

# Nonlinear Model Predictive Control and Real-Time Optimization

Moritz Diehl

Systems Control and Optimization Laboratory  
Department of Microsystems Engineering (IMTEK) &  
Department of Mathematics  
University of Freiburg

**universität freiburg**

NTNU Trondheim  
February 17, 2026

# Complex Sensor Actuator Systems

## SENSORS

- GPS
- acceleration
- radar
- vision
- ...



How to connect ?

## ACTUATORS

- flight surfaces
- steering wheel
- motor speeds
- joint torques
- ...



# Complex Sensor Actuator Systems

## SENSORS

- GPS
- acceleration
- radar
- vision
- ...



How to connect ?  
linear controllers ?

## ACTUATORS

- flight surfaces
- steering wheel
- motor speeds
- joint torques
- ...



# Classical Linear Controllers / Linear Filters

Map from one time series into another

$$\dots, y_{i-1}, y_i, y_{i+1}, \dots \longrightarrow \dots, u_{i-1}, u_i, u_{i+1}, \dots$$

# Classical Linear Controllers / Linear Filters

Map from one time series into another

$$\boxed{\dots, y_{i-1}, y_i, y_{i+1}, \dots} \longrightarrow \dots, u_{i-1}, \boxed{u_i}, u_{i+1}, \dots$$

Special case: linear time invariant (LTI) filters

$$u_k = \sum_{i=0}^N a_i y_{k-i}$$

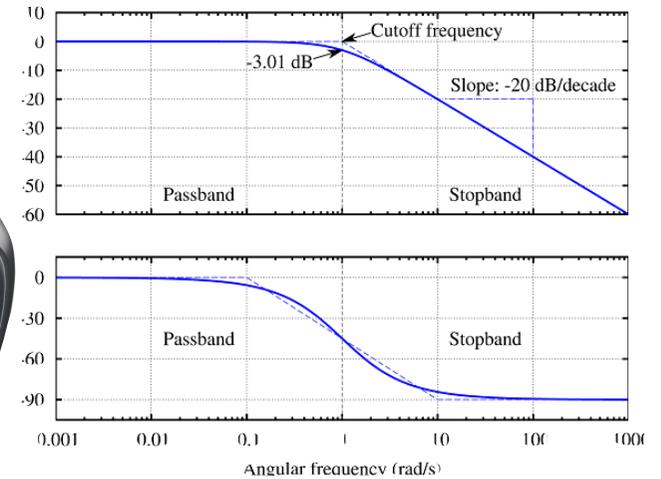
action output = weighted sum of past measurement inputs

# Linear Filters are Everywhere...

## AUDIO SYSTEMS:

- Dolby,
- Echo and other effects
- active noise control

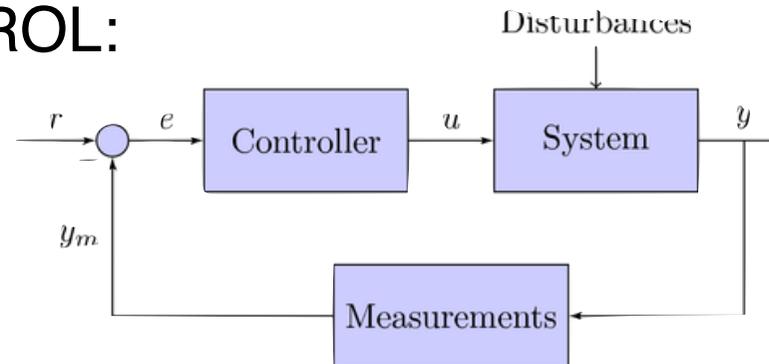
...



## FEEDBACK CONTROL:

- PID controller,
- Kalman filter,
- LQR,

...

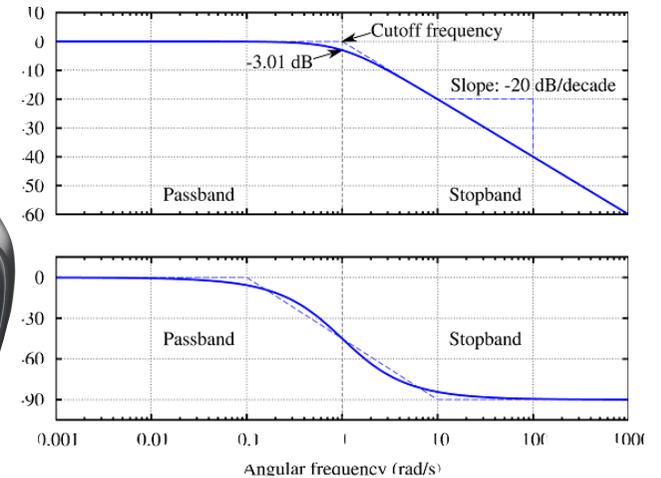


# Linear Filters are Everywhere...

## AUDIO SYSTEMS:

- Dolby,
- Echo and other effects
- active noise control

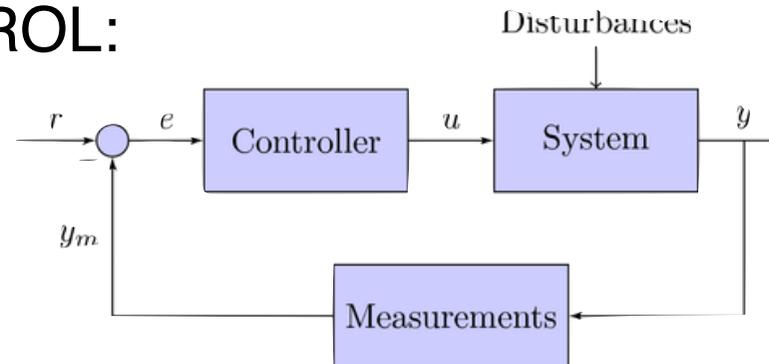
...



## FEEDBACK CONTROL:

- PID controller,
- Kalman filter,
- LQR,

...



...but they need lots of tuning to cope with constraints and nonlinearities.

# Complex Sensor Actuator Systems

## SENSORS

- GPS
- acceleration
- radar
- vision
- ...



How to connect ?  
linear controllers,

## ACTUATORS

- flight surfaces
- steering wheel
- motor speeds
- joint torques
- ...



# Complex Sensor Actuator Systems

## SENSORS

- GPS
- acceleration
- radar
- vision
- ...



How to connect ?  
linear controllers,  
fuzzy logic,  
neural networks, or:

## ACTUATORS

- flight surfaces
- steering wheel
- motor speeds
- joint torques
- ...



# Complex Sensor Actuator Systems

## SENSORS

- GPS
- acceleration
- radar
- vision
- ...



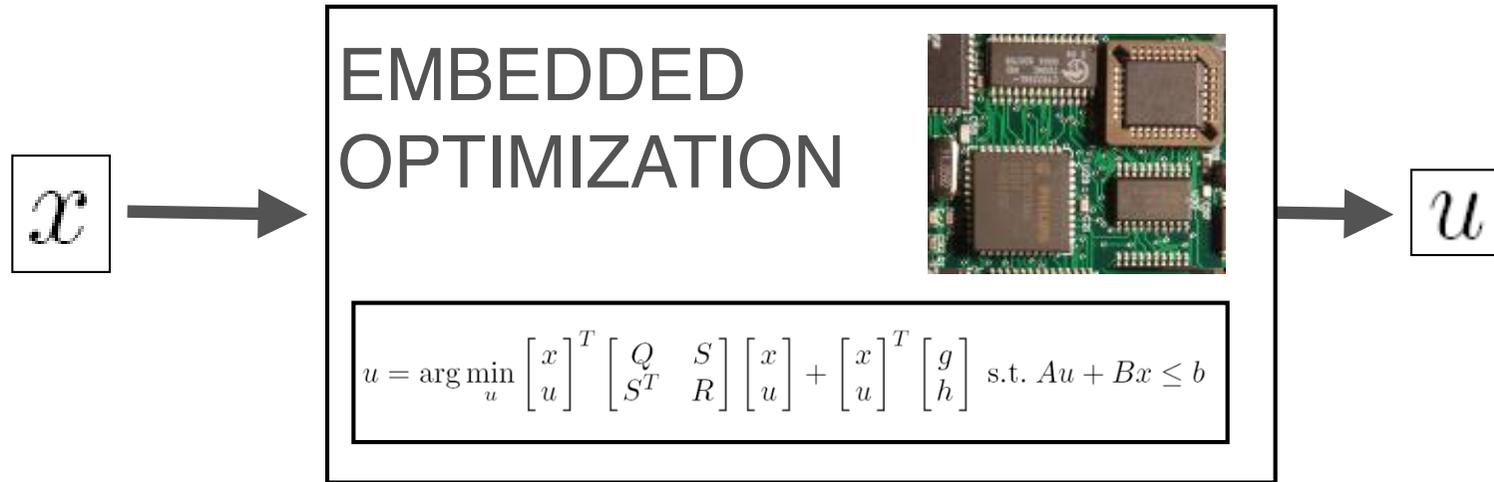
How to connect ?  
linear controllers,  
fuzzy logic,  
neural networks, or:  
**embedded optimisation**

## ACTUATORS

- flight surfaces
- steering wheel
- motor speeds
- joint torques
- ...



# Embedded Optimization: a CPU-Intensive Map



Solve, in real-time and repeatedly, an optimization problem that depends on the incoming stream of input data, to generate a stream of output data.

# The ubiquity of parametric convex optimization

THEOREM [Baes, D., Necoara, 2008]

Every continuous map

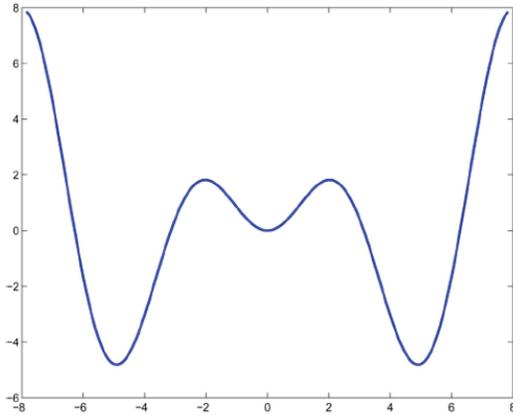
$$\begin{aligned}\mu : \mathbb{R}^{n_x} &\rightarrow \mathbb{R}^{n_u} \\ x &\mapsto u = \mu(x)\end{aligned}$$

can be represented as parametric convex program (PCP):

$$\mu(x) = \arg \min_u g(u, x) \quad \text{s.t.} \quad (u, x) \in \Gamma$$

PCP: objective and feasible set jointly convex in parameters and variables  $(x, u)$ .

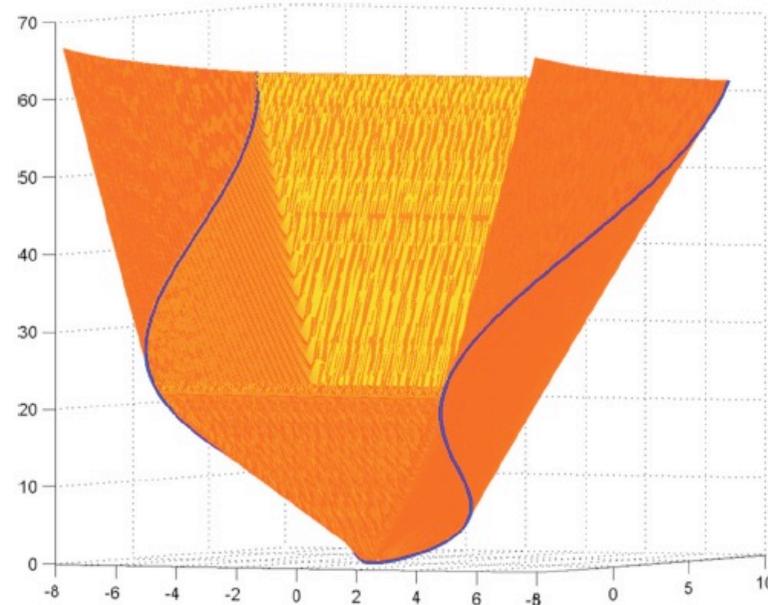
# (Sketch of Proof)



Given: graph of  $\mu(x)$

Construct epigraph  $E$  of  $g(u, x)$

1. “Bend” graph of  $\mu(x)$  using strictly convex  $g^0(x)$
2. Add upward rays.
3. Take convex hull.
4. Show that minima are preserved.

