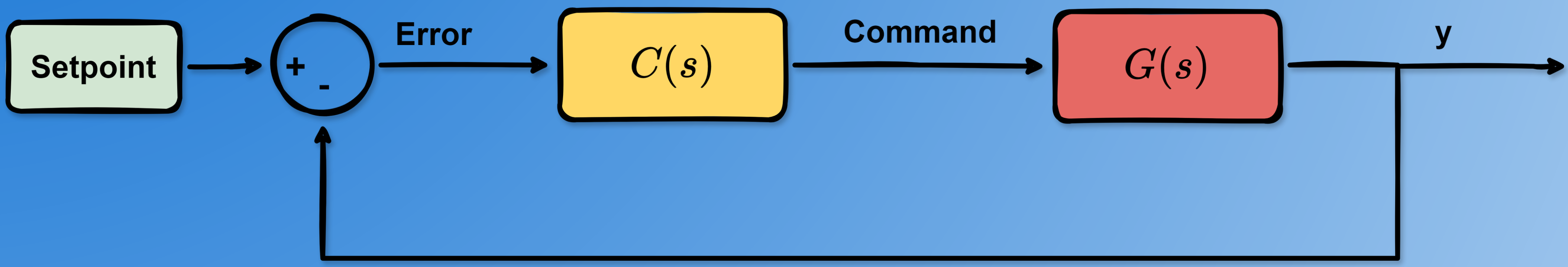
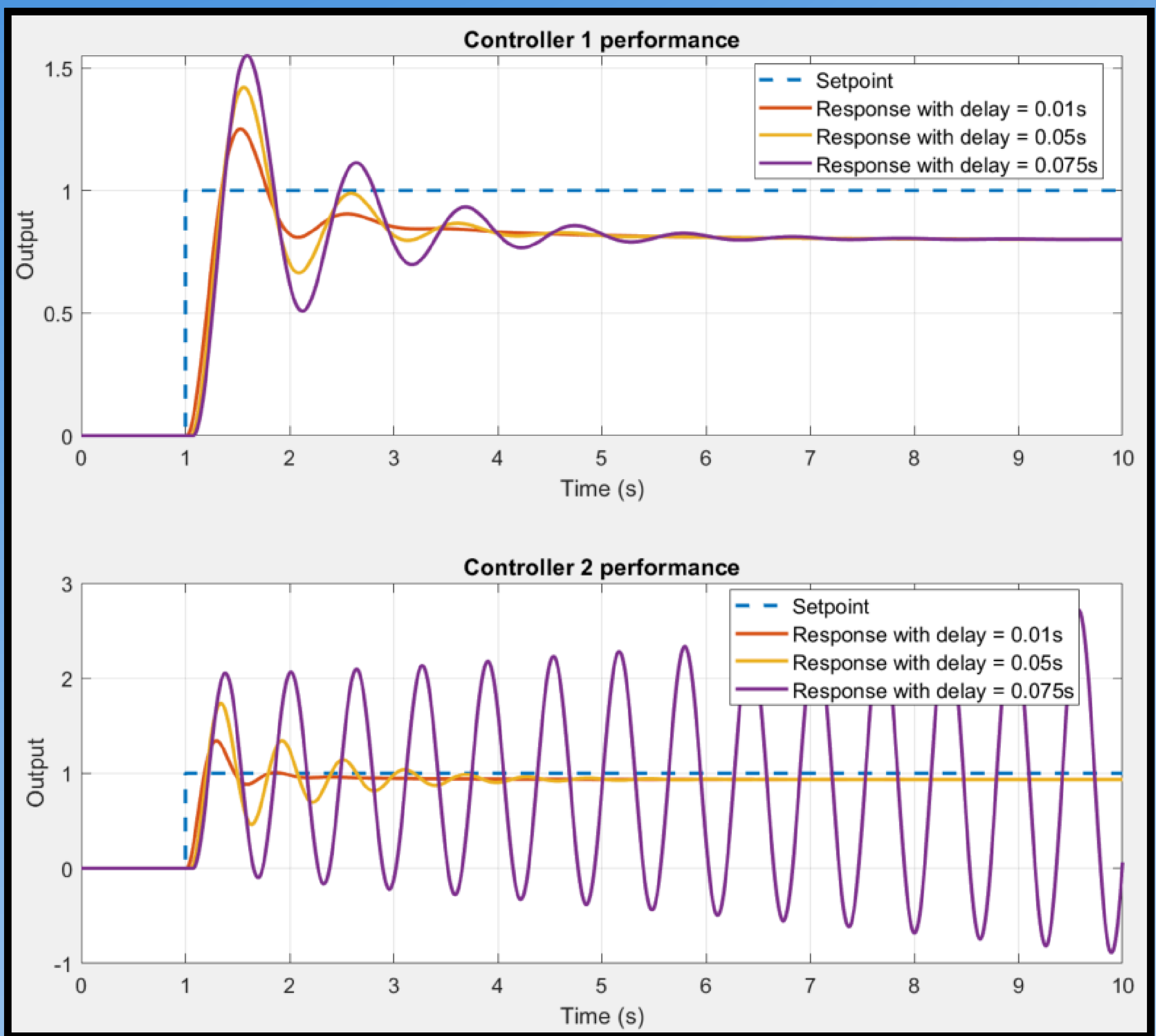
Phase Margin vs Delay Margin





