Introduction to Automatic Control

1 - Introduction & Modelling

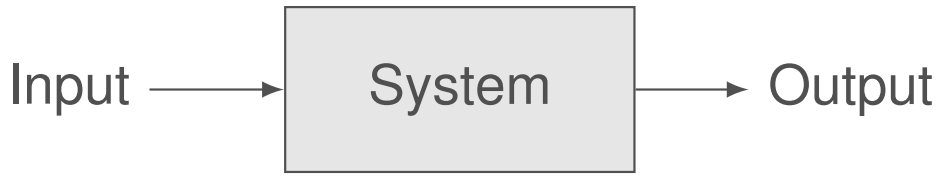
Dr Sebastian East

University of BRISTOL

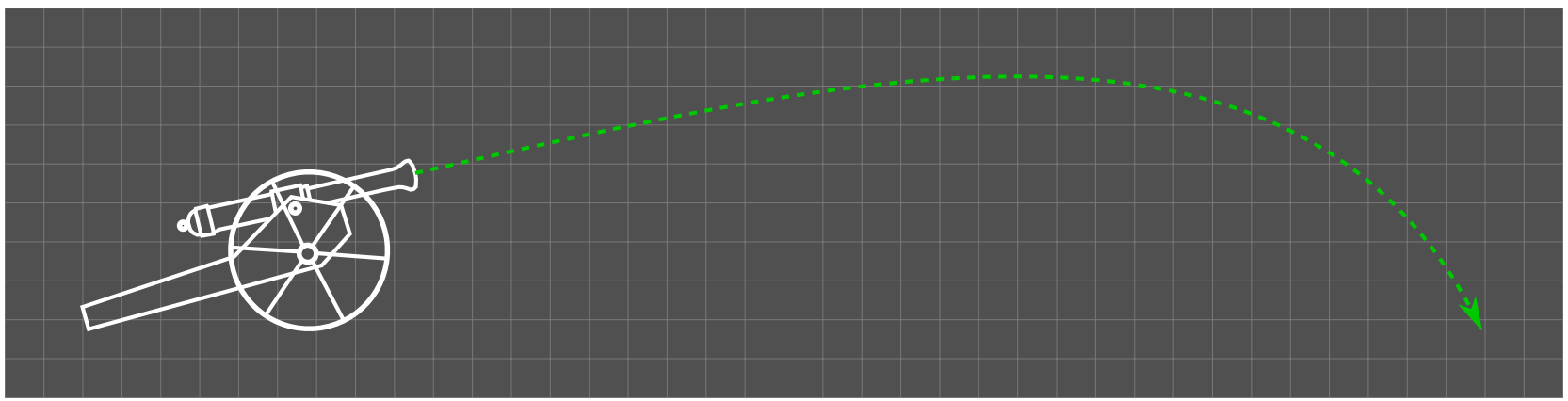
Contents

Open Loop Systems

- **Open loop system:** input is *independent* of the output:

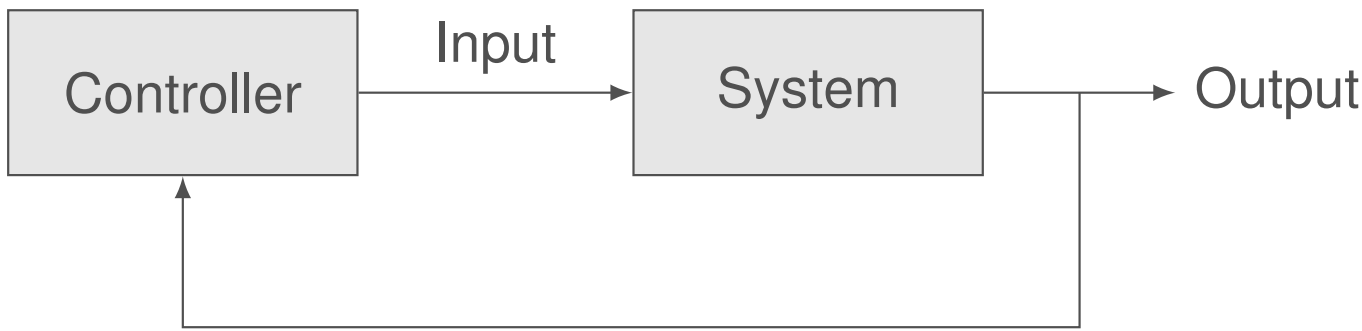


- **Example:** Cannon
  - Input = Barrel orientation
  - Output = Range



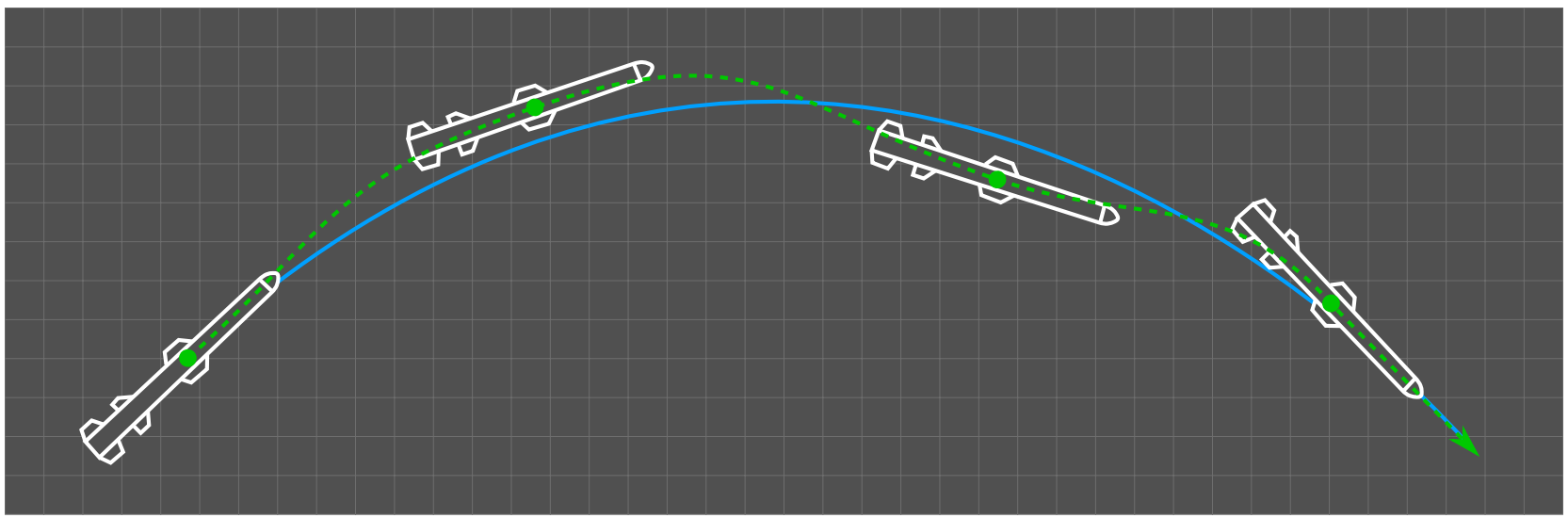
Closed Loop Systems

- **Closed loop system:** input is dependant on the output.
- Key mechanism is *feedback*.
- Input is determined by a *controller* based on the output.



Feedback Example

- Missile
  - Inputs = Thrust, Steering
  - Outputs = Speed, Orientation, Position
- Controller **continuously** updates the inputs to ensure that the rocket follows the required trajectory.
- **Robust** to uncertainty



Feedback Control in Aerospace

- Air/spacecraft are generally *unstable*
- **Wright Brothers (1903):** invented a *controllable* aircraft.