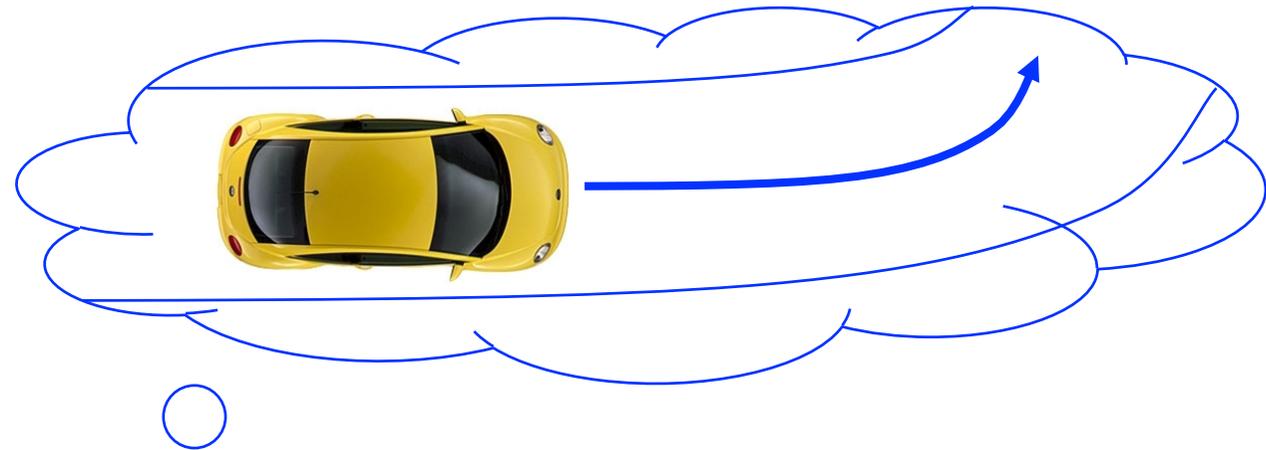


$$S := \{(x, \mu(x), t) \mid x \in \Omega, g^0(x) \leq t\}$$

$$E := \text{conv}(S)$$

# Prime Example: Model Predictive Control (MPC)

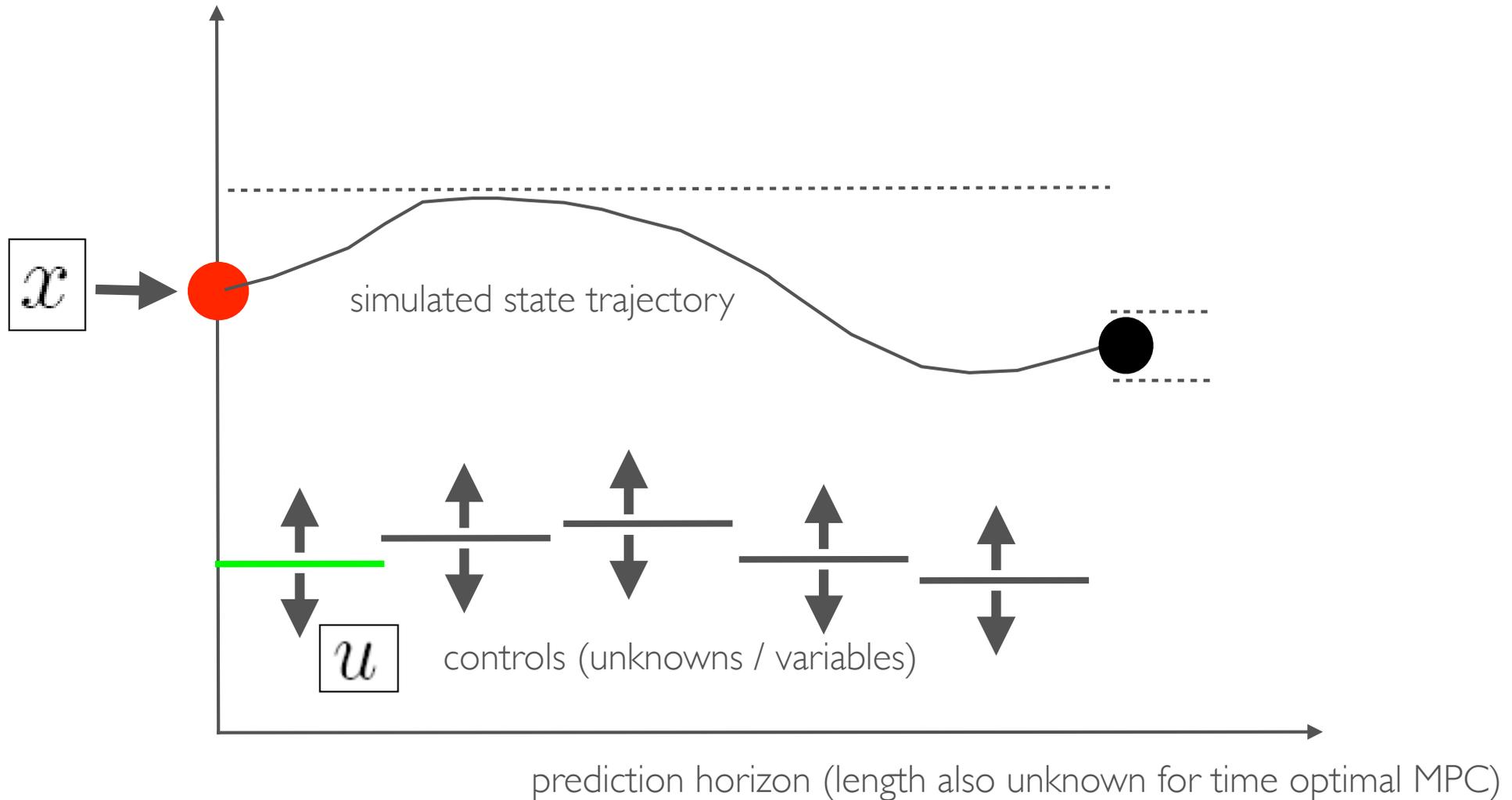
Always look a bit into the future



Example: driver predicts and optimizes, and therefore slows down before a curve

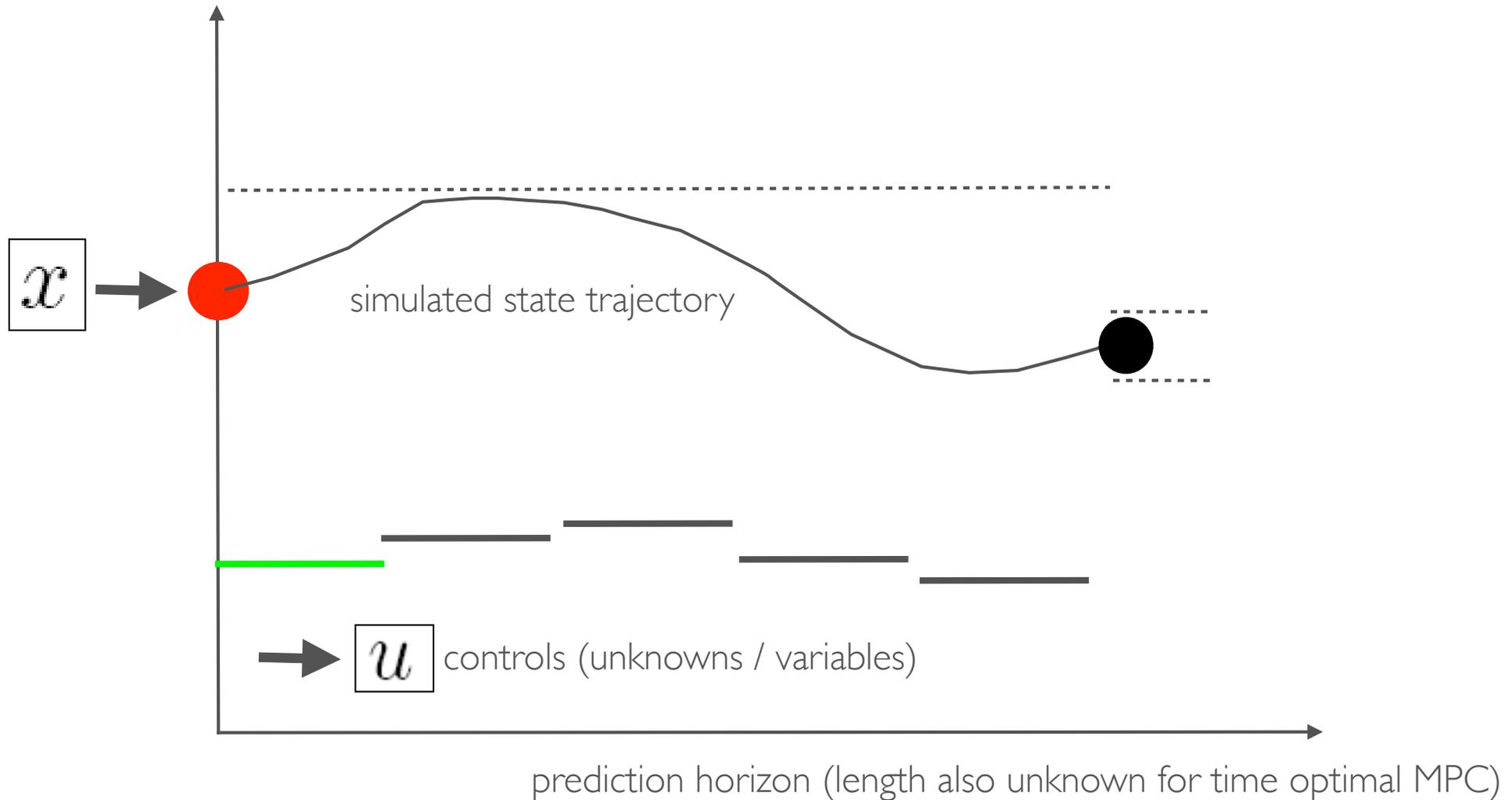
# Open Loop Optimal Control Problem in MPC

For given system state  $\mathbf{x}$ , which controls  $\mathbf{u}$  lead to the best objective value without violation of constraints ?



# Open Loop Optimal Control Problem in MPC

For given system state  $\mathbf{x}$ , which controls  $\mathbf{u}$  lead to the best objective value without violation of constraints ?



MPC creates a map from the initial value  $\mathbf{x}$  to the first control  $\mathbf{u}$   
(which in fact approximates the optimal feedback control from the HJB equation)

# MPC Example: Point-To-Point Motions [PhD Vandembrouck 2012]



Fast oscillating systems (cranes, plotters, wafer steppers, ...)

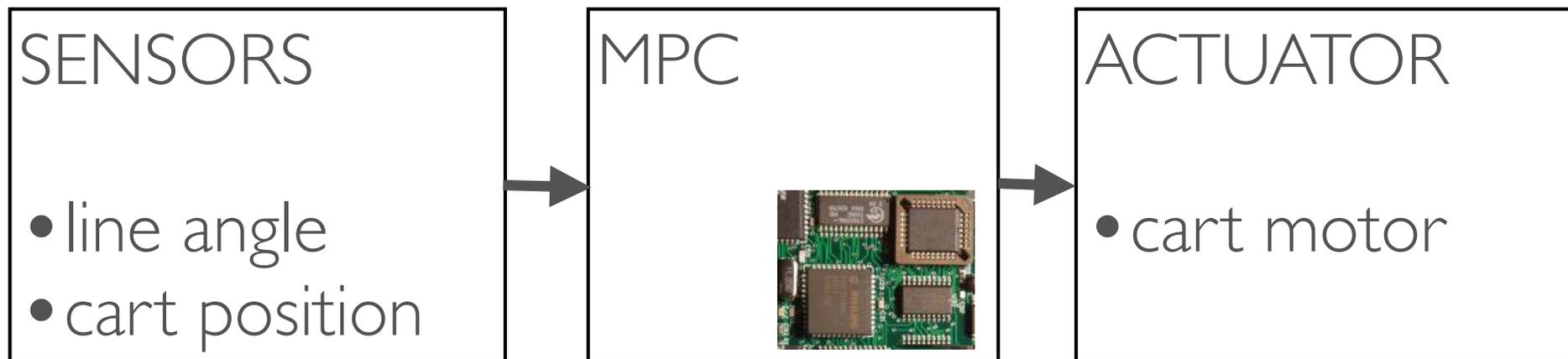
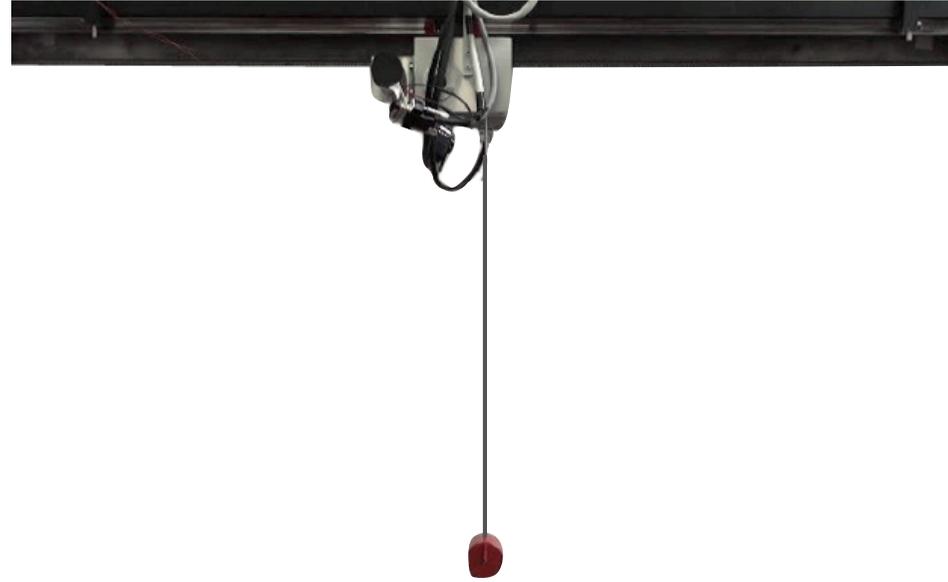
Control aims:

- reach end point as fast as possible
- do not violate constraints
- no residual vibrations

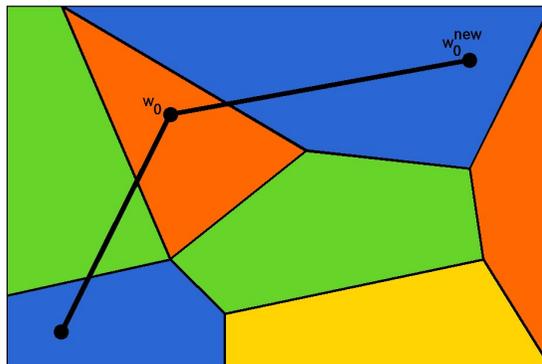
Idea: formulate as embedded optimization problem in form of Model Predictive Control (MPC)



# Time Optimal MPC of a Crane



# qpOASES



- “Quadratic Programming with the Online Active Set Strategy”
- Implements an parametric active set method with dense or sparse linear algebra in C/C++
- Open-source (LGPL): <https://projects.coin-or.org/qpOASES>, developed by Hans Joachim Ferreau with Christian Kirches, Andreas Potschka, ...
- Interfaced to C++, MATLAB, Simulink, CasADi, ...

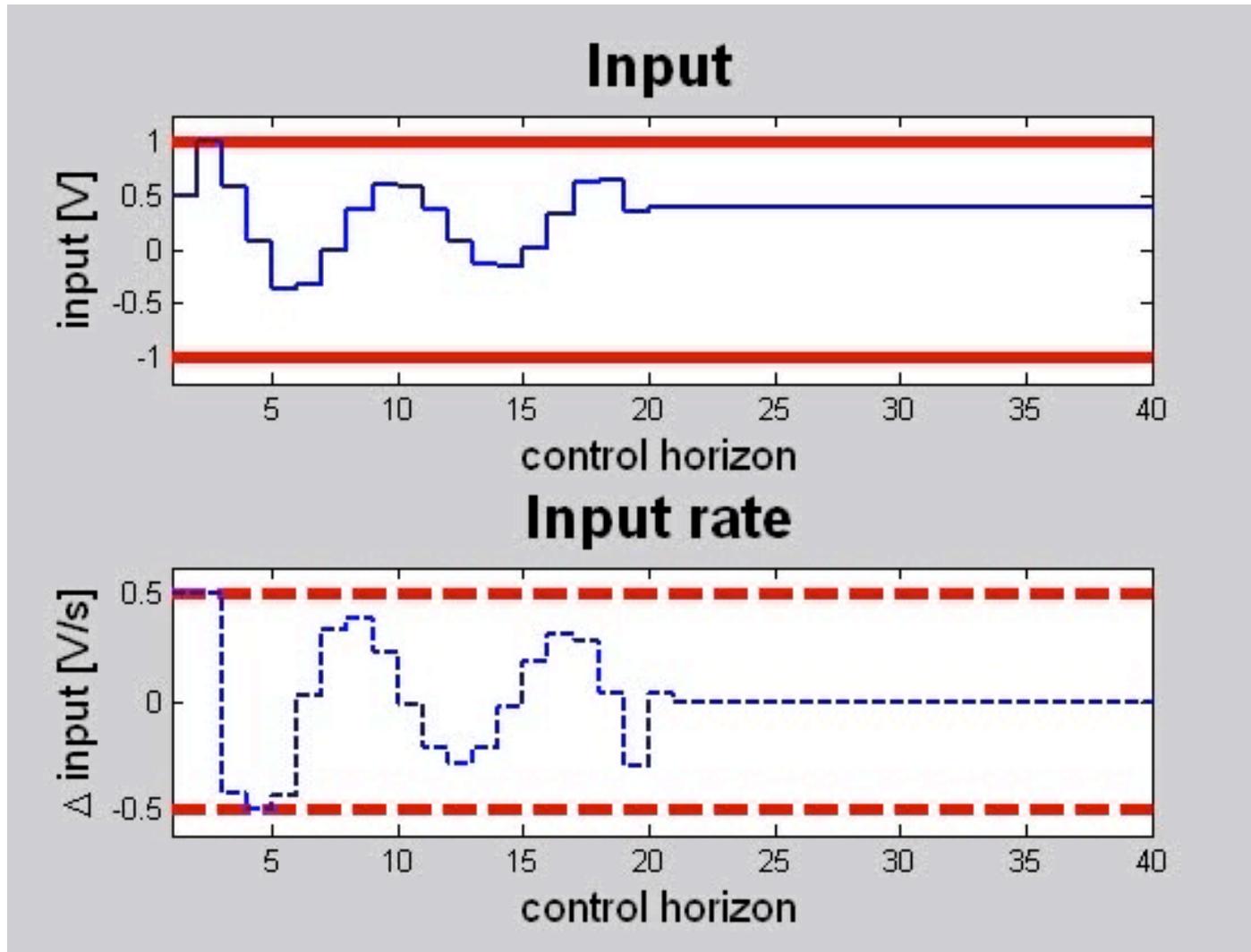


# Time Optimal MPC of a Crane

Univ. Leuven [Vandenbrouck, Swevers, D.]



# Optimal solutions varying in time (inequalities matter)

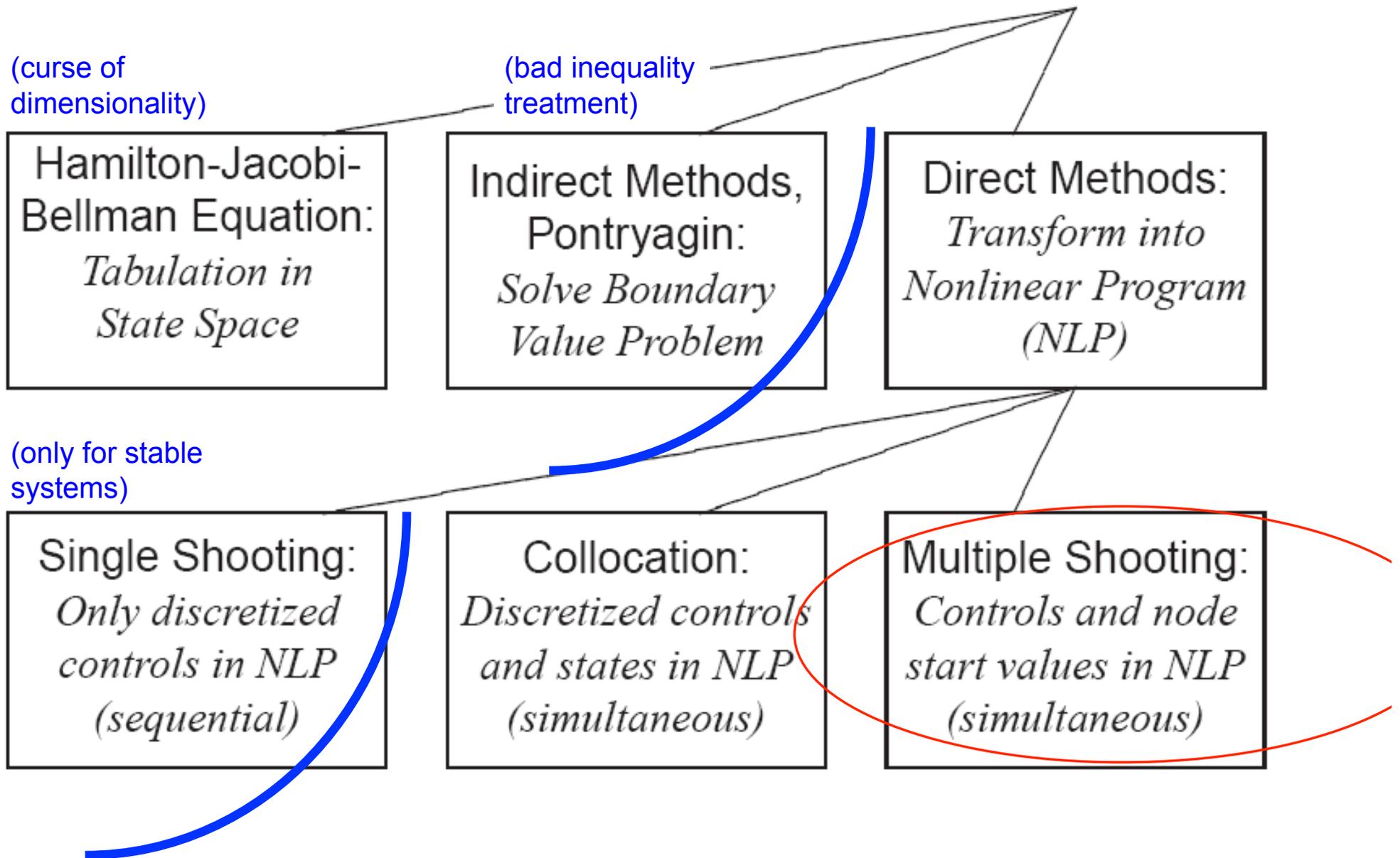


Solver qpOASES [PhD H.J. Ferreau, 2011], [Ferreau, Kirches, Potschka, Bock, D. , A parametric active-set algorithm for quadratic programming, Mathematical Programming Computation, 2014]

# Overview

- Embedded Optimization and Model Predictive Control (MPC)
- **Real-Time Optimal Control Methods**
- Progress in Numerical Integration Methods
- Progress in Structured Quadratic Programming
- Some real-world NMPC applications

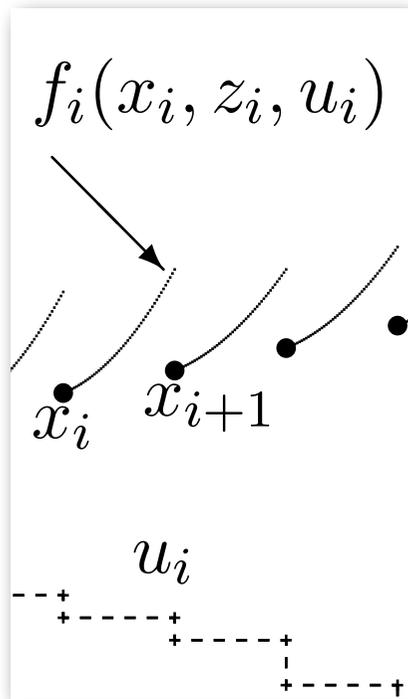
# Optimal Control Solution Methods - Family Tree



# Direct Multiple Shooting [Bock and Plitt, 1981] [Leineweber et al. 1999]



**Hans Georg Bock**



- Discretize controls piecewise on a coarse grid

$$u(t) = u_i \quad \text{for } t \in [t_i, t_{i+1}]$$

- Solve relaxed DAE on each interval  $[t_i, t_{i+1}]$  numerically, starting with artificial initial value  $x_i, z_i$ . Obtain trajectory pieces and state at end of interval,  $f_i(x_i, z_i, u_i)$ .

$$\begin{aligned} & \text{minimize}_{x, z, u} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 &= & 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) &= & 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) &= & 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) &\leq & 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) &\leq & 0. \end{aligned}$$

# Dynamic Optimization Problem in NMPC

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) && + && E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 &= & 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) &= & 0, & i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) &= & 0, & i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) &\leq & 0, & i = 0, \dots, N-1, \\ & && r(x_N) &\leq & 0. \end{aligned}$$

Structured parametric Nonlinear Program

Initial Value  $\bar{x}_0$  usually not known beforehand (“online data” in MPC)

Discrete time dynamics often come from ODE/DAE simulation, in direct multiple shooting method [Bock and Plitt, 1984]

# Dynamic Optimization Problem in NMPC

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) & + & E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

