Rien Quirynen

ACADO Code Generation tool

including material from B. Houska and M. Vukov

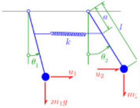
TEMPO Spring School on NMPC

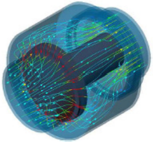
University of Freiburg

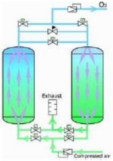
Outline

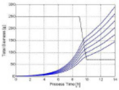
Outline

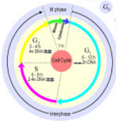
Nonlinear Dynamic Systems

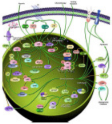


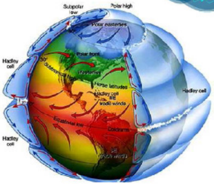




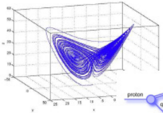


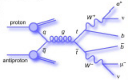


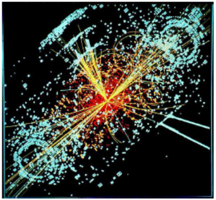


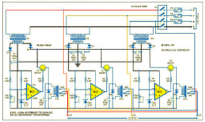


$$\boxed{\dot{x}(t) = f(x(t), \ldots)}$$









Optimal Control

Many Fields of Application:

- Optimal Motions in Robotics
- Operation of a Chemical Plant
- Seasonal Heat Storage
- Kite Power

Problems:

- Optimize Parameters/Controls
- Uncertainties/Disturbances

`www.acadotoolkit.org`

Key Properties of ACADO Toolkit [Houska et al 2009]

- Open Source (LGPL)
- **A**utomatic **C**ontrol **A**nd **D**ynamic **O**ptimization
- User friendly interface close to mathematical syntax

www.acadotoolkit.org

Key Properties of ACADO Toolkit [Houska et al 2009]

- Open Source (LGPL)
- **A**utomatic **C**ontrol **A**nd **D**ynamic **O**ptimization
- User friendly interface close to mathematical syntax

Multiplatform support

- C++: Linux, OS X, Windows
- MATLAB

ACADO Toolkit:

- **A**utomatic **C**ontrol **A**nd **D**ynamic **O**ptimization
- Open Source (LGPL) www.acadotoolkit.org

ACADO Toolkit:

- **A**utomatic **C**ontrol **A**nd **D**ynamic **O**ptimization
- Open Source (LGPL) www.acadotoolkit.org

List of Developers:

Moritz Diehl
Scientific advisor

Hans Joachim Ferreau
Main developer

Boris Houska
Main developer

Filip Logist
Multi-objective optimization

Rien Quirynen
Code generation

Dries Telen
Optimal Experimental Design

Mattia Valerio
Multi-objective optimal control

Milan Vukov
Code generation for MPC & MHE

Tutorial Example: Time Optimal Control of a Rocket

Mathematical Formulation:

$$\underset{s(\cdot),v(\cdot),m(\cdot),u(\cdot),T}{\text{minimize}} \quad T$$

subject to

$$\begin{aligned}
\dot{s}(t) &= v(t) \\
\dot{v}(t) &= \tfrac{u(t)-0.2\,v(t)^2}{m(t)} \\
\dot{m}(t) &= -0.01\,u(t)^2
\end{aligned}$$

$$\begin{aligned}
s(0) &= 0 \quad & s(T) &= 10 \\
v(0) &= 0 \quad & v(T) &= 0 \\
m(0) &= 1
\end{aligned}$$

$$\begin{aligned}
-0.1 &\leq v(t) \leq 1.7 \\
-1.1 &\leq u(t) \leq 1.1 \\
5 &\leq T \leq 15
\end{aligned}$$

Tutorial Example: Time Optimal Control of a Rocket

Mathematical Formulation:

$$\underset{s(\cdot),v(\cdot),m(\cdot),u(\cdot),T}{\text{minimize}} \quad T$$

subject to

$$\begin{aligned}
\dot{s}(t) &= v(t) \\
\dot{v}(t) &= \frac{u(t)-0.2\,v(t)^2}{m(t)} \\
\dot{m}(t) &= -0.01\,u(t)^2
\end{aligned}$$

$$\begin{aligned}
s(0) &= 0 & s(T) &= 10 \\
v(0) &= 0 & v(T) &= 0 \\
m(0) &= 1
\end{aligned}$$

$$\begin{aligned}
-0.1 &\leq v(t) \leq 1.7 \\
-1.1 &\leq u(t) \leq 1.1 \\
5 &\leq T \leq 15
\end{aligned}$$

