# MODEL PREDICTIVE CONTROL: DREAMS, POSSIBILITIES, AND REALITY

**Alberto Bemporad**

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

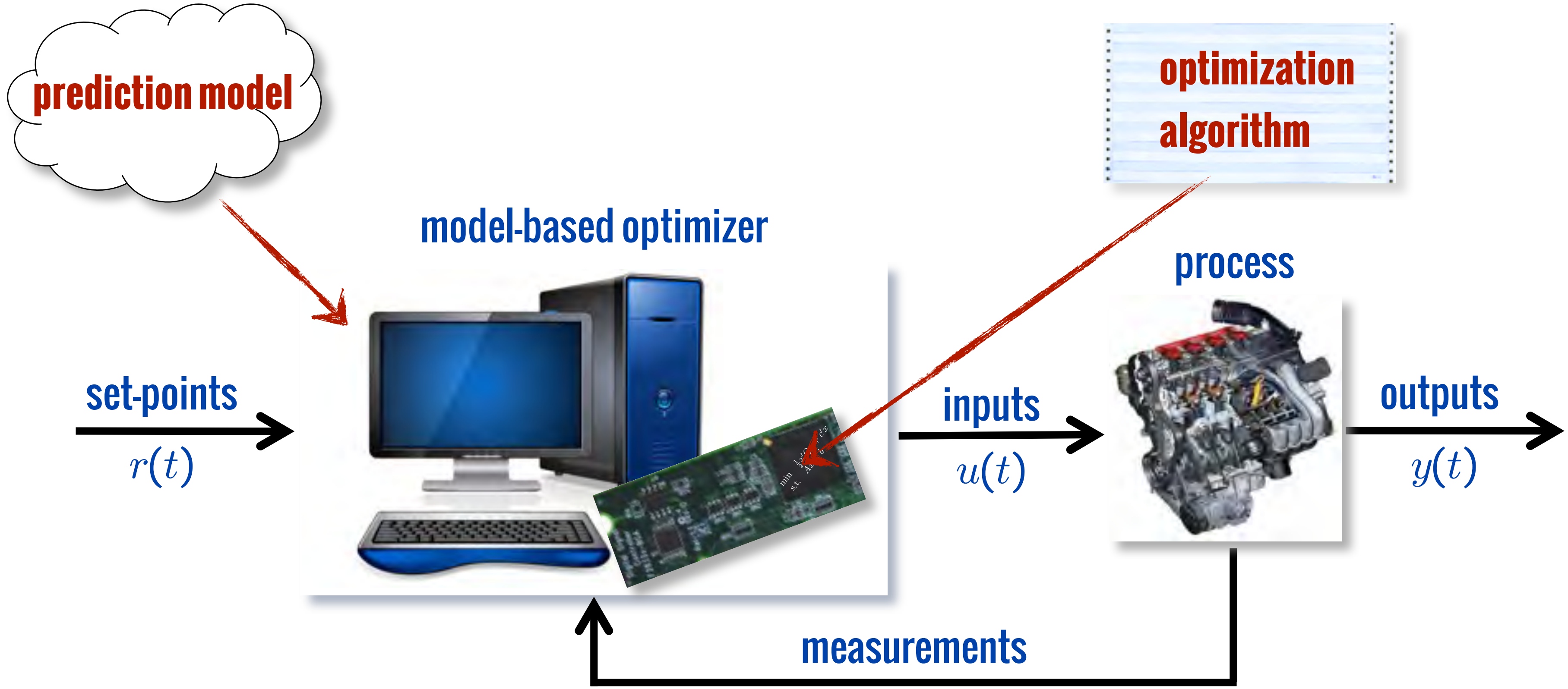
IMT SCHOOL FOR ADVANCED STUDIES LUCCA

NMPC'18, Madison, WI - August 19, 2018

# OUTLINE

- Embedded optimization for MPC (**dreams**, **possibilities**)
- Data-driven MPC (**dreams**, **possibilities**)
- MPC in the automotive industry (**reality**)
- Conclusions

# MODEL PREDICTIVE CONTROL (MPC)




simplified                                                                 likely
Use a dynamical model of the process to predict its future evolution and choose ~~the “best” control~~ action
a good

# MPC IN INDUSTRY

- The MPC concept for process control dates back to the 60's


Discrete Dynamic Optimization
Applied to On-Line Optimal Control
MARSHALL D. RAFAL and WILLIAM F. STEVENS

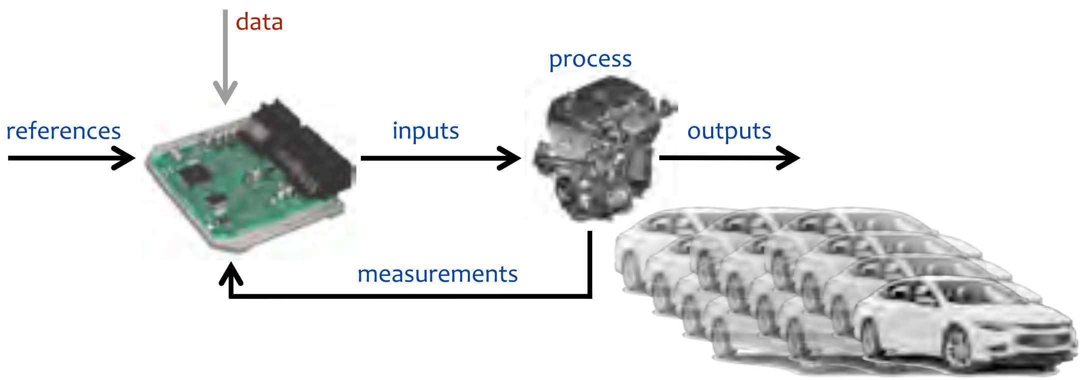
(Rafal, Stevens, AiChE Journal, 1968)

- MPC used in the process industries since the 80's

(Qin, Badgewell, 2003) (Bauer, Craig, 2008)

- Research in MPC is still very active ! (274 authors in NMPC'18)

- Most **automotive OEMs** are looking into MPC solutions today

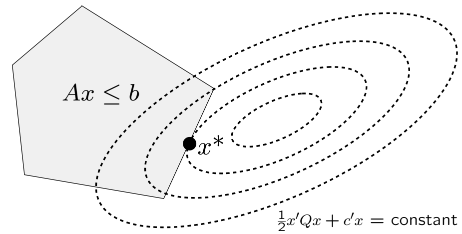


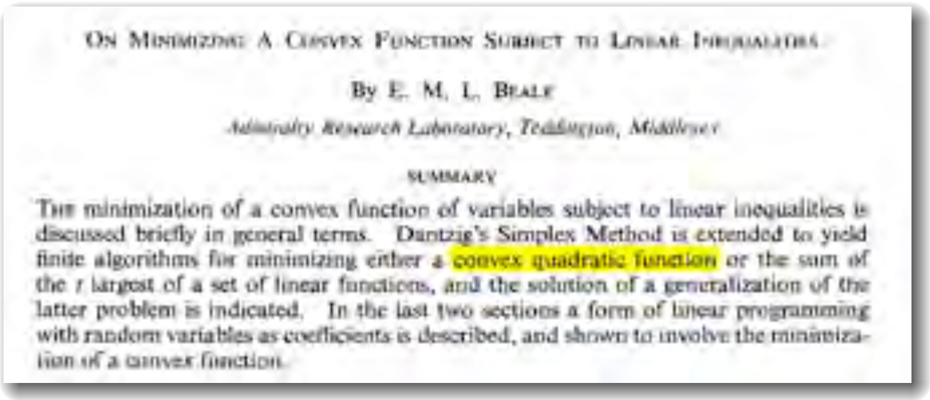
# EMBEDDED LINEAR MPC AND QUADRATIC PROGRAMMING

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

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

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



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 $r$ 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.

