Simulation 2
Quarter Car Model

Student: Andres F. Hoyos

Consider a two-mass quarter model with the following attributes:

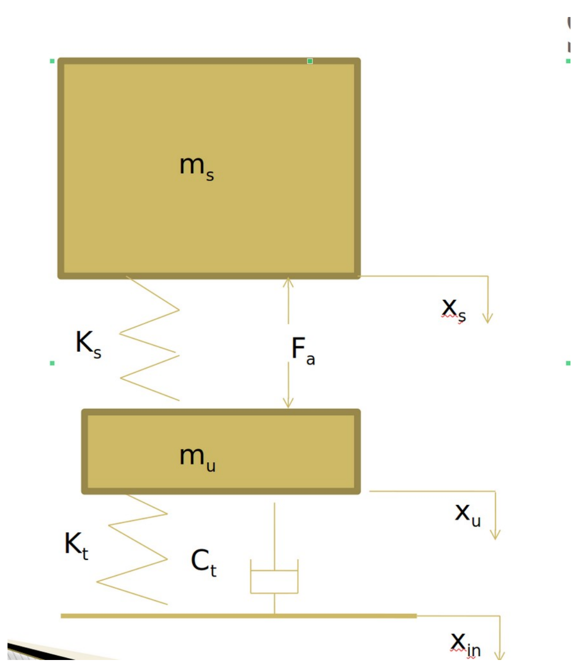
Ks=100 lbs/in
Kt=1000 lbs/in
Ws=1000 lbs
Wu=100 lbs
Ct=0.01 lbs-sec/in

Part 1: Linear Analysis

Determine the optimal suspension damping parameter using frequency response methods and human sensitivity to vibrations. Construct a time domain simulation of the two-mass quarter car model and validate the choice of suspension damping parameter with results of the frequency analysis.

Consider inputs consisting of:
1.) A one inch step function beginning at t=1s.
2.) A ¼ inch step function beginning at t=1s.
3.) A +/- 1 inch sine wave input at 1 Hz.
4.) A +/- ¼ inch sine wave at 10 Hz.
5.) A 2 inch pothole 3 ft wide at 50 mph.

Figure 1: Quarter car model diagram.



The equations of motion for the quarter car model are:

$$\ddot{x}_s = -\left(\frac{K_s}{m_s}\right) * x_s - \left(\frac{C_s}{m_s}\right) * \dot{x}_s + \left(\frac{K_s}{m_s}\right) * x_u + \left(\frac{C_s}{m_s}\right) * \dot{x}_u \quad ,$$

$$\ddot{x}_u = \left(\frac{K_s}{m_u}\right) x_s + \left(\frac{C_s}{m_u}\right) \dot{x}_s + \left(\frac{-(K_t + K_s)}{m_u}\right) x_u + \left(\frac{-(C_t - C_s)}{m_u}\right) \dot{x}_u + \left(\frac{K_t}{m_u}\right) x_{\text{in}} + \left(\frac{C_t}{m_u}\right) \dot{x}_{\text{in}} \quad .$$

The linear system can be expressed in state-space form:

$$\begin{bmatrix} \dot{x}_s \\ \ddot{x}_s \\ \dot{x}_u \\ \ddot{x}_u \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-K_s}{m_s} & \frac{-C_s}{m_s} & \frac{K_s}{m_s} & \frac{C_s}{m_s} \\ 0 & 0 & 0 & 1 \\ \frac{K_s}{m_u} & \frac{C_s}{m_u} & \frac{-(K_s + K_t)}{m_u} & \frac{-(C_s + C_t)}{m_u} \end{bmatrix} \cdot \begin{bmatrix} x_s \\ \dot{x}_s \\ x_u \\ \dot{x}_u \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{K_t}{m_u} & \frac{C_t}{m_u} \end{bmatrix} \cdot \begin{bmatrix} x_{\text{in}} \\ \dot{x}_{\text{in}} \end{bmatrix}$$

The C matrix will determine which output is required from the system. For the **isolation function**,

Determining the optimal suspension damping coefficient:

A simulation in Matlab was developed to determine the optimal damper coefficient selection. There is an important trade-off when selecting a correct damping coefficient value given that increasing the value of C_s decreases the wheel-hop and ride transmissibilities at the natural frequencies, but it can significantly amplify the isolation function and the transmissibility functions specially in the 4-8 Hz region where the humans are more sensitive to vertical vibrations.

