MENG366

State Space Modeling

Dr. Saeed Asiri
saeed@asiri.net

Basic Concepts related to State Space

- **State**

  State Variables

  The smallest set of variables {$q_1, q_2, \ldots, q_n$} such that the knowledge of these variables at time $t = t_0$, together with the knowledge of the input for $t \geq t_0$ completely determines the behavior (the values of the state variables) of the system for time $t \geq t_0$.

- **State Vector**

  All the state variables {$q_1, q_2, \ldots, q_n$} can be looked on as components of state vector.

- **State Space**

  A space whose coordinates consist of state variables is called a state space. Any state can be represented by a point in state space.

State Space Representation

- **State Space Representation**
  - Two parts:
    - A set of first order ODEs that represents the derivative of each state variable q_i as an algebraic function of the set of state variables $\{q_i\}$ and the inputs $\{u_i\}$.

$$\begin{cases} \dot{q}_1 = f_1(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \\ \dot{q}_2 = f_2(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \\ \vdots \\ \dot{q}_n = f_n(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \end{cases}$$

    - A set of equations that represents the output variables as algebraic functions of the set of state variables $\{q_i\}$ and the inputs $\{u_i\}$.

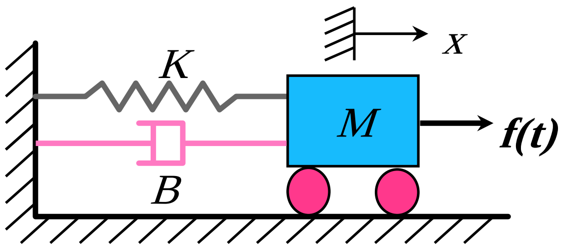
$$\begin{cases} y_1 = g_1(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \\ y_2 = g_2(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \\ \vdots \\ y_p = g_p(q_1, q_2, q_3, \dots, q_n, u_1, u_2, u_3, \dots, u_m) \end{cases}$$

State Space Representation

Example:

EOM:

$$M\ddot{x} + B\dot{x} + Kx = f(t)$$



Q: What information about the mass do we need to know to be able to solve for $x(t)$ for $t \geq t_0$?

Input: $f(t), \quad t \geq t_0$

Initial Conditions (ICs):

$x(t_0) \qquad q_1 = x(t)$

$\dot{x}(t_0) \qquad q_2 = \dot{x}(t)$

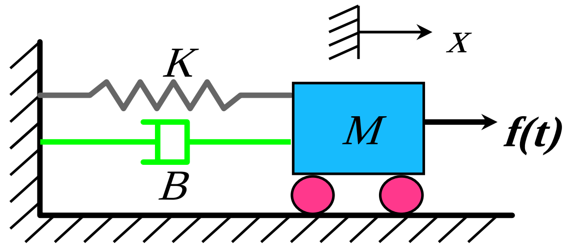
Rule of Thumb

Number of state variables = Sum of orders of EOMs

State Space Representation

- Example

EOM
$$M\ddot{x} + B\dot{x} + Kx = f(t)$$



State Variables:
$$q_1 = x,$$
$$q_2 = \dot{x}$$

Outputs:
$$y_1 = x, \qquad y_2 = -B\dot{x}$$

State Space Representation:

State equation

$$\begin{cases} \dot{q}_1 = \dot{x} = q_2 \\ \dot{q}_2 = \ddot{x} = \dfrac{1}{M}(-Bq_2 - Kq_1) + \dfrac{1}{M} f(t) \end{cases}$$

Output equation

$$\begin{cases} y_1 = x = q_1 \\ y_2 = -B\dot{x} = -Bq_2 \end{cases}$$

Matrix Form

$$\underbrace{\begin{bmatrix}\dot{q}_1 \\ \dot{q}_2\end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix}0 & 1 \\ -\dfrac{K}{M} & -\dfrac{B}{M}\end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix}q_1 \\ q_2\end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix}0 \\ \dfrac{1}{M}\end{bmatrix}}_{\mathbf{B}} \underbrace{[f]}_{u},$$

$$\underbrace{\begin{bmatrix}y_1 \\ y_2\end{bmatrix}}_{y} = \underbrace{\begin{bmatrix}1 & 0 \\ 0 & -B\end{bmatrix}}_{\mathbf{C}} \cdot \underbrace{\begin{bmatrix}q_1 \\ q_2\end{bmatrix}}_{\mathbf{x}} + \underbrace{0_{2\times1}}_{D} \cdot u$$

