UPPSALA UNIVERSITET

Intro. Computer Control Systems: F10

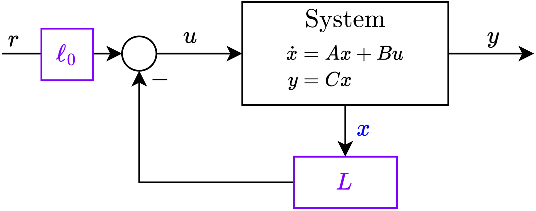
State observers

Fabio Bonassi

Dept. Information Technology, Div. Systems and Control

fabio.bonassi@it.uu.se

Recap: state-feedback control

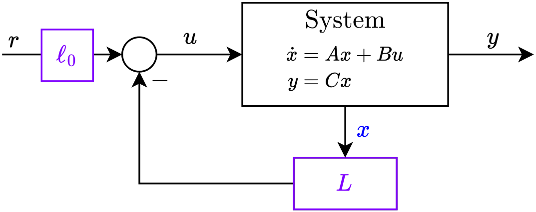


State-feedback control law: $u = -Lx + \ell_0 r$

- Closed-loop dynamics: $\dot{x} = \big(A - BL\big)x + B\ell_0 r$

fabio.bonassi@it.uu.se

Recap: state-feedback control

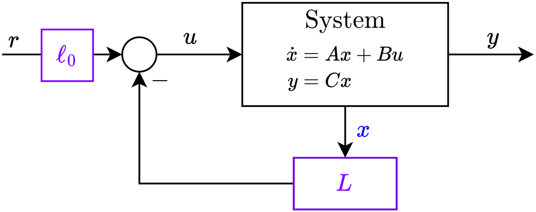


State-feedback control law: $u = -Lx + \ell_0 r$

- Closed-loop dynamics: $\dot{x} = \big(A - BL\big)x + B\ell_0 r$
- Task: design L so that $(A - BL)$ has the desired eigenvalues

fabio.bonassi@it.uu.se

Recap: state-feedback control



State-feedback control law: $u = -Lx + \ell_0 r$

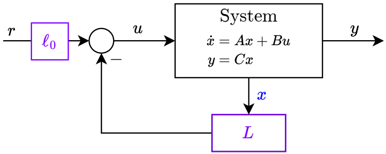
- Closed-loop dynamics: $\dot{x} = \big(A - BL\big)x + B\ell_0 r$
- Task: design L so that $(A - BL)$ has the desired eigenvalues

We can place the eigenvalues $\Leftrightarrow$ the system is controllable $\Leftrightarrow$ the controllability matrix $\mathcal{S}$ is non-singular

$$\mathcal{S} = \begin{bmatrix} B & AB & \dots & A^{n-1}B \end{bmatrix}$$

fabio.bonassi@it.uu.se

Recap: pole placement algorithm

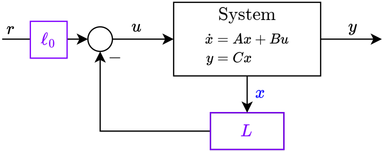


Once we are sure that the system is controllable, we use the **pole placement algorithm** to find L:

- Compute the characteristic poly. $\varphi(\lambda) = \det(sI - (A - BL))$

fabio.bonassi@it.uu.se

Recap: pole placement algorithm

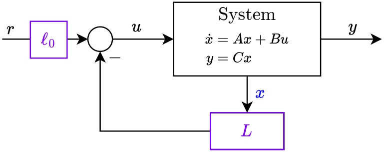


Once we are sure that the system is controllable, we use the **pole placement algorithm** to find L:

- Compute the characteristic poly. $\varphi(\lambda) = \det(sI - (A - BL))$
- Write the desired characteristic poly. $(\lambda - p_1) \cdot \ldots \cdot (\lambda - p_n)$

fabio.bonassi@it.uu.se

Recap: pole placement algorithm

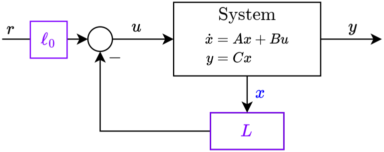


Once we are sure that the system is controllable, we use the **pole placement algorithm** to find L:

- Compute the characteristic poly. $\varphi(\lambda) = \det(sI - (A - BL))$
- Write the desired characteristic poly. $(\lambda - p_1) \cdot \ldots \cdot (\lambda - p_n)$
- Match the coefficients of the two polynomial

fabio.bonassi@it.uu.se

Recap: pole placement algorithm

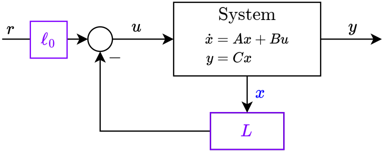


Once we are sure that the system is controllable, we use the **pole placement algorithm** to find L:

- Compute the characteristic poly. $\varphi(\lambda) = \det(sI - (A - BL))$
- Write the desired characteristic poly. $(\lambda - p_1) \cdot \ldots \cdot (\lambda - p_n)$
- Match the coefficients of the two polynomial

fabio.bonassi@it.uu.se

Recap: pole placement algorithm



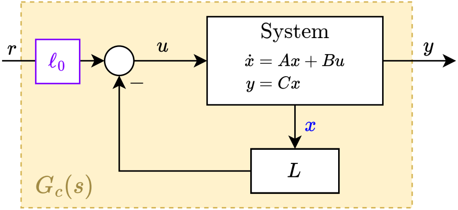
Once we are sure that the system is controllable, we use the **pole placement algorithm** to find L:

- Compute the characteristic poly. $\varphi(\lambda) = \det(sI - (A - BL))$
- Write the desired characteristic poly. $(\lambda - p_1) \cdot \ldots \cdot (\lambda - p_n)$
- Match the coefficients of the two polynomial

If the system is in **controllable canonical form**, this procedure is a lot easier!

fabio.bonassi@it.uu.se

Recap: Setting unitary closed-loop static gain



The gain ℓ_0 is design so that the static gain of the closed-loop transfer function $G_c(s)$ is 1

$$\boxed{\ell_0 = -\frac{1}{C(A - BL)^{-1}B}}$$

fabio.bonassi@it.uu.se

F9: Quiz! `www.menti.com` code 3587 8302

- How to design a state-feedback controller

fabio.bonassi@it.uu.se

Today's lecture

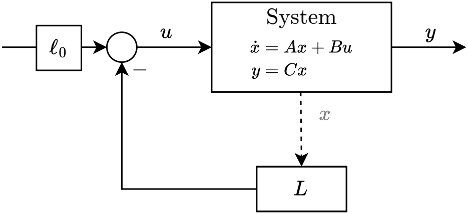
- Can we use state-feedback control when we can't measure x?
- State observer design with pole placement
- State-feedback control using state estimates $\rightarrow$ general linear feedback

fabio.bonassi@it.uu.se

Introduction to state observers

fabio.bonassi@it.uu.se

Why do we need a state observer?



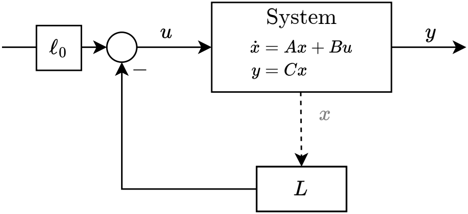
When we designed the state feedback law

$$u = -Lx + \ell_0 r$$

we assumed to known/measure the state x.

fabio.bonassi@it.uu.se

Why do we need a state observer?



When we designed the state feedback law

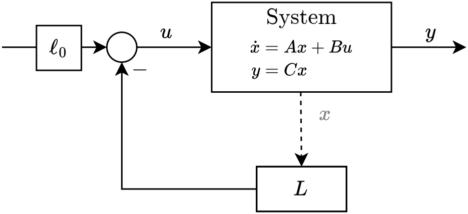
$$u = -Lx + \ell_0 r$$

we assumed to known/measure the state x.

This is (almost) never the case! Some state may require an expensive sensor.

fabio.bonassi@it.uu.se

Why do we need a state observer?



When we designed the state feedback law

$$u = -Lx + \ell_0 r$$

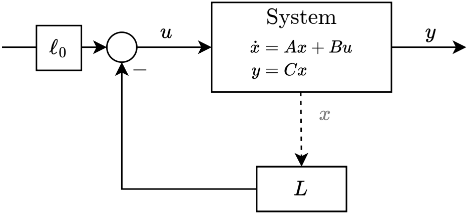
we assumed to known/measure the state x.

This is (almost) never the case! Some state may require an expensive sensor. Examples:

- *Inverted pendulum* – we only measure angles (via encoders)

fabio.bonassi@it.uu.se

Why do we need a state observer?



When we designed the state feedback law

$$u = -Lx + \ell_0 r$$

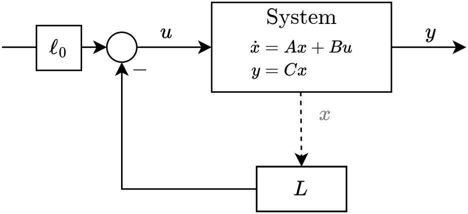
we assumed to known/measure the state x.

This is (almost) never the case! Some state may require an expensive sensor. Examples:

- *Inverted pendulum* – we only measure angles (via encoders)
- *Cars* – the speed is estimated from the wheels' angular velocity

fabio.bonassi@it.uu.se

Why do we need a state observer?



When we designed the state feedback law

$$u = -Lx + \ell_0 r$$

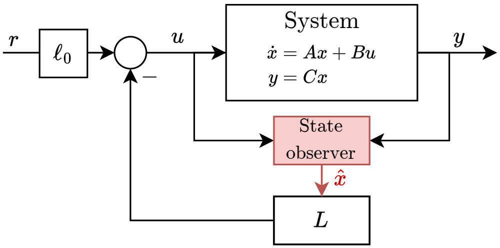
we assumed to known/measure the state x.

This is (almost) never the case! Some state may require an expensive sensor. Examples:

- *Inverted pendulum* – we only measure angles (via encoders)
- *Cars* – the speed is estimated from the wheels' angular velocity
- *LEGO NXT* – States estimated using distance from centerline

UPPSALA UNIVERSITET

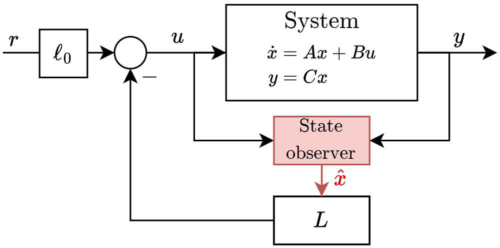
The separation principle



Idea: Build a **state observer** yielding a state estimate $\widehat{x}$.

fabio.bonassi@it.uu.se

The separation principle



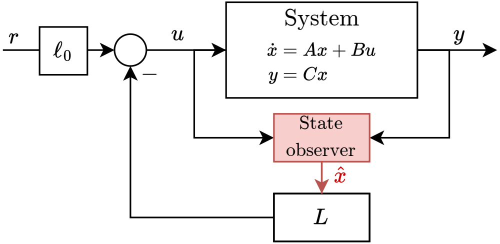
Idea: Build a **state observer** yielding a state estimate $\widehat{x}$.

