Explicit Model Predictive Control
via Multiparametric Programming

MPC of linear systems

linear model $\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$

$R = R' \succ 0$
$Q = Q' \succeq 0$
$P = P' \succeq 0$

performance index $\min_U x_N' P x_N + \sum_{k=0}^{N-1} x_k' Q x_k + u_k' R u_k$ $\quad x_0 = x(t)$

$$\begin{array}{ll} \min_U & \frac{1}{2}U'HU + x'(t)F'U + \cancel{\frac{1}{2}x'(t)Yx(t)} \\ \text{s.t.} & GU \leq W + Sx(t) \end{array}$$

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

constraints $\begin{cases} u_{\min} \leq u_k \leq u_{\max} \\ y_{\min} \leq Cx_k \leq y_{\max} \end{cases}$

MPC implemented by solving a **(convex) Quadratic Program (QP)**

Routinely used in the process industries

MPC based on convex piecewise affine costs

(Propoi, 1963)
(Bemporad, Borrelli, Morari, 2003)

linear model $\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$
$x_0 = x(t)$

performance index $\min_z \|Px_N\|_\infty + \sum_{k=0}^{N-1} \|Qx_k\|_\infty + \|Ru_k\|_\infty$

$$\begin{array}{ll} \min_z & \begin{bmatrix} 1 & \dots & 1 | & 0 & \dots & 0 \end{bmatrix}' \cancel{z} \\ \text{subj. to} & Gz \leq W + Sx(t), \end{array}$$

$$z = \begin{bmatrix} \epsilon_0^u \\ \vdots \\ \epsilon_{N-1}^u \\ \epsilon_1^x \\ \vdots \\ \epsilon_N^x \\ \hline u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

constraints $\epsilon_k^x \geq \|Qx_k\|_\infty, \ \epsilon_k^u \geq \|Ru_k\|_\infty, \ \epsilon_N^x \geq \|Px_N\|_\infty$

$u_{\min} \leq u_k \leq u_{\max}$ (more generally: $u_k \in \mathcal{U}$, $\mathcal{U}$=polyhedron)

$y_{\min} \leq y_k \leq y_{\max}$ (more generally: $y_k \in \mathcal{Y}$, $\mathcal{Y}$=polyhedron)

MPC implemented by solving a **Linear Program (LP)**

(holds for **any** sum of convex piecewise affine costs) (Schechter, 1987)

KKT optimality conditions

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

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 exist vectors of **Lagrange multipliers** $\lambda \in \mathbb{R}^m$, $\nu \in \mathbb{R}^p$ such that

$$\begin{array}{rcl} \nabla f(U^*) + \sum_{i=1}^m \lambda_i \nabla g_i(U^*) + \sum_{j=1}^p \nu_j \nabla h_j(U^*) & = & 0 \\ \lambda_i g_i(U^*) & = & 0, \quad \forall i = 1, \dots, m \\ \lambda_i & \geq & 0 \\ g_i(U^*) & \leq & 0 \\ h_j(U^*) & = & 0, \quad \forall j = 1, \dots, p \end{array}$$

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

KKT conditions for QP

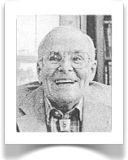
$$\min_U \quad f(U) \triangleq \frac{1}{2}U'HU + c'U$$
$$\text{s.t.} \quad AU \le b$$

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

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

$$HU + c + A'\lambda = 0$$
$$\lambda_i(A_i'U - b_i) = 0$$
$$\lambda \ge 0$$
$$AU - b \le 0$$

Harold W. Kuhn
(1925 -)



Albert W. Tucker
(1905 - 1995)

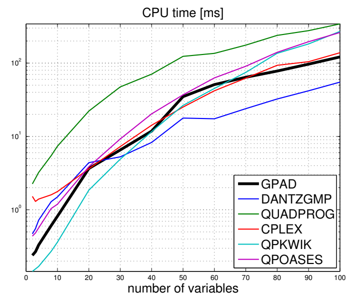
William Karush
(1917 - 1997)

W. Karush, "Minima of functions of several variables with inequalities as side constraints", Master's thesis, Dept. of Mathematics, Univ. of Chicago, 1939

Solution methods for QP

Most used algorithms for solving QP problems:

- ***active set*** *methods* (small/medium size)
- ***interior point*** *methods* (large scale)
- *gradient projection*
- *conjugate gradient*
- *augmented Lagrangian*
  (and *alternating direction method of multipliers*)



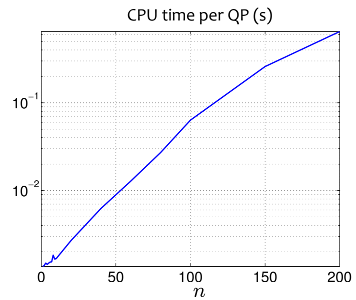
Average CPU time on random QP problems with n variables and $2n$ constraints

Solvers: see **http://plato.asu.edu/guide.html**

```
>>x=qpsol(Q,f,A,b,VLB,VUB,x0,solver)
```
(Hyb-Tbx)

Linear MPC - Numerical example

- Linear MPC of random square MIMO systems
  - n outputs, n inputs, $3n$ states
  - prediction horizon N=10, control horizon N_u=2
  - constraints: $-1 \le u_k \le 1,\ \ -1 \le y_k \le 1$
  - QP size: $(mn+1)$ variables, $(2Nn+2mn)$ constraints



n	#vars	# constraints	CPU time (s)
1	3	24	0.00136
5	11	120	0.00149
20	41	480	0.00270
100	201	2400	0.06432
150	301	3600	0.25873
200	401	4800	0.64981

Macbook Air 2.13 GHz (this mac !)
Inter Core 2 Duo 4GB RAM
MPC Toolbox 4.0, MATLAB R2011b
New active set QP QPKWIK in EML (dense matrices)

Large-scale system with **200** inputs and **200** outputs w/ constraints: **less than 1 s** !

Pros and cons of on-line optimization

✓ Continuous update of best decision under constraints, reaction to unexpected events (disturbances)

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

✖ Computation time may be too long: ok for large sampling times (>10 ms) but not for fast-sampling applications (< 1 ms).

✖ Requires relatively expensive hardware (microprocessor)

✖ Software complexity: solver code must be embedded

✖ Real-time: Worst-case CPU time often hard to estimate

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

On-Line vs off-line optimization

$$\min_U \ \frac{1}{2}U'HU + x'(t)FU + \frac{1}{2}x'(t)Yx(t)$$
$$\text{s.t.} \quad GU \le W + Sx(t)$$

- **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

$$\min_U \ h(U, x)$$
$$\text{s.t.} \quad g(U, x) \le 0$$

and a set X of parameters, determine:

- the **set of feasible parameters** $X^*$of all x∈X for which the problem admits a solution (that is $g(U,x)$≤0 for some U)
- the **value function** $V^* : X^* \to \mathbb{R}$ associating the optimal value $V^*(x)$ to each x
- An **optimizer function** $U^* : X^* \to \mathbb{R}^s$

Multiparametric quadratic programming

(Bemporad et al., 2002)

$$\min_U \ \frac{1}{2}U'HU + x'FU + \cancel{\frac{1}{2}x'Yx}$$
$$\text{s.t.} \quad GU \le W + Sx$$

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

- ***Objective***: solve the QP off line **for all** x∈X to get the optimizer function U^*, and therefore the MPC control law $u(x)$=$[I\ 0\ \dots\ 0]U^*(x)$ **explicitly**

- Assumptions:

  $\begin{bmatrix} H & F \\ F' & Y \end{bmatrix} \succeq 0$ always satisfied if mpQP comes from an MPC problem !

  $H = H' \succ 0$ always satisfied if weight matrix R>0

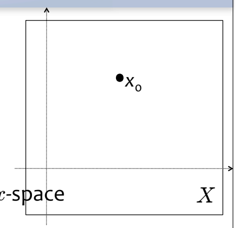
Linearity of solution

Fix $x_0 \in X$

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

⇨ identify active constraints at $U^*(x_0)$

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



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):

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

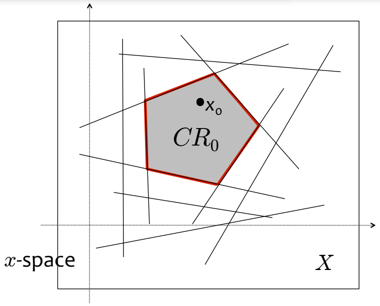
➡ In some neighborhood of x_0, λ and U are explicit affine functions of x !

