# Modelling dynamic systems

Michela Mulas

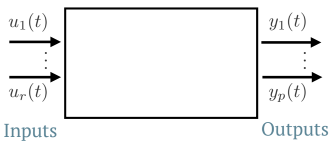
Today:

- Complex Numbers (revision for LAPLACE)
- Linearization EXAMPLE
- TF2SS (reading material for EASTER)

## Brief recap

### Last weeks, we did...

- Describe dynamical systems with **Input-output representation**:




$$\mathbf{u}(t) = [u_1(t) \dots u_r] \in \mathbb{R}^r$$
$$\mathbf{y}(t) = [y_1(t) \dots y_p] \in \mathbb{R}^p$$

A **SISO** system has $r = 1$ and $p = 1$.

A **MIMO** system has $r$-inputs and $p$-outputs.

- Focus on SISO systems

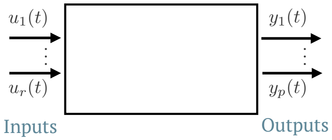
  The IO model for a SISO system is given by one differential equation as:

$$h\Big(\underbrace{y(t), \dot{y}(t), \dots, y^n(t)}_{\text{output}}, \underbrace{u(t), \dot{u}(t), \dots, u^m(t)}_{\text{input}}\Big) = 0$$

## Brief recap

### Last weeks, we did...

- Describe dynamical systems with **Input-output representation**:




$$\mathbf{u}(t) = [u_1(t)\dots u_r] \in \mathbb{R}^r$$
$$\mathbf{y}(t) = [y_1(t)\dots y_p] \in \mathbb{R}^p$$

A **SISO** system has $r = 1$ and $p = 1$.

A **MIMO** system has $r$-inputs and $p$-outputs.

- Focus on LTI-SISO systems

  An IO model is said to be **linear time invariant** if and only if the input-output relation is given by a linear differential equation and if and only if the IO relation is (explicitly) time independent.

$$a_0 y(t) + a_1 \dot{y}(t) + \dots + a_n y^{(n)}(t) = b_0 u(t) + b_1 \dot{u}(t) + \dots + b_n u^{(m)}(t)$$

$$a_0 Y(s) + a_1 s Y(s) + \dots + a_n s^n Y(s) = b_0 U(s) + b_1 s Y(s) + \dots + b_m s^m Y(s)$$

$$(a_0 + a_1 s + \dots + a_n s^n) Y(s) = (b_0 + b_1 s + \dots + b_m s^m) U(s)$$

$$Y(s) = \frac{N(s)}{D(s)} U(s)$$

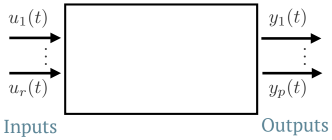
TRANSFER FUNCTION

$$Y(s) \rightsquigarrow y(t)$$

## Brief recap

### Last weeks, we did...

- Describe dynamical systems with **Input-output representation**:




$$\mathbf{u}(t) = [u_1(t) \dots u_r] \in \mathbb{R}^r$$
$$\mathbf{y}(t) = [y_1(t) \dots y_p] \in \mathbb{R}^p$$

A **SISO** system has $r = 1$ and $p = 1$.

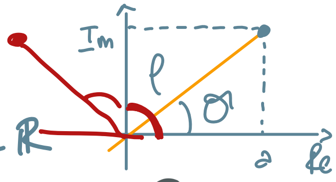
A **MIMO** system has $r$-inputs and $p$-outputs.

- Review the Laplace transform to easy the resolution of differential equations (e.g., using partial-fraction expansions)
  - ⇝ **How do we handle complex roots?**



COMPLEX NUMBERS (Back 2 Basics)

Imaginary units $j \triangleq \sqrt{-1}$




CARTESIAN FORM $c = a + jb$ $\quad c \in \mathbb{C};\ a, b \in \mathbb{R}$

Real part of c   Imaginary part of c

Conjugate of c ⇝ $c' = a - jb$

Check L4E2a

POLAR FORM $c = \rho e^{j\theta}$ $\quad \rho \geq 0,\ \theta \in \mathbb{R}$

Modulus or magnitude of c: $|c| = \sqrt{a^2 + b^2} = \rho$

Angle or phase of c: $\angle c = \theta = \arctan\left(\frac{b}{a}\right)$

COMPLEX EXPONENTIAL

$$e^c = e^{a+jb} = e^a e^{jb}$$
$$= e^a(\cos b + j\sin b)$$

EULER FORMULA

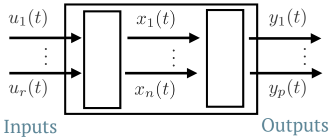
$$\cos\theta = \frac{e^{j\theta} + e^{-j\theta}}{2}$$

$$\sin\theta = \frac{e^{j\theta} - e^{-j\theta}}{2j}$$

## Brief recap

### Last weeks, we did...

- Describe dynamical systems with **State-Space representation**:




The **state of a system describes enough about the system to determine the future output behaviour** of $\mathbf{y}(t)$ given any external forces $\mathbf{u}(t)$ affecting the system.

