RECENT ADVANCES IN EMBEDDED MODEL PREDICTIVE CONTROL

Alberto Bemporad

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

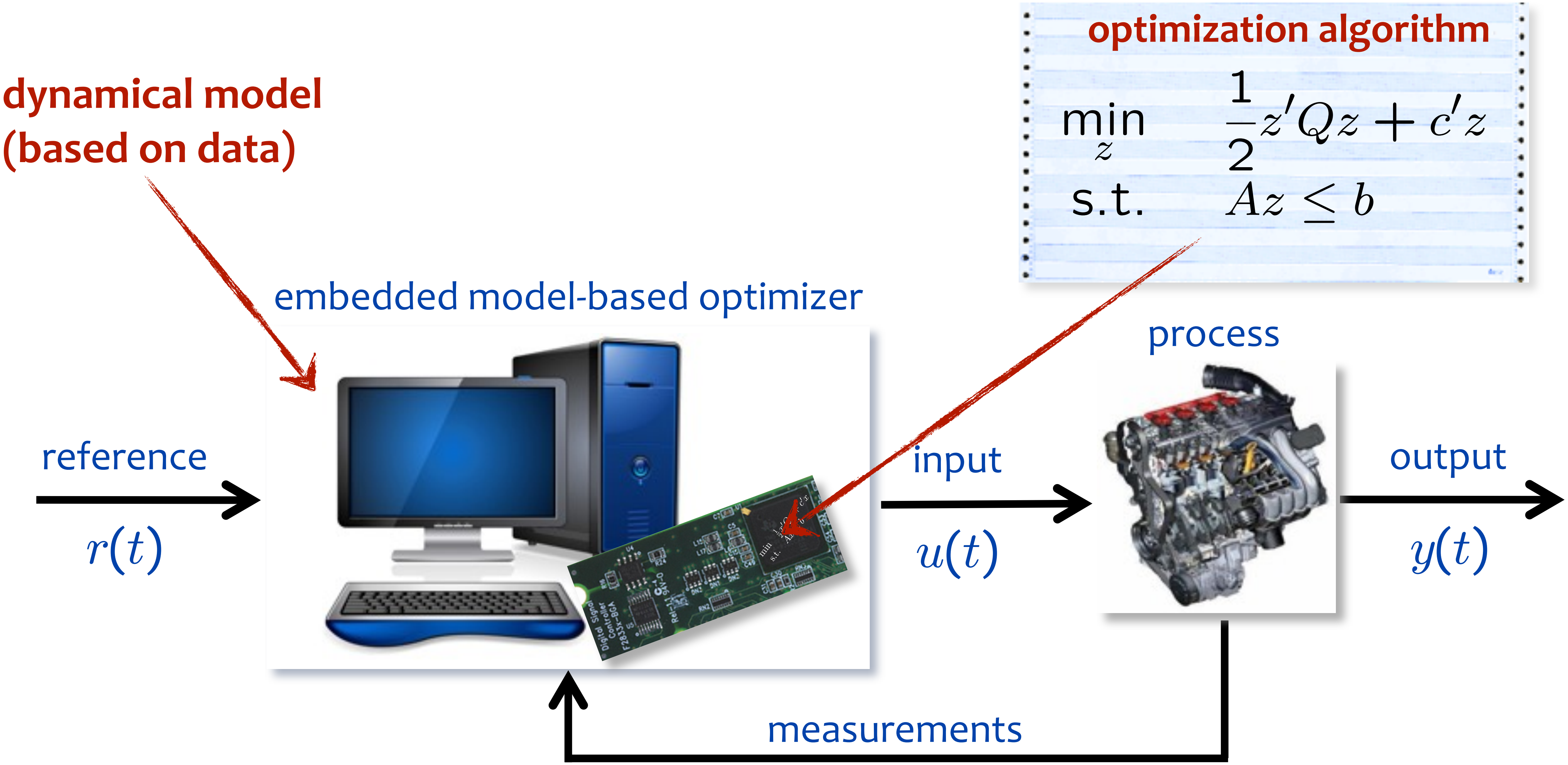
www.odys.it

IMT INSTITUTE FOR ADVANCED STUDIES LUCCA

ODYS Advanced Controls & Optimization

KTH, May 24, 2016

MODEL PREDICTIVE CONTROL (MPC)



Use a dynamical ***model*** *of the process to* ***predict*** *its future evolution and choose the "best"* ***control*** *action*

MODEL PREDICTIVE CONTROL (MPC)

- At time t: consider optimal control problem over a future horizon of N steps

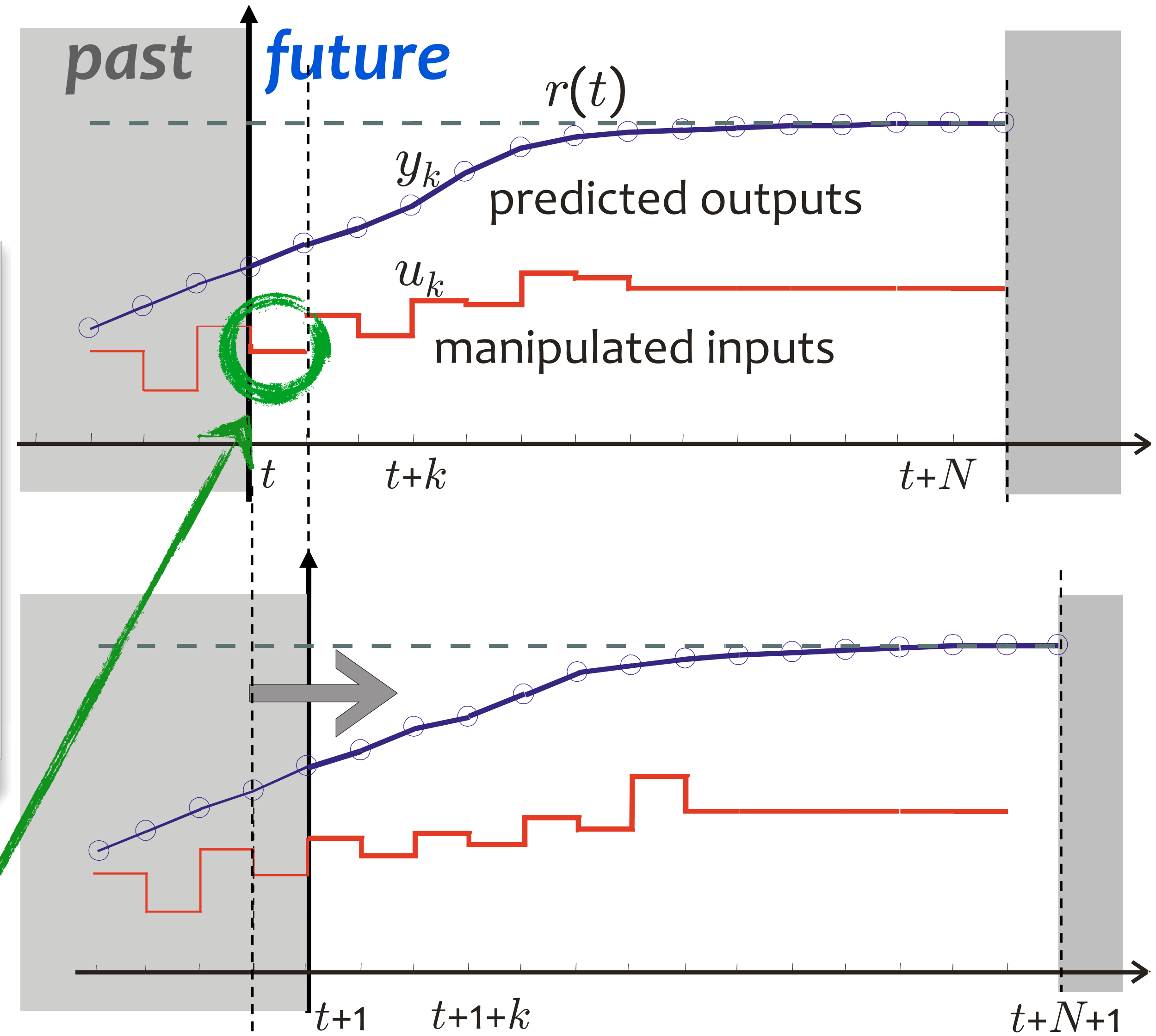
penalty on tracking error

penalty on actuation

$$
\begin{aligned}
\min \quad & \sum_{k=0}^{N-1} \|W^y(y_k - r(t))\|^2 + \|W^u(u_k - u^{\mathrm{ref}}(t))\|^2 \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k, t) \\
& y_k = g(x_k, u_k, t) \\
& \text{constraints on } u_k, y_k \\
& x_0 = x(t)
\end{aligned}
$$

← feedback !

optimization problem



- Solve problem w.r.t. $\{u_0,\dots,u_{N-1}\}$
- Apply the first optimal move $u(t)=u_0^*$, throw the rest of the sequence away
- At time $t+1$: Get new measurements, repeat the optimization. And so on ...

Used in process industries since the 80's

ONE OF THE FIRST PAPERS ON MPC

(Rafal, Stevens, *AiChE Journal*, 1968)

Discrete Dynamic Optimization Applied to On-Line Optimal Control

MARSHALL D. RAFAL and WILLIAM F. STEVENS
Northwestern University, Evanston, Illinois

A general method has been developed for controlling deterministic systems described by linear or linearized dynamics. The discrete problem has been treated in detail. Step-by-step optimal controls for a quadratic performance index have been derived. The method accommodates upper and lower limits on the components of the control vector.

A small binary distillation unit was considered as a typical application of the method. The control vector was made up of feed rate, reflux ratio, and reboiler heat load. Control to a desired state and about a load upset was effected.

Calculations are performed quite rapidly and only grow significantly with an increase in the dimension of the control vector. Extension to much larger distillation units with the same controls thus seems practical.

The advent of high-speed computers has made possible the on-line digital control of many chemical engineering processes. In on-line control a three-step procedure is adhered to:

1. Sense the current state.
2. Calculate a suitable control action.
3. Apply this control for a period of time known as the sampling period.

The present study proposes a method for performing step 2. The technique developed is based on linearized dynamics. The strongly nonlinear binary distillation unit provides a suitable system for this study. While much has been published recently (*2, 3, 8*) on modeling distillation, little if anything has appeared on the optimal control of such units.

In recent years, a good deal has been published by Kalman, Lapidus, and others (*4* to *7*) on the control of linear or linearized nonlinear systems by minimizing a quadratic function of the states resulting from a sequence of control actions. Their controls are always unconstrained, although the introduction of a quadratic penalty function limits this effect somewhat. The general constrained problem has been treated numerically (*1*) for a single control variable. It was Wanninger (*10, 11*) who first chose to look at the problem on a one-step-at-a-time basis rather than considering a sequence of controls. However, he made no attempt to solve completely the resulting quadratic programming problem.

The approach taken in the present work is to set up the problem on a one-step basis. This is quite compatible with the on-line digital control scheme. The problem is then shown to be a special case of the quadratic programming problem and as such has a special solution. The particulars concerning the theory underlying the solution scheme and its implementation on a digital computer have been presented (*9*). In addition, a derivation of the theorems upon which the computational algorithm is based is presented in the Appendix.

The authors wish to be very careful to point out that *optimal*, as used herein, refers only to a single step of control. Even for truly linear systems, the step-by-step optimal control need not be overall optimal. A recent text by Athans and Falb (*1a*) presents both the virtues and defects of such a one-step method. In the present work, the one-step approach is taken because it is amenable to practical solution of the problem and is well suited to nonlinear situations where updating linearization is useful.

THE PROBLEM

The system under consideration is described by a set of matrix differential equations:

$$\dot{X}(t) = AX(t) + BM(t) + \delta(t) \qquad (1)$$

Marshall D. Rafal is with Esso Research and Engineering Company., Florham Park, New Jersey.