State Space Representation

- **Obtaining State Space Representation**
  - Identify State Variables
    - *Rule of Thumb:*
      - *Nth order ODE requires N state variables.*
      - *Position and velocity of inertia elements are natural state variables for translational mechanical systems.*
  - Eliminate all algebraic equations written in the modeling process.
  - Express the resulting differential equations in terms of state variables and inputs in coupled first order ODEs.
  - Express the output variables as algebraic functions of the state variables and inputs.
  - For linear systems, put the equations in matrix form.

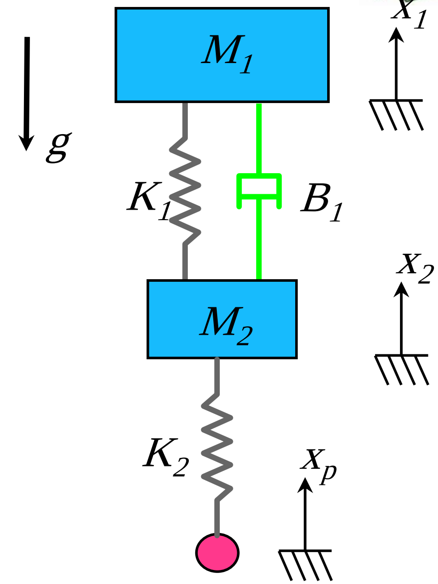
$$\dot{\mathbf{x}} = \mathbf{A}\cdot \underbrace{\mathbf{x}}_{\text{State Vector}} + \mathbf{B}\cdot \underbrace{\mathbf{u}}_{\text{Input Vector}}$$

$$\underbrace{\mathbf{y}}_{\text{Output Vector}} = \mathbf{C}\cdot\mathbf{x} + \mathbf{D}\cdot\mathbf{u}$$

State Space Representation

- Exercise

Represent the 2 DOF suspension system in a state space representation. Let the system output be the relative position of mass M_1 with respect to M_2.



$$M_1\ddot{x}_1 + B_1\dot{x}_1 - B_1\dot{x}_2 + K_1x_1 - K_1x_2 = 0$$

$$M_2\ddot{x}_2 - B_1\dot{x}_1 + B_1\dot{x}_2 - K_1x_1 + (K_1 + K_2)x_2 = K_2x_p$$

State Variables: $q_1 = x_1, \quad q_2 = \dot{x}_1, \quad q_3 = x_2, \quad q_4 = \dot{x}_2$

Output: $y = x_1 - x_2$ ***Input:*** x_p

State Space Representation:

$$\underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_1}{M_1} & -\frac{B_1}{M_1} & \frac{K_1}{M_1} & \frac{B_1}{M_1} \\ 0 & 0 & 0 & 1 \\ \frac{K_1}{M_2} & \frac{B_1}{M_2} & -\frac{K_1+K_2}{M_2} & -\frac{B_1}{M_2} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{K_2}{M_2} \end{bmatrix}}_{\mathbf{B}} \underset{\text{u}}{x_p} \quad ,$$

$$\text{y} = \underbrace{\begin{bmatrix} 1 & 0 & -1 & 0 \end{bmatrix}}_{\mathbf{C}} \cdot \mathbf{x} + \underset{\text{D}}{0} \cdot \text{u}$$

Input/Output Representation

- Input/Output Model

  Uses one nth order ODE to represent the relationship between the input variable, u(t), and the output variable, y(t), of a system.

  For linear time-invariant (LTI) systems, it can be represented by :

$$a_n y^{(n)} + \cdots + a_2 \ddot{y} + a_1 \dot{y} + a_0 y = b_m u^{(m)} + \cdots + b_2 \ddot{u} + b_1 \dot{u} + b_0 u(t)$$

  where

$$y^{(n)} = \left(\frac{d}{dt}\right)^n y$$

  – To solve an input/output differential equation, we need to know

    Input: $u(t), \quad t \geq 0$

    Initial Conditions (ICs): $\underbrace{y(0), \quad \dot{y}(0), \quad \cdots, \quad y^{(n-1)}(0)}_{n}$

  – To obtain I/O models:
    - Identify input/output variables.
    - Derive equations of motion.
    - Combine equations of motion by eliminating all variables except the input and output variables and their derivatives.