- Focus on SISO: We have $n$ **differential equations**

They link the derivative of each state variable to the input and the output to the state variables and the input:

$$\begin{cases} \dot{x}(t) = f_1(x_1, \ldots, x_n(t), u(t), t) \\ \vdots \quad \vdots \\ \dot{x}_n(t) = f_n(x_1, \ldots, x_n(t), u(t), t) \\ y(t) = g(x_1, \ldots, x_n(t), u(t), t) \end{cases}$$

S-S
NONLINEAR

- Focus on LTI-SISO systems

**linear** if and only if the equations of state equation and the output transformation are linear equations

**time-invariant** if and only if the state equation and the output transformation equations are time independent.

$$\begin{cases} \dot{x}_1(t) = a_1 x_1(t) + \cdots + a_1 x_n(t) + b_1 u(t) \\ \vdots \\ \dot{x}_n(t) = a_n x_1(t) + \cdots + a_n x_n(t) + b_n u(t) \\ y(t) = c_1 x_1(t) + \cdots + c_n x_n(t) + d_1 u(t) + \cdots + d_n u(t) \end{cases}$$

$\rightsquigarrow \frac{dx_1}{dt}$

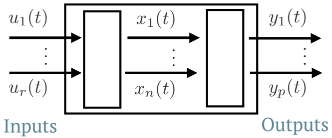
S-S
LINEAR

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ y(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$

## Brief recap

### Last weeks, we did...

- Describe dynamical systems with **State-Space representation**:




The **state of a system describes enough about the system to determine the future output behaviour** of $\mathbf{y}(t)$ given any external forces $\mathbf{u}(t)$ affecting the system.

- Focus on LTI-SISO systems which can be written in matrix form as

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ y(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t) \end{cases}$$



$\boxed{\text{LAGRANGE FORMULA}}$ for the continuous-time linear sys.

$\dot{x} = Ax + Bu$ with initial conditions $x(0) = x_0 \in \mathbb{R}^n$ there exists an unique sol.

$$x(t) = \underbrace{e^{At}x_0}_{x_n} + \underbrace{\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau}_{x_f}$$

The $\boxed{\text{exponential matrix } e^{At}}$ is defined as:

$$e^{At} \triangleq I + At + \frac{A^2t^2}{2} + \dots + \frac{A^nt^n}{n!}$$

## Brief recap

**Last weeks, we did...**

If $A \in \mathbb{R}^{n \times n}$ is diagonalizable $A = T \Lambda T^{-1}$ then

$$\Lambda = T^{-1} A T = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}$$

$T$ is the modal matrix of $A$

$$e^{At} = T \begin{bmatrix} e^{\lambda_1 t} & & & 0 \\ & e^{\lambda_2 t} & & \\ & & \ddots & \\ 0 & & & e^{\lambda_n t} \end{bmatrix}$$

The STATE TRAJECTORY is the NATURAL RESPONSE

$$x(t) = e^{At} x(0) = T e^{\Lambda t} T^{-1} x_0 = [v_1 \dots v_n] \begin{bmatrix} e^{\lambda_1 t} & & 0 \\ & \ddots & \\ 0 & & e^{\lambda_n t} \end{bmatrix}$$

$$= [v_1 e^{\lambda_1 t} \dots v_n e^{\lambda_n t}] \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = \sum_{i=1}^{n} \alpha \, e^{\lambda_i t} v_i$$

$v_i$ = eigenvectors of $A$

$\lambda_i$ = eingenvalues of $A$

$\alpha_1 = T^{-1} x_0 \in \mathbb{R}^n$

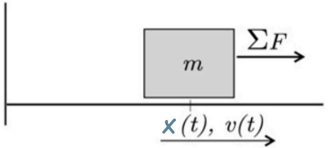
The evolution of the system depends on $\lambda_i$ of $A$ of the system

