Exercises 3a: MAE Unit 3

Name: Josep Famadas Alsamora

Collaboration Info: I have used some information from mathworks.com.

Exercise 1. Curves intersection.

Explanation:

(1) First of all, I plotted the three circles using the function 'ezplot' and defining x and y as 'syms'.

(2) (I solved this part basing on a code I found in mathworks) Here, in order to find the intersection, I created 3 functions containing the equations of the circles by pairs and a 4th function containing the 3 equations. Then, in the "main" script I used the functions and the function 'fsolve' to find all the intersection points. Finally, I plotted, over the circles, the intersection points as 'o' and the centers as '*'.

MATLAB Code:

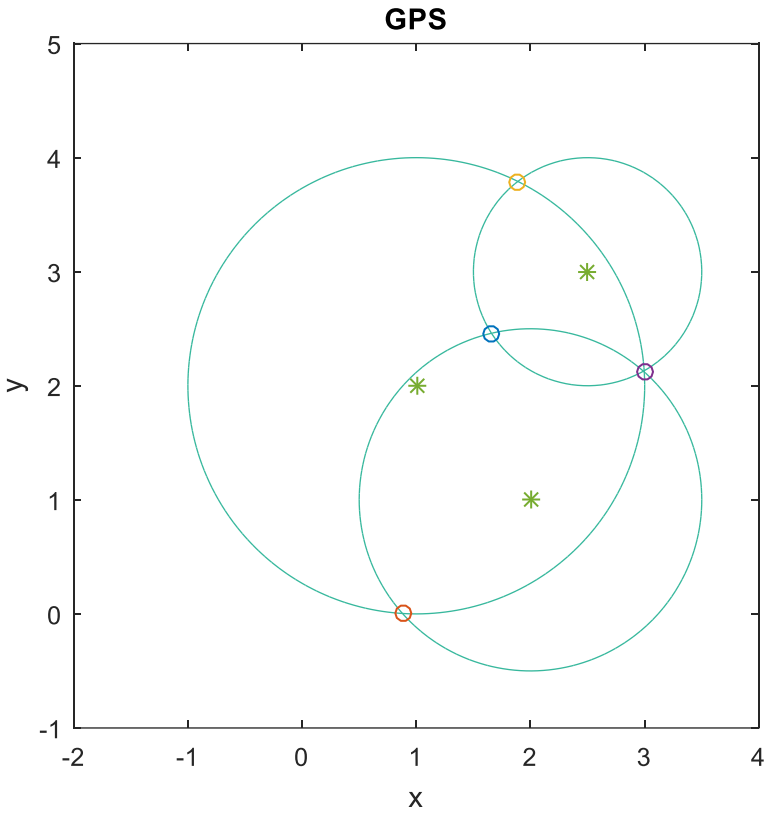
```
%% (1)

P = [2 1;2.5 3;1 2];
R = [1.5 1 2];
syms x y
f1 = '(x-2)^2+(y-1)^2=1.5^2';
f2 = '(x-2.5)^2+(y-3)^2=1^2';
f3 = '(x-1)^2+(y-2)^2=2^2';
ezplot(f1)
axis equal;
hold on;
ezplot(f2)
ezplot(f3)
title('GPS');
axis([-2 4 -1 5]);


%% (2)

inici = [0 0];
inici23=[2,3];

inter12 = @Equations12;
x12 = fsolve(inter12,inici);
plot(x12(1),x12(2),'o');

inter13 = @Equations13;
x13 = fsolve(inter13,inici);
plot(x13(1),x13(2),'o');

inter23 = @Equations23;
x23 = fsolve(inter23,inici23);
plot(x23(1),x23(2),'o');
```

```
inter123 = @Equations123;
x123 = fsolve(inter123,inici);
plot(x123(1),x123(2),'o');

x = P(1:end,1);
y = P(1:end,2);
plot(x,y,'*');
```

Functions

```
function [F] = Equations12( x )
F = zeros(1,2);
F(1) = (x(1)-2).^2+(x(2)-1).^2-1.5.^2;
F(2) = (x(1)-2.5).^2+(x(2)-3).^2-1.^2;
end

function [F] = Equations13( x )
F = zeros(1,2);
F(1) = (x(1)-2).^2+(x(2)-1).^2-1.5.^2;
F(2) = (x(1)-1).^2+(x(2)-2).^2-2.^2;
end


function [F] = Equations23( x )
F = zeros(1,2);
F(1) = (x(1)-2.5).^2+(x(2)-3).^2-1.^2;
F(2) = (x(1)-1).^2+(x(2)-2).^2-2.^2;
end


function [F] = Equations123( x )
F = zeros(1,3);
F(1) = (x(1)-2).^2+(x(2)-1).^2-1.5.^2;
F(2) = (x(1)-2.5).^2+(x(2)-3).^2-1.^2;
F(3) = (x(1)-1).^2+(x(2)-2).^2-2.^2;
end
```

Results:



Coments:

At the beginning I had a problem when solving the intersection23, Xavi Timoneda told me that I had to change the initial point.

Exercise 2. Complex variable functions.