Input/Output Representation

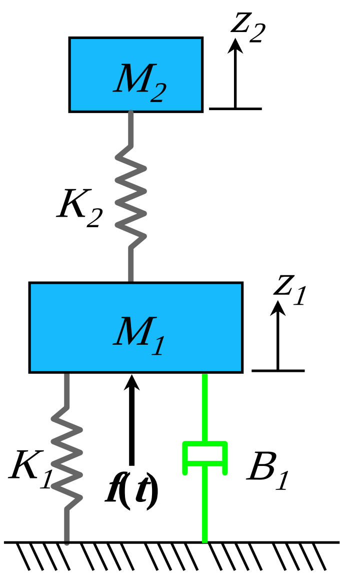
- **Example**

  Vibration Absorber

  EOM: $M_1\ddot{z}_1 + B_1\dot{z}_1 + (K_1 + K_2)z_1 - K_2 z_2 = f(t)$

  $M_2\ddot{z}_2 + K_2 z_2 - K_2 z_1 = 0$

  - Find input/output representation between input $f(t)$ and output z_2.



- Need to eliminate z_1 and its time derivative of all orders

$$z_1 = \frac{1}{K_2}\{M_2\ddot{z}_2 + K_2 z_2\},$$

$$\dot{z}_1 = \frac{1}{K_2}\{M_2 z_2^{(3)} + K_2\dot{z}_2\},$$

$$\ddot{z}_1 = \frac{1}{K_2}\{M_2 z_2^{(4)} + K_2\ddot{z}_2\}$$

$$M_1\underbrace{\frac{1}{K_2}\{M_2 z_2^{(4)} + K_2\ddot{z}_2\}}_{\ddot{z}_1} + B_1\underbrace{\frac{1}{K_2}\{M_2 z_2^{(3)} + K_2\dot{z}_2\}}_{\dot{z}_1} + (K_1 + K_2)\underbrace{\frac{1}{K_2}\{M_2\ddot{z}_2 + K_2 z_2\}}_{z_1} - K_2 z_2 = f(t)$$

$$z_2^{(4)} + \frac{B_1}{M_1}z_2^{(3)} + \frac{M_1K_2 + M_2(K_1 + K_2)}{M_1M_2}\ddot{z}_2 + \frac{B_1K_2}{M_1M_2}\dot{z}_2 + \frac{K_1K_2}{M_1M_2}z_2 = \frac{K_2}{M_1M_2}f$$

Input/Output Models
vs
State-Space Models

- **State Space Models:**
  - consider the internal behavior of a system
  - can easily incorporate complicated output variables
  - have significant computation advantage for computer simulation
  - can represent multi-input multi-output (MIMO) systems and nonlinear systems
- **Input/Output Models:**
  - are conceptually simple
  - are easily converted to frequency domain transfer functions that are more intuitive to practicing engineers
  - are difficult to solve in the time domain (solution: Laplace transformation)

State Space Equations

Dynamic Equations

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

Observation Equations

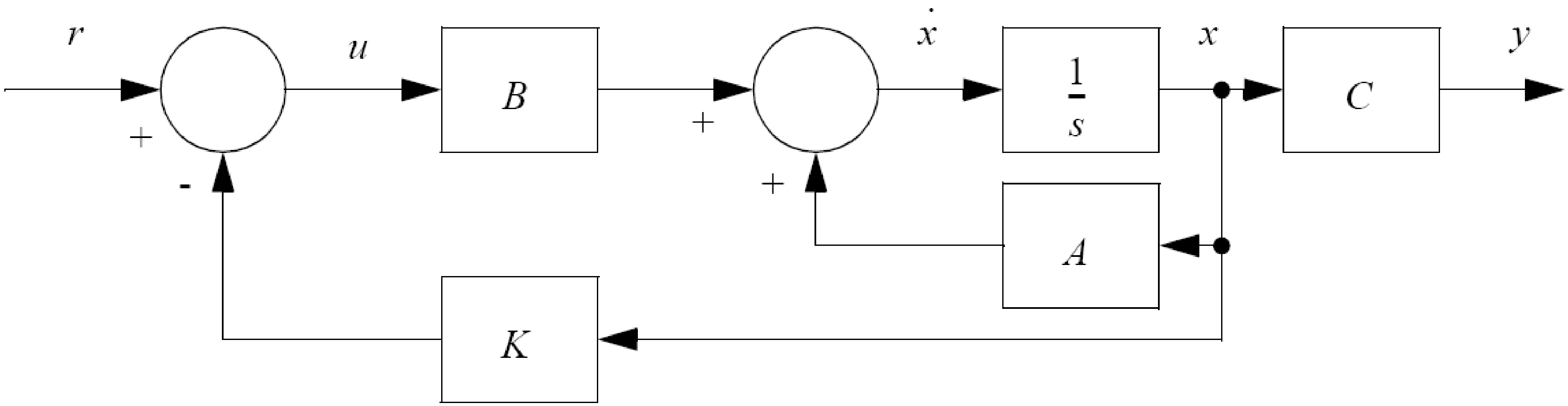
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