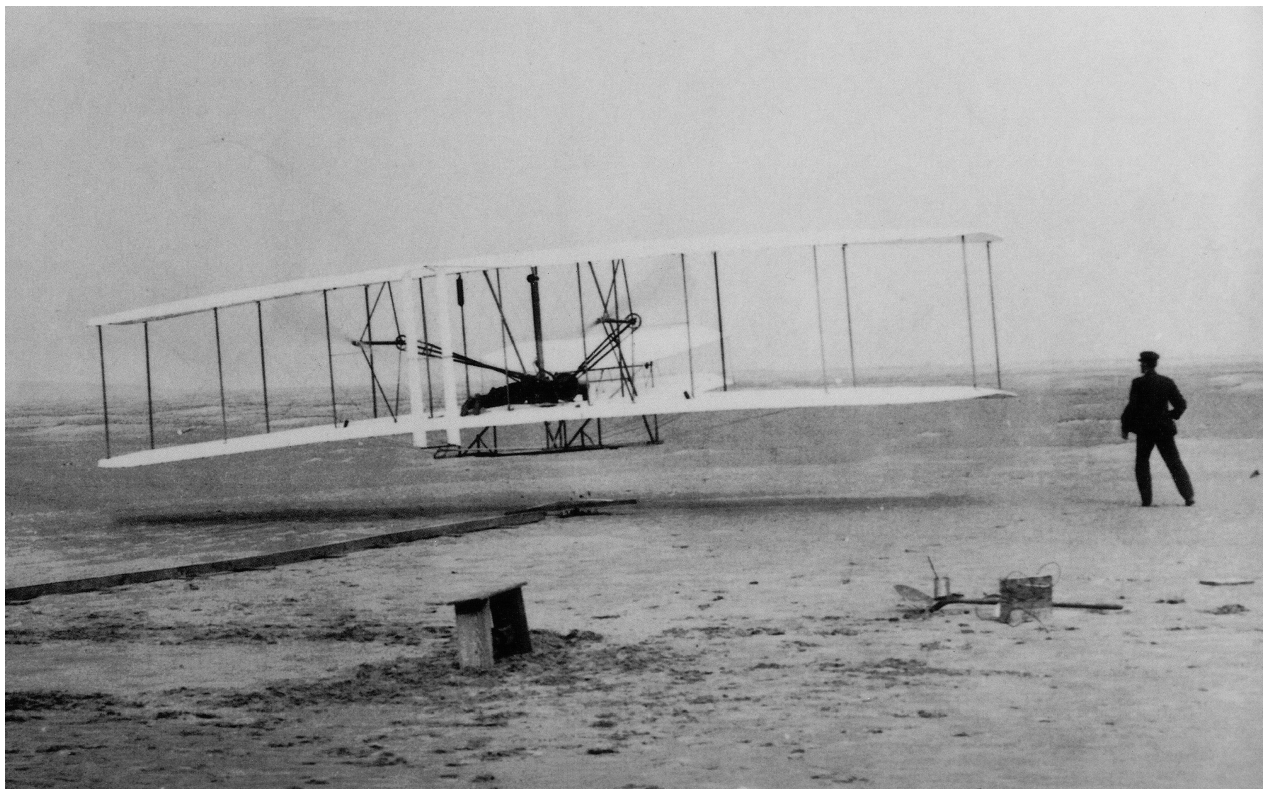
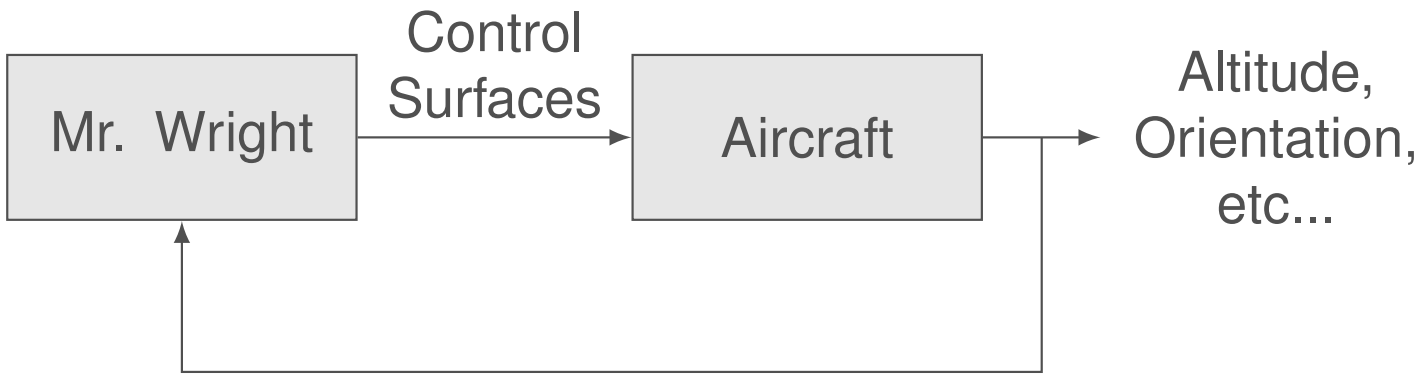


Image downloaded from `www.nasa.gov/multimedia/imagegallery/image_feature_976.html` on 15/07/2021.

Automatic Control

- The controller (pilot) is a **human**.



- **Automatic control** is where the human-in-the-loop is replaced by an **engineered system**.
  - \+ Humans exceptionally good at high level reasoning
  - – Still many (sub-)systems where human control is suboptimal

Examples

- Autopilot
  - Challenging (and taxing) to maintain completely level flight

- Fly-by-wire
  - Relaxing stability margins improves manoeuvrability

Images downloaded from `airbus.com/aircraft/passenger-aircraft/cockpits.html` and `worldwarwings.com/wp-content/uploads/2016/02/su-loop1-735x413.jpg` on 15/07/2021.

Examples Continued

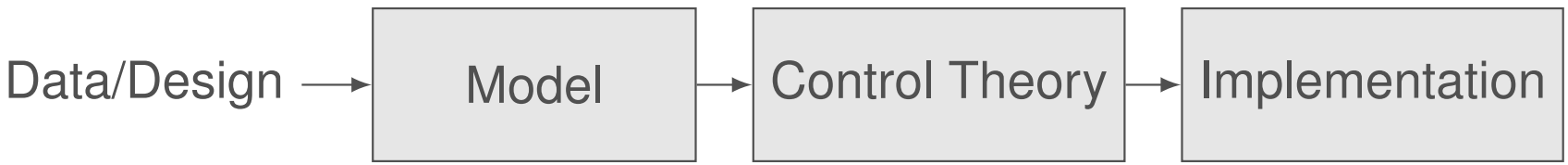
- Multi-rotorcraft
  - Pilot cannot control the rotor speeds directly

- Rockets
  - Longitudinally unstable (dynamically similar to the inverted pendulum)

Images downloaded from `transportup.com/wp-content/uploads/2020/06/download.jpg` and `nasa.gov/images/content/739339main_Space_Shuttle_Challenger_04-04-1983.jpg` on 15/07/2021.

Control System Engineering

- *Process* by which a controller is designed for a given system.



- Mathematical modelling is an integral component
- **Control theory** is the mathematical toolbox used to analyse and design feedback systems
- Hardware implementation is an important consideration
- In practice, it is an *iterative*, *holistic* process.

Control Theory

- **Control Theory** is the mathematical toolbox used to analyze and design control systems.

