Convex Optimization - Homework 2

Kian Khaneghahi

A Homework Solution
Presented to the Faculty
of Science and Research Branch
Azad University for the Degree
of Master of Science

Department of
Control Engineering
Supervisor: Professor Ali Moarefian

15th of February 2021

Abstract

This report contains the solution to Homework 2 of the corresponding convex optimization course with professor Ali Moarefian. The questions are also provided for convenience.

Contents

List of Tables

List of Figures

Chapter 1

Question 1

Consider the system below :

$$\dot{x} = \begin{bmatrix} -1 & 1 \\ 0 & -2 \end{bmatrix} x \tag{1.1}$$

Find the lyapunov function $V(x) = x^T P x$ with $P > 0$ so that the given system is asymptotically stable. Note that it needs to be in the LMI form and also plot the transient response of the system at the end with initial conditions $x_0 = (1 \quad 1)^T$.

1.1 LMI Representation

In order for the system to be asymptotically stable, it is required that $\dot{V}(x) < 0$. Therefore, first $\dot{V}$ is derived as the following.

$$\dot{V} = \dot{x}^T P x + x^T P \dot{x} < 0 \tag{1.2}$$

and by substituting $\dot{x}$ from relation 1.1 and showing it in parametric form we achieve the following.

$$\dot{V} = (Ax)^T P x + x^T P A x < 0 \tag{1.3}$$

$$\dot{V} = x^T A^T P x + x^T P A x < 0 \tag{1.4}$$

by factorizing x^T from the left and x^T from the right we reach the following.

$$\dot{V} = x^T [A^T P + PA] x < 0 \tag{1.5}$$

and due to the quadratic nature of this inequality the inner expression should be negative definite that is the following relation.

$$A^T P + PA < 0 \qquad , \qquad P > 0 \tag{1.6}$$

Relation 1.6 is also known as the lyapunov relation for linear systems. Because A is a constant and only P is the optimization variable, therefore, relation 1.6 is the desired LMI form for the SDP[1] optimization Problem.

1.2 Solving the SDP Problem In Matlab

Due to the simplicity in usage, the CVX convex optimization library for MATLAB is used. To solve for P in relation 1.6, the following Matlab code is used. As seen in the code, the strict inequalities are replaced with non-strict inequalities only not compared to zero but a very small matrix and this is on account of CVX's regulations. Also, we enclose the SDP problem in `cvx_begin sdp` and `cvx_end`.

```
A = [-1 1;0 -2];
cvx_begin sdp
```

[1] Semi-definite Programming

```
variable P(2,2) symmetric
% Contraints(Lyapanov LMI)
A'*P + P'*A ≤ -0.000001*eye(2); % A'*P + P*A < 0
P ≥ 0.000001*eye(2); % P > 0
cvx_end
cvx_status
P
```

