

Model Predictive Control

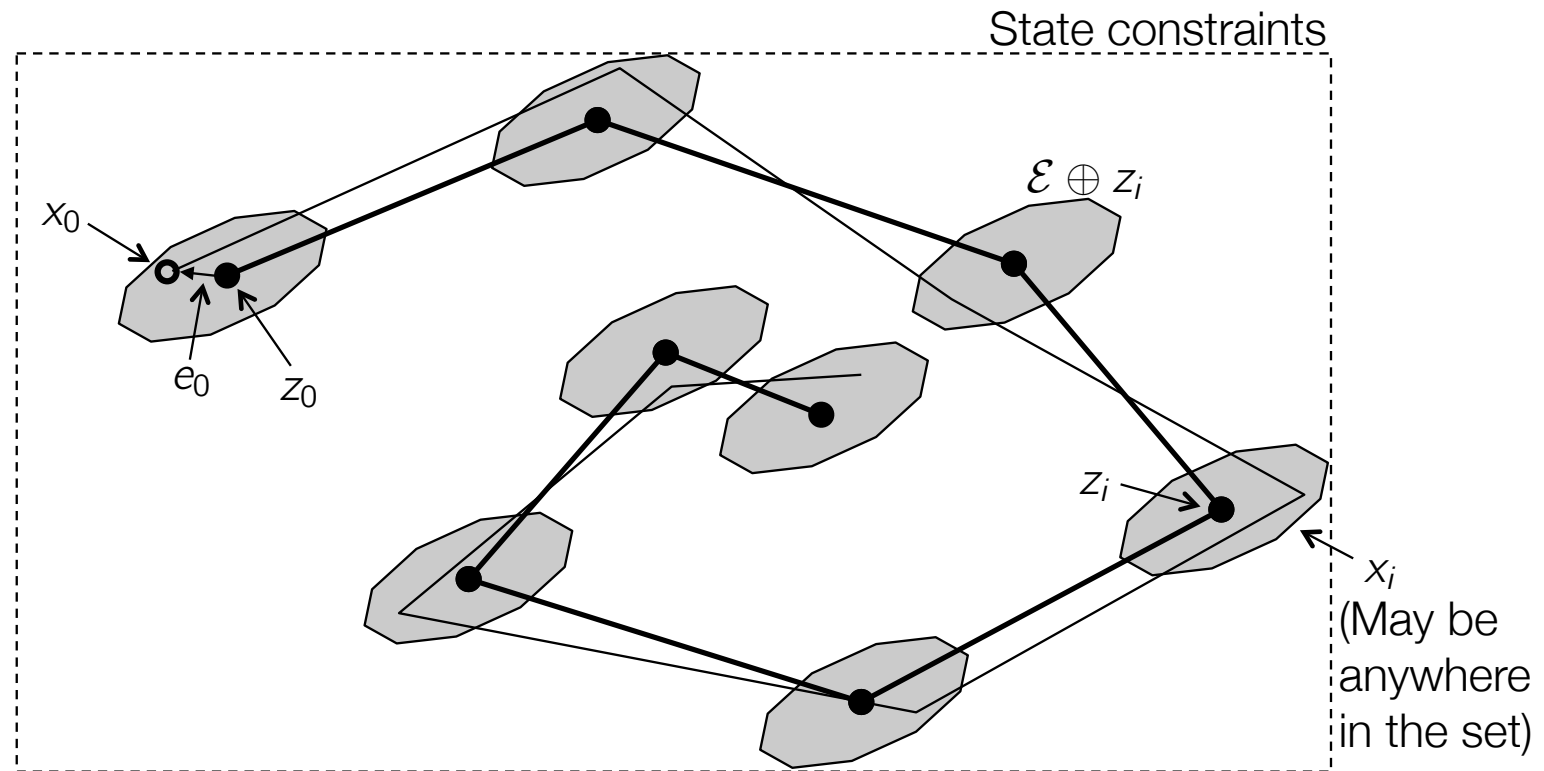
Lecture: Explicit MPC

Colin Jones

Laboratoire d'Automatique, EPFL

26 April, 2012

Tube-MPC : The Idea



We must ensure that all possible state trajectories satisfy the constraints
This is now equivalent to ensuring that $z_i \oplus \mathcal{E} \subset \mathbb{X}$
(Satisfying input constraints is now more complex - more later)

Tube MPC - Summary

Idea:

- Split input into two parts: One to steer system (v), one to compensate for the noise (Ke)

$$u = Ke + v$$

- Optimize for the nominal trajectory, ensuring that any deviations stay within constraints

Benefits:

- Less conservative than open-loop robust MPC (we're now actively compensating for noise in the prediction)
- Works for unstable systems
- Optimization problem to solve is simple

Cons:

- Sub-optimal MPC (optimal is extremely difficult)
- Reduced feasible set when compared to nominal MPC
- We need to know what $\mathbb{W}$ is (this is usually not realistic)

Nominal MPC for Uncertain Systems - Summary

Idea

- Ignore the noise and hope it works

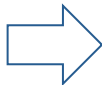
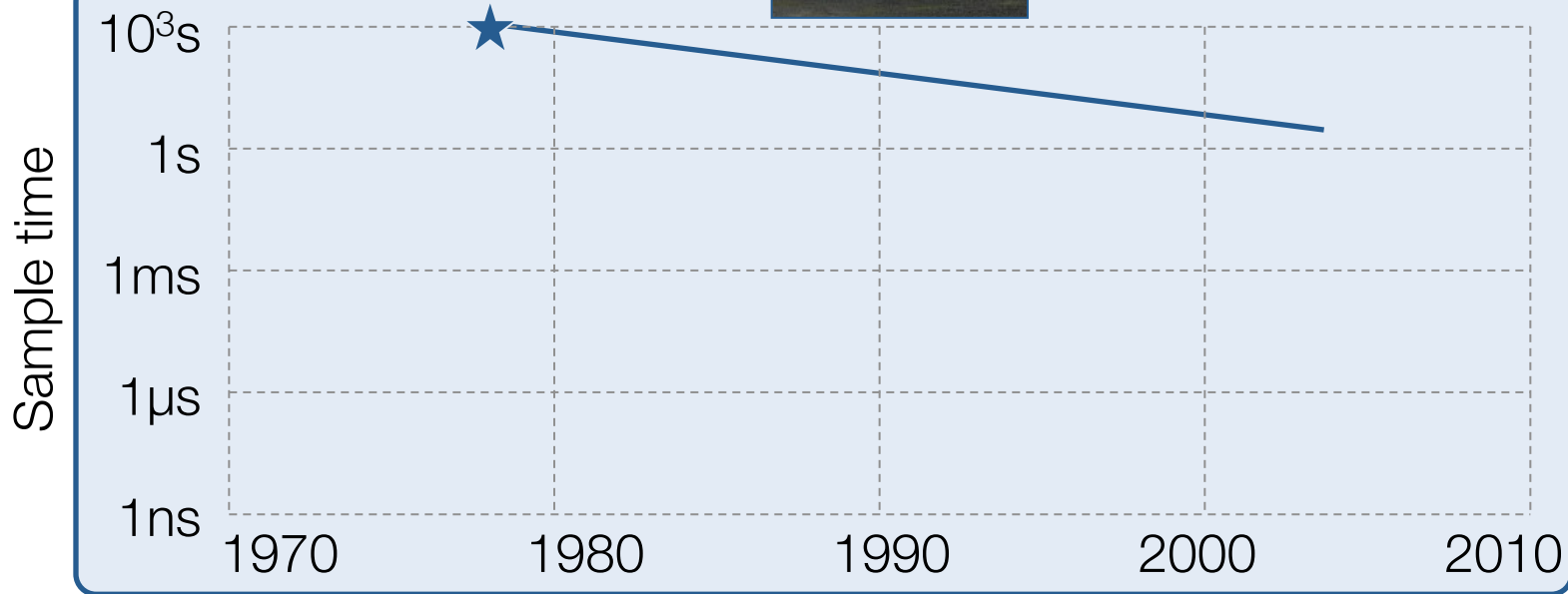
Benefits

- Simple
- No knowledge of the noise set $\mathbb{W}$ required - 'just works'
- Often very effective in practice (this is what most practitioners do anyway)
- Feasible set is large (we can find a solution, but it may be garbage)
- Region of attraction may be larger than other approaches

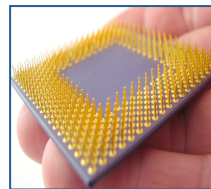
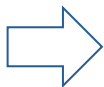
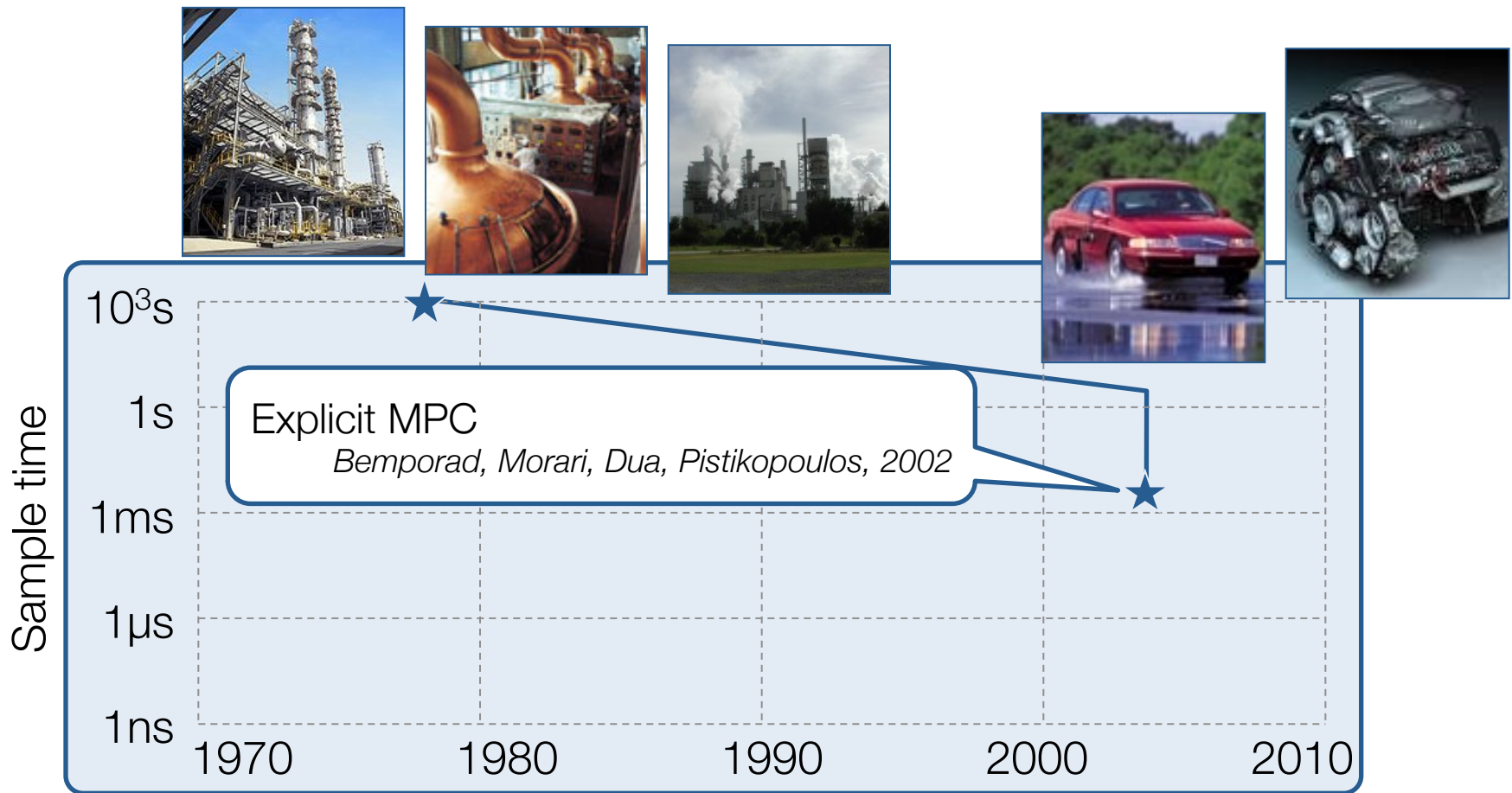
Cons

- Very difficult to determine region of attraction (set of states in which the controller works)
- Hard to tune - no obvious way to tradeoff robustness against performance

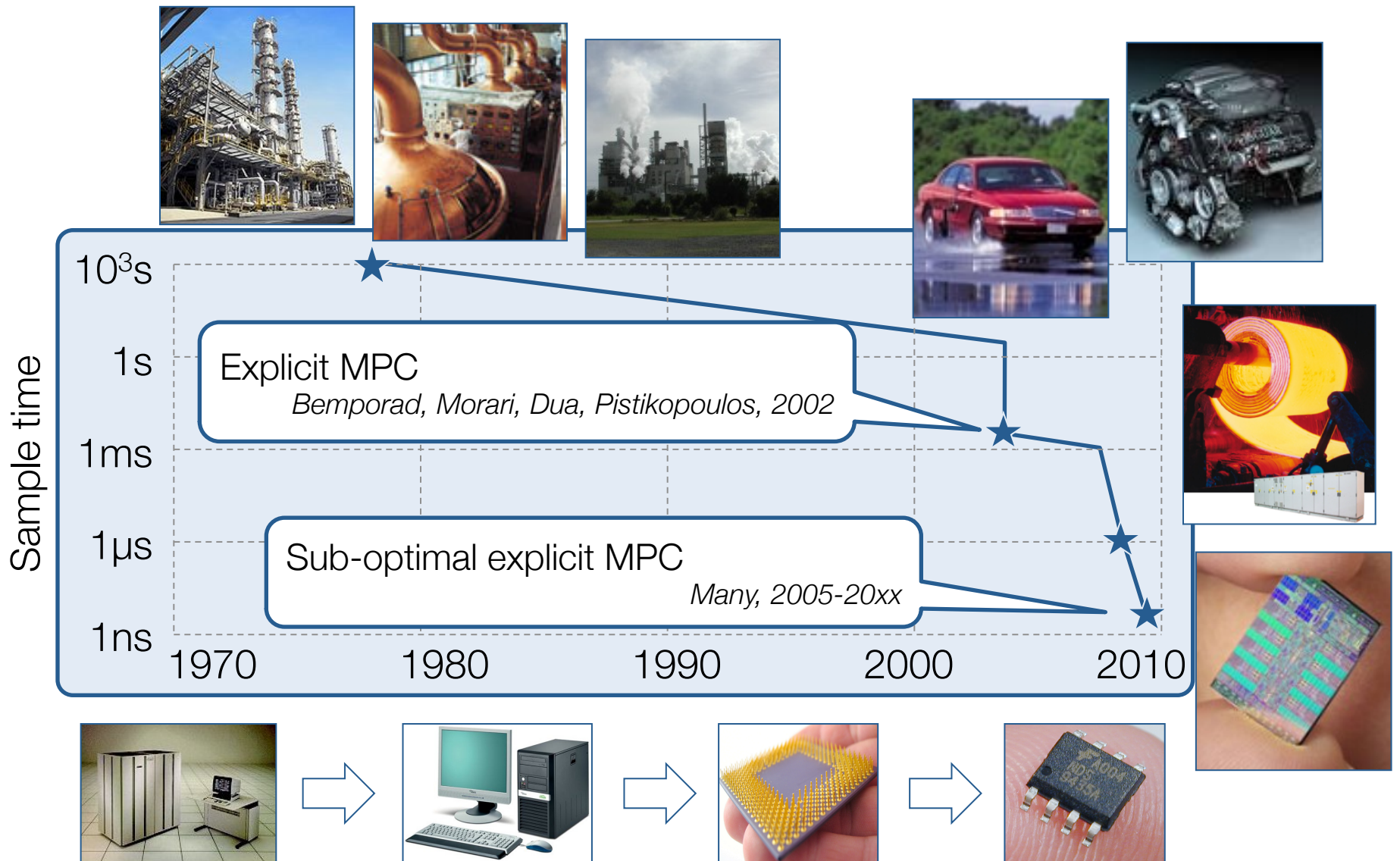
Evolution of MPC – beyond Moore's Law



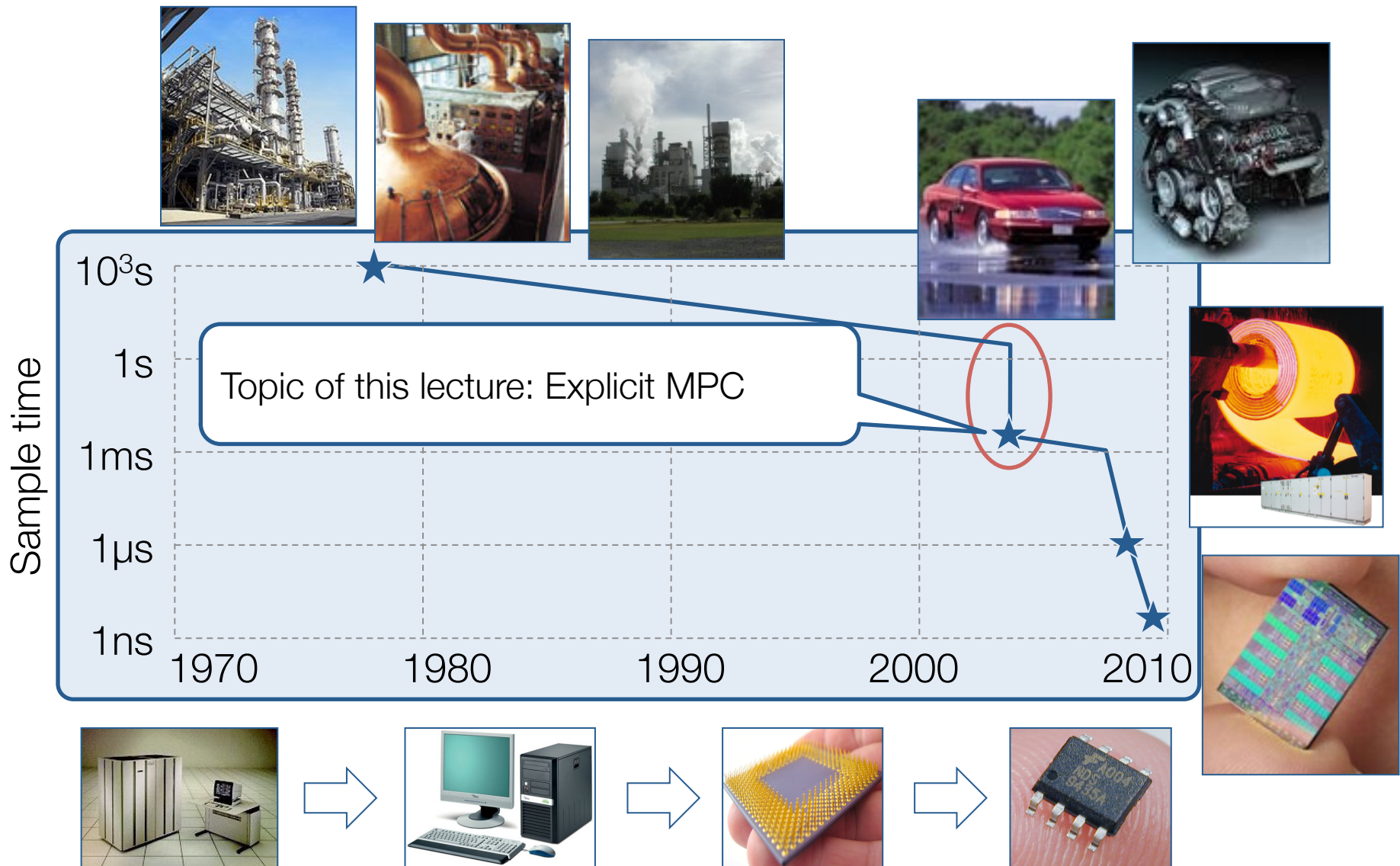
Evolution of MPC – beyond Moore's Law



Evolution of MPC – beyond Moore's Law



Evolution of MPC – beyond Moore's Law



Receding Horizon Control Synthesis

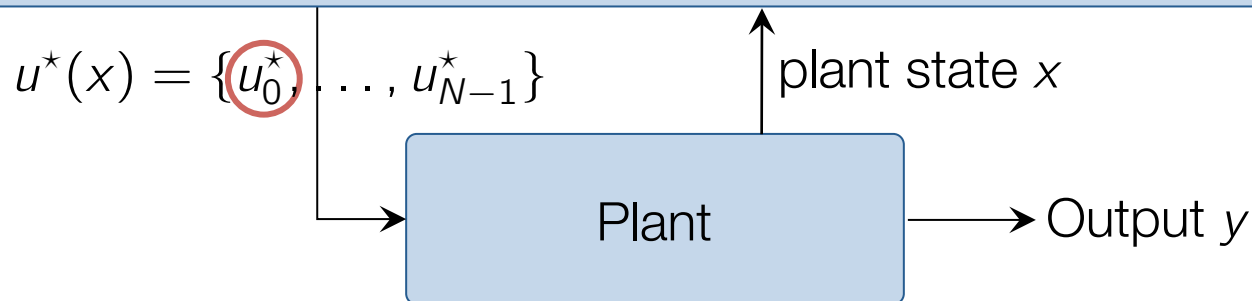
$$u^*(x) := \operatorname{argmin} \quad x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i$$

measurement

system model

constraints

performance weights



Solve optimization problem each time sample

- Computationally complex and relatively slow
- *Not real-time*

Receding Horizon Control Synthesis

OFFLINE | ONLINE

Parametric solver

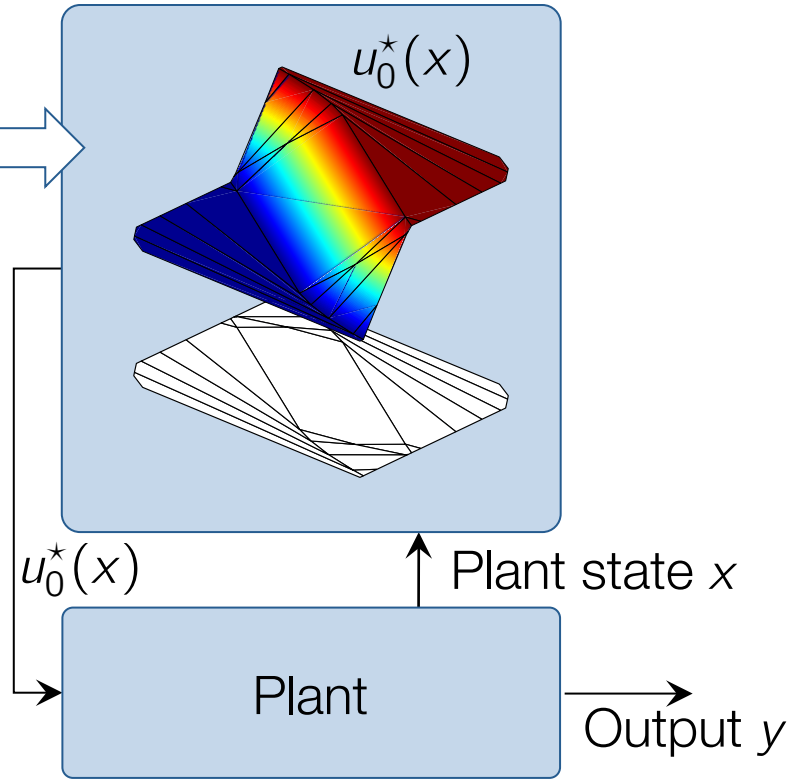
$$u^*(x) := \operatorname{argmin} \quad x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i$$

s.t. $x_0 = x$ measurement

$x_{i+1} = Ax_i + Bu_i$ system model

$Cx_i + Du_i \leq b$ constraints

$R \succ 0, Q \succ 0$ performance weights



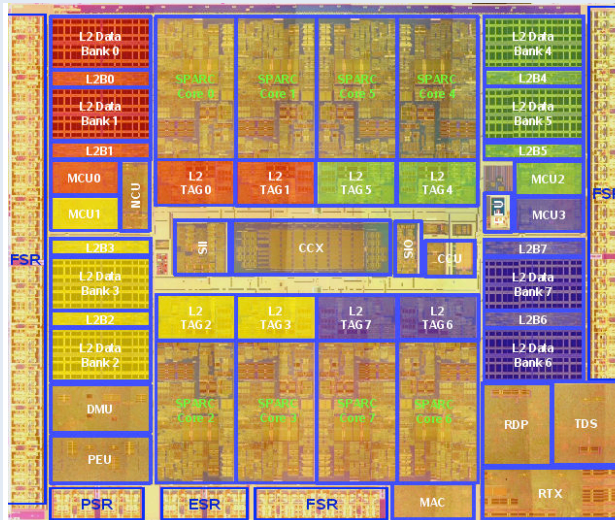
- Optimization problem is function parameterized by state
- Control law piecewise affine for linear systems/constraints
- Pre-compute control law as function of state x

Result : Online computation dramatically reduced and *real-time*

Example : How fast is fast?

