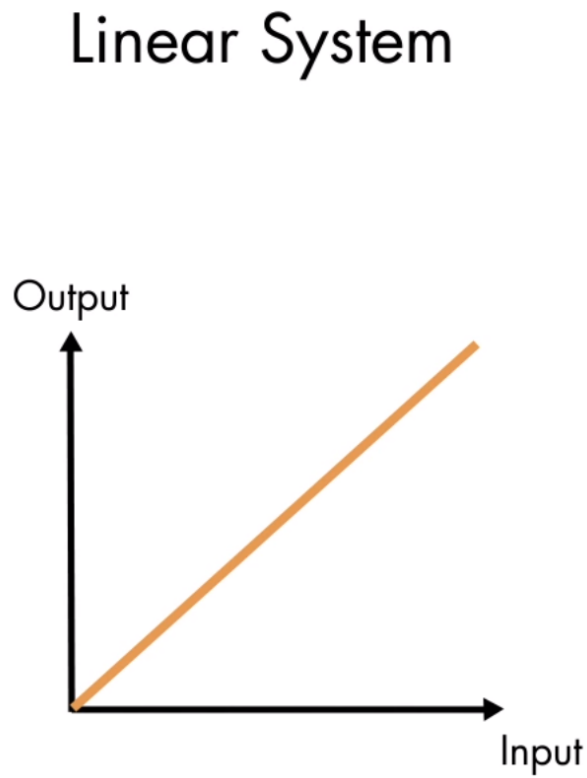
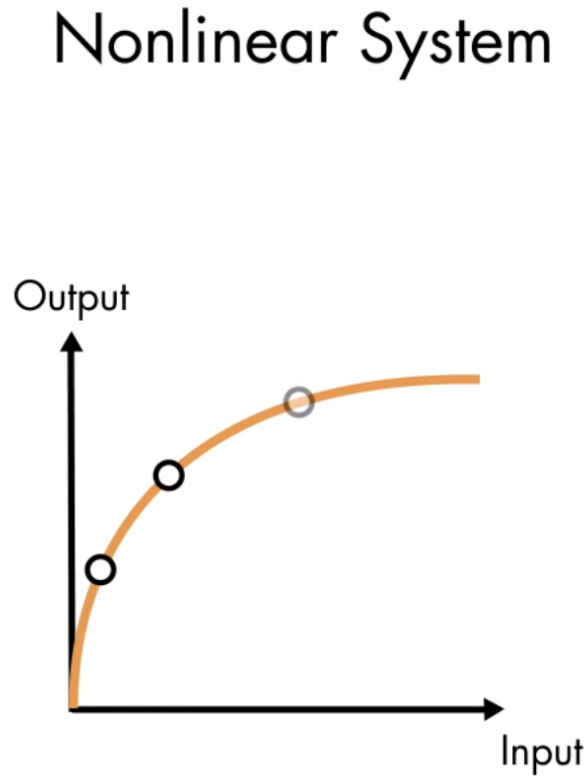
5ELEN018W - Robotic Principles
Lecture 9–10: More on Bode Plots, Linearisation and PID Control Implementation

