SSY281 - Multi Predictive Control

MPT and Persistent Feasibility

Assignment - 6

ID-Number 43

Fikri Farhan Witjaksono fikrif@student.chalmers.se

March 4, 2020

1 MPT

1.1 Question 1 : Find the V-representation and H-representation of the given polyhedron

Let we define the following matrices A and b

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \\ -1 & -1 \\ 1 & 1 \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \tag{1}$$

In **V-representation** [1], The polyhedron $\mathcal{P}$ is formed by the intersection of m inequalities and m_e equalities as shown below

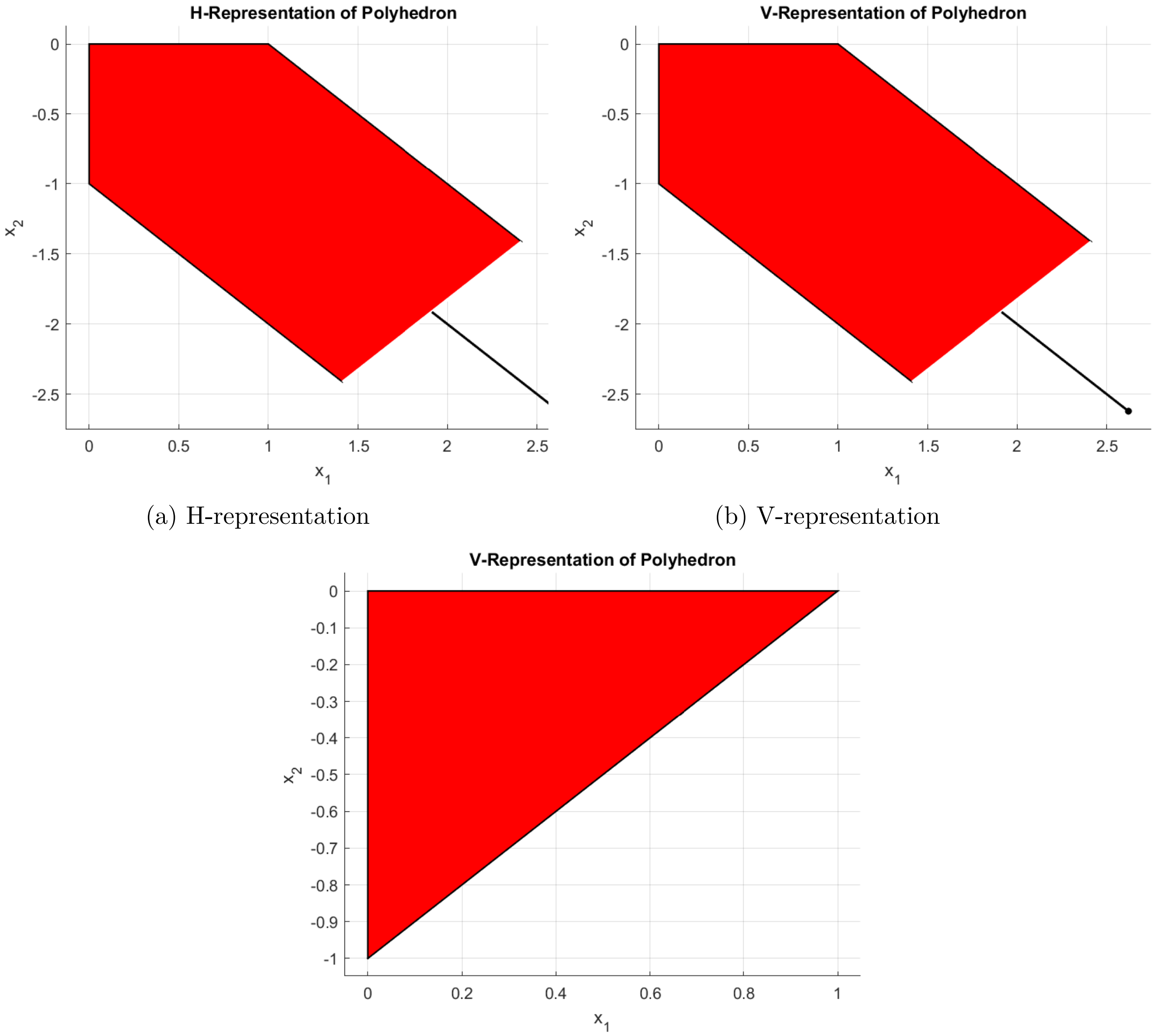
$$\mathcal{P} = \{x \in \mathrm{I\!R^n} \mid \mathrm{Ax \leq b, A_e x = b_e}\} \tag{2}$$

