Automatic Control II

Computer exercise 3

LQG Design

Preparations:
- Read Chapters 5 and 9 in the textbook by Glad and Ljung.
- All preparation exercises in Section 3.
Contents

1 Introduction 1

2 LQG: Short theoretical background 1

3 Preparation exercises 2

4 Balancing a stick 4

5 The two wheeled inverted pendulum process 7

6 Simulation 9
   6.1 State feedback: LQ design 9
   6.2 Optimal state estimation: The Kalman filter 11
1 Introduction

In this computer exercise, controller synthesis with LQG design will be investigated. The synthesis will be performed on two variants of the inverted pendulum. In Section 4 the system is a stick balancing on a finger, and in Section 5 the system is a two wheeled inverted pendulum, which is actuated by DC motors. This is the same system that will be used in the process lab.

To be well prepared for this computer exercise one should have read the Chapters 6, 8.5 and 9 in the course book by Glad-Ljung thoroughly. It may also be beneficial to have read the Chapters 5 and 7.

For the computer exercise, access to Matlab and the Control System Toolbox is necessary. There are also a number of .m-files that are needed.

2 LQG: Short theoretical background

The standard LQG problem is to minimize a quadratic criterion for a linear system. That is, to find the controller that, when applied to the system, minimizes the criterion. The standard model (as in Equation (9.4) in Glad-Ljung) of the system is

\[ \dot{x} = Ax + Bu + Nw \]  
\[ z = Mx \]  
\[ y = Cx + v \] 

where \( z \) is the performance signal, \( y \) is the measured output signal, and \( w \) and \( v \) are the white noise-signals with covariance matrices

\[ Eww^T = R_1, \quad Evv^T = R_2, \quad Ewv^T = R_{12} \] 

For simplicity, assume that \( w \) and \( v \) are independent, so that \( R_{12} = 0 \).

The control error, i.e., the discrepancy between the performance signal and the reference signal is denoted \( e = z - r \). It is of course desirable to keep \( e \) as small as possible. The standard formulation of LQG, though, treats the regulation problem, i.e., when \( r \equiv 0 \). This means that \( e = z \). The criterion to be minimized then is

\[ V = E \lim_{T \to \infty} \frac{1}{T} \int_0^T (z(t)^T Q_1 z(t) + u(t)^T Q_2 u(t)) \, dt \] 

where \( Q_1 \) and \( Q_2 \) are symmetric matrices. \( Q_2 \) is positive definite and \( Q_1 \) is positive semidefinite.

When treating the servo problem, when \( r \neq 0 \), the criterion should be appropriately modified. However, here \( r \) will be confined to be piece-wise constant, like set point changes. In this case LQG will give a feedback gain that is the same as for the regulation problem.
The optimal control law has the familiar observer based state feedback structure (see Theorems 9.1 and 9.2 in Glad–Ljung)

\[ u(t) = -L\hat{x}(t) + L_r r(t) \]  

where \( \hat{x}(t) \) is the optimal estimate of \( x(t) \), obtained by the Kalman filter for the system (1)–(3). The gain \( L \) depends only on \( Q_1 \) and \( Q_2 \), while the Kalman filter depends only on \( R_1 \) and \( R_2 \) (and \( R_{12} \) if nonzero).

Note that the LQG-controller is optimal only for the idealized situation, that is, when the system is exactly described by (1)–(3) and when \( w \) and \( v \) are white Gaussian noises. This is a very idealized situation. In practical cases one or more of the conditions for the LQG framework may not be fulfilled. In this case study the system is not even a linear model. Hence, it can not be expected that the obtained controller will be optimal. Still, for many systems the LQG approach is useful due to its simplicity and to its ability to generate controllers that behave rather well also in nonidealized situations.

In practice \( R_1 \) and \( R_2 \) are unknown and may, together with \( Q_1 \) and \( Q_2 \), be regarded as design variables. There are some guidelines in how to choose these design matrices.

For simplicity the discussion is confined to diagonal choices of \( Q_1 \) and \( Q_2 \). Each element in these matrices can be interpreted as a penalty of the corresponding signal component. The rule of thumb then is that this signal component will be small in the closed loop simulation if it is penalized considerably. That is, if the corresponding diagonal entry in the diagonal matrices are chosen large. Still, it is stressed that this is only a rule of thumb and that the design procedure should be regarded as an iterative procedure supported by closed loop simulations.

