EE231 Lecture Note 6

Linear Quadratic Regulator

1 Linear Quadratic Regulator (LQR)

In pole placement design, we start with selecting the pole locations. However, the relationship between the pole locations and the system performance is not always clear. The Linear Quadratic Regulator (LQR) design method, allows us to determine the feedback matrix *K* directly for the 'optimum' controller performance. Optimality is defined based on a performance criterion that is specified by the designer. For regulation control, i.e., the objective of the controller is to bring the state back to the origin, the performance criterion is to minimize the following cost function

$$J = \int_0^\infty (x^T Qx + Ru^2)dt \tag{1}$$

Q is an n×n, positive definite matrix and *Q* is often diagonal. *R* is a positive number for the single-input case. *Q* is the weighting matrix for the state *x* and R is the weighting for the input *u*. According to (1), the area under x^TQx+Ru^2 from t=0 to t=∞ (J*) is the cost as shown in Figure 1. Since the objective of the regulation control is to steer *x(t)* to 0 as quickly as possible, the optimum condition is the case where the area under x^TQx+Ru^2 is minimized.

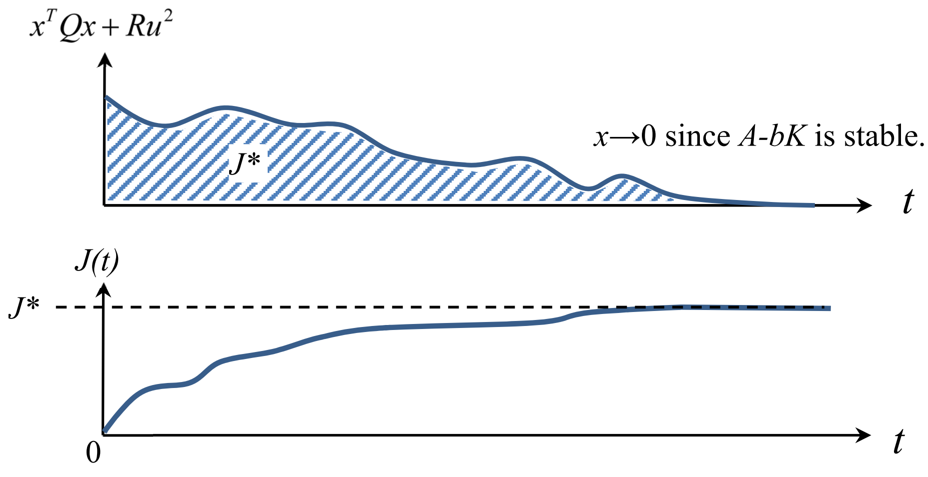

Figure 1

The matrix Q must be positive definite (i.e. $x^TQx>0$ for non-zero x) because if Q is not positive definite (i.e. $x^TQx<0$ for some x), there might be a negative area in Figure 1. In this case, some non-zero states are rewarded instead of penalized by the cost function. If Q is a diagonal matrix with all the positive diagonal elements, Q is positive definite.

A diagonal Q weights each state variable individually and independently from other state variables. For example, in the following 3×3 Q matrix, $q_{ii}>0$ is the weighting factor on the state variable x_i. A higher value of q_{ii} put a higher penalty on x_i for not being zero. In other words, the controller should try 'harder' to force this x_i to zero than to other state variables in order to achieve optimality.

$$Q = \begin{bmatrix} q_{11} & 0 & 0 \\ 0 & q_{22} & 0 \\ 0 & 0 & q_{33} \end{bmatrix} \quad \text{and} \quad x^TQx = q_{11}x_1^2 + q_{22}x_2^2 + q_{33}x_3^2 \geq 0 \tag{2}$$

R, a scalar for a single input system, is the weighting on the cost of the control effort 'u'. A higher weighting on the input results in a slower converging system but the controller would not need to 'work as hard', i.e., the input 'u' is smaller in magnitude. This is for the consideration of the limitations of the actuators (motors, heating elements, etc.). An aggressive design results in a high-performing system but only in theory. In reality, the actuator might not be able to produce high enough control (very high torque from the motor or very high power from the heating elements, e.g.) to achieve the designed performance.

For LQR design, the designer only needs to determine the relative cost factor (i.e., the weighting) associated with each state variable and with the input.