where $A \in \mathrm{I\!R^{mxn}}$, $b \in \mathrm{I\!R^m}$, $A_e \in \mathrm{I\!R^{m_e xn}}$, $b_e \in \mathrm{I\!R^{m_e}}$ are the data representing the halfspace and hyperplanes respectively.

On the other hand, in **H-representation** [1], The polyhedron $\mathcal{P}$ is formed by convex combination of n_v vertices and n_r rays as shown below

$$\mathcal{P} = \{x \in \mathrm{I\!R^n} \mid x = \lambda^T V \; + \; \gamma^T R, \; \lambda, \; \gamma \geq 0, \; 1^T \lambda = 1\} \tag{3}$$

where $V \in \mathrm{I\!R^{nxn_v}}$, $R \in \mathrm{I\!R^{nxn_r}}$ represents vertices and rays respectively. V-representation is represented with vertices and rays that goes to infinity.



(a) H-representation

(b) V-representation

(c) V-representation without the tail

Figure 1: H- and V- representation of the Polyhedron

The V-representation would be exactly the same except if we have set the representation using only the vertices, without taking into account the rays that goes to infinity $[-\infty, \infty]$, we will get a whole different shape than figure 1(b) (e.g. triangle) as shown in the previous page.

1.2 Question 2 : Compute the Minkowski sum and Pontraygin difference

The Minkowski sum and Pontraygin difference set could be expressed mathematically as shown below

$$Minkowski = (P \oplus Q) = \{\mathbf{p} + \mathbf{q} \,|\, \mathbf{p} \in Q,\ \mathbf{q} \in B\} \tag{4}$$
$$Pontraygin = (P \ominus Q) = \{\mathbf{p} - \mathbf{q} \,|\, \mathbf{p} \in Q,\ \mathbf{q} \in B\}$$

Minkowski sum and Pontraygin difference could be described as the sum and differentiation of two different position vector in Euclidean space. The result of this calculation is as shown below