$e^{\lambda_i t}$ is the i-th MODE

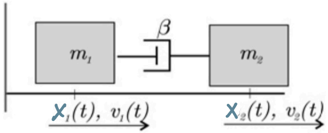
## Brief recap

MODELLING MECHANICAL SYSTEMS ⇝ NEWTON LAW

**Last weeks, we did...**

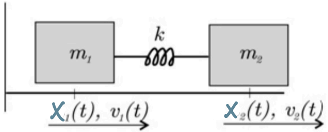



$$\Sigma F = m\frac{dv(t)}{dt} = m\frac{d^2x(t)}{dt}$$




VISCOUS FRICTION

$$F_1(t) = \beta\left(v_2(t) - v_1(t)\right) = -F_2(t)$$




ELASTIC COUPLING

$$F_1(t) = K\left(x_1(t) - x_2(t)\right) = -F_2(t)$$

$x_i(t)$ is the position of the body $i$

$v_i(t)$ is the velocity of the body $i$

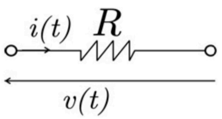
with respect to a fixed (inertial) reference frame

$F_i(t)$ is the force acting on the body $i$

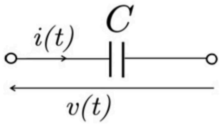
$$\left.\begin{array}{l} m \text{ is the mass} \\ \beta \text{ is the friction coefficient} \\ k \text{ is the spring constant} \end{array}\right\} \text{constant !!}$$

## Brief recap MODELLING ELECTRICAL SYSTEMS ⇝ KIRCHOFF' LAWS

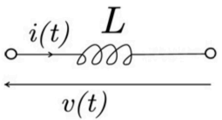
**Last weeks, we did...**




RESISTOR $v(t) = R\,i(t)$



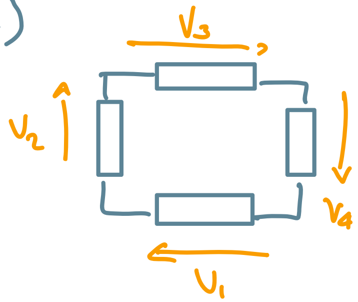

CAPACITOR $i(t) = C\,\frac{dv(t)}{dt}$




INDUCTOR $v(t) = L\,\frac{di(t)}{dt}$

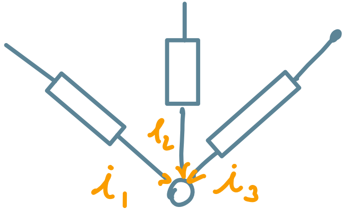
ELECTRIC SYSTEMS (example)

KIRCHOFF' VOLTAGE LAW
balance of voltage in a closed circuit




$V_1 + V_2 + V_3 + V_4 = 0$

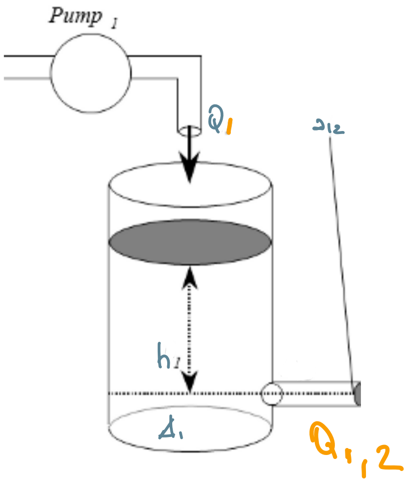
KIRCHOFF' CURRENT LAW
balance of the currents at a node




$i_1 + i_2 + i_3 = 0$

## Brief recap MODELLING HYDRAULIC SYSTEMS ⟶ MASS BALANCES

**Last weeks, we did...**




ASSUMPTIONS ⟶ the fluid is perfect
(no shear stress, no viscosity, no heat conduction, ...)

subject only to gravity

constant density $\rho$ (incompressible fluid)

