Problem 2-A

# **CODE:**

% PROBLEM 2-A

% Clear workspace

close all

clear

clc

% Add parser and solver to path

addpath(genpath('C:\Users\tsamak\Downloads\MathWorks\Toolboxes\archives\required\YALMIP'))

addpath(genpath('C:\Users\tsamak\Downloads\MathWorks\Toolboxes\archives\required\SeDuMi'))

% Define the system matrices A, B, and C

A = [-1.01887 0.90506; 0.82225 -1.07741];

B = [0.00203; -0.00164];

C = [15.87875, 1.48113];

D = 0;

% Define the LMI variables

P = sdpvar(2, 2);

Gamma\_ep = sdpvar(1, 1);

% Define the LMI constraints

LMI1 = [-Gamma\_ep\*eye(1), C\*P\*C'; C\*P\*C', -eye(1)] <= 0;

LMI2 = A\*P+P\*A'+B\*B' <= 0;

LMI3 = -P <= 0;

% Set up the objective

Objective = Gamma\_ep;

% Define the solver settings (use an LMI solver like YALMIP with a solver of your choice)

options = sdpsettings('verbose', 1, 'solver', 'sedumi');

% Solve the LMI problem

solution = optimize([LMI1, LMI2, LMI3], Objective, options);

if solution.problem == 0

% Extract the optimal solution

optimal\_Gamma\_ep = value(Gamma\_ep)^(0.25);

% Display the result

fprintf('Energy-to-Peak Gain (Gamma\_ep): %.4f\n', optimal\_Gamma\_ep);

else

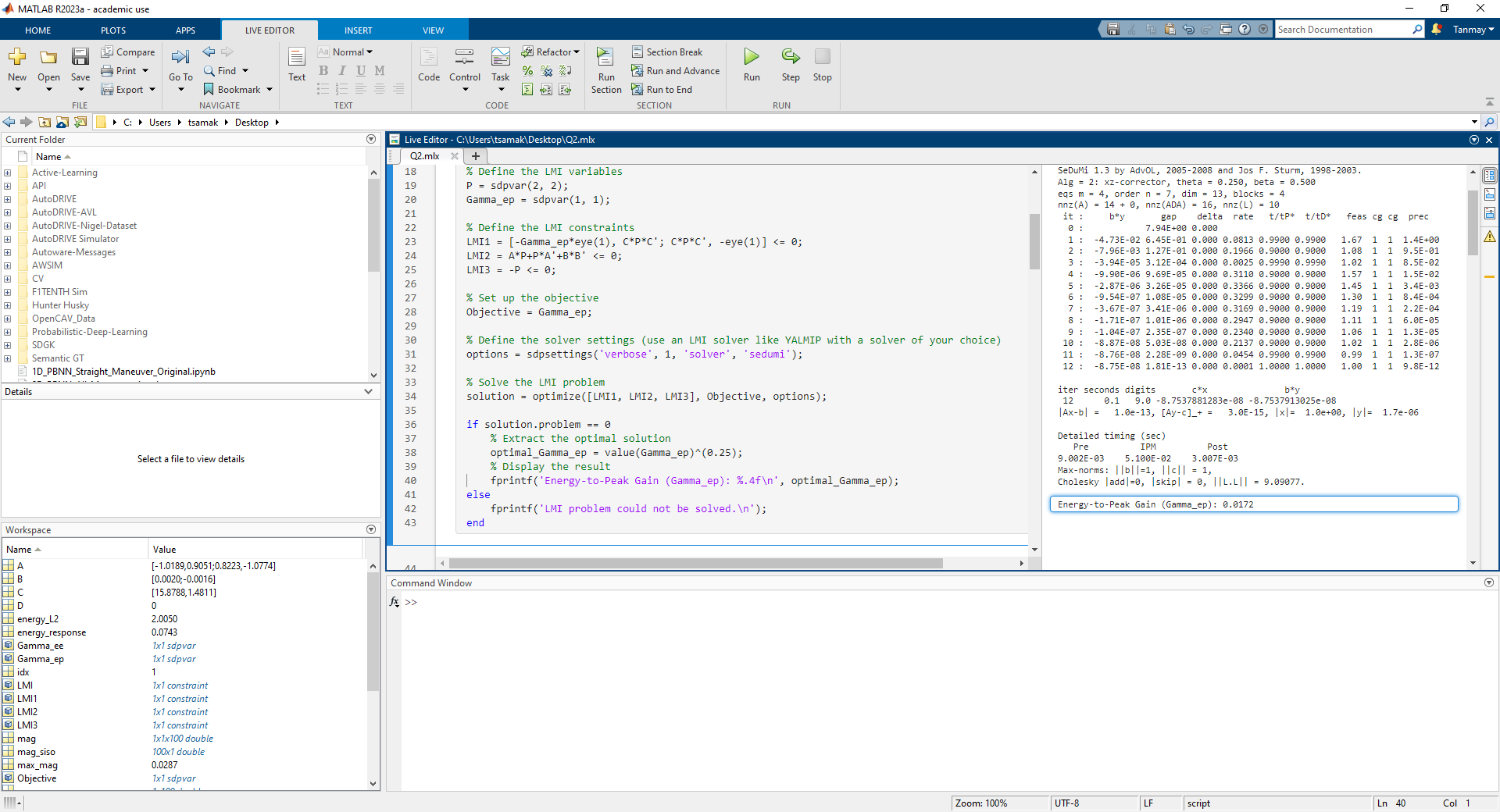
fprintf('LMI problem could not be solved.\n');

