# Task 1: Open-loop and Closed-loop Analysis

* Given Open-loop Transfer Function

4

H(s) = s3 + 5s2 + 9s + 5

* Code:

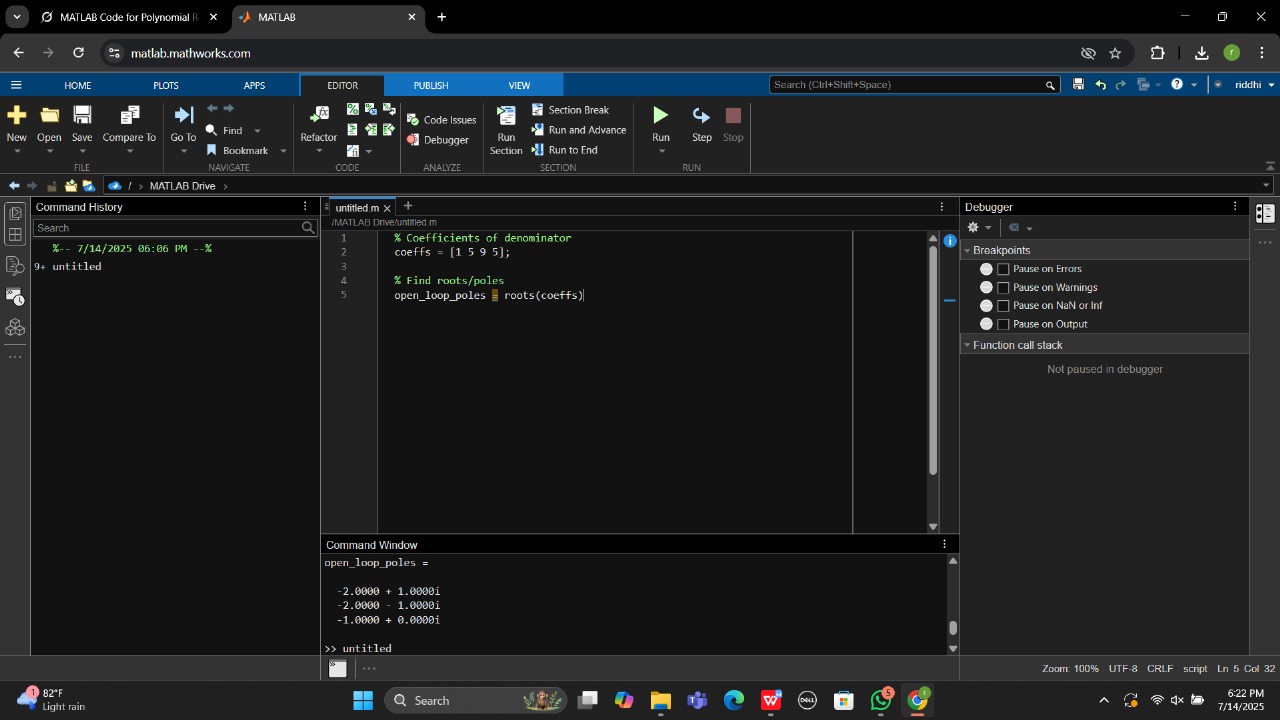
% Coefficients of denominator

coeffs = [1 5 9 5];

% Find roots/poles

open\_loop\_poles = roots(coeffs)

* Output:



**Plot the Block Diagram**

**+--------+ + +**

**-->| K | -------------> | H(s) | +**

**+--------+ +--------+ |**

**^ |**

**| |**

**+ (-) +**

### Closed-loop Transfer Function

For a unity feedback system, the closed-loop transfer function T(s) T(s) T(s) is defined as:

T(s)=H(s)1+H(s)T(s) = \frac{H(s)}{1 + H(s)}T(s)=1+H(s)H(s)​

Given the open-loop transfer function:

H(s)=45s3+5s2+9s+5H(s) = \frac{4}{5s^3 + 5s^2 + 9s + 5}H(s)=5s3+5s2+9s+54​

Substitute H(s) H(s) H(s) into the closed-loop transfer function:

T(s)=45s3+5s2+9s+51+45s3+5s2+9s+5T(s) = \frac{\frac{4}{5s^3 + 5s^2 + 9s + 5}}{1 + \frac{4}{5s^3 + 5s^2 + 9s + 5}}T(s)=1+5s3+5s2+9s+54​5s3+5s2+9s+54​​

Simplify the expression:

T(s)=45s3+5s2+9s+5+4=45s3+5s2+9s+9T(s) = \frac{4}{5s^3 + 5s^2 + 9s + 5 + 4} = \frac{4}{5s^3 + 5s^2 + 9s + 9}T(s)=5s3+5s2+9s+5+44​=5s3+5s2+9s+94​

Thus, the closed-loop transfer function is:

T(s)=45s3+5s2+9s+9T(s) = \frac{4}{5s^3 + 5s^2 + 9s + 9}T(s)=5s3+5s2+9s+94​

* Code:

k = 2;

a3 = 1;

a2 = 5;

a1 = 9;

a0 = 5 + 4 \* k;

