Emir Arda Bayer
22201832

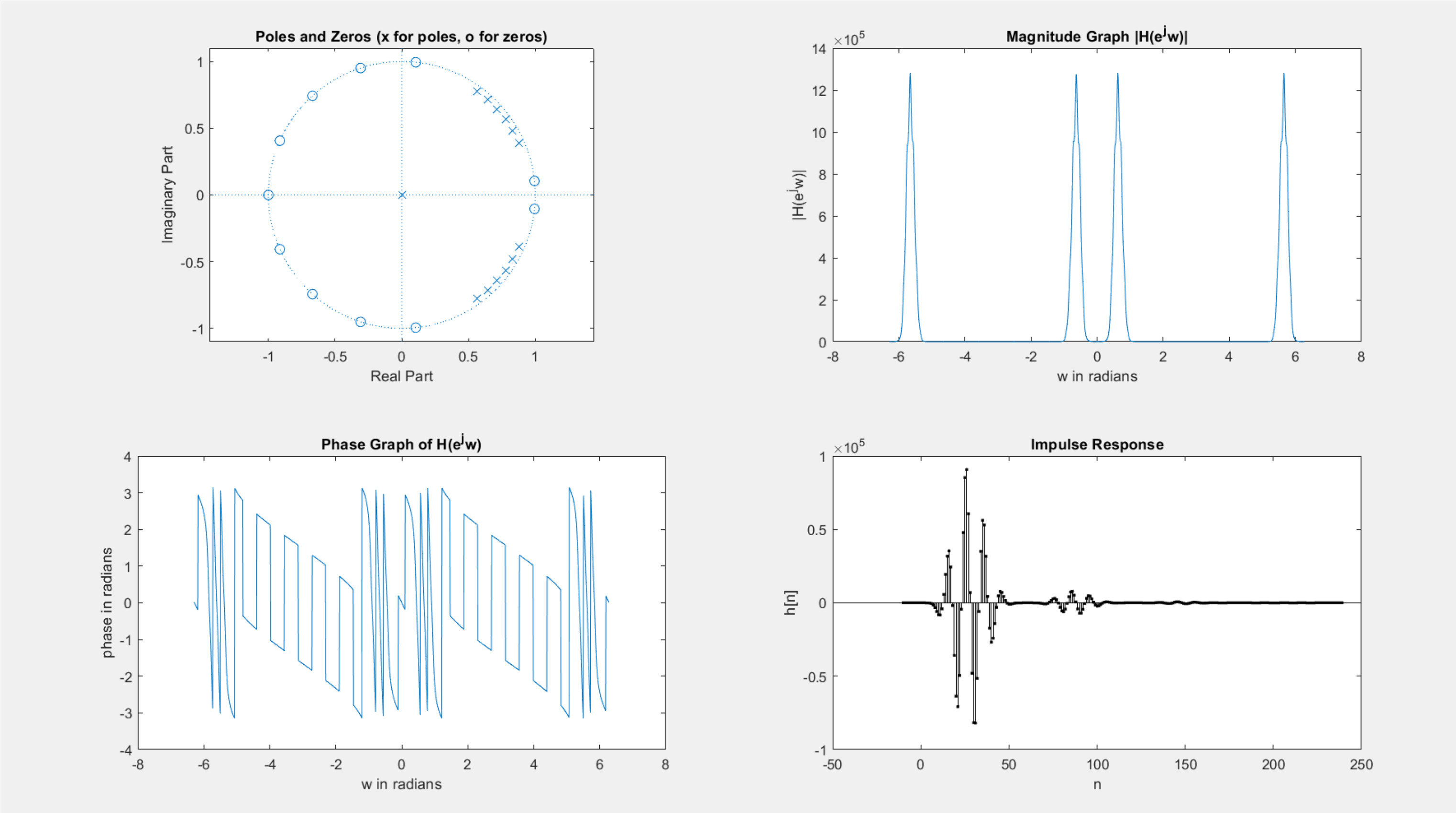

Figure 1: Graphs for Q1

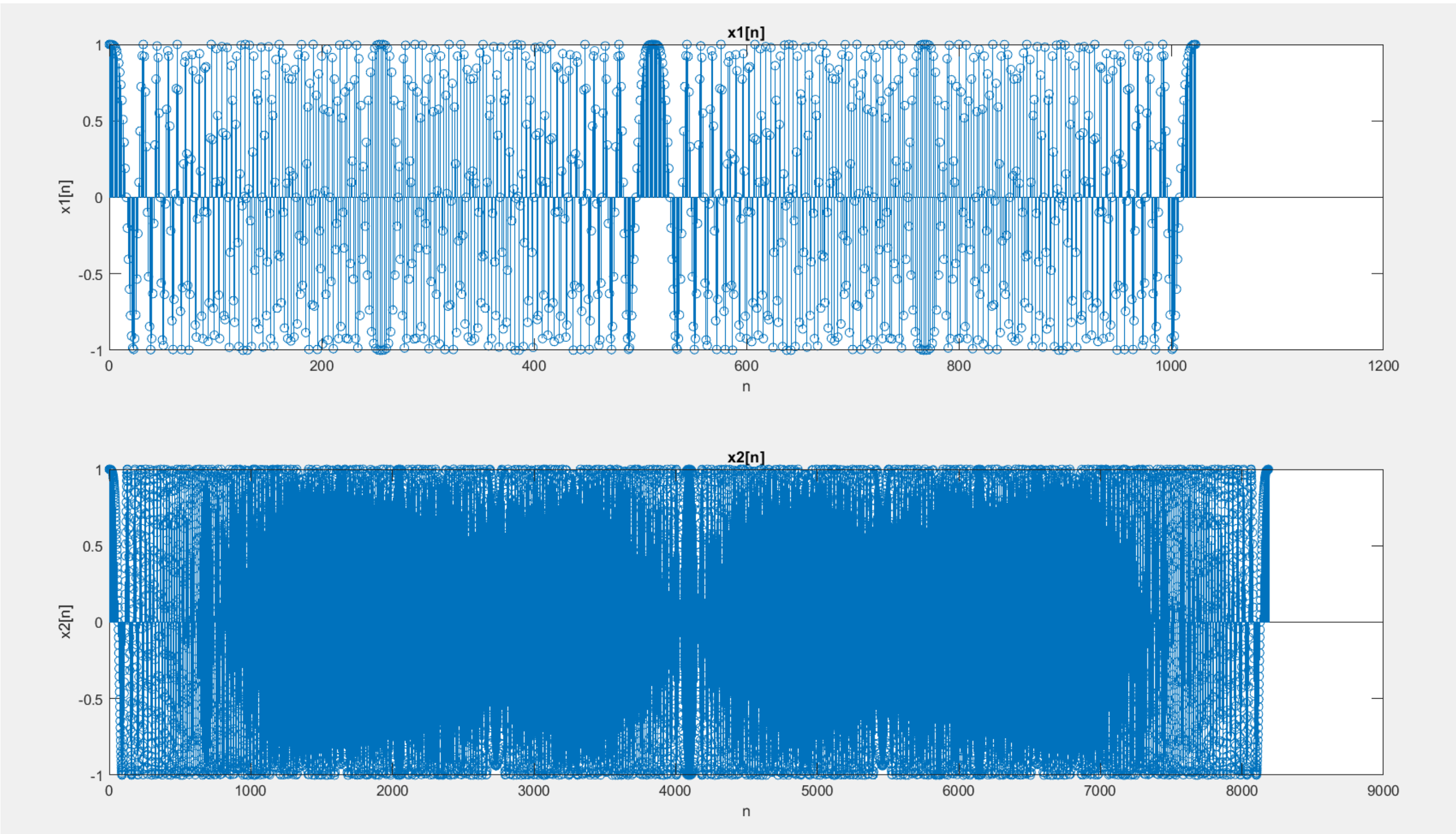