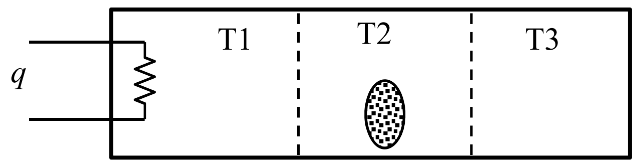

Figure 2

For example, in the 3-zone thermal chamber as shown above, the sample is placed in the middle section, and therefore it is most important that the temperature T2 is controlled precisely. For that, the cost associated with T2 should be high. Also, the heating element has only limited power output so the controller should not ask for high heat input. So, the cost of the input should also be high. The following is a possible choice of Q and R.

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad R = 5 \tag{3}$$

The absolute magnitudes of Q and R are not important. It is the <u>relative magnitude</u> that determines the relative cost. For example, if the Q and R are multiplied by 10, the result of the design (i.e., the value of *K*) remains unchanged.

The following Matlab command is LQR design command for given A, b, Q, and R.

```
[K,P,CLP]=lqr(A,B,Q,R);
```
(4)

where *K* is the state feedback vector, CLP contains the closed-loop poles of the system (i.e., eigenvalues of A-bK), and P will be explained later in this lecture.

Example:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -0.1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} u \tag{5}$$

All figures in Figure 3 are the response of the system to the initial state: x(0)= [1 1 1]^T.

The top two figures in Figure 3 show the response of two pole-placement designs. The one on the left is from the closed loop poles at (-5, -2+2j, -2-2j) and the one on the right at (-10, -4+4j, -4-4j). The corresponding feedback gain *K* is shown in the figures. Notice the high control effort, i.e., the magnitude of input ‘u’ is required to place the poles further to the left on the s- s-plane. The figures on the 2nd to the last rows in Figure 3 are the responses from LQR designs. Notice that the response of the state variable (x1, x2, or x3) or the input (u) that is weighted heavily tends to converge to zero faster. The last two cases in Figure 3 are the same because the relative magnitudes of Q and R are the same.