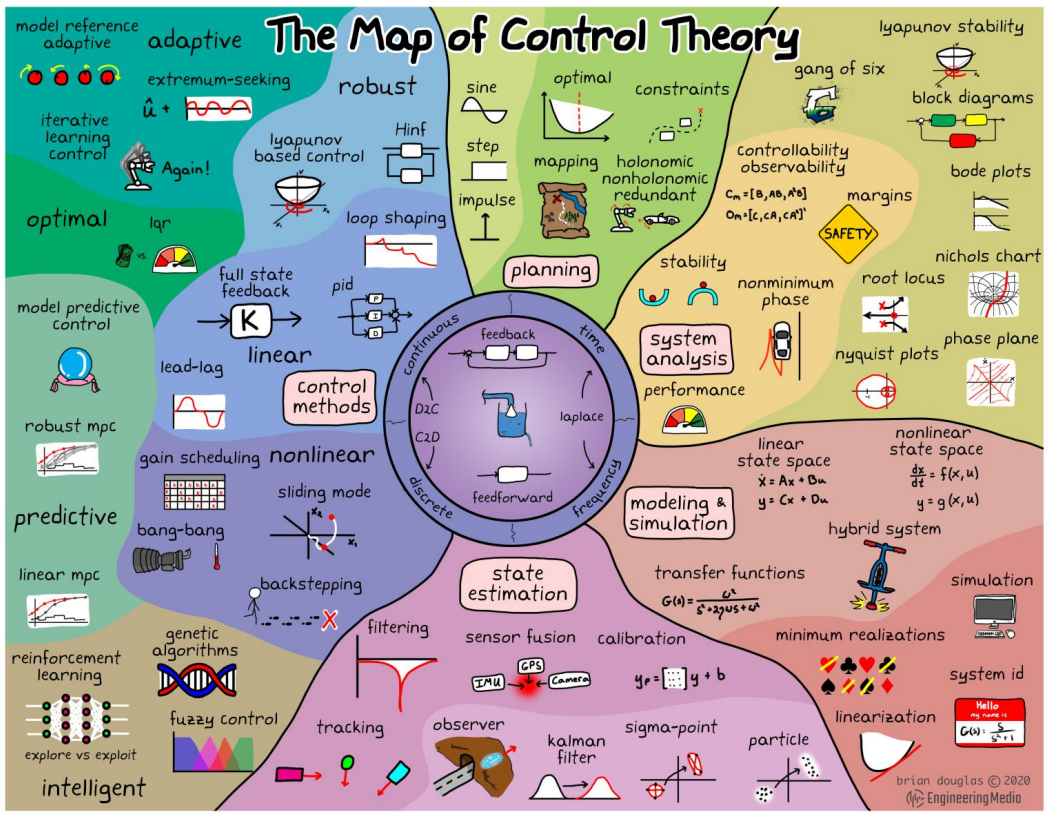


Image downloaded from `engineeringmedia.com/map-of-control` on 04/06/2021. Credit to Brian Douglas.

In this course

- You will learn how to design controllers for a wide range of aerospace systems:

- **This term**
  1. Feedback Systems

- **Next Term**
  2. Single-input/single-output controller design
  3. Multi-input/multi-output controller design
  4. Optimal Controller design

Contents

About Me

- Dr Sebastian East
- Joined UoB $\approx$ 1.5 years ago from University of Oxford
  - Second year teaching this unit.
  - Control part expanded into TB1.
  - Feedback **strongly** encouraged[1].
- Research interests in Control Theory, Numerical Optimization, and Machine Learning.

[1] ...enforced?

Content Delivery

Delivery

- Each week:
  - ‘Asynchronous’ - approx. 2 hours (equivalent) of video content on blackboard.
  - Problem sheet
  - ‘Synchronous’ - 1 hour lecture going through examples, and opportunity to ask questions in person. **No more hybrid**.

Assessment

- 50% of Timed Assessment at End of Term

Schedule - TB1

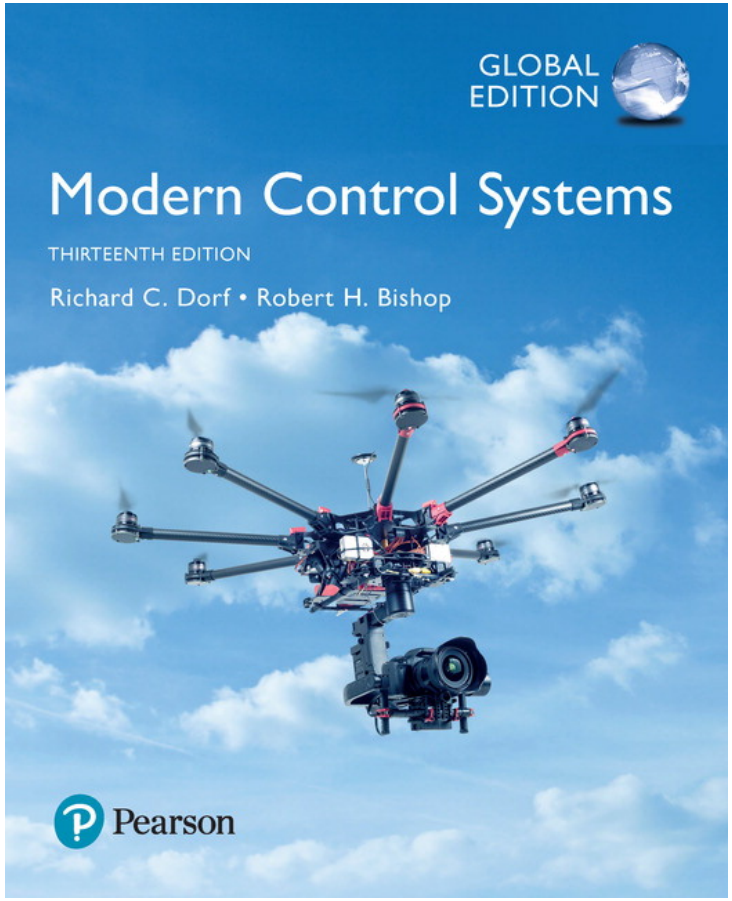
Week	Theme	Topics
1 2 3 4 5 6 7 8		Steve
9 10 11 12	Feedback	Intro & Mathematical Modelling PID Control & Block Diagrams Stability Analysis Performance & Sensitivity

Course Textbook

- **Modern Control Systems**
  Dorf & Bishop

- All course theory can be found in this textbook
  - Reordered for this course.
  - Textbook contains additional content not covered here.

- Available online through UoB Library.
- **10 Copies** in Queens Library.

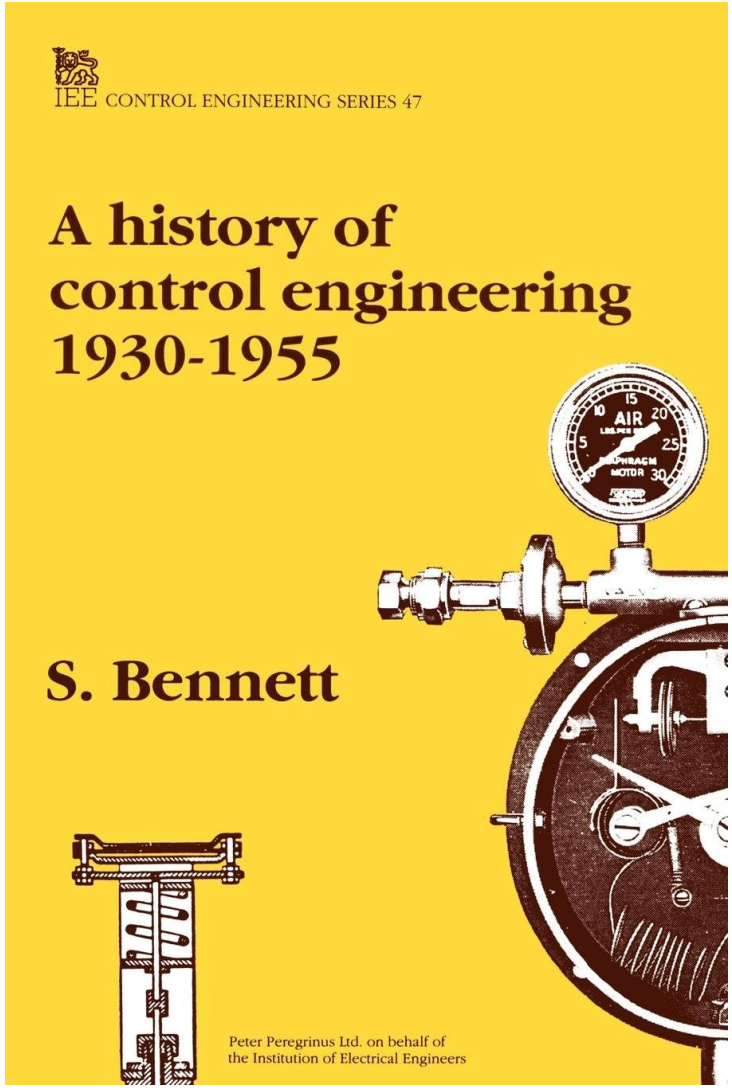


Alternative Textbooks

- Similar content can be found in any textbook on control engineering
- Many can be found in Queen's Library, Bay 13.
  - Nise
    Control Systems Engineering
  - Franklin, Powell, Emami-Naeini
    Feedback Control of Dynamic Systems
  - Ogata
    Modern Control Engineering

