EE571 HW Feature Demo

Template Showcase   2025-09-01

Problem 1: This demonstrates the basic problem command with automatic numbering.

This is how you create a basic problem. The numbering happens automatically (Problem 1, Problem 2, etc.).

Problem A.1: This demonstrates custom problem numbering.

You can override the automatic numbering by providing a custom label in square brackets.

Problem 3: Find the derivative of $f(x) = 3x^2 + 2x - 5$ and evaluate it at $x = 2$.

This shows how to include the actual question or description within the problem command.
Using the power rule:

$$\begin{aligned} f(x) &= 3x^2 + 2x - 5 \\ f'(x) &= 6x + 2 \\ f'(2) &= 6(2) + 2 && = 12 + 2 = 14 \end{aligned}$$

Problem 4: This problem demonstrates sub-problems with multiple parts.

(a)

Find the transfer function of the system.
Given the differential equation $\ddot{y} + 3\dot{y} + 2y = 5u(t)$:

$$\begin{aligned} s^2Y(s) + 3sY(s) + 2Y(s) &= 5U(s) \\ G(s) &= \frac{Y(s)}{U(s)} && = \frac{5}{s^2 + 3s + 2} \end{aligned}$$

(b)

Factor the denominator and find the poles.

$$\begin{aligned} s^2 + 3s + 2 &= (s+1)(s+2) \\ \text{Poles: } s_1 = -1, \quad s_2 && = -2 \end{aligned}$$

(iii)

Determine system stability (custom sub-problem numbering).
Since both poles have negative real parts, the system is stable.

Problem 5: Demonstrate the hwmath environment and new comparison shorthand operators.

The hwmath environment provides automatic alignment using shorthand commands:

(a)

Basic alignment with
`eq` shorthand:

$$\begin{aligned}
\mathcal{L}\{f(t)\} &= F(s) \\
\mathcal{L}\{\dot{f}(t)\} &= sF(s) - f(0) \\
\mathcal{L}\{\ddot{f}(t)\} &= s^2F(s) - sf(0) - \dot{f}(0)
\end{aligned}$$

(b)

All comparison operators including new additions:

$$\begin{aligned}
f(x) &= x^2 + 1 \\
f(x) &> 0 \text{ for all } x \\
f(2) &> f(1) \\
f(3) &>> f(0) \\
0 &< f(x) \\
f(-1) &<< f(2) \\
\omega_n &\geq 5 \text{ rad/s} \\
\text{Error} &\leq 2\% \\
K_p &\neq 0 \\
G(j\omega) &\approx H(j\omega) \text{ at high freq}
\end{aligned}$$

(c)

Complex mathematical expressions with mixed operators:

$$\begin{aligned}
\mathbf{A} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -4 \end{bmatrix}, \quad \mathbf{B} &&= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\
\lambda_1 &< \lambda_2 &&< \lambda_3 \\
|\lambda_{\max}| &>> |\lambda_{\min}| \\
\text{Re}(\lambda_i) &< 0 \text{ for all } i
\end{aligned}$$

Problem 6: Show numbered equations for referencing.

For equations that need to be referenced later, use hwmathnumbered:

$$\begin{aligned}
E &= mc^2 && (1) \\
F &= ma && (2) \\
P &= VI && (3) \\
|H(j\omega)| &> 0.707 \text{ for } \omega &&< \omega_c \quad (4)
\end{aligned}$$

These equations are numbered automatically and can be referenced in the text.

Problem 7: Demonstrate the note command for emphasis.

(a)

Basic note without custom text:

NOTE

This is the default "NOTE" highlighting.

(b)

Custom note with specific text:

IMPORTANT: Check stability margins before proceeding with the design.

(c)

Multiple notes with different messages:

WARNING: Ensure proper gain margins

Calculate the open-loop transfer function first.

CRITICAL: Verify closed-loop stability

The system must remain stable under all operating conditions.

Problem 8: Demonstrate the new example environment for highlighting important examples.

The new `example` environment creates a boxed area with a custom header for showcasing worked examples:

Example 4-1: PID Controller Design

Given a second-order plant:

$$G_p(s) = \frac{10}{s(s+2)}$$

Design a PID controller $G_c(s) = K_p + \frac{K_i}{s} + K_d s$ to achieve:

- Steady-state error $< 2\%$ for step input
- Phase margin $> 45°$
- Gain margin > 6 dB

Solution: Using root locus design, we select: $K_p = 5$, $K_i = 2$, $K_d = 0.5$.

Example 7-2: Frequency Response Analysis

For the transfer function $H(j\omega) = \frac{1}{1+j\omega/\omega_c}$:
Find the magnitude and phase at $\omega = \omega_c$.
Solution: At $\omega = \omega_c$:

$$|H(j\omega_c)| = \frac{1}{\sqrt{1+1}} = \frac{1}{\sqrt{2}} = 0.707 \tag{5}$$

$$\angle H(j\omega_c) = -\tan^{-1}(1) = -45° \tag{6}$$

This is the 3-dB point (half-power frequency).

