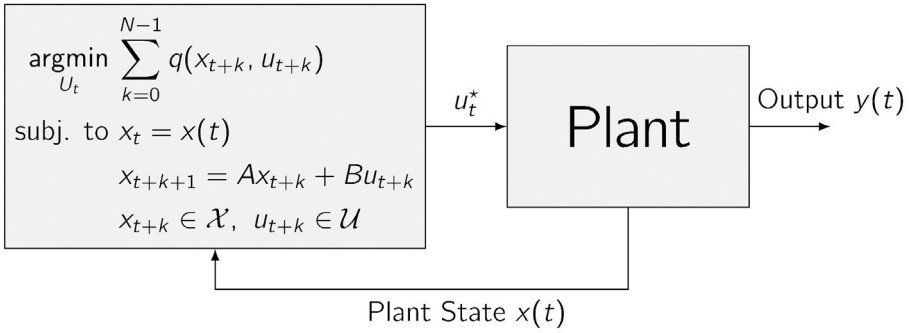
Lecture 11: Explicit MPC

I. Reference Tracking

- **Initial Remarks: Overview for Online and Offline**

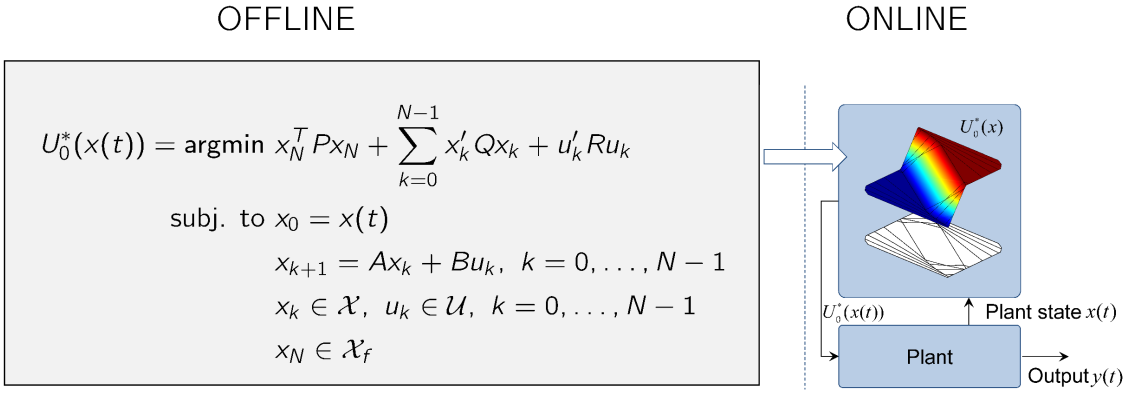
Consider the MPC scheme we currently have:



This requires at each time step an online solution of an optimization problem.

Now, we hope to cut down the cost of online computation and get the controller as soon as possible:

Recall in lecture 6, when we talked about the 2-norm feedback control of CFTOC, we derive that one can explicitly get a **control law** that depends on the initial condition $x(0)$. That gives us inspirations to design MPC as follows:



This is the explicit MPC. The core idea is:

1. The optimization problem is **parameterized by the state**
2. Pre-compute control law as a function of state x
3. The control law is piecewise affine for linear system/constraints

This would let online computation dramatically reduced and can be done in real-time, and the tool we use is **Multi-Parametric (MP) programming**. Therefore, this lecture will cover content about MP programming and its applications in MPC.

- **Simple Example of MP-Programming**

Consider the following problem:

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

$$\text{subj. to. } z \le 1 + x$$

Where $x \in \mathbb{R}$ is a parameter.

The goals:

1. Find $z^*(x) = \text{argmin}_z J(z,x)$
2. Find all x for which the problem has a solution
3. Compute the value function $J^*(x)$

Obviously, this is a convex optimization problem, and the constraints satisfy the Slater condition

Define Lagrangian: $\mathcal{L}(z,x,\lambda) = f(z,x) + \lambda(z - x - 1)$

The KKT conditions are:

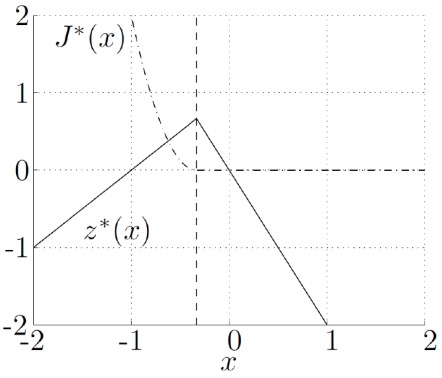
$$\begin{aligned} z + 2x + \lambda &= 0 \\ z - x - 1 &\le 0 \\ \lambda(z - x - 1) &= 0 \\ \lambda &\ge 0 \end{aligned}$$

The complementarity condition implies:

$$\begin{aligned} z - x - 1 &= 0 \\ \lambda &\ge 0 \end{aligned} \Rightarrow \begin{cases} z^*(x) = x + 1 \\ x \le -\frac{1}{3} \\ J^*(x) = \frac{9}{2}x^2 + 3x + \frac{1}{2} \end{cases} \quad \text{or} \quad \begin{aligned} z - x - 1 &< 0 \\ \lambda &= 0 \end{aligned} \Rightarrow \begin{cases} z^*(x) = -2x \\ x > -\frac{1}{3} \\ J^*(x) = 0 \end{cases}$$

Therefore, the complete optimal solution $z^*(x)$ is:

$$z^*(x) = \begin{cases} x + 1, & \text{if } x \le -\frac{1}{3} \\ -2x, & \text{if } x \ge -\frac{1}{3} \end{cases}$$



And the optimal value $J^*(x)$ is:

$$J^*(x) = \begin{cases} \frac{9}{2}x^2 + 3x + \frac{1}{2}, & \text{if } x \le -\frac{1}{3} \\ 0, & \text{if } x \ge -\frac{1}{3} \end{cases}$$

We can observe that the optimal solution is affine and the optimal value is quadratic, and we **switch the solution depending on which constraints are active**.

Note:

1. In this method, since the case is simple enough, we only have one constraint, so we can list all the cases for complementarity conditions (one is active, the other would be inactive), but if the number

of constraints grows up, the complexity of listing all the complementarity conditions would be combinatorial exploded.

2. Therefore, in real practice, we do not do that naive way to enumerate all the cases. **It is because many of the combinations of the constraints may lead to infeasible solutions, and there are also redundant constraints in certain constraint combinations.**
3. In practice, the algorithm is as follows: First, pick a random feasible point and calculate the region where the same constraints are active. Then, to proceed to the next possible critical region, we choose to flip the sign of a single constraint to do the transition and keep tracking the already calculated region, repeat this process, and we will finally fill up the whole parameter (state) space.

