Table of Contents I

- Introduction to Luenberger observers
  - how to select the poles of the estimator
  - Self-assessment material

notes

- this is the table of contents of this document; each section corresponds to a specific part of the course

Introduction to Luenberger observers

notes

Contents map

developed content units	taxonomy levels
observer	u1, e1

prerequisite content units	taxonomy levels
feedback	u1, e1
continuous time LTI systems	u1, e1

notes

Main ILO of sub-module "Introduction to Luenberger observers"

Derive the error dynamics equation for state estimators and explain its significance for observer stability

Describe the meaning of the gain matrix L in Luenberger observers

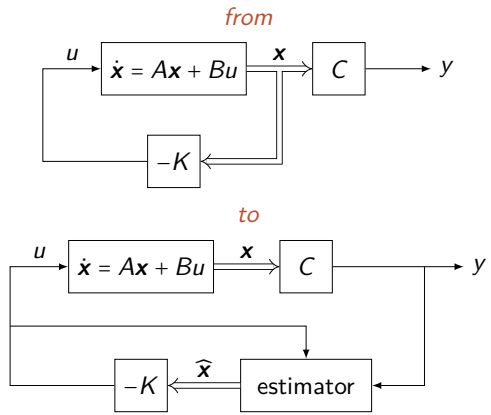
List rules of thumb to select estimator poles based on controller dynamics and system characteristics

Discuss the trade-offs between fast and slow observers in the presence of process and measurement noise

notes

- by the end of this module you shall be able to do this

Control-law design for full-state feedback

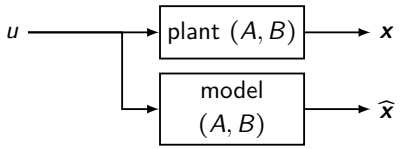


notes

- Let me walk you through this important transition in control design. On the left, we have our ideal scenario where we can measure all states directly - but in reality, this is often impossible or too expensive. On the right is what we'll achieve: estimating the states we can't measure. This is crucial because our full-state feedback controller needs all states to work properly.
- Think of this like driving a car - you can't directly measure everything (like engine temperature or tire friction), but you can estimate them from what you can observe (speed, RPM, etc.).

First idea for how to estimate $\boldsymbol{x}$: open loop estimator

$$\text{I know} \begin{cases} \boldsymbol{x}(0) \\ u(k) \\ A, B \end{cases}$$



$$\implies \text{I can simulate} \begin{cases} \widehat{\boldsymbol{x}}(0) = \boldsymbol{x}(0) \\ \dot{\widehat{\boldsymbol{x}}} = A\widehat{\boldsymbol{x}} + Bu \end{cases}$$

Dynamics:

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \qquad \widehat{\boldsymbol{x}}(t) = e^{At}\boldsymbol{x}(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau$$

though, is this strategy robust? *no! if there is uncertainty the estimation error*

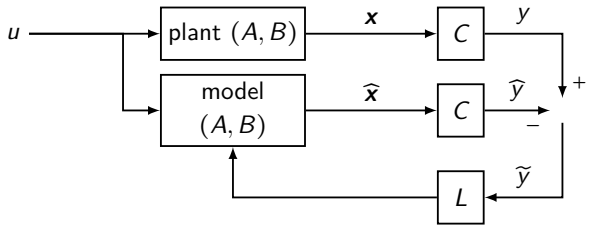
$\widetilde{\boldsymbol{x}}(t) := \boldsymbol{x}(t) - \widehat{\boldsymbol{x}}(t)$ *may diverge*

notes

- At first glance, this seems reasonable - if we know the initial state and the model is perfect, why not just simulate the system? But here's the catch: in the real world, nothing is perfect.
- Imagine you're trying to predict where a ball will land after you throw it. If your initial position measurement is slightly off, or if you don't account perfectly for wind resistance, your prediction will be wrong. And the longer you predict into the future, the worse it gets!
- That's exactly what happens here - any small error in initial conditions or model parameters makes our estimate drift away from reality over time. We need a better approach.