Figure 2: Graphs for Q2

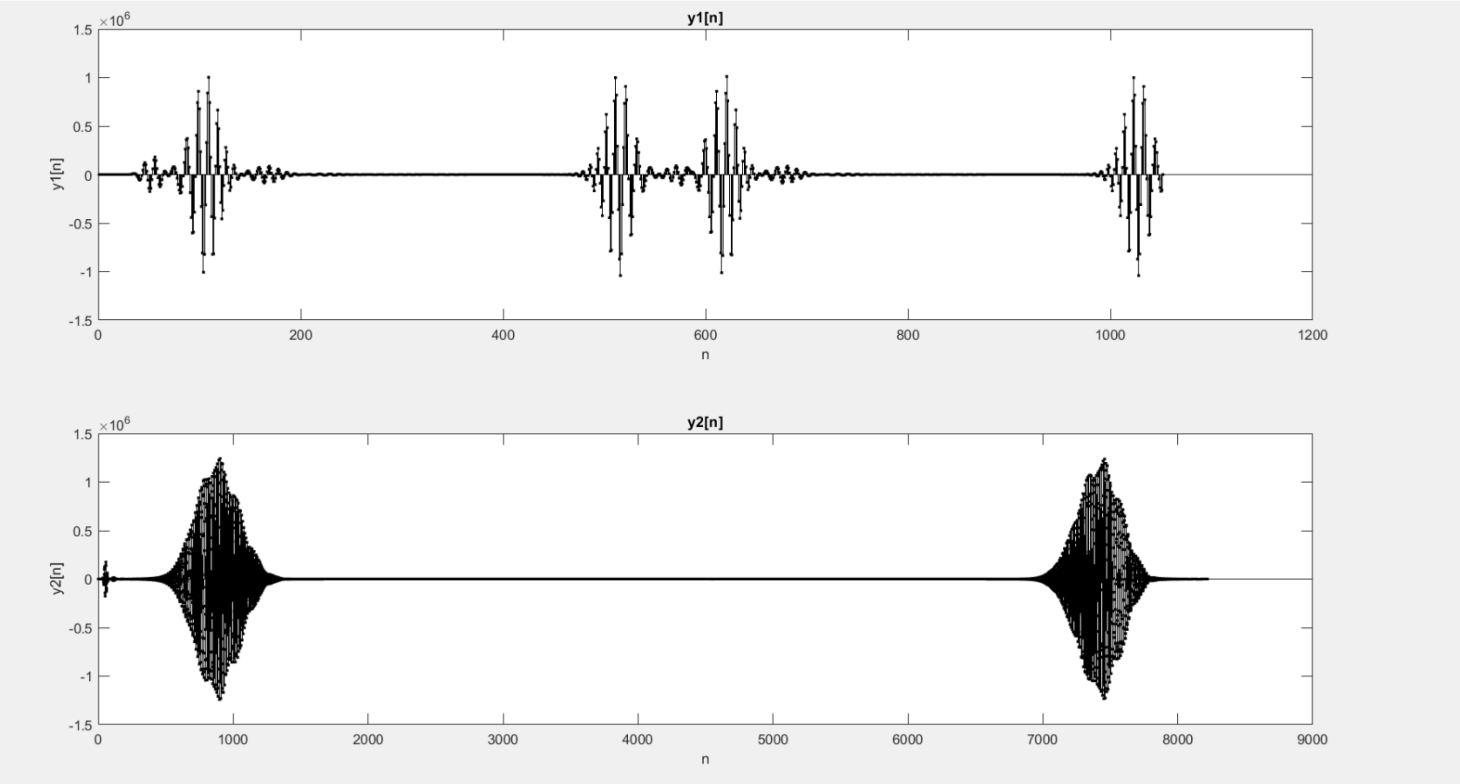

Figure 3: y1[n] and y2[n]