For more details on the algorithm, refer to the textbook and related papers.

II. Multi-Parametric Quadratic Programming (mp-QP)

- **mp-QP: Problem Formulation**

Problems and Goals

$$J^*(x) = \min_z \ \frac{1}{2} z^{\mathrm{T}} H z$$

$$\text{subj. to. } Gz \leq w + Sx$$

Where $H \succ 0, z \in \mathbb{R}^s, x \in \mathbb{R}^n$ and $G \in \mathbb{R}^{m \times s}$

Given a closed and bounded polyhedral set $\mathcal{K} \subseteq \mathbb{R}^n$ of parameters denoted by $\mathcal{K}^* \subseteq \mathcal{K}$ the region of parameters $x \in \mathcal{K}$ such that the **problem is feasible**:

$$\mathcal{K}^* = \{x \in \mathcal{K} : \exists\, z,\ Gz \leq w + Sx\}$$

Goals:

1. find $z^*(x) = \operatorname{argmin}_z J(z,x)$
2. find all x for which the problem has a solution
3. compute the value function $J^*(x)$

Active Set and Critical Region

Let $I = \{1, \ldots, m\}$ be the set of constraints indices

Definition (Active and Inactive Set)

We define the active set at $x, A(x)$ and its complement, $NA(x)$ as:

$$A(x) = \{i \in I,\ G_i z^*(x) - S_i x = w_i\}$$

$$NA(x) = \{i \in I,\ G_i z^*(x) - S_i x < w_i\}$$

Where G_i, S_i and w_i are the i-th row of G, S and w, respectively.

Definition (Critical Region)

$\mathcal{CR}_A$ is the set of parameters x for which the same set $A \subseteq I$ of constraints is active at the optimum. For a given $\bar{x} \in \mathcal{K}^*$, let $(A, NA) = (A(\bar{x}), NA(\bar{x}))$, then:

$$\mathcal{CR}_A = \{x \in \mathcal{K}^*,\ A(x) = A\}$$

- **mp-QP: Global properties of the solution**

The following theorem summarizes the properties of the mp-QP solution:

Theorem (Solution of mp-QP)

1. The feasible set $\mathcal{K}^*$ is a **polyhedron.**
2. The optimizer function $z^*(x):\mathcal{K}\to\mathbb{R}^m$ is:

   Continuous

   polyhedral piecewise affine over $\mathcal{K}^*$. It is affine in each critical region $\mathcal{CR}_i$, every $\mathcal{CR}_i$ is a polyhedron and $\cup\mathcal{CR}_i=\mathcal{K}^*$
3. The value function $J^*(x):\mathcal{K}\to\mathbb{R}$ is:

   continuous

   convex

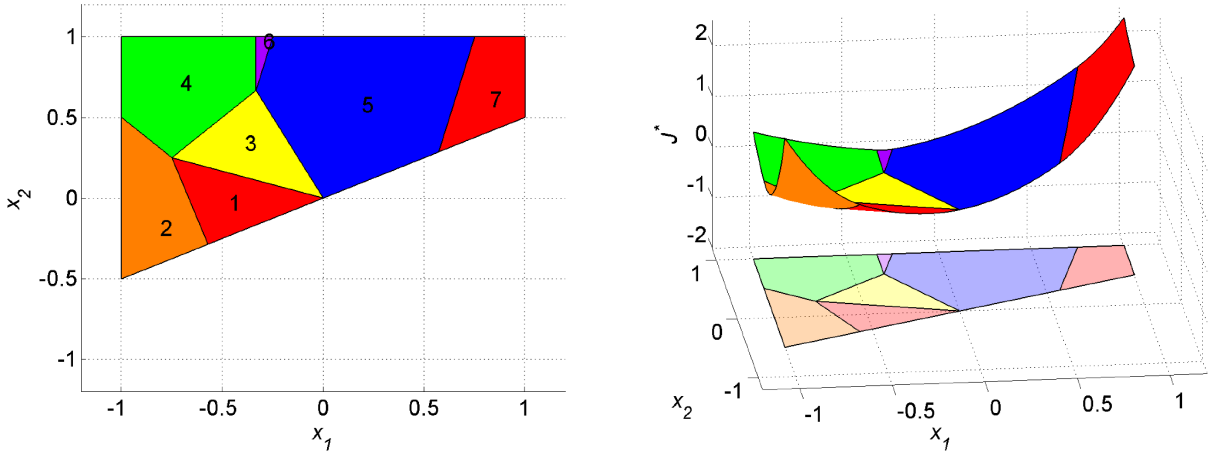
   polyhedral piecewise quadratic over $\mathcal{K}^*$, it is quadratic in each critical region $\mathcal{CR}_i$

- **mp-QP: Example**

Consider the example:

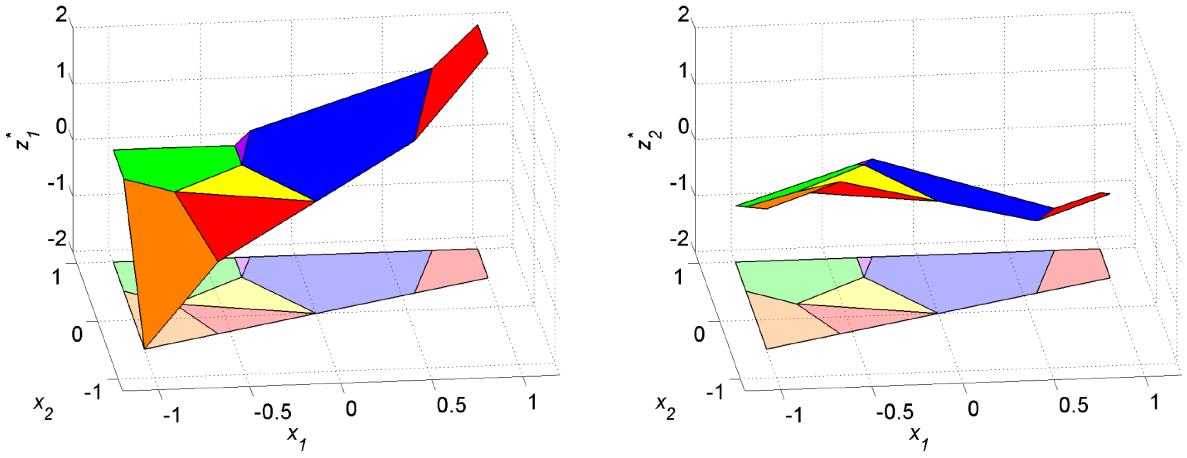
$$\begin{aligned}\min_{z(x)}\ & \frac{1}{2}(z_1^2+z_2^2)\\ \text{subj. to.}\quad & z_1\le 1+x_1+x_2\\ & -z_1\le 1-x_1-x_2\\ & z_2\le 1+x_1-x_2\\ & -z_2\le 1-x_1+x_2\\ & z_1-z_2\le x_1+3x_2\\ & -z_1+z_2\le -2x_1-x_2\\ & -1\le x_1\le 1\\ & -1\le x_2\le 1\end{aligned}$$

