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 for aligned equations.

The hwmath environment provides automatic alignment on equals signs using the `textbackslash eq` shorthand:

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

Problem 6: Show numbered equations for referencing.

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

$$E = mc^2 \tag{1}$$
$$F = ma \tag{2}$$
$$P = VI \tag{3}$$

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

Problem 7: Demonstrate complex mathematical formatting capabilities.

(a)

Matrix operations and state-space representation.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \end{aligned}$$

Where the system matrices are:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -3 & -4 \end{bmatrix}, \quad \mathbf{B} \quad = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

(b)

Convolution integral and Laplace transforms.
The convolution of two functions:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

Laplace transform definition:

$$\mathcal{L}\{f(t)\} = \int_{0}^{\infty} f(t)e^{-st}dt$$

Problem 8: Demonstrate the note command for highlighting important information.

The note command displays text in bold, red, and large font:
IMPORTANT: Check all calculations before submitting
TODO: Add numerical verification
FIX ME: Verify units are consistent
NOTE
This is useful for marking sections that need attention during review.

Problem 9: Demonstrate the hwmatlab environment for MATLAB code.

(a)

MATLAB function with caption.

Listing 1: Second Order System Analysis

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

    s = tf('s');
    G = K * wn^2 / (s^2 + 2*zeta*wn*s + wn^2);

    [y, t] = step(G, tfinal);

    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);
```

Problem 10: Demonstrate the hwpython environment for Python code.

(a)

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)
```

```python
    # 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(f'Second Order System (wn={wn}, zeta={zeta}, K={K})')
    plt.grid(True, alpha=0.3)
    plt.legend()
    plt.show()

    return y, t
```

(b)

Python data analysis without caption.

```python
# System parameters
wn = 5.0        # Natural frequency
zeta = 0.3      # Damping ratio
K = 2.0         # Gain

# Performance metrics
overshoot = np.exp(-zeta*np.pi/np.sqrt(1-zeta**2)) * 100
settling_time = 4 / (zeta * wn)
peak_time = np.pi / (wn * np.sqrt(1 - zeta**2))

print(f"Overshoot: {overshoot:.1f}%")
print(f"Settling time: {settling_time:.2f} s")
print(f"Peak time: {peak_time:.2f} s")
```

Problem 11: Demonstrate the hwterminal environment for terminal output.

The hwterminal environment displays terminal commands and output with appropriate styling:

```
>> wn = 5; zeta = 0.3; K = 1;
>> s = tf('s');
>> G = K*wn^2/(s^2 + 2*zeta*wn*s + wn^2);
>> step(G);
>> stepinfo(G)

ans =
       RiseTime: 0.2927
   SettlingTime: 2.6667
    SettlingMin: 0.9000
    SettlingMax: 1.3625
      Overshoot: 36.2467
     Undershoot: 0
           Peak: 1.3625
       PeakTime: 0.6283
```

Listing 3: MATLAB Simulation Output

```
$ pip install control matplotlib numpy
Collecting control
  Downloading control-0.9.4-py3-none-any.whl (719 kB)
     ======================================= 719.0/719.0 kB 5.2 MB/s
Collecting matplotlib
  Downloading matplotlib-3.7.2-cp311-cp311-macosx_10_12_x86_64.whl (7.8 MB)
     ======================================= 7.8/7.8 MB 8.4 MB/s
Successfully installed control-0.9.4 matplotlib-3.7.2 numpy-1.24.3
```

Listing 4: Python Package Installation

Problem 12: Showcase advanced mathematical typesetting capabilities.

(a)

Partial derivatives and multivariable calculus.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad \text{(Laplace's equation)}$$
$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = 0$$

(b)

Complex analysis and frequency domain.

$$H(j\omega) = \frac{K}{j\omega + a}$$
$$|H(j\omega)| = \frac{K}{\sqrt{\omega^2 + a^2}}$$
$$\angle H(j\omega) = -\arctan\left(\frac{\omega}{a}\right)$$

Problem 13: Demonstrate piecewise functions and special mathematical constructs.

(a)

Unit step and impulse functions.

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$$
$$\delta(t) = \begin{cases} \infty & t = 0 \\ 0 & t \neq 0 \end{cases}, \quad \int_{-\infty}^{\infty} \delta(t) dt = 1$$

(b)

System response types based on damping.

$$\text{Response type} = \begin{cases} \text{Overdamped} & \zeta > 1 \\ \text{Critically damped} & \zeta = 1 \\ \text{Underdamped} & 0 < \zeta < 1 \\ \text{Undamped} & \zeta = 0 \end{cases}$$

Problem 14: Show integration with standard LaTeX environments and packages.

The homework class works seamlessly with standard LaTeX features:

- Standard lists and enumerations
- Mathematical symbols: $\alpha, \beta, \gamma, \Delta, \Omega$
- Greek letters and special symbols: $\infty, \partial, \nabla, \sum, \int$
- Text formatting: **bold**, *italic*, `monospace`

1. First numbered item
2. Second numbered item with math: $E = mc^2$

3. Third item with inline code: `plot(x, y)`

Problem 15: Comprehensive example combining all features.

FINAL EXAMPLE: Combines multiple features

(a)

Design a control system.
Given plant: $G_p(s) = \frac{10}{s(s+2)}$
Design a PID controller: $G_c(s) = K_p + \frac{K_i}{s} + K_d s$

$$G_{ol}(s) = G_c(s)G_p(s) \tag{4}$$

$$G_{cl}(s) = \frac{G_{ol}(s)}{1 + G_{ol}(s)} \tag{5}$$

(b)

MATLAB implementation.

Listing 5: PID Controller Design and Analysis

```
% Plant transfer function
s = tf('s');
Gp = 10 / (s * (s + 2));

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

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

% Analysis
step(Gcl);
title('Closed-Loop Step Response');
grid on;

% Display controller
fprintf('PID Controller: Kp=%.1f, Ki=%.1f, Kd=%.1f\n', Kp, Ki, Kd);
```

(*)

Bonus: Python verification.

Listing 6: Python Control Systems Verification

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

# Plant transfer function
Gp = ct.tf([10], [1, 2, 0])

# PID controller
Kp, Ki, Kd = 5, 2, 0.5
Gc = ct.tf([Kd, Kp, Ki], [1, 0])

# Closed-loop system
Gcl = ct.feedback(Gc * Gp, 1)

# Step response
```

```
t = np.linspace(0, 5, 1000)
y, t = ct.step_response(Gcl, t)

plt.figure(figsize=(10, 6))
plt.plot(t, y, 'b-', linewidth=2)
plt.xlabel('Time (s)')
plt.ylabel('Output')
plt.title('PID Controlled System Step Response')
plt.grid(True, alpha=0.3)
plt.show()

print(f"Controller: PID({Kp}, {Ki}, {Kd})")
```

This example demonstrates the complete integration of mathematical typesetting, code environments, problem structuring, and highlighting features available in the homework class.

Appendix A

This is the first appendix. It contains additional derivations and supplementary material.

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 Reference Tables

T

his appendix contains useful reference tables and formulas.

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	$\sin(\omega t) \cdot u(t)$	$\frac{\omega}{s^2+\omega^2}$

Table 1: Common Laplace Transform Pairs

Appendix C

T

his is the third appendix, showing automatic lettering continues (Appendix C).

NOTEAppendices are useful for including detailed calculations, reference materials, or code listings that would interrupt the flow of the main document.