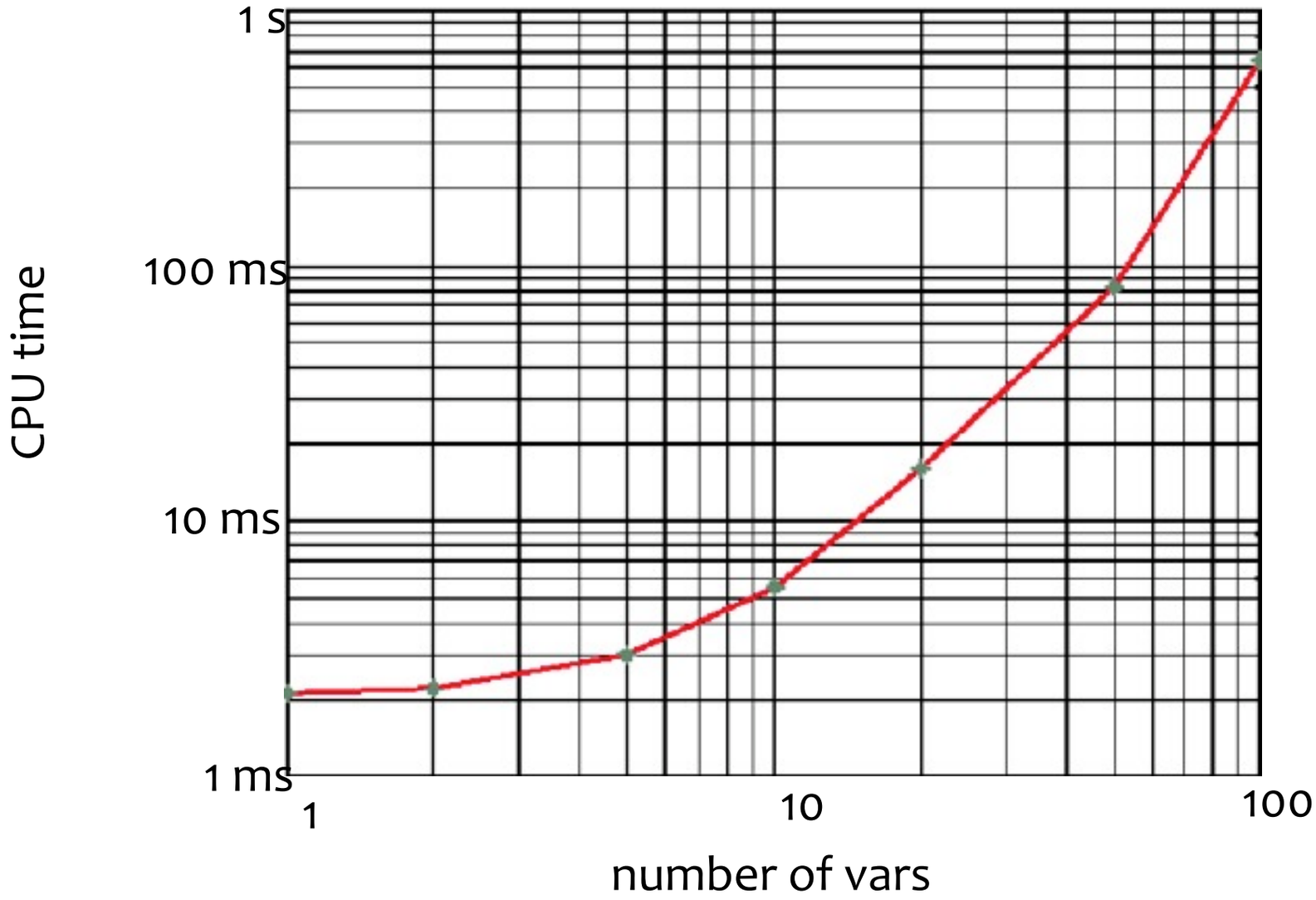
Hybrid Toolbox

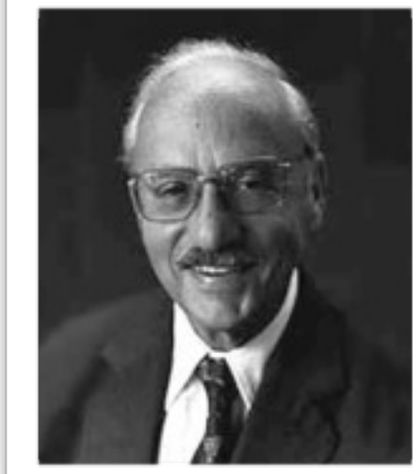
QPSOL

Much more efficient QP solvers are available !

(CPLEX, NAG, DANTZGMP, and many others ...)

QP Complexity





George Dantzig
(1914 - 2005)

- Non-optimized Dantzig's routine (MPC Toolbox for Matlab)
- Tests in Matlab 5.3 on Pentium II 300 MHz

Explicit Form of Model Predictive Control via Multiparametric Programming

MPC: Pros and Cons

- PRO: systematic design approach
  - Multivariable systems
  - State and input constraints
  - Stability guarantees
  - Reference preview, delays, …

- CON: computation complexity !
  - Large sampling time/fast hardware
  - Software reliability

Main drawbacks of on-line implementation

- Excellent LP/QP/MIP/NLP solvers exist today (*"LP is a technology" – S. Boyd*)

but ...

- Computation time may be too long: ok for large sampling times (>10 ms) but not for fast-sampling applications (< 1 ms).
  Worst-case CPU time often hard to estimate

- Requires relatively expensive hardware (not suitable on inexpensive 8-bit μ-controllers with few kB RAM)

- Software complexity: control profile $u(x)$ hard to understand, solver code difficult to certify
  (bad in safety critical apps)

Any way to use MPC **without** on-line solvers ?

On-Line vs. Off-Line Optimization

$$\begin{aligned} \min_{U} \quad & \frac{1}{2}U'HU + x'(t)F'U + \frac{1}{2}x'(t)Yx(t) \\ \text{subj. to} \quad & GU \leq W + Sx(t), \end{aligned}$$

- **On-line** optimization: given $x(t)$ solve the problem at each time step t (the control law u=$u(x)$ is **implicitly** defined by the QP solver)

→ Quadratic Program (QP)

- **Off-line** optimization: solve the QP **for all** $x(t)$ to find the control law u=$u(x)$ **explicitly**

→ multi-parametric Quadratic Program (mp-QP)

Multiparametric programming problem

Given the optimization problem

$$\begin{array}{ll} \min\limits_{U} & h(U, x) \\ \text{s.t.} & g(U, x) \leq 0 \end{array}$$

determine:

- The **feasible parameter set** X_f of all $x \in X$ for which the problem admits a solution ($g(U, x) \leq 0$ for some U)

- The **value function** $V^* : X_f \rightarrow \mathbb{R}$ that at each x associates the optimal value $V^*(x)$

- An **optimizer function** $U^* : X_f \rightarrow \mathbb{R}^\ell$

Multiparametric Quadratic Programming

(Bemporad et al., 2002)

$$\begin{aligned} \min_{U} \quad & \frac{1}{2}U'HU + x'F'z + \frac{1}{2}xYx \\ \text{subj. to} \quad & GU \leq W + Sx, \end{aligned}$$

$$U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

- Objective: solve the QP off line **for all** $x \in X$ to find the MPC control law $u=u(x)$ **explicitly**

- Assumptions:

$$\begin{bmatrix} H & F \\ F' & Y \end{bmatrix} \succeq 0$$

always satisfied if mpQP comes from an optimal control problem !

$$H \succ 0$$

always satisfied if weight matrix $R>0$

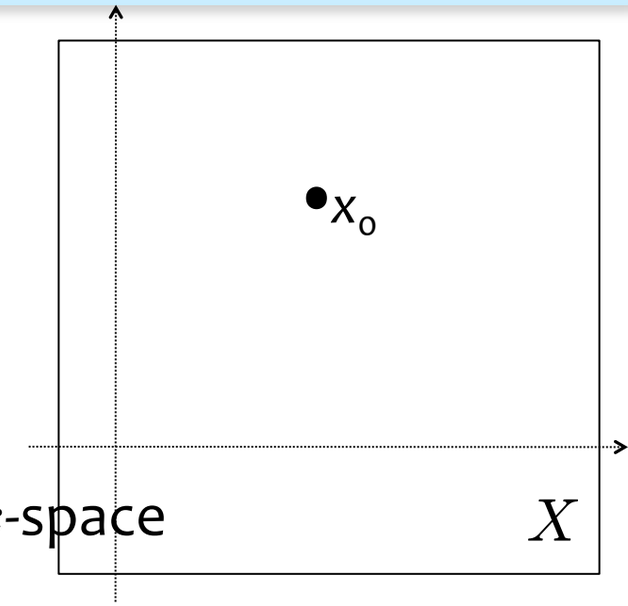
Linearity of the Solution

Fix $x_0 \in X$

⇒ solve QP to find $U^*(x_0),\ \lambda^*(x_0)$

⇒ identify active constraints $\tilde{G},\ \tilde{W},\ \tilde{S}$ at $U^*(x_0)$

⇒ form matrices by collecting active constraints: $\tilde{G}z^*(x_0) - \tilde{W} - \tilde{S}x_0 = 0$



KKT optimality conditions:

$$
\begin{array}{ll}
(1)\ HU + Fx + G'\lambda = 0, & (2)\ \tilde{G}U - \tilde{W} - \tilde{S}x = 0 \\
(3)\ \lambda_i(G^iU - W^i - S^ix) = 0, & (4)\ \hat{G}U \le \hat{W} + \hat{S}x \\
(5)\ \tilde{\lambda}_i \ge 0,\ \hat{\lambda}_i = 0 &
\end{array}
$$

$\hat{G}$=rows of G not in $\tilde{G}$ (inactive constraints)

From (1): $U = -H^{-1}(Fx + \tilde{G}'\tilde{\lambda})$

From (2):

$$\tilde{\lambda}(x) = -(\tilde{G}H^{-1}\tilde{G}')^{-1}(\tilde{W} + (\tilde{S} + \tilde{G}H^{-1}F)x).$$

$$U(x) = H^{-1}[\tilde{G}'(\tilde{G}H^{-1}\tilde{G}')^{-1}(\tilde{W}+(\tilde{S}+\tilde{G}H^{-1}F)x)-Fx]$$

⇒ In some neighborhood of x_0, λ and U are explicit affine functions of x !

(Zafiriou, 1990)

Multiparametric QP algorithm

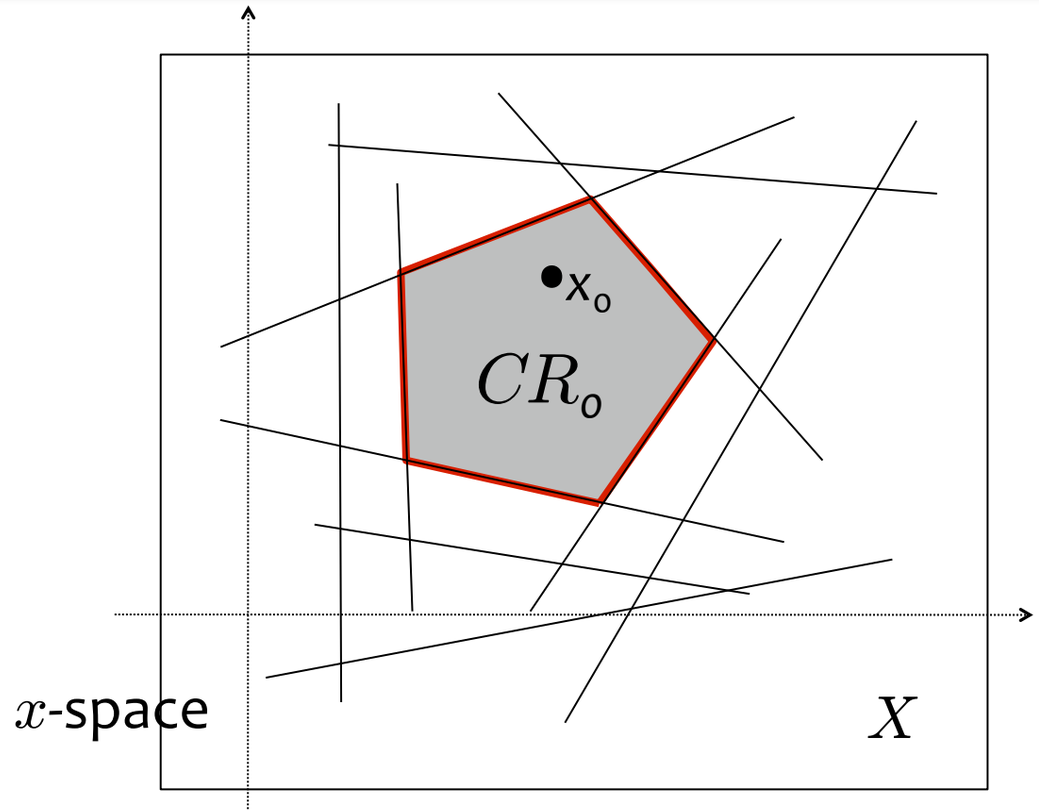
- Impose primal and dual feasibility:

$$\begin{aligned} &\hat{G}U(x) \leq \hat{W} + \hat{S}x \\ &\tilde{\lambda}(x) \geq 0 \end{aligned}$$

from (4)

from (5)

➡ linear inequalities in x !



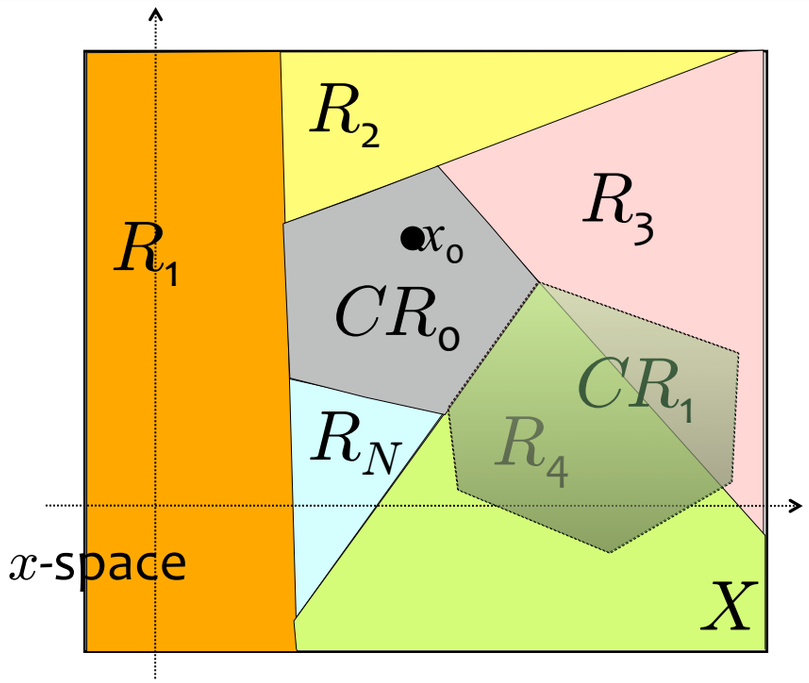
- Remove redundant constraints (this requires solving LP's):

➡ critical region CR_0

$$CR_0 = \{x \in X : \ \mathcal{A}x \leq \mathcal{B}\}$$

- CR_0 is the set of all and only parameters x for which $\tilde{G},\ \tilde{W},\ \tilde{S}$ is the optimal combination of active constraints at the optimizer

Multiparametric QP algorithm



Method #1: Split and proceed iteratively

(Bemporad, Morari, Dua, Pistikopoulos, 2002)

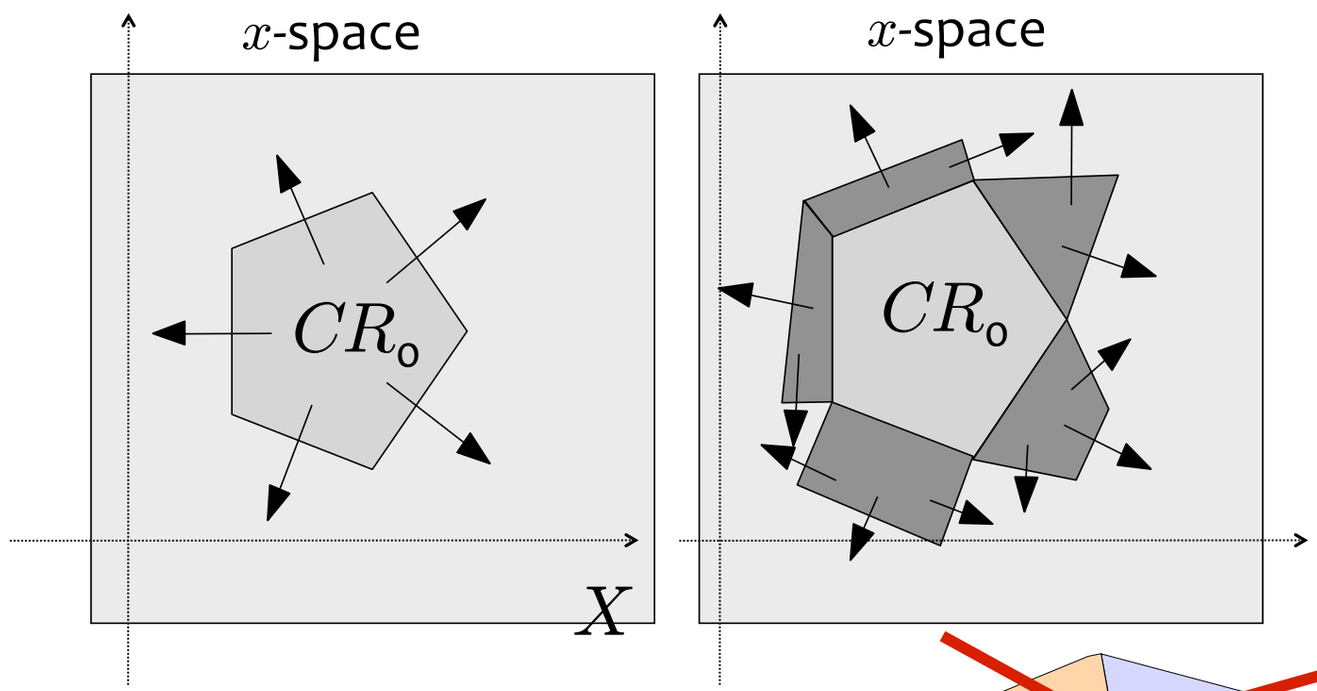
Recursions terminate because combinations of active constraints are finite