AUTOMOTIVE APPLICATIONS OF MPC

Bemporad, Bernardini, Borrelli, Cimini, Di Cairano, Esen, Giorgetti, Graf-Plessen, Hrovat, Kolmanovsky, Levijoki, Ripaccioli, Trimboli, Tseng, Yanakiev, ... (2001-2016)

Powertrain

- direct-inj. engine control
- A/F ratio control
- magnetic actuators
- robotized gearbox
- power MGT in HEVs
- cabin heat control in HEVs
- electrical motors

Vehicle dynamics

- traction control
- active steering
- semiactive suspensions
- autonomous driving

Ford Motor Company

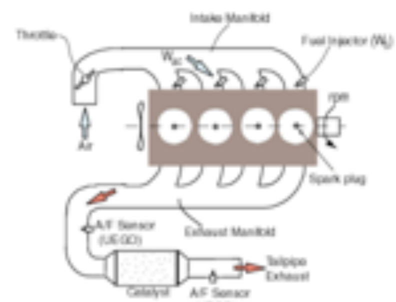
Jaguar

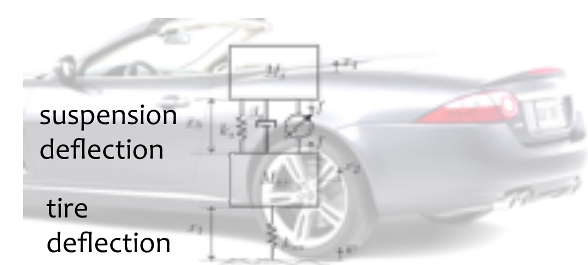
DENSO Automotive

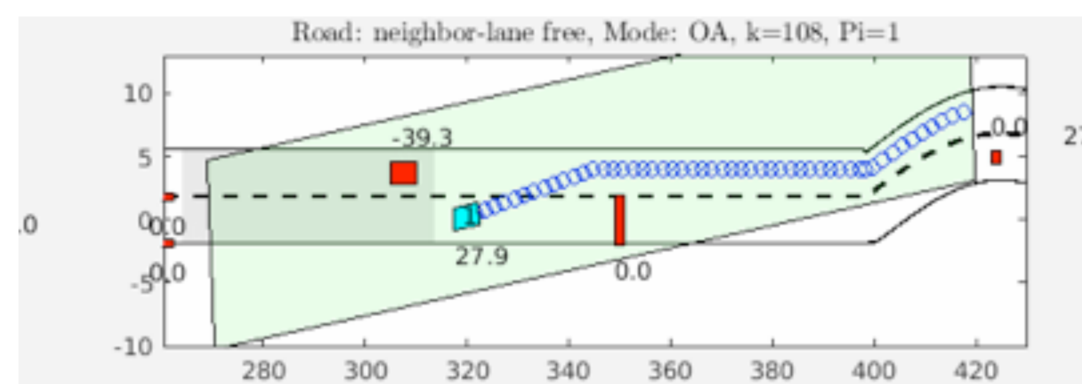
FIAT

General Motors

ODYS Advanced Controls & Optimization







MPC TOOLBOXES

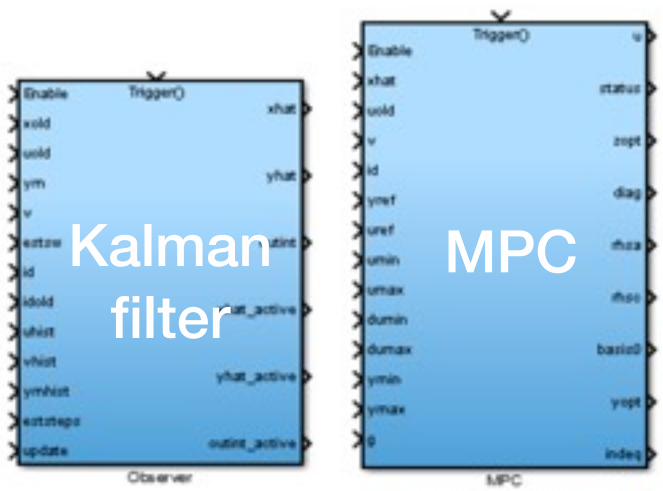
- **MPC Toolbox (The Mathworks, Inc.)**
  (Bemporad, Ricker, Morari, 1998-present)
  - Part of Mathworks' official toolbox distribution
  - Great for **education and research**

- **Hybrid Toolbox**
  (Bemporad, 2003-present)

> 6k downloads

  - Free download: `http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox/`
  - Great for **research and education**

- **ODYS Toolbox**
  (Bemporad, Bernardini, 2013-present)
  - Provides flexible and customized MPC control **design** and **seamless integration** in production systems
  - Real-time code written in plain C
  - **Designed for production**



ODYS
Advanced Controls & Optimization

AEROSPACE APPLICATIONS OF MPC

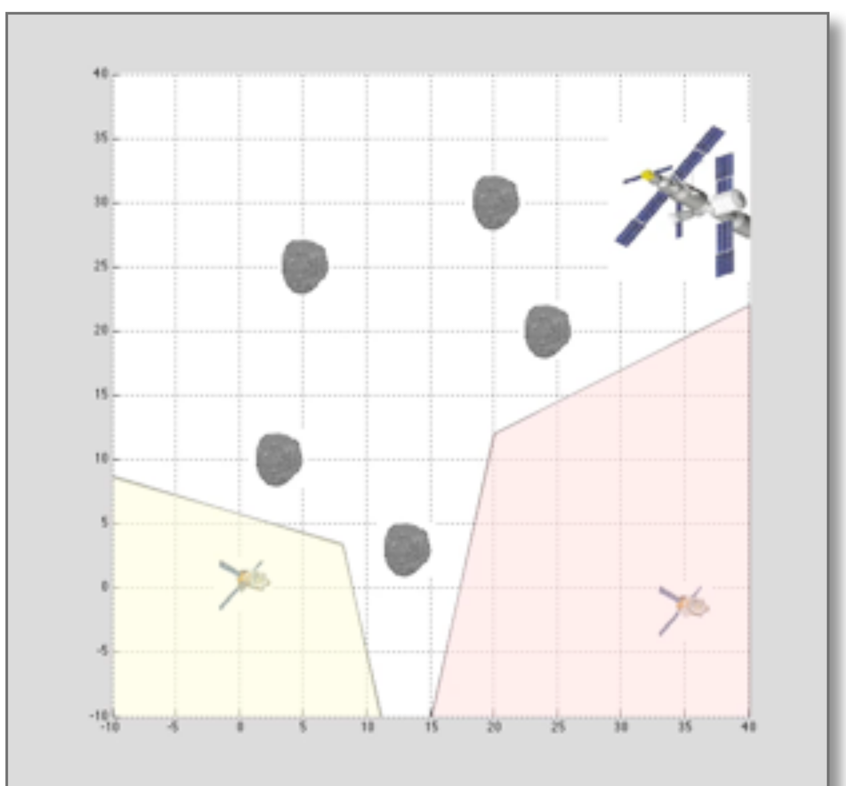
- MPC capabilities explored in new space applications
- New MATLAB MPC Toolboxes developed (**MPCTOOL** and **MPCSofT**)

(Bemporad, 2010) (Bemporad, 2012)

powered descent

(Pascucci, Bennani, Bemporad, 2016)

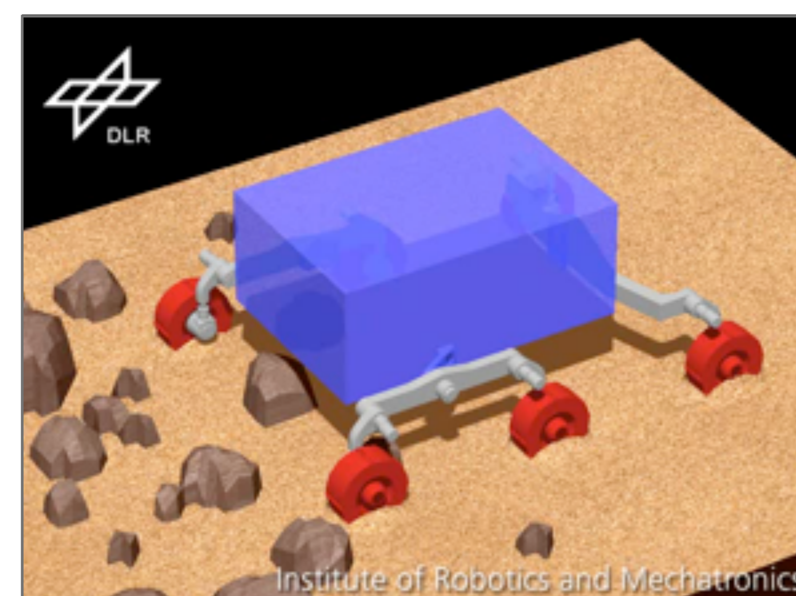
cooperating UAVs



(Bemporad, Rocchi, 2011)

planetary rover

(Krenn et. al., 2012)

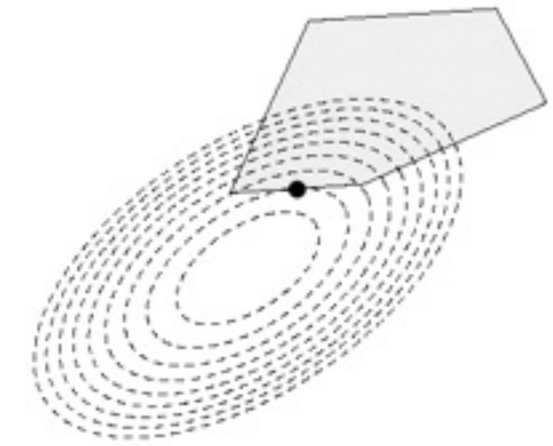


EMBEDDED LINEAR MPC AND QUADRATIC PROGRAMMING

- Linear MPC requires solving a **Quadratic Program (QP)**

$$
\begin{array}{ll}
\min\limits_{z} & \frac{1}{2}z'Hz + x'(t)F'z + \cancel{\frac{1}{2}x'(t)Yx(t)} \\
\text{s.t.} & Gz \leq W + Sx(t)
\end{array}
$$

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



On Minimizing A Convex Function Subject to Linear Inequalities

By E. M. L. Beale

Admiralty Research Laboratory, Teddington, Middlesex

SUMMARY

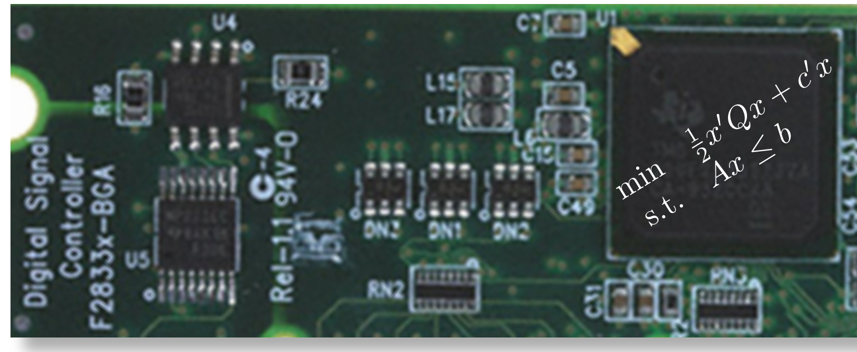
The minimization of a convex function of variables subject to linear inequalities is discussed briefly in general terms. Dantzig's Simplex Method is extended to yield finite algorithms for minimizing either a convex quadratic function or the sum of the *t* largest of a set of linear functions, and the solution of a generalization of the latter problem is indicated. In the last two sections a form of linear programming with random variables as coefficients is described, and shown to involve the minimization of a convex function.

(Beale, 1955)

A rich set of good QP algorithms is available today, still more research is required to have an impact in real applications !

MPC IN A PRODUCTION ENVIRONMENT

embedded model-based optimizer



Requirements for production:

1. **Speed (throughput)**: solve optimization problem within sampling interval

2. **Robustness** (e.g., with respect to numerical errors)

3. Be able to run on **limited hardware** (e.g., 150 MHz) with **little memory**

4. **Worst-case execution time** must be (tightly) estimated

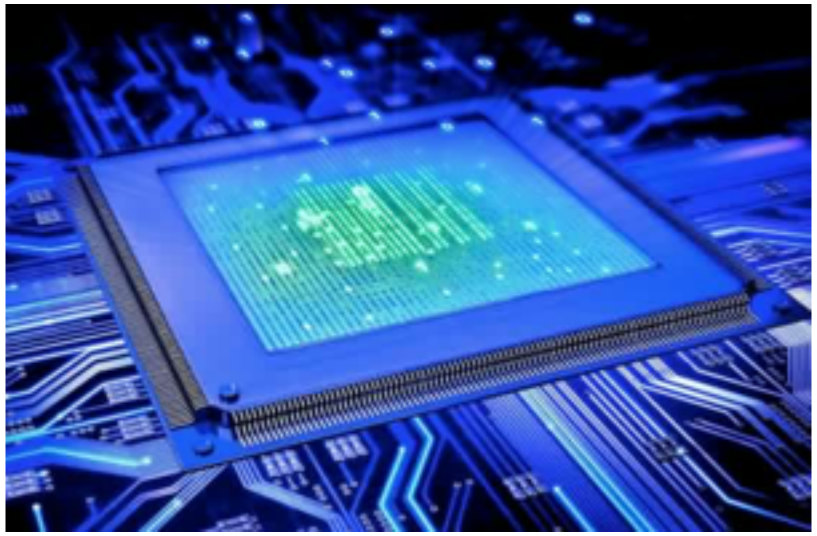
5. **Code simple** enough to be validated/verified/certified
(in general, it must be understandable by production engineers)

EMBEDDED SOLVERS IN INDUSTRIAL PRODUCTION

- Multivariable MPC controller
- Sampling frequency = 40 Hz (= 1 QP solved every 25 ms)
- Vehicle operating ~1 hr/day for ~360 days/year on average
- Controller may be run on 10 million vehicles

≈ 520,000,000,000,000 QP/yr

and none of them can fail !