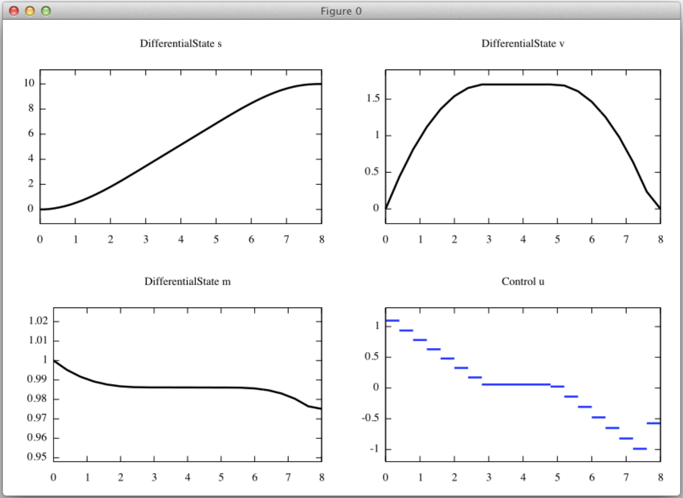
```
DifferentialState                s,v,m;
Control                             u;
Parameter                           T;
DifferentialEquation     f( 0.0, T );
OCP ocp( 0.0, T );
ocp.minimizeMayerTerm( T );

f << dot(s) == v;
f << dot(v) == (u-0.2*v*v)/m;
f << dot(m) == -0.01*u*u;
ocp.subjectTo( f                    );

ocp.subjectTo( AT_START, s ==  0.0 );
ocp.subjectTo( AT_START, v ==  0.0 );
ocp.subjectTo( AT_START, m ==  1.0 );
ocp.subjectTo( AT_END  , s == 10.0 );
ocp.subjectTo( AT_END  , v ==  0.0 );

ocp.subjectTo( -0.1 <= v <=  1.7   );
ocp.subjectTo( -1.1 <= u <=  1.1   );
ocp.subjectTo(  5.0 <= T <= 15.0   );
OptimizationAlgorithm algorithm(ocp);
algorithm.solve();
```

Optimization Results



Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)
- **Feedback control (NMPC) and closed loop simulation**

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)
- **Feedback control (NMPC) and closed loop simulation**
- **Robust optimal control**

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)
- **Feedback control (NMPC) and closed loop simulation**
- **Robust optimal control**

$\longrightarrow$ **"standard" ACADO Toolkit**

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)
- **Feedback control (NMPC) and closed loop simulation**
- **Robust optimal control**
- **Real-Time MPC and Code Export**

Implemented Problem Classes

- **Optimal control of dynamic systems**
  (ODE, DAE)
- **Multi-objective optimization**
  (joint work with Filip Logist)
- **State and parameter estimation/OED**
  (joint work with Dries Telen and Filip Logist)
- **Feedback control (NMPC) and closed loop simulation**
- **Robust optimal control**
- Real-Time MPC and Code Export

$\longrightarrow$ **"ACADO Code Generation"**

Outline

Automatic Code Generation

Mathematical Formulation

$$\min_{x,u} \quad \int_0^T x^2 + u^2 \mathrm{d}t$$
$$\text{s.t.} \quad \dot{x} = f(x,u), \quad x(0) = x_0, \quad -1 \leq u \leq 1\ .$$

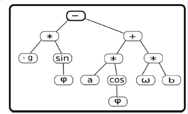
ACADO Syntax

```
DifferentialState     x;
Control               u;

DifferentialEquation f;
f << dot(x) == u + ...;

ocp.minLagrangeTerm( x*x+u*u );
ocp.subjectTo( f );
ocp.subjectTo( -1 <= u <= 1 );
```

Symbolic Structure Detection



Algorithm

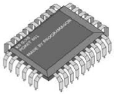
- Multiple Shooting
- Real-Time Gauss Newton
- Online Active Set Strategy

Optimized C-Code

```
r[1] = a[15]*c[17] + a[16]*c[19] + ... ;
r[2] = sin(a[1]*a[2]) + a[4] + ... ;
r[3] = cos(r[1])/exp(c[4])+ r[1] +... ;
```

Customized Solver Implemented on Chip/FPGA:

Measurement x_0



Optimal Decision u*

ACADO Toolkit

Software and algorithms for . . .

- Dynamic optimization
- code generation tool
- Fast NMPC and MHE
- MATLAB interface

ACADO Toolkit

Software and algorithms for ...

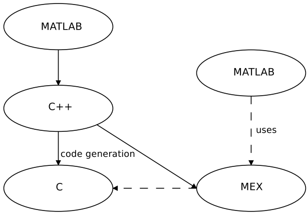
- Dynamic optimization
- code generation tool
- Fast NMPC and MHE
- MATLAB interface

Why code generation?

