Real-time Optimization for Distributed Model Predictive Control

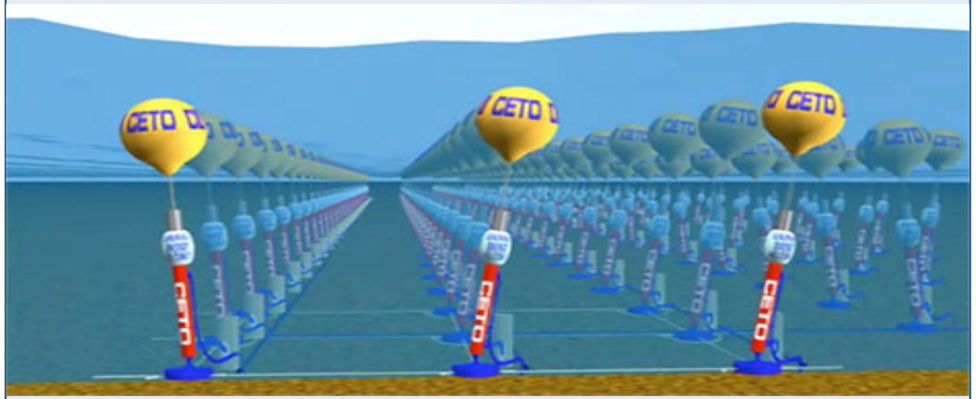
Manfred Morari & Colin Jones

Christian Conte, Davide Raimondo,
Stefan Richter, Sean Summers,
Joe Warrington, Melanie Zeilinger

Automatic Control Laboratory, ETH Zürich

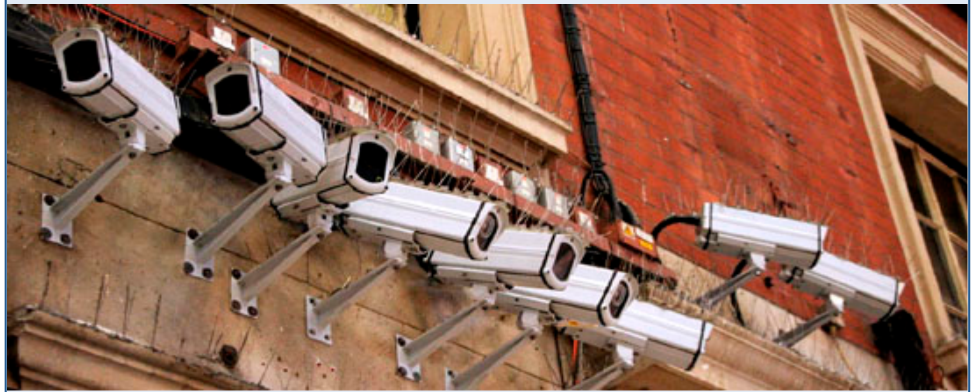
Distributed MPC : Motivating Examples

Wave power

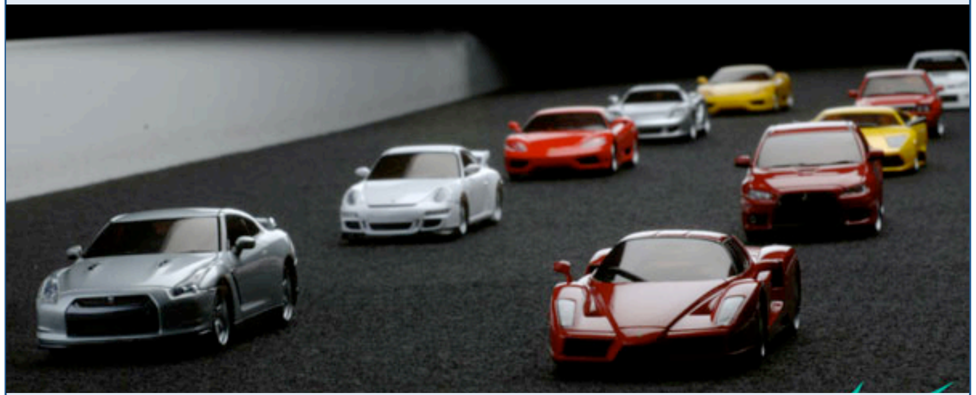


Picture from Carnegie

Camera systems



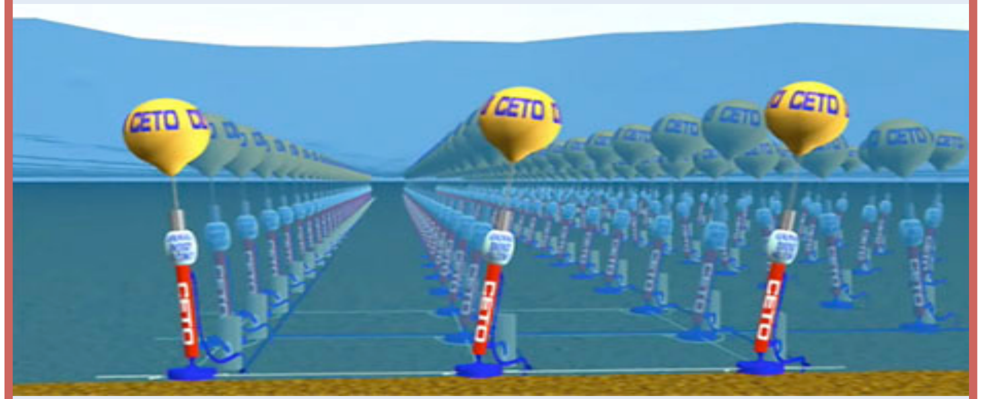
Racing



Power grids

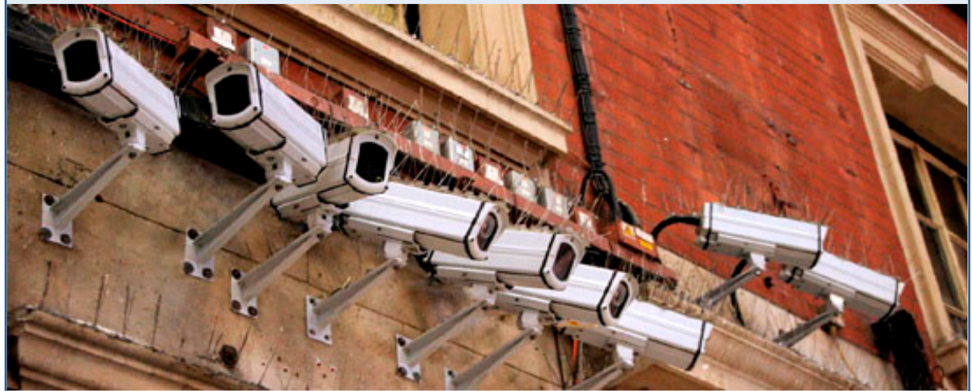
Distributed MPC : Motivating Examples

Wave power



Picture from Carnegie

Camera systems



Racing

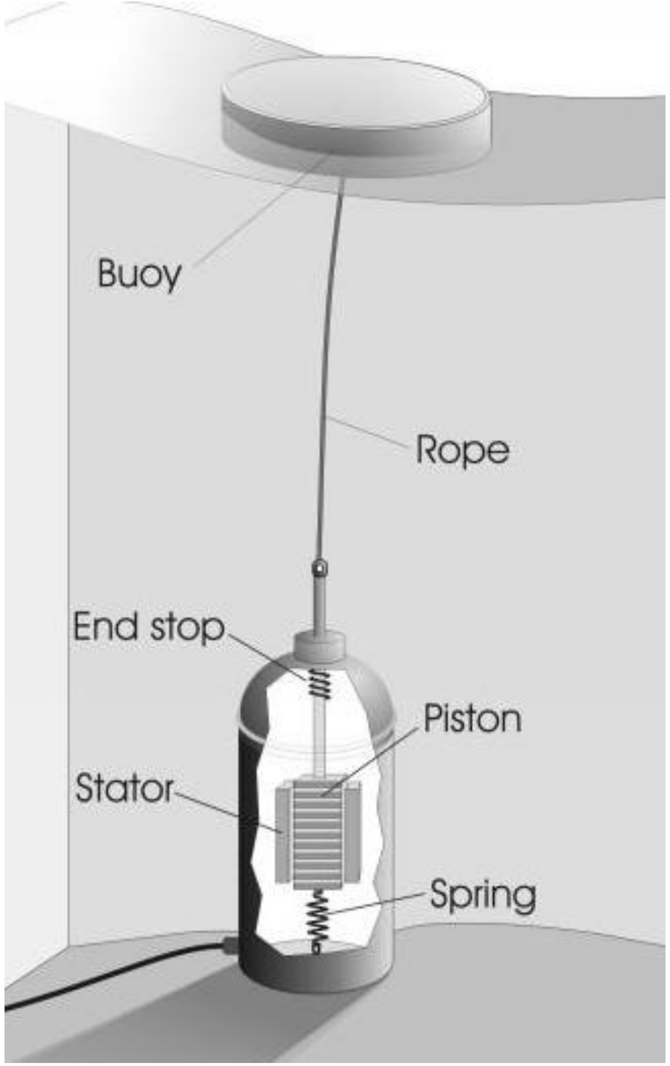
Power grids

Wave power: The heaving buoy

- ~1MW per meter of wave crest[1]
  - Energy density ~800x wind
- Global potential ~10 TW[2]
  - Exploitable > 2TW[3]
  - 20% world consumption[4]
- Floating buoy attached to generator on seabed
  - Heaving motion $\Rightarrow$ Electrical energy
  - System dynamics $\Rightarrow$ ~Second order

$$M\ddot{x} = F_w - ks - b\dot{x} - F_u$$

Wave impact (F_w) — Mechanical System (ks) — Generator (F_u)



Picture courtesy of Uppsala University

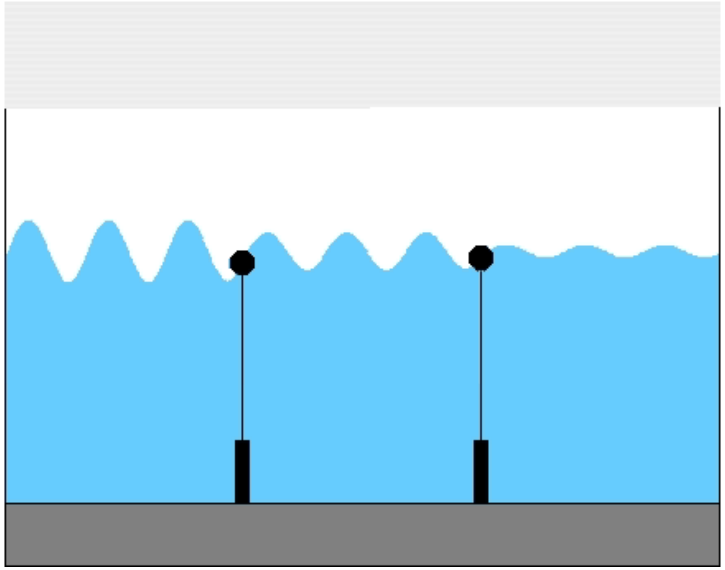
1. Survey of Energy Resources, WEC, 2007
2. Panicker, Power resource estimate of ocean surface waves (2003)
3. Thorpe, Wave Power: Moving towards Commercial Viability (1999)
4. BP statistical review of world energy (2008)

Wave farms are highly coupled

Combined cost function

- Maximize total energy

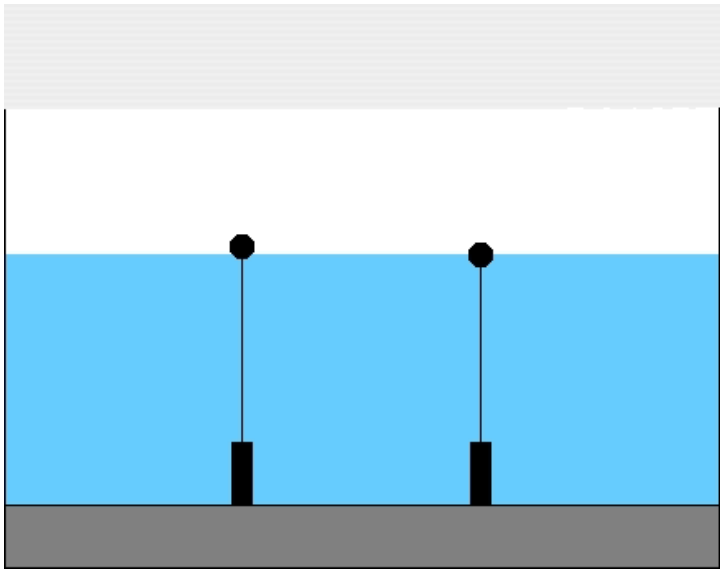
$$\max E_{\text{total}} := \sum_i \int_t \text{power}_i$$