x is the state vector of the system, nx1;
n is the order of the system;
u is the input vector, mx1;
y is the output vector, px1;
A is the system (or coefficient) matrix, nxn;
B is the input (or driving) matrix, nxm;
C is the observation matrix, pxn;
D is the feedforward matrix, pxm.

MATLAB function:

```
SYS = SS(A,B,C,D
SYS = SS(A,B,C,D,T)
SYS = SS
SYS = SS(D)
SYS =
SS(A,B,C,D,LTISYS)
SYS = SS(SYS)
```

State Space Equations



Block diagram of a state based controller

Solving the State Equations- Free Response

$$\mathbf{x}_{\mathbf{IC}}(t) = \mathcal{L}^{-1}\{[s\mathbf{I} - \mathbf{A}]^{-1}\}\mathbf{x}(0) = \Phi(t,0)\mathbf{x}(0) = \mathbf{exp}(\mathbf{A}t)\mathbf{x}(0)$$

where $\Phi(t,0)$ is called the state transition matrix and

$\mathbf{exp}(\mathbf{A}t)$ is called the matrix exponential function and is defined as,

$$\mathbf{exp}(\mathbf{A}t) = \mathbf{e}^{\mathbf{A}t} = \mathbf{I} + \mathbf{A}t + \mathbf{A}^2 t^2 + \cdots + \frac{1}{i!}\mathbf{A}^i t^i + \cdots$$

MATLAB function:

```
INITIAL(SYS,X0)
INITIAL(SYS,X0,TFINAL)
INITIAL(SYS,X0,T)
INITIAL(SYS1,SYS2,...,X0,T)
```

Solving the State Equations- Forced Response

$$\mathcal{L}^{-1}\{g(s)u(s)\} = \int_{-\infty}^{\infty} g(t-\tau)u(\tau)d\tau \qquad \text{Convolution Integral}$$

$$\Rightarrow \quad \mathbf{x}_{\text{forced}}(t) = \int_{0}^{t} \mathbf{exp}\{\mathbf{A}(t-\tau)\}\mathbf{Bu}(\tau)d\tau$$

MATLAB function:
IMPULSE : impulsive input
STEP: step input
LSIM: arbitrary input

State Space Representations

For a specific transfer function or block diagram representation of a system there is no unique state space representation.

We may choose state variables that are:

1) *physically meaningful* or
2) *mathematically convenient*.

Some representations are particularly useful and have been standardized as *"canonical" forms*.

State Space Model to Transfer Function Model

MATLAB function:
```
[NUM,DEN] =
SS2TF(A,B,C,D,iu)
```

$$s\mathbf{x}(s) - \mathbf{x}(0) = A\mathbf{x}(s) + B\mathbf{u}(s)$$

Solving for $\mathbf{x}(s)$,

$$\mathbf{x}(s) = \mathrm{inv}(sI-A)\mathbf{x}(0) + \mathrm{inv}(sI-A)B\mathbf{u}(s).$$

So,

$$\mathbf{y}(s) = C\mathbf{x}(s) + D\mathbf{u}(s)$$

$$= C\,\mathrm{inv}(sI-A)B\mathbf{u}(s) + D\mathbf{u}(s) \quad \text{for } \mathbf{x}(0) = \mathbf{0}.$$

$$=[C\,\mathrm{inv}(sI-A)B + D]\mathbf{u}(s)$$

$$=\{[C\,\mathrm{adj}(sI-A)B + D\det(sI-A)]/\det(sI-A)\}\mathbf{u}(s)$$

$\det(sI-A) = 0$ is the *characteristic equation* of the system and $[C\,\mathrm{adj}(sI-A)B + D\det(sI-A)]$ is the pxm *matrix of system zeros.*

