ME 190 HW 1 John Phung 9/12/2017

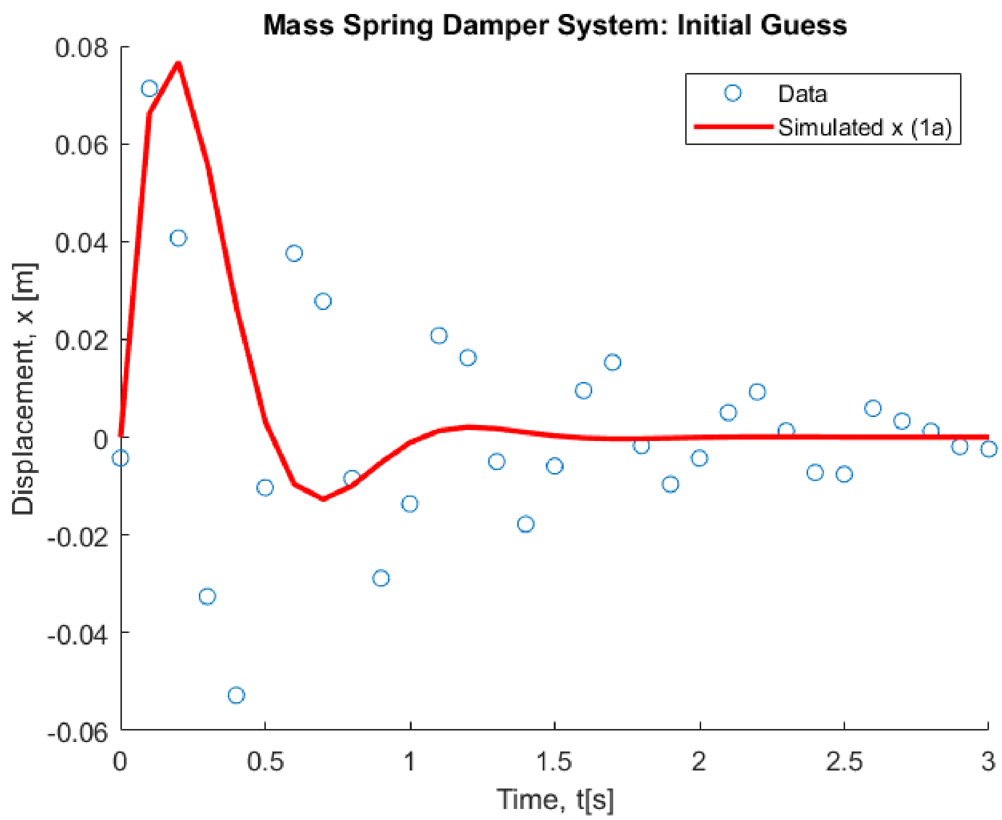
Table of Contents

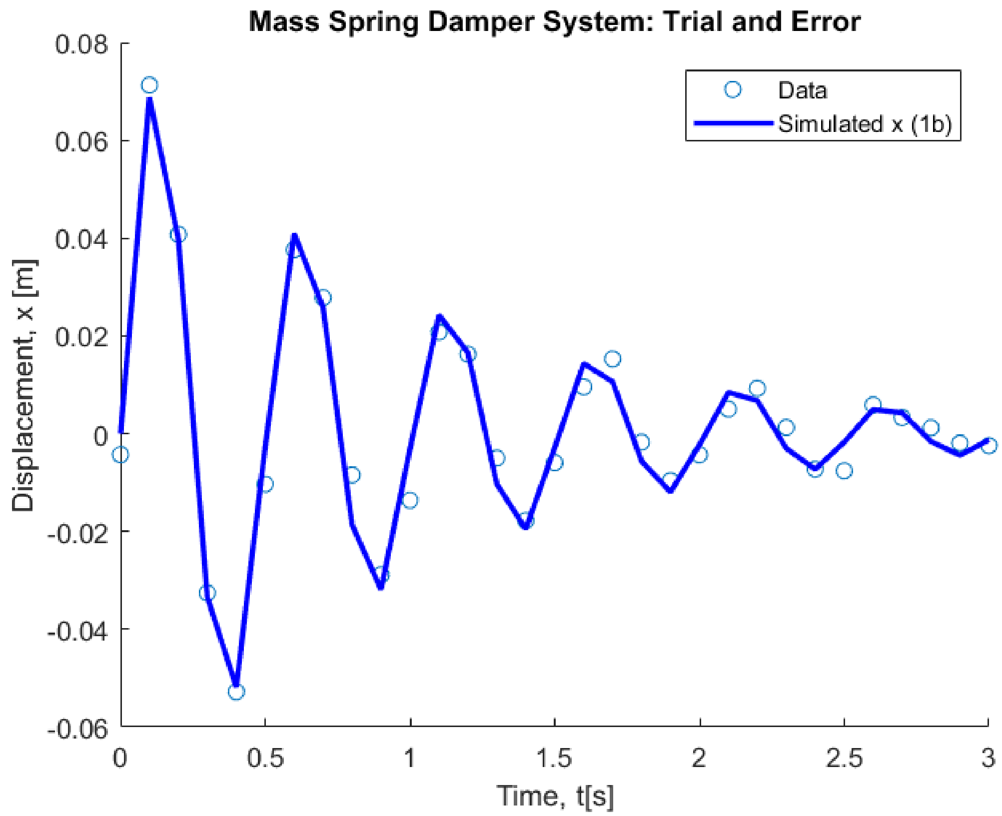
Problem 1. Identification by Trial and Error

1a) Write Script simulating and plots the response of the system

```
A = xlsread('ImpulseResponseData');

t = A (:, 1);
x = A (:, 2);

m = 1
w1 = 7 %arbitrary
z1 = 0.5 %arbitrary
x_func1 = (1./(w1.*sqrt(1-z1^2))).*exp(-z1.*w1.*t).*sin(w1.*t.*sqrt(1-
z1^2));

figure (1)
scatter (t, x)
title('Mass Spring Damper System: Initial Guess')
xlabel('Time, t[s]')
ylabel ('Displacement, x [m]')
hold on
plot (t, x_func1, 'r', 'linewidth', 2)
legend ('Data', 'Simulated x (1a)');
hold off

% 1b) Use trial and error

w = 12.5 %trial and error
z = 0.08 %trial and error
x_func = (1./(w.*sqrt(1-z^2))).*exp(-z.*w.*t).*sin(w.*t.*sqrt(1-z^2));

figure ('Name', '1b Trial & Error')
scatter (t, x)
title('Mass Spring Damper System: Trial and Error')
xlabel('Time, t[s]')
ylabel ('Displacement, x [m]')
hold on
plot (t, x_func, 'b', 'linewidth', 2)
legend ('Data', 'Simulated x (1b)');
hold off
```

```
m =

     1


w1 =

     7


z1 =

    0.5000


w =

   12.5000


z =

    0.0800
```