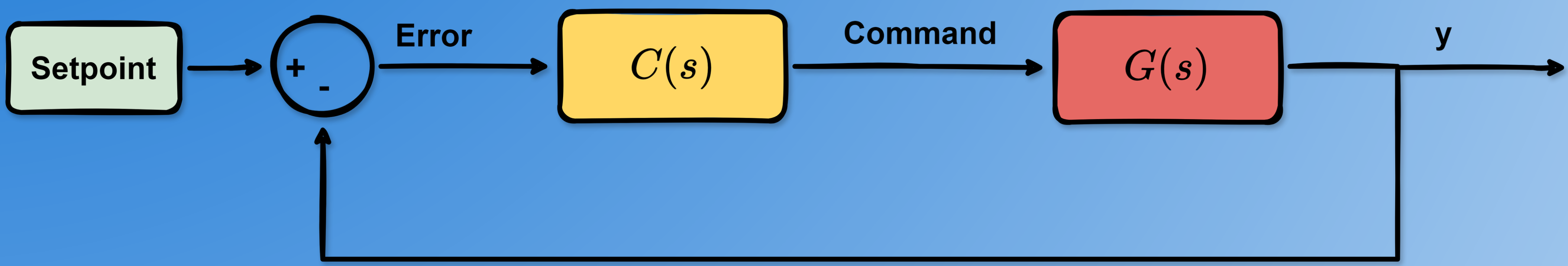

```
%% Plot results

% Create figure
figure;

legend_V = cell(length(delay_V)+1, 1);

% Cycle through controllers
for C_idx = 1:length(C_V)

    % Subplot for controller
    subplot(length(C_V), 1, C_idx);

    % Get setpoint - always the same
    stp = out_V{C_idx, 1}.logsout.get('setpoint').Values.Data;
    t_stp = out_V{C_idx, 1}.logsout.get('setpoint').Values.Time;

    % Plot setpoint
    plot(t_stp, stp, '--', 'LineWidth', 2);
    hold on;

    % Setpoint legend
    legend_V{1} = 'Setpoint';

    % Cycle through delays
    for d_idx = 1:length(delay_V)

        % Get response data
        y = out_V{C_idx, d_idx}.logsout.get('y').Values.Data;
        t_y = out_V{C_idx, d_idx}.logsout.get('y').Values.Time;

        % Plot response
        plot(t_y, y, 'LineWidth', 2);

        % Response legend
        legend_V{d_idx + 1} = ['Response with delay = ' num2str(delay_V{d_idx}) 's'];

    end

    hold off;
    title(['Controller ' num2str(C_idx) ' performance']);
    xlabel('Time (s)');
    ylabel('Output');
    legend(legend_V, 'FontSize', 12);
    set(gca, 'FontSize', 12);
    grid on;

end
```