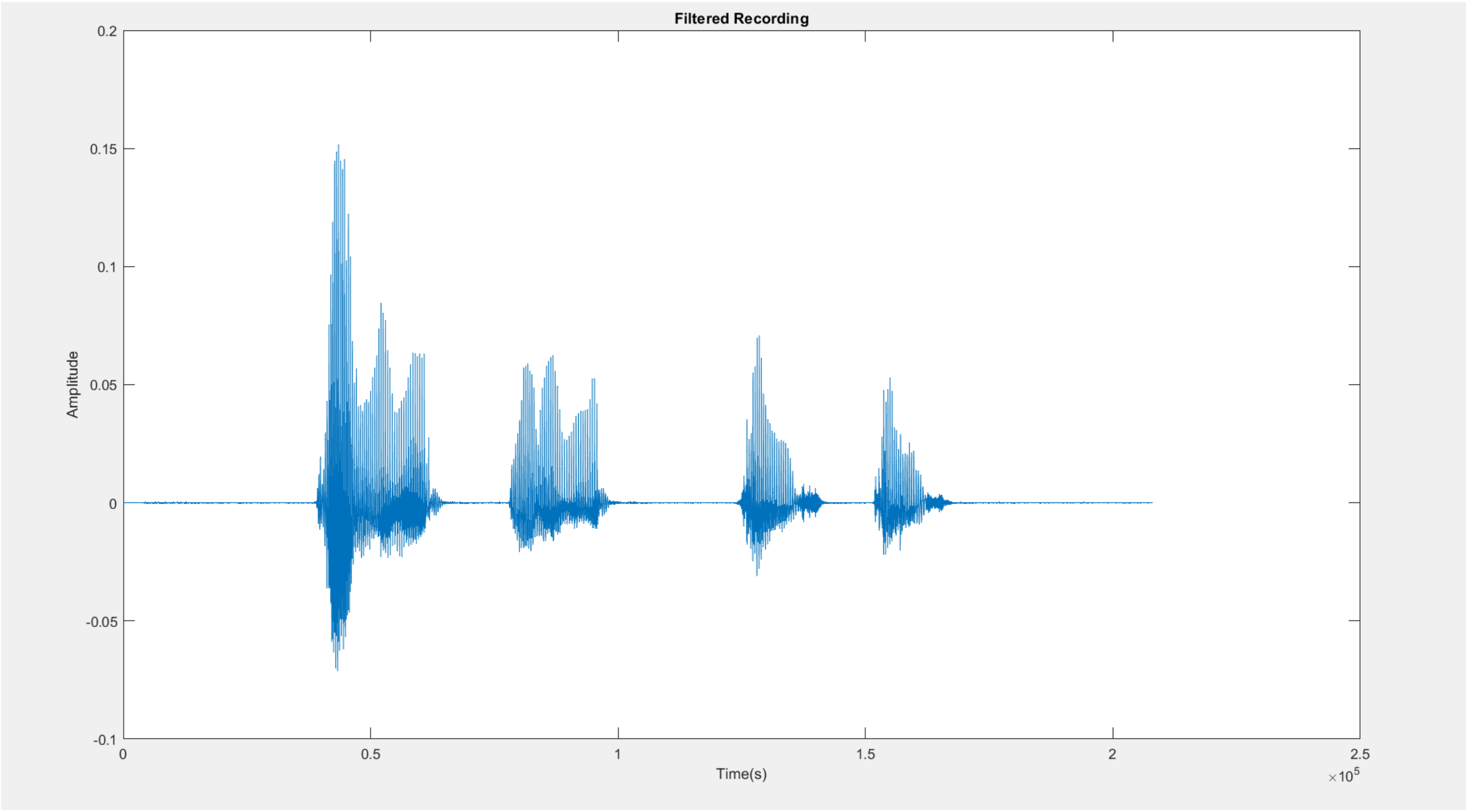

Figure 4: Filtered Recording

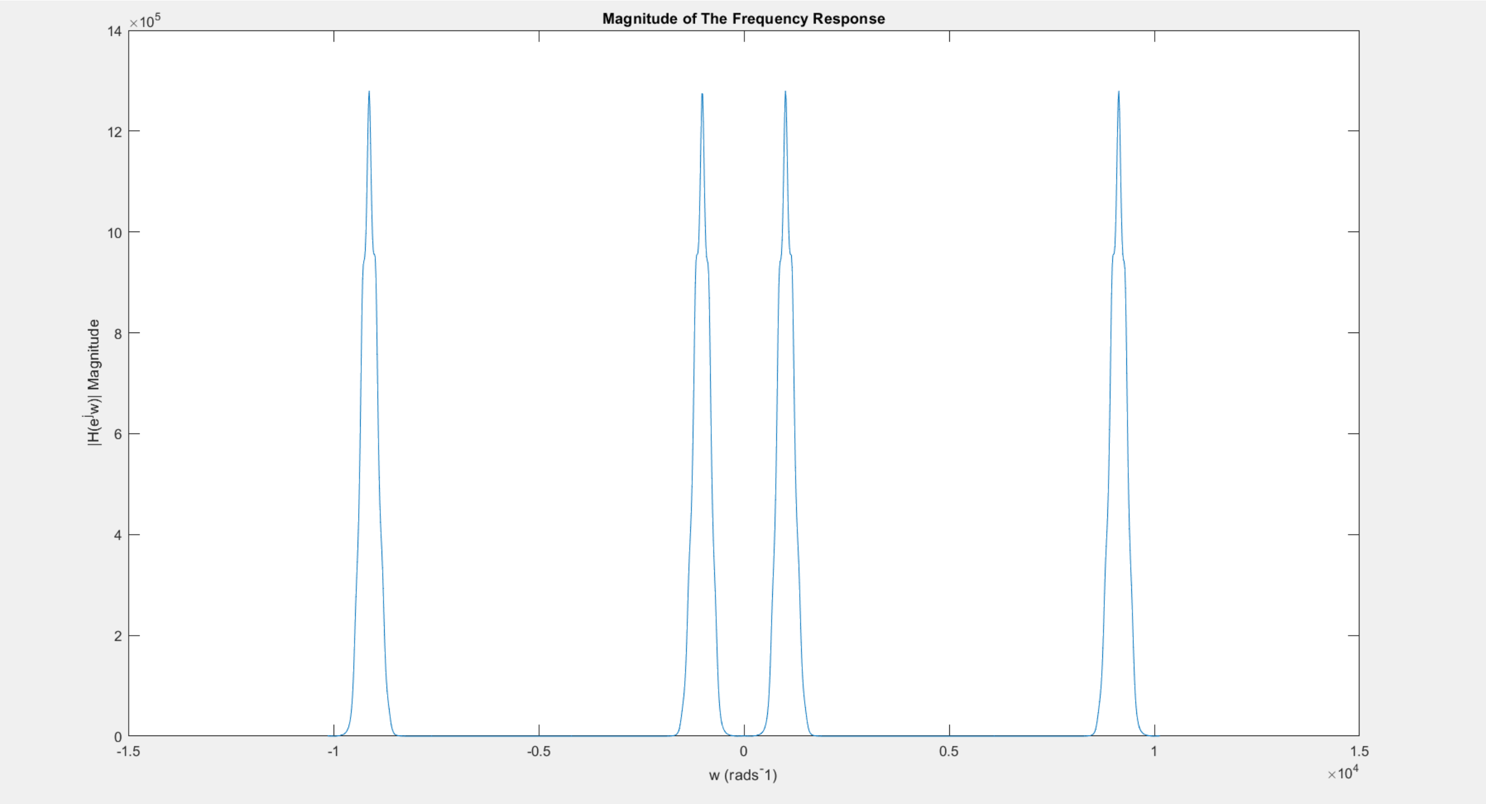

Figure 5: Magnitude of Frequency Response for $y_r(t)$

Written Explanations for Solutions

In this lab, the students designed IIR filters regarding certain specifications as below.

The filter specifications are as follows:

* Stable (Naturally, this is needed for all practical filters.)

* Bandpass

* Real valued $h[n]$.

* Order of the filter is $5 + N_1$.

* Cutoff frequencies are $\min\{\frac{\pi}{M_1}, \frac{\pi}{M_2}\}$ and $\max\{\frac{\pi}{M_1}, \frac{\pi}{M_2}\}$

* A stopband and a passband, which are as flat as possible, are desirable.

* Causal; $h[n]$.

Figure 6: Specifications on the Manual

I found my values to be N1=8, M1=10, N2=1, M2=3, N1+5=13; and found my cutoff frequencies as π/10 and π/3. Due to the conjugate symmetry of the frequency response due to z-transform, my passbands are (π/10, π/3) and (-π/10, -π/3). The order further implies that there are 13 poles of the z-transform, and 11 is selected to be the amount of zeros to suppress frequencies in the stopband. The poles are supposed to be in the unit circle, while the zeros are uniformly distributed among the stopbands. Later, MATLAB is used to obtain the impulse response of the filter, where the coefficients are found from using the nominator from the z transform. We can define H(z) = B(z) / A(z) where B(z) has the zeros as the roots and A(z) has the