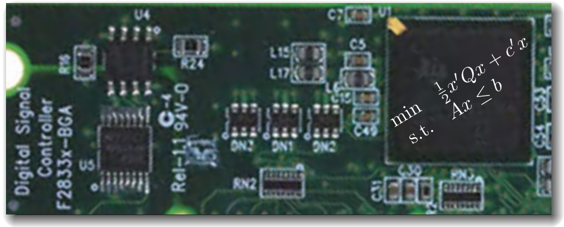
(Beale, 1955)

A rich set of good QP algorithms is available today

- Which QP algorithms are suitable for implementation in **embedded systems** ?

# MPC IN A PRODUCTION ENVIRONMENT

## Key requirements for deploying MPC in production:



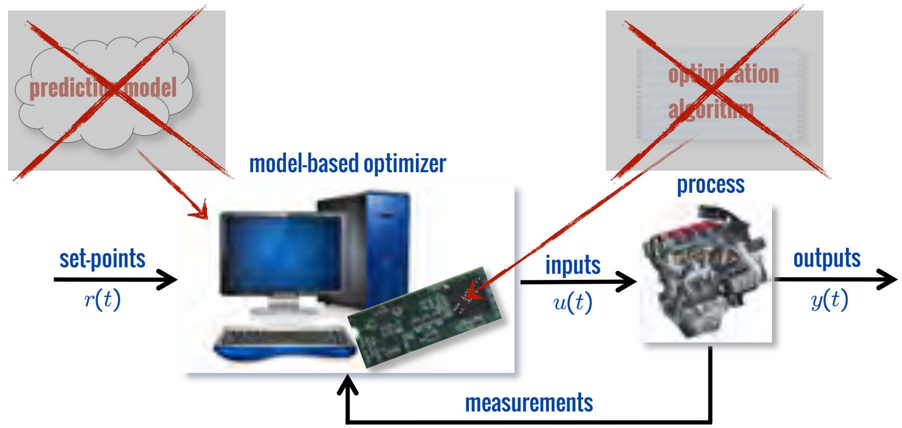

1. **speed (throughput)**
   - worst-case execution time less than sampling interval
   - also fast on average (to free the processor to execute other tasks)

2. limited **memory and CPU power** (e.g., 150 MHz / 50 kB)

3. **numerical robustness** (single precision arithmetic)

4. **certification** of worst-case execution time


5. **code simple enough** to be validated/verified/certified
   (library-free C code, easy to check by production engineers)




**Key challenges**

- **Numerical solvers**
  - Fast/robust/compact/certifiable embedded optimization solvers
  - Solver and MPC problem should be coupled for most efficiency
  - Should we avoid real-time optimization (explicit MPC/ function regression)?

- **Prediction models**
  - Getting the predicition model(s) is usually the most time-consuming phase
  - Leverage on modern SYS-ID / machine learning methods (=less physical modeling)
  - Should we avoid identifying open-loop prediction models at all?

# EMBEDDED OPTIMIZATION - SOME POSSIBILITIES

# FAST GRADIENT PROJECTION

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

- Solve (dual) QP by **fast gradient method**

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

$$\begin{array}{rcl} K & = & Q^{-1}G' \\ g & = & Q^{-1}Fx \\ L & \geq & \frac{1}{\lambda_{\max}(GQ^{-1}G')} \\ \beta_k & = & \max\{\frac{k-1}{k+2}, 0\} \end{array}$$

$$\begin{array}{rcl} w^k & = & y^k + \beta_k(y^k - y^{k-1}) \\ z^k & = & -Kw^k - g \\ s^k & = & \frac{1}{L}Gz^k - \frac{1}{L}(W + Sx) \\ y^{k+1} & = & \max\left\{w^k + s^k, 0\right\} \end{array}$$

```
while k<maxiter
  beta=max((k-1)/(k+2),0);
  w=y+beta*(y-y0);
  z=-(iMG*w+iMc);
  s=GL*z-bL;

  y0=y;


  % Termination
  if all(s<=epsGL)
    gapL=-w'*s;
    if gapL<=epsVL
      return
    end
  end

  y=w+s;
  k=k+1;
end
```

- Very **simple to code**
- Convergence rate: $f(x^k) - f(x^*) \leq \frac{2L}{(k+2)^2}\|z_0 - z^*\|_2^2$
- Tight bounds on maximum number of iterations
- Extended to **mixed-integer quadratic programming (MIQP)** (Naik, Bemporad, 2017)

# REGULARIZED ADMM FOR QUADRATIC PROGRAMMING

(Banjac, Stellato, Moehle, Goulart, Bemporad, Boyd, 2017)

- Robust "regularized" ADMM iterations:

$$
\begin{aligned}
z^{k+1} &= -(Q + \rho A^T A + \epsilon I)^{-1}(c - \epsilon z_k + \rho A^T(u^k - z^k)) \\
s^{k+1} &= \min\{\max\{A z^{k+1} + y^k, \ell\}, u\} \\
u^{k+1} &= u^k + A z^{k+1} - s^{k+1}
\end{aligned}
$$

- Works for any $Q \succeq 0$, $A$, and choice of $\epsilon > 0$
- **Simple** to code, **fast**, **robust**, and **free**
- Only needs to factorize $\begin{bmatrix} Q + \epsilon I & A' \\ A & -\frac{1}{\rho} I \end{bmatrix}$ once
- Algorithm implemented in osQP solver

  **(Python interface: $\approx$ 20,000 downloads)**

`http://osqp.org`

- Extended to solve **mixed-integer quadratic programming** problems

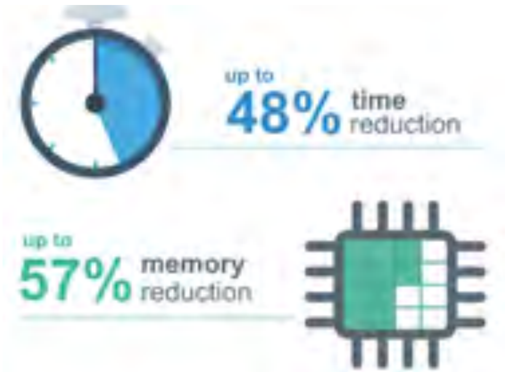
  (Stellato, Naik, Bemporad, Goulart, Boyd, 2018)