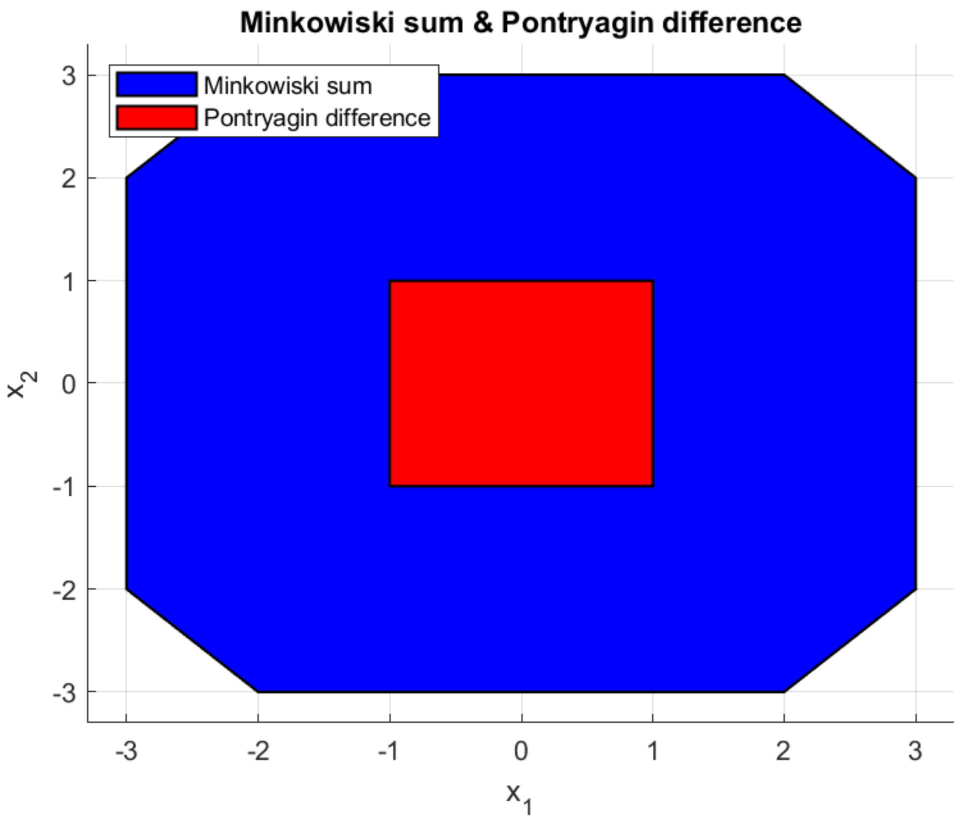


Figure 2: Minkowski sum and Pontraygin Difference plot

Here we could observe that the Pontraygin difference set is a subset of the Minkowski sum set $(P \oplus Q) \subseteq (P \ominus Q)$ since it represent is a smaller set than the Minkowski sum set.

2 Persistent Feasibility

2.1 Question 3: Show that set S is positively invariant

Assume that we have the following autonomous system

$$x^+ = Ax, \quad A = \begin{bmatrix} 0.8 & 0.4 \\ -0.4 & 0.8 \end{bmatrix} \tag{5}$$

as well as the following system

$$A_{in} = \begin{bmatrix} 1.0 & 0 \\ 0 & 1.0 \\ -1.0 & 0 \\ 0 & -1.0 \\ 1.0 & 1.0 \\ 1.0 & -1.0 \\ -1.0 & 1.0 \\ -1.0 & -1.0 \end{bmatrix}, \quad b_{in} = \begin{bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \\ 1.5 \\ 1.5 \\ 1.5 \\ 1.5 \end{bmatrix} \tag{6}$$

By definition, in order to find positive invariant sets, Suppose that we have these conditions to satisfy

1. $x(k+1) = Ax(k) \longrightarrow$ Assuming A-stable
2. $x \in X \longrightarrow$ such that it satisfy state constraint
3. $C \doteq$ such that $x(0) \in C \longrightarrow x(k) \in C, \ \forall k \in 1, 2, ..\infty$

The 3 conditions above will only be satisfied if

$$C \subseteq \mathcal{X} \tag{7}$$

In order to do that, we could use the following algorithm which is called as infinite step backward reachable sets algorithm as follows.
Suppose that we have

$$\text{Input} : A, \mathcal{X} \tag{8}$$

$$\text{Output} : C$$

$$\Omega_0 = \mathcal{X} \tag{9}$$

Repeat,

$$\Omega_{k+1} = Pre(\Omega_k) \cap \mathcal{X} \tag{10}$$

until

$$\Omega_{k+1} = \Omega_k \tag{11}$$

The algorithm above shows that in order to prove that we have a positive invariant set, we need to firstly assume that we have initial Ω equal to the State Constraint (X). Then, by repeating equation (10) above, until the Ω converges at some point before infinity, then we could conclude that we will have a positive invariant set. When we have satisfied eq (11), we will have as following

$$\Omega_{k+3} = \Omega_{k+2} = \Omega_{k+1} = \Omega_k \tag{12}$$

Hence, by using algorithm above, we would be able to guaranteed to have a positive invariant sets. In MATLAB, we could find that by creating while which keep iterating until condition (11) is satisfied.