constant external pressure

MASS (VOLUME) BALANCE

$$\sum Q_{ij} = \frac{dV_i(t)}{dt} = A_i \frac{dh_i(t)}{dt}$$

TORRICELLI LAW ⟶ $Q_{ij} = -a_{ij}\sqrt{2g\,h_i(t)}$

$A_i$ = cross-sectional area of the tank i

$h_i$ = fluid level in the tank i

$Q_{ij}$ = volumetric flow rate from tank j to tank i

$a_{ij}$ = area of the orifice of the tube from tank j to tank i

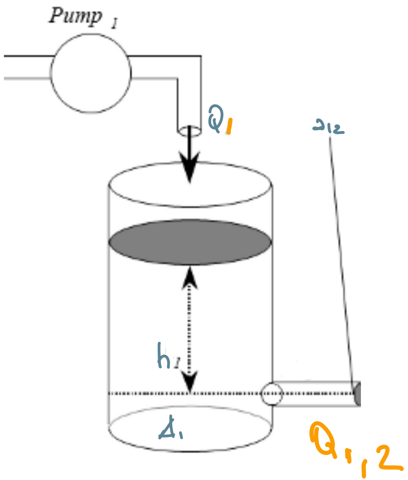
$g$ = gravitational acceleration

NONLINEAR $\xrightarrow{\text{TAYLOR}}$ LINEAR

## Brief recap

MODELLING HYDRAULIC SYSTEMS ⇝ MASS BALANCES

### Last weeks, we did...

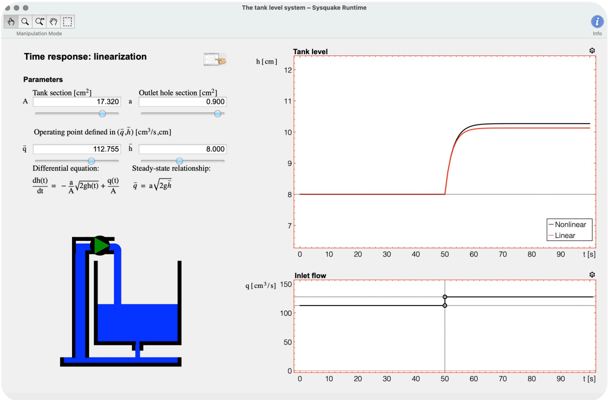



ASSUMPTIONS → the fluid is perfect (no shear stress, no viscosity, no heat conduction, ... )

- subject only to gravity
- constant density $\rho$ (incompressible fluid)
- constant external pressure

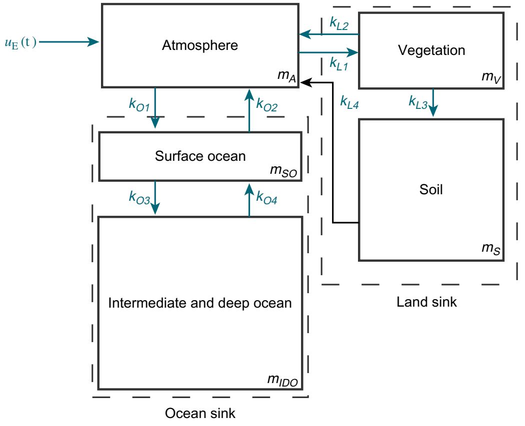
LINEARIZATION

BACK to LOG



## Brief recap

**Exercise L7E0**: The Figure shows a schematic description of the global carbon cycle.




- $m_A(t)$ represents the amount of carbon in gigatons (GtC) present in the atmosphere of earth.
- $m_V(t)$ the amount in vegetation.
- $m_s(t)$ the amount in soil.
- $m_{SO}(t)$ the amount in surface ocean.
- $m_{IDO}(t)$ the amount in intermediate and deep-ocean reservoirs.
- $u_E(t)$ stands for the human generated $CO_2$ emission in (GtC/yr).



SURFACE OCEAN $\quad \frac{dm_{SO}}{dt} = k_{O1} m_A - (k_{O2} + k_{O3}) m_{SO} + k_{O4} m_{IDO}$

INTERMEDIATE DEEP OCEAN: $\quad \frac{dm_{IDO}}{dt} = k_{O3} m_{SO} - k_{O4} m_{IDO}$

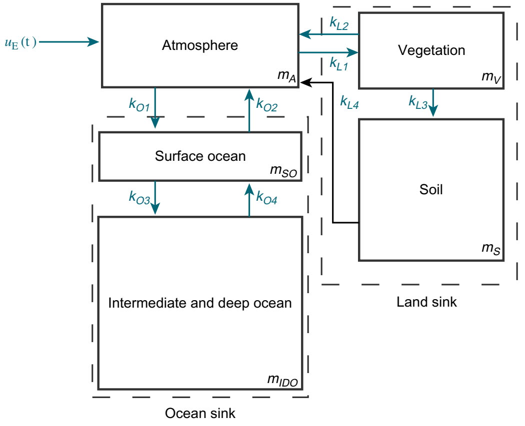
VEGETATION: $\quad \frac{dm_V}{dt} = k_{L1} m_A - (k_{L2} + k_{L3}) m_V$

ATMOSPHERE: $\quad \frac{dm_A}{dt} = u_E - (k_{O1} + k_{L1}) m_A + k_{L2} m_V + k_{O2} m_{SO} + k_{L4} m_S$