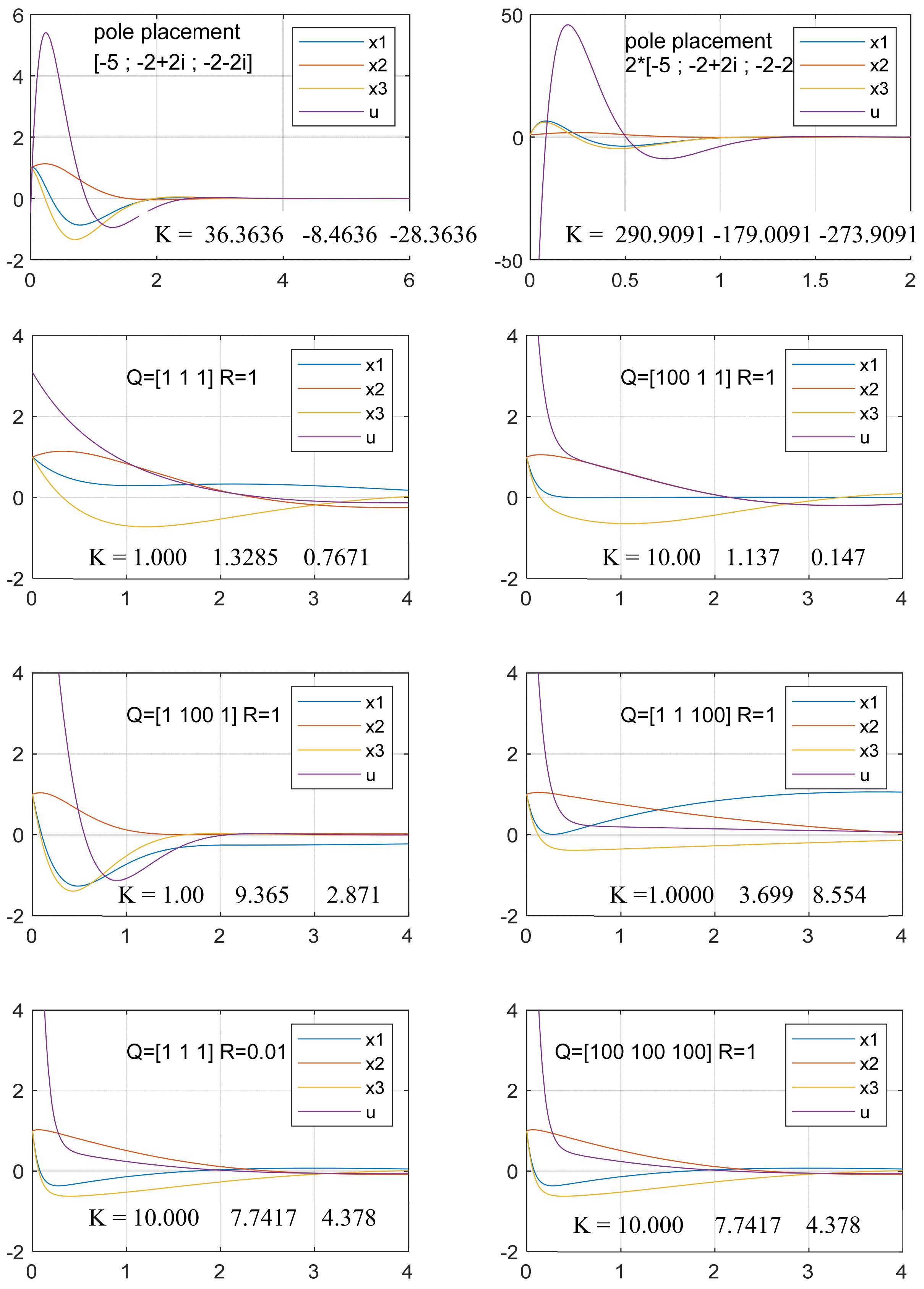
pole placement
[-5 ; -2+2i ; -2-2i]
x1
x2
x3
u
K = 36.3636 -8.4636 -28.3636
pole placement
2*[-5 ; -2+2i ; -2-2
x1
x2
x3
u
K = 290.9091 -179.0091 -273.9091
Q=[1 1 1] R=1
x1
x2
x3
u
K = 1.000 1.3285 0.7671
Q=[100 1 1] R=1
x1
x2
x3
u
K = 10.00 1.137 0.147
Q=[1 100 1] R=1
x1
x2
x3
u
K = 1.00 9.365 2.871
Q=[1 1 100] R=1
x1
x2
x3
u
K =1.0000 3.699 8.554
Q=[1 1 1] R=0.01
x1
x2
x3
u
K = 10.000 7.7417 4.378
Q=[100 100 100] R=1
x1
x2
x3
u
K = 10.000 7.7417 4.378

Figure 3

The following MATLAB script generates some of the figures in Figure 3.

```
clear all

T=0:0.01:6

A=[0 1 0; 0 0 1; 0 -0.1 -1]

B=[1;0;1]

P=[-5 ; -2+2i ; -2-2i] ;

K1=place(A,B,P)

H=ss(A-B*K1,B,K1,0)

X0=[1; 1; 1];

figure(1)

[Y,T1,X]=initial(H,X0,T)

plot(T,X,T,Y)

legend('x1','x2','x3','u');

text(1,5.5,'pole placement')

text(1,4.5,'[-5 ; -2+2i ; -2-2i]')

axis([0 6 -2 6])

grid on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Q=[1 0 0; 0 1 0; 0 0 1];

R=1;

[K2,S,e]=lqr(A,B,Q,R);

H=ss(A-B*K2,B,K2,0)

figure(2)

[Y,T1,X]=initial(H,X0,T)

plot(T,X,T,Y)

legend('x1','x2','x3','u');

text(1,5,'Q=[1 1 1] R=1')

axis([0 6 -2 6])

grid on
```

2 Derivation of LQR solution

The objective is to find a state feedback controller (i.e., the state feedback gain *K*)

$$u = -Kx \tag{6}$$

for the state equation

$$\dot{x} = Ax + Bu \tag{7}$$

such that both following two conditions are met.

(i) *x(t)* converges to zero. This means that system (7) must be controllable.

