Review of Automatic Control

State space models

Per Mattsson

per.mattsson@hig.se

Introduction

- We have seen how to represent a linear system with a transfer function $G(s)$

$$Y(s) = G(s)U(s)$$

Introduction

- We have seen how to represent a linear system with a transfer function $G(s)$

$$Y(s) = G(s)U(s)$$

$$\xrightarrow{u(t)} \boxed{G} \xrightarrow{y(t)}$$

- This form is especially useful when we study how the system reacts to different frequencies in the input.

per.mattsson@hig.se

Introduction

- We have seen how to represent a linear system with a transfer function $G(s)$

$$Y(s) = G(s)U(s)$$

- This form is especially useful when we study how the system reacts to different frequencies in the input.
- Another popular type of linear models are the state space models.

per.mattsson@hig.se

State space models

- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

per.mattsson@hig.se

State space models

- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

per.mattsson@hig.se

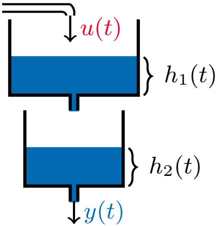
State space models

- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

Example: Level-system



per.mattsson@hig.se

State space models

- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

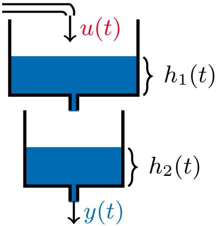
The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

Example: Level-system

- All past inputs: $x(t) = \{u(\tau), \tau < t\}$

  (infinite amount of information).



per.mattsson@hig.se

State space models

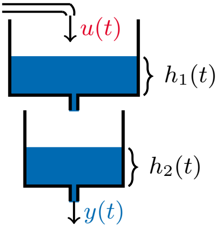
- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

Example: Level-system

- All past inputs: $x(t) = \{u(\tau), \tau < t\}$
  (infinite amount of information).
- Water level: $x(t) = \begin{bmatrix} h_1(t) \\ h_2(t) \end{bmatrix}$.



per.mattsson@hig.se

State space models

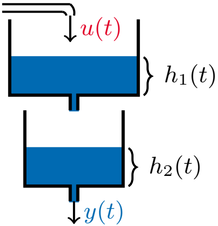
- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

Example: Level-system

- All past inputs: $x(t) = \{u(\tau), \tau < t\}$ (infinite amount of information).
- Water level: $x(t) = \begin{bmatrix} h_1(t) \\ h_2(t) \end{bmatrix}$.
- Volume: $x(t) = \begin{bmatrix} V_1(t) \\ V_2(t) \end{bmatrix} = \begin{bmatrix} A_1 h_1(t) \\ A_2 h_2(t) \end{bmatrix}$.



per.mattsson@hig.se

State space models

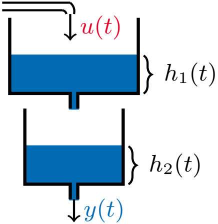
- In a dynamical system, the output $y(t)$ at time t does not only depend on the current input $u(t)$, but also on $u(\tau)$, $\tau < t$.

The state of a system

The state $x(t)$ of a system contains all the information necessary to unambiguously determine future outputs if future inputs are known.

Example: Level-system

- All past inputs: $x(t) = \{u(\tau), \tau < t\}$
  (infinite amount of information).
- Water level: $x(t) = \begin{bmatrix} h_1(t) \\ h_2(t) \end{bmatrix}$.
- Volume: $x(t) = \begin{bmatrix} V_1(t) \\ V_2(t) \end{bmatrix} = \begin{bmatrix} A_1 h_1(t) \\ A_2 h_2(t) \end{bmatrix}$.
- Infinite number of ways to choose the states.



per.mattsson@hig.se

State space form

Linear system

An LTI-system can be written on state-space form as

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t) + Du(t),\end{aligned}$$

where $x(t)$ is the state vector, and A, B, C, D are matrices.

per.mattsson@hig.se

State space form

Linear system

An LTI-system can be written on state-space form as

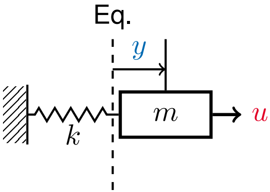
$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t), \end{aligned}$$

where $x(t)$ is the state vector, and A, B, C, D are matrices.