Model

https://github.com/simorxb/delay-margin

SIMONE BERTONI
CONTROL LAB

Control Architecture



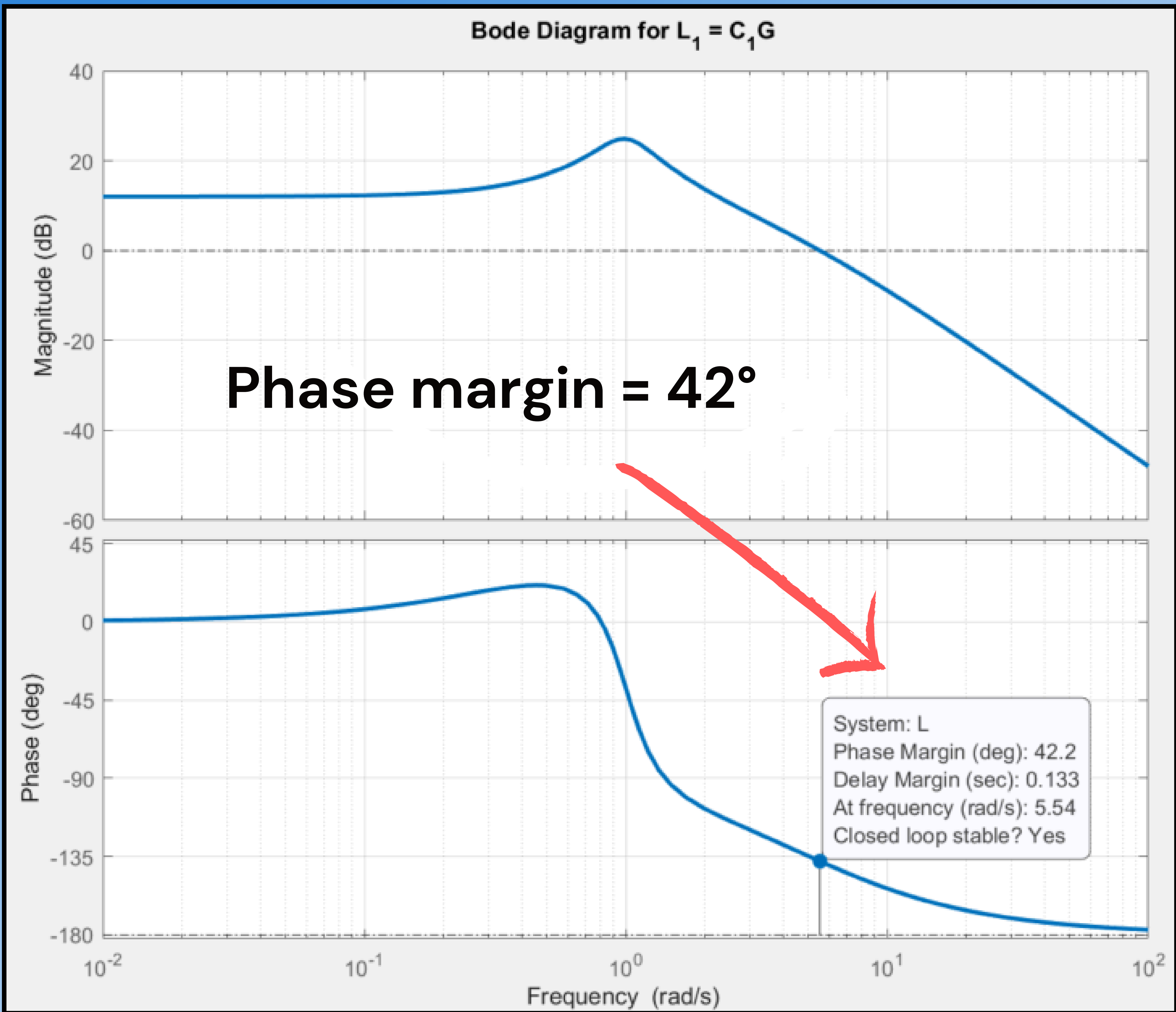
Plant

- $G(s) = \frac{1}{S^2+0.5s+1}$

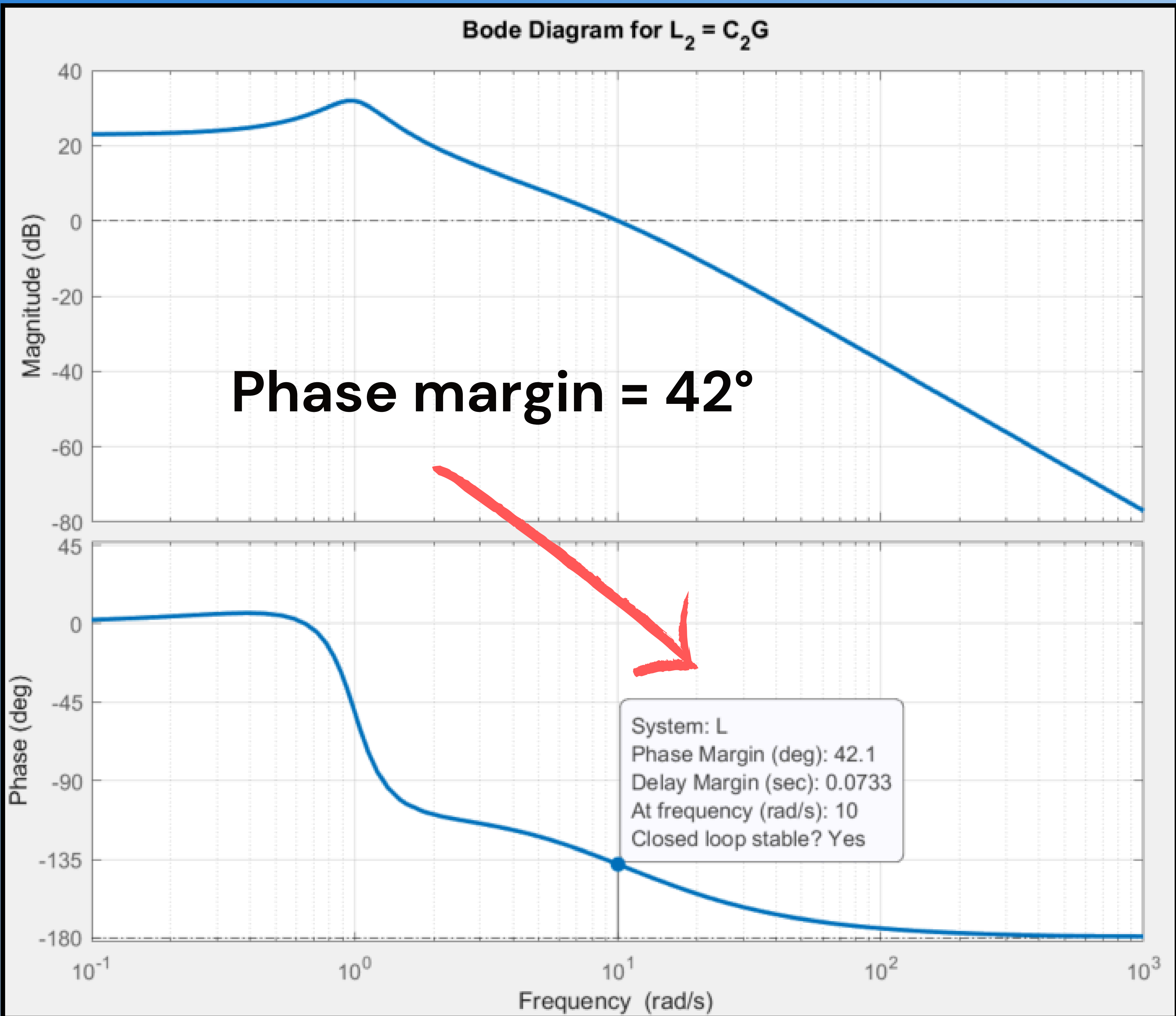
Controllers - 2 Options

- $C_1(s) = \frac{4(2s+1)}{0.2s+1}$