Explanation:

(1) First of all I defined G with the function 'freqs' and the data given and I found the modulus and the argument using 'abs' and 'angle' respectively.
(2) Here I created G2 using the function 'tf' and plotted its bode diagram with the function 'bode'.
(3) Now I used the function 'Nyquist' to plot the representation between -1 and 1.

MATLAB Code:

```
%% (1)

w = [0, 1, 2, 5, inf];
zeros = 3;
poles = [1 5 6 0];

G = freqs(zeros,poles,w);

mags = abs(G);
args = angle(G);

%% (2)
w = logspace(-1,5);
G2 = tf(zeros,poles);
bode(G2,w);

%% (3)

w=logspace(-1,1);
nyquist(G2,w)
```

Results:

(1)
For `w = [0, 1, 2, 5, inf];`

The results are (respectively):

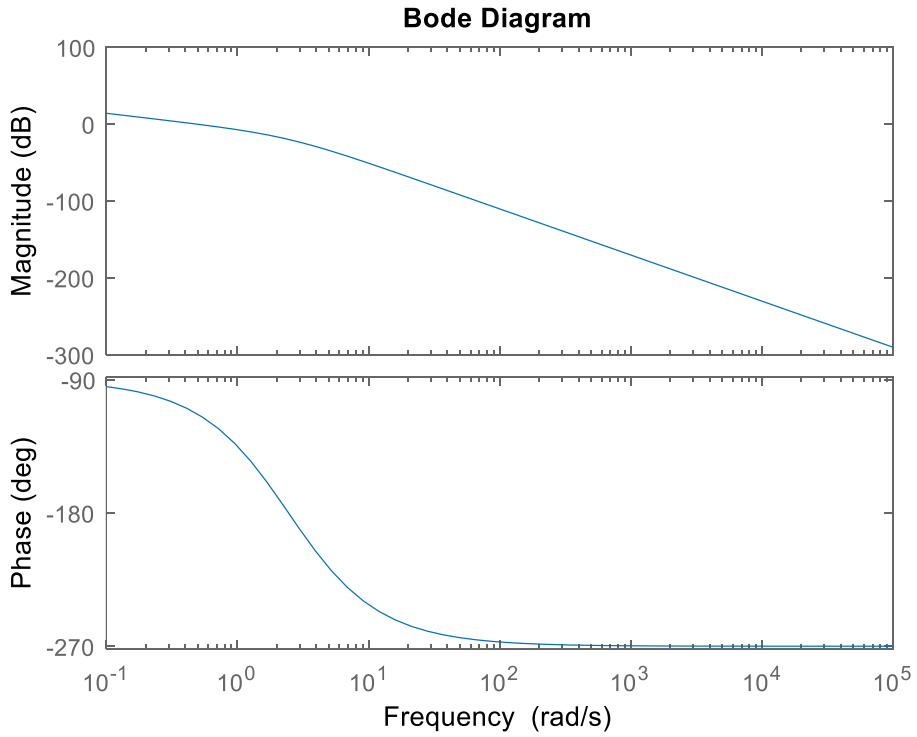
mags =

Inf 0.4243 0.1471 0.0191 0

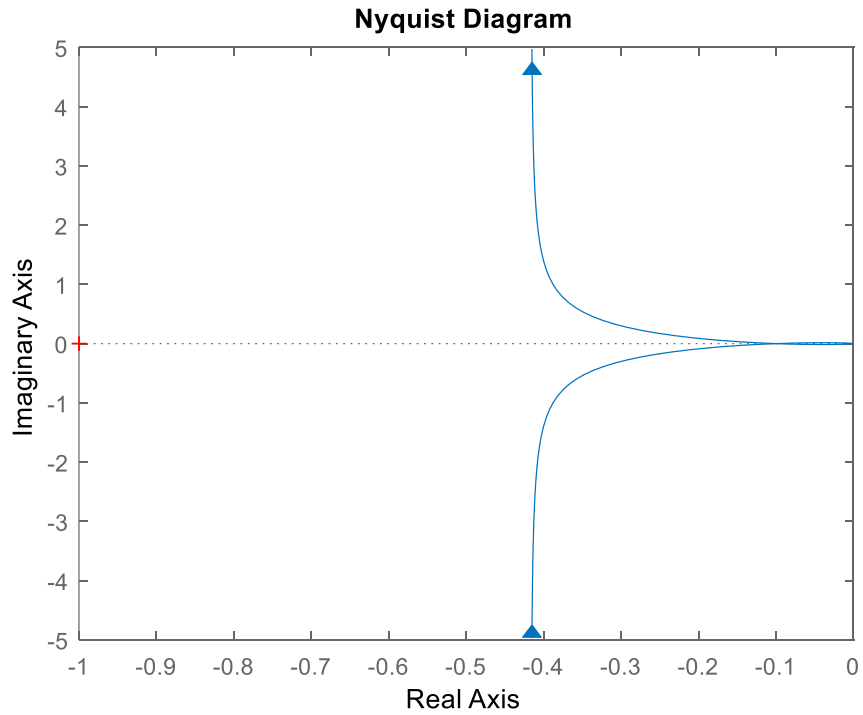
args =

0 -2.3562 -2.9442 2.4917 0

(2)



(3)



Comments:

Exercise 3. Time response.