Coupled dynamics

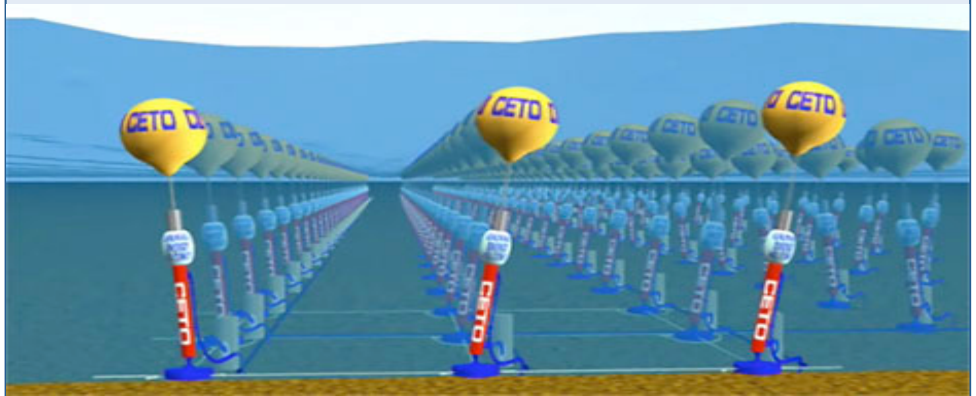
- Buoy causes a circular wave
- Perturbs motion of adjacent buoys

$$\dot{x}_i = f(x_1, \ldots, x_n, u_1, \ldots, u_n)$$



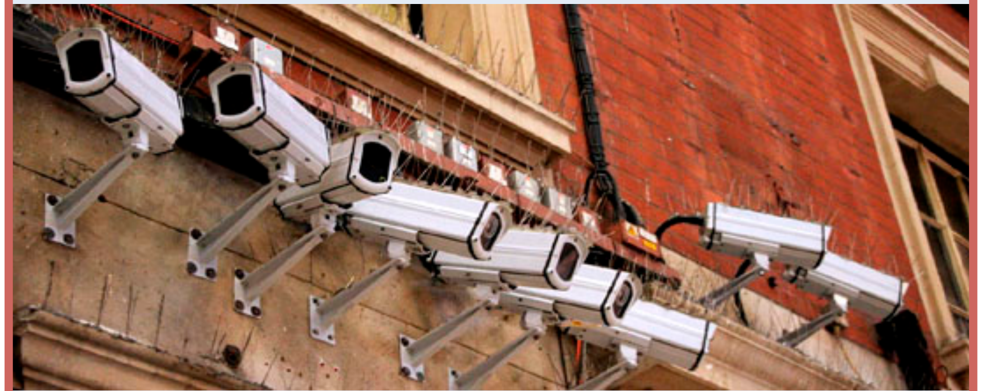
Distributed MPC : Motivating Examples

Wave power



Picture from Carnegie

Camera systems



Racing

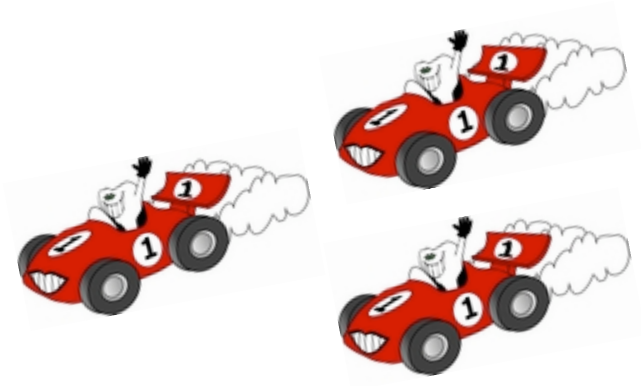
Power grids

Smart camera networks : Surveillance and motion capture

Goal: cooperatively detect and track human targets

- Unsupervised identification of camera network topology
- Distributed estimation of a relative mapping between adjacent cameras' field of views
- Optimal coverage of monitored site to search for anomalous events
- Moving object tracking with PTZ cameras and target hand-off

IfA Vision Lab



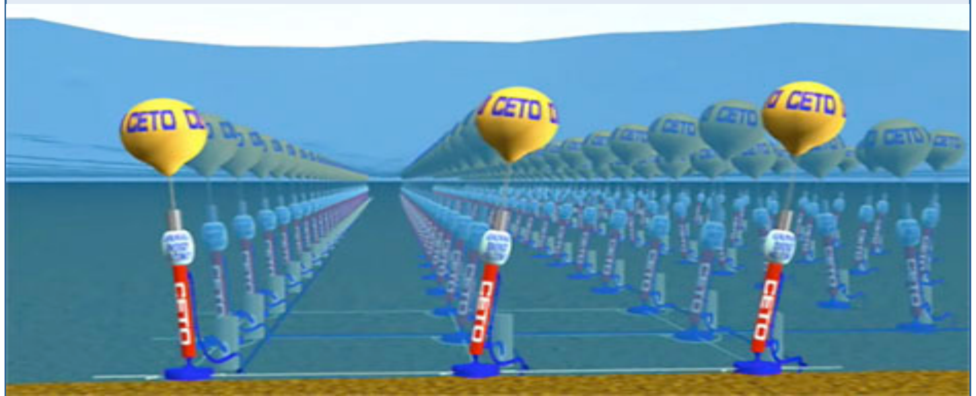
- Pan-tilt-zoom Ulisse Compact Cameras
- Support of Videotec S.p.A.

VIDEOTEC

FeedNetback co-design for networked control systems

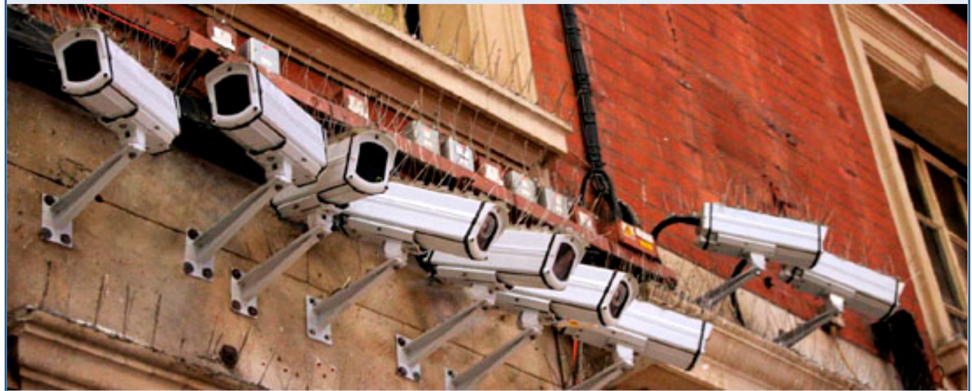
Distributed MPC : Motivating Examples

Wave power

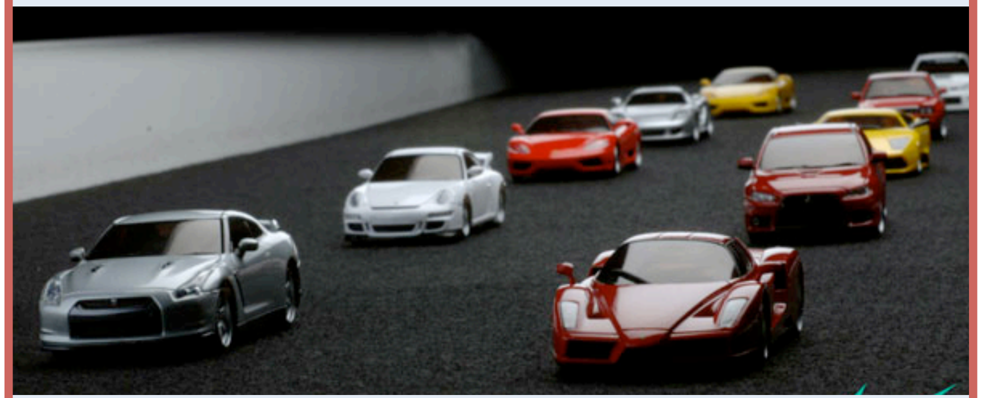


Picture from Carnegie

Camera systems



Racing



Power grids

Micro-scale Race Cars

- 1:43 scale cars – 106mm
- Top speed: 5 m/s
  (774 km/h scale speed)
- Full differential steering
- Position-sensing: External vision
- Sampling rate: 60Hz

Project goals:

1. Beat all human opponents!
2. Demonstrate real-time MPC maximizing car performance
3. Plan optimal path online in dynamic race environment

Challenges:

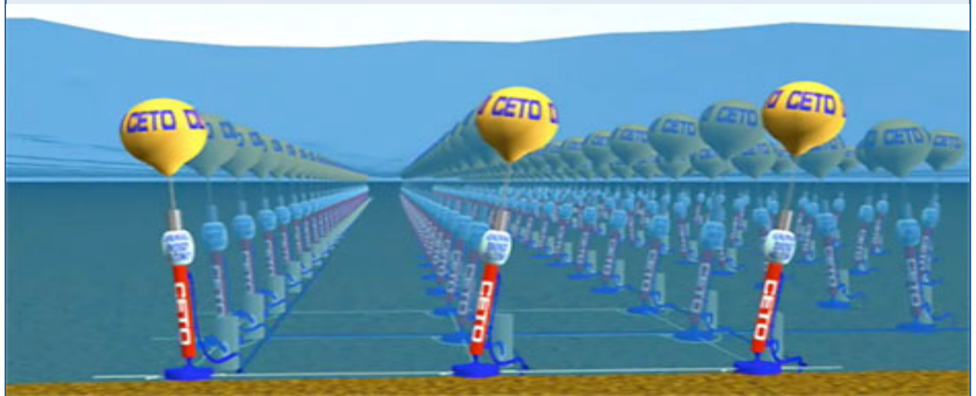
Highly nonlinear dynamics
Multiple unpredictable opponents
High-speed planning and control

Optimal Race Planning

[S. Colass, F. Engler, M. Osswald and C.N. Jones 2009]

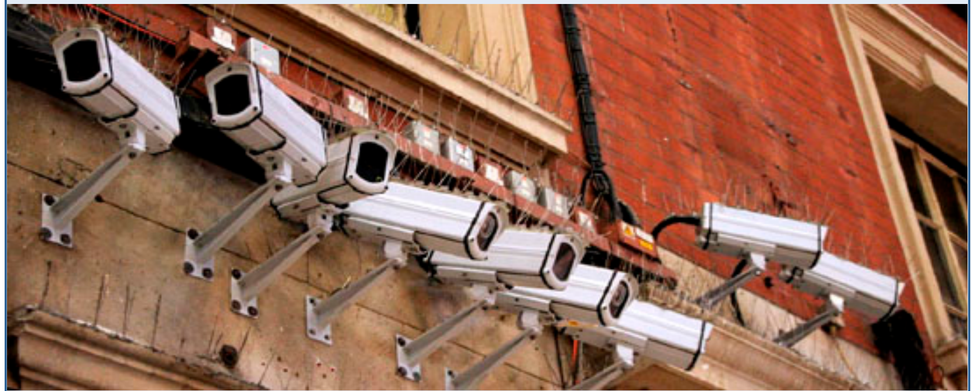
Distributed MPC : Motivating Examples

Wave power



Picture from Carnegie

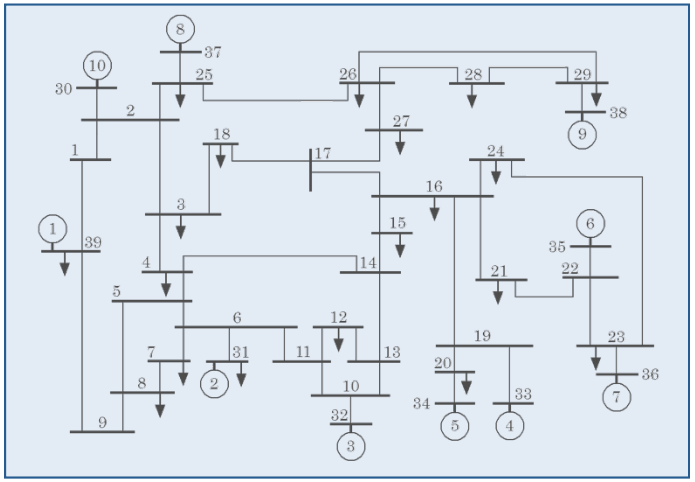
Camera systems



Racing

Power grids

Price Control of Power Grids



- Current grid:
  - Many loads, generators, transmission lines
  - Strongly coupled but with own objectives
- Market mechanisms break as renewables e.g., wind power share increases:
  - Flow schedule violates line limits
  - Failure to establish a clearing price

Goal: Minimize total generation cost, satisfy loads and line constraints

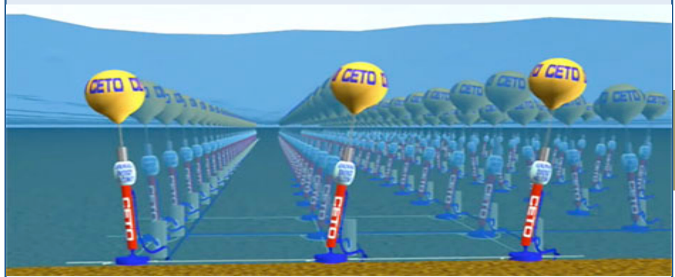
- Keep complex generation decisions *localized*:
  - Cost function of operating point, penalties for output changes, startup/shutdown events, capacity for ancillary services…

Idea: Distribute optimization and communicate via price signals

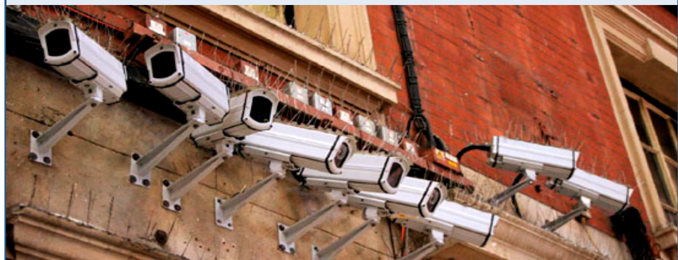
[J. Warrington and S. Mariethoz, 2009]
E-PRICE: Price-based Control of Electrical Power Systems

