Recap: Model Predictive Control (MPC)

- MPC solves a N-step lookahead problem (called **Optimal Control** problem) in a receding horizon fashion, in order to approximate the infinite-horizon DP:

Terminal Cost Approximation

$$J_0(\bar{x}_0) = \min_{X,U} \hat{J}_N(x_N) + \sum_{i=0}^{N-1} g_i(x_i, u_i)$$

$$\begin{aligned}
\text{s.t. } & x_{i+1} = f_i(x_i, u_i), \quad \forall i \in \{0, 1, 2, ...\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0, 1, 2, ...\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0, 1, 2, ...\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}$$

Terminal set constraint

Establishing MPC Properties

- We saw, last time, that for the following properties to hold:
  - **(1) Recursive Feasibility**
  - **(2) Asymptotic Stability**

- The terminal set $\mathcal{X}_f$ is a **invariant set** under some local control policy $v(x_k)$:

$$x_{k+1} = f(x_k, v(x_k)) \in \mathcal{X}_f, \, \forall x_k \in \mathcal{X}_f$$

$$\mathcal{X}_f \subseteq \mathcal{X}, \, v(x_k) \in \mathcal{U}, \, \forall x_k \in \mathcal{X}_f$$

- And the terminal cost approximation needs to be a Lyapunov Function over the terminal set $\mathcal{X}_f$

$$\hat{J}_k(x_{k+1}) - \hat{J}_k(x_k) \leq -g(x_k, v(x_k)), \, \forall x_k \in \mathcal{X}_f$$

Linear MPC

- Consider the Linear MPC case:

$$
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \hat{J}_N(x_N) + \sum_{i=0}^{N-1} x_i^\top Q x_i + u_i^\top R u_i \\
\text{s.t.} \quad & x_{i+1} = Ax_i + Bu_i, \quad \forall i \in \{0,1,2,...\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0,1,2,...\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0,1,2,...\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}
$$

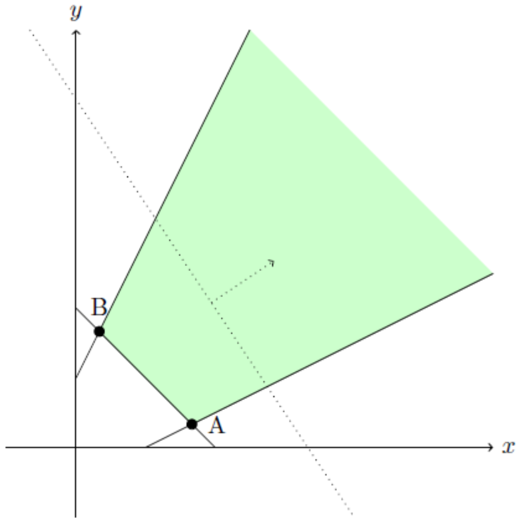
- Where the constraints are Polyhedrons. We will study today how to construct invariant sets for the linear case.

Polyhedrons and Polytopes

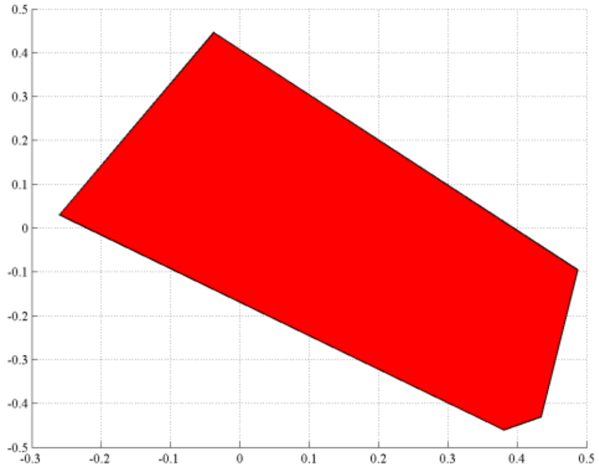
- A set $\mathcal{S}$ is a polyhedron if it can be described by a finite number of closed halfspaces:

$$\mathcal{S} = \{x \in \mathbb{R}^n : d_1^\top x \leq b_1, ..., d_m^\top x \leq b_m\} = \{x \in \mathbb{R}^n : Dx \leq b\}$$

- And a polytope is a bounded polyhedron:



(unbounded) polyhedron



Polytope (bounded polyhedron)

Representation of Polytopes

- There are two ways we can represent polytopes:

- (1) H-representation: we represent by the halfspaces:

$$\mathcal{S} = \{x \in \mathbb{R}^n : Dx \leq b\}$$

- (2) V-representation: we represent by convex combination of it's extreme points:

$$\mathcal{S} = \{x \in \mathbb{R}^n : x = \sum_{k=1}^{K} \lambda_k \bar{x}_k, \quad \sum_{k=1}^{K} \lambda_k = 1, 0 \leq \lambda_k \leq 1\}$$

- Where $(\bar{x}_0, \ldots, \bar{x}_K)$ are it's extreme points.

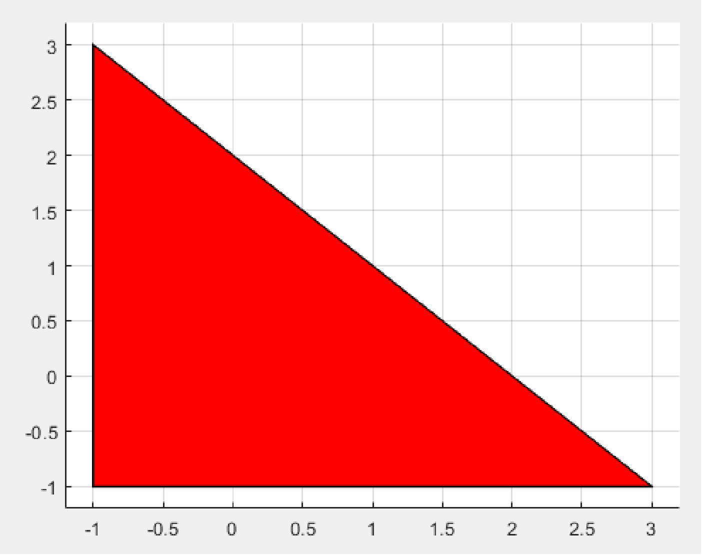
Example:Polytopes

- Consider the following polytope, with the H-represention:

$$\mathcal{S} = \{x \in \mathbb{R}^2 : x_1 + x_2 \leq 2, -x_1 \leq 1, -x_2 \leq 1\}$$

- We can write the V-representation as follows:

$$\mathcal{S} = \{x \in \mathbb{R}^2 : x = \lambda_1 \begin{bmatrix} 3 \\ -1 \end{bmatrix} + \lambda_2 \begin{bmatrix} -1 \\ 3 \end{bmatrix} + \lambda_3 \begin{bmatrix} -1 \\ -1 \end{bmatrix},$$
$$\lambda_1 + \lambda_2 + \lambda_3 = 1, 1 \geq \lambda_1, \lambda_2, \lambda_3 \geq 0\}$$