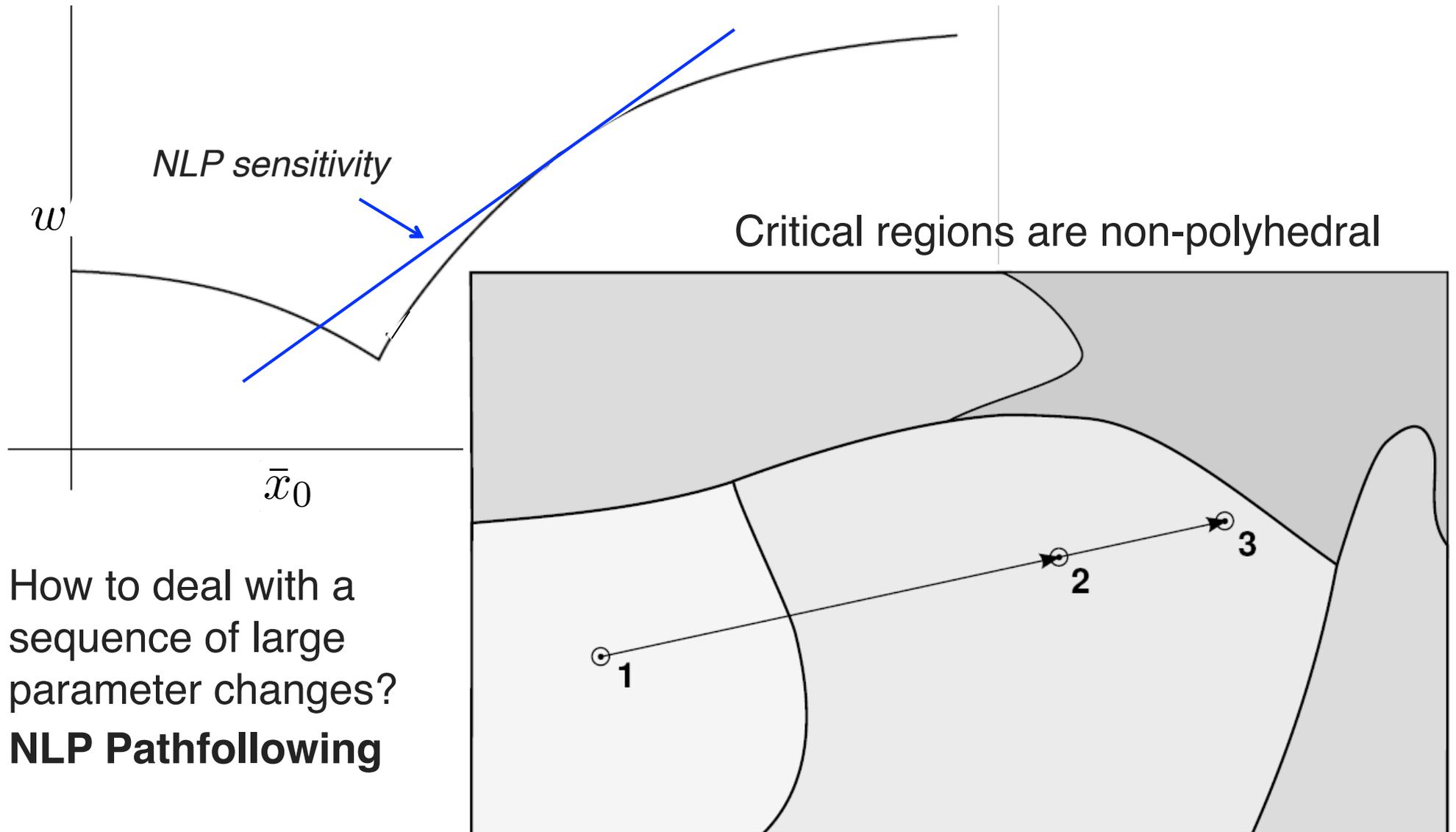
Summarize as

$$\begin{aligned} & \underset{w \in \mathbb{R}^n}{\text{minimize}} && \phi(w) \\ & \text{subject to} && g(w) + M\bar{x}_0 = 0 \\ & && w \in \Omega \end{aligned}$$

with convex  $\phi$  and  $\Omega$

# Nonlinear MPC = parametric Nonlinear Programming

Solution manifold is piecewise differentiable (kinks at active set changes)



# Sequential Convex Programming (SCP) and Real-Time Iteration (RTI)

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$



$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w_k) + g'(w_k)(w - w_k) + M \cdot (\bar{x}_0)_{k+1} = 0 \\ & w \in \Omega \end{array}$$

Repeat each sampling instant  $k$  two steps:

Step 1: Linearize nonlinear constraints at  $w_k$  to obtain convex problem (right).  
(numerical integrations, nonlinear function and derivative evaluations)

Step 2: Obtain new value of parameter  $(\bar{x}_0)_{k+1}$  and solve convex problem  
to obtain new iterate  $w_{k+1}$ . (quadratic program solution)

[D., Bock, Schloeder, Findeisen, Nagy, Allgower, JPC, 2002]

[Zavala, Anitescu, SICON, 2010]

[Tran Dinh, Savorgnan, D., SIOPT, 2013]

# Sequential Convex Programming (SCP) and Real-Time Iteration (RTI)

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$



$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w_k) + g'(w_k)(w - w_k) + M \cdot (\bar{x}_0)_{k+1} = 0 \\ & w \in \Omega \end{array}$$

Repeat each sampling instant  $k$  two steps:

Step 1: Linearize nonlinear constraints at  $w_k$  to obtain convex problem (right).  
(numerical integrations, **nonlinear function and derivative evaluations**)

Step 2: Obtain new value of parameter  $(\bar{x}_0)_{k+1}$  and solve convex problem  
to obtain new iterate  $w_{k+1}$ . (quadratic program solution)

[D., Bock, Schloeder, Findeisen, Nagy, Allgower, JPC, 2002]

[Zavala, Anitescu, SICON, 2010]

[Tran Dinh, Savorgnan, D., SIOPT, 2013]

# Sequential Convex Programming (SCP) and Real-Time Iteration (RTI)

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$



$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w_k) + g'(w_k)(w - w_k) + M \cdot (\bar{x}_0)_{k+1} = 0 \\ & w \in \Omega \end{array}$$

Repeat each sampling instant  $k$  two steps:

Step 1: Linearize nonlinear constraints at  $w_k$  to obtain convex problem (right).  
(numerical integrations, nonlinear function and derivative evaluations)

Step 2: Obtain new value of parameter  $(\bar{x}_0)_{k+1}$  and solve convex problem  
to obtain new iterate  $w_{k+1}$ . (**quadratic program solution**)

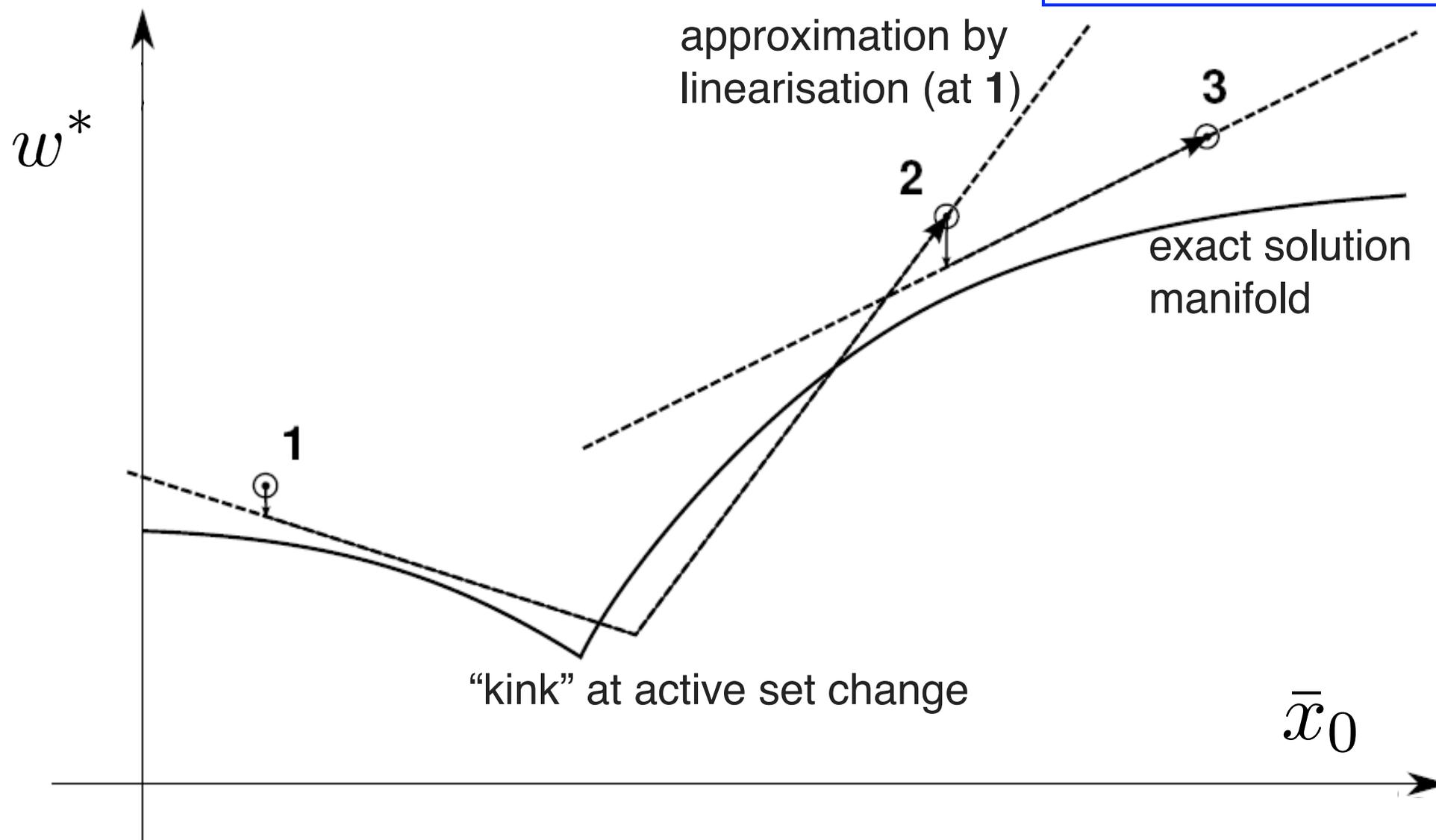
[D., Bock, Schloeder, Findeisen, Nagy, Allgower, JPC, 2002]

[Zavala, Anitescu, SICON, 2010]

[Tran Dinh, Savorgnan, D., SIOPT, 2013]

# Real-Time Iteration

$$\begin{array}{ll} \text{minimize} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$



Tangential prediction even across active set changes

Can divide computations in "preparation" and "feedback phase" [D. 2001]

# Real-Time Iterations [PhD Diehl 2001, Heidelberg]

- 1) Keep states in problem - use direct multiple shooting [1]
- 2) Exploit convexity via Generalized Gauss-Newton [2]
- 3) Use tangential predictors for short feedback delay [3]
- 4) Iterate while problem changes (Real-Time Iterations) [4]
- 5) Auto-generate custom solvers in plain-C [5,6,7] (no malloc)

[1] Bock & Plitt, IFAC WC, 1984

[2] Bock 1983

[3] Bock, D. et al, 1999

[4] D. et al., 2002 / 2005

[5] Mattingley & Boyd, 2009

[6] Houska et al.: Automatica, 2011

[7] Verschueren et al.: Math.Prog. C, 2022

# Computations in one Real-Time Iteration

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

## Sparse QP

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2) Condense sparse QP

## Condensed small QP

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

3) Solve condensed QP

# Computations in one Real-Time Iteration

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

## Sparse QP

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

## Condensed small QP

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

Can prepare without knowing  $\bar{x}_0$   
“Preparation phase”

1) Linearize constraints:  
Integration & sensitivities

2) Condense sparse QP

3) Solve condensed QP

# Computations in one Real-Time Iteration

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

## Sparse QP

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2) Condense sparse QP

## Condensed small QP

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

3) Solve condensed QP

“Feedback phase”

# Real-Time Iteration Contraction

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$

Regard primal dual iterates  $z_k = (w_k, \lambda_k)$

and exact solutions  $z_k^*$  solving the full nonconvex problem for  $(\bar{x}_0)_k$

Iterations are driven by parameter changes  $\Delta\bar{x}_{0,k} := (\bar{x}_0)_{k+1} - (\bar{x}_0)_k$

We can establish a contraction estimate for the primal dual errors:

$$\|z_{k+1} - z_{k+1}^*\| \leq (c_1 + c_2 \|z_k - z_k^*\|) \|z_k - z_k^*\| + (c_3 + c_4 \|\Delta\bar{x}_{0,k}\|) \|\Delta\bar{x}_{0,k}\|$$

Contraction rate depends on bounds on nonlinearity, Jacobian error, and on strong regularity constant.

Contraction rate is independent of active set changes.

[Tran Dinh, Savorgnan, Diehl, SIOPT, 2013]

[Zanelli, Tran Dinh, Diehl, Automatica, 2021]

# \*\*\* Advanced-Step Real-Time Iteration (AS-RTI) \*\*\*

[Nurkanovic et al. 2020]

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) + E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

Sparse QP

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) + E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2) Condense sparse QP

Condensed small QP (feedback)

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

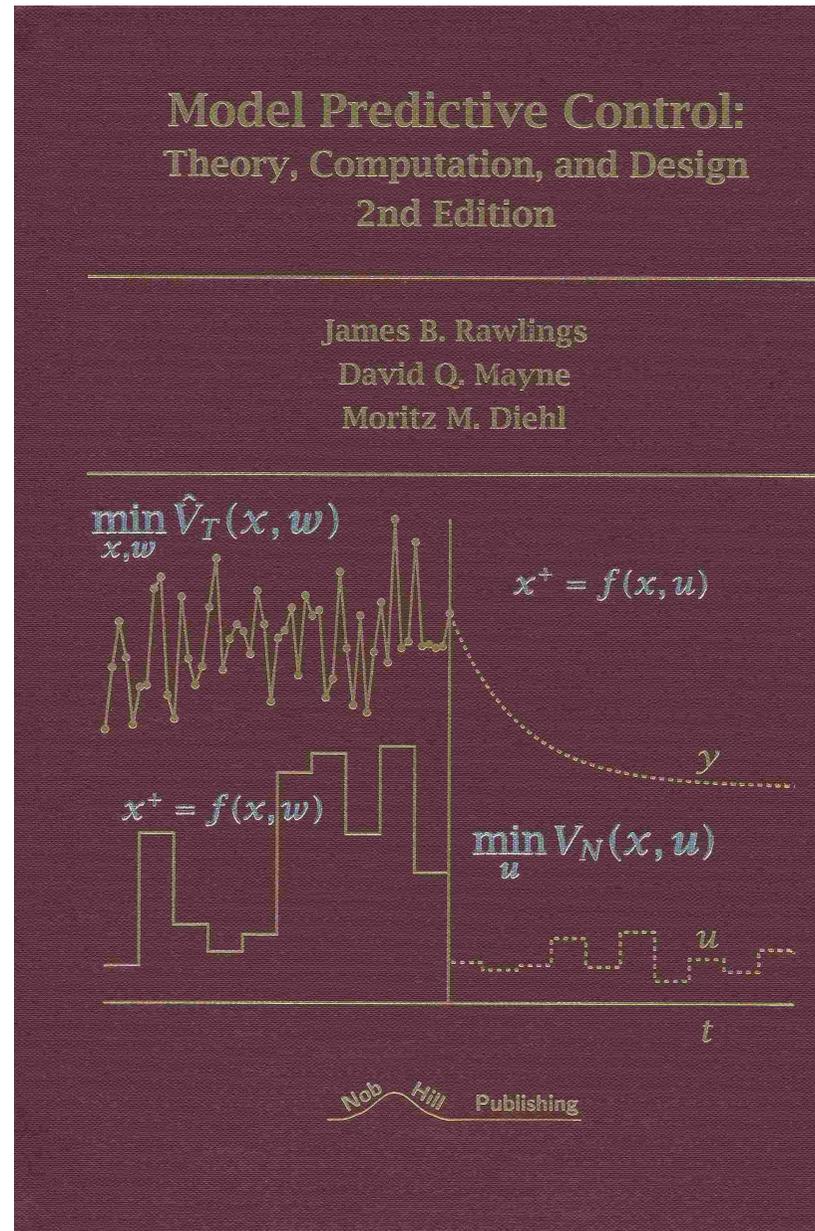
3) Solve condensed QP for  
current state

Condensed small QP (predicted state)

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

4) Solve condensed QP  
again for predicted next  
state, for NLP initialisation

Also see MPC textbook, Ch. 8 on “Numerical Optimal Control”  
(2nd edition, fourth printing, 618 pages, Nob Hill 2022)



# Overview

- Embedded Optimization and Model Predictive Control (MPC)
- Real-Time Optimal Control Methods
- **Progress in Numerical Integration Methods**
- Progress in Structured Quadratic Programming
- Some real-world NMPC applications

# Numerical Integrators for Embedded Optimization

## General features:

- offline precomputation possible (model preprocessing, code generation)
- first and second order derivatives (Jacobian, Hessian) needed
- use implicit integration methods (with root-finding) for stiff and for DAE systems (e.g. Gauss-Legendre or Radau IIA collocation methods)

## Some recent developments:

- **Lifted implicit integrators** perform only one root-finding iteration, combine advantages of direct collocation and direct multiple shooting
- **Inexact Newton with Iterated Sensitivities (INIS)** can work with inexact inverses of the implicit systems [SIAM J. Opt., 28(1), 74-95, 2018]
- **General Nonlinear Static Feedback Structure (GNSF) Structure** can be exploited [Frey et al. , ECC 2019]
- **Casados-Integrators** make acados numerics available in CasADi [Frey et al. , ECC 2023]
- **Gauss-Newton Runge-Kutta (GNRK) Methods** can efficiently compute Gauss-Newton Hessian for continuous least-squares integrals [in prep.]

# Numerical Integrators for Embedded Optimization

## General features:

- offline precomputation possible (model preprocessing, code generation)
- first and second order derivatives (Jacobian, Hessian) needed
- use implicit integration methods (with root-finding) for stiff and for DAE systems (e.g. Gauss-Legendre or Radau IIA collocation methods)

## Some recent developments:

- **Lifted implicit integrators** perform only one root-finding iteration, combine advantages of direct collocation and direct multiple shooting
- **Inexact Newton with Iterated Sensitivities (INIS)** can work with inexact inverses of the implicit systems [SIAM J. Opt., 28(1), 74-95, 2018]
- ➔ **General Nonlinear Static Feedback Structure (GNSF) Structure** can be exploited [Frey et al. , ECC 2019]
- **Casados-Integrators** make acados numerics available in CasADi [Frey et al. , ECC 2023]
- **Gauss-Newton Runge-Kutta (GNRK) Methods** can efficiently compute Gauss-Newton Hessian for continuous least-squares integrals [in prep.]