MATLAB: » `sys = ss(A,B,C,D);`

per.mattsson@hig.se

Example: Spring

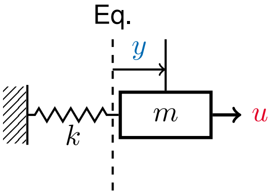


Using Newton's second law, and Hooke's law we get:

$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

per.mattsson@hig.se

Example: Spring



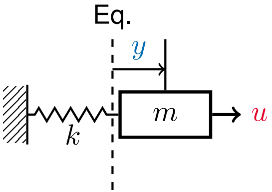
Using Newton's second law, and Hooke's law we get:

$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.

per.mattsson@hig.se

Example: Spring



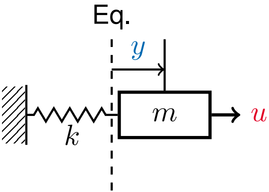
Using Newton's second law, and Hooke's law we get:

$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \tag{1}$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$.

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

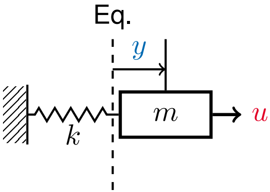
$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} =$$

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

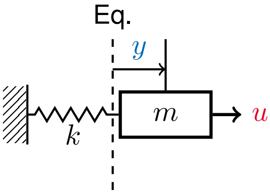
$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} \qquad\qquad & \\ & \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} \quad \\ \\ \end{bmatrix}}_{B} u(t),$$

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

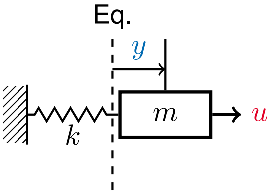
$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ & \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \end{bmatrix}}_{B} u(t),$$

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

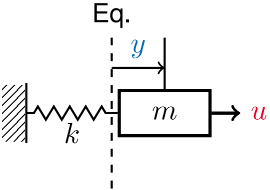
$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}}_{B} u(t),$$

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

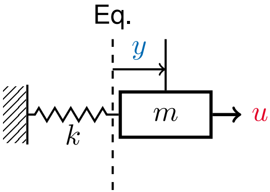
- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}}_{B} u(t),$$

$$y(t) = \underbrace{\begin{bmatrix} \quad\quad \end{bmatrix}}_{C} x(t) + \underbrace{\begin{bmatrix} \quad \end{bmatrix}}_{D} u(t).$$

per.mattsson@hig.se

Example: Spring



Using Newton's second law, and Hooke's law we get:

$$\ddot{y}(t) = -\frac{k}{m}y(t) + \frac{1}{m}u(t) \qquad (1)$$

- If the position and velocity are known, we can compute future outputs if future inputs are known.
- Let $x(t) = \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix}$. From (1) we get

$$\dot{x}(t) = \begin{bmatrix} \dot{y}(t) \\ \ddot{y}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}}_{B} u(t),$$

$$y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} x(t) + \underbrace{\begin{bmatrix} 0 \end{bmatrix}}_{D} u(t).$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s)$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t) + Du(t)\end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$, and

$$Y(s) = CX(s) + DU(s) =$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t) + Du(t)\end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$, and

$$Y(s) = CX(s) + DU(s) = C(sI - A)^{-1}BU(s) + DU(s).$$

per.mattsson@hig.se

HÖGSKOLAN I GÄVLE

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$, and

$$Y(s) = CX(s) + DU(s) = C(sI - A)^{-1}BU(s) + DU(s).$$

Hence, the transfer function is given by

$$\boxed{G(s) = \qquad\qquad\qquad\qquad}$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t) + Du(t)\end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$, and

$$Y(s) = CX(s) + DU(s) = C(sI - A)^{-1}BU(s) + DU(s).$$

Hence, the transfer function is given by

$$\boxed{G(s) = C(sI - A)^{-1}B + D.}$$

per.mattsson@hig.se

From state space form to transfer function

$$\boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}} \longrightarrow \boxed{Y(s) = G(s)U(s)}$$