Separation principle

Since the system is linear, we can

fabio.bonassi@it.uu.se

The separation principle



Idea: Build a **state observer** yielding a state estimate $\widehat{x}$.

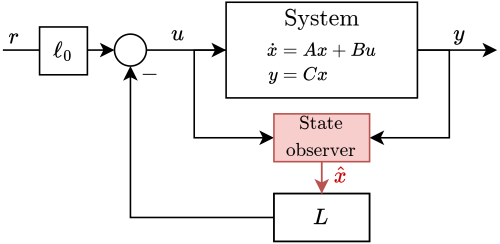
Separation principle

Since the system is linear, we can

- Design the state-feedback $u = -Lx + \ell_0 r$ as if we knew x

fabio.bonassi@it.uu.se

The separation principle



Idea: Build a **state observer** yielding a state estimate $\widehat{x}$.

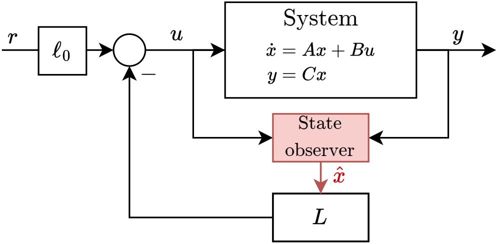
Separation principle

Since the system is linear, we can

- Design the state-feedback $u = -Lx + \ell_0 r$ as if we knew x
- Design a state observer yielding an estimate $\widehat{x}(t) \xrightarrow{t\to\infty} x(t)$

UPPSALA UNIVERSITET

The separation principle



Idea: Build a **state observer** yielding a state estimate $\widehat{x}$.

Separation principle

Since the system is linear, we can

- Design the state-feedback $u = -Lx + \ell_0 r$ as if we knew x
- Design a state observer yielding an estimate $\widehat{x}(t) \xrightarrow{t\to\infty} x(t)$
- Use such a state estimate in the controller $u = -L\widehat{x} + \ell_0 r$

fabio.bonassi@it.uu.se

The separation principle

Separation principle

Since the system is linear, we can

- Design the state-feedback $u = -Lx + \ell_0 r$ as if we knew x
- Design a state observer yielding an estimate $\widehat{x}(t) \xrightarrow{t\to\infty} x(t)$
- Use such a state estimate in the controller $u = -L\widehat{x} + \ell_0 r$

If these two component work well on their own, their combination will still work well.

fabio.bonassi@it.uu.se

The separation principle

Separation principle

Since the system is linear, we can

- Design the state-feedback $u = -Lx + \ell_0 r$ as if we knew x
- Design a state observer yielding an estimate $\widehat{x}(t) \xrightarrow{t \to \infty} x(t)$
- Use such a state estimate in the controller $u = -L\widehat{x} + \ell_0 r$

If these two component work well on their own, their combination will still work well.

In other words, we just have to find a suitable state observer.

fabio.bonassi@it.uu.se

First (naïve) approaches to observer design

Naïve idea #1

Inverting the output transformation

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *Since we know that $y = Cx$, can't we invert this relationship?*

fabio.bonassi@it.uu.se

Naïve idea #1

Inverting the output transformation

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *Since we know that $y = Cx$, can't we invert this relationship?*
No!

fabio.bonassi@it.uu.se

Naïve idea #1

Inverting the output transformation

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *Since we know that $y = Cx$, can't we invert this relationship?*
No!

- If $n > 1$ the output transformation $y = Cx$ is not invertible: infinite number of x:s corresponding to the same y

fabio.bonassi@it.uu.se

Naïve idea #1

Inverting the output transformation

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *Since we know that $y = Cx$, can't we invert this relationship?*

No!

- If $n > 1$ the output transformation $y = Cx$ is not invertible: infinite number of x:s corresponding to the same y
- Even if $n = 1$, $x = \frac{y}{C}$ would be a bad estimate (noise)

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Naïve idea #2

Open-loop state observer

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *We know the equations of the system and the input we applied...can't we just estimate* $\widehat{x}$ *by solving*

$$\widehat{x}(t) = \underbrace{\widehat{x}_0 e^{At}}_{\text{Depends upon } \textbf{unknown } \widehat{x}_0} + \underbrace{\int_0^t e^{A\tau} Bu(t-\tau)\, d\tau}_{\text{Forced motion (known)}} \quad ?$$

fabio.bonassi@it.uu.se

Naïve idea #2

Open-loop state observer

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *We know the equations of the system and the input we applied...can't we just estimate* $\hat{x}$ *by solving*

$$\hat{x}(t) = \underbrace{\hat{x}_0 e^{At}}_{\substack{\text{Depends upon} \\ \textbf{unknown } \hat{x}_0}} + \underbrace{\int_0^t e^{A\tau} Bu(t-\tau)d\tau}_{\substack{\text{Forced motion} \\ \text{(known)}}} \quad ?$$