Method #2: add/withdraw constraints from active set

$$\hat{G}^i z(x) \le \hat{W}^i + \hat{S}^i x \quad \Rightarrow \text{add}$$

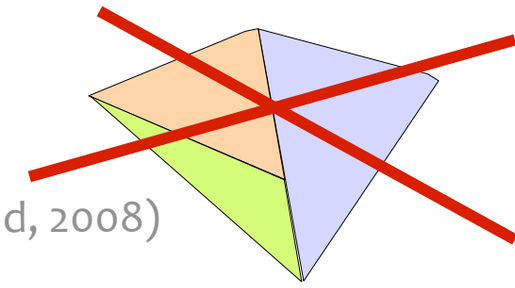
$$\tilde{\lambda}_j(x) \ge 0 \quad \Rightarrow \text{withdraw}$$

(Tøndel, Johansen, Bemporad, 2003)



Method #3: exploit the *facet-to-facet* property

(Spjøtvold, Kerrigan, Jones, Tøndel, Johansen, 2006)



(Spjøtvold, 2008)

Step out ϵ outside each facet, solve QP, get new region, iterate. (Baotic, 2002)

Properties of multiparametric-QP

Theorem 1 *Consider a multi-parametric quadratic program with* $H \succ 0$, $\begin{bmatrix} H & F \\ F' & Y \end{bmatrix} \succeq 0$. *The set* X^* *of parameters* x *for which the problem is feasible is a polyhedral set, the value function* $V^* : X^* \mapsto \mathbb{R}$ *is piecewise quadratic, convex and continuous and the optimizer* $U^* : X^* \mapsto \mathbb{R}^r$ *is piecewise affine and continuous.*

$$U^*(x) = \arg\min_U \ \tfrac{1}{2}U'HU + x'F'U \quad \text{subj. to} \quad GU \leq W + Sx$$

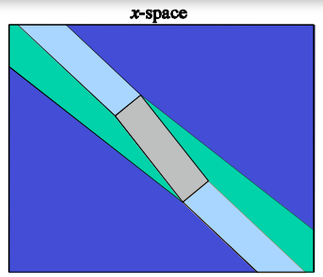
continuous,
piecewise affine

$$V^*(x) = \tfrac{1}{2}x'Yx + \min_U \ \tfrac{1}{2}U'HU + x'F'U \quad \text{subj. to} \quad GU \leq W + Sx$$

convex, continuous,
piecewise quadratic,
C^1 (if no degeneracy)

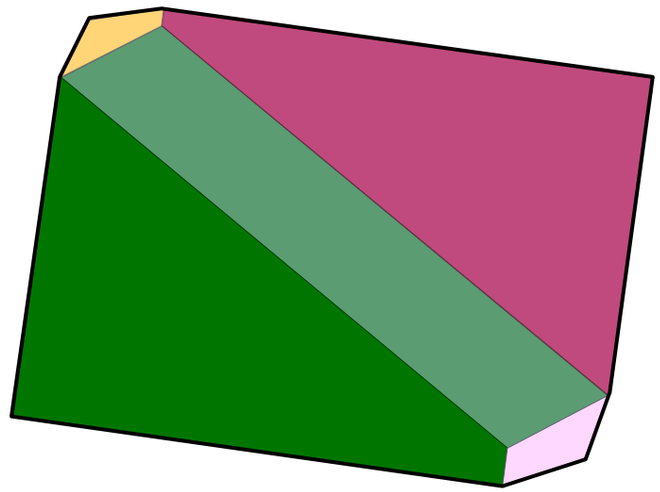
Corollary: The linear MPC controller is a continuous piecewise affine function of the state

$$u(x) = \begin{cases} F_1x + g_1 & \text{if} \quad H_1x \leq K_1 \\ \vdots & \vdots \\ F_Mx + g_M & \text{if} \quad H_Mx \leq K_M \end{cases}$$



Set of Feasible Parameters X^*

Why is the set X^* of parameters for which the QP problem is solvable a convex polyhedral set ?



$$X^* = \{x : \exists U \text{ such that } GU \leq W + Sx\}$$

X^* is the projection of a polyhedron onto the parameter space. Therefore X^* is a polyhedron.

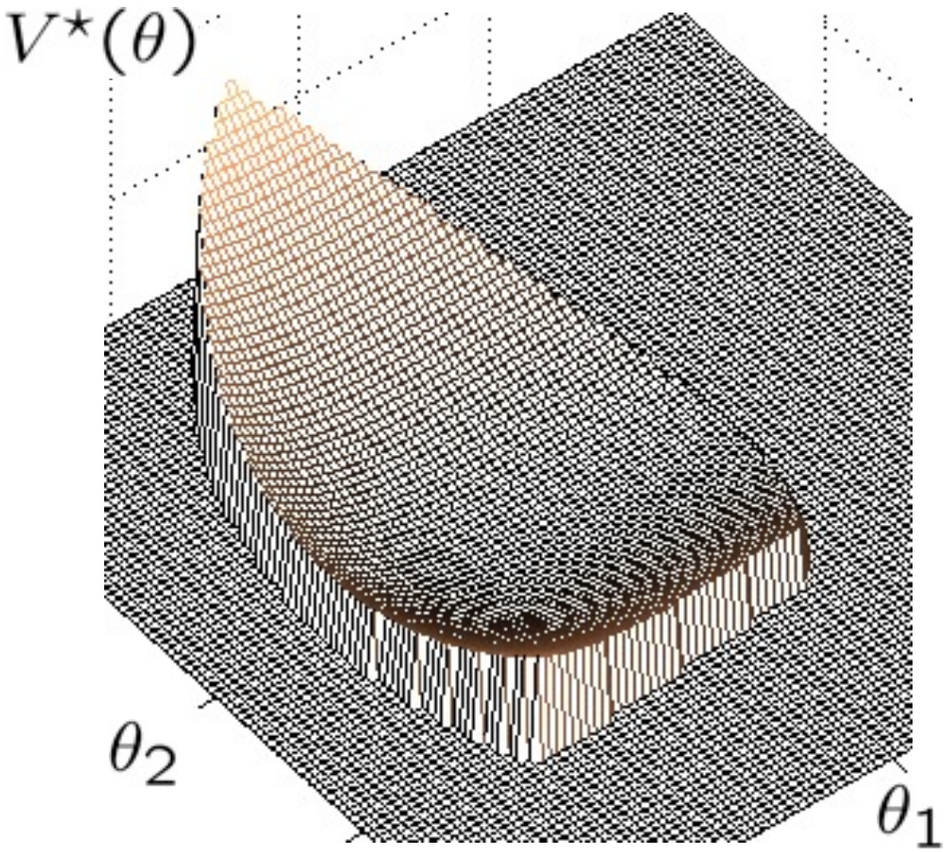
$$X^* = \text{Proj}_x \left\{ \begin{bmatrix} U \\ x \end{bmatrix} : \begin{bmatrix} G & -S \end{bmatrix} \begin{bmatrix} U \\ x \end{bmatrix} \leq W \right\}$$

Multiparametric Convex Programming

$$\begin{aligned} \min_x \quad & f(U,x) \\ \text{s.t.} \quad & g_i(U,x) \le 0 \quad (i=1,\dots,p) \\ & AU + Bx + d = 0 \end{aligned}$$

Lemma *Let f, g_i be jointly convex functions of (U,x) $(\forall i = 1,\dots,p)$. Then X^* is a convex set and $V^\star$ is a convex function of x.*

(Mangasarian, Rosen, 1964)

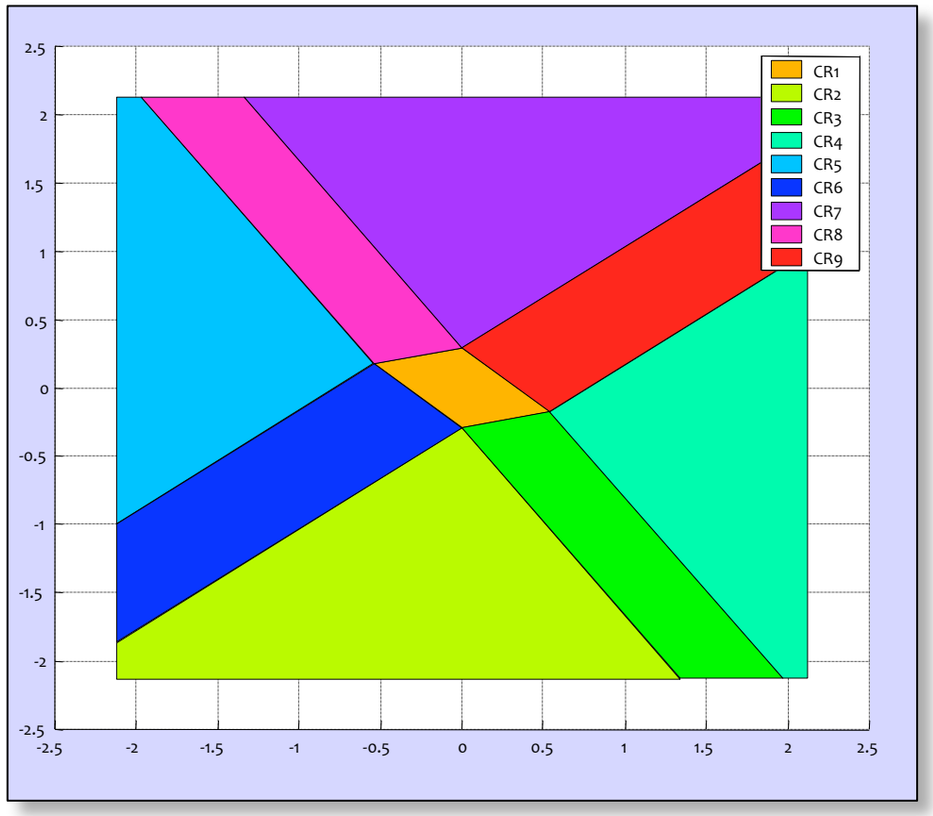


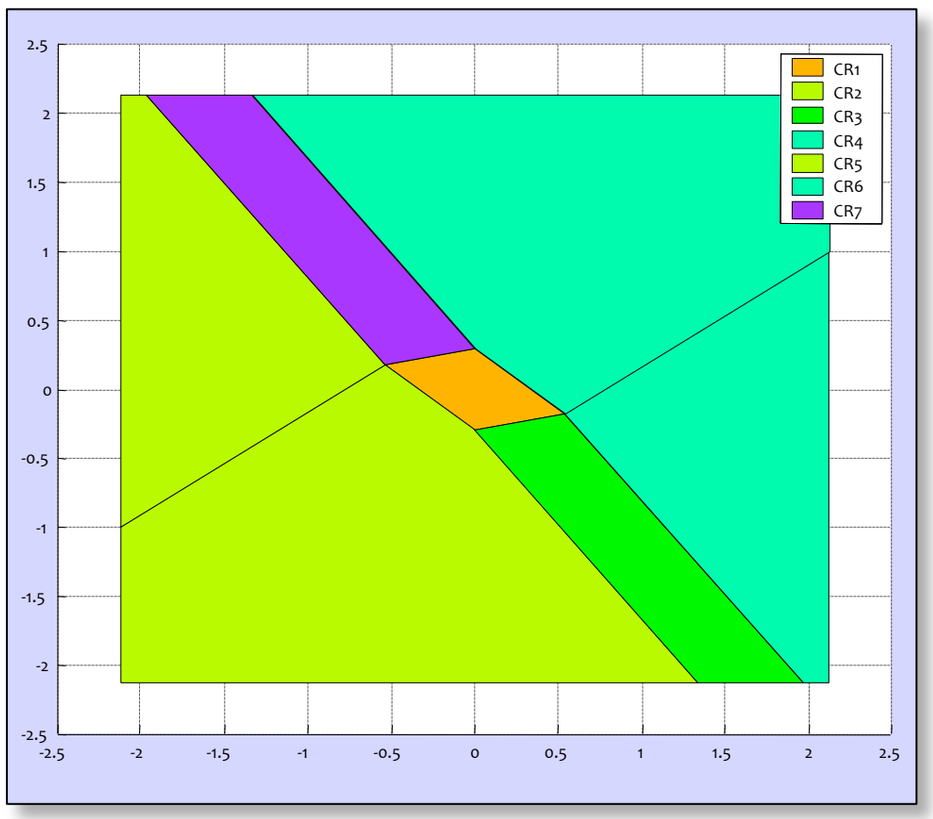
V^* and X^* may not be easy to express analytically

(approximate solutions possible)

(Bemporad, Filippi, 2003)

Complexity Reduction





$$U(x) \triangleq [u'_0(x) \; u'_1(x) \; \ldots \; u'_{N-1}(x)]'$$

Regions where the first component of the solution is the same can be joined (when their union is convex).

(Bemporad, Fukuda, Torrisi, *Computational Geometry*, 2001)

Double Integrator Example

- System: $y(t) = \frac{1}{s^2}u(t)$

sampling + ZOH
T_s=1 s

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \end{aligned}$$

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

- Control objective: minimize $\sum_{t=0}^{\infty} y^2(t) + \frac{1}{100}u^2(t)$

$$u(t+k) = K_{LQ}x(t+k|t), \ \forall k \ge N_u$$

- Optimization problem: for N_u=2

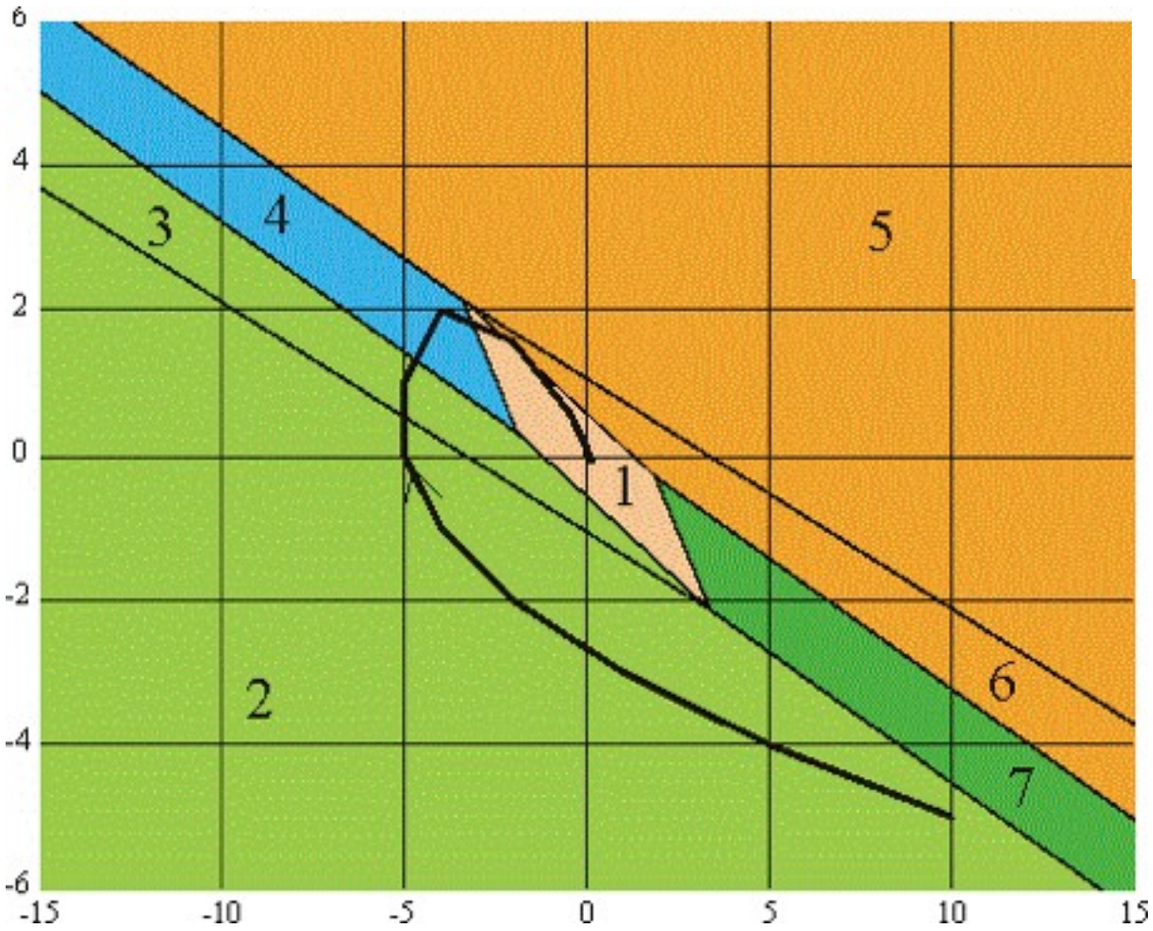
$$H = \begin{bmatrix} 0.8365 & 0.3603 \\ 0.3603 & 0.2059 \end{bmatrix}, \ F = \begin{bmatrix} 0.4624 & 1.2852 \\ 0.1682 & 0.5285 \end{bmatrix}$$

$$G = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \ W = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \ S = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