The explicit solution is defined over $i=1,\dots,7$ different regions $\mathcal{P}_i=\{x\in\mathbb{R}^2\mid A_ix\le b_i\}$ in the parameter space x_1-x_2 and the objective function is piecewise quadratic:



And the primal solution is given as piecewise affine function $z(x) = F_i + g_i x$ if $x \in \mathcal{P}_i$

$$z^*(x) = \begin{cases} \begin{bmatrix} 0.5 & 1.5 \\ -0.5 & -1.5 \end{bmatrix} x & \text{,if } x \in \mathcal{P}_1 \\ \begin{bmatrix} 2 & 2 \\ 1 & -1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \end{bmatrix} & \text{,if } x \in \mathcal{P}_2 \\ \vdots \\ \vdots \end{cases}$$



Note:

1. We can observe that there are 8 constraints, and if we enumerate them, there should be a lot more possible region due to the combinatorial complexity. Here there exist only 7 critical regions since for all other possible combinations, the problem would be infeasible. Recall our definition, the solution should be defined on $\mathcal{K}^*$ where the problem is feasible and that $\mathcal{K}^* = \cup \mathcal{CR}_i$. In the following example, the critical regions can even be further merged depending on our requirements and can produce a simpler outcome, but here we talk about the raw result of the problem.
2. For a simpler but more detailed example of the process of hand-calculating mp-QP, especially on the determination of critical regions, refer to Homework 6, Ex2.

III. Multi-Parametric Linear Programming (mp-LP)

- **mp-LP: Problem Formulation**

Problems and Goals

$$J^*(x) = \min_z \; c^{\mathrm{T}} z$$

$$\text{subj. to. } Gz \leq w + Sx$$

Where $z \in \mathbb{R}^s$, $x \in \mathbb{R}^n$ and $G \in \mathbb{R}^{m \times s}$

Given a closed and bounded polyhedral set $\mathcal{K} \subseteq \mathbb{R}^n$ of parameters denoted by $\mathcal{K}^* \subseteq \mathcal{K}$ the region of parameters $x \in \mathcal{K}$ such that the **problem is feasible**:

$$\mathcal{K}^* = \{x \in \mathcal{K} : \exists\, z,\ Gz \leq w + Sx\}$$

Goals:

1. find $z^*(x) = \text{argmin}_z\ J(z,x)$
2. find all x for which the problem has a solution
3. compute the value function $J^*(x)$

- **mp-LP: Global properties of the solution**

The following theorem summarizes the properties of the mp-LP solution:

Theorem (Solution of mp-LP)

1. The feasible set $\mathcal{K}^*$ is a **polyhedron.**
2. The optimizer function $z^*(x): \mathcal{K} \to \mathbb{R}^m$ is:

   continuous

   polyhedral piecewise affine over $\mathcal{K}^*$. It is affine in each critical region $\mathcal{CR}_i$, every $\mathcal{CR}_i$ is a polyhedron and $\cup \mathcal{CR}_i = \mathcal{K}^*$

   Otherwise, it is always possible to choose such a continuous and PPWA optimizer function $z^*(x)$.
3. The value function $J^*(x): \mathcal{K} \to \mathbb{R}$ is:

   continuous

   convex

   polyhedral piecewise affine over $\mathcal{K}^*$, it is quadratic in each critical region $\mathcal{CR}_i$

Note:

It can be observed that there are two main differences in mp-LP case:

1. The objective function now is polyhedral piecewise affine instead of piecewise quadratic
2. Since in every critical region we are solving a linear programm, and linear programming is possible to have multiple solutions, and that might lead to discontinuity of the solution. So we need to emphasize that we **should and are able to pick a certain solution from the multiple solutions and maintain the continuous PPWA property of the solution**.

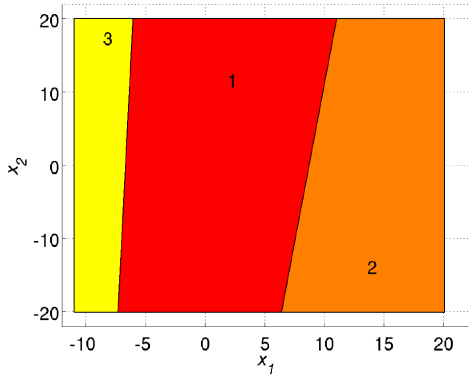
- **mp-LP: Example**

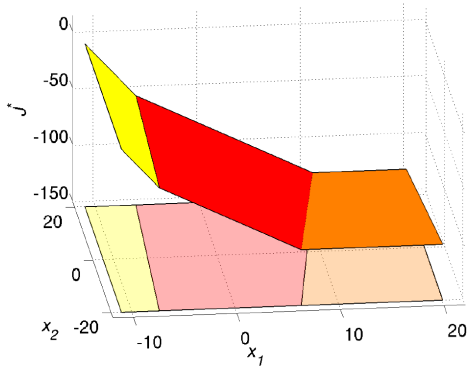
Consider the example:

$$\min_{z(x)} -3z_1 - 8z_2$$

$$\begin{aligned}
\text{subj. to.} \quad & z_1 + z_2 \le 13 + x_1 \\
& 5z_1 - 4z_2 \le 20 \\
& -8z_1 + 22z_2 \le 121 + x_2 \\
& -4z_1 - z_2 \le -8 \\
& -z_1 \le 0 \\
& -z_2 \le 0 \\
& -11 \le x_1 \le 20 \\
& -20 \le x_2 \le 20
\end{aligned}$$

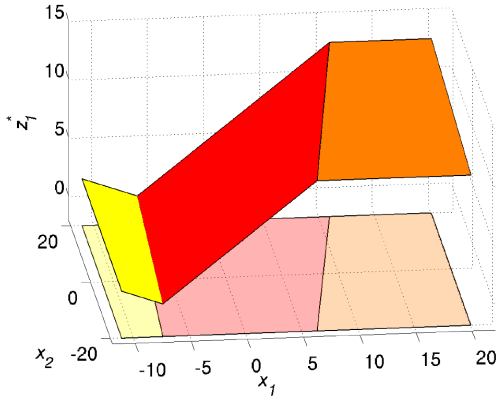
The explicit solution is defined over $i=1,\ldots,3$ different regions $\mathcal{P}_i=\{x\in\mathbb{R}^2 \mid A_i x\leq b_i\}$ in the parameter space x_1-x_2 and the objective function is piecewise quadratic:

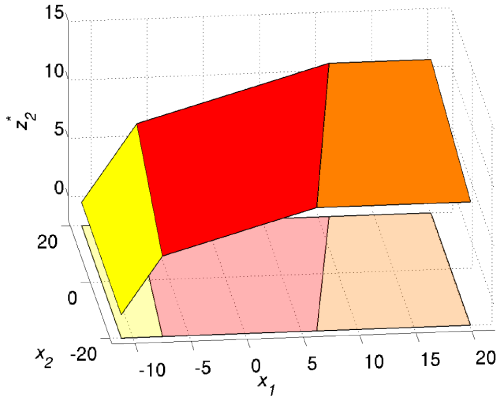




And the primal solution is given as piecewise affine function $z(x)=F_i+g_i x$ if $x\in\mathcal{P}_i$

$$z^*(x)=\begin{cases}\begin{bmatrix}0.733 & -0.033\\ 0.267 & -0.033\end{bmatrix}x+\begin{bmatrix}5.5\\ 7.5\end{bmatrix} & ,\text{if } x\in\mathcal{P}_1\\ \begin{bmatrix}0 & 0.051\\ 0 & 0.064\end{bmatrix}x+\begin{bmatrix}11.846\\ 9.808\end{bmatrix} & ,\text{if } x\in\mathcal{P}_2\\ \begin{bmatrix}-0.333 & 0\\ 1.333 & 0\end{bmatrix}x+\begin{bmatrix}-1.667\\ 14.667\end{bmatrix} & ,\text{if } x\in\mathcal{P}_3\end{cases}$$





III. MP-Programming for Constrained Finite Time Optimal Control

This part of the notes is basically the same as lecture 6, but now with the detailed learning of mp-programming, we can look back and get a better understanding of all these contents

- **Recall: Quadratic CFTOC with Substitution**

Recall our CFTOC with quadratic cost and solution with substitution in lecture 6:

$$J_0^*(x(0))=\min_{U_0}\ x_N^{\mathrm{T}}Px_N+\sum_{k=0}^{N-1}x_k^{\mathrm{T}}Qx_k+u_k^{\mathrm{T}}Ru_k$$

$$\begin{aligned}\text{subj. to. }\ & x_{k+1}=Ax_k+Bu_k,\ k=0,\ldots,N-1\\ & x_k\in\mathcal{X},\ u_k\in\mathcal{U},\ k=0,\ldots,N-1\\ & x_N\in\mathcal{X}_f\\ & x_0=x(0)\end{aligned}$$

with $P \succeq 0, Q \succeq 0, R \succ 0, N$ is the time horizon, and the feasible set $\mathcal{X}, \mathcal{U}, \mathcal{X}_f$ are polyhedra.

Step 1: Rewrite the cost as:

$$\begin{aligned} J_0(x(0), U_0) &= x_N^{\mathrm{T}} P x_N + \sum_{k=0}^{N-1} x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k \\ &= U_0^{\mathrm{T}} H U_0 + 2x(0)^{\mathrm{T}} F U_0 + x(0)^{\mathrm{T}} Y x(0) \\ &= \begin{bmatrix} U_0^{\mathrm{T}} & x(0)^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} H & F^{\mathrm{T}} \\ F & Y \end{bmatrix} \begin{bmatrix} U_0^{\mathrm{T}} & x(0)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \end{aligned}$$

Note: $\begin{bmatrix} H & F^{\mathrm{T}} \\ F & Y \end{bmatrix} \succeq 0$ since $J_0(x(0), U_0) \geq 0$ by assumption.

Step 2: Rewrite the constraints compactly as

$$G_0 U_0 \leq w_0 + E_0 x(0)$$

Step 3: Rewrite the optimal control problem as the standard QP form:

$$J_0^*(x(0)) = \min_{U_0} \begin{bmatrix} U_0^{\mathrm{T}} & x(0)^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} H & F^{\mathrm{T}} \\ F & Y \end{bmatrix} \begin{bmatrix} U_0^{\mathrm{T}} & x(0)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$$

$$\text{subj. to } G_0 U_0 \leq w_0 + E_0 x(0)$$

Thus, for a given $x(0)$, U_0^* can be found via a QP solver.

Note that the feasible set $\mathcal{X}, \mathcal{U}, \mathcal{X}_f$ are all polyhedra., in other words, they are given by:

$$\mathcal{X} = \{x \mid A_x x \leq b_x\} \quad \mathcal{U} = \{u \mid A_u u \leq b_u\} \quad \mathcal{X}_f = \{x \mid A_f x \leq b_f\}$$

So the G_0, E_0 and w_0 in the above formulation is constructed as below:

$$G_0 = \begin{bmatrix} \underbrace{\begin{matrix} A_u & 0 & \dots & 0 \\ 0 & A_u & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_u \end{matrix}}_{\mathcal{U}} \\ \underbrace{\begin{matrix} 0 & 0 & \dots & 0 \\ A_x B & 0 & \dots & 0 \\ A_x A B & A_x B & \ddots & 0 \\ \vdots & \vdots & \dots & \vdots \\ A_x A^{N-2} B & A_x A^{N-3} B & \dots & 0 \end{matrix}}_{\mathcal{X}} \\ \underbrace{\begin{matrix} A_f A^{N-1} B & A_f A^{N-2} B & \dots & A_f B \end{matrix}}_{\mathcal{X}_f} \end{bmatrix}, \quad E_0 = \begin{bmatrix} \underbrace{\begin{matrix} 0 \\ 0 \\ \vdots \\ 0 \end{matrix}}_{\mathcal{U}} \\ \begin{matrix} -A_x \\ -A_x A \\ -A_x A^2 \\ \vdots \end{matrix} \\ \underbrace{\begin{matrix} -A_x A^{N-1} \end{matrix}}_{\mathcal{X}} \\ \underbrace{\begin{matrix} -A_f A^N \end{matrix}}_{\mathcal{X}_f} \end{bmatrix}, \quad w_0 = \begin{bmatrix} \underbrace{\begin{matrix} b_u \\ b_u \\ \vdots \\ b_u \end{matrix}}_{\mathcal{U}} \\ \begin{matrix} b_x \\ b_x \\ b_x \\ \vdots \end{matrix} \\ \underbrace{\begin{matrix} b_x \end{matrix}}_{\mathcal{X}} \\ \underbrace{\begin{matrix} b_f \end{matrix}}_{\mathcal{X}_f} \end{bmatrix}$$

- **Recall: 2-Norm State Feedback Solution using Multi-parametric Programming**

In lecture 6, when solving the feedback solution for the above CFTOC, we take the steps as follows:

Step 1: Define $z \triangleq U_0 + H^{-1} F^{\mathrm{T}} x(0)$ and transform the problem into

$$\hat{J}(x(0)) = \min_z \ z^{\mathrm{T}} H z$$

$$\text{subj. to } G_0 z \leq w_0 + S_0 x(0)$$

Where $S_0 \triangleq E_0 + G_0 H^{-1} F^{\mathrm{T}}$ and $\hat{J}^*(x(0)) = J_0^*(x(0)) - x(0)^{\mathrm{T}}(Y - FH^{-1}F^{\mathrm{T}})x(0)$.

