Swiss German University

Differential Equation Code Report

Subject	: **Differential Equation**
Lecturer	: **Dr. Rusman Rusyadi**
Faculty	: **Mechatronics**
Date of Submission	:**30/10/2024**

Experiment: **Python Control Library Test and Experiment**

Tiara Mae Muljana

I. Abstract

This report will analyze and experiment on a dynamic mechanical system using Python's control functions, focusing on modeling, control design, and response analysis.

II. Objective

- Test and execute the given Phyton code
- Learn the code and understand the purpose of each function
- Experiment with varying system and controller parameters.
- Understand parameter effects on stability, response, and control tuning.

III. Analysis and Experiment

```
# System parameters
m = 0.5
l = 1
k = 0.5
J = m*l*l
g = 9.81
```

The initial parameters (m, I, k, J, g) define a system that may represent its component:

- m: Mass of the object (0.5 kg).
- I: Length or distance to the center of mass (1 meter).
- k: Damping coefficient (0.5 N·s).
- J: Moment of inertia, defined as m*I*I.
- g: Acceleration due to gravity (9.81 m/s²).

```
# Non-linear system:
# tau = J*ddtheta + k*dtheta + m*g*l*sin(theta)

# Linearised system in theta = pi/2:
# tau = J*ddtheta + k*dtheta - m*g*l*theta

# Transfer function
P = 1/(J*s**2 + k*s - m*g*l)
```

As stated from the code above the nonlinear system equation is:

$$\tau = J\theta'' + k\theta' + mglsin(\theta)$$

where τ is torque, θ’’ and θ’ are angular acceleration and velocity, and θ is the angular displacement. To simplify, the system is linearized around θ=π/2, yielding the transfer function:

$$P(s) \ = \ \frac{1}{Js^2 + ks + mgl}$$

```
# Plot root-locus of P
plt.figure()
ct.root_locus(P)

```

The root locus plot visually represents how the poles of the closed-loop system vary with different feedback gains, indicating stability. By running “ct.root_locus(P)”, we analyze the stability of “P(s)” and its response to control.

```
# Controller transfer function - lead-lag
Tc = 0.2
z = -3
K = 2
C = K*(s-z)/(Tc*s+1)
```

The control function, C(s), applies a lead-lag controller, a design that shifts poles and zeros to achieve desired stability and response:

- Tc: Time constant (0.2 s) that affects system response speed.
- z: Zero location (-3) that affects transient behavior.
- K: Gain (2) that scales the control effect.

This controller is defined as:

$$C(s) \ = \ K\frac{s - z}{Tc * s + 1}$$

This configuration, C*P, represents the plant P under the influence of the controller C.

```
# Close-loop transfer function
G = ct.feedback(C*P, 1)

# Plot root-locus of C*P
plt.figure()
ct.root_locus(C*P)
```