1. **optimization:**
   - eliminate computations
   - known dimensions and sparsity patterns
   - no dynamic memory
   - code reorganization, ...
2. **customization:** precision, language, libraries, ...

ACADO Toolkit

Software and algorithms for . . .



Why code generation?

1. **optimization:**
   - eliminate computations
   - known dimensions and sparsity patterns
   - no dynamic memory
   - code reorganization, . . .
2. **customization:** precision, language, libraries, . . .

Outline

Optimal Control Problem (OCP)

- parametric: initial condition

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \|F(t,x(t),u(t)) - \bar{y}(t)\|_2^2 \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0, T]
\end{aligned}
$$

Optimal Control Problem (OCP)

- parametric: initial condition
- tracking MPC

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \| F(t, x(t), u(t)) - \bar{y}(t) \|_2^2 \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t, x(t), u(t)) \\
& 0 \geq h(x(t), u(t)) \\
& 0 \geq r(x(0), x(T)) \\
& \forall t \in [0, T]
\end{aligned}
$$

Optimal Control Problem (OCP)

- parametric: initial condition
- tracking MPC
- nonlinear model

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \|F(t,x(t),u(t)) - \bar{y}(t)\|_2^2 \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0,T]
\end{aligned}
$$

Optimal Control Problem (OCP)

Tracking MPC

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \|F(t,x(t),u(t)) - \bar{y}(t)\|_2^2 \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0,T]
\end{aligned}
$$

Economic MPC

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T l(t,x(t),u(t)) \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0,T]
\end{aligned}
$$

Optimal Control Problem (OCP)

Tracking MPC: continuous

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \|F(t,x(t),u(t)) - \bar{y}(t)\|_2^2 \,\mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0,T]
\end{aligned}
$$

Optimal Control Problem (OCP)

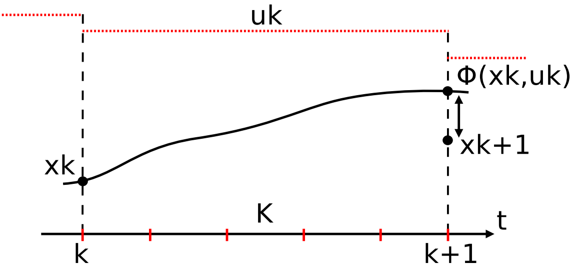
Tracking MPC: continuous $\rightarrow$ *shooting discretization*

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^T \|F(t,x(t),u(t)) - \bar{y}(t)\|_2^2 \, \mathrm{d}t \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = f(t,x(t),u(t)) \\
& 0 \geq h(x(t),u(t)) \\
& 0 \geq r(x(0),x(T)) \\
& \forall t \in [0,T]
\end{aligned}
$$

$$
\begin{aligned}
\min_{x,u} \quad & \sum_{i=0}^{N-1} \|F_i(x_i,u_i) - \bar{y}_i\|_2^2 + \|F_N(x_N)\|_2^2 \\
\text{s.t.} \quad & 0 = x_0 - \bar{x}_0 \\
& 0 = x_{i+1} - \Phi_i(x_i,u_i) \\
& 0 \geq h_i(x_i,u_i) \\
& 0 \geq r(x_0,x_N) \\
& \forall i = 0,\ldots,N-1
\end{aligned}
$$

Task of the integrator in RTI

- $x_{k+1} = \Phi_k(x_k, u_k)$
- nonlinear equality constraint

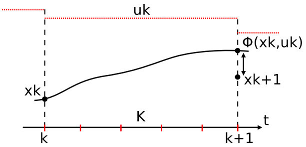


Task of the integrator in RTI

- $x_{k+1} = \Phi_k(x_k, u_k)$
- nonlinear equality constraint

$$\downarrow$$

- linearization at $\bar{w}_k = (\bar{x}_k, \bar{u}_k)$



$$0 = \mathbf{\Phi_k(\bar{w}_k)} - x_{k+1} + \frac{\partial \mathbf{\Phi_k}}{\partial \mathbf{w}}(\mathbf{\bar{w}_k})(w_k - \bar{w}_k)$$

- integration and sensitivity generation is typically a major computational step

The 3-stage model structure

Linear Input System

$$C_1\dot{x}^{[1]} = A_1 x^{[1]} + B_1 u$$

$\downarrow$

Nonlinear System

$$0 = f_2(x^{[1]}, x^{[2]}, \dot{x}^{[1]}, \dot{x}^{[2]}, z, u)$$

$\downarrow$

Linear Output System

$$C_3\dot{x}^{[3]} = A_3 x^{[3]} + f_3(x^{[1]}, x^{[2]}, \dot{x}^{[1]}, \dot{x}^{[2]}, z, u)$$