end

# **OUTPUT:**

Energy-to-Peak Gain (Gamma\_ep): 0.0172

# **SCREENSHOT:**



Problem 2-B

# **CODE:**

% PROBLEM 2-B

% Define the state-space system

A = [-1.01887 0.90506; 0.82225 -1.07741];

B = [0.00203; -0.00164];

C = [15.87875 1.48113];

D = 0;

% Define the disturbance signal w\_g(t)

t = 0:0.01:2; % Time vector from 0 to 2 with a step size of 0.01

w\_g = 2 \* (t >= 0 & t <= 1); % Pulse disturbance, 2 for 0 ≤ t ≤ 1, 0 otherwise

plot(t,w\_g)

title('Pulse Disturbance')

% Calculate the L2-norm (energy) of the disturbance signal

energy\_L2 = sqrt(trapz(t, w\_g.^2));

disp(['Energy of the disturbance signal: ', num2str(energy\_L2)]);

% Simulate the system response

sys = ss(A, B, C, D);

[y, t, x] = lsim(sys, w\_g, t);

% Estimate the energy of the response to a pulse disturbance

energy\_response = sqrt(trapz(t, y.^2)); % L2 norm

fprintf('Estimated energy of system response: %f\n', energy\_response);

if optimal\_Gamma\_ep >= energy\_response

fprintf('System response is consistent with Gamma\_ep.\n');

else

fprintf('System response is not consistent with Gamma\_ep.\n');

end

% Plot the response

subplot(2, 1, 1);

plot(t, y);

xlabel('Time (s)');

ylabel('Vertical Acceleration (y)');

title('System Response to Disturbance');

subplot(2, 1, 2);

plot(t, x);

xlabel('Time (s)');

ylabel('State Vector (x)');

legend('\alpha', 'q');

# **OUTPUT:**

|  |  |
| --- | --- |
| A graph of a pulse  Description automatically generated |  |
| Disturbance Signal | System Response |