ODYS Advanced Controls & Optimization

FAST GRADIENT PROJECTION FOR (DUAL) QP

(Nesterov, 1983)
(Patrinos, Bemporad, IEEE TAC, 2014)

- Apply **fast gradient method** to dual QP:

$$\begin{array}{ll}\min_z & \frac{1}{2}z'Hz + x'F'z \\ \text{s.t.} & Gz \leq W + Sx\end{array}$$

$$\beta_k = \begin{cases} 0 & k = 0 \\ \frac{k-1}{k+2} & k > 0\end{cases}$$

$$\boxed{\begin{array}{rcl} w_k &=& y_k + \beta_k(y_k - y_{k-1}) \\ z_k &=& -Kw_k - Jx \\ s_k &=& \frac{1}{L}Gz_k - \frac{1}{L}(Sx + W) \\ y_{k+1} &=& \max\{y_k + s_k, 0\}\end{array}}$$

$$y_{-1} = y_0 = 0$$

$$\begin{array}{rcl} K &=& H^{-1}G' \\ J &=& H^{-1}F'\end{array}$$

- Termination criterion #1: **primal feasibility**

feasibility tol

$$s_k^i \leq \frac{1}{L}\epsilon_G, \quad \forall i = 1, \ldots, m$$

- Termination criterion #2: **primal optimality**

$$f(z_k) - f^* \leq f(z_k) - \phi(w_k) = -w_k' s_k L \leq \epsilon_V$$

dual function

optimality tol

$$-w_k' s_k \leq \frac{1}{L}\epsilon_V$$

FAST GRADIENT PROJECTION FOR (DUAL) QP

(Patrinos, Bemporad, IEEE TAC, 2014)

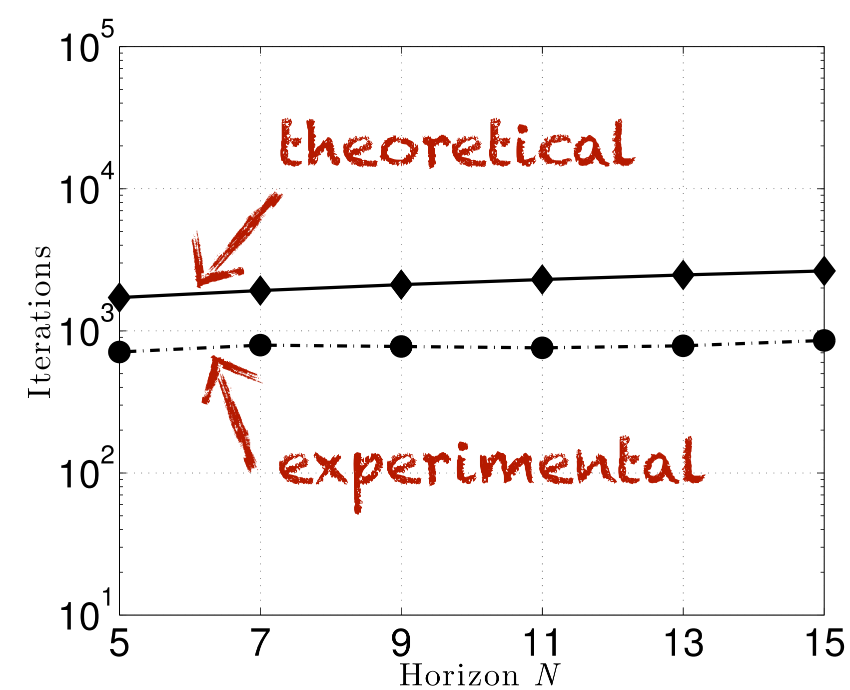
- Main on-line operations involve only **simple linear algebra**

```
while keepgoing && (i<maxiter),

    beta=(i-1)/(i-2).*(i>0);

    w=y+beta*(y-y0);
    z=-(iMG*w+iMc);
    s=GLz-bL;

    y0=y;

    % Check termination conditions
    if all(s<=epsGL),
        gapL=-w'*s;
        if gapL<=epsVL,
            return
        end
    end

    y=max(w+s,0);

    i=i+1;
end
```

- Convergence rate:

$$f(z_{k+1}) - f^* \leq \frac{2L}{(k+2)^2}\|z_0 - z^*\|^2$$

- Tight bounds on maximum number of iterations



- Can be used to warm-start other methods

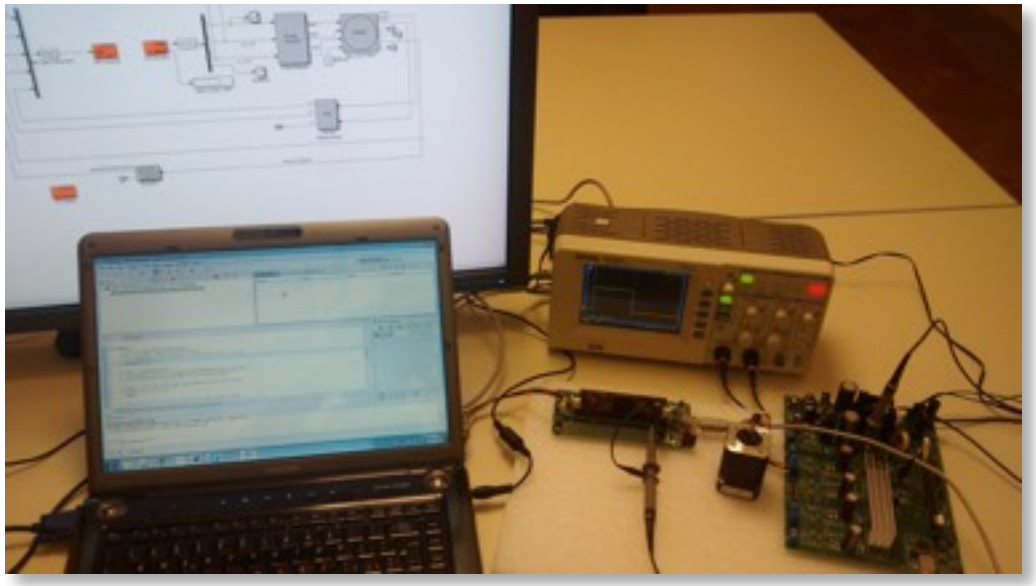
- Currently extended to mixed-integer problems

(Naik, Bemporad, work in progress)

EXPERIMENTS WITH EMBEDDED QP

TMS320F28335 controlCARD
(Real-time Control Applications)

- 32-bit Floating Point (IEEE-754);
- 150MHz clock;
- 68KB Ram / 512KB Flash.



var × constr.	GPAD	AS	ADMM	FBN
4×16	332 μs (18)	120 μs (3)	1.42 ms (62)	208 μs (2)
8×24	1.1 ms (22)	446 μs (5)	4 ms (77)	396 μs (2)
12×32	2.59 ms (27)	1.19 ms (7)	8.25 ms (82)	652 μs (2)

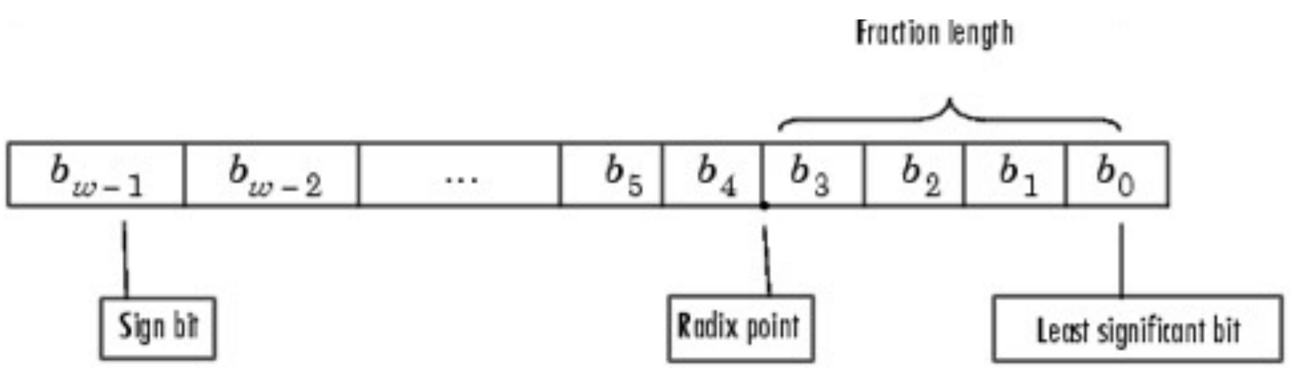
- **Active set** (AS) methods are usually the best on small problems:
  - excellent quality solutions within few iterations
  - less sensitive to preconditioning (= behavior is more predictable)
  - no need for advanced linear algebra packages

* GPAD = Dual Accelerated Gradient Projection (Patrinos, Bemporad, 2014)
* FBN = Forward-Backwards Netwon (proximal method) (Patrinos, Guiggiani, Bemporad, 2014)
* ADMM = Alternating Directions Method of Multipliers (Boyd et al., 2010)

FINITE-PRECISION ARITHMETICS

How about numerical robustness ?

- **Fixed-point arithmetics** is very attractive for embedded control:



- Pros:
  - Computations are fast and cheap
  - Hardware support in all platforms

- Cons:
  - Accumulation of quantization errors
  - Limited range (numerical overflow)

HARDWARE TESTS (FLOATING VS FIXED POINT)

(Patrinos, Guiggiani, Bemporad, 2013)

- Gradient projection works in fixed-point arithmetics

$$\max_i g_i(z_k) \leq \frac{2LD^2}{k+1} + L_v\epsilon_z^2 + 4D\epsilon_\xi$$

exponentially decreasing with number p of fractional bits

max constraint violation

Table 1
Fixed-point hardware implementation

Size [variables/constraints]	Time [ms]	Time per iteration [μs]	Code Size [KB]
10/20	22.9	226	15
20/40	52.9	867	17
40/80	544.9	3382	27
60/120	1519.8	7561	43

fixed point

Table 2
Floating-point hardware implementation

Size [variables/constraints]	Time [ms]	Time per iteration [μs]	Code Size [KB]
10/20	88.6	974	16
20/40	220.1	3608	21
40/80	2240	13099	40
60/120	5816	30450	73

floating point

32-bit Atmel SAM3X8E

ARM Cortex-M3 processing unit

84 MHz, 512 KB of flash memory

and 100 KB of RAM

fixed-point about 4x faster than floating-point

CAN WE SOLVE QP'S USING LEAST SQUARES ?

The **Least Squares (LS)** problem is probably the most studied problem in numerical linear algebra

(Legendre, 1805)

(Gauss, <= 1809)

$$v = \arg\min \|Av - b\|_2^2$$

In MATLAB: >> `v=A\b % (1 character ! )`

- **Nonnegative Least Squares (NNLS):**

$$\begin{array}{ll} \min_v & \|Av - b\|_2^2 \\ \text{s.t.} & v \geq 0 \end{array}$$

ACTIVE-SET METHOD FOR NONNEGATIVE LEAST SQUARES

(Lawson, Hanson, 1974)

$$\begin{array}{ll} \min_v & \|Av - b\|_2^2 \\ \text{s.t.} & v \geq 0 \end{array}$$

1) $\mathcal{P} \leftarrow \emptyset$, $v \leftarrow 0$;
2) $w \leftarrow A'(Av - b)$;
3) **if** $w \geq 0$ **or** $\mathcal{P} = \{1, \dots, m\}$ **then go to** Step 11;
4) $i \leftarrow \arg\min_{i \in \{1,\dots,m\} \setminus \mathcal{P}} w_i$, $\mathcal{P} \leftarrow \mathcal{P} \cup \{i\}$;
5) $y_{\mathcal{P}} \leftarrow \arg\min_{z_{\mathcal{P}}} \|((A')_{\mathcal{P}})' z_{\mathcal{P}} - b\|_2^2$, $y_{\{1,\dots,m\} \setminus \mathcal{P}} \leftarrow 0$;
6) **if** $y_{\mathcal{P}} \geq 0$ **then** $v \leftarrow y$ **and go to** Step 2;
7) $j \leftarrow \arg\min_{h \in \mathcal{P}:\ y_h \leq 0} \left\{ \frac{v_h}{v_h - y_h} \right\}$;
8) $v \leftarrow v + \frac{v_j}{v_j - y_j}(y - v)$;
9) $\mathcal{I} \leftarrow \{h \in \mathcal{P} : v_h = 0\}$, $\mathcal{P} \leftarrow \mathcal{P} \setminus \mathcal{I}$;
10) **go to** Step 5;
11) $v^* \leftarrow v$; **end**.