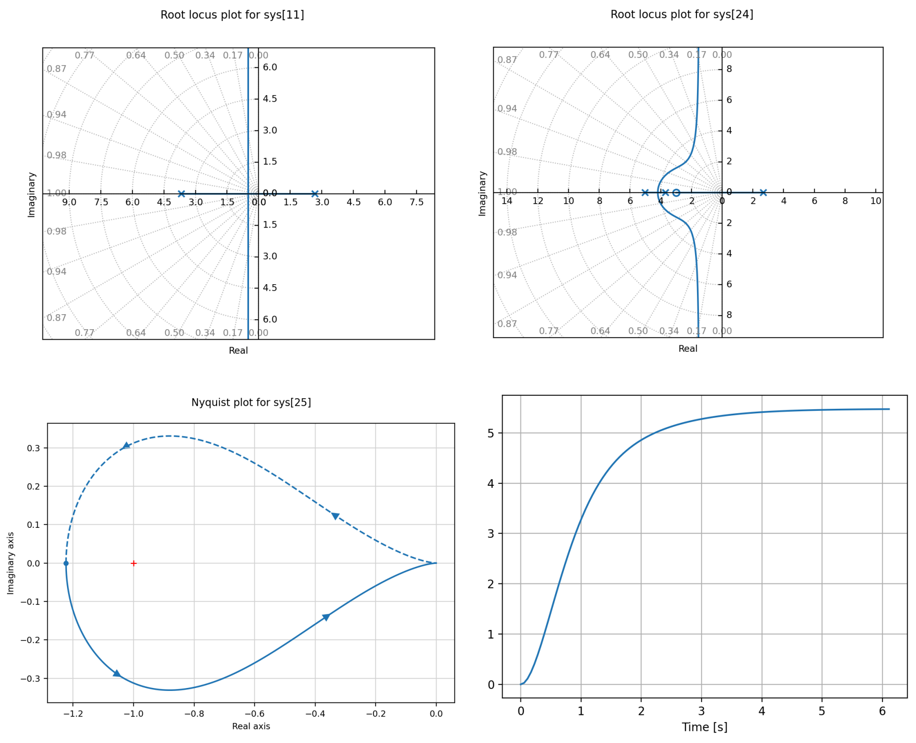
The closed-loop transfer function, “G = ct.feedback(C*P, 1)”, represents the overall system response when feedback control is applied. “1” represents a unit feedback. This setup helps maintain the desired output and reduces errors by adjusting the controller’s response.

```
# Plot Nyquist diagram
plt.figure()
ct.nyquist_plot(C*P)
```

The Nyquist plot helps examine system stability by mapping the frequency response of” C*P”. Peaks near or encircling the critical point (-1,0) may indicate instability or oscillations in the system. Running “ct.nyquist_plot(C*P)” visualizes the frequency response and helps determine if “G” has an appropriate stability margin.

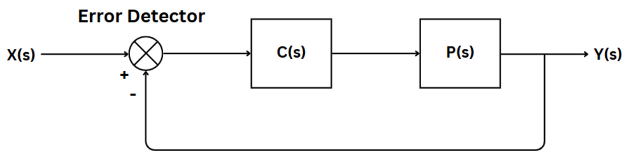
```
# Ideal step response
t, yout = ct.step_response(G)
plt.figure()
plt.plot(t, yout)
plt.grid()
plt.xlabel("Time [s]")
```

The step response represents how the closed-loop system reacts to a unit step input, simulating a sudden disturbance or desired setpoint shift. “t, yout = ct.step_response(G)” will generate time-domain data. Plotting this response shows how “G” reaches steady state, which is crucial in assessing performance criteria like settling time, overshoot, and steady-state error.



Graphs with the initial value

After running the code we will have this 4 graph displaying the Root Locus of the System Transfer Function P(s), Root Locus of the Open-Loop System with Controller C(s)P(s), Nyquist Plot of the System with Controller C(s)P(s), and Step Response of the Closed-Loop System G(s)



The block diagram flow for this code can be broken down into distinct blocks representing each component and their interactions. Here’s an outline of the block flow:

1. System Dynamics Block (Plant P(s))

- Inputs: Torque (τ)
- Outputs: Angular position/angle (θ)

2. Controller Block (C(s))

- Inputs: Error (difference between desired and actual angle)
- Outputs: Modified Torque (τ')
- Description: The controller modifies the input to stabilize and optimize the system's performance. Here, the lead-lag controller transfer function C adjusts the control action by providing phase lead and gain to manage the system's response time and stability.

