University of Cyprus
Department of Electrical and Computer Engineering

# ECE 327: Lecture 3

# State Space Representation, Controllability and Observability

## State space representation

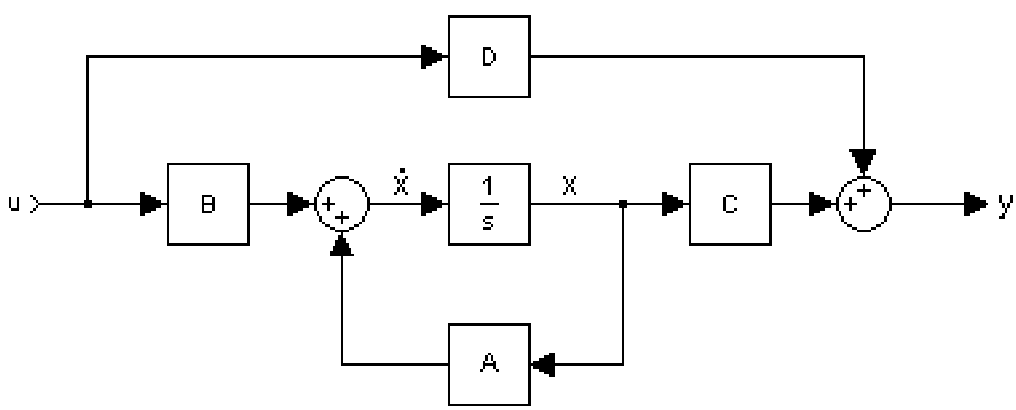
A system is represented in state space by the following equations:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

where

- $x(t)$: State vector
- $u(t)$: Input/control vector
- $y(t)$: Output vector
- $A \in \mathbb{R}^{n \times n}$: System matrix. It relates how the current state affects the state change $\dot{x}$
- $B \in \mathbb{R}^{n \times m}$: Input/control matrix.
- $C \in \mathbb{R}^{l \times n}$: Output matrix. Determines the relationship between the system state and the system output
- $D \in \mathbb{R}^{l \times m}$: Feedthrough/feedforward matrix. Allows the system input to affect the system output directly

The first equation is called the state equation and the second the output equation. In order to create a state space system, we need to know the matrices $A, B, C, D$.




We can **define a state space system** by using the function `sys = ss(A,B,C,D)`.

Given that we have already defined our state space system in the workspace as we did with the above command, we can extract matrices $A, B, C, D$ using `[A,B,C,D]=ssdata(sys)`.

We can **convert from a transfer function to a state space system**, in other words, if we have a transfer function, we can find the state space representation that corresponds to this transfer function. To achieve this use

```
[A,B,C,D] = tf2ss(num,den);

sys = ss(A,B,C,D)
```

Function `tf2ss(num,den)` returns matrices $A, B, C, D$. Note that, we have to assign 4 output variables, in this case $A, B, C, D$, otherwise Matlab will not return the complete result. Then, we create the state space model with the command `ss`. An alternative way of converting to a state space model from a transfer function, is by using `sys_ss = ss(sys)`, provided that, we have already stored the transfer function inside the variable `sys`. This method though has a disadvantage, since the matrices $A, B, C, D$ are not saved as variables in the workspace. On the other hand, in order to **convert from a state space model to a transfer function**, we can use

```
[num,den] = ss2tf(A,B,C,D);

sys = tf(num,den)
```

Function `ss2tf(A,B,C,D)` returns the numerator and the denominator of the transfer function. Then, we create the transfer function with the command `tf`. Alternatively, if we have already defined our system in state space form inside the variable `sys` we can use `sys_tf = tf(sys)`.

**Example:** Consider the following transfer function

$$G(s) = \frac{1}{s^2 + 2s + 1}$$

Find the state space model, that corresponds to this transfer function.

```
num = 1;
den = [1 2 1];
[A,B,C,D] = tf2ss(num,den); % Convert from tf to ss.
sys = ss(A,B,C,D)
```

```
sys =

  A =
       x1  x2
   x1  -2  -1
   x2   1   0

  B =
       u1
   x1   1
   x2   0

  C =
       x1  x2
   y1   0   1
```

```
D =
     u1
 y1   0

Continuous-time state-space model.
```

## State-Variable Modeling

State equations may be obtained from an $n$th-order differential equation or directly from the system model by identifying appropriate state variables.