poles. As in the Appendix these zeros and poles are used in obtaining the impulse response with infinite duration. Recursion is used to obtain impulse response for a finite range with for loops as in the Appendix with 'fresz' and 'fresp' being the lists carrying the elements of recursion, revealing the impulse response below.

Impulse Response Array

```
  1.0e+04 *

 Columns 1 through 11

      0         0         0         0         0         0         0         0         0         0         0

 Columns 12 through 22

      0         0   -0.0002   -0.0026   -0.0144   -0.0542   -0.1519   -0.3343   -0.5885   -0.8187   -0.8334

 Columns 23 through 33

  -0.4077    0.5716    1.9380    3.1733    3.5413    2.4385   -0.1851   -3.5828   -6.3655   -7.0760   -4.9466

 Columns 34 through 44

  -0.4387    4.7855    8.5294    9.0805    6.0677    0.6942   -4.7971   -8.1649   -8.1871   -5.1481   -0.5885

 Columns 45 through 55

   3.5049    5.6208    5.3045    3.1706    0.4342   -1.7314   -2.6773   -2.4138   -1.4214   -0.3095    0.4772

 Columns 56 through 66

   0.7837    0.7010    0.4311    0.1576   -0.0214   -0.0931   -0.0926   -0.0631   -0.0332   -0.0139   -0.0045

 Columns 67 through 77

  -0.0011   -0.0002   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000   -0.0002   -0.0012   -0.0047

 Columns 78 through 88

  -0.0131   -0.0289   -0.0508   -0.0707   -0.0720   -0.0352    0.0494    0.1674    0.2740    0.3058    0.2106

 Columns 89 through 99
```