Distributed MPC Challenges

Coupled…



…inputs



…objectives

…constraints

…dynamics

Common Problem

Execute control action with objectives

- Stability
- Constraint satisfaction
- Performance guarantee
- Real time execution guarantee

Outline

Motivating examples

A key challenge : Fast, fixed-time optimization

- **Interior-point methods : Milli-seconds**
- Fast gradient methods : Micro-seconds
- Explicit methods : Nano-seconds

Summary

High-speed Model Predictive Control

$$J^*(x) = \min_{\mathbf{u}=[u_0,\ldots,u_{N-1}]} V_N(x,\mathbf{u}) \triangleq \frac{1}{2}x_N^T P x_N + \sum_{i=0}^{N-1} \frac{1}{2}x_i^T Q x_i + \frac{1}{2}u_i^T R u_i$$

$$\begin{aligned}
\text{s. t.} \quad & x_{i+1} && = Ax_i + Bu_i \ , && \text{linear nominal system} \\
& (x_i, u_i) && \in \mathbb{X} \times \mathbb{U} \ , && \text{polytopic constraints} \\
& x_N && \in \mathcal{X}_F \ , && \text{terminal set} \\
& x_0 && = x \ ,
\end{aligned}$$

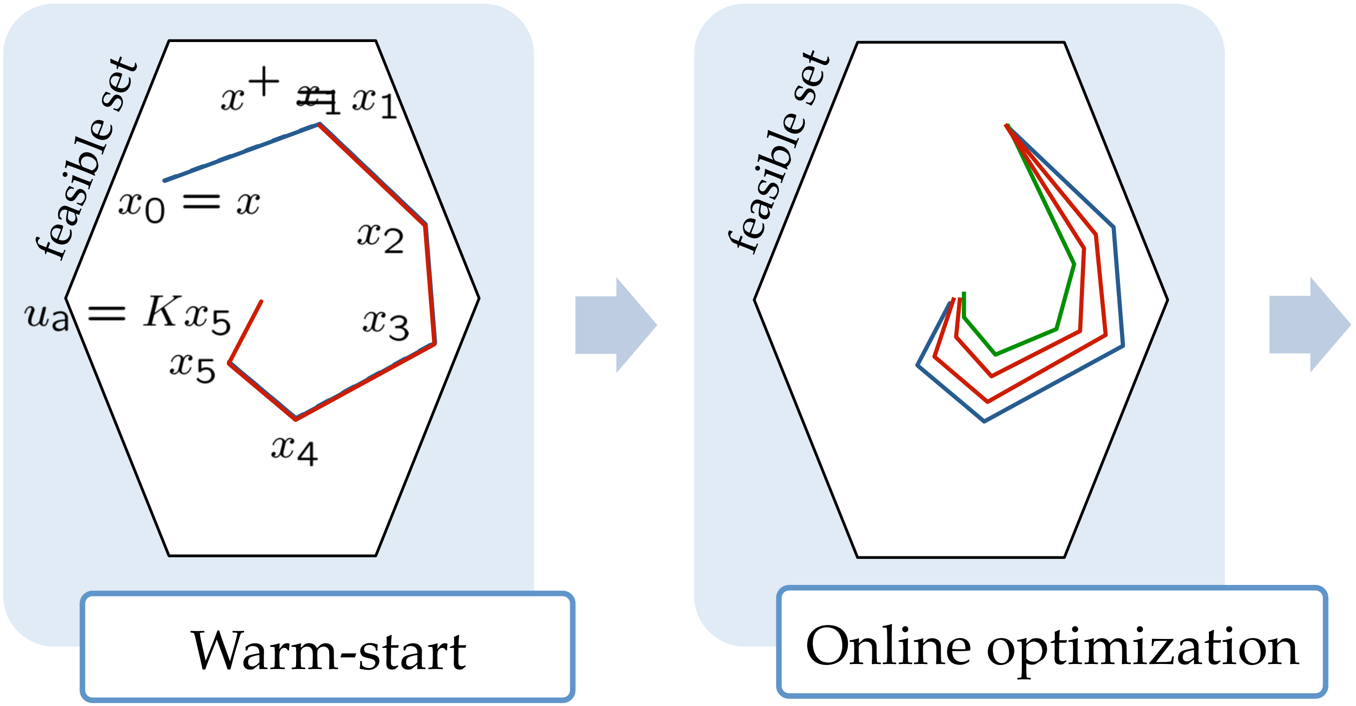
Optimal MPC controller:

- Input and state constraints are satisfied
  → Recursive feasibility
- $J^*(x)$ is a convex Lyapunov function
  → Stability of the closed-loop system

Goal: Feasibility/Stability/Tracking for suboptimal MPC controller with real-time constraint

[M.N. Zeilinger, C.N. Jones, D. M. Raimondo, M. Morari, CDC 2009]

Optimal MPC scheme (Not Real-time!)



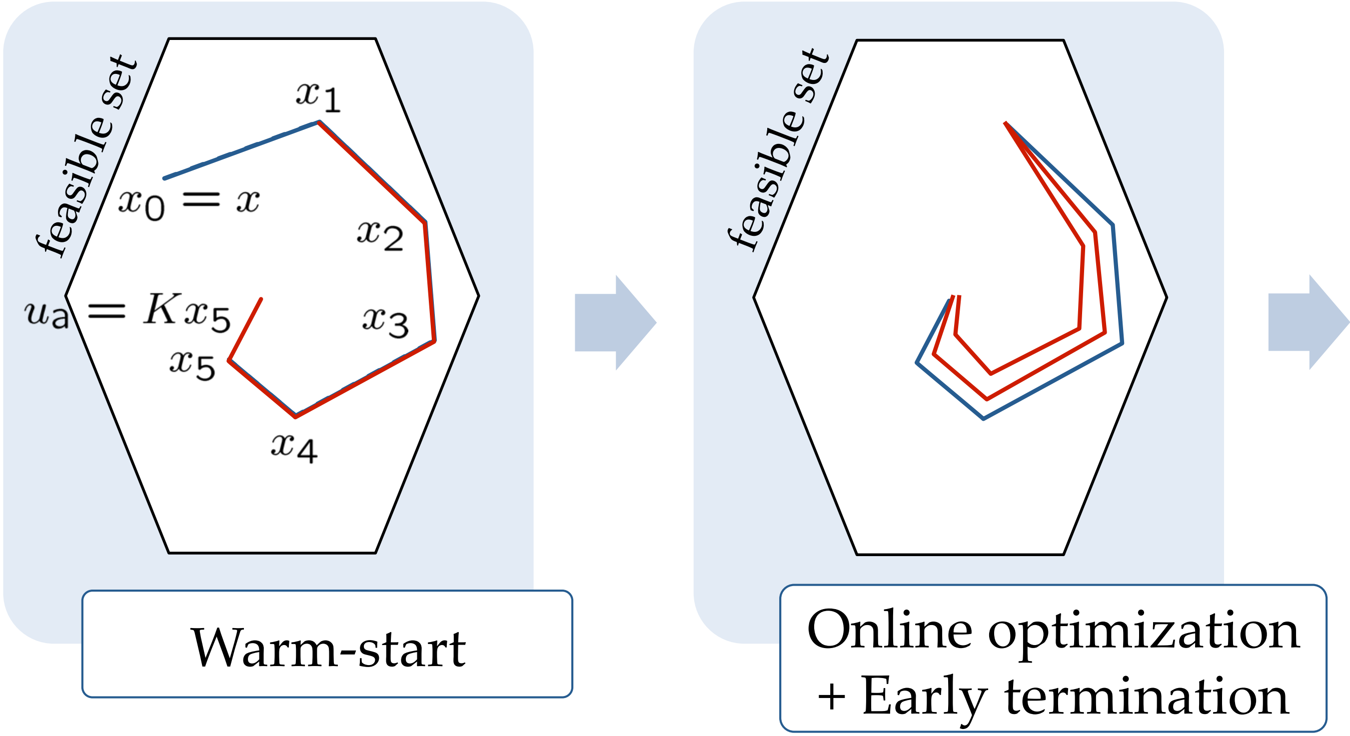
Optimal solution

→ Properties?

Optimal MPC:

- Recursively feasible
- Stabilizing
- Unknown computation time…

Real-time MPC scheme



Suboptimal solution

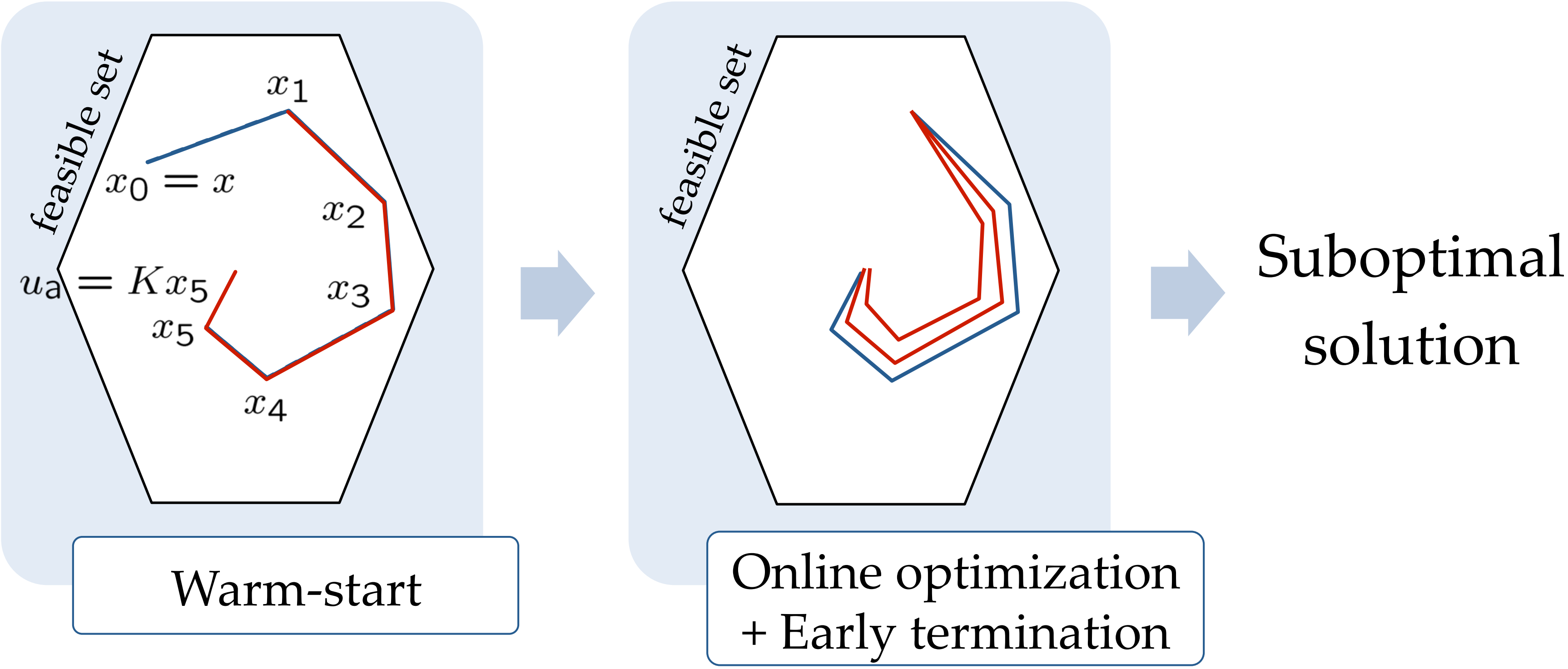
→ Properties?