(ii) The cost *J** as defined in (8) is minimized. *Q* and *R* are positive definite weighting matrices. Recall that, a matrix Q is positive definite if, for any vector $x \neq 0$, $x^T Q x > 0$. This ensures that all non-zero states contribute to the cost.

$$\begin{aligned} J^* &= \int_0^\infty (x^T Qx + u^T Ru)dt = \int_0^\infty (x^T Qx + x^T K^T RKx)dt \\ &= \int_0^\infty x^T \left(Q + K^T RK\right) x dt \end{aligned} \tag{8}$$

Consider the following three equations:

$$\dot{x} = Ax + Bu = (A - BK)x \tag{9}$$

$$J(t) = \int_0^t x^T \left(Q + K^T RK\right) x dt \tag{10}$$

$$\frac{d}{dt} J(t) = x^T \left(Q + K^T RK\right) x \, . \tag{11}$$

- Equation (9) is the closed-loop system equation.
- Equation (10) is the 'up to time *t*' cost of the system in (9) and $J(\infty) = J^*$.
- Equation (11) is derived from (10) by differentiating both sides of (10).

x(t) is the state trajectory of the closed-loop system (9) and this *x(t)* is used in (10) and (11).

Now let's construct a <u>scalar</u> differential equation (12) where *Q* and *R* are the same as those in the cost function (8), and *P* (to be determined) is a positive definite matrix.

$$\frac{d}{dt}\left(x^T Px\right) = -x^T \left(Q + K^T RK\right) x \, . \tag{12}$$

Note that, although we use the same symbol *x(t)* in (12) and in (9), (10), or (11), there is no reason to believe that state trajectory *x(t)* of (9) satisfies (12) unless we can derive a *K* and a *P* such that the state trajectory *x(t)* satisfies (12). In order to find such *K* and *P*, we now assert that

(a) the closed-loop system (9) is stable and

(b) *x(t)* in (9), (10) and (11) also satisfy (12).

From these assertions, we make the following three observations.

a) At t=0, the cost is zero, i.e., $J(0)=0$ and it converges to $J^*=J(\infty)$ as $t\to\infty$.

b) x^TPx converges to 0 as$\to\infty$ since (10) is stable, i.e., $x(t)\to 0$.

c) The rate of change of *J(t)* and the rate of change of x^TPx have the same magnitude but opposite sign --- *J(t)* always goes up while x^TPx always goes down as shown in Figure 4.

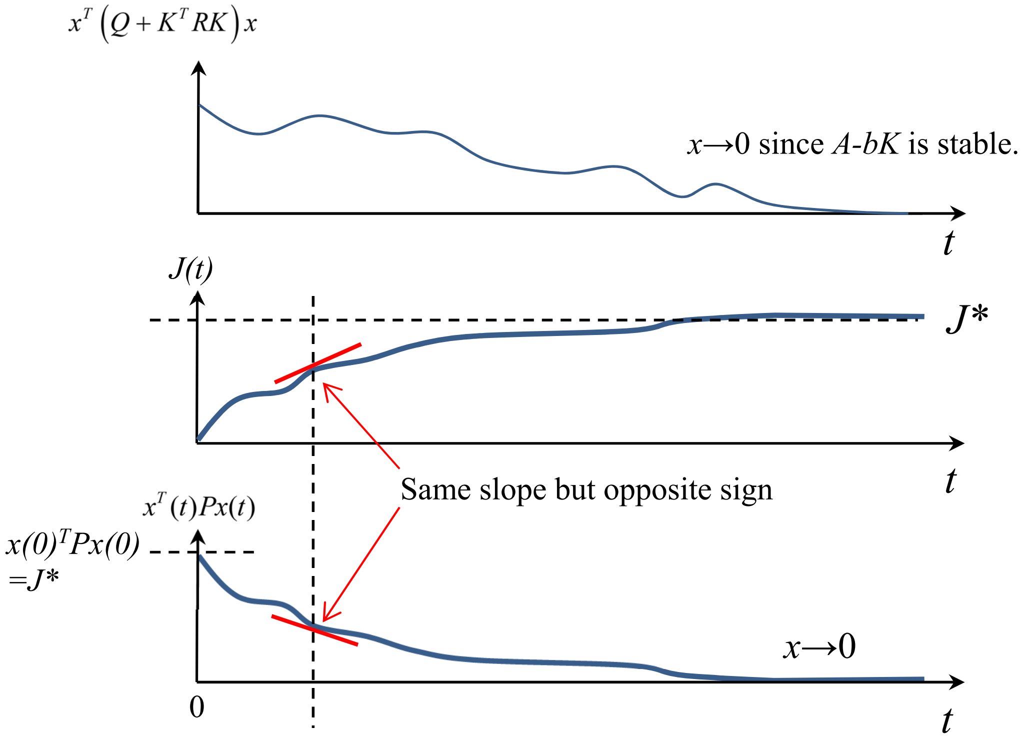

Figure 4

From observation (a), *J*(t) in (10) starts from 0 and ends at *J**. From observations (b), x^TPx ends at 0. Since the rates of change of *J(t)* and x^TPx have the same magnitude, from (11) and (12), the total change of *J(t)* and x^TPx must be the same! We can conclude that the initial value of x^TPx