Sequential Quadratic Programming (SQP - RTI)

$$\underset{X,\,U}{\text{minimize}} \quad \sum_{i=0}^{N-1} \|F_i(x_i, u_i) - \bar{y}_i\|_2^2 + \|F_N(x_N)\|_2^2$$

$$\text{subject to} \quad G_{\mathrm{eq}}(\cdot) = \begin{bmatrix} x_0 - \bar{x}_0 \\ x_1 - \phi_0(x_0, u_0) \\ \vdots \end{bmatrix} = 0$$

$$G_{\mathrm{ineq}}(\cdot) = \begin{bmatrix} h_0(x_0, u_0) \\ \vdots \\ r(x_0, x_N) \end{bmatrix} \leq 0$$

Sequential Quadratic Programming (SQP - RTI)

$$\underset{X,\,U}{\text{minimize}} \quad \Phi_{\mathrm{quad}}(X, U; X^{[k]}, U^{[k]}, Y^{[k]}, \lambda^{[k]})$$

$$\text{subject to} \quad G_{\mathrm{eq,lin}}(\cdot) = \begin{bmatrix} x_0 - \bar{x}_0 \\ x_1 - \phi_0(x_0^{[k]}, u_0^{[k]}) - \begin{bmatrix} A_0^{[k]} & B_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \end{bmatrix} = 0$$

$$G_{\mathrm{ineq,lin}}(\cdot) = \begin{bmatrix} h_0(x_0^{[k]}, u_0^{[k]}) + \begin{bmatrix} C_0^{[k]} & D_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \\ r(\bar{x}_0, x_N^{[k]}) + C_N^{[k]}(x_N - x_N^{[k]}) \end{bmatrix} \leq 0$$

Sequential Quadratic Programming (SQP - RTI)

$$\underset{X,\,U}{\text{minimize}} \quad \Phi_{\mathrm{quad}}(X, U; X^{[k]}, U^{[k]}, Y^{[k]}, \lambda^{[k]})$$

$$\text{subject to} \quad G_{\mathrm{eq,lin}}(\cdot) = \begin{bmatrix} x_0 - \bar{x}_0 \\ x_1 - \phi_0(x_0^{[k]}, u_0^{[k]}) - \begin{bmatrix} A_0^{[k]} & B_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \end{bmatrix} = 0$$

$$G_{\mathrm{ineq,lin}}(\cdot) = \begin{bmatrix} h_0(x_0^{[k]}, u_0^{[k]}) + \begin{bmatrix} C_0^{[k]} & D_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \\ r(\bar{x}_0, x_N^{[k]}) + C_N^{[k]}(x_N - x_N^{[k]}) \end{bmatrix} \leq 0$$

Sequential Quadratic Programming (SQP - RTI)

$$\underset{X,\,U}{\text{minimize}} \quad \Phi_{\text{quad}}(X, U; X^{[k]}, U^{[k]}, Y^{[k]}, \lambda^{[k]})$$

$$\text{subject to} \quad G_{\text{eq,lin}}(\cdot) = \begin{bmatrix} x_0 - \bar{x}_0 \\ x_1 - \phi_0(x_0^{[k]}, u_0^{[k]}) - \begin{bmatrix} A_0^{[k]}, & B_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \end{bmatrix} = 0$$

$$G_{\text{ineq,lin}}(\cdot) = \begin{bmatrix} h_0(x_0^{[k]}, u_0^{[k]}) + \begin{bmatrix} C_0^{[k]}, & D_0^{[k]} \end{bmatrix} \begin{bmatrix} x_0 - x_0^{[k]} \\ u_0 - u_0^{[k]} \end{bmatrix} \\ \vdots \\ r(\bar{x}_0, x_N^{[k]}) + C_N^{[k]}(x_N - x_N^{[k]}) \end{bmatrix} \leq 0$$

Objective quadratic subproblem

- Gauss-Newton: easy, convex, fast
- Exact Hessian: $B_k = \nabla_W^2 \mathcal{L}(\cdot)$

How to solve the structured convex QP?

$$\min_{\Delta X, \Delta U} \sum_{i=0}^{N-1} \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix}^\top \begin{bmatrix} Q_i & S_i \\ S_i^\top & R_i \end{bmatrix} \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix} + \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix}^\top \begin{bmatrix} q_i \\ r_i \end{bmatrix} + x_N^\top Q_N x_N + x_N^\top q_N$$

$$\text{s.t.} \quad G_{\mathrm{eq,lin}}(\cdot) = \begin{bmatrix} \Delta x_0 - d_0 \\ \Delta x_1 - d_1 - [A_0, B_0] \begin{bmatrix} \Delta x_0 \\ \Delta u_0 \end{bmatrix} \\ \vdots \end{bmatrix} = 0$$

$$G_{\mathrm{ineq,lin}}(\cdot) = \begin{bmatrix} c_0 + [C_0, D_0] \begin{bmatrix} \Delta x_0 \\ \Delta u_0 \end{bmatrix} \\ \vdots \\ c_N + C_N \Delta x_N \end{bmatrix} \leq 0$$