General approach for real-time MPC:

- Use of warm-start method
- Exploitation of structure inherent in MPC problems
- Early termination of the online optimization

[Ferreau et al., 2008], [Wang et al., 2008],…

Real-time MPC scheme - Current methods



Suboptimal solution during online optimization steps

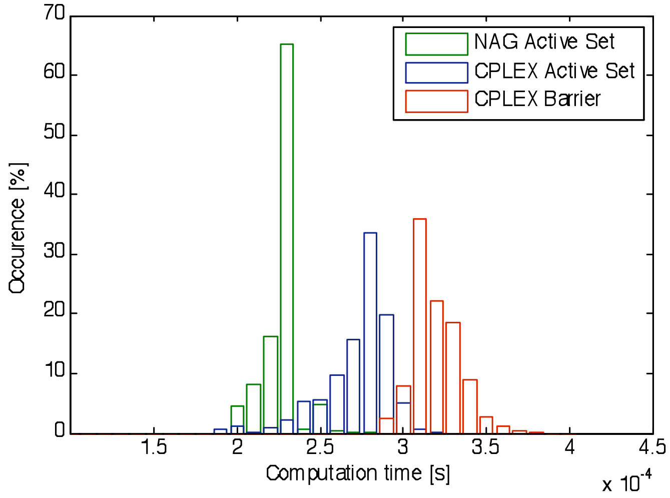
- can be infeasible
- can destabilize the system
- can cause steady-state offset

Online computation times for varying states

Example problem:

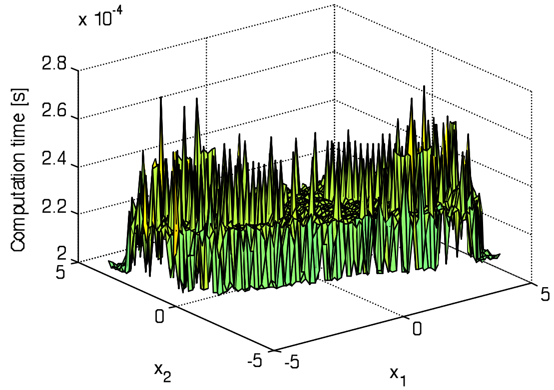
$$x^+ = \begin{bmatrix} 1.2 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u \ ,$$

$\|x\|_\infty \leq 5, \|u\|_\infty \leq 1,$
$N = 5, Q = I, R = 1$

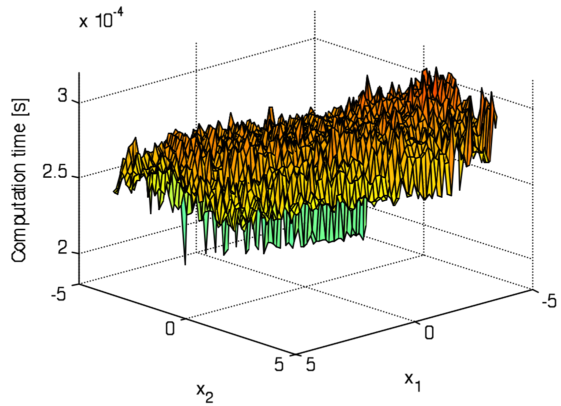


Executed on a 2.53GHz Intel Xeon running Linux;
Computation time is measured as time spent on C routine, averaged over 1000 calls of the optimizer for each sampled state.

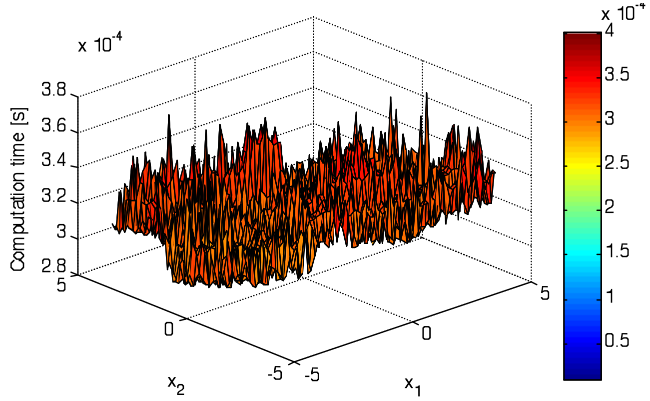
NAG – Active Set method



CPLEX – Active Set method



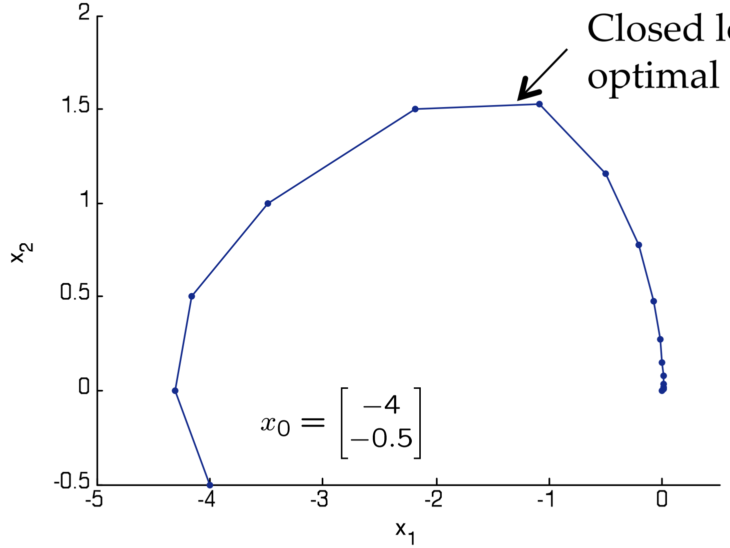
CPLEX – Barrier method



→ Computation times for solving the optimal MPC problem vary with the state of the system

Example: Effects of limited computation time

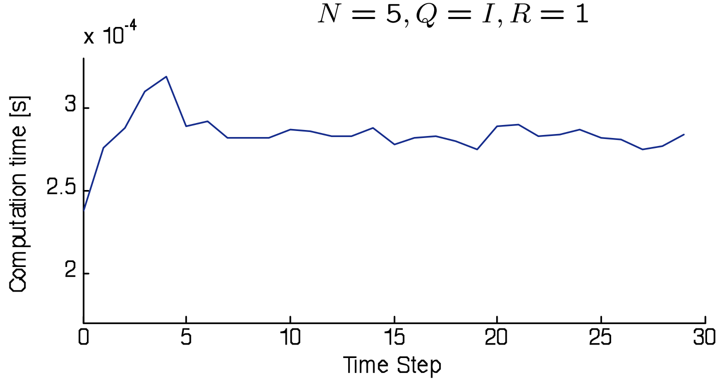
Solve MPC problem using CPLEX Active Set method



Example problem:

$$x^+ = \begin{bmatrix} 1.2 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u \ ,$$

$$\|x\|_\infty \leq 5, \|u\|_\infty \leq 1,$$
$$N = 5, Q = I, R = 1$$

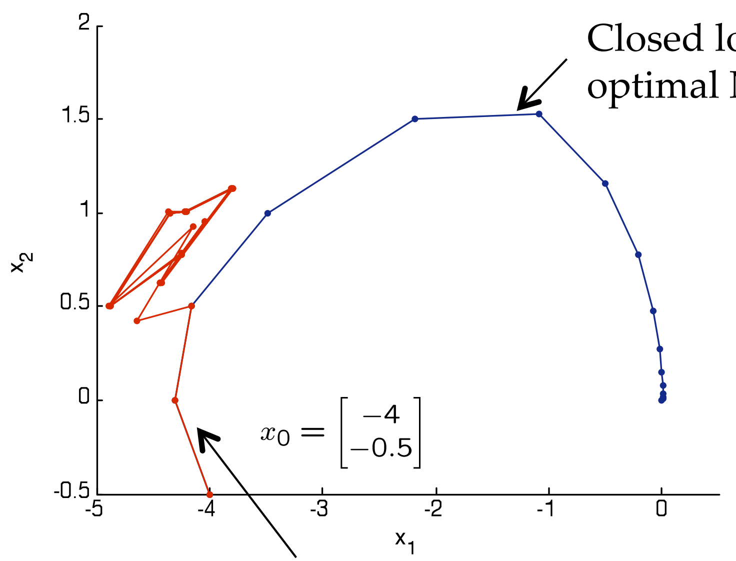


Now: Require computation time to be less than 27ms at every sampled state

→ Restrict algorithm to 5 online optimization steps

Example: Effects of limited computation time

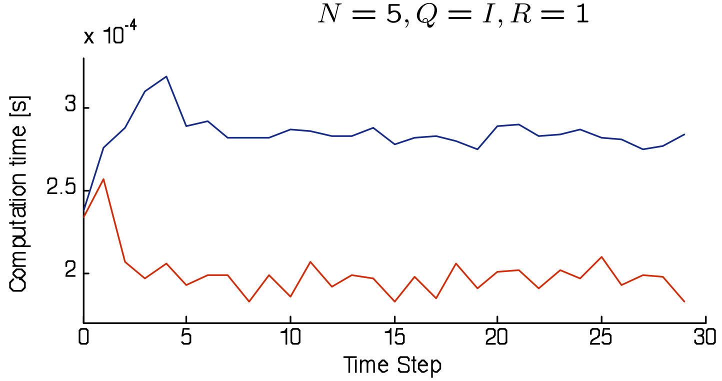
Solve MPC problem using CPLEX Active Set method



Example problem:

$$x^+ = \begin{bmatrix} 1.2 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u \ ,$$

$$\|x\|_\infty \leq 5, \|u\|_\infty \leq 1,$$
$$N = 5, Q = I, R = 1$$



→ System does not converge to the origin

Limits on the online computation time can destroy the stability properties of optimal MPC

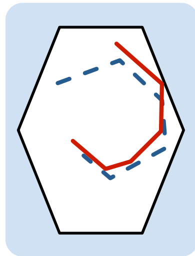
Real-time MPC
with stability and robustness guarantees

- Guarantees on
  - **Real-time ← Early termination**
  - Feasibility
  - Stability
  - Steady-state tracking
- Implementation for large-scale systems
- Fast implementation

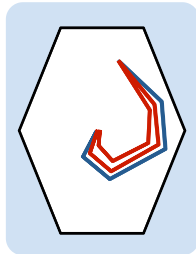
[M.N. Zeilinger, C.N. Jones, D. M. Raimondo, M. Morari, CDC 2009]

Real-time MPC method
- Constraint satisfaction

Consider uncertain system: $x^+ = Ax + Bu + w$
where $w \in W$ is a bounded disturbance .



- Robust MPC: Initial feasible solution for all disturbances
  e.g. [Limon et *al.*, 2009] and references therein



- Optimization maintains feasibility at all times

Here: Tube-based robust MPC: [Mayne et *al.*, 2005]

$$\min_{\{\bar{x}_0,\bar{\mathbf{u}}\}} \bar{V}_N(x,\bar{x}_0,\bar{\mathbf{u}}) \triangleq \frac{1}{2}\bar{x}_N^T P \bar{x}_N + \sum_{i=0}^{N-1} \frac{1}{2}\bar{x}_i^T Q \bar{x}_i + \frac{1}{2}\bar{u}_i^T R \bar{u}_i$$

$$\begin{aligned} \text{s.t.} \quad & \bar{x}_{i+1} && = A\bar{x}_i + B\bar{u}_i \ , \\ & (\bar{x}_i,\bar{u}_i) && \in \bar{\mathbb{X}} \times \bar{\mathbb{U}} \ , \qquad \bar{\mathbb{X}} = \mathbb{X} \ominus \mathcal{Z}, \bar{\mathbb{U}} = \mathbb{U} \ominus K\mathcal{Z} \\ & \bar{x}_N && \in X_f \ , \\ & x && \in \bar{x}_0 \oplus \mathcal{Z} \ , \end{aligned}$$

→ Ellipsoidal invariant sets can be computed for all system sizes
→ Resulting optimization problem is a convex QCQP

Real-time MPC with stability and robustness guarantees

- Guarantees on
  - Real-time ← Early termination
  - Feasibility ← Robust MPC formulation
  - **Stability ← Lyapunov constraint**
  - **Steady-state tracking ← Lyapunov constraint**
- Implementation for large-scale systems ← Convex QCQP
- Fast implementation

[M.N. Zeilinger, C.N. Jones, D. M. Raimondo, M. Morari, CDC 2009]

Real-time MPC - Fast Implementation

- Tracking formulation and Lyapunov constraint significantly modify structure of matrices in Newton step computation compared to literature. [Rao et *al*., 1998, Wang et *al*., 2008]

- Matrices can be transformed into arrow structure, which can be solved efficiently with same complexity as standard MPC problems [Rao et *al*.,1998; Hansson, 2000; Wang et *al*.,2008]

→ Fast solution of the tracking problem with guaranteed stability for all suboptimal iterates → for all time constraints!

- Custom solver in C++ was developed extending fast MPC solver described in literature [Wang et *al*., 2008]

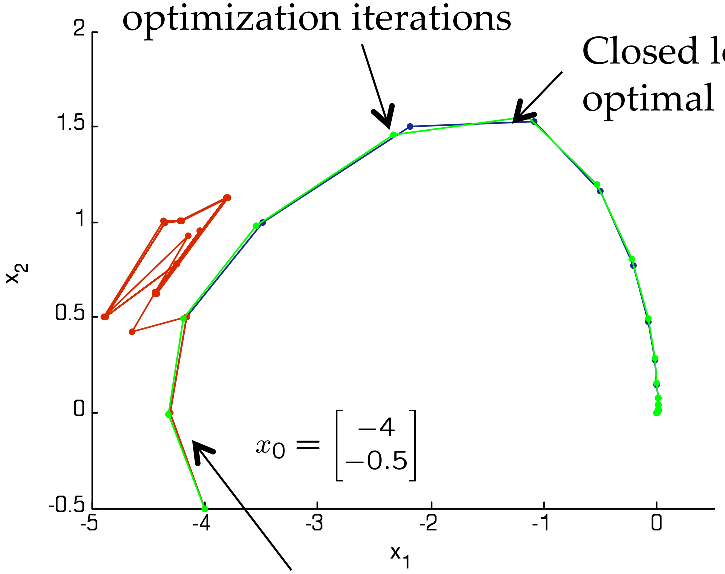
→ Computation times that are faster or equal compared to methods with no guarantees

[M.N. Zeilinger, C.N. Jones, D. M. Raimondo, M. Morari, CDC 2009]

Example: Effects of limited computation time

Closed loop trajectory using real-time MPC control law with Lyapunov constraint with a limit of 5 online optimization iterations