- $C_2(s) = \frac{14(s+1)}{0.1s+1}$

Controller 1
Bode & Margins

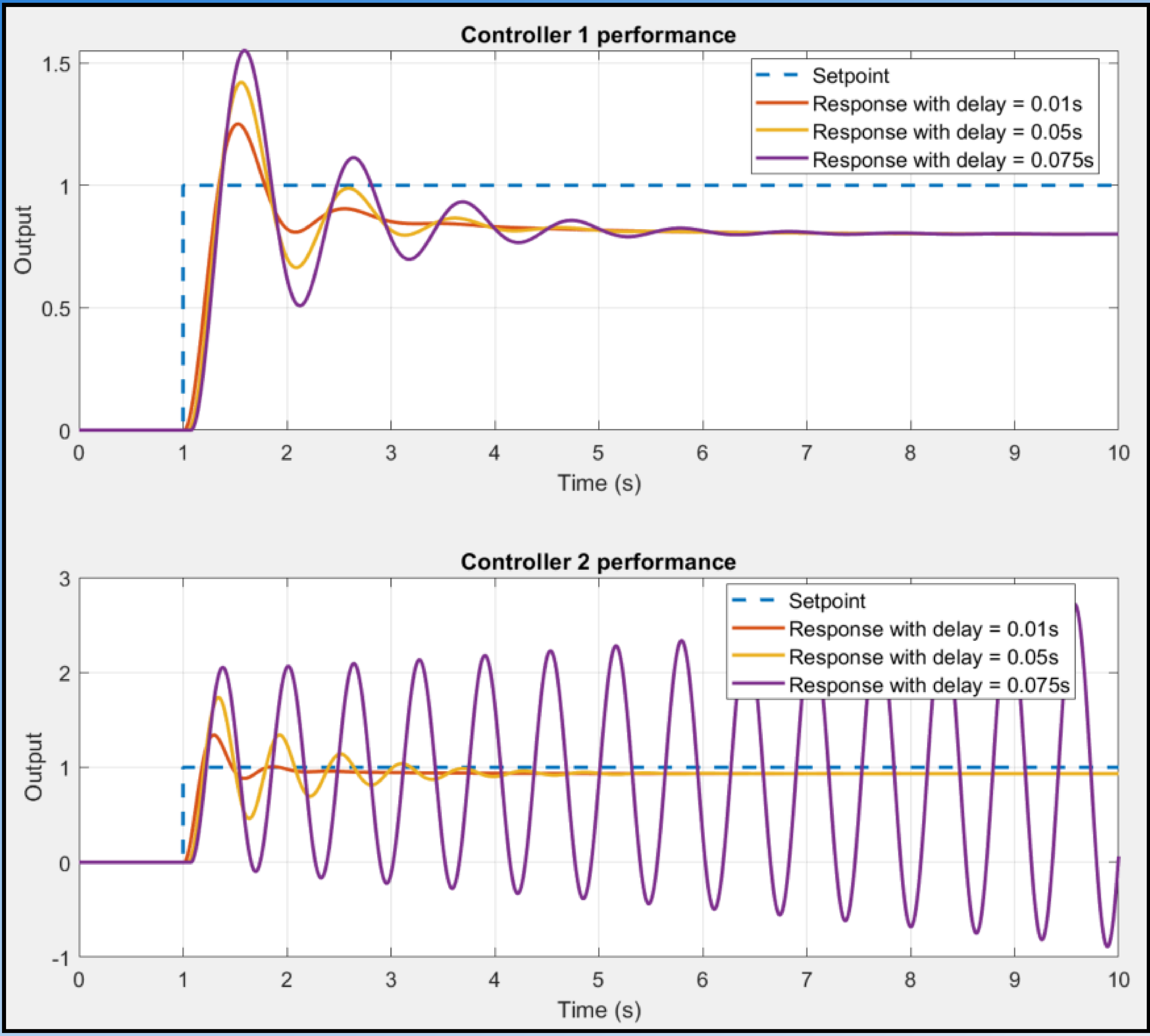


Controller 2
Bode & Margins



Simulation

Let’s simulate the close loop system with 3 different delays in the loop: 0.01s, 0.05s, 0.075s.