Obs: We call this observer "open-loop" because we just simulate the system without any correction term (feedback) based the measured output y

Naïve idea #2

Open-loop state observer

$$\text{System}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Q: *We know the equations of the system and the input we applied...can't we just estimate* $\hat{x}$ *by solving*

$$\hat{x}(t) = \underbrace{\hat{x}_0 e^{At}}_{\substack{\text{Depends upon} \\ \textbf{unknown } \hat{x}_0}} + \underbrace{\int_0^t e^{A\tau} Bu(t-\tau)\, d\tau}_{\substack{\text{Forced motion} \\ \text{(known)}}} \quad ?$$

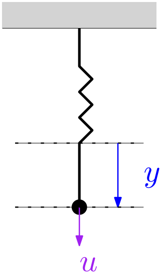
We can guess the initial state $\hat{x}_0$, but we don't know it exactly!

Obs: We call this observer "open-loop" because we just simulate the system without any correction term (feedback) based the measured output y

Build intuition from simple systems

Open-loop state observer

We consider the following mechanical system (spring + damper)



State-space form:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -k/m & -\mu/m \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u(t), \quad x(0) = x_0$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

fabio.bonassi@it.uu.se

Build intuition from simple systems

Open-loop state observer

- We switch off the input $(u(t) = 0)$

fabio.bonassi@it.uu.se

Build intuition from simple systems

Open-loop state observer

- We switch off the input ($u(t) = 0$)
- The initial conditions of the real system are $x_0 = [0, 1]^T$

fabio.bonassi@it.uu.se

Build intuition from simple systems

Open-loop state observer

- We switch off the input ($u(t) = 0$)
- The initial conditions of the real system are $x_0 = [0, 1]^T$
- The initial guess for our open-loop observer is $\hat{x}_0 = [0, 3]^T$

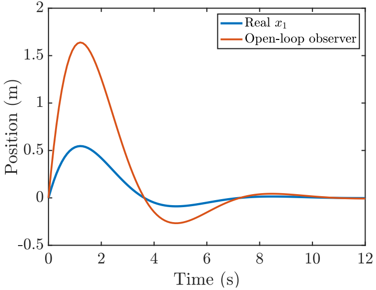
fabio.bonassi@it.uu.se

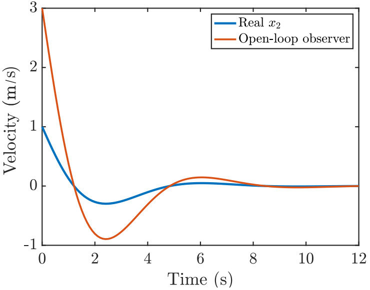
UPPSALA UNIVERSITET

Build intuition from simple systems

Open-loop state observer

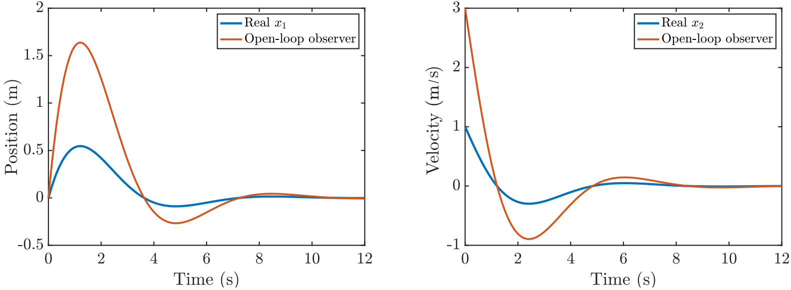
- We switch off the input ($u(t) = 0$)
- The initial conditions of the real system are $x_0 = [0, 1]^T$
- The initial guess for our open-loop observer is $\hat{x}_0 = [0, 3]^T$





Build intuition from simple systems
Open-loop state observer

- We switch off the input ($u(t) = 0$)
- The initial conditions of the real system are $x_0 = [0, 1]^T$
- The initial guess for our open-loop observer is $\hat{x}_0 = [0, 3]^T$



For asymptotically stable systems the open-loop observers eventually converges (when the free motion dies out) $\implies$ **slow!**

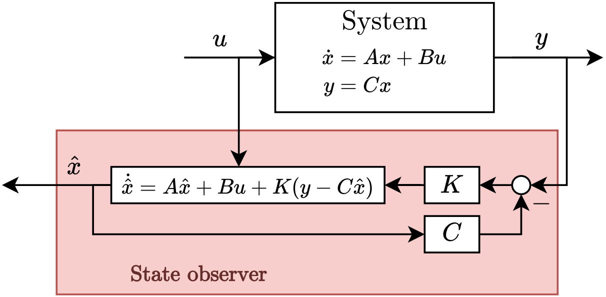
fabio.bonassi@it.uu.se

Dynamic state observers for state estimation

fabio.bonassi@it.uu.se

Dynamical state observer

Structure

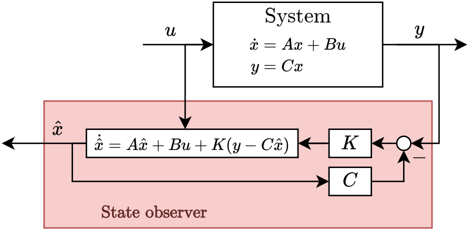


Idea: use the output prediction error $y - C\widehat{x}$ (a.k.a. **innovation**) to correct the state estimate

fabio.bonassi@it.uu.se

Dynamical state observer

Structure



Idea: use the output prediction error $y - C\hat{x}$ (a.k.a. **innovation**) to correct the state estimate