# Exploiting Linear Substructures in ODE/DAE Models

## Generalized Nonlinear Static Feedback structure (GNSF)

$$E \begin{bmatrix} \dot{x}^{[1]} \\ z^{[1]} \end{bmatrix} = Ax^{[1]} + Bu + C\phi(\underbrace{L_{\dot{x}}\dot{x}^{[1]} + L_x x^{[1]} + L_z z^{[1]}}_{=:y}, \underbrace{L_u u}_{\hat{u}}) + c \quad (3a)$$

$$E^{\text{LO}} \begin{bmatrix} \dot{x}^{[2]} \\ z^{[2]} \end{bmatrix} = A^{\text{LO}} x^{[2]} + f_{\text{LO}}(\dot{x}^{[1]}, x^{[1]}, z^{[1]}, u). \quad (3b)$$

Every ODE/DAE can be brought into this format. Gains are largest if output dimension of nonlinearity function  $\phi$  is smallest

# Implementation inside acados (successor of ACADO)

GNSF-IRK integrator implemented in acados using BLASFEO, including:

- ▶ Precomputation phase
- ▶ Simulation
- ▶ Efficient forward & adjoint sensitivity generation
- ▶ Output of algebraic variables  $z$  at start  $t_0$
- ▶ and corresponding sensitivities

[Master thesis Jonathan Frey]



# Numerical Experiments with Wind Turbine ODE Model

For numerical experiments, a wind turbine model was used:

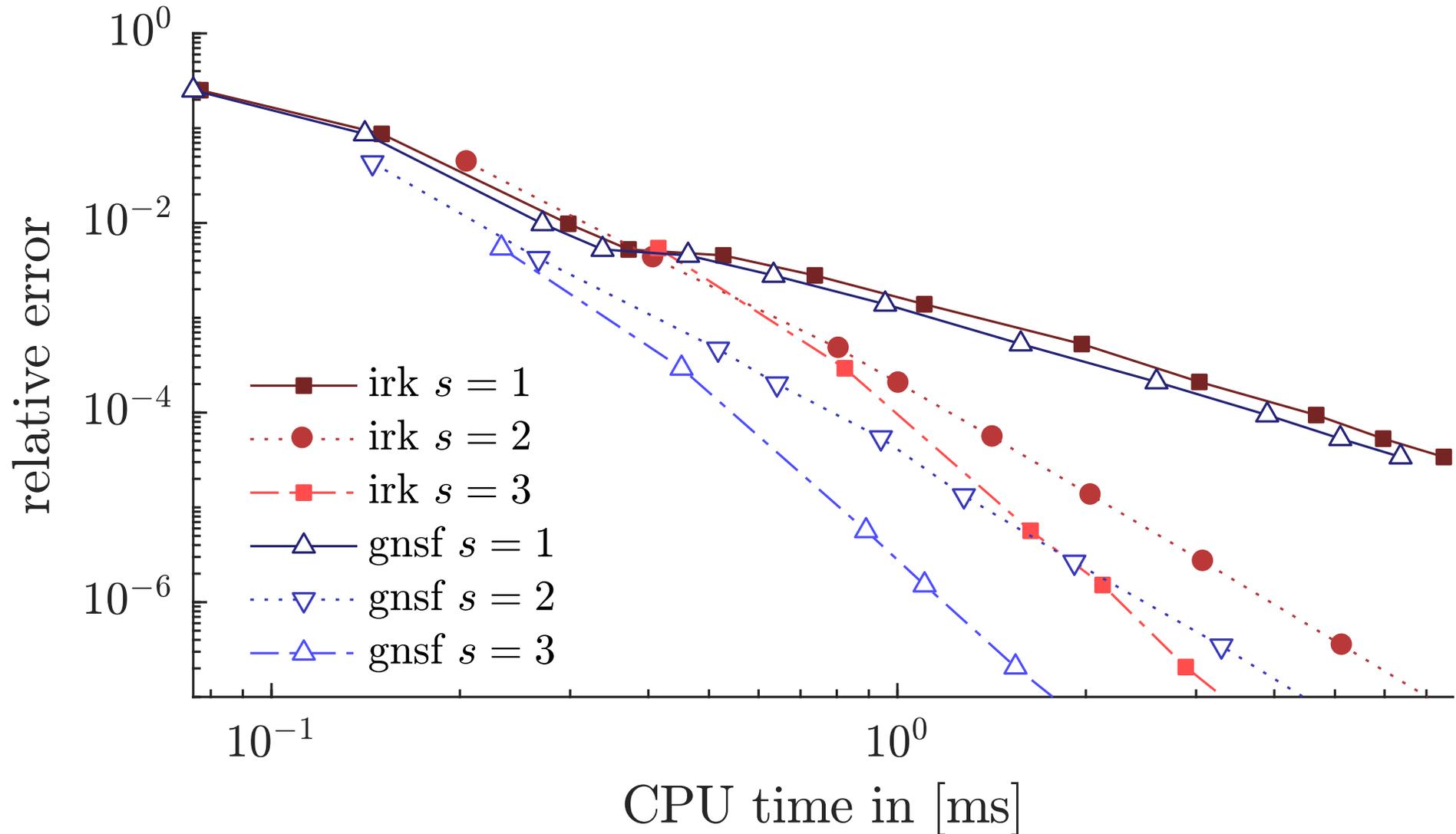
- ▶ highly nonlinear
- ▶ polynomials to model aerodynamic coefficients
- ▶ used in the eco4wind project, provided by Senvion

Model dimensions				GNSF dimensions				
$n_x$	$n_u$	$n_z$	$n_{\text{param}}$	$n_{x_1}$	$n_{z_1}$	$n_{\text{out}}$	$n_y$	$n_{\hat{u}}$
13	2	0	1	11	0	5	8	1



# Numerical Comparisons with Different Stepsizes

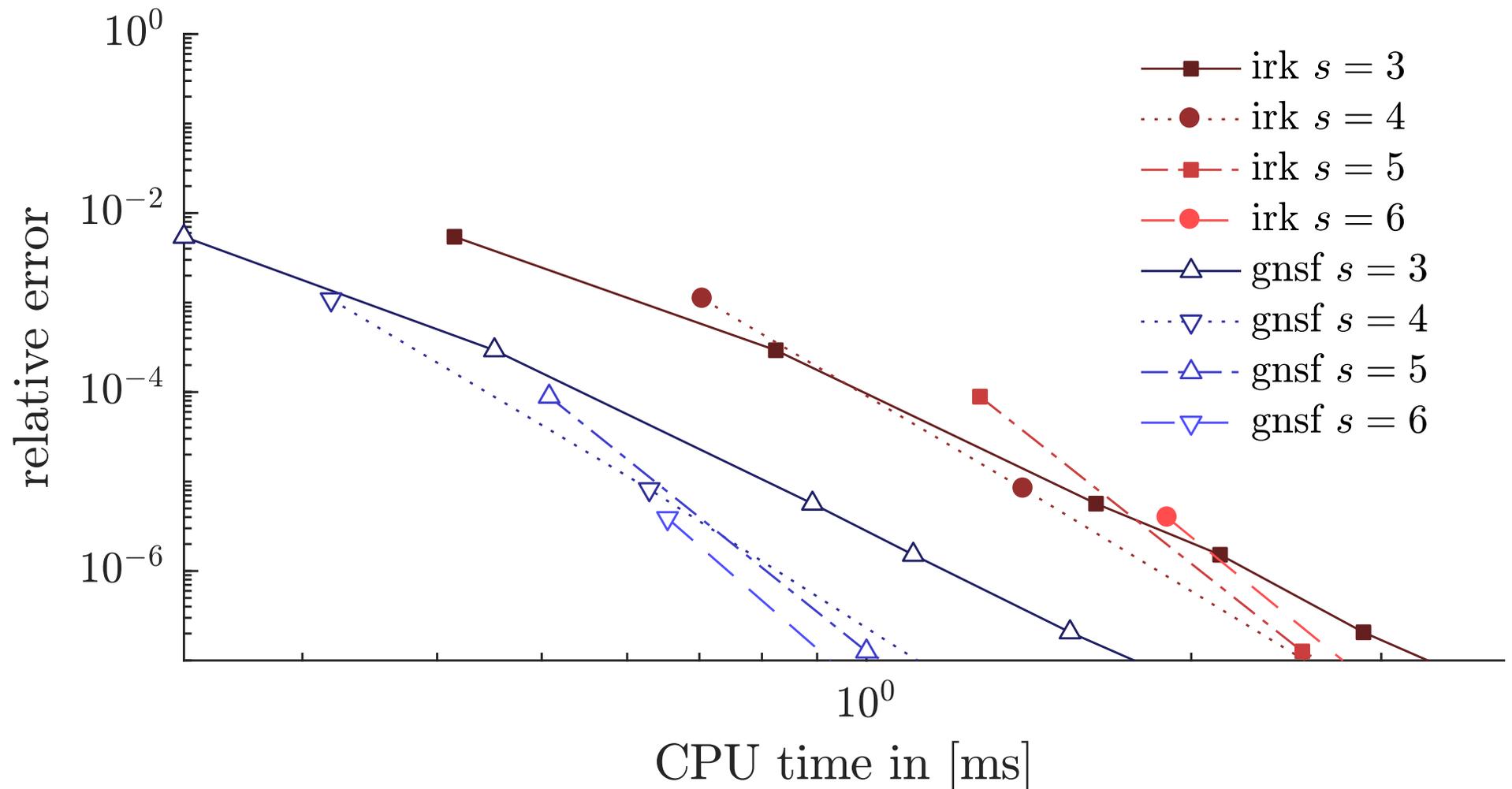
CPU time includes forward derivatives, as needed in optimal control  
Gauss-Legendre Collocation Integrators with  $s=1$  to  $s=3$  stages (2-6th order)



# Numerical Comparisons with Different Stepsizes

CPU time includes forward derivatives, as needed in optimal control

Gauss-Legendre Collocation Integrators with  $s=3$  to  $s=6$  stages (6th-12th order)



**Speedup of GNSF integrators: factor 2-3 (depending on desired accuracy)**

# Overview

- Embedded Optimization and Model Predictive Control (MPC)
- Real-Time Optimal Control Methods
- Progress in Numerical Integration Methods
- **Progress in Structured Quadratic Programming**
- Some real-world NMPC applications

# Computations in one Real-Time Iteration (RTI)

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

## Sparse Quadratic Program (QP)

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

2) Solve sparse QP either  
- via condensing, or  
- via a Riccati type algorithm, or  
- via a combination of both

# Computations in one Real-Time Iteration (RTI)

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

## Sparse Quadratic Program (QP)

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

2) Solve sparse QP either  
- **via condensing**, or  
- via a Riccati type algorithm, or  
- via a combination of both

# Computations in one Real-Time Iteration (RTI)

## NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints:  
Integration & sensitivities

## Sparse Quadratic Program (QP)

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2a) **Condense** sparse QP

## Condensed small QP

$$\begin{aligned} & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\ & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0. \end{aligned}$$

2b) Solve condensed QP  
with small dense QP solver e.g.  
qpOASES

# Case Study: Quadratic Programming improvements 2012-2016 (all algorithms re-activated on same computer on 14.6.2017)



**Andrea Zanelli**



**Dimitris Kouzoupis**

# Case Study: Quadratic Programming improvements 2012-2016 (all algorithms re-activated on same computer on 14.6.2017)



**Andrea Zanelli**

Comparison of different algorithmic QP solution approaches, using **ACADO Code Generation** on Linux Laptop, CPU i5 6200U with 2.7 GHz



**Dimitris Kouzoupis**

Hanging Chain Optimal Control Benchmark

- 15 states, 3 controls, state and control constraints,
- vary MPC control horizon length from  $N=10$  to  $N=100$  intervals
- direct multiple shooting leads to sparse NLP with  $N*(15+3)$  variables,  $N*3$  state constraints,  $N*6$  input bounds (1800 variables, 300 state constraints, 600 input bounds for  $N=100$ )

# Case Study: Quadratic Programming improvements 2012-2016 (all algorithms re-activated on same computer on 14.6.2017)



**Andrea Zanelli**

Comparison of different algorithmic QP solution approaches, using **ACADO Code Generation** on Linux Laptop, CPU i5 6200U with 2.7 GHz



**Dimitris Kouzoupis**

Hanging Chain Optimal Control Benchmark

- 15 states, 3 controls, state and control constraints,
- vary MPC control horizon length from  $N=10$  to  $N=100$  intervals
- direct multiple shooting leads to sparse NLP with  $N*(15+3)$  variables,  $N*3$  state constraints,  $N*6$  input bounds (1800 variables, 300 state constraints, 600 input bounds for  $N=100$ )



**Rien Quirynen**

- always use: Numerical integration with code generated Implicit Runge Kutta (IRK-GL2) method [Quirynen 2012], two integration steps per interval of 100 ms.

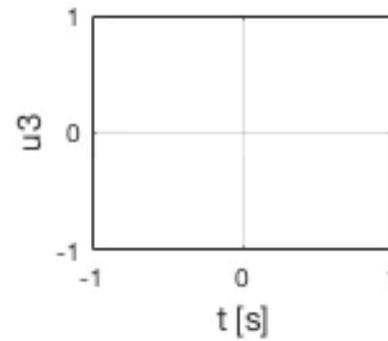
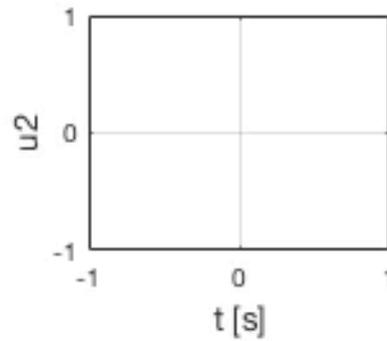
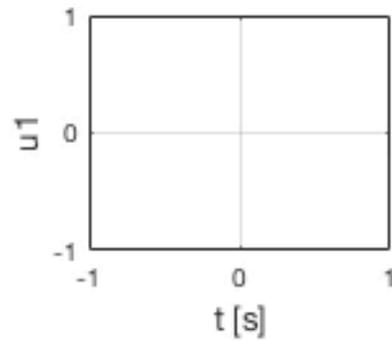
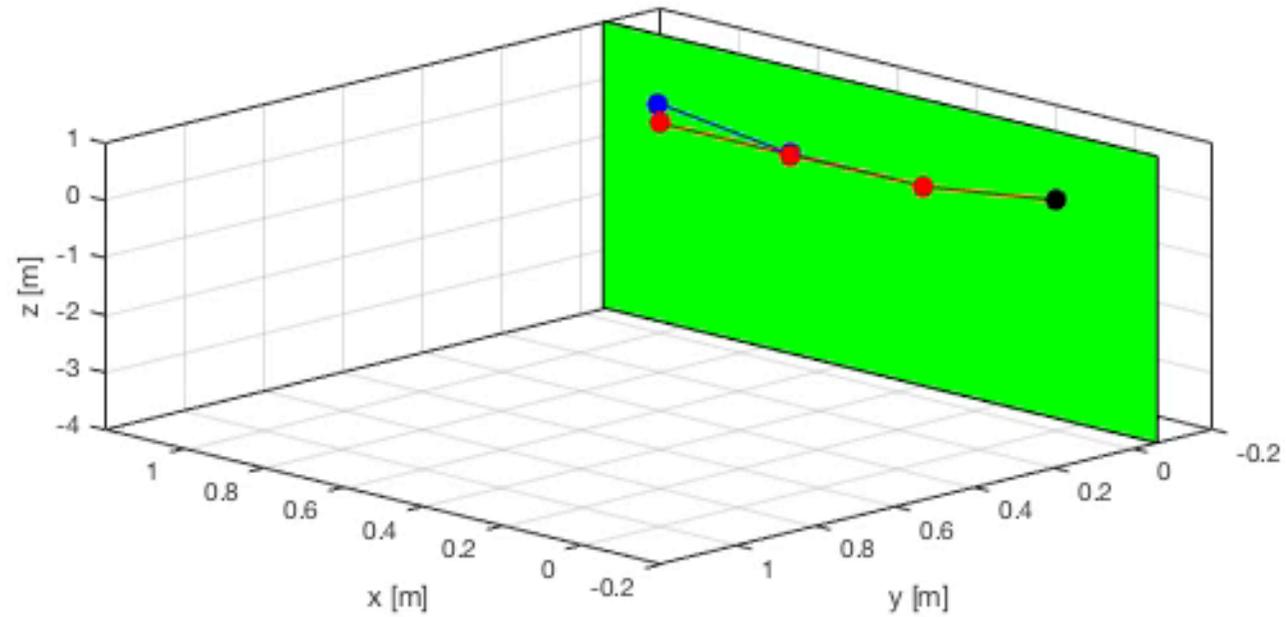
# Hanging Chain Benchmark



**Andrea Zanelli**



**Dimitris Kouzoupis**



# 2012: ACADO Code Generation with **Condensing**

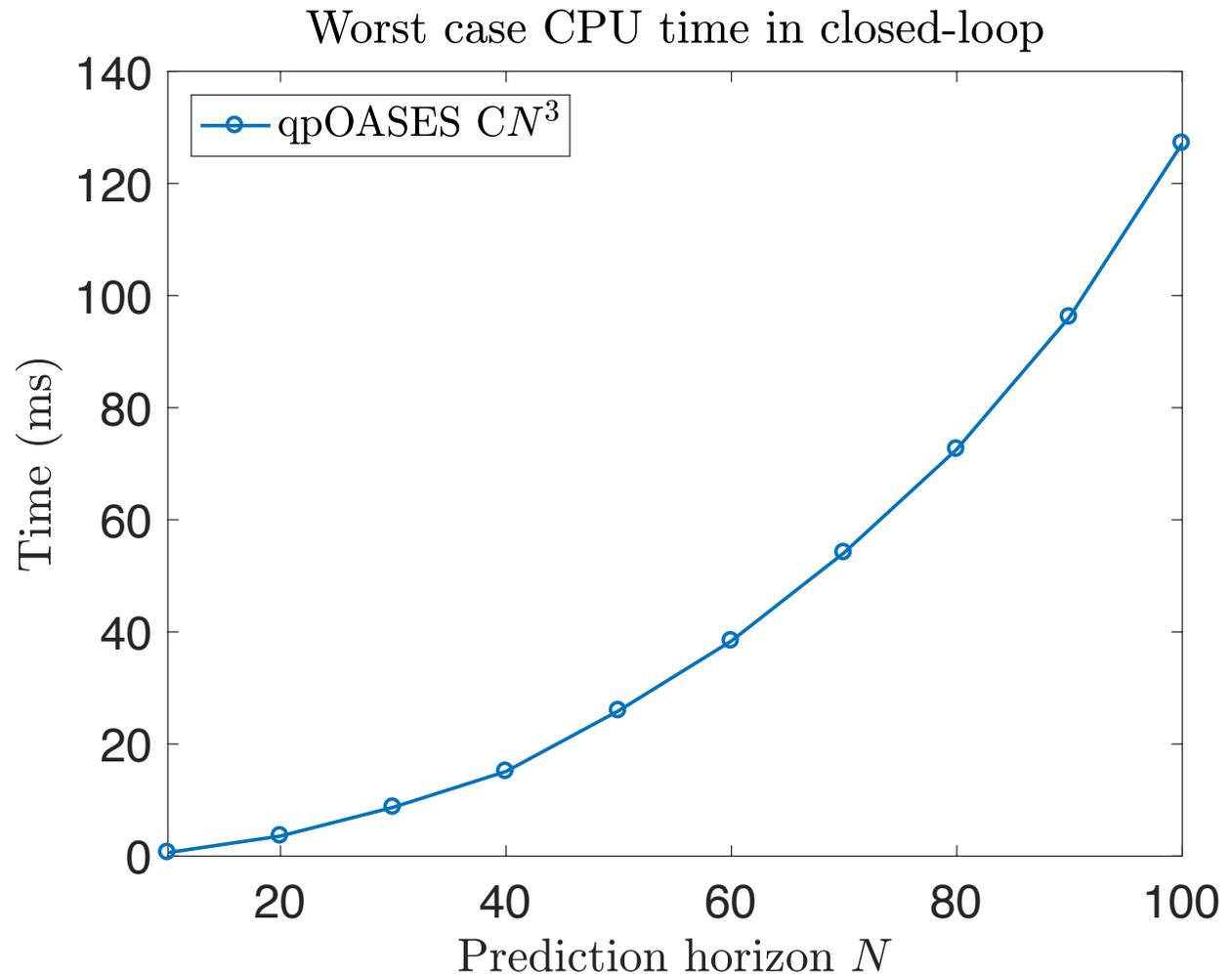
- efficient block sparse condensing with  **$O(N^3)$  complexity**
- qpOASES to solve “condensed” QPs (with  $3*N$  variables)