State Space Model to Transfer Function Model

State equations as functions of time

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

In the s-domain

$$sX - X_0 = AX + BU$$

$$X(sI - A) = BU + X_0$$

$$X = (sI - A)^{-1}BU + (sI - A)^{-1}X_0$$

$$Y = CX + DU$$

$$Y = C((sI - A)^{-1}BU + (sI - A)^{-1}X_0) + DU$$

$$Y = (C(sI - A)^{-1}B + D)U + C(sI - A)^{-1}X_0$$

Assuming the system starts at rest,

$$Y = (C(sI - A)^{-1}B + D)U$$

$$\frac{Y}{U} = (C(sI - A)^{-1}B + D) \qquad \text{(the transfer function)}$$

Controller Canonical Form

Consider the transfer function

$$\frac{y(s)}{u(s)} = \sum_{i=0}^{n-1} b_i s^i \Big/ \sum_{j=0}^{n} a_j s^{n-j}, a_0 = 1.$$

Define

$$y = \sum_i b_i x_{c_i} \quad \text{,and} \quad \dot{x}_{c_i} = x_{c_{i-1}}, i = 2, \ldots n$$

Structure of the state space model

$$\dot{\mathbf{x}}_{\mathbf{c}} = \mathbf{A}_{\mathbf{c}} \mathbf{x}_{\mathbf{c}} + \mathbf{B}_{\mathbf{c}} \mathbf{u}$$

$$\mathbf{A}_{\mathbf{c}} = \begin{bmatrix} -a_1 - a_2 - a_3 \ldots\ldots\ldots\ldots - a_{n-1} - a_n \\ \mathbf{I}_{(n-1)x(n-1)} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{0}_{(n-1)x1} \end{bmatrix}_{(nxn)}$$

$$\mathbf{B}_{\mathbf{c}} = \begin{bmatrix} 1 \\ \vdots \\ 0 \end{bmatrix}_{(nx1)} , \mathbf{C}_{\mathbf{c}} = \begin{bmatrix} b_1 & \cdots & b_n \end{bmatrix}, \ \mathbf{D}_{\mathbf{c}} = \mathbf{0}.$$

Controller Canonical Form

Controller canonical form is particularly useful:

- The coefficients of the numerator and denominator polynomials of the transfer function appear *directly* in the state variable model.
- *All other elements are either 0 or 1.*
- The *state variable model can be written by inspection*, and vice versa. This is used in MATLAB to compute the state space model from the transfer function model with the function **tf2ss**.
- *It replicates itself if state variable feedback is used.*

Controller Canonical Form

Consider a state variable feedback control law.

$$u(t) = -\mathbf{K}\mathbf{x}_\mathbf{c}(t), \mathbf{K} = \left[k_1 \ldots\ldots k_n\right]$$

$$\dot{\mathbf{x}}_\mathbf{c} = \mathbf{A}_\mathbf{c}\mathbf{x}_\mathbf{c} - \mathbf{B}_\mathbf{c}\mathbf{K}\mathbf{x}_\mathbf{c} = \left[\mathbf{A}_\mathbf{c} - \mathbf{B}_\mathbf{c}\mathbf{K}\right]\mathbf{x}_\mathbf{c}$$

$$\left[\mathbf{A}_\mathbf{c} - \mathbf{B}_c\mathbf{K}\right] = \begin{bmatrix} -(a_1 + k_1) \ldots\ldots\ldots\ldots -(a_n + k_n) \\ \mathbf{I}_{(n-1)x(n-1)} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{0}_{(n-1)x1} \end{bmatrix}$$

Thus, the closed-loop characteristic equation is:

$$s^n + (a_1 + k_1)s^{n-1} + (a_2 + k_2)s^{n-2} + \ldots\ldots\ldots + (a_n + k_n) = 0$$

The feedback coefficients determine the closed-loop poles.