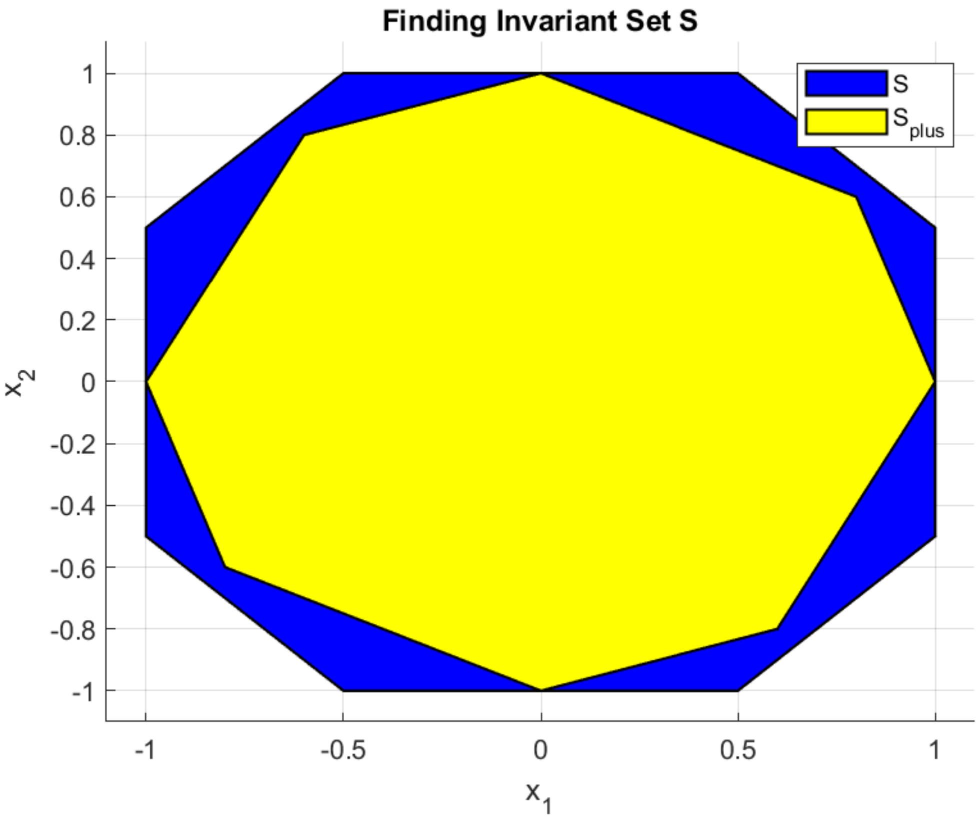


Figure 3: Polytope representation of S and S^+

Alternatively, assuming that A is invertible, we could prove that S is positively invariant for this system by proving that if we have the new set S^+ as x from the system inputted to the inequality constraint in the set S as follows

$$x^+ = Ax \Rightarrow A^{-1}x^+ = x \tag{13}$$

$$A_{in}x \leq b_{in} \Rightarrow A_{in}A^{-1}x^+ \leq b_{in}$$

The result of plotting the polytope of both sets is as shown above. Observing the figure above, we could see that the set S^+ is a subset of S ($S \subseteq S^+$), since it is wholly inside the set of S. Consequently, the next step of the initial state $(x_0 \in S) \longrightarrow (x(t, x_0) \in S\ , \forall t \geq 0)$, would be inside the set of S as well. Intuitively, this means that once an initial trajectory of the system starts from inside the set S, it will never leave it again. *Hence, we could conclude that the set S is positively invariant for this system.*

2.2 Question 4 : Compute the reachable set and plot it

Assume that we have the following non-autonomous system

$$x^+ = Ax + Bu, \quad A = \begin{bmatrix} 0.8 & 0.4 \\ -0.4 & 0.8 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{14}$$

with inequality input constraint $-1 \leq u \leq 1$. By definition, The reachable set could be found by finding the set that is reachable in one step from S. The set could be expressed mathematically as below

$$\begin{aligned} \text{Reach(x)} &= \{x \mid \exists\, x(0) \in X,\ \exists\, u(0) \in U s.t.\ x = Ax(0) + Bu(0)\} \\ &= \{x(1) \mid \exists\, x(0) \in X,\ \exists\, u(0) \in U s.t.\ x = Ax(0) + Bu(0)\} \end{aligned} \tag{15}$$