3. Feedback Loop (Closed-Loop System)

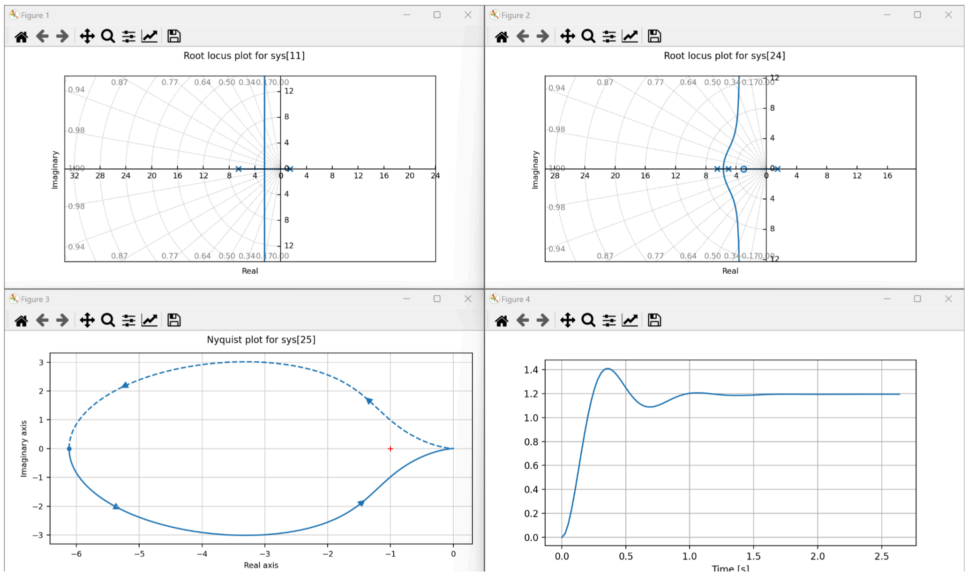
- Inputs: Desired angle (X(s))
- Outputs: Actual angle (Y(s))
- Description: The feedback loop takes the difference between the desired angle and the actual angle (error signal) and feeds it into the controller C(s). The controller output is applied to the system P(s) to reduce the error. This closed-loop structure ensures that the system continuously corrects its response to meet the desired outcome.

4. Response Analysis Blocks

- Root Locus Plot Block: Analyzes how the closed-loop poles move with changes in feedback gain, indicating system stability.
- Nyquist Plot Block: Shows the frequency response to assess stability and robustness.
- Step Response Block: Simulates the closed-loop system's time-domain response to a step input, demonstrating settling time, overshoot, and steady-state behavior.

To see how each parameter affects the system, try experimenting with values for mass m, damping k, and controller gain K. Here are a few trials:

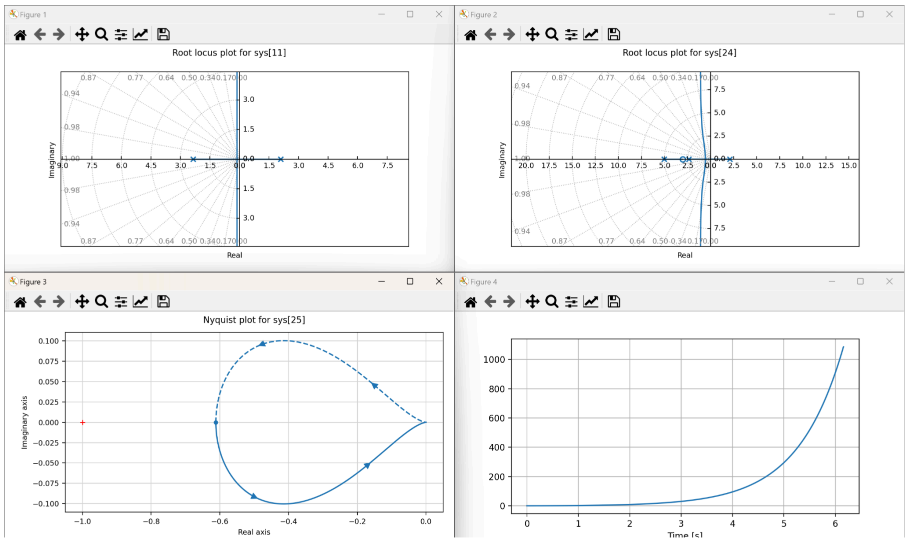
1. Changing the value of mass (m)



Changing m value from 0.5 to 0.1

- Increasing the mass mmm of the system generally leads to a slower response due to the larger inertia. This results in a decrease in the natural frequency (ωn\omega_nωn) of the system, causing the poles to shift closer to the imaginary axis in the s-plane, which may increase the settling time and lead to longer oscillations. Conversely, decreasing mmm reduces inertia, increasing the system's responsiveness and potentially leading to faster settling times and more pronounced overshoot. The overall dynamic behavior becomes more sluggish with increased mass, requiring careful tuning of the controller to maintain desired performance.