Example 12-5: State-Space Controllability

Given the state-space system:

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

Determine if the system is controllable.
Solution: The controllability matrix is:

$$\mathcal{C} = [B \quad AB] = \begin{bmatrix} 0 & 1 \\ 1 & -3 \end{bmatrix}$$

Since $\det(\mathcal{C}) = -1 \neq 0$, the system is controllable.

Problem 9: Demonstrate the hwgraphic command for including images.

(a)

Command syntax and features:
The
`hwgraphic` command provides automatic centering with optional titles and scaling:

`hwgraphic{path}[optional title][optional scale]`
Key features:

- Automatic centering
- Optional title above the image
- Optional scaling (default: 0.8 textwidth)
- Proper spacing before and after

(b)

Example usage patterns:

- `hwgraphic{figure.pdf}` — Basic inclusion with default width
- `hwgraphic{figure.pdf}[Plot Title]` — With title
- `hwgraphic{figure.pdf}[Title][0.6]` — With title and custom scale
- `hwgraphic{figure.pdf}[][0.5]` — Custom scale, no title

Problem 10: Demonstrate code listing environments.

(a)

MATLAB implementation with caption:

Listing 1: Second Order System Response Function

```
function [y, t] = second_order_response(wn, zeta, K, tfinal)
    % Calculate step response of second order system
    % Inputs:
    %   wn: natural frequency (rad/s)
    %   zeta: damping ratio
    %   K: gain
    %   tfinal: final simulation time

    % Create transfer function
    num = [K * wn^2];
    den = [1, 2*zeta*wn, wn^2];
    G = tf(num, den);

    % Calculate and plot step response
    [y, t] = step(G, tfinal);

    % Create plot
    figure;
    plot(t, y, 'LineWidth', 2);
    xlabel('Time (s)');
    ylabel('Amplitude');
    title('Step Response');
    grid on;
end
```

(b)

Simple MATLAB commands without caption:

```
% Define system parameters
wn = 5;          % Natural frequency (rad/s)
zeta = 0.3;      % Damping ratio
K = 2;           % Gain

% Calculate and plot response
[y, t] = second_order_response(wn, zeta, K, 2);
```

(c)

Python implementation with caption:

Listing 2: Control Systems Analysis in Python

```
import numpy as np
import matplotlib.pyplot as plt
import control as ct

def second_order_response(wn, zeta, K, tfinal):
    """
    Calculate step response of second order system
    Parameters:
        wn: natural frequency (rad/s)
        zeta: damping ratio
        K: gain
        tfinal: final time for simulation
    """

    # Create transfer function
    num = [K * wn**2]
    den = [1, 2*zeta*wn, wn**2]
    G = ct.tf(num, den)

    # Calculate step response
    t = np.linspace(0, tfinal, 1000)
    y, t = ct.step_response(G, t)

    # Plot results
    plt.figure(figsize=(10, 6))
    plt.plot(t, y, linewidth=2, label='Step Response')
    plt.xlabel('Time (s)')
    plt.ylabel('Amplitude')
    plt.title('Second Order System Step Response')
    plt.grid(True)
    plt.legend()
    plt.show()

    return y, t

# Example usage
wn = 5.0        # Natural frequency
zeta = 0.3      # Damping ratio
K = 2.0         # Gain
tfinal = 2.0    # Simulation time

y, t = second_order_response(wn, zeta, K, tfinal)
```

```
print(f"Final value: {y[-1]:.3f}")
```

(d)

Simple Python code without caption:

```
# Quick calculation
import numpy as np

# System parameters
A = np.array([[0, 1], [-2, -3]])
B = np.array([[0], [1]])

# Check controllability
C = np.hstack([B, A @ B])
rank = np.linalg.matrix_rank(C)
print(f"Controllability matrix rank: {rank}")
```

(e)

Terminal output with caption:

```
$ pip install control numpy matplotlib
Collecting control
  Downloading control-0.9.2-py3-none-any.whl (709 kB)
Collecting numpy>=1.17
  Using cached numpy-1.21.2-cp39-cp39-macosx_10_9_x86_64.whl (17.0 MB)
Collecting matplotlib>=3.0
  Using cached matplotlib-3.4.3-cp39-cp39-macosx_10_9_x86_64.whl (7.2 MB)
Successfully installed control-0.9.2 numpy-1.21.2 matplotlib-3.4.3

$ python control_analysis.py
System is controllable: True
Poles: [-1. -2.]
Step response calculated successfully
Final value: 2.500
```

Listing 3: Package Installation and Testing

(f)

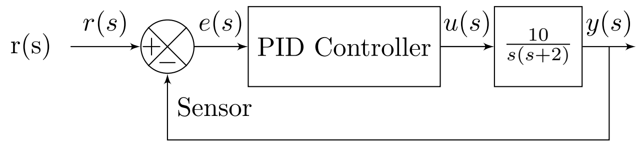
Simple terminal commands without caption:

```
$ cd ~/project
$ python -m pytest tests/
======================= test session starts ========================
collected 15 items

tests/test_control.py ........                [ 53%]
tests/test_analysis.py .......                [100%]

======================= 15 passed in 2.34s ========================
```

Problem 11: Demonstrate the hwblocks environment and blox package features for control system diagrams.