Delay Margin

Controller 1 performs a lot better than **Controller 2** when the system is subject to time delay.

Even though the closed-loop systems have the same phase margin!

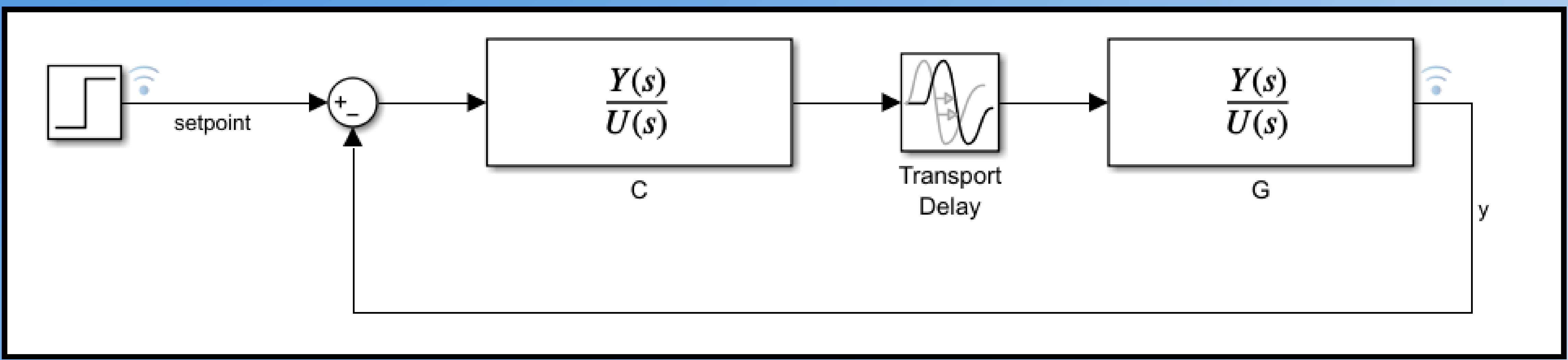
WHY??

Delay margin! Let's define:

- P_m: phase margin in radians
- ω_p: Frequency (rad/s) at which the magnitude of $C(j\omega_p)G(j\omega_p)$ is = 0 dB
- D_m: time delay margin in seconds

If you look back to the Bode diagrams you'll see that delay margin for **Controller 1** is 0.13s and for **Controller 2** is 0.073s!