must be the same as the final cost J^*! In other words, $x^T(0)P(0)x(0) = J^*$. This same idea can also be derived by equation (13). Note that, in (13), we used the assertion that $x(t)$ satisfies both (12) and (10).

$$\begin{aligned}
J^* &= \int_0^\infty \left\{ x^T\left(Q + K^T RK\right)x \right\} dt = \int_0^\infty -\frac{d}{dx}\left(x^T Px\right) dt = -x^T(t)\,Px(t)\,|_0^\infty \\
&= -x^T(\infty)\,Px(\infty) + x^T(0)\,Px(0) \\
&= x^T(0)\,Px(0), \qquad\qquad \left(\text{since } x(\infty) = 0 \text{ for stable system.}\right)
\end{aligned} \tag{13}$$

This observation shows that we should look for the P that renders the smallest $x^T(0)Px(0)$, i.e., the lowest cost J^*.

From the assertions, we can substitute (9) into (12) and obtain (14).

$$\begin{aligned}
\frac{d}{dt}\left(x^T Px\right) &= \dot{x}^T Px + x^T P\dot{x} + x^T \dot{P}x \qquad (\dot{P} = 0 \text{ since it is constant.}) \\
&= x^T (A - BK)^T Px + x^T P(A - BK)x \\
&= x^T \left[(A - BK)^T P + P(A - BK) \right] x
\end{aligned} \tag{14}$$

In this lecture, we only consider the case that P is a constant matrix, i.e. $\dot{P} = 0$. We now equate the right sides of (12) and (14), and obtain (15). We will find the matrix P and K based on this identity.

$$-x^T\left(Q + K^T RK\right)x = x^T\left[(A - BK)^T P + P(A - BK)\right]x \tag{15}$$

Since (15) must be true for any x, it implies (16) or equivalently (17).

$$Q + K^T RK = -\left[(A - BK)^T P + P(A - BK)\right] \tag{16}$$

$$\left[(A - BK)^T P + P(A - BK)\right] + Q + K^T RK = 0 \tag{17}$$

There are two unknowns in (17), P and K. There are many P and K pairs that satisfy (17). Since $x^T(0)Px(0)$ is the cost J^*, we are looking for the pair that has the 'smallest' P and the K in this pair is the answer we are looking for.

3 Solution of LQR problem (a scalar case)

To see how to derive *P* and *K*, for simplicity, we first consider the scalar case. (18) is the scalar case of (17).

$$2p(a-bk)+q+k^2r=0 \tag{18}$$

From (18), *p* can be expressed as:

$$p=\frac{-(q+k^2r)}{2(a-bk)} \tag{19}$$

Note that *p* is a function of *k*. Since the objective is to find the smallest *p* for some *k*, we take the derivative of (19) with respect to k.

$$\frac{dp}{dk}=\frac{brk^2-2ark-bq}{2(a-bk)^2} \tag{20}$$

To find the minimum value of p, we set the numerator in (20) to zero, i.e.,

$$brk^2-2ark-bq=0\,. \tag{21}$$

Now, equations (18) and (21) form a two-equation-two-unknown problem and, therefore, *p* and *k* can be solved from these two equations. To simplify the equations, we multiply (18) by *b* and add the product to (21). Equation (22) is result.

$$2p\left(ba-b^2k\right)+2brk^2-2ark=0 \tag{22}$$

Equation (22) can be simplified to (23) and then (24).

$$p=\frac{-2brk^2+2ark}{2\left(ba-b^2k\right)}=\frac{2rk\left(a-bk\right)}{2b\left(a-bk\right)}=\frac{rk}{b} \tag{23}$$

$$k=\frac{bq}{r} \tag{24}$$

Substituting (23) into (18), we obtain:

$$2pa-\frac{b^2p^2}{r}+q=0 \tag{25}$$

Solving (24) for *p*, we obtain:

$$p = \frac{2a \pm \sqrt{4a^2 + 4\left(b^2/r\right)q}}{2\left(b^2/r\right)} \tag{26}$$

Since *p*>0, the solution is :

$$p = \frac{2a + \sqrt{4a^2 + 4\left(b^2/r\right)q}}{2\left(b^2/r\right)} \tag{27}$$

Once *p* is determined, *k* can be found by (23).