For this reason, a balanced approach that takes into account the transmissibility plots and the isolation plot is used. The following code in Matlab was developed to solve the exercise:

```
%% Simulation 1:
% Parameters:
Ks = 100; %lbs/in
Kt = 1000; % lbs/in
Ws = 1000; % lbs
Wu = 100; % lbs
Ct = 0.01; % lbs-sec/in
g = 386.06; % in/sec^2
ms = Ws / g; % Sprung mass lbs*sec^2/in
mu = Wu / g; % Unsprung mass in lbs*sec^2/in
freq_low_bound = 4*2*pi; % 4 Hz = 25.13 rad/sec
```

```
freq_up_bound = 8*2*pi; % 8 Hz = 50.27 rad / sec
Cs_array = logspace(log10(0.1), log10(100), 10); % In lbs/in/sec
% Legend arrays:
legendEntries = cell(1, 2*length(Cs_array) + 2); % Initialize the cell array
legendCounter = 1; % Initialize a counter for legend entries
legendEntriesBottom = cell(1, length(Cs_array) + 2);
legendCounterBottom = 1; % Initialize a counter for legend entries
figure(1); % Create the first figure outside the loop
subplot(2, 1, 1)
xscale log;
grid on;
hold on;
subplot(2, 1, 2)
xscale log;
grid on;
hold on;
for i=1:length(Cs_array)
Cs = Cs_array(i);
% plot:
subplot(2, 1, 1);
[mgs, mgu, mg_iso, w] = twomass_rel_damp(Ks, Kt, Cs, Ct, ms, mu);
semilogx(w, mgs, 'o-'); % Plot the magnitude vs frequency for the sprung mass
legendEntries{legendCounter} = sprintf('Sprung mass Cs = %.2f, Ct = %.2f', Cs, Ct);
legendCounter = legendCounter + 1;
semilogx(w, mgu, '--'); % Plot the magnitude vs frequency for the unsprung mass
legendEntries{legendCounter} = sprintf('Unsprung mass Cs = %.2f, Ct = %.2f', Cs, Ct);
legendCounter = legendCounter + 1;
% plot:
subplot(2, 1, 2);
mg_iso_db = 20*log(mg_iso);
semilogx(w, mg_iso_db, 'o-'); % Plot the magnitude vs frequency for the sprung mass
legendEntriesBottom{legendCounterBottom} = sprintf('Sprung mass Cs = %.2f, Ct = %.2f', Cs, Ct);
legendCounterBottom = legendCounterBottom + 1;
end
% Initial guess for Cs
Cs_initial = 3.0; % Starting point for the search
options = optimoptions('fmincon','Display','iter','Algorithm','sqp'); % FIXME: Use me
A = []; b = []; Aeq = []; Beq = []; lb = 1.0; ub = 100;
[Cs_optimal, cost_optimal] = fmincon(@objectiveFunction, Cs_initial, A, b, Aeq, Beq, lb, ub, [],
options); % TODO: Add options to show optimization by iter
% Output the optimal Cs
fprintf('Optimal Cs: %f, with a cost of %f\n', Cs_optimal, cost_optimal);
subplot(2, 1, 1);
[mgs, mgu, mg_iso, w] = twomass_rel_damp(Ks, Kt, Cs_optimal, Ct, ms, mu);
mg_iso_db = 20*log(mg_iso);
semilogx(w, mgs, 'o-', LineWidth=3.0); % Plot the magnitude vs frequency for the sprung mass
legendEntries{legendCounter} = sprintf('Sprung mass transm. for selected opt. damper Cs = %.2f, Ct =
%.2f', Cs_optimal, Ct);
legendCounter = legendCounter + 1;
```

```
semilogx(w, mgu, '--', LineWidth=3.0); % Plot the magnitude vs frequency for the unsprung mass
legendEntries{legendCounter} = sprintf('Unsprung mass transm. for selected opt. damper Cs = %.2f,
Ct = %.2f', Cs_optimal, Ct);
legendCounter = legendCounter + 1;
title('Relative damping transmissibility plot varying Cs');
legend(legendEntries{1:2*length(Cs_array) + 2}); % Create the legend for the first figure
xlabel('frequency [rad/sec]');
ylabel('amplitude ratio');
hold off;
subplot(2, 1, 2);
semilogx(w, mg_iso_db, '--', LineWidth=3.0); % Plot the magnitude vs frequency for the unsprung
mass
legendEntriesBottom{legendCounterBottom} = sprintf('Sprung mass isolation func. for selected opt.
damper Cs = %.2f, Ct = %.2f', Cs_optimal, Ct);
legendCounterBottom = legendCounterBottom + 1;
% human_sensitivity = 4*ones(length(w), 1);
% semilogx(w, human_sensitivity, '--', LineWidth=3);
legendEntriesBottom{legendCounterBottom} = sprintf('Human sensitivity');
legend(legendEntriesBottom{:}); % Create the legend for the first figure
xline(freq_low_bound);
xline(freq_up_bound);
ylabel('magnitude (dB)')
hold off;
%% The system with the optimal damper is loaded as a state-space in Matlab for
% Bode plots:
% Mathematical model:
Cs = Cs_optimal;
A_qc = [
0, 1, 0, 0;
-Ks/ms, -Cs/ms, Ks/ms, Cs/ms;
0, 0, 0, 1;
Ks/mu, Cs/mu, -(Ks+Kt)/mu, -(Cs+Ct)/mu];
B_qc = [
0, 0;
0, 0;
0, 0;
Kt/mu, Ct/mu];
C_qc = [
1, 0, 0, 0;
0, 0, 1, 0];
D_qc = [
0, 0;0, 0];
% Assuming the optimal damper value and other parameters are defined
% Generate the state-space model
sys_qc = ss(A_qc, B_qc(:,1), C_qc(2,:), D_qc(1,1));
figure;
bode(sys_qc);
%% Isolation function:
C_isolation = [0, 1, 0, 0];
```