The first thing to do is to define the optimization variable P by `variable ... P(2,2)symmetric` and then the LMI constraints are defined. After `cvx_end`, `cvx_status` is placed to give us the information on whether the problem is solved or not solved. The answer provided by matlab for this problem is given in the following.

```
Calling SDPT3 4.0: 6 variables, 3 equality constraints
------------------------------------------------------------

num. of constraints =  3
dim. of sdp    var  =  4,   num. of sdp  blk  =  2
*******************************************************************
SDPT3: Infeasible path-following algorithms
*******************************************************************
version  predcorr  gam  expon  scale_data
HKM      1      0.000   1        0
it pstep dstep pinfeas dinfeas  gap      prim-obj      dual-obj ...
      cputime
-------------------------------------------------------------------
 0|0.000|0.000|3.1e+01|2.0e+01|4.0e+02| 0.000000e+00  ...
    0.000000e+00| 0:0:00| chol  1  1
 1|1.000|0.877|6.9e-06|2.5e+00|4.6e+01| 0.000000e+00  ...
    3.632281e-06| 0:0:00| chol  1  1
```

```
2|1.000|0.992|6.2e-06|2.9e-02|5.3e-01| 0.000000e+00 ...
    4.192551e-08| 0:0:00| chol  1  1
3|1.000|1.000|1.3e-06|1.0e-03|1.7e-02| 0.000000e+00 ...
    1.429407e-09| 0:0:00| chol  1  1
4|1.000|1.000|8.8e-08|1.0e-04|1.4e-03| 0.000000e+00 ...
    1.421835e-10| 0:0:00| chol  1  1
5|1.000|1.000|1.6e-09|1.0e-05|1.1e-04| 0.000000e+00 ...
    1.412408e-11| 0:0:00| chol  1  1
6|1.000|1.000|8.1e-11|1.0e-06|8.9e-06| 0.000000e+00 ...
    1.404284e-12| 0:0:00| chol  1  1
7|1.000|0.989|4.5e-12|1.1e-08|9.8e-08| 0.000000e+00 ...
    1.530650e-14| 0:0:00| chol  1  1
8|1.000|0.984|3.5e-14|1.7e-10|1.6e-09| 0.000000e+00 ...
    1.510172e-16| 0:0:00|
stop: max(relative gap, infeasibilities) < 1.49e-08
-------------------------------------------------------------------
number of iterations   =  8
primal objective value =  0.00000000e+00
dual   objective value =  1.51017179e-16
gap := trace(XZ)       = 1.57e-09
relative gap           = 1.57e-09
actual relative gap    = -1.51e-16
rel. primal infeas (scaled problem)   = 3.53e-14
rel. dual     "        "       "      = 1.73e-10
rel. primal infeas (unscaled problem) = 0.00e+00
rel. dual     "        "       "      = 0.00e+00
norm(X), norm(y), norm(Z) = 9.9e+00, 7.8e-11, 3.0e-10
norm(A), norm(b), norm(C) = 6.0e+00, 1.0e+00, 1.0e+00
Total CPU time (secs)  = 0.42
CPU time per iteration = 0.05
termination code       =  0
DIMACS: 3.5e-14  0.0e+00  1.7e-10  0.0e+00  -1.5e-16  1.6e-09
-------------------------------------------------------------------
```

```


------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +0


cvx_status =

    'Solved'


P =

    3.1903    0.9751
    0.9751    2.0878
```

As illustrated in the result, the optimized P for this system to make it asymptotically stable is shown bellow.

$$P = \begin{bmatrix} 3.1903 & 0.9751 \\ 0.9751 & 2.0878 \end{bmatrix} \tag{1.7}$$

1.3 The Transient Response Of The System

The transient Response of the given system with the optimization variable values from the previous section and initial conditions $x_0 = (1 \quad 1)^T$, is calculated by the `ode45` function in the $[0 \quad 9]$ time span. The code for this section is typed as the following.

```
% plotting the transient state for x = [1 1]'
tspan =[0 9];
```

```
X0 = [1;1];
[t,y] = ode45(@(t,X) A*X,tspan,X0);
figure(1);
plot(t,y)
```

Consequently, the plot of the transient response of the system is depicted in figure 1.1 and as illustrated, both states have stabilized towards zero and therefore the system is asymptotically stable.