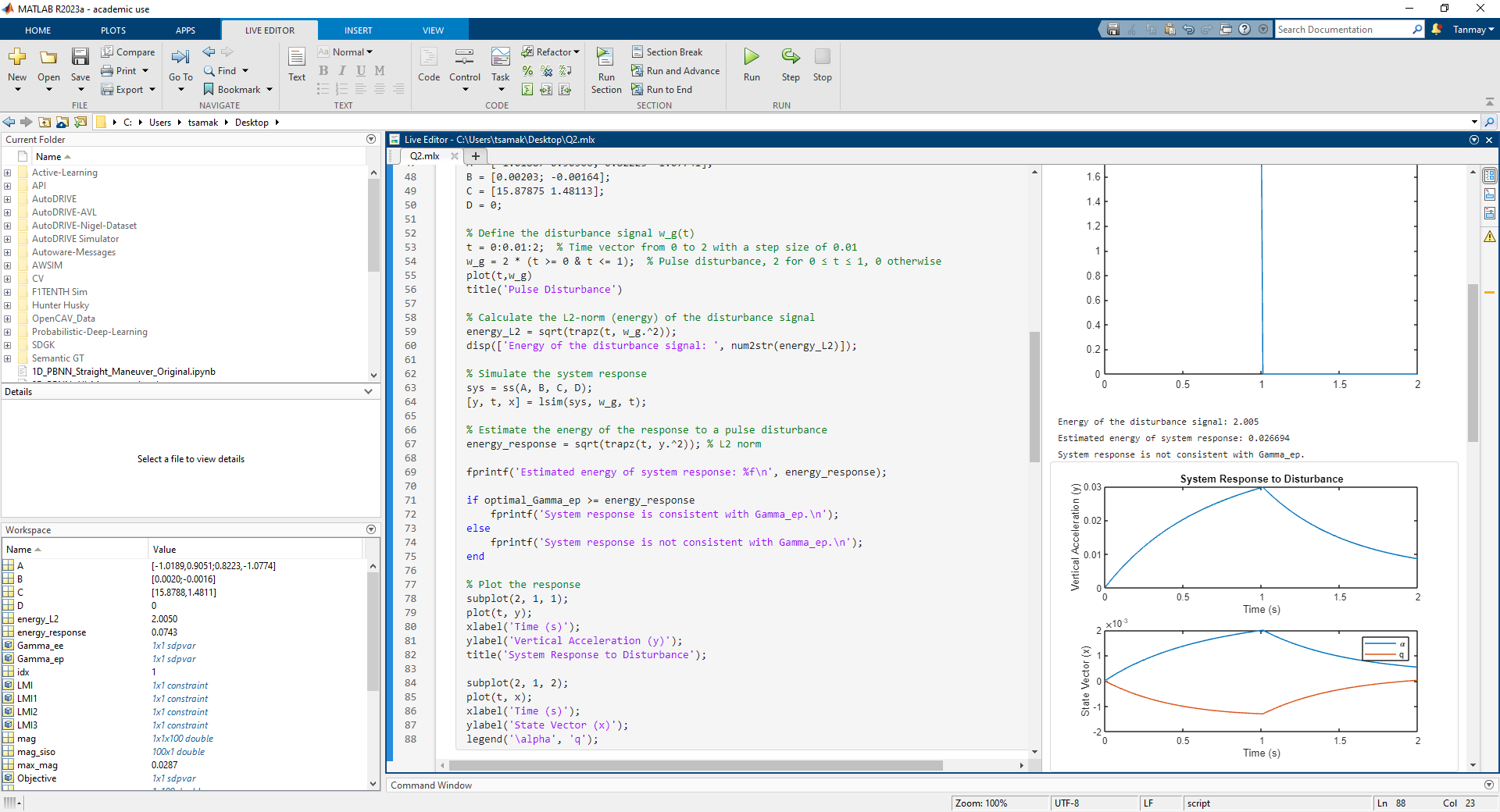
Energy of the disturbance signal: 2.005

Estimated energy of system response: 0.026694

System response is not consistent with Gamma\_ep.

The inconsistency indicates that the system is not very robust to given disturbance.