(cost function is normalized by max svd(H))

mp-QP solution

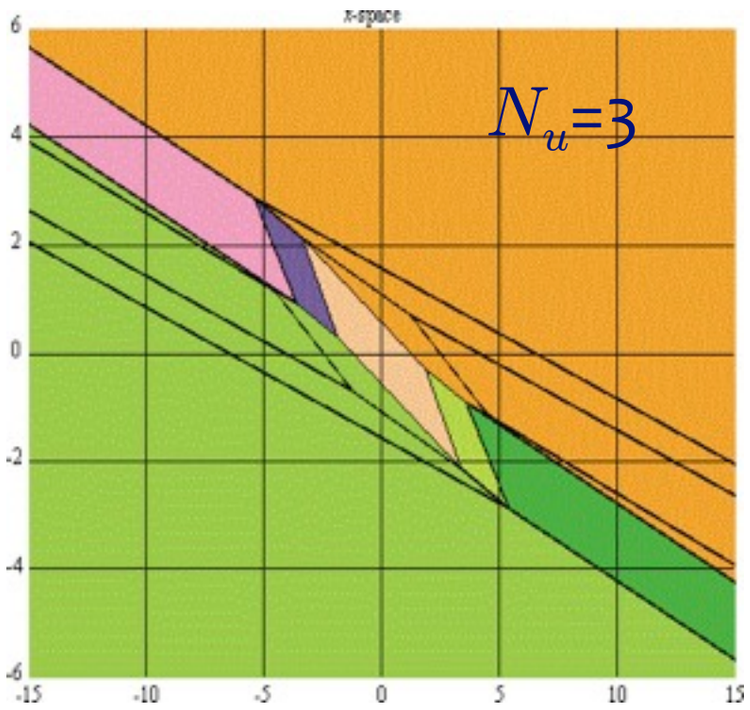
N_u=2

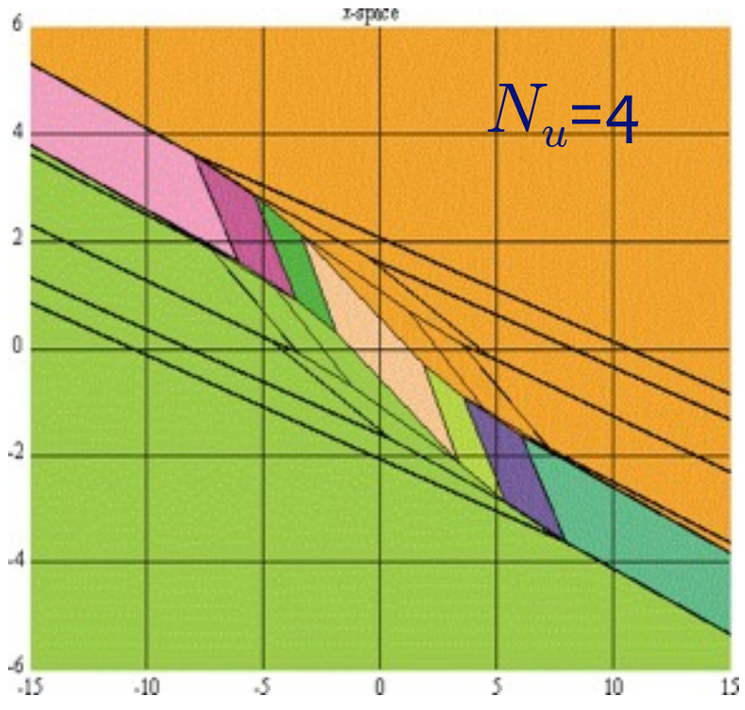


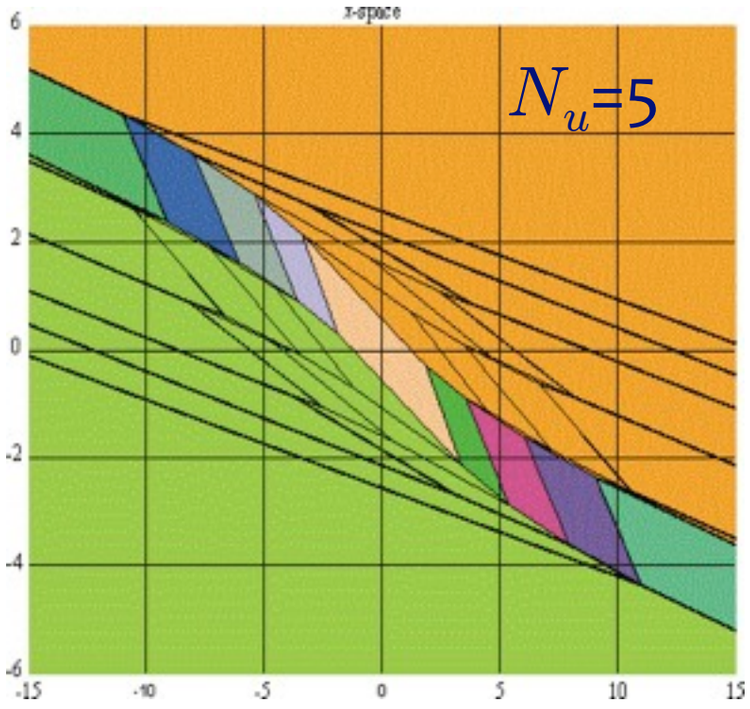
$$u(x) = \begin{cases} \begin{bmatrix} -0.8166 & -1.7499 \end{bmatrix} x & \text{if } \begin{bmatrix} -0.8166 & -1.7499 \\ 0.8166 & 1.7499 \\ 0.6124 & 0.4957 \\ -0.6124 & -0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} & \text{(Region \#1)} \\ 1.0000 & \text{if } \begin{bmatrix} 0.3864 & 1.0738 \\ 0.2970 & 0.9333 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#2)} \\ 1.0000 & \text{if } \begin{bmatrix} 0.9712 & 2.6991 \\ -0.2970 & -0.9333 \\ 0.8166 & 1.7499 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#3)} \\ \begin{bmatrix} -0.5528 & -1.5364 \end{bmatrix} x + 0.4308 & \text{if } \begin{bmatrix} -0.9712 & -2.6991 \\ 0.3864 & 1.0738 \\ 0.6124 & 0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#4)} \\ -1.0000 & \text{if } \begin{bmatrix} -0.3864 & -1.0738 \\ -0.2970 & -0.9333 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#5)} \\ -1.0000 & \text{if } \begin{bmatrix} -0.9712 & -2.6991 \\ 0.2970 & 0.9333 \\ -0.8166 & -1.7499 \end{bmatrix} x \le \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#6)} \\ \begin{bmatrix} -0.5528 & -1.5364 \end{bmatrix} x - 0.4308 & \text{if } \begin{bmatrix} -0.3864 & -1.0738 \\ 0.9712 & 2.6991 \\ -0.6124 & -0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} & \text{(Region \#7)} \end{cases}$$

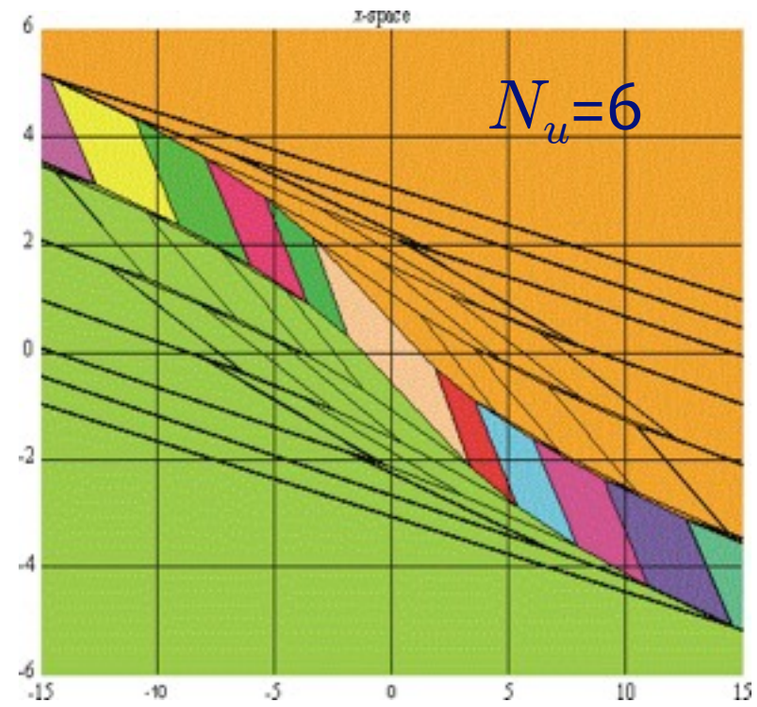
go to demo **`/demos/linear/doubleintexp.m`** (Hyb-Tbx)

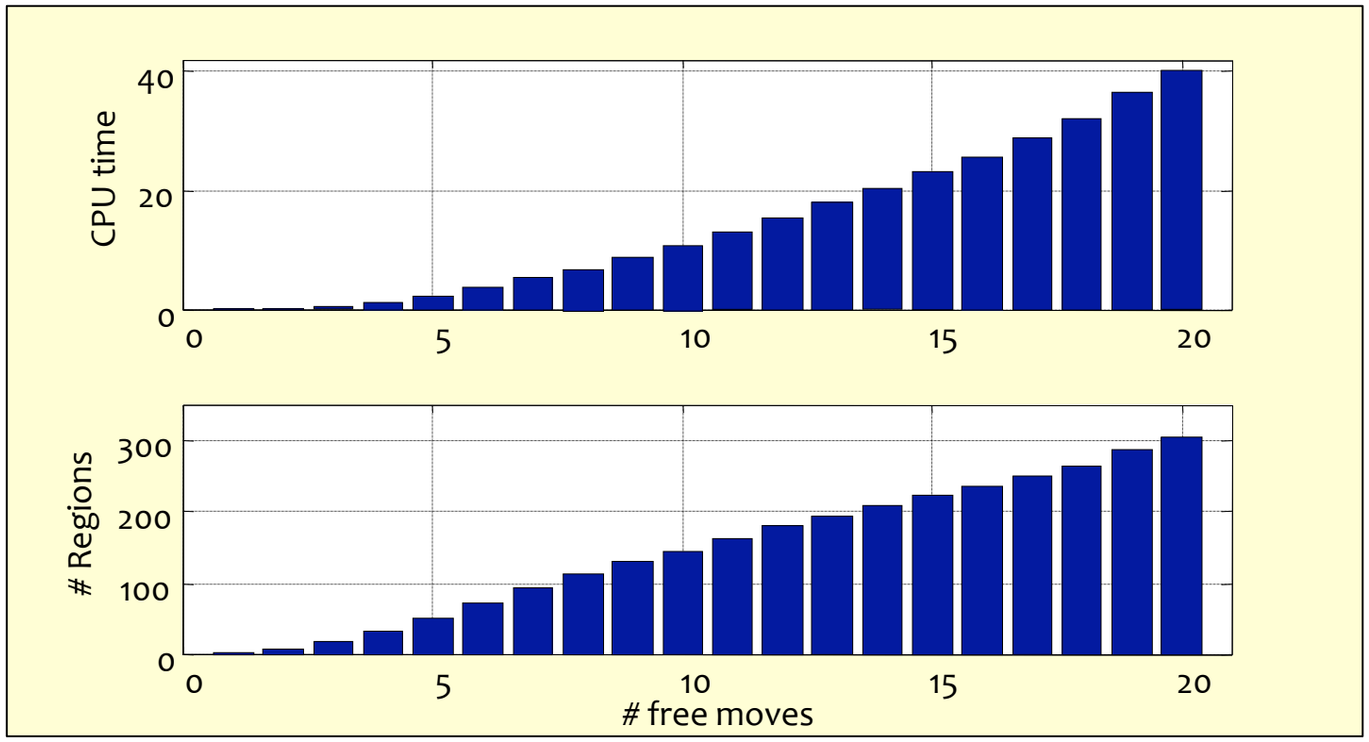
Complexity











(is the number of regions finite for $N_u \to \infty$?)

Complexity

- Worst-case complexity analysis:

$$M \triangleq \sum_{\ell=0}^{q} \binom{q}{\ell} = 2^q$$ combinations of active constraints

- Usually the number of regions is much smaller, as many combinations of active constraints are never feasible and optimal at any parameter vector x

- Strongest dependence on the number q of constraints

- Strong dependence on the number N_u of free moves

- Weak dependence on the number n of parameters x

- Example:

states\horizon	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$
n=2	3	6.7	13.5	21.4	19.3
n=3	3	6.9	17	37.3	77
n=4	3	7	21.65	56	114.2
n=5	3	7	22	61.5	132.7
n=6	3	7	23.1	71.2	196.3
n=7	3	6.95	23.2	71.4	182.3
n=8	3	7	23	70.2	207.9

Data averaged over 20 randomly generated single-input single-output systems subject to input saturation (q=2N)

Extensions

- Tracking of reference $r(t)$:

$$\Delta u(t) = f(x(t), u(t-1), r(t))$$

- Rejection of measured disturbance $v(t)$:

$$\Delta u(t) = f(x(t), u(t-1), v(t))$$

- Soft constraints:

$$u(t) = f(x(t))$$

$$y_{\min} - \epsilon \leq y(k) \leq y_{\max} + \epsilon$$

- Variable constraints:

$$u(t) = f(x(t), u_{\min}(t), \ldots, y_{\max}(t))$$

$$\begin{aligned} u_{\min}(t) &\leq u(k) \leq u_{\max}(t) \\ y_{\min}(t) &\leq y(k) \leq y_{\max}(t) \end{aligned}$$

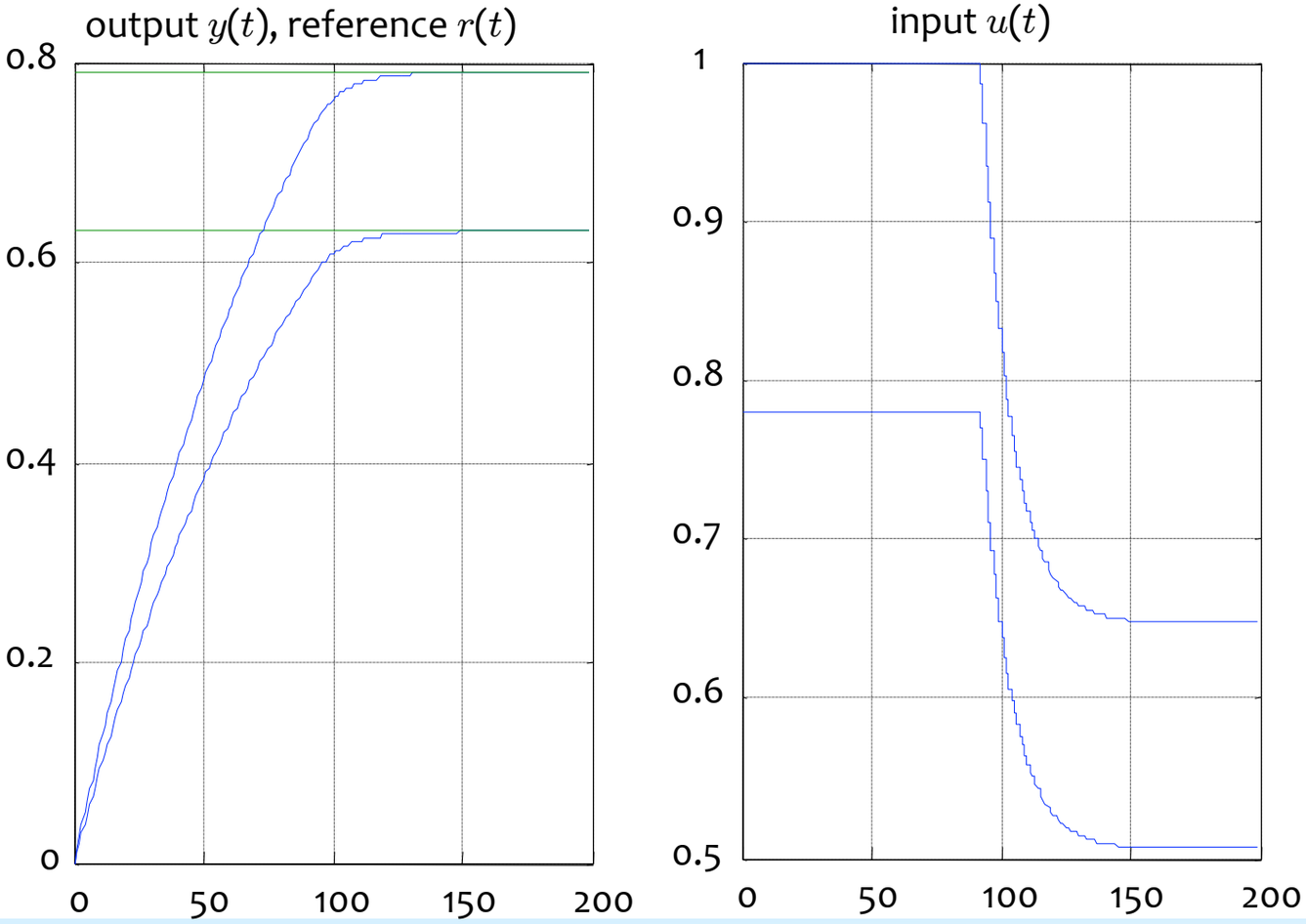
- Other models (hybrid, uncertain) and other norms ($\|\cdot\|_1$, $\|\cdot\|_\infty$)