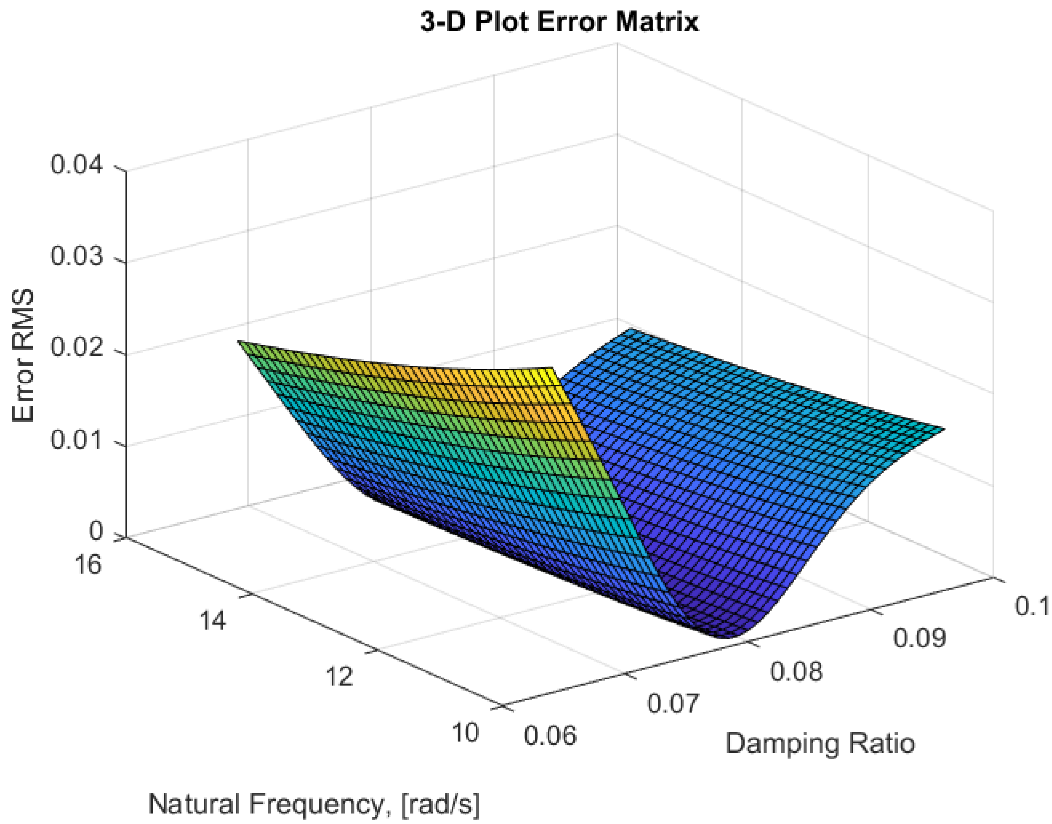
Problem 2. Error Function Development

2b) Write a short script that calculates RMS error

```
e_rms = CalcRMSError (w, z, t, x, A)

%2c) Exhaustive neighborhood search process for finding z & w

z_vec = z*[0.8:0.01:1.2];
w_vec = w*[0.8:0.01:1.2];
for i = 1:length (z_vec)
    for j = 1:length(w_vec)
        E_Matrix (i,j) = CalcRMSError (w_vec(j), z_vec(i), t, x, A);
    end
end

% 2d) Find minimum value of E_Matrix. Use "disp"

minError = min ( min (E_Matrix));
min_z = z_vec (find (minError));
min_w = w_vec (find (minError));
disp (['Optimal value of z is: ' num2str(min_z) ' [rad/s]'])
disp (['Optimal value of w is: ' num2str(min_w) ' [rad/s]'])

% 2e) 3-D Plot of error matrix using "surf"
```

```
figure ('Name', '2e 3-D Plot Error Matrix');
surf(z_vec, w_vec, E_Matrix);
title('3-D Plot Error Matrix');
xlabel('Damping Ratio');
ylabel('Natural Frequency, [rad/s]');
zlabel('Error RMS');
```

```
e_rms =

   5.3445e-04

Optimal value of z is: 0.064 [rad/s]
Optimal value of w is: 10 [rad/s]
```