Let X is the convex hull set of V

$$X = conv(V) \tag{16}$$

where conv(V) indicates the convex hull of the set V. Then the reachable set could be expressed as below

$$\text{Reach(x)} = (\underbrace{A \cdot v}_{\text{initial forward reachable set}} \oplus \underbrace{B \cdot u}_{\text{Additional direction from the input}}) \tag{17}$$

Note that the dot multiplication indicates set multiplication and v and u indicates the V-representation of the system (e.g using the vertices). From the equation above, theoretically we can say that the effect of u is that it expands the initial boundary of forward reachable set through the operation of Minkowski sum. By the method presented above, we could find the forward reachable set as shown in the figure below

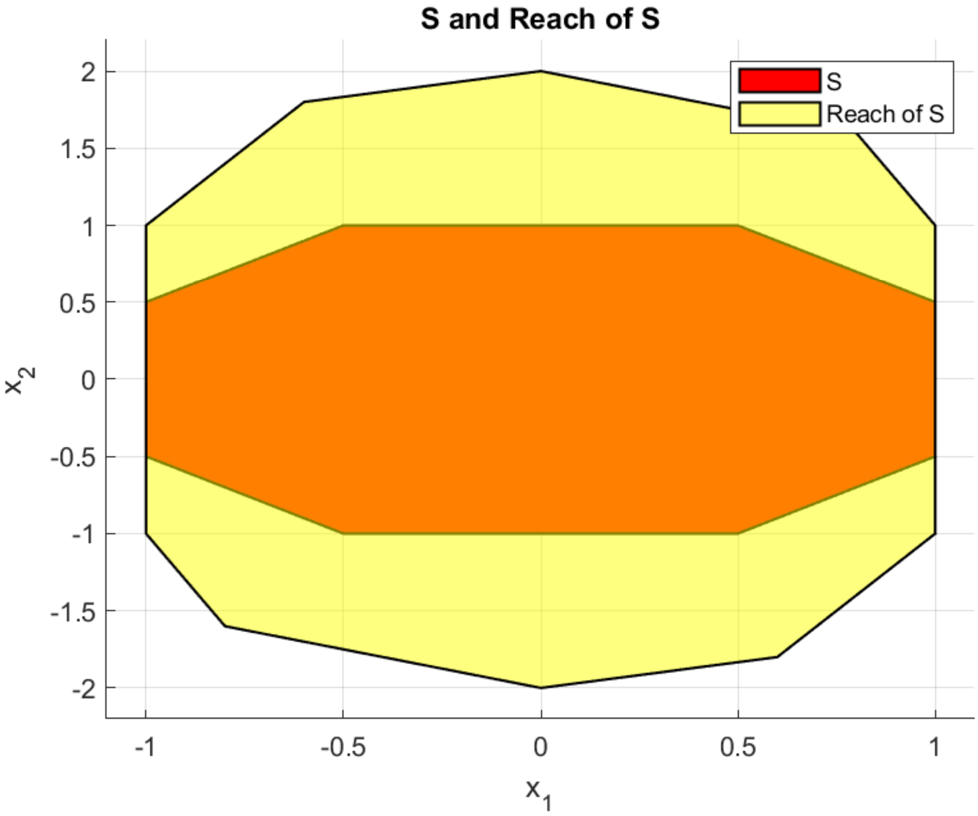


Figure 4: The polytope representation of S and Reach of S

2.3 Question 5: Calculate the one step Pre of S

Using the same system as in Question 4 above, we could first show the definition of Pre of x as follows

$$\text{Pre(x)} = \{x \mid \exists\, x(k+1) \in X,\ \exists\, u \in U\ s.t.\ x = Ax(0) + Bu(0)\} \tag{18}$$

$$= \{x(0) \mid \exists\, x(1) \in X,\ \exists\, u(0) \in U\ s.t.\ x = Ax(0) + Bu(0)\}$$