How to solve the structured convex QP?

$$\min_{\Delta X, \Delta U} \sum_{i=0}^{N-1} \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix}^\top \begin{bmatrix} Q_i & S_i \\ S_i^\top & R_i \end{bmatrix} \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix} + \begin{bmatrix} \Delta x_i \\ \Delta u_i \end{bmatrix}^\top \begin{bmatrix} q_i \\ r_i \end{bmatrix} + x_N^\top Q_N x_N + x_N^\top q_N$$

$$\text{s.t.} \quad G_{\mathrm{eq,lin}}(\cdot) = \begin{bmatrix} \Delta x_0 - d_0 \\ \Delta x_1 - d_1 - [A_0, B_0] \begin{bmatrix} \Delta x_0 \\ \Delta u_0 \end{bmatrix} \\ \vdots \end{bmatrix} = 0$$

$$G_{\mathrm{ineq,lin}}(\cdot) = \begin{bmatrix} c_0 + [C_0, D_0] \begin{bmatrix} \Delta x_0 \\ \Delta u_0 \end{bmatrix} \\ \vdots \\ c_N + C_N \Delta x_N \end{bmatrix} \leq 0$$

structure exploiting, embedded convex solvers:
FORCES, *qpDUNES*, *HPMPC*, ...

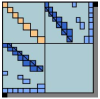
How to solve the structured convex QP?

structure exploiting, embedded convex solvers:

OR condensing, $O(N^2)$ complexity

$$
\begin{aligned}
\underset{x_0,u_0,\ldots,x_N}{\text{minimize}} \quad & \frac{1}{2}\sum_{k=0}^{N-1} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} Q_k & S_k \\ S_k^T & R_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} + \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} g_k^{\mathrm{x}} \\ g_k^{\mathrm{u}} \end{bmatrix}^T \\
& + \tfrac{1}{2} x_N^T Q_{\mathrm{e}} x_N + x_N^T g_{\mathrm{e}}^{\mathrm{x}} \\
\text{subject to} \quad & x_{k+1} = A_k x_k + B_k u_k + c_k, \quad \text{for} \quad k = 0,\ldots,N-1 \\
& x_k^{\mathrm{lo}} \leq x_k \leq x_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N, \\
& u_k^{\mathrm{lo}} \leq u_k \leq u_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N-1, \\
& b_k^{\mathrm{lo}} \leq C_k x_k + D_k u_k \leq b_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N-1, \\
& b_{\mathrm{e}}^{\mathrm{lo}} \leq C_{\mathrm{e}} x_N \leq b_{\mathrm{e}}^{\mathrm{up}},
\end{aligned}
$$

$$
\begin{aligned}
\underset{u}{\text{minimize}} \quad & \frac{1}{2} u^T H_{\mathrm{C}} u + u^T g_{\mathrm{C}} \\
\text{subject to} \quad & u^{\mathrm{lo}} \leq u \leq u^{\mathrm{up}} \\
& b_{\mathrm{C}}^{\mathrm{lo}} \leq A_{\mathrm{C}} u \leq b_{\mathrm{C}}^{\mathrm{up}}
\end{aligned}
$$

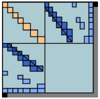


How to solve the structured convex QP?

structure exploiting, embedded convex solvers:

OR condensing, $O(N^2)$ complexity

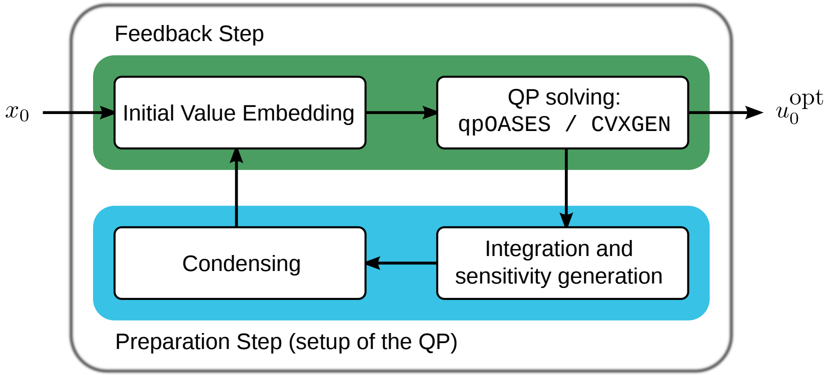
$$
\begin{aligned}
\underset{x_0,u_0,\ldots,x_N}{\text{minimize}} \quad & \frac{1}{2}\sum_{k=0}^{N-1}\begin{bmatrix}x_k\\u_k\end{bmatrix}^T\begin{bmatrix}Q_k & S_k\\S_k^T & R_k\end{bmatrix}\begin{bmatrix}x_k\\u_k\end{bmatrix}+\begin{bmatrix}x_k\\u_k\end{bmatrix}^T\begin{bmatrix}g_k^{\mathrm{x}}\\g_k^{\mathrm{u}}\end{bmatrix}^T \\
& +\tfrac{1}{2}x_N^T Q_{\mathrm{e}} x_N + x_N^T g_{\mathrm{e}}^{\mathrm{x}} \\
\text{subject to} \quad & x_{k+1} = A_k x_k + B_k u_k + c_k, \quad \text{for} \quad k = 0,\ldots,N-1 \\
& x_k^{\mathrm{lo}} \le x_k \le x_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N, \\
& u_k^{\mathrm{lo}} \le u_k \le u_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N-1, \\
& b_k^{\mathrm{lo}} \le C_k x_k + D_k u_k \le b_k^{\mathrm{up}}, \quad \text{for} \quad k = 0,\ldots,N-1, \\
& b_{\mathrm{e}}^{\mathrm{lo}} \le C_{\mathrm{e}} x_N \le b_{\mathrm{e}}^{\mathrm{up}},
\end{aligned}
$$



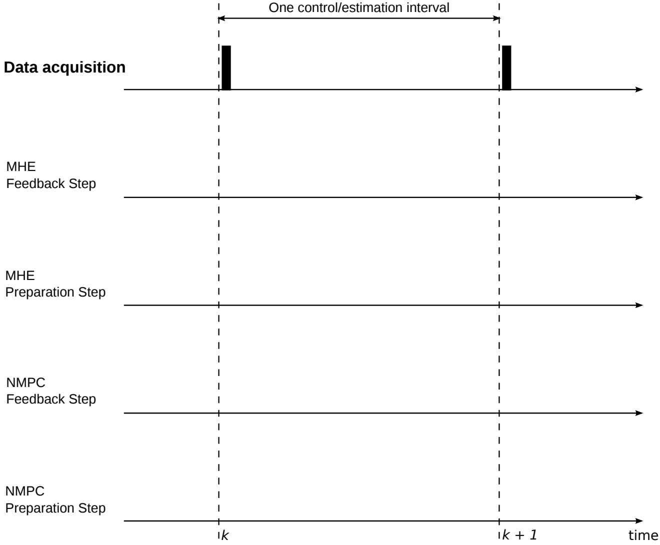
$$
\begin{aligned}
\underset{u}{\text{minimize}} \quad & \frac{1}{2}u^T H_{\mathrm{C}} u + u^T g_{\mathrm{C}} \\
\text{subject to} \quad & u^{\mathrm{lo}} \le u \le u^{\mathrm{up}} \\
& b_{\mathrm{C}}^{\mathrm{lo}} \le A_{\mathrm{C}} u \le b_{\mathrm{C}}^{\mathrm{up}}
\end{aligned}
$$

$\longrightarrow$ solve the condensed QP with a dense linear algebra QP solver, e.g. qpOASES, www.qpoases.org