# ODYS QP SOLVER

- General purpose QP solver designed for **industrial production**

$$
\begin{array}{ll}
\min_z & \frac{1}{2}z'Qz + c'z \\
\text{s.t.} & Az \leq b \\
 & Ez = f
\end{array}
$$


- Implements a dual-active set method for QP + **optimized for MPC**
- Emphasis on **robustness** (especially in **single precision**), **speed** of execution, **low memory** requirements
- Completely written in **ANSI-C** (no libraries)
- Currently evaluated for production by some major OEMs




**`http://odys.it/qp`**

# QP SOLVERS - AN EXPERIMENTAL COMPARISON

- **Experimental setup**:
  - PC with MATLAB/Simulink
  - RS232 adapter
  - TMS320F28335 DSP (**150 MHz**)

| vars × constr. | ODYS QP | GPAD | ADMM |
|---|---|---|---|
| 4× 16 | 0.12 ms | 0.33 ms | 1.4 ms |
| 8× 24 | 0.44 ms | 1.1 ms | 4 ms |
| 12×32 | 1.2 ms | 2.6 ms | 8.2 ms |

- Active set (AS) methods are usually the best on small/medium problems:
  - excellent quality solutions within few iterations
  - behavior is more predictable (less sensitive to preconditioning)
  - no linear algebra libraries required

# SOLVING QP'S USING LEAST SQUARES

- Least Squares (LS): (Legendre, 1805) (Gauss, 1809)

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

In MATLAB:

```
>> v=A\b
```

(one character !)

- Nonnegative Least Squares (NNLS): (Lawson, Hanson, 1974)

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

very simple to solve
(750 chars in Embedded
MATLAB)

- Bounded-Variable Least Squares (BVLS): (Stark, Parker, 1995)

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

**See NMPC'18 talk TuMRb1.2**

(Saraf, Zanon, Bemporad, NMPC, 2018)

# SOLVING QP'S VIA NONNEGATIVE LEAST SQUARES

(Bemporad, 2016)

- Complete the squares and transform QP to **least distance problem** (LDP)

$$\begin{array}{ll} \min_z & \frac{1}{2}z'Qz + c'z \\ \text{s.t.} & Gz \leq g \end{array} \quad \xrightarrow[u \triangleq Lz + L^{-T}c]{Q = L'L} \quad \begin{array}{ll} \min_u & \frac{1}{2}\|u\|^2 \\ \text{s.t.} & Mu \leq d \end{array}$$

- An LDP can be solved by the NNLS (Lawson, Hanson, 1974)

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

$$\begin{array}{rcl} M & = & GL^{-1} \\ d & = & b + GQ^{-1}c \end{array}$$

- If residual $= 0$ then QP is infeasible. Otherwise set

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

- Extended to solving mixed-integer QP's (Bemporad, NMPC, 2015)

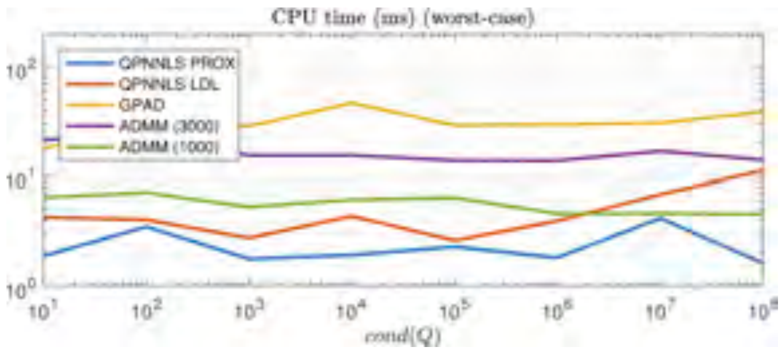
# SOLVING QP'S VIA NNLS AND PROXIMAL POINT ITERATIONS

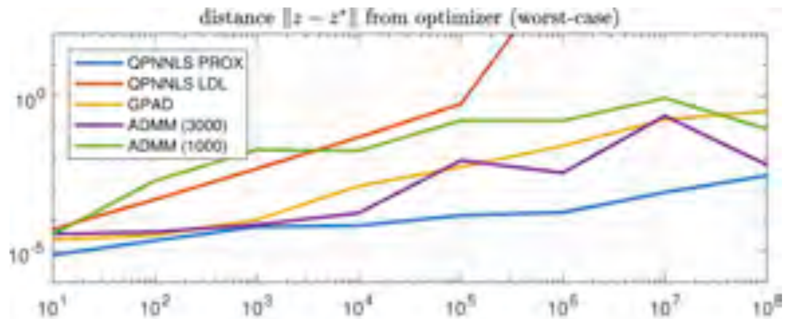
(Bemporad, 2018)

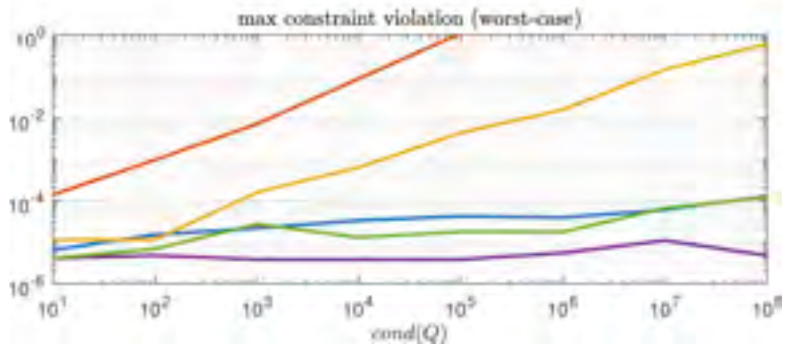
- Solve QP via NNLS within **proximal-point iterations**

$$
\begin{array}{rl}
z_{k+1} = \arg\min_z & \frac{1}{2}z'Qz + c'z + \frac{\epsilon}{2}\|z - z_k\|_2^2 \\
\text{s.t.} & Az \leq b \\
& Gx = g
\end{array}
$$

- Advantage: **numerical robustness**, as $Q + \epsilon I$ can be arbitrarily well conditioned