Now looking back, it can be observed that the CFTOC problem is exactly in the form we discussed above, it is an mp-QP, and now we know why we can get the solution using the following procedure:

Step 2: Solve the mp-QP to get the explicit solution $z^*(x(0))$

Step 3: Obtain $U_0^*(x(0))$ from $z^*(x(0))$

Main result:

1. The **open loop optimal control function** can be obtained by solving the mp-QP problem and calculating $U_0^*(x(0)), \forall x(0) \in \mathcal{X}_0$ as $U_0^* = z(x(0)) - H^{-1}F^{\mathrm{T}}x(0)$
2. The first component of the multiparametric solution has the form:

$$u^*(0) = f_0(x(0)),\ \forall x(0) \in \mathcal{X}_0$$

And $f_0 : \mathbb{R}^n \to \mathbb{R}^m$ is continuous and piecewise affine on polyhedra

$$f_0(x) = F_0^i x + g_0^i \ \text{ if } \ x \in CR_0^i,\ i = 1, \ldots, N_0^r$$

3. The polyhedral sets $CR_0^i = \{x \in \mathbb{R}^n \mid H_0^i x \le K_0^i\},\ i = 0 \cdots, N_0^r$ are a partition of the feasible polyhedron $\mathcal{X}_0$
4. The value function $J_0^*(x(0))$ is convex and piecewise quadratic on polyhedra.

Example: simple double Integrator.

Consider the double integrator:

$$x(t+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

Subject to constraints:

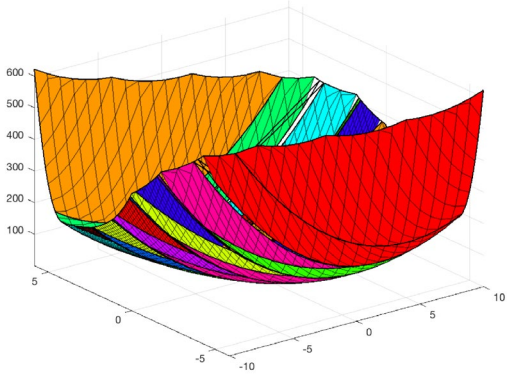
$$-1 \le u(k) \le 1,\ k = 0, \ldots, 5$$

$$\begin{bmatrix} -10 \\ -10 \end{bmatrix} \le x(k) \le \begin{bmatrix} 10 \\ 10 \end{bmatrix},\ k = 0, \ldots, 5$$

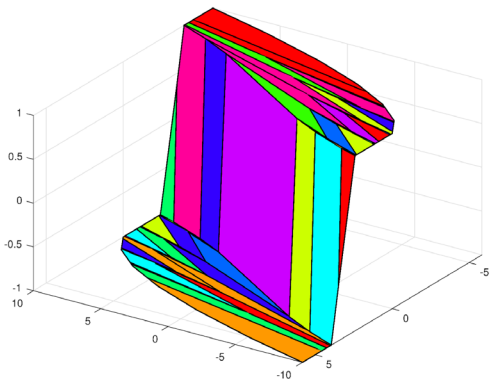
Compute the **state feedback** optimal controller $u^*(0)\ (x(0))$ solving the CFTOC problem with:

$$N = 6,\ Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ R = 0.1,\ \mathcal{X}_f = \mathbb{R}^2 \text{ and let } P \text{ be a solution of ARE}$$

We can get the solution as the following figures shows:

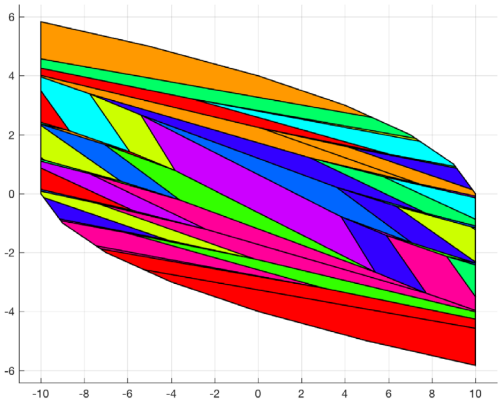


Value function $J^*(0)\ (N_0^r = 61)$

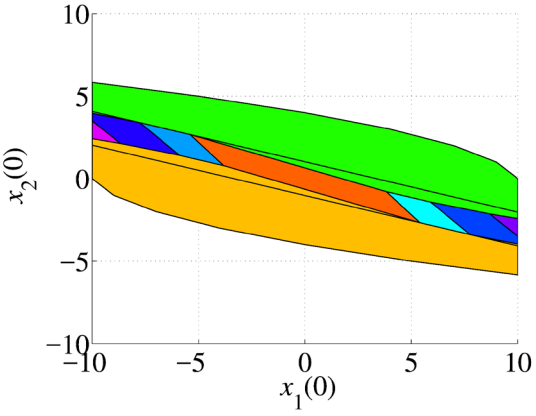


Optimal PWWA control $u^*(0)\ (N_0^r = 61)$

If fact, the above is the raw output of multi-parametric programming, it has 61 critical regions. We can observe that there are a lot of polyhedral region that use the same saturated input, if we combine the region where the input are the same, we can further simplified the result as follows:



Original critical region $(N_0^r = 61)$



Merged critical region $(N_0^r = 13)$

- **Recall: 1-Norm and inf-Norm Minimization CFTOC with Substitution**

Similarly, in lecture 6, we discussed the CFTOC for 1-norm and ∞-norm minimization, i.e. piece-wise linear cost function:

$$J_0^*(x(0)) = \min_{U_0} \ \|Px_N\|_p + \sum_{k=0}^{N-1} \|Qx_k\|_p + \|Ru_k\|_p$$

$$\begin{aligned} \text{subj. to} \quad & x_{k+1} = Ax_k + Bu_k, \ k = 0, \ldots, N-1 \\ & x_k \in \mathcal{X}, \ u_k \in \mathcal{U}, \ k = 0, \ldots, N-1 \\ & x_N \in \mathcal{X}_f \\ & x_0 = x(0) \end{aligned}$$