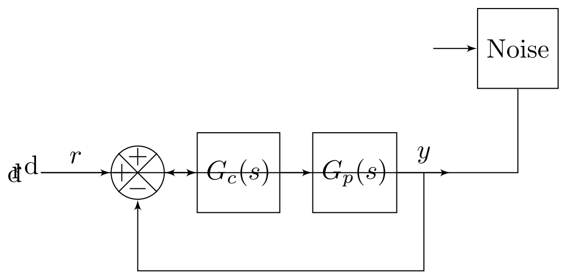
(a)

Classic PID feedback control system:



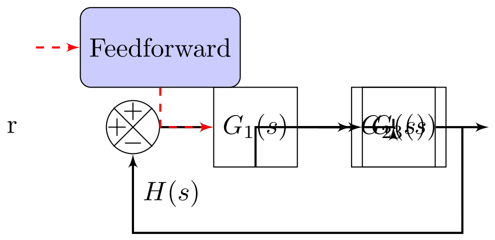
(b)

Multi-input system with disturbance and noise using chains:



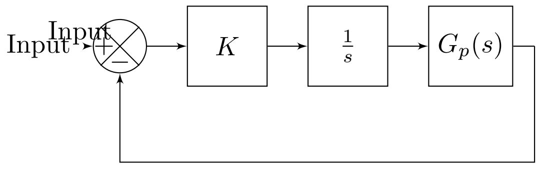
(c)

Advanced system with multiple branches and custom styling:



(d)

Demonstrating blox loop command for rapid prototyping:



Problem 12: Integration with Standard LaTeX Features

This problem demonstrates that the homework class works well with standard LaTeX environments like tables, figures, and lists, while maintaining consistent formatting.

(a)

Standard itemize and enumerate lists work normally:

1. First numbered item
2. Second numbered item with math: $x = \frac{-b\pm\sqrt{b^2-4ac}}{2a}$
3. Third item with sub-list:
   - Bullet point A
   - Bullet point B with inline code: `control.tf([1], [1, 1])`

(b)

Tables integrate seamlessly:

Parameter	**Symbol**	**Value**
Natural Frequency	ω_n	5 rad/s
Damping Ratio	ζ	0.3
Gain	K	2.0

Appendix A

This is the first appendix. It contains additional derivations and supplementary material that supports the main homework solutions.

Extended derivation of the quadratic formula:

$$\begin{aligned}
ax^2 + bx + c &= 0 \\
ax^2 + bx &= -c \\
x^2 + \frac{b}{a}x &= -\frac{c}{a} \\
x^2 + \frac{b}{a}x + \left(\frac{b}{2a}\right)^2 &= -\frac{c}{a} + \left(\frac{b}{2a}\right)^2 \\
\left(x + \frac{b}{2a}\right)^2 &= \frac{b^2 - 4ac}{4a^2} \\
x + \frac{b}{2a} &= \pm\frac{\sqrt{b^2 - 4ac}}{2a} \\
x &= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
\end{aligned}$$

Appendix B: Reference Tables and Constants

This appendix contains useful reference tables and formulas commonly used in control systems analysis.

Transform	**Time Domain**	**Frequency Domain**
Unit Step	$u(t)$	$\frac{1}{s}$
Ramp	$t \cdot u(t)$	$\frac{1}{s^2}$
Exponential	$e^{-at} \cdot u(t)$	$\frac{1}{s+a}$
Sine	$\sin(\omega t) \cdot u(t)$	$\frac{\omega}{s^2+\omega^2}$
Cosine	$\cos(\omega t) \cdot u(t)$	$\frac{s}{s^2+\omega^2}$

Table 1: Common Laplace Transform Pairs

Appendix Example A-1: Root Locus Design Steps

1. Plot the open-loop poles and zeros 2. Determine the root locus branches 3. Find breakaway and break-in points 4. Calculate asymptotes 5. Determine gain values for desired pole locations 6. Verify closed-loop performance

Appendix C: MATLAB Code Listings

This appendix shows automatic lettering continues (Appendix C) and contains detailed code implementations.

```
% Complete control system design script
function design_pid_controller()
    % Plant definition
    s = tf('s');
    Gp = 10/(s*(s+2));

    % PID controller design
    Kp = 5; Ki = 2; Kd = 0.5;
    Gc = pid(Kp, Ki, Kd);

    % Closed-loop system
    T = feedback(Gc*Gp, 1);

    % Analysis
    step(T);
    margin(Gc*Gp);

    fprintf('Design complete\n');
end
```

All features of the homework class have been demonstrated in the order they appear in the class file definition. This organization makes it easier to understand the feature hierarchy and locate specific functionality.