Using the Laplace transform we get

$$sX(s) = AX(s) + BU(s) \iff (sI - A)X(s) = BU(s).$$

hence $X(s) = (sI - A)^{-1}BU(s)$, and

$$Y(s) = CX(s) + DU(s) = C(sI - A)^{-1}BU(s) + DU(s).$$

Hence, the transfer function is given by

$$\boxed{G(s) = C(sI - A)^{-1}B + D.}$$

MATLAB: » `sys = ss(A,B,C,D); G = tf(sys);`

per.mattsson@hig.se

From transfer function to state space

$$\boxed{Y(s) = G(s)U(s)} \longrightarrow \boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}}$$

per.mattsson@hig.se

From transfer function to state space

$$\boxed{Y(s) = G(s)U(s)} \longrightarrow \boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}}$$

- For every transfer function, there exists an infinite number of state space representations.

per.mattsson@hig.se

From transfer function to state space

$$\boxed{Y(s) = G(s)U(s)} \longrightarrow \boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}}$$

- For every transfer function, there exists an infinite number of state space representations.
- MATLAB: » `sys = ss(G)` (Gives *one* state space realization. `canon` can be used to find other)

per.mattsson@hig.se

From transfer function to state space

$$\boxed{Y(s) = G(s)U(s)} \longrightarrow \boxed{\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}}$$

- For every transfer function, there exists an infinite number of state space representations.
- MATLAB: » `sys = ss(G)` (Gives *one* state space realization. `canon` can be used to find other)
- Canonical forms.

per.mattsson@hig.se

Canonical forms

Observable canonical form

A SISO-system with the transfer function

$$G(s) = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \cdots + b_{n-1} s + b_n}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n}.$$

Observable canonical form:

$$\dot{x}(t) = \begin{bmatrix} -a_1 & 1 & 0 & \cdots & 0 \\ -a_2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{n-1} & 0 & 0 & \cdots & 1 \\ -a_n & 0 & 0 & \cdots & 0 \end{bmatrix} x(t) + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix} x(t)$$

per.mattsson@hig.se

Canonical forms

Controllable form

A SISO-system with the transfer function

$$G(s) = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \cdots + b_{n-1} s + b_n}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n}.$$

Controllable canonical form:

$$\dot{x}(t) = \begin{bmatrix} -a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} x(t).$$

per.mattsson@hig.se

Some concepts

- Let n be the number of states (elements in $x(t)$)

per.mattsson@hig.se

Some concepts

- Let n be the number of states (elements in $x(t)$)
- **Observable** if and only if the matrix

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

has full rank. Observable canonical form $\Rightarrow$ observable.

per.mattsson@hig.se

Some concepts

- Let n be the number of states (elements in $x(t)$)
- **Observable** if and only if the matrix

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

  has full rank. Observable canonical form $\Rightarrow$ observable.
- **Controllable** if and only if the matrix

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

  has full rank. Controllable canonical form $\Rightarrow$ Controllable.

per.mattsson@hig.se

Some concepts

- Let n be the number of states (elements in $x(t)$)
- **Observable** if and only if the matrix

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

has full rank. Observable canonical form $\Rightarrow$ observable.

- **Controllable** if and only if the matrix

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

has full rank. Controllable canonical form $\Rightarrow$ Controllable.

- **Minimal realization** if and only if both controllable and observable. For a minimal realization, it is not possible to find a state space representation with fewer states.

per.mattsson@hig.se