Closed loop trajectory using optimal MPC control law

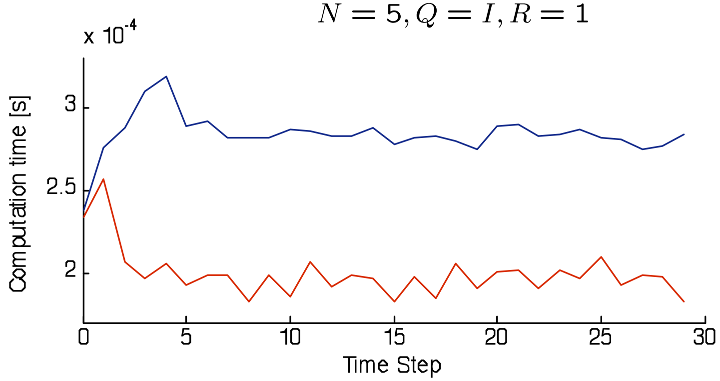


Closed loop trajectory using suboptimal MPC control law, with a limit of 5 online optimization iterations

Example problem:

$$x^+ = \begin{bmatrix} 1.2 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u \ ,$$

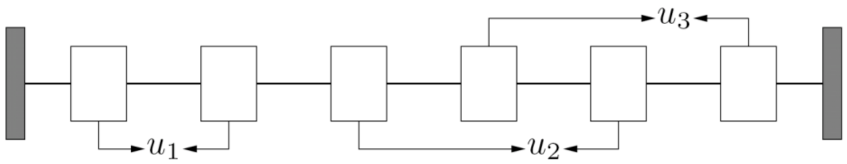
$$\|x\|_\infty \leq 5, \|u\|_\infty \leq 1,$$
$$N = 5, Q = I, R = 1$$



Limits on the online computation time can destroy the stability properties of optimal MPC

Numerical Examples

Oscillating masses example



- Problem: 12 states, 3 inputs
- Fast MPC with guarantees: horizon N=10

→ Computation of 5 Newton steps in **2 msec**

Comparison: CPLEX **26 msec**, SEDUMI **252 msec**

Closed loop performance loss in % for varying iteration numbers

$k_{\max}$	1	2	3	4	5	6	7	8
$\triangle J_{cl}$	1.39	1.32	1.10	0.88	0.70	0.55	0.44	0.33

→Optimal ~44 iterations

Random example

- Problem: 30 states, 8 inputs, horizon N=10

→ QCQP with 410 optimization variables and 1002 constraints

→ Computation of 5 Newton steps in **10 msec**

[M.N. Zeilinger, C.N. Jones, D. M. Raimondo, M. Morari, CDC 2009]

Outline

Motivating examples

A key challenge : Fast, fixed-time optimization

- Interior-point methods : Milli-seconds
- **Fast gradient methods : Micro-seconds**
- Explicit methods : Nano-seconds

Summary

Structured Optimization: Input constrained MPC

- Linear system, input constraints only
- Gradient-based optimization
  - Very simple
  - Easy to parallelize
  - Fast for large number of states

⇒ Can pre-compute required number of online iterations

Require: $U_0 \in \mathbb{U}^N$, $V_0 = U_0$

for $i = 1$ to $i_{\max}$ **do**

  $U_i = \pi_{\mathbb{U}^N}\left(V_{i-1} - \frac{1}{L}\nabla J_N(V_{i-1}; x)\right)$

  $V_i = U_i + b_i(U_i - U_{i-1})$

end for

- Work per iteration
  - 1 matrix-vector product
  - 2 vector sums
  - 1 projection (more later)

[Y. Nesterov, 1983]

[S. Richter, C.N. Jones and M. Morari, CDC 2009]

Fast Gradient Method for MPC

Observe:

Input-constrained MPC problem has a "simple" feasible set

$$\mathbb{U}^N := \mathbb{U} \times \mathbb{U} \times \ldots \times \mathbb{U}$$

→ Projection can be separated: $\pi_{\mathbb{U}^N}\left(\overline{U}\right) = \begin{bmatrix} \pi_{\mathbb{U}}\left(\overline{u}_0\right) \\ \pi_{\mathbb{U}}\left(\overline{u}_1\right) \\ \vdots \\ \pi_{\mathbb{U}}\left(\overline{u}_{N-1}\right) \end{bmatrix}$, where $\overline{U} = \begin{bmatrix} \overline{u}_0 \\ \overline{u}_1 \\ \vdots \\ \overline{u}_{N-1} \end{bmatrix}$

Missing Pieces

Require: $U_0 \in \mathbb{U}^N$, $V_0 = U_0$
for $i = 1$ to $i_{\max}$ **do**
$U_i = \pi_{\mathbb{U}^N}\left(V_{i-1} - \frac{1}{L}\nabla J_N(V_{i-1})\right)$
$V_i = U_i + b_i(U_i - U_{i-1})$
end for

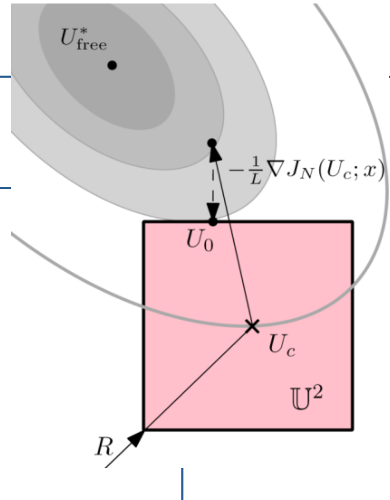
Intuition:
Choice of initial iterate influences number of iterations

Two Initialization Strategies ⇔ Two Different Lower Bounds on $i_{\max}$:

→ Cold-Starting

→ Warm-Starting

Main Complexity Results



Proposition (Cold-Starting)

If in the fast gradient method

→ the sequence U_c is the center of the feasible set $\mathbb{U}^N$, and

→ the initial iterate is given by $U_0 = \pi_{\mathbb{U}^N}\left(U_c - \frac{1}{L}\nabla J_N(U_c;x)\right)$,

an ϵ-solution is obtained after $i_{\max} \geq \left\lceil (\ln 2\epsilon - \ln LR^2)/\ln\left(1-\sqrt{\frac{1}{\kappa}}\right)\right\rceil$ iterations.

Proposition (Warm-Starting)

Assume an ϵ-solution $U_\epsilon = \left(u_{\epsilon,0}, u_{\epsilon,1}, \ldots, u_{\epsilon,N-1}\right)$ was obta[…]
previous time-step.

If in the fast gradient method the initial iterate is defined by

$$U_0 = \left(u_{\epsilon,1}, \ldots, u_{\epsilon,N-1}, u_N\right), \quad u_N \in \mathbb{U},$$

an ϵ-solution is obtained after $i_{\max} \geq \left\lceil (\ln \epsilon - \ln 2\delta)/\ln\left(1-\sqrt{\frac{1}{\kappa}}\right)\right\rceil$ iterations,

where $\delta = \max_{x\in\mathbb{X}_0} J_N(U_w;x) - J^*_N(x)$.

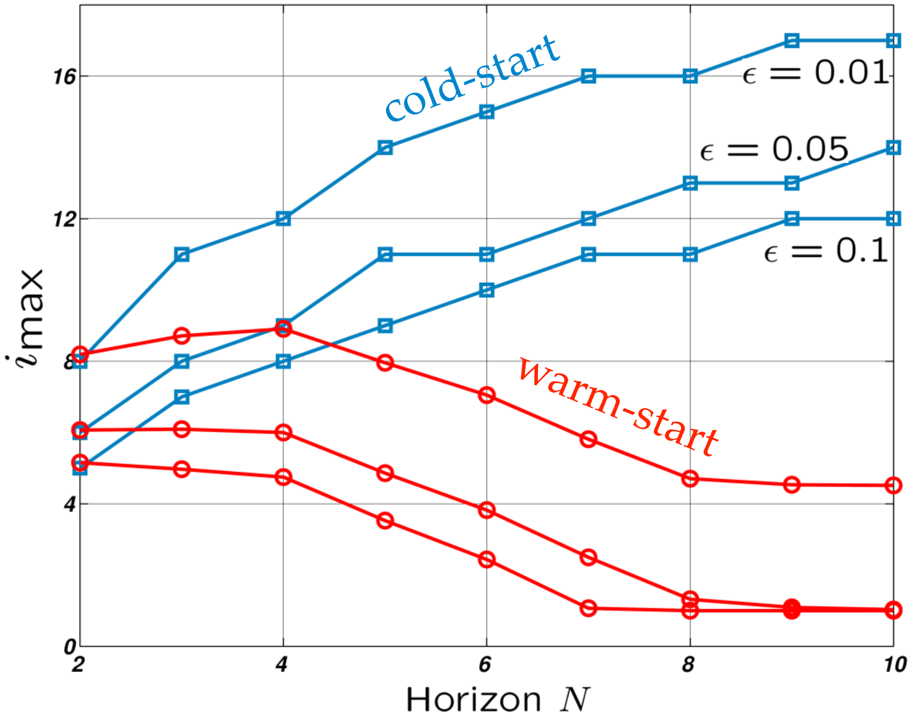
→ Bound depends on set of initial states
→ Hard to compute (*Bilevel Problem*) but can be recast as a Mixed Integer LP

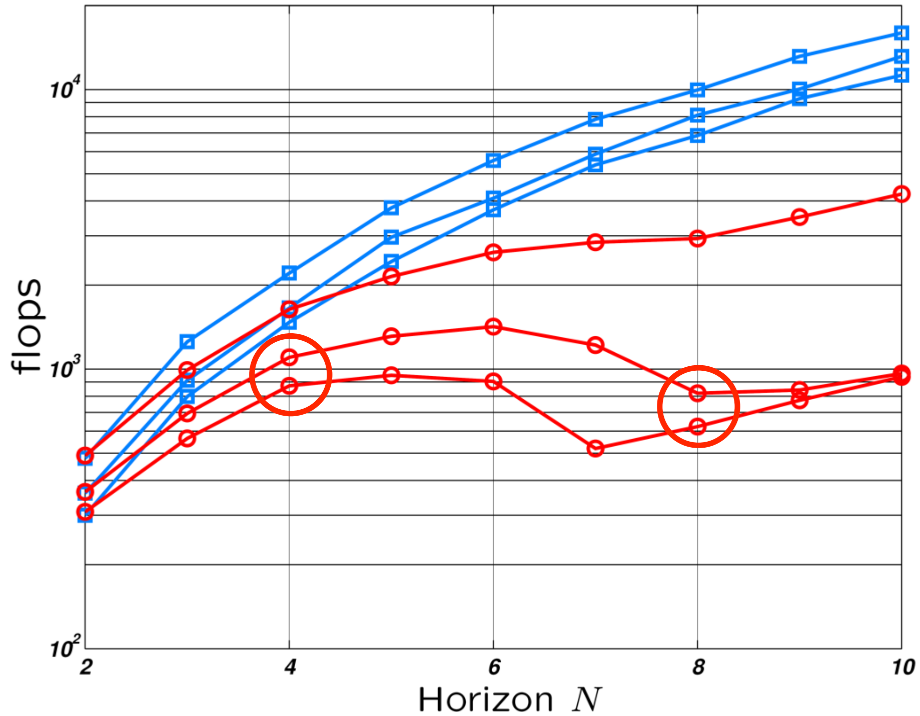
[S. Richter, C.N. Jones and M. Morari, CDC 2009]

Illustrative Example

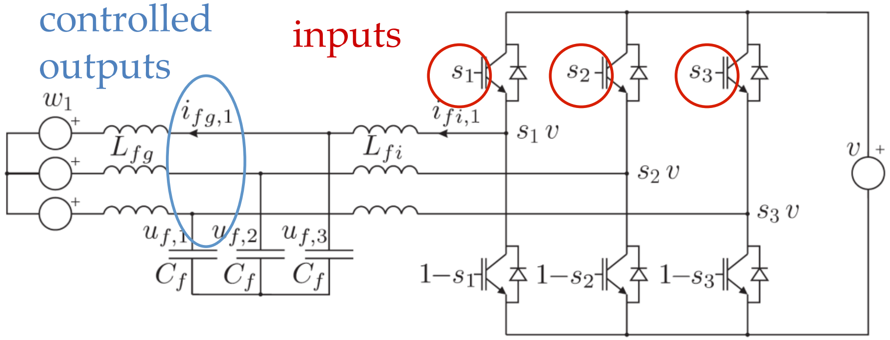
4 states/2 inputs system: $x^+ = \begin{bmatrix} 0.7 & -0.1 & 0.0 & 0.0 \\ 0.2 & -0.5 & 0.1 & 0.0 \\ 0.0 & 0.1 & 0.1 & 0.0 \\ 0.5 & 0.0 & 0.5 & 0.5 \end{bmatrix} x + \begin{bmatrix} 0.0 & 0.1 \\ 0.1 & 1.0 \\ 0.1 & 0.0 \\ 0.0 & 0.0 \end{bmatrix} u + w$

- → Set of initial states $\mathbb{X}_0 = \{x \mid \|x\|_\infty \leq 10\}$
- → Set of feasible inputs $\mathbb{U} = \{u \mid \|u\|_\infty \leq 1\}$
- → State disturbance $w \in \mathbb{W} = \{w \mid \|w\|_\infty \leq 0.25\}$
- → Weight matrices $Q = I_n, \quad R = 0.1 I_m$