(Zafiriou, 1990)

Multiparametric QP algorithm

- Impose primal and dual feasibility:

$$\tilde{G}U(x) \leq \tilde{W} + \tilde{S}x \quad \text{from (4)}$$
$$\tilde{\lambda}(x) \geq 0 \quad \text{from (5)}$$



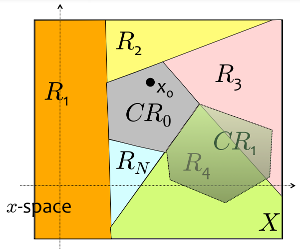
linear inequalities in x !

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

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

- 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 solver #1



Method #1: Split and proceed iteratively

(Bemporad, Morari, Dua, Pistikopoulos, 2002)

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

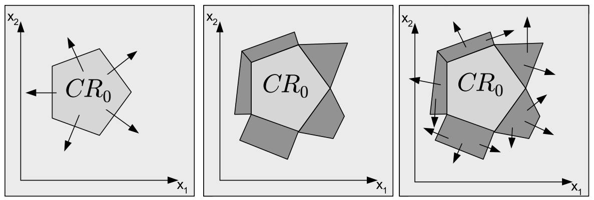
$$R_i = \{x \in X : \mathcal{A}^i x > \mathcal{B}^i,\ \mathcal{A}^j x \leq \mathcal{B}^j,\ \forall j < i\}$$

Note: while CR_i is characterizing a set of active constraints, R_i is not

1) Use the above splitting only as a search procedure, don't split the CR
2) Remove duplicates of CR already found

Multiparametric QP solver #2, #3

(Tøndel, Johansen, Bemporad, 2003)



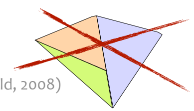
Active set of neighboring region obtained by adding/removing active constraints based on knowledge of the type of crossed hyperplane in x-space

$\tilde{G}^i U(x) \leq \tilde{W}^i + \tilde{S}^i x$ → constraint $\#i$ **added** to active set (to maintain **feasibility** of solution)

$\tilde{\lambda}_j(x) \geq 0$ → constraint $\#j$ **withdrawn** from active set (to maintain **optimality** of solution)

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 $H \succ 0$, $\left[\begin{smallmatrix} H & F \\ F' & Y \end{smallmatrix}\right] \succeq 0$. 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.*

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

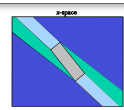
continuous, piecewise affine

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

convex, continuous, piecewise quadratic, C¹ (if no degeneracy)

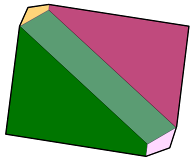
Corollary: The linear MPC controller is a continuous piecewise affine function of the state

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



Proof of theorem

1) X^* is convex and polyhedral



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

Let $x_\alpha = \alpha x_1 + (1-\alpha)x_2 \in X^*$, $x_1, x_2 \in X^*$, $\alpha \in [0,1]$

Let $U_\alpha \triangleq \alpha U^*(x_1) + (1-\alpha)U^*(x_2)$. Vector U_α satisfies the constraints

$$\begin{array}{rcl} GU_\alpha & = & \alpha GU^*(x_1) + (1-\alpha)GU^*(x_2) \\ & \le & \alpha(W + Sx_1) + (1-\alpha)(W + Sx_2) = W + Sx_\alpha \end{array}$$

Therefore $x_\alpha \in X^*$, $\forall x_1, x_2 \in X^*$, $\forall \alpha \in [0,1]$

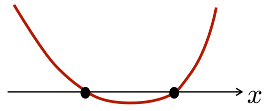
Note: X^* is the projection of a polyhedron onto the parameter space

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

hence X^* is a polyhedron, and therefore convex.

Proof of theorem

2) V^* is a convex function of x



Since U_α satisfies the constraints

$$GU_\alpha \le W + Sx_\alpha$$

by optimality of $V^*(x_\alpha)$ and convexity of $f(U,x) = \frac{1}{2}\left[\begin{smallmatrix} U \\ x \end{smallmatrix}\right]' \left[\begin{smallmatrix} H & F \\ F' & Y \end{smallmatrix}\right] \left[\begin{smallmatrix} U \\ x \end{smallmatrix}\right]$

$$\begin{array}{rcl} V^*(x_\alpha) & \le & f(U_\alpha, x_\alpha) \\ & \le & \alpha f(U^*(x_1), x_1) + (1-\alpha) f(U^*(x_2), x_2) \\ & = & \alpha V^*(x_1) + (1-\alpha)V^*(x_2) \end{array}$$

Proof of theorem

3) Continuity of U^* and V^* with respect to x

Let $U^*(x) = L_i x + M_i$ when $x \in CR_i$. U^* is linear and therefore continuous on the interior of each critical region CR_i.

Consider a parameter x on the boundary between two regions, $x \in CR_i \cap CR_j$. By construction, both $L_i x + M_i$ and $L_j x + M_j$ satisfy the optimality conditions.

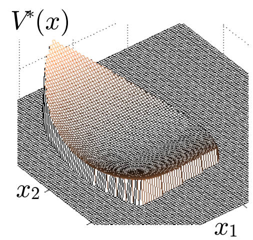
By strict convexity of the optimization problem ($H \succ 0$), the optimum is unique, so $L_i x + M_i = L_j x + M_j$, $\forall x \in CR_i \cap CR_j$.

This proves continuity of U^* across boundaries of critical regions.

As V^* is the composition of two continuous functions, $V^*(x) = f(U^*(x), x)$, it is also continuous. $\square$

Multiparametric convex programming

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



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^* is a convex function of x.*

If f,g_i are also **continuous** in (U,x) then $V^*(x)$ is also continuous in x

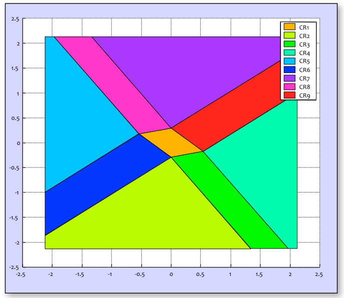
(Mangasarian, Rosen, 1964)

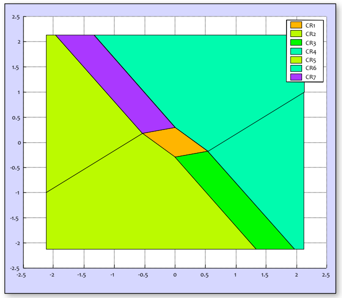
h, g convex and continuous in $(z,x) \Rightarrow V^*(x)$ convex and continuous

V^* and X^* may not be easy to express analytically. Approximate solutions possible

(Bemporad, Filippi, 2003)

Complexity reduction





$$U(x) \triangleq [u_0'(x)\ u_1'(x)\ \dots\ u_{N-1}'(x)]'$$

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

LQ gain

- Control objective: $\min \sum_{k=0}^{\infty} y_k^2 + \frac{1}{100}u_k^2$

$$u_k = K_{LQ} x_k,\ \forall k \ge N_u$$
$$N_u = N = 2$$

$$\Rightarrow \min \left(\sum_{k=0}^{1} y_k^2 + \frac{1}{100}u_k^2 \right) + x_2' \begin{bmatrix} 2.1429 & 1.2246 \\ 1.2246 & 1.3996 \end{bmatrix} x_2$$

solution of algebraic Riccati equation

- Optimization problem

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

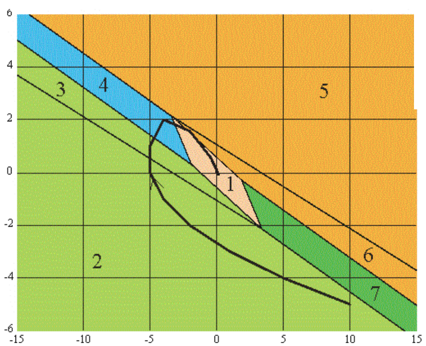
(cost function is normalized by max svd(H))