Operations on Polytopes

- Consider the following polytope:

$$\mathcal{P} = \{(x, y) \in \mathbb{R}^{n+m} : D^x x + D^y y \leq d\}$$

- A projection of a polytope is the following set:

$$\mathrm{proj}_x(\mathcal{P}) = \{x \in \mathbb{R}^n : \exists y \in \mathbb{R}^m : D^x x + D^y y \leq d\}$$

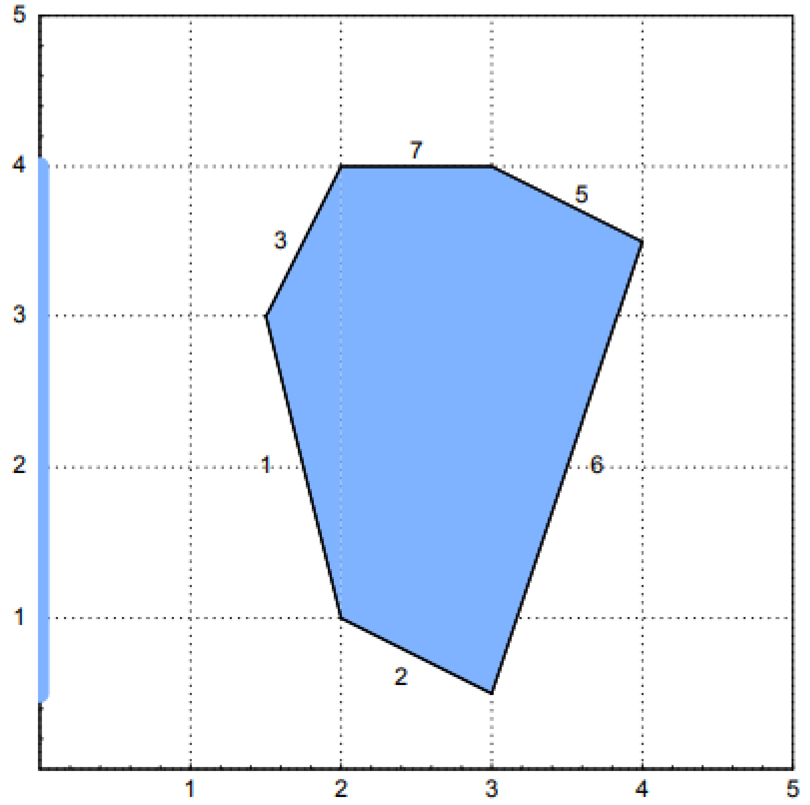
- It turns out that a projection of polytope is also a polytope.

- A Projection of polytope can be obtained by many different methods:
    - e.g.: Fourier-Motzkin elimination

Polyhedrons and Polytopes

- Consider the polytope given by the following inequalities

$$
\begin{aligned}
-4x_1 - x_2 &\leq -9 && (1)\\
-x_1 - 2x_2 &\leq -4 && (2)\\
-2x_1 + x_2 &\leq 0 && (3)\\
-x_2 - 6x_2 &\leq -6 && (4)\\
x_1 + 2x_2 &\leq 11 && (5)\\
6x_1 + 2x_2 &\leq 17 && (6)\\
x_2 &\leq 4 && (7)
\end{aligned}
$$



Operations on Polytopes

- We can combine polytope with affine mappings. Suppose we have a polytope:

$$\mathcal{S} = \left\{ x \in \mathbb{R}^n : Dx \leq d \right\}, \forall k \geq 0$$

- And an affine mapping:

$$m : x \in \mathbb{R}^n \rightarrow Ax + b$$

- Then the composition $\mathcal{S} \circ m$ is also a polyhedron:

$$\mathcal{S} \circ m = \{x \in \mathbb{R}^n : DAx \leq d - Db\}$$

Operations on Polytopes

- Consider the following Polytope

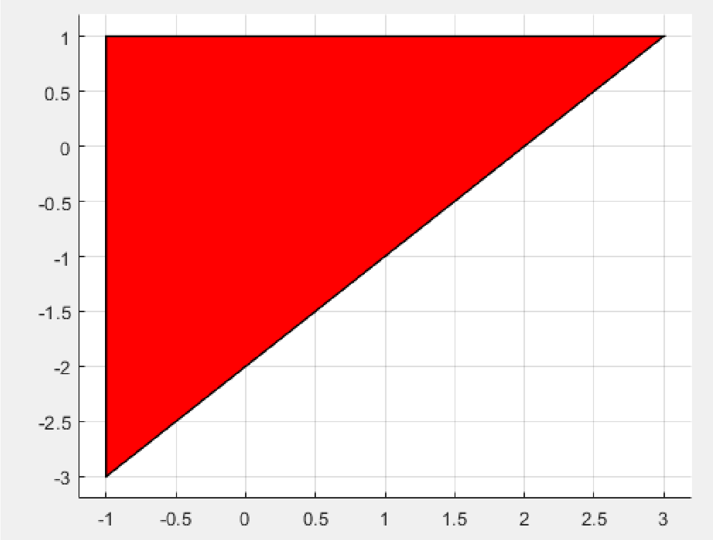
$$\mathcal{S} = \{x \in \mathbb{R}^2 : x_1 + x_2 \leq 2, -x_1 \leq 1, -x_2 \leq 1\}$$

- And an affine mapping (a clock-wise rotation):

$$m : x \in \mathbb{R}^n \rightarrow \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} x$$

- Then the new polyhedron is:

$$\mathcal{S} \circ m = \{x \in \mathbb{R}^2 : x_1 - x_2 \leq 2, -x_1 \leq 1, x_2 \leq 1\}$$



Operations on Polytopes

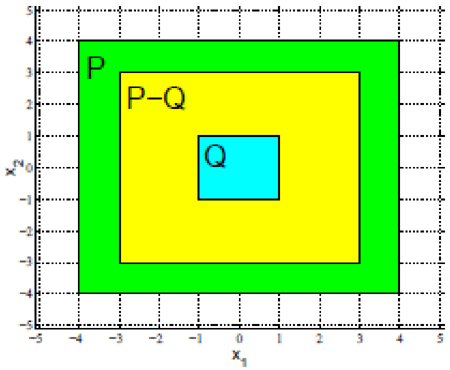
- The Pontryagin Difference (Minkowski difference) of two polytopes $\mathcal{P}$ and $\mathcal{R}$:

$$\mathcal{P} \ominus \mathcal{R} = \{x \in \mathbb{R}^n : x + y \in \mathcal{P}, \forall y \in \mathcal{Q}\}$$

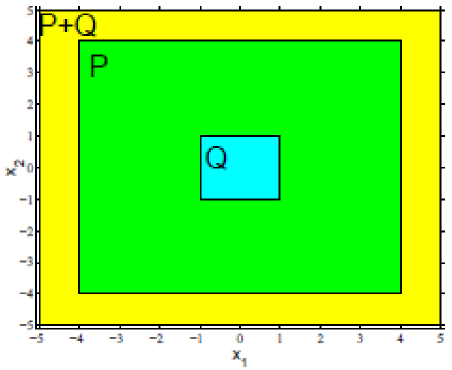
- And the Minkowski sum of two polytopes $\mathcal{P}$ and $\mathcal{R}$:

$$\mathcal{P} \oplus \mathcal{R} = \{x \in \mathbb{R}^n : \exists y \in \mathcal{P}, \exists z \in \mathcal{R}, x = y + z\}$$

- These operations are illustrated below:



(a) Pontryagin difference $\mathcal{P} \ominus \mathcal{Q}$.