**single precision** arithmetic
30 vars, 100 constraints
(Macbook Pro 3 GHz Intel Core i7)

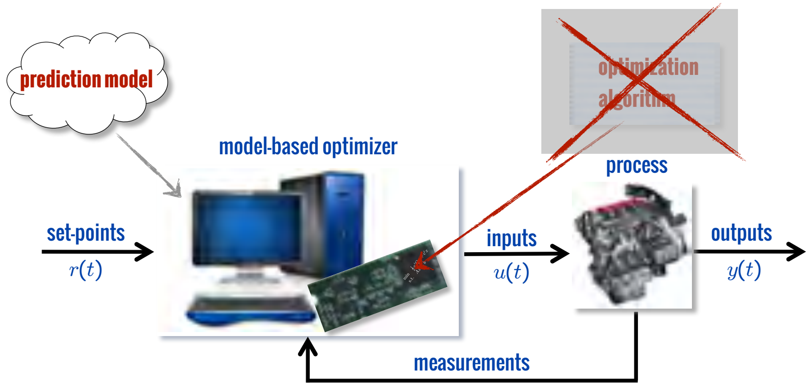
- Extended to solve **MIQP** problems

(Naik, Bemporad, 2017) $\Rightarrow$ **TuAPo1.15**

# EXPLICIT MPC

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

  (Bemporad, Morari, Dua, Pistikopoulos, 2000)
  (Seron, De Donà, Goodwin, 2000)
  (Johansen, Petersen, Slupphaug, 2000)




$$z^*(x) = \arg\min_z \quad \frac{1}{2}z'Qz + x'F'z$$
$$\text{s.t.} \quad Gz \leq W + Sx$$

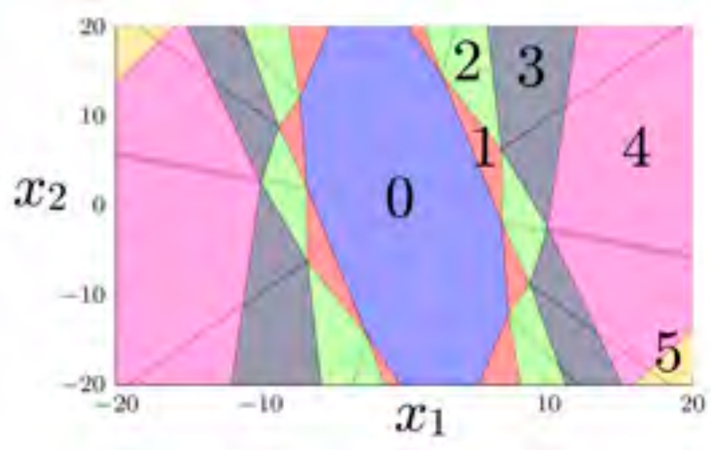
- Explicit MPC enables implementing MPC **without an on-line solver**

- New mpQP solver based on NNLS available (Bemporad, 2015) and included in **MPC Toolbox** since version R2014b

Is explicit MPC better than on-line QP (=implicit MPC) ?

# COMPLEXITY CERTIFICATION FOR ACTIVE-SET QP SOLVERS

- **Result**: The **number of iterations** to solve the QP via a dual active-set method is a **piecewise constant function** of the parameter $x$




(Cimini, Bemporad, 2017)

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

- Examples (from MPC Toolbox):

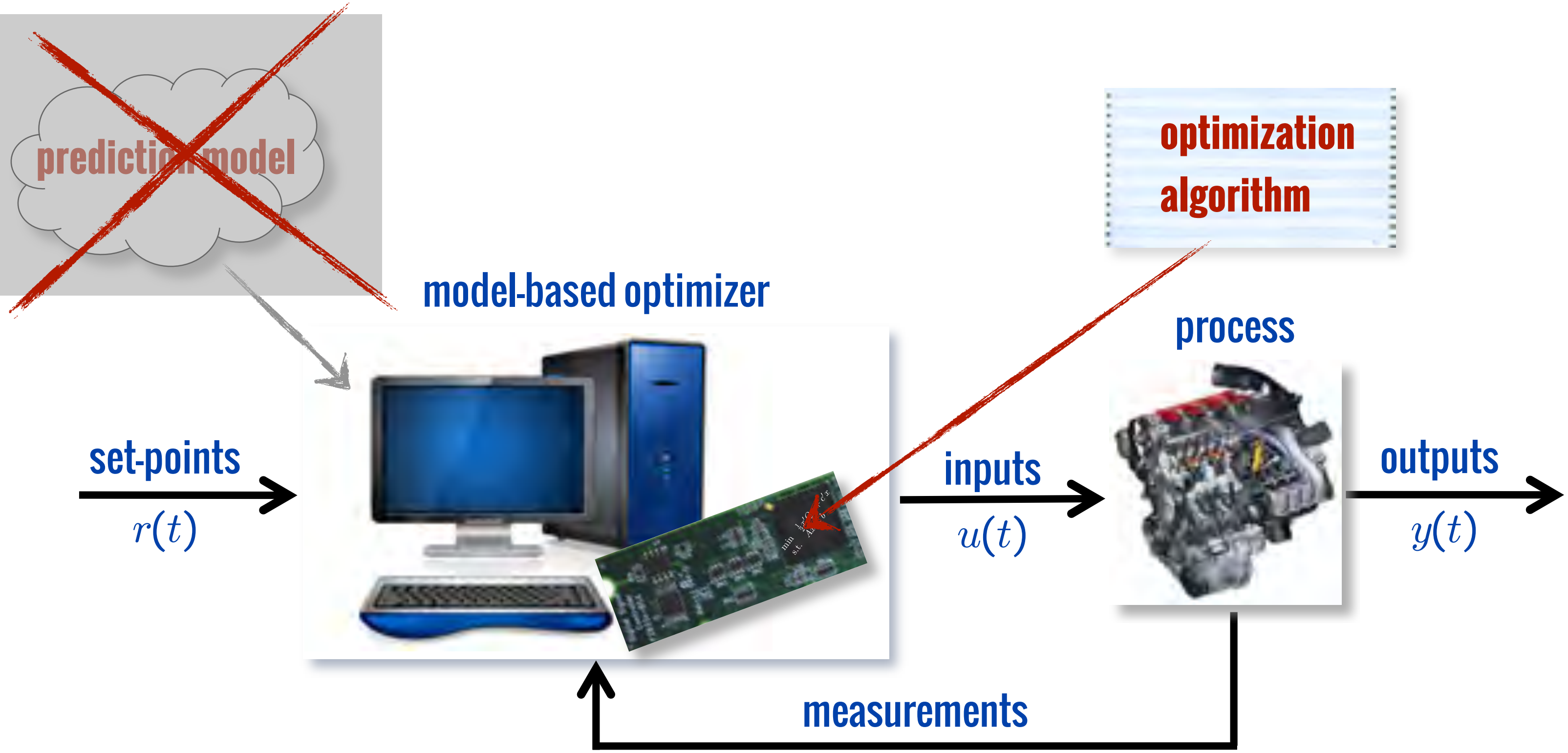
| | inverted pendulum | DC motor | nonlinear demo | AFTI F16 |
|---|---|---|---|---|
| Explicit MPC | | | | |
| max flops | 3382 | 1689 | 9184 | 16434 |
| max memory (kB) | 55 | 30 | 297 | 430 |
| Implicit MPC | | | | |
| max flops | 3809 | 2082 | 7747 | 7807 |
| sqrt | 27 | 9 | 37 | 33 |
| max memory (kB) | 15 | 13 | 20 | 16 |