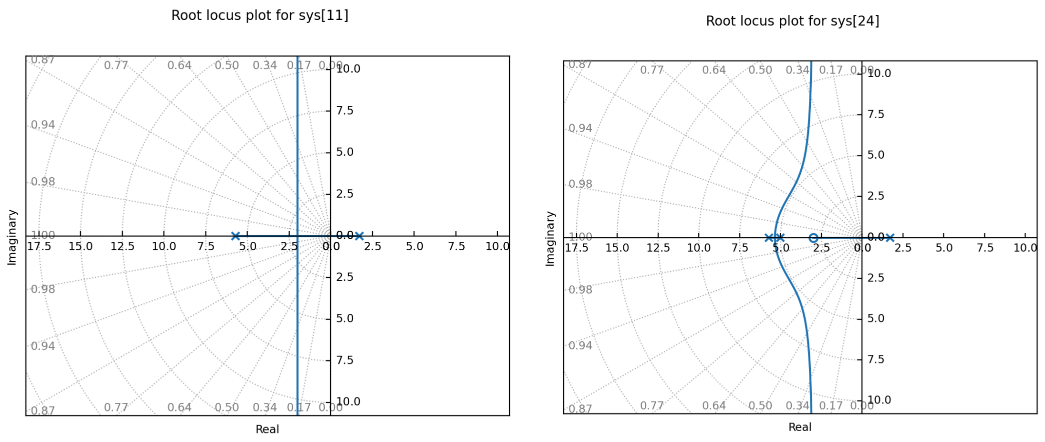
2. Changing the value of length (l)

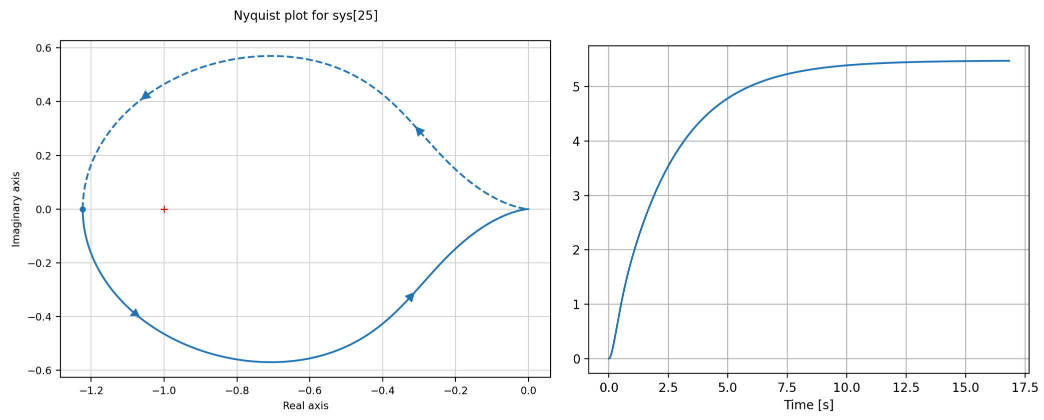


Changing the value of l from 1 to 2

- Increasing the length lll of the pendulum-like system affects both the moment of inertia and the gravitational torque. A longer length increases the gravitational force's leverage, enhancing the effect of gravity on the system and shifting the system's poles, which can lead to greater oscillatory behavior and slower settling times. Conversely, reducing lll decreases the moment of inertia, enhancing the system's responsiveness and potentially leading to faster settling times with less overshoot. The effect of length on system dynamics emphasizes the balance between inertia and gravitational influences.