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

Double integrator example - mp-QP solution

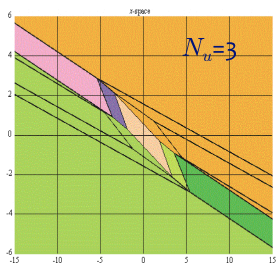
$$u(x) = \begin{cases} \begin{bmatrix} -0.8166 & -1.7499 \end{bmatrix} x & \text{if } \begin{bmatrix} -0.8166 & -1.7499 \\ 0.8166 & 1.7499 \\ 0.6124 & 0.4957 \\ -0.6124 & -0.4957 \end{bmatrix} x \le \begin{bmatrix} 1.0000 \\ 1.0000 \\ 1.0000 \\ 1.0000 \end{bmatrix} & \text{(Region \#1)} \\ 1.0000 & \text{if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

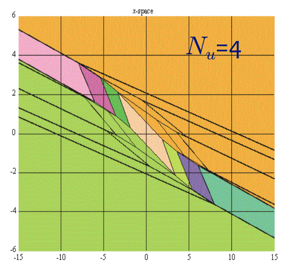
N_u=2

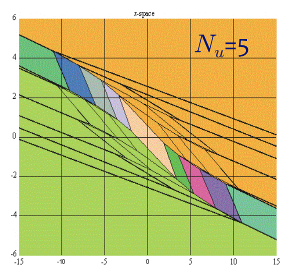


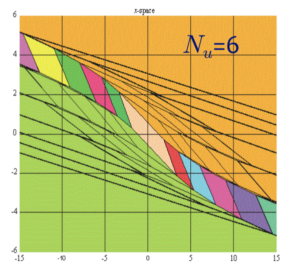
go to demo **/demos/linear/doubleintexp.m** (Hyb-Tbx)

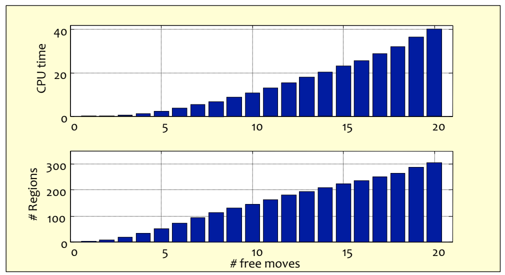
Double integrator example - Complexity of solution











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

Applicability of explicit MPC approach

- Tracking of reference $r(t)$: optimization vector $= \Delta U$

  MPC law $=$ PWA function of $\begin{bmatrix} x(t) \\ u(t-1) \\ r(t) \end{bmatrix}$

- Rejection of measured disturbance $v(t)$: optimization vector $= \Delta U$

  MPC law $=$ PWA function of $\begin{bmatrix} x(t) \\ u(t-1) \\ v(t) \end{bmatrix}$

- Soft constraints: optimization vector $= \left[\begin{smallmatrix} U \\ \epsilon \end{smallmatrix}\right]$

  MPC law $=$ PWA function of $x(t)$

  $y_{\min} - V_{\min}\epsilon \le y_k \le y_{\max} + V_{\max}\epsilon$

- Variable constraints: optimization vector $= U$

  $$\begin{array}{ccccc} u_{\min}(t) & \le & u_k & \le & u_{\max}(t) \\ y_{\min}(t) & \le & y_k & \le & y_{\max}(t) \end{array}$$

  MPC law $=$ PWA function of $\begin{bmatrix} x(t) \\ u_{\min}(t) \\ \vdots \\ y_{\max}(t) \end{bmatrix}$

- Any combination of the above lead to a multiparametric QP solution

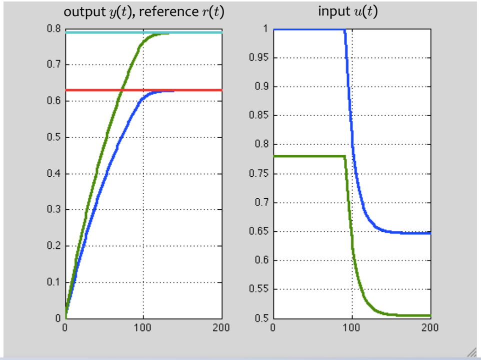
Reference tracking, MIMO system

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

- Constraints: $\left[\begin{smallmatrix} -1 \\ -1 \end{smallmatrix}\right] \le u(t) \le \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$

- Control objective: $\min \displaystyle\sum_{k=0}^{19} \|y_k - r(t)\|^2 + \frac{1}{10}\|\Delta u_k\|^2$ N=20

  $u_k \equiv u_0, \ \forall k \ge 1$ N_u=1



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

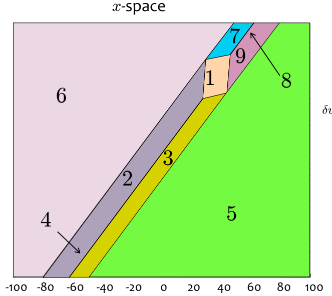
(Hyb-Tbx)

Reference tracking, MIMO system

MPC law

state-space partition

x-space



Polyhedral partition of the state-space for $u(t$-1)=[0 0]' and $r(t)$=[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}$$

Hybrid Toolbox for MATLAB

(Bemporad, 2003-2012)

Supported by

Features:

- **Explicit** MPC control (via multi-parametric programming)
- Simulink library
- C-code generation
- **Hybrid** models: design, simulation, verification
- Control design for linear systems w/ constraints and hybrid systems
- Interfaces to several QP/LP and Mixed-Integer Programming solvers

4000+ download requests since October 2004

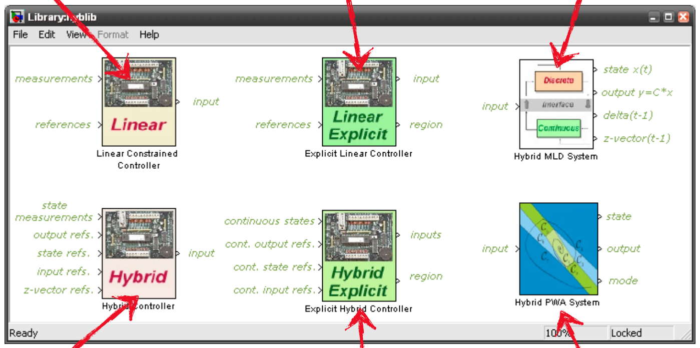
http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox/

Hybrid Toolbox - Simulink library

linear MPC based on on-line QP (M-code)

explicit linear MPC (C-code)

MLD dynamics (M-code)



hybrid MPC based on on-line MIP (M-code)

explicit hybrid MPC (C-code)

PWA dynamics (M-code)

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)

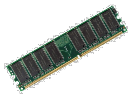
Average time on 100 random 3D parameters (2N constraints)

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

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

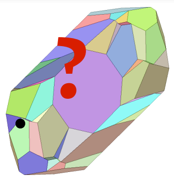
→ Regions can be visited more efficiently than linear search

→ Number of regions can be usually reduced (e.g. suboptimal solutions)



Point location problem

In which region of the partition is $x(t)$?



- Store all regions and search linearly through them
- Exploit properties of mpLP solution to locate $x(t)$ from value function (also extended to mpQP) (Baotic, Borrelli, Bemporad, Morari 2008)
- Organize regions on a tree for logarithmic search (Tøndel, Johansen, Bemporad, 2003)
- For mpLP, recast as weighted nearest neighbour problem (logarithmic search) (Jones, Grieder, Rakovic, 2003)
- Exploit reachability analysis (Spjøtvold, Rakovic, Tøndel, Johansen, 2006) (Wang, Jones, Maciejowski, 2007)
- Use bounding boxes and trees (Christophersen, Kvasnica, Jones, Morari, 2007)

Complexity

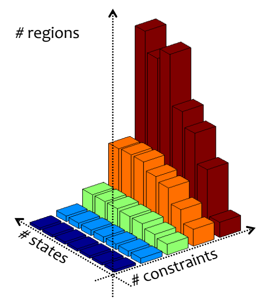
- Number n_r of regions = number of combinations of active constraints at optimality
  - Mainly depends on number q of constraints: $n_r \leq \sum_{h=0}^{q} \binom{q}{h} = 2^q$

    (this is a worst-case estimate, most of the combinations are never optimal !)
  - Also depends on #free variables
  - Weakly depends on #states