Dynamical state observer

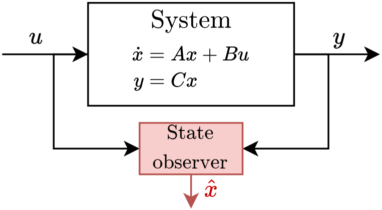
$$\dot{\hat{x}} = A\hat{x} + Bu + K(y - C\hat{x}) \qquad \text{where } K = \begin{bmatrix} \kappa_1 \\ \vdots \\ \kappa_n \end{bmatrix}$$

the **innovation gain** $K \in \mathbb{R}^{n \times 1}$ needs to be designed properly

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain

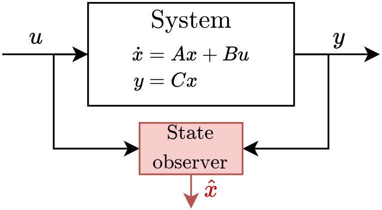


To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain



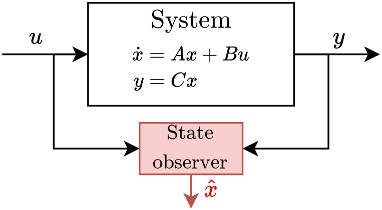
To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

$$\dot{\varepsilon} = \dot{x} - \dot{\widehat{x}}$$

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain

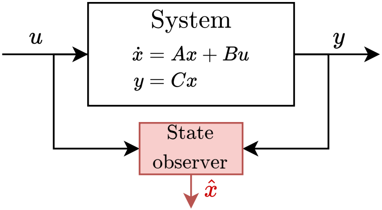


To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \hat{x}$ evolves

$$\dot{\varepsilon} = \dot{x} - \dot{\hat{x}} = \underbrace{Ax + Bu}_{\dot{x}}$$

UPPSALA UNIVERSITET

Dynamical state observer
Design of the innovation gain



To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

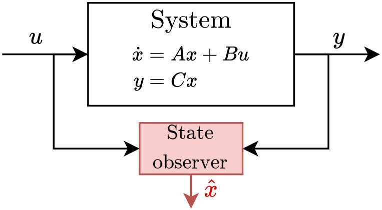
$$\dot{\varepsilon} = \dot{x} - \dot{\widehat{x}} = \underbrace{Ax + Bu}_{\dot{x}} - \underbrace{\big(A\widehat{x} + Bu + K(Cx - C\widehat{x})\big)}_{\dot{\widehat{x}}}$$

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Dynamical state observer

Design of the innovation gain



To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

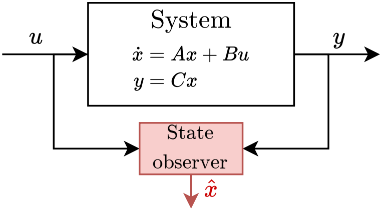
$$\dot{\varepsilon} = \dot{x} - \dot{\widehat{x}} = \underbrace{Ax + Bu}_{\dot{x}} - \underbrace{\big(A\widehat{x} + Bu + K(Cx - C\widehat{x})\big)}_{\dot{\widehat{x}}}$$

$$= (A - KC)x - (A - KC)\widehat{x}$$

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain



To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

$$\dot{\varepsilon} = \dot{x} - \dot{\widehat{x}} = \underbrace{Ax + Bu}_{\dot{x}} - \underbrace{\left(A\widehat{x} + Bu + K(Cx - C\widehat{x})\right)}_{\dot{\widehat{x}}}$$

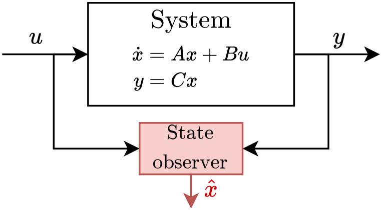
$$= (A - KC)x - (A - KC)\widehat{x} = (A - KC)(x - \widehat{x})$$

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Dynamical state observer

Design of the innovation gain



To understand how to design the observer, let's see how the state estimation error $\varepsilon \triangleq x - \widehat{x}$ evolves

$$\dot{\varepsilon} = \dot{x} - \dot{\widehat{x}} = \underbrace{Ax + Bu}_{\dot{x}} - \underbrace{\big(A\widehat{x} + Bu + K(Cx - C\widehat{x})\big)}_{\dot{\widehat{x}}}$$

$$= (A - KC)x - (A - KC)\widehat{x} = (A - KC)(x - \widehat{x})$$

The state estimation error evolves according to

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain $\leftrightarrow$ pole placement problem

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain $\leftrightarrow$ pole placement problem

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

- $\varepsilon \to 0$ iff $(A - KC)$ has **asymptotically stable eigenvalues**

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain $\leftrightarrow$ pole placement problem

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

- $\varepsilon \to 0$ iff $(A - KC)$ has **asymptotically stable eigenvalues**
- The eigenvalues of $(A - KC)$ depend on K, we can place them where we want

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain $\leftrightarrow$ pole placement problem