State Space Representations

Example 1.1: Conversion from Transfer-Function Model to State-Space Model

$$\frac{Y(s)}{U(s)} = \frac{s}{s^3 + 14s^2 + 56s + 160}$$

MATLAB code:

```
n=[1 0]; d=[1 14 56 160];

[Ac,Bc,Cc,Dc]=tf2ss(n,d)  %Controller canonical form

system=tf(n,d)

[a,b,c,d]=ssdata(system)   % Not controller canonical form
```

State Space Representations

Example 1.1:

```
                s
  -------------------------------
  s^3 + 14 s^2 + 56 s + 160
```

```
Ac =
   -14   -56  -160
     1     0     0
     0     1     0
Bc =
     1
     0
     0
Cc =
     0     1     0
Dc =
     0
```

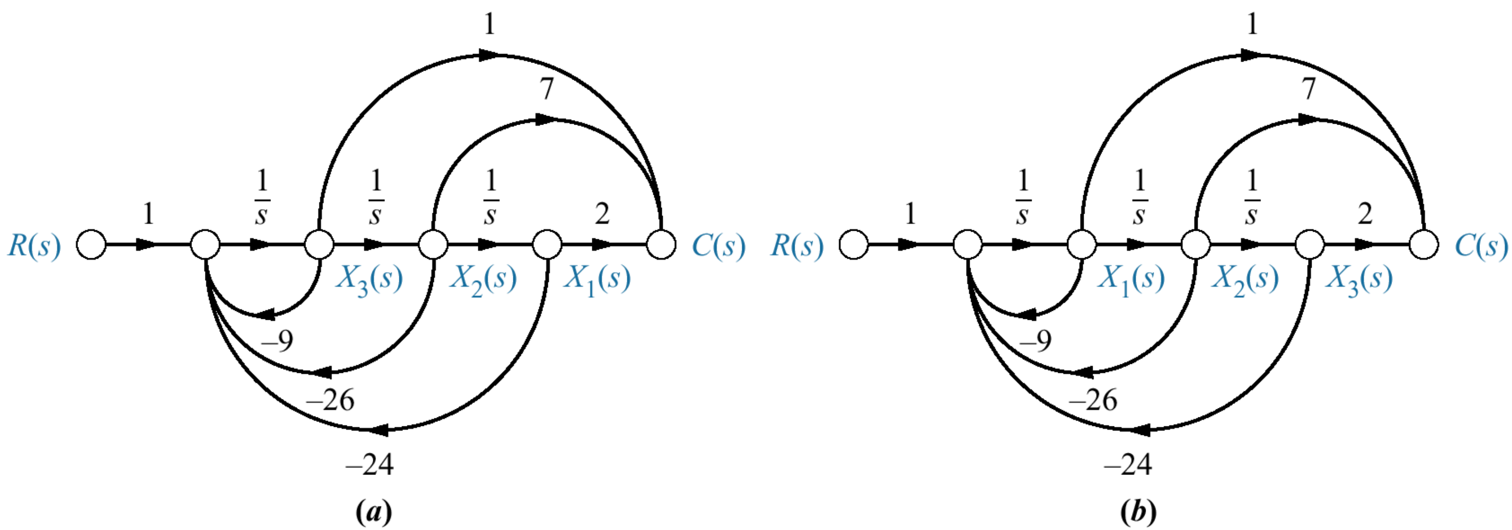
```
a =
   -14    -7    -5
     8     0     0
     0     4     0
b =
    0.2500
         0
         0
c =
         0    0.5000         0
d =
     0
```

»

Similarity between phase-variable & controller canonical forms

Signal-flow graphs for obtaining forms for

$G(s) = C(s)/R(s) = (s^2 + 7s + 2)/(s^3 + 9s^2 + 26s + 24)$:



a. phase-variable form; **b.** controller canonical form

Modal Canonical Form

MATLAB function:

```
[R,P,K] = RESIDUE(B,A)
[B,A] = RESIDUE(R,P,K)
```

For non-repeated roots

$$y(s)/u(s) = \sum_{i=1}^{n} r_i / (s - p_i) \quad \text{for} \quad p_i \neq p_j \quad i \neq j$$

$$Let \qquad x_{m_i}(s)/u(s) = r_i / (s - p_i)$$

So,

$$\dot{x}_{m_i}(t) = p_i x_{m_i}(t) + r_i u(t)$$

$$\mathrm{y(t)} = \sum_{i=1}^{\mathrm{n}} x_{m_i}(t) = [1 \ldots\ldots 1]\mathbf{x}$$