- Example

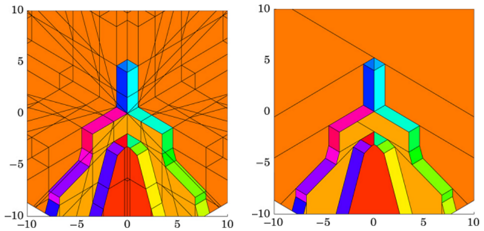
average on 20 random SISO systems (input saturation)

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



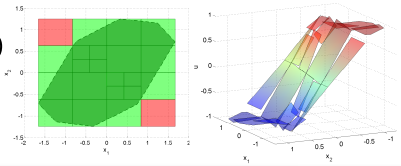
Region reduction and approximations

- Join regions more efficiently
  (Geyer, Torrisi, Morari, 2008)



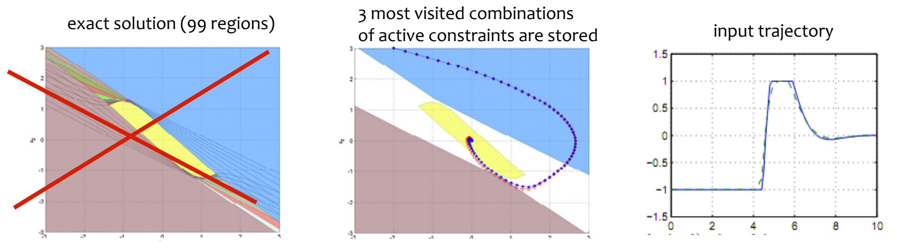
(a) Original set of 252 polyhedra. (c) Set of 39 polyhedra resulting from OCR.

- Change cost function (e.g. minimum time)
  (Grieder, Morari, 2003)
- Relax KKT conditions (suboptimal mpQP)
  (Bemporad, Filippi, 2003)
- Use orthogonal trees (suboptimal solutions)
  (Johansen, Grancharova, 2003)
  (Liang, Heemels, Bemporad, CDC 2011)



Interpolation methods

- Interpolate solution from reduced number of regions
  (Pannocchia, Rawlings, Wright, 2007) (Christophersen, Zeilinger, Jones, Morari, 2007)



(Alessio, Bemporad, NMPC 2008)

$\beta_i(x) = \max_j \{H_i^j x - K_i^j\}$ max violation (how much x is outside region $H_i x \leq K_i$

$$\bar{u}(x) = \left(\sum_{i=1,\ldots,L} \frac{1}{\beta_i(x)}\right)^{-1} \sum_{i=1,\ldots,L} \frac{1}{\beta_i(x)}(F_i x + g_i)$$

or set $\bar{u}(x) = F_h x + g_h$

$$\beta_h(x) = \min_{i=1,\ldots,L} \beta_i(x)$$

- Other approaches: (Jones, Morari, 2010) (Kvasnica, Fikar, 2010)

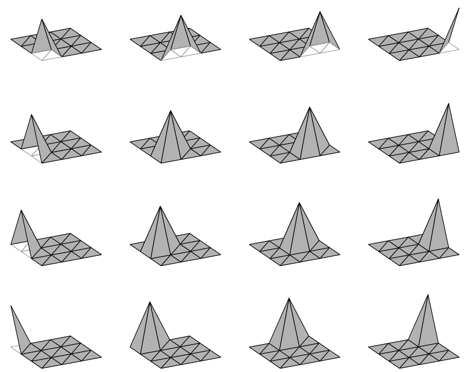
Approximate MPC solutions

- Use gridding methods from *dynamic programming*
- Use any *function approximation* technique to get control law from M samples $u_i = u^*(x_i)$ of the exact MPC law, then check performance and prove stability

  - Lookup tables (linear interpolation, inverse distance weighting, ...)
  - Neural networks (Parisini, Zoppoli 1995)
  - Hybrid (PWA) system identification
  - NL identification (Canale, Fagiano, Milanese NMPC'08)

PWA approximation of MPC over simplices

- ***Approximate*** a given linear MPC controller by using **canonical PWA** functions over **simplicial partitions** (**PWAS**) (Bemporad, Oliveri, Poggi, Storace , IEEE TAC, 2011)



$$\hat{u}(x) = \sum_{k=1}^{N_v} w_k\, \phi_k(x) = w'\phi(x)$$

approximate MPC law

(Julian, Desages, Agamennoni, 1999)

Weights w_k optimized off-line to best approximate a given MPC law

PWA approximation of MPC over simplices

- **Extremely cheap**: **PWAS functions can be directly implemented on FPGA**, or even **ASIC** (Application Specific Integrated Circuits)

- **Extremely fast** computations (**10-100 nanoseconds**)

fit criterion

Control	p	Latency (A - B) [ns]
L^2	7	170 - 31
	31	238 - 45
	63	272 - 46
L^∞	7	170 - 31
	31	238 - 45
	63	272 - 46

Architecture A: mainly serial

Architecture B: fully parallel

online time to compute MPC (Xilinx Spartan 3 FPGA)

partitions per dimension

Exact explicit MPC: 52 regions
383 ns (avg) - 486 ns (max)

MIMO system dynamics

$$x_{k+1} = \begin{bmatrix} 1.2 & 1 \\ 0 & 1.1 \end{bmatrix} x_k + \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} u_k$$

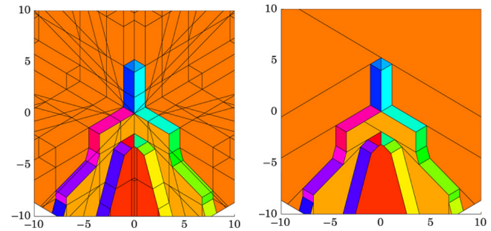
$$y_k = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} x_k$$

subject to saturation on u and y

- Certified closed-loop stability by constructing a **PWA Lyapunov fnc**
- Fulfillment of constraints on inputs (soft constraints on states)

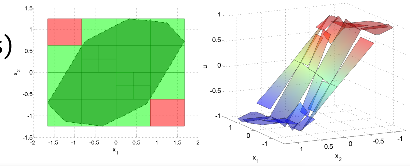
Region reduction and approximations

- Join regions more efficiently
  (Geyer, Torrisi, Morari, 2008)



(a) Original set of 252 polyhedra. (c) Set of 39 polyhedra resulting from OCR.

- Change cost function (e.g. minimum time)
  (Grieder, Morari, 2003)
- Relax KKT conditions (suboptimal mpQP)
  (Bemporad, Filippi, 2003)
- Use orthogonal trees (suboptimal solutions)
  (Johansen, Grancharova, 2003)
  (Liang, Heemels, Bemporad, CDC 2011)



Example: AFTI-16

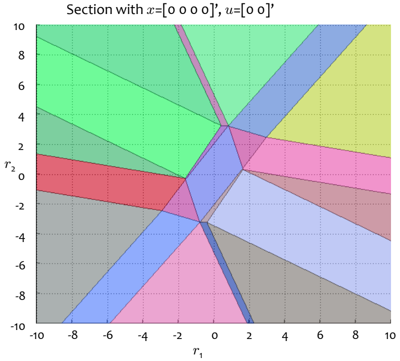
- Linearized model:

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

- 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: $-7.6636, -0.0075 \pm 0.0556j, 5.4530$)

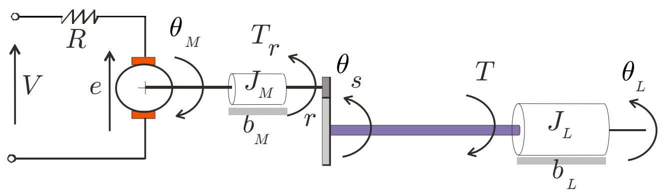
Explicit controller: 8 parameters, 51 regions

Section with x=[0 0 0 0]', u=[0 0]'



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

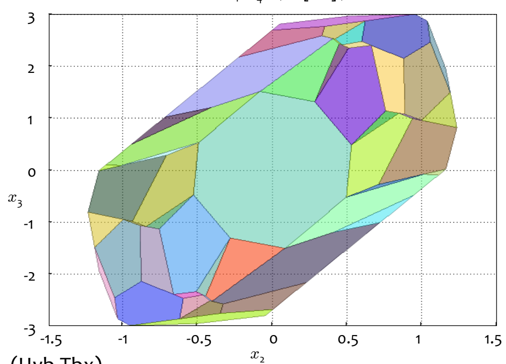
MPC of a DC servomotor



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

Explicit controller:
7 parameters, 101 regions

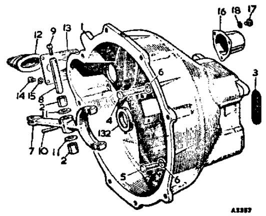
Section with x_1=x_4=0, r=[0 0]', u=0



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

Dry clutch engagement

(Bemporad, Borrelli, Glielmo, Vasca, 2001)