Temperature Regulation of Multi-Core Processor

- Goals
 - Track workload requests
 - Minimize power usage
 - Respect temperature limits
- Quadratic nonlinear dynamics
 - Convex PWA approximation
- Stringent computational and storage requirements

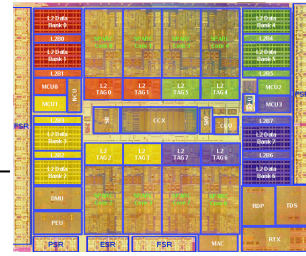


$$J^*(x_0, w) = \min_{f_i} \sum_{t=0}^N \sum_{i=0}^t (w_i - f_i)$$
$$\text{s.t. } x_{i+1} = Ax_i + Bf_i^2$$
$$\sum_{i=0}^t w_i \leq \sum_{i=0}^t f_i$$
$$x_i \leq T_{\max}$$
$$f_{\min} \leq f_i \leq f_{\max}$$

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

Example : How fast is fast?

Temperature Regulation of Multi-Core Processor



Work to do at time i

Frequency of processors
at time i
(work that can be done)

$$J^*(x_0, w) = \min_{f_i} \sum_{t=0}^N \sum_{i=0}^t (w_i - f_i)$$

$$\text{s.t. } x_{i+1} = Ax_i + Bf_i^2$$

$$\sum_{i=0}^t w_i \leq \sum_{i=0}^t f_i$$

$$x_i \leq T_{\max}$$

$$f_{\min} \leq f_i \leq f_{\max}$$

Temperature x is a quadratic
function of frequency
(Have to approximate)

Can't do work before it's
requested

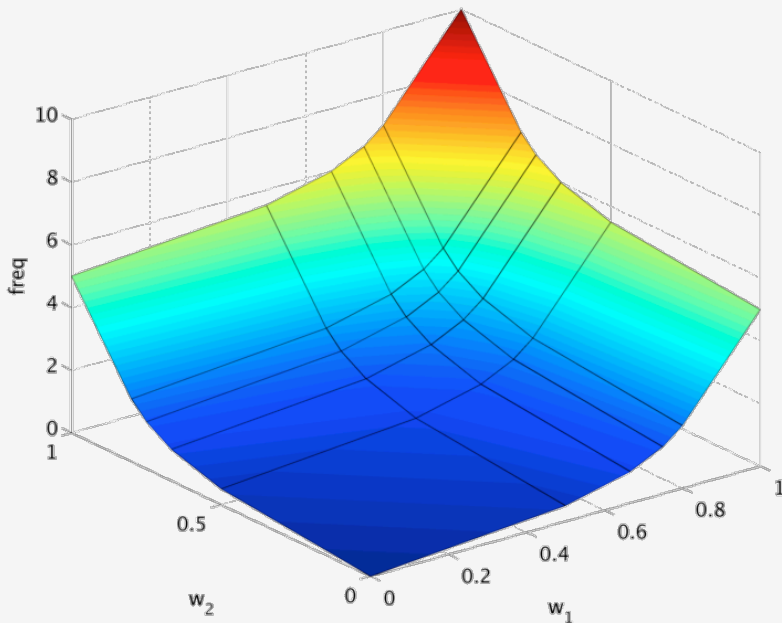
Don't overheat

Clock frequency is
bounded

Multi-core thermal regulation : Control law

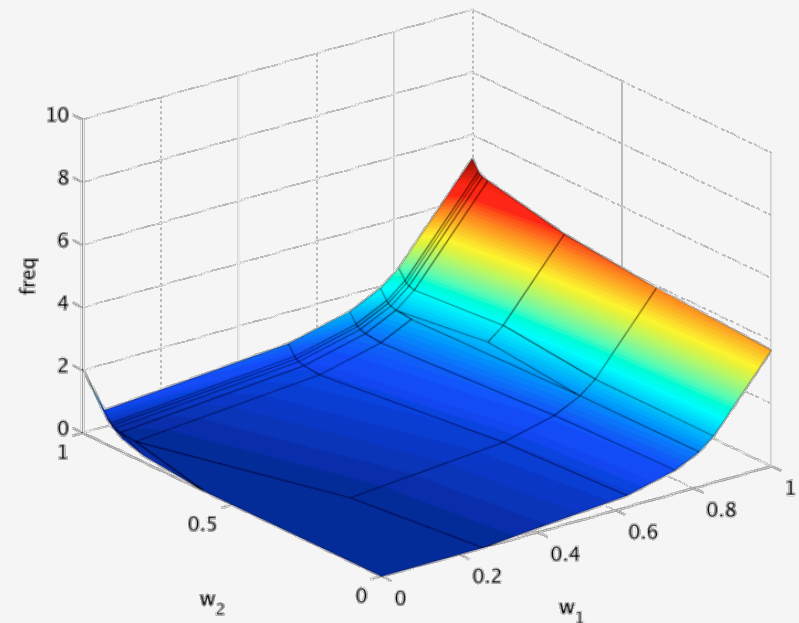
Cold chip

Do all requested work



Warm chip

Move work to cooler cores



Example : How fast is fast?

Offline processing

- Time to compute control law : 196 sec

Online processing

- Required storage : 17'969 numbers (71 kB)
- Required online computation : 10'737 FLOPS

Result

- Possible to compute control action in *145 ns*
 - (Assuming 70 GFLOPS)
- Compare to commercial optimizer CPLEX : 4'120'000 FLOPS (~59 us)

Speed == Money

- Can use much slower/cheaper computational platform

Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Example

MPC → Parametric programming

$$J^*(x) = \min x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i$$

$$\text{s.t. } x_0 = x$$

$$x_{i+1} = A x_i + B u_i$$

$$C x_i + D u_i \leq b$$

Linear, quadratic or convex
piecewise affine cost functions.
Tracking and regulation.

Linear (or affine) dynamics

Linear constraints on states
and inputs



Equivalent
representation

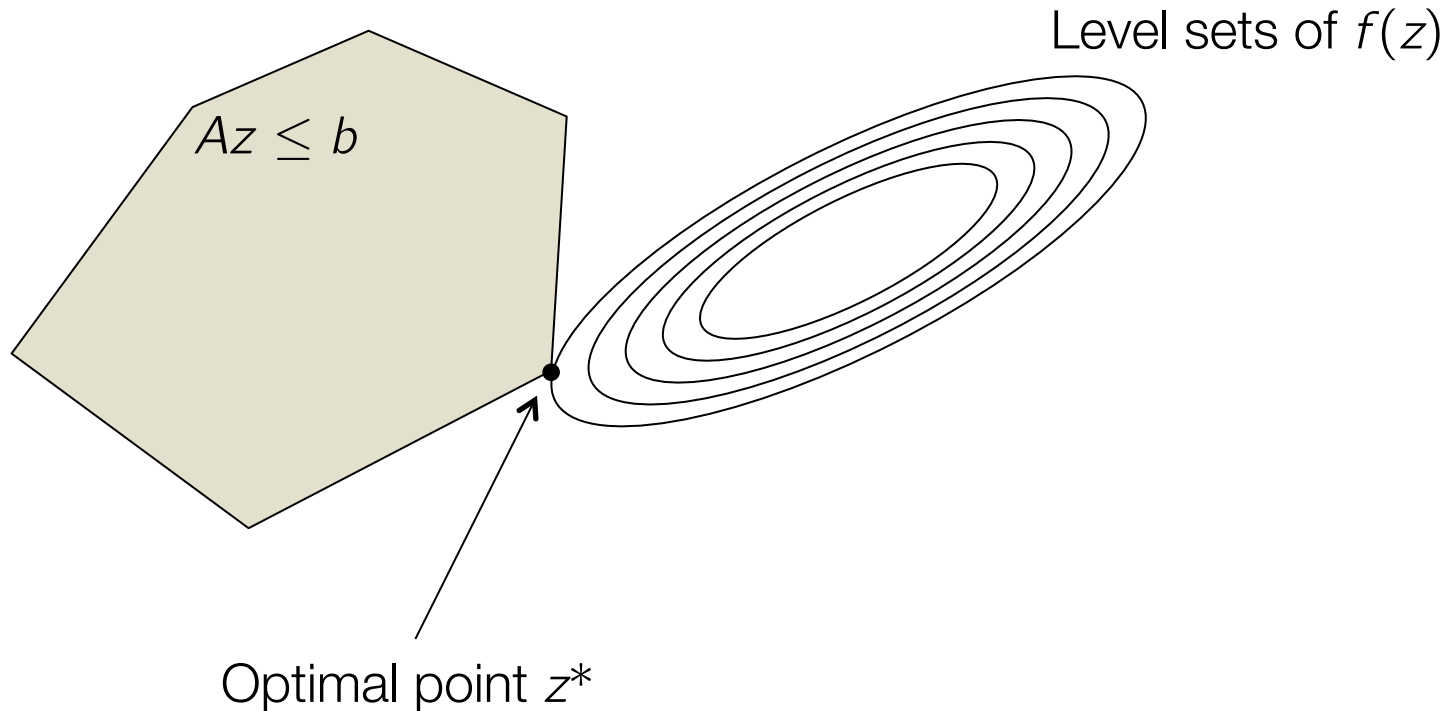
$$J^*(x) = \min_u \frac{1}{2} u^T Q u + (F x + f)^T u$$

$$\text{s.t. } G u \leq E x + e$$

It is also possible to represent piecewise affine systems or mixed-logic dynamic systems as *parametric mixed-integer programs*, but this is not covered in this tutorial.

QP Optimality Conditions

$$\begin{aligned} \min f(z) &:= \frac{1}{2} z^T Q z \\ \text{s.t. } Az &\leq b \end{aligned}$$



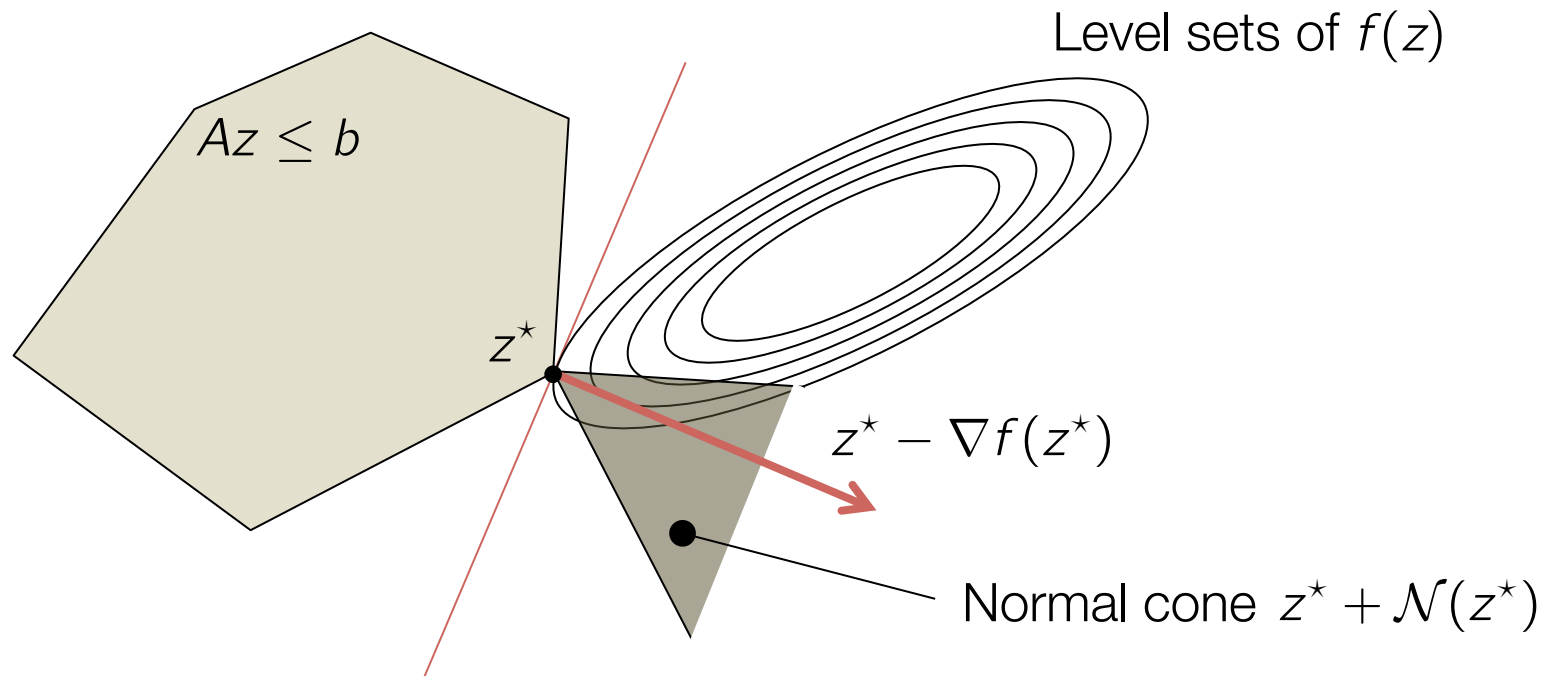
QP Optimality Conditions

$$\begin{aligned} \min f(z) &:= \frac{1}{2} z^T Q z \\ \text{s.t. } Az &\leq b \end{aligned}$$

Necessary optimality condition:

$$-\nabla f(z^*) \in \mathcal{T}(z^*)^* = \mathcal{N}(z^*)$$

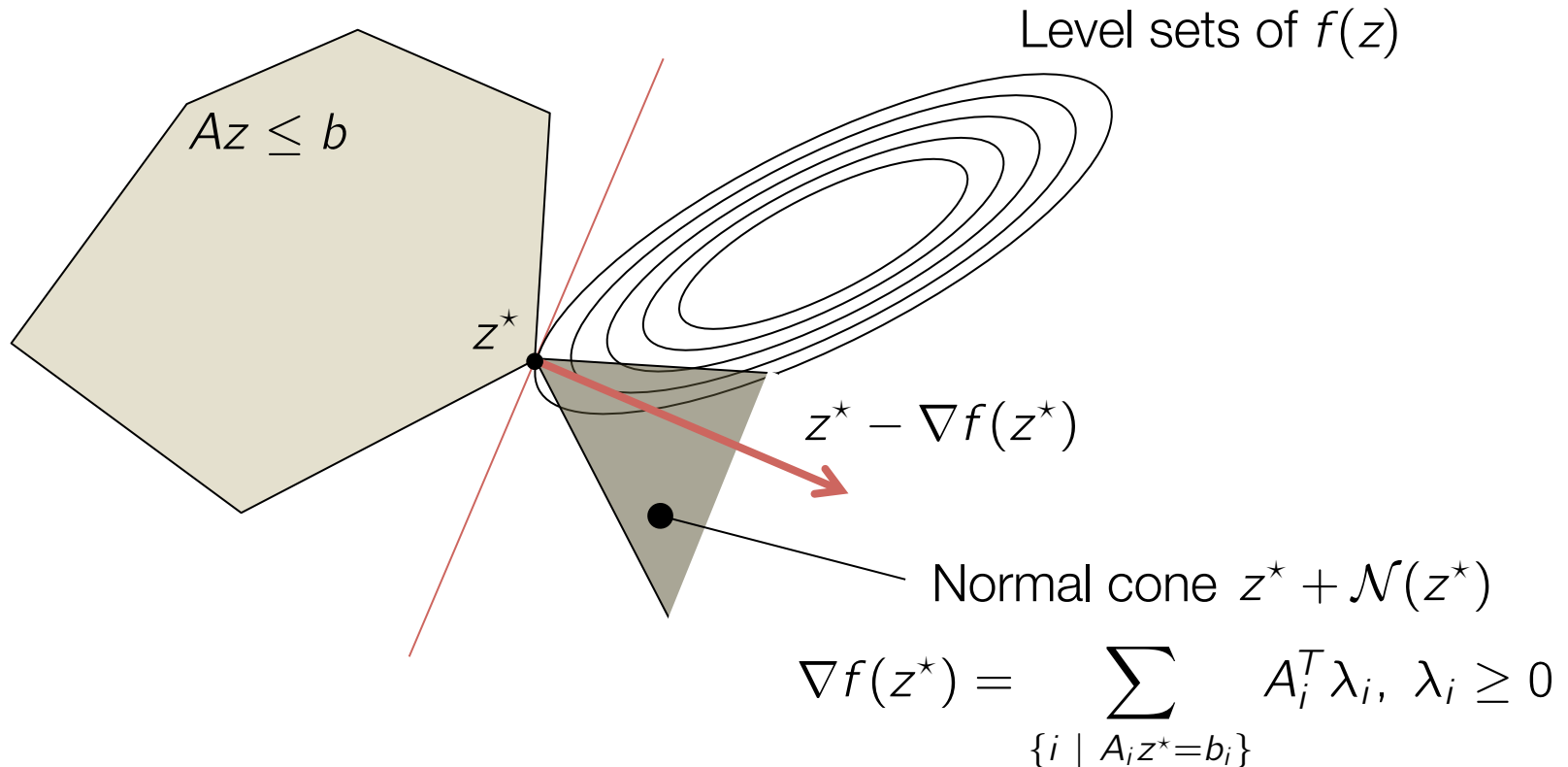
(Negative gradient is in the normal cone)



QP Optimality Conditions

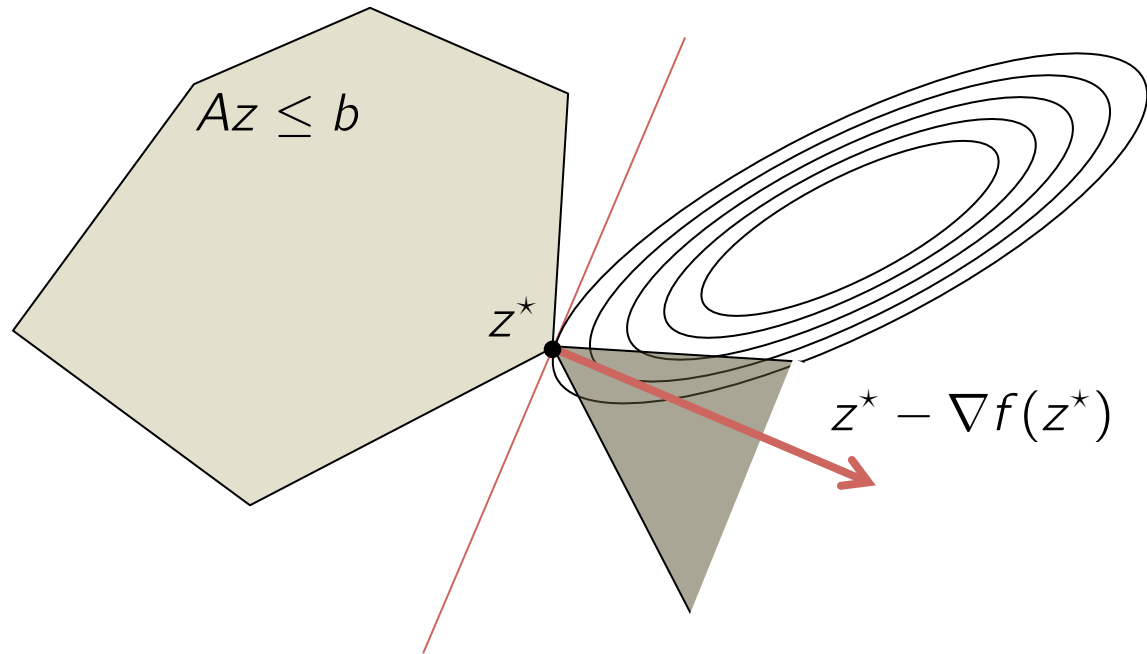
$$\begin{aligned} \min f(z) &:= \frac{1}{2} z^T Q z \\ \text{s.t. } Az &\leq b \end{aligned}$$

Necessary optimality condition:
 $-\nabla f(z^*) \in \mathcal{T}(z^*)^* = \mathcal{N}(z^*)$
(Negative gradient is in the normal cone)



QP Optimality Conditions

$$\begin{aligned} \min f(z) &:= \frac{1}{2} z^T Q z \\ \text{s.t. } &Az \leq b \end{aligned}$$



KKT Necessary and Sufficient Optimality Conditions for ConvexQPs

$Qz = A^T \lambda, \lambda \geq 0$	Gradient is in the normal cone
$Az \leq b$	Optimal point must be feasible
$\lambda^T (Az - b) = 0$	Normal cone contains only active constraints

Simple Parametric Programming Example

One-dimensional example

$$f^*(x) = \min_z \frac{1}{2}z^2 + 2xz$$

$$\text{s.t. } z \geq x - 1$$

$$z \geq 0$$

Find:

- Optimizer $z(x)$
- All x for which problem has a solution
- Value function $f^*(x)$