Simulink Model



Matlab Code Design

```matlab
%% Design

% Define transfer function G
s = tf('s');
G = 1/(s^2 + 0.5*s + 1);

% Define controllers
C_V{1} = 4*(2*s+1)/(0.2*s+1);
C_V{2} = 14*(s+1)/(0.1*s+1);

for C_idx = 1:length(C_V)

    % Controller
    C = C_V{C_idx};

    % Loop function
    L = C * G;

    % Closed-loop transfer function
    T = feedback(L, 1);

    % Plot Bode diagram for L
    figure;
    bode(L);
    title(['Bode Diagram for L_' num2str(C_idx) ' = C_' num2str(C_idx) 'G']);
    grid on;

end
```

Matlab Code Simulation

```matlab
%% Simulation

% Define delays
delay_V = {0.01, 0.05, 0.075};

% Clear out_V
out_V = cell(length(C_V), length(delay_V));

% Cycle through controllers
for C_idx = 1:length(C_V)

    % Controller
    C = C_V{C_idx};

    % Cycle through delays
    for d_idx = 1:length(delay_V)

        % Delay
        delay = delay_V{d_idx};

        % Simulate
        out = sim("Delay_Margin.slx");

        % Store output data
        out_V{C_idx, d_idx} = out;

    end

end
```

Matlab Code Plot

```
%% Plot results

% Create figure
figure;

legend_V = cell(length(delay_V)+1, 1);

% Cycle through controllers
for C_idx = 1:length(C_V)

    % Subplot for controller
    subplot(length(C_V), 1, C_idx);

    % Get setpoint - always the same
    stp = out_V{C_idx, 1}.logsout.get('setpoint').Values.Data;
    t_stp = out_V{C_idx, 1}.logsout.get('setpoint').Values.Time;

    % Plot setpoint
    plot(t_stp, stp, '--', 'LineWidth', 2);
    hold on;

    % Setpoint legend
    legend_V{1} = 'Setpoint';

    % Cycle through delays
    for d_idx = 1:length(delay_V)

        % Get response data
        y = out_V{C_idx, d_idx}.logsout.get('y').Values.Data;
        t_y = out_V{C_idx, d_idx}.logsout.get('y').Values.Time;

        % Plot response
        plot(t_y, y, 'LineWidth', 2);

        % Response legend
        legend_V{d_idx + 1} = ['Response with delay = ' num2str(delay_V{d_idx}) 's'];

    end

    hold off;
    title(['Controller ' num2str(C_idx) ' performance']);
    xlabel('Time (s)');
    ylabel('Output');
    legend(legend_V, 'FontSize', 12);
    set(gca, 'FontSize', 12);
    grid on;

end
```

PID Controller Course

https://simonebertonilab.com

Understand the control theory

★★★★★ *April 28, 2024*

I think the most important thing is to understand the meaning behind the mathematical formula. I guess this is the mission of Simone in this course and from my point of view he fully achivied this target. I hope to see in the future other courses (e.g advanced controls) structered in the same way with the same passion and examples.

Thank you Simone. Show less

Emidio Verified

★★★★★

Very helpful and practical

Yoav Golan

I enjoyed this course very much. I learned a lot of practical knowledge in a short time. Simone is very clear and teaches well, thank you! In the future, I would be very interested if Simone added a course with more subjects, such as cascading controllers, rate limiting, and how the controllers look in actual code. Thanks again!

★★★★★

Intuitive and Practical

Ranya Badawi

Simone's explanation of PID control was very intuitive. This is a great starter course to gain a fundamental understanding and some practical knowledge of PID controllers. I highly recommend it. For future topics, I'd be interested in frequency response, transfer functions, Bode plots (including phase/gain margin), Nyquist plots, and stability.

★★★★★

Very good sharing of experience

Romy Domingo Bompart Ballache

I have background in control system for power electronics, I see every lesson very useful.

Great course

★★★★★ *April 15, 2024*

Right to the point, easy to follow and very practical. I missed the zero/pole placement and phase margin analysis. It would also be interesting if you could provide other plants examples. Anyway, a great course to help designing and tuning a PID controller.

Leonardo Starling Verified

A different way to learn PID !

★★★★★ *May 31, 2023*

The teacher explains PID in a clear way adding his experience there where formulas alone cannot do much. Furthermore, each topic covered is included in a practical example to better fix ideas.

Michele De Palma