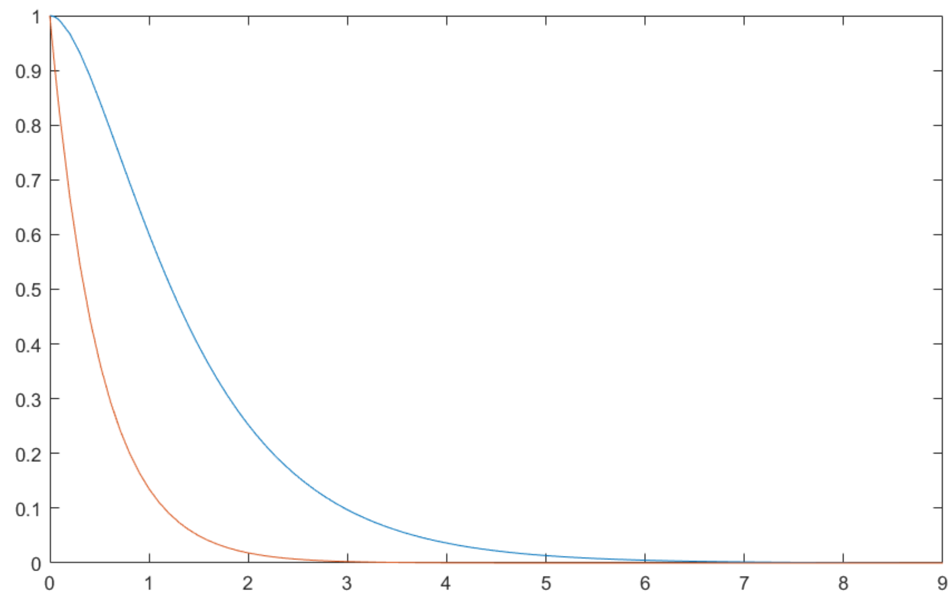

Figure 1.1: The transient response of the system with initial conditions x_0

Chapter 2

Question 2

Consider the system below :

$$\dot{x} = \begin{bmatrix} -1 & 1 \\ 0 & -2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{2.1}$$

Find the proper state feedback rule $u = -Kx$ so that the closed-loop system is asymptotically stable. Note that it needs to be in the LMI form and also plot the transient response of the system at the end with initial conditions $x_0 = (1 \quad 1)^T$.

2.1 LMI Representation

As proved in Chapter 1, we can show the asymptotically stability of the system by applying the following inequality for the closed loop system from relation 1.1.

$${A_{cl}}^T P + P A_{cl} < 0 \qquad , \qquad P > 0 \tag{2.2}$$

If we substitute $u = -Kx$ into relation 2.1, A_{cl} becomes $A_{cl} = A - BK$ and the following inequality is acquired.

$$(A - BK)^T P + P(A - BK) < 0 \qquad , \qquad P > 0 \tag{2.3}$$

If P^{-1} is multiplicated to the right and the left of the inequality, the following relations are obtained.

$$P^{-1}[(A - BK)^T P + P(A - BK)]P^{-1} < 0 \qquad , \qquad P > 0 \tag{2.4}$$

$$P^{-1}A^T - P^{-1}K^T B^T + AP^{-1} - BKP^{-1} < 0 \qquad , \qquad P > 0 \tag{2.5}$$

In order to form the LMI inequalities, we need to change the variables by setting $Q = P^{-1}$ and $Y = KP^{-1}$ and subsequently the previous relation becomes the following.

$$AQ + QA^T - BY - Y^T B^T < 0 \qquad , \qquad Q > 0 \tag{2.6}$$

Relation 2.6 is in the form of LMI because A and B are constants and Q and Y are the optimization variables. Therefore, we can form the SDP problem accordingly.

2.2 Solving the SDP Problem In Matlab

To solve for Q and Y in relation 2.6, the following Matlab code is used. As seen in the code, the strict inequalities are replaced with non-strict inequalities only not compared to zero but a very small matrix and this is on account of CVX's regulations. Also, we enclose the SDP problem in `cvx_begin sdp` and `cvx_end`.

```
A = [1 1;0 -2];
B = [0;1];
cvx_begin sdp
```

```
variable Q(2,2) symmetric
variable Y(1,2)
A*Q + Q*A' -B*Y-Y'*B' ≤ -0.000001*eye(2);
Q ≥ 0.000001*eye(2);
cvx_end
cvx_status
P = inv(Q)
K = Y * P
```

