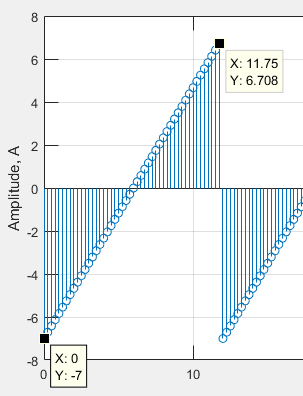
# DIGITAL SIGNAL PROCESSING

# Coursework

# M2.2) Sawtooth and square functions

Figure 1 - Sawtooth + Rectangle functions

In this task both of the required “sawtooth” and “square” functions were used with provided parameters which can be noted as the titles for the subplots of **figure 1**.

It can be noted that the sawtooth function does not reach the applied **amplitude A** on the positive half-cycle. This is due to a set number of samples per period (N = 13) which acts as a sampling frequency. If the ‘**sampling frequency’** increases, the higher frequency component of the sawtooth function, such as the positive peak to negative peak conversion, will be more visible and their amplitude values can be more accurately measured. This can be seen in **figure 2** where the amount of samples per period (sampling frequency) has increased by a factor of 4. This inevitably allowed for a more accurate measurement of high frequency components of the waveform which can be seen as the value converges closer to the amplitude of 7. It is important to note that in an ideal sawtooth waveform of this type, it will be **impossible** for it to reach a positive 7 value as this will indicate that it has reached a value of 7 and -7 at the same time, which is impossible.

No such observations cannot be made for the square waveform as the amplitude of it only varies between two values rather than a wider range. However, the inputted duty cycle of 60% is not a truly represented with the available number of samples N=13. As 8/13=61.5% duty cycle, this value can also be improved by increasing the sampling rate further.

Figure 2 - Sawtooth with 4x samples per period

Both waveforms were plotted using a **stem** function which plots discrete sequence data points at varying x-axis values specified by a desired array with desired variation.

# M2.4) Sinusoidal sequence

Figure 3 – Generated sinusoidal sequence and Generated sequences with varying frequencies

The sinusoid in figure 3 has been plotted using the stem function and inputting a general sinusoid equation into it. The values of it can be varied extensively.

Figure 3 shows the generated waveforms with varying angular frequencies. The x-axis, length, is the number of samples that were requested. However, a sampling frequency component has been added to account for the increase in frequency of the waveforms. Otherwise the waveform quality would start to decrease with increased frequency.

Theoretical periods of waveforms are calculated as follows:

Where w is the angular frequency of the provided waveform.

The fundamental period is a function of discrete time and can be calculated using the following equation:

**N** represents the difference of the number of samples needed to reach the same amplitude value.

**K** represents the number of periods the same amplitude value. For example in table 1, for an angular frequency of 0.34\*pi rad/s, it took 25 samples, or 3 periods to reach the exact same amplitude value of 3

The measured period value is received by measuring the distance between the peaks using the Matlab cursor.

Table - Periods of the waveforms

|  |  |  |  |
| --- | --- | --- | --- |
| Angular frequency(rad) | Theoretical (s) | Measured(s) | Natural period N/k - Discrete |
| 0.14\*pi | 14.29 | 14 | 100/7 |
| 0.24\*pi | 8.33 | 8.3 | 25/3 |
| 0.34\*pi | 5.88 | 5.8 | 100/17 |
| 0.68\*pi | 2.94 | 2.9 | 50/17 |
| 0.75\*pi | 2.67 | 2.7 | 8/3 |

