Linear Quadratic Regulator and Observer Design for a Flexible Joint

Kevin M. Passino and Nicanor Quijano

Dept. Electrical Engineering, The Ohio State University
2015 Neil Avenue, Columbus, OH 43210-1272

April 8, 2002

Abstract

First, using full state feedback you design a linear quadratic regulator (LQR) for endpoint position control in the face of flexibility effects for the flexible joint experiment. "Full state feedback" is achieved by using a tachometer and encoder to measure the angular velocity and position of the DC servo output shaft, an encoder for the link angle, and a derivative filter to produce the angular rate of the link angle. Next, you will develop an observer for the joint angle and its rate, along with the DC servo shaft angular rate, and then use its estimated state as the input to the LQR for position control. Successful observer design results in overcoming flexibility effects using only the encoder for the shaft position (i.e., without *any* sensor for the link angle or its rate).

Contents

1 Linear Quadratic Regulator

1.1 Model Development and Analysis

First, you will develop a state space model for the flexible joint that will be used in the LQR and observer designs. For this you should refer to the model development section on the flexible link that is in the design challenges document that is on the web. Suppose that the state space model

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{1}$$

has an $n \times 1$ state x and a 1×1 input u. Recall that for the flexible link

$$x = \left[\theta, \alpha, \dot{\theta}, \dot{\alpha}\right]^\top$$

1. Find the A, B, and C matrices for a state space description of the plant. Assume the $[A, 3]$ configuration.
2. Is the plant "controllable"? Use Matlab to provide your answer.

1.2 LQR Development

Suppose that we have sensors to measure the entire state and that we use a controller (regulator)

$$u = -Kx$$

that seeks to drive the state to zero. Regulator design entails finding the $n \times 1$ gain vector K. You could use pole placement via Ackerman's formula. Here, we use the LQR methodology to specify the gain K. For this, let

$$J = \int_0^\infty \left(x^\top Q x + u^\top R u\right) d\tau$$

and we seek to find the gain vector K to *minimize* this "cost function." Minimization of J results in moving x to zero with as little control energy and state deviations as possible, with the balance between control energy and state deviations specified via the Q and R matrices. Here, assume the $n \times n$ Q matrix is diagonal with diagonal elements $q_i \geq 0$ (each providing a weight for a different element of the deviation of the state) and the scalar $R > 0$. The values for Q and R are used as design parameters. How do you tune them? If you choose all the $q_i = 0$ this represents that you do not care what type of excursions the state has while the control tries to force the state to zero. High values of q_i relative to R mean that you are willing to use lots of control energy to keep state excursions small while driving it to zero. Clearly, you cannot pick $R = 0$ as this results in allowing infinite control energy to force the state to zero, typically then very fast.

Finding the K to minimize J involves solving a "Ricatti equation." You can use the Matlab "`lqr`" command to directly solve for the gain vector K given A, B, Q, and R. This is the approach you will use in all your LQR designs.

Next, you will design the LQR:

1. Develop a simulation for the closed-loop system so that you can test your various designs. For your simulation you may want to use the Matlab command "`initial`".
2. Find values of Q and R that result in as fast of a response as possible but with an overshoot of less than 5 % when the intial condition has zero for all components except the intial output shaft position can be up to 30 deg. from its zero position. Also, the control input to the plant should not exceed ± 5V (simply check this via simulation) for your choice of Q and R. Why?
3. Show the closed-loop response for the gains you choose.

2 Observer Design

2.1 Observability and Observer Equations

Test the observability of the system for different uses of sensors. Use the model above and Matlab to show that the state is observable using only the DC servo output shaft position θ (i.e., no sensing of the link angle). This shows that there is the potential to design an observer to produce an estimate of the state. Then, using the separation principle we can use this state estimate as the input to a state feedback controller to obtain a regulator for the system. In this way we implement the regulator *without* the use of two sensors (the link angle encoder and tachometer for shaft angular velocity) that were needed in the last section (in a practical application this would lead to a significant cost savings).

Suppose that we use the A and B that you found earlier and

$$y = [1\ 0\ 0\ 0]x = C_1 x$$

so that we are only measuring θ. The observer that produces the estimate of the state, $\hat{x}$, is of the form

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ \hat{y} &= C_1 \hat{x} \end{aligned}$$

The "observer gain" vector is the $n \times 1$ vector L and it must be chosen. If you let $e = x - \hat{x}$ then

$$\dot{e} = (A - LC_1)e$$

for the dynamics on the estimation error. Show this. Clearly, the choice of L can lead to different estimation error dynamics (e.g., rate at which the estimation error goes to zero).

2.2 Observer Gain Selection Via Pole Placement

You can use the Matlab commmand "`place`" (just like you can for full state feedback control design) to find the value of the gain vector L to specify where you want the poles of the observer. One way to do this is to use the following Matlab code fragment (clearly $C_1 = C(1,:)$):

```
rho=?; % Value not included - you tune it

P=rho*[-1, -1.5, -2, -2.5]  % These are simply suggested values

Ltilde = place(A',C(1,:)',P);

L=Ltilde' % Observer gains to place poles

obspoles=eig(A-L*C(1,:))
```

Answer the following questions:

1. What is P above? What is the effect of the scalar $\rho > 0$ (rho)?
2. Explain why the transposes of A, C_1, and $\tilde{L}$ are used? Derive the formulas to show why.

Next, suppose that you want to form the closed-loop system when the control

$$u = -K\hat{x}$$

is used (yes, now we do not use state feedback, but the state estimate is fed back instead). The following Matlab code fragment is useful:

```
Atot=[A          -B*K;
   L*C(1,:)  A-L*C(1,:)-B*K];

Btot=0*ones(8,1);

Ctot=[C(1,:) 0*ones(1,n)];

totaleigenvalues=eig(Atot)
```

Questions:

1. What does this code fragment do?
2. What is the state of the system that it represents?
3. Derive the above formulas in the code fragment. Show your work.

2.3 Observer/LQR Testing

1. Form the closed-loop system and use "`initial`" in Matlab to simulate it. Also, provide the input to the plant on a plot in additioon to the states and their estimates (on the same plots).
2. Find values of ρ (rho) and P so that you get a good observer response, one that is faster than the response of the LQR designed earlier, so that when it is used with the state feedback controller it will provide a reasonably good estimate of the state quickly so that the LQR can control the plant properly and give a reasonably good transient behavior.
3. Compare the closed-loop response when the LQR uses full state feedback vs. when it uses the state estimate. Your observer design should be good enough so that you can meet specifications as in the state feedback case (see above).

3 Additional Challenges

Finally, the following tasks are *optional* (but lots of fun!):

1. Design and test a "reduced order observer."
2. Develop a set of simulations to illustrate the "peaking phenomenon" (see Khalil's book on Nonlinear Systems for an explanation of that).
3. Develop and test a Kalman filter for estimating the state. For this you will need a noise model. A reasonable one for this system is to assume noise only in the output channel that is zero mean with a normal distribution and a variance that you could treat as a design parameter (of course, alternatively, you could study the characteristics of the encoder and try to specify a noise model).
4. Using the handout on the flexible joint model find a model for another spring configuration and test all your designs on this different model (this is a rather simple and very incomplete robustness test).