With $p = 1$ or $p = \infty$, P, Q, R are full column rank matrices. The ∞-norm case is transformed as:

$$\min_{z_0} = \varepsilon_0^x + \ldots + \varepsilon_N^x + \varepsilon_0^u + \ldots + \varepsilon_{N-1}^u$$

$$\begin{aligned} \text{subj. to} \quad & -\mathbf{1}_n \varepsilon_k^x \leq \pm Q\left[A^k x_0 + \sum_{j=0}^{k-1} A^j B u_{k-1-j}\right] \\ & -\mathbf{1}_n \varepsilon_N^x \leq \pm P\left[A^N x_0 + \sum_{j=0}^{N-1} A^j B u_{N-1-j}\right] \\ & -\mathbf{1}_n \varepsilon_N^u \leq R + \pm Ru_k \\ & A^k x_0 + \sum_{j=0}^{k-1} A^j B u_{k-1-j} \in \mathcal{X} \\ & A^N x_0 + \sum_{j=0}^{N-1} A^j B u_{N-1-j} \in \mathcal{X}_f \\ & u_k \in \mathcal{U} \\ & x_0 = x(0) \end{aligned}$$

- **Recall: ∞-Norm State Feedback Solution using Multi-parametric Programming**

Step 1: The above formulation can be solved by stacking up the decision variable as doing transform:

$$\min_{z_0} \ c_0^{\mathrm{T}} z_0$$

$$\text{subj. to. } \bar{G}_0 z_0 \leq \bar{w}_0 + \bar{S}_0 x(0)$$

Where $z_0 = \{\varepsilon_0^x, \ldots, \varepsilon_N^x, \varepsilon_0^u \ldots, \varepsilon_{N-1}^u, u_0^{\mathrm{T}}, \ldots, u_{N-1}^{\mathrm{T}}\} \in \mathbb{R}^s$, $s \triangleq (m+1)N + N + 1$, and

$$\bar{G}_0 = \begin{bmatrix} G_\varepsilon & 0 \\ 0 & G_0 \end{bmatrix}, \ \bar{S}_0 = \begin{bmatrix} S_\varepsilon \\ S_0 \end{bmatrix}, \ \bar{w}_0 = \begin{bmatrix} w_\varepsilon \\ w_0 \end{bmatrix}$$

It is not hard to see that this is a mp-LP problem, so our solution procedure is also:

Step 2: Solve the mp-LP to get the explicit solution $z^*(x(0))$

Step 3: Obtain $U_0^*(x(0))$ from $z^*(x(0))$

Main result:

1. The **open loop optimal control function** can be obtained by solving the mp-LP problem and calculating $z_0^*(x(0))$
2. The component $u_0^* = [0 \ \ldots \ 0 \ \ I_m \ \ 0 \ \ldots \ 0] z_0^*(x(0))$ of the multiparametric solution has the following form:

   $$u^*(0) = f_0(x(0)), \ \forall x(0) \in \mathcal{X}_0$$

   And $f_0 : \mathbb{R}^n \to \mathbb{R}^m$ is continuous and piecewise affine on polyhedra

   $$f_0(x) = F_0^i x + g_0^i \ \text{ if } \ x \in \mathcal{CR}_0^i, \ i = 1, \ldots, N_0^r$$
3. The polyhedral sets $\mathcal{CR}_0^i = \{x \in \mathbb{R}^n \mid H_0^i x \leq K_0^i\}$, $i = 0 \cdots, N_0^r$ are a partition of the feasible polyhedron $\mathcal{X}_0$
4. In the case of multiple optimizers, a continuous piecewise affine control law exists. (If objective in LP is parallel to the constraint, then it might be infinitely mainly solutions, and we should properly make it become continuous)
5. The value function $J_0^*(x(0))$ is convex and piecewise linear on polyhedra.

- **Summary:**

To summarize the above two sections, we in fact try to answer the following question:

Why is it so significant to emphasize the multi-parametric programming and its result in the context of CFTOC, even MPC?

1. **Theoretically, it can be regarded as an extension of LQR to constrained case**

   The first and biggest motivation starts from the LQR control. In LQR (or LQG), no matter whether we use the forward approach (shooting or collocation) or backward approach (dynamic programming), we all return to the conclusion that the controller is a linear controller and the optimal cost is quadratic. In other words, we explicitly derive the specific form of the controller and optimal cost.

However, LQR can only be applied to systems without constraints. In the case of constrained control, we also want to take a step further into exploring the specific form of the controller and optimal cost. Luckily, using multi-parametric programming, we know that the controller is piecewise affine and the optimal cost is piecewise quadratic.

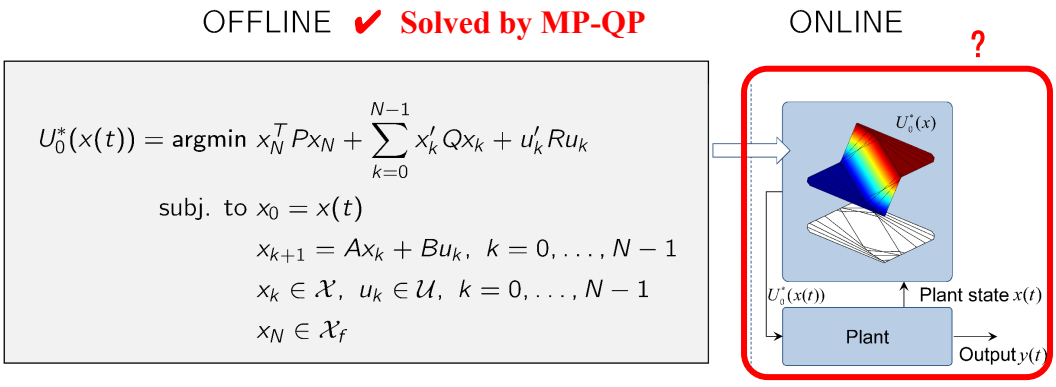
2. **Practically, it can perfectly fit into the MPC scheme**

   The determination of a specific form of the controller and optimal cost brings not only theoretical value, but also benefits practice. It is because MPC almost did everything implicitly, i.e., generally, things can not be determined in advance but need online computing, which is also a big challenge in MPC application. With the specific form of controller and optimal cost being determined, we can now turn part of the MPC from implicit to explicit, and do the calculations in advance. This idea leads to the explicit MPC, which would be introduced in detail in the following sections.

   Also, the MP-programming tells us the controller and optimal cost are parametrized by the initial state. At first glance, this result means open-loop since it only relies on the initial state. But note that MPC only executes the first step of the computed control sequence and regards the next time step as the new initial step, so every calculation is about the initial state. In this sense, the result is no longer open-loop, but closed-loop! We get a feedback policy for every time step, which is much more powerful than just an open-loop control. All these have to attribute to the nature of the MPC scheme: receding horizon fashion.