Estimator design

Idea: use feedback



$$\begin{cases} \widehat{x}(0) = x(0) \\ \dot{\widehat{x}} = A\widehat{x} + Bu + L(y - C\widehat{x}) \end{cases}$$

notes

- Here's our breakthrough idea: let's use the actual measurements we do have (y) to correct our estimates continuously. This is the power of feedback!
- The matrix L is like a set of "correction knobs" - it tells us how much we should adjust each state estimate based on the difference between what we measure and what our model predicts we should measure.
- Think of it like adjusting your shower temperature: you don't just set it once and hope for the best - you continuously feel the water and adjust based on the difference between what you feel and what you want.

What are the dynamics of the error $\widetilde{x} := x - \widehat{x}$?

$$\begin{cases} \dot{x} &= Ax + Bu \\ \dot{\widehat{x}} &= A\widehat{x} + Bu + L(y - C\widehat{x}) \end{cases}$$

$$\Downarrow$$

$$\dot{\widetilde{x}} = \dot{x} - \dot{\widehat{x}} = A(x - \widehat{x}) + Bu - Bu - L(y - C\widehat{x})$$

$$\Downarrow$$

$$\dot{\widetilde{x}} = (A - LC)\widetilde{x}$$

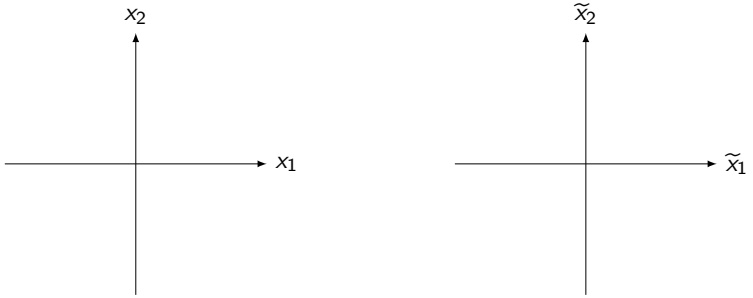
What decides the stability and speed of the dynamics of the error? *The eigenvalues of $A - LC$!*

notes

- This derivation shows something beautiful - the error dynamics are autonomous (don't depend on u) and linear! This means we can completely control how the estimation error behaves.
- The key insight? By choosing L properly, we can make A-LC have any eigenvalues we want (assuming observability), just like we did with state feedback!
- This is why we spent so much time on pole placement earlier - the same concepts apply here, but now we're placing poles for the estimator rather than the controller.

Dynamics of the error vs. dynamics of the state

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + Bu \qquad \dot{\tilde{\boldsymbol{x}}} = (A - LC)\,\tilde{\boldsymbol{x}}$$



notes

- Notice the parallel between these two diagrams. On the left, the actual system dynamics that we're trying to control. On the right, the error dynamics that we're designing through L.
- The crucial difference? The error dynamics don't have a control input u - they're entirely determined by A-LC. This makes our estimator design problem slightly different from our controller design problem.
- Visualize this: if the error dynamics are stable, all error trajectories will spiral into the origin - meaning our estimates will converge to the true states!

How can we design L?

For $(A, B, C, 0)$ fully observable & in observation canonical form:

$$A = \begin{bmatrix} -a_1 & 1 & 0 & \dots & 0 \\ -a_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & 1 \\ -a_n & 0 & 0 & \dots & 0 \end{bmatrix} \qquad B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}$$

$$\det(sI - A) = s^n + a_1 s^{n-1} + \dots + a_n$$

$$L = \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_n \end{bmatrix} \implies LC = \begin{bmatrix} L_1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ L_n & 0 & \cdots & 0 \end{bmatrix}$$

notes

- Here we're using the observer canonical form because it makes the pole placement problem particularly straightforward - notice how the characteristic polynomial coefficients appear explicitly in the first column of A.
- The structure of C (just [1 0 ... 0]) is crucial here - it means each element of L only affects one term in the characteristic polynomial of A-LC.
- This is similar to how the controller canonical form made pole placement easy for state feedback - there's a beautiful duality between these problems that we'll explore more later.