4 LQR solution -- The Riccati Equation

The derivation for vector cases is much more complex. Here, we will simply extend the results (24) and (25) to their vector counterparts (28) and (29), respectively.

$$K = R^{-1}B^T P \tag{28}$$

$$A^T P + PA - P^T BR^{-1}B^T P + Q = 0 \tag{29}$$

Equation (29) is known as the algebraic Riccati equation. As a side note, if we did not assume that *P* is a constant, the $\dot{P}$ from (14) will be carried to (29) and that would make it a general Riccati differential equation.

Both *K* and *P* can be found by using MATLAB command in (30).

```
[K,P,CLP] = lqr(A,B,Q,R,N)
```
(30)

In (29), `CLP` is the closed pole, i.e., the eigenvalues of *A-BK*. *N* is the weighting associated with the cross terms between the input *u* and state *x*. N is usually zero and is not considered in this note.

Example: The system given in (5) and the following weightings are used for this simulation.

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad R = 1.$$

This is one of the cases in Figure 3. The Matlab `lqr` command returned the following values.

$$K = \begin{bmatrix} 1 & 9.3655 & 2.8715 \end{bmatrix} \qquad P = \begin{bmatrix} 8.6050 & -4.7336 & -7.6050 \\ -4.7336 & 49.6444 & 14.0991 \\ -7.6050 & 14.0991 & 10.4765 \end{bmatrix}$$

Figure 5 shows the cost function *J(t)* and $x(t)^T Px(t)$. As shown, the relationship between these two waveforms are exactly as shown in Figure 4.

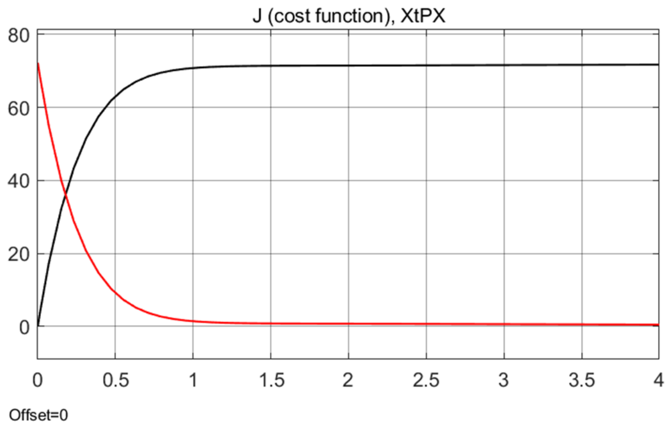

Figure 5

Figure 6 shows the magnitude square of the state vector *x(t)* (i.e., $x(t)^T x(t)$. As shown, unlike $x(t)^T Px(t)$, $x(t)^T x(t)$ is not a monotonically decreasing function.

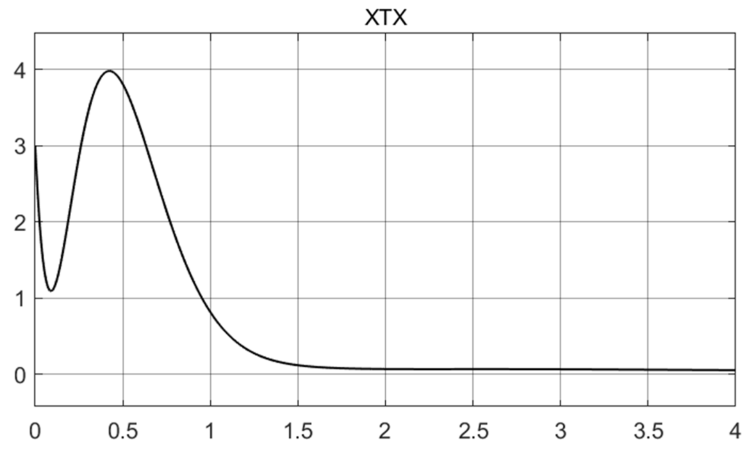

Figure 6