-0.0160 -0.3094 -0.5497 -0.6110 -0.4272 -0.0379 0.4132 0.7365 0.7841 0.5240 0.0599

Columns 100 through 110

-0.4142 -0.7051 -0.7070 -0.4446 -0.0508 0.3027 0.4854 0.4581 0.2738 0.0375 -0.1495

Columns 111 through 121

-0.2312 -0.2084 -0.1227 -0.0267 0.0412 0.0677 0.0605 0.0372 0.0136 -0.0018 -0.0080

Columns 122 through 132

-0.0080 -0.0054 -0.0029 -0.0012 -0.0004 -0.0001 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000

Columns 133 through 143

-0.0000 -0.0000 -0.0000 -0.0001 -0.0004 -0.0011 -0.0025 -0.0044 -0.0061 -0.0062 -0.0030

Columns 144 through 154

0.0043 0.0145 0.0237 0.0264 0.0182 -0.0014 -0.0267 -0.0475 -0.0528 -0.0369 -0.0033

Columns 155 through 165

0.0357 0.0636 0.0677 0.0452 0.0052 -0.0358 -0.0609 -0.0610 -0.0384 -0.0044 0.0261

Columns 166 through 176

0.0419 0.0396 0.0236 0.0032 -0.0129 -0.0200 -0.0180 -0.0106 -0.0023 0.0036 0.0058

Columns 177 through 187

0.0052 0.0032 0.0012 -0.0002 -0.0007 -0.0007 -0.0005 -0.0002 -0.0001 -0.0000 -0.0000

Columns 188 through 198

-0.0000 -0.0000 0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0001

Columns 199 through 209

-0.0002 -0.0004 -0.0005 -0.0005 -0.0003 0.0004 0.0012 0.0020 0.0023 0.0016 -0.0001

Columns 210 through 220

-0.0023 -0.0041 -0.0046 -0.0032 -0.0003 0.0031 0.0055 0.0058 0.0039 0.0004 -0.0031

Columns 221 through 231

-0.0053 -0.0053 -0.0033 -0.0004 0.0023 0.0036 0.0034 0.0020 0.0003 -0.0011 -0.0017

Columns 232 through 242

-0.0016 -0.0009 -0.0002 0.0003 0.0005 0.0005 0.0003 0.0001 -0.0000 -0.0001 -0.0001

Columns 243 through 250

-0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000 -0.0000 0

The impulse response h[n] is purely real as in Figure 1. This case happens since the zeros and poles are distributed symmetrically along the y-axis, as can be seen in Figure 1. Furthermore, the magnitude and the phase of the frequency response are in the same figure, which are found by using the H(z) found before and replacing z with e^jw. For Q2, the chirp signal with sampling frequency $\sqrt{\frac{\pi}{512}}$ is plotted as in Figure 2, being cos($\frac{n^2\pi}{512}$) with n between 0 and 1023. While this signal was portrayed as x1[n], other signal x2[n] = cos($\frac{n^2\pi}{8192}$), is as in Figure 2, where x2[n] is cos(α$(n * Ts)^2$) for n between 0 and 8192 and Ts=1000 rad^-2.

For Q3, the x1[n] is used in recursion with the following equation, and the final equation can be seen in handwritten solution in Figure 8 and also in the Appendix.

$$y[n] = -\sum_{k=1}^{K} a_k y[n-k] + \sum_{l=0}^{L} b_l x[n-l]$$

Figure 7: Recursion Equation Provided in the Manual

The output with the inputted sampled chirp signal is as in Figure 3. When the sampling rate increases, the output changes as in Figure 3, top to bottom. The output is the response to frequency placed into the filter. For this reason, cos(αt^2), with frequency 2αt demonstrates which time point the signal is evaluated, as the instantaneous frequency would be the itself with α being 0.5. The chirp signal thereby helps in representing the frequency response when applied as the input. Here, y1[n] is the frequency response of the filter. Furthermore, it can be stated that the resolution can be found by the sampling rate of the chirp signal, and value with high magnitude is necessary to properly observe the frequency response of a filter. It can also be stated that the chirp signal is not stable, and convolution with a chirp signal could diverge.

For Q4, it can be stated that sampling with periodic approach forms the periodic output xr(t), even though xα(t) is not periodic. This implies that there is a change of output, one periodic one aperiodic. The chirp signal has the behavior of increasing frequency, while the output has periodically changing frequency, implying that the input reaches higher notes while the output oscillates.

Furthermore regarding Q5, it can be dictated that yr(t) and y2[n] are both periodic, one being discrete and one being continuous. yr(t) is formed by sampling its discrete signal. Every time point in the discrete signal can be sampled by lasting them for sample rate time amounts, finally creating a continuous signal. Another name for this process is interpolation.

The system's cut off frequencies are constant for every signal, and they are symmetric to the y-axis in its plot as in Figure 5 where the magnitude of the frequency response is plotted for $y_r(t)$. The system impulse response's cut off frequencies are as in the filter. This filter suppresses the frequencies in the stopband and allow the frequencies in the bandpass region. This implies only certain frequencies within a signal passes the filter, just as in smartphone microphones that suppress air/wind noise frequencies. We can conclude that such filters change sound signals in desired ways, making them clearer to the human ear, removing certain undesired parts of the input signal.