### 3 Preparation exercises

Consider a system given in the standard state space form:

\[
\dot{x} = Ax + Bu + Nw
\]  
\[
z = Cx
\]  
\[
y = z + v
\]

The system is controlled with the LQG-controller

\[
\dot{x} = A\hat{x} + Bu + K(y - C\hat{x})
\]  
\[
u = -L\hat{x} + L_r r
\]

where (10) is the Kalman filter and (11) is the control law. \( r \) is the reference signal.
Exercise 3.1: The state estimation error is \( \hat{x} = x - \hat{x} \). What is the dynamics for \( \hat{x} \)?

Answer:

Exercise 3.2: Give the state space description for the closed loop system. Use \( \begin{bmatrix} x \\ \hat{x} \end{bmatrix} \) as state vector.

Answer:

For the next exercise the following identity — regarding block triangular matrices — is useful:

\[
\begin{bmatrix}
\Gamma & \Phi \\
0 & \Psi
\end{bmatrix}^{-1} = \begin{bmatrix}
\Gamma^{-1} & -\Gamma^{-1}\Phi\Psi^{-1} \\
0 & \Psi^{-1}
\end{bmatrix}
\]

\( \Gamma \) and \( \Psi \) are square nonsingular matrices, and \( \Phi \) is a matrix of compatible dimension.
**Exercise 3.3:** The Kalman filter does not affect the dynamics from reference signal to output signal. Show this by computing the transfer function from reference to output. Use the state space description obtained in the previous task.

**Answer:**

---

**4 Balancing a stick**

Example 7.1 in Glad-Ljung concerns the problem of balancing a stick on the finger. For simplicity the situation is idealized to the case where all motion is confined to a vertical plane. Also, the finger is confined to move along a horizontal line, in the $\xi$-direction. The input signal is the acceleration in this direction, i.e., $u = \ddot{\xi}$. The output signal is the angle of the stick with respect to a vertical line (see Figure 7.7 in Glad-Ljung). The second order differential equation (7.32) in Glad-Ljung describes the relation between the input and the output signals. Introducing the state variables $x_1 = y$ and $x_2 = \dot{y}$ yields the nonlinear state space description

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{2g}{l} \sin x_1 - \frac{2}{l} u \cos x_1 \\
y &= x_1
\end{align*}
\]

If the system is linearized around $x = 0$ and $u = 0$, the linear (and hence approximative) state space model is

\[
\begin{align*}
\dot{x} &= \begin{bmatrix} 0 & 1 \\ \frac{2g}{l} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ -\frac{2}{l} \end{bmatrix} u \\
y &= \begin{bmatrix} 1 & 0 \end{bmatrix} x
\end{align*}
\]

The corresponding transfer function is

\[
G(s) = \frac{-2/l}{s^2 - \frac{2g}{l}}
\]
The system has two poles, in $\pm \sqrt{\frac{2g}{l}}$, and is thus unstable.

Consider the case when the stick is hit on its top. This can be modeled as an impulse disturbance (remember the analogy between impulses and white noise) on the angle acceleration. Hence it enters the system like the input signal, but with opposite sign. The standard model of the system then is

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ \frac{2g}{l} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ -2 \end{bmatrix} u + \begin{bmatrix} 0 \\ \frac{2}{l} \end{bmatrix} w$$

$$z = \begin{bmatrix} 1 & 0 \end{bmatrix} x$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x + v$$

where $w$ is the process noise, representing the disturbance, and $v$ is the measurement noise.

LQG is applied to design controllers for the system. Set $Q_1 \equiv 1$ and $R_1 \equiv 1$ (this is no restriction, since it is only the ratios, $Q_1/Q_2$ and $R_1/R_2$, that affects the controllers for SISO systems). Before you proceed with the labs, execute the `initpend.m` file to add the required files to the path.

First set $R_2 = 10^{-6}$. For $Q_2 = 10^{-2}, 10^{-3}$ and $10^{-4}$, generate controllers with `lqgstick`. To use this command, type $\text{reg1=lqgstick(Q1,Q2,R1,R2)}$; and so on. Use `tfstick` to study the transfer functions. Type `tfstick(reg1,reg2,reg3)`. 

**Exercise 4.1:** How does the magnitude of $Q_2$ affect (a) the bandwidth of $G_{ry}(s)$? (b) the bandwidth of $S(s)$ (where $S(s)$ is the sensitivity function of the system)? (See Section 6.6 in Glad-Ljung for definitions on bandwidth.)

**Answer:**

Simulate the system with `simstick` for the different controllers. This is done by writing `simstick(reg1,reg2,reg3)`. This will simulate a unit step on the reference signal, and a unit impulse disturbance (a hit on the top of the stick). The responses on the output and input signals are shown. It is the linear model that is simulated.
Exercise 4.2: How does the magnitude of $Q_2$ affect (a) the output signal? (b) the input signal?