**Hans Joachim Ferreau**



**Milan Vukov**



# 2013: Code generated sparse QP solver **FORCES**

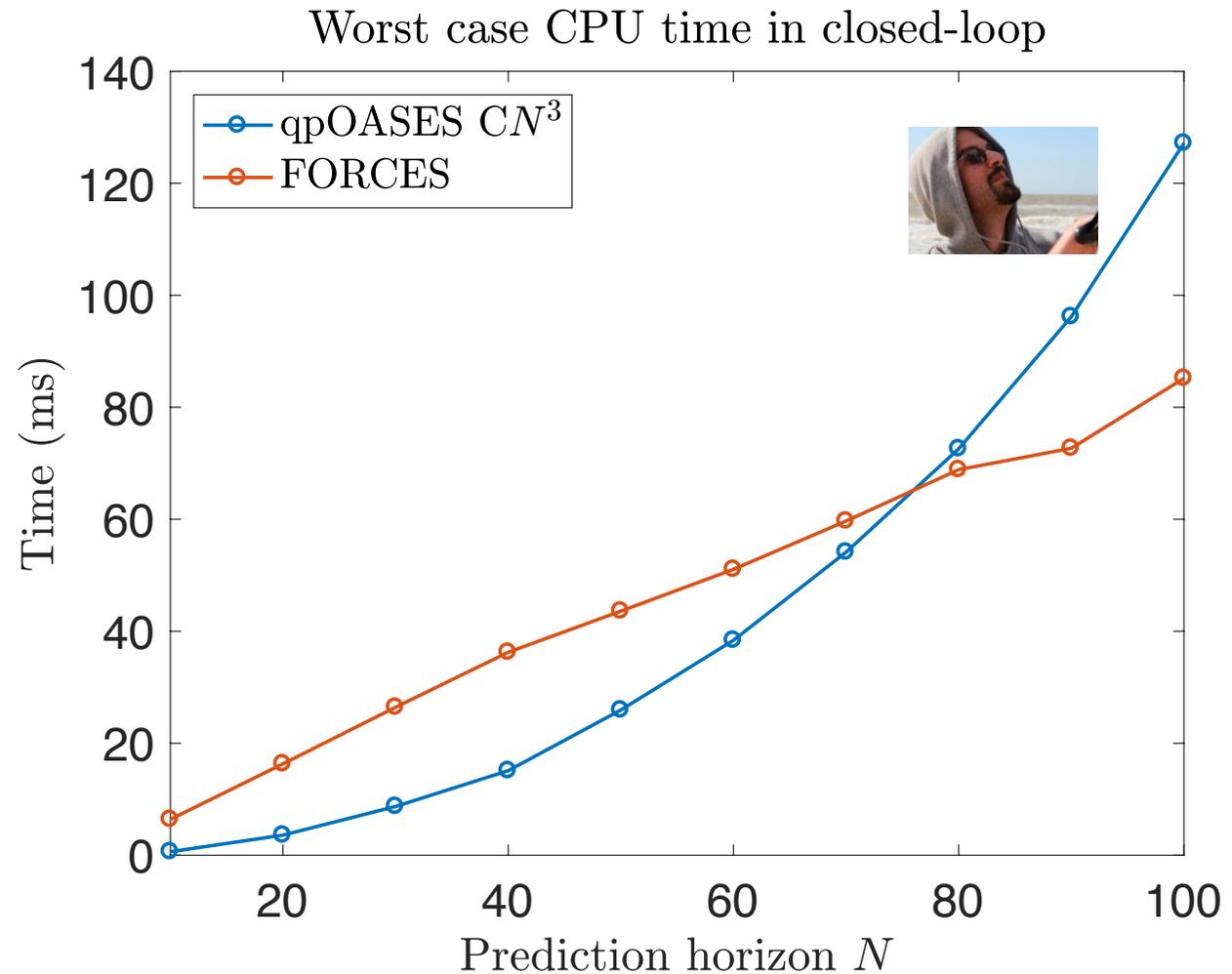
- use interior point method with sparse linear algebra (cf. Steinbach 1995)
- code generate Riccati solvers with  **$O(N)$  complexity**
- include FORCES as QP solver in ACADO [Vukov, Domahidi et al., CDC, 2013]



**Alexander Domahidi**  
[PhD ETH 2013]

# 2013: Code generated sparse QP solver **FORCES**

- use interior point method with sparse linear algebra (cf. Steinbach 1995)
- code generate Riccati solvers with  **$O(N)$  complexity**
- include FORCES as QP solver in ACADO [Vukov, Domahidi et al., CDC, 2013]



Alexander Domahidi  
[PhD ETH 2013]

# 2013: A Surprising Improvement in **Condensing**

- reorder block matrix multiplications, reduce  **$O(N^3)$  to  $O(N^2)$  complexity!**
- independently discovered by G. Frison and J. Andersson.  
Implemented efficiently by M. Vukov in ACADO.



**Milan Vukov**



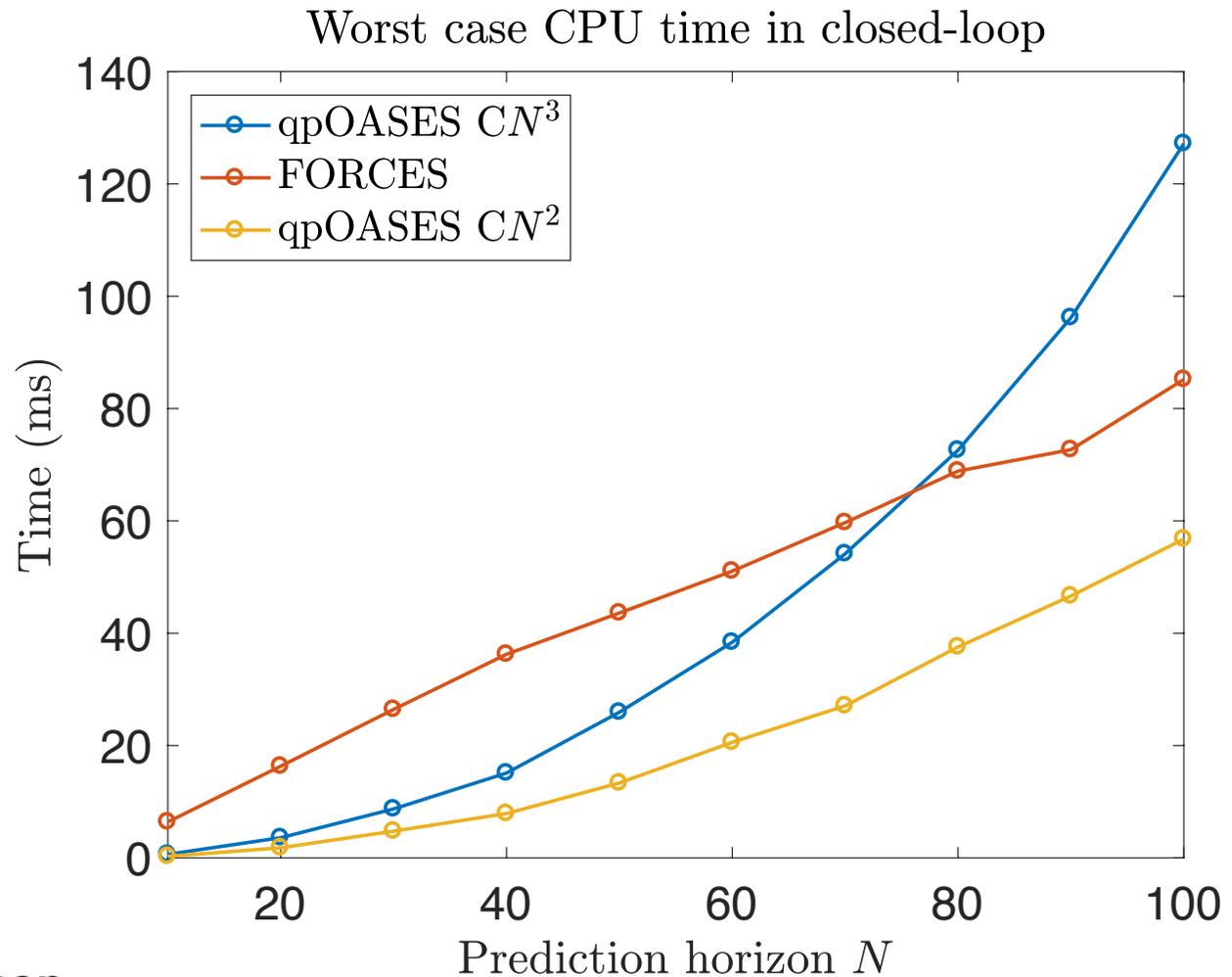
**Gianluca Frison**



**Joel Andersson**

# 2013: A Surprising Improvement in **Condensing**

- reorder block matrix multiplications, reduce  **$O(N^3)$  to  $O(N^2)$  complexity!**
- independently discovered by G. Frison and J. Andersson.  
Implemented efficiently by M. Vukov in ACADO.



Milan Vukov



Gianluca Frison



Joel Andersson

# 2014: The dual Newton strategy **qpDUNES**



**Janick Frasch**

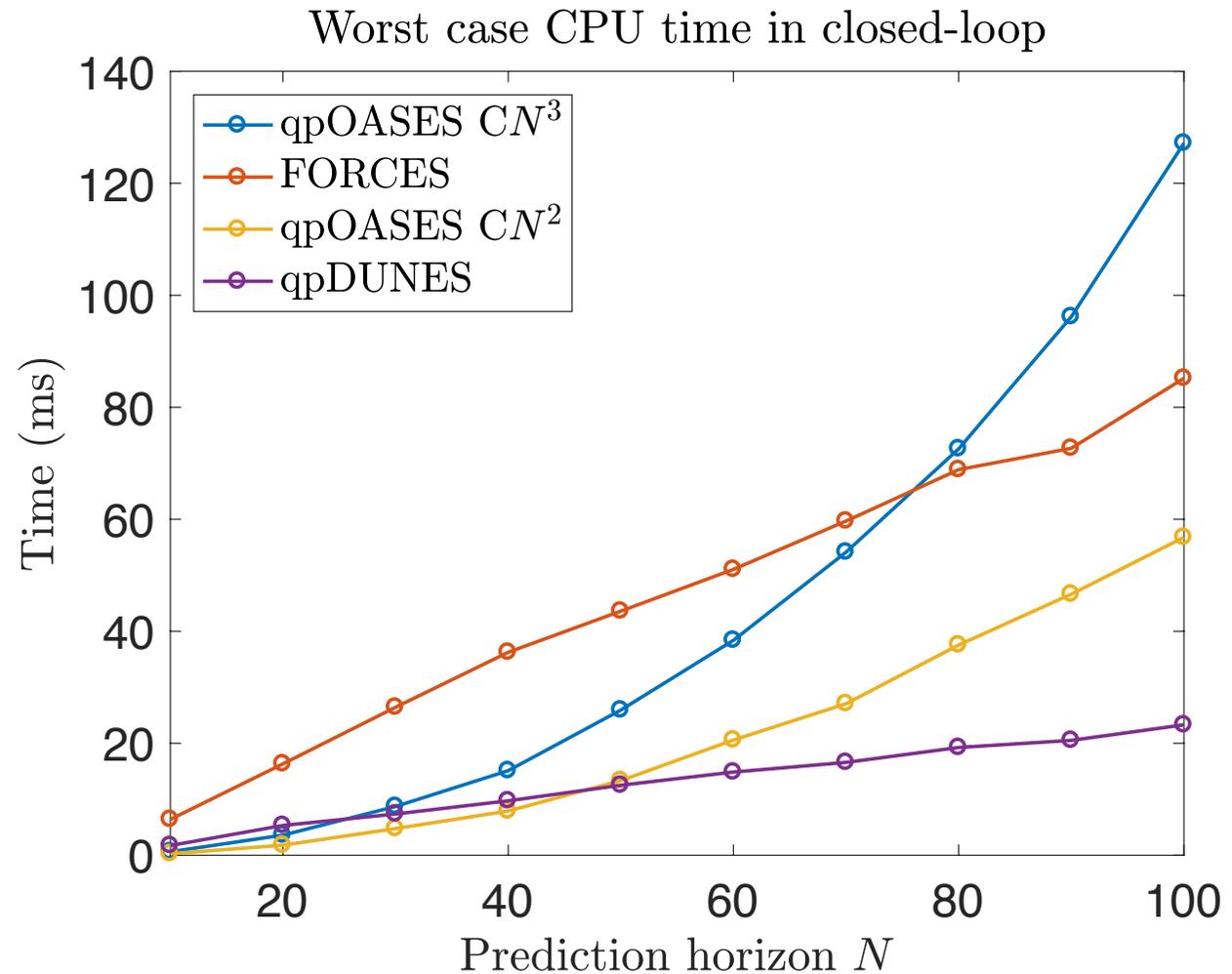
- qpDUNES (Frasch et al. 2015) uses Lagrangian decomposition, solves many small QPs independently, and performs sparse Cholesky factorisations

# 2014: The dual Newton strategy qpDUNES



Janick Frasch

- qpDUNES (Frasch et al. 2015) uses Lagrangian decomposition, solves many small QPs independently, and performs sparse Cholesky factorisations



# 2015: Efficient Register Management in **HPMPC**

- use interior point method with Riccati solver of  **$O(N)$  complexity**
- use linear algebra routines tailored to embedded optimization (now in BLASFEO) obtaining near peak CPU performance (**Gianluca Frison**)
- include in ACADO (M. Vukov, A. Zanelli, G. Frison)



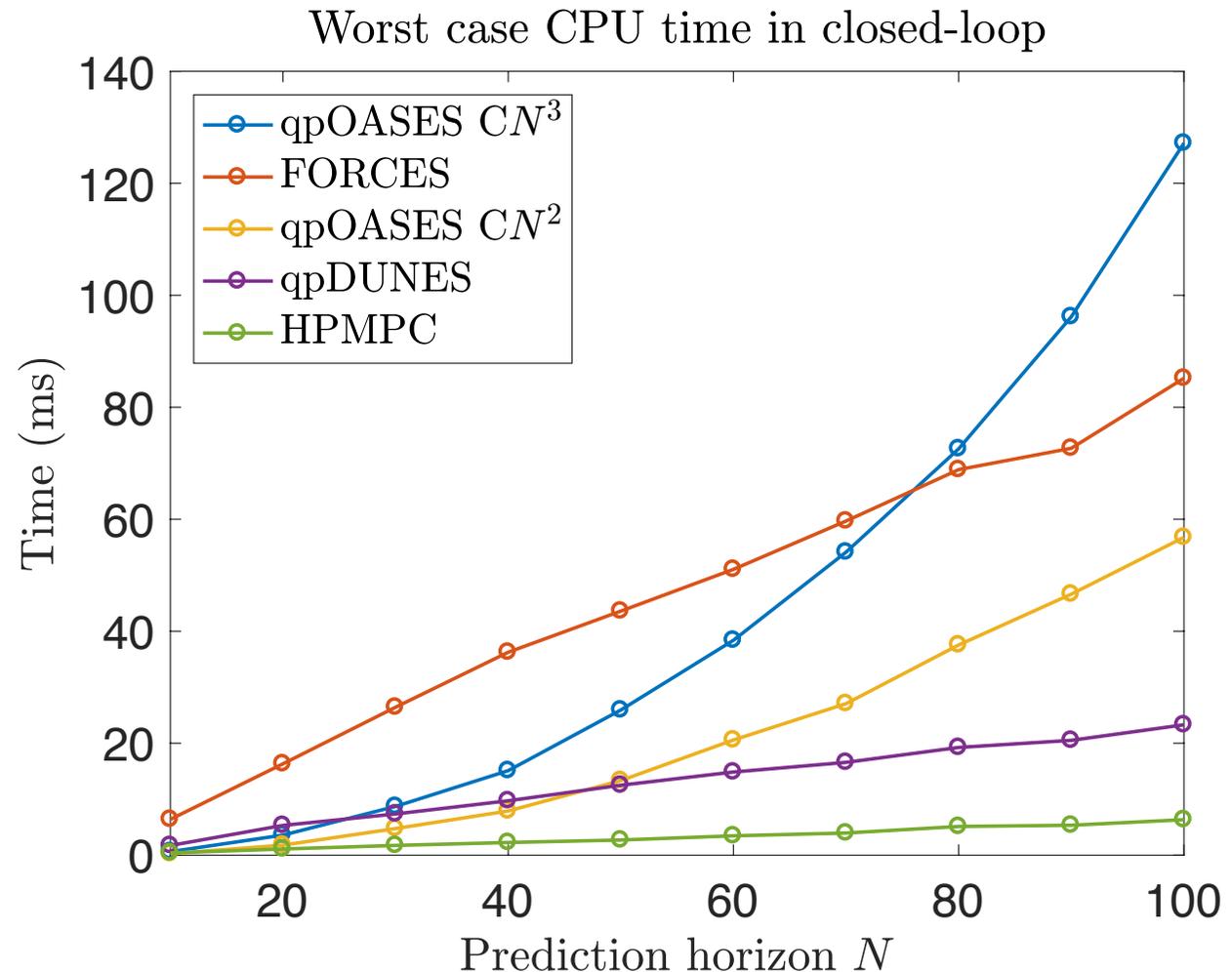
**Gianluca Frison**

# 2015: Efficient Register Management in HPMPC

- use interior point method with Riccati solver of  **$O(N)$  complexity**
- use linear algebra routines tailored to embedded optimization (now in BLASFEO) obtaining near peak CPU performance (**Gianluca Frison**)
- include in ACADO (M. Vukov, A. Zanelli, G. Frison)