Application to AC-DC Converter



Control of an AC-DC Converter

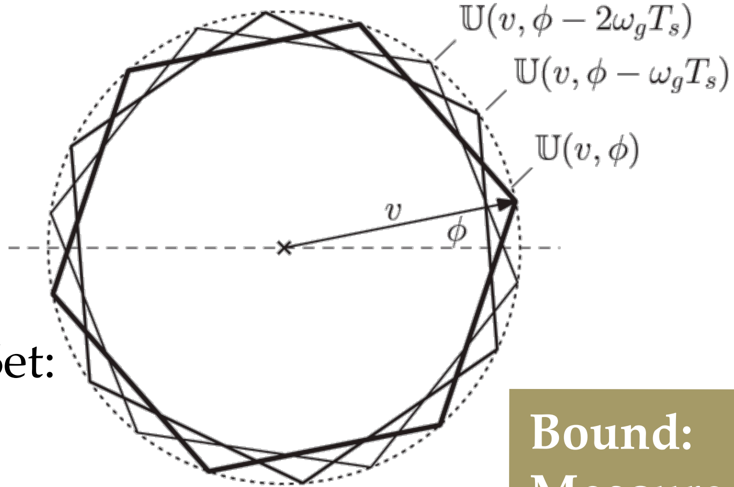
- Marginally stable system
  in d-q coordinates: 6 states / 2 inputs / 2 disturbances / 2 controlled outputs

- Reference tracking MPC

$$J_N^*(q) := \min \frac{1}{2}\|\delta x_N\|_{Q_N}^2 + \frac{1}{2}\sum_{k=0}^{N-1} \|\delta x_k\|_Q^2 + \|\delta u_k\|_R^2$$

$$\begin{aligned} \text{s.t.} \quad \delta x_k &= x_k - x_{ss} \ , \\ \delta u_k &= u_k - u_{ss} \ , \\ x_{k+1} &= Ax_k + Bu_k + B_w w \ , \\ u_k &\in \mathbb{U}(v, \phi + k\omega_g T_s) \ , \\ x_0 &= x \end{aligned}$$

- Rotating/Scaling Feasible Set:



- Implementation Platform:
  600 MHz DSP, 16-bit **fixed point** arithmetic

Bound:	**125 µs**
Measured:	**< 50 µs**
Memory:	**< 1KB**
Relative accuracy:	**< 1e-3**

[S. Richter, S. Mariéthoz and M. Morari, ACC 2010]

Outline

Motivating examples

A key challenge : Fast, fixed-time optimization

- Interior-point methods : Milli-seconds
- Fast gradient methods : Micro-seconds
- **Explicit methods : Nano-seconds**

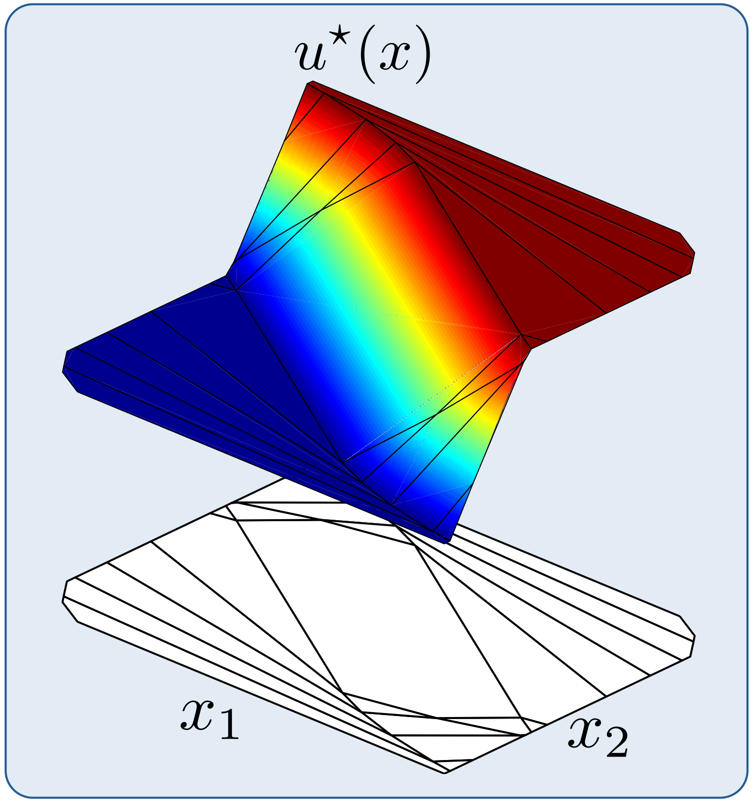
Summary

Explicit MPC : Online => Offline Processing

- Optimization problem is function parameterized by state
- Control law piecewise affine for PWA systems / constraints
- Pre-compute control law as function of state x

Result : Online computation dramatically reduced

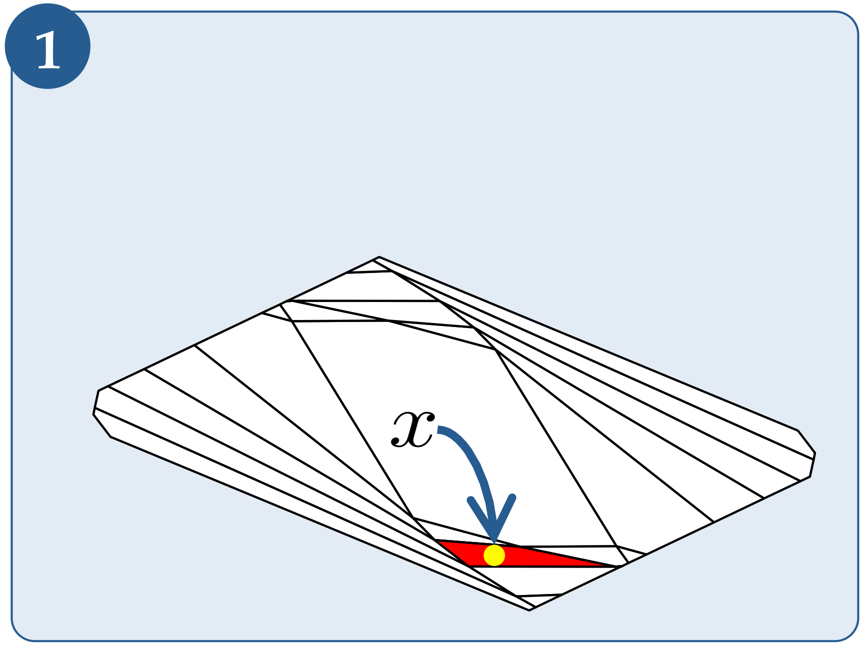
$$
\begin{aligned}
u^\star(x) = \underset{u_i}{\operatorname{argmin}}\ & V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i) \\
\text{s.t.}\ \ & x_{i+1} = f(x_i, u_i) \\
& (x_i, u_i) \in \mathcal{X} \times \mathcal{U} \\
& x_N \in \mathcal{X}_N \\
& x_0 = x
\end{aligned}
$$

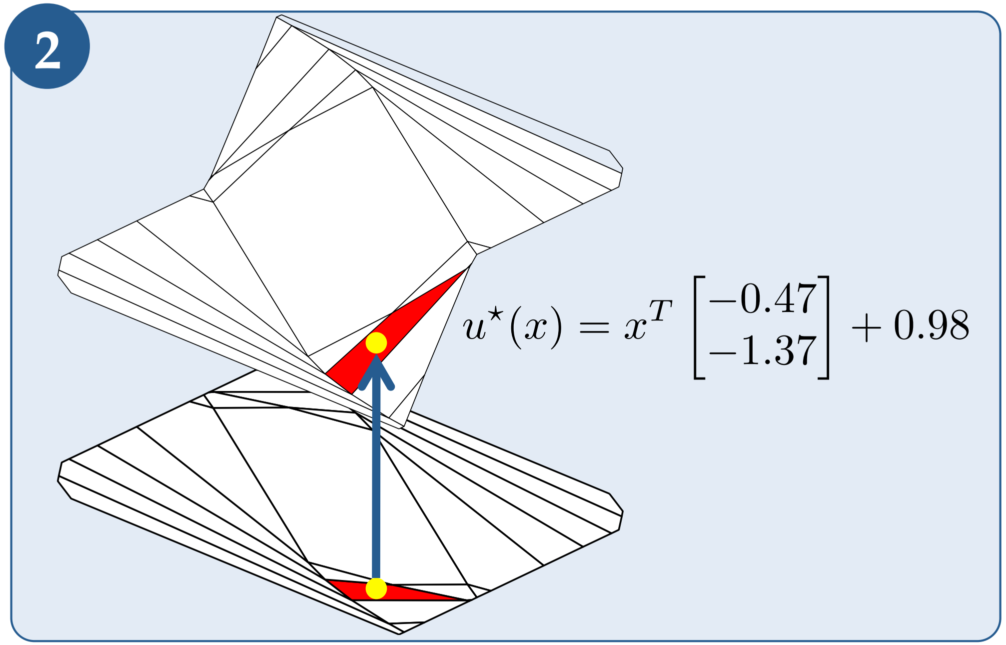


[M.M. Seron, J.A. De Doná and G.C. Goodwin, 2000]
[T.A. Johansen, I. Peterson and O. Slupphaug, 2000]
[A. Bemporad, M. Morari, V. Dua and E.N. Pistokopoulos, 2000]

Online speed depends on number of control law regions

- Online evaluation reduced to:
  1. Point location
  2. Evaluation of affine function
- Online complexity is governed by point location
  – Function of number of regions in cell complex
  – Milli- to microseconds possible only ***if small number of regions!!***





Real-time ⇔ synthesize control law of *specified* complexity

- Explicit MPC may not satisfy given real-time constraint
  - Complexity independent of available processing power
  - Number of regions (complexity) is exponentially sensitive to
    - State dimension
    - Input dimension
    - Small changes in system dynamics

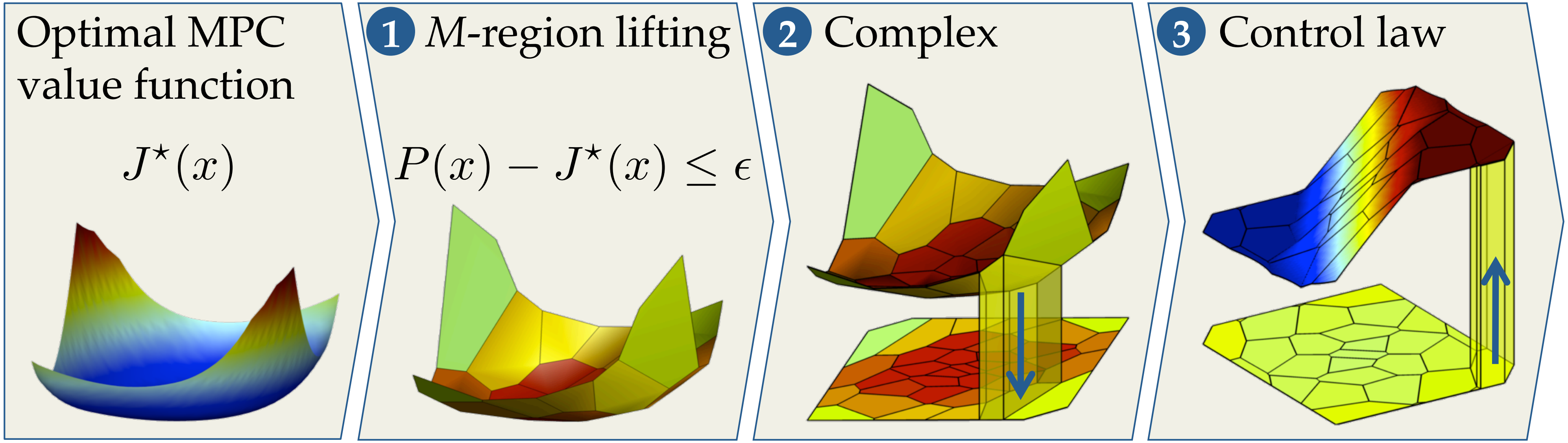
Idea : Real-time explicit MPC with complexity as input

Algorithm properties:

- Tradeoff between complexity and optimality
  - Real-time synthesis
  - Control extremely high-speed systems
- Process any convex MPC problem
- Synthesis of control law to software is verifiable

[C.N. Jones and M. Morari, TAC 2010]

Real-time explicit MPC : Offline processing



Given optimal controller:

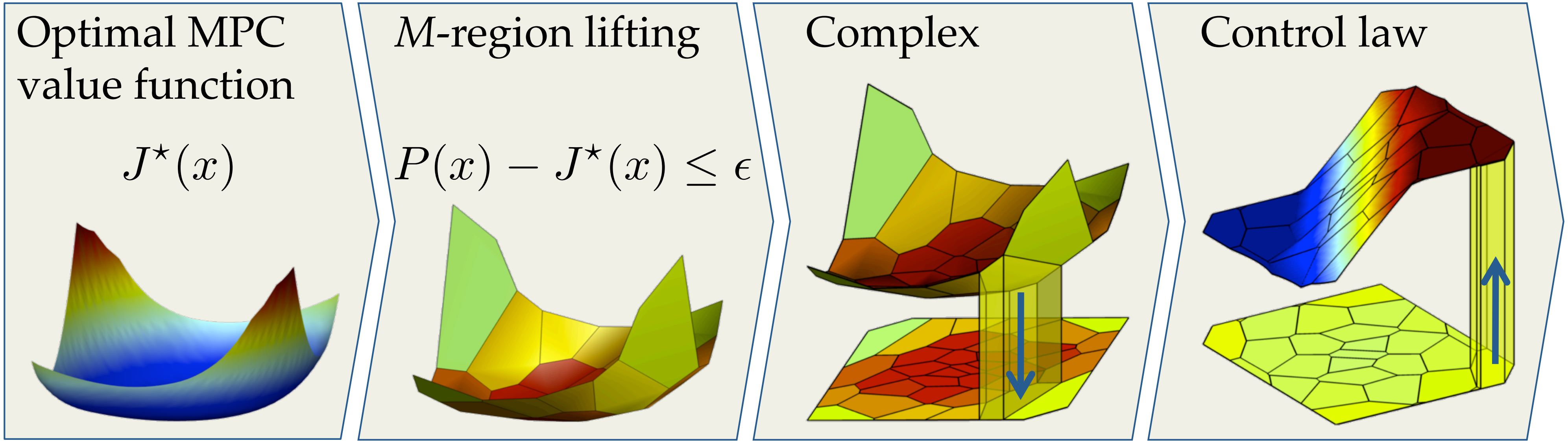
1. Compute convex polyhedral function of M facets
2. Define complex as projection of lifting facets
3. Interpolate optimal control law at vertices of complex

$$J^\star(x_0) = \min_{u_i} V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i)$$
$$\text{s.t. } x_{i+1} = f(x_i, u_i)$$
$$(x_i, u_i) \in \mathcal{X} \times \mathcal{U}$$
$$x_N \in \mathcal{X}_N$$