- KKT conditions

$$\nabla_z \mathcal{L} = z + 2x - \lambda - \nu = 0 \quad \text{Stationarity}$$

$$x - 1 - z \leq 0, \quad z \geq 0 \quad \text{Primal feasibility}$$

$$\lambda, \nu \geq 0 \quad \text{Dual feasibility}$$

$$\lambda(z - x - 1) = \nu z = 0 \quad \text{Complementarity}$$

Simple Parametric Programming Example

$$\nabla_z \mathcal{L} = z + 2x - \lambda - \nu = 0 \quad \text{Stationarity}$$

$$x - 1 - z \leq 0, \quad z \geq 0 \quad \text{Primal feasibility}$$

$$\lambda, \nu \geq 0 \quad \text{Dual feasibility}$$

$$\lambda(z - x - 1) = \nu z = 0 \quad \text{Complementarity}$$

Four complementarity cases:

$$\begin{array}{ll} \lambda = 0 & z \geq x - 1 \\ \nu = 0 & z \geq 0 \end{array} \rightarrow \begin{cases} z^*(x) &= -2x \\ f^*(x) &= -2x^2 \\ x &\leq 0 \end{cases}$$

$$\begin{array}{ll} \lambda = 0 & z \geq x - 1 \\ \nu \geq 0 & z = 0 \end{array} \rightarrow \begin{cases} z^*(x) = 0 \\ f^*(x) = 0 \\ 0 \leq x \leq 1 \end{cases}$$

$$\begin{array}{ll} \lambda \geq 0 & z = x - 1 \\ \nu = 0 & z \geq 0 \end{array} \rightarrow \begin{cases} z^*(x) &= x - 1 \\ f^*(x) &= \frac{5}{2}x^2 - 3x + \frac{1}{2} \\ x &\geq 1 \end{cases}$$

$$\begin{array}{ll} \lambda \geq 0 & z = x - 1 \\ \nu \geq 0 & z = 0 \end{array} \rightarrow \begin{cases} z^*(x) &= 0 \\ f^*(x) &= 0 \\ x &= 1 \end{cases}$$

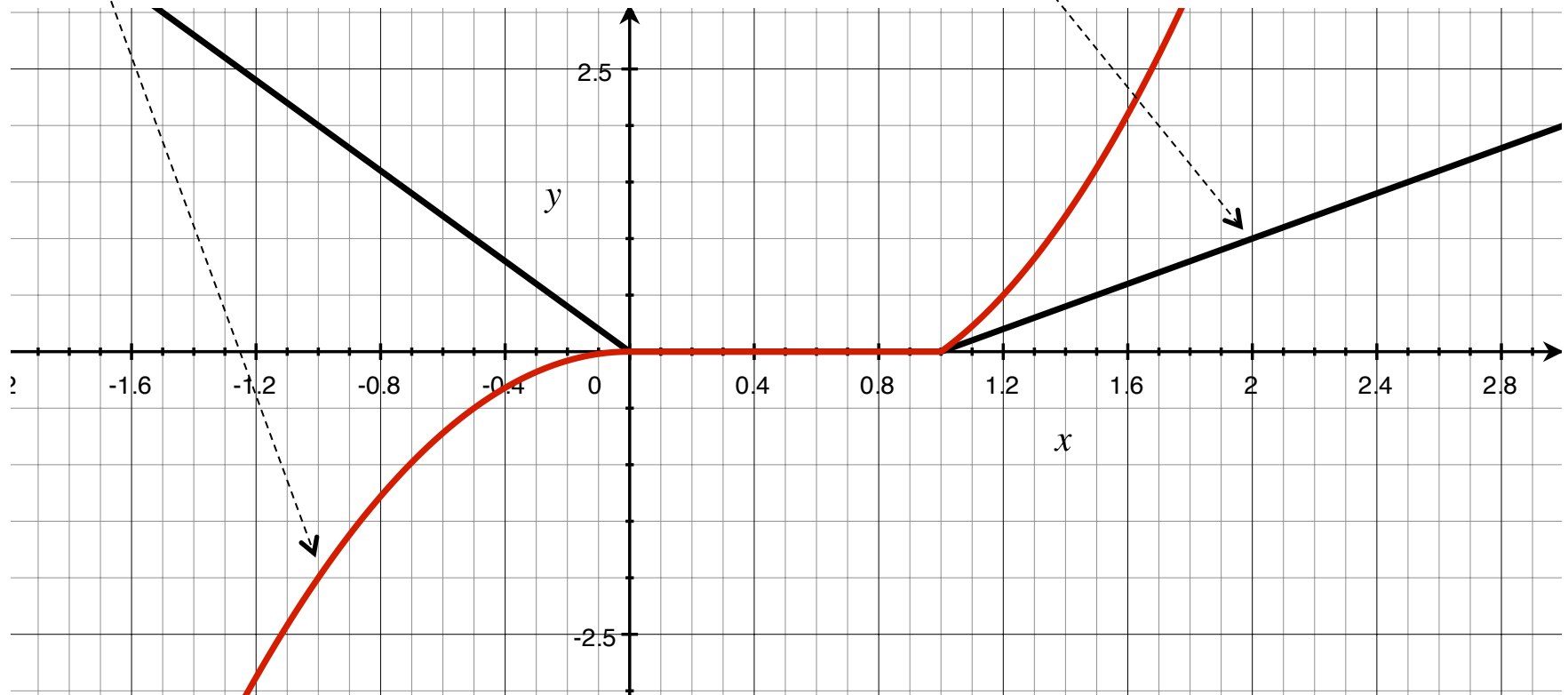
Simple Parametric Programming Example

Optimal value: Piecewise quadratic

$$f^*(x) = \begin{cases} -2x^2 & x \leq 0 \\ 0 & 0 \leq x \leq 1 \\ \frac{5}{2}x^2 - 3x + \frac{1}{2} & x \geq 1 \end{cases}$$

Optimizer : Piecewise affine

$$z^*(x) = \begin{cases} -2x & x \leq 0 \\ 0 & 0 \leq x \leq 1 \\ x - 1 & x \geq 1 \end{cases}$$



Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Example

General Formulation: Parametric Linear Complementarity

Parametric Linear Complementarity Problem

Given matrices M , q and Q , find functions $w(x)$, $z(x)$ such that

$$w - Mz = q + Qx$$

$$w^T z = 0$$

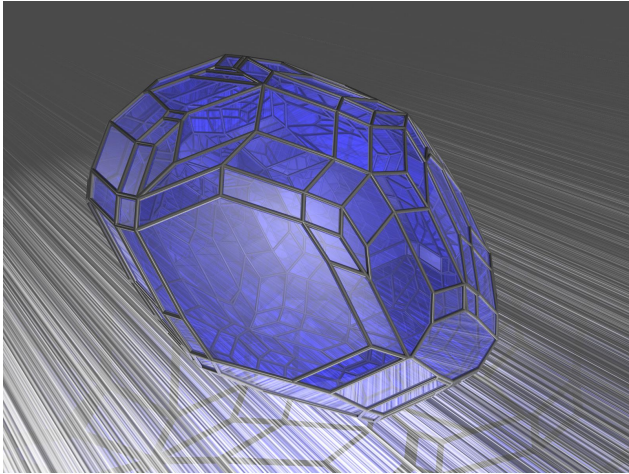
$$w, z \geq 0$$

- General form of the KKT conditions for quadratic programming
- Encompasses RIM linear and quadratic programming
 - Linear parameters in the cost and RHS
- Includes ‘standard’ MPC problems

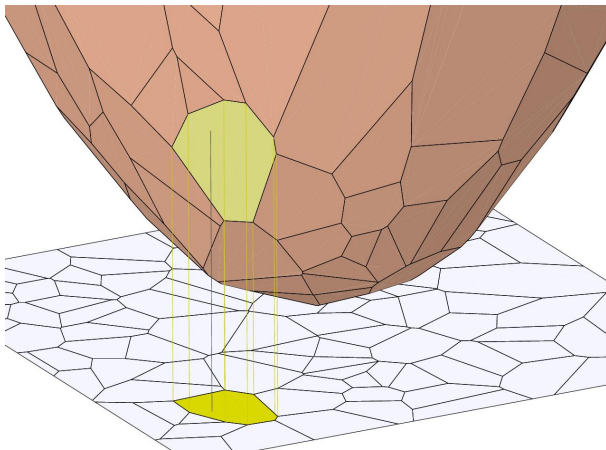
... but much more general

Geometric Problems Solvable by Parametric Linear Complementary Programming

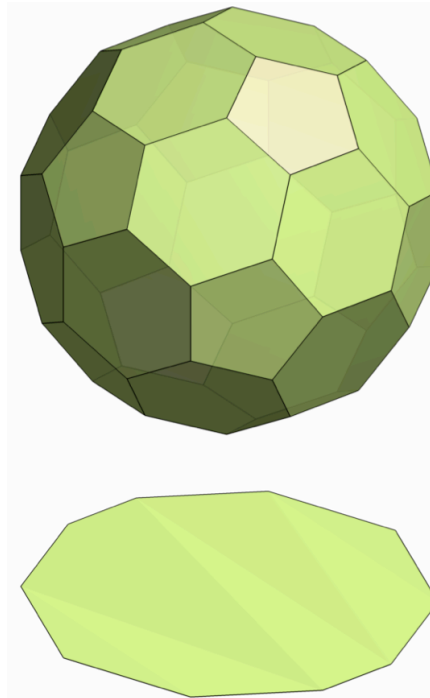
Invariant Sets



Voronoi diagrams



Polytopic projection

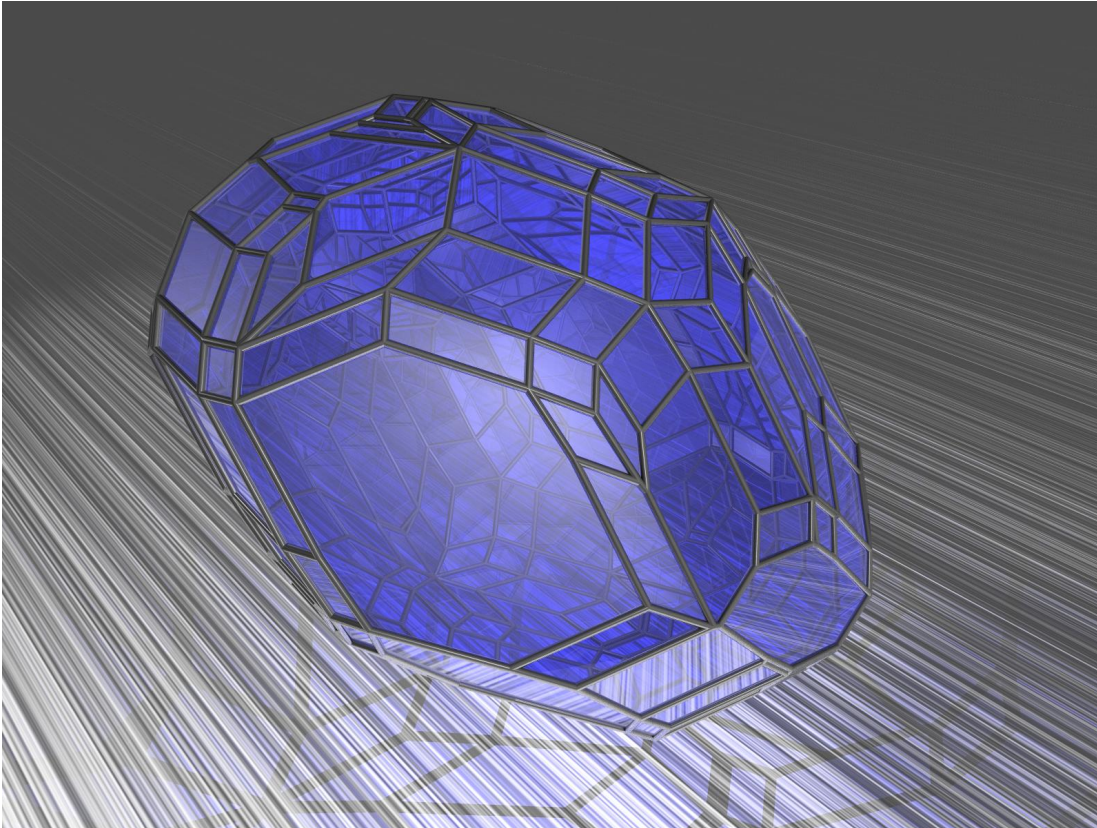


Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Invariant sets



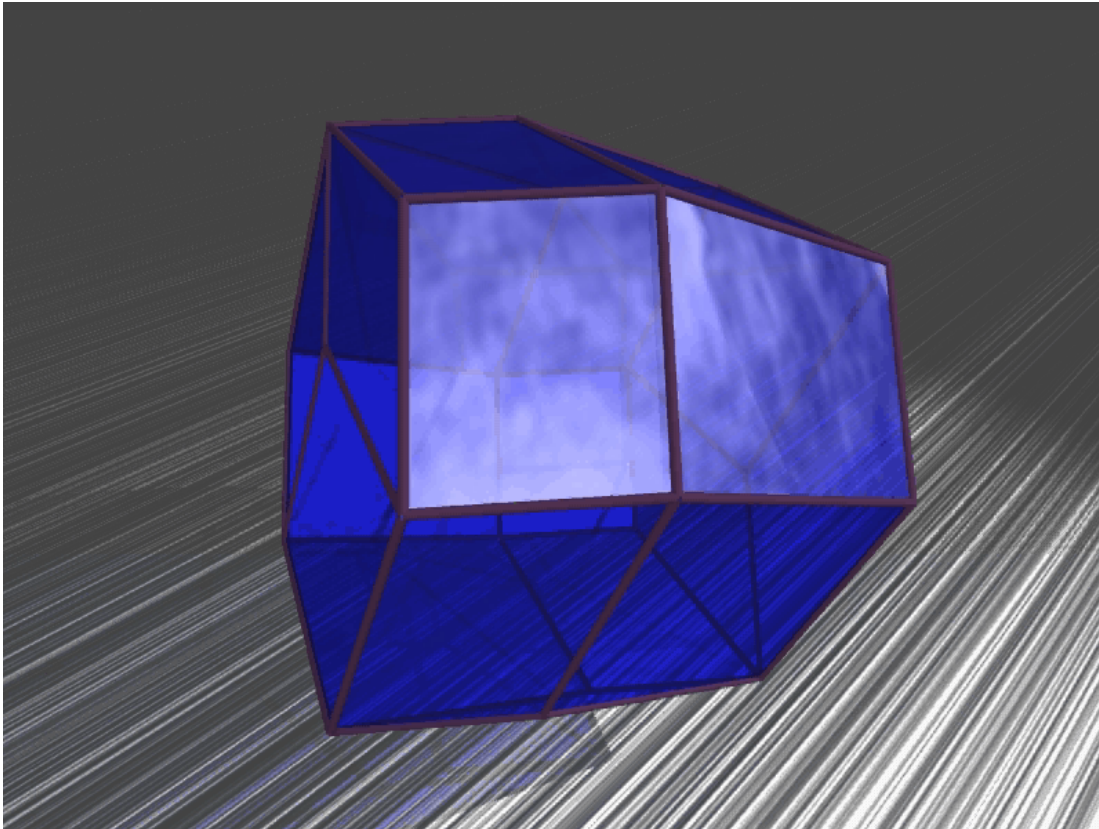
Longitudinal axis of Boeing 747

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Projection



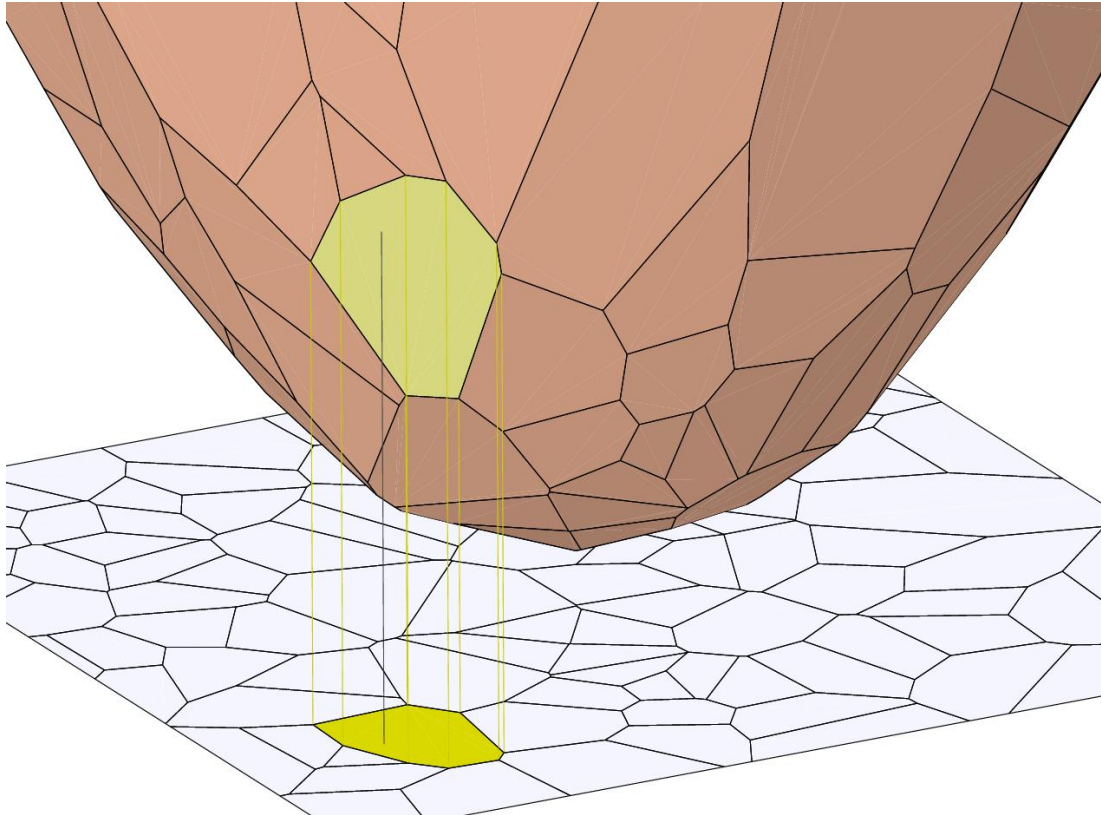
Projection of 5-dimensional hypercube

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Voronoi diagram



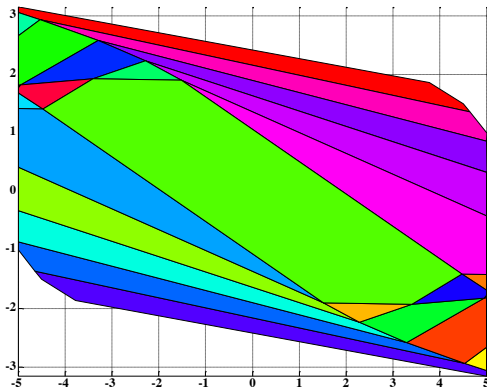
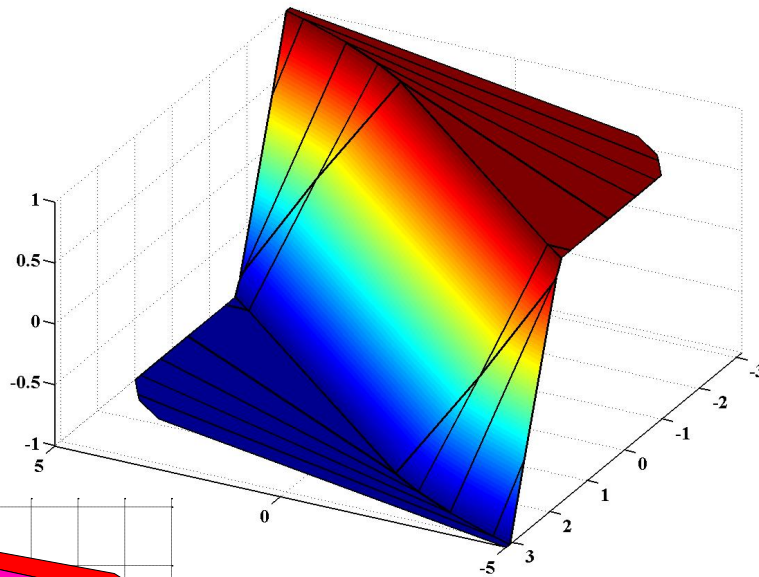
$$R(x_i) := \{x \mid \|x - x_i\| \leq \|x - x_j\|, \forall j \neq i\}$$

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Explicit MPC



Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Convex

Linear
Complementarity
Problem

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

Parametric MILP
Parametric MIQP

*(Bimatrix games
Portfolio selection
Option pricing
Energy markets
Structural mech)*

Explicit MPC:

- Hybrid systems
- PWA systems
- Max-plus algebra
- MLD systems
- ...

Convex

Linear
Complementarity
Problem

Non-Convex

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Geometric Problems Solvable by Parametric Linear Complementary Programming

This lecture covers the
'convex' class
(LCPs with sufficient
matrices)

Convex

Linear
Complementarity
Problem

Parametric LP
Parametric QP
Projection
Affine map
Minkowski sum
Convex hull
Redundancy Elim
Voronoi/Delaunay
...

Explicit MPC
Dynamic Prog
Theorem Proving
Robotic Gripping
...

Conversion of pQP to pLCP

Parametric quadratic
optimization problem:

$$\begin{aligned} J^*(x) &:= \min_u \frac{1}{2} u^T Q u + (F x + f)^T u \\ \text{s.t. } & G u \geq E x + e \\ &u \geq 0 \end{aligned}$$



KKT Conditions:

$$Q u + F x + f - G^T \lambda - \nu = 0$$

Stationarity

$$-s + G u = E x + e, \quad u \geq 0$$

Primary feasibility

$$\lambda, \nu \geq 0$$

Dual feasibility

$$\nu^T u = 0, \quad \lambda^T s = 0$$

Complementarity

Note: Quadratic program is in slightly different form to make the derivation of the LCP simpler. This is always possible through a simple change of variables.

Conversion of pQP to pLCP

KKT Conditions:

→	$Qu + Fx + f - G^T \lambda - \nu = 0$	Stationarity
→	$-s + Gu = Ex + e, u \geq 0$	Primary feasibility
	$\lambda, \nu \geq 0$	Dual feasibility
	$\nu^T u = 0, \lambda^T s = 0$	Complementarity

Stationarity

Primal feasibility

Primal and dual feasibility

Complementarity

→	$\begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} Q & -G^T \\ G & 0 \end{bmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{bmatrix} F \\ -E \end{bmatrix} x + \begin{bmatrix} f \\ -e \end{bmatrix}$
	$\nu, s, u, \lambda \geq 0$
	$\nu^T u = s^T \lambda = 0$

Conversion of pQP to pLCP

KKT Conditions:

$$\begin{aligned}
 &\rightarrow Qu + Fx + f - G^T \lambda - \nu = 0 && \text{Stationarity} \\
 &\rightarrow -s + Gu = Ex + e, \quad u \geq 0 && \text{Primary feasibility} \\
 &\quad \lambda, \nu \geq 0 && \text{Dual feasibility} \\
 &\quad \nu^T u = 0, \quad \lambda^T s = 0 && \text{Complementarity}
 \end{aligned}$$

Stationarity

Primal feasibility

Primal and dual feasibility

Complementarity

$$\begin{aligned}
 &\rightarrow \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} Q & -G^T \\ G & 0 \end{bmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{bmatrix} F \\ -E \end{bmatrix} x + \begin{bmatrix} f \\ -e \end{bmatrix} \\
 &\quad \nu, s, u, \lambda \geq 0 \\
 &\quad \nu^T u = s^T \lambda = 0
 \end{aligned}$$

Standard pLCP:

$$\begin{aligned}
 &Iw - Mz = Qx + q \\
 &w, z \geq 0, \quad w^T z = 0
 \end{aligned}$$

Note: Optimizer is linear transform of pLCP solution $u^*(x) = \begin{bmatrix} I & 0 \end{bmatrix} z(x)$

Parametric Linear Complementarity

Parametric Linear Complementarity Problem

Given matrices M , q and Q , find functions $w(x)$, $z(x)$ such that

$$w - Mz = q + Qx$$

$$w^T z = 0$$

$$w, z \geq 0$$

- Recall:
 - MPC control law is a linear transformation of z
 - Can represent ‘standard’ MPC problems as pLCPs
- Key questions:
 - What is the domain of w and z ?
 - What class of functions are w and z ? (and hence the control law)
 - How can we efficiently compute w and z ?

Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Example

Geometry of the LCP

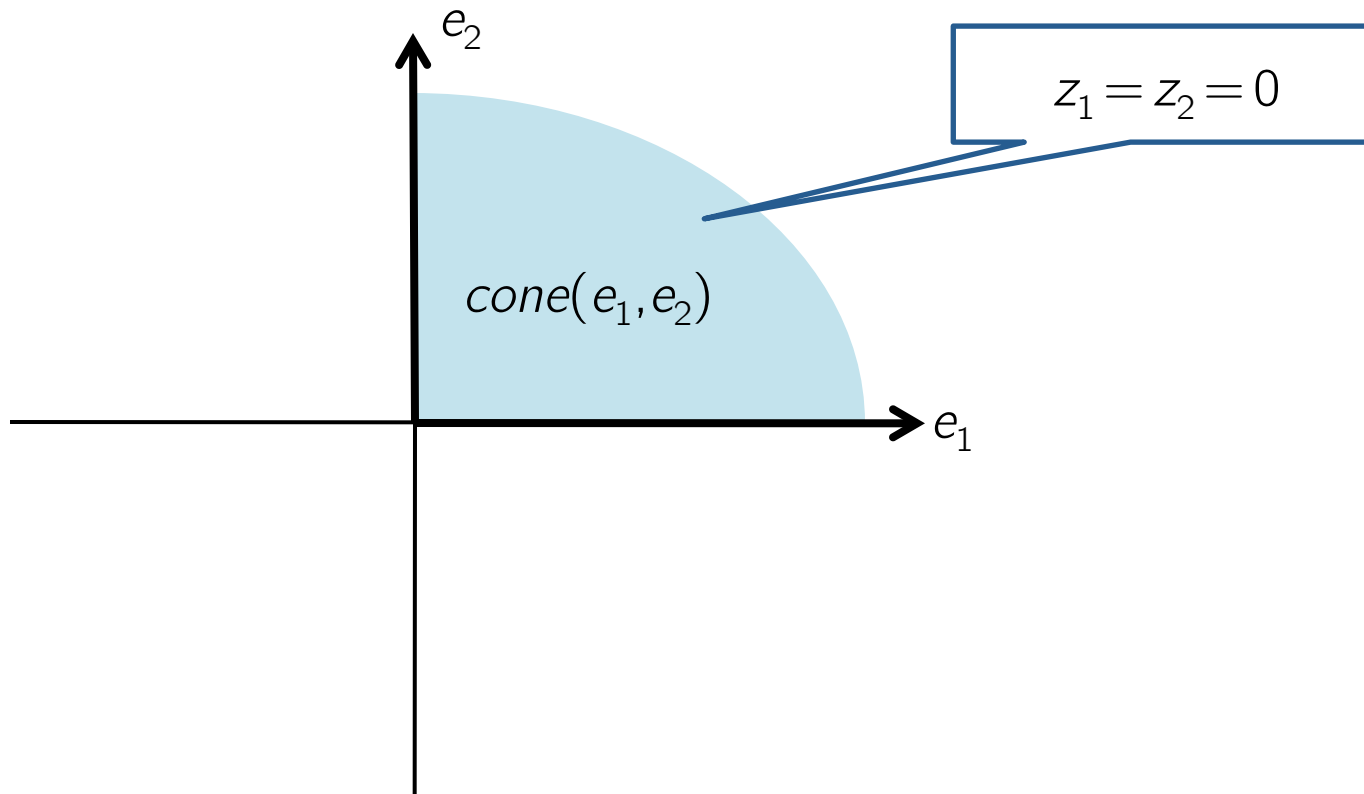
LCP feasibility conditions

1. $w^T z = 0, w, z \geq 0$ $\rightarrow$ either w_i or z_i is zero for all i
2. $Iw - Mz = q$ $\rightarrow$ q is in the cone of non-zero variables

Geometry of the LCP

LCP feasibility conditions

1. $w^T z = 0, w, z \geq 0$ $\rightarrow$ either w_i or z_i is zero for all i
2. $Iw - Mz = q$ $\rightarrow$ q is in the cone of non-zero variables



Geometry of the LCP

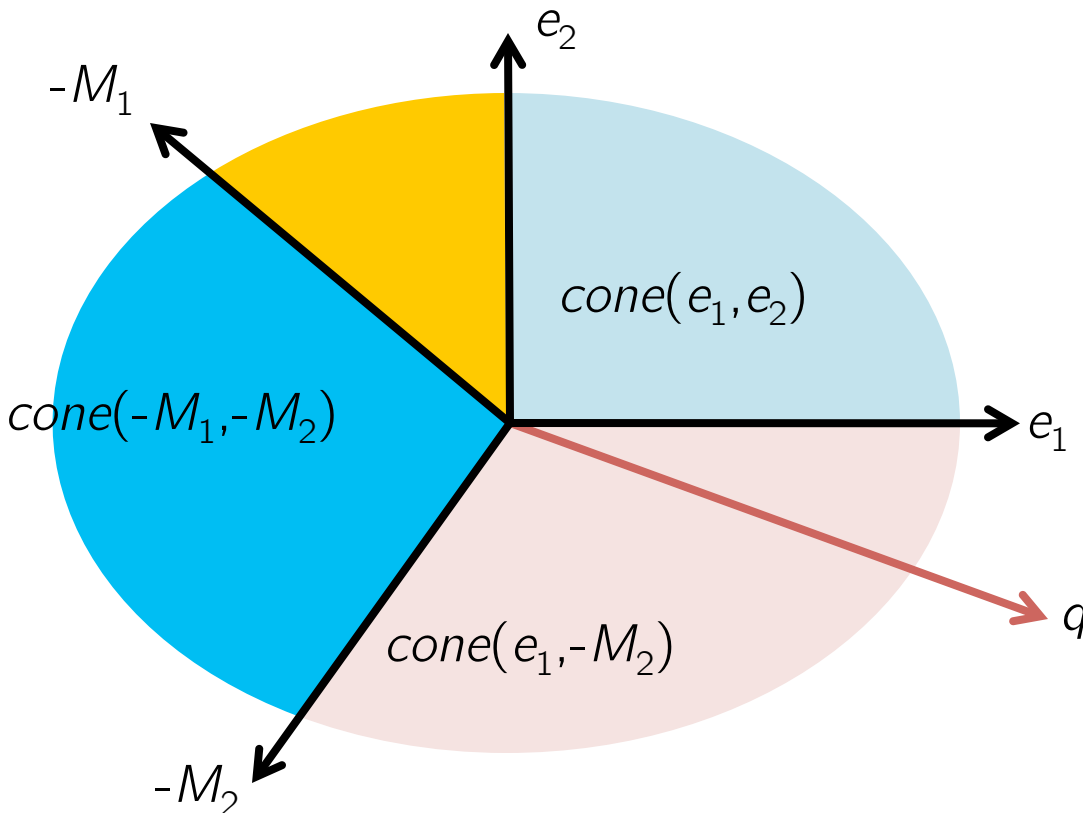
LCP feasibility conditions

1. $w^T z = 0, w, z \geq 0$

→ either w_i or z_i is zero for all i

2. $Iw - Mz = q$

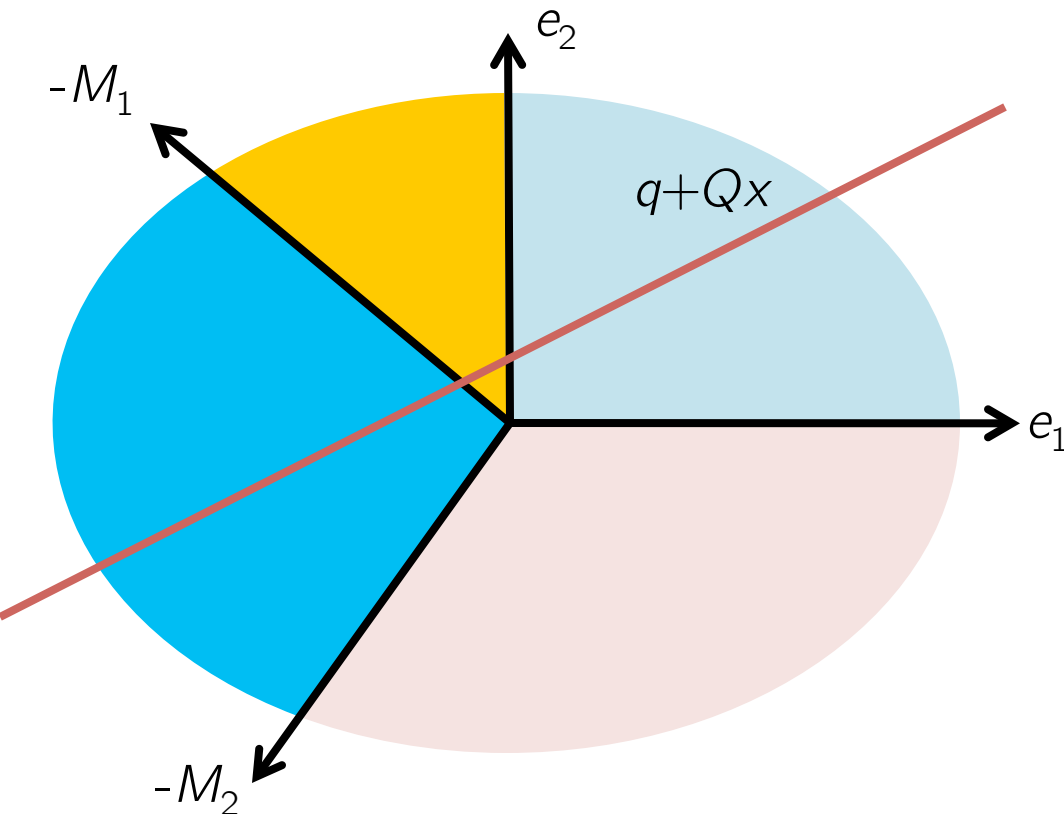
→ q is in the cone of non-zero variables



Goal: Find cone containing q

Geometry of the Parametric LCP

Goal: Find all cones intersecting $q + Qx$



$$\begin{aligned} w(x) - Mz(x) &= q + Qx \\ w(x)^T z(x) &= 0 \\ w(x), z(x) &\geq 0 \end{aligned}$$

Re-visit Simple Example

Parametric QP

$$\begin{aligned} f^*(x) = \min_z \quad & \frac{1}{2}z^2 + 2xz \\ \text{s.t.} \quad & z \geq x - 1 \\ & z \geq 0 \end{aligned}$$

KKT Conditions

$\nabla_z \mathcal{L} = z + 2x - \lambda - \nu = 0$	Stationarity
$x - 1 - z + s = 0, \quad s, z \geq 0$	Primal feasibility
$\lambda, \nu \geq 0$	Dual feasibility
$\lambda(z - x - 1) = \nu z = 0$	Complementarity

Re-visit Simple Example

Parametric QP

$$f^*(x) = \min_z \frac{1}{2}z^2 + 2xz$$

s.t. $z \geq x - 1$
 $z \geq 0$

KKT Conditions

$$\begin{aligned} \nabla_z \mathcal{L} &= z + 2x - \lambda - \nu = 0 && \leftarrow \text{Stationarity} \\ x - 1 - z + s &= 0, \quad s, z \geq 0 && \leftarrow \text{Primal feasibility} \\ \lambda, \nu &\geq 0 && \leftarrow \text{Dual feasibility} \\ \lambda(z - x - 1) &= \nu z = 0 && \leftarrow \text{Complementarity} \end{aligned}$$

Equivalent pLCP

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} z \\ \lambda \end{pmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} \nu \\ s \end{pmatrix}^T \begin{pmatrix} z \\ \lambda \end{pmatrix} = 0 \quad \nu, s, z, \lambda \geq 0$$

Re-visit Simple Example

Parametric QP

$$f^*(x) = \min_z \frac{1}{2}z^2 + 2xz$$

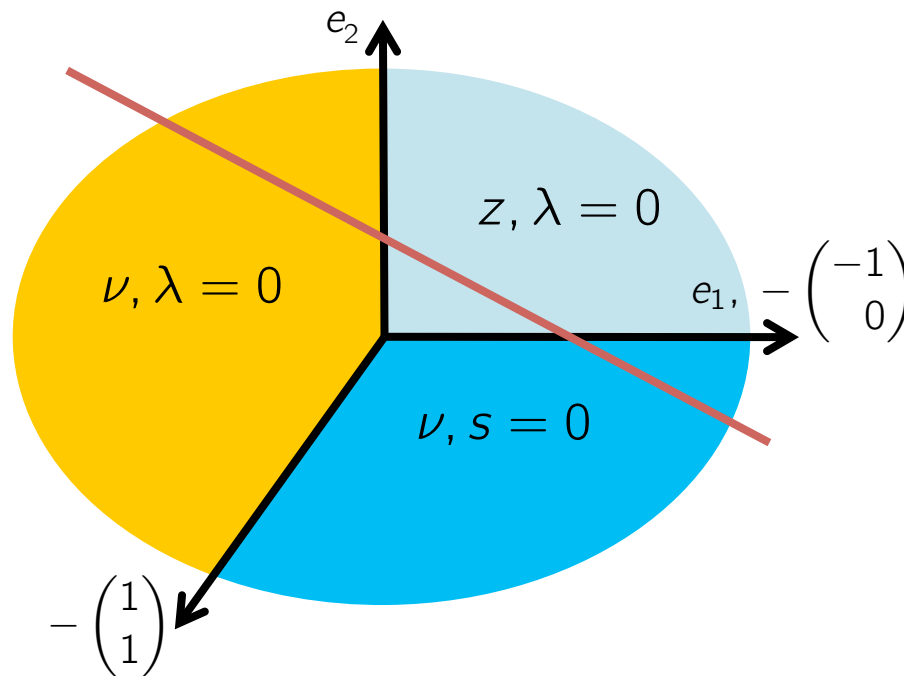
$$\text{s.t. } z \geq x - 1$$

$$z \geq 0$$

Parametric Linear Complementarity Problem

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} z \\ \lambda \end{pmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} \nu \\ s \end{pmatrix}^T \begin{pmatrix} z \\ \lambda \end{pmatrix} = 0 \quad \nu, s, z, \lambda \geq 0$$



Re-visit Simple Example

Parametric QP

$$f^*(x) = \min_z \frac{1}{2}z^2 + 2xz$$

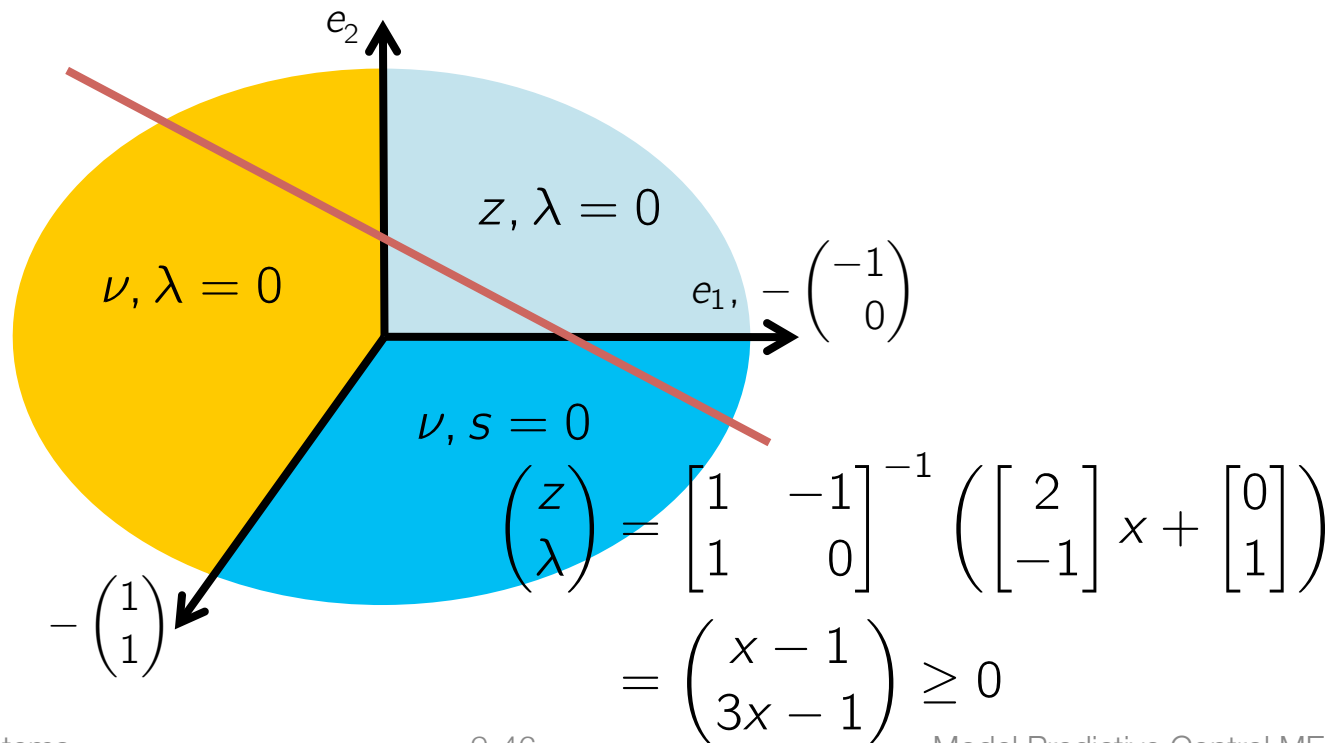
$$\text{s.t. } z \geq x - 1$$

$$z \geq 0$$

Parametric Linear Complementarity Problem

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} z \\ \lambda \end{pmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} \nu \\ s \end{pmatrix}^T \begin{pmatrix} z \\ \lambda \end{pmatrix} = 0 \quad \nu, s, z, \lambda \geq 0$$



Re-visit Simple Example

Parametric QP

$$f^*(x) = \min_z \frac{1}{2}z^2 + 2xz$$

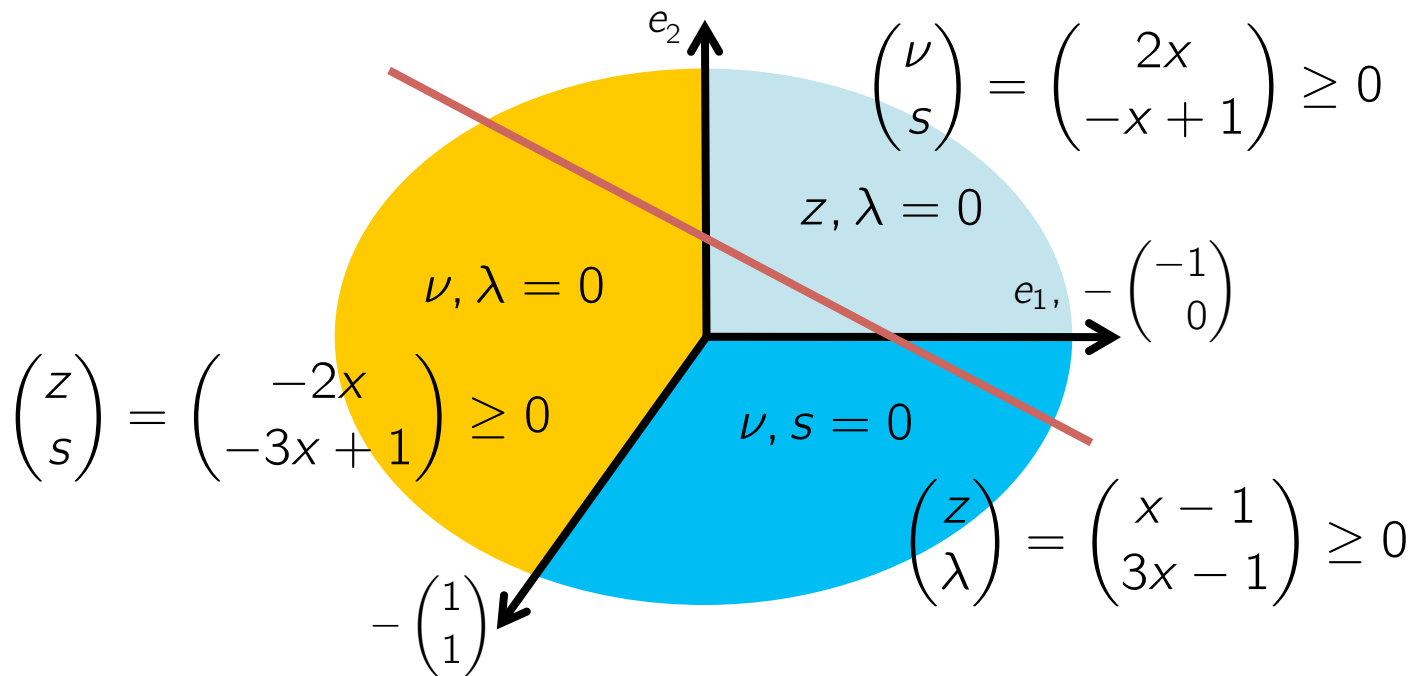
$$\text{s.t. } z \geq x - 1$$

$$z \geq 0$$

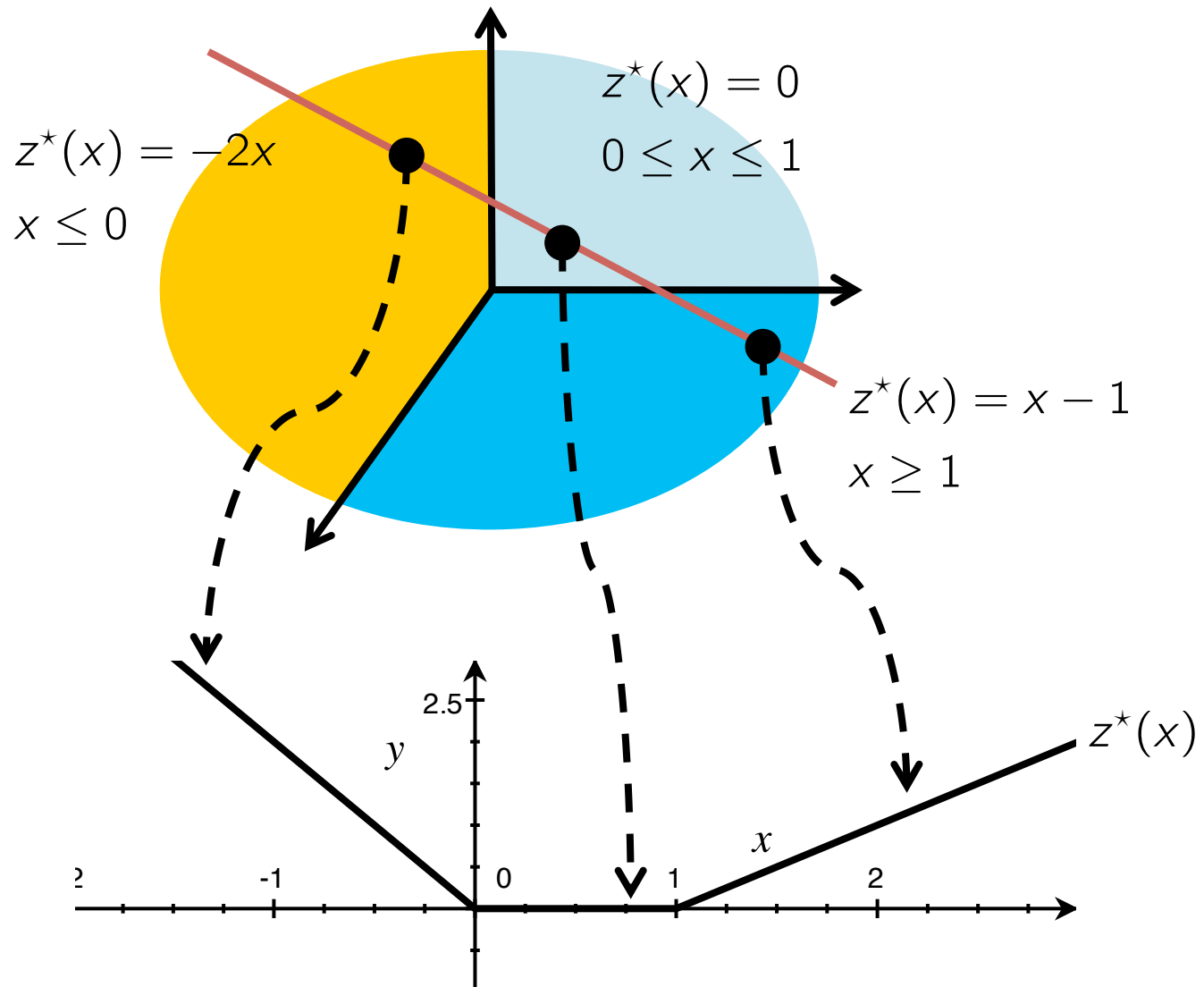
Parametric Linear Complementarity Problem