SOIL: $\quad \frac{dm_S}{dt} = k_{L3} m_V - k_{L4} m_S$

## Brief recap

**Exercise L7E0**: The Figure shows a schematic description of the global carbon cycle.




- The atmospheric mass balance in the atmosphere can be expressed as:

$$\frac{dm_A(t)}{dt} = u_E(t) - (k_{O1} + k_{L1})m_A(t) + k_{L2}m_V(t) + k_{O2}m_{SO}(t) + k_{L4}m_S(t)$$

$k$'s are the exchange coefficients (yr$^{-1}$)

$$\begin{bmatrix} \dot m_{SO} \\ \dot m_{IDO} \\ \dot m_V \\ \dot m_A \\ \dot m_S \end{bmatrix} = \overset{A}{\left| \begin{matrix} -(k_{02}+k_{03}) & k_{04} & 0 & k_{01} & 0 \\ k_{03} & -k_{04} & 0 & 0 & 0 \\ 0 & 0 & -(k_{L2}+k_{c3}) & k_{L1} & 0 \\ k_{02} & 0 & k_{L2} & -(k_{01}+k_{L1}) & k_{L4} \\ 0 & 0 & k_{L3} & 0 & -k_{L4} \end{matrix} \right|} \begin{bmatrix} m_{SO} \\ m_{IDO} \\ m_V \\ m_A \\ m_S \end{bmatrix} + \overset{B}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}} u_E$$

$$y = \underset{C}{[0\ 0\ 0\ 1\ 0]} \begin{bmatrix} m_{SO} \\ m_{IDO} \\ m_V \\ m_A \\ m_S \end{bmatrix}$$

## Today's goal

### Today's lecture is about ...

- Convert state space model to transfer function model.
- Convert transfer function model to state space model.

### Reading list

- Nise, *Control Systems Engineering* (6th Edition)[1]
- Ogata and Severo, *Engenharia de Controle Moderno* (3rd Edition)

[1] Today's lecture is mainly based on Ch.3 of Nise.
Same concepts can be found in Ogata, Ch.3, Sec. 4

## Converting from state space to a transfer function

Given the state and output equations:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$$
$$y = \mathbf{C}\mathbf{x} + \mathbf{D}u$$

SS

- Take the Laplace transform assuming zero initial conditions:

$$s\mathbf{X}(s) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}U(s)$$
$$Y(s) = \mathbf{C}\mathbf{X}(s) + \mathbf{D}U(s)$$

- Solve for $\mathbf{X}(s)$ the state equation and substitute it on the output equation:

$$Y(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}U(s) + \mathbf{D}U(s) = \left[\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right]U(s)$$

TF

$$G(s) = \frac{Y(s)}{U(s)}$$

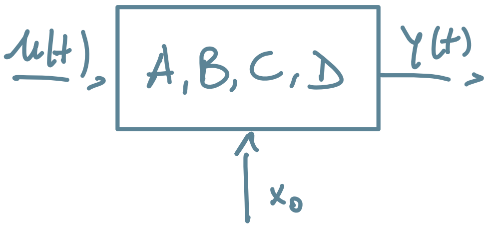
## Converting from state space to a transfer function

Given the state and output equations:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

- We call the matrix $\left[\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right]$ the **transfer function matrix**, since it relates the output vector, $\mathbf{Y}(s)$, to the input vector, $\mathbf{U}(s)$.
- If $\mathbf{U}(s) = U(s)$ and $\mathbf{Y}(s) = Y(s)$ are scalars, we can find the transfer function as:

$$T(s) = \frac{Y(s)}{U(s)} = \left[\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right]$$




(LTI)
STATE-SPACE


(LTI)
INPUT/OUTPUT

## Converting from state space to a transfer function

**Example**: Given the system defined by the equations below, find the transfer function, $T(s) = Y(s)/U(s)$, where $U(s)$ is the input and $Y(s)$ is the output.

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 10 \\ 0 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \mathbf{x}$$



MATRIX INVERSE (B2B)

$$(sI - A)^{-1} = \frac{\operatorname{adj}(sI - A)}{\det(sI - A)} \sim \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$\operatorname{adj} A = C^T = \begin{pmatrix} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \\ -\begin{vmatrix} a_{12} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{vmatrix} \\ +\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \end{pmatrix}^T$$

## Converting from state space to a transfer function

**Example**: Given the system defined by the equations below, find the transfer function, $T(s) = Y(s)/U(s)$, where $U(s)$ is the input and $Y(s)$ is the output.