(b) Minkowski sum $\mathcal{P} \oplus \mathcal{Q}$.

(figure from Borelli and Morari)

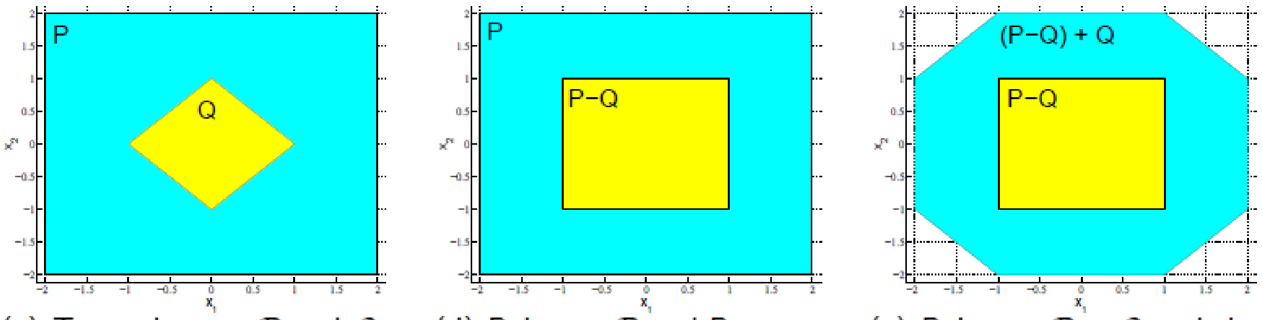
Polyhedrons and Polytopes

- The Minkowski operations are set operations and they behave differently from scalar operations.

- For example it holds, that in general:

$$(\mathcal{P} \ominus \mathcal{R}) \oplus \mathcal{R} \subseteq \mathcal{P}$$

- Which can be a strict inclusion as shown by the example:



(c) Two polytopes $\mathcal{P}$ and $\mathcal{Q}$. (d) Polytope $\mathcal{P}$ and Pontryagin difference $\mathcal{P} \ominus \mathcal{Q}$. (e) Polytope $\mathcal{P} \ominus \mathcal{Q}$ and the set $(\mathcal{P} \ominus \mathcal{Q}) \oplus \mathcal{Q}$.

(figure from Borelli and Morari)

Operations on Polytopes

- The Minkowski sum of two polytopes $\mathcal{P}$ and $\mathcal{R}$:

$$\mathcal{P} = \{y \in \mathbb{R}^n : D^y y \leq d^y\} \quad \mathcal{R} = \{z \in \mathbb{R}^n : D^z z \leq d^z\}$$

- It holds that

$$\begin{aligned}
\mathcal{P} \oplus \mathcal{R} = & \\
&= \{x \in \mathbb{R}^n : \exists y : \ D^y y \leq d^y, \exists z : \ D^z z \leq d^z, x = y + z\} \\
&= \{x \in \mathbb{R}^n : \exists y : \ D^y y \leq d^y, D^z(x - y) \leq d^z\} \\
&= \{x \in \mathbb{R}^n : \exists y : \begin{bmatrix} 0 & D^y \\ D^z & -D^z \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} d^y \\ d^z \end{bmatrix}\} \\
&= \mathrm{proj}_x\big(\{(x, y) \in \mathbb{R}^{n+n} : \begin{bmatrix} 0 & D^y \\ D^z & -D^z \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} d^y \\ d^z \end{bmatrix}\}\big)
\end{aligned}$$

Operations on Polytopes

- The Minkowski difference can be computed for two polytopes $\mathcal{P}$ and $\mathcal{R}$:

$$\mathcal{P} = \{y \in \mathbb{R}^n : D^y y \leq d^y\} \quad \mathcal{R} = \{z \in \mathbb{R}^n : D^z z \leq d^z\}$$

- As follows:

$$\mathcal{P} \ominus \mathcal{R} = \{x \in \mathbb{R}^n : D^y x \leq d^y - H(D^z, \mathcal{R})\}$$

- Where i-th element of $H(D^z, \mathcal{R})$ is:

$$H_i(D^z, \mathcal{R}) = \max_{x \in \mathcal{R}} \{D_i^y x\}$$

- Which is a Linear Program, which can be solved very efficiently.

Predecessor Set: Autonomous System

- Now let's return to our linear dynamics:

$$x_{k+1} = Ax_k, \forall k \geq 0$$

- Subject to the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^n : Hx \leq h \right\}, \forall k \geq 0$$

- Then the precursor set $P(\mathcal{X})$ that is the set of all points that if we start at $P(\mathcal{X})$ we do not leave $\mathcal{X}$:

$$P(\mathcal{X}) = \{x \in \mathbb{R}^n : Ax \in \mathcal{X}\}$$

Predecessor Set Computation

- So we write the Predecessor set as:

$$P(\mathcal{X}) = \{x \in \mathbb{R}^n : Ax \in \mathcal{X}\} = \{x \in \mathbb{R}^n : HAx \leq h\}$$

- So the Predecessor set is in essence a composition a Linear Transformation with a Polytope:

- **(1)** Linear transformation $x \to Ax$

- **(2)** Polytope: $P(\mathcal{X}) = \{x \in \mathbb{R}^n : HAx \leq h\}$

- So we can write compactly:

$$P(\mathcal{X}) = \mathcal{X} \circ A$$

Example: 2nd-order autonomous system

- Consider the following system:

$$x_{k+1} = Ax_k = \begin{bmatrix} 1.5 & 0 \\ 1 & -1.5 \end{bmatrix} x_k$$

- Subject to the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

- Let's find a set $P(\mathcal{X})$ that is the set of all points that if we start at $P(\mathcal{X})$ we do not leave $\mathcal{X}$:

$$P(\mathcal{X}) = \{ x \in \mathbb{R}^2 : Ax \in \mathcal{X} \}$$

Example: 2nd-order autonomous system

- To find P, let's re-write the constraints as follows:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