Reference Tracking, MIMO System

- System: $y(t) = \dfrac{10}{100s+1}\begin{bmatrix} 4 & -5 \\ -3 & 4 \end{bmatrix} u(t)$ sampling + ZOH (T_s=1 s)
- Constraints: $-1 \le u_1, u_2 \le 1$
- Control objective:

$$\min \sum_{k=0}^{19} \|y(t+k|t) - r(t)\|^2 + \frac{1}{10}\|\Delta u(t+k)\|^2$$

$$u(t+k) = u(t), \ \forall k \ge 1$$



N=20

N_u=1

go to demo **`linear/mimo.m`**

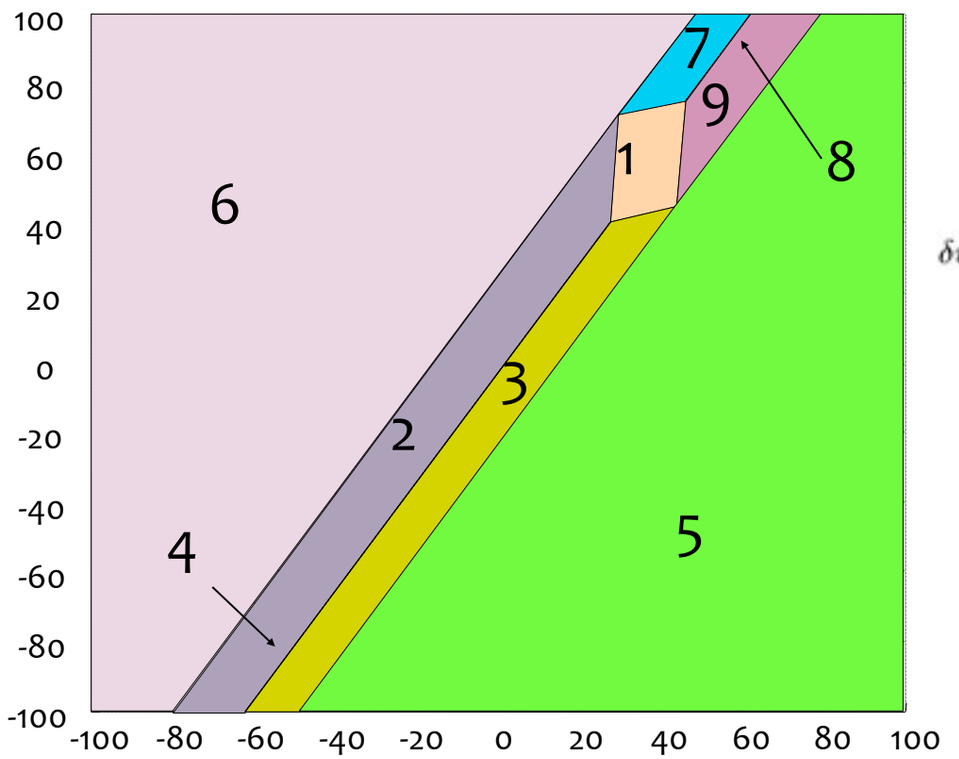
(Hyb-Tbx)

Reference Tracking, MIMO System

MPC law

State-space partition

x-space



Polyhedral partition of the state-space for u=[0 0]' and r=[0.63 0.79]'

$$
\delta u = \begin{cases}
\begin{bmatrix} -0.1251 & 0.0084 \\ 0.0168 & -0.0668 \end{bmatrix} x + \begin{bmatrix} -0.8535 & 0.1143 \\ 0.1143 & -0.9107 \end{bmatrix} u + \begin{bmatrix} 2.5583 & 3.1741 \\ 1.8949 & 2.5583 \end{bmatrix} r & \text{if } \begin{bmatrix} -0.1251 & 0.0084 \\ 0.0168 & -0.0668 \\ 0.1251 & -0.0084 \\ -0.0168 & 0.0668 \end{bmatrix} x + \begin{bmatrix} 0.1465 & 0.1143 \\ 0.1143 & 0.0893 \\ -0.1465 & -0.1143 \\ -0.1143 & -0.0893 \end{bmatrix} u + \begin{bmatrix} 2.5583 & 3.1741 \\ 1.8949 & 2.5583 \\ -2.5583 & -3.1741 \\ -1.8949 & -2.5583 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} \quad \text{(Region \#1)} \\
\begin{bmatrix} 0.0000 & 0.0000 \\ 0.1144 & -0.0733 \end{bmatrix} x + \begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -0.9999 \end{bmatrix} u + \begin{bmatrix} 0.0000 & 0.0000 \\ -0.1016 & 0.0813 \end{bmatrix} r + \begin{bmatrix} 1.0000 \\ 0.7804 \end{bmatrix} & \text{if } \begin{bmatrix} 0.5210 & -0.3337 \\ -0.0643 & 0.0412 \\ 0.1251 & -0.0084 \end{bmatrix} x + \begin{bmatrix} -0.0000 & 0.0005 \\ 0.0000 & -0.0001 \\ -0.1465 & -0.1143 \end{bmatrix} u + \begin{bmatrix} -0.4625 & 0.3700 \\ 0.0570 & -0.0456 \\ -2.5583 & -3.1741 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#2)} \\
\begin{bmatrix} -0.1466 & 0.0938 \\ 0.0000 & 0.0000 \end{bmatrix} x + \begin{bmatrix} -0.9998 & -0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 0.1333 & -0.0999 \\ -0.0000 & 0.0000 \end{bmatrix} r + \begin{bmatrix} 1.2798 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.5239 & 0.3353 \\ 0.0643 & -0.0411 \\ -0.0168 & 0.0668 \end{bmatrix} x + \begin{bmatrix} 0.0007 & -0.0000 \\ -0.0001 & 0.0000 \\ -0.1143 & -0.0893 \end{bmatrix} u + \begin{bmatrix} 0.4763 & -0.3572 \\ -0.0584 & 0.0438 \\ -1.8949 & -2.5583 \end{bmatrix} r \le \begin{bmatrix} -1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#3)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 1.0000 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} 0.5239 & -0.3353 \\ -0.5210 & 0.3337 \end{bmatrix} x + \begin{bmatrix} -0.0007 & 0.0000 \\ 0.0000 & -0.0005 \end{bmatrix} u + \begin{bmatrix} -0.4763 & 0.3572 \\ 0.4625 & -0.3700 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#4)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} -1.0000 \\ 1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.0643 & 0.0412 \\ -0.0643 & 0.0411 \end{bmatrix} x + \begin{bmatrix} 0.0000 & -0.0001 \\ 0.0001 & -0.0000 \end{bmatrix} u + \begin{bmatrix} 0.0570 & -0.0456 \\ 0.0584 & -0.0438 \end{bmatrix} r \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#5)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} 0.0643 & -0.0411 \\ 0.0643 & -0.0412 \end{bmatrix} x + \begin{bmatrix} -0.0001 & 0.0000 \\ -0.0000 & 0.0001 \end{bmatrix} u + \begin{bmatrix} -0.0584 & 0.0438 \\ -0.0570 & 0.0456 \end{bmatrix} r \le \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#6)} \\
\begin{bmatrix} 0.0000 & 0.0000 \\ 0.1144 & -0.0733 \end{bmatrix} x + \begin{bmatrix} -1.0000 & 0.0000 \\ -0.0000 & -0.9999 \end{bmatrix} u + \begin{bmatrix} 0.0000 & 0.0000 \\ -0.1016 & 0.0813 \end{bmatrix} r + \begin{bmatrix} -1.0000 \\ -0.7804 \end{bmatrix} & \text{if } \begin{bmatrix} 0.0643 & -0.0412 \\ -0.5210 & 0.3337 \\ -0.1251 & 0.0084 \end{bmatrix} x + \begin{bmatrix} -0.0000 & 0.0001 \\ 0.0000 & -0.0005 \\ 0.1465 & 0.1143 \end{bmatrix} u + \begin{bmatrix} -0.0570 & 0.0456 \\ 0.4625 & -0.3700 \\ 2.5583 & 3.1741 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#7)} \\
\begin{bmatrix} -0.1466 & 0.0938 \\ 0.0000 & 0.0000 \end{bmatrix} x + \begin{bmatrix} -0.9998 & -0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} 0.1333 & -0.0999 \\ -0.0000 & 0.0000 \end{bmatrix} r + \begin{bmatrix} -1.2798 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.0643 & 0.0411 \\ 0.5239 & -0.3353 \\ 0.0168 & -0.0668 \end{bmatrix} x + \begin{bmatrix} 0.0001 & -0.0000 \\ -0.0007 & 0.0000 \\ 0.1143 & 0.0893 \end{bmatrix} u + \begin{bmatrix} 0.0584 & -0.0438 \\ -0.4763 & 0.3572 \\ 1.8949 & 2.5583 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ -1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#8)} \\
\begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & -1.0000 \end{bmatrix} u + \begin{bmatrix} -1.0000 \\ -1.0000 \end{bmatrix} & \text{if } \begin{bmatrix} -0.5239 & 0.3353 \\ 0.5210 & -0.3337 \end{bmatrix} x + \begin{bmatrix} 0.0007 & -0.0000 \\ -0.0000 & 0.0005 \end{bmatrix} u + \begin{bmatrix} 0.4763 & -0.3572 \\ -0.4625 & 0.3700 \end{bmatrix} r \le \begin{bmatrix} 1.0000 \\ -1.0000 \end{bmatrix} \quad \text{(Region \#9)}
\end{cases}
$$

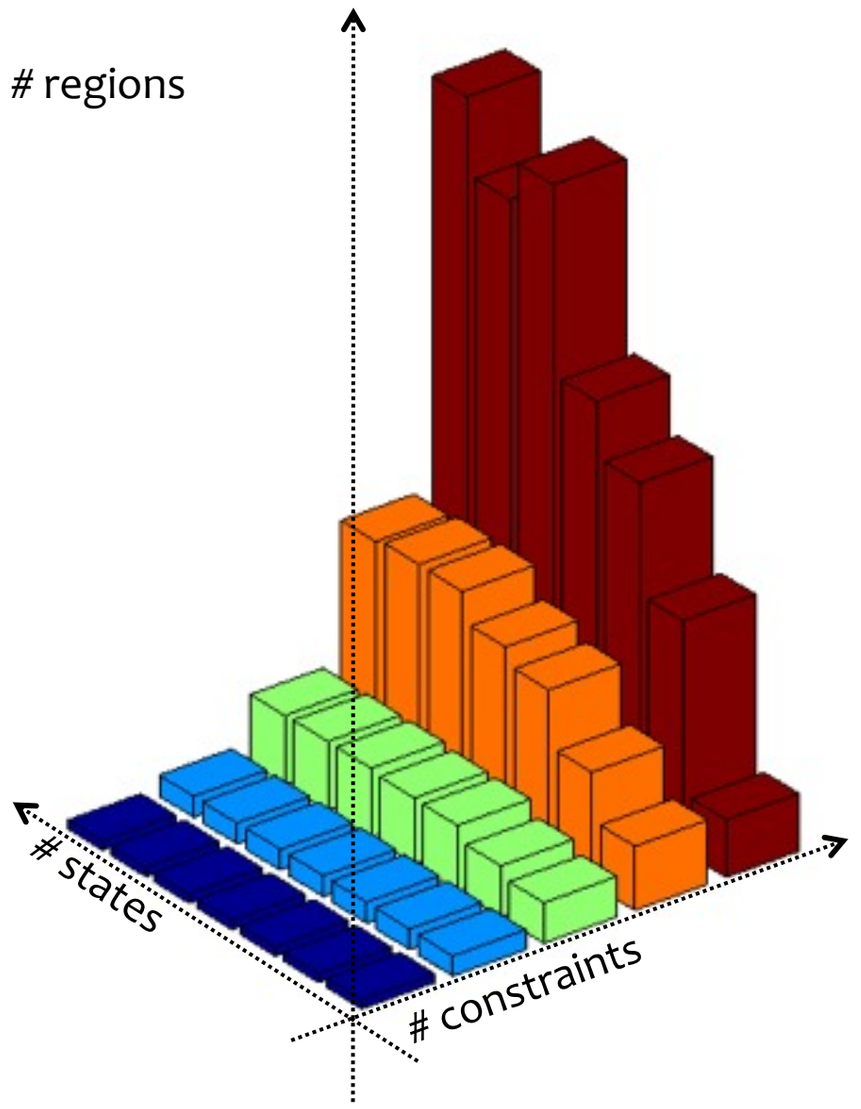
Complexity

- Number of regions $\leq$ # combinations of active constraints

  - Largely depends on #constraints (usually combinatorially)

  - Also depends on #free variables

  - Weakly depends on #states



- Example

average on 20 random SISO systems (input saturation)

states\horizon	$N=1$	$N=2$	$N=3$	$N=4$	$N=5$
n=2	3	6.7	13.5	21.4	19.3
n=3	3	6.9	17	37.3	77
n=4	3	7	21.65	56	114.2
n=5	3	7	22	61.5	132.7
n=6	3	7	23.1	71.2	196.3
n=7	3	6.95	23.2	71.4	182.3
n=8	3	7	23	70.2	207.9

Explicit MPC typically limited to 6-8 free control moves and 8-12 states+references

Complexity - QP vs. Explicit

$2N$	QP (ms) average	worst	explicit (ms) average	worst	regions	[storage kb]
4	1.1	1.5	0.005	0.1	25	16
8	1.3	1.9	0.023	1.1	175	78
20	2.5	2.6	0.038	3.3	1767	811
30	5.3	7.2	0.069	4.4	5162	2465
40	10.9	13.0	0.239	15.6	11519	5598

(Intel Centrino 1.4 GHz)

Average time on 100 random 3D parameters (2N constraints)

Worst-case time on 100 random 3D parameters (2N constraints)

⇒ Need to visit regions more efficiently than linear search

⇒ Need to reduce number of regions

Example: AFTI-16

- Linearized model:

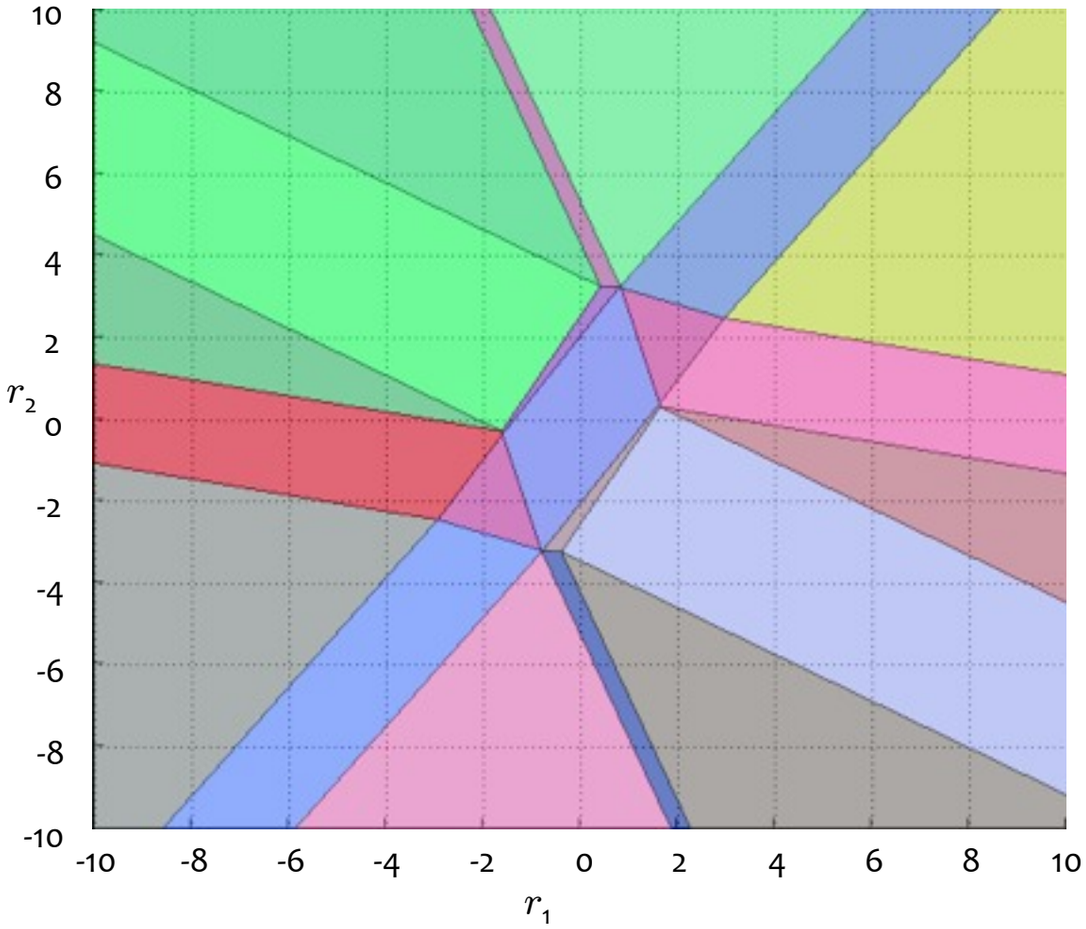
$$\begin{cases} \dot{x} = \begin{bmatrix} -.0151 & -60.5651 & 0 & -32.174 \\ -.0001 & -1.3411 & .9929 & 0 \\ .00018 & 43.2541 & -.86939 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x + \\ \qquad \begin{bmatrix} -2.516 & -13.136 \\ -.1689 & -.2514 \\ -17.251 & -1.5766 \\ 0 & 0 \end{bmatrix} u \\ y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x, \end{cases}$$

- Inputs: elevator and flaperon angle
- Outputs: attack and pitch angle
- Sampling time: $T_s = .05$ s (+ zero-order hold)
- Constraints: max 25° on both angles
- Open-loop response: unstable
  (open-loop poles: $-7.6636, -0.0075 \pm 0.0556j, 5.4530$)