3. Changing the value of the damping coefficient (k)



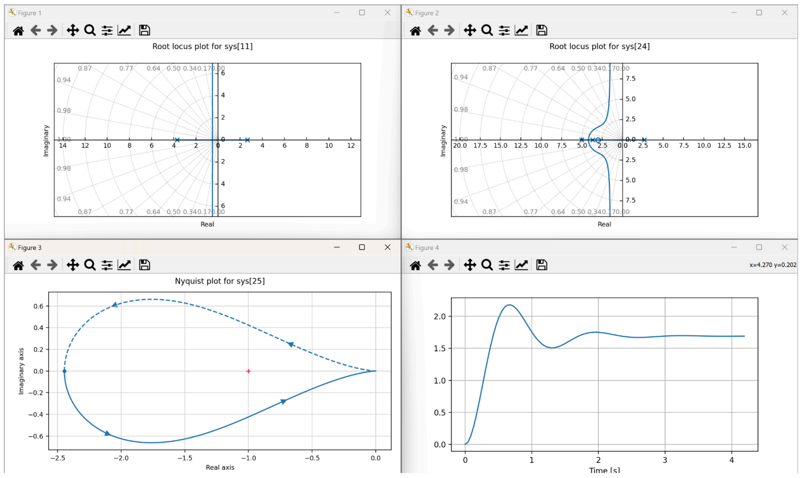


Changing k value from 0.5 to 2

- Increasing k from 0.5 to 2, will simulate greater resistance. This increases the damping in the system. As k increases:
  1. The impact on the root locus of P(s) graph are the poles of the system will move towards the left half of the s-plane, indicating better stability. The root locus will show a more stable configuration with reduced oscillation.
  2. Impact on Root Locus of C(s)P(s) graph the yield poles will be more favorable for stability and control.
  3. The impact on Nyquist plot will be a more stable system will result in a plot that remains clear of the critical point (-1, 0), indicating no encirclements, and maintaining stability.
  4. The impact on step response graph will be reducing oscillations but potentially increasing settling time.

Increasing the damping coefficient k generally results in improved stability and faster settling times due to reduced oscillations and overshoot. As k increases, the damping ratio increases, causing the poles to move left in the s-plane and leading to a quicker return to equilibrium after disturbances. However, excessively high damping can make the system sluggish and overly damped, which might counterintuitively increase the settling time. Lowering k reduces damping, leading to longer settling times and increased overshoot, making the system more oscillatory.

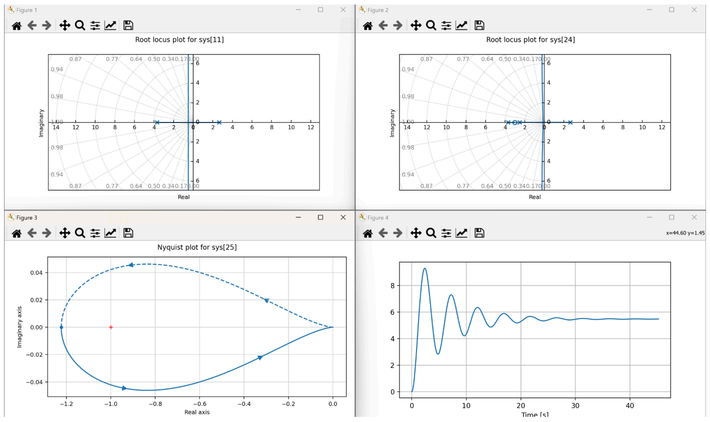
4. Changing the value of gain (K)



Changing the value of K from 2 to 4

- Modifying the controller gain K directly impacts the system's responsiveness. Increasing K typically leads to faster responses and reduced settling times as the controller applies stronger corrective actions. However, excessively high gain can increase overshoot and instability, causing oscillations and potentially leading to system saturation. Decreasing K leads to a more sluggish response with longer settling times and larger steady-state errors, which may be beneficial for stability but could compromise performance.