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \nu \\ s \end{pmatrix} - \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} z \\ \lambda \end{pmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} \nu \\ s \end{pmatrix}^T \begin{pmatrix} z \\ \lambda \end{pmatrix} = 0 \quad \nu, s, z, \lambda \geq 0$$



Re-visit Simple Example



Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Examples

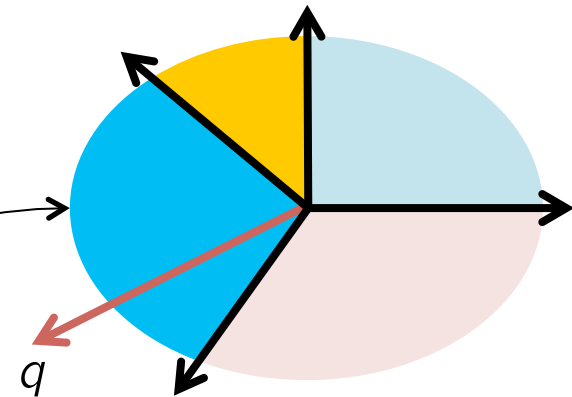
Algebra of the LCP

- Define the matrix $A := [I \ -M] \in \mathbb{R}^{n \times 2n}$
- The index set $B \subset \{1, \dots, 2n\}$ is a *basis* if
 - B contains n elements $|B| = n$
 - Columns of A indexed by B are full-rank $\text{rank } A_B = n$
- B is a complementary basis if

$$i \in B \Leftrightarrow i + n \notin B \text{ for all } i \in \{1, \dots, n\}$$

i.e., the columns satisfy the complementarity conditions $w^T z = 0, \ w, z \geq 0$
- Complementary bases define complementary cones

$$\mathcal{C}(B) := \{q \in \mathbb{R}^n \mid A_B^{-1} q \geq 0\}$$



Basis B 'solves' the LCP (M, q) if and only if $q \in \mathcal{C}(B)$

Solution Properties

What are $w(x)$ and $z(x)$ in $\mathcal{C}(B)$?

- $w^T z = 0 \quad \rightarrow \quad B \text{ is a complementary basis}$
- $A \begin{bmatrix} w \\ z \end{bmatrix} = q + Qx \quad \rightarrow \quad \begin{bmatrix} w \\ z \end{bmatrix}_B = A_B^{-1}(q + Qx)$
- $w, z \geq 0 \quad \rightarrow \quad A_B^{-1}(q + Qx) \geq 0$

Defines a polyhedral *critical region* in which B is the solution

$$CR(B) := \{x \mid A_B^{-1}(q + Qx) \geq 0\}$$

Solution is *affine* in each complementary cone

$$\begin{bmatrix} w \\ z \end{bmatrix}_B = A_B^{-1}(q + Qx)$$

Control law is piecewise affine defined over a polyhedral partition!

Simple Solution Algorithm

Simple Parametric LCP Solver

- For each complementary basis B
 - If $CR(B)$ is non-empty

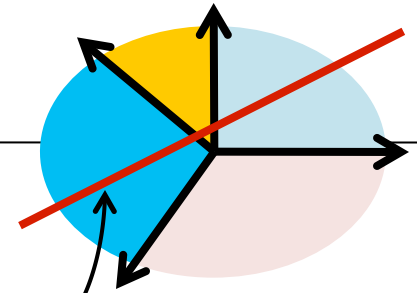
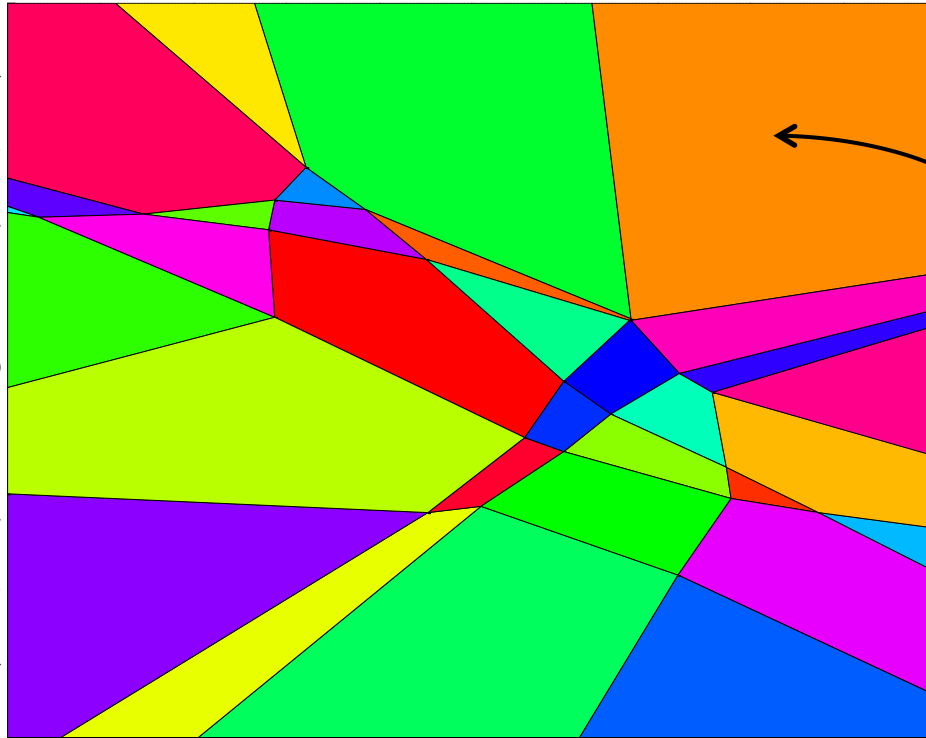
$$\begin{bmatrix} w(x) \\ z(x) \end{bmatrix}_B = A_B^{-1}(q + Qx) \quad \text{for all } x \in CR(B)$$

- Works for all pLCPs (including ‘non-convex’ examples)
 - Testing non-emptiness of a polyhedron is easy (Linear Program)
- Consider MPC problem with: 2 states, 2 inputs, horizon of 5 and upper/lower bounds on states and inputs
 - $2^{20} = 1'048'576$ possible bases!
- We need a more efficient algorithm!

Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Examples

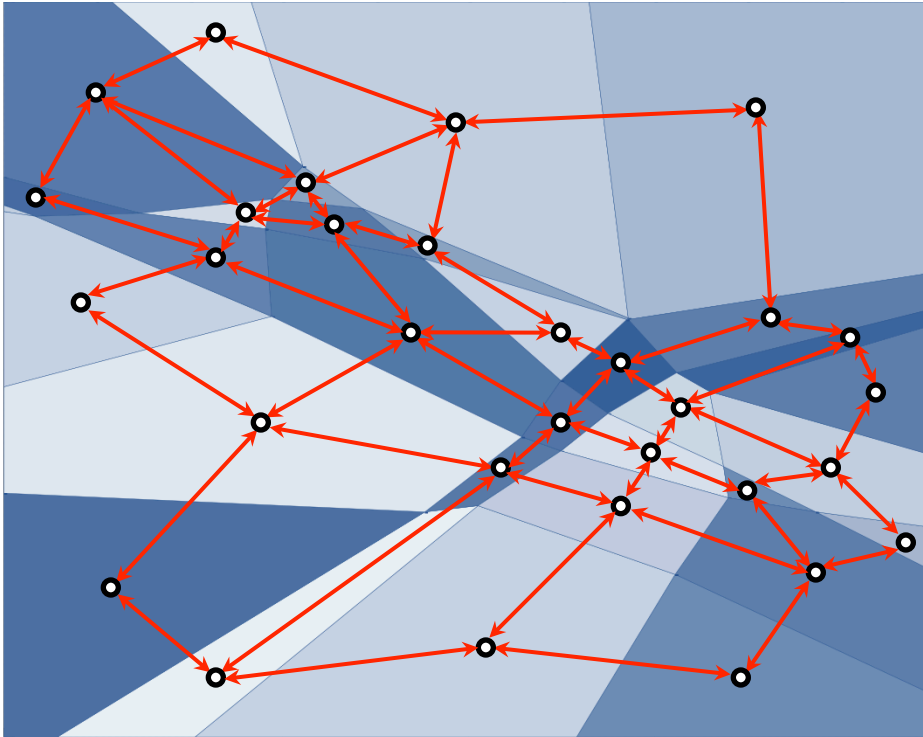
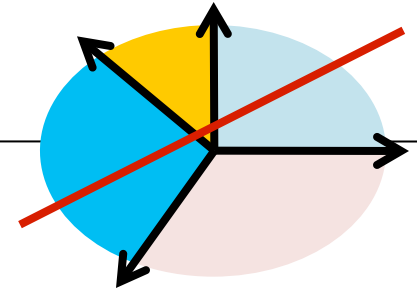
Efficient Enumeration Algorithm



Critical
Regions

P-matrix example.
Figure generated using MPT toolbox

Efficient Enumeration Algorithm



Define graph $\mathcal{G}$

Vertices: Non-empty CRs

Edges: Adjacent CRs

Find all critical regions by
standard graph enumeration

When does this work?

- Connected graph
- Non-overlapping critical regions

Well behaving matrix classes

Definition : Sufficient matrix

A matrix $M \in \mathbb{R}^{n \times n}$ is called *column sufficient* if it satisfies the implication

$$[z_i(Mz)_i \leq 0 \text{ for all } i] \implies [z_i(Mz)_i = 0 \text{ for all } i] .$$

The matrix M is called *row sufficient* if its transpose is column sufficient. If M is both column and row sufficient, then it is called *sufficient*.

- Weaker form of semi-definite matrices

Proposition

Positive semi-definite matrices are sufficient.

*Note that this applies to non-symmetric PSD matrices too

Convex quadratic programs give rise to LCPs with sufficient matrices

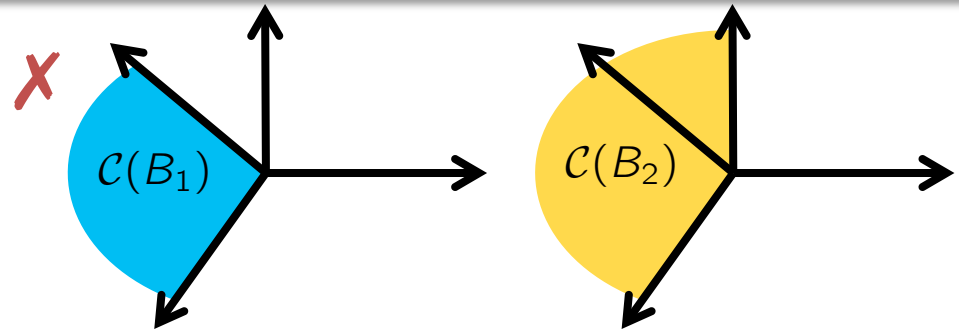
LCPs with Sufficient Matrices

Proposition

If M is a sufficient matrix, then the relative interiors of any two distinct complementary cones are disjoint.

Cones cannot overlap:

⇒ **Solution is unique!**

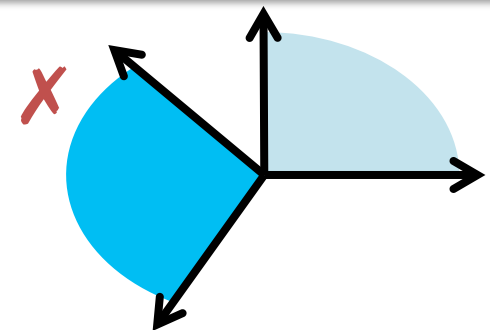


Proposition

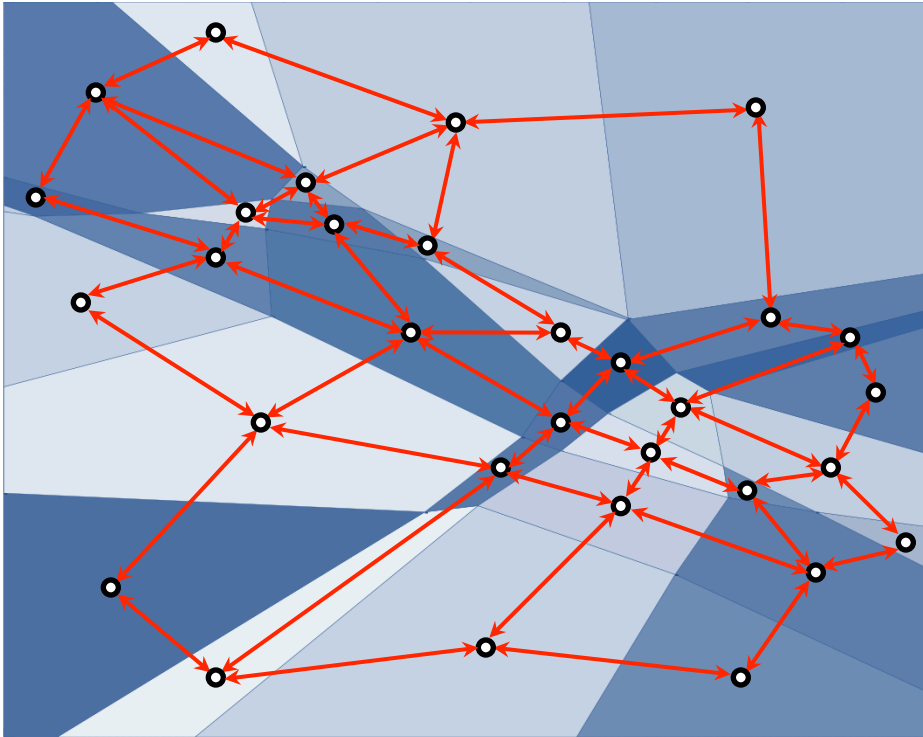
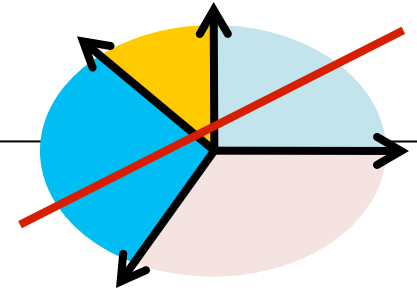
If M is a sufficient matrix, then the union of all complementary cones $K(M)$ is a convex polyhedral cone $K(M) = \text{cone}([I \ -M])$.

Domain is connected:

⇒ **Neighbour graph is connected!**



Efficient Enumeration Algorithm



Define graph $\mathcal{G}$

Vertices: Non-empty CRs

Edges: Adjacent CRs

Find all critical regions by
standard graph enumeration

When does this work?

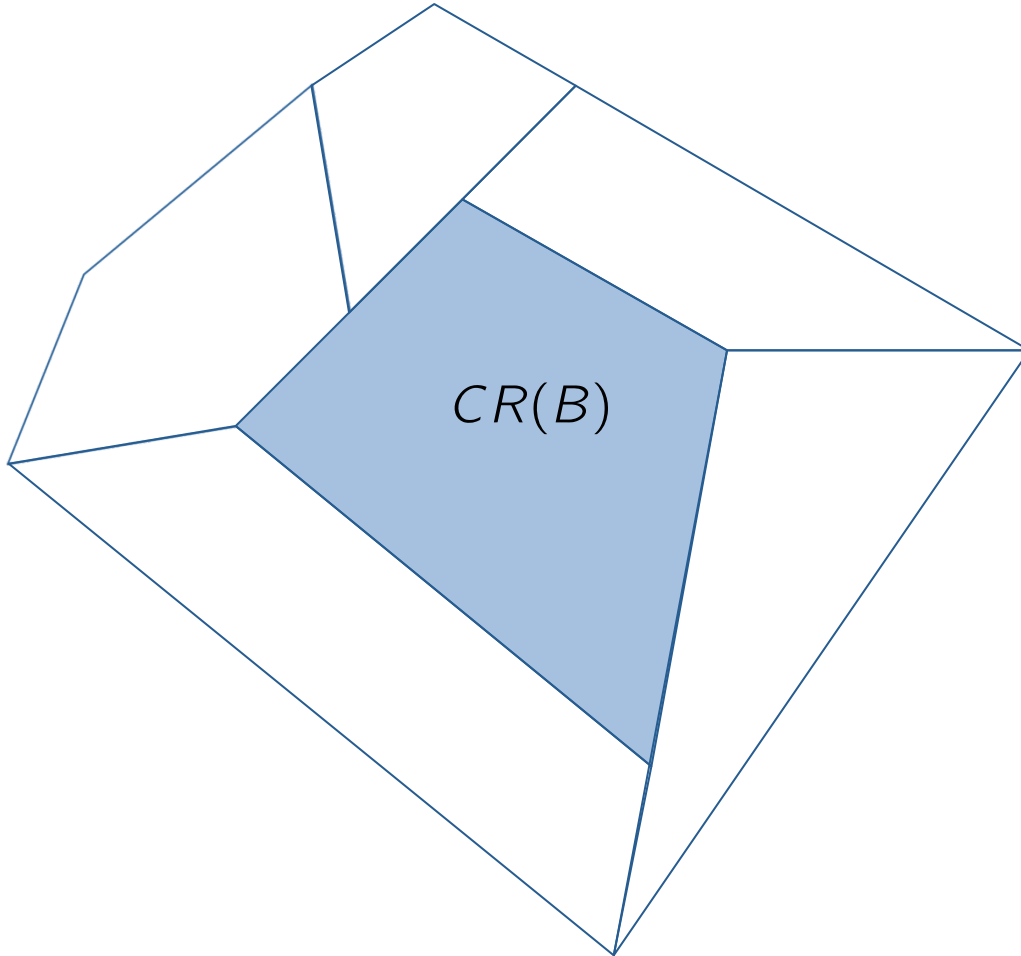
- Connected graph ✓
- Non-overlapping critical regions ✓



If we can compute the
neighbours in poly-time,
then the enumeration
algorithm is polynomial

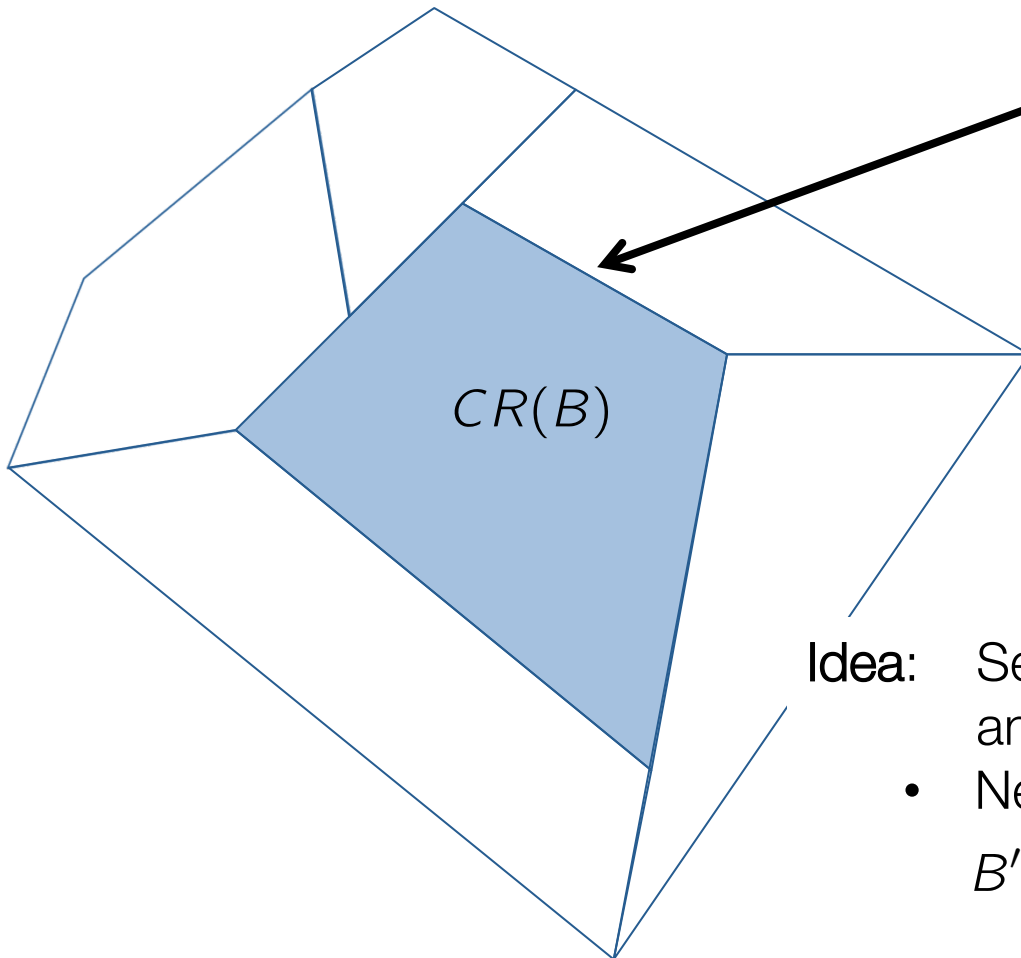
Computing adjacent regions : The idea

Key operation : Find neighbours of $CR(B) := \{x \mid A_B^{-1}(q + Qx) \geq 0\}$



Computing adjacent regions : The idea

Key operation : Find neighbours of $CR(B) := \{x \mid A_B^{-1}(q + Qx) \geq 0\}$



What happens here?