- QP certification algorithm currently used in production

ODYS Advanced Controls & Optimization

# DATA-DRIVEN MPC - SOME POSSIBILITIES

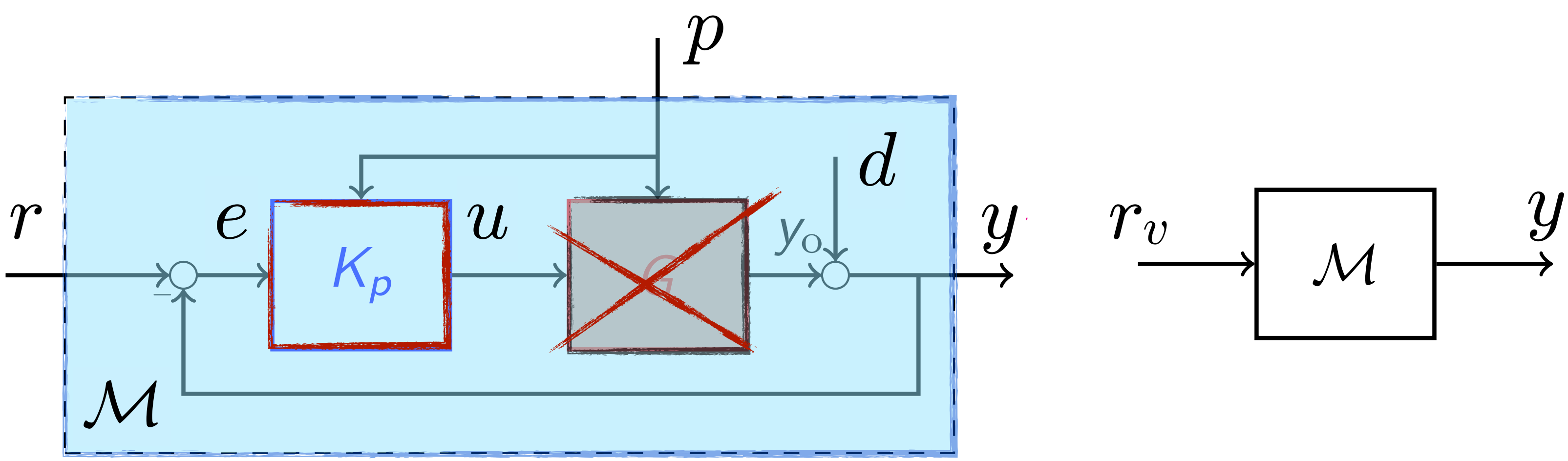
# DATA-DRIVEN MPC




- Can we design an MPC controller **without** first identifying a model of the open-loop process ?

# DATA-DRIVEN DIRECT CONTROLLER SYNTHESIS

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

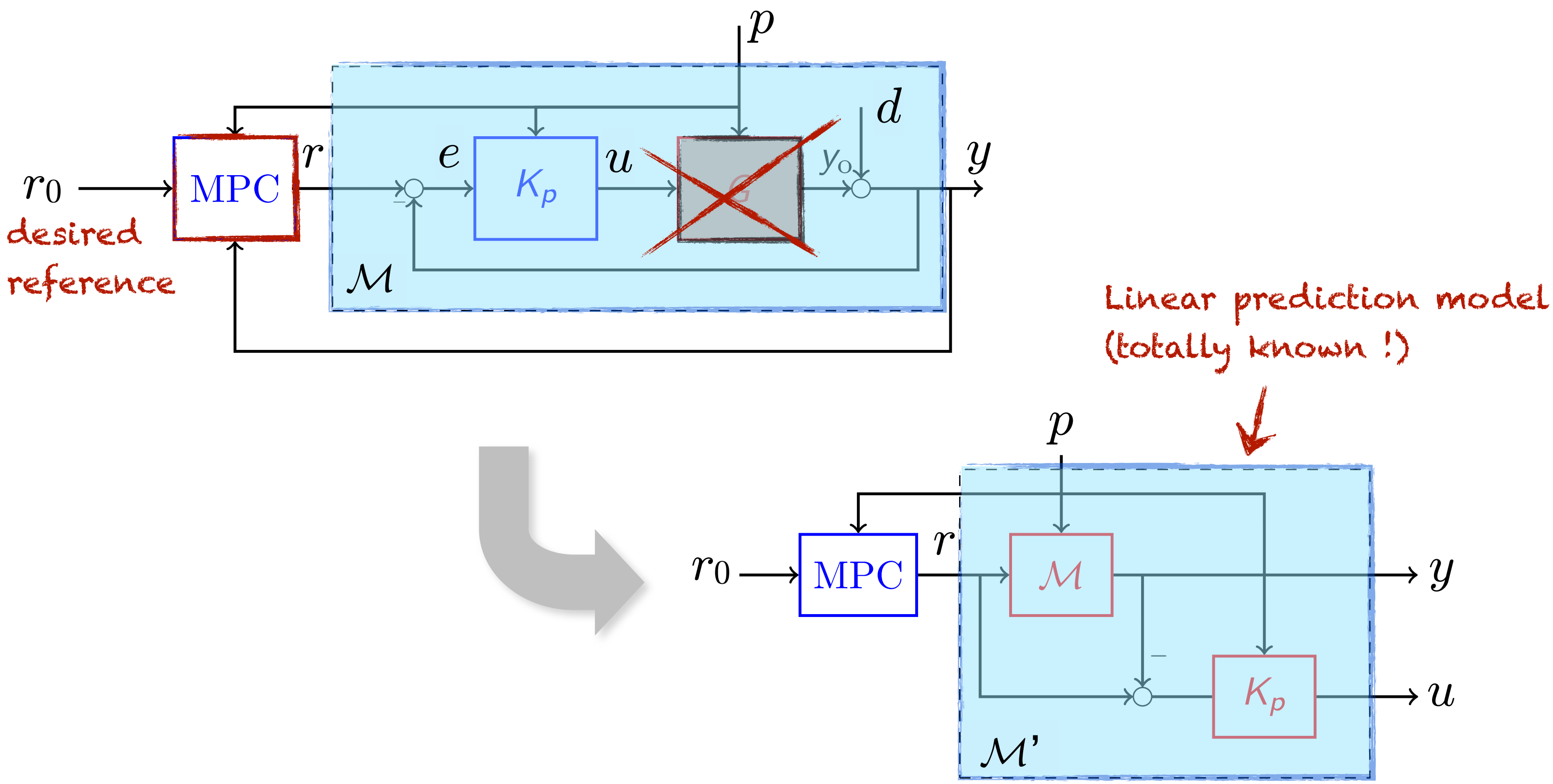



- Collect a set of **data** $\{u(t), y(t), p(t)\}, t = 1, \dots, N$
- Specify a **desired closed-loop linear model** $\mathcal{M}$ from $r$ to $y$
- Compute $r_v(t) = \mathcal{M}^{\#} y(t)$ from **pseudo-inverse model** $\mathcal{M}^{\#}$ of $\mathcal{M}$
- **Identify** linear (LPV) model $K_p$ from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