Dr Dimitris C. Dracopoulos

Linear vs Nonlinear Systems

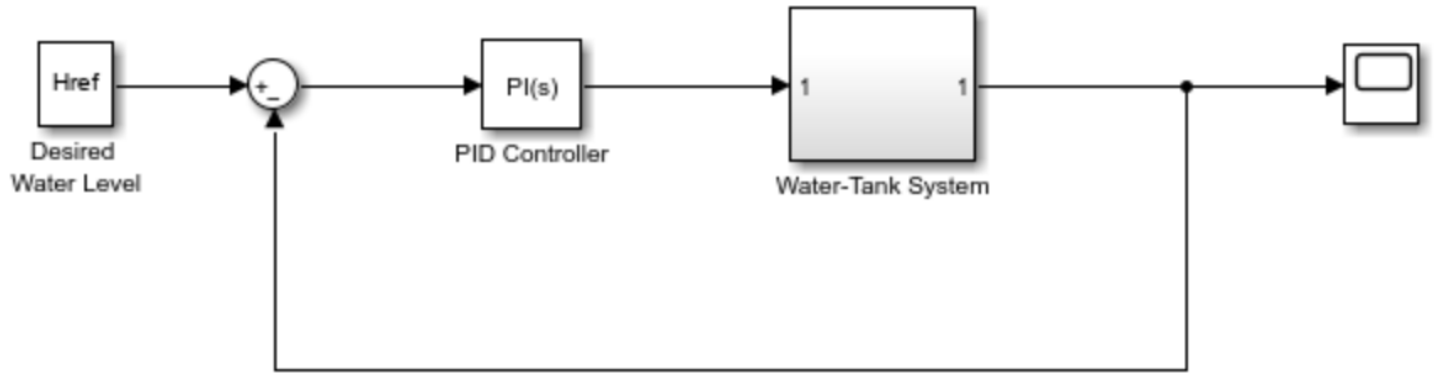


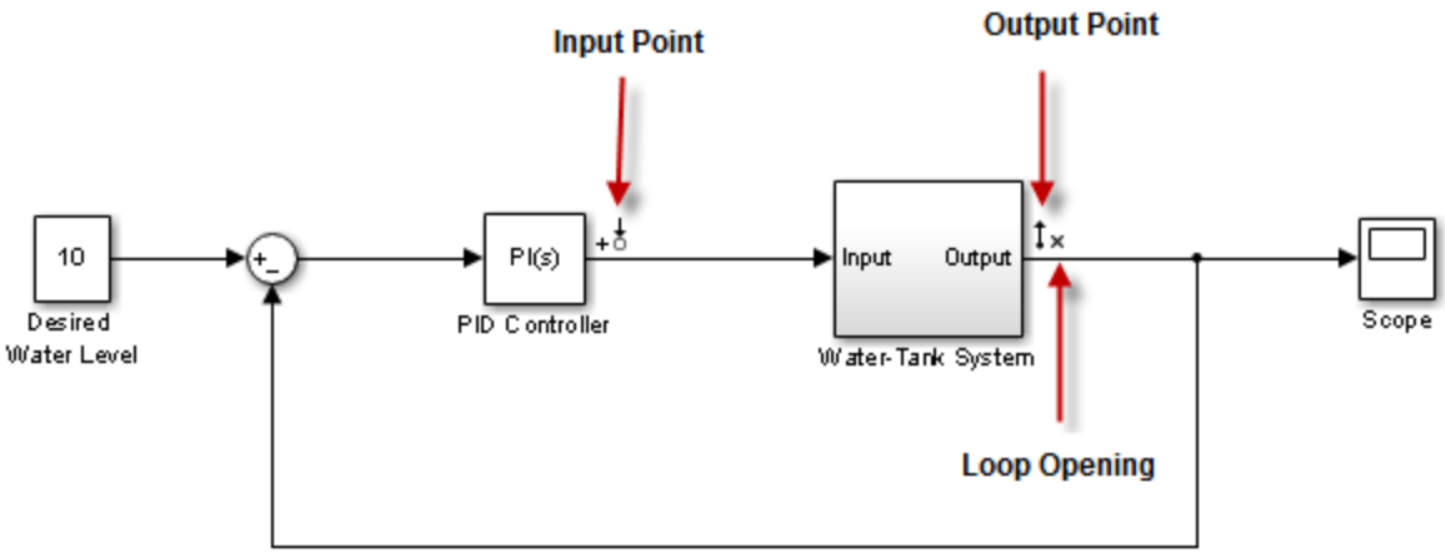


- In real life all systems are nonlinear, however many of them can be linearised about their operation point.