NNLS algorithm: *While maintaining primal var v feasible, keep switching active set until dual var w is also feasible*

- NNLS algorithm is very simple (**`750 chars in Embedded MATLAB`**), the key operation is to solve a **standard LS problem** at each iteration (via QR, LDL, or Cholesky factorization)

SOLVING QP'S VIA NONNEGATIVE LEAST SQUARES

(Bemporad, IEEE TAC, 2016)

- Use NNLS to solve strictly convex QP

$$\begin{array}{ll} \min\limits_{z} & \frac{1}{2}z'Qz + c'z \\ \text{s.t.} & Gz \leq g \end{array}$$

QP

$$u \triangleq Lz + L^{-T}c$$

complete the squares

$$Q = L'L$$
$$M = GL^{-1}$$
$$d = b + GQ^{-1}c$$

$$\begin{array}{ll} \min\limits_{u} & \frac{1}{2}\|u\|^2 \\ \text{s.t.} & Mu \leq d \end{array}$$

Least Distance Problem

$$\begin{array}{ll} \min\limits_{y} & \frac{1}{2}\left\|\begin{bmatrix} M' \\ d' \end{bmatrix} y + \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\|_2^2 \\ \text{s.t.} & y \geq 0 \end{array}$$

Nonnegative Least Squares

residual $= 0$?

yes

QP problem infeasible

no

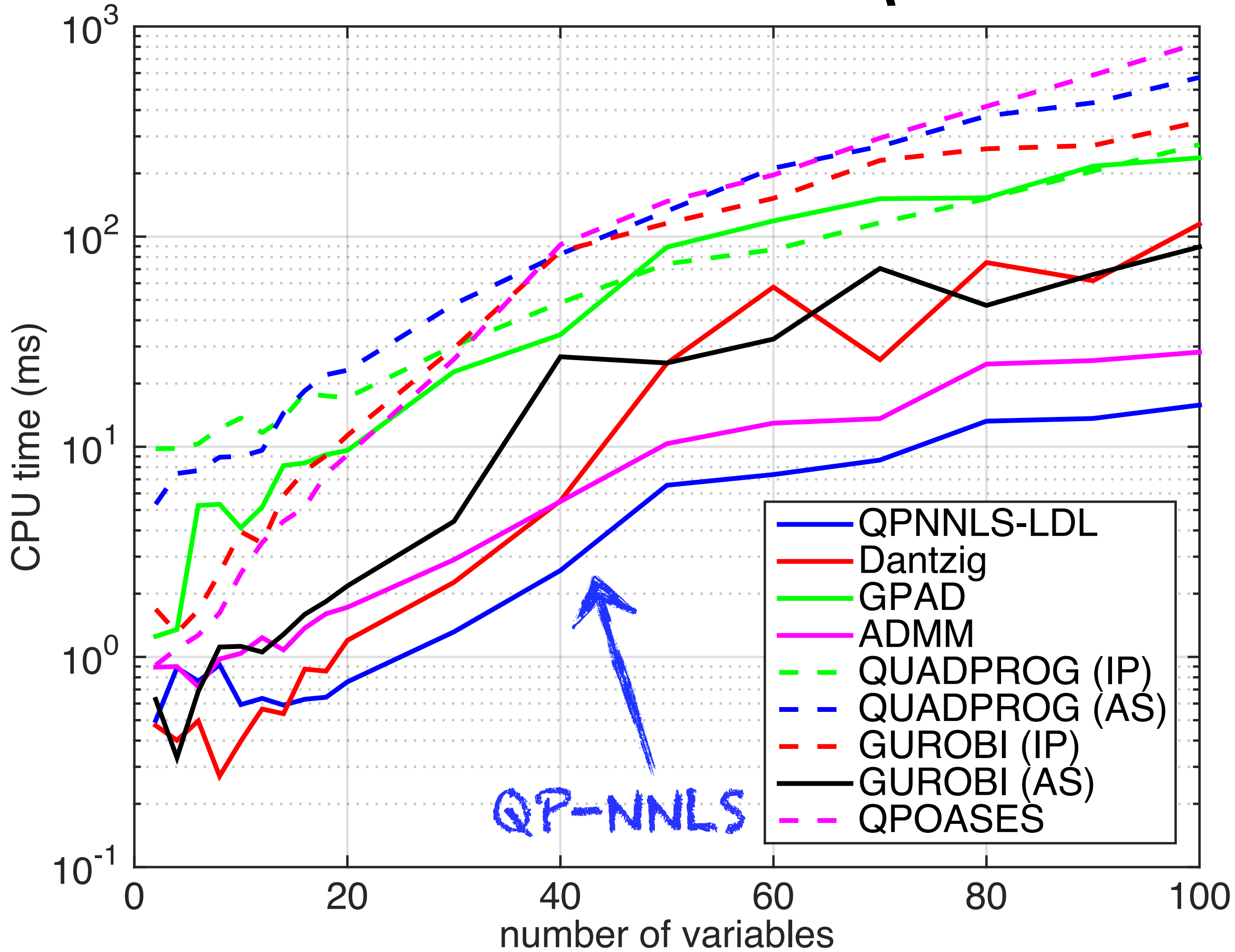
$$z^* = -\frac{1}{1+d'y^*}L^{-1}M'y^* - Q^{-1}c$$

retrieve primal solution

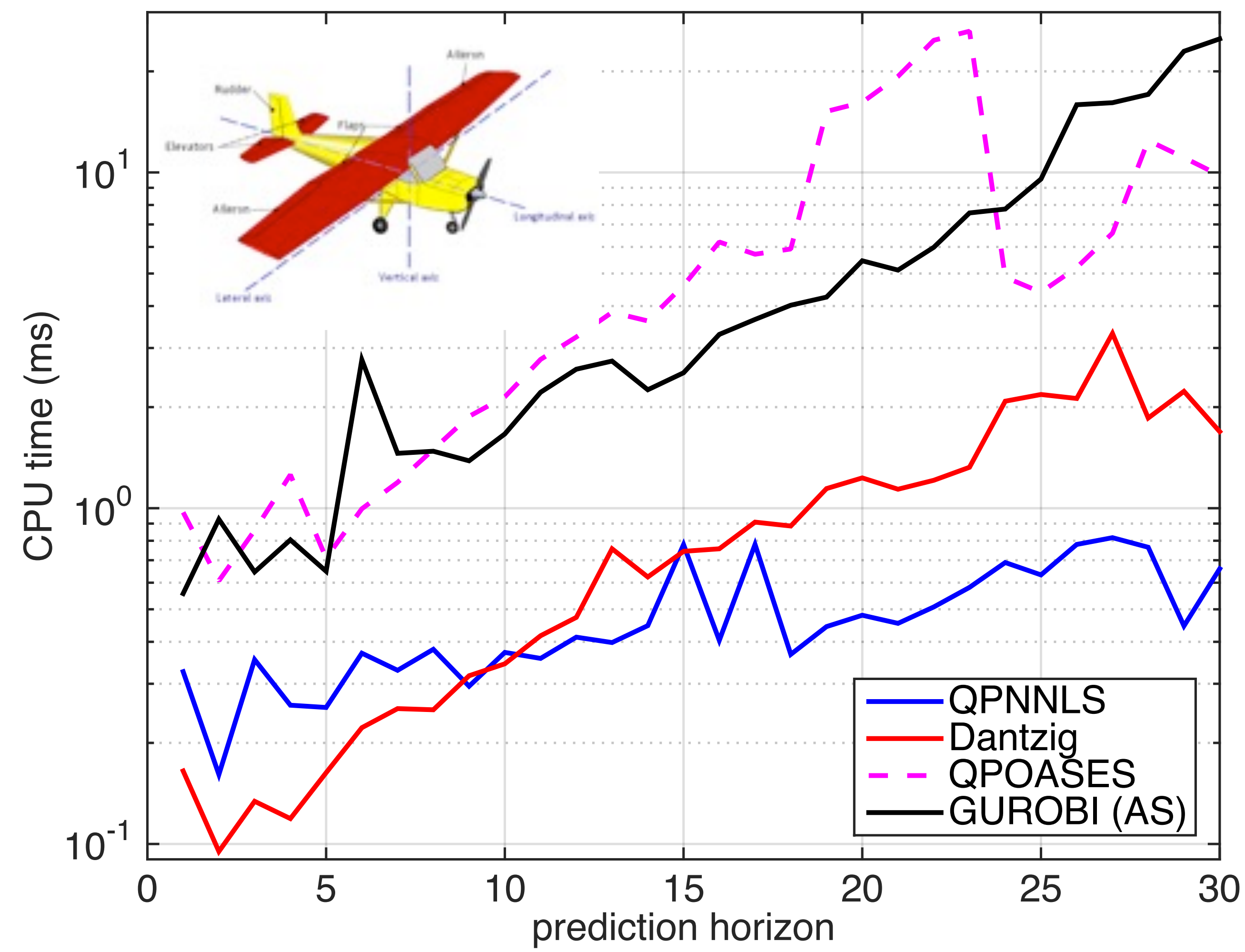
SOLVING QP VIA NNLS: NUMERICAL RESULTS

(Bemporad, IEEE TAC, 2016)

worst-case over 100 random QP instances



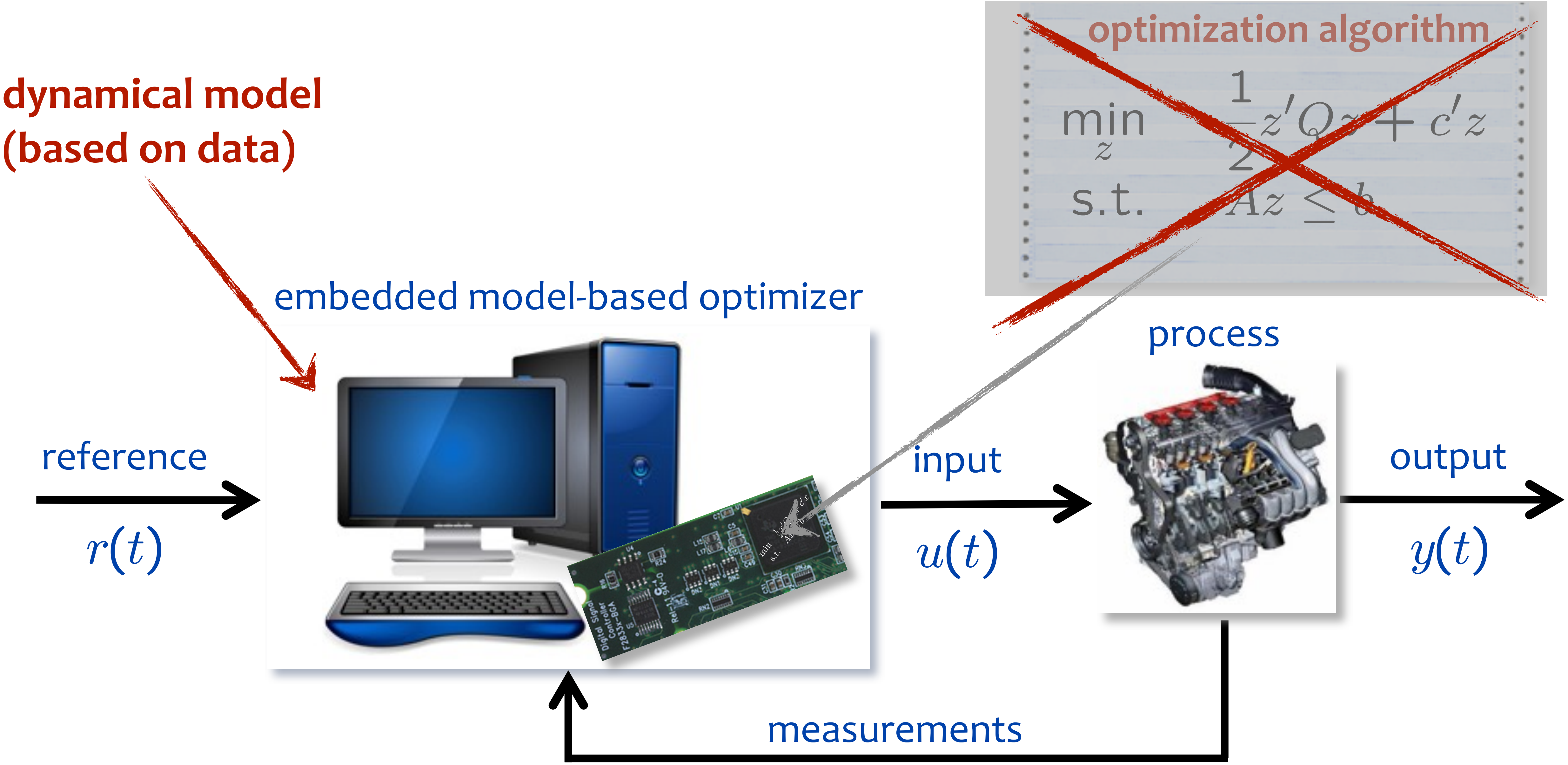
worst-case occurred during entire simulation*



* Step t=0 not considered for QPOASES not to penalize the benefits of the method with warm starting

- A rather **fast** and relatively **simple-to-code** QP solver
- Extended to solving mixed-integer QP's (Bemporad, NMPC 2015)

EMBEDDED MPC WITHOUT SOLVING QP'S ON LINE



- Can we implement MPC **without** an embedded **optimization solver** ?

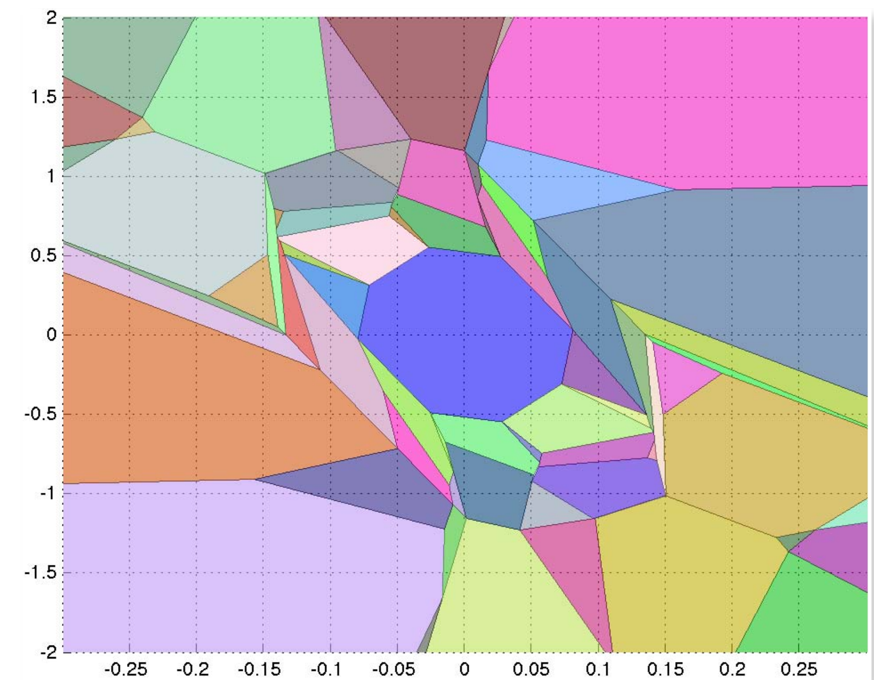
YES !