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

- $\varepsilon \to 0$ iff $(A - KC)$ has **asymptotically stable eigenvalues**
- The eigenvalues of $(A - KC)$ depend on K, we can place them where we want
- **Pole placement problem that we know how to solve! :)**

fabio.bonassi@it.uu.se

Dynamical state observer

Design of the innovation gain ↔ pole placement problem

$$\boxed{\dot{\varepsilon} = (A - KC)\,\varepsilon}$$

- $\varepsilon \to 0$ iff $(A - KC)$ has **asymptotically stable eigenvalues**
- The eigenvalues of $(A - KC)$ depend on K, we can place them where we want
- **Pole placement problem that we know how to solve! :)**

Pole placement algorithm for observer design

Given the desired eigenvalues $p_1, p_2, ..., p_n$, the innovation gain

$$K = \begin{bmatrix} \kappa_1 & ... & \kappa_n \end{bmatrix}^T$$

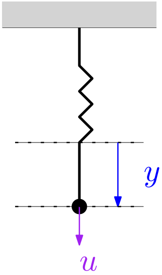
is computed by setting the characteristic polynomial of $(A - KC)$ equal to the *desired characteristic polynomial*

$$\underbrace{\det\left(\lambda I - (A - KC)\right)}_{\varphi(\lambda)} = \underbrace{(\lambda - p_1)\cdot(\lambda - p_2)\cdot ... \cdot(\lambda - p_n)}_{\text{desired characteristic polynomial}}$$

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Example 1 – spring + damper



State-space form:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\mu}{m} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)$$

For simplicity we will consider $k = m = \mu = 1$.

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Computing the innovation gain K

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Task – Set the state estimation error's eigenvalues to $p_1 = p_2 = -10$.

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Computing the innovation gain K

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Task – Set the state estimation error's eigenvalues to $p_1 = p_2 = -10$.

`Step 1.` Find the characteristic polynomial

$$\begin{aligned}
\varphi(\lambda) &= \det(\lambda I - (A - KC)) \\
&= \det\left(\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} - \left(\begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix}\begin{bmatrix} 1 & 0 \end{bmatrix}\right)\right) \\
&= \det\left(\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} - \left(\begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} - \begin{bmatrix} \kappa_1 & 0 \\ \kappa_2 & 0 \end{bmatrix}\right)\right) \\
&= \det\begin{bmatrix} \lambda + \kappa_1 & -1 \\ 1 + \kappa_2 & \lambda + 1 \end{bmatrix} = \lambda^2 + (\kappa_1 + 1)\lambda + (\kappa_1 + \kappa_2 + 1)
\end{aligned}$$

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Computing the innovation gain K

`Step 2.` Compute the desired characteristic polynomial

$$(\lambda+10)(\lambda+10) \quad \rightarrow \quad \lambda^2 + 20\lambda + 100$$

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Computing the innovation gain K

Step 2. Compute the desired characteristic polynomial

$$(\lambda+10)(\lambda+10) \quad \rightarrow \quad \lambda^2 + 20\lambda + 100$$

Step 3. Set the two polynomial equal

$$\lambda^2 + (\kappa_1 + 1)\lambda + (\kappa_1 + \kappa_2 + 1) = \lambda^2 + 20\lambda + 100$$

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Computing the innovation gain K

`Step 2.` Compute the desired characteristic polynomial

$$(\lambda+10)(\lambda+10) \quad \rightarrow \quad \lambda^2 + 20\lambda + 100$$

`Step 3.` Set the two polynomial equal

$$\lambda^2 + (\kappa_1 + 1)\lambda + (\kappa_1 + \kappa_2 + 1) = \lambda^2 + 20\lambda + 100$$

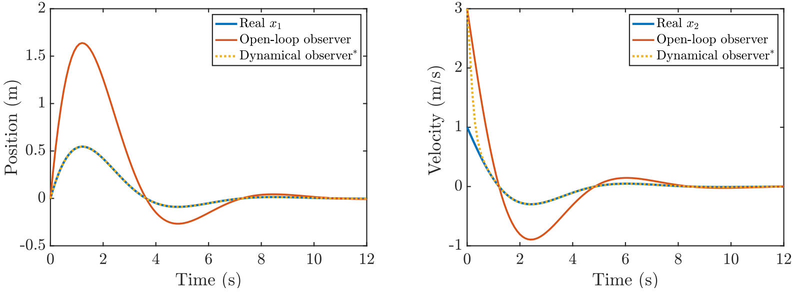
The K we are looking for is

$$\begin{cases} \kappa_1 + 1 = 20 \\ \kappa_1 + \kappa_2 + 1 = 100 \end{cases} \quad \Leftrightarrow \quad K = \begin{bmatrix} 19 \\ 80 \end{bmatrix}$$

fabio.bonassi@it.uu.se

Example 1 – spring + damper

Simulation of the resulting state observer



Real system with $x_0 = [0, 1]^T$

Open-loop observer $\dot{\hat{x}} = A\hat{x} + Bu$ with $\hat{x}_0 = [0, 3]^T$

$\boxed{\text{Dynamical observer } \dot{\hat{x}} = A\hat{x} + Bu + K(y - C\hat{x}) \text{ with } \hat{x}_0 = [0, 3]^T}$

fabio.bonassi@it.uu.se

Tips for the choice of the observer poles

- Make them (≈ 1 decade) faster than the closed-loop poles

fabio.bonassi@it.uu.se

Tips for the choice of the observer poles

- Make them (≈ 1 decade) faster than the closed-loop poles
- Not too fast: the faster we want the observer to be, the bigger $K \rightarrow$ the noise on the output gets amplified!