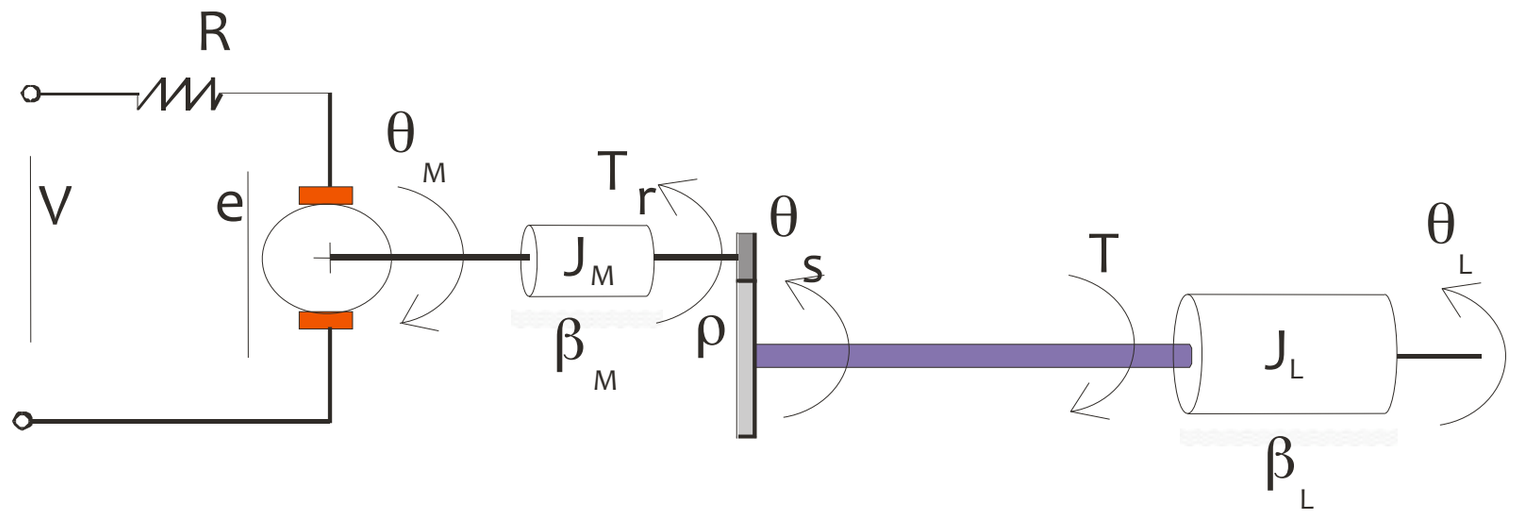
Explicit controller: 8 parameters, 51 regions

go to demo **`linear/afti16.m`** (Hyb-Tbx)

Section with x=[o o o o]', u=[o o]'



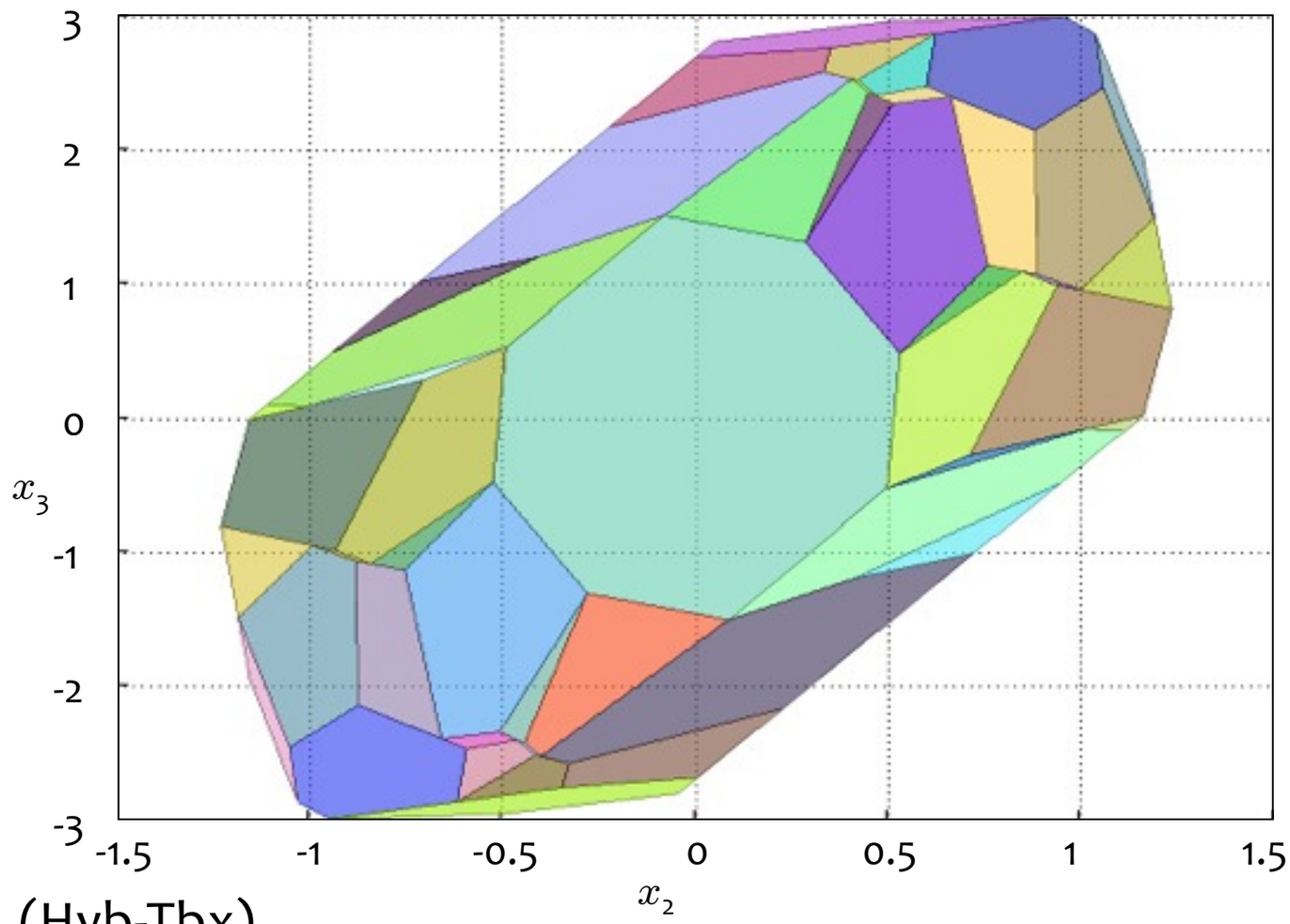
MPC of a DC Servomotor



$$
\begin{aligned}
N &= 10 \\
N_u &= 2 \\
w_y &= \{1000, 0\} \\
w_{\delta u} &= .05 \\
u &\in [-220, 220] \text{ V} \\
y_2 &\in [-78.5398, 78.5398] \text{ Nm}
\end{aligned}
$$

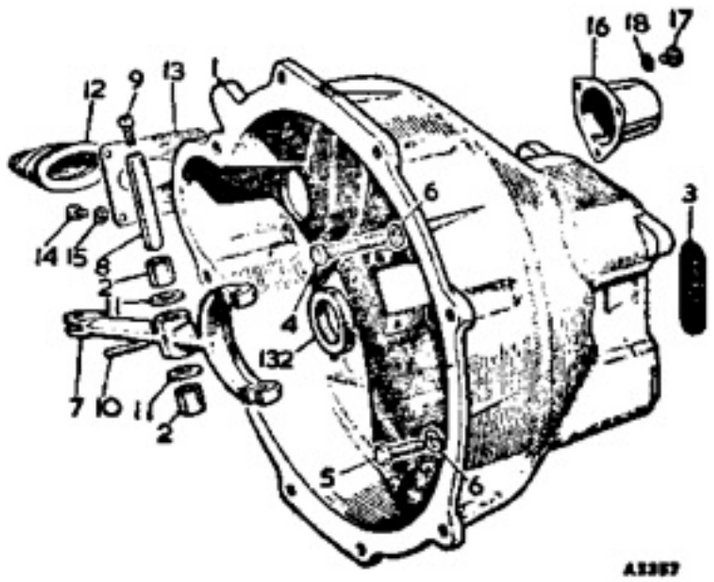
Explicit controller:
7 parameters, 101 regions

Section with x_1=x_4=0, r=[0 0]', u=0



go to demo **`linear/dcmotor.m`** (Hyb-Tbx)

Dry Clutch Engagement



(Bemporad, Borrelli, Glielmo, Vasca, 2001)

Slip phase: $\omega_e > \omega_v$

$$\begin{aligned} I_e\dot{\omega}_e &= T_{in} - b_e\omega_e - T_{cl} \\ I_v\dot{\omega}_v &= T_{cl} - b_v\omega_v - T_l \end{aligned}$$

$$T_{cl} = kF_n\text{sign}(\omega_e - \omega_v)$$

Clutch engaged: $\omega_e=\omega_v=\omega$

$$(I_e + I_v)\dot{\omega} = T_{in} - (b_e + b_v)\omega - T_l$$

- Control Objective
  - small friction losses
  - fast engagement
  - driver comfort
- Constraints:
  - clutch force
  - clutch force derivative
  - minimum engine speed

I_e	engine inertia
ω_e	crackshaft rotor speed
T_{in}	engine torque
b_e	crankshaft friction coefficient
T_{cl}	torque trasmitted by clutch
I_v	vehicle moment of inertia
ω_v	clutch disk rotor speed
b_v	clutch friction coefficient
T_l	equivalent load torque

Linear MPC Design

- Linear model during slip

  + disturbance model

$$\begin{aligned} x_1 &\triangleq \omega_e \\ x_2 &\triangleq \omega_e - \omega_v \\ u &\triangleq F_n \end{aligned}$$

$$\begin{aligned} \dot{x}_1 &= -\frac{b_e}{I_e}x_1 + \frac{T_{in}}{I_e} - \frac{k}{I_e}u \\ \dot{x}_2 &= \left(-\frac{b_e}{I_e} + \frac{b_v}{I_v}\right)x_1 - \frac{b_v}{I_v}x_2 + \frac{T_{in}}{I_e} + \frac{T_\ell}{I_v} - k\left(\frac{1}{I_e} + \frac{1}{I_v}\right)u \\ \ddot{T}_{in} &= 0 \\ \dot{T}_\ell &= 0 \end{aligned}$$

- Sampling

  T=10 ms

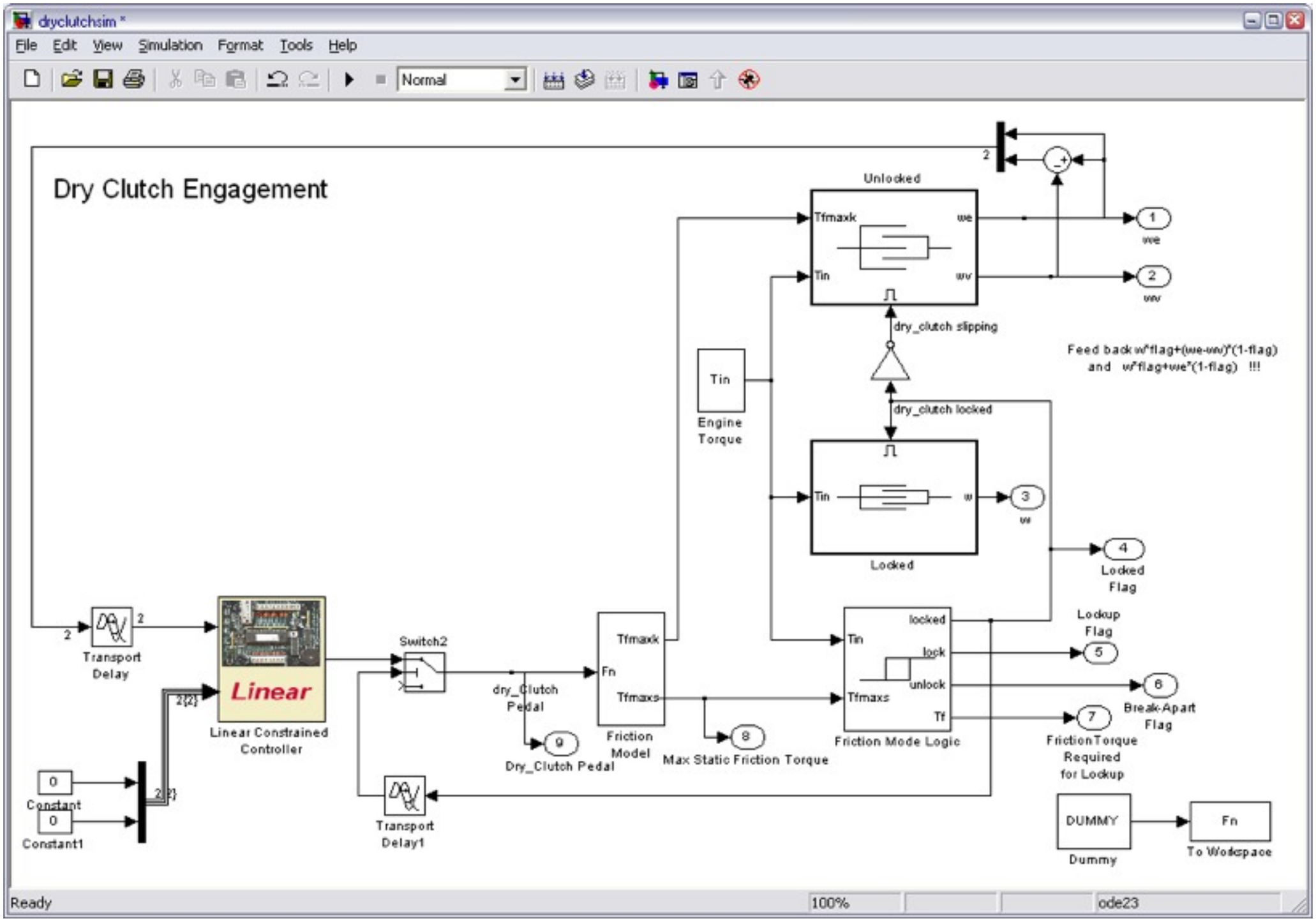
$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ T_{in}(t+1) \\ \dot{T}_{in}(t+1) \\ T_l(t+1) \\ u(t) \end{bmatrix} = \begin{bmatrix} 0.9985 & 0 & 0.0500 & 0.0002 & 0 & -0.0049 \\ -0.0011 & 0.9996 & 0.0500 & 0.0002 & 0.0129 & -0.0062 \\ 0 & 0 & 1.0000 & 0.0100 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ T_{in}(t) \\ \dot{T}_{in}(t) \\ T_l(t) \\ u(t-1) \end{bmatrix} + \begin{bmatrix} -0.0049 \\ -0.0062 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \Delta u(t)$$

- Control objective:

$$\min_{\{\Delta u(t),\ \ldots,\ \Delta u(t+m-1)\}} \sum_{k=0}^{N-1} Q x_2^2(t+k|t) + R\Delta u^2(t+k) + x'(t+N|t)Px(t+N|t)$$

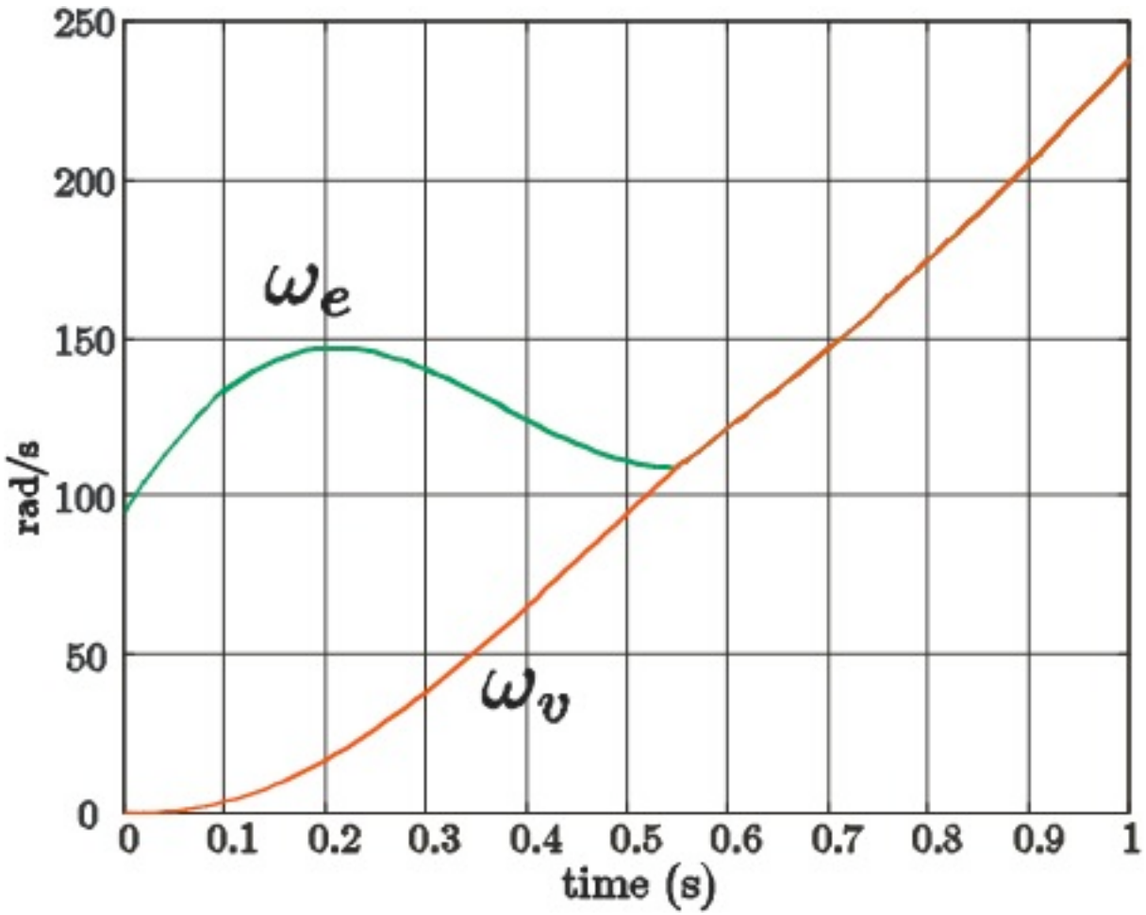
- Constraints: $0 \le \Delta u \le 80$ N, $0 \le u \le 5000$ N, $x_1 \ge 50$ rad/s, $x_2 \ge 0$