HANDWRITTEN OBSERVATIONS/SOLUTIONS

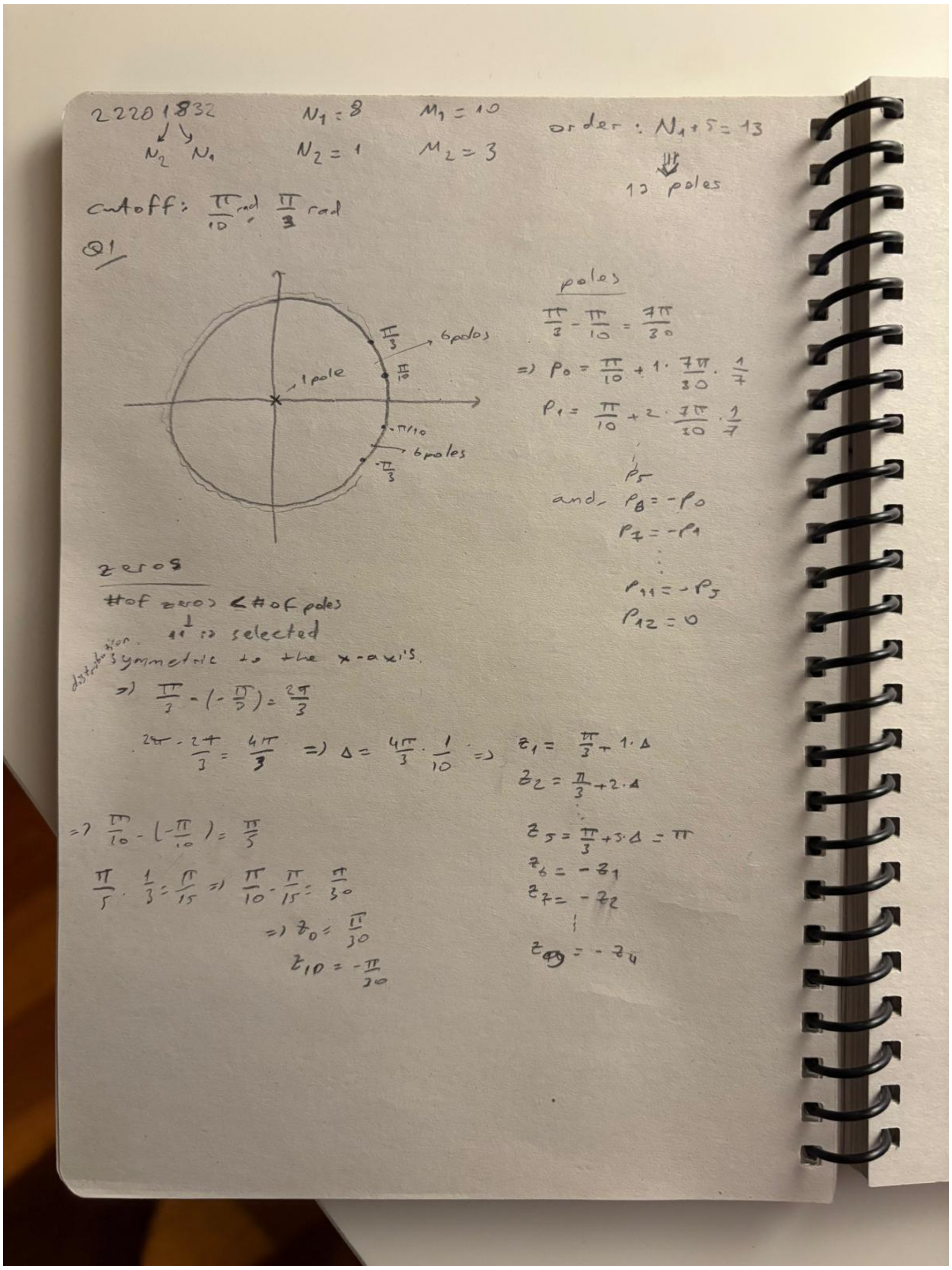

Figure 8: Handwritten Solutions Page 1

Emir Arda Bayer
22201832

$x_a(t) = \cos(\alpha t^2) \rightarrow T = -t \mp \sqrt{t^2 \mp 2\pi k}$ $\quad \frac{d(t^2)}{dt} = 2t$

$\alpha = 1$ $\quad$ cont function!

- $T_s = 1000$ rads^{-2} $\Rightarrow x_2[n] = \cos(\alpha(n \cdot T_s)^2) = \cos\left(\frac{\pi n^2}{8192}\right)$

  $n \in (0:8192)$

- sampling frequency $\alpha = \sqrt{\frac{\pi}{512}} \rightarrow \cos\left(\frac{n^2 \pi}{512}\right)$, $n \in (0\ 8\ 1023)$

  $x_1[n]$

$$y[n] = -\sum_{k=1}^{13} A(k+1)\, y[n-k] + \sum_{l=0}^{11} B(k+1) \cdot x[n-k]$$

$\rightarrow$ using recursion on MATLAB

from eq.