# DATA-DRIVEN MPC

- Design a linear MPC (reference governor) to generate the reference $r$

(Bemporad, Mosca, 1994) (Gilbert, Kolmanovsky, Tan, 1994)

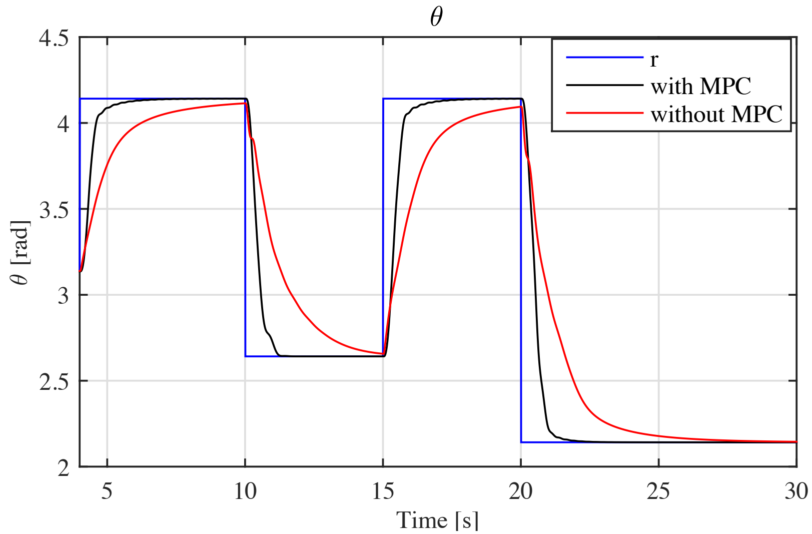


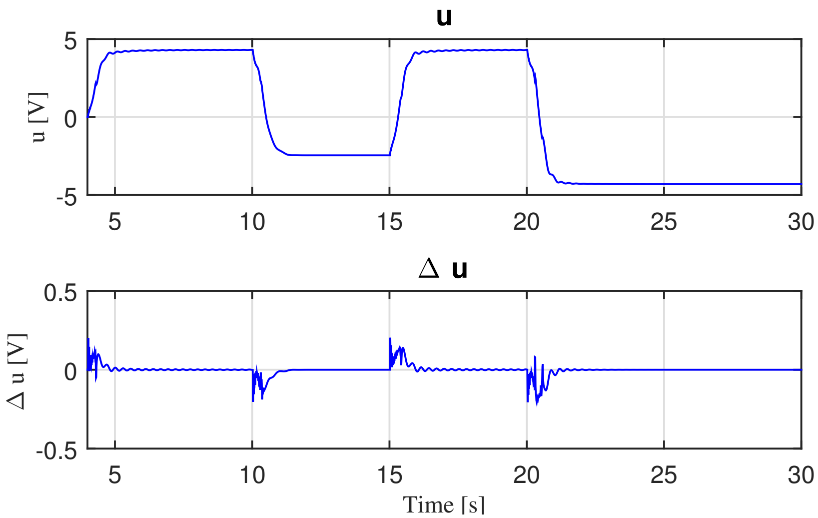

- MPC designed to handle input/output constraints and improve performance

(Piga, Formentin, Bemporad, 2017)

# DATA-DRIVEN MPC - AN EXAMPLE

- Experimental results: MPC handles soft constraints on $u$, $\Delta u$ and $y$

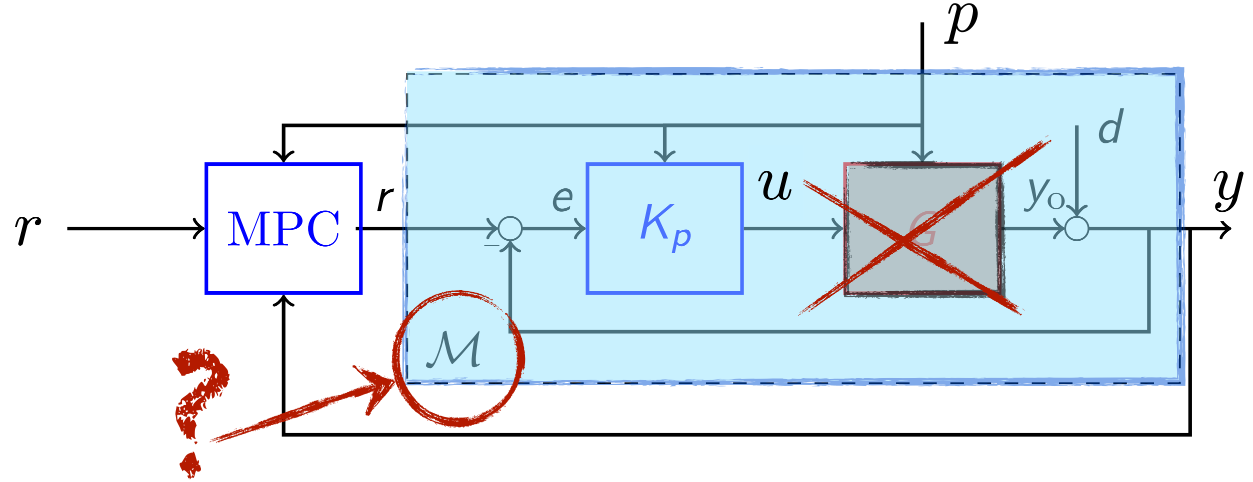

desired tracking performance achieved

constraints on input increments satisfied

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

# OPTIMAL DATA-DRIVEN MPC

- **Question**: How to choose the reference model $\mathcal{M}$ ?




- Can we choose $\mathcal{M}$ from data so that $K_p$ is an **optimal controller** ?