Slip phase: $\omega_e > \omega_v$

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

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

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

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

- **Control objectives:**
  - 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

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

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

- Sample time

  T=10 ms

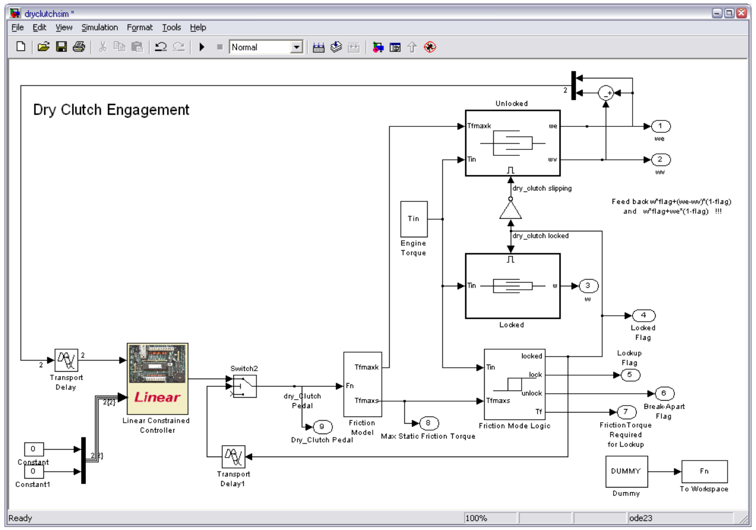
$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ T_{in}(t+1) \\ \dot{T}_{in}(t+1) \\ T_l(t+1) \\ u(t) \end{bmatrix} = \begin{bmatrix} 0.9985 & 0 & 0.0500 & 0.0002 & 0 & -0.0049 \\ -0.0011 & 0.9996 & 0.0500 & 0.0002 & 0.0129 & -0.0062 \\ 0 & 0 & 1.0000 & 0.0100 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0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

- Control objective:

$$\min_{\Delta u_0,\ldots,\Delta u_{N_u-1}} \sum_{k=0}^{N-1} Q x_{2,k}^2 + R\Delta u_k^2 + x_N' P x_N$$

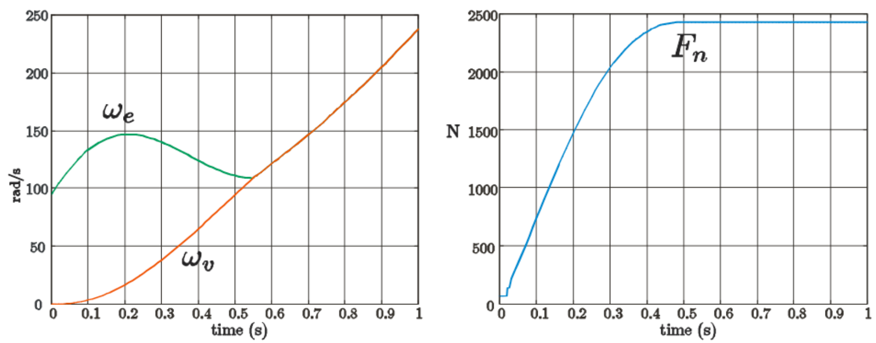
- Constraints: $0 \leq \Delta u \leq 80$ N, $0 \leq u \leq 5000$ N, $x_1 \geq 50$ rad/s, $x_2 \geq 0$

MPC tuning



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

MPC simulation



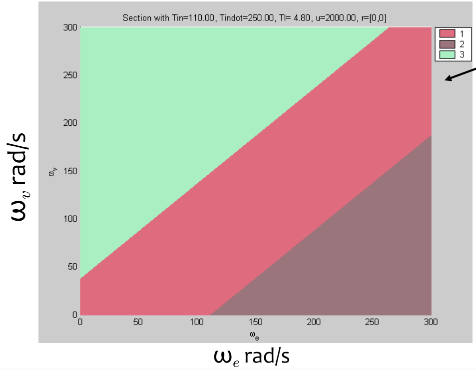
Controller	τ^*	$F_n(\tau^*)$	E_d	$\dot{x}_2(\tau^*)$
1	0.57 s	2333 Nm	6676 J	230 rad/s^2
2	0.76 s	2515 Nm	9777 J	109 rad/s^2
3	0.83 s	2572 Nm	10838 J	62 rad/s^2

Choice:
Q=2, R=1
N=10, N_u=1

Explicit MPC controller

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

+ linear observer



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

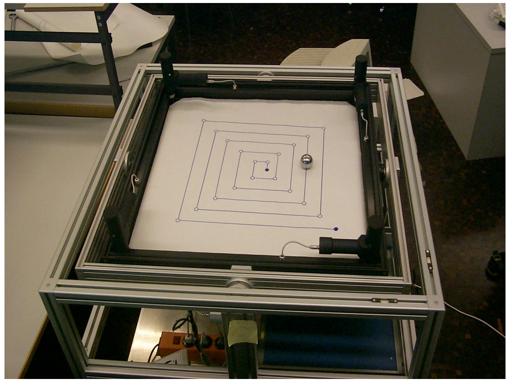
Alternative: explicit **hybrid** MPC

- 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

Ball & Plate System

Host

TCP/IP **Code** **Data**

Target

u_1, u_2

a,b

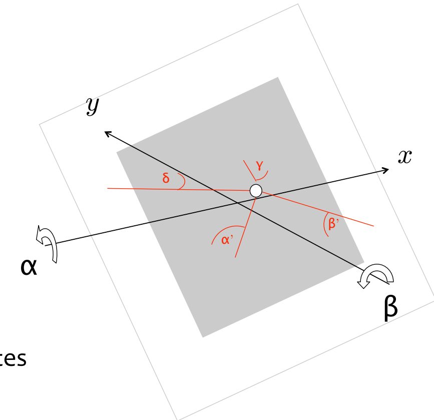
x, y

Ball & Plate

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



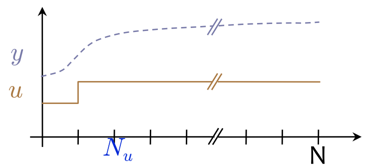
- Model: LTI 14 states
  Constraints on inputs and states

MPC tuning

Sampling time: T_s = 30 ms

Prediction horizon: N = 50

Free control moves: N_u = 2



Output constraint horizon: 1 (*soft constraint*)

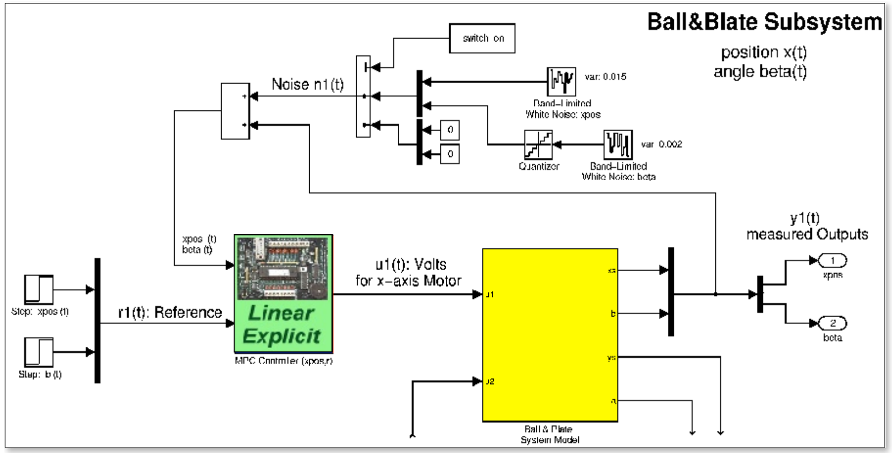
Input constraint horizon: 1 (*hard constraint*)