$$y[n] = -\sum_{k=1}^{K} a_k\, y[n-k] + \sum_{l=0}^{L} b_l\, x[n-l]$$

Figure 9: Handwritten Solutions Page 2

Appendix

MATLAB CODE

```
% n1 = 8;
% n2 = 1;
% m1 = 10;
% m2 = 3;
% N1 + 5 = 13 (num of poles);

clc
clear
close all
% 22201832 Emir A. Bayer EEE321 Lab6
syms z
syms t


%% Q1

% H(z) = B(z) / A(z)

%zeros
delta = (4 * pi/3) / 10;

z0 = pi / 30;
z1 = pi / 3 + 1 * delta;
z2 = pi / 3 + 2 * delta;
z3 = pi / 3 + 3 * delta;
z4 = pi / 3 + 4 * delta;
z5 = pi / 3 + 5 * delta;
z6 = -z1;
z7 = -z2;
z8 = -z3;
z9 = -z4;
z10 = -z0;

zeros_list = [z0, z1, z2, z3, z4, z5, z6, z7, z8, z9, z10];
fresz = [exp(1j*zeros_list(1));exp(1j*zeros_list(2));exp(1j*zeros_list(3));...
 exp(1j*zeros_list(4));exp(1j*zeros_list(5));exp(1j*zeros_list(6));...
 exp(1j*zeros_list(7));exp(1j*zeros_list(8));exp(1j*zeros_list(9));...
 exp(1j*zeros_list(10));exp(1j*zeros_list(11))];

B(z) = (z-fresz(1)) * (z-fresz(2)) * (z-fresz(3)) * (z-fresz(4)) * (z-fresz(5)) *...
(z-fresz(6)) * (z-fresz(7)) * (z-fresz(8)) * (z-fresz(9)) * (z-fresz(10)) * (z-fresz(11));

```

```
%poles
deltap = (7 * pi/30) / 7;

p0 = pi / 10 + 1 * deltap;
p1 = pi / 10 + 2 * deltap;
p2 = pi / 10 + 3 * deltap;
p3 = pi / 10 + 4 * deltap;
p4 = pi / 10 + 5 * deltap;
p5 = pi / 10 + 6 * deltap;
p6 = -p0;
p7 = -p1;
p8 = -p2;
p9 = -p3;
p10 = -p4;
p11 = -p5;
p12 = 0;

%0.96 is selected for placing the poles inside the unit circle forstability
poles_list = [p0, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11, p12];
fresp = [0.96*exp(1j*poles_list(1));0.96*exp(1j*poles_list(2));0.96*exp(1j*poles_list(3));...
 0.96*exp(1j*poles_list(4));0.96*exp(1j*poles_list(5));0.96*exp(1j*poles_list(6));...
 0.96*exp(1j*poles_list(7));0.96*exp(1j*poles_list(8));0.96*exp(1j*poles_list(9));...
 0.96*exp(1j*poles_list(10));0.96*exp(1j*poles_list(11));0.96*exp(1j*poles_list(12));...
 0*exp(1j*poles_list(13))];

A(z) = (z-fresp(1)) * (z-fresp(2)) * (z-fresp(3)) * (z-fresp(4)) * (z-fresp(5)) *...
(z-fresp(6)) * (z-fresp(7)) * (z-fresp(8)) * (z-fresp(9)) * (z-fresp(10)) * (z-fresp(11)) *...
(z-fresp(12));

H(z) = B(z) / A(z);
deg = -2*pi : 0.01 : 2*pi;
he = H(exp(deg * 1j));

up = round(double(coeffs(B(z), z)), 8);
down = round(double(coeffs(A(z), z)), 8);
[Acf, Bcf] = zp2tf(fresz, fresp, 1/0.96);

impulse = zeros(1,250);
impulse(11) = 1;
outi = zeros(1,250);
```

```
for i=-10:238
    if i>=3
        outi(i+11) = -Bcf(2)*outi(i+10) - Bcf(3)*outi(i+9) - Bcf(4)*outi(i+8) - Bcf(5)*outi(i+7)...
           -Bcf(6)*outi(i+6) - Bcf(7)*outi(i+5) - Bcf(8)*outi(i+4) - Bcf(9)*outi(i+3)...
           -Bcf(10)*outi(i+2) - Bcf(11)*outi(i+1) -Bcf(12)*outi(i) -Bcf(13)*outi(i-1)...
           -Bcf(14)*outi(i-2) + Acf(4)*impulse(i+11) + Acf(5)*impulse(i+10) + Acf(6)*impulse(i+9)...
           + Acf(7)*impulse(i+8) + Acf(8)*impulse(i+7) + Acf(9)*impulse(i+6) + Acf(10)*impulse(i+5)...
           + Acf(11)*impulse(i+4) + Acf(12)*impulse(i+3) + Acf(13)*impulse(i+2) + Acf(14)*impulse(i+1);
    else
        outi(i+11) = 0;
    end
end
h = outi;
disp(h)
save('h.mat', 'h');


figure(1)

subplot(2,2,1);
zplane(fresz, fresp);
title('Poles and Zeros (x for poles, o for zeros)');

subplot(2,2,2);
plot(deg, abs(double(he)));
title('Magnitude Graph |H(e^jw)|');
xlabel('w in radians');
ylabel('|H(e^jw)|');

subplot(2,2,3);
plot(deg, angle(double(he)));
title('Phase Graph of H(e^jw)');
xlabel('w in radians');
ylabel('phase in radians');

subplot(2,2,4);
stem(-10:239, h, '.k');
title('Impulse Response');
xlabel('n');
ylabel('h[n]');