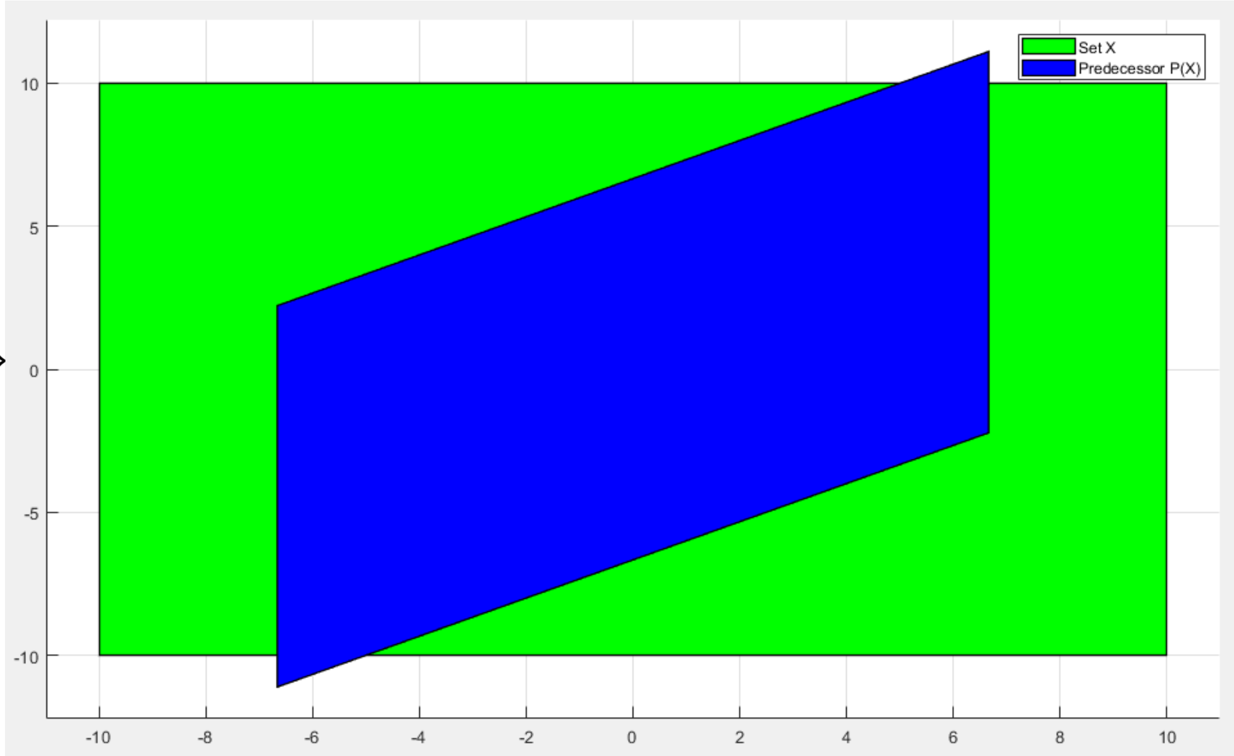
- And we write:

$$P(\mathcal{X}) = \mathcal{X} \circ A = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 1.5 & 0 \\ 1 & -1.5 \\ -1.5 & 0 \\ 0 & 1.5 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}$$

Example: 2^{nd}-order autonomous system

- This amount to the figure:

$$P(\mathcal{X}) = \mathcal{X} \circ A = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 1.5 & 0 \\ 1 & -1.5 \\ -1.5 & 0 \\ 0 & 1.5 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}$$



Predecessor Set: System with inputs

- Now let's focus on the following system:

$$x_{k+1} = Ax_k + Bu_k, \forall k \geq 0$$

- Subject to the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^n : Hx \leq h \right\}, \forall k \geq 0$$

$$u_k \in \mathcal{U} = \left\{ u \in \mathbb{R}^m : H_u u \leq h_u \right\}, \forall k \geq 0$$

Predecessor Set: System with inputs

- The we write the Predecessor set as:

$$P(\mathcal{X}) = \left\{x \in \mathbb{R}^n : \;\; \exists u \in \mathbb{R} \text{ s.t.: } \begin{bmatrix} HA & HB \\ 0 & H_u \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} h \\ h_u \end{bmatrix} \right\}$$

- But note that the set above is actually a projection of the following polyhedron:

$$\mathcal{Y} = \left\{(x, u) \in \mathbb{R}^{n+m} : \;\; \begin{bmatrix} HA & HB \\ 0 & H_u \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} h \\ h_u \end{bmatrix} \right\}$$

- So we can write:

$$\mathcal{P}(\mathcal{X}) = \text{proj}_x(\mathcal{Y})$$

- In addition, there are compositions between linear transformations.

Predecessor Set: System with inputs

- Then we can equivalently write:

$$
\begin{aligned}
P(\mathcal{X}) &= \left\{x : \ \exists u \in \mathcal{U} \text{ s.t.: } Ax + Bu \in \mathcal{X}\right\} \\
&= \left\{x : \ \exists z \in \mathcal{X}, \exists u \in \mathcal{U} \text{ s.t.: } Ax = z - Bu\right\} \\
&= \left\{x : \ Ax = \mathcal{X} \oplus (-B) \circ \mathcal{U}\right\}
\end{aligned}
$$

- So we can write compactly:

$$P(\mathcal{X}) = (\mathcal{X} \oplus (-B \circ \mathcal{U})) \circ A$$

Example: 2^{nd}-order unstable system

- Consider the following unstable system:

$$x_{k+1} = Ax_k + Bu_k = \begin{bmatrix} 1.5 & 0 \\ 1.0 & -1.5 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_k$$

- Subject to the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

$$u_k \in \mathcal{U} = \left\{ u \in \mathbb{R} : -5 \leq u \leq 5 \right\}, \forall k \geq 0$$

Example: 2nd-order unstable system

- Again consider the Precursor set $P(\mathcal{X})$

$$P(\mathcal{X}) = \left\{x \in \mathbb{R}^2 : \; \exists u \in \mathcal{U} \text{ s.t.: } Ax + Bu \in \mathcal{X}\right\}$$

- Again let's rewrite the constraint sets:

$$x_k \in \mathcal{X} = \left\{x \in \mathbb{R}^2 : \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

$$u_k \in \mathcal{U} = \left\{u \in \mathbb{R} : \begin{bmatrix} 1 \\ -1 \end{bmatrix} u \leq \begin{bmatrix} 5 \\ 5 \end{bmatrix} \right\}$$