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 10 \\ 0 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \mathbf{x}$$

The solution resolves around finding the term $(s\mathbf{I} - \mathbf{A})^{-1}$. We have all other terms already defined.

$$(s\mathbf{I} - \mathbf{A}) = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -2 & -3 \end{bmatrix} = \begin{bmatrix} s & -1 & 0 \\ 0 & s & -1 \\ 1 & 2 & s+3 \end{bmatrix}$$



$$(s\mathbf{I} - \mathbf{A})^{-1} = \frac{adj(s\mathbf{I} - \mathbf{A})}{det(s\mathbf{I} - \mathbf{A})} = \frac{\begin{bmatrix} (s^2+3s+2) & s+3 & 1 \\ -1 & s(s+3) & s \\ -s & -(2s+1) & s^2 \end{bmatrix}}{s^3+3s^2+2s+1}$$

Substituting it, together with $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$, we obtain the final result for the transfer function:

$$T(s) = \left[\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right] =$$

$$= \frac{10(s^2+3s+2)}{s^3+3s^2+2s+1}$$

## Converting from state space to a transfer function

### Matlab/Octave

```
A2=[0 1 0;0 0 1;-1 -2 -3];      % Matrix A
B2=[10;0;0];                    % Vector B
C2=[1 0 0];                     % Vector C
D2=0;                           % Scalar D
I=eye(size(B2,1));               % Create identity matrix

[num2,den2] = ss2tf(A2,B2,C2,D2);
T2=tf(num2,den2)

% Using the Symbolic toolbox
syms s
T=C2*((s*I-A2)^-1)*B2+D2;
```

```
import control as ctrl

A2=[[0, 1, 0],[0, 0, 1],[-1, -2, -3]];   # Matrix A
B2=[10, 0, 0];                           # Vector B
C2=[1, 0, 0];                            # Vector C
D2=0;                                    # Scalar D

TF2 = ctrl.ss2tf(A2, B2, C2, D2)

# Alternatively
sys_ss = ctrl.ss(A2, B2, C2, D2)
TF2 = ctrl.ss2tf(sys_ss)
```

## Converting a transfer function to state space

**Exercise L7E1**: Convert the state and output equation below to a transfer function.

$$\dot{\mathbf{x}} = \begin{bmatrix} -4 & -1.5 \\ 4 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 2 \\ 0 \end{bmatrix} u(t)$$
$$y = \begin{bmatrix} 1.5 & 0.625 \end{bmatrix} \mathbf{x}$$



$A = \begin{bmatrix} -4 & -1.5 \\ 4 & 0 \end{bmatrix}$ $\quad B = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$ $\quad C = \begin{bmatrix} 1.5 & 0.625 \end{bmatrix}$ $\quad D = 0$

$(sI - A) = \begin{bmatrix} s+4 & 1.5 \\ -4 & s \end{bmatrix}$

$\mathrm{adj}(sI - A) = \begin{bmatrix} s & -1.5 \\ 4 & s+4 \end{bmatrix}$

$\det(sI - A) = s^2 + 4s + 6$

$$(sI - A)^{-1} = \frac{\mathrm{adj}(sI - A)}{\det(sI - A)} = \frac{1}{s^2+4s+6} \cdot \begin{bmatrix} s & -1.5 \\ 4 & s+4 \end{bmatrix}$$

$$T(s) = \frac{1}{s^2+4s+6} \begin{bmatrix} 1.5 & 0.625 \end{bmatrix} \begin{bmatrix} s & -1.5 \\ 4 & s+4 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} =$$

$$= \frac{1}{s^2+4s+6} \begin{bmatrix} 1.5s + 2.5 & 0.625s + 0.25 \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} =$$

$$= \frac{3s+5}{s^2+4s+6}$$

## Converting a transfer function to state space

One advantage of the state space representation is that it can be used for the simulation of physical systems on the digital computer.

If we want to simulate a system that is represented by a transfer function, we must first convert the transfer function representation to state space.

- At first we select a set of state variables (**phase variables**) where each subsequent state variable is defined to be the derivative of the previous state variable.
- We begin by showing how to represent a general, $n$th order, linear differential equation with constant coefficients in state space in the phase-variable form.
- We will then show how to apply this representation to transfer function.