Weight on position error: 5

Weight on input voltage changes: 1

Implementation

- Solve mp-QP and implement explicit MPC

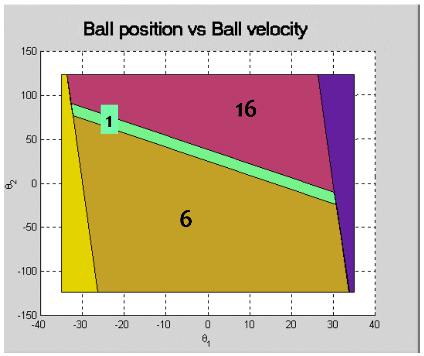


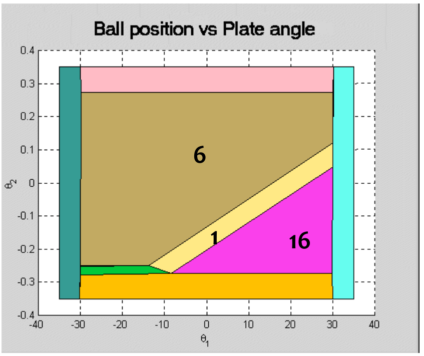
E.g: Real-Time Workshop + xPC Toolbox

Explicit MPC solution

Controller: x: 22 regions, y: 23 regions

x-MPC: sections at $\alpha_x=0,\ \mathring{\alpha}_x=0,\ u_x=0,\ r_x=18,\ r_\alpha=0$



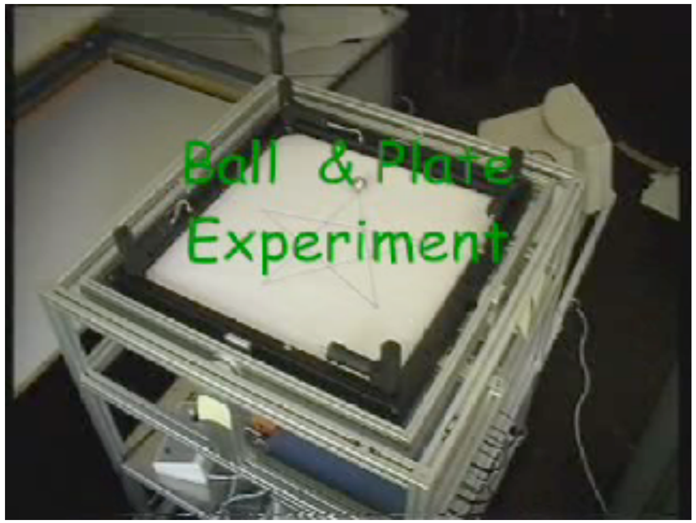


Region 1: LQR Controller (near equilibrium)

Region 6: Saturation at -10

Region 16: Saturation at +10

Ball and plate experiment



Ball and plate experiment

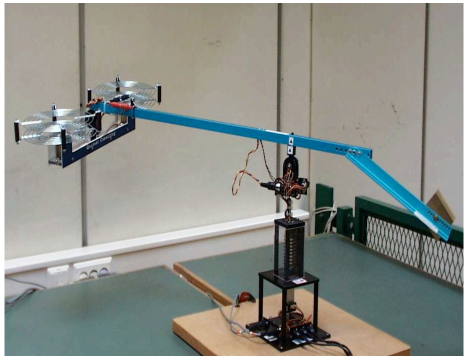
Ball and plate experiment in LEGO, using explicit MPC and Hybrid Toolbox

(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:

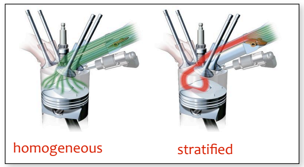
Horizon	Time (s)	# polyhedra
1	1	33
2	19	395
3	163	2211

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

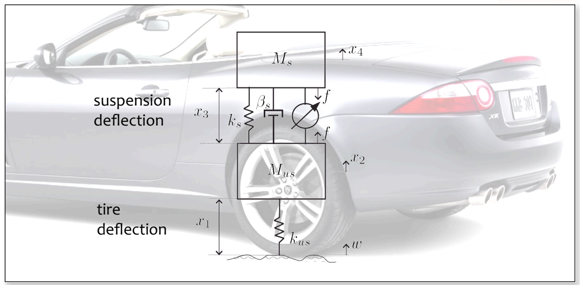
Automotive applications of MPC

traction control
(Borrelli, Bemporad, Fodor, Hrovat, 2001)



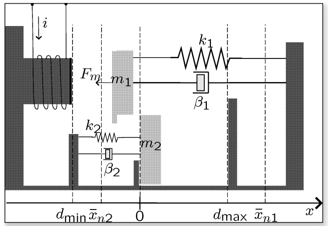
engine control
(N.Giorgetti,G. Ripaccioli, AB, I. Kolmanovsky, D.Hrovat, 2006)

idle speed control
(Di Cairano, Bemporad, Kolmanovsky, Hrovat, 2006)



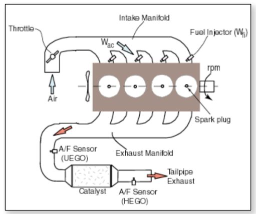
semiactive suspensions (Giorgetti, Bemporad, Tseng, Hrovat, 2005)

Automotive applications of MPC



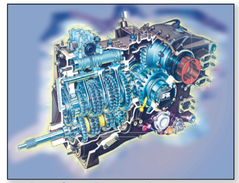
magnetic actuators
(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2008)

hybrid electric vehicles
(Ripaccioli, Bemporad, Assadian, Dextereit, Di Cairano, Kolmanovsky, 2009)



air-to-fuel ratio
(Trimboli, Di Cairano, Bemporad, Kolmanovsky, 2009)

active steering
(Bernardini, Di Cairano, Bemporad, Tseng, 2009)



robotized gearbox
(Bemporad, Borodani, Mannelli, 2003)

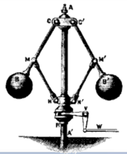
Explicit MPC for idle speed control

(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*

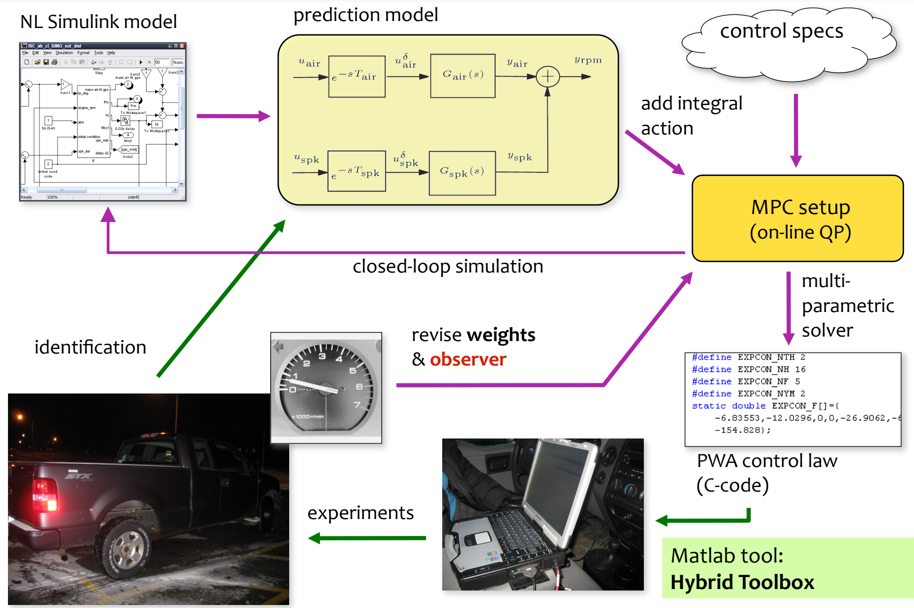
- Objectives and specs:
  - *regulate engine speed* at constant rpm
  - *saturation* limits on airflow and spark
  - *lower bound* on engine speed ($\geq$450 rpm)

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



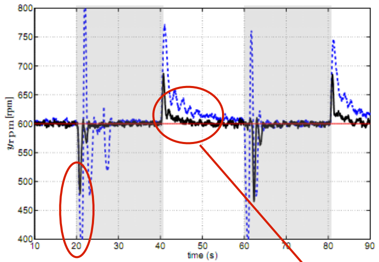
- Problem suitable for MPC design (Hrovat, 1996)