Suppose we have x

$$x = \{x \mid Hx \leq 1\} \tag{19}$$

Next, we could define pre(x) using the inequality state constraint and input constraint that is represented by unified constraint H (e.g. using structure in MATLAB) as follows

$$H_U \cdot u \leq h_U \tag{20}$$

$$H_S \cdot s \leq h_s$$

where S is the polyhedron that represent the inequality constraint on the state and U is the polyhedron that represent the inequality constraint on the input. Including the constraint in equation (19) in the equation (17) will give us

$$Pre(x) \rightarrow Hx(1) \leq 1 \tag{21}$$

$$H[Ax(0) + Bu(0)] \leq \begin{bmatrix} h_S \\ h_U \end{bmatrix}$$

Expressing in matrix form gives

$$\begin{bmatrix} H_S A & H_S B \\ 0 & H_U \end{bmatrix} \begin{bmatrix} x(0) \\ u(0) \end{bmatrix} \leq \begin{bmatrix} h_S \\ h_U \end{bmatrix} \tag{22}$$

Finally, Pre(x) is

$$\text{Pre(x)} : Proj_x(\begin{bmatrix} H_S A & H_S B \\ 0 & H_U \end{bmatrix} \begin{bmatrix} x(0) \\ u(0) \end{bmatrix} \leq \begin{bmatrix} h_S \\ h_U \end{bmatrix} \tag{23}$$

The final expression in equation (22) contain the projection of the polytope on x space term which in MATLAB, we could use the projection command. Hence, using the method, we could plot Pre and S set as shown below

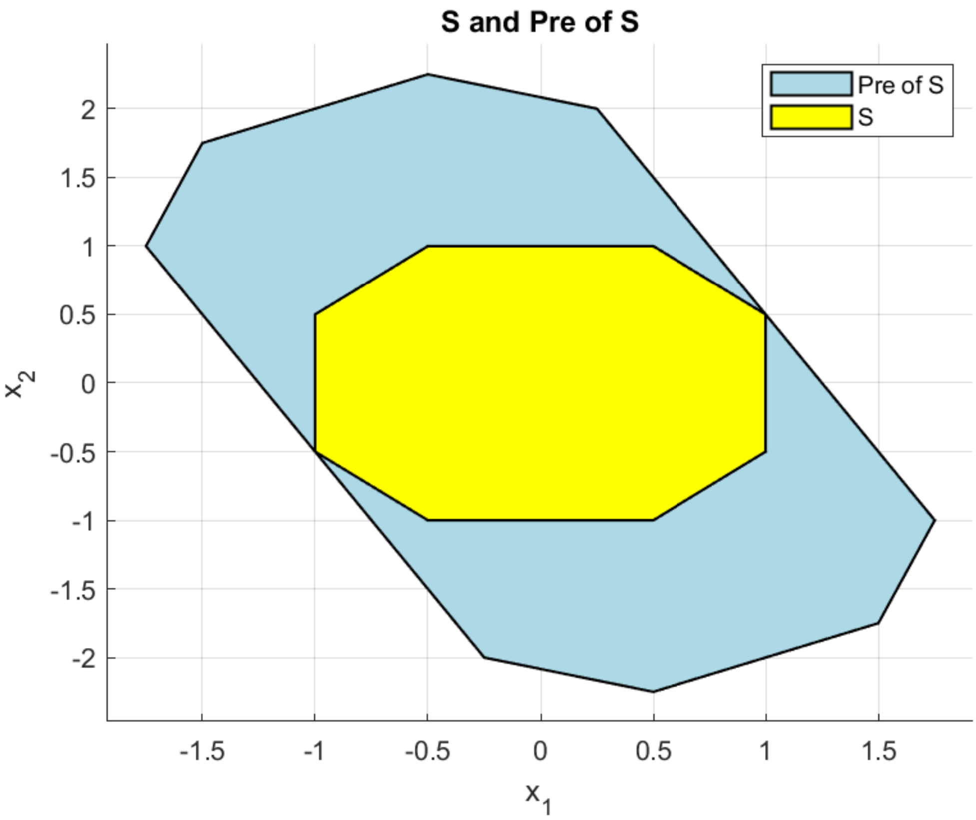


Figure 5: The Pre of S and S polytope representation

We could observe that the $Pre(S) \subseteq S$ as expected theoretically in figure 4 above.