Explanation:

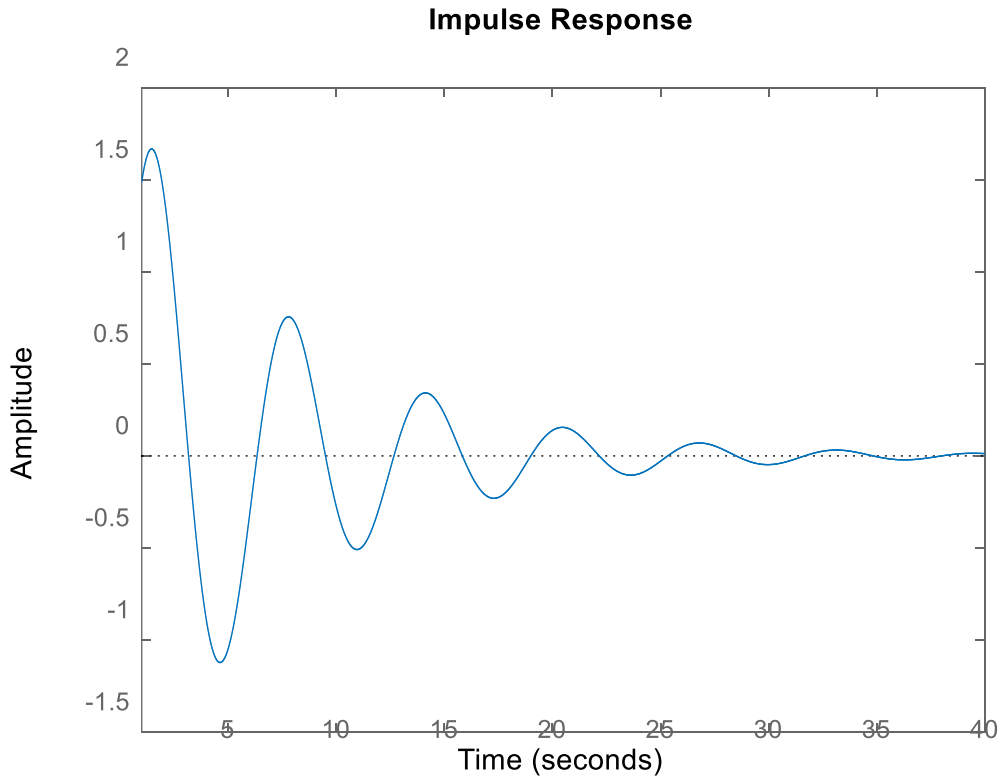
(1) First of all, I created H using the function 'tf' with the poles and zeros given and then I plotted it through the time using 'impulse'
(2) Now I did the same but using the function 'pulse'.

MATLAB Code:

```
%% (1)

zeros = 2;
poles = [1, 0.25, 1];

H = tf(zeros,poles);
figure(1)
impulse(H,(1:0.01:40))

%% (2)
figure(2)
step(H,(1:0.01:40))
```

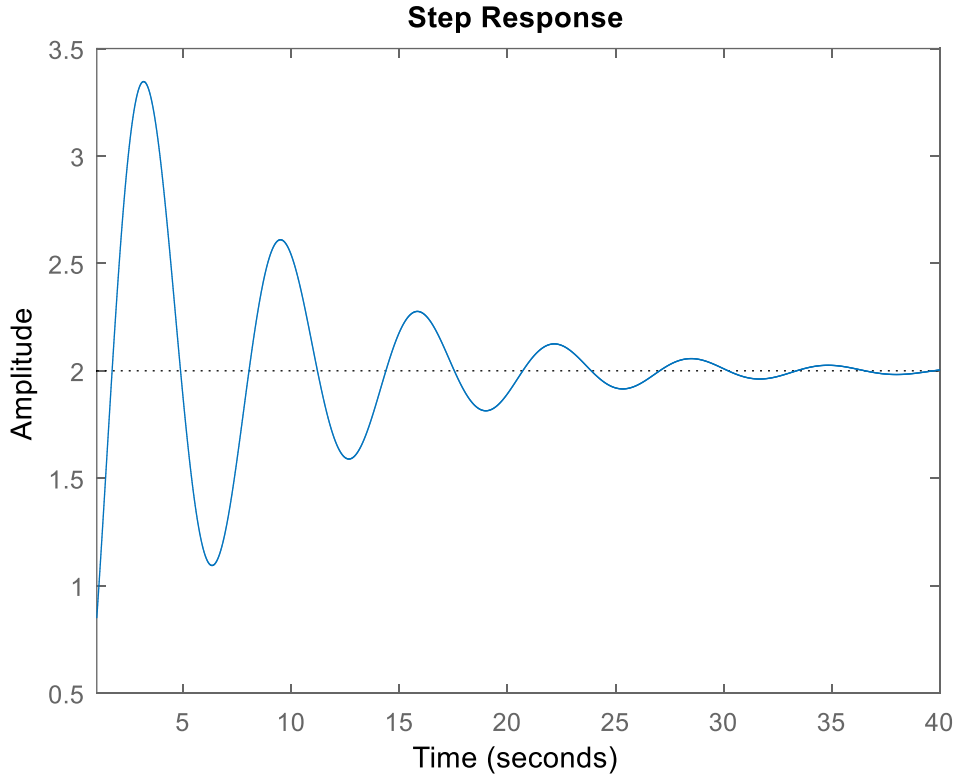
Results:

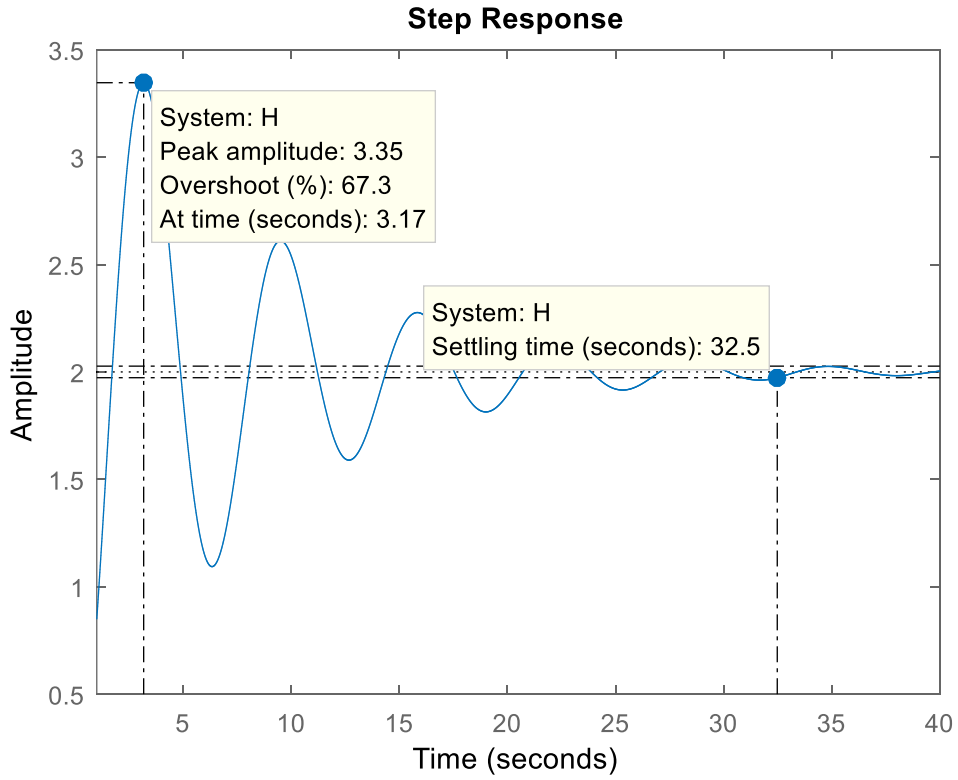
(1)



(2)

The first graphic is the Step Response and in the second one I have added the required points.





Comments:

Exercise 4. Decoding Morse signal.

Explanation:

(1) First of all, I just loaded the signal using 'audioread('morse.wav')'.
(2) Here I plotted and played the signal.
(3) In this point, I used the spectral estimation method explained in the theory pdf to estimate the spectral density but using more points.
(4) I created a low-pass filter with the specified bandwidth using the function 'fir1'. Then I filtered the signal multiplied by a sinus (with the specified frequency) with the function 'filter'. Finally I ploted the result with the function 'stairs'
(5) Here I got the derivative function using 'diff' and plotted it.
(6) I could not find a way to solve it.