Modal Canonical Form - Vector-Matrix Format

The state variables in this format are *uncoupled*. they are called the *modes*. In vector-matrix notation,

$$\dot{\mathbf{x}}_{\mathbf{m}}(t) = \begin{bmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & p_n \end{bmatrix} \mathbf{x}_{\mathbf{m}}(t) + \begin{bmatrix} r_1 \\ \vdots \\ \vdots \\ r_n \end{bmatrix} u(t)$$

and

$$y = [1 \ldots 1]\mathbf{x}_{\mathbf{m}}.$$

Definition of Controllability

•If an input can be found that takes *every* state variable from an *arbitrary initial state* to a *desired final state* in a *finite amount of time*, the system is said to be *controllable*; otherwise the system is *uncontrollable*.

•The *controllability* of a system in *modal canonical form* can be determined by *inspection*.

• *A system is controllable iff all of its modes can be affected by the control (input) variable(s).*

Consider a scalar system

$$\dot{x} = px + ru$$

$$x(T) = e^{p(T-t_0)} x(t_0) + \int_{t_0}^{T} e^{p(T-\tau)} r u(\tau) d\tau$$

For u = constant = U,

$$U = \frac{x(T) - e^{p(T-t_0)}}{r \int_{t_0}^{T} e^{p(T-\tau)} d\tau}$$

Thus, the scalar system is *controllable iff r is not zero* and the controllability of each mode can be assessed by inspection of the modal canonical form.

Transforming State Equations

$$\mathbf{x} = \mathbf{T}\mathbf{z}$$

then $\quad \dot{\mathbf{x}} = \mathbf{T}\dot{\mathbf{z}} = \mathbf{A}\mathbf{T}\mathbf{z} + \mathbf{B}\mathbf{u}$

or $\quad \dot{\mathbf{z}} = \mathbf{T}^{-1}\mathbf{A}\mathbf{T}\mathbf{z} + \mathbf{T}^{-1}\mathbf{B}\mathbf{u} = \mathbf{F}\mathbf{z} + \mathbf{G}\mathbf{u}$

and $\quad \mathbf{y} = \mathbf{C}\mathbf{T}\mathbf{z} + \mathbf{D}\mathbf{u} = \mathbf{H}\mathbf{z} + \mathbf{J}\mathbf{u}.$

So that the state space models are related by the matrix transformations :

$$\mathbf{F} = \mathbf{T}^{-1}\mathbf{A}\mathbf{T}$$

$$\mathbf{G} = \mathbf{T}^{-1}\mathbf{B}$$

$$\mathbf{H} = \mathbf{C}\mathbf{T}$$

$$\mathbf{J} = \mathbf{D}.$$

Transforming to control canonical form

Suppose we wanted to transform an arbitrary state variable description (**A,B,C,D**) to the control canonical form.
Is it possible?

From the general transformation equations,

$$\mathbf{A_c T} = \mathbf{T}\ \ \mathbf{A}.$$

Considering the three-dimensional case and letting:

$$\mathbf{T} = \begin{bmatrix}\mathbf{t_1} & \mathbf{t_2} & \mathbf{t_3}\end{bmatrix}^T,$$

Transforming to control canonical form

$$\begin{bmatrix} -a_1 & -a_2 & -a_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \mathbf{t}_2^{\mathrm{T}} \\ \mathbf{t}_3^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \mathbf{t}_2^{\mathrm{T}} \\ \mathbf{t}_3^{\mathrm{T}} \end{bmatrix} \mathbf{A}$$

From the last row, $\mathbf{t}_2^{\mathrm{T}} = \mathbf{t}_3^{\mathrm{T}} \mathbf{A}$.

From the second row, $\mathbf{t}_1^{\mathrm{T}} = \mathbf{t}_2^{\mathrm{T}} \mathbf{A}$.