The first thing to do is to define the optimization variables Q and Y by `variable Q(2,2)symmetric` and `variable Y(1,2)` and then the LMI constraints are defined. After `cvx_end`, `cvx_status` is placed to give us the information on whether the problem is solved or not solved. The optimized value for P and K is achieved by their relations with Q and Y which leads to `P = inv(Q)` and `K = Y * P`. The answer provided by matlab for this problem is given in the following.

```
Calling SDPT3 4.0: 6 variables, 1 equality constraints
-----------------------------------------------------------

num. of constraints =  1
dim. of sdp    var  =  4,   num. of sdp  blk  =  2
******************************************************************
SDPT3: Infeasible path-following algorithms
******************************************************************
version  predcorr  gam  expon  scale_data
HKM      1      0.000   1        0
it pstep dstep pinfeas dinfeas  gap      prim-obj      dual-obj ...
       cputime
------------------------------------------------------------------
```

```
0|0.000|0.000|3.0e+01|2.0e+01|4.0e+02| 0.000000e+00  ...
    0.000000e+00| 0:0:00| chol  1  1
1|1.000|0.842|4.9e-15|3.2e+00|4.6e+01| 0.000000e+00  ...
    1.001469e-05| 0:0:00| chol  1  1
2|1.000|0.991|3.1e-15|3.9e-02|5.4e-01| 0.000000e+00  ...
    1.192444e-07| 0:0:00| chol  1  1
3|1.000|1.000|4.7e-16|1.0e-03|1.3e-02| 0.000000e+00  ...
    3.083601e-09| 0:0:00| chol  1  1
4|1.000|1.000|8.4e-15|1.0e-04|1.1e-03| 0.000000e+00  ...
    3.074719e-10| 0:0:00| chol  1  1
5|1.000|1.000|4.0e-15|1.0e-05|8.7e-05| 0.000000e+00  ...
    3.065916e-11| 0:0:00| chol  1  1
6|1.000|0.989|4.9e-15|1.1e-07|9.5e-07| 0.000000e+00  ...
    3.341823e-13| 0:0:00| chol  1  1
7|1.000|0.985|4.7e-16|1.6e-09|1.4e-08| 0.000000e+00  ...
    3.434237e-15| 0:0:00|
stop: max(relative gap, infeasibilities) < 1.49e-08
-------------------------------------------------------------------
number of iterations   =  7
primal objective value =  0.00000000e+00
dual   objective value =  3.43423697e-15
gap := trace(XZ)       = 1.40e-08
relative gap           = 1.40e-08
actual relative gap    = -3.43e-15
rel. primal infeas (scaled problem)   = 4.69e-16
rel. dual     "        "       "      = 1.58e-09
rel. primal infeas (unscaled problem) = 0.00e+00
rel. dual     "        "       "      = 0.00e+00
norm(X), norm(y), norm(Z) = 1.1e+01, 1.1e-09, 4.1e-09
norm(A), norm(b), norm(C) = 3.6e+00, 1.0e+00, 1.0e+00
Total CPU time (secs)  = 0.38
CPU time per iteration = 0.05
termination code       =  0
```

```
DIMACS: 4.7e-16  0.0e+00  1.6e-09  0.0e+00  -3.4e-15  1.4e-08
---------------------------------------------------------------------

-------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +0


cvx_status =

    'Solved'


P =

    1.6153    0.6210
    0.6210    0.3733


K =

    9.3297    2.0072
```

As illustrated in the result, the optimized P and K for this system to make it asymptotically stable is shown bellow.

$$P = \begin{bmatrix} 1.6153 & 0.6210 \\ 0.6210 & 0.3733 \end{bmatrix} \tag{2.7}$$

$$K = \begin{bmatrix} 9.3297 & 2.0072 \end{bmatrix} \tag{2.8}$$

2.3 The Transient Response Of The System

The transient Response of the given system with the optimization variable values from the previous section and initial conditions $x_0 = (1 \quad 1)^T$, is calculated by the `ode45` function in the $[0 \quad 9]$ time span. The code for this section is typed as the following.

```
tspan =[0 9];
X0 = [1;1];
[t,y] = ode45(@(t,X) (A-B*K)*X,tspan,X0);
figure(2);
plot(t,y)
```

Consequently, the plot of the transient response of the system is depicted in figure 2.1 and as illustrated, both states have stabilized towards zero and therefore the system is asymptotically stable.