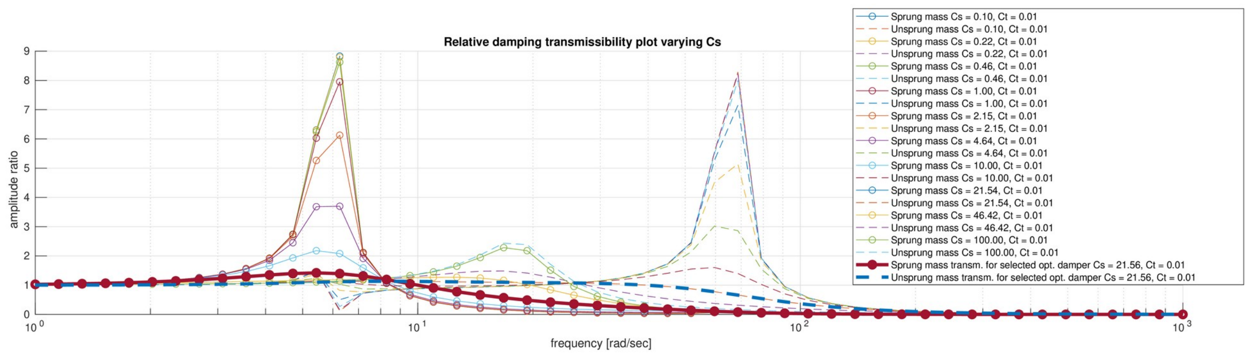
```
% Bode:
figure;
hold on;
xscale log;
grid on;
[mag, phase, wout] = bode(sys_qc);
magdB = 20*log10(mag); % Convert magnitude to dB
semilogx(squeeze(wout), squeeze(magdB), 'o-', LineWidth=3.0); % Plot the magnitude vs frequency
for the sprung mass
yline(4);
xline(freq_low_bound);
xline(freq_up_bound);
%% Functions:
% Objective function that the optimization algorithm will minimize
function cost = objectiveFunction(Cs)
% Parameters:
Ks = 100; %lbs/in
Kt = 1000; % lbs/in
Ws = 1000; % lbs
Wu = 100; % lbs
Ct = 0.01; % lbs-sec/in
g = 386.06; % in/sec^2
ms = Ws / g; % Sprung mass lbs*sec^2/in
mu = Wu / g; % Unsprung mass in lbs*sec^2/in
% Assume twomass_rel_damp is modified to accept Cs and return a cost metric
[mgs, mgu, mg_iso, w] = twomass_rel_damp(Ks, Kt, Cs, Ct, ms, mu);
mg_iso_db = 20*log(mg_iso);
% Cost function formulation:
relevantIndices = (w >= 4*2*pi) & (w <= 8*2*pi);
relevantIndices2 = (w >= 4) & (w <= 8);
cost = 2*mean(mgs(relevantIndices)) + 10*mean(mgs(relevantIndices2) +
0.075*mean(abs(mg_iso(relevantIndices))));
% cost = mean(mg_iso(relevantIndices)); %max(mgs(relevantIndices));Cs = Cs_array(i);
end
function [mgs, mgu, mg_iso, w] = twomass_rel_damp(Ks, Kt, Cs, Ct, ms, mu)
% Mathematical model:
A=[ 0, 1, 0, 0;
-Ks/ms, -Cs/ms, Ks/ms, Cs/ms;
0, 0, 0, 1;
Ks/mu, Cs/mu, -(Ks+Kt)/mu, -(Cs+Ct)/mu];
B=[0, 0;
0, 0;
0, 0;
Kt/mu, Ct/mu];
C=[1, 0, 0, 0;
0, 0, 1, 0];
D=[0, 0;0, 0];
%twomass calculates the frequency response of a two-mass
[mag, phase, w]=bode(ss(A,B(:,1), C(1,:), [0]),logspace(0,3)); % Outputs Xs
```