Also, since $\mathbf{B}_{\mathbf{c}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \mathbf{TB} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \mathbf{t}_2^{\mathrm{T}} \\ \mathbf{t}_3^{\mathrm{T}} \end{bmatrix} \mathbf{B}$,

$$1 = \mathbf{t}_1^T \mathbf{B} = \mathbf{t}_2^T \mathbf{AB} = \mathbf{t}_3^T \mathbf{A}^2 \mathbf{B}$$

$$0 = \mathbf{t}_2^T \mathbf{B} = \mathbf{t}_3^T \mathbf{A} \ \mathbf{B}$$

$$0 = \mathbf{t}_3^T \mathbf{B}$$

or

$$\mathbf{t}_3^T \left[\mathbf{B}, \mathbf{AB}, \mathbf{A}^2 \mathbf{B}\right] = \left[0 \quad 0 \quad 1\right].$$

Transforming to control canonical form

Solving for $\mathbf{t}_3^T$,

$$\mathbf{t}_3^T = [0 \quad 0 \quad 1][\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B}]^{-1}$$

From which we can solve for $\mathbf{t}_2^T$ and $\mathbf{t}_1^T$ from the equation derived above if, and only if, $[\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B}]^{-1}$ exists!

<u>General Case:</u> In general we define the <u>controllability matrix, **C**,</u> as:

$$C = [\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \quad \cdots \quad \mathbf{A}^{\mathbf{n-1}}\mathbf{B}]$$

$$\mathbf{t}_n^T = [0 \quad \cdots \quad 0 \quad 1]C^{-1}$$

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_n^T \mathbf{A}^{\mathbf{n-1}} \\ \mathbf{t}_n^T \mathbf{A}^{\mathbf{n-2}} \\ \vdots \\ \mathbf{t}_n^T \end{bmatrix}$$

MATLAB function:

```
CO = CTRB(A,B)
CO = CTRB(SYS)
```

Controllability - an alternate viewpoint

- We have just seen that an *arbitrary* state space representation can be transformed to the *control canonical* form *iff the controllability matrix is nonsingular.*

- Once in controller canonical form the closed-loop poles can be arbitrarily placed using state variable feedback.

- Thus, *a system is controllable iff its closed-loop poles can be arbitrarily selected using state variable feedback.*

- All *viewpoints of controllability are compatible.*

Example 1.2: Forming the Controllability Matrix

For the canonical state-space model defined in Example 1.2.1.1

CO = ctrb(Ac,Bc)

```
CO =
     1   -14   140
     0     1   -14
     0     0     1
```

Now we can check its determinant to make sure its inverse exists:

det(CO)

ans = 1

Therefore, the system is controllable.

Example 1.3: An Uncontrollable System

Let **a=[1 1;0 1];b=[1;0];CO=ctrb(a,b);det(CO)**

```
ans =
    0
```

Since the controllability matrix is singular the system is uncontrollable. Note the transfer function model has pole-zero cancellation at s = -1. This is symptomatic of an uncontrollable mode.

[zeros,poles]=ss2zp(a,b,[1 1],0)

```
zeros =
   1.0000
poles =
    1
    1
```

Transforming to modal canonical form

$$\dot{\mathbf{z}} = \mathbf{V}^{-1}\dot{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{A}\mathbf{x} + \mathbf{V}\mathbf{B}\mathbf{u} = \mathbf{V}^{-1}\mathbf{A}\mathbf{V}\ \ \mathbf{z} + \mathbf{V}^{-1}\mathbf{B}\mathbf{u} = \mathbf{A}_{\mathbf{m}}\mathbf{z} + \mathbf{B}_{\mathbf{m}}\mathbf{u}.$$

So, $\quad \mathbf{A}_{\mathbf{m}} = \mathbf{V}^{-1}\mathbf{A}\mathbf{V} \quad$ or $\quad \mathbf{V}\mathbf{A}_{\mathbf{m}} = \mathbf{A}\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{bmatrix} \begin{bmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & p_n \end{bmatrix}.$

Thus, $\quad \mathbf{A}\mathbf{v}_i = p_i \mathbf{v}_i \quad$ for $i = 1 \cdots n$ **Eigenvalue Problem**

where

$\{p_i\}$ are the eigenvalues of $\mathbf{A}$, i.e., $\det(\mathbf{A} - \mathrm{p}_i \mathbf{I}) = 0$; and

$\{\mathbf{v}_i\}$ are the eigenvectors of $\mathbf{A}$.

MATLAB function:

```
E = EIG(X)
[V,D] = EIG(X)
[V,D] = EIG(X,'nobalance')
E = EIG(A,B)
[V,D] = EIG(A,B)
```