**Gianluca Frison**



# 2016: Partial Condensing in HPMPC

- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



**Daniel Axehill**



**Gianluca Frison**

# 2016: Partial Condensing in HPMPC

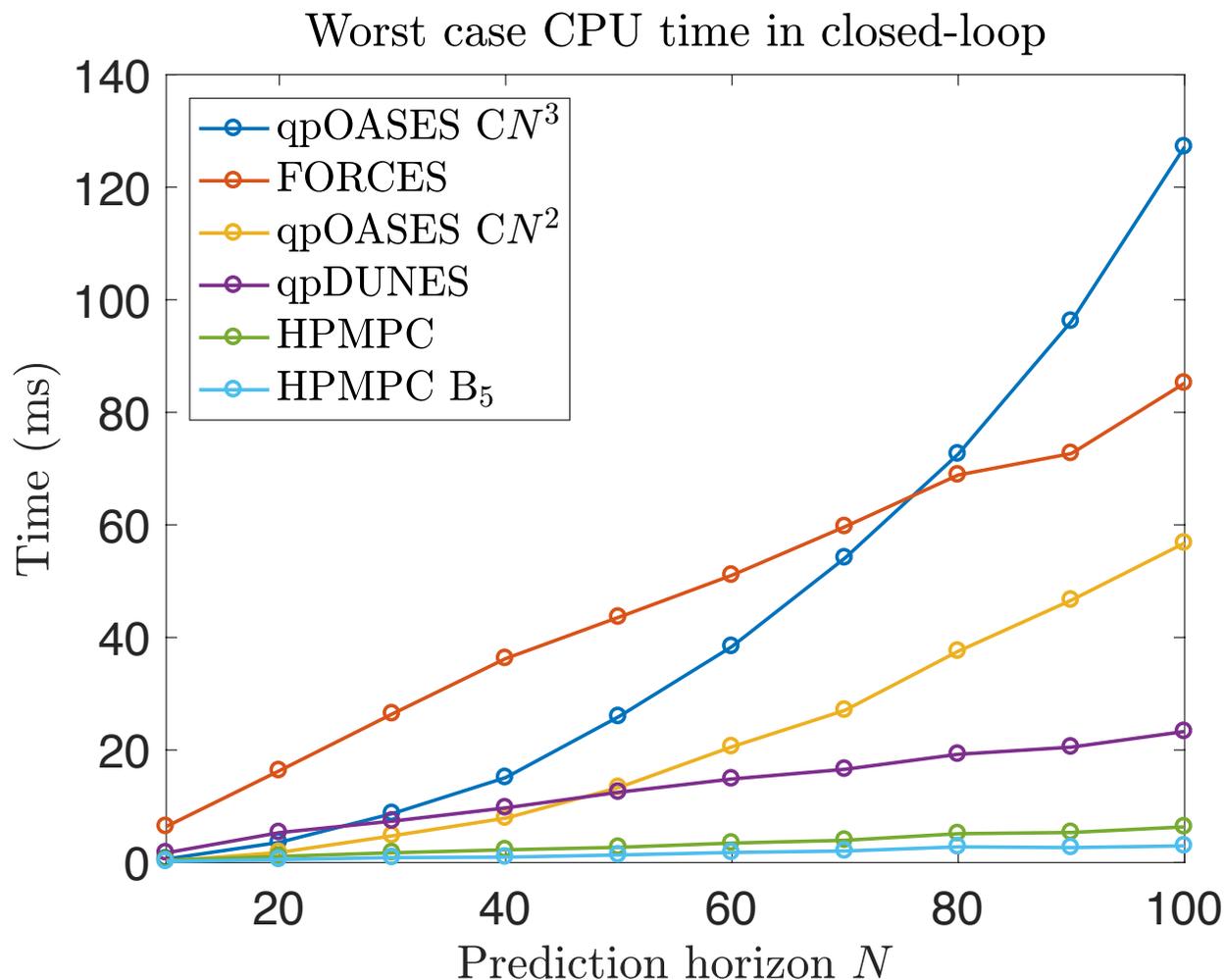
- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



Daniel Axehill



Gianluca Frison



# 2016: Partial Condensing in HPMPC

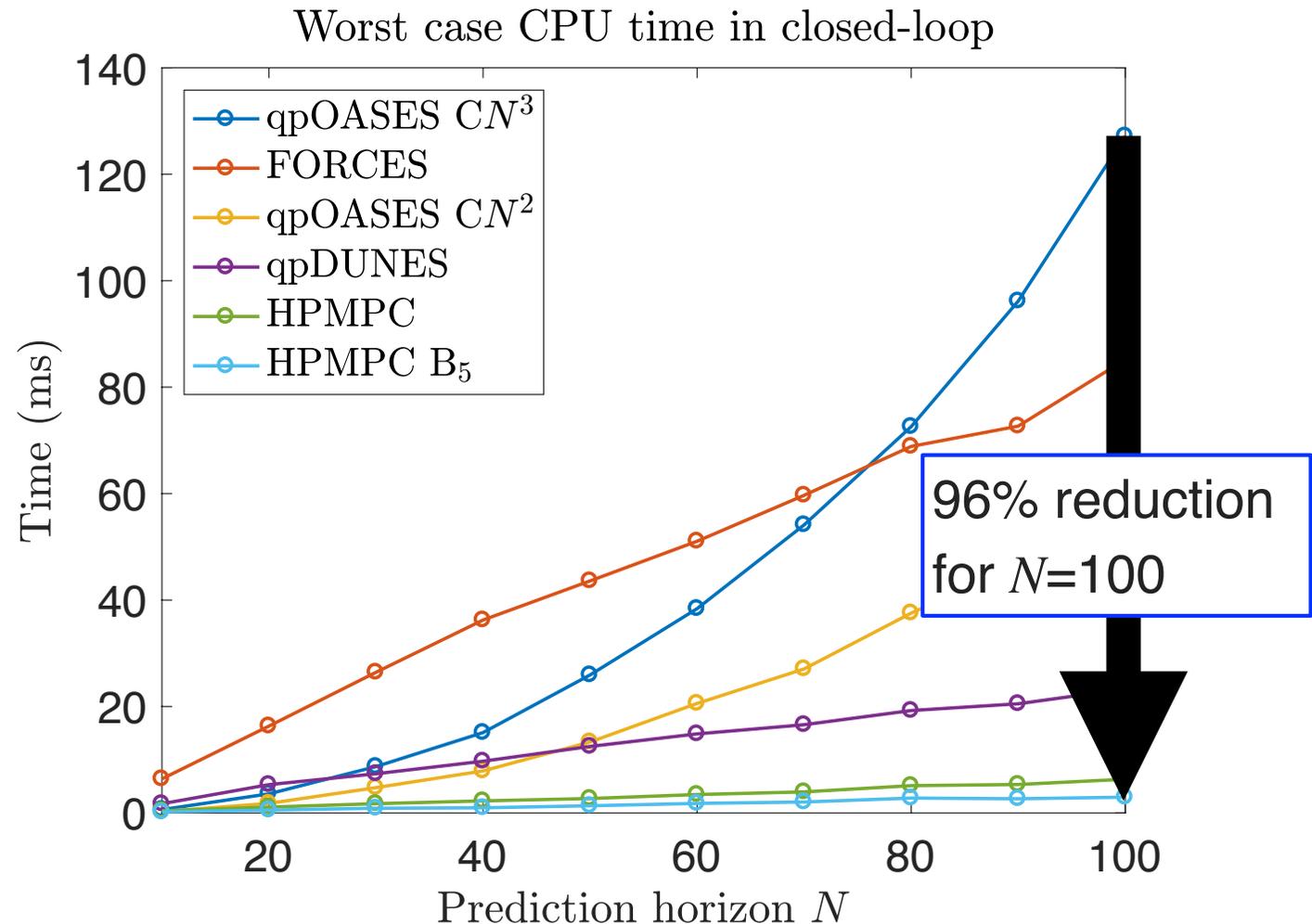
- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



Daniel Axehill



Gianluca Frison



# 2016: Partial Condensing in HPMPC

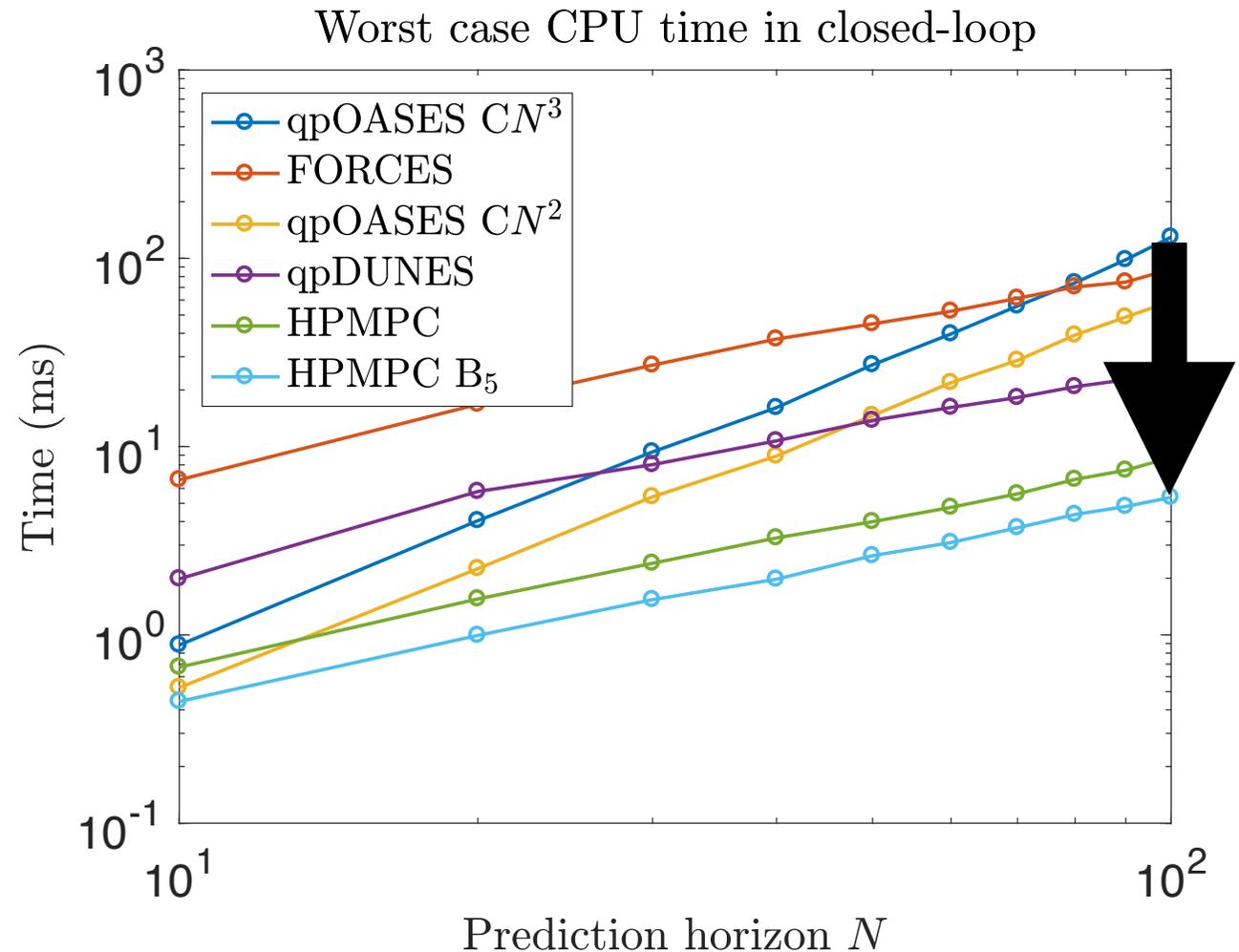
- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



Daniel Axehill



Gianluca Frison



# 2016: Partial Condensing in HPMPC

- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)

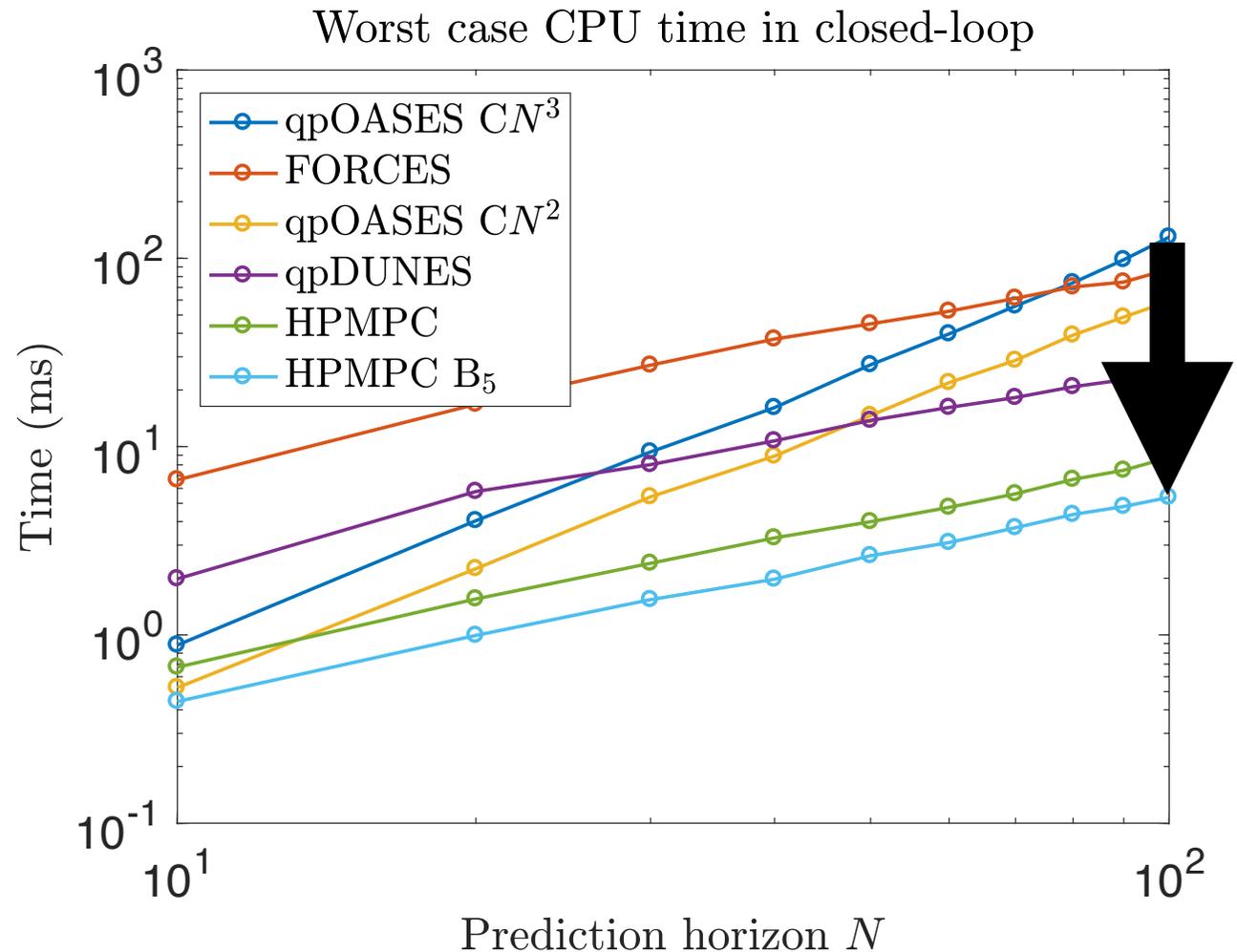


Daniel Axehill



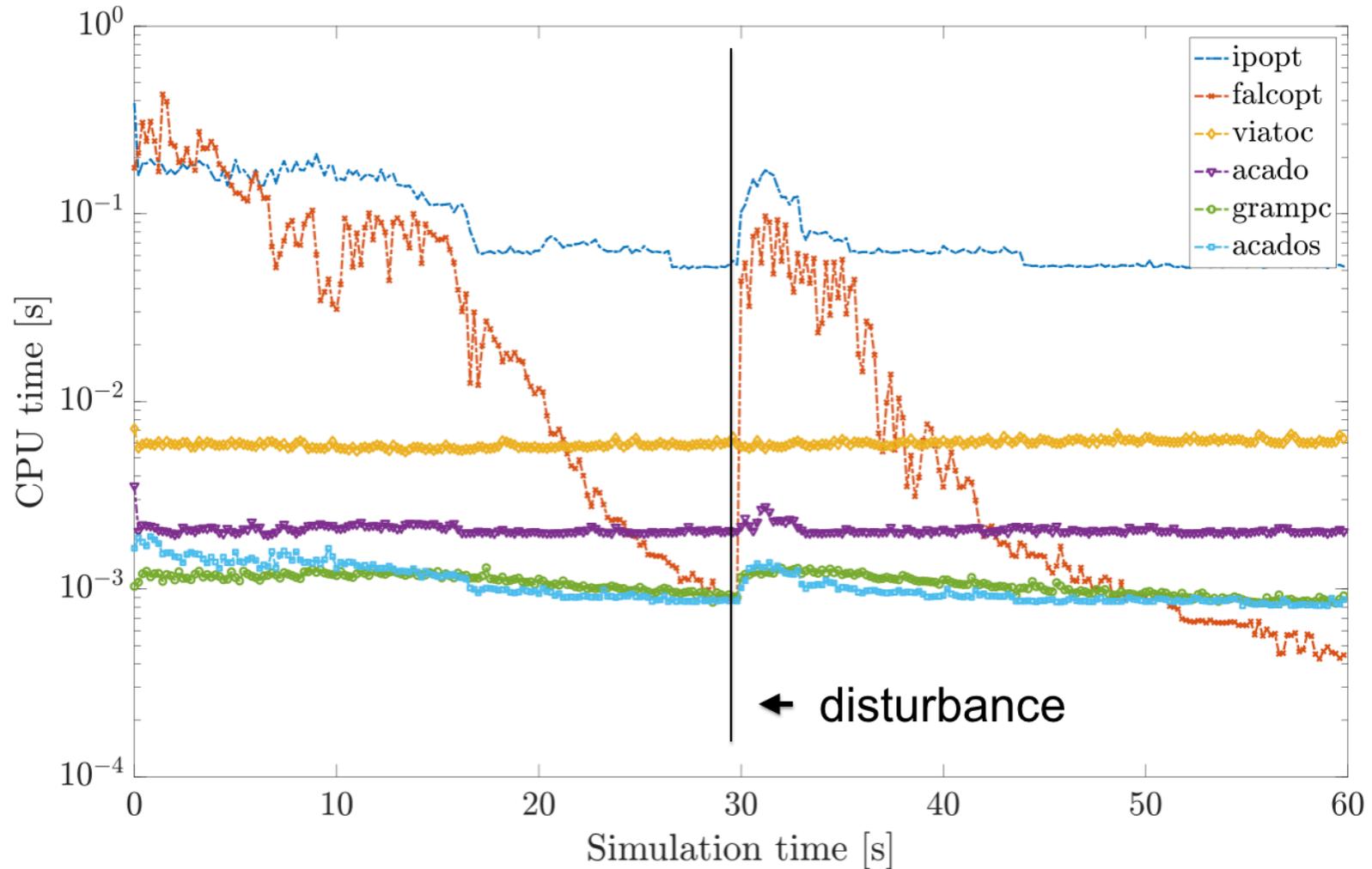
Gianluca Frison

“Basic Linear Algebra  
Subroutines for Embedded  
Optimization (BLASFEO)”