Recommended Reading

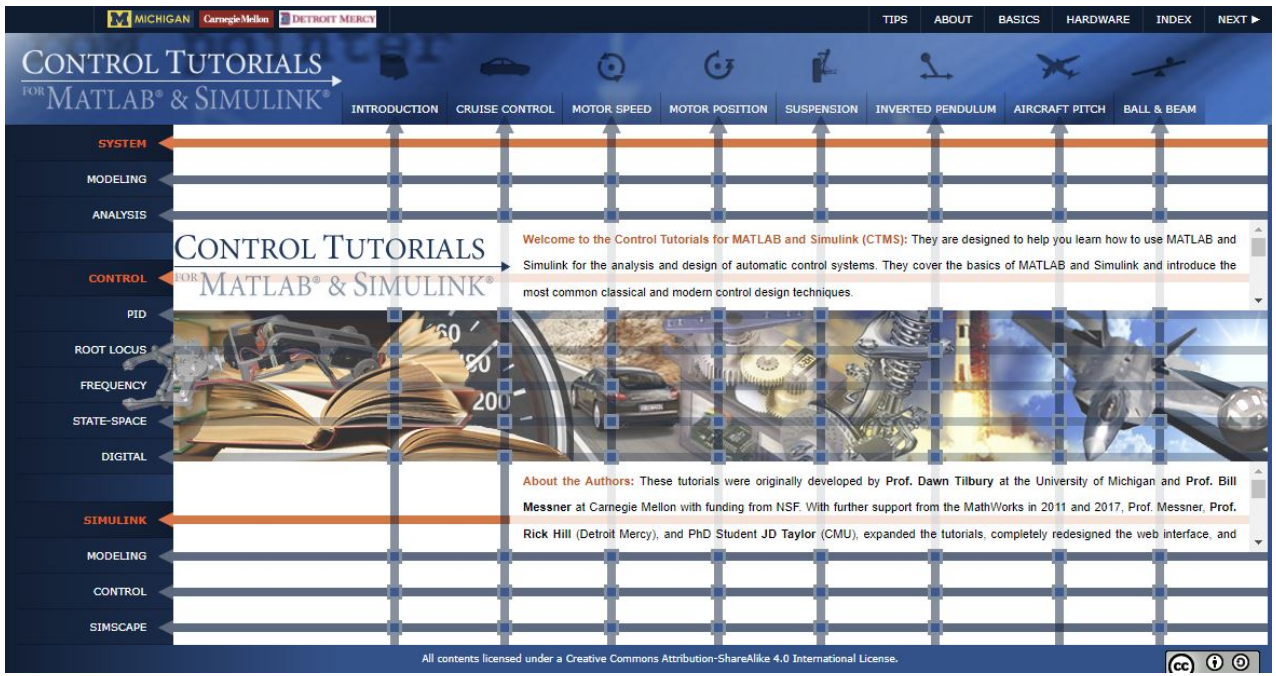
- **History of Control Engineering** Bennet
- **Historical** overview of control.
- Not ‘necessary’ for course.
- **1 copy of each** in Queens Library.



Online Tutorials

- **Control Tutorials for Matlab & Simulink**[2]
  Michigan, Carnegie Mellon, Detroit Mercy.

- Additional examples of
  - Modelling, System Analysis, Controller Design, Simulink.
- Helpful for **coursework**.



[2] ctms.engin.umich.edu

Contents

Differential Equations

- Differential equations are common mathematical tool for modelling engineering systems.
- Equation that is solved to obtain a **function** using knowledge of its derivatives

$$Ax = b \quad x = ? \quad \Rightarrow \quad \frac{\mathrm{d}x}{\mathrm{d}t} = b(t) \quad x(t) = ?$$

- Control theory is primarily concerned with systems that are modelled using **ODEs**:
  - Unknown function depends on derivatives w.r.t. single variable
  - Often used to describe the behaviour of **lumped states**.
  - Example: **Newton's Second Law**

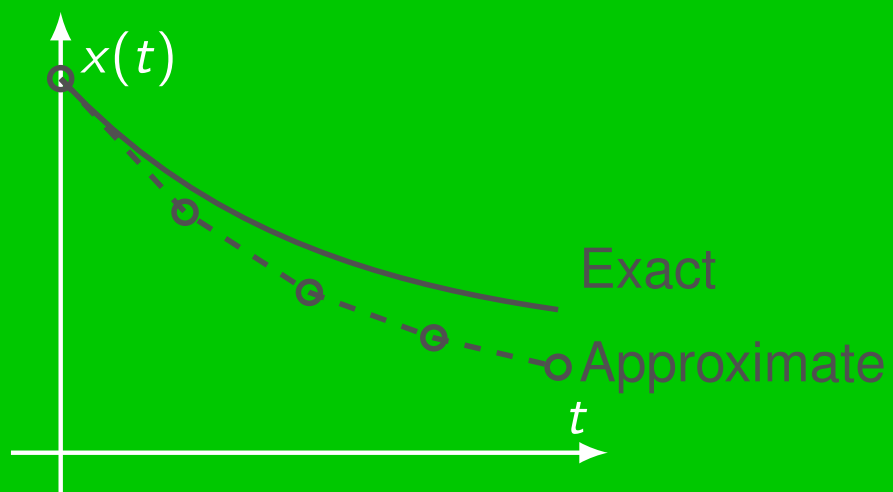
$$m\ddot{x} = \sum_i f_i$$

Numerical Solution of ODEs

- In general, *no analytical solution exists*
- Solution can be approximated using *numerical integration*

Euler's method

$$x(t + \delta_t) := x(t) + \delta_t \frac{dx}{dt}$$



Solving ODEs in Matlab

- Create a function of first derivatives, save as `dynamics.m`

```
function [xdot] = dynamics(t, x)
xdot = ...
end
```

- Simulate using `ode45` function

```
[t, x] = ode45(@dynamics, [0, T], [x_0]);
```

- `ode45` uses an *adaptive step, 4th order Runge-Kutta method*.

Numerical methods are *not* examinable in this course.

First Order Equivalent System

- `ode45` can only be used to solve n-dimensional, first order ODE.
- An n-th order ODE can be converted to n first order coupled ODEs by creating new variables for the lower order derivatives.

The third order ODE

$$a\dddot{x} + b\ddot{x} + c\dot{x} + dx = 0$$

is equivalent to the three ODEs

$$\dot{x}_0 = x_1, \quad \dot{x}_1 = x_2, \quad \dot{x}_2 = -\frac{1}{a}\left[bx_2 + cx_1 + dx_0\right]$$

using the change of variables

$$x_0 = x, \quad x_1 = \dot{x}, \quad x_2 = \ddot{x}, \quad \dot{x}_2 = \dddot{x}.$$

Example: Lorenz System

- Represents fluid flow in the atmosphere
- Example of a **chaotic** system
- Dynamics given by

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = x(\rho - z) - y$$
$$\dot{z} = xy - \beta z$$

where σ, ρ, and β are constants.