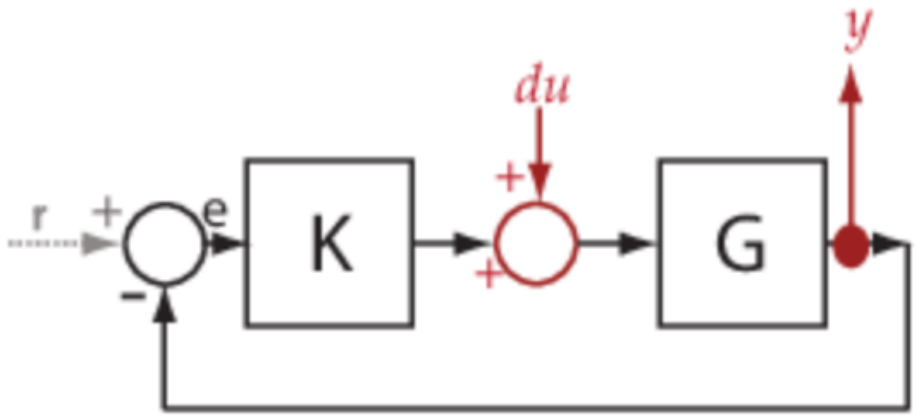
Linear systems are easier to analyse and prove mathematically their behaviour and properties.

How to Specify the Portion of a Model to Linearise





Example 1

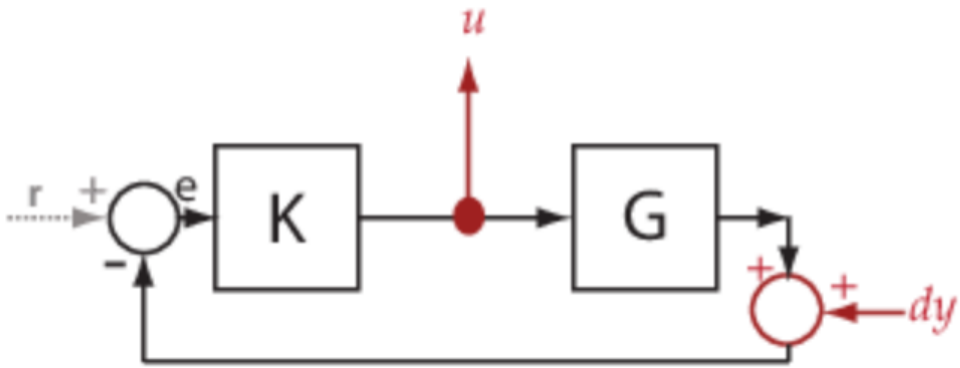


Show that the transfer function for $\frac{y}{du}$ is given by $\frac{G}{1+GK}$

$$(y * K + du) * G = u \tag{1}$$

Solve manually for $\frac{y}{du}$ or using Sympy. **Both ways were shown in the lecture!**

Example 2



Show that the transfer function for $\frac{u}{dy}$ is given by $\frac{-K}{1+KG}$

$$(u * G + dy) * K = u \tag{2}$$

Solve manually for $\frac{u}{dy}$ or using Sympy. **Both ways were shown in the lecture!**

Bode Plots Revisited

Consider the watertank of the previous lecture and tutorial:

$$\frac{d}{dt}Vol = A\frac{dH}{dt} = bV - \alpha\sqrt{H} \tag{3}$$

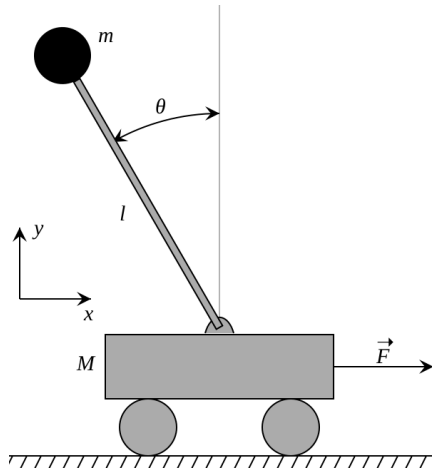
- $A = 20$
- $\alpha = 2$
- $b = 5$
- $H_{ref} = 10$ (reference signal - desired)
- Initial condition $H(0) = 1$
- PID parameters: $P = 1.599340, P_I = 0.079967, P_D = 0.$

Plot the Bode diagram.