$$\begin{bmatrix} w \\ z \end{bmatrix}_{B_i} = (A_B^{-1}(q + Qx))_i \rightarrow 0$$

- One of the positive variables is going to zero
- Any further and the problem would become infeasible

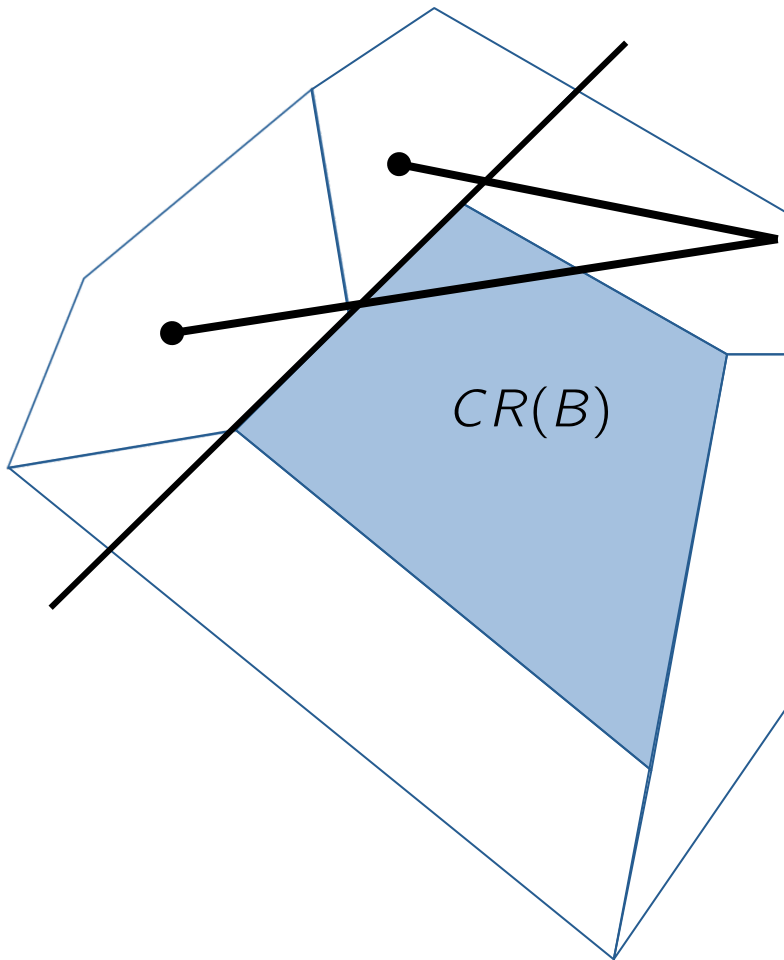
Idea: Set offending variable to zero and allow its complement to increase

- New basis is complementary

$$B' = B \setminus \{i\} \cup \{\bar{i}\}$$

Cost : One linear program to determine if i forms a facet

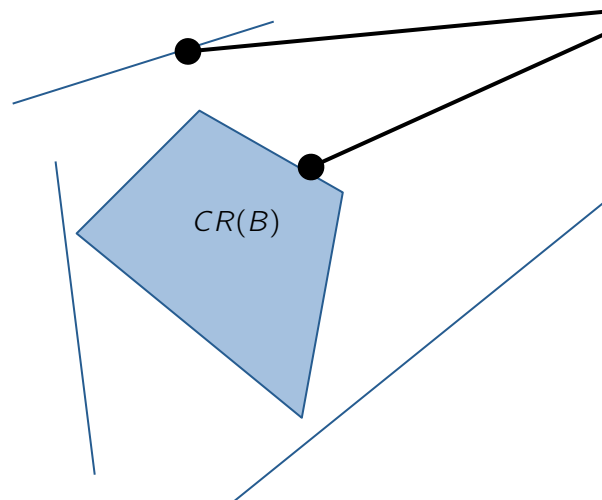
Computing adjacent regions : The complexity



- Multiple regions can neighbour along a single facet
- Requires an *exchange pivot* (those that differ by two elements)
$$B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}$$
- Cost : n linear programs to determine if i, j forms an adjacent region

Computing adjacent regions

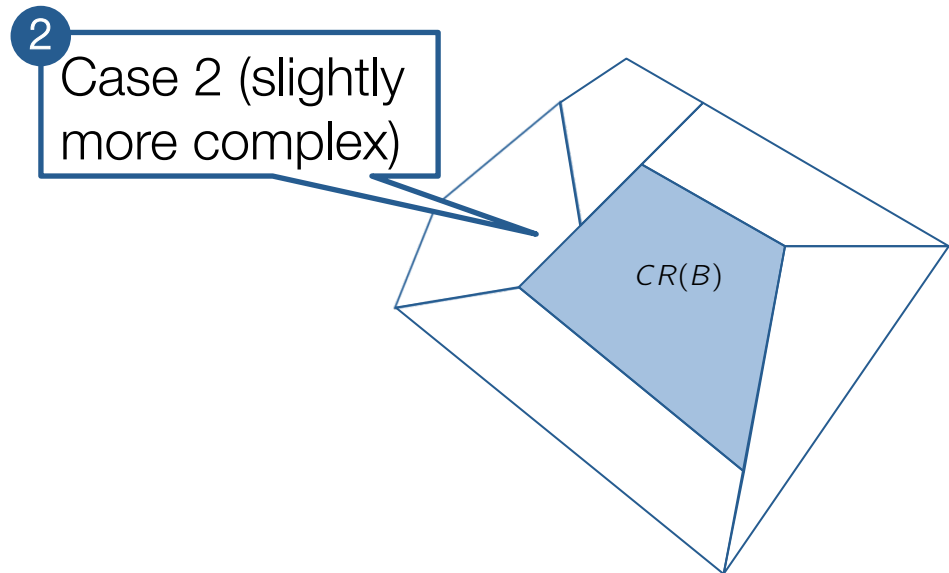
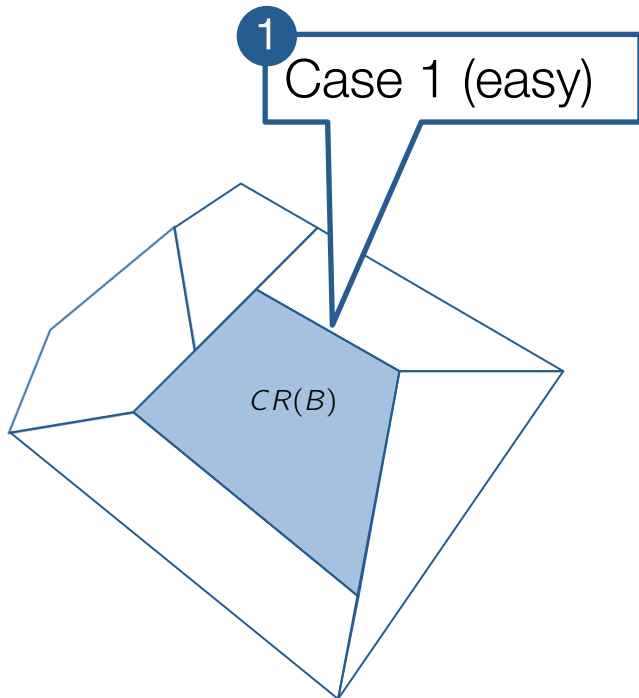
- * for each row $(A_B^{-1}(q + Qx))_i \geq 0$ which forms a facet of $CR(B)$
 - 1 Case 1 : Diagonal pivot
return $B \setminus \{i\} \cup \{\bar{i}\}$
 - 2 Case 2 : Exchange pivot
for each j if $CR(B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}) \cap CR(B) \neq \emptyset$
return $B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}$



* Some rows form facets, some don't.
Telling the difference requires a linear program.

Computing adjacent regions

- * for each row $(A_B^{-1}(q + Qx))_i \geq 0$ which forms a facet of $CR(B)$
 - 1 Case 1 : Diagonal pivot
return $B \setminus \{i\} \cup \{\bar{i}\}$
 - 2 Case 2 : Exchange pivot
for each j if $CR(B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}) \cap CR(B) \neq \emptyset$
return $B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}$



Computing adjacent regions

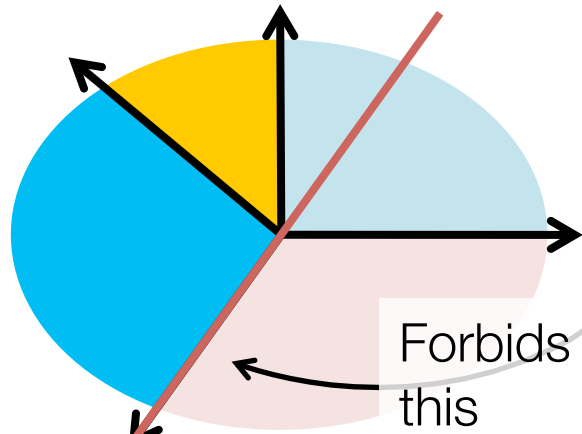
- * for each row $(A_B^{-1}(q + Qx))_i \geq 0$ which forms a facet of $CR(B)$
 - 1 Case 1 : Diagonal pivot
return $B \setminus \{i\} \cup \{\bar{i}\}$
 - 2 Case 2 : Exchange pivot
for each j if $CR(B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}) \cap CR(B) \neq \emptyset$
return $B \setminus \{i, j\} \cup \{\bar{i}, \bar{j}\}$

Worst case : n^2 linear programs

- n = number of variables

Total complexity $\leq$ (Number of regions) $\cdot$ (number of variables)²

The fine print : Degeneracy



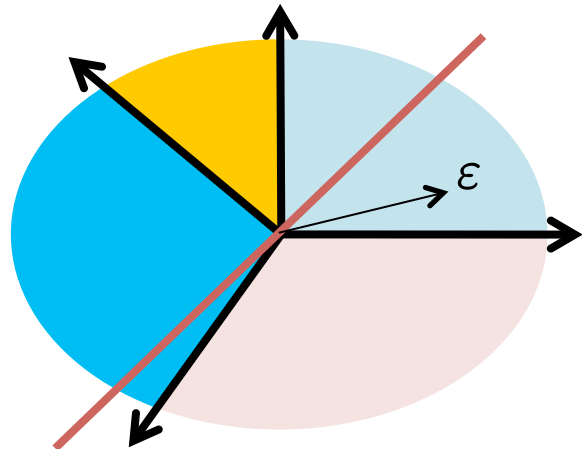
All previous statements rely on S being in *general position*

S intersects $CR(B) \Rightarrow S$ intersects $int(C(B))$

- Multiple solutions for same parameter
- Neighbourhood statements do not hold
 - Enumeration algorithm does not work

Forbids
this

$$S := \{Qx + q, x \in \mathbb{R}^d\}$$



We can ***simulate*** general position artificially

- Adds complexity to the algorithm
- Restores all positive properties

$$S_\epsilon := \{Qx + q + \bar{\epsilon}, x \in \mathbb{R}^d\}$$

Properties of Model Predictive Control Laws

Model Predictive Control

$$J^*(x) = \min x_N^T Q_f x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i$$

$$\text{s.t. } x_0 = x$$

$$x_{i+1} = Ax_i + Bu_i$$

$$Cx_i + Du_i \leq b$$

Theorem

- Feasible set X^* is polyhedral (closed and convex)
- The optimizer $u^*(x) : X^*$ is a piecewise affine function defined over a polyhedral partition
- The pLCP algorithm selects a continuous optimizer

Also applies for convex polyhedral cost functions

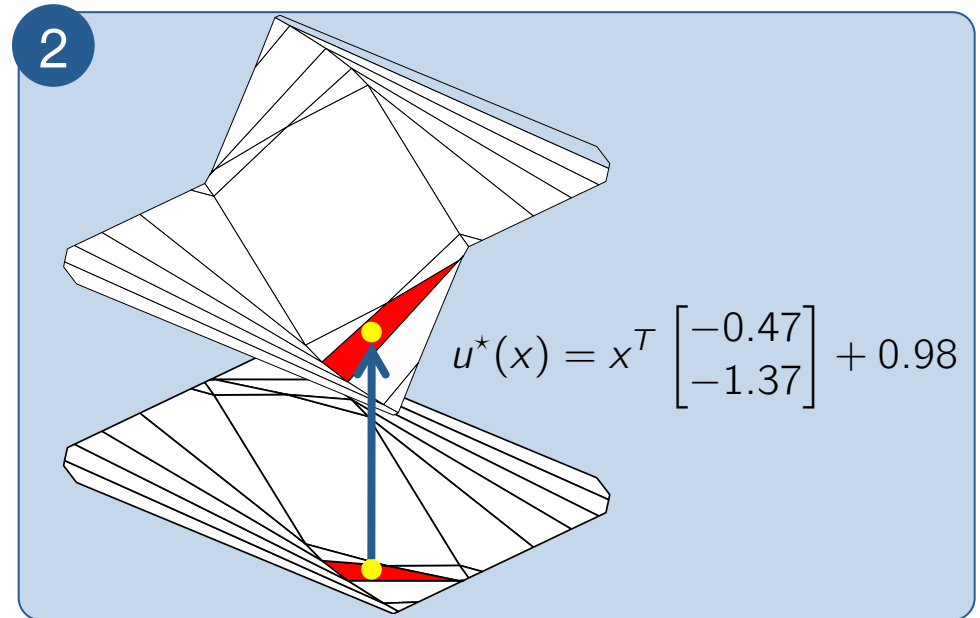
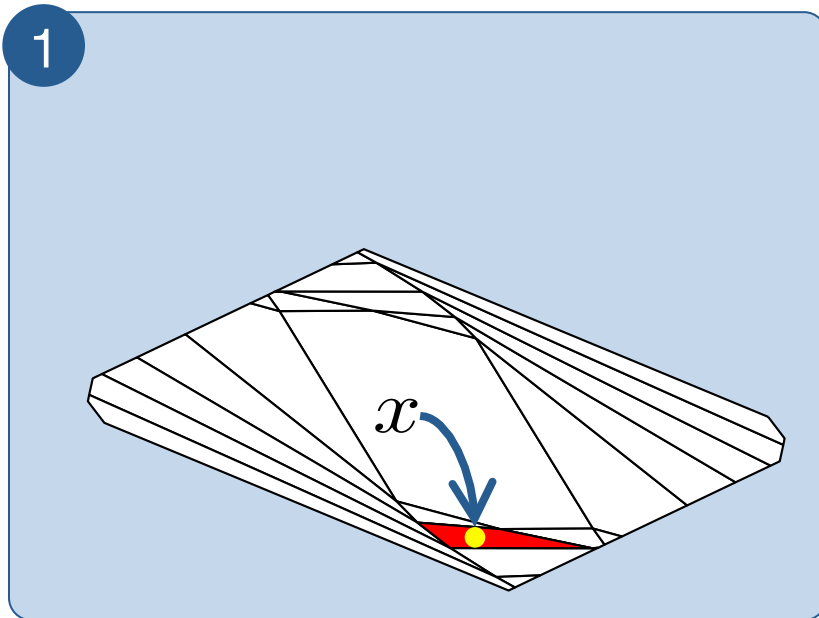
Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Example

Online evaluation : Point location

Calculation of piecewise affine function:

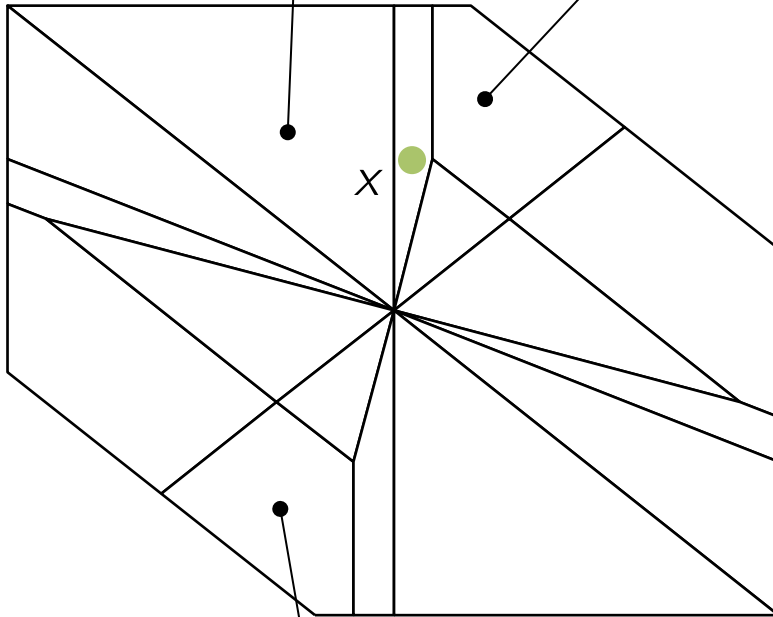
- 1 Point location
- 2 Evaluation of affine function



Point Location – Sequential search

$$CR(B_1) = \{x \mid A_1x + b_1 \leq 0\}$$

$$CR(B_2) = \{x \mid A_2x + b_2 \leq 0\}$$



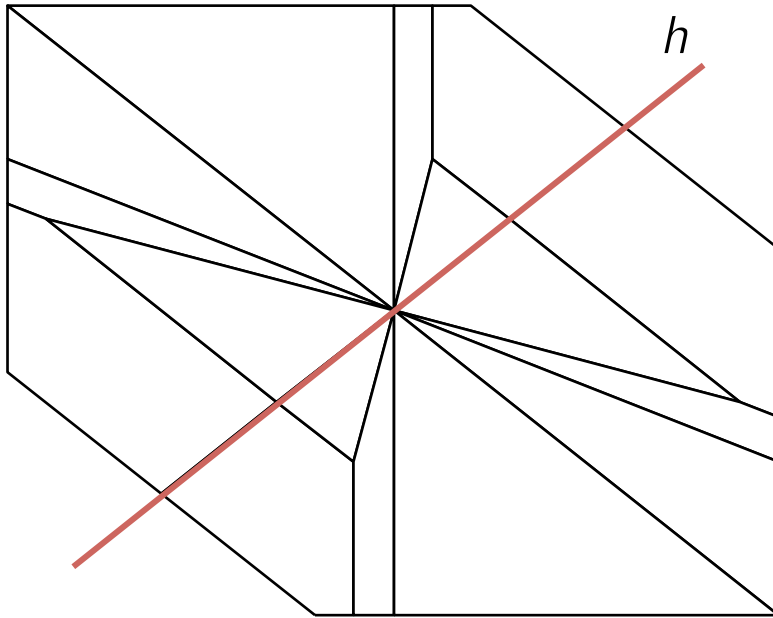
$$CR(B_3) = \{x \mid A_3x + b_3 \leq 0\}$$

Sequential search

```
for each  $i$   
  if  $A_ix + b_i \leq 0$  then  
     $x$  is in region  $i$ 
```

- Very simple
- Linear in number of regions

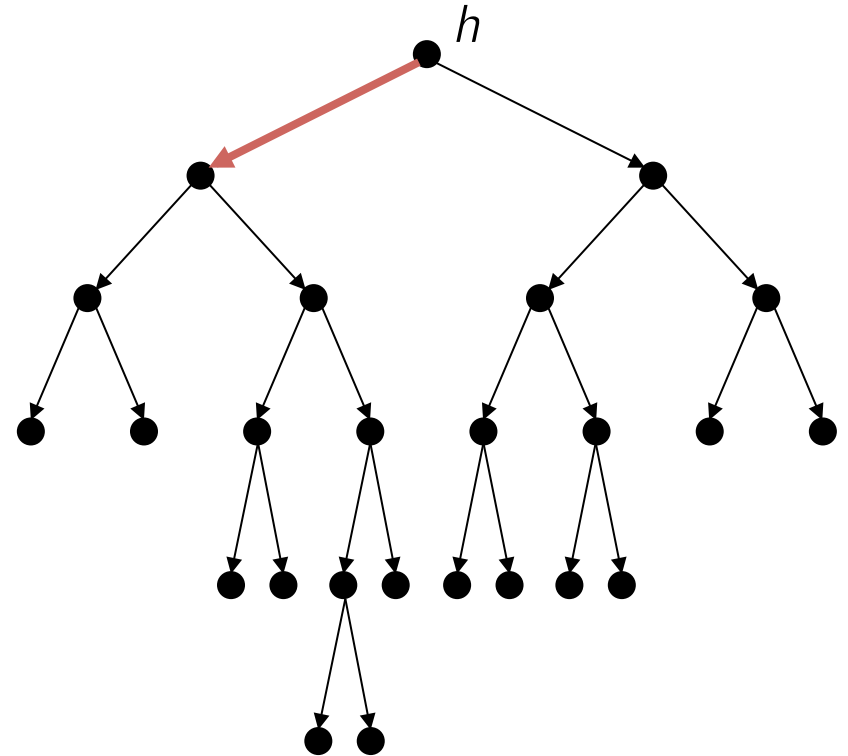
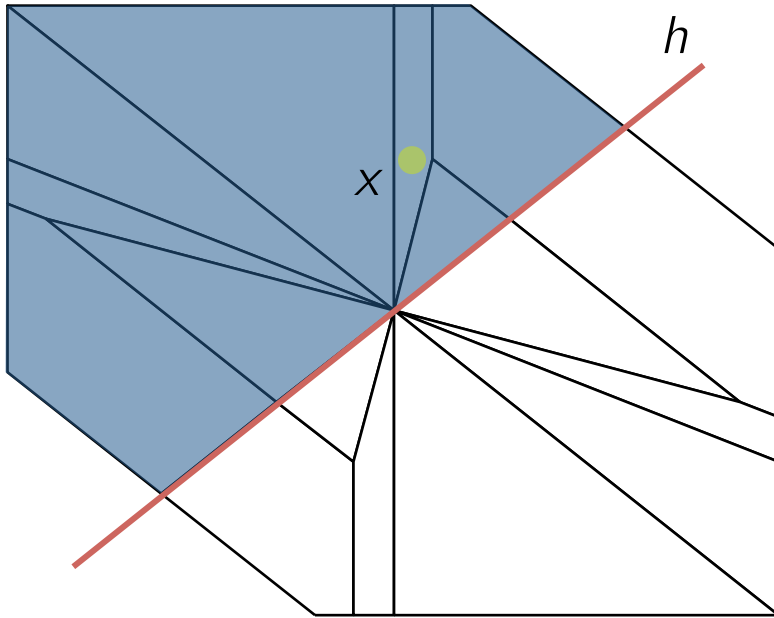
Point Location – Logarithmic search