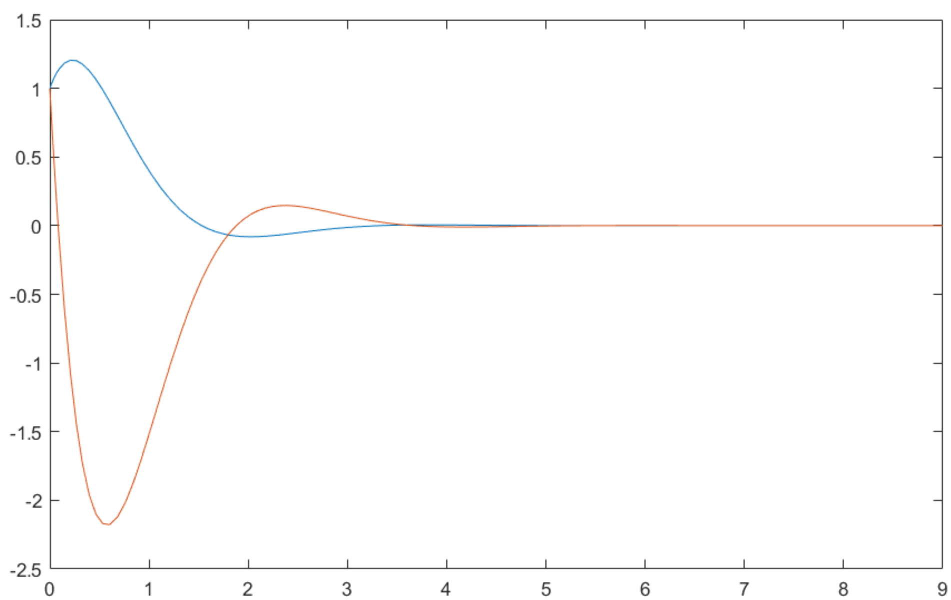

Figure 2.1: The transient response of the system with initial conditions x_0

Chapter 3

Question 3

Consider the system below :

$$\dot{x} = \begin{bmatrix} k & 1 \\ 0 & -2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{3.1}$$

Find the proper state feedback rule $u = -Kx$ so that the closed-loop system is asymptotically stable for $k \in \begin{bmatrix} -1 & 1 \end{bmatrix}$. Note that it needs to be in the LMI form and also plot the transient response of the system at the end with initial conditions $x_0 = (1 \quad 1)^T$ in which assume k varies according to the following relation.

$$k = \begin{cases} 1 & t \in [0 \quad 3] \\ 0 & t \in (3 \quad 6] \\ -1 & t \in (6 \quad 9] \end{cases} \tag{3.2}$$

3.1 LMI Representation

As proved in Chapter 2, we can show the asymptotically stability of the state feedback closed loop system by applying the following inequality for the closed loop system from relation 2.6. As noted before, $Q = P^{-1}$ and $Y = KP^{-1}$.

$$AQ + QA^T - BY - Y^T B^T < 0 \quad , \quad Q > 0 \tag{3.3}$$

In order to solve the SDP for $k \in \begin{bmatrix} -1 & 1 \end{bmatrix}$ it is only required that the convex optimization problem is solved at its convex hull. Explicitly, it can be shown that the convex hull can be formed by substituting $k = 1$ and $k = -1$ in relation 3.3 and acquiring the following inequalities.

$$A_1Q + QA_1{}^T - BY - Y^T B^T < 0 \quad , \quad Q > 0 \tag{3.4}$$

$$A_2Q + QA_2{}^T - BY - Y^T B^T < 0 \quad , \quad Q > 0 \tag{3.5}$$

$$A_1 = \begin{bmatrix} 1 & 1 \\ 0 & -2 \end{bmatrix} \tag{3.6}$$

$$A_2 = \begin{bmatrix} -1 & 1 \\ 0 & -2 \end{bmatrix} \tag{3.7}$$

Relations 3.4 and 3.5 are in the form of LMI because $A1$, $A2$ and B are constants and Q and Y are the optimization variables. Therefore, we can form the SDP problem accordingly.