# Current development: acados - plain C-code library

**benchmark - chain of masses<sup>2</sup>: 33 states, 3 controls, horizon length 40**

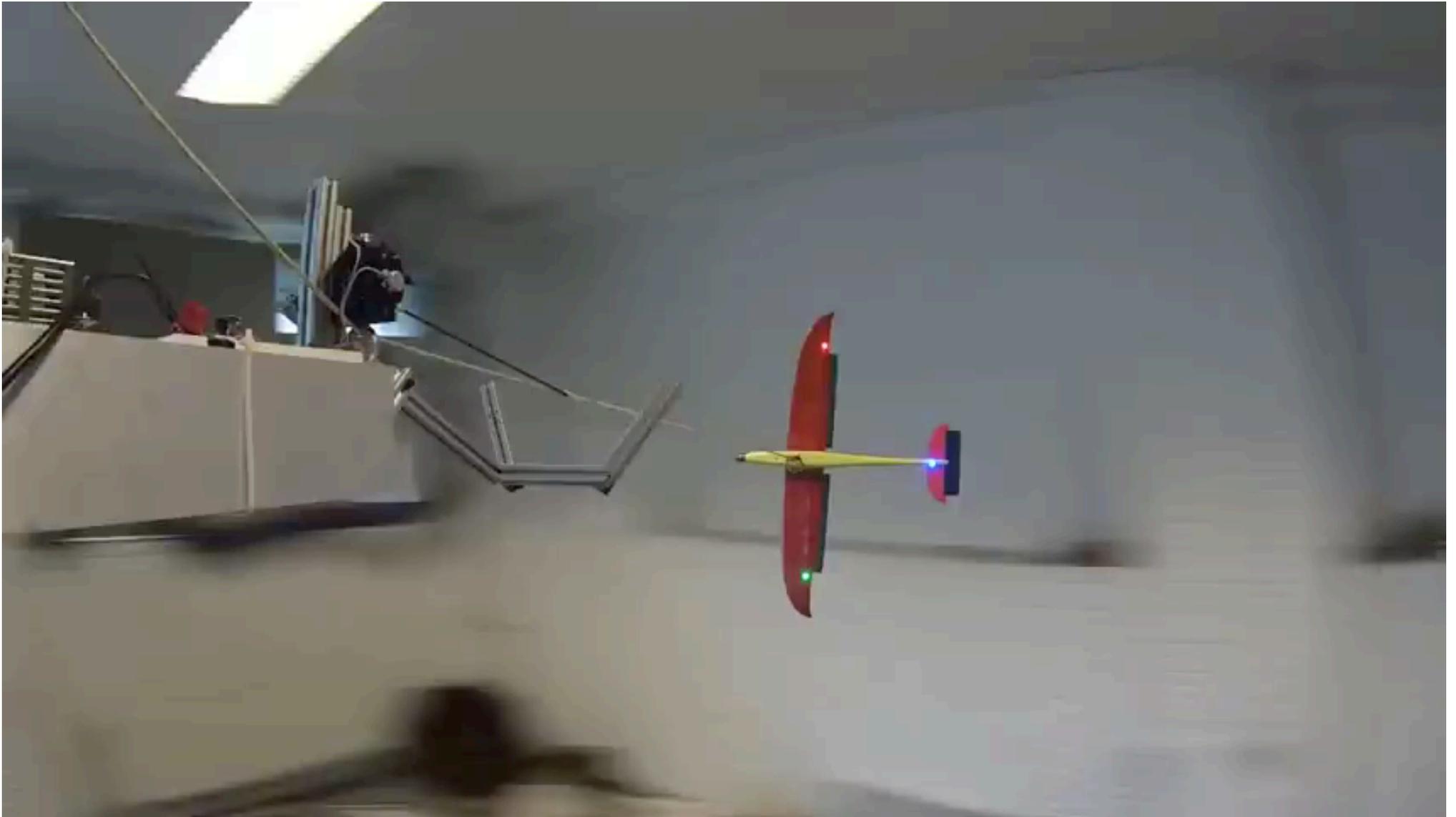


# Overview

- Embedded Optimization and Model Predictive Control (MPC)
- Real-Time Optimal Control Methods
- Progress in Numerical Integration Methods
- Progress in Structured Quadratic Programming
- **Some real-world NMPC applications**

# 1) Flight Carousel for Tethered Airplanes

Experiments within the ERC Project HIGHWIND Leuven/Freiburg





Milan Vukov

Moving Horizon Estimation and Nonlinear  
Model Predictive Control on the Flight Carousel  
(sampling time 50 Hz, using ACADO Code Generation)

# Closed loop experiments with NMPC & NMHE

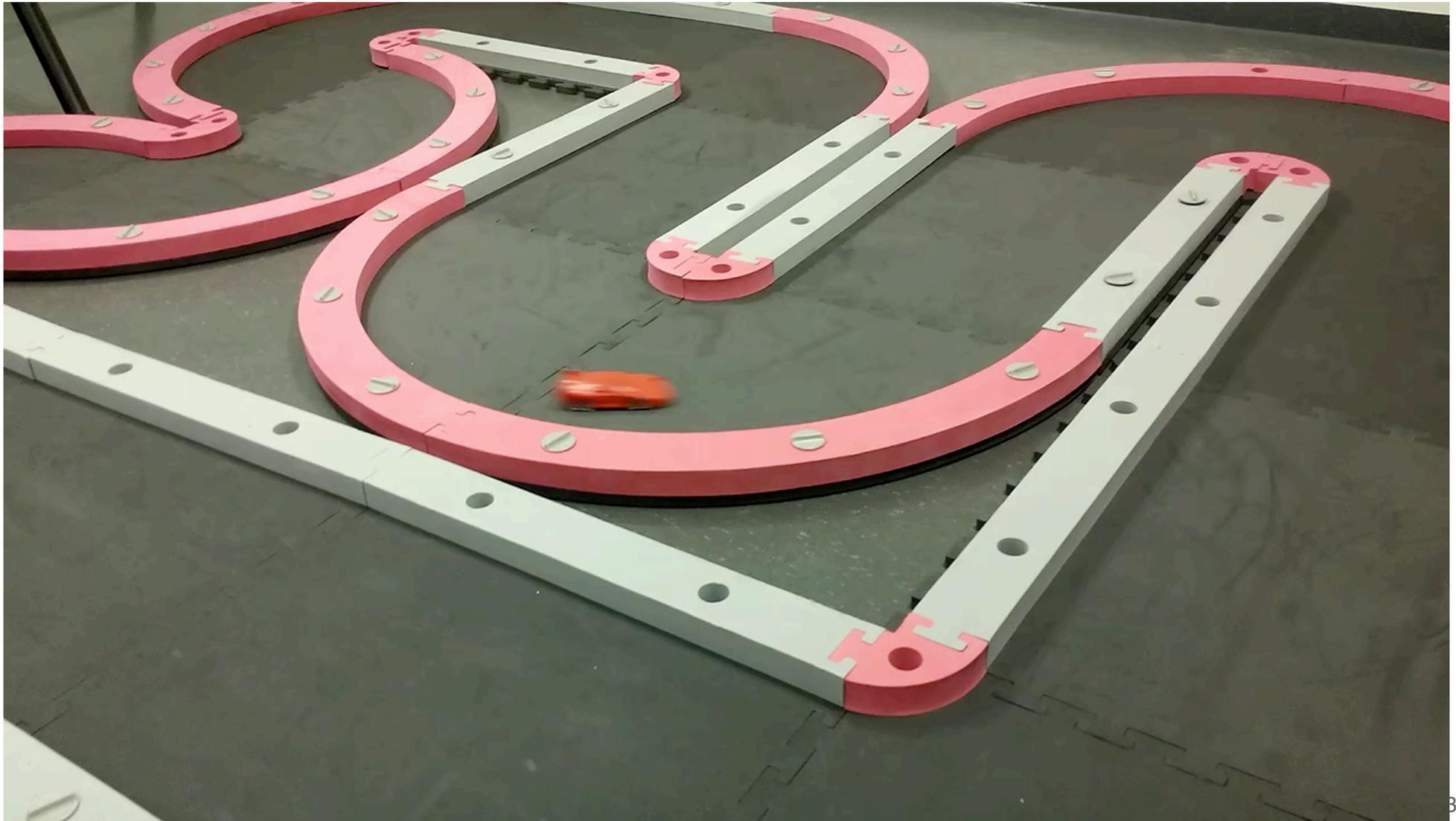


## 2) Nonlinear MPC Example: time-optimal “racing” of model cars



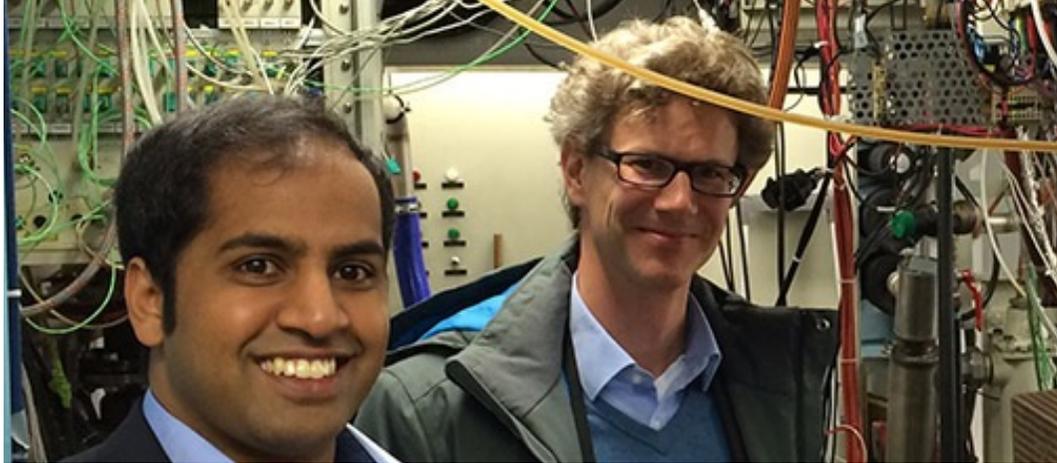
Freiburg/Leuven/ETH/Siemens-PLM. 100 Hz sampling time using ACADO [Verschueren, De Bruyne, Zanon, Frasch, D. CDC 2014] (Nonlinear MPC video from 22.6.2016 in Freiburg)

**Robin Verschueren**



### 3) Nonlinear MPC of Two-Stage Turbocharger with ACADO

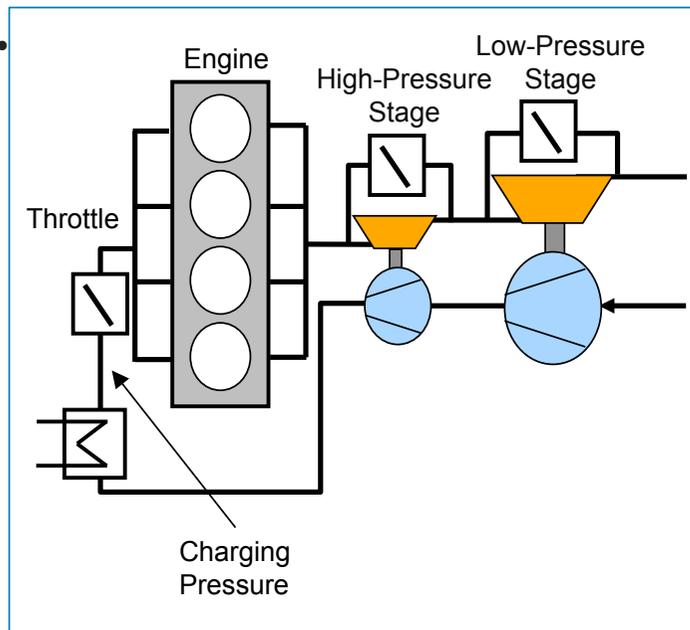
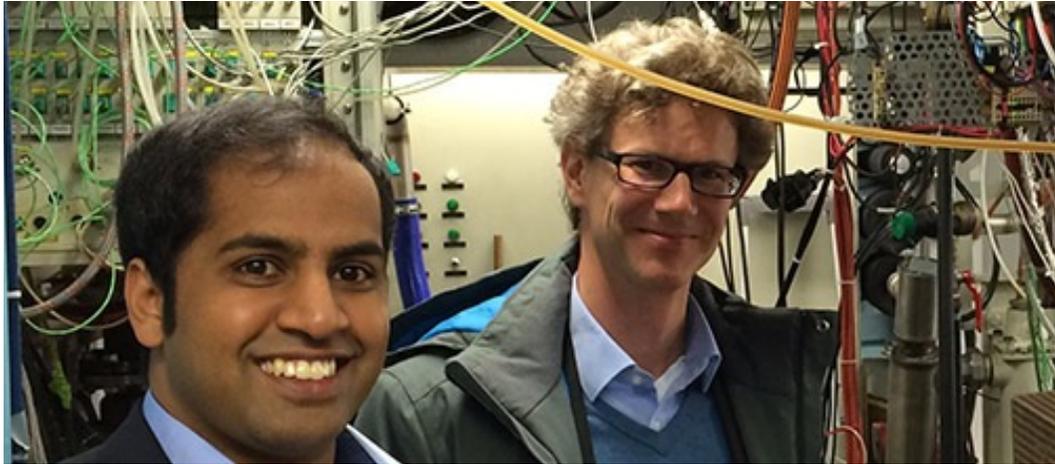
Cooperation with Dr. Thiva Albin (RWTH Aachen) and **Rien Quirynen**



•

# 3) Nonlinear MPC of Two-Stage Turbocharger with ACADO

Cooperation with Dr. Thiva Albin (RWTH Aachen) and **Rien Quirynen**



## Actuated Variables

- Wastegate-High Pressure
- Wastegate-Low Pressure

## Controlled Variable / Sensors

- Charging pressure
- No sensors in the exhaust gas path



test car of RWTH Aachen

# 3) Nonlinear MPC of Two-Stage Turbocharger with ACADO

Cooperation with Dr. Thiva Albin (RWTH Aachen) and **Rien Quirynen**

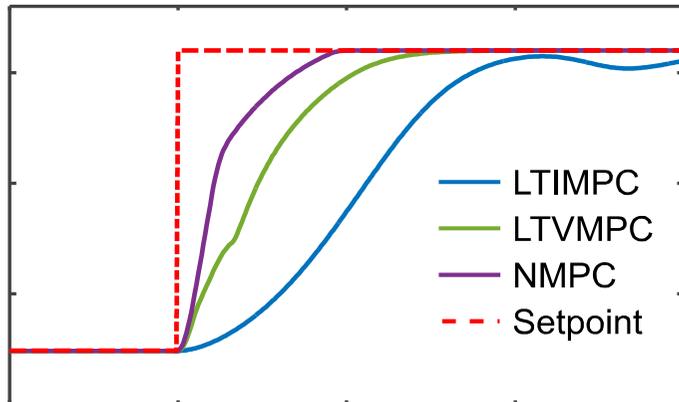
- use nonlinear DAE model with 4 states, 2 controls
- use ACADO Code Generation from MATLAB
- export C-code into Simulink
- deploy on dSPACE Autobox



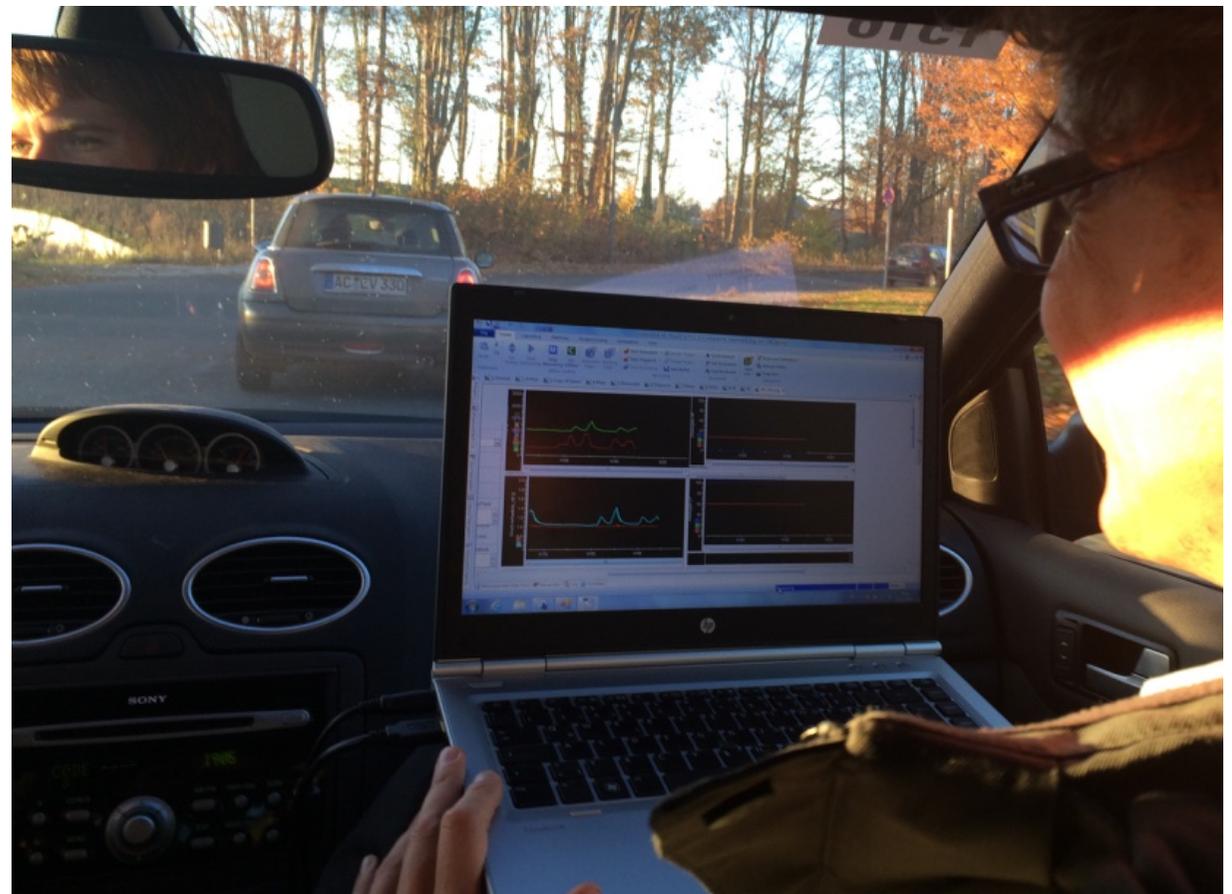
$$\begin{aligned} \frac{d}{dt} \left( \frac{1}{2} J_{tc, hp} n_{tc, hp}^2 \right) &= P_{t, hp} - P_{c, hp} \\ \frac{d}{dt} \left( \frac{1}{2} J_{tc, lp} n_{tc, lp}^2 \right) &= P_{t, lp} - P_{c, lp} \\ P_{c, hp} &= \dot{m}_{c, hp} c_p T_{uc, hp} \frac{1}{\eta_{s, c, hp}} \left( \Pi_{c, hp}^{\frac{\kappa-1}{\kappa}} - 1 \right) \\ P_{c, lp} &= \dot{m}_{c, lp} c_p T_{amb} \frac{1}{\eta_{s, c, lp}} \left( \Pi_{c, lp}^{\frac{\kappa-1}{\kappa}} - 1 \right) \\ P_{t, hp} &= \dot{m}_{t, hp} c_p T_{ut, hp} \eta_{s, t, hp} \left( 1 - \Pi_{t, hp}^{\frac{1-\kappa}{\kappa}} \right) \\ P_{t, lp} &= \dot{m}_{t, lp} c_p T_{ut, lp} \eta_{s, t, lp} \left( 1 - \Pi_{t, lp}^{\frac{1-\kappa}{\kappa}} \right) \end{aligned}$$

### 3) Nonlinear MPC of Two-Stage Turbocharger with ACADO

Nonlinear MPC superior to Linear MPC in simulations:



Implemented in test car of RWTH Aachen and tested on a test drive and the road.