fabio.bonassi@it.uu.se

Conditions for state estimation

fabio.bonassi@it.uu.se

Observability of state-space systems

Observability & Observability matrix

Definition: Observability matrix

We define the **observability matrix** $\mathcal{O} \in \mathbb{R}^{n \times n}$ as

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

fabio.bonassi@it.uu.se

Observability of state-space systems

Observability & Observability matrix

Definition: Observability matrix

We define the **observability matrix** $\mathcal{O} \in \mathbb{R}^{n \times n}$ as

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Definition: Observable system (Def. 8.4 G&L)

A system is said **observable** if there exists no initial state x^* for which the resulting output signal y is identically zero.

fabio.bonassi@it.uu.se

Observability of state-space systems

Observability & Observability matrix

Definition: Observability matrix

We define the **observability matrix** $\mathcal{O} \in \mathbb{R}^{n \times n}$ as

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Definition: Observable system (Def. 8.4 G&L)

A system is said **observable** if there exists no initial state x^* for which the resulting output signal y is identically zero[1].

[1] Informally, this means that states always have some effect on output y

fabio.bonassi@it.uu.se

Observability of state-space systems

A condition for the design of state observers

Result 8.9 G&L

- $\det(\mathcal{O}) \neq 0 \Leftrightarrow$ the system is *observable*
- system is observable $\Leftrightarrow$ admits *observable canonical form*

Note that $\det(\mathcal{O}) \neq 0$ means $\text{rank}(\mathcal{O}) = n$.

Result 9.2 G&L

The system is observable $\Leftrightarrow$ the **state observer's pole placement** design problem can be solved for **arbitrary real and complex-conjugate eigenvalues**

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Example 2

A non-observable system

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0$$

fabio.bonassi@it.uu.se

Example 2
A non-observable system

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0$$

To check the observability, we need to compute the observability matrix

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \quad \rightarrow \quad \text{Singular!}$$

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Example 2
A non-observable system

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0$$

To check the observability, we need to compute the observability matrix

$$\mathcal{O} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} \quad \rightarrow \quad \text{Singular!}$$

The system is not observable. *Where is this coming from?*

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Example 2

Investigating the non-observability

As we did in the previous lecture, let's compute the transfer function $G(s)$ corresponding to this LTI

fabio.bonassi@it.uu.se

Example 2

Investigating the non-observability

As we did in the previous lecture, let's compute the transfer function $G(s)$ corresponding to this LTI

$$
\begin{aligned}
G(s) = C(sI - A)^{-1}B + D &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s+1 & 0 \\ 0 & s-1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
&= \frac{1}{(s+1)(s-1)} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s-1 & 0 \\ 0 & s+1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
&= \frac{1}{(s+1)(s-1)} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s-1 \\ s+1 \end{bmatrix} = \frac{\cancel{(s-1)}}{\cancel{(s-1)}(s+1)}
\end{aligned}
$$

fabio.bonassi@it.uu.se

Example 2

Investigating the non-observability

As we did in the previous lecture, let's compute the transfer function $G(s)$ corresponding to this LTI

$$
\begin{aligned}
G(s) = C(sI - A)^{-1}B + D &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s+1 & 0 \\ 0 & s-1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
&= \frac{1}{(s+1)(s-1)} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s-1 & 0 \\ 0 & s+1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
&= \frac{1}{(s+1)(s-1)} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s-1 \\ s+1 \end{bmatrix} = \frac{\cancel{(s-1)}}{\cancel{(s-1)}(s+1)}
\end{aligned}
$$

Pole/zero cancellation!

fabio.bonassi@it.uu.se

About poles/zeros cancellations

Poles/zeros cancellations can cause

- Loss of controllability $\rightarrow$ cancelled poles are an uncontrollable part of the system

fabio.bonassi@it.uu.se

About poles/zeros cancellations

Poles/zeros cancellations can cause

- Loss of controllability $\rightarrow$ cancelled poles are an uncontrollable part of the system
- Loss of observability $\rightarrow$ cancelled poles are an unobservable part of the system

fabio.bonassi@it.uu.se

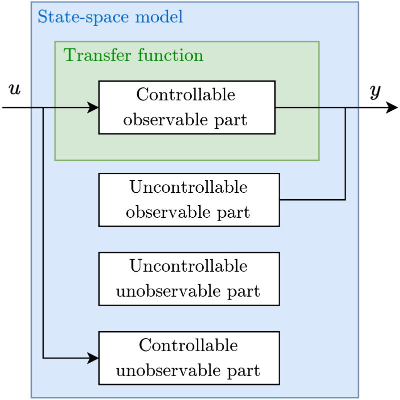
About poles/zeros cancellations

Poles/zeros cancellations can cause

- Loss of controllability → cancelled poles are an uncontrollable part of the system
- Loss of observability → cancelled poles are an unobservable part of the system
- Both → cancelled poles are an uncontrollable & unobservable part of the system