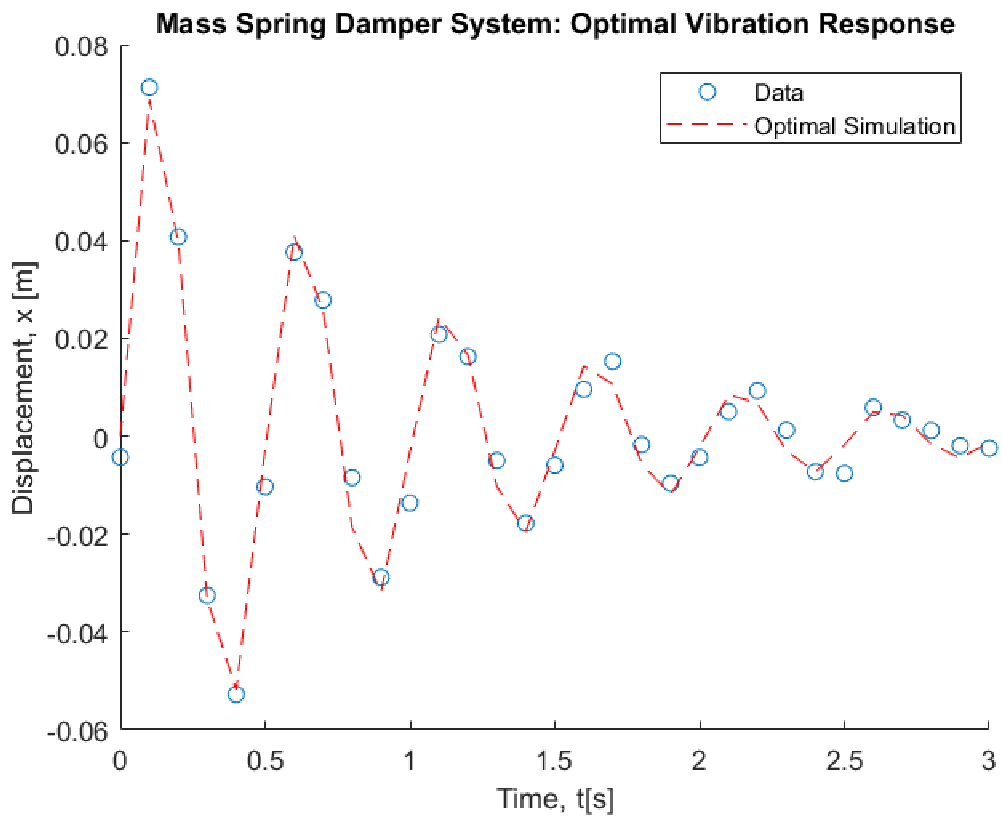
Problem 3. Automated Optimization of the Error