How can we design L?

$$A = \begin{bmatrix} -a_1 & 1 & 0 & \dots & 0 \\ -a_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & 1 \\ -a_n & 0 & 0 & \dots & 0 \end{bmatrix} \qquad LC = \begin{bmatrix} L_1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ L_n & 0 & \cdots & 0 \end{bmatrix}$$

$$\Downarrow$$

$$A - LC = \begin{bmatrix} -a_1 - L_1 & 1 & 0 & \dots & 0 \\ -a_2 - L_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & 1 \\ -a_n - L_n & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\Downarrow$$

$$\det\left(sI - (A - LC)\right) = s^n + (a_1 + L_1)\, s^{n-1} + \ldots + (a_n + L_n)$$

notes

- See how elegantly this works? Each L_i directly adjusts one coefficient of the characteristic polynomial. Want to change the s^{n-1} term? Just adjust L_1!
- This means pole placement for the observer is exactly as straightforward as it was for the controller in controller canonical form - we can directly match coefficients to place the poles wherever we want.
- Remember: the poles of A-LC determine how quickly our estimates converge to the true states. Faster poles mean quicker convergence, but as we'll see, there are tradeoffs.

Using `acker` for designing L

```
% controller design
K = acker( A, B, afPoles );

% observer design
L = ( acker( A', C', afPoles ) )';
```


notes

- Notice the beautiful symmetry here! Designing an observer is exactly like designing a controller, but for the dual system (A transposed, B replaced by C transposed).
- This duality is profound - it means all the intuition you developed for controller design carries over to observer design.
- In practice, we just use acker twice - once for K, once for L. The only trick is transposing the matrices for the observer case.

how to select the poles of the estimator

notes

Estimator poles selection – rules of thumb

- the observer should be $2 \to 6$ times faster than the controller
- the more the sensors are noisy, the slower the observer should be
- the slower the observer, the less resilient the controller is to disturbances

algorithms for designing L:

- dominant second order poles
- LQR

notes

- Here's the practical wisdom: make your observer faster than your controller, but not too fast. Why? Because the observer needs to settle before the controller acts.
- But there's a catch: if your measurements are noisy (and they always are to some degree), a super-fast observer will amplify that noise in the estimates. It's like overreacting to every little fluctuation in your shower temperature.
- The 2-6x rule is a good starting point, but real engineering requires tuning based on your specific system's needs and limitations.

Estimator poles selection – connections with estimation theory

$$\begin{cases} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + Bu \\ y &= C\boldsymbol{x} \end{cases} \quad \mapsto \quad \begin{cases} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + Bu + \boldsymbol{w} \\ y &= C\boldsymbol{x} + \nu \end{cases}$$

Estimator as before:

$$\dot{\widehat{\boldsymbol{x}}} = A\widehat{\boldsymbol{x}} + Bu + L\left(y - C\widehat{\boldsymbol{x}}\right)$$

New error dynamics:

$$\dot{\widetilde{\boldsymbol{x}}} = (A - LC)\,\widetilde{\boldsymbol{x}} + \boldsymbol{w} - L\nu$$

Trade-off:

- L big $\Longrightarrow$ effect of $\boldsymbol{w}$ is negligible but ν is amplified
- L small $\Longrightarrow$ effect of ν is negligible but $\boldsymbol{w}$ has bigger influences

optimal strategy in a statistical sense requires Kalman filtering (not in this course)

notes

- This is where theory meets reality. In practice, we always have process noise (w) and measurement noise (v). The choice of L is all about balancing these.
- Big L means we trust our measurements more, quickly correcting any discrepancies - great if measurements are accurate but bad if they're noisy.
- Small L means we trust our model more, smoothing out measurement noise but being slower to respond to actual changes.
- The Kalman filter (which you might see in advanced courses) provides the mathematically optimal balance, but for now, these intuitive guidelines will serve you well.
- Remember: control engineering is as much art as science - these tradeoffs require judgment calls based on your specific application.

Summarizing

Derive the error dynamics equation for state estimators and explain its significance for observer stability

Describe the meaning of the gain matrix L in Luenberger observers

List rules of thumb to select estimator poles based on controller dynamics and system characteristics

Discuss the trade-offs between fast and slow observers in the presence of process and measurement noise

notes

- in this module we saw this

Self-assessment material

notes

Question 1

Why is an open-loop estimator generally not considered a robust method for state estimation?