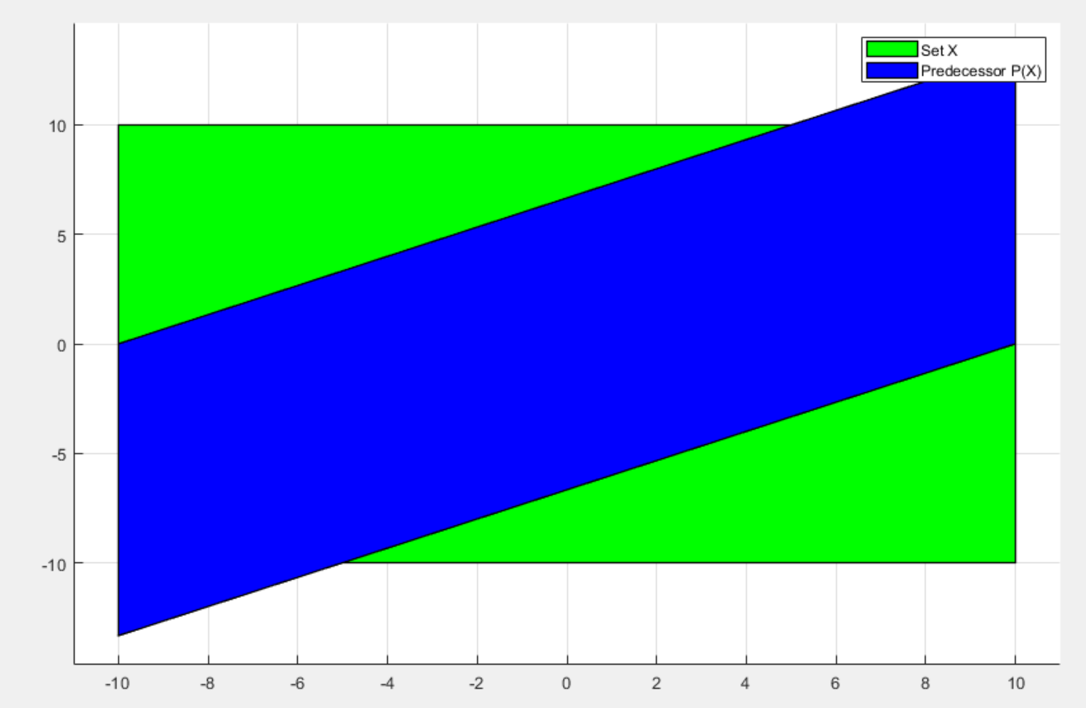
Example: 2nd-order unstable system

- So we can write:

$$P(\mathcal{X}) = (\mathcal{X} \oplus (-B \circ \mathcal{U})) \circ A$$

- Obtaining:

$$P(\mathcal{X}) = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 1.5 & 0 \\ -1.5 & 0 \\ -1 & 1.5 \\ 1 & -1.5 \end{bmatrix} x \leq \begin{bmatrix} 15 \\ 15 \\ 10 \\ 10 \end{bmatrix} \right\}$$



Invariant Set Computation

- Now we are ready to provide an Algorithm to compute Invariant sets for Linear MPC.

- A set $\mathcal{O} \subseteq \mathcal{X}$ **positively invariant** for any closed-loop systems $x_{k+1} = Ax_k$ we have:

$$x_0 \in \mathcal{O} \Rightarrow x_k \in \mathcal{O},\ k \geq 1$$

- In words: "a set is invariant if once our systems starts from it, it stays in it forever."

- And recall that the largest of such sets is called the maximal positively invariant set, and we called it $\mathcal{O}_\infty$.

Invariant Set Computation

- In turns out that there exists a way to "test" if a set is invariant or not:

- A set $\mathcal{O} \subseteq \mathcal{X}$ is positively invariant **if and only if:**

$$\mathcal{O} \subseteq \mathcal{P}(\mathcal{O})$$

- And note that:

$$\mathcal{O} \subseteq \mathcal{P}(\mathcal{O}) \Leftrightarrow \mathcal{P}(\mathcal{O}) \cap \mathcal{O} = \mathcal{O}$$

- The idea now is to generate a sequence of sets $\{\Omega_k\}_{k=1}^{\infty}$, such that $\Omega_{k+1} \subseteq \Omega_k$ for all $k \geq 0$. And terminate when:

$$\mathcal{P}(\Omega_k) \cap \Omega_k = \Omega_k$$

Invariant Set Computation

- We state the Algorithm below:

Algorithm 1 Algorithm for Invariant Set Computation

Input: Linear system matrix A and state constraint set $\mathcal{X}$

1: Let $\Omega_0 = \mathcal{X}$
2: **for** $k = 0, 1, 2, 3...$ **do** (obtaining new samples)
3: Let: $\Omega_{k+1} = \mathcal{P}(\Omega_k) \cap \Omega_k$
4: If $\Omega_{k+1} = \Omega_k$, Set $\mathcal{O}_\infty \leftarrow \Omega_{k+1}$; Then break
5: **end for**

Output: The Maximal Invariant Set $\mathcal{O}_\infty$

Example: 2nd-order autonomous system

- Consider again the following system:

$$x_{k+1} = Ax_k = \begin{bmatrix} 0.5 & 0 \\ 1 & -0.5 \end{bmatrix} x_k$$

- Subject to the following constraints:

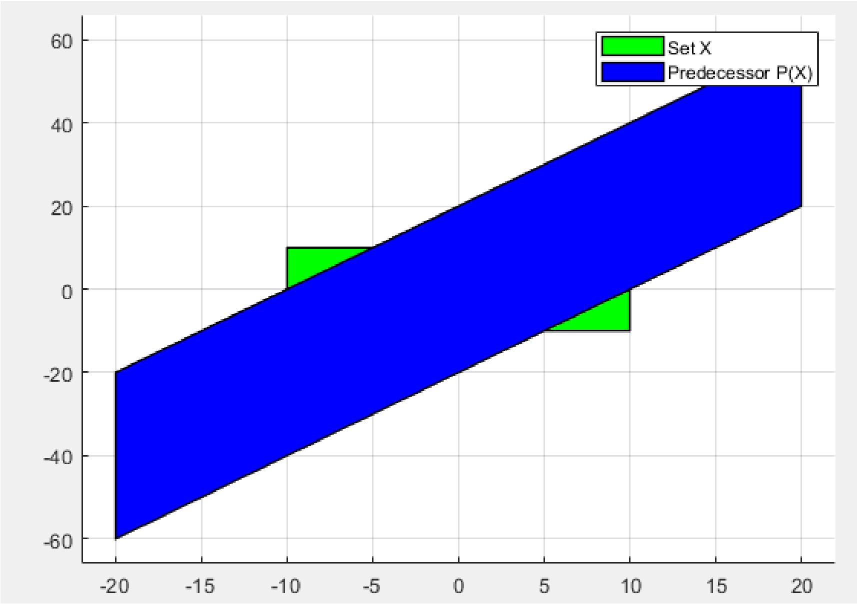
$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

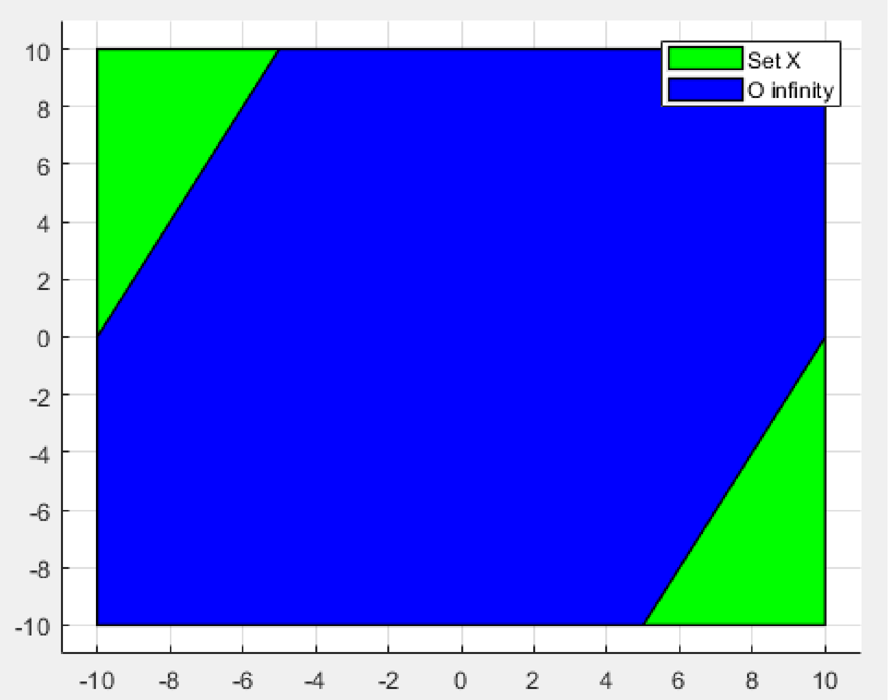
- We apply the Algorithm to obtain the maximal invariant set associated with this system.