## Converting a transfer function to state space

Consider the differential equation:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \ldots + a_1\frac{dy}{dt} + a_0 y = b_0 u$$

- A convenient way to choose state variables is to choose the output, $y(t)$, and its $(n-1)$ derivatives as the state variables.
- This choice is called the **phase-variable choice**.
- Choosing the state variable $x_i$, we get:

$$x_1 = y$$
$$x_2 = \frac{dy}{dt}$$
$$x_3 = \frac{d^2y}{dt^2}$$
$$\vdots$$
$$x_n = \frac{d^{n-1}y}{dt^{n-1}}$$

- ... and differentiating both side:

$$\dot{x}_1 = \frac{dy}{dt}$$
$$\dot{x}_2 = \frac{d^2y}{dt^2}$$
$$\dot{x}_3 = \frac{d^3y}{dt^3}$$
$$\vdots$$
$$\dot{x}_n = \frac{d^n y}{dt^n}$$

## Converting a transfer function to state space

Consider the differential equation:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_1 \frac{dy}{dt} + a_0 y = b_0 u$$

- ... and differentiating both side:

$$\begin{aligned}
\dot{x}_1 &= \frac{dy}{dt} \\
\dot{x}_2 &= \frac{d^2 y}{dt^2} \\
\dot{x}_3 &= \frac{d^3 y}{dt^3} \\
&\vdots \\
\dot{x}_n &= \frac{d^n y}{dt^n}
\end{aligned}$$

## Converting a transfer function to state space

Consider the differential equation:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1}y}{dt^{n-1}} + \ldots + a_1\frac{dy}{dt} + a_0 y = b_0 u$$

- Substituting the definitions, the state equations are evaluate as

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= x_3 \\
&\vdots \\
\dot{x}_{n-1} &= x_n \\
\dot{x}_n &= -a_0 x_1 - a_1 x_2 - \ldots - a_{n-1}x_n + b_0 u
\end{aligned}$$

- In vector-matrix form, it becomes:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \overset{A}{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots 1 & \\ -a_0 & -a_1 & -a_2 & -a_3 & -a_4 & -a_5 & \cdots & -a_{n-1} \end{bmatrix}} \overset{X}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}} + \overset{B}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ b_0 \end{bmatrix}} u$$

- This is the **phase-variable form of the state equations**.

- Since the solution to the differential equation is $y(t)$, or $x_1$, the output equation is:

$$y = \overset{C}{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}$$

## Converting a transfer function to state space

In summary, to convert a transfer function into state equations in phase-variable form...

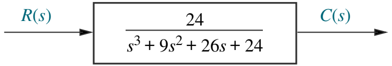
- First convert the transfer function to a differential equation by cross-multiplying and taking the inverse Laplace transform, assuming zero initial conditions.
- Then, represent the diff. equation in state space in phase variable form.

**Example**: Find the state space representation in phase-variable form for the transfer function below.

$$R(s) \longrightarrow \boxed{\frac{24}{s^3 + 9s^2 + 26s + 24}} \longrightarrow C(s)$$

## Converting a transfer function to state space

**Example**: Find the state space representation in phase-variable form for the transfer function shown here.




1. Find the associated differential equation. Being,

$$\frac{C(s)}{R(s)} = \frac{24}{s^3 + 9s^2 + 26s + 24}$$

$$(s^3 + 9s^2 + 26s + 24)C(s) = 24R(s)$$

⇝ The corresponding differential equation is found by taking the inverse Laplace transform, assuming zero initial conditions:

$$\dddot{c} + 9\ddot{c} + 26\dot{c} + 24c = 24r$$



$$\frac{C(s)}{R(s)} = \frac{24}{s^3+9s^2+26s+24}$$

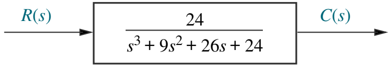
$$(s^3+9s^2+26s+24)C(s) = 24\,R(s)$$

$$\frac{d^3c}{dt^3} + \frac{9d^2c}{dt^2} + 26\frac{dc}{dt} + 24c = 24r$$

$$\frac{d^3c}{dt^3} = -\frac{9d^2c}{dt^2} - 26\frac{dc}{dt} - 24c + 24r$$

## Converting a transfer function to state space

**Example**: Find the state space representation in phase-variable form for the transfer function shown here.




2. Select the state variables. Choosing the state variables as successive derivatives, we get:

$$x_1 = c$$
$$x_2 = \dot{c}$$
$$x_3 = \ddot{c}$$

⇝ Differentiating both sides and making use of the previous definitions, we obtain the combined state and output equations:

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dot{x}_3 = -24x_1 - 26x_2 - 9x_3$$



2. (cont.) In vector-matrix form:

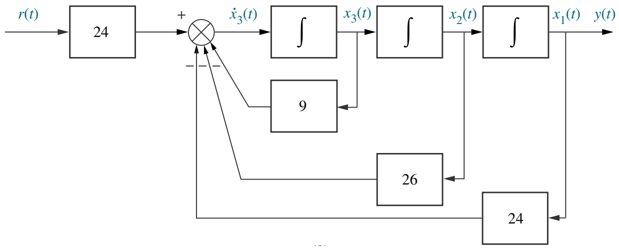
$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -24 & -26 & -9\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}0\\ 0\\ 24\end{bmatrix} r$$

$$y = \begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}$$

⇝ Notice that the third row of the system matrix has the same coefficients as the denominator of the transfer function but negative and in reverse order.

