The Path From LQG to Robust Control Theory

Bite-Sized Notes

Sam Vial

22 May 2024

Introduction

At the beginning of week eight, we finally had all of the pieces in place for a Linear Quadratic Gaussian (LQG) system. An LQG system combines a linear quadratic regulator (LQR) control scheme with a Kallman filter state estimator. We had proven that for a system with linear dynamics and a quadratic cost function an LQR provided optimum controls to minimize the quadratic cost. Likewise we had shown that a Kallman filter was an optimum state estimator (minimizing uncertainty) for these systems. Via the Separation Prinicipal we showed that LQR and Kallman filters maintained their optimality when combined. We had even learned how to approximate linear dynamics and quadratic cost functions for systems without them so that they might get in on this good thing we have going.

Indeed much of the quarter was spent assembling, piece by piece, the tools necessary to implement LQG systems. Where do you go from this lofty perch of optimality? Consider the question, "With respect to what is LQG an optimal control/estimation framework?" Or more to the point, "In what ways are LQG schemes not optimal?" One answer came in a famously terse article published in 1978 by John Doyle [1]. Doyle titled his article, *Guaranteed Margins for LQG Regulators*. The abstract reads simply: "There are none."

These notes will examine Doyle's findings, discuss what they say about the stability of LQG systems, and see why they offer a natural transition from state space controls to the frequency domain and the field of robust control theory.

System Example of LQG Fragility

Define The System

Doyle exposes the vulnerability of LQG systems by exploring an example. Consider the simple system shown below. The system has linear dynamics, one dimension controls, and gaussian white noise process disturbance. You can see that only x_1 is measured, and the measurement is also subject to gaussian white noise.

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 1 \\ 1 \end{bmatrix} w$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + v$$

Here, our process noise and gaussian noise are $w \sim \mathcal{N}(0, \sigma_w)$ and $v \sim \mathcal{N}(0, \sigma_v)$, respectively, and $\sigma_w < 0$, $\sigma_v = 0$. If you consider the vector of w's to be its own multidimensional random variable, $\mathbf{w} \sim \mathcal{N}(0, S_w)$, where $S_w = \begin{bmatrix} \sigma_w & \sigma_w \\ \sigma_w & \sigma_w \end{bmatrix}$, the system may be represented as:

$$\dot{x} = Ax + Bu + w$$
$$\dot{y} = Cx + v$$

Note: Even though v is a scalar random variable, we will refer to the covariance of v as S_v later on. You may assume that $S_v \in \mathbb{R}^{(1,1)}$, and $S_v = \sigma_v$ for our example.

Build LQG

To construct an LQG system, we will need a Q and R for our cost function. We will use simple stand-ins.

$$R = 1$$
$$Q = q\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Here $q > 0$. Great. Now, we may begin building our LQG. We will Start by finding the cost matrix P using the continuous time infinite horizon Algebraic Riccati Equation (ARE):

$$\dot{P} = A^T P + PA - PBR^{-1}B^T P + Q = 0$$

It is presumed that $\dot{P}$ goes to 0 over time. The code cell below solves the ARE for P numerically. We may then use P to solve for our control gain, $K = R^{-1}B^T P$, such that $u = -Kx$. The code below also performs this step.

In [ ]:
```
import numpy as np
import scipy as sp

# Define matrices and variables
A = np.array([[1.0, 1.0], [0.0, 1.0]])
B = np.array([[0.0], [1.0]])
q = 2.25
Q = q*np.ones((2,2))
R = np.array([[1.0]])


# Solve the continuous time ARE for P
P = sp.linalg.solve_continuous_are(A, B, Q, R)
print("Steady state P:")
print(P)

# Solve for control gain K
K = B.T@P
print("Control Gain K: ", K[0])

# Check against analytic solution
```

```
K_a = (2 + np.sqrt(4+q)) * np.ones(2)
print("Analytically derived K: ", K_a)
```

```
Steady state P:
[[9.  4.5]
 [4.5 4.5]]
Control Gain K:  [4.5 4.5]
Analytically derived K:  [4.5 4.5]
```

Doyle somehow derived an analytic solution for the control gain (presumably using the time he saved for writing an abstract). He came up with:

$$K = (2 + \sqrt{4+q}) \begin{bmatrix} 1 & 1 \end{bmatrix}$$

The above code computes and displays this analytic solution for comparison to our numerical soluton (given same q). We can confirm that we are on the right track.