EXPLICIT MODEL PREDICTIVE CONTROL AND MULTIPARAMETRIC QP

(Bemporad, Morari, Dua, Pistikopoulos, 2002)

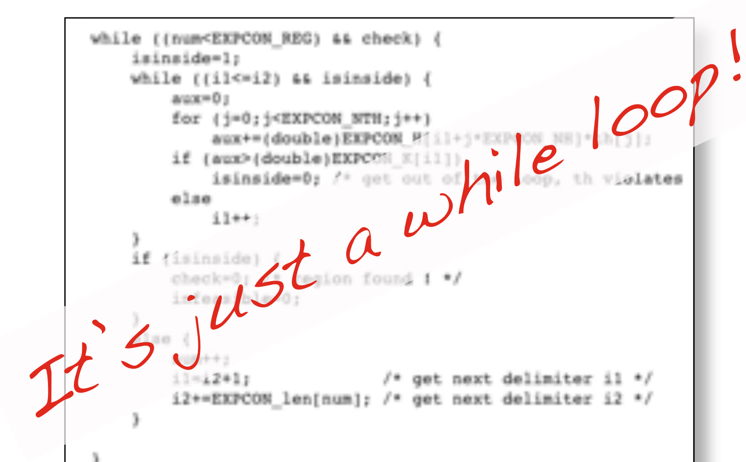
The multiparametric solution of a strictly convex QP is **continuous** and **piecewise affine**

$$
\begin{aligned}
z^*(x) = \arg\min_z \quad & \tfrac{1}{2} z' H z + x' F' z \\
\text{s.t.} \quad & G z \le W + S x
\end{aligned}
$$



Corollary: The linear MPC control law is continuous & piecewise affine !

$$
z^* = \begin{bmatrix} u_0^* \\ u_1^* \\ \vdots \\ u_{N-1}^* \end{bmatrix}
\qquad
u(x) = \begin{cases} F_1 x + g_1 & \text{if } H_1 x \le K_1 \\ \vdots & \quad \vdots \\ F_M x + g_M & \text{if } H_M x \le K_M \end{cases}
$$



NNLS FOR MULTIPARAMETRIC QP

- A variety of mpQP solvers is available

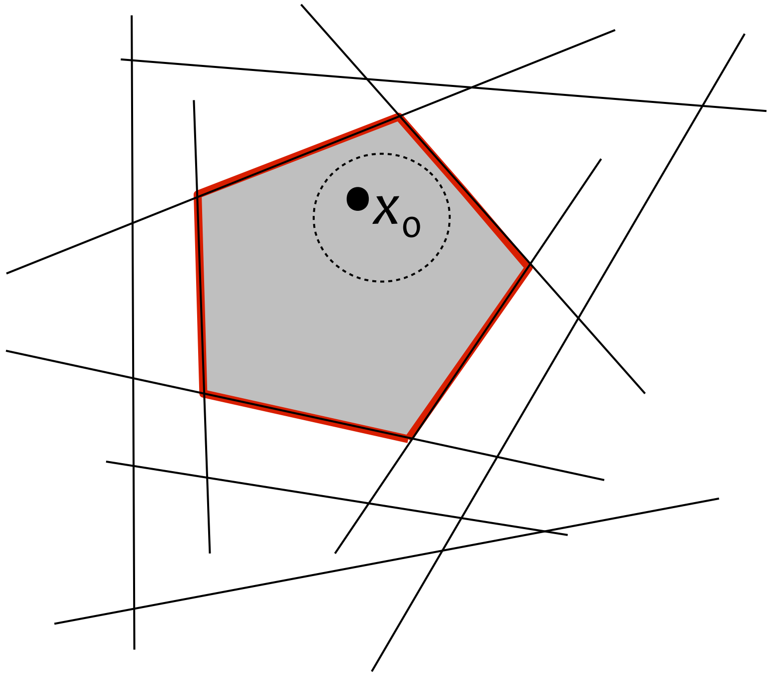
(Bemporad *et al.*, 2002) (Baotic, 2002)
(Tøndel, Johansen, Bemporad, 2003)
(Spjøtvold *et al.*, 2006)(Patrinos, Sarimveis, 2010)

- Most computations are spent in **operations on polyhedra** (=critical regions)

$$\begin{array}{rcl} \hat{G}z^*(x) & \leq & \hat{W} + \hat{S}x \\ \tilde{\lambda}^*(x) & \geq & 0 \end{array}$$

feasibility of primal solution
feasibility of dual solution

- checking **emptiness of polyhedra**
- removal of **redundant inequalities**
- checking **full-dimensionality** of polyhedra



- All such operations are usually done via **linear programming (LP)**

NNLS FOR MULTIPARAMETRIC QP

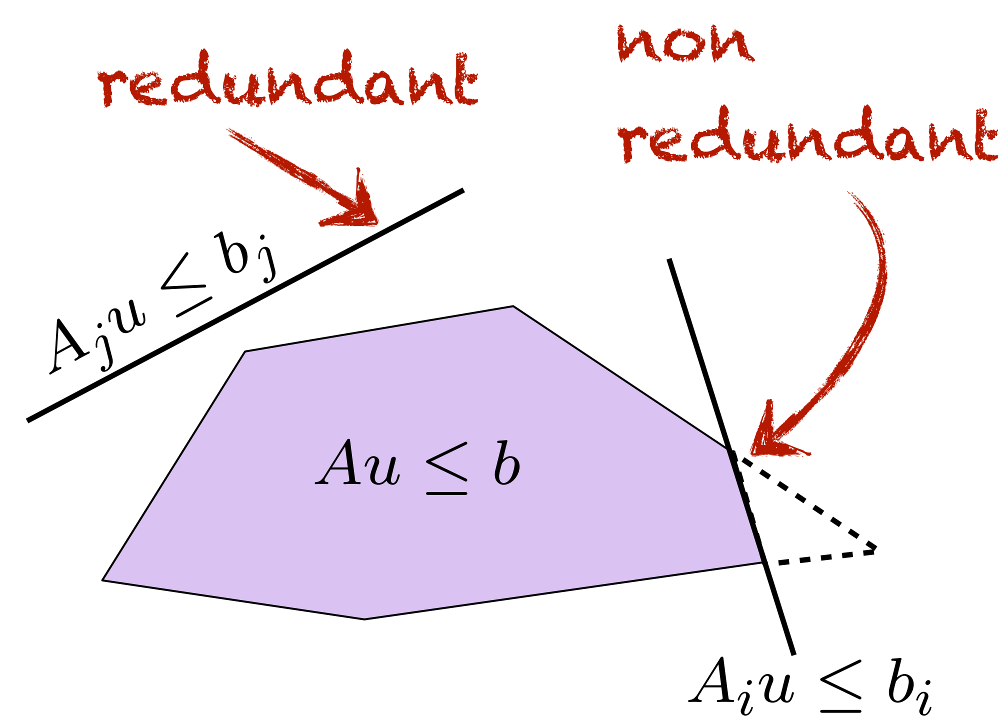
(Bemporad, IEEE TAC 2015)

- Key result:

A polyhedron $P = \{u \in \mathbb{R}^n : \ Au \leq b\}$
is nonempty iff

$$(v^*, u^*) = \arg\min_{v,u} \ \|v + Au - b\|_2^2 \quad \text{s.t.} \ v \geq 0, u \text{ free}$$

has zero residual $\|v^* + Au^* - b\|_2^2 = 0$



- Numerical results on elimination of redundant inequalities:

m	NNLS	LP
2	0.0006	0.0046
4	0.0019	0.0103
6	0.0038	0.0193
8	0.0071	0.0340
10	0.0111	0.0554
12	0.0178	0.0955
14	0.0263	0.1426
16	0.0357	0.1959

random polyhedra of $\mathbb{R}^m$ with $10m$ inequalities

NNLS = compiled Embedded MATLAB
LP = compiled C code (GLPK)

CPU time = seconds (this Mac)

- Many other polyhedral operations can be also tackled by NNLS

NNLS FOR SOLVING MPQP PROBLEMS

(Bemporad, IEEE TAC, 2015)

- New mpQP algorithm based on **NNLS** + **dual QP formulation** to compute active sets and deal with degeneracy

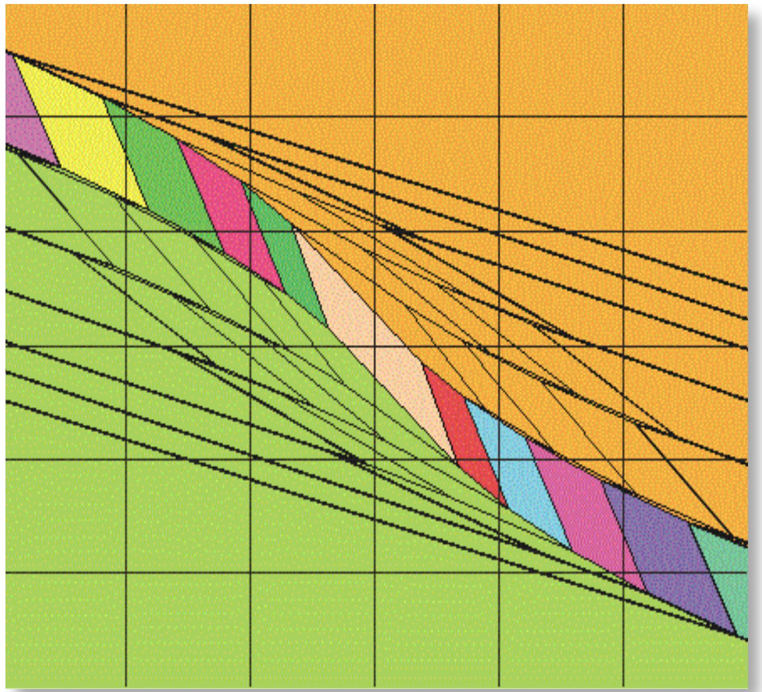
- Comparison with:
  - Hybrid Toolbox (Bemporad, 2003)
  - Multiparametric Toolbox 2.6 (with default opts) (Kvasnica, Grieder, Baotic, 2006)

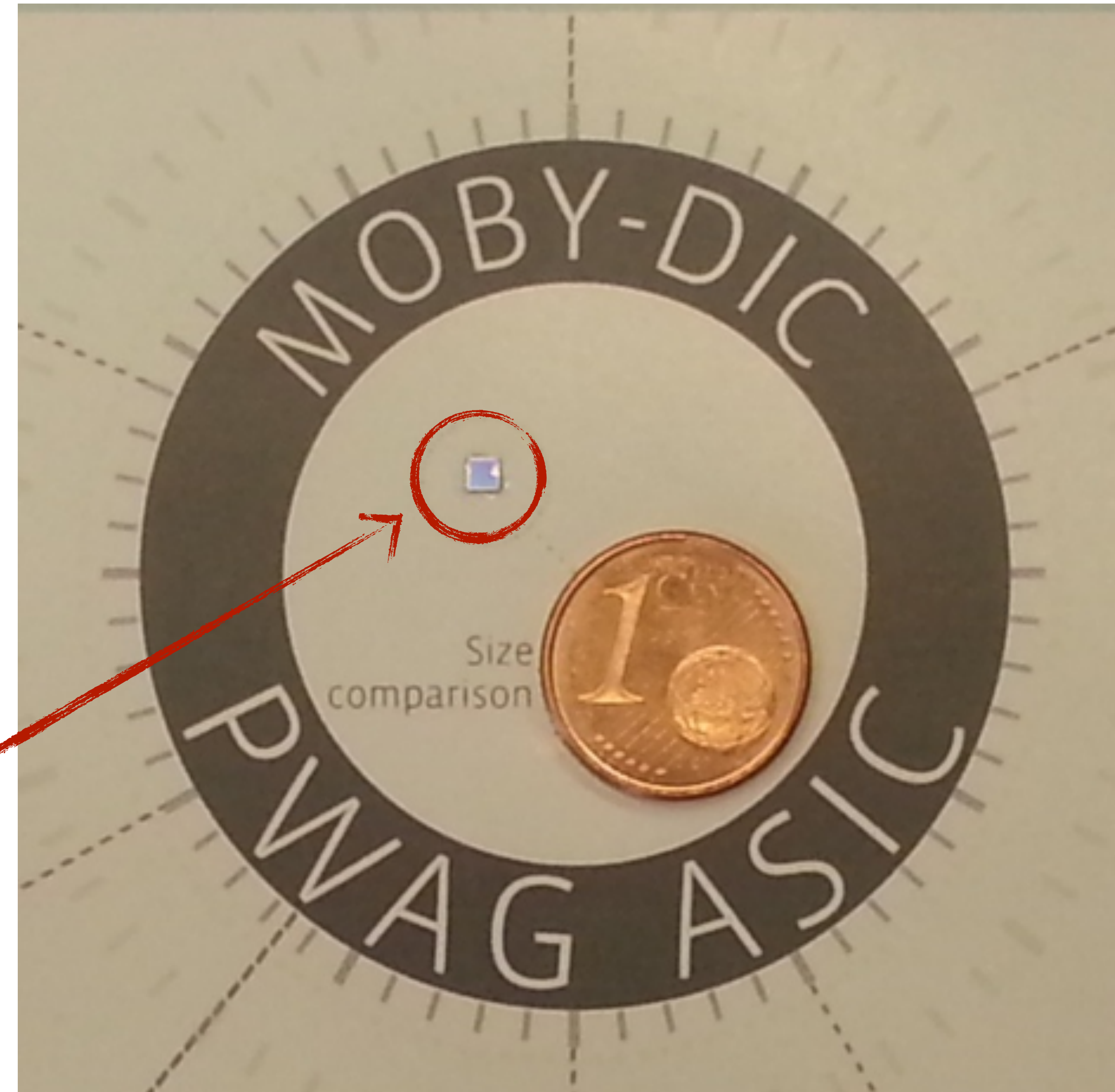
q	m	Hybrid Tbx	MPT	NNLS
4	2	0.0174	0.0256	0.0026
4	3	0.0263	0.0356	0.0038
4	4	0.0432	0.0559	0.0061
4	5	0.0650	0.0850	0.0097
4	6	0.0827	0.1105	0.0126
8	2	0.0347	0.0396	0.0050
8	3	0.0583	0.0680	0.0092
8	4	0.0916	0.0999	0.0140
8	5	0.1869	0.2147	0.0322
8	6	0.3177	0.3611	0.0586
12	2	0.0398	0.0387	0.0054
12	3	0.1121	0.1158	0.0191
12	4	0.2067	0.2001	0.0352
12	5	0.6180	0.6428	0.1151
12	6	1.2453	1.3601	0.2426
20	2	0.1029	0.0763	0.0152
20	3	0.3698	0.2905	0.0588
20	4	0.9069	0.7100	0.1617
20	5	2.2978	1.9761	0.4395
20	6	6.1220	6.2518	1.2853