You need to create a subsystem for the part of the plant that you would like to generate the Bode plot! The output of the PID controller (input of the subsystem) should be the first signal to the Bode plot block. The output of the subsystem should be the second signal selected for the Bode plot block.

The Inverted Pendulum Control Problem - PID Control

Consider the inverted pendulum control problem we have seen in the last tutorial implementing a PID controller using Simulink:



$$(M+m)\,\ddot{x} + m\ell\cos\theta\,\ddot{\theta} - m\ell\dot{\theta}^2\sin\theta = F - b_c\dot{x}, \tag{4}$$

$$m\ell\cos\theta\,\ddot{x} + m\ell^2\,\ddot{\theta} - mg\ell\sin\theta = -b_p\dot{\theta}. \tag{5}$$

Implement a PID controller for it using Python (not Simulink). For the values of the parameters see the last tutorial specification.

PID Controller in Python (cont'ed))

The implementation will appear here after the lecture!

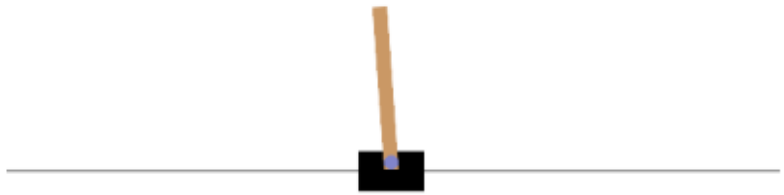
The Gymnasium Environment

Installing Gymnasium
Execute in a a terminal (inside Anaconda, if you are using Anaconda):

```
pip install swig
pip install gymnasium[toy_text]
pip install gymnasium[box2d]
```

You can follow the link below for more detailed instructions:
https://github.com/Farama-Foundation/Gymnasium

The Cart Pole Problem in Gymnasium



A simplified version of the inverted pendulum problem.

- Decision: Force to the left or right.

Action Space:

- 0: Push cart to the left
- 1: Push cart to the right

The Cart Pole System (cont'd)

Observation Space:
A 4-dimensional vector:

- Cart position x
- Cart velocity $\dot{x}$
- Pole Angle θ
- Angular velocity $\dot{\theta}$

https://gymnasium.farama.org/environments/classic_control/cart_pole/

Example of Robot Agent (Random Policy) for Cart Pole Balancing

```python
import gymnasium as gym
import time

# Create our training environment - a cart with a pole that needs balancing
env = gym.make("CartPole-v1", render_mode="human")

# Reset environment to start a new episode
observation, info = env.reset()
# observation: what the agent can "see" - cart position, velocity, pole angle, etc.
# info: extra debugging information (usually not needed for basic learning)

episode_over = False
total_reward = 0

while not episode_over:
    # Choose an action: 0 = push cart left, 1 = push cart right
    action = env.action_space.sample()  # Random action for now - real agents will be smarter!

    # Take the action and see what happens
    observation, reward, terminated, truncated, info = env.step(action)

    # reward: +1 for each step the pole stays upright
    # terminated: True if pole falls too far (agent failed)
    # truncated: True if we hit the time limit (500 steps)

    total_reward += reward
    episode_over = terminated or truncated

print(f"Episode finished! Total reward: {total_reward}")
time.sleep(3)  # wait for 3 secs

env.close()
```

https://gymnasium.farama.org/introduction/basic_usage/

PID Controller for Cart Pole Balancing in Gymnasium

Write some code to implement a PID controller for the cart-pole system and apply it in the Gymnasium environment (tutorial exercise).

PID Controller in Python (cont'ed))

The implementation will appear here after the tutorial!