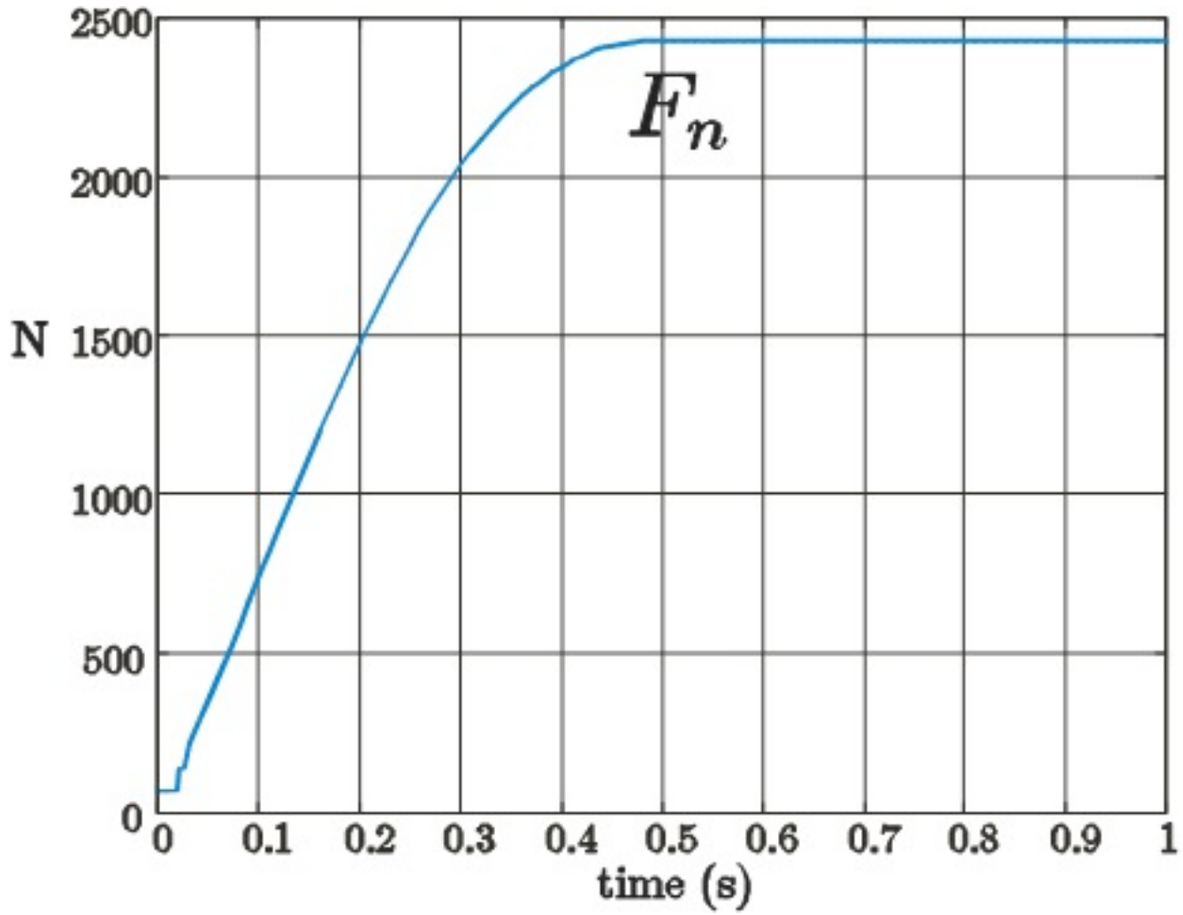
MPC Tuning



go to demo **/demos/dryclutch/dryclutch.m** (Hyb-Tbx)

MPC simulation





Controller	τ^*	$E_n(\tau^*)$	E_d	$\dot{x}_2(\tau^*)$
1	0.57 s	2333 Nm	6676 J	230 rad/s^2
2	0.76 s	2515 Nm	9777 J	109 rad/s^2
3	0.83 s	2572 Nm	10838 J	62 rad/s^2

Choice:

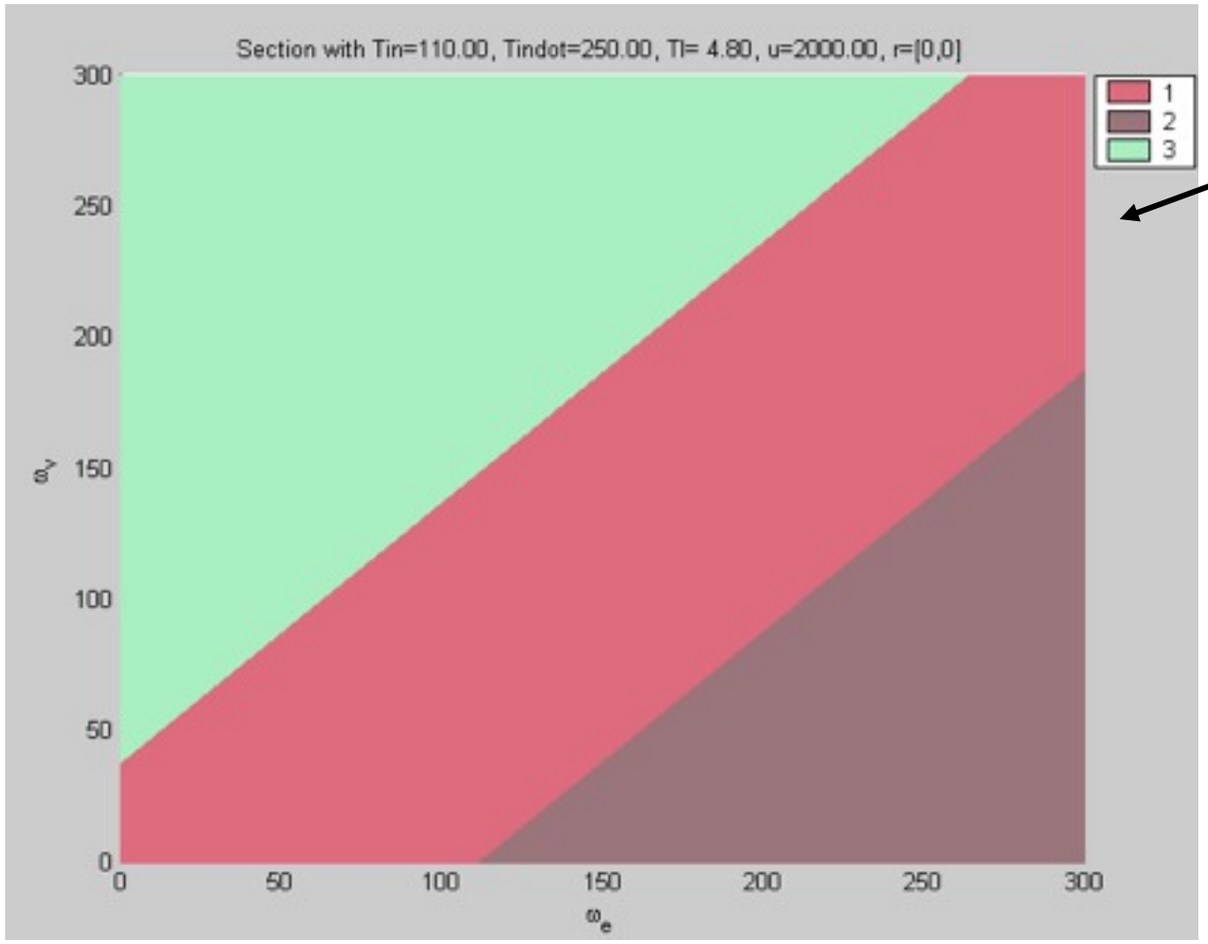
Q=2, R=1

N=10, N_u=1

MPC Controller - Explicit Solution

$$\Delta u(\theta) = \begin{cases} [-0.003786\ 0.5396\ 0.1703\ 0.006058\ 0.04411]\,x + \\ -0.02101\,u + [0\ -0.5409]\,r & \text{if } \begin{bmatrix} -4.732e{-}005 & 0.006745 & 0.002129 & 7.572e{-}005 & 0.0005513 \\ 0.003786 & -0.5396 & -0.1703 & -0.006058 & -0.04411 \end{bmatrix} x + \\ & \begin{bmatrix} -0.0002626 \\ 0.02101 \end{bmatrix} u + \begin{bmatrix} 0 & -0.006762 \\ 0 & 0.5409 \end{bmatrix} r \le \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ & \text{(Region \#1)} \\ 80 & \text{if } [4.732e{-}005\ -0.006745\ -0.002129\ -7.572e{-}005\ -0.0005513]\,x + \\ & 0.0002626\,u + [0\ 0.006762]\,r \le [-1] \\ & \text{(Region \#2)} \\ 0 & \text{if } [-0.003786\ 0.5396\ 0.1703\ 0.006058\ 0.04411]\,x + \\ & -0.02101\,u + [0\ -0.5409]\,r \le [0] \\ & \text{(Region \#3)} \end{cases}$$

+ Linear Observer



ω_v rad/s

ω_e rad/s

T_{in}=110 Nm, $\dot{T}_{in}$=250 Nm/s
T_l=4.8 Nm, F_n=2000 Nm, r=[0 0]'

Alternative: explicit **hybrid** MPC

- Switching model (slipping mode, engaged mode)
- Design an explicit MPC controller for the switched system

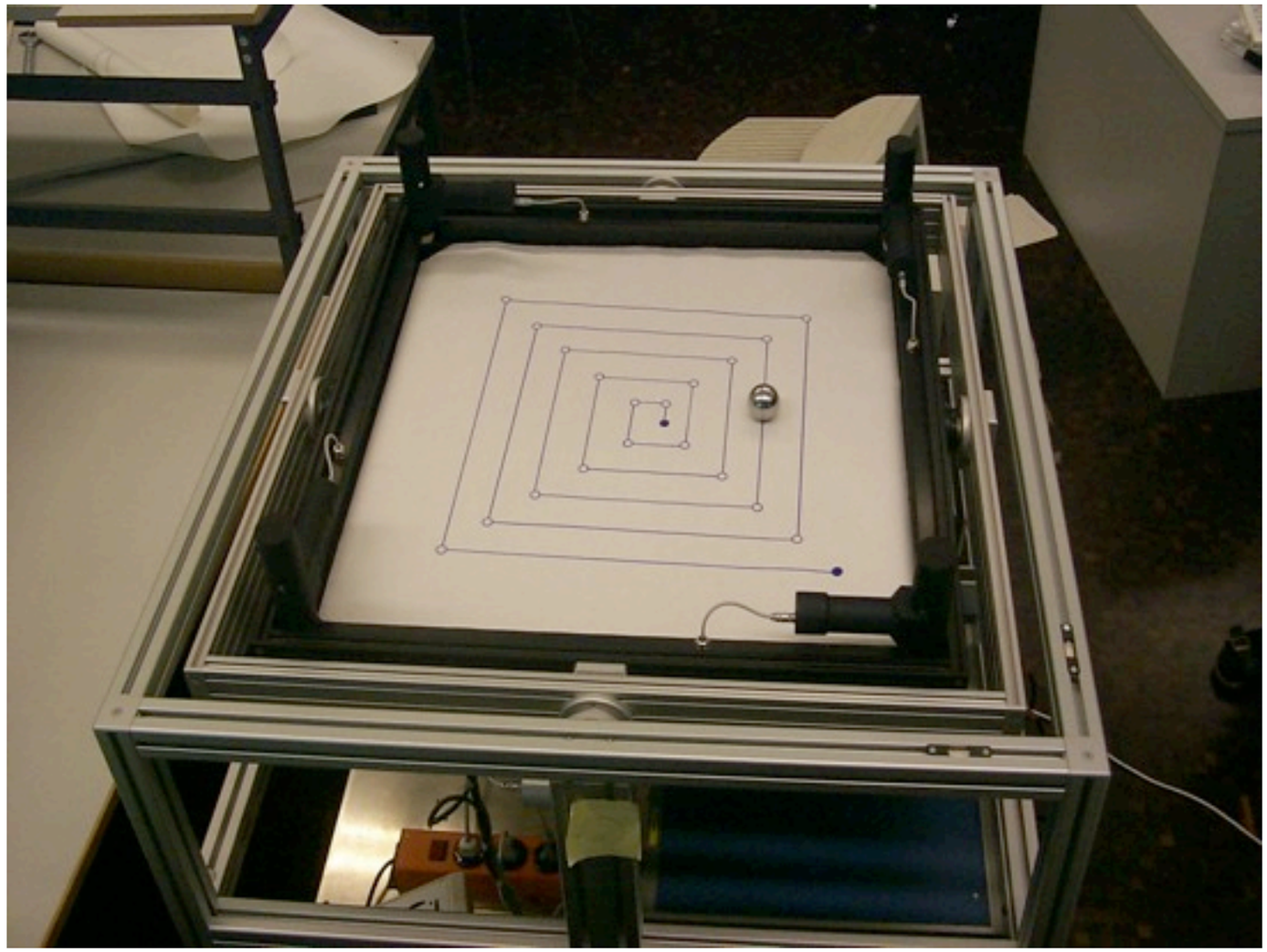
MPC Regulation of a Ball on a Plate

Task:

- Tune an MPC controller by simulation, using the ***MPC Simulink Toolbox***

- Get the ***explicit solution*** of the MPC controller.

- Validate the controller on ***experiments***.



Ball & plate: experimental setup

Host

Ball & Plate System

TCP/IP

Code

Data

Ball & Plate

Target

u_1, u_2

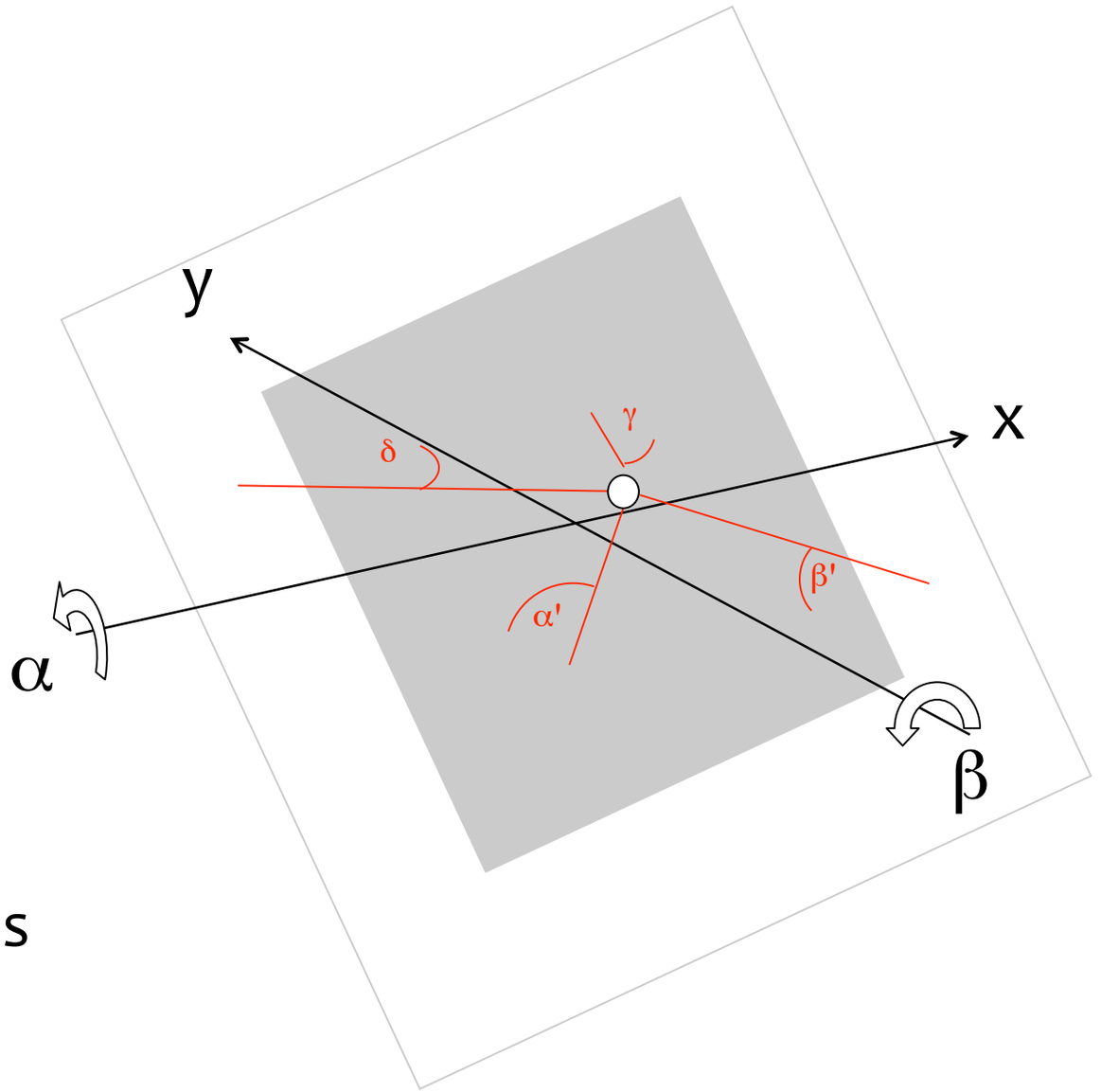
a,b

x, y

Ball & plate: model and specifications

- Specifications:

  Angle: -17 deg … +17deg
  Plate: -30 cm …+30 cm
  Input Voltage: -10 V… +10 V
  Computer: PENTIUM166
  Sampling Time:30 ms



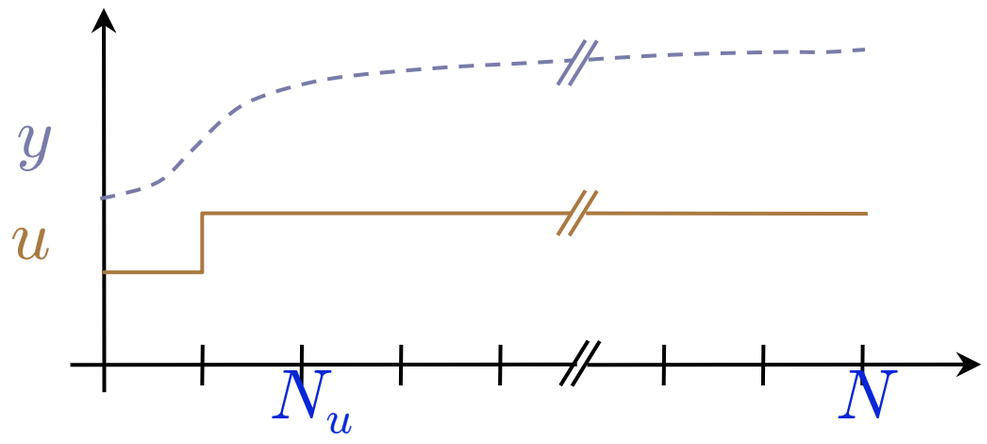
- Model: LTI 14 states
  Constraints on inputs and states