Next, we will do basically the same thing, but on the estimator side of our LQR. We will solve an ARE for the steady state Σ, and then solve for our Kallman filter gain, $L = \Sigma C^T S_v^{-1}$. The ARE in this case is

$$\dot{\Sigma} = A\Sigma + \Sigma A^T - \Sigma C^T S_v^{-1} C\Sigma + S_w = 0$$

Doyle's Analytic solution for the Kallman filter gain was

$$L = (2 + \sqrt{4+\sigma_w}) \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

The code below solves this second ARE for Σ, calculates L numerically, and then displays Doyle's analytic solution for confirmation.

In [ ]:
```
C = np.array([1, 0])
sig_w = 5
S_w = sig_w * np.ones((2,2))
S_v = np.array([1])
# R_k = r_k * np.ones((2,2))
# Cov_p = sigma_p * np.ones((2,2))

# Solve the continuous time ARE for
Sigma = sp.linalg.solve_continuous_are(A.T, C.reshape(2,1), S_w, S_v )
print("Steady state Sigma:")
print(Sigma)

# Solve for Kallman gain L
L = Sigma @ C.reshape(2,1) @ S_v
print("Kallman Gain, L: ", L)

# Check against analytic solution
L_a = (2 + np.sqrt(4+sig_w)) * np.ones(2)
print("Analytically derived L: ", L_a)
```

```
Steady state Sigma:
[[ 5.  5.]
 [ 5. 10.]]
Kallman Gain, L:  [5. 5.]
Analytically derived L:  [5. 5.]
```

Let us simplify our control and Kallman gains further for the sake of coming notation, and say

$$K = f\begin{bmatrix}1 & 1\end{bmatrix} \qquad \text{and} \qquad L = d\begin{bmatrix}1\\1\end{bmatrix}$$

Putting all of this together, our full system dynamics become:

$$\begin{bmatrix} A & BK \\ LC & A - LC - BK \end{bmatrix}\begin{bmatrix} x \\ \hat{x} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix}$$

Here $\hat{x}$ is our extimate of x from observation. This system is comes from these a synthesis of the following equations:

$$\begin{aligned} \dot{x} &= Ax + Bu & u &= -K\hat{x} \\ \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) & y &= Cx \\ & & \hat{y} &= C\hat{x} \end{aligned}$$

The last step in constructing this LQG system is to say that the closed loop controller has a nominal scalar gain, m. That brings our full system matrix to:

$$\begin{bmatrix} A & mBK \\ LC & A - LC - BK \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & -mf & -mf \\ d & 1 & 1-d & 1 \\ d & 0 & -d-f & 1-f \end{bmatrix}$$

Exposing Instability

Recall that in dynamic systems, the primary requirment is that the eigenvalues of the system dynamics all be less than zero. With that in mind, consider the charactaristic equation of our system matrix.

$$\begin{aligned} 0 =& \lambda^4 + (d + f - 4)\lambda^3 + (df - 2(d+f) + b)\lambda^2 \\ &+ (d + f - 4 + 2df(m-1))\lambda + (1 + (1-m)df) \end{aligned}$$

Focus on the final two terms (the only terms containing m). Note that in order for all of the eigenvalues to be less than 0, these last two terms must be greater than 0.

$$1 + (1-m)df > 0 \qquad\qquad d + f - 4 + 2df(m-1) > 0$$
$$m < 1 + \frac{1}{df} \qquad\qquad m > \frac{4 - d - f}{2df} + 1$$

When d and f (or equivalently, q and σ_w) become sufficiently large, $m \gtrsim 1$ AND $m \lesssim 1$. Since, g and f cannot actually go to infinity, the system will technically be stable as long as it

exhibits a gain of exactly one. However, if it deviates slightly in either direction, it will become unstable! There is the rub: **An LQG system has no guarunteed stability margin. In other words, the linear (or linearized) system dynamics can become arbitrarily close to having positive eigenvalues - and thus becoming unstable.**

This is obviously not a desirable position to be in. Most system models involve at least some simplifications and/or approximations. If any one of them cause reality to deviate from the model, they could throw the system into unstable behavior. How to approach this problem?

Transition to Robust Control Theory

The potential fragility of LQG systems act as a natural invitation to explore the field of robust controls. In robust controls, we may learn to translate our system from state space into the frequency domain where we may map the gain and phase response of the system to perterbations. With this information we may learn how to trade some of LQG's optimality for improved stability.

Sources

1. J. Doyle, "Guaranteed margins for LQG regulators," in *IEEE Transactions on Automatic Control*, vol. 23, no. 4, pp. 756-757, August 1978, doi: 10.1109/TAC.1978.1101812.
2. Brunton, S. L., & Kutz, J. N. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108380690
3. OpenAI. (2024). ChatGPT (v4). Retrieved from https://www.openai.com