IV. Explicit MPC: Online Evaluation

Again, we return to this graph:



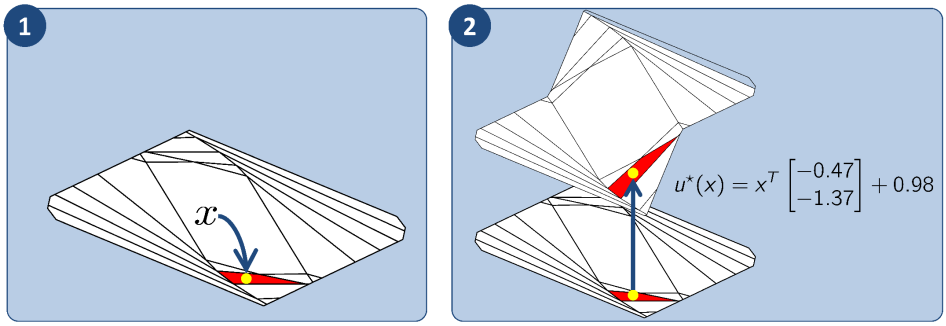
In the framework we introduced at the beginning of the lecture, the left part is now done since MP-Programming proved that the controller and optimal cost can be solved in advance. The following discussion is based on the assumption that we have stored all the critical regions and corresponding controller and optimal cost., i.e., the offline phase is completed.

- **Online evaluation: Point location**

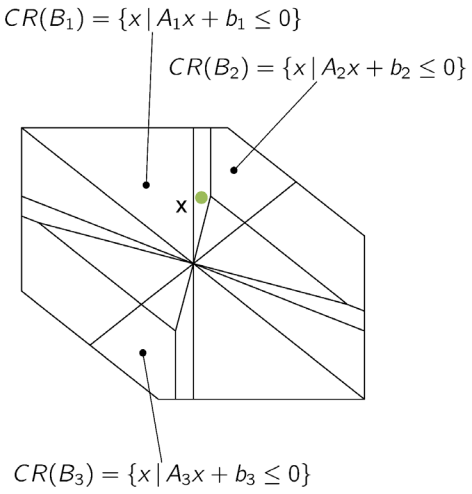
It is not hard to see the rest to be done are two things, as the following figure shows:

1. Point location, i.e. find the critical region given an initial point
2. Evaluation of affine function to get control and quadratic function to get optimal cost

The hard part should be the point location since the evaluation is simply matrix multiplication.



Method 1: Sequential Search (brute force)



The first and very naïve way is to search the regions one by one. i.e., sequential search for each $\mathcal{CR}_i$, if $A_i x \leq b_i$ then x is in region $\mathcal{CR}_i$

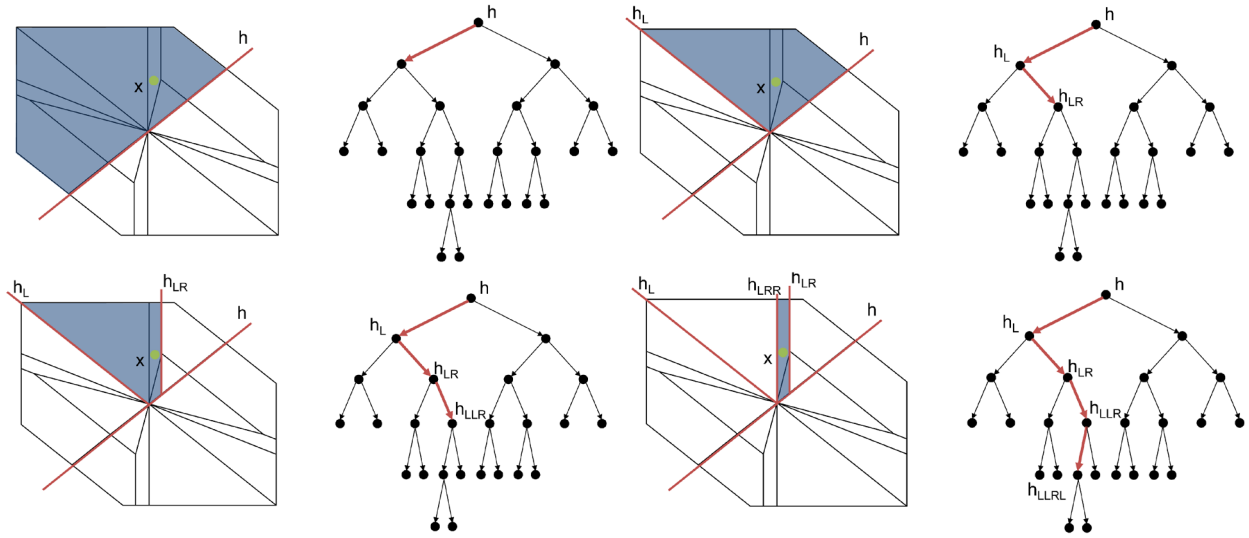
This search is very simple, and the computation complexity is apparently linear in the number of regions.

Note: there are also a lot of tricks to speed up the search, e.g.:

1. The state is not likely to "jump" very far if the system is not hybrid, so we can add some heuristics in the search to first find the neighboring regions
2. The computing can be done parallelly if hardware permits.

Method 2: Logarithmic Search

The second method is that we offline construction of the search tree by finding a hyperplane that separates regions into two equal-sized sets and repeating for left and right sets. Therefore, each time we check the point, we filter out half of the not corrected region, so the complexity is logarithmic.



Note: However, this requires pre-sorting those regions offline, and unfortunately, the computation for dividing the regions as described above is combinatorial or exponential in the number of regions.

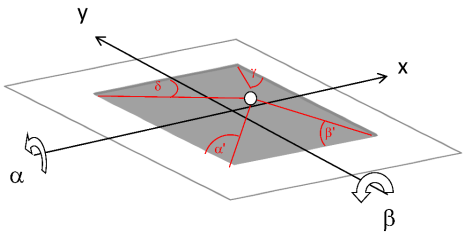
Summary:

Sequential search: Very simple; Works for all problems

Search tree: Potentially logarithmic; Significant offline processing (reasonable for < 1000 regions)

There are not many other options for special cases

- **Example: Ball and Plate**

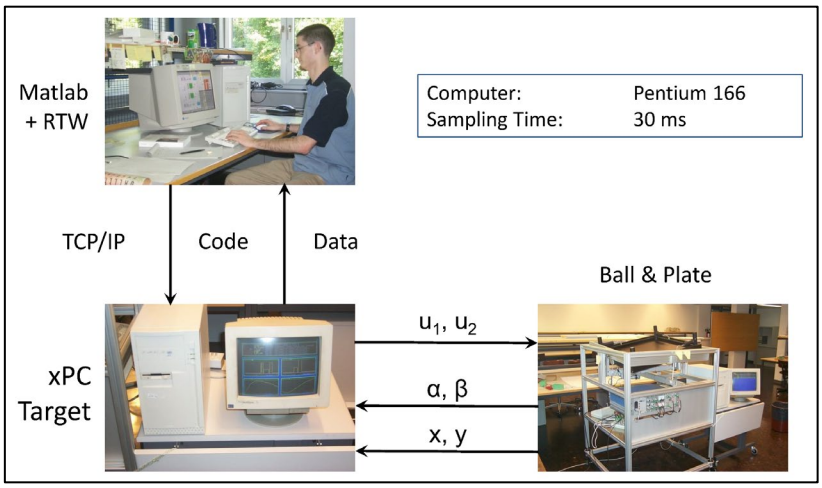


Linearized model: four states for each axis: plate angle, ball position, plate angular speed, and ball speed.