Example: 2nd-order autonomous system

- This amount to:

$$\mathcal{O}_\infty = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 0.89 & -0.45 \\ -0.89 & 0.45 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} x \leq \begin{bmatrix} 8.94 \\ 8.94 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}$$





Control Invariant Set Computation

- This algorithm can be extended to Control Invariant sets.

- Recall that a set $\mathcal{C} \subseteq \mathcal{X}$ is **<u>control invariant</u>** for a system $x_{k+1} = Ax_k + Bu_k$ if:

$$x_0 \in \mathcal{C} \Rightarrow \exists u_k \in \mathcal{U} \text{ s.t.: } x_k \in \mathcal{C},\ k \geq 1$$

- In words: "a set is control invariant if once our systems starts from it, there is always a feasible control such that once applied the system inside the set."

- And the largest of such sets is the maximal control invariant set, and we called it $\mathcal{C}_\infty$.

Invariant Set Computation

- As before, a set $\mathcal{C} \subseteq \mathcal{X}$ is control invariant if and only if:

$$\mathcal{C} \subseteq \mathcal{P}(\mathcal{C})$$

- Then we can state a very similar Algorithm to compute control invariant sets:

Algorithm 1 Algorithm for Invariant Set Computation

Input: Linear system matrix A and state constraint set $\mathcal{X}$

1: Let $\Omega_0 = \mathcal{X}$

2: **for** $k = 0, 1, 2, 3...$ **do** (obtaining new samples)

3: Let: $\Omega_{k+1} = \mathcal{P}(\Omega_k) \cap \Omega_k$

4: If $\Omega_{k+1} = \Omega_k$, Set $\mathcal{C}_\infty \leftarrow \Omega_{k+1}$; Then break

5: **end for**

Output: The Maximal Control Invariant Set $\mathcal{C}_\infty$

Example: 2nd-order unstable system

- Consider again the following unstable system:

$$x_{k+1} = Ax_k + Bu_k = \begin{bmatrix} 1.5 & 0 \\ 1.0 & -1.5 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_k$$

- Subject to the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \le \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \ge 0$$

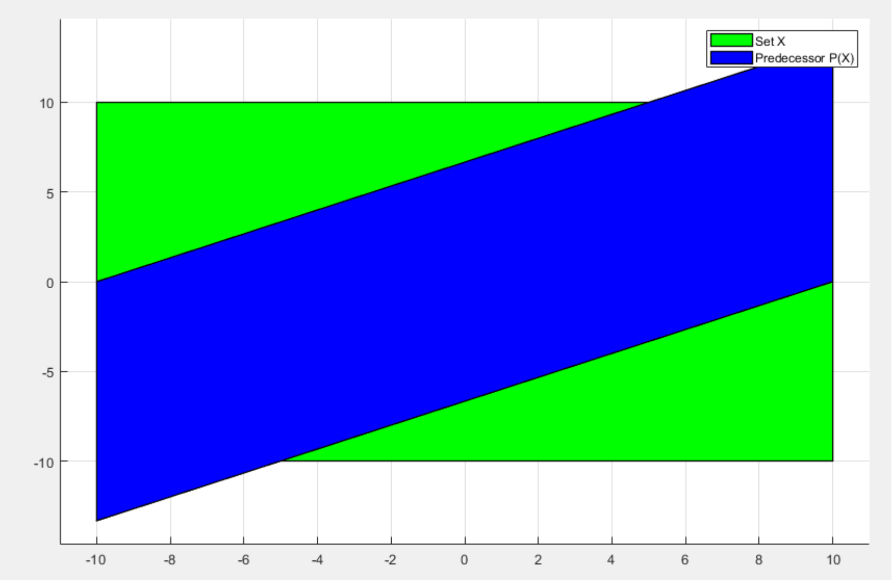
$$u_k \in \mathcal{U} = \left\{ u \in \mathbb{R} : -5 \le u \le 5 \right\}, \forall k \ge 0$$

- We apply the Algorithm as before.

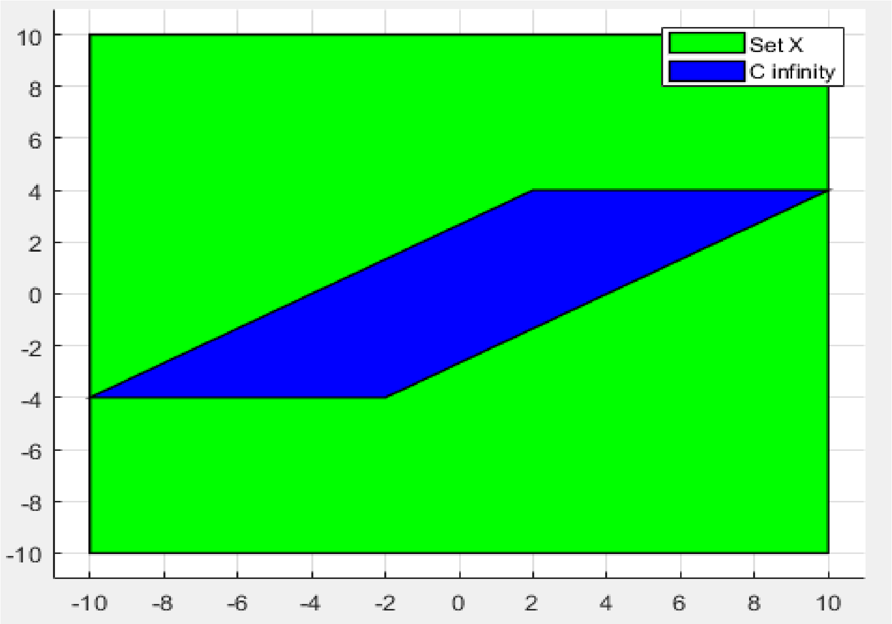
Example: 2nd-order unstable system

- Which amounts to:

$$\mathcal{C}_\infty = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 0 & 1 \\ 0 & -1 \\ 0.55 & -0.83 \\ -0.55 & -0.83 \\ 1 & 0 \\ -1 & 0 \end{bmatrix} x \leq \begin{bmatrix} 4 \\ 4 \\ 2.22 \\ 2.22 \\ 6.67 \\ 6.67 \end{bmatrix} \right\}$$