# **SCREENSHOT:**



Problem 2-C

# **CODE:**

% PROBLEM 2-C

% Define the system matrices A, B, and C

A = [-1.01887 0.90506; 0.82225 -1.07741];

B = [0.00203; -0.00164];

C = [15.87875, 1.48113];

D = 0;

% Define the LMI variables

P = sdpvar(2, 2);

Gamma\_ee = sdpvar(1, 1);

% Define the LMI constraints

LMI = [A'\*P + P\*A, P\*B, C'; B'\*P, -Gamma\_ee\*eye(1), D'; C, D, -Gamma\_ee\*eye(1)] <= 0;

% Set up the objective

Objective = Gamma\_ee;

% Define the solver settings (use an LMI solver like YALMIP with a solver of your choice)

options = sdpsettings('verbose', 1, 'solver', 'sedumi');

% Solve the LMI problem

solution = optimize(LMI, Objective, options);

if solution.problem == 0

% Extract the optimal solution

optimal\_Gamma\_ee = value(Gamma\_ee);

% Estimate the energy of the response to a pulse disturbance (adjust the time horizon as needed)

T = 10; % Adjust the time horizon as needed

sys = ss(A, B, C, 0);

t = 0:0.01:T;

[y, t] = lsim(sys, ones(size(t)), t); % Pulse disturbance input is ones

energy\_response = sqrt(trapz(t, y.^2)); % L2 norm

fprintf('Energy-to-Energy Gain (Gamma\_ee): %f\n', optimal\_Gamma\_ee);

fprintf('Estimated energy of system response: %f\n', energy\_response);

if optimal\_Gamma\_ee >= energy\_response

fprintf('System response is consistent with Gamma\_ee.\n');

else

fprintf('System response is not consistent with Gamma\_ee.\n');

end

else

fprintf('LMI problem could not be solved.\n');

end

# **OUTPUT:**

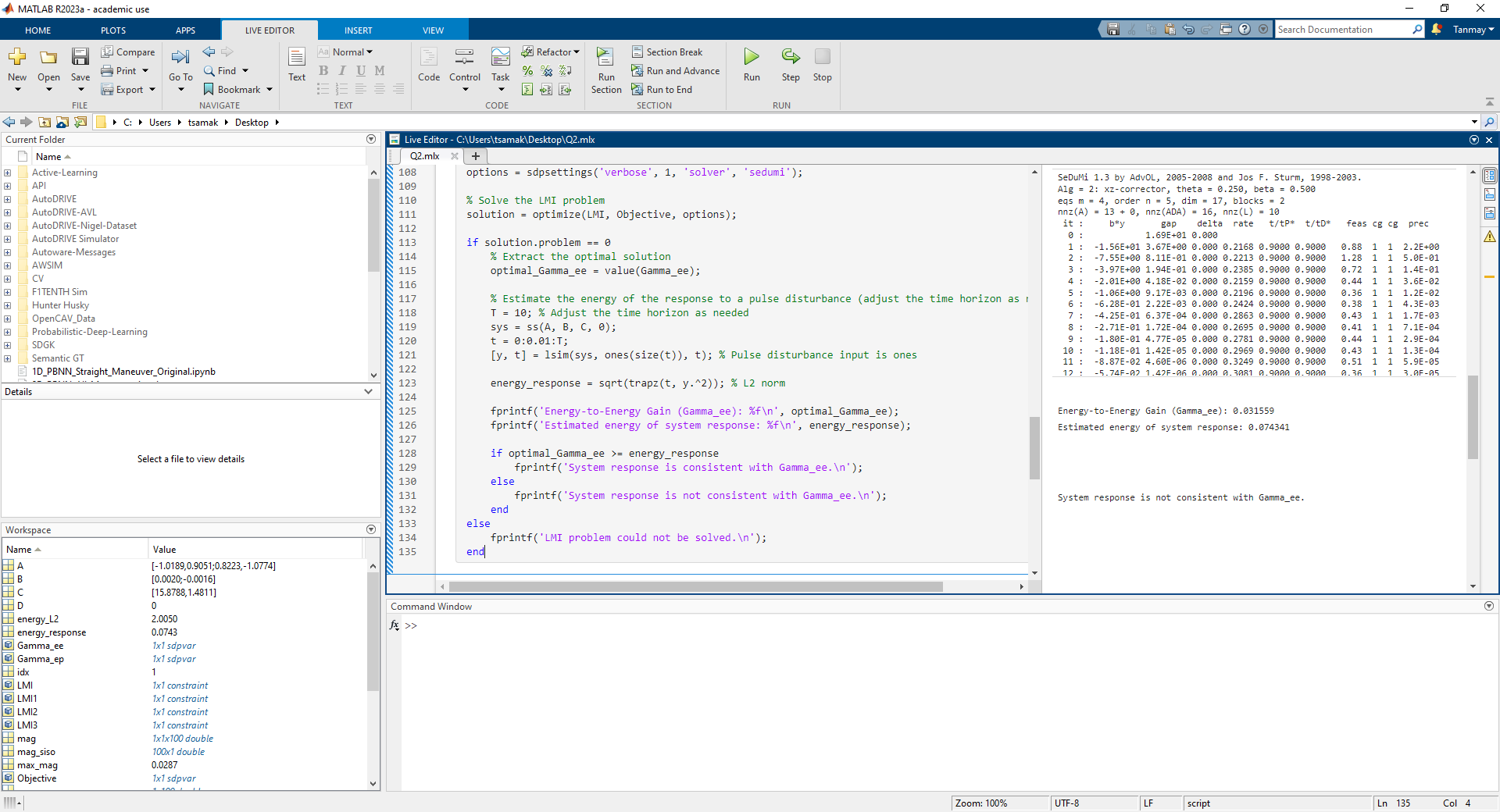
Energy-to-Energy Gain (Gamma\_ee): 0.031559