- Included in MPC Toolbox 5.0 (≥R2014b)

The MathWorks (Bemporad, Morari, Ricker, 1998-2015)

HARDWARE (ASIC) IMPLEMENTATION OF EXPLICIT MPC





Technology: 90 nm, 9 metal-layer from Taiwan Semiconductor Manufacturing Company

Chip size: 1860x1860 μm^2

Memory sizes: 24 KB (tree memory); 30 KB (parameter memory)

Power supply: 2.5V (ring); 1.2V (core)

Maximum frequency: 107.5 MHz (with two inputs)

Power consumption: 38.08 mW@107.5MHz;

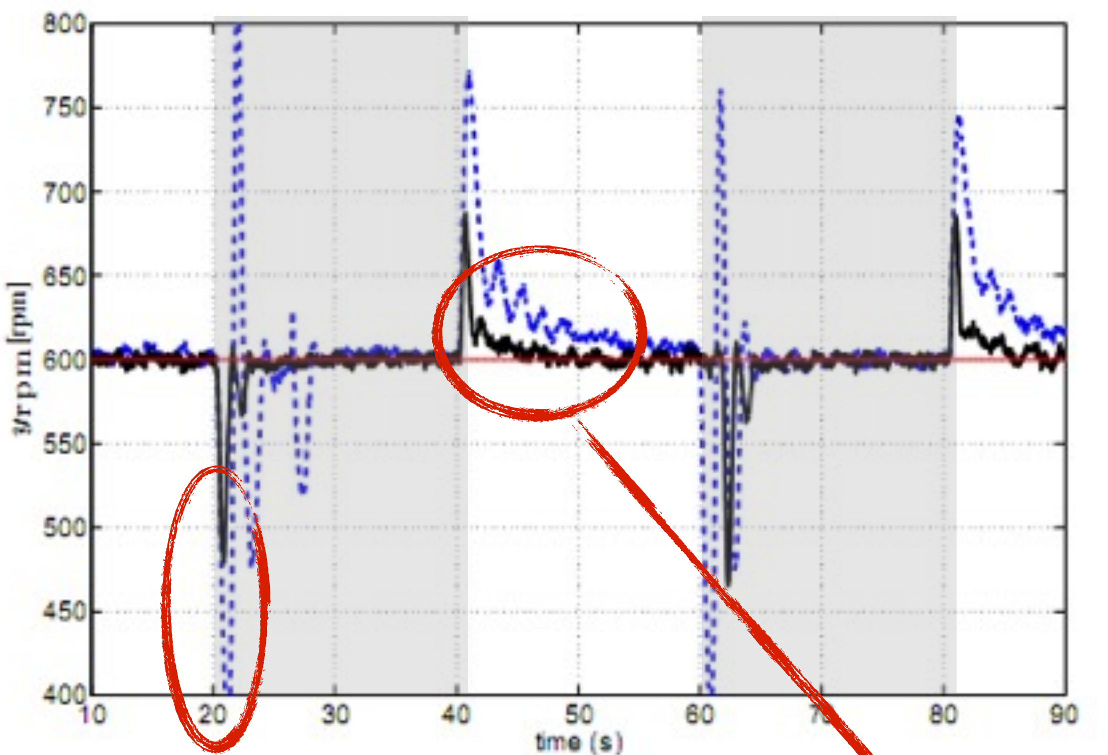
$$\frac{1}{107.5\ \text{MHz}} = 9.3\ \text{ns}$$

SEVENTH FRAMEWORK PROGRAMME

http://www.mobydic-project.eu/

EXPLICIT MPC FOR IDLE SPEED - EXPERIMENTS

(Di Cairano, Yanakiev, Bemporad, Kolmanovsky, Hrovat, 2011)



Load torque (power steering)

peak reduced by 50%

convergence 10s faster

- Sampling time = 30 ms
- Explicit MPC implemented in dSPACE MicroAutoBox rapid prototyping unit
- **Observer tuning** as much important as tuning of MPC weights !

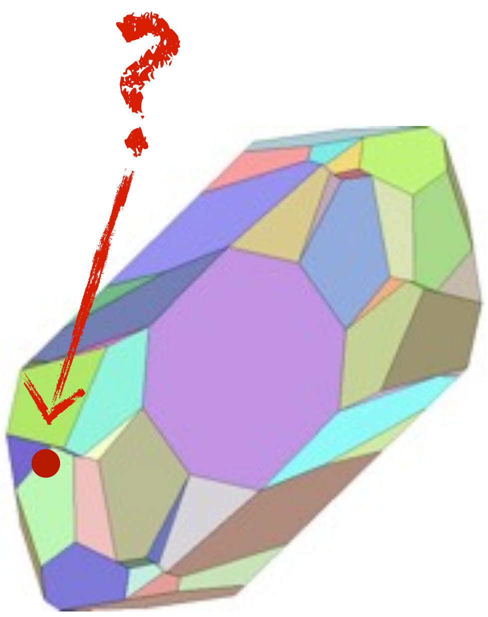
————	explicit MPC
············	baseline controller (linear)
————	set-point

COMPLEXITY OF MULTIPARAMETRIC SOLUTIONS

- The number of regions depends (exponentially) on the number of possible **combinations of active constraints**

  (weak dependence on the number of states and references)

- Explicit MPC gets less attractive when number of regions grows: too much **memory** required, too much **time** to locate state $x(t)$



- Fast **on-line** QP solvers may be preferable

When is implicit preferable to explicit MPC ?

COMPLEXITY CERTIFICATION FOR ACTIVE SET QP SOLVERS

(Cimini, Bemporad, 2016)

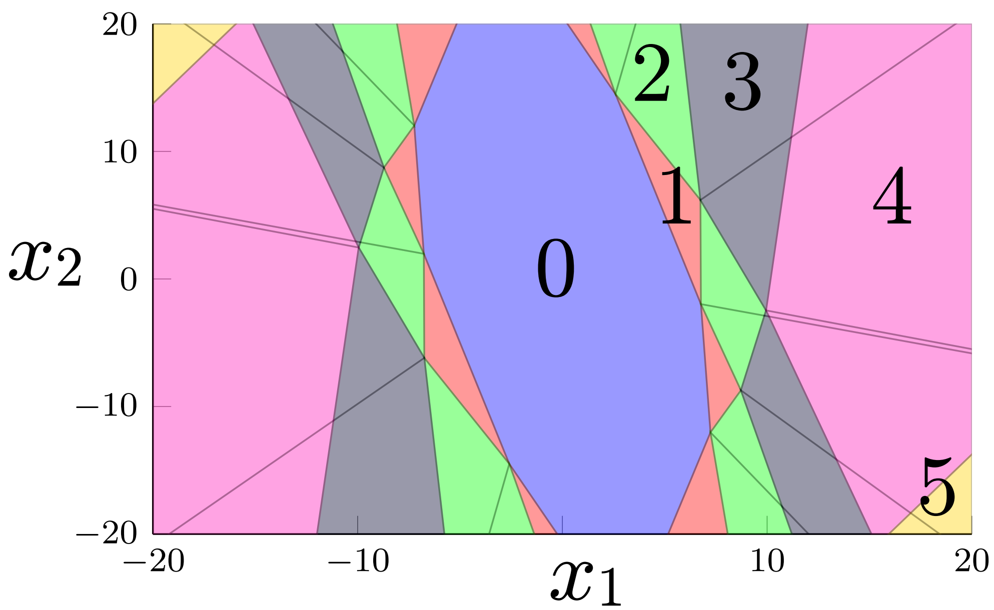
- Consider a dual active-set QP solver
  (Goldfarb, Idnani, 1983)

$$
\begin{array}{rl}
z^*(x) = \arg\min_z & \frac{1}{2}z'Hz + x'F'z \\
\text{s.t.} & Gz \leq W + Sx
\end{array}
$$

- What is the worst-case number of iterations over x to solve the QP ?

- **Key result:**

The number of iterations to solve the QP is a piecewise constant function of the parameter x



We can **exactly** quantify how many iterations (flops) the QP solver takes in the worst-case !

COMPLEXITY CERTIFICATION FOR ACTIVE SET QP SOLVERS

- Examples (from MPC Toolbox):

	inv. pend.	DC motor	nonlin. demo	AFTI 16
# vars	5	3	6	5
# constraints	10	10	18	12
# params	9	6	10	10
Explicit MPC				
# regions	87	67	215	417
max flops	3382	1689	9184	16434
max memory (kb)	55	30	297	430
Implicit MPC				
max iters	11	9	13	16
max flops	3809	2082	7747	7807
sqrt	27	9	37	33
max memory (kb)	15	13	20	16

explicit MPC is faster
in the worst-case

online QP is faster
in the worst-case

- It is possible to combine explicit and on-line QP for best tradeoff

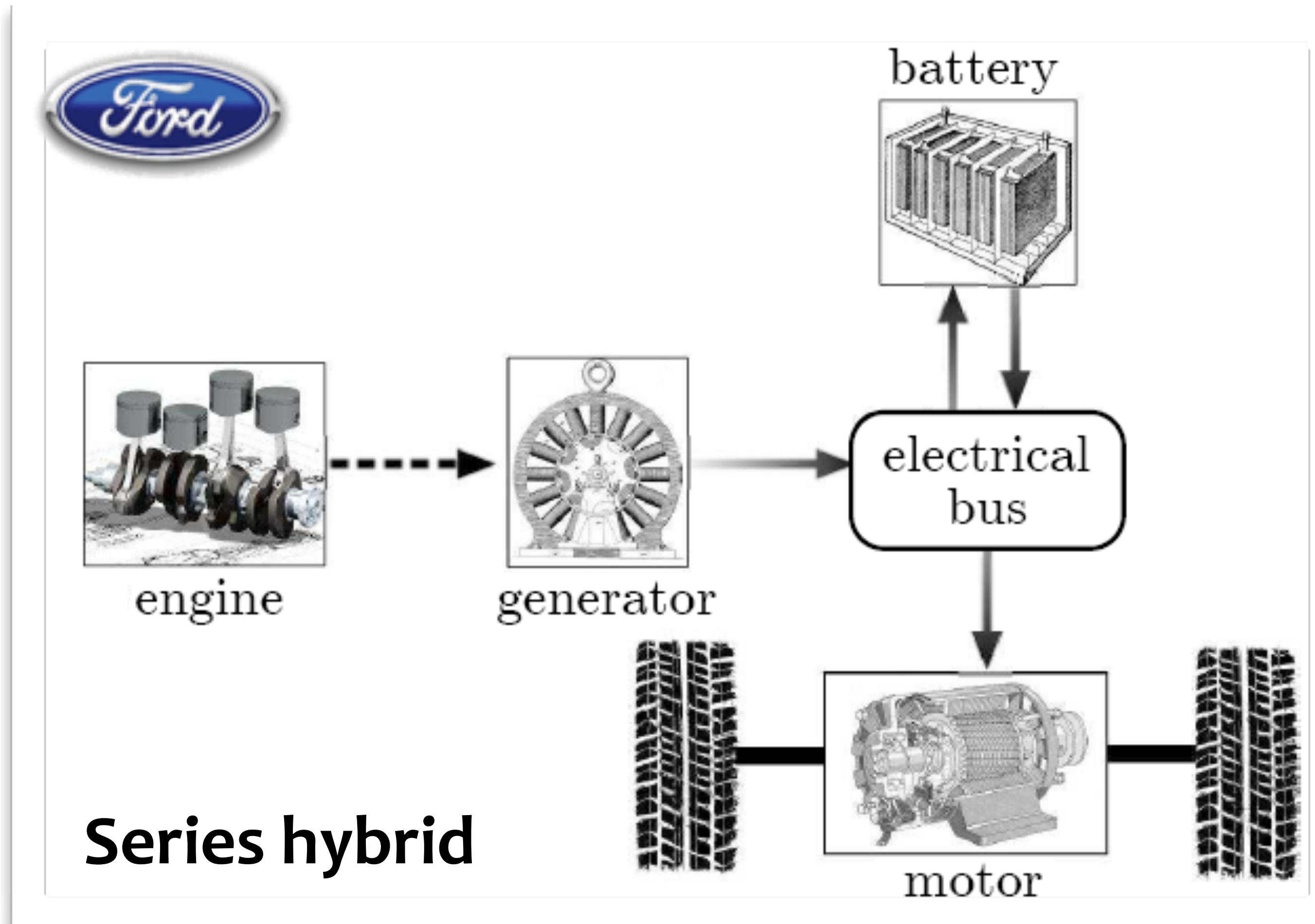
(Cimini, Bemporad, 2016)

MPC FOR POWER MANAGEMENT IN HEVS

Control problem: decide optimal generation of **mechanical power** (from engine) and **electrical power** (from battery) to satisfy **driver's power request**

What will the future power request from the driver be ?