# M3.1) Components of a Transfer Function

for 0<r<1

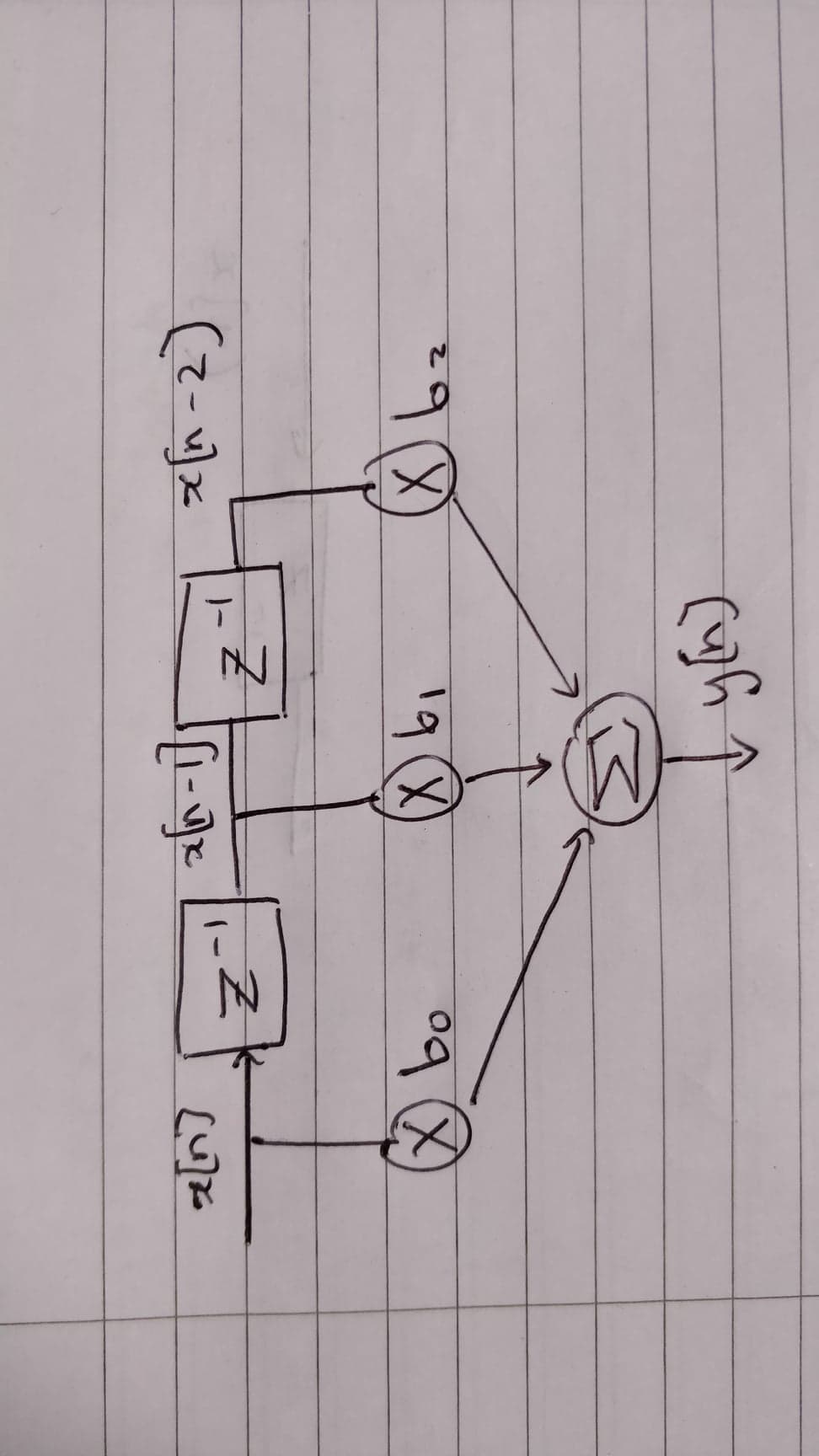
This transfer function represents a Finite Impulse Response (FIR) filter as it has no components on the numerator. Since it is a Discrete Time Fourier Tranform (DTFT) it is expected that in the frequency domain, the output of the transfer function will be periodic, therefore it will represent a mirror image every 2\*pi radians. This means that a range of 2\*pi on the x-axis should be enough to showcase all of the needed information.

Figure 4 - IIR diagrammatic representation

This diagram represents the given transfer function. The b(0,1,2) values are as follows:

b0 = 1

b1 = -2rcos

b2 = r^2

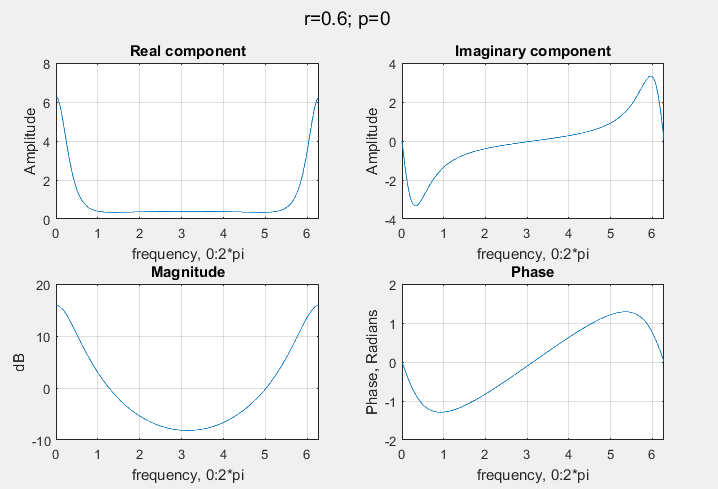


Figure 4 – Components at r=0.6, p=0

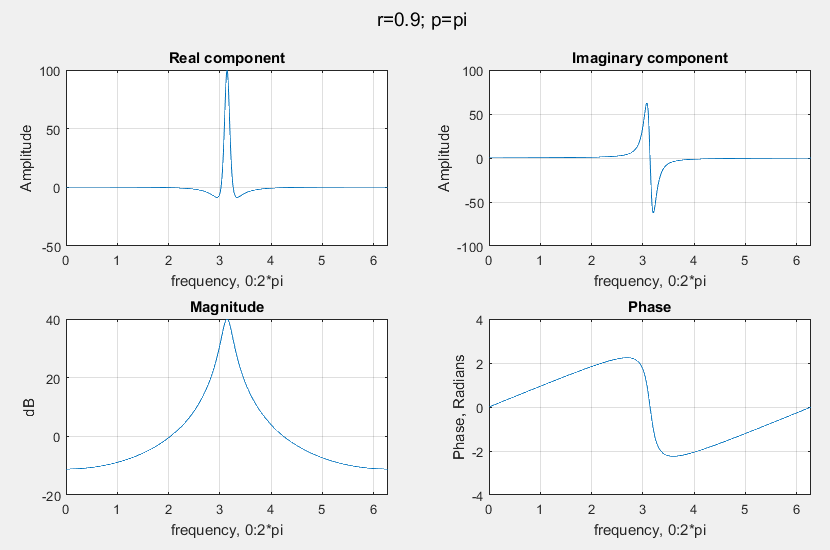


Figure 5– Components at r=0.9, p=pi

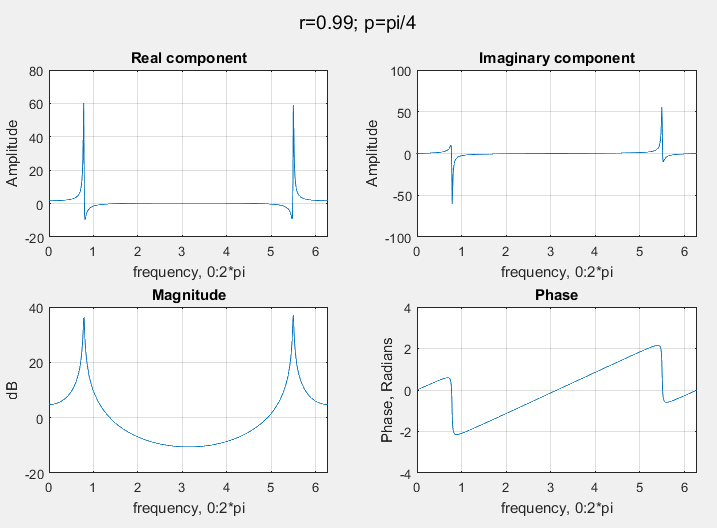


Figure 6 – Components at r=0.99, p=pi/4

Figures 4,5 and 6 show the characteristics of the filter as the values of ‘r’ and phase change. The ‘r’ value acts as a **scalar** for all characteristics within the given range of 0<r<1. As the value of r increases, without the phase variation, the amplitudes of all characteristics increase.

The phase component **shifts** the response of the filter onto itself. The left and right peaks of each characteristic **converge** into each other and co**nstructively interfere** the most at ‘pi’ radians. That is why there is high amplitude response of real, imaginary and magnitude characteristics when phase difference is at pi which can be seen in figure 5.

Changing these the scalar and phase values changes the inherent nature of the filter by changing the positions of poles and zeros. Originally it seems to act as a **band stop filter**, however as the values of ‘r’ and the phase changed, it became a **band pass filter** which is very clear at the phase value of pi radians. The scalar component varies the amplification and attenuation of the filter.

# M3.9) Convolution

Inbuilt MATLAB function conv(A, B) produces a linear convolution of 2 vectors and will have a final size of An +