```

```
%% Q2


n1range = 0:1023;
n2range = 0:8191;
samplerate1 = sqrt(pi/512000);
samplerate2 = sqrt(pi/8192000);
xa(t) = cos(1000*t^2);
x1n = xa(n1range .* samplerate1);
x2n = xa(n2range .* samplerate2);

figure(2)
subplot(2,1,1);
stem(n1range,x1n);
title('x1[n]');
xlabel('n');
ylabel('x1[n]');

subplot(2,1,2);
stem(n2range,x2n);
title('x2[n]');
xlabel('n');
ylabel('x2[n]');



%% Q3

x3n1 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 x1n];
y3n1 = zeros(1,1052);
for i=-10:1040
    if i>=3
        y3n1(i+11) = -Bcf(2)*y3n1(i+10) - Bcf(3)*y3n1(i+9) - Bcf(4)*y3n1(i+8) - Bcf(5)*y3n1(i+7)...
           -Bcf(6)*y3n1(i+6) - Bcf(7)*y3n1(i+5) - Bcf(8)*y3n1(i+4) - Bcf(9)*y3n1(i+3)...
           -Bcf(10)*y3n1(i+2) - Bcf(11)*y3n1(i+1) -Bcf(12)*y3n1(i) -Bcf(13)*y3n1(i-1)...
           -Bcf(14)*y3n1(i-2) + Acf(4)*x3n1(i+11) + Acf(5)*x3n1(i+10) + Acf(6)*x3n1(i+9)...
           + Acf(7)*x3n1(i+8) + Acf(8)*x3n1(i+7) + Acf(9)*x3n1(i+6) + Acf(10)*x3n1(i+5)...
           + Acf(11)*x3n1(i+4) + Acf(12)*x3n1(i+3) + Acf(13)*x3n1(i+2) + Acf(14)*x3n1(i+1);
    else
        y3n1(i+11) = 0;
    end
end
```

```
y31 = y3n1;
save('y31.mat', 'y31');
save('x31.mat', 'x3n1');

figure(3);
subplot(2,1,1);
stem(y3n1, '.k');
title('y1[n]');
xlabel('n');
ylabel('y1[n]');


x3n2 = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 x2n];
y3n2 = zeros(1,8208);
for i=-10:8208
    if i>=3
        y3n2(i+11) = -Bcf(2)*y3n2(i+10) - Bcf(3)*y3n2(i+9) - Bcf(4)*y3n2(i+8) - Bcf(5)*y3n2(i+7)...
           -Bcf(6)*y3n2(i+6) - Bcf(7)*y3n2(i+5) - Bcf(8)*y3n2(i+4) - Bcf(9)*y3n2(i+3)...
           -Bcf(10)*y3n2(i+2) - Bcf(11)*y3n2(i+1) -Bcf(12)*y3n2(i) -Bcf(13)*y3n2(i-1)...
           -Bcf(14)*y3n2(i-2) + Acf(4)*x3n2(i+11) + Acf(5)*x3n2(i+10) + Acf(6)*x3n2(i+9)...
           + Acf(7)*x3n2(i+8) + Acf(8)*x3n2(i+7) + Acf(9)*x3n2(i+6) + Acf(10)*x3n2(i+5)...
           + Acf(11)*x3n2(i+4) + Acf(12)*x3n2(i+3) + Acf(13)*x3n2(i+2) + Acf(14)*x3n2(i+1);
    else
        y3n2(i+11) = 0;
    end
end
y32 = y3n2;
save('y32.mat', 'y32');
save('x32.mat', 'x3n2');
figure(3);
subplot(2,1,2);
stem(y3n2, '.k');
title('y2[n]');
xlabel('n');
ylabel('y2[n]');



%% remaining questions

[me, samplerateme] = audioread("lab5recording.mp3");
xn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 me'];
yn = zeros(1,340928);
```

```
[me, samplerateme] = audioread("lab5recording.mp3");
xn = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 me'];
yn = zeros(1,340928);
for i =-10:34000
    if i>=3
        yn(i+11) = -Bcf(2)*yn(i+10) - Bcf(3)*yn(i+9) - Bcf(4)*yn(i+8) - Bcf(5)*yn(i+7)...
           -Bcf(6)*yn(i+6) - Bcf(7)*yn(i+5) - Bcf(8)*yn(i+4) - Bcf(9)*yn(i+3)...
           -Bcf(10)*yn(i+2) - Bcf(11)*yn(i+1) -Bcf(12)*yn(i) -Bcf(13)*yn(i-1)...
           -Bcf(14)*yn(i-2) + Acf(4)*xn(i+11) + Acf(5)*xn(i+10) + Acf(6)*xn(i+9)...
           + Acf(7)*xn(i+8) + Acf(8)*xn(i+7) + Acf(9)*xn(i+6) + Acf(10)*xn(i+5)...
           + Acf(11)*xn(i+4) + Acf(12)*xn(i+3) + Acf(13)*xn(i+2) + Acf(14)*xn(i+1);
    else
        yn(i+11) = 0;
    end
end

figure(4);
plot(me)
title('Filtered Recording');
xlabel('Time(s)');
ylabel('Amplitude');
audiowrite('iirme.m4a', yn, samplerateme)


figure(5);
plot((deg/samplerate2), abs(double(he)));
title('Magnitude of The Frequency Response');
xlabel('w (rads^-1)');
ylabel('|H(e^jw)| Magnitude');

load("y32.mat", "y32");
ts = sqrt(pi/(8207000));
samplerate32 = 1/ts;
player = audioplayer(y32, samplerate32);
T = samplerate32 .* length(y32);
continuee = 1;
while continuee
    play(player);
    pause(T);
    stop(player);
end
```