Result : Piecewise polynomial controller of M regions

[C.N. Jones and M. Morari, TAC 2010]

Real-time explicit MPC : Properties



Real-time explicit MPC:

Is computable in micro- to nanoseconds <= Lifting function

Satisfies constraints <= Barycentric interpolation

Stabilizes the system

Complexity / performance tradeoff

[C.N. Jones and M. Morari, TAC 2010]

ε -approx controller is stable if ε < 1

$$J(u) := V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i)$$

$$J^\star(x_0) := \min_{u_i} \; J(u)$$

$$\text{s.t. } x_{i+1} = f(x_i, u_i)$$

$$(x_i, u_i) \in \mathcal{X} \times \mathcal{U}$$

$$x_N \in \mathcal{X}_N$$

Thm: $x^+ = f(x, \tilde{u}(x))$

is stable if

$$J^\star(x) \le J(\tilde{u}(x)) \le J^\star(x) + \epsilon l(x, 0)$$

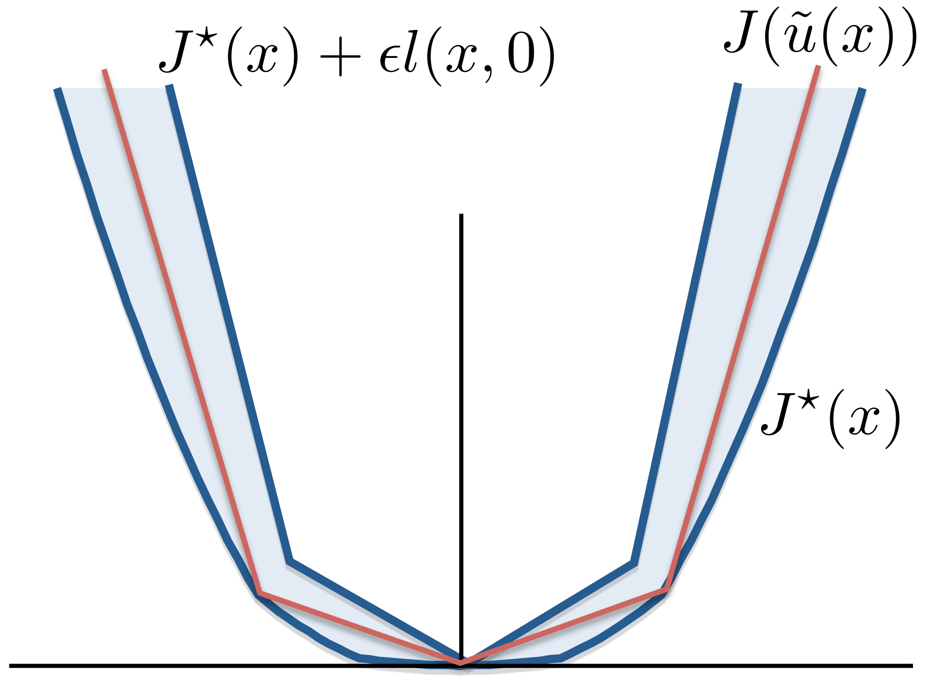
for $\epsilon < 1$

Sufficiently close to optimal

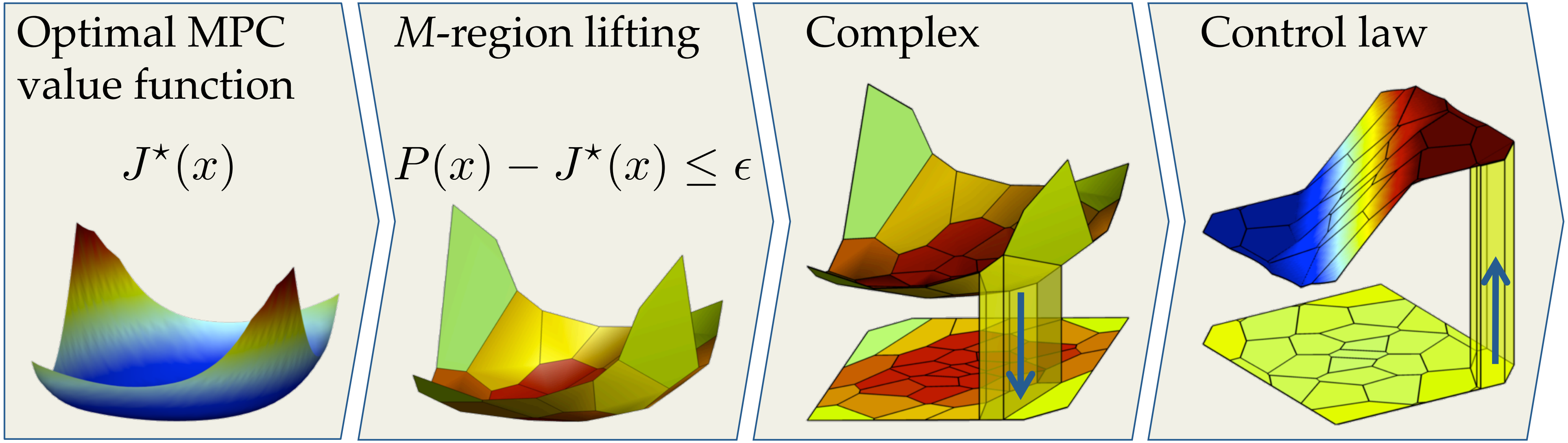
⇒ Stabilizing

Idea:

- Find a lifting sufficiently close to optimal and use it to define $\tilde{u}(x)$



Real-time explicit MPC : Properties



Real-time explicit MPC:

Is computable in micro- to nanoseconds <= Lifting function

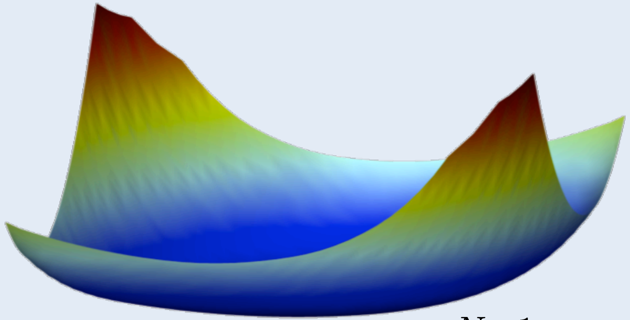
Satisfies constraints <= Barycentric interpolation

Stabilizes the system <= Error less than one

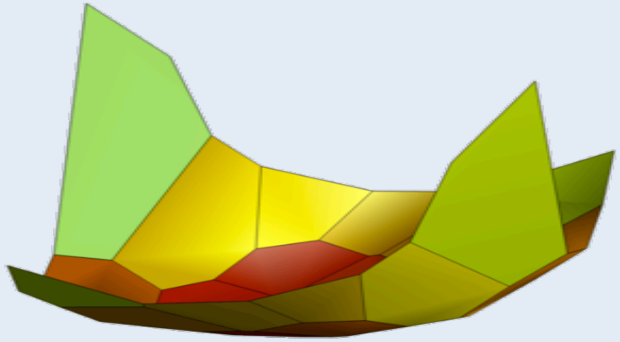
Complexity/performance tradeoff

[C.N. Jones and M. Morari, TAC 2010]

M-region approximation => Double description method



$$J^\star(x_0) = \min_{u_i} V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i)$$
$$\text{s.t. } x_{i+1} = f(x_i, u_i)$$
$$(x_i, u_i) \in \mathcal{X} \times \mathcal{U}$$
$$x_N \in \mathcal{X}_N$$



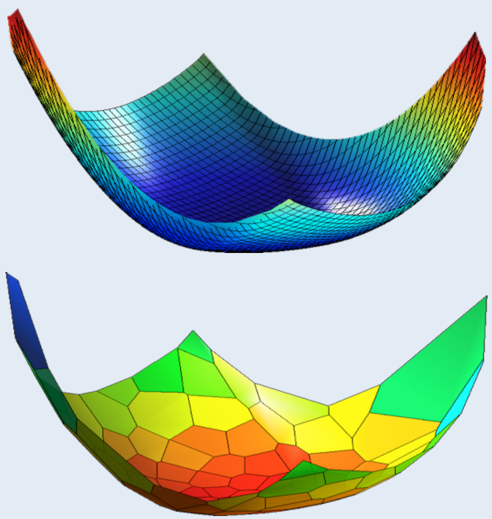
$$P(x)$$

- Approximate convex parametric programming
- Open problem in many areas:
  - Vertex enumeration, Projection, Non-negative matrix factorization…
  - These problems are known to be NP-hard

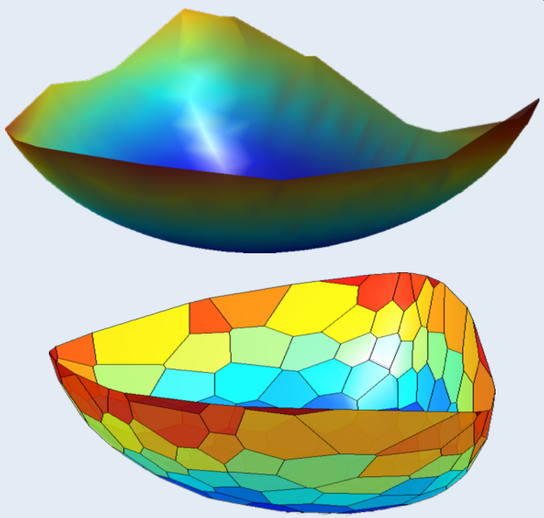
$\Rightarrow$ Poly-time greedy-optimal algorithm

Double description method : Algorithm properties

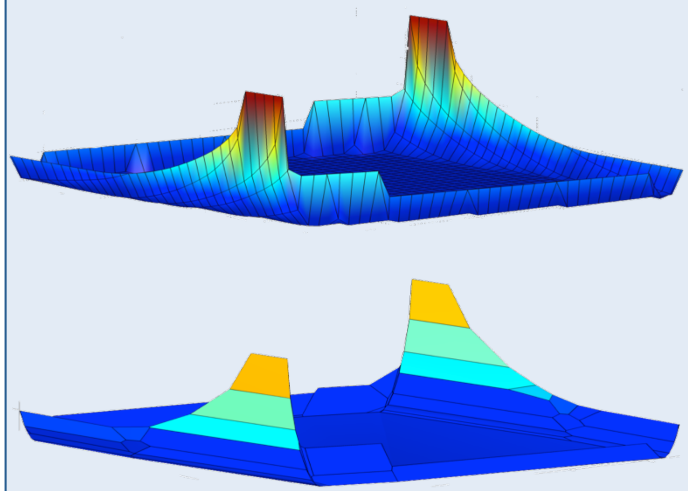
- Lifting of M regions <= Iterate algorithm M times
- Monotonic decrease in Hausdorff distance
  - Complexity / performance tradeoff via M
- There exists a minimum M for stability
  - ε-error in finite time ⇒ will find a Lyapunov function
  - Once stable, always stable



Quadratic programming



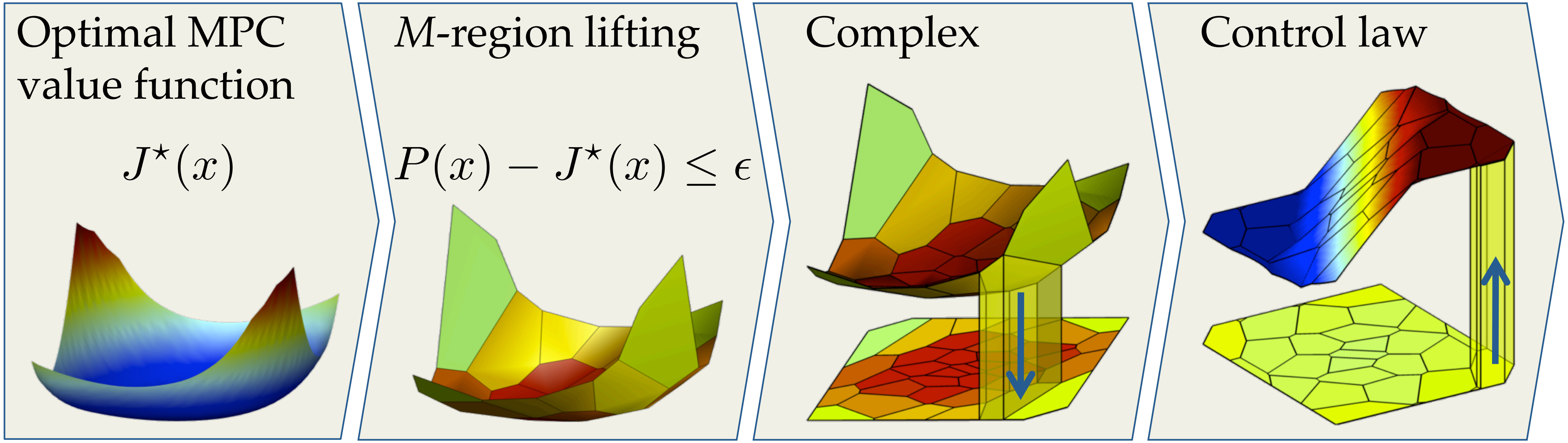
Geometric programming



Second-order cone programming

[C.N. Jones and M. Morari, TAC 2010]