2.4 Question 6: Design a RHC using the parameters below

2.4.1 PROBLEM FORMULATION

Suppose we have a non-autonomous system

$$x^{+} = Ax + Bu, \quad A = \begin{bmatrix} 0.9 & 0.4 \\ -0.4 & 0.9 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{24}$$

Design a RHC with $P_f = 0$, $Q = I_{2x2}$, $R = 1$ and $\mathcal{X}_f = 0_{2x1}$. Let $x(0) = [2 \;\; 0]^T$ be the initial condition and consider the following constraints.

$$|x_i(k)| \leq 3, \; |u(k)| \leq 0.1 \;\; \forall k \in \{0, 1, 2, ...\}, \; \forall i \in \{1, 2\} \tag{25}$$

2.4.2 1. Find the shortest horizon N such that RHC is feasible

The RHC could be solved by the same method that was used in Assignment 2 except for the fact that in this assignment, there is an additional condition that the last set of state $x(N) = 0$. In this problem, we could form the inequality constraints that we have in a matrix form ($A_{ineq}x = b_{ineq}$) as follows

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ u \\ x_1 \\ x_2 \\ u \end{bmatrix} \leq \begin{bmatrix} 3 \\ 3 \\ 0.1 \\ 3 \\ 3 \\ 0.1 \end{bmatrix} \tag{26}$$

In Matlab, we will have the designed RHC controller trying to direct the terminal set of state $x(N)$ to 0. At the same time, the QP solver will check if we can find a feasible solution with the constraints that we have. We can know the result of this through the **exitflag** output which if it shows output **1**, then we will have a feasible solution. On the other hand, if we have output **exitflag=-2**, it means that we will not have a feasible solution. Consequently, it means that we have to extend the prediction horizon input until at the last iteration, we have **exitflag=-8**,which means that we have found the shortest horizon N which gives feasible RHC so that MATLAB is unable to compute a next step direction anymore.
From the method above, by trial and error, it is found that the shortest prediction horizon N=26 for feasible RHC

2.4.3 2. Determine if the RHC is still feasible until convergence to the origin

Suppose we have a non-autonomous system

$$x(k+1) = Ax(k) + Bu(k) \tag{27}$$

where $x \in X$ and $u \in U$. Then, suppose that we chose $\mathcal{X}_f$ to be control invariant set of the system by using **invariant set** command, that is, assuming that X_f is maximal control invariant set for the system. By definition, in control invariant sets, it is possible satisfy controllability condition as well as satisfying the 2 different constraints that we have (e.g. state constraint and input constraint). Thus, theoretically, if we start inside the control invariant set, we will be able to guarantee the persistent feasibility of the system without having the need to use large prediction horizon.

In this case, by setting $N = 2$ as well as X(N) which is defined in previous question to be inside the set $\mathcal{X}_f$, we will be able to prove that RHC is still feasible at the origin if the quadprog function solver used in MATLAB can give a solution (e.g. exitflag = 1). *Hence, we could conclude that using $N = 2$ will still be feasible until convergence to the origin.*

2.4.4 3. Plot the feasible initial states' sets for the controller designed in Question 6.1 and 6.2

To plot the initial states, we could take the feasible state at N=1 from question 6.1 and 6.2 and then use the command **ReachableSet** specifically using backward function in question 4 to calculate the x(0) from x(1). The result of this is a polyhedron.

In the **question 6.1**, the **number of optimization variables** needed is **78 variables** and the **number of inequality constraints are in total 150**. On the other hand, in the **question 6.2**, **the number of optimization variable** needed is only **6 variables** and **the number of constraints** are only **14 inequality constraints**. This represents a major improvement in terms of computational cost, hence we could conclude that *the second method gives a much more efficient calculation than the second one to find a feasible RHC due to the fact that it is a much smaller optimization problem.*