After 36 iterations



Invariant Set Computation in MPC

- Let's return to the MPC problem:

$$
\begin{aligned}
J_0(\bar{x}_0) = \min_{X,U} \quad & \hat{J}_N(x_N) + \sum_{i=0}^{N-1} x_i^\top Q x_i + u_i^\top R u_i \\
\text{s.t. } \; & x_{i+1} = A x_i + B u_i, \quad \forall i \in \{0, 1, 2, ...\} \\
& x_i \in \mathcal{X}, \quad \forall i \in \{0, 1, 2, ...\} \\
& u_i \in \mathcal{U}, \quad \forall i \in \{0, 1, 2, ...\} \\
& x_0 = \bar{x}_0 \\
& x_N \in \mathcal{X}_f
\end{aligned}
$$

- We saw last time that a good choice for the terminal cost function approximation

$$\hat{J}_N(x_N) = x_N^\top P x_N$$

Invariant Set Computation in MPC

- Where the matrix P is obtained by solving the unconstrained infinite-horizon LQR:

$$K = -(B^\top KB + R)^{-1}B^\top PA$$

$$P = A^\top(P - PB(B^\top PB + R)^{-1}B^\top P)A + Q$$

- And we take $\mathcal{X}_f$ to be the maximum invariant set for the closed-loop system:

$$x_{k+1} = (A + BK)x_k \in \mathcal{X}_f, \quad \forall k \in \{0, 1, 2, ...\}$$

$$\mathcal{X}_f \subseteq \mathcal{X},\, Kx_k \in \mathcal{U},\, \forall x_k \in \mathcal{X}_f$$

- Which can be computed by our the invariant set computation Algorithm.

Example: Linear MPC

- Let's consider again the following system:

$$x_{k+1} = Ax_k + Bu_k = \begin{bmatrix} 1.5 & 0 \\ 1.0 & -1.5 \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u_k$$

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

$$u_k \in \mathcal{U} = \left\{ u \in \mathbb{R} : -5 \leq u \leq 5 \right\}, \forall k \geq 0$$

- With the following cost matrices:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = 1$$

Example: Linear MPC

- First we solver the Algebraic Ricatti Equation:

$$K = -(B^\top K B + R)^{-1} B^\top P A$$

$$P = A^\top (P - PB(B^\top P B + R)^{-1} B^\top P) A + Q$$

- And we obtain:

$$K = \begin{bmatrix} -0.21 & -1.69 \end{bmatrix} \qquad P = \begin{bmatrix} 8.35 & -10.55 \\ -10.55 & 20.64 \end{bmatrix}$$

- Then the closed-loop system of the LQR problem becomes:

$$x_{k+1} = \begin{bmatrix} 1.29 & -1.69 \\ 1 & -1.5 \end{bmatrix} x_k$$

Example: Linear MPC

- Now we need to find the Invariant Set associated with this system and that satisfies the following constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -10 \\ -10 \end{bmatrix} \leq \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\}, \forall k \geq 0$$

$$Kx_k \in \mathcal{U} = \left\{ x \in \mathbb{R}^2 : -5 \leq Kx \leq 5 \right\}, \forall k \geq 0$$

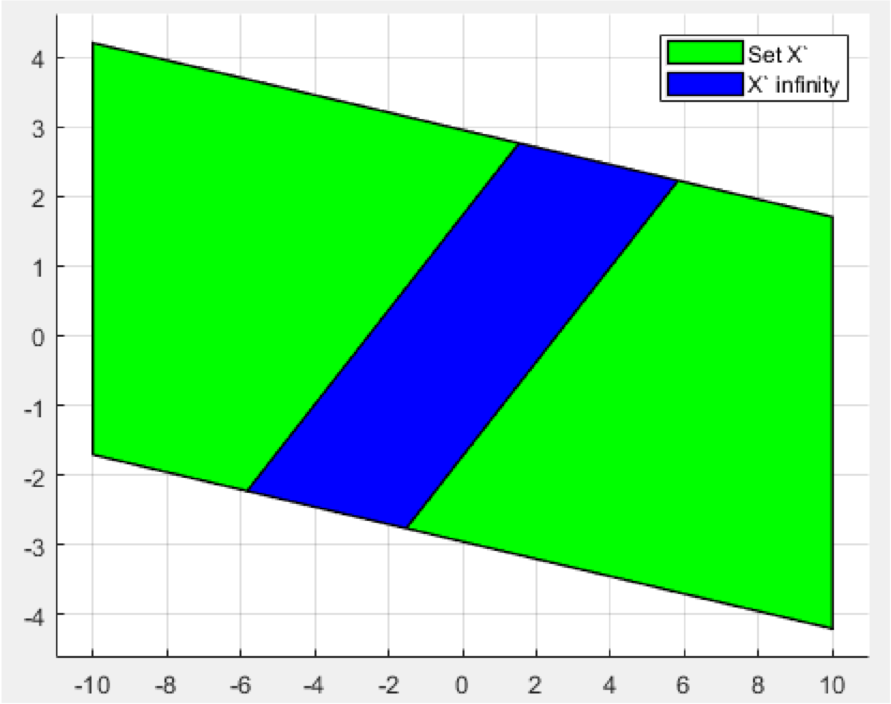
- Thus the “joint” feasible region is:

$$x_k \in \mathcal{X}' = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ -0.21 & -1.69 \\ 0.21 & 1.69 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 5 \\ 5 \end{bmatrix} \right\}, \forall k \geq 0$$

Example: Linear MPC

- Applying our Algorithm, it yields the following:

$$x_k \in \mathcal{X}_f = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -0.56 & 0.83 \\ 0.56 & -0.83 \\ -0.13 & -0.99 \\ 0.13 & 0.99 \end{bmatrix} x \leq \begin{bmatrix} 1.43 \\ 1.43 \\ 2.93 \\ 2.93 \end{bmatrix} \right\}$$



Robust Set Computation

- Now let's re-introduce disturbances to our linear system:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

- Where the disturbance vector w_k will belong to some polytope $\mathcal{W}$:

$$w_k \in \mathcal{W} = \left\{ w \in \mathbb{R}^n : H_w w \leq h_w \right\}, \forall k \geq 0$$

- First let's suppose there are no controls, so we have an autonomous system:

$$x_{k+1} = Ax_k + w_k$$

- How can we extend the notion of predecessor set to the case with disturbances?