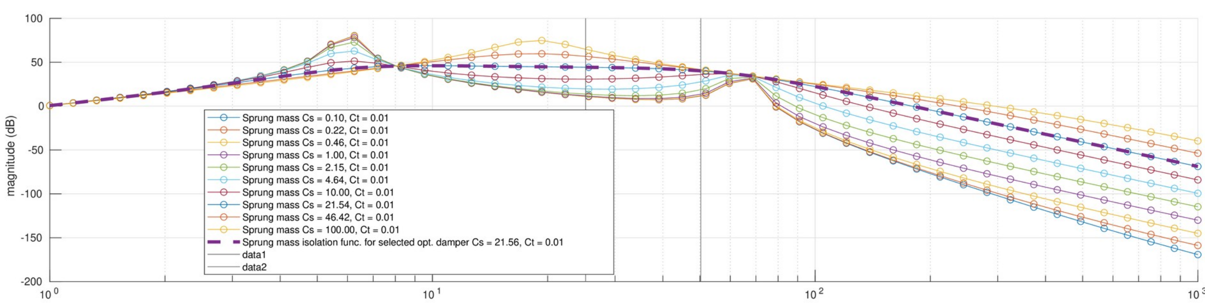
```
mgs(1:50)=mag;
[mag, phase, w]=bode(ss(A,B(:,1), C(2,:), [0]),logspace(0,3)); % Outputs Xu
mgu(1:50)=mag;
[mag, phase, w]=bode(ss(A,B(:,1), [0, 1, 0, 0], [0]),logspace(0,3)); % Outputs Xu
mg_iso(1:50)=mag;
end
```