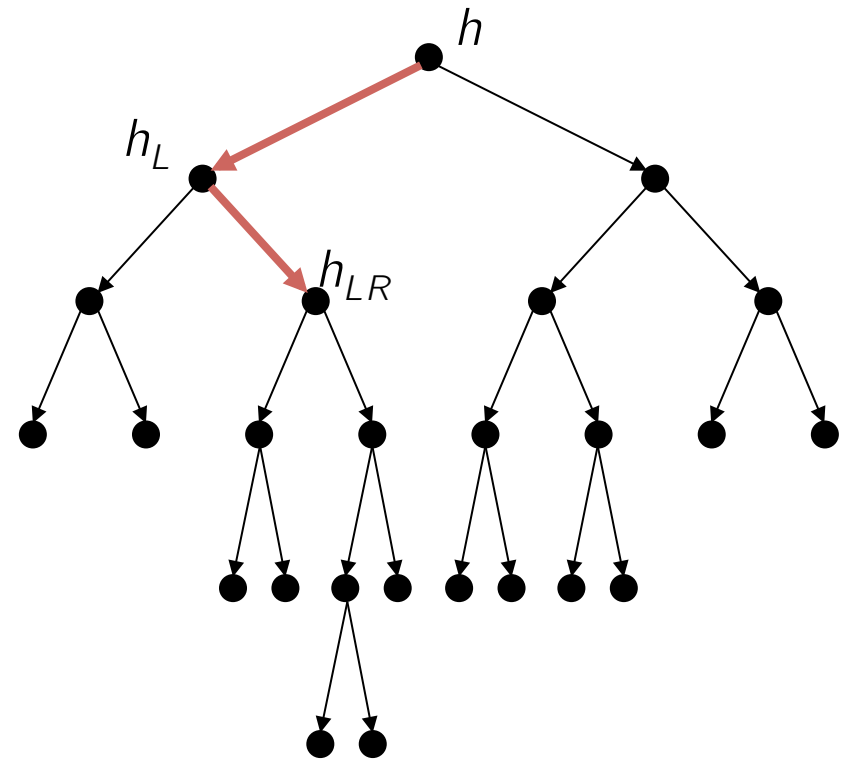
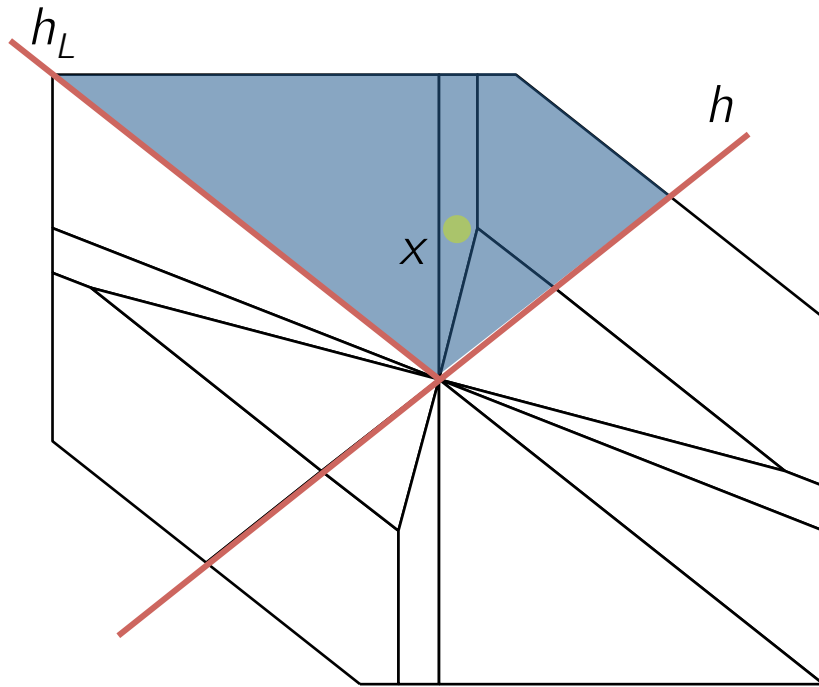
Offline construction of search tree

- Find hyperplane that separates regions into two equal sized sets
- Repeat for left and right sets

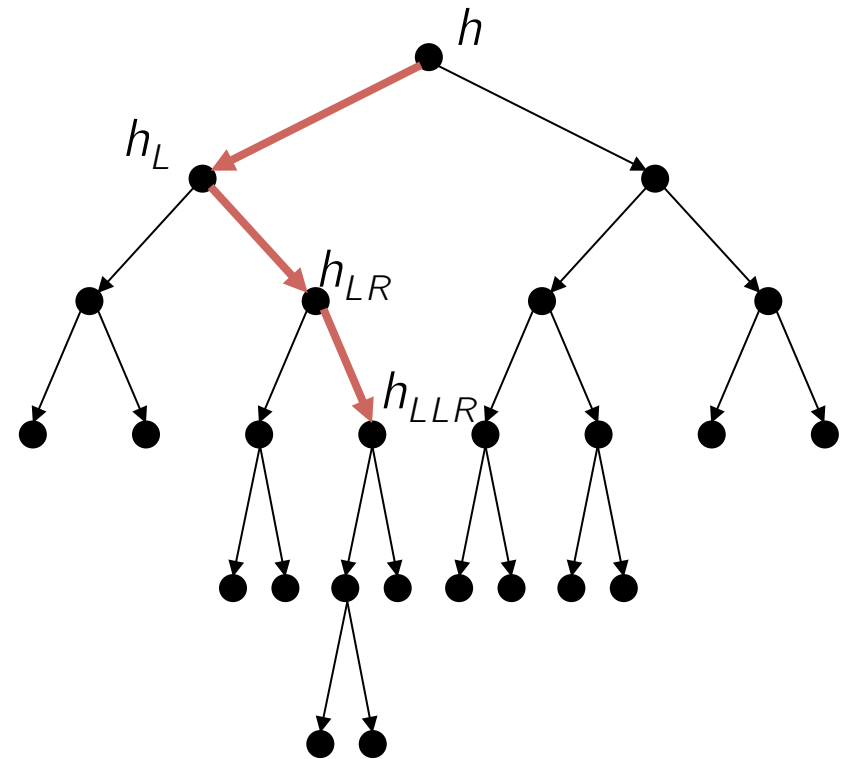
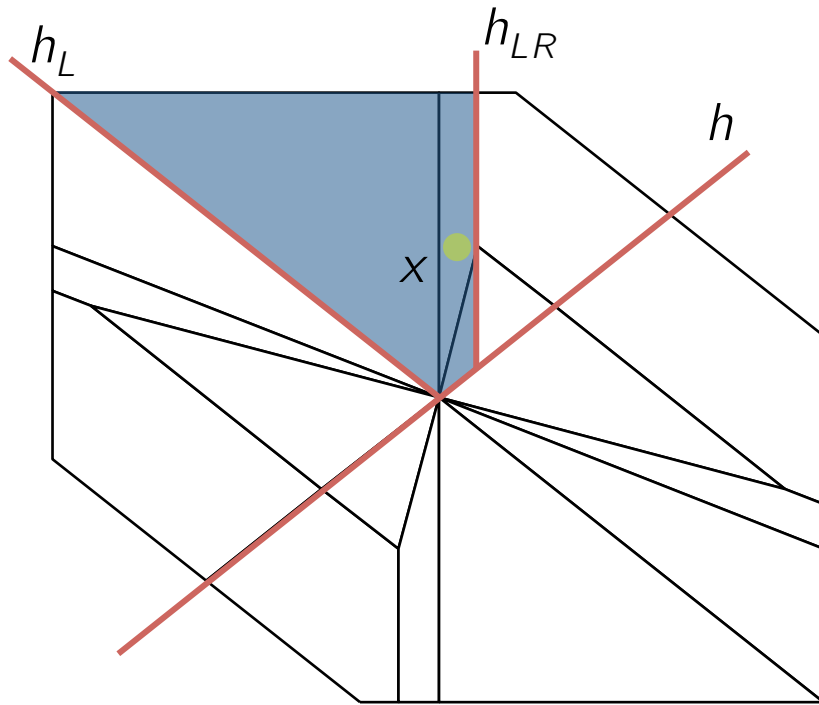
Point Location – Logarithmic search



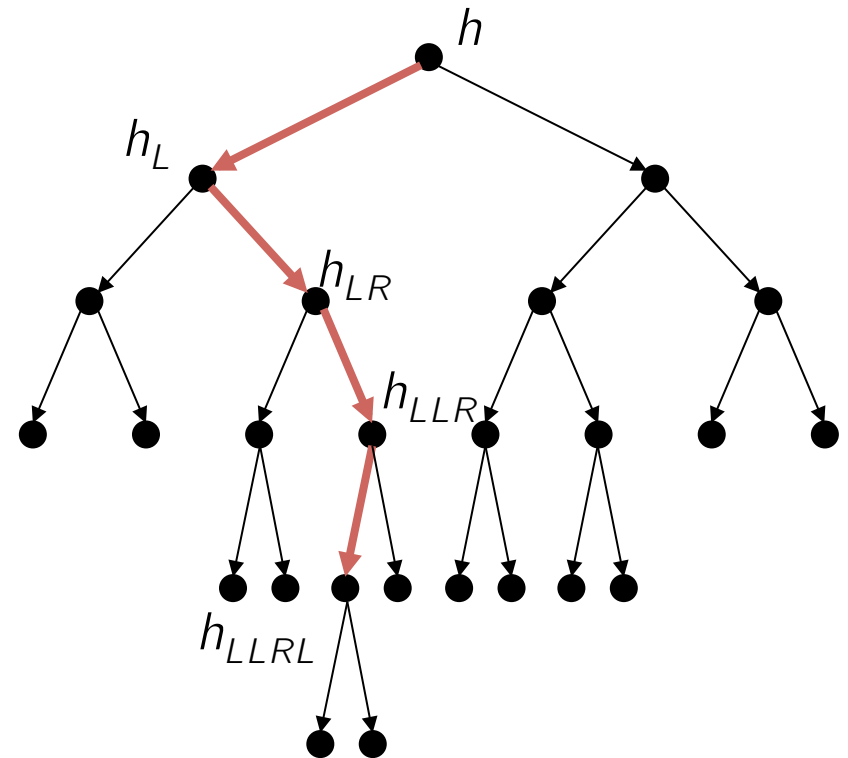
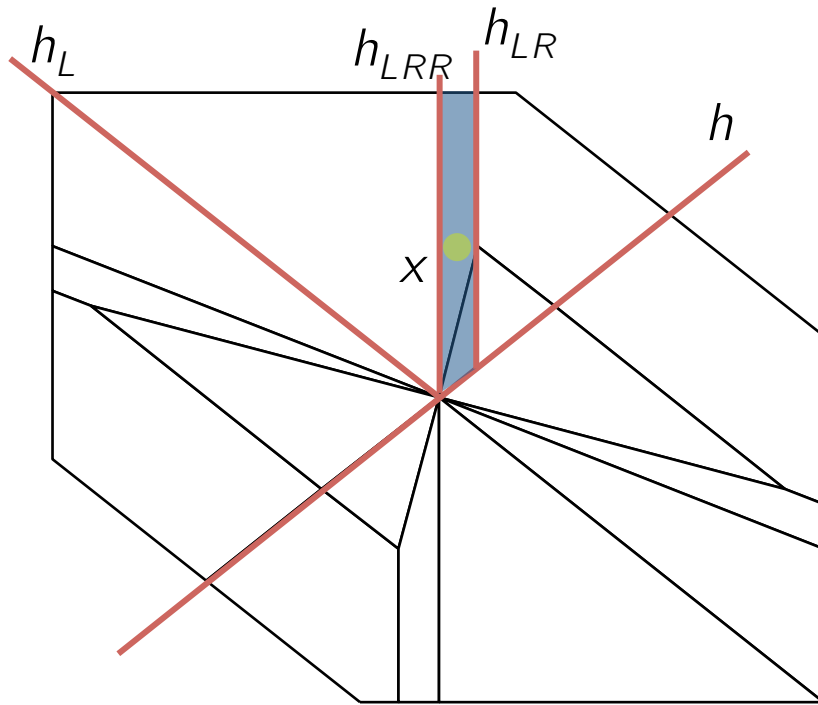
Point Location – Logarithmic search



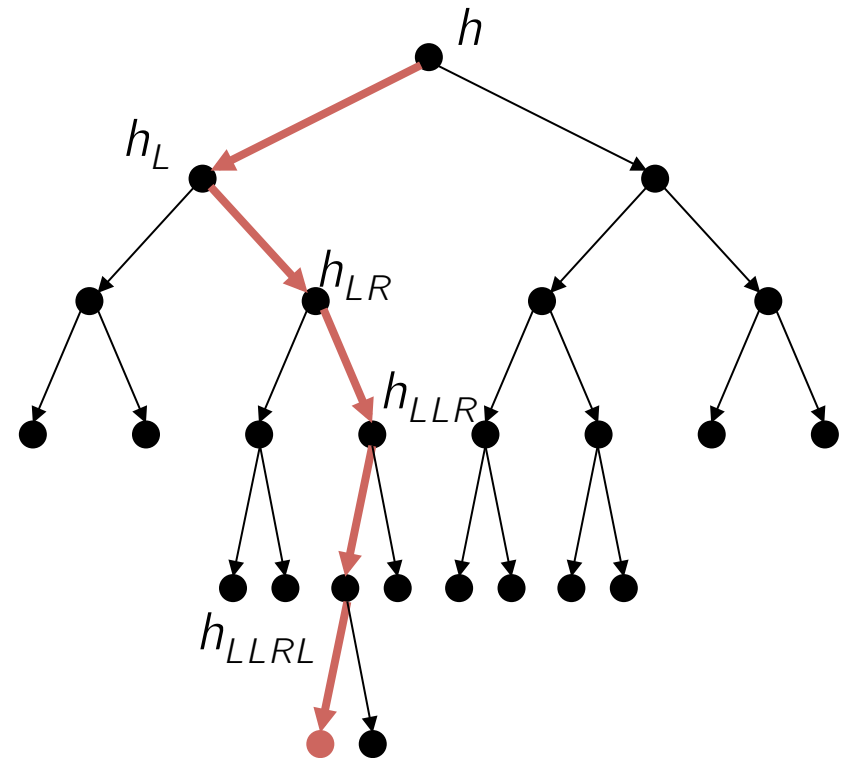
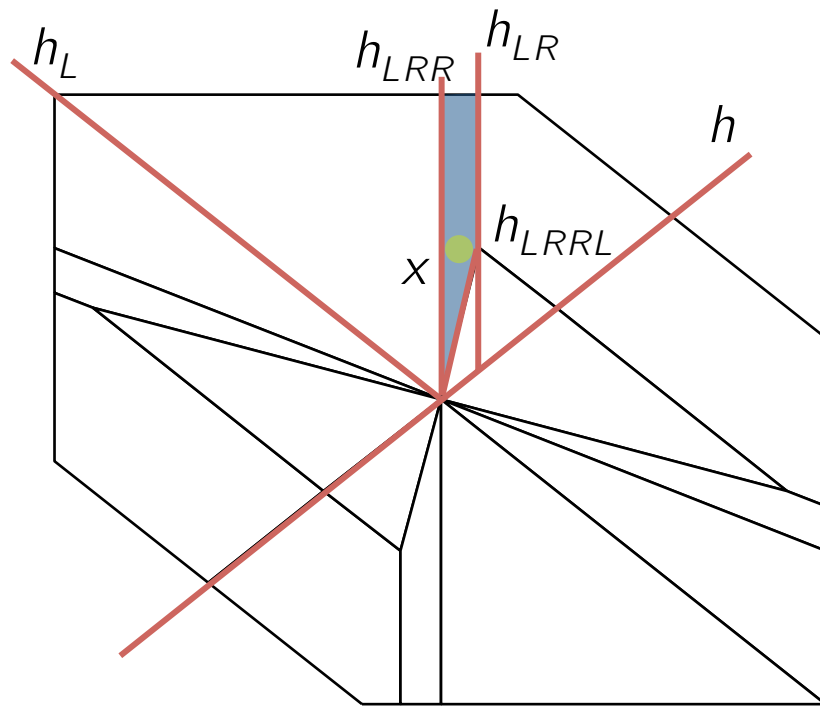
Point Location – Logarithmic search



Point Location – Logarithmic search



Point Location – Logarithmic search



Point Location

- Sequential search
 - Very simple
 - Works for all problems
- Search tree
 - Potentially logarithmic
 - Significant offline processing (reasonable for $< 1'000$ regions)
- Many other options for special cases

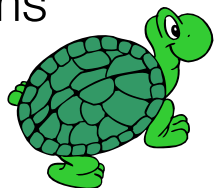
Outline

- Motivating Example
- MPC = Parametric Quadratic Programming
- Parametric Linear Complementarity Problems
 - The Geometry
 - The Algebra
 - Efficient Solution Methods
- Online Computation : Point Location Problem
- Example

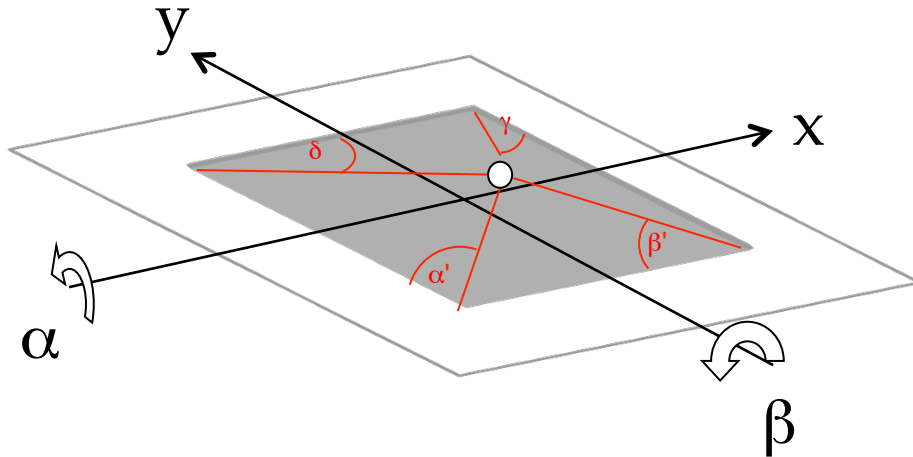
Applications by the Automatic Control Lab



18 ns	Multi-core thermal management (EPFL)	<i>[Zanini et al 2010]</i>
10 μ s	Voltage source inverters	<i>[Mariethoz et al 2008]</i>
20 μ s	DC/DC converters (STM)	<i>[Mariethoz et al 2008]</i>
25 μ s	Direct torque control (ABB)	<i>[Papafotiou 2007]</i>
50 μ s	AC / DC converters	<i>[Richter et al 2010]</i>
5 ms	Electronic throttle control (Ford)	<i>[Vasak et al 2006]</i>
20 ms	Traction control (Ford)	<i>[Borrelli et al 2001]</i>
30 ms	Ball and plate	
50 ms	Autonomous vehicle steering (Ford)	<i>[Besselmann et al 2008]</i>
500 ms	Energy efficient building control (Siemens)	<i>[Oldewurtel et al 2010]</i>



Ball and Plate



- Linearized model : four states for each axis
- Constraints on inputs and states
 - Plate angle
 - Ball position
 - Acceleration
- MPC objective : Path tracking



Ball and Plate System

Matlab
+ RTW



Computer: Pentium 166
Sampling Time: 30 ms

TCP/IP

Code

Data

xPC
Target



u_1, u_2

α, β

x, y

Ball & Plate

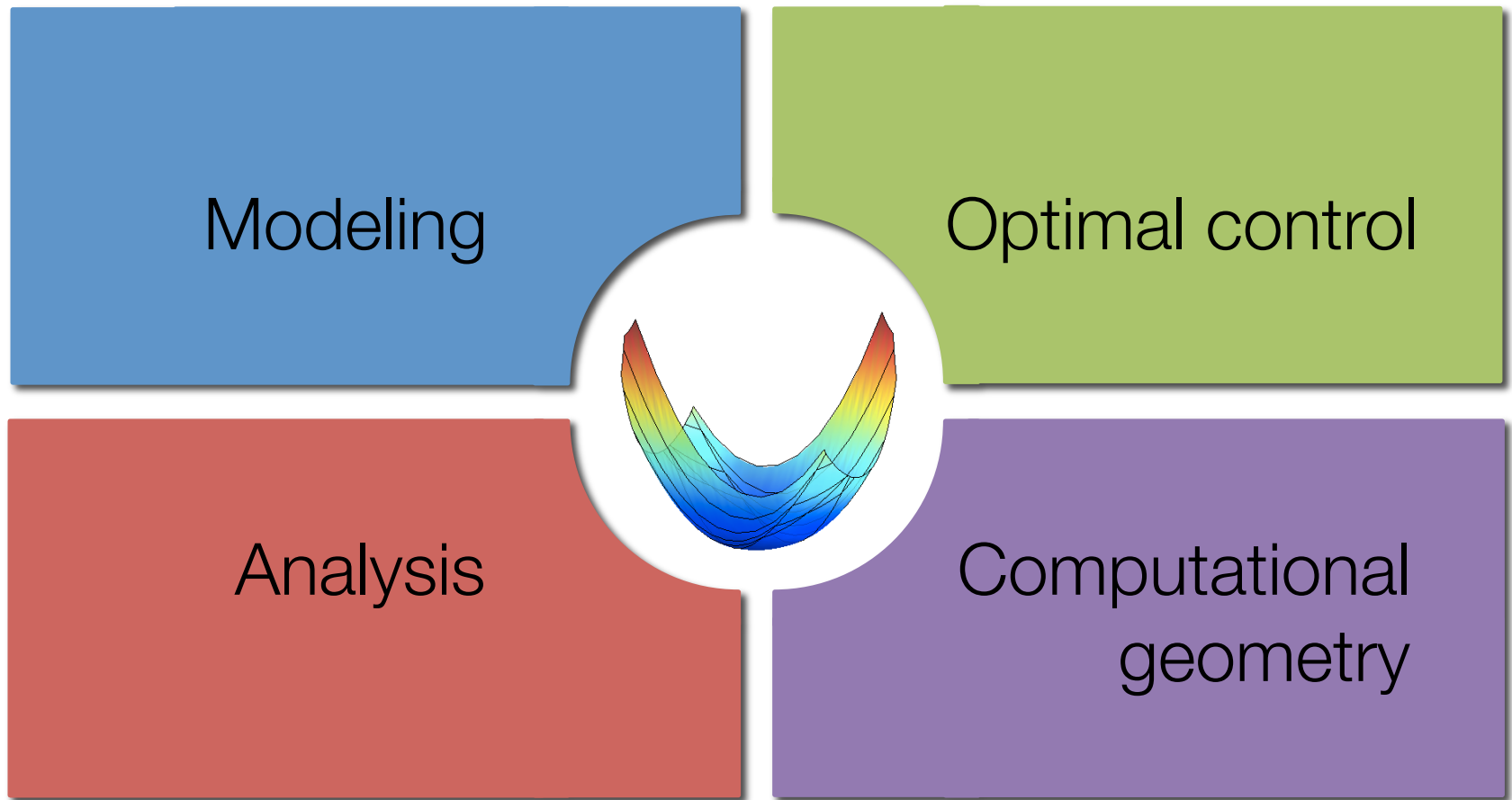


MPC Problem

- 4 states + 1 tracking variable = 5 parameters
- Move-blocking reduces complexity
 - Horizon of 10
 - Inputs 2-10 must be equal

$$\begin{aligned} \min \quad & \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2 \\ \text{s.t.} \quad & x_0 = x \\ & x_{i+1} = Ax_i + Bu_i \\ & y_i = Cx_i \\ & u_{\min} \leq u_i \leq u_{\max} \\ & y_{\min} \leq y_i \leq y_{\max} \\ & x_{\min} \leq x_i \leq x_{\max} \\ & u_{i+1} = u_i, \quad i = \{1, \dots, 9\} \end{aligned}$$