3.2 Solving the SDP Problem In Matlab

To solve for Q and Y in relations 3.4 and 3.5, the following Matlab code is used. As seen in the code, the strict inequalities are replaced with non-strict inequalities only not compared to zero but a very small matrix and this is on account of CVX's regulations. Also, we enclose the SDP problem in `cvx_begin sdp` and `cvx_end`.

```
A1 = [1 1;0 -2];
A2 = [-1 1;0 -2];
B = [0;1];
cvx_begin sdp
variable Q(2,2) symmetric
variable Y(1,2)
A1*Q + Q*A1' -B*Y-Y'*B' <= -0.000001*eye(2);
A2*Q + Q*A2' -B*Y-Y'*B' <= -0.000001*eye(2);
Q >= 0.000001*eye(2);
cvx_end
cvx_status
P = inv(Q)
K = Y * P
```

The first thing to do is to define the optimization variables Q and Y by `variable Q(2,2)symmetric` and `variable Y(1,2)` and then the LMI constraints are defined. After `cvx_end`, `cvx_status` is placed to give us the information on whether the problem is solved or not solved. The optimized value for P and K is achieved by their relations with Q and Y which leads to `P = inv(Q)` and `K = Y * P`. The answer provided by matlab for this problem is given in the following.

```
Calling SDPT3 4.0: 9 variables, 4 equality constraints
------------------------------------------------------------

num. of constraints =  4
dim. of sdp    var  =  6,   num. of sdp  blk  =  3
*******************************************************************
SDPT3: Infeasible path-following algorithms
*******************************************************************
```

```
version  predcorr  gam  expon  scale_data
HKM      1      0.000   1        0
it pstep dstep pinfeas dinfeas  gap      prim-obj      dual-obj ...
       cputime
-------------------------------------------------------------------
0|0.000|0.000|3.2e+01|2.4e+01|6.0e+02| 0.000000e+00  ...
    0.000000e+00| 0:0:00| chol  1  1
1|1.000|0.877|1.3e-05|3.1e+00|5.9e+01| 0.000000e+00  ...
    1.256735e-05| 0:0:00| chol  1  1
2|1.000|0.992|8.6e-06|3.6e-02|6.8e-01| 0.000000e+00  ...
    1.459847e-07| 0:0:00| chol  1  1
3|1.000|1.000|2.1e-06|1.0e-03|1.8e-02| 0.000000e+00  ...
    4.037023e-09| 0:0:00| chol  1  1
4|1.000|1.000|2.4e-07|1.0e-04|1.4e-03| 0.000000e+00  ...
    4.022374e-10| 0:0:00| chol  1  1
5|1.000|1.000|5.0e-09|1.0e-05|1.2e-04| 0.000000e+00  ...
    3.998145e-11| 0:0:00| chol  1  1
6|1.000|1.000|2.5e-10|1.0e-06|9.4e-06| 0.000000e+00  ...
    3.972065e-12| 0:0:00| chol  1  1
7|1.000|0.989|1.4e-11|1.1e-08|1.0e-07| 0.000000e+00  ...
    4.329562e-14| 0:0:00| chol  1  1
8|1.000|0.986|7.9e-14|1.5e-10|1.5e-09| 0.000000e+00  ...
    4.881214e-16| 0:0:00|
stop: max(relative gap, infeasibilities) < 1.49e-08
-------------------------------------------------------------------
number of iterations   =  8
primal objective value =  0.00000000e+00
dual   objective value =  4.88121421e-16
gap := trace(XZ)       = 1.48e-09
relative gap           = 1.48e-09
actual relative gap    = -4.88e-16
rel. primal infeas (scaled problem)   = 7.95e-14
rel. dual     "        "       "      = 1.55e-10
```

```
rel. primal infeas (unscaled problem) = 0.00e+00
rel. dual     "         "        "      = 0.00e+00
norm(X), norm(y), norm(Z) = 1.1e+01, 1.9e-10, 4.8e-10
norm(A), norm(b), norm(C) = 5.4e+00, 1.0e+00, 1.0e+00
Total CPU time (secs)  = 0.42
CPU time per iteration = 0.05
termination code       =  0
DIMACS: 7.9e-14  0.0e+00  1.5e-10  0.0e+00  -4.9e-16  1.5e-09
-------------------------------------------------------------------
------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +0
cvx_status =
    'Solved'
P =
    5.0733    1.4263
    1.4263    0.5821
K =
    30.2762    7.0098
```