To illustrate how we select a set of state variables, consider an $n$th-order linear plant model described by the differential equation:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \ldots + a_1\frac{dy}{dt} + a_0 y = u(t).$$

A state model for this system is not unique, but depends on the choice of a set of state variables. A useful set of state variables, referred to as *phase variables*, is defined by

$$x_1 = y,\ x_2 = \dot{y},\ x_3 = \ddot{y}, \ldots, x_n = y^{n-1}.$$

We express $\dot{x}_k = x_{k+1}$ for $k = 1, 2, \ldots, n-1$, and then solve for $d^n y/dt^n$ and replace $y$ and its derivatives by the corresponding state variables to give

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= x_3 \\
&\vdots \\
\dot{x}_{n-1} &= x_n \\
\dot{x}_n &= -a_0 x_1 - a_1 x_2 - \ldots - a_{n-1} x_n + u(t)
\end{aligned}$$

or, in matrix form

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & & & & \\ 0 & 0 & 0 & \ldots & 1 \\ -a_0 & -a_1 & -a_2 & \ldots & -a_{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} u(t)$$

and the output equation is

$$y = (1 \quad 0 \quad 0 \ldots \quad 0)x.$$

## Controllablity and Observability

Consider the control system presented in the state-space form

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) \qquad \text{(eqn.1)}$$

A system is said to be **controllable** when the plant input $u$ can be used to transfer the plant from any initial state to any arbitrary state in a finite time. The plant described by (eqn.1) with the system matrix having dimension $n \times n$ is completely state controllable if and only if the controllability matrix

$$\mathscr{C} = [B \;\; AB \;\; A^2B \;\; \dots A^{n-1}B]$$

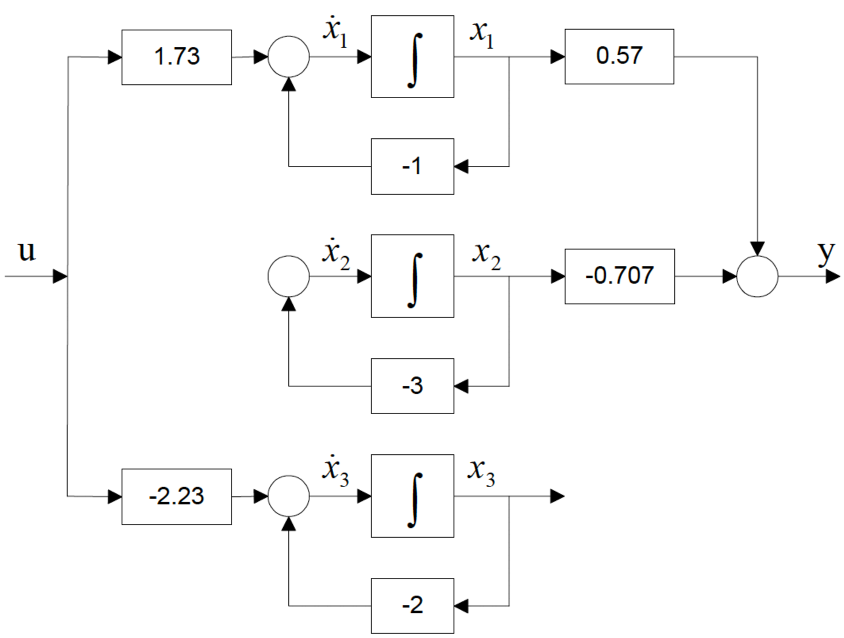
has rank $n$. The MATLAB function $\mathscr{C} = \text{ctrb}(A, B)$ returns the matrix $\mathscr{C}$ and determines whether or not the system is state controllable.

A system is said to be **observable** if the initial vector $x(0) = x_0$ can be found from the measurement of $u(t)$ and $y(t)$. The plant described by (eqn.1) is completely state observable if the inverse of the observability matrix

$$\mathscr{O} = [C \;\; CA \;\; CA^2 \;\; \dots CA^{n-1}]$$

exists (also called nonsingular). The MATLAB function $\mathscr{O} = \text{obsv}(A, C)$ returns the matrix $\mathscr{O}$ and determines whether or not the system is state observable.

**Example:** Consider the following block diagram:




From the above diagram, discuss whether the system is controllable and observable.

- No matter, how we change $u(t)$, $x_2(t)$ is not affected. Thus, $x_2$ is not controllable ($x_1$ and $x_3$ are controllable).
- $x_3(t)$ cannot be observed from $y(t)$. Thus, $x_3$ is not observable ($x_1$ and $x_2$ are observable).

**Frequency Domain Controllability and Observability Test:**

**If there are no zero-pole cancellations in the transfer function of a single-input single-output system, then the system is both controllable and observable. If a zero-pole cancellation occurs, then the system is either uncontrollable or unobservable or both uncontrollable and unobservable.**

From the above statement, it follows that a single-input single-output dynamic system is irreducible if and only if it is both controllable and observable. Such a system realization is called the ***minimal realization***. If the system is either uncontrollable and/or unobservable it can be represented by a system whose order has been reduced by removing uncontrollable and/or unobservable modes. The MATLAB function `minreal(sys)` returns the minimal realization or pole-zero cancellation.