[driving a happy M.D. to Aachen Hbf on 2.11.2015]

## 4) Electrical Compressor Control at ABB (Norway)

- work of Dr. Joachim Ferreau and Dr. Thomas Besselmann, ABB
- nonlinear MPC with qpOASES and ACADO, 1ms sampling time
- first tests at 48 MW Drive
- currently, 15% of Norwegian Gas Exports are controlled by Nonlinear MPC



## 4) Electrical Compressor Control at ABB (Norway)

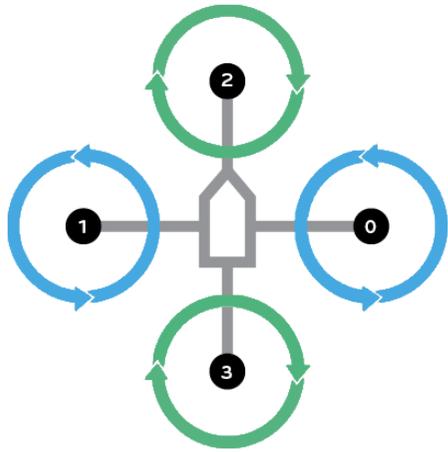
- work of Dr. Joachim Ferreau and Dr. Thomas Besselmann, ABB
- nonlinear MPC with qpOASES and ACADO, 1ms sampling time
- first tests at 48 MW Drive
- currently, 15% of Norwegian Gas Exports are controlled by Nonlinear MPC



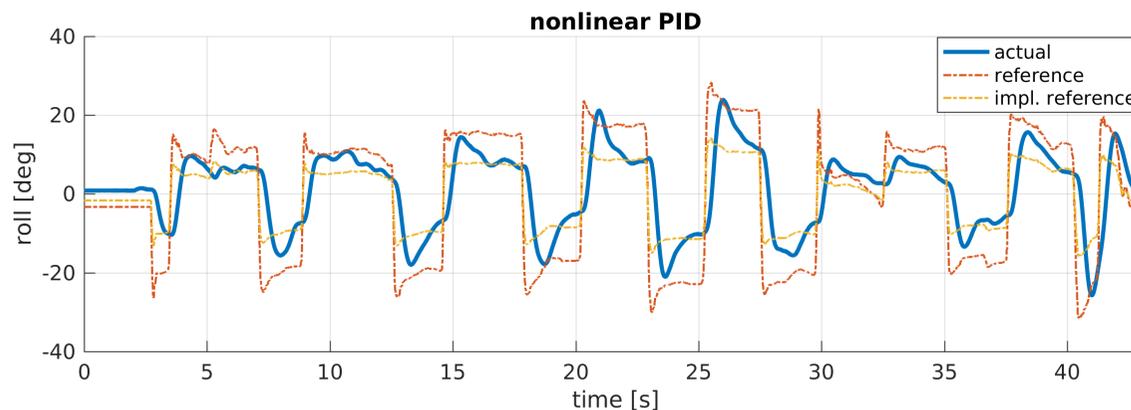
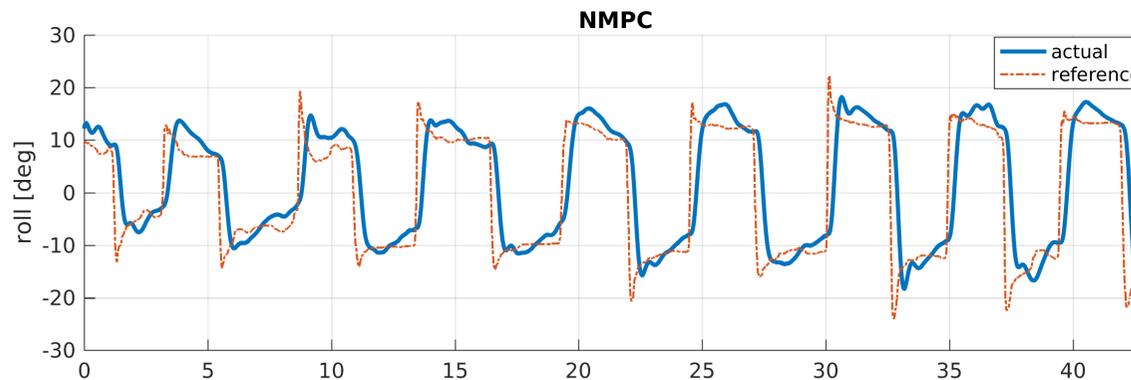
Joachim Ferreau (email from 7.3.2016):

The NMPC installations in Norway (actually 5 compressors at two different sites) are doing fine since last autumn – roughly 80 billion NMPC instances solved by now. In addition, they have proven to work as expected when handling external voltage dips.

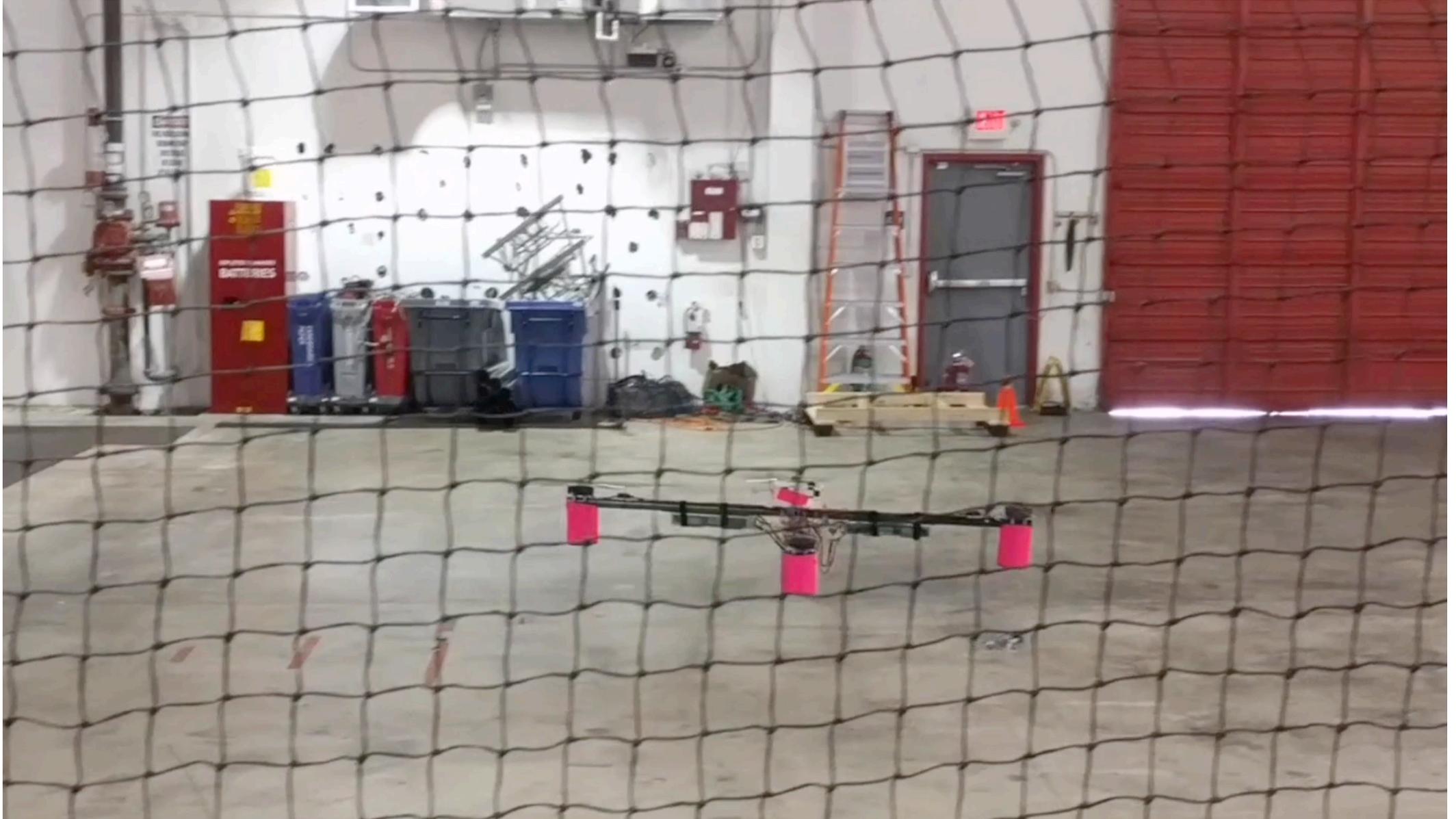
# 5) Human sized quadcopter control



- work by Greg Horn (Kittyhawk, California) and Andrea Zanelli (Freiburg) in April 2017
- aim is to track roll angle commands better than a custom PID
- nonlinear MPC with 11 states, 4 controls, 10 intervals
- use **HPMPC**, custom linear algebra **BLASFEO**, and “partial tightening” NMPC scheme within **acados** environment
- achieve 2 ms per optimisation on ARM cortex A9 @900 MHz



## 5) Human sized quadcopter control (NMPC) at Kittyhawk



# Conclusions and outlook

**Model Predictive Control (MPC)** uses more CPU resources than standard techniques, but allows the development of **more powerful** controllers with **wider range of validity**

good numerical methods can solve nonlinear optimal control problems at **milli- and microsecond sampling times** on embedded systems

**open source software (CasADi, qpOASES, ACADO, HPIPM, BLASFEO, acados)** well-tested in dozens of embedded MPC applications: cranes, wafer steppers, model race cars, combustion engines, electrical drives, tethered airplanes, power converters,...

Latest open source developments in the Freiburg team regard

- mixed-integer nonlinear optimal control algorithms
- nonsmooth optimal control algorithms (see previous summer school)
- combinations of MPC & Reinforcement Learning (“differentiable acados”)

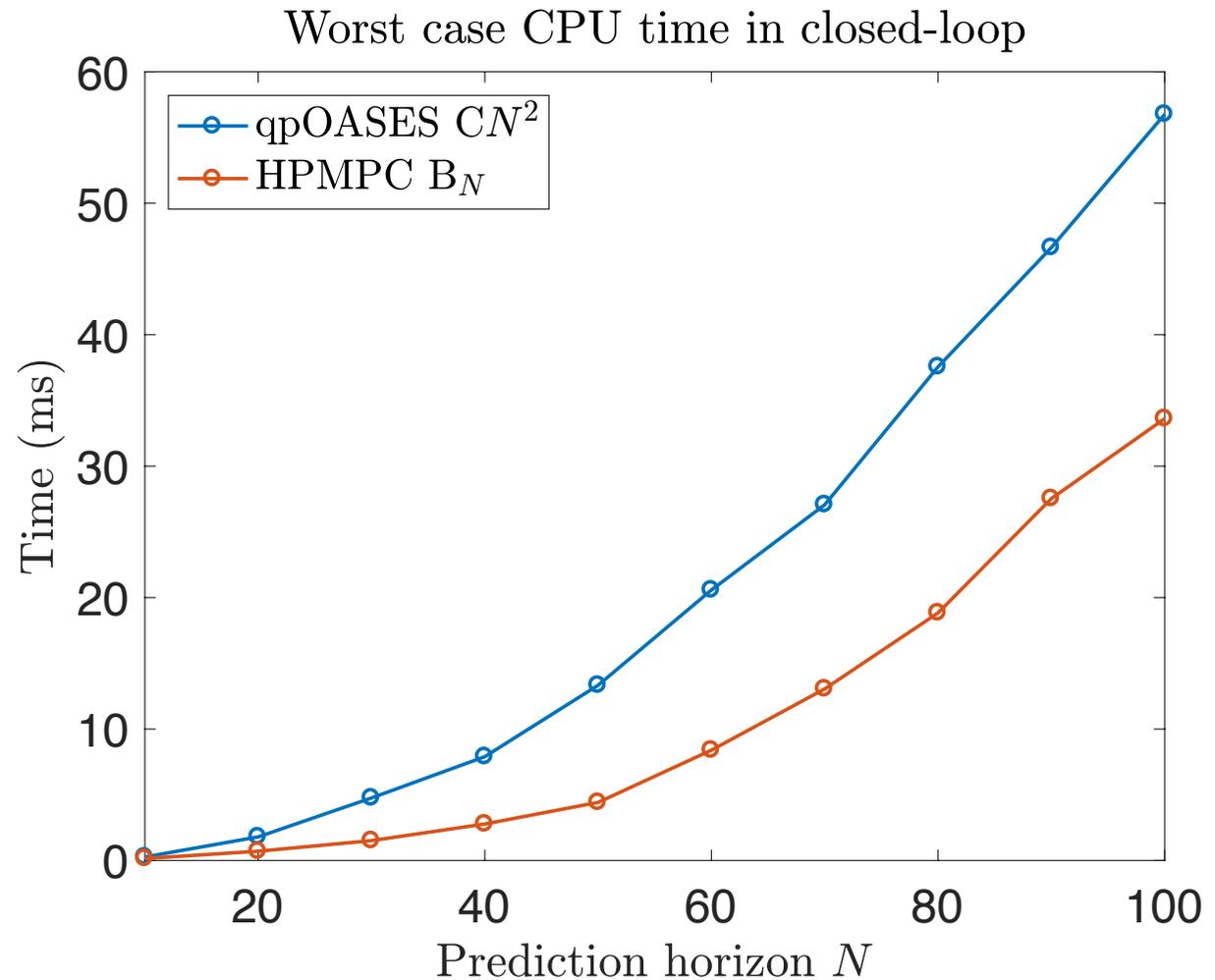
# Appendix

# 2017: Full condensing with BLASFEO and HPMPC

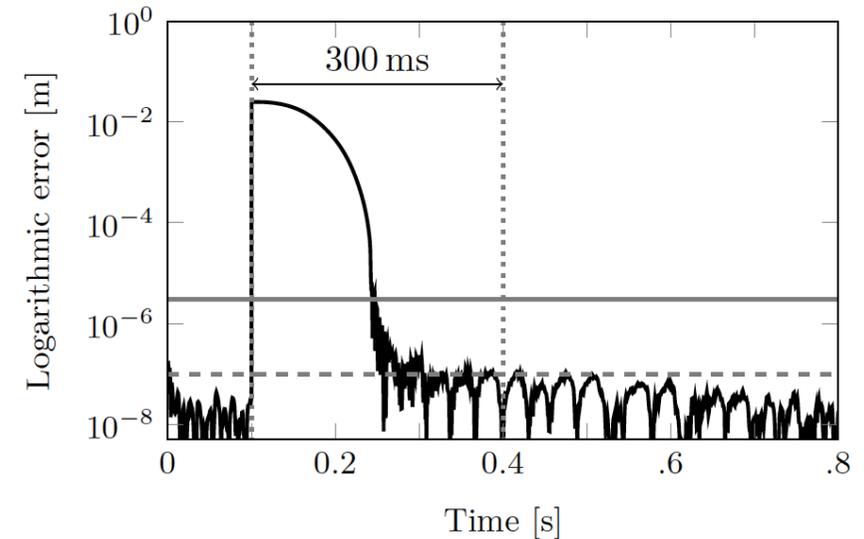
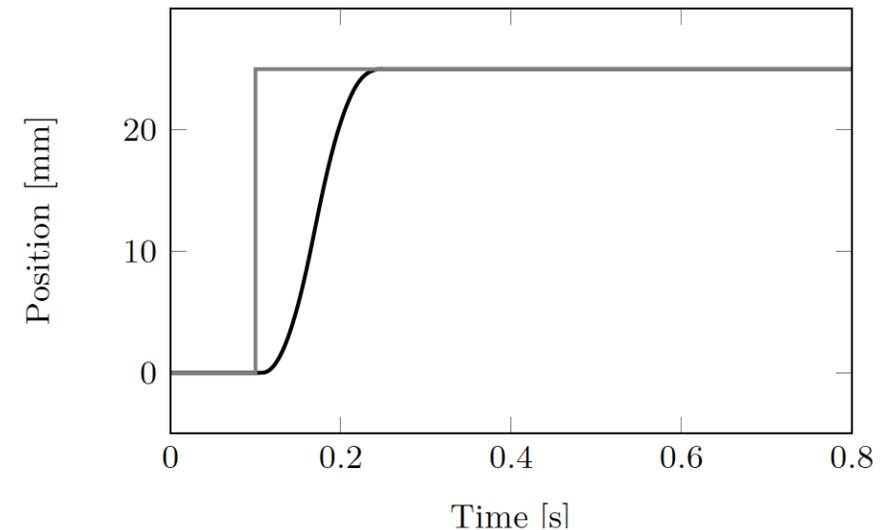
- block size in HPMPC with partial condensing equal to prediction horizon.



**Gianluca Frison**



# Time Optimal MPC at ETEL (CH): 25cm step, 100nm accuracy



TOMPC at 250 Hz (+PID with 12 kHz)

Lieboud's results after 1 week at ETEL:

- 25 cm step in 300 ms
- 100 nm accuracy

equivalent to: „fly 2,5 km with MACH15,  
stop with 1 mm position accuracy“