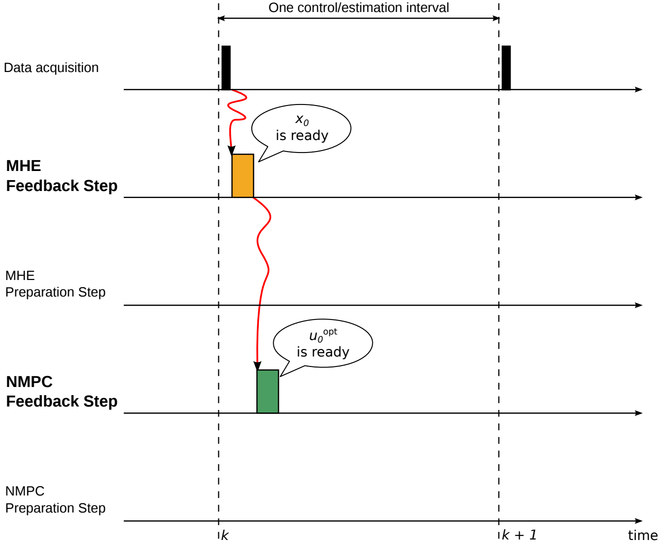
The RTI workflow for fast NMPC



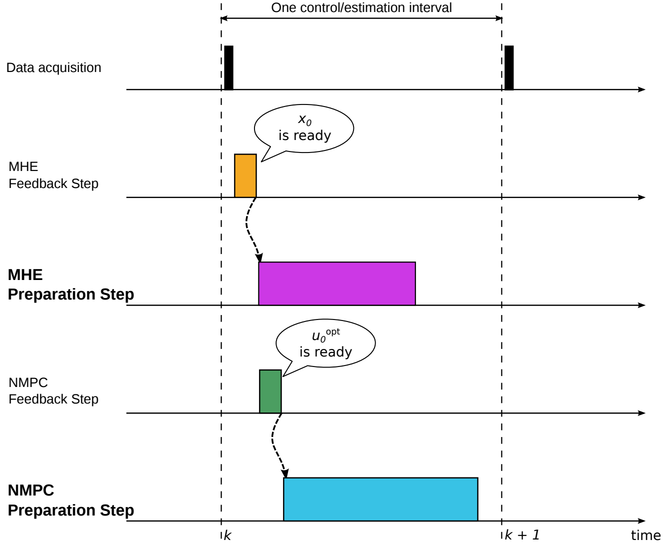
The RTI workflow for fast NMPC (& MHE)



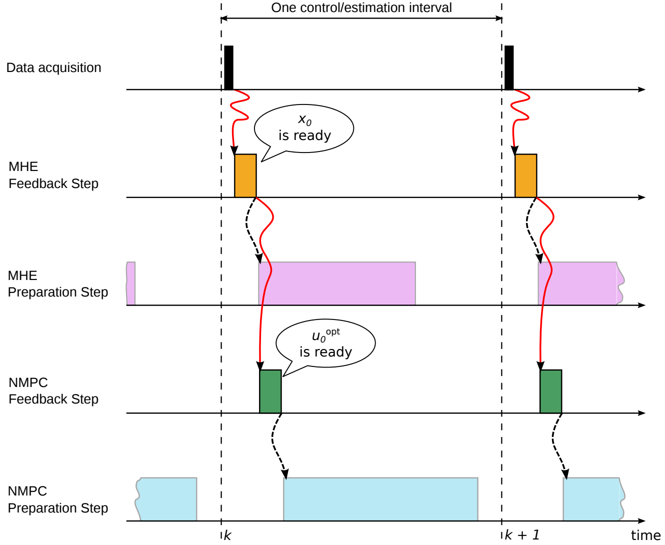
The RTI workflow for fast NMPC (& MHE)



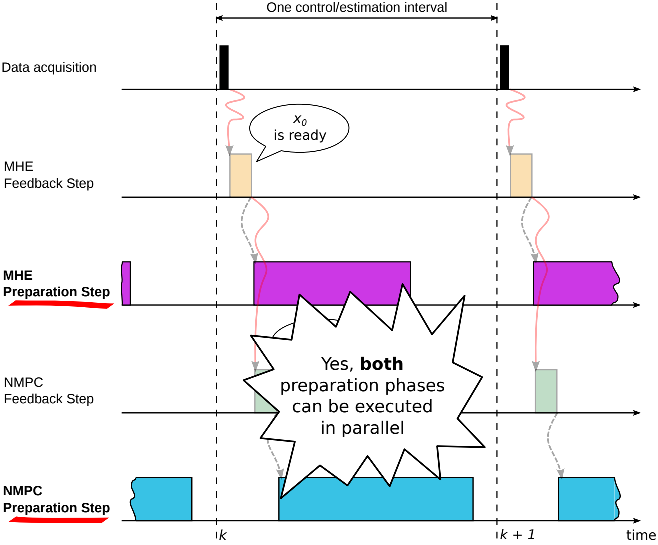
The RTI workflow for fast NMPC (& MHE)



The RTI workflow for fast NMPC (& MHE)



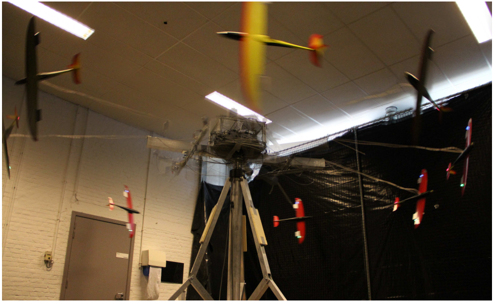
The RTI workflow for fast NMPC (& MHE)



Outline

ERC HIGHWIND project [1]

MHE and NMPC implementation on an experimental test set-up for launch/recovery of an airborne wind energy (AWE) system [Geebelen, 2013], located at KU Leuven (new carousel in Freiburg).