Estimated energy of system response: 0.074341

System response is not consistent with Gamma\_ee.

The inconsistency indicates that the system is not very robust to given disturbance.

# **SCREENSHOT:**



Problem 2-D

# **CODE:**

% PROBLEM 2-D

% Define the state-space system

A = [-1.01887 0.90506; 0.82225 -1.07741];

B = [0.00203; -0.00164];

C = [15.87875 1.48113];

D = 0;

sys = ss(A, B, C, D);

% Define a range of frequencies

omega = logspace(-1, 2, 100); % Adjust the frequency range as needed

% Calculate the frequency response of the system

[mag, ~, ~] = bode(sys, omega);

% Extract the magnitude for the SISO system

mag\_siso = squeeze(mag);

% Find the peak magnitude and its corresponding frequency

[max\_mag, idx] = max(mag\_siso);

peak\_freq = omega(idx);

% Plot the magnitude response

figure;

semilogx(omega, 20\*log10(mag\_siso), 'b'); % Plot in decibels

title('Magnitude Response |G(j\omega)|');

xlabel('Frequency (rad/s)');

ylabel('Magnitude (dB)');

grid on;

% Display the peak magnitude and frequency

fprintf('Peak Magnitude (dB): %.4f dB at Frequency %.4f rad/s\n', 20\*log10(max\_mag), peak\_freq);

fprintf('Energy-to-Energy Gain (Gamma\_ee): %.4f\n', max\_mag);

if abs(optimal\_Gamma\_ee-max\_mag) <= 0.01

fprintf('Peak is equivalent to Gamma\_ee.\n');

else

fprintf('Peak is not equivalent to Gamma\_ee.\n');

end

# **OUTPUT:**

A graph of a function

Description automatically generated

Peak Magnitude (dB): -30.8437 dB at Frequency 0.1000 rad/s

Energy-to-Energy Gain (Gamma\_ee): 0.0287

Peak is equivalent to Gamma\_ee.

# **SCREENSHOT:**