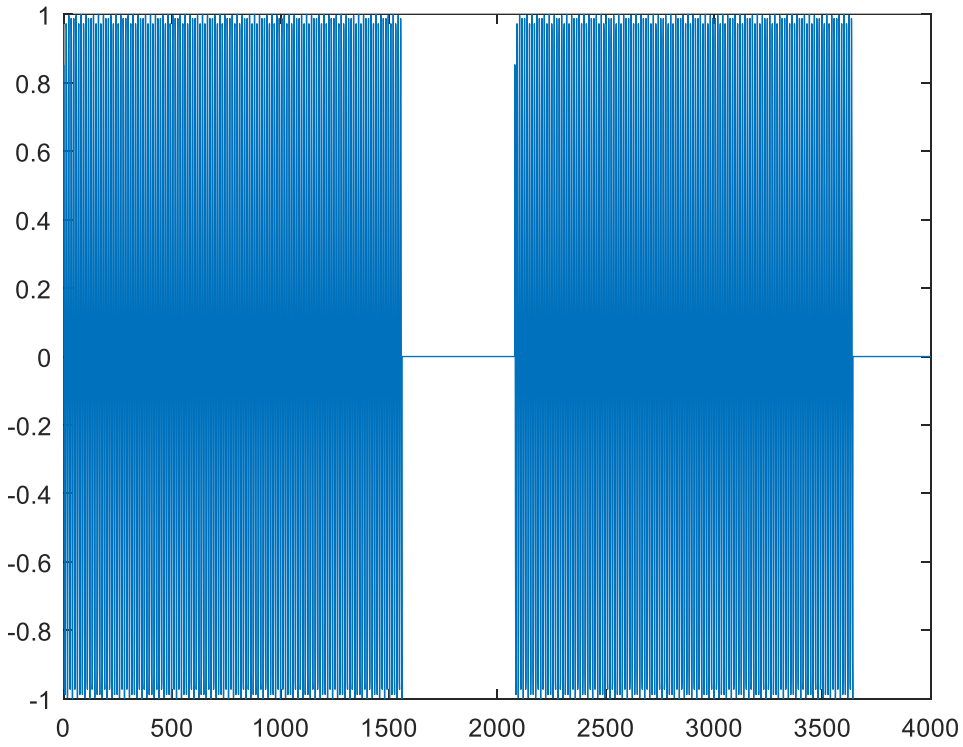
MATLAB Code:

```
%% (1)

[signal,Fs] = audioread('morse.wav');
dot_duration = 0.065;
tone_frequency = 700;
sampling_frequency = 8000;

%% (2)

sound(signal,Fs);

plot(signal(1:4000));

%% (3)
N = 4000;
x = signal(1:4000);

M=4*512;
K=N/M;
w=triang(M);
Sxx=zeros(1,M);
for i=1:K
    xi=x( (i-1)*M+1 : i*M )'.*w';
    Xi=fft(xi);
    Sxxi=(abs(Xi).^2)/M;
    Sxx=Sxx+Sxxi;
end
Sxx=Sxx/K;

f = linspace(0,Fs/2,M/2);
plot(f,10*log10(Sxx(1:M/2)));

%% (4)

t = linspace(0,(length(signal)-1)/Fs,length(signal));
sinus = sin(2*pi*700*t);
```

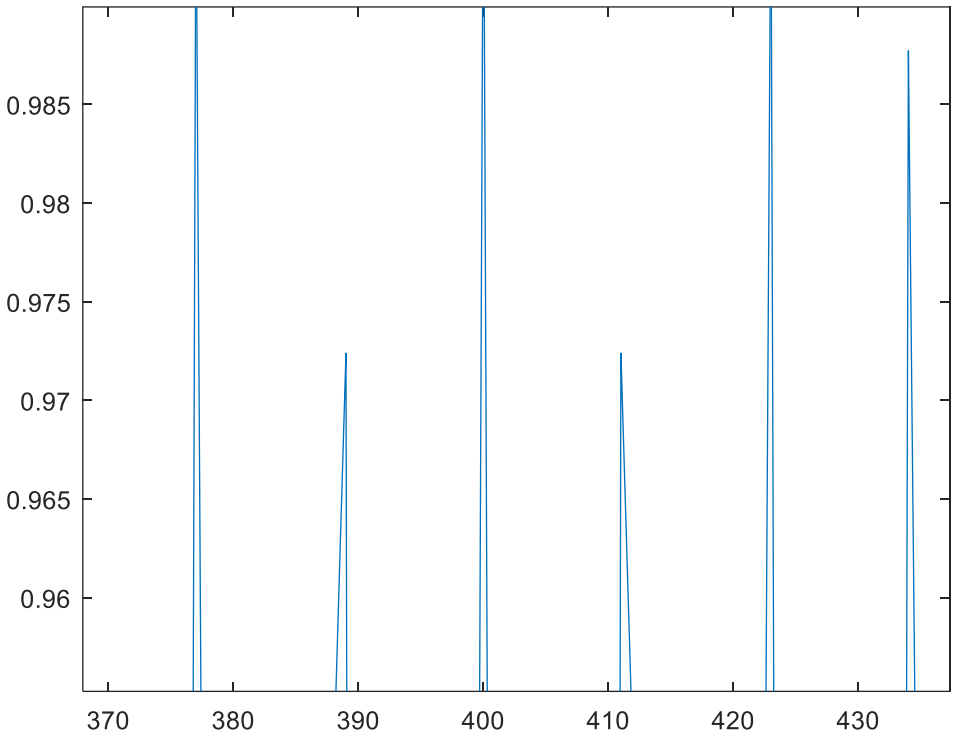
```
fx = sinus'.*signal;
dig_BW=5/(dot_duration*Fs);
fil = fir1(64,dig_BW);

fx_BB = filter(fil,1,fx);
stairs(fx_BB);


%% (5)

deriv = diff(fx_BB);
plot(deriv)
```

Results:

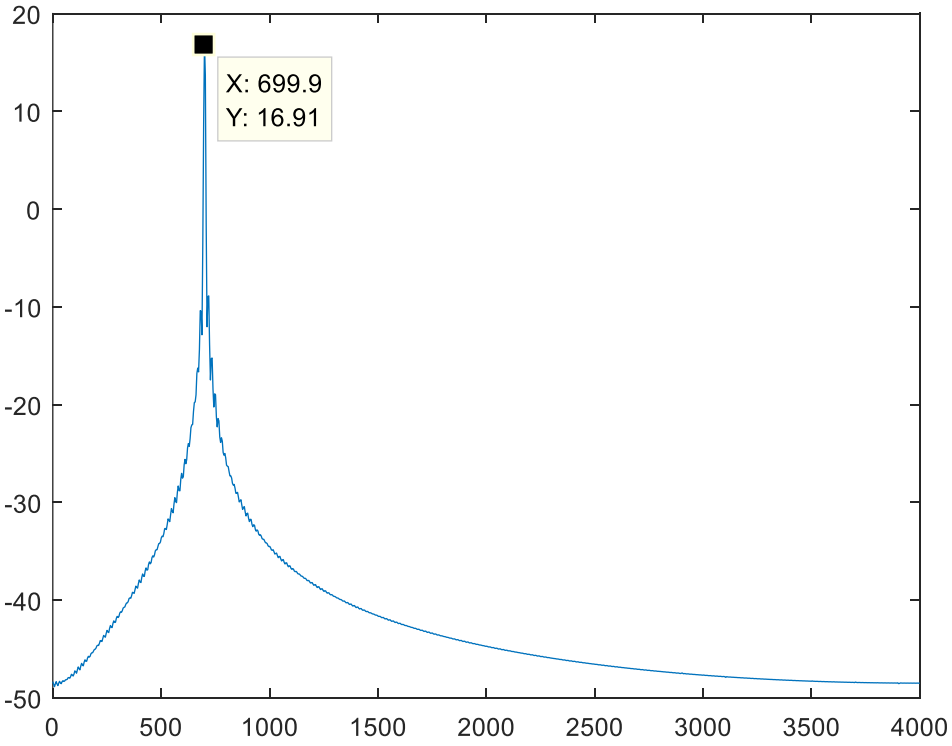
(2)



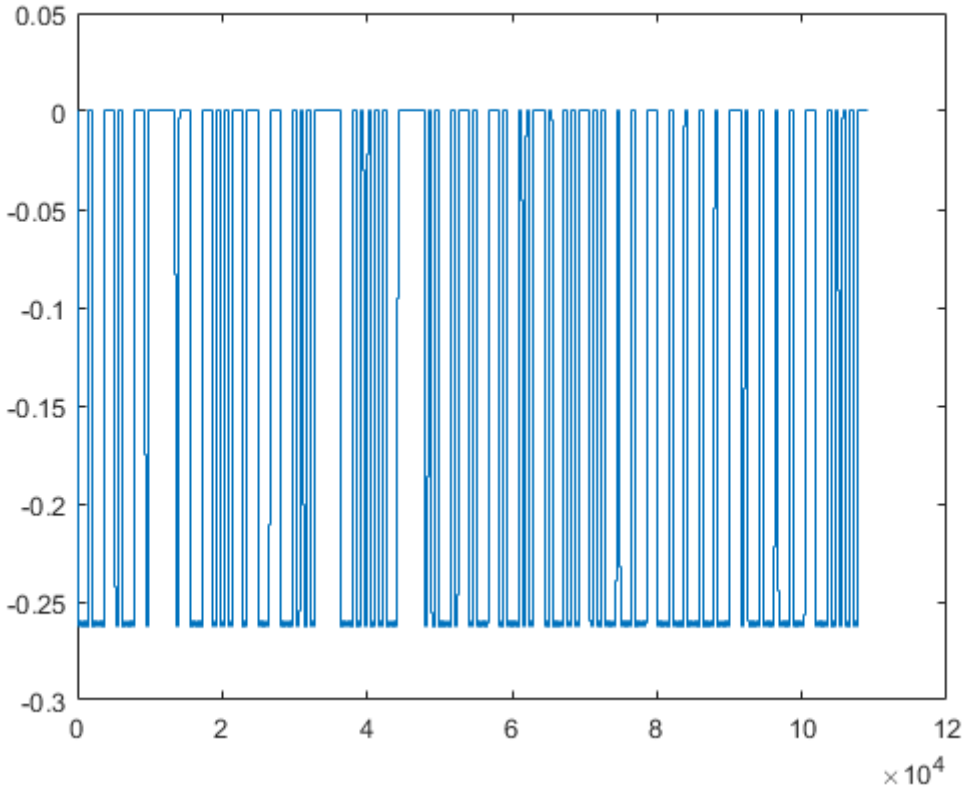
Zoomed:



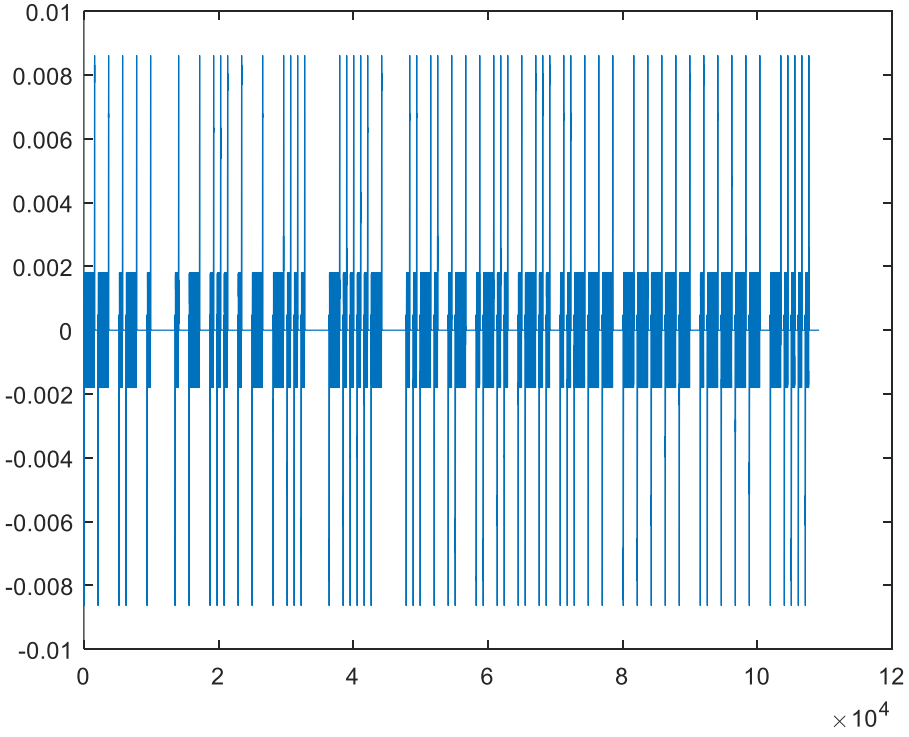
(3) We can see that the pick is approximately at X = 700 Hz.



(4)


0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
0
2
4
6
8
10
12
×10^4

(5)


0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-0.006
-0.008
-0.01
0
2
4
6
8
10
12
×10^4

Coments:

Exercise 5. Lineal Regression, Polynomial Fitting.

Explanation:

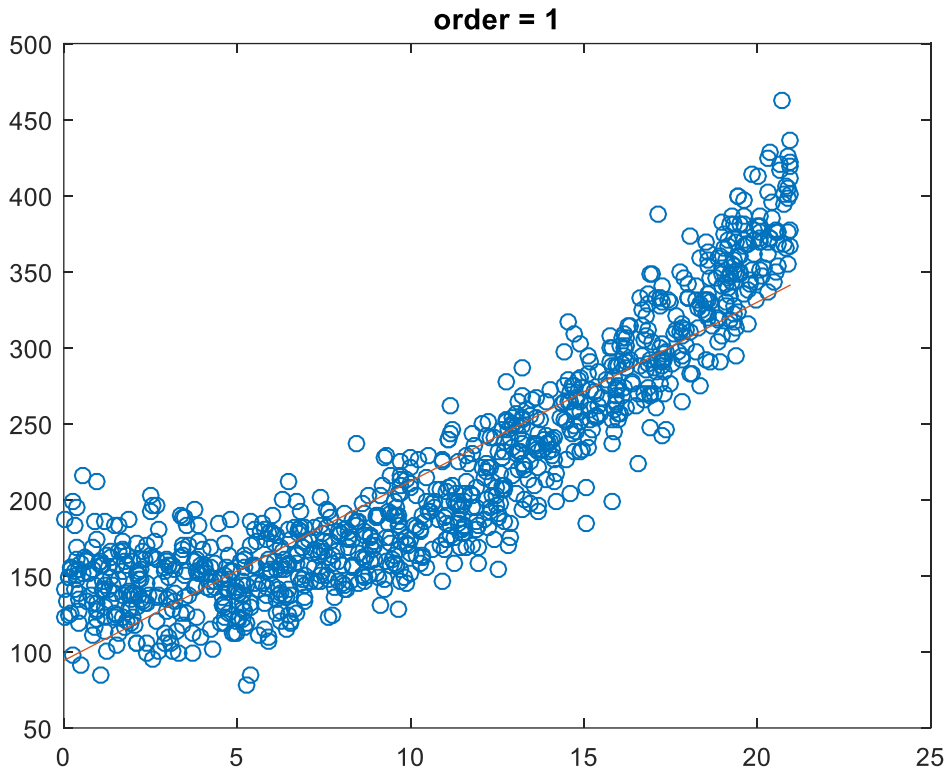
(1) First of all, I loaded the data using 'load('data5.mat')', I plotted it and over this plot I put the regression line that I found using 'polyfit' and 'polyval'. Finally, I found the error as the norm of the rest between the line and the points.