Example: Matlab Implementation

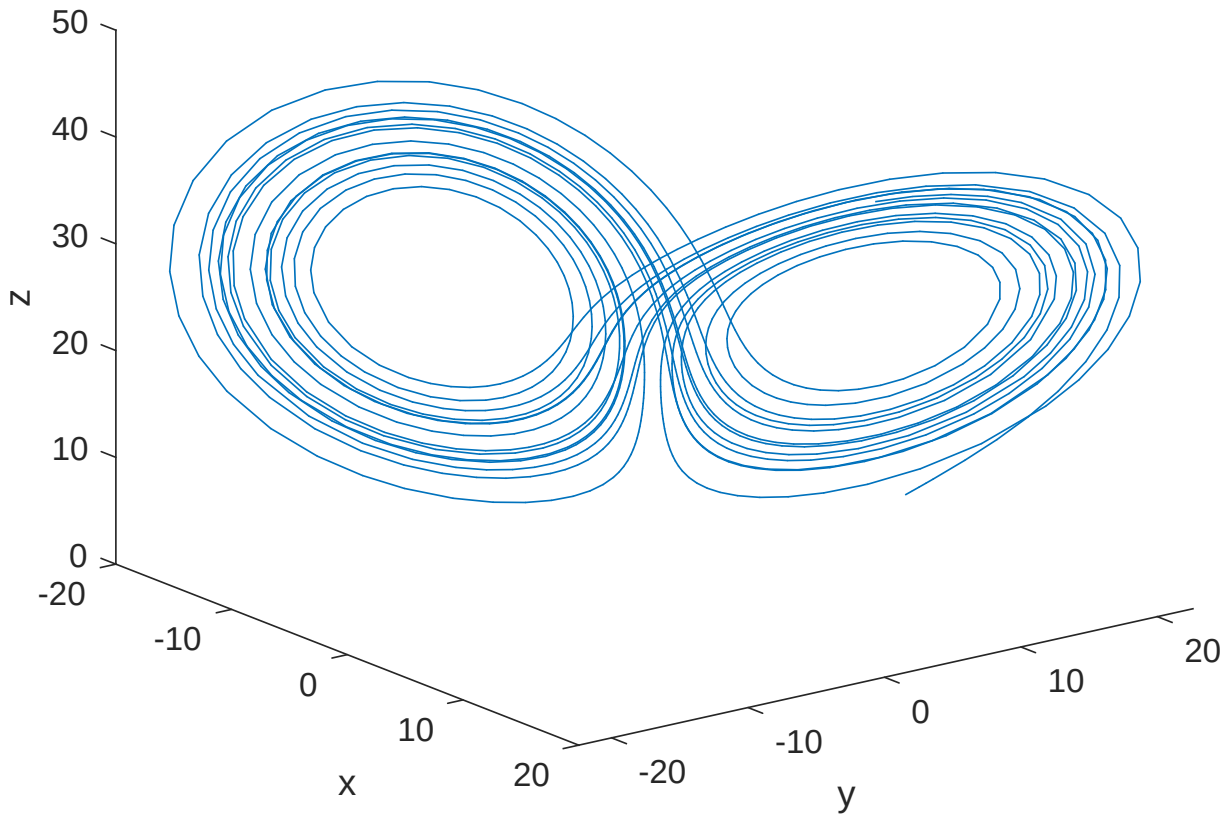
- Dynamics function with parameters $\sigma = 10$, $\rho = 28$, $\beta = 8/3$

```
function [vdot] = dynamics(t, v)
x = v(1); y = v(2); z = v(3);
sigma = 10; rho = 28; beta = 8 / 3;
vdot = [sigma * (y - x); x * (rho - z) - y; x * y - beta * z];
end
```

- Integrate over 20s, with $x(0) = 10$, $y(0) = 10$, $z(0) = 10$
- Plot in 3d

```
[t, v] = ode45(@dynamics, [0, 20], [10, 10, 10]);
plot3(v(:,1), v(:,2), v(:,3))
view([53 23]); xlabel('x'); ylabel('y'); zlabel('z');
```

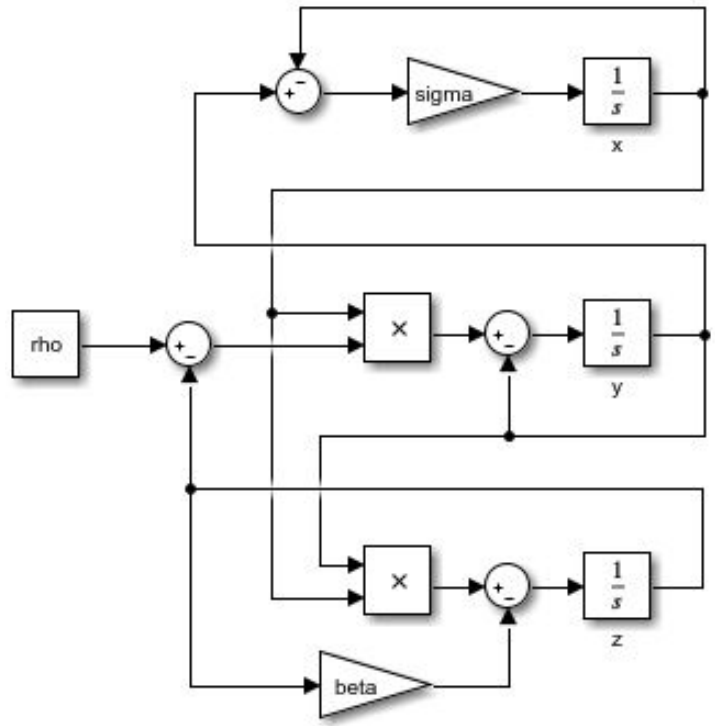
Example: Output



Simulink

$$\dot{x} = \sigma(y - x), \quad \dot{y} = x(\rho - z) - y, \quad \dot{z} = xy - \beta z$$

- Simulink is a **graphical** method for solving ODEs
- You should already have experience from **Sensors & Signals**.
- Much easier to implement controllers
  - PID controller block.



General Method

- General method for implementing ODEs in Simulink:

Given ODE(s):

1. For each ODE, rearrange to obtain highest order derivative ($\frac{\mathrm{d}^n x}{\mathrm{d}t^n}$).
2. Sequentially compose n integrator ($\frac{1}{s}$) blocks in Simulink
   - Output of ith block is the $(n-i)$th derivative.
3. Use block algebra to obtain input to first integrator.

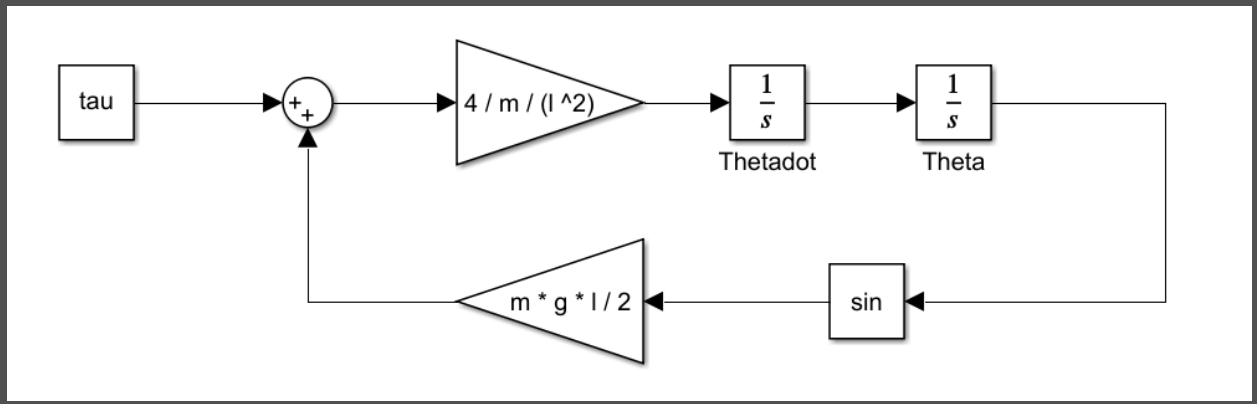
Example

- Dynamics of pendulum given by

$$m\left(\frac{l}{2}\right)^2 \ddot{\theta} = \frac{mgl}{2}\sin\theta(t) + \tau(t)$$

States: $\theta(t)$, $\dot{\theta}(t)$ Control input: $\tau(t)$

- Simulink model:



You can now complete problem 1

Contents

LTI ODEs

- The theory we will develop in this course is related *entirely* to **linear time-invariant** ODEs.