Potential answers:

I: **(wrong)** Because it can track the states accurately even with uncertainties.
II: **(wrong)** Because it uses feedback to correct errors in real time.
III: **(correct)** Because in the presence of model uncertainties or disturbances, the estimation error may diverge.
IV: **(wrong)** Because it relies on noisy measurements, which destabilize the estimation.
V: **(wrong)** I do not know

Solution 1:

An open-loop estimator only simulates the system based on initial conditions and the input. If there is any model uncertainty or disturbance, the estimated state $\widehat{\boldsymbol{x}}(t)$ will deviate from the true state $\boldsymbol{x}(t)$, causing the error $\widetilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \widehat{\boldsymbol{x}}(t)$

notes

- see the associated solution(s), if compiled with that ones :)

Question 2

What determines the stability and speed of convergence of the estimation error $\widetilde{\boldsymbol{x}} = \boldsymbol{x} - \widehat{\boldsymbol{x}}$ in a full-state observer?

Potential answers:

I: **(wrong)** The eigenvalues of matrix B.
II: **(wrong)** The input $u(t)$ and measurement noise.
III: **(wrong)** The initial state $\boldsymbol{x}(0)$.
IV: **(correct)** The eigenvalues of the matrix $A - LC$.
V: **(wrong)** I do not know

Solution 1:

The dynamics of the estimation error $\widetilde{\boldsymbol{x}}$ are governed by the differential equation $\dot{\widetilde{\boldsymbol{x}}} = (A - LC)\widetilde{\boldsymbol{x}}$. Therefore, the eigenvalues of $A - LC$ determine how the estimation error evolves over time, in particular whether it converges to zero and how fast.

notes

- see the associated solution(s), if compiled with that ones :)

Question 3

In the observer canonical form, how does the gain matrix L affect the characteristic polynomial of the observer error dynamics?

Potential answers:

I: **(wrong)** It multiplies the coefficients of the system matrix A.
II: **(correct)** It adds to the coefficients of the characteristic polynomial of A.
III: **(wrong)** It replaces the eigenvalues of A with the eigenvalues of B.
IV: **(wrong)** It subtracts from the output matrix C to reduce measurement noise.
V: **(wrong)** I do not know

Solution 1:

When the system is in observer canonical form, the matrix LC has a specific structure such that $A - LC$ modifies the coefficients of the characteristic polynomial of A. Specifically, if the original polynomial is $s^n + a_1 s^{n-1} + \cdots + a_n$, the

notes

- see the associated solution(s), if compiled with that ones :)

Question 4

Why is it generally recommended for the observer poles to be 2 to 6 times faster than the controller poles?

Potential answers:

I: **(wrong)** To ensure the controller has enough time to react to the observer.
II: **(correct)** To make the estimation error dynamics faster than the control system, enabling accurate feedback.
III: **(wrong)** To slow down the observer and reduce noise amplification.
IV: **(wrong)** To match the sampling rate of the digital controller.
V: **(wrong)** I do not know

Solution 1:

Placing the observer poles faster than the controller poles ensures that the estimation error converges quickly compared to the control dynamics. This allows the controller to act as if it had access to the true state, improving the overall performance of the closed-loop system. However, the poles should not be placed arbitrarily fast due to practical limitations like noise sensitivity.

notes

- see the associated solution(s), if compiled with that ones :)

Question 5

What is a trade-off involved when choosing faster poles for the observer?

Potential answers:

I: **(correct)** Faster poles increase sensitivity to measurement noise.
II: **(wrong)** Faster poles always improve estimation accuracy, regardless of noise.
III: **(wrong)** Slower poles make the observer more responsive.
IV: **(wrong)** Faster poles reduce computational complexity.
V: **(wrong)** I do not know

Solution 1:

While faster poles improve the speed of convergence of the observer, they also amplify measurement noise due to higher observer gains. This makes the observer sensitive to high-frequency disturbances, creating a trade-off between speed and robustness. Slower poles may be preferable in the presence of significant measurement noise.

notes

- see the associated solution(s), if compiled with that ones :)

Recap of sub-module "Introduction to Luenberger observers"

- one may estimate the states of a system by means of making the estimated state be so that it dynamically matches the measured values
- this strategy though is as valid as the model is, as a description of the system
- the situation is though in practice not as simple as seen here - indeed the here presented case is for "fully observable" systems (a concept that you'll see in systems theory) and thus not applicable all the times (but extensible to!)

notes

- the most important remarks from this sub-module are these ones