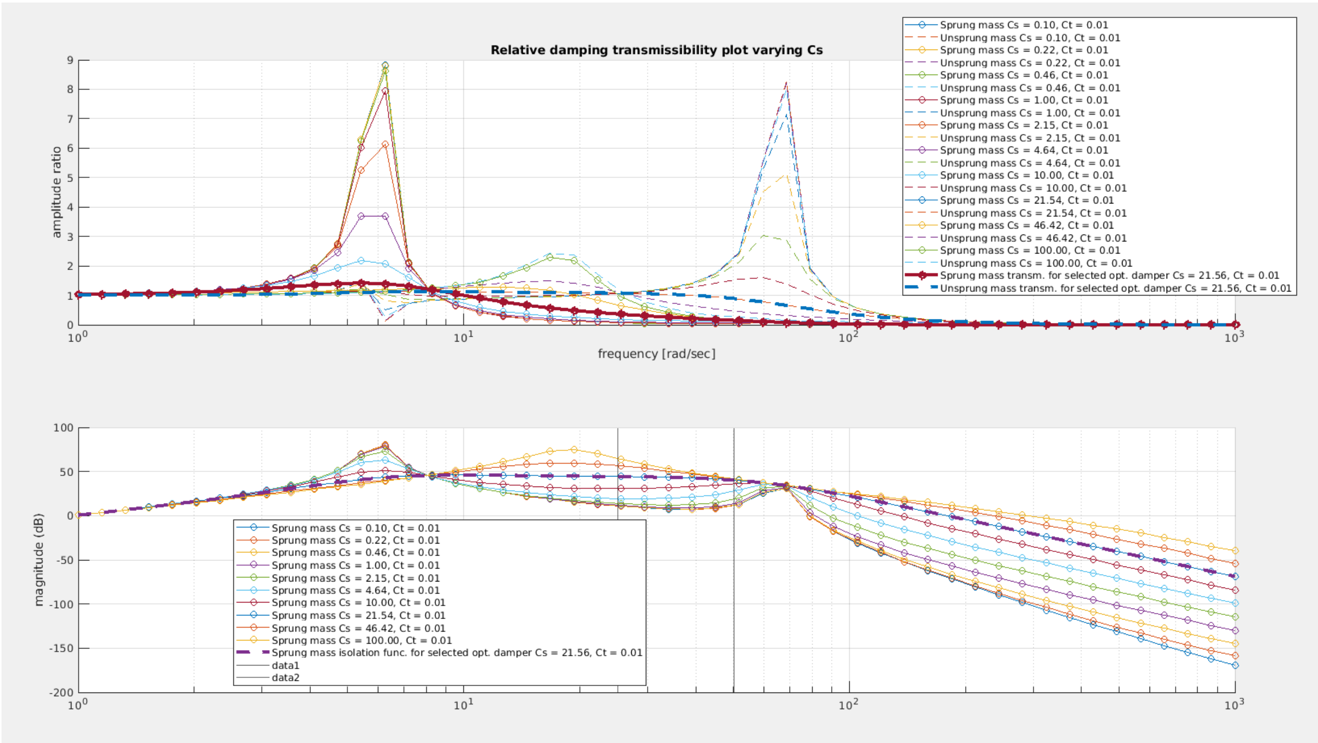
The cost function assigns a weight of 2 for the average of the transmissibilities in the 4 – 8 Hz region, then sums it with the mean of transmissibilities in the range between 4 to 8 radians per second with a weight of 10 where the transmissibilites are amplified due to the wheel hop frequency at around 2*pi rad/seconds and finally, it is summed with the mean between 4 – 8 Hz of the isolation plot.

The optimal damping coefficient value C_s is calculated to be 21.56 lbs-sec/in

The output for the code is:






Relative damping transmissibility plot varying Cs
amplitude ratio
frequency [rad/sec]
Sprung mass Cs = 0.10, Ct = 0.01
Unsprung mass Cs = 0.10, Ct = 0.01
Sprung mass Cs = 0.22, Ct = 0.01
Unsprung mass Cs = 0.22, Ct = 0.01
Sprung mass Cs = 0.46, Ct = 0.01
Unsprung mass Cs = 0.46, Ct = 0.01
Sprung mass Cs = 1.00, Ct = 0.01
Unsprung mass Cs = 1.00, Ct = 0.01
Sprung mass Cs = 2.15, Ct = 0.01
Unsprung mass Cs = 2.15, Ct = 0.01
Sprung mass Cs = 4.64, Ct = 0.01
Unsprung mass Cs = 4.64, Ct = 0.01
Sprung mass Cs = 10.00, Ct = 0.01
Unsprung mass Cs = 10.00, Ct = 0.01
Sprung mass Cs = 21.54, Ct = 0.01
Unsprung mass Cs = 21.54, Ct = 0.01
Sprung mass Cs = 46.42, Ct = 0.01
Unsprung mass Cs = 46.42, Ct = 0.01
Sprung mass Cs = 100.00, Ct = 0.01
Unsprung mass Cs = 100.00, Ct = 0.01
Sprung mass transm. for selected opt. damper Cs = 21.56, Ct = 0.01
Unsprung mass transm. for selected opt. damper Cs = 21.56, Ct = 0.01
magnitude (dB)
Sprung mass Cs = 0.10, Ct = 0.01
Sprung mass Cs = 0.22, Ct = 0.01
Sprung mass Cs = 0.46, Ct = 0.01
Sprung mass Cs = 1.00, Ct = 0.01
Sprung mass Cs = 2.15, Ct = 0.01
Sprung mass Cs = 4.64, Ct = 0.01
Sprung mass Cs = 10.00, Ct = 0.01
Sprung mass Cs = 21.54, Ct = 0.01
Sprung mass Cs = 46.42, Ct = 0.01
Sprung mass Cs = 100.00, Ct = 0.01
Sprung mass isolation func. for selected opt. damper Cs = 21.56, Ct = 0.01
data1
data2