fabio.bonassi@it.uu.se

Kalman decomposition

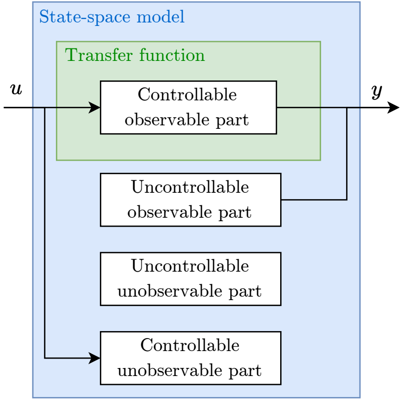


$G(s)$ only represents the controllable & observable part of an LTI

fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Kalman decomposition

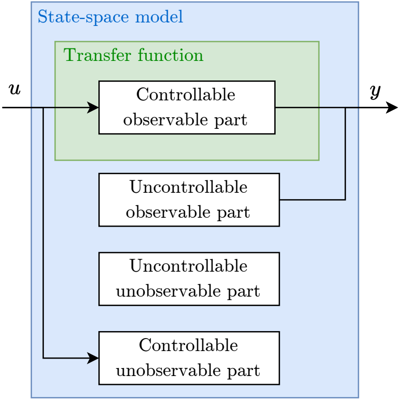


$G(s)$ only represents the controllable & observable part of an LTI

- This is why state-space is more informative!

fabio.bonassi@it.uu.se

Kalman decomposition



$G(s)$ only represents the controllable & observable part of an LTI

- This is why state-space is more informative!
- Acceptable if the "missing" part are asymptotically stable

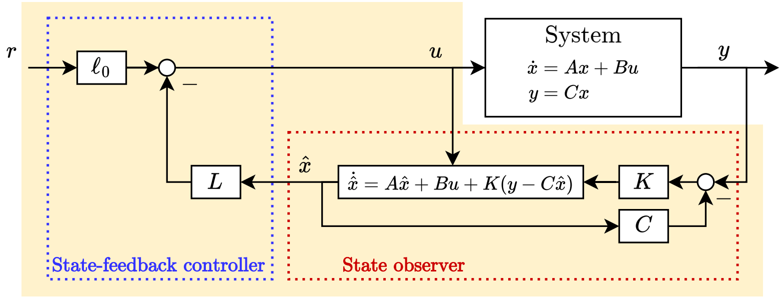
fabio.bonassi@it.uu.se

UPPSALA UNIVERSITET

Connecting back to general linear feedback

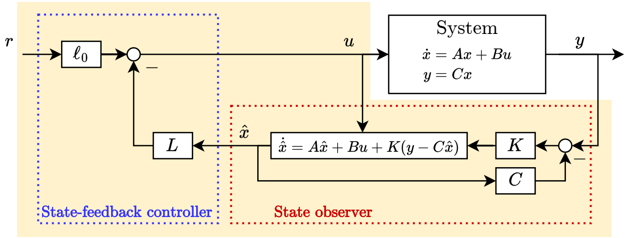
fabio.bonassi@it.uu.se

State-feedback with estimated states → general linear feedback



fabio.bonassi@it.uu.se

State-feedback with estimated states → general linear feedback



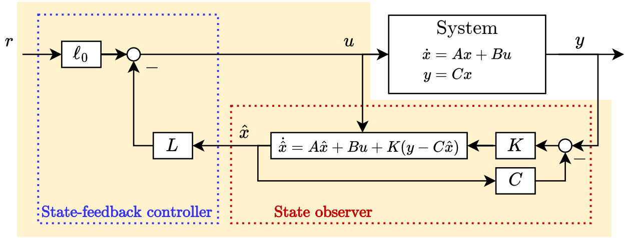
General linear feedback form (Ch. 9.5 G&L)

The state-feedback controller based on the estimated state is a **general linear feedback controller**

$$U(s) = F_r(s)R(s) - F_y(s)Y(s)$$

fabio.bonassi@it.uu.se

State-feedback with estimated states → general linear feedback



General linear feedback form (Ch. 9.5 G&L)

The state-feedback controller based on the estimated state is a **general linear feedback controller**

$$U(s) = F_r(s)R(s) - F_y(s)Y(s)$$

where

$$F_r(s) = \left[1 - L(sI - A + KC + BL)^{-1}B\right]\ell_0$$

$$F_y(s) = L(sI - A + KC + BL)^{-1}K$$

fabio.bonassi@it.uu.se

UPPSALA
UNIVERSITET

Conclusions

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate
- We can use the pole placement algorithm also for the design of the innovation gain K

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate
- We can use the pole placement algorithm also for the design of the innovation gain K
- The combo "state-feedback control" + "state observer"

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate
- We can use the pole placement algorithm also for the design of the innovation gain K
- The combo "state-feedback control" $+$ "state observer"
  - Works well if these two components work well individually

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate
- We can use the pole placement algorithm also for the design of the innovation gain K
- The combo "state-feedback control" + "state observer"
  - Works well if these two components work well individually
  - We can design L and K separately

fabio.bonassi@it.uu.se

Conclusions

- If we can't measure the state we need a state observer
- A good observer uses the innovation to correct its state estimate
- We can use the pole placement algorithm also for the design of the innovation gain K
- The combo "state-feedback control" + "state observer"
  - Works well if these two components work well individually
  - We can design L and K separately
  - is a great way to design a *general linear feedback* controller!

fabio.bonassi@it.uu.se

Useful resources

Useful videos

- Motivations for full-state estimation – control bootcamp
- Observability – control bootcamp
- Full-state estimation – control bootcamp
- Observability example (inverted pendulum)

fabio.bonassi@it.uu.se