Constraints on inputs and states: Plate angle, Ball position, Plate tilting acceleration

MPC objective: path tracking

The hardware system overview is as follows:



Problem Formulation and Solution:

4 states + 1 tracking variable = 5 parameters

Move-blocking reduces complexity: Horizon of 10. Inputs 2-10 must be equal.

MPT

```
% Linear discrete-time prediction model
model=LTISystem('A', A, 'B', B, 'C', C);

% Input constraints
model.u.min = -10; model.u.max = 10;

% Output constraints
model.y.min = -30; model.y.max = 30;

% State constraints
model.x.min = [-30; -15; -15*pi/180; -1];
model.x.max = [30; 15; 15*pi/180; 1];

% Penalties in the cost function
model.y.penalty = QuadFunction(100);
model.u.penalty = QuadFunction(0.1);

% Adjustment via input blocking
model.u.with('block'); model.u.from = 1; model.u.to = 9;

% Time varying reference signal
model.y.with('reference'); model.y.reference = 'free';
```

```
% Online MPC object
online_ctrl = MPCController( model, 9 )
```

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

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

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

$$y_i = Cx_i$$

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

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

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

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

Explicit solution (per dimension)

Regions : 529

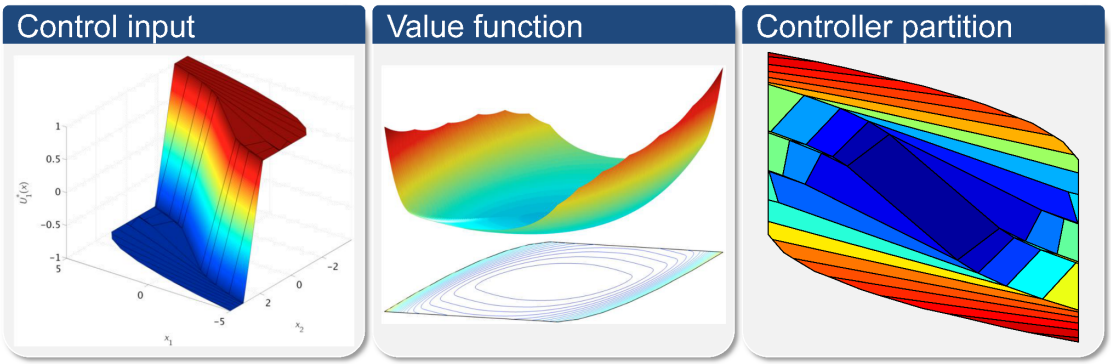
Storage : 48'000 numbers (192 kB)

Computation : 89'000 FLOPS (~1ms)

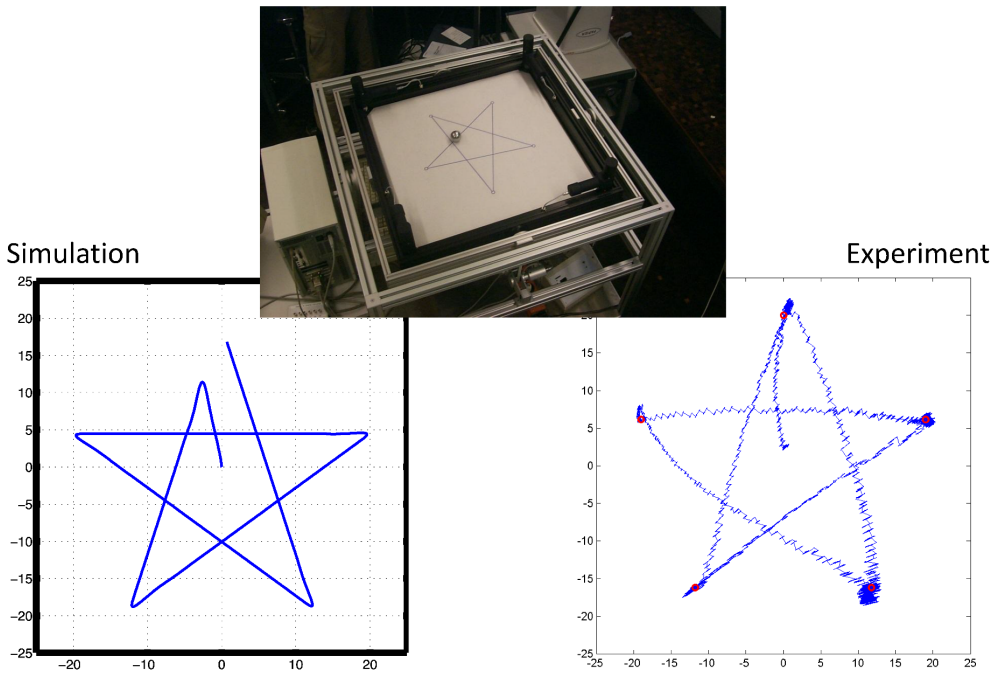
```
% Plot control law (primal solution)
explicit_ctrl.optimizer.fplot('primal')
```

```
% plot the objective function
explicit_ctrl.optimizer.fplot('obj')
```

```
% Plot controller partition
explicit_ctrl.optimizer.plot
```



Experiment Results:



V. Summary:

1. Linear MPC + Quadratic or linear-norm cost ⇒ Controller is a PWA function
2. We can pre-compute this function offline efficiently
3. Online evaluation of a PWA function is very fast (ns-μs)
4. We can only do this for relatively low-dimensional systems! (3-6 states). For larger systems, either calculating the critical regions would be too hard, or finding the region where the state lies would be too hard. Clever choices must be made depending on specific hardware and problem target.

Note:

1. This is to say, in real applications, the first choice would always be doing online optimization, especially nowadays, we have more powerful computation resources than before.
2. Also, explicit MPC is not flexible. If the parameters of the model are slightly tuned or perturbed, the whole process needs to be re-done