# OPTIMAL DATA-DRIVEN MPC

(Selvi, Piga, Bemporad, 2018)

- **Idea**: parameterize desired closed-loop model $\mathcal{M}(\theta)$ and optimize

$$\min_{\theta} J(\theta) = \frac{1}{N}\sum_{t=0}^{N-1} \underbrace{W_y(r(t)-y_p(\theta,t))^2 + W_{\Delta u}\Delta u_p^2(\theta,t)}_{\text{performance index}} + \underbrace{W_{\rm fit}(u(t)-u_v(\theta,t))^2}_{\text{identification error}}$$

- **Solution**: solve a (non-convex) nonlinear program to get optimal $\theta$

- Each function evaluation requires synthesizing $K_p(\theta)$ from data and simulating the nominal model and control law

$$y_p(\theta,t) = \mathcal{M}(\theta)r(t) \qquad u_p(\theta,t) = K_p(\theta)(r(t)-y_p(\theta,t))$$

$$\Delta u_p(\theta,t) = u_p(\theta,t) - u_p(\theta,t-1)$$

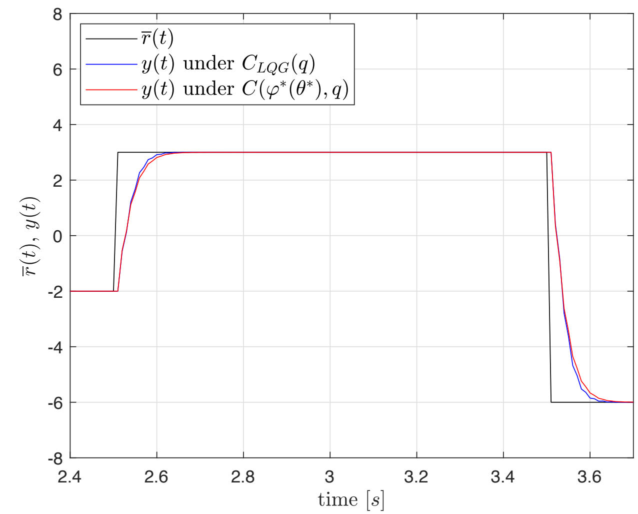
# OPTIMAL DATA-DRIVEN MPC

(Selvi, Piga, Bemporad, 2018)

- **Results**: linear process

$$G(z) = \frac{z - 0.4}{z^2 + 0.15z - 0.325}$$

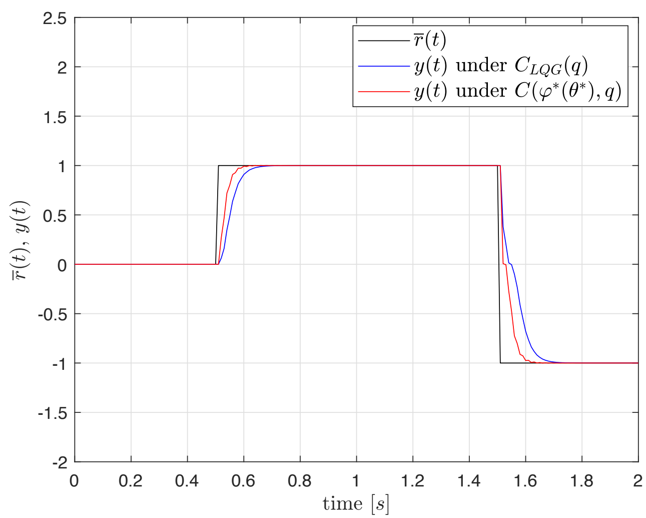
The data-driven controller is **only 1.3% worse** than model-based LQR



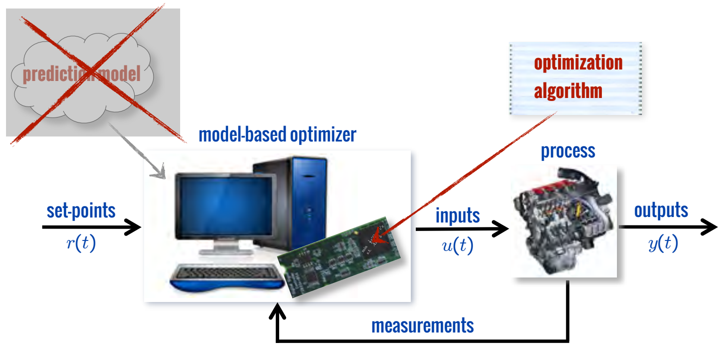

- **Results**: nonlinear (Wiener) process

$$\begin{aligned} y_L(t) &= G(z)u(t) \\ y(t) &= |y_L(t)| \arctan(y_L(t)) \end{aligned}$$

The data-driven controller is **24% better** than LQR based on identified open-loop model !



# LEARNING MPC FROM DATA: DREAMS AND POSSIBILITIES




- **Dream**: learn an MPC controller **without** a prediction model, that optimizes a given index

- **Possibilities**:
  - **Q-learning**: optimize parameters of Q-function defining the MPC law from data. Parameters can also include model coeffs, but not necessarily (Gros, Zanon, 2018)
  - **Policy gradient methods**: learn MPC policy coefficients directly from data using stochastic gradient descent (Ferrarotti, Bemporad, work in progress)
  - **Direct weight optimization**: only collect a database of input/state trajectories, then optimally interpolate them online
    (Salvador, Muñoz de la Peña, Alamo, Bemporad, 2018) $\Rightarrow$ **TuAPo1.1**
  - More in this conference ...

> **Note**: when open-loop model is used as a tuning parameter, very often **learned model $\neq$ best open-loop model** (unless model perfectly fits data)

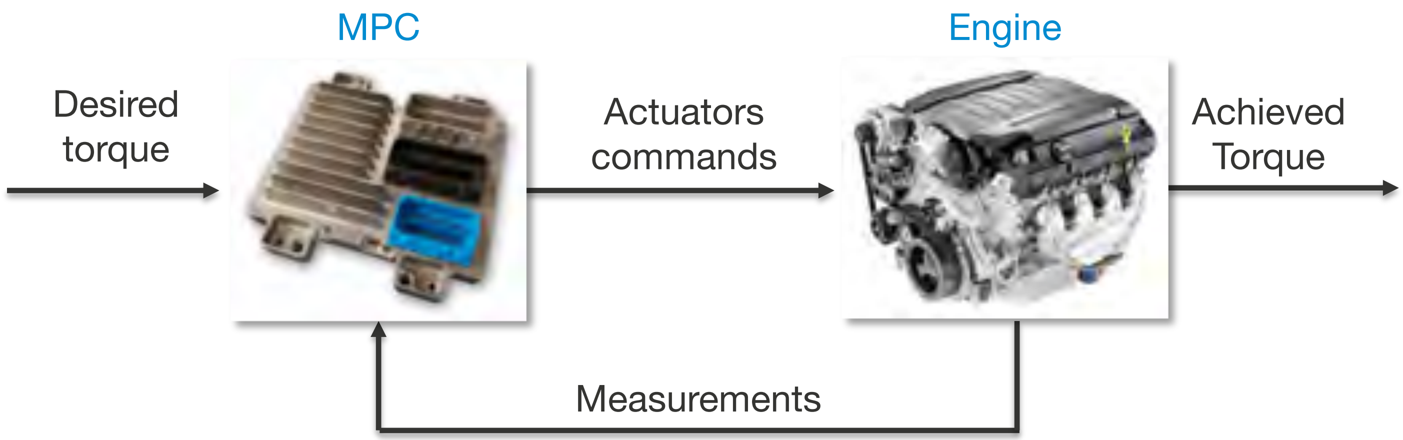
# MPC FOR AUTOMOTIVE APPLICATIONS

# MPC FOR AUTONOMOUS DRIVING

- Coordinate **torque request** and **steering** to achieve **safe and comfortable** autonomous driving with no collisions
- MPC combines **path planning**, **path tracking**, and **obstacle avoidance**
- **Stochastic prediction models** are used to account for uncertainty and driver's behavior