Explicit MPC design flow

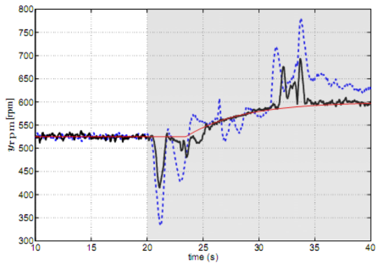


Explicit MPC for idle speed - Experiments

(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

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

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)

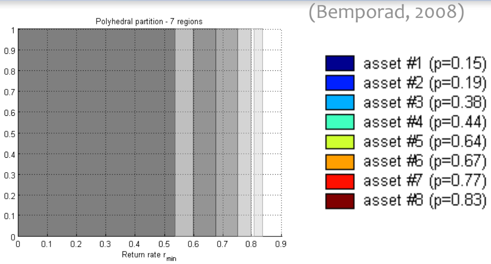
Constraint: guarantee a minimum expected return

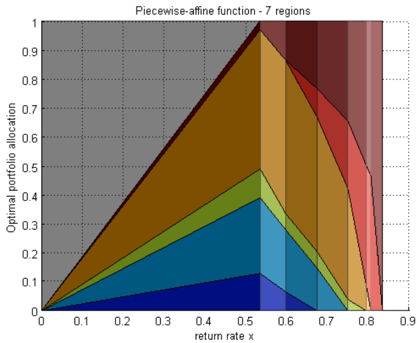
mpQP in portfolio optimization

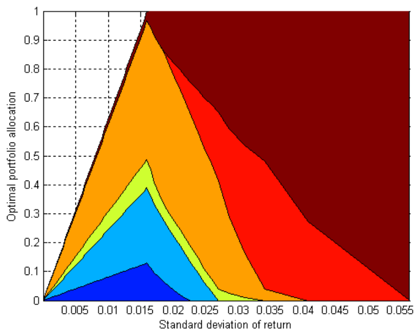
(Bemporad, 2008)

Multiparametric QP solution

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







The origins of (multi)parametric programming

(mono)-parametric LP

THE COMPUTATIONAL ALGORITHM FOR THE PARAMETRIC OBJECTIVE FUNCTION[1]

Saul Gass
U. S. Air Force[2]

and

Thomas Saaty
Melpar, Inc.[3]

(Gass and Saaty, 1955)

Let $\delta \leq \lambda \leq \phi$ be an arbitrary interval on the real line
For each λ in this interval, find a vector $x = (x_1, x_2, \dots x_n)$ which minimizes

$$\sum_{j=1}^{n} (d_j + \lambda d'_j)\, x_j,$$

$$x_j \geq 0 \qquad j = 1, \dots, n,$$

$$\sum_{j=1}^{n} a_{ij} x_j = a_{i0} \qquad i = 1, \dots, m,$$

$$\begin{array}{ll} \min\limits_z & (c_1 + x \cdot c_2)' z \\ \text{s.t.} & Gz = W \\ & z \geq 0 \end{array} \qquad x \in \mathbb{R}$$

Also extended to 2 parameters in 1955 by Gass and Saaty

multi-parametric convex programming

INEQUALITIES FOR STOCHASTIC NONLINEAR PROGRAMMING PROBLEMS

O. L. Mangasarian and J. B. Rosen*

Shell Development Company, Emeryville, California
(Received December, 1962)

(Mangasarian and Rosen, 1964)

$$\begin{array}{rl} V^*(x) = \min\limits_z & h(z, x) \\ \text{s.t.} & g(z, x) \leq 0 \end{array}$$

h, g **convex** in (z, x)

LEMMA 1. *The scalar function $\alpha(a) \equiv \min_z \{\theta(z,a) | f(z,a) \geqq 0\}$ is a convex function of the vector a provided that θ is a convex function of the vector $[z'a']$ and each component of f is a concave function of $[z'a']$.*

h, g convex in (z, x) $\Rightarrow$ $V^*(x)$ convex

LEMMA 2. *The scalar function $\alpha(a) \equiv \min_z \{\theta(z,a)(z,a) | f(z,a) \geqq 0\}$ is a convex and continuous function of the vector a provided that θ is a convex and continuous function of the vector $[z'a']$, and each component of f is a concave and continuous function of $[z'a']$.*

h, g convex and continuous in (z, x) $\Rightarrow$ $V^*(x)$ convex and continuous

multi-parametric LP

MANAGEMENT SCIENCE
Vol. 18, No. 7, March, 1972
Printed in U.S.A.

MULTIPARAMETRIC LINEAR PROGRAMMING*

TOMAS GAL† AND JOSEF NEDOMA‡

The multiparametric linear programming (MLP) problem for the right-hand sides (RHS) is to maximize $z = c^T x$ subject to $Ax = b(\lambda)$, $x \geqq 0$, where $b(\lambda)$ can be expressed in the form

$$b(\lambda) = b^* + F\lambda,$$

where F is a matrix of constant coefficients, and λ is a vector-parameter.
The multiparametric linear programming (MLP) problem for the prices or objective function coefficients (OFC) is to maximize $z = c^T(\nu)x$ subject to $Ax = b$, $x \geqq 0$, where $c(\nu)$ can be expressed in the form $c(\nu) = c^* + H\nu$, and where H is a matrix of constant coefficients, and ν a vector-parameter.

(Gal and Nedoma, 1972)

$$\begin{array}{ll} \min_z & c'z \\ \text{s.t.} & Gz = W + Sx \\ & z \geq 0 \end{array} \qquad \begin{array}{ll} \min_z & (c_1 + x'c_2)'z \\ \text{s.t.} & Gz = W \\ & z \geq 0 \end{array}$$

$$x \in \mathbb{R}^n$$

multi-parametric NLP - 1983

Introduction to Sensitivity and Stability Analysis in Nonlinear Programming

ANTHONY V. FIACCO
Operations Research Department
Institute for Management Science and Engineering
School of Engineering and Applied Science
The George Washington University
Washington, D.C.

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

(Fiacco, 1983)

Very general treatment of multiparametric programming

Multiparametric programming algorithms

Problem	$z^*(x)$	$V^*(x)$	
mp-LP	continuous, PWA	convex (cont.), PWA	(Gal, Nedoma, 1972) (Gal 1995) (Borrelli, Bemporad, Morari, 2003)
mp-QP	continuous, PWA	convex (cont.) PWQ, C¹ (if no degen.)	(Bemporad, Morari, Dua, Pistikopoulos, 2002) (Tøndel, Bemporad, Johansen, 2003a) (Seron, De Doná, Goodwin, 2000) (Baotic, 2002)
mp-MILP	PWA	(nonconvex) PWA	(Acevedo, Pistikopoulos, 1997) (Dua, Pistikopoulos, 2000)
mp-LCP	continuous, PWA	[undefined]	(Jones, Morari, 2006) (Columbano, Fukuda, Jones, 2008)
mp-convex (mp-SDP)	PWA (approx.)	convex (approx.)	(Bemporad, Filippi. 2003)
mp-IP	PW constant	PWA	(Bemporad, 2003) (Crema, 1999)
mp-convex PWQ	PWA	convex PWQ	(Patrinos, Sarimveis, 2011)

Ways to handle *degeneracy* in mpQP/mpLP have been studied

(Tøndel, Bemporad, Johansen, 2003b) (Jones, Kerrigan, Maciejowski. 2007)

Explicit MPC based on linear programming

On-Line vs. Off-Line Optimization

$$V^*(x(t)) = \min_{\xi} \begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 \end{bmatrix} \xi$$
$$\text{s.t.} \quad Gz \leq W + Sx(t)$$

$$\xi \triangleq [\epsilon_0^u \ \dots \ \epsilon_{N-1}^u \ \epsilon_1^x \ \dots \ \epsilon_N^x \ u_0', \dots, u_{N-1}']'$$

- **On-line** optimization: given $x(t)$ solve the problem at each time step t (the control law $u{=}u_0(x)$ is **implicitly** defined by the LP solver)

→ Linear Program (LP)

- **Off-line** optimization: solve the LP **for all** $x(t)$ to find the control law $u{=}u_0(x)$ **explicitly**

→ multi-parametric Linear Program (mp-LP)

Multiparametric LP

(Gal, Nedoma, 1972)
(Borrelli, Bemporad, Morari, 2003)

$$\begin{array}{ll} \min_{\xi} & f'\xi \\ \text{s.t.} & G\xi \leq W + Sx \end{array}$$

For a given parameter x_0:

- solve LP to find ξ_0^* (assume no degeneracy)
- identify active constraints $\tilde{G}\xi_0^* = \tilde{W} + \tilde{S}x,\ \tilde{G} \in \mathbb{R}^{n\times n}$ full rank
- from active constraints $\tilde{G}\xi^*(x) = \tilde{W} + \tilde{S}x$:

→

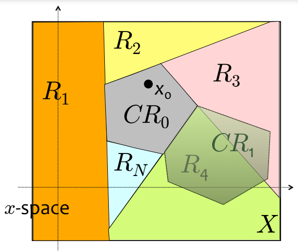
$\xi^*(x) = \tilde{G}^{-1}(\tilde{W} + \tilde{S}x)$ optimizer

$\hat{G}(\tilde{G}^{-1}\tilde{S})x + \hat{G}(\tilde{G}^{-1}W) \leq \hat{W} + \hat{S}x$ critical region CR_0

$V^*(x) = f'\xi^*(x) = f'\tilde{G}^{-1}(\tilde{W} + \tilde{S}x)$ value function

$f + \tilde{G}'\tilde{\lambda} = 0 \ \Rightarrow \ \tilde{\lambda}(x) = -(\tilde{G}^{-1})'f$ dual variables

Multiparametric LP



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

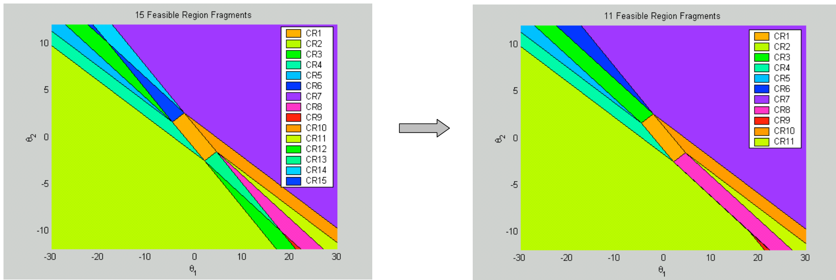
$$R_i = \{x \in X : \ \mathcal{A}^i x > \mathcal{B}^i, \ \mathcal{A}^j x \leq \mathcal{B}^j, \ \forall j < i\}$$

Note: while CR_i is characterizing a set of active constraints, R_i is not

→ 1) Use the above splitting only as a search procedure, don't split the CR

2) Remove duplicates of CR already found

Union of regions



$$z(x) \triangleq [\epsilon_0^u(x) \ \dots \ \epsilon_{N-1}^u(x) \ \epsilon_1^x(x) \ \dots \ \epsilon_N^x(x) \ u_0'(x) \ \dots \ u_{N-1}'(x)]'$$

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)

Properties of multiparametric LP

Theorem The set X^* of parameters for which the mpLP problem is feasible is a convex polyhedron. The functions $V^*(x)$ and $\xi^*(x)$ are piecewise affine and continuous over X^*, and $V^*(x)$ is also convex on X^*.

(Gal, Nedoma, 1972)
(Borrelli, Bemporad, Morari, 2003)

$$\begin{aligned}\xi^*(x) &= \arg\min f'\xi \\ &\quad \text{s.t.}\, G\xi \le W + Sx\end{aligned}$$

piecewise affine, continuous
(if optimizer is always unique)

$$\begin{aligned}V^*(x) &= \min f'\xi \\ &\quad \text{s.t.}\, G\xi \le W + Sx\end{aligned}$$

continuous, piecewise affine, convex

Corollary 1: The value function $V^*(x)$ is continuous piecewise affine

Corollary 2: The MPC controller is continuous piecewise affine !

Triple integrator example

- System:

$$y(t) = \frac{1}{s^3} u(t)$$

sampling + ZOH
T=0.1 s

$$\begin{aligned}x(t+1) &= \begin{bmatrix} 1.05 & 0.46 & 1.02 \\ 0 & 0.73 & -0.01 \\ 0 & 0.86 & 0.99 \end{bmatrix} x(t) + \begin{bmatrix} 0.15 \\ 0.86 \\ 0.45 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t)\end{aligned}$$

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

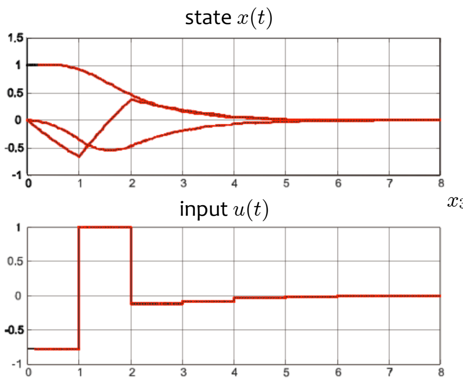
- Control objective: minimize: $\displaystyle\sum_{k=0}^{2} |y(k)| + \left|\frac{1}{8000} u(k)\right|$

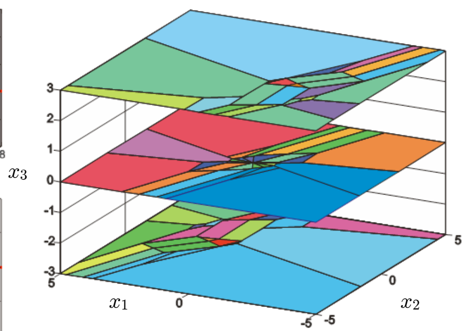
- LP problem:

$$f = [1\ 1\ 1\ 1\ 0\ 0],$$

$$G = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1332.1 & 0 \\ 0 & 0 & -1 & 0 & -7319.1 & 0 \\ 0 & 0 & -1 & 0 & -3848.8 & 0 \\ 0 & 0 & -1 & 0 & 1332.1 & 0 \\ 0 & 0 & -1 & 0 & 7319.1 & 0 \\ 0 & 0 & -1 & 0 & 3848.8 & 0 \\ 0 & 0 & 0 & -1 & -8706.3 & -1332.1 \\ 0 & 0 & 0 & -1 & -5295.8 & -7319.1 \\ 0 & 0 & 0 & -1 & -10116 & -3848.8 \\ 0 & 0 & 0 & -1 & 8706.3 & 1332.1 \\ 0 & 0 & 0 & -1 & 5295.8 & 7319.1 \\ 0 & 0 & 0 & -1 & 10116 & 3848.8 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}, \quad W = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 8935.8 & 3915.4 & 8689.4 \\ 0 & 6227.3 & -146.38 \\ 0 & 7319.1 & 8423 \\ -8935.8 & -3915.4 & -8689.4 \\ 0 & -6227.3 & 146.38 \\ 0 & -7319.1 & -8423 \\ 9394 & 14467 & 17678 \\ 0 & 4436.2 & -252.3 \\ 0 & 12615 & 8220.7 \\ -9394 & -14467 & -17678 \\ 0 & -4436.2 & 252.3 \\ 0 & -12615 & -8220.7 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Explicit MPC Controller





# constraints	# variables	# critical regions
20	6	68

Main Issues in LP-Based MPC

- Idle control performance: $u^*(t) = 0 \quad x(t) \neq 0$

– On-line optimization: Not detected
– Off-line optimization: Detected

- Multiple optima: $u^*(t) = \{u_1^*(t), \ldots, u_K^*(t)\}$

– On-line optimization: Not detected
Arbitrary choice
– Off-line optimization Detected
Multiple explicit solution exploited

Robust (explicit) MPC

Multiparametric solutions: Min-max MPC

$$\boxed{x_{k+1} = A(w_k)x_k + B(w_k)u_k + Ev_k}$$ uncertain linear model

$$A(w) = A_0 + \sum_{i=1}^{q} A_i w_i, \quad B(w) = B_0 + \sum_{i=1}^{q} B_i w_i$$ w, v belong to polytopes

- open-loop prediction, ∞-norms: solved via mpLP $(A(w) \equiv A_0)$ (Bemporad, Borrelli, Morari, 2003)
- closed-loop prediction, ∞-norms:
  - mpLP iterations (dynamic programming solution) (Bemporad, Borrelli, Morari, 2003)
  - mpLP solving single LP problem of Scokaert-Mayne (Kerrigan, Maciejowski, 2004)
- min-max MPC with quadratic costs (Ramirez, Camacho, 2006) (Munoz, Alamo, Ramirez, Camacho, 2007)

Explicit min-max MPC control law is piecewise affine