- LTI ODEs have the form

$$a_0 x(t) + a_1 x'(t) + a_2 x''(t) + \cdots + a_n x^{(n)}(t) = u(t)$$

  - Have a single, unique solution (subject to initial conditions)
  - $u(t)$ is the **input** function

- Linear ODEs have many important properties:
  + Solutions obey superposition
  + Can be solved algebraically using Laplace transform

(Very) Common LTI System Models

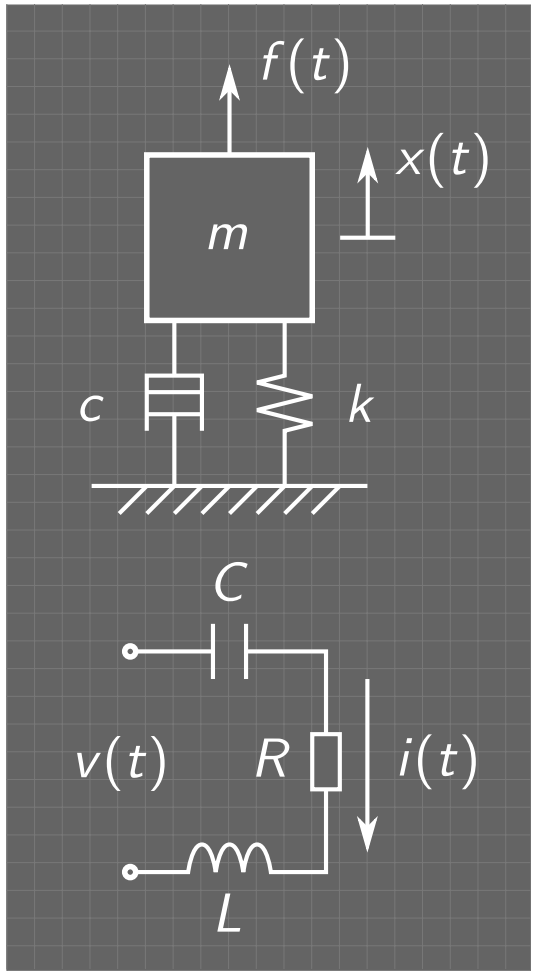
- Mechanical

$$m\ddot{x} = -c\dot{x} - kx(t) + f(t)$$

- Electrical

$$L\frac{di}{dt} + Ri(t) + \frac{1}{C}\int_0^t i(\tau)d\tau = v(t)$$

- Complex systems are often modelled using many of these simple representations



Solving Linear ODEs

- Many methods for solving **specific** forms of linear ODE
- Laplace transform is a very **general** method for solving LTI ODEs
- Reduces solution to an **algebraic problem**

Laplace Transform

$$F(s) = \mathcal{L}[f(t)] := \int_0^\infty e^{-st} f(t)\, \mathrm{d}t$$

Good video on intuition behind Laplace Transform at `https://www.youtube.com/watch?v=ZGPtPkTft8g`.

Transfer Functions

- Recall that the solution of a LTI system is given by the **convolution** of the input function and **impulse response**, $g(t)$:

$$x(t) = \int_{-\infty}^{\infty} g(\tau)u(t-\tau)\mathrm{d}\tau$$

- The Laplace transform of a convolution integral results in **multiplication**[3]:

$$X(s) = G(s)U(s)$$

- $G(s)$ is the **transfer function** of the system.

Given system's transfer function $G(s)$:

1. Take Laplace transform of input function
2. Multiply $G(s)$ with $U(s)$
3. Take inverse Laplace transform to obtain $x(t)$

[3] See if you can remember/prove this.

Obtaining the Transfer Function

- Recall that, assuming that the **initial conditions are zero**:

$$\mathcal{L}\left[\frac{\mathrm{d}f}{\mathrm{d}t}\right] = sF(s), \quad \mathcal{L}\left[\int_0^t f(\tau)\,\mathrm{d}\tau\right] = \frac{1}{s}F(s)$$

The transfer function can be obtained replacing all derivatives (integrals) with $sF(s)$ ($F(s)/s$), then rearranging terms for $\frac{X(s)}{U(s)}$

Example

The transfer function for the LTI system

$$a\ddot{x} + b\dot{x} + cx = d\dot{u} + eu$$

is given by

$$G(s) = \frac{X(s)}{U(s)} = \frac{ds + e}{as^2 + bs + c}$$

LTI Systems with Matlab

- The Matlab `tf` function is useful for simulating LTI systems using Matlab:
  - `step` simulates the step response
  - `bode` generates the bode plot
  - More available in documentation
- **Example** The system

$$G(s) = \frac{ds + e}{as^2 + b + c}$$

can be analyzed with

```
system = tf([d, e], [a, b, c]);
step(system);
bode(system);
```

Simple Example

System given by

$$\ddot{x} + x = u$$

1. Take Laplace transform

$$s^2 X(s) + X(s) = U(s)$$

2. Rearrange for transfer function

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

3. Take Laplace transform of (step) input

$$U(s) = \frac{1}{s}$$

4. Determine output in s domain

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

5. Determine output in time domain (inverse Laplace transform)

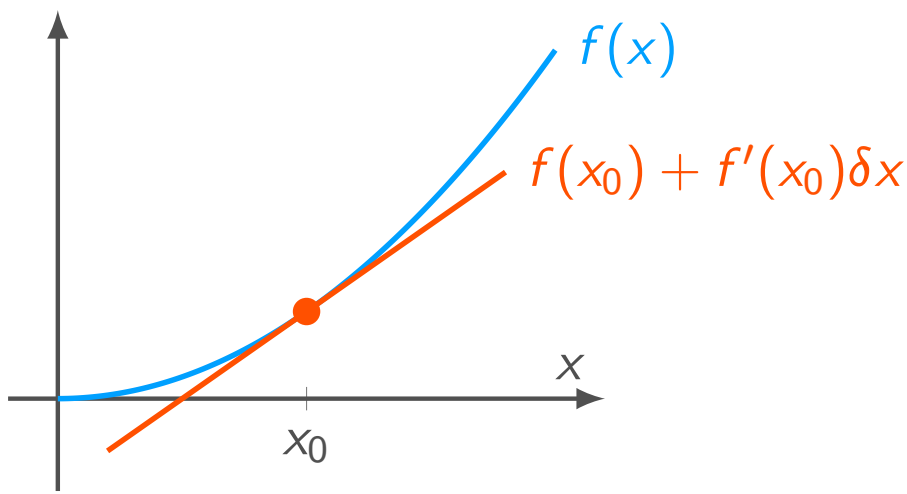
$$x(t) = 1 - \cos t$$

You can now complete problems 2 & 3

Contents

Linearization

- We are typically interested in controlling a system to hold a **set point**
  - e.g. autopilot holds level flight at given altitude
- Therefore, we can use a locally linear approximation of dynamics



- Only works functions that are **differentiable** at x_0

First Order Case

- Consider first order ODE with single input, u:

$$\dot{x} = f(x, u)$$

- We are interested in the dynamics close to a set-point (x_0, u_0).
  - x_0 typically has some physical significance (e.g. level flight)
  - u_0 typically chosen so that $f(x, u) = 0$ (i.e. trim condition)

- Approximate f with first terms of Taylor's series:

$$\begin{aligned}\dot{x} &\approx f(x_0, u_0) + \frac{\partial f(x_0, u_0)}{\partial x}(x - x_0) + \frac{\partial f(x_0, u_0)}{\partial u}(u - u_0) \\ &= \text{const.} + \frac{\partial f(x_0, u_0)}{\partial x}x + \frac{\partial f(x_0, u_0)}{\partial u}u\end{aligned}$$

- The R.H.S is a linear ODE

Cancelling Disturbance Term

- Want to choose u_0 so that $f(x_0, u_0) = 0$, as then the constant term (disturbance) can be removed using a change of variables:
  - $\Delta x := x - x_0$ ($\implies \Delta\dot{x} = \dot{x}$) and $\Delta u := u - u_0$

$$\dot{x} \approx f(x_0, u_0) + \frac{\partial f(x_0, u_0)}{\partial x}(x - x_0) + \frac{\partial f(x_0, u_0)}{\partial u}(u - u_0)$$

$$\implies \quad \Delta\dot{x} \approx \frac{\partial f(x_0, u_0)}{\partial x}\Delta x + \frac{\partial f(x_0, u_0)}{\partial u}\Delta u$$

Higher Order ODEs

- Higher order ODEs follow the same method:

$$\ddot{x} = f(\dot{x}, x, \dots) \implies$$
$$\ddot{x} \approx \text{const.} + \frac{\partial f(\dot{x}_0, x_0, \dots)}{\partial \dot{x}}(\dot{x} - \dot{x}_0) + \frac{\partial f(\dot{x}_0, x_0, \dots)}{\partial x}(x - x_0) + \dots$$