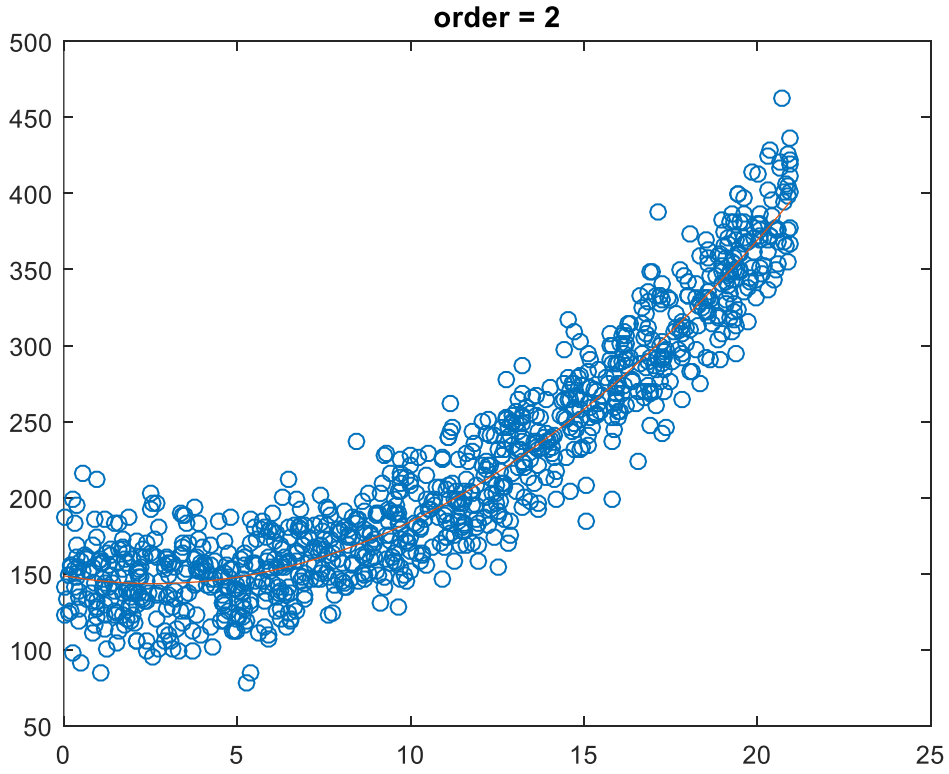
(2) Now I did the same but changing one parameter of 'polyfit' from 1 to 2 in order to get a second degree polynomic and try to see if that decreases the error. I also tried with a 3rd order polynomic.

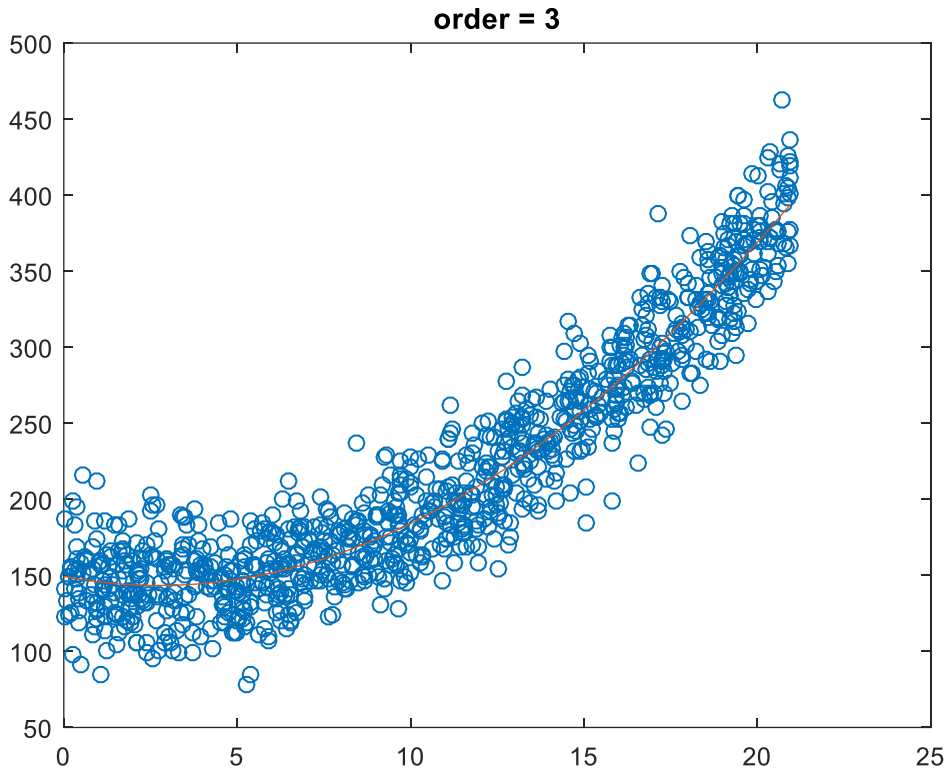
MATLAB Code:

```
%% (1)

load('data5.mat');
figure(1)
plot(U,V,'o');
p = polyfit(U,V,1);

V1 = polyval(p,U);
hold on;
plot(U,V1);
title('order = 1');
hold off;
err1 = norm(V1-V)

%% (2)

figure(2)
plot(U,V,'o');
p = polyfit(U,V,2);

V2 = polyval(p,U);
hold on;
plot(U,V2);
title('order = 2');
hold off;
err2 = norm(V2-V)

figure(3)
plot(U,V,'o');
p = polyfit(U,V,3);

V3 = polyval(p,U);
hold on;
plot(U,V3);
title('order = 3');
hold off;
err3 = norm(V3-V)
```

Results:


order = 1
500
450
400
350
300
250
200
150
100
50
0
5
10
15
20
25


order = 2
500
450
400
350
300
250
200
150
100
50
0
5
10
15
20
25



err1 =

1.1028e+03

err2 =

772.9220

err3 =

772.8619

Coments:

We can observe that when we increase from 1 to 2 the order, the decrase in error is significant. But from 2 to 3 the error only decreases 0.061 (which is less than a 0.01%), so it is not worth at all to use the 3rd order polynomic.