MPC Tuning

Sampling time: T_s = **30** ms

Prediction horizon: N = **50**

Free control moves: N_u = **2**



Output constraint horizon: **1** (*soft constraint*)

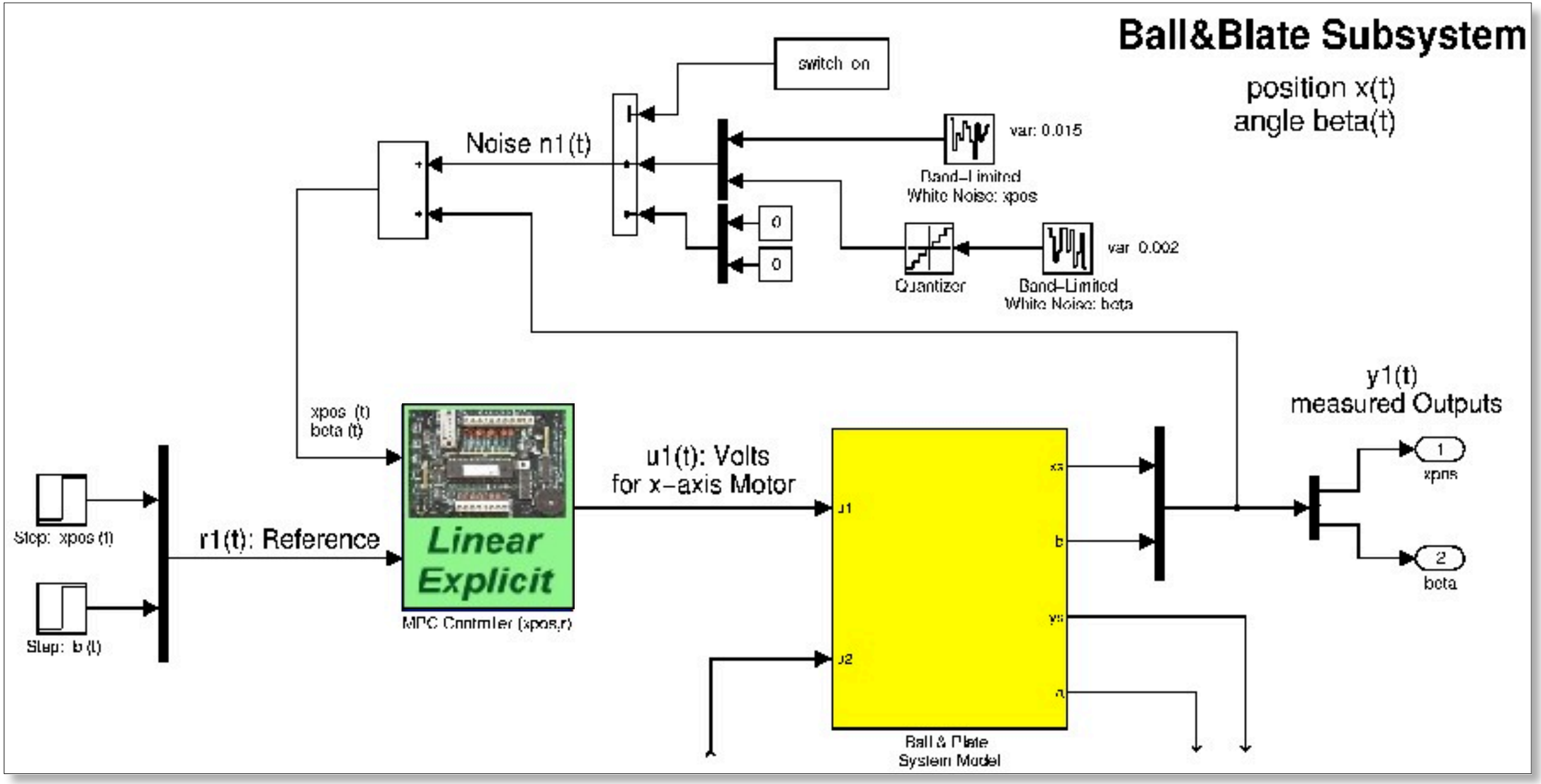
Input constraint horizon: **1** (*hard constraint*)

Weight on position error: **5**

Weight on input voltage changes: **1**

Implementation

- Solve mp-QP and implement explicit MPC

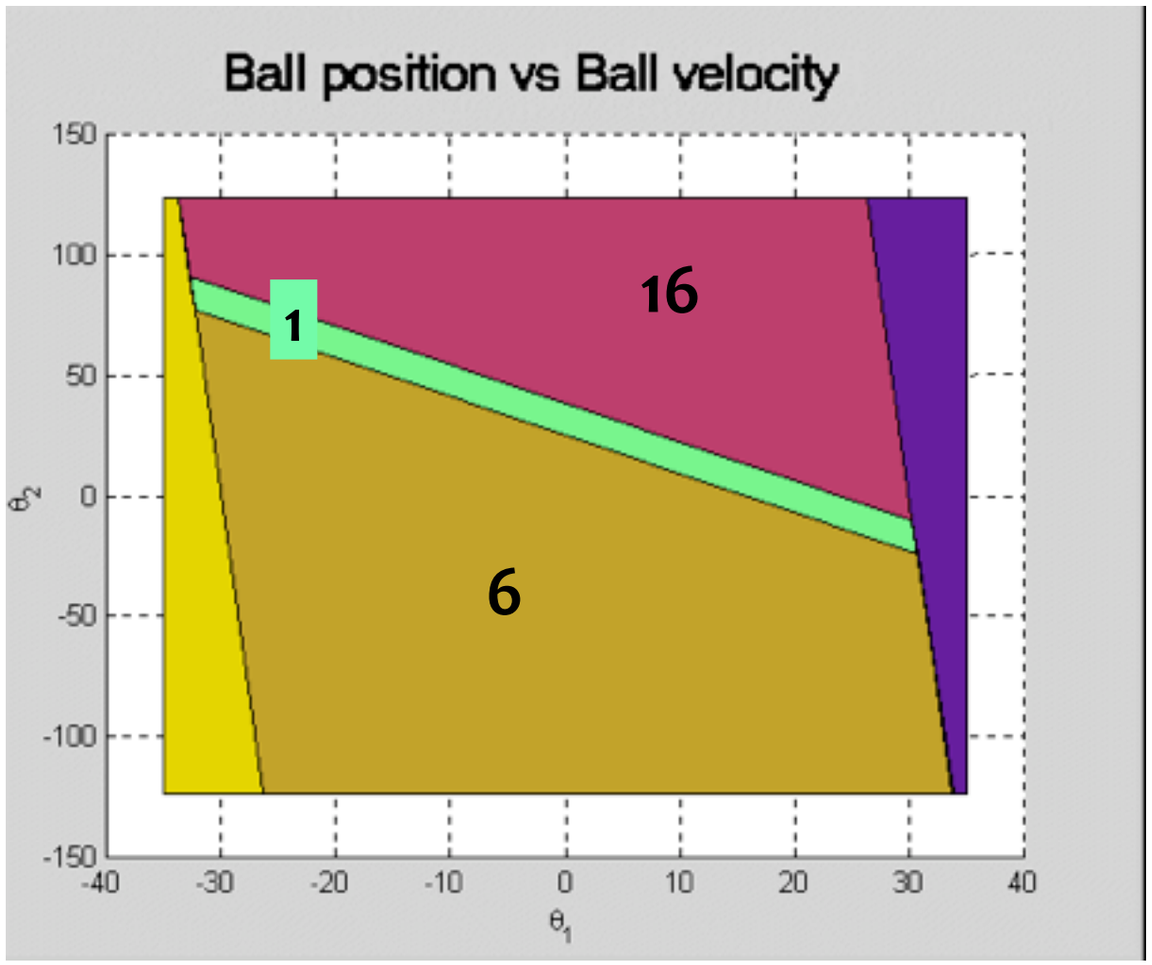


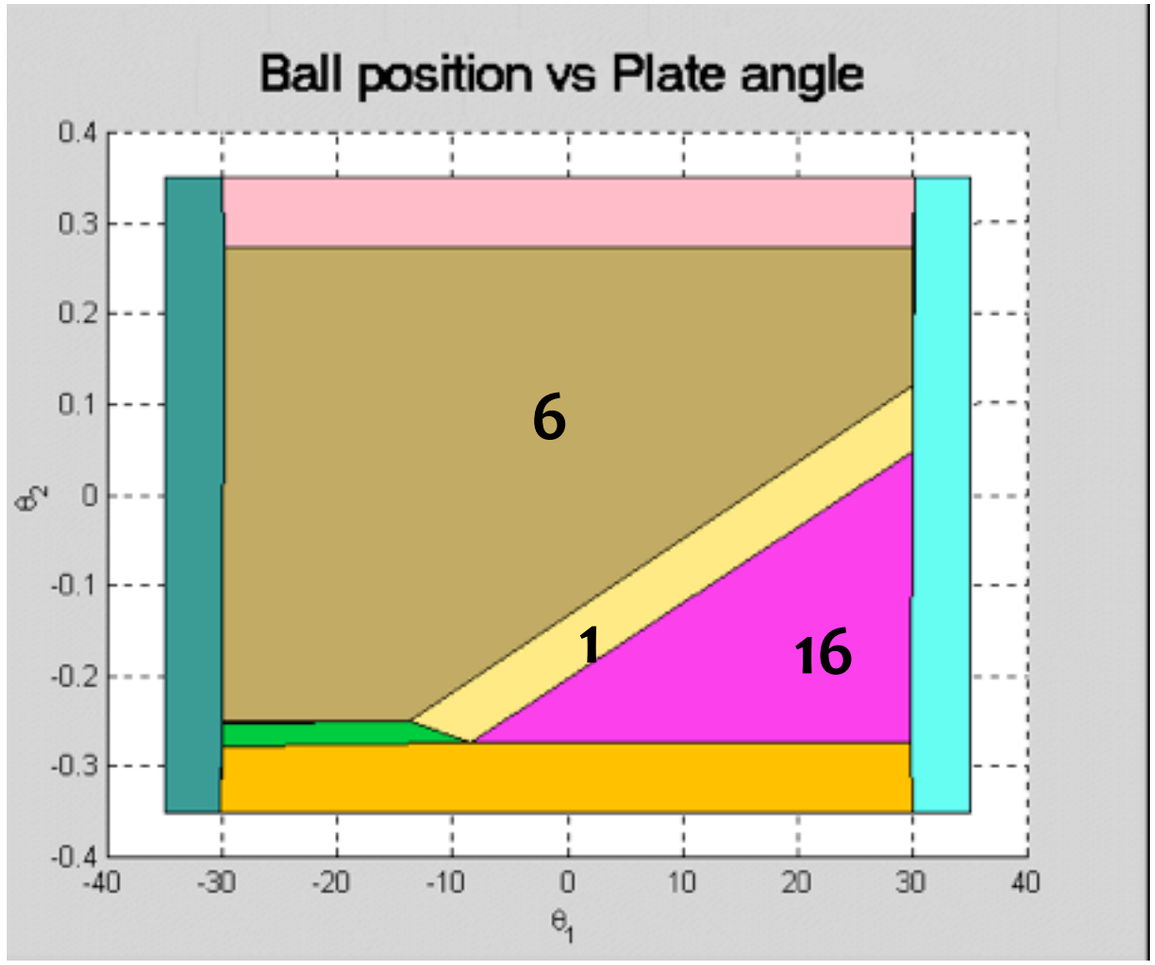
E.g: Real-Time Workshop + xPC Toolbox

Explicit MPC Solution

Controller: x: 22 Regions, y: 23 Regions

x-MPC: sections at α_x=0, α_x=0°, u_x=0, r_x=18, r_α=0



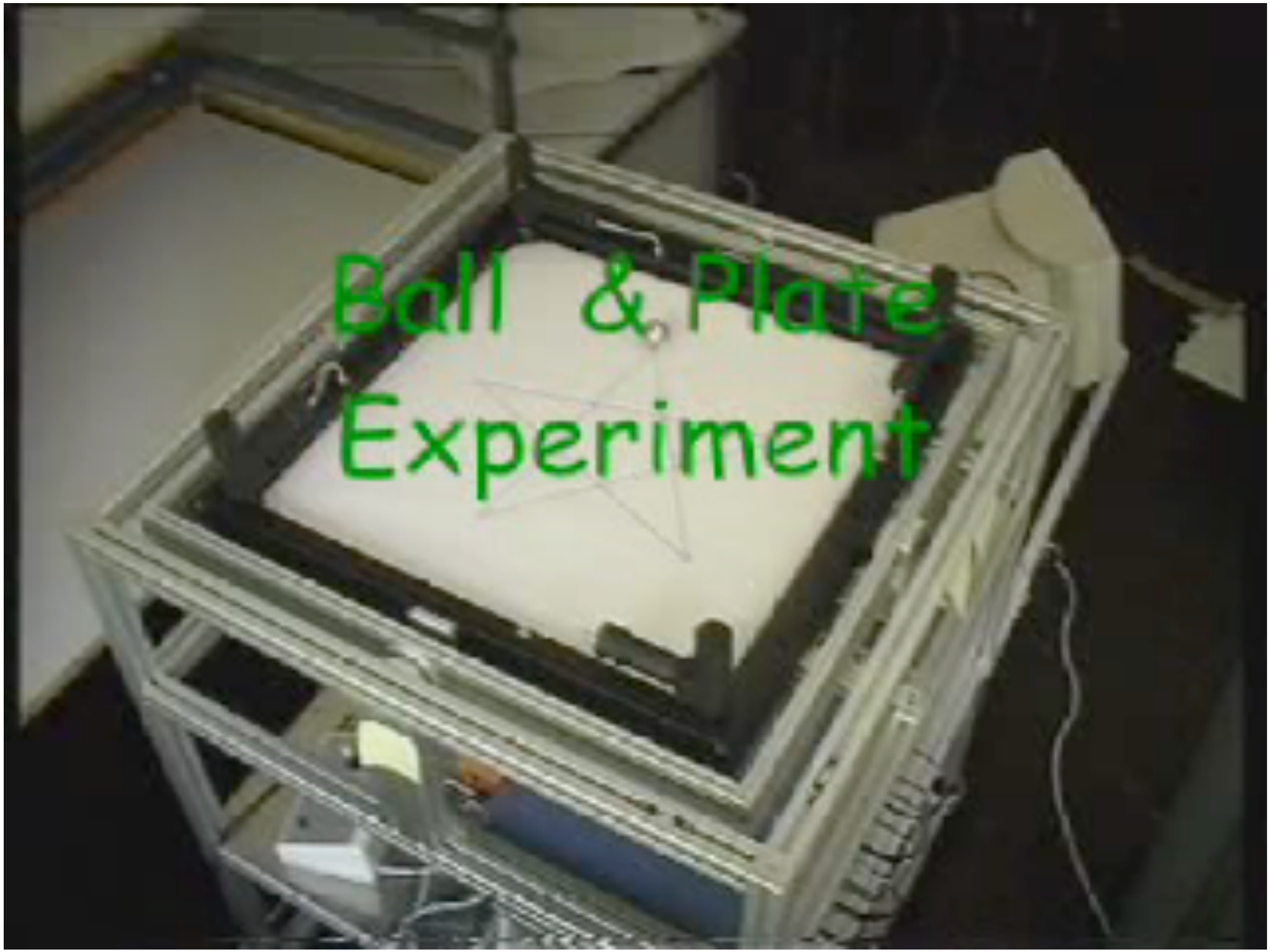


Region 1: LQR Controller (near equilibrium)

Region 6: Saturation at -10

Region 16: Saturation at +10

Ball and plate experiment



Ball and plate experiment

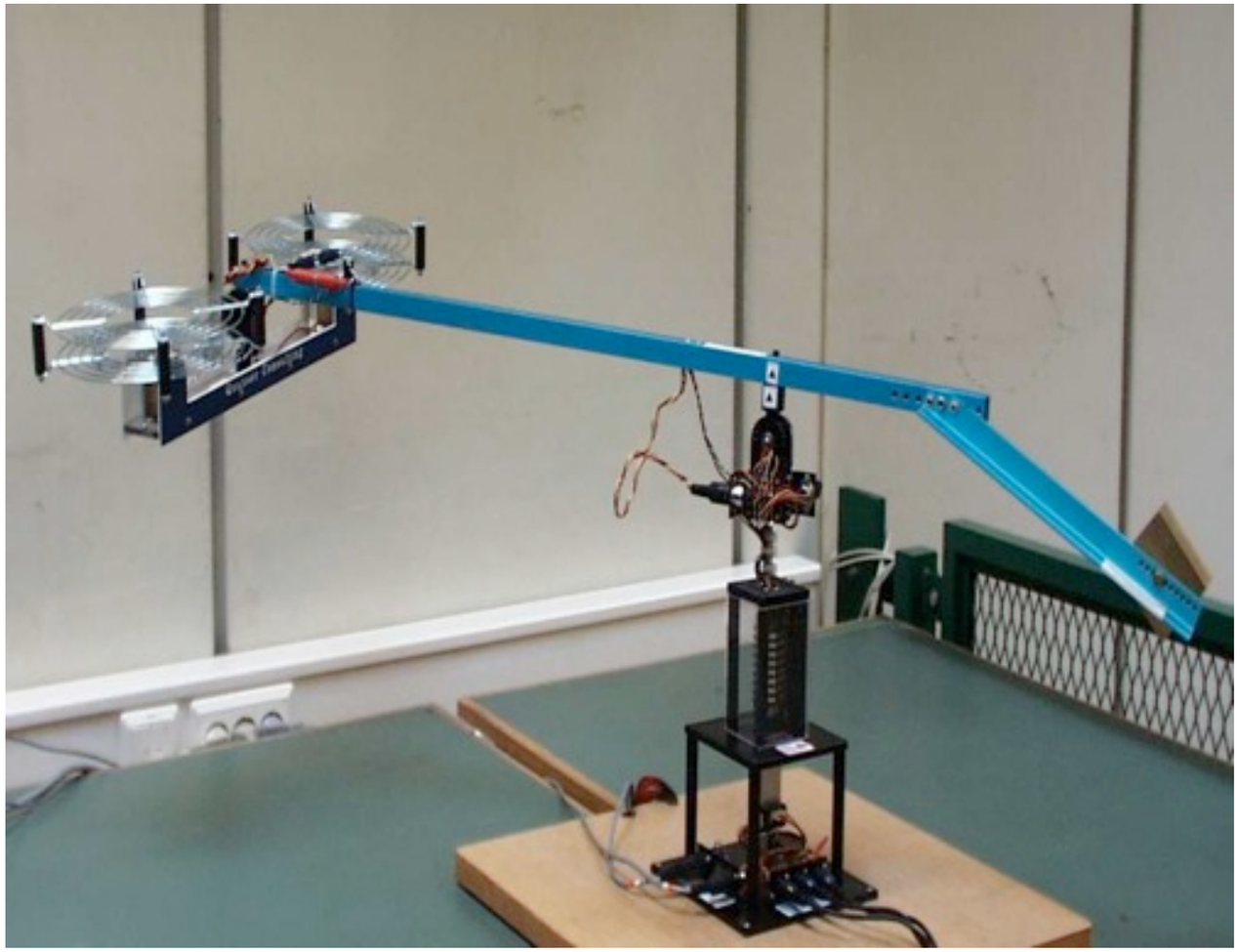
Ball and plate experiment in LEGO, using explicit MPC and Hybrid Toolbox

(by Daniele Benedettelli, Univ. of Siena, July 2008)

- 20Hz sampling frequency
- camera used for position feedback
- explicit MPC coded using integer numbers

Example - Laboratory Helicopter

(Tøndel, 2003)



6 states, 2 inputs

Upper/lower constraints on both inputs and two of the states.