As illustrated in the result, the optimized P and K for this system to make it asymptotically stable is shown bellow.

$$P = \begin{bmatrix} 5.0733 & 1.4263 \\ 1.4263 & 0.5821 \end{bmatrix} \tag{3.8}$$

$$K = \begin{bmatrix} 30.2762 & 7.0098 \end{bmatrix} \tag{3.9}$$

3.3 The Transient Response Of The System

The transient Response of the given system with the optimization variable values from the previous section and initial conditions $x_0 = (1 \quad 1)^T$, is calculated by the `ode45` function in the $[0 \quad 9]$ time span. The code for this section is typed as the following.

```
tspan =[0 9];
X0 = [1;1];
[t,y] = ode45(@(t,X) sf_sys(K,X,t),tspan,X0);
figure(3);
plot(t,y)
```

In the code the function `sf_sys(K,X,t)` is given by the following code which is different for different time intervals due to relations 3.1, 3.2, 3.6 and 3.7.

```
function X_dot = sf_sys(K,X,t)
    A1 = [1 1;0 -2];
    A2 = [0 1;0 -2];
    A3 = [-1 1;0 -2];
    B = [0;1];
    if t≥0 && t≤3
        A = A1;
    elseif t>3 && t≤6
```

```
        A = A2;
    elseif t>6 && t≤9
        A = A3;
    else
        A = A3;
    end
    u = - K*X;
    X_dot = A*X + B*u;
end
```

Consequently, the plot of the transient response of the system is depicted in figure 3.1 and as illustrated, both states have stabilized towards zero and therefore the system is asymptotically stable.

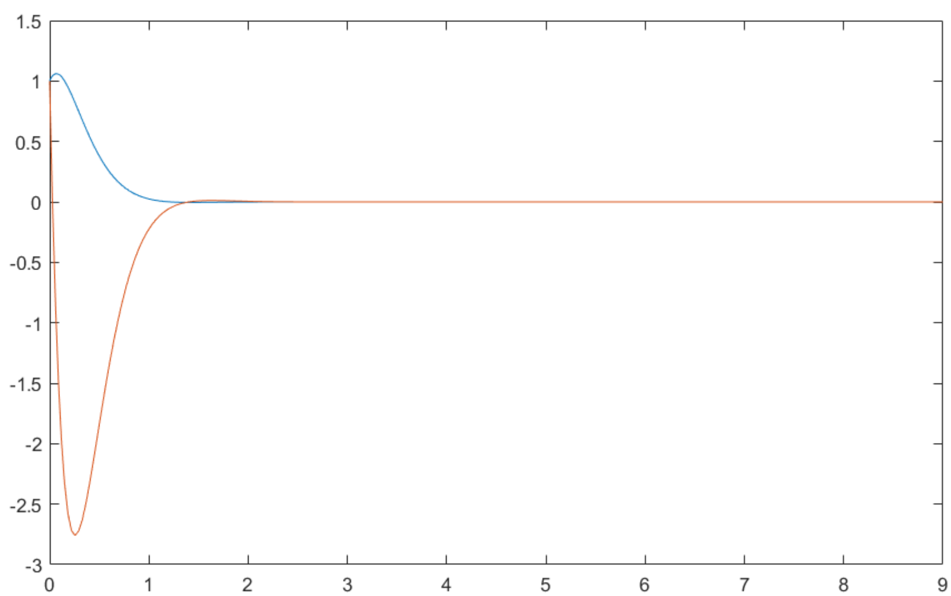

Figure 3.1: The transient response of the system with initial conditions x_0

Chapter 4

Conclusion

The first system that was given had open loop stability and this was proved by satisfying the lyapunov inequality. Then, a system with open loop instability was introduced which was stabilized by finding the optimized state feedback rule. The last system was almost the same as the second one however this system had parametric uncertainty in the system model and this problem was overcomed by solving the LMI form of the second system for the convex hull of the last system that concluded with finding the optimized state feedback to ensure asymptotically stability. For all three systems, the transient response was plotted and this verified the optimization results for stability.

4.1 Future Works

The same methods can be used to ensure stability for a state feedback system with an observer.