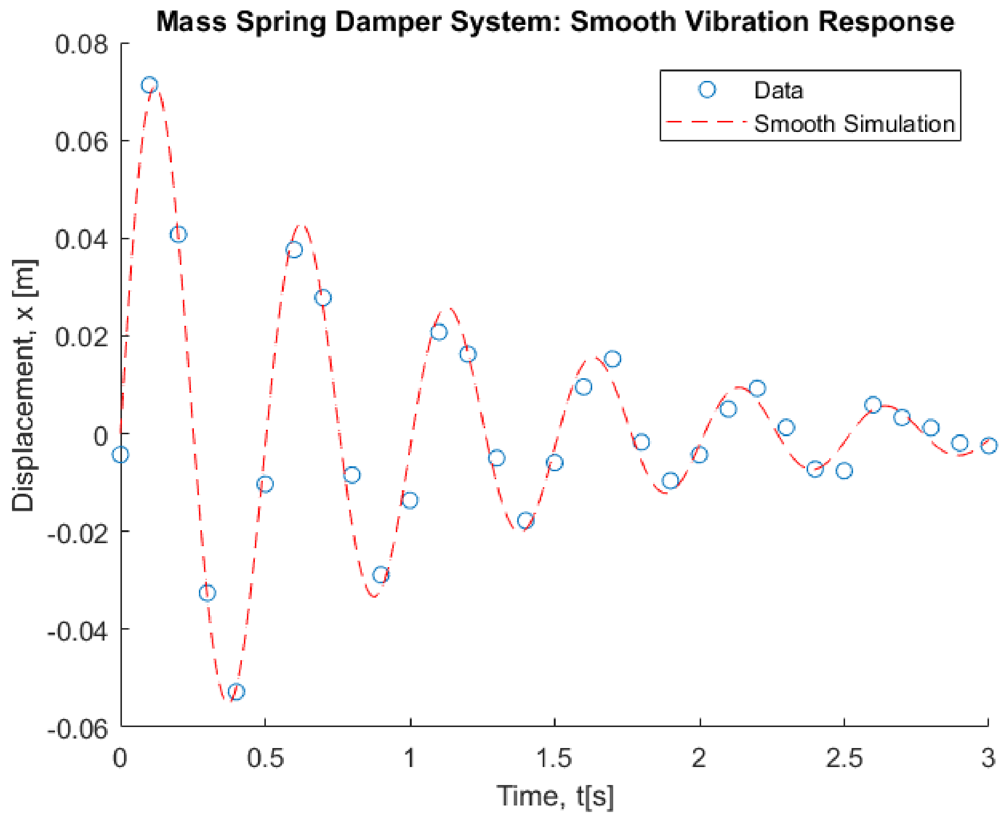
3b) Nelder-Mean simplex search algo minimize the RMS error

```
q0 = [z, w];
e_rms2 = CalcRMSError2 (q0);

[q_opt e_opt] = fminsearch('CalcRMSError2', q0)  %Question what is
 q_opt and e_opt
```

```
% 3c) Optimal values of z and w to plot simulated response. Do you
 observe any improvement compared to the repsonse obtained in the
 guessed values in 1.b?

OptSim = x_sim (q0(2), q0(1), t, A);

figure ('Name', '3c Optimal Plot')
scatter (t, x)
title('Mass Spring Damper System: Optimal Vibration Response')
xlabel('Time, t[s]')
ylabel ('Displacement, x [m]')
hold on
plot (t, OptSim, '--r')
legend ('Data', 'Optimal Simulation');
hold off

%{
Using optimal simulated values (red dashed) returned curves closeer to
 the given data
%}

% 3d Plot smoother response of similated output.

t_sim = [0:0.001:3];
SmoothSim = x_sim (q0(2), q0(1), t_sim, A);

figure ('Name', '3d Smooth Plot')
scatter (t, x)
title('Mass Spring Damper System: Smooth Vibration Response')
xlabel('Time, t[s]')
ylabel ('Displacement, x [m]')
hold on
plot (t_sim, SmoothSim, '--r')
legend ('Data', 'Smooth Simulation');
hold off

% 3e) Compute values of c & k

k = m*q_opt(2)^2
c = 2*q_opt(1)*sqrt(k*m)

%{
Yes I am satisfied with the response of the model vs the measurement
 samples. If optimizing and identifying the unknown parameters of a
 system is the second step toward
developing a controller for a mechatronic system. The fist step in
 development must be modeling and analysis.
%}


q_opt =

    0.0873   12.2776
```

```
e_opt =

   1.4645e-04

k =

  150.7389

c =

    2.1427
```





Appendix

```
type x_sim.m;

type CalcRMSError.m;

type CalcRMSError2.m;

% End of HW1


function ImpResponse = x_sim (w, z, t, A)

ImpResponse = (1./(w.*sqrt(1-z^2))).*exp(-z.*w.*t).*sin(w.*t.*sqrt(1-
z^2));

end

function e_rms = CalcRMSError (w, z, t, x, A)

Error = (x - x_sim(w, z, t, A)).^2;
e_rms = sum (Error);

end

function e_rms2 = CalcRMSError2 (q)
```

```
%Calc RMS using optimized values.

 t = evalin ('base', 't');
 x = evalin ('base', 'x');
 Error = (x - x_sim(q(2), q(1), t, x)).^2;
 e_rms2 = sum (Error);

end
```

Published with MATLAB® R2017a