Multi-Parametric Toolbox



Real-Time Model Predictive Control via Multi-Parametric Programming: Theory and Tools

Michal Kvasnica

ISBN: 3639206444

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
model.A = A;  model.B = B;
model.C = C;  model.D = D;

% Input constraints
model.umax = 10;  model.umin = -10;
% Output constraints
model.ymax = 30;  model.ymin = -30;
% State constraints
model.xmax = [30; 15; 15*pi/180; 1];
model.xmin = -model.xmax;
```

$$\begin{aligned} \min \quad & \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2 \\ \text{s.t.} \quad & x_0 = x \\ & x_{i+1} = Ax_i + Bu_i \\ & y_i = Cx_i \\ & u_{\min} \leq u_i \leq u_{\max} \\ & y_{\min} \leq y_i \leq y_{\max} \\ & x_{\min} \leq x_i \leq x_{\max} \\ & u_{i+1} = u_i, \quad i = \{1, \dots, 9\} \end{aligned}$$

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
```

```
model.A = A; model.B = B;
```

```
model.C = C; model.D = D;
```

```
% Input constraints
```

```
model.umax = 10; model.umin = -10;
```

```
% Output constraints
```

```
model.ymax = 30; model.ymin = -30;
```

```
% State constraints
```

```
model.xmax = [30; 15; 15*pi/180; 1];
```

```
model.xmin = -model.xmax;
```

$$\min \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2$$

$$\text{s.t. } x_0 = x$$

$$x_{i+1} = Ax_i + Bu_i$$

$$y_i = Cx_i$$

$$u_{\min} \leq u_i \leq u_{\max}$$

$$y_{\min} \leq y_i \leq y_{\max}$$

$$x_{\min} \leq x_i \leq x_{\max}$$

$$u_{i+1} = u_i, \quad i = \{1, \dots, 9\}$$

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
```

```
model.A = A; model.B = B;
```

```
model.C = C; model.D = D;
```

```
% Input constraints
```

```
model.umax = 10; model.umin = -10;
```

```
% Output constraints
```

```
model.ymax = 30; model.ymin = -30;
```

```
% State constraints
```

```
model.xmax = [30; 15; 15*pi/180; 1];
```

```
model.xmin = -model.xmax;
```

$$\min \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2$$

$$\text{s.t. } x_0 = x$$

$$x_{i+1} = Ax_i + Bu_i$$

$$y_i = Cx_i$$

$$u_{\min} \leq u_i \leq u_{\max}$$

$$y_{\min} \leq y_i \leq y_{\max}$$

$$x_{\min} \leq x_i \leq x_{\max}$$

$$u_{i+1} = u_i, \quad i = \{1, \dots, 9\}$$

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
model.A = A;  model.B = B;
model.C = C;  model.D = D;

% Input constraints
model.umin = -10;
% Output constraints
model.ymin = -30;
% State constraints
model.xmin = -model.xmax;
```

MPT

```
% Prediction horizon
problem.N = 10;
% Control horizon
problem.Nc = 1;
% Quadratic performance objective
problem.norm = 2;
% Penalty on outputs
problem.Qy = 100;
% Penalty on control inputs
problem.R = 0.1;
% Tracking
problem.tracking = 2;
```

$$\begin{aligned} \min \quad & \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2 \\ \text{s.t.} \quad & x_0 = x \\ & x_{i+1} = Ax_i + Bu_i \\ & y_i = Cx_i \\ & u_{\min} \leq u_i \leq u_{\max} \\ & y_{\min} \leq y_i \leq y_{\max} \\ & x_{\min} \leq x_i \leq x_{\max} \\ & u_{i+1} = u_i, \quad i = \{1, \dots, 9\} \end{aligned}$$

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
```

```
model.A = A; model.B = B;
```

```
model.C = C; model.D = D;
```

```
% Input constraints
```

```
model.ymax = 10; model.ymin = -10;
```

```
% Output constraints
```

```
model.ymax = 30; model.ymin = -30;
```

```
% State constraints
```

```
model.xmax = [30; 15; 15*pi/180; 1];
```

```
model.xmin = -model.xmax;
```

MPT

```
% Prediction horizon
```

```
problem.N = 10;
```

```
% Control horizon
```

```
problem.Nc = 1;
```

```
% Quadratic performance objective
```

```
problem.norm = 2;
```

```
% Penalty on outputs
```

```
problem.Qy = 100;
```

```
% Penalty on control inputs
```

```
problem.R = 0.1;
```

```
% Tracking
```

```
problem.tracking = 2;
```

$$\min \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2$$

$$\text{s.t. } x_0 = x$$

$$x_{i+1} = Ax_i + Bu_i$$

$$y_i = Cx_i$$

$$u_{\min} \leq u_i \leq u_{\max}$$

$$y_{\min} \leq y_i \leq y_{\max}$$

$$x_{\min} \leq x_i \leq x_{\max}$$

$$u_{i+1} = u_i, \quad i = \{1, \dots, 9\}$$

Multi-Parametric Toolbox Formulation

MPT

% linear discrete-time prediction model

model.A = A; model.B = B;

model.C = C; model.D = D;

% Input constraints

model.umin = -10; model.umax = 10;

% Output constraints

model.ymin = -30; model.ymax = 30;

% State constraints

model.xmin = [-model.ymax; -model.umin];

model.xmax = [model.ymax; model.umin];

MPT

% Prediction horizon

problem.N = 10;

% Control horizon

problem.Nc = 1;

% Quadratic performance objective

problem.norm = 2;

% Penalty on outputs

problem.Qy = 100;

% Penalty on control inputs

problem.R = 0.1;

% Tracking

problem.tracking = 2;

$$\begin{aligned} \min \quad & \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2 \\ \text{s.t.} \quad & x_0 = x \\ & x_{i+1} = Ax_i + Bu_i \\ & y_i = Cx_i \\ & u_{\min} \leq u_i \leq u_{\max} \\ & y_{\min} \leq y_i \leq y_{\max} \\ & x_{\min} \leq x_i \leq x_{\max} \\ & u_{i+1} = u_i, \quad i = \{1, \dots, 9\} \end{aligned}$$

Multi-Parametric Toolbox Formulation

MPT

```
% linear discrete-time prediction model
model.A = A; model.B = B;
model.C = C; model.D = D;

% Input constraints
model.umin = -10; model.umax = 10;
% Output constraints
model.ymin = -30; model.ymax = 30;
% State constraints
model.xmin = -model.xmax;
model.xmax = [30; 15; 15*pi/180; 1];
```

MPT

```
% Prediction horizon
problem.N = 10;
% Control horizon
problem.Nc = 1;
% Quadratic performance objective
problem.norm = 2;
% Penalty on outputs
problem.Qy = 100;
% Penalty on control inputs
problem.R = 0.1;
% Tracking
problem.tracking = 2;
```

$$\begin{aligned} \min \quad & \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2 \\ \text{s.t.} \quad & x_0 = x \\ & x_{i+1} = Ax_i + Bu_i \\ & y_i = Cx_i \\ & u_{\min} \leq u_i \leq u_{\max} \\ & y_{\min} \leq y_i \leq y_{\max} \\ & x_{\min} \leq x_i \leq x_{\max} \\ & u_{i+1} = u_i, \quad i = \{1, \dots, 9\} \end{aligned}$$

Explicit Solution

MPT

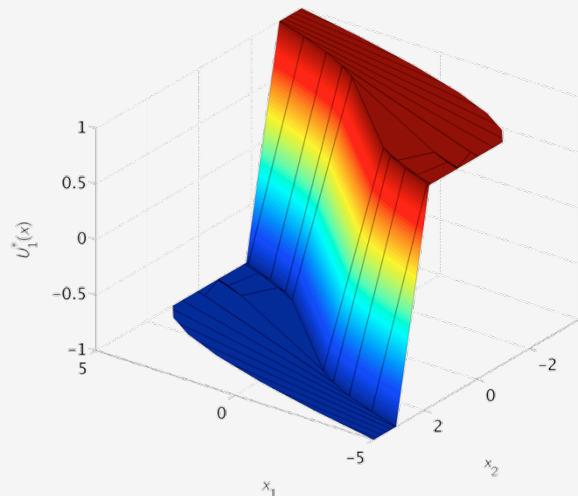
```
mpt_controller = mpt_control(model, problem, 'explicit');
```

```
% Plot controller partition  
plot(mpt_controller)
```

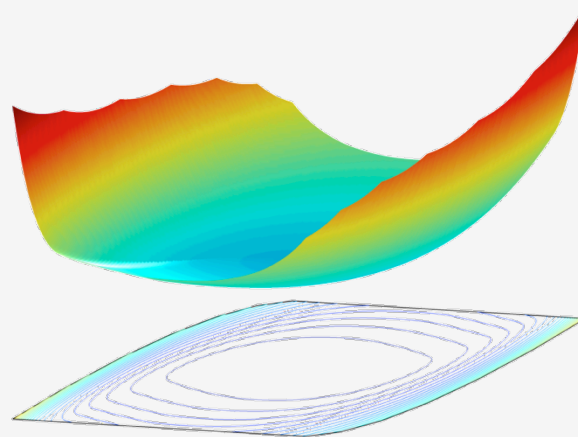
```
% Plot control law  
plotU(mpt_controller)
```

```
% Plot optimal value function  
plotJ(mpt_controller)
```

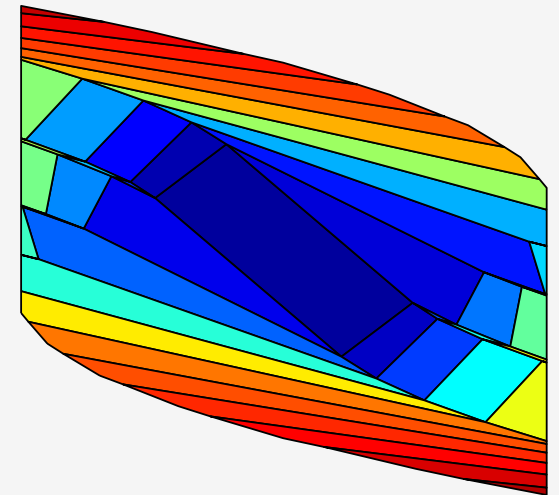
Control input



Value function



Controller partition



Exporting Explicit Solutions

Search tree and export to flat c-file

MPT mpt_searchTree(mpt_controller)
mpt_exportST(mpt_controller)

```
#define MPT_NU 1
#define MPT_NX 2
static float MPT_ST[] = {
-5.225216e-01, 8.526260e-01, 2.007907e+00, 2.000000e+00,
-6.669647e-01, 7.450893e-01, 3.576296e+00, 4.000000e+00,
6.347758e-01, -7.726964e-01, 2.699058e+00, 6.000000e+00,
-2.990218e-01, -9.542463e-01, -2.286671e-01, 8.000000e+00,
-4.546596e-01, -8.906653e-01, 3.576036e-01, 1.000000e+01,
3.630512e-01, 9.317692e-01, 4.490581e-02, 1.200000e+01,
4.546596e-01, 8.906653e-01, 3.576036e-01, 1.400000e+01,
-2.528581e-01, -9.675034e-01, -4.781395e-01, -5.000000e+00,
-6.811331e-01, 7.321596e-01, 4.632449e+00, 1.600000e+01,
-3.630512e-01, -9.317692e-01, 4.490581e-02, -5.000000e+00,
-6.347758e-01, 7.726964e-01, 2.699058e+00, 1.800000e+01,
6.811331e-01, -7.321596e-01, 4.632449e+00, 2.000000e+01,
6.669647e-01, -7.450893e-01, 3.576296e+00, 2.200000e+01,
5.845669e-01, 8.113455e-01, 7.438927e-01, -4.000000e+00,
5.225216e-01, -8.526260e-01, 2.007907e+00, 2.400000e+01,
2.528581e-01, 9.675034e-01, 1.703078e+00, -4.000000e+00,
2.990218e-01, 9.542463e-01, 1.489549e+00, -4.000000e+00,
3.630512e-01, 9.317692e-01, 1.263985e+00, -4.000000e+00,
4.546596e-01, 8.906653e-01, 1.018374e+00, -4.000000e+00,
2.528581e-01, 9.675034e-01, -4.781395e-01, -4.000000e+00,
2.990218e-01, 9.542463e-01, -2.286671e-01, -4.000000e+00,
}
```

Export to flat c-file, sequential search

MPT mpt_exportc(mpt_controller)

```
#ifndef tmwtypes_h
/* RTW is used, switch to real_T data types to avoid problems with TLC compilation */
static long mpt_searchTree(const real_T *X, real_T *U)
#else
static long mpt_searchTree(const float *X, float *U)
#endif
{
    int ix, iu;
    long node = 1, row;
    float hx, k;

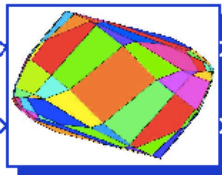
    /* initialize U to zero */
    for (iu=0; iu<MPT_NU; iu++) {
        U[iu] = 0;
    }

    /* find region which contains the state x0 */
    while (node > 0) {
        hx = 0;
        row = (node-1)*(MPT_NX+3);
        for (ix=0; ix<MPT_NX; ix++) {
            hx = hx + MPT_ST[row+ix]*X[ix];
        }
        k = MPT_ST[row+MPT_NX];
    }
}
```

RTW

state
measurements

state/output
references



MPT Controller

control
action

region
number

- Explicit MPC block in Simulink
- Compatible with RTW

Ball and Plate Explicit Controller

- 4 states + 1 tracking variable = 5 parameters
- Move-blocking reduces complexity
 - Horizon of 10
 - Inputs 2-10 must be equal

$$J^*(x, y_t) = \min \sum_{i=0}^9 100 \|y_i - y_t\|_2^2 + 0.1 \|u_i\|_2^2$$

s.t. $x_0 = x$

$$x_{i+1} = Ax_i + Bu_i$$
$$y_i = Cx_i$$
$$u_{\min} \leq u_i \leq u_{\max}$$
$$y_{\min} \leq y_i \leq y_{\max}$$
$$u_{i+1} = u_i, \quad i = \{1, \dots, 9\}$$

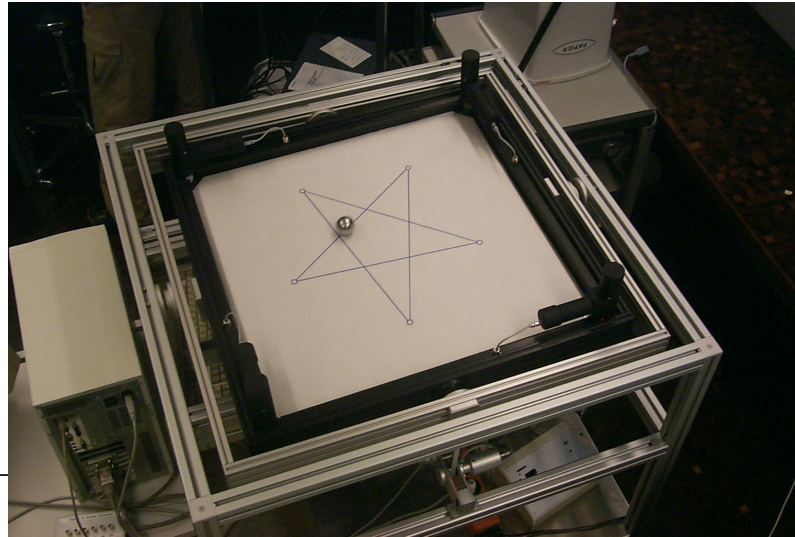
Explicit solution (per dimension)

Regions : 529

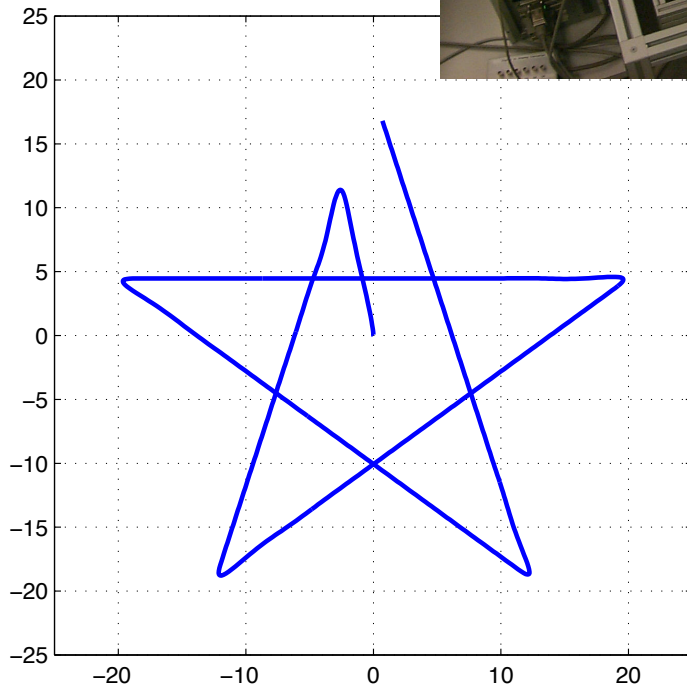
Storage : 48'000 numbers
(192 kB)

Computation : 89'000 FLOPS
(~1ms)

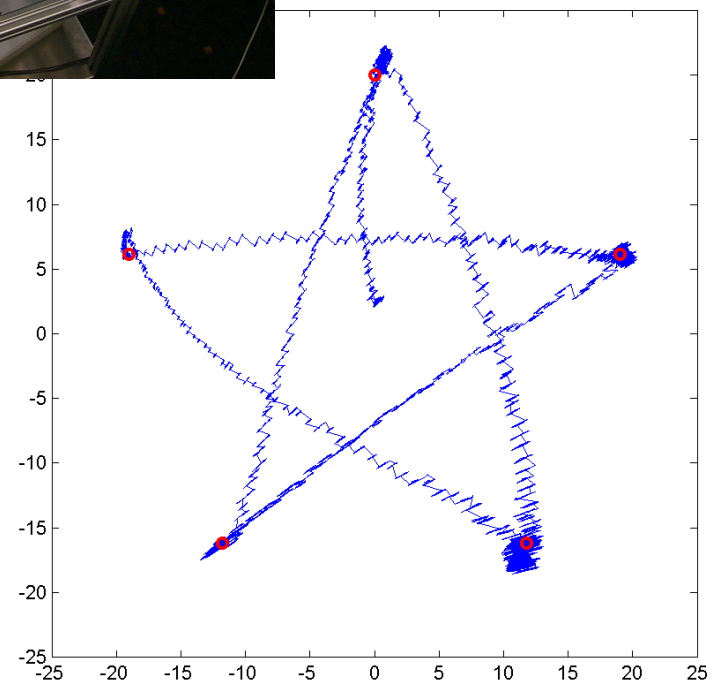
Ball and Plate : Pentagram



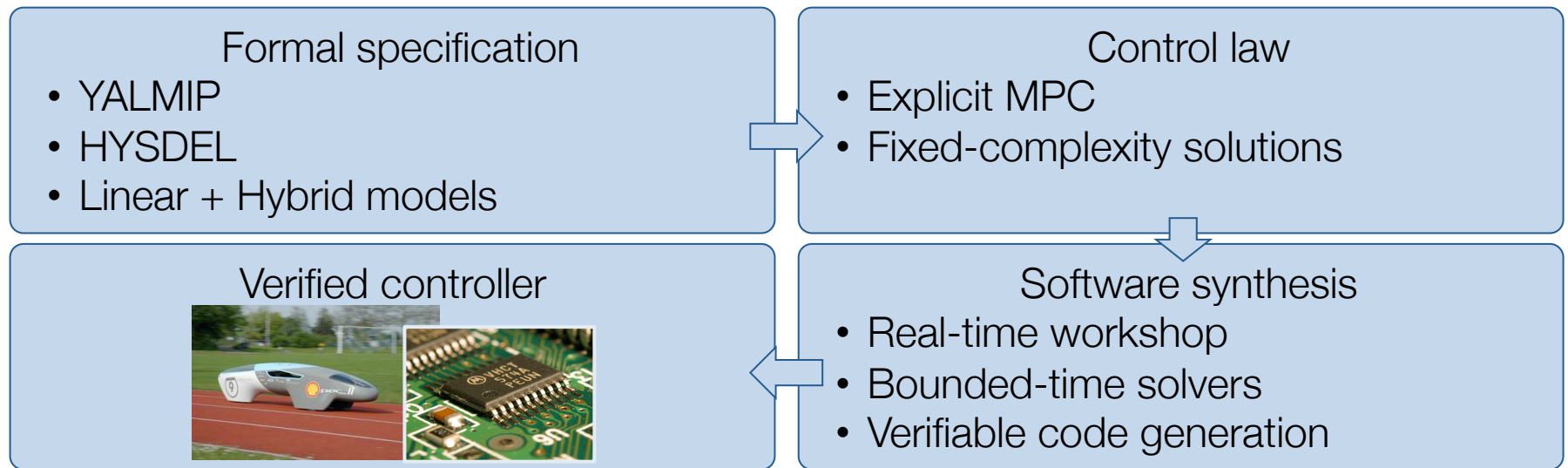
Simulation



Experiment

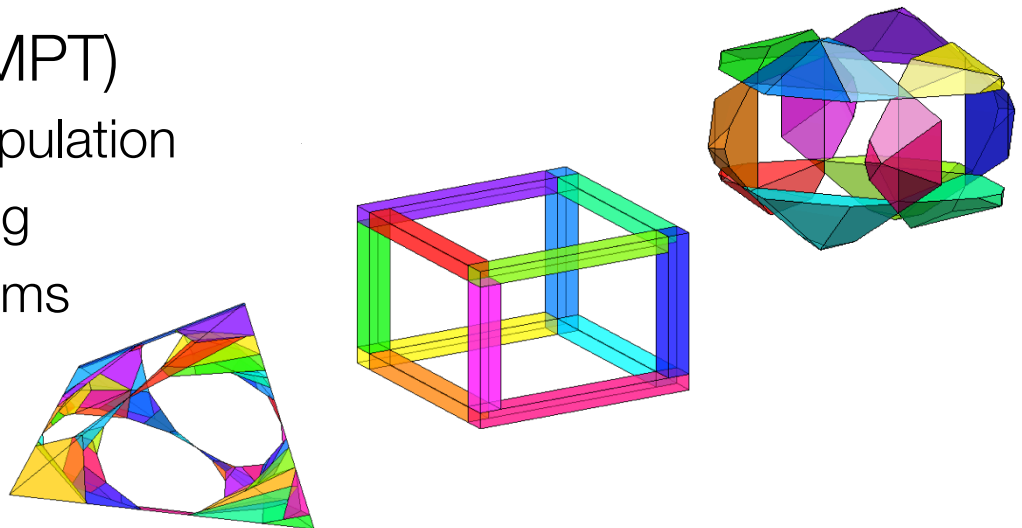


Real-time MPC Software Toolbox



Multi-Parametric Toolbox (MPT)

- (Non)-Convex Polytopic Manipulation
- Multi-Parametric Programming
- Control of PWA and LTI systems



Summary

- Linear MPC + Quadratic or linear-norm cost $\rightarrow$ Controller is PWA function
- We can pre-compute this function offline efficiently
- Online evaluation of a PWA function is *very* fast (ns - μ s)
- We can only do this for very small systems! (3-6 states)