- Note that we have defined $\Delta x := x - x_0$ and $\Delta \dot{x} := \frac{\mathrm{d}}{\mathrm{d}t}\Delta x$, so $\Delta \dot{x} = \dot{x}$ (and $\Delta \dot{x} \neq \dot{x} - \dot{x}_0$). Therefore, for a second-order example with no input:

$$\ddot{x} \approx \frac{\partial f(\dot{x}_0, x_0)}{\partial \dot{x}}(\dot{x} - \dot{x}_0) + \frac{\partial f(\dot{x}_0, x_0)}{\partial x}(x - x_0) \implies$$
$$\Delta \ddot{x} \approx -\frac{\partial f(\dot{x}_0, x_0)}{\partial \dot{x}}\dot{x}_0 + \frac{\partial f(\dot{x}_0, x_0)}{\partial \dot{x}}\Delta \dot{x} + \frac{\partial f(\dot{x}_0, x_0)}{\partial x}\Delta x$$

General Method

Linearization

1. Given ODE f and set-point x_0
2. Choose u_0 so that $f(x_0, u_0) = 0$
3. Differentiate f w.r.t. each state derivative and control input
4. Evaluate partial derivatives at (x_0, u_0)
5. Change variables

Remember that linearized ODE solution is for Δx and Δu.

Need to add x_0 and u_0 to solutions to get x and u.

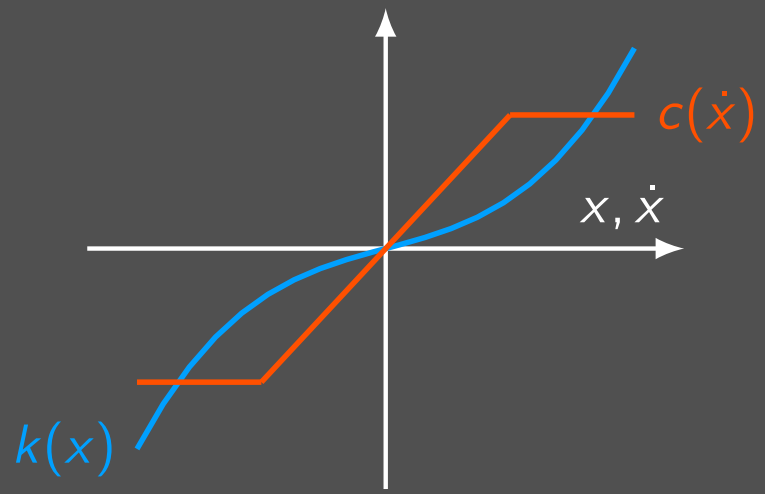
Example: Nonlinear Mass Spring Damper

1. Mass spring damper with nonlinear damping and stiffness

$$\ddot{x} = \frac{1}{m}\left[-k(x) - c(\dot{x}) + u\right],$$

where

$$k(x) = 0.5x + x^3, \quad c(\dot{x}) = \begin{cases} 2\dot{x} & -0.5 \leq \dot{x} \leq 0.5 \\ \text{sign}(\dot{x}) & \text{otherwise} \end{cases}$$



Set point $x = 1, \dot{x} = 0$

Example Continued

$$\ddot{x} = \frac{1}{m}\left[-k(x) - c(\dot{x}) + u\right],$$

2. Trim condition: $-k(1) - c(0) = -1.5 \implies u_0 = 1.5$.
3. Derivatives:

$$\frac{\partial f}{\partial x} = -\frac{1}{m}(0.5 + 3x^2) \quad \frac{\partial f}{\partial \dot{x}} = -\frac{2}{m} \quad \frac{\partial f}{\partial u} = \frac{1}{m}$$

4. Evaluate at set point $\dot{x}_0 = 0$, $x_0 = 1$, $u_0 = 1.5$:

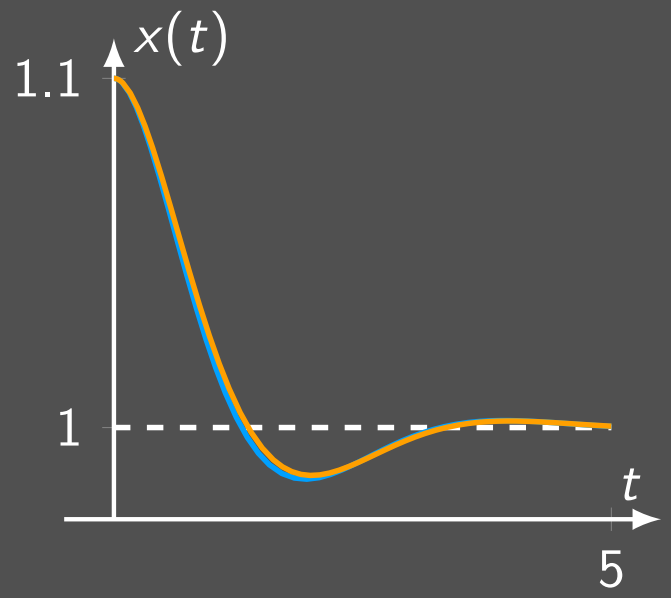
$$\frac{\partial f(\dot{x}_0, x_0, u_0)}{\partial x} = -\frac{3.5}{m} \quad \frac{\partial f(\dot{x}_0, x_0, u_0)}{\partial \dot{x}} = -\frac{2}{m} \quad \frac{\partial f(\dot{x}_0, x_0, u_0)}{\partial u} = \frac{1}{m}$$

5. Collect terms

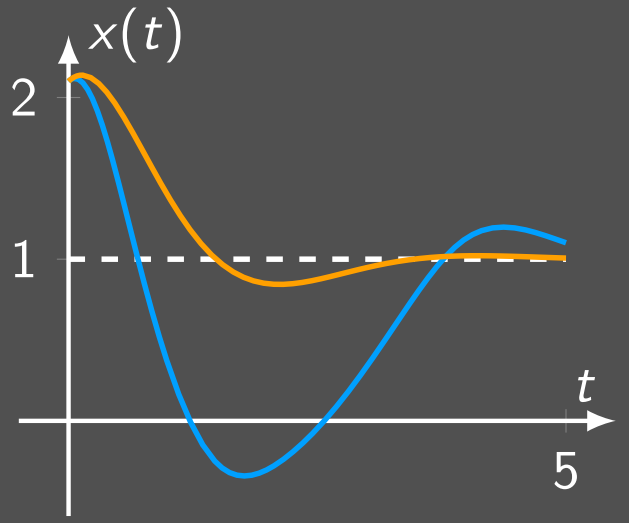
$$\Delta\ddot{x} \approx \frac{1}{m}\left[-3.5\Delta x - 2\Delta\dot{x} + \Delta u\right]$$

Visualised

- Initial conditions set at $x(0) = 1.1$, $\dot{x}(0) = 0$ ($\Delta x(0) = 0.1$, $\Delta \dot{x} = 0$).
- Simulated with $u(t) = 1.5$ ($\Delta u(t) = 0$).
- Linearized system behaviour very similar to nonlinear system.



- Initial conditions changed to $x(0) = 2.1$, $\dot{x}(0) = 0.6$ ($\Delta x(0) = 1.1$, $\Delta \dot{x} = 0.6$).
- Linearized system behaviour very **different** to nonlinear system.