Real-time explicit MPC : Properties

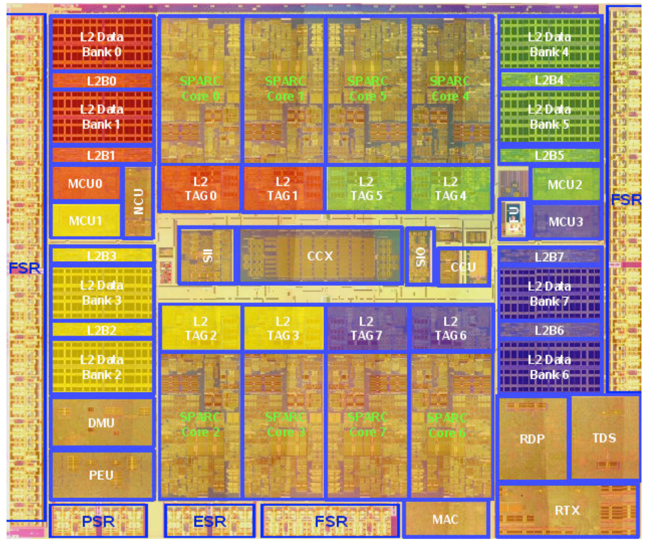


Real-time explicit MPC:

Is computable in micro- to nanoseconds	<= Lifting function
Satisfies constraints	<= Barycentric interpolation
Stabilizes the system	<= Error less than one
Complexity / performance tradeoff	<= M-region lifting

[C.N. Jones and M. Morari, TAC 2010]

Example :
Temperature Regulation of Multi-Core Processor

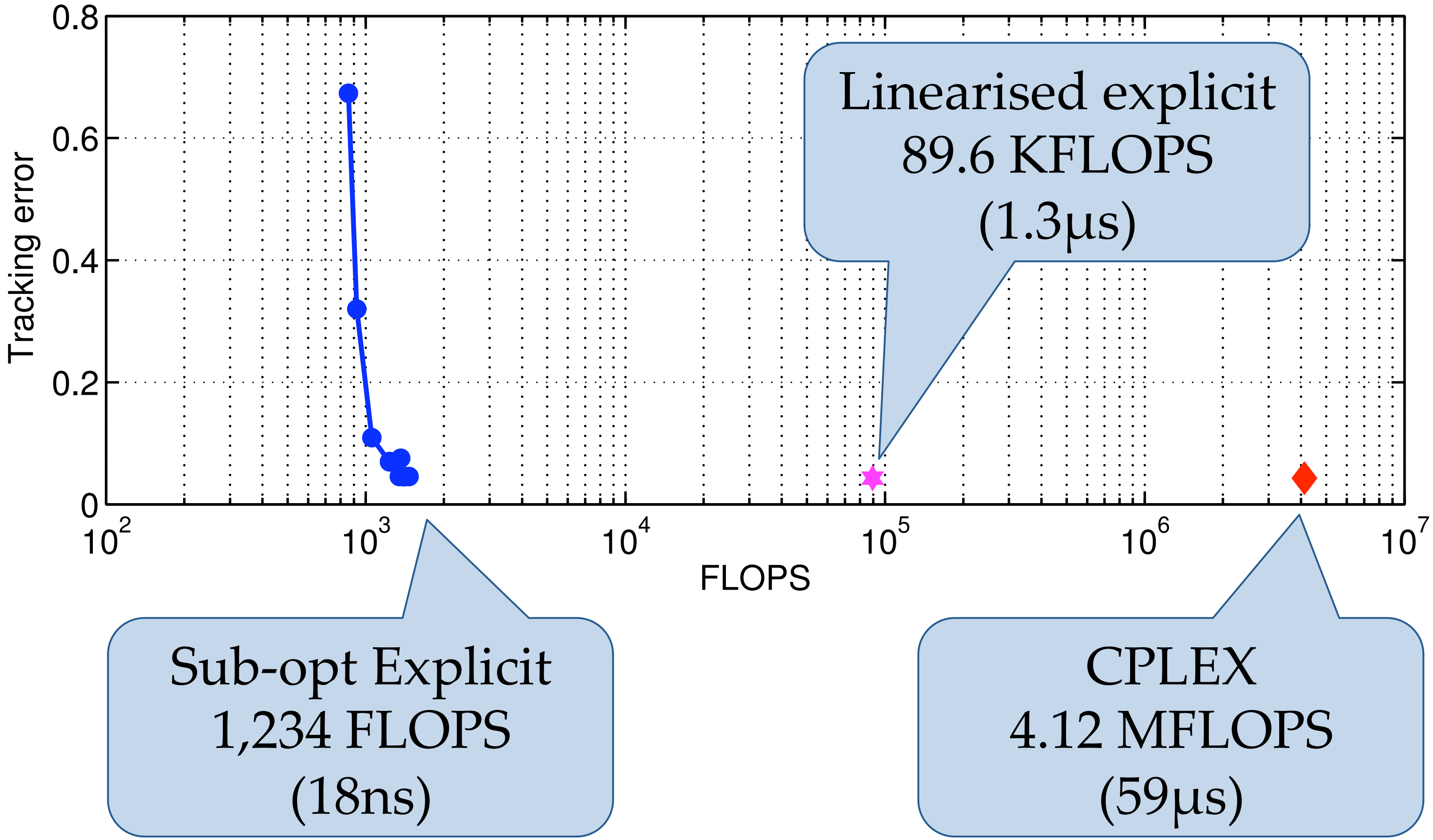


- Goals
  - Track workload requests
  - Minimize power usage
  - Respect temperature limits
- Quadratic nonlinear dynamics
  - Exact convex relaxation
- Stringent computational and storage requirements

$$
\begin{aligned}
J^\star(x_0, w) = \min_{f_i} \ & \sum_{t=0}^{N} \sum_{i=0}^{t} (w_i - f_i) \\
\text{s.t.} \ & x_{i+1} = Ax_i + Bf_i^2 \\
& \sum_{i=0}^{t} w_i \le \sum_{i=0}^{t} f_i \\
& x_i \le T_{\max} \\
& f_{\min} \le f_i \le f_{\max}
\end{aligned}
$$

[F. Zanini, C.N. Jones, D. Atienza, and G. De Micheli, 2010]

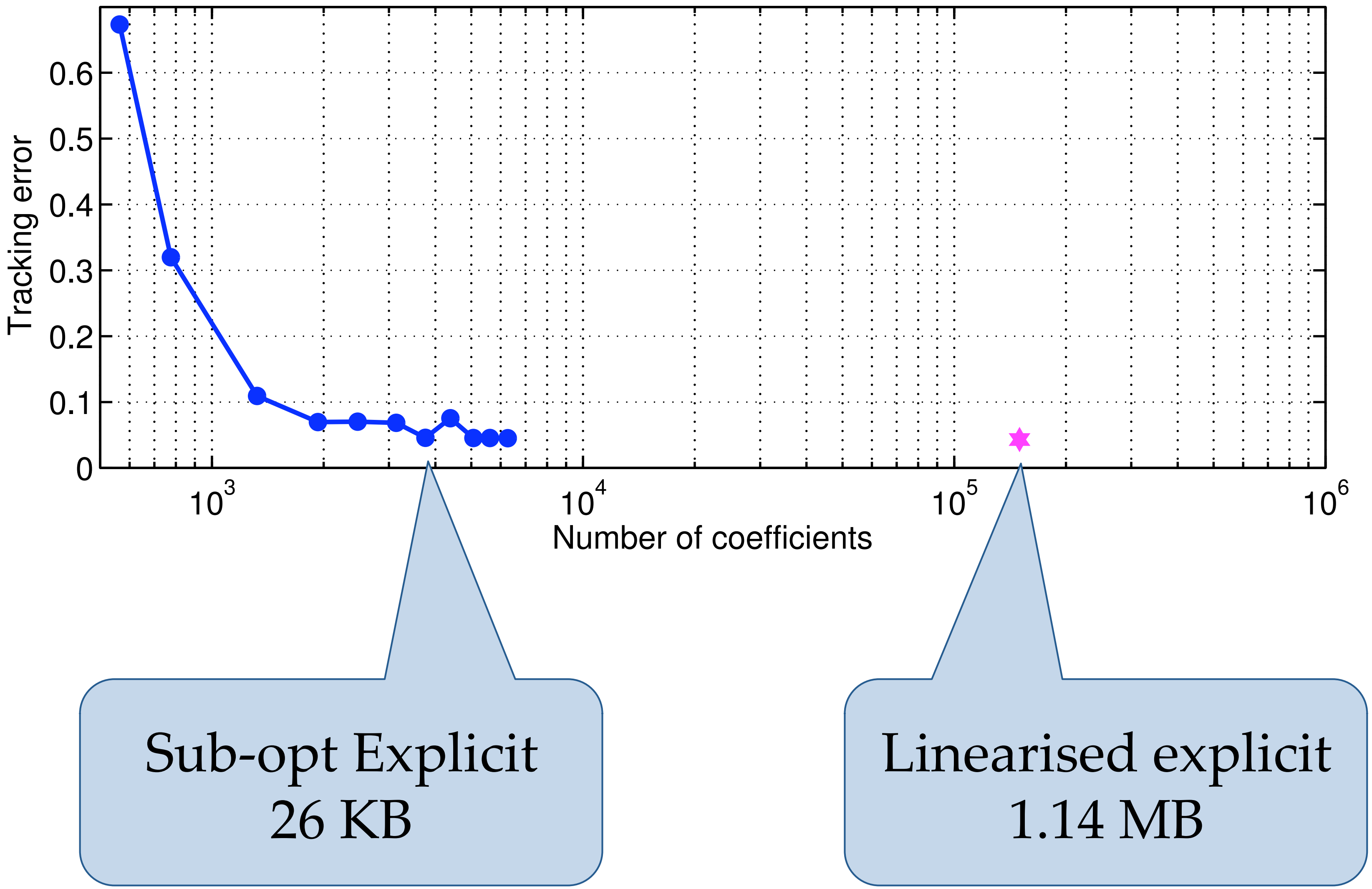
Computational results for QCQP : >3,000× faster



(Assuming 70 GFLOPS / sec – e.g., Intel Core i7 965 XE)

>3,000× / 72× faster than CPLEX / lin. explicit

Computational results for QCQP : 45× less storage



45× less storage

Outline

Motivating examples

A key challenge : Fast, fixed-time optimization

- Interior-point methods : Milli-seconds
- Fast gradient methods : Micro-seconds
- Explicit methods : Nano-seconds

Summary

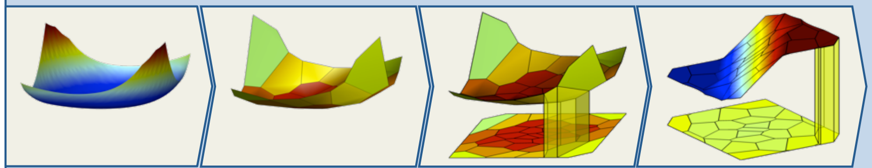
Summary

Formal specification
- YALMIP
- HYSDEL
- Linear + Hybrid models

Control law
- Explicit MPC
- Fixed-complexity solutions

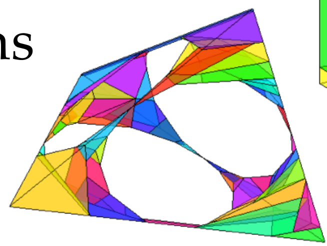
Software synthesis
- Real-time workshop
- Bounded-time solvers
- Verifiable code generation

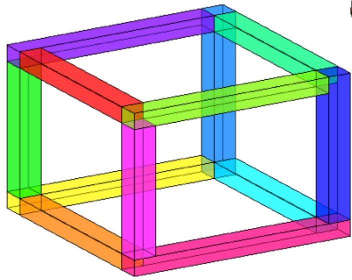
Verified controller

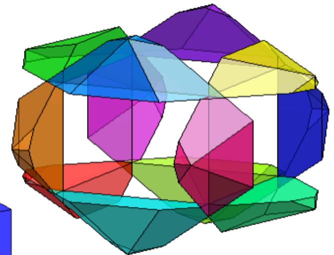


Multi-Parametric Toolbox (MPT)

- (Non)-Convex Polytopic Manipulation
- Multi-Parametric Programming
- Control of PWA and LTI systems
- > 22,000 downloads to date







MPT 3.0 coming in 2010