Answer:

Now let $Q_2 = 10^{-3}$. For $R_2 = 10^{-5}, 10^{-6}$ and $10^{-7}$, generate controllers with \texttt{lqgstick}.

Exercise 4.3: How does the magnitude of $R_2$ affect (a) the bandwidth of $G_{ry}(s)$? (b) the bandwidth of $S(s)$?

Answer:

Simulate the system for the different controllers.
Exercise 4.4: How does the magnitude of $R_2$ affect (a) the output signal? (b) the input signal?

Answer:

5 The two wheeled inverted pendulum process

In this part LQG control of a two wheel inverted pendulum system (the same system that will be used in the process lab) will be studied. The system is a robot balancing on two wheels, as shown in Fig. 1.

To be able to control the system with an LQ-controller, the dynamics are linearized around the origin (an equilibrium point). The origin is defined as $x_0 = [0 0 0 0 0 0]^T$. That is, the robot is standing still at a vertical position. The state space model is of the form:
\[ \begin{align*} 
\dot{x} &= Ax + Bu + Nw, \\
z &= Mx, \\
y &= Cx + v, 
\end{align*} \]

with the state vector \( x \), measurement vector \( y \), performance signal \( z \) and the controllable input signal \( u \) given by

\[ \begin{align*} 
x &= \begin{bmatrix} \xi & \phi & \psi & \dot{\xi} & \dot{\phi} & \dot{\psi} \end{bmatrix}^T, \\
y &= \begin{bmatrix} \theta_l & \theta_r & \dot{\psi} \end{bmatrix}^T, \\
z &= \begin{bmatrix} \phi & \xi \end{bmatrix}^T, \\
u &= \begin{bmatrix} u_l & u_r \end{bmatrix}^T, 
\end{align*} \]

where

- \( \xi \) - traveled distance
- \( \phi \) - yaw angle
- \( \psi \) - pitch angle
- \( u_l \) - voltage applied to left motor
- \( u_r \) - voltage applied to right motor
- \( \theta_l \) - left shaft angle (measured by a tachometer in the motor assembly)
- \( \theta_r \) - right shaft angle (measured by a tachometer in the motor assembly)

The process noise is modeled as a disturbance on the input signal, which means that \( N = B \). Numerically the matrices are given by

\[ A = \begin{bmatrix} 
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & -1.9 & -28.1 & 0 & 1.1 \\
0 & 0 & 0 & 0 & -30.5 & 0 \\
0 & 0 & 88.1 & 295.4 & 0 & -11.8 
\end{bmatrix}, \]

and

\[ B = \begin{bmatrix} 
0 & 0 \\
0 & 0 \\
0 & 0 \\
1.1 & 1.1 \\
15.5 & -15.5 \\
-11.9 & -11.9 
\end{bmatrix}, \]
\[ C = \begin{bmatrix} 25 & -2 & -1 & 0 & 0 & 0 \\ 25 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \]

As the performance signal is given by \( z = [\phi \ ˆ{\xi}]^T \), it follows that

\[ M = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \]

The poles of the linearized system are given by

\[ p = [0 \ -40.9 \ 7.4 \ -6.4 \ 0 \ -30.5] . \]

6 Simulation

In the simulations the nonlinear model will be used. The simulations are performed under circumstances that should be as similar to the situation in a real experiment as possible. For instance, the controller is implemented as a sampling controller, the measurements are quantized, and so on. Most important here, though, is that the input signal is bounded; \( u \) must lie within the interval \([-6, 6]\) Volts. This must be taken into account when the controller is designed.

Notice that the controller design is based on the linear model. Also, the transfer functions reflects the effects on the linear model. However, the goal is to get a controller that works on the nonlinear model in the simulations.

To perform a simulation, first create a controller by using the controller generating command for the specific task, e.g.

\[ \text{reg} = \text{lq.twip.lab}(\text{Q1}, \text{Q2}) \]

to simulate this controller use the to wheeled inverted pendulum GUI: \text{twip gui}. Open it by typing \text{twip gui} in matlabs command prompt and hit enter. To simulate, enter the name of the controller in the text box under “Controller selection” in the upper right corner, i.e. “reg” in the example give here, and press the “simulate” button to perform the simulation. In the simulation a step of 45° in the yaw reference at time \( t = 0 \) s is applied and a step in the velocity reference of 0.1 m/s will be applied at time \( t = 0.5 \) s.