Robust Set Computation

- We define now the **robust** predecessor set as follows:

$$P(\mathcal{X}, \mathcal{W}) = \left\{ x \in \mathbb{R}^n : \ Ax + w \in \mathcal{X}, \, \forall w \in \mathcal{W} \right\}$$

- Similarly for the system with control inputs:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

- We define:

$$P(\mathcal{X}, \mathcal{W}) = \left\{ x \in \mathbb{R}^n : \ \exists u \in \mathcal{U} \text{ s.t.: } Ax + Bu + w \in \mathcal{X}, \, \forall w \in \mathcal{W} \right\}$$

Robust Set Computation

- First let's focus on the autonomous system:

$$x_{k+1} = Ax_k + w_k$$

- With the constraint sets:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : Hx \leq h \right\}, \forall k \geq 0$$

$$w_k \in \mathcal{W} = \left\{ w \in \mathbb{R}^n : H_w w \leq h_w \right\}, \forall k \geq 0$$

- And Predecessor set:

$$P(\mathcal{X}, \mathcal{W}) = \left\{ x \in \mathbb{R}^n : \ Ax + w \in \mathcal{X}, \forall w \in \mathcal{W} \right\}$$

Robust Set Computation

- We can write it as:

$$P(\mathcal{X},\mathcal{W}) = \{x \in \mathbb{R}^n : HAx \leq h - Ww,\ \forall w \in \mathcal{W}\}$$

- Which we can be written as:

$$P(\mathcal{X},\mathcal{W}) = \{x \in \mathbb{R}^n : HAx \leq \bar{h}\} \qquad \bar{h}_i = \min_{w \in \mathcal{W}} \{h_i - H_i w\}$$

- And using our Minkowski operation is the same as:

$$P(\mathcal{X},\mathcal{W}) = \{x \in \mathbb{R}^n : Ax + w \in \mathcal{X},\ \forall w \in \mathcal{W}\} = \{x \in \mathbb{R}^n : Ax \in \mathcal{X} \ominus \mathcal{W}\}$$

$$P(\mathcal{X},\mathcal{W}) = (\mathcal{X} \ominus \mathcal{W}) \circ A$$

Robust Set Computation

- Now back to a control system with disturbances:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

- With constraints:

$$x_k \in \mathcal{X} = \left\{ x \in \mathbb{R}^2 : Hx \leq h \right\}, \forall k \geq 0$$

$$u_k \in \mathcal{U} = \left\{ u \in \mathbb{R}^m : H_u u \leq h_u \right\}, \forall k \geq 0$$

$$w_k \in \mathcal{W} = \left\{ w \in \mathbb{R}^n : H_w w \leq h_w \right\}, \forall k \geq 0$$

Robust Set Computation

- We write the Predecessor set:

$$P(\mathcal{X},\mathcal{W}) = \left\{x \in \mathbb{R}^n : \ \exists u \in \mathcal{U} \text{ s.t.: } Ax + Bu + w \in \mathcal{X}, \forall w \in \mathcal{W}\right\}$$

- As

$$P(\mathcal{X},\mathcal{W}) = \left\{x \in \mathbb{R}^n : \ \exists u \in \mathbb{R} \text{ s.t.: } \begin{bmatrix} HA & HB \\ 0 & H_u \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} h - Hw \\ h_u \end{bmatrix}, \forall w \in \mathcal{W}\right\}$$

- Which we write as:

$$P(\mathcal{X},\mathcal{W}) = \left\{x \in \mathbb{R}^n : \ \exists u \in \mathbb{R} \text{ s.t.: } \begin{bmatrix} HA & HB \\ 0 & H_u \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \leq \begin{bmatrix} \bar{h} \\ h_u \end{bmatrix}, \forall w \in \mathcal{W}\right\}$$

$$\bar{h}_i = \min_{w \in \mathcal{W}} \{h_i - H_i w\}$$

Robust Set Computation

- Using the Minkowski Operation it becomes:

$$P(\mathcal{X}, \mathcal{W}) = ((\mathcal{X} \ominus \mathcal{W}) \oplus (-B \circ \mathcal{U})) \circ A$$

- We can summarize all Predecessor set operations for system with and without disturbances as follows:

	$x_{k+1} = Ax_k + w_k$	$x_{k+1} = Ax_k + Bu_k + w_k$
$P(\mathcal{X})$	$\mathcal{X} \circ A$	$(\mathcal{X} \oplus (-B \circ \mathcal{U})) \circ A$
$P(\mathcal{X}, \mathcal{W})$	$(\mathcal{X} \ominus \mathcal{W}) \circ A$	$((\mathcal{X} \ominus \mathcal{W}) \oplus (-B \circ \mathcal{U})) \circ A$

Robust Invariant Set Computation

- We extend the definition of **positively invariant** for any closed-loop systems $x_{k+1} = Ax_k + w_k$ with disturbances:

$$x_0 \in \mathcal{O} \Rightarrow x_k \in \mathcal{O},\ k \geq 1,\ \forall w_k \in \mathcal{W}$$

- In words: “a set is robust invariant if once our systems starts from it, it stays in it forever for any possible disturbance value.”

- And, as always the largest of such sets is called the maximal robust invariant set, and we called it $\mathcal{O}_\infty$.

Invariant Set Computation

- All our arguments carry forward in this case as well: a set $\mathcal{O} \subseteq \mathcal{X}$ if and only if:

$$\mathcal{O} \subseteq \mathcal{P}(\mathcal{O}, \mathcal{W})$$

- And the Algorithm remains the same:

Algorithm 1 Algorithm for Invariant Set Computation

Input: Linear system matrix A, state constraint set $\mathcal{X}$, and disturbance set $\mathcal{W}$.

1: Let $\Omega_0 = \mathcal{X}$

2: **for** $k = 0, 1, 2, 3...$ **do** (obtaining new samples)

3: Let: $\Omega_{k+1} = \mathcal{P}(\Omega_k, \mathcal{W}) \cap \Omega_k$

4: If $\Omega_{k+1} = \Omega_k$, Set $\mathcal{O}_\infty \leftarrow \Omega_{k+1}$; Then break

5: **end for**

Output: The Maximal Robust Invariant Set $\mathcal{O}_\infty$

Control Invariant Set Computation

- The same is true Control Invariant sets. A set is robust control invariant for a system $x_{k+1} = Ax_k + Bu_k + w_k$ if:

$$x_0 \in \mathcal{C} \Rightarrow \exists u_k \in \mathcal{U} \text{ s.t.: } x_k \in \mathcal{C},\ \forall w_k \in \mathcal{W},\ k \geq 1$$

- In words: “a set is control invariant if once our systems starts from it, there is always a feasible control such that once applied the system inside the set for any possible disturbance value.”

- And the largest of such sets is the maximal control invariant set, and we called it $\mathcal{C}_\infty$.

Invariant Set Computation

- As before, a set $\mathcal{C} \subseteq \mathcal{X}$ is control invariant if and only if:

$$\mathcal{C} \subseteq \mathcal{P}(\mathcal{C}, \mathcal{W})$$

- Then we can state a very similar Algorithm to compute control invariant sets:

Algorithm 1 Algorithm for Invariant Set Computation

Input: Linear system matrix A, state constraint set $\mathcal{X}$, and disturbance set $\mathcal{W}$.

1: Let $\Omega_0 = \mathcal{X}$
2: **for** $k = 0, 1, 2, 3...$ **do** (obtaining new samples)
3: Let: $\Omega_{k+1} = \mathcal{P}(\Omega_k, \mathcal{W}) \cap \Omega_k$
4: If $\Omega_{k+1} = \Omega_k$, Set $\mathcal{C}_\infty \leftarrow \Omega_{k+1}$; Then break
5: **end for**

Output: The Maximal Robust Control Invariant Set $\mathcal{C}_\infty$