

# Static State Feedback Problem

Find feedback (gain) matrix  $K$  such that

$$\dot{x}(t) = A x(t) + B u(t)$$

$$u(t) = K x(t)$$

is stable

i.e. Find  $K$  such that  $A+BK$  is Hurwitz

LMI representation:  
(BMI)

Find  $X > 0$ ,  $K$ :

$$X(A+BK) + (A+BK)^T X < 0$$

↳ Above problem is bilinear in  $K$  and  $X$

Equivalent problem:

Find  $P > 0$ ,  $Z$ :

~~$$AP + PA^T + BZ + Z^T B^T < 0$$~~

$$AP + PA^T + BZ + Z^T B^T < 0$$

with  $u(t) = \underbrace{ZP^{-1}}_K x(t)$

} system  $(A, B)$  is static state feedback stabilizable iff sol<sup>n</sup> to this problem exists. (i.e. some  $P > 0$  and  $Z$  exist that satisfy this LMI)

## Python Code:

```
import cvxpy as cp
import numpy as np

# Define system matrices for SYS1
A1 = np.array([[ -4, 1], [0, 2]])
B1 = np.array([[1], [0]])

# Define system matrices for SYS2
A2 = np.array([[ -3, 2], [4, 1]])
B2 = np.array([[0], [1]])

# Define dimensions
n = 2 # Dimension of state vector
m = 1 # Dimension of control input

# Define the decision variables
P1 = cp.Variable((n, n), symmetric=True)
Z1 = cp.Variable((m, n))

P2 = cp.Variable((n, n), symmetric=True)
Z2 = cp.Variable((m, n))

# Define the LMI constraints for SYS1
constraints1 = [A1@P1 + P1@A1.T + B1@Z1 + Z1.T@B1.T << 0, P1 >> 0.0001*np.eye(n)]

# Define the LMI constraints for SYS2
constraints2 = [A2@P2 + P2@A2.T + B2@Z2 + Z2.T@B2.T << 0, P2 >> 0.0001*np.eye(n)]

# Create an optimization problem for SYS1
problem1 = cp.Problem(cp.Minimize(0), constraints1)

# Create an optimization problem for SYS2
problem2 = cp.Problem(cp.Minimize(0), constraints2)

# Solve the LMI for SYS1
problem1.solve()

# Solve the LMI for SYS2
problem2.solve()

# Check the optimization results for SYS1
if problem1.status == cp.OPTIMAL:
```

```

    print('SYS1 is stabilizable')
    K1 = Z1.value @ np.linalg.inv(P1.value)
    print("K1 = ", K1)
    Acl1 = A1 + B1@K1
    print("Acl1 = A1 + B1@K1 =\n", Acl1)
    print("Eigenvalues of Acl1:", np.linalg.eig(Acl1)[0])
else:
    print('SYS1 is not stabilizable')
    K1 = None

# Check the optimization results for SYS2
if problem2.status == cp.OPTIMAL:
    print('\nSYS2 is stabilizable')
    K2 = Z2.value @ np.linalg.inv(P2.value)
    print("K2 = ", K2)
    Acl2 = A2 + B2@K2
    print("Acl2 = A2 + B2@K2 =\n", Acl2)
    print("Eigenvalues of Acl2:", np.linalg.eig(Acl2)[0])
else:
    print('SYS2 is not stabilizable')
    K2 = None

```

## Output:

SYS1 is not stabilizable

SYS2 is stabilizable

K2 = [[-5.862771 -1.10285985]]

Acl2 = A2 + B2@K2 =

[[ -3. 2. ]

[-1.862771 -0.10285985]]

Eigenvalues of Acl2: [-1.55142992+1.2756123j -1.55142992-1.2756123j]

Since real parts of eigenvalues of Acl2 are negative, it is successfully verified that system SYS2 is in fact stabilized using static state feedback control.

## Screenshot:

The screenshot displays a Jupyter Notebook environment with a Python script for analyzing the stability of two systems (SVS1 and SVS2) using Linear Matrix Inequalities (LMIs). The script is written in a Jupyter cell and includes the following key components:

- Imports:** `import cvxpy as cp` and `import numpy as np`.
- System Matrices:** Defines system matrices for SVS1 and SVS2. For SVS1,  $A_1 = \begin{bmatrix} -4 & 1 \\ 0 & 2 \end{bmatrix}$  and  $B_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ . For SVS2,  $A_2 = \begin{bmatrix} -3 & 2 \\ 0 & 1 \end{bmatrix}$  and  $B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ .
- Dimensions:**  $n = 2$  (Dimension of state vector) and  $m = 1$  (Dimension of control input).
- Decision Variables:**  $P_1$ ,  $P_2$ ,  $Z_1$ , and  $Z_2$  are defined as symmetric positive semi-definite matrices.
- LMIs:** Constraints are defined for each system:  $A_i P_i + P_i B_i + B_i^T P_i + Z_i \leq 0$  for  $i = 1, 2$ .
- Optimization:** Two optimization problems are created and solved using CVXPY's `minimize` function.
- Results:** The script checks the status of the optimization problems. If optimal, it calculates the feedback gain  $K_i = -P_i^{-1} B_i^T P_i^{-1}$  and the closed-loop system matrix  $A_{cl_i} = A_i + B_i K_i$ . It then prints the eigenvalues of the closed-loop matrices.

The output of the script, displayed in the Jupyter cell's output area, shows that SVS1 is not stabilizable, while SVS2 is stabilizable. The calculated feedback gain for SVS2 is  $K_2 = [-5.862771 \quad -1.18285985]$ , and the eigenvalues of the closed-loop matrix  $A_{cl2}$  are  $[-1.862771 \quad -0.18285985]$ .