You can now complete problems 4 & 5

Contents

Frequency Domain

- The information we can gain from the analyzing a system in the time domain is limited
- The **frequency response** provides a much more complete description of a system's input/output behaviour
  - Input function can be decomposed into sin/cosine waves **(Fourier transform)**.
  - System output is sum of response to each sin/cosine wave **(superposition)**

Frequency Response

- Recall again that

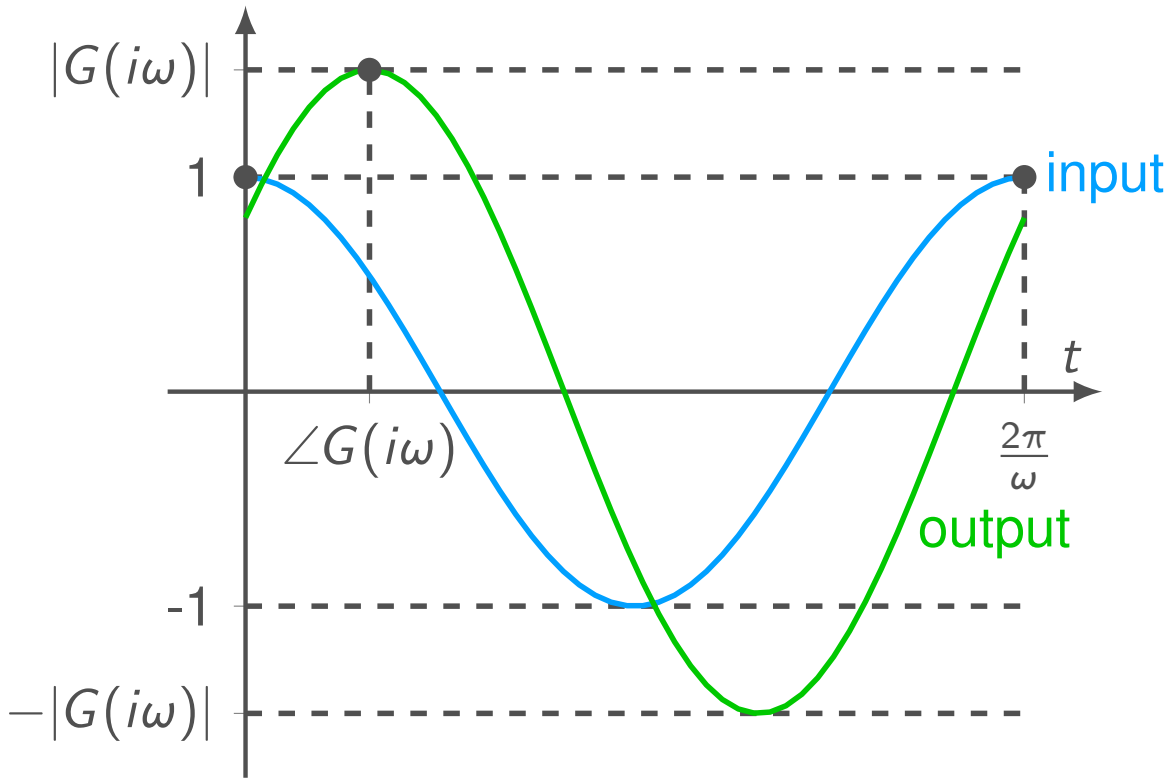
$$x(t) = \int_{-\infty}^{\infty} g(\tau)u(t-\tau)\mathrm{d}\tau,$$

- Assume that input is a unit sinusoid with frequency ω, $u(t) = e^{i\omega t}$.

$$x(t) = \int_{-\infty}^{\infty} g(\tau)e^{i\omega(t-\tau)}\mathrm{d}\tau = \underbrace{\int_{-\infty}^{\infty} g(\tau)e^{-i\omega\tau}\mathrm{d}\tau}_{G(s) \text{ with } s=i\omega} \cdot e^{i\omega t} = G(i\omega) \cdot e^{i\omega t}$$

- Therefore, $x(t)$ is also a sinusoid with frequency ω, but with **amplitude** $|G(i\omega)|$, and **out of phase** with the input by $\angle G(i\omega)$.

Visualised



Frequency response is given by the argument and magnitude of $G(i\omega)$.

Bode Plots

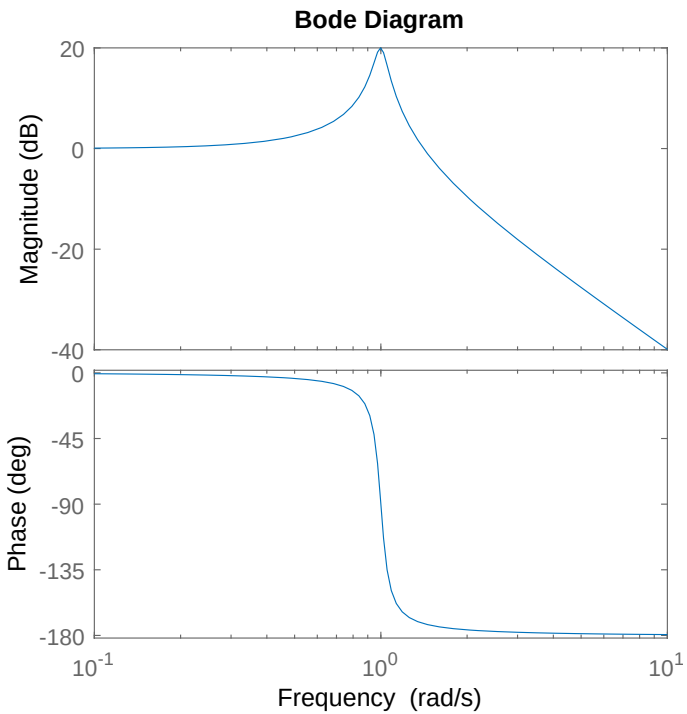
- A **Bode plot** is a graphical representation of a system's frequency response

- 2 Axes plotted against $\omega \in (0, \infty)$:

Axis 1 The amplitude ratio in decibels $(20 \log_{10} |G(i\omega)|)$

Axis 2 The phase difference in degrees $\left(\frac{180}{\pi} \angle G(i\omega)\right)$

- Can be generated in Matlab using the `bode` function.
  - Shown for $G(s) = 1/(s^2 + 0.1s + 1)$



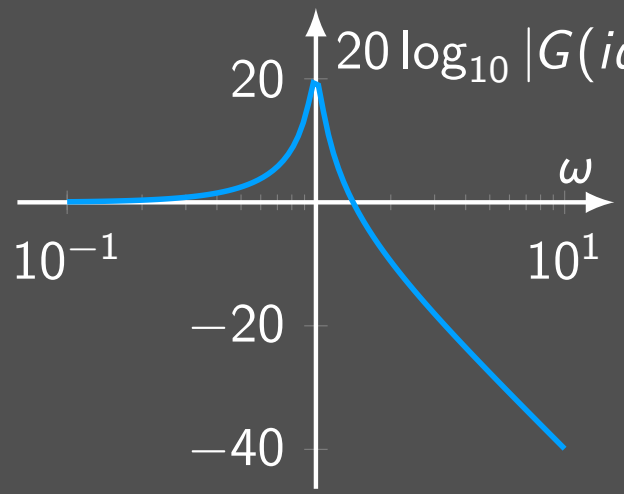
Example

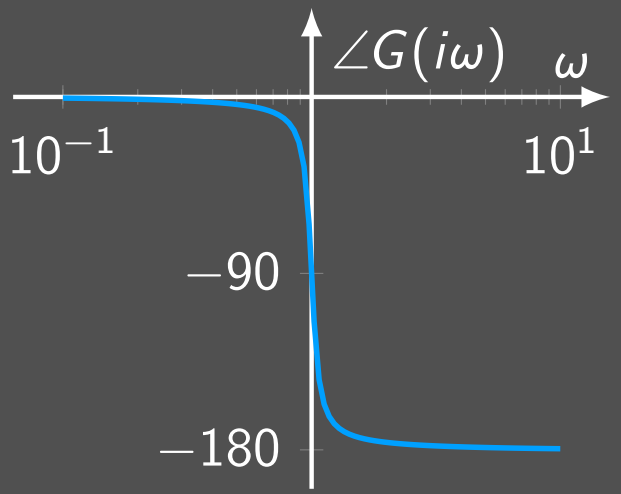
- Transfer function from previous slide $G(s) = \frac{1}{s^2+0.1s+1}$
- Amplitude of frequency response

$$|G(i\omega)| = \frac{1}{|1-\omega^2+i0.1\omega|} = \frac{1}{\sqrt{(1-\omega^2)^2+(0.1\omega)^2}}$$

- Phase of frequency response

$$\angle G(i\omega) = 0 - \angle(1-\omega^2+i0.1\omega) = -\tan^{-1}\left(\frac{0.1\omega}{1-\omega^2}\right)$$





You can now complete problems 6,7 & 8