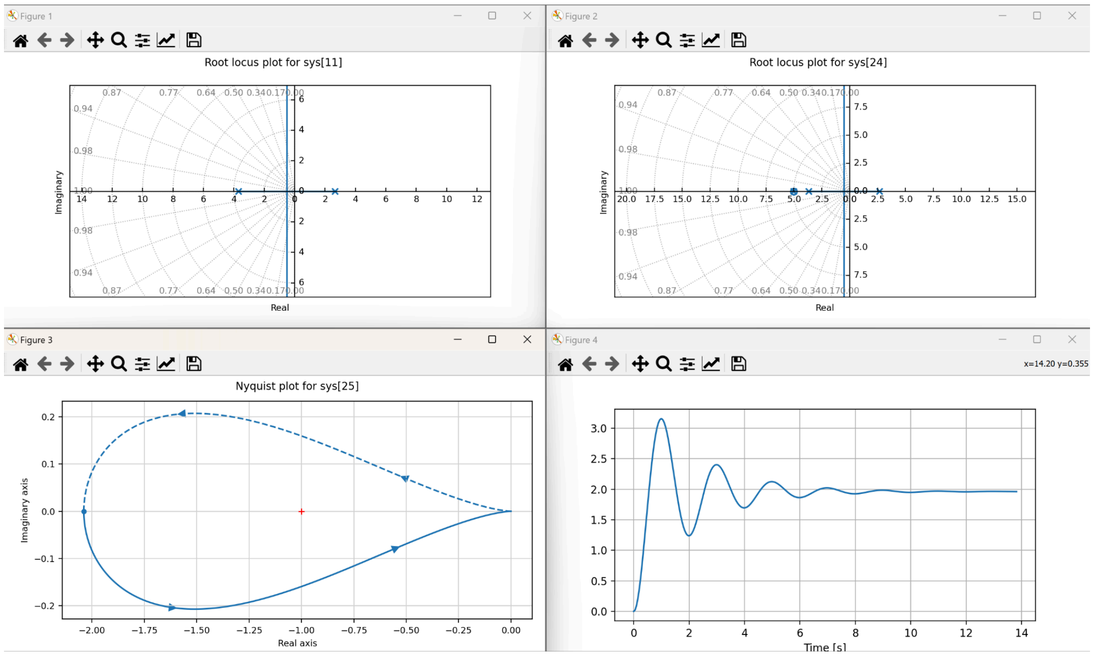
5. Changing the value of time constant (Tc)



Changing the value of Tc from 0.2 to 0.4

- The time constant Tc in a lead-lag controller affects the speed of the controller's response. Increasing Tc slows down the controller’s response, leading to longer settling times and potentially greater overshoot as the system reacts more sluggishly to changes. Conversely, decreasing Tc enhances the controller's responsiveness, allowing for quicker adjustments to disturbances and improved settling times. However, a response that is too rapid can lead to instability or excessive oscillation if not carefully tuned, highlighting the need for balance in controller design.

6. Changing zero (z)



Changing the value of z from -3 to -5

- Modifying the zero z in the controller's transfer function influences the system's transient response and phase characteristics. Increasing z (moving it closer to the right in the s-plane) generally results in decreased phase lead, which can slow the system’s response and lead to longer settling times. This is because a higher zero can introduce lag, making the system less reactive to input changes and potentially increasing overshoot and oscillations. Conversely, moving z further left (more negative) enhances the phase lead, leading to a faster response with reduced settling times and minimal overshoot, as the controller can more effectively counteract the lagging dynamics of the system. However, if z is set too far to the left, it may lead to increased risk of instability and overshoot, requiring careful tuning to ensure that the benefits of increased responsiveness do not come at the cost of system stability.

References :

- **Code Source**
  https://github.com/simorxb/Python-Control-Library.git