[1] Joint work: A. Wagner, M. Vukov, M. Zanon, K. Geebelen

ERC HIGHWIND project

Problem specific info

- Nonlinear dynamics: 22 states and 3 inputs
- Nonlinear measurement functions (for camera and IMU)
- Sensors:
  - Camera measurements 12 data @ 10 Hz with delay
  - IMU measurements 6 data @ 500 Hz
  - encoder measurements 2 data @ 10 Hz
- Sampling frequency: 10 Hz

ERC HIGHWIND project

Timing results: MHE & NMPC

		Average	Worst case
MHE	Preparation phase	3.76 ms	3.76 ms
	Estimation phase	0.75 ms	0.78 ms
	Overall execution time	4.51 ms	**4.54 ms**
MPC	Preparation phase	3.56 ms	3.56 ms
	Feedback phase	0.50 ms	0.61 ms
	Overall execution time	4.06 ms	**4.17 ms**

MHE applied on an induction motor [Frick, 2012] [2]

Dynamic system properties:

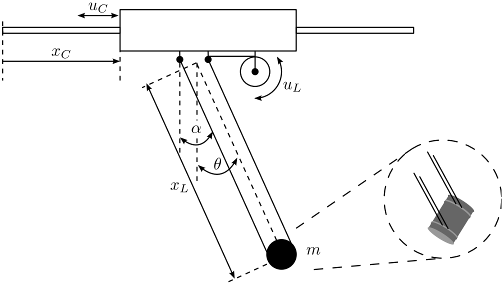
- 5 states, 2 controls
- 6 estimation intervals
- sampling freq.: **1.5 kHz**

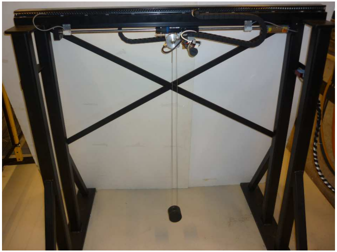
Execution times:

- one RTI on a 3 GHz Intel CPU: **30 μs** (double precision)
- one RTI on a 1 GHz TI low power DSP: **270 μs** (single precision)

[2] Joint work with ETH Zürich (D. Frick, A. Domahidi, S. Mariethoz, M. Morari)

Overhead crane [Debrouwere, 2014]





Overhead crane [3]

linear input	$\rightarrow$	nonlinear	$\rightarrow$	linear output
6		2		0

	unstructured	structured
integration method	**220 µs**	**67 µs**
condensing	6 µs	6 µs
QP solution (qpOASES)	16 µs	16 µs
remaining operations	3 µs	3 µs
one real-time iteration	245 µs	92 µs

Table : $T = 1.0$ s, $N = 10$ and 4^{th} order Gauss method ($h = 0.025$ s)

[3] Intel i7-3720QM 6MB cache, 2.60 GHz

Overhead crane [3]

linear input	$\rightarrow$	nonlinear	$\rightarrow$	linear output
6		2		0

	unstructured	structured
integration method	**220 µs**	**67 µs**
condensing	6 µs	6 µs
QP solution (qpOASES)	16 µs	16 µs
remaining operations	3 µs	3 µs
one real-time iteration	245 µs	92 µs

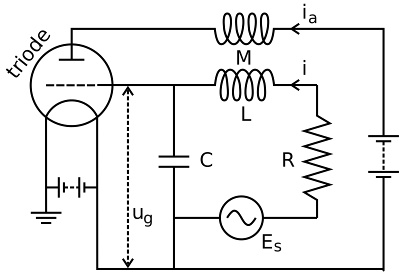
Table : $T = 1.0$ s, $N = 10$ and 4^{th} order Gauss method ($h = 0.025$ s)

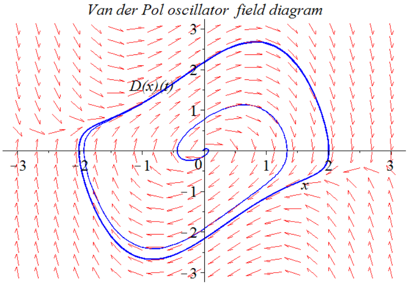
$\Rightarrow$ **integration speedup factor $\sim$ 3**

[3] Intel i7-3720QM 6MB cache, 2.60 GHz

Outline

Let's control the Van der Pol oscillator





Van der Pol oscillator x,x' phase diagram for epsilon=α = 1

$$\mathrm{dot}(x_1) = (1 - x_2^2)x_1 - x_2 + u$$
$$\mathrm{dot}(x_2) = x_1$$

Thank you for your attention!

Questions?