$P_{req}(w(t))$ = driver's power request

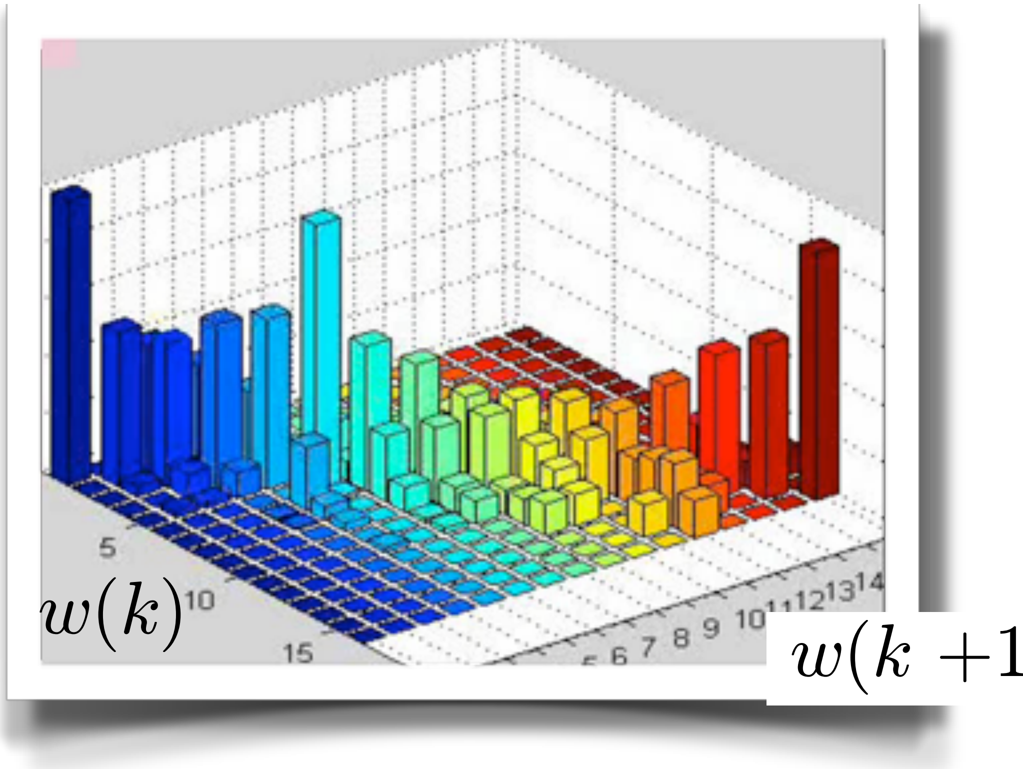


$$P_{req}(k) = P_{el}(k) + P_{mec}(k) - P_{br}(k)$$

(Di Cairano, Bernardini, Bemporad, Kolmanovsky, IEEE CST, 2014)

SIMULATION RESULTS ON REAL DRIVING DATA

- Driver's power request $w(k)$ modeled as a **stochastic** process (Markov chain)



- Real-world driving cycles (acquired on vehicle)

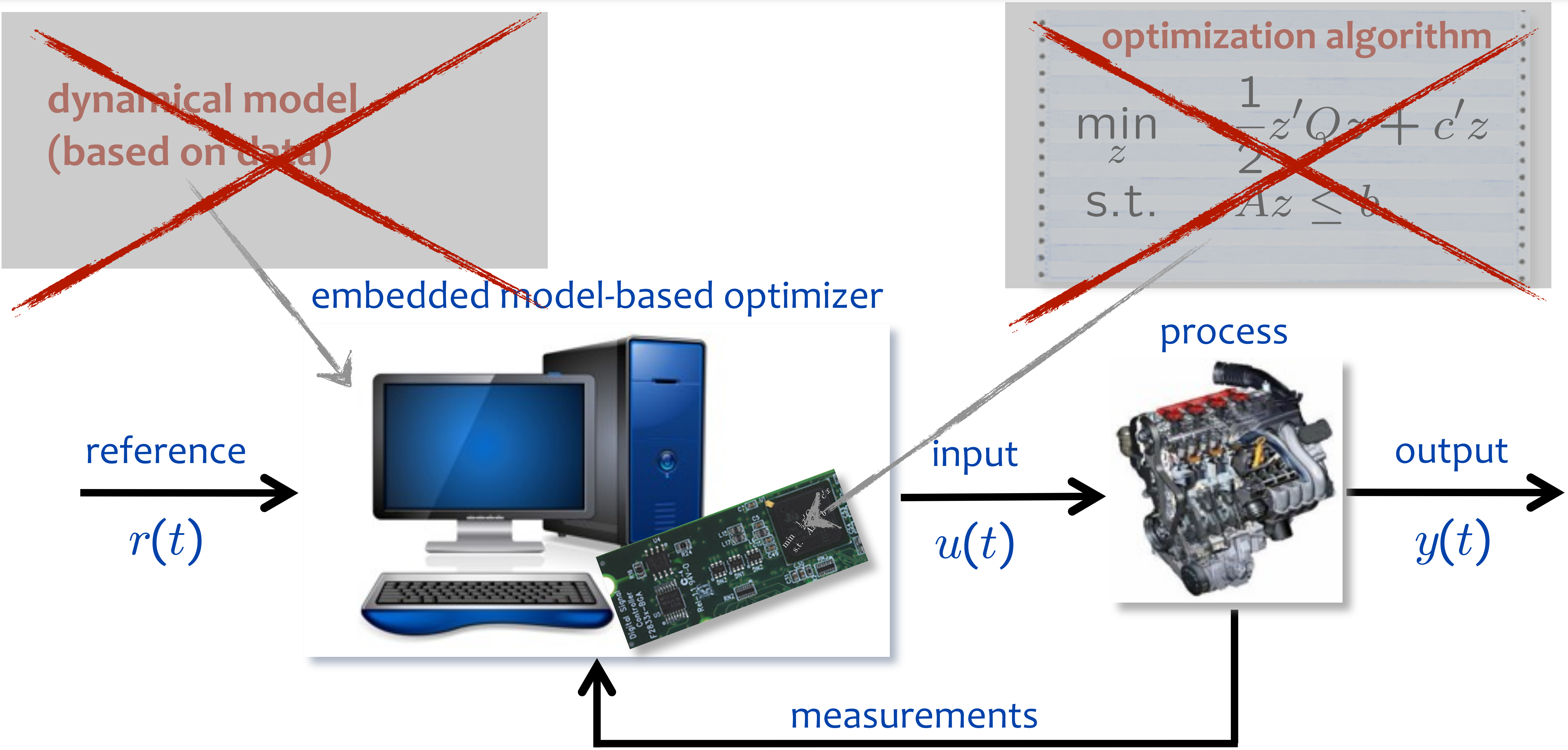
	$\|\Delta P\|$	Fuel cons.	*SoC* gain/loss	Equiv. fuel cons.	impr. wrt FTMPC
	Trace #1 - smooth accelerations				
FTMPC	37.84kW	243g	-0.05%	244g	–
→ SMPCL	14.32kW	244g	0.90%	225g	8.04%
PMPC	14.08kW	223g	-0.08%	224g	8.19%
	Trace #2 - steep accelerations				
FTMPC	80.61kW	327g	0.11%	323g	–
→ SMPCL	35.74kW	320g	1.16%	287g	11.34%
PMPC	30.67kW	287g	0.17%	282g	12.73%

deterministic MPC approach

stochastic MPC

best achievable

EMBEDDED MPC WITHOUT A MODEL

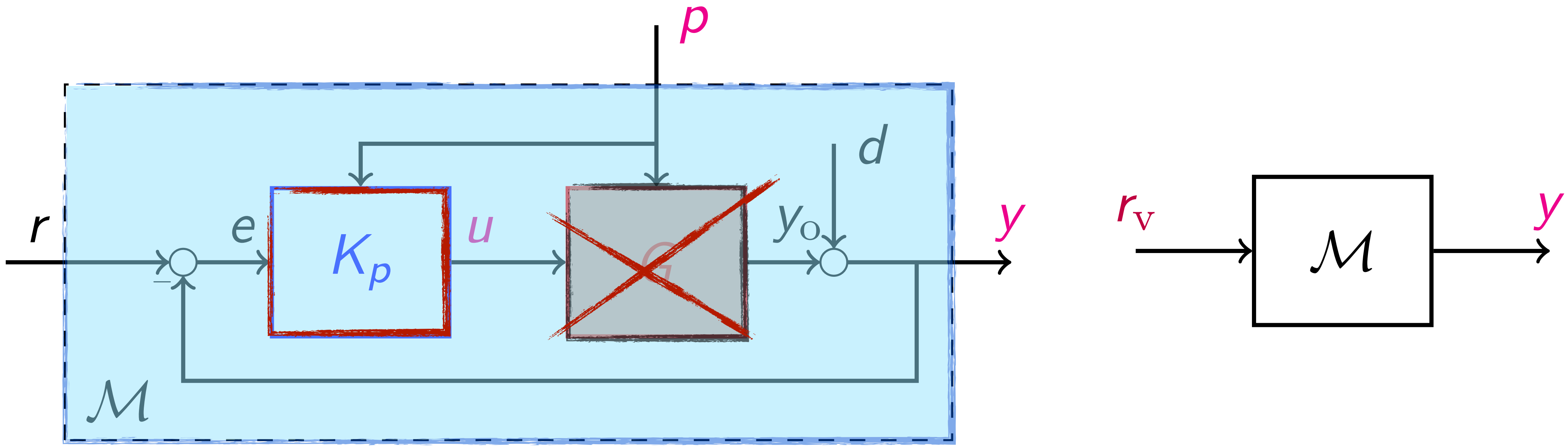


- Can we implement MPC **without** even a **model** of the process ?

YES !