Size of mp-QP with N=3:

dim(U)=6

dim(x)=6

24 constraints

Off-line computation times:

Horizon	Time (s)	# polyhedra
1	1	33
2	19	395
3	163	2211

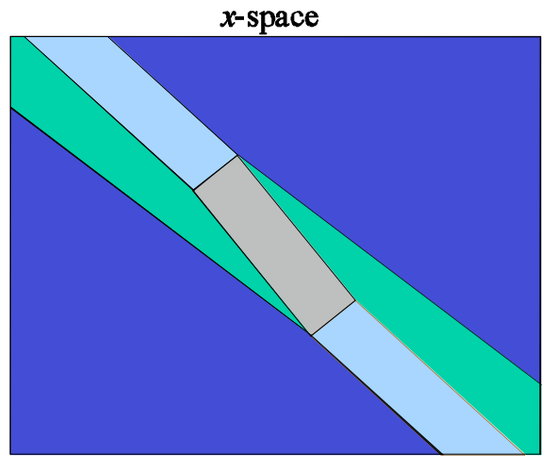
On-line computation times:

Horizon	Implicit solution	Explicit solution
1	19 ms	0.3 ms
2	20 ms	9 ms
3	25 ms	52 ms

Comments on Explicit MPC

- Multiparametric Quadratic Programs (mp-QP) can be solved efficiently
- Model Predictive Control (MPC) can be solved off-line via mp-QP
- Explicit solution of MPC controller $u=f(x)$ is Piecewise Affine

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



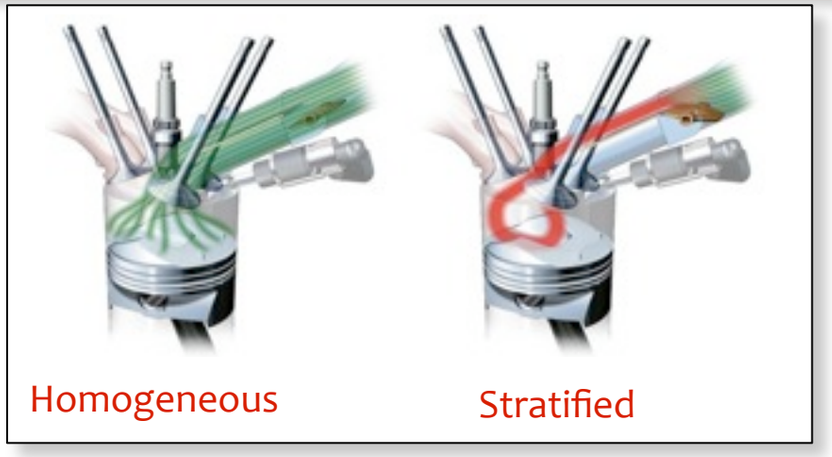
⇒ **Eliminate heavy on-line computation for MPC**

⇒ **Make MPC suitable for fast/small/cheap processes**

Automotive applications of explicit MPC

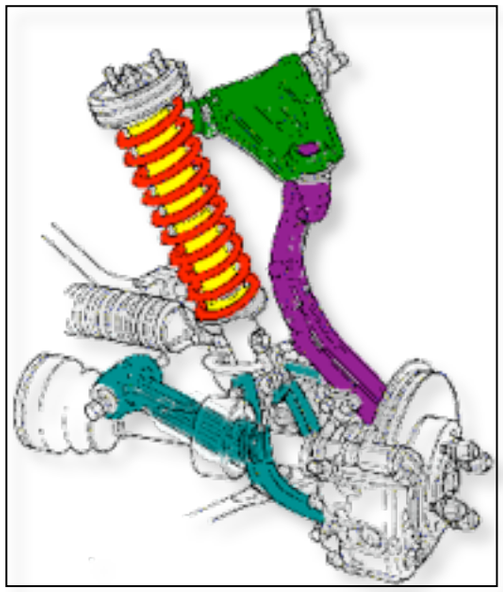
traction control

(Borrelli, Bemporad, Fodor, Hrovat, 2001)



engine control

(N.Giorgetti,G. Ripaccioli, AB, I. Kolmanovsky, D.Hrovat, 2006)

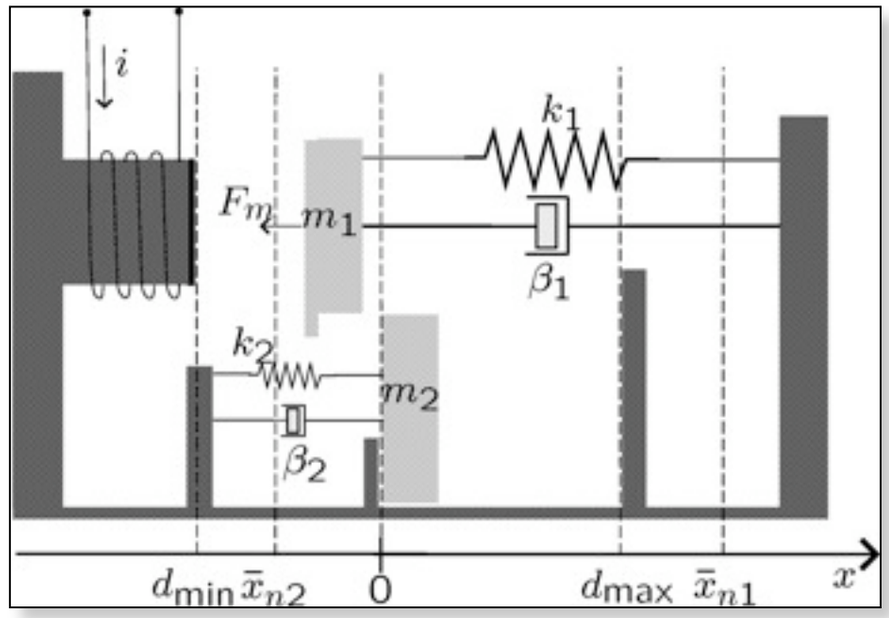


semiactive suspensions

(Giorgetti, Bemporad, Tseng, Hrovat, 2005)

idle speed control

(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2008)



magnetic actuators

(Di Cairano, Bemporad, Kolmanovsky, Hrovat, 2006)

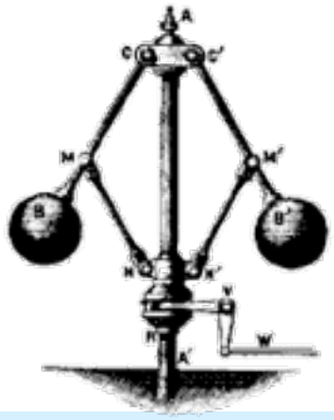
Explicit MPC for idle speed control

(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2008)

- Ford pickup truck, V8 4.6L gasoline engine
- Process:
  - *1 output* (engine speed) to regulate
  - *2 inputs* (airflow, spark advance)
  - input *delays*

- Objectives and specs:
  - *regulate engine speed* at constant rpm
  - *saturation* limits on airflow and spark
  - *lower bound* on engine speed ($\geq$450 rpm)

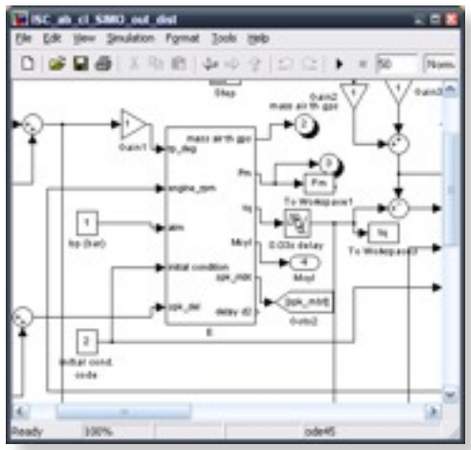
- Related to most classical problem in control: Watt's governor (1787)



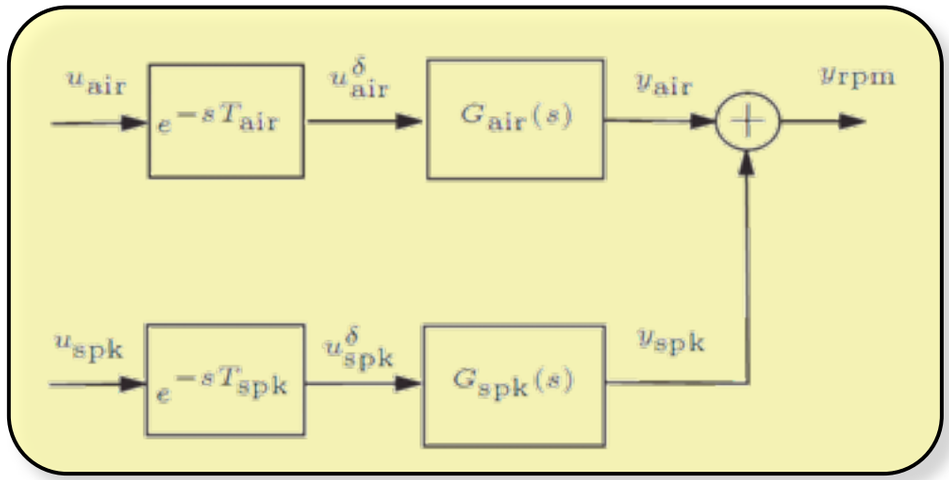
- Problem suitable for MPC design (Hrovat, 1996)

Explicit MPC Design Flow

NL Simulink model



prediction model



control specs

add integral action

MPC setup (on-line QP)

multi-parametric solver

```
#define EXPCON_NTH 2
#define EXPCON_NH 16
#define EXPCON_NF 5
#define EXPCON_NYM 2
static double EXPCON_F[]={
    -6.83553,-12.0296,0,0,-26.9062,-6
    -154.828};
```

PWA control law (C-code)

closed-loop simulation

identification

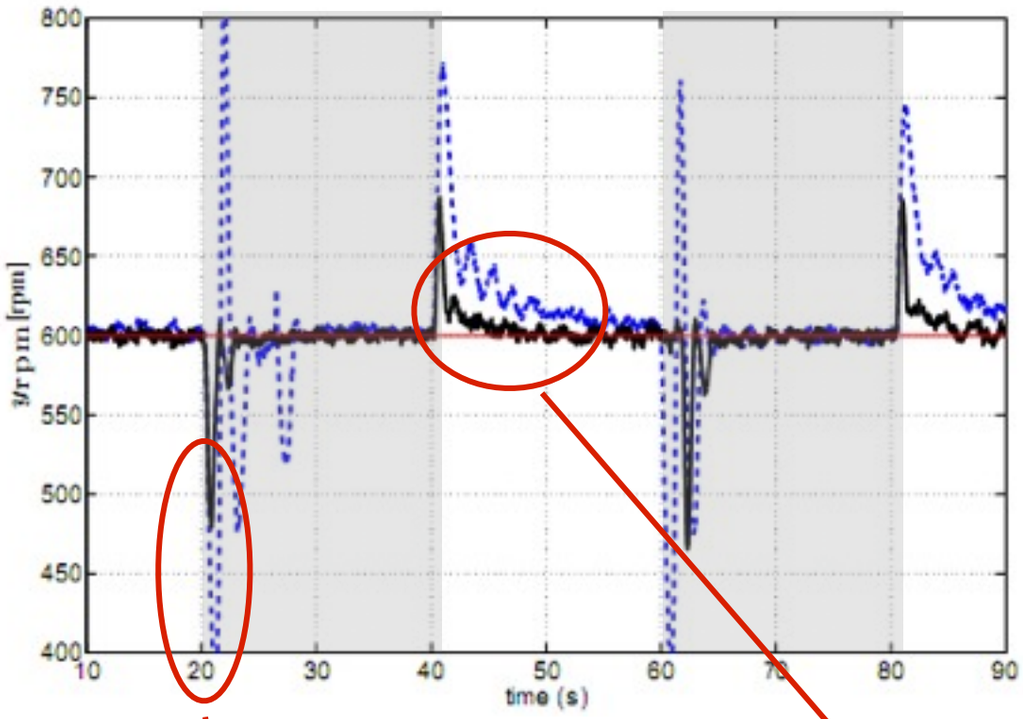
revise **weights** & **observer**

experiments

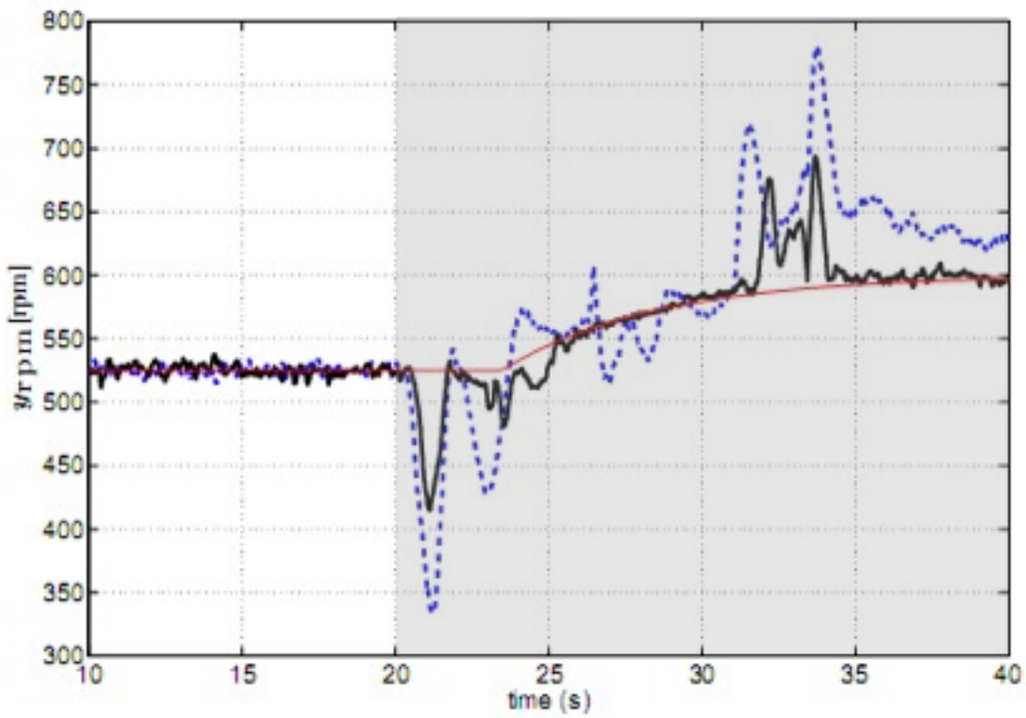
Matlab tool:
Hybrid Toolbox

Explicit MPC for idle speed - Experiments

(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2008)



Load torque (power steering)



Power steering + air conditioning

peak reduced by 50%

convergence 10s faster

explicit MPC

baseline controller (linear)

set-point

mpQP in portfolio optimization

(Bemporad, NMPC plenary, 2008)

Markowitz portfolio optimization

$$\begin{aligned} \min \quad & z'\Sigma z \\ \text{s.t.} \quad & p'z \geq x \\ & [1 \ \ldots \ 1]\, z = 1 \\ & z \geq 0 \end{aligned}$$

z_i = money invested in asset i

p_i = expected return of asset i

Σ_{ij} = covariance of assets i, j

x = expected minimum return of portfolio

Objective: minimize variance (=risk)

Constraint: guarantee a minimum expected return

mpQP in portfolio optimization

Multiparametric QP solution

$$\begin{aligned} \min \quad & z'\Sigma z \\ \text{s.t.} \quad & p'z \geq x \\ & [1 \ \ldots \ 1]\, z = 1 \\ & z \geq 0 \end{aligned}$$

(Bemporad, 2008)