6.1 State feedback: LQ design

One method to design the controller is to first find a state feedback gain \( L \) that would work well if pure state feedback was applicable. Then, once such \( L \) is
found, the Kalman filter is chosen so that the true but unknown state vector
 can be replaced with its best estimate in order to compute the control signal\textsuperscript{1}.

Before the experiments begin you should note that the model consists of two
independent subsystems. The system for the yaw angle (the turning angle)
is independent of the velocity-pitch system. The control synthesis could thus
actually be performed on each subsystem independently of each other. It is
however convenient to treat the whole system as one single system and apply
the LQ framework to it. This decoupling of the two subsystems occurs in the
linearized model only, in the nonlinear state space model for the system the
subsystems are actually not independent.

To begin with we will focus on the velocity-pitch subsystem. Since this subsys-
tem is unstable it is more crucial to be able to control this than to control the
yaw angle. Design an LQ controller with

\[
Q_1 = \begin{bmatrix}
q_\phi & 0 \\
0 & q_\xi
\end{bmatrix}
\]

(12)

\[
Q_2 = \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]

(13)

where $q_\phi = 10^{-10}$. By selecting this small value on $q_\phi$ the controller will basically ignore controlling the yaw angle. Note though that it can not be put to zero since the LQ methodology can then not find a positive definite solution to the Riccati equation. To create the controller use the function

\[
\text{reg} = \text{lq\_twip\_lab}(Q_1, Q_2)
\]

and simulate using the \textit{twip\_gui}. Note that the simulation should be performed \textbf{without} noise!

**Exercise 6.1:** \textit{Try some different values of $q_\xi$ to see how the step response of the velocity changes.}

**Answer:**

\textsuperscript{1}It is of course possible to do it the other way around as well, i.e., to first determine the
Kalman filter, and then $L$, or one could try to determine both simultaneously.
After some trials you will notice that the step response is very little affected by the choice of $q\dot{\xi}$. The reason for this is that the system have a zero in the r.h.s. of the complex plane, which puts an upper bound on the performance, and the system also have a pole in the l.h.s. of the plane which puts a lower bound of the performance of the control. The performance then gets "trapped" in between this lower and upper bound and is little affected by the choice of $q\dot{\xi}$. This is a fundamental limitation posed by the system, and could not be resolved by using a "better" linear controller. The key to understand here is that even though the system is controllable it is not always possible to control it arbitrarily. These fundamental limitations are explored in the course Automatic control III.

In the next step you will try to control the yaw angle. This subsystem does not posses the fundamental limitations present in the velocity-pitch system, and hence you should be able to affect the step response more significantly in this case. Try some different values on $q\phi$.

**Exercise 6.2:** Comment on the step response on the yaw angle in terms of fastness (fast/slow) and also in terms of input signal magnitude (large/small). What is your choice for $Q_2$?

<table>
<thead>
<tr>
<th>Answer:</th>
</tr>
</thead>
</table>

**6.2 Optimal state estimation: The Kalman filter**

Stick to $Q_1$ and the $Q_2$ obtained above. As for the state feedback control problem discussed above, the Kalman filter depends only on the process and measurement noise intensities, $R_1$ and $R_2$. In practice these are unknown and may be regarded as design parameters. A priori knowledge about the noise intensities can be useful in the tuning procedure.

Once again it is sufficient to consider only one parameter. It is rather natural to compare the intensities of the process noise $R_1$ and the measurement noise $R_2$. Set $R_1 = I$ and $R_2 = rI$, where $r > 0$ is varied. *Note that both $R_1$ and $R_2$ are $2 \times 2$ symmetric positive definite matrices.* Choosing $r$ small means that the measurement noise is considered small compared to the process noise. The corresponding Kalman filter will then become fast, since it will put more trust in the measurements than in the system dynamics. And, vice versa, a large
$r$ will give a slower Kalman filter. In this system the measurement noise is probably small (primarily caused by quantization in the sensors). It is harder to judge the size of the process noise. It includes the disturbances in the system as well as model uncertainties. Particular in this case where the real system is nonlinear, while the model is linear, this should be taken into account. The system is unstable as well, which invokes the controller to be fast enough (and particular the state estimator). All together this suggest that $r$ should be chosen quite small. Try some different values on $r$. To see any essential effect in the behavior of the system, $r$ should be changed by a factor of ten or hundred. Generate a controller using

```matlab
reg = lqg_twip(Q1,Q2,R1,R2)
```

and use the `twip_gui` to simulate the system. Note that the simulation should be performed with noise!

**Exercise 6.3:** What $r$ have you chosen?

**Answer:**

12