## Converting a transfer function to state space

We can create an equivalent block diagram of the system to help visualize the state variables.




- We draw three integral blocks and label each output as one of the state variables, $x_i(t)$.
- Since the input to each integrator is $x_i(t)$, we use the equations to determine the combination of input and output signals to each integrator.

## Converting a transfer function to state space

### Matlab/Octave

```
% From the transfer function to the state space
num=24;
den=[1 9 26 24];

% Convert the transfer function to SS form
[A,B,C,D]=tf2ss(num,den);

% Alternative realisation
T=tf(num,den);
Tss=ss(T);
```



```
import control as ctrl

% From the transfer function to the state space
### From transfer function to state space
num = 24
den = [1, 9, 26, 24]

sys1 = ctrl.tf2ss(num, den)

### Alternatively
TF_ss=ctrl.tf(num,den)
sys2 = ctrl.ss(TF2)
```

## Converting a transfer function to state space

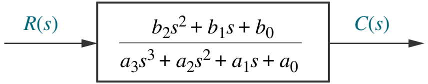
**Exercise L7E2**: Find the state-space representation in phase-variable form for the system below:

$$R(s) \longrightarrow \boxed{\frac{100}{s^4 + 20s^3 + 10s^2 + 7s + 100}} \longrightarrow C(s)$$

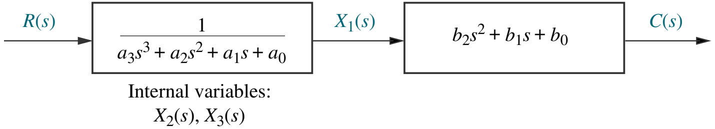
ToDo !!

## Converting a transfer function to state space

If a transfer function with **the order of the numerator is less than the order of the denominator**, the numerator and denominator can be handled separately.

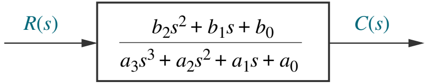



- First we separate the transfer function into two cascaded transfer functions:
  - The first is the **denominator**
  - The second is the **numerator**

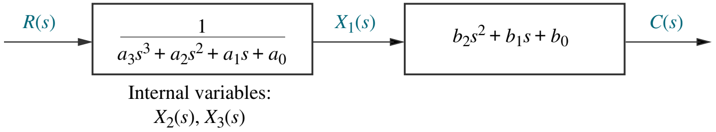


## Converting a transfer function to state space

If a transfer function with **the order of the numerator is less than the order of the denominator**, the numerator and denominator can be handled separately.




- The **first transfer function** with just the denominator is converted to the phase-variable representation in state space as demonstrated before.
  - Phase variable $x_1$ is the output, and the rest of the phase variables are the internal variables of the first block.



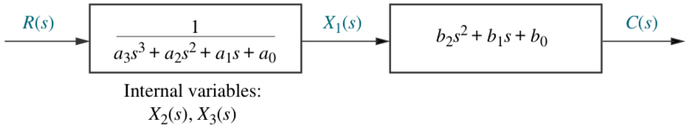



- The **second transfer function** with just the numerator yields:

$$Y(s) = C(s) = (b_2s^2 + b_1s + b_0)X_1(s)$$

- After taking the inverse Laplace transform with zero initial conditions, we get

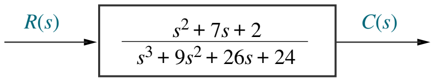
$$y(t) = b_2\frac{d^2x_1}{dt^2} + b_1\frac{dx_1}{dt} + b_0x_1$$



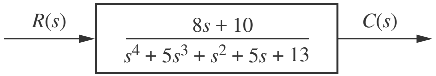

- Hence, the second block simply forms a specified linear combination of the state variables developed in the first block.
- From another perspective, **the denominator of the transfer function yields the state equations, while the numerator yields the output equation**.

## Converting a transfer function to state space

**Exercise L7E3**: Find the state space representation in phase-variable form for the system below:




**Exercise L7E4**: Find the state space representation in phase-variable form for the system below:




ToDo!!