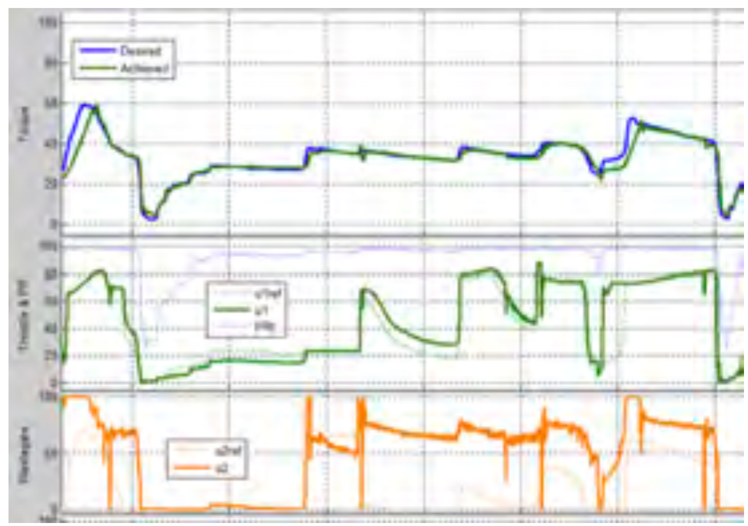
# MPC OF GASOLINE TURBOCHARGED ENGINES

- Optimize **engine actuators** (throttle, wastegate, intake/exhaust cams) to make **engine torque** track set-points, maximizing efficiency and satisfying **constraints**




**QP solved in real-time on ECU**

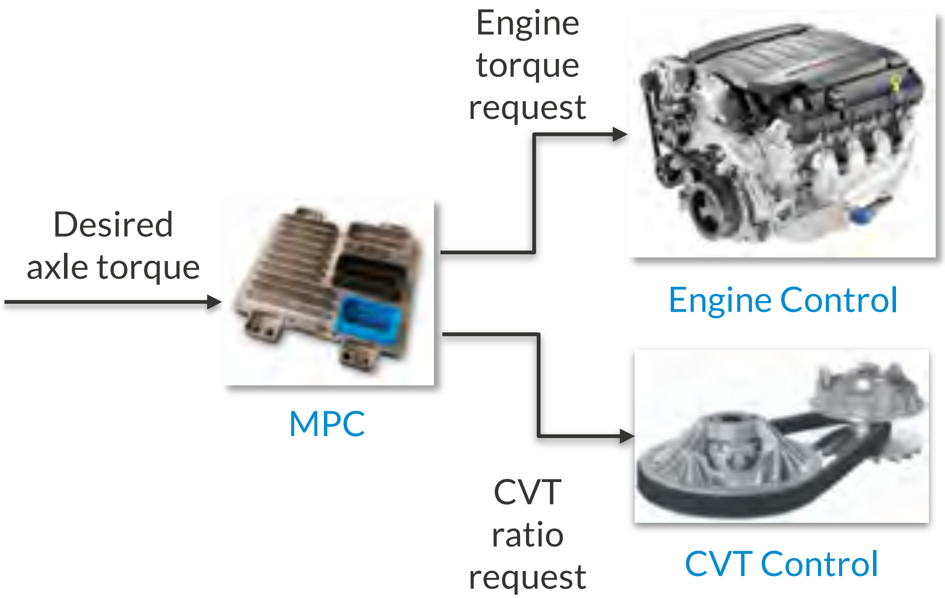
(Bemporad, Bernardini, Long, Verdejo, 2018)

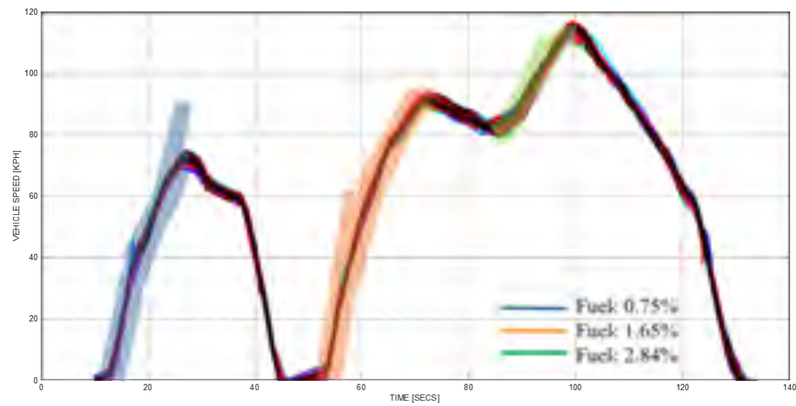


engine operating at low pressure (66 kPa)

# SUPERVISORY MPC OF POWERTRAIN WITH CVT

- Coordinate **engine torque request** and continuously variable transmission (CVT) **ratio** to improve fuel economy and drivability
- Real-time MPC is able to take into account **coupled dynamics** and **constraints**, optimizing performance also during transients




US06 Double Hill driving cycle

(Bemporad, Bernardini, Livshiz, Pattipati, 2018)

# CONCLUSIONS

- Long history of success of MPC in the **process industries**, now spreading to other industries (especially **automotive**)

- A lot of excellent technical **possibilities** for MPC are available today **(see this conference!)**
- **Key enablers** for MPC to become a **reality** in embedded production systems:

  1. Fast, robust, and simple to code **QP solvers**, with proved execution time
  2. Good **system identification / machine learning** methods to deal with data
  3. **Production managers** that are willing to adopt new advanced control technologies

**Is MPC a mature technology for industrial production?**

# EMBEDDED MPC DESIGN: REALITY

**General Motors** and **ODYS** have developed a multivariable constrained MPC system for torque tracking in turbocharged gasoline engines. The control system is **scheduled for production by GM in fall 2018**.

- Multivariable system, **4 inputs, 4 outputs**. QP solved **in real time on ECU**
  (Bemporad, Bernardini, Long, Verdejo, 2018)

- Supervisory **MPC for powertrain control** also in production in 2018
  (Bemporad, Bernardini, Livshiz, Pattipati, 2018)

**First known mass production of MPC in the automotive industry**

http://www.odys.it/odys-and-gm-bring-online-mpc-to-production

ODYS
Advanced Controls & Optimization