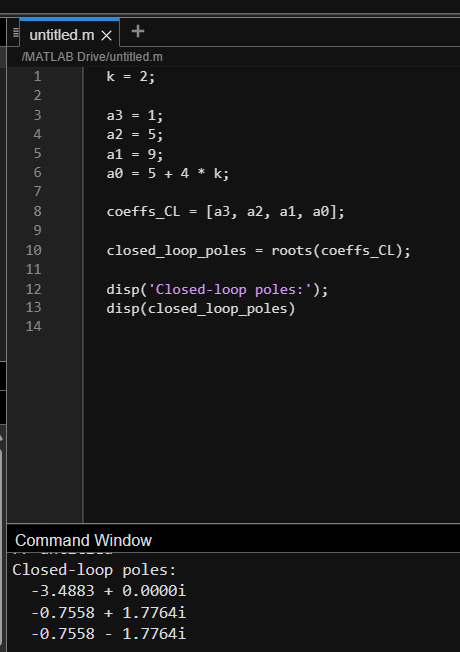
coeffs\_CL = [a3, a2, a1, a0];

closed\_loop\_poles = roots(coeffs\_CL);

disp('Closed-loop poles:');

disp(closed\_loop\_poles)

* Output:



**Task 2: MATLAB Function for Pole Calculation**

num = [4];

den = [1 5 9];

H = tf(num, den);

figure;

rlocus(H);

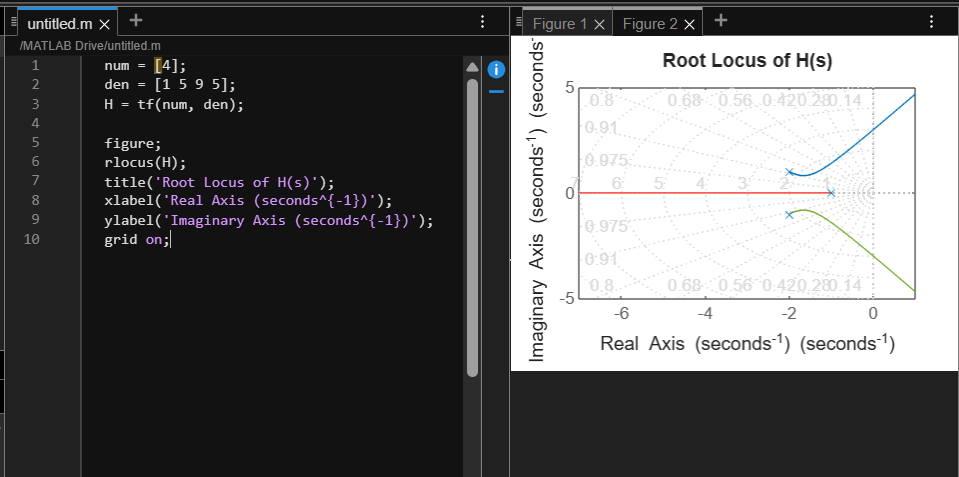
title('Root Locus of H(s)');

xlabel('Real Axis (seconds^{-1})');

ylabel('Imaginary Axis (seconds^{-1})');

grid on;

* Output:



**Task 3: Develop a PID controller in MATLAB for the system H(s) subject to the following** **constraints.**

Design a **PID controller** for a given plant (system) H(s)H(s)H(s), subject to the following **time-domain performance constraints**:

### ****System Transfer Function****

The open-loop plant is:

H(s)=1s3+5s2+9s+5H(s) = \frac{1}{s^3 + 5s^2 + 9s + 5}H(s)=s3+5s2+9s+51​

This is a **third-order stable system** (denominator polynomial).

### ****Performance Requirements****

You must design a **PID controller** such that the **closed-loop system** meets the following:

| **Specification** | **Constraint** |
| --- | --- |
| Peak Overshoot (MpM\_p) | < 10% |
| Rise Time (trt\_r) | < 1 second |
| Settling Time (2%) (tst\_s) | < 5 seconds |

function [err, M, t\_r, t\_s, gm, pm] = pid\_analysis(P, I, D)

numerator = 4;

denominator = [1 5 9 5];

plant\_model = tf(numerator, denominator);

pid\_model = D \* tf([1 0], 1) + P \* tf([1 0], 1) + I \* tf(1, [1 0]);

loop\_open = series(pid\_model, plant\_model);

loop\_closed = feedback(loop\_open, 1);

time\_array = 0:0.01:20;

[response, t] = step(loop\_closed, time\_array);

err = sqrt(mean((1 - response).^2));

metrics = stepinfo(loop\_closed);

M = metrics.Overshoot;

t\_r = metrics.RiseTime;

t\_s = metrics.SettlingTime;

[Gm, Pm] = margin(loop\_open);

gm = 20 \* log10(Gm);

pm = Pm;

end

[P\_val, I\_val, D\_val] = deal(1.2, 0.8, 0.5);

[err\_val, M\_val, t\_r\_val, t\_s\_val, gm\_val, pm\_val] = pid\_analysis(P\_val, I\_val, D\_val);

fprintf('RMS Error = %.4f\n', err\_val);

fprintf('Overshoot = %.2f%%\n', M\_val);

fprintf('Rise Time = %.2f sec\n', t\_r\_val);

fprintf('Settling Time = %.2f sec\n', t\_s\_val);

fprintf('Gain Margin = %.2f dB\n', gm\_val);

fprintf('Phase Margin = %.2f degrees\n', pm\_val);