DATA-DRIVEN DIRECT CONTROLLER SYNTHESIS

(Campi, Lecchini, Savaresi, 2002)
(Formentin et al., 2015)

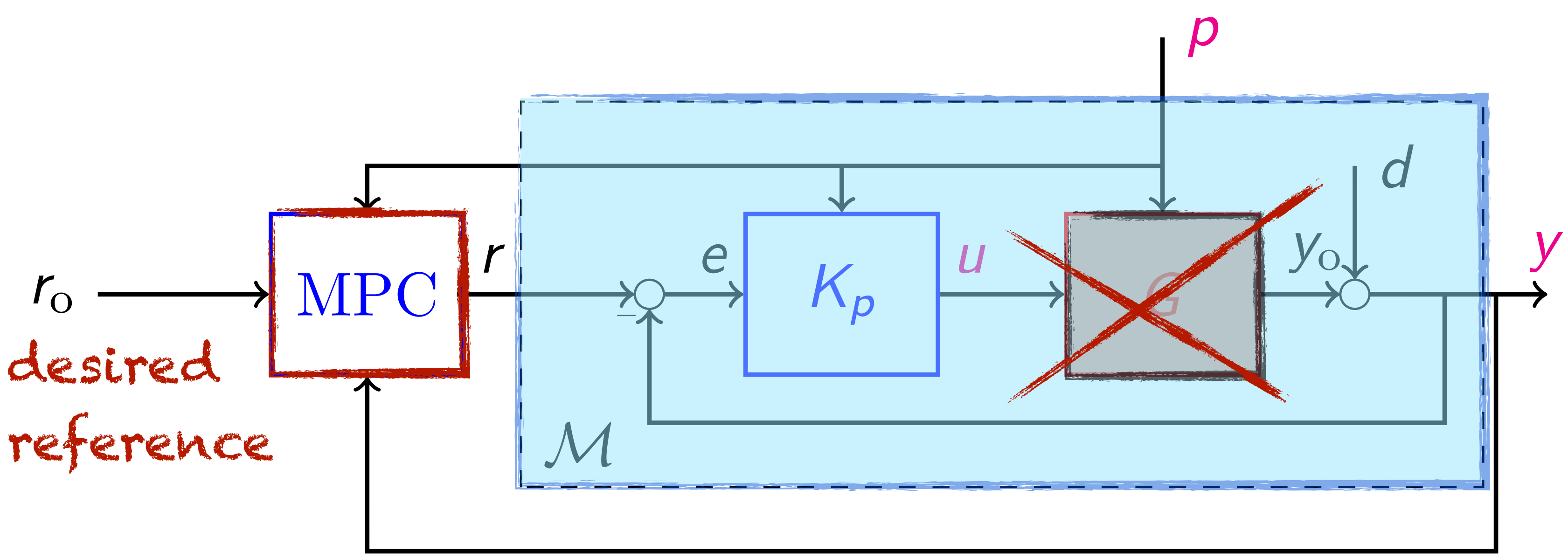


- Collect a set of observations $\{u(k),y(k),p(k)\}$, $k=1,\ldots,N$
- Specify a desired closed-loop linear model $\mathcal{M}$ from r to y
- Compute $r_v(k)=\mathcal{M}^{\#}y(k)$ from pseudo-inverse model $\mathcal{M}^{\#}$ of $\mathcal{M}$
- Identify linear (LPV) model from $e_v=r_v$ - y (virtual tracking error) to u

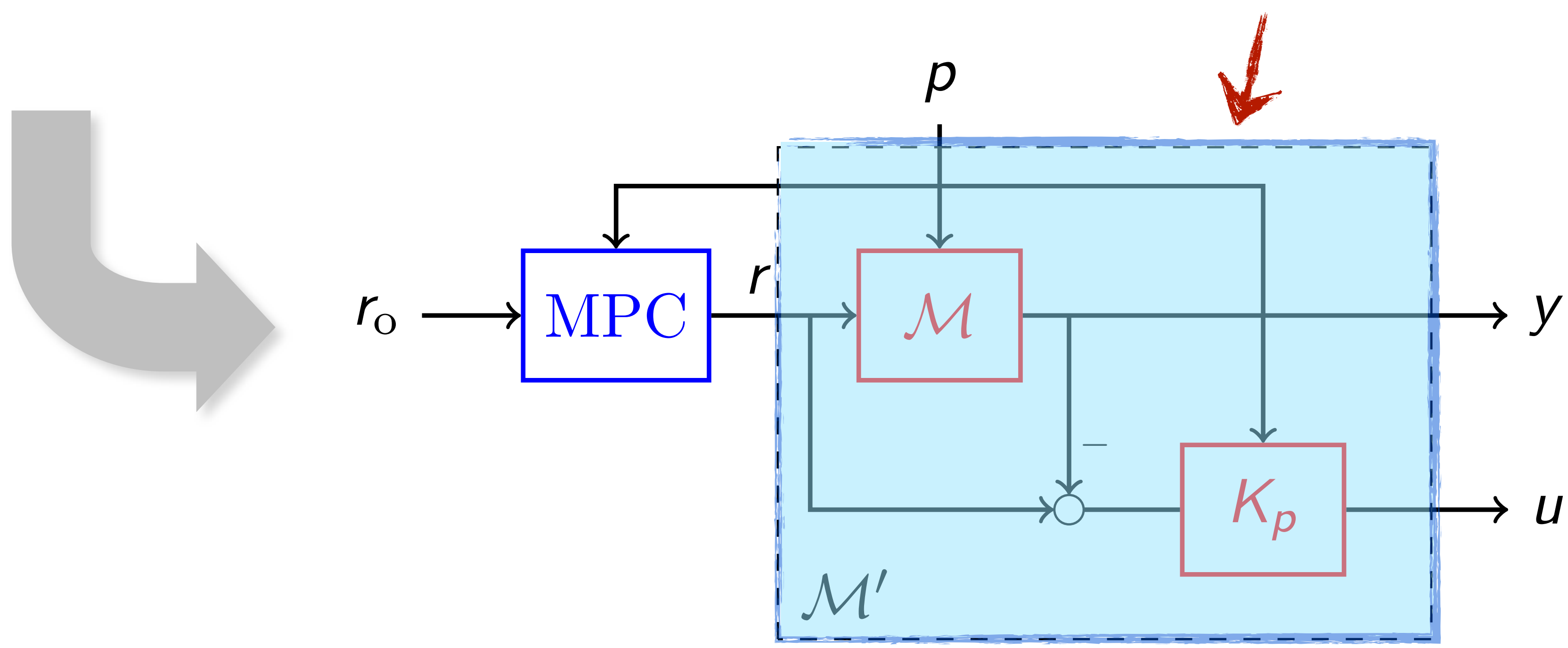
DATA-DRIVEN MPC SYNTHESIS OF CONTROLLERS

- Design a linear MPC controller (reference governor) to generate command r

(Bemporad, 1997)



Linear prediction model (totally known !)



MPC can handle constraints on inputs and outputs, and improve closed-loop performance

(Piga, Formentin, Bemporad, 2016)

- DC motor equations

$$\begin{bmatrix} \dot{\theta}(\tau) \\ \dot{\omega}(\tau) \\ \dot{I}(\tau) \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ \frac{mgl}{J} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\sin(\theta(\tau))}{\theta(\tau)} \right) \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} V(\tau)$$

(not used!)

$$y(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ I(\tau) \end{bmatrix}$$

- Desired closed-loop behavior $\mathcal{M}$ (=first-order low-pass filter):

$$x_M(k+1) = 0.99 x_M(k) + 0.01 r(k)$$
$$\theta_M(k) = x_M(k)$$

- Chosen control structure for K_p :

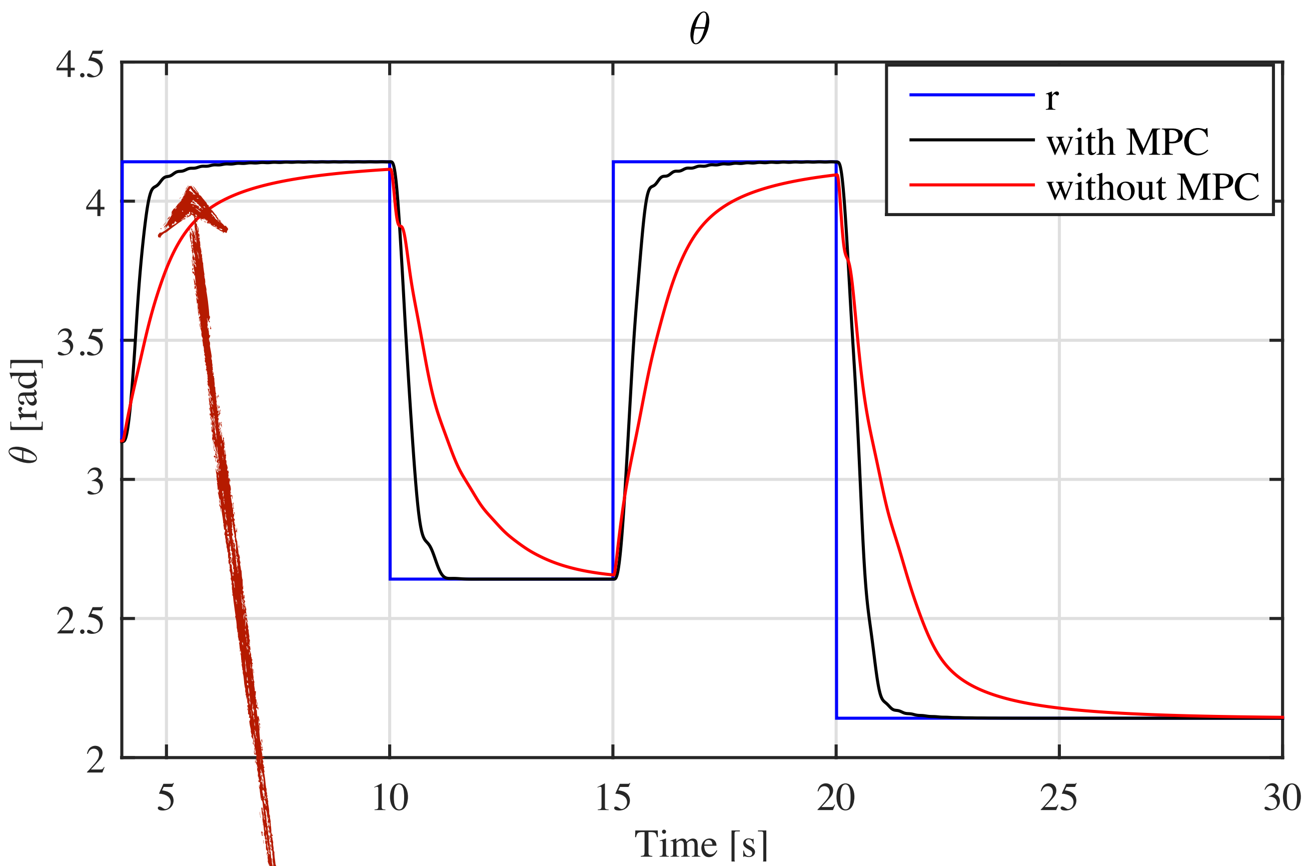
$$\xrightarrow{e_V} \boxed{\frac{z}{z-1}} \xrightarrow{e_I} \boxed{K'_p} \xrightarrow{u}$$

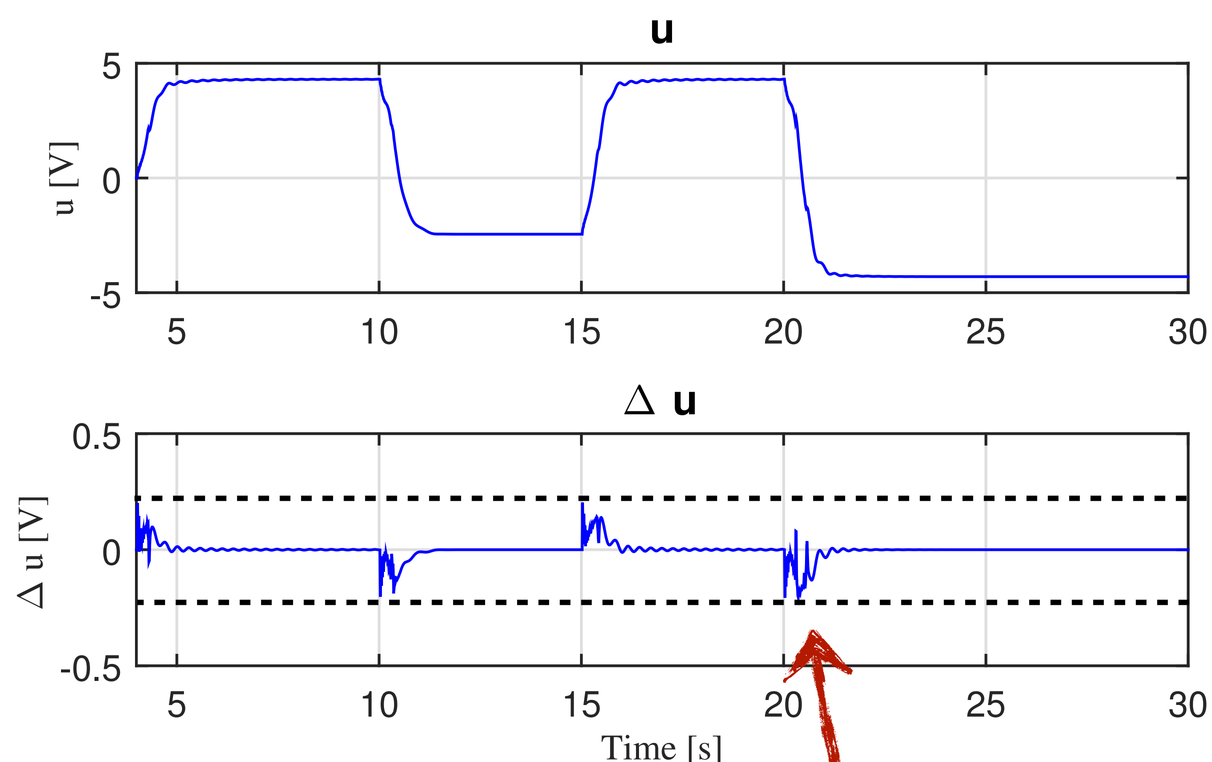
$$K'_p : u(k) = \sum_{i=1}^{4} a_i(\theta(k)) u(k-i) + \sum_{j=0}^{3} b_j(\theta(k)) e_I(k-j)$$

- MPC design w/ soft constraints on inputs, outputs and input increments

DATA-DRIVEN MPC SYNTHESIS OF CONTROLLERS - AN EXAMPLE

- Experimental results





desired tracking performance achieved

constraints on input increments satisfied

No model of open-loop process identified to design the MPC controller !

CONCLUSIONS

- MPC can easily handle **multivariable control** problems with **constraints** in an **optimized** way, it's **easy to design** and **reconfigure,** it handles **uncertainty**

- Long history of success in the *process industries* since the 80's, now spreading in the automotive industry (and many others)

- MATLAB design/calibration tools and production-ready C-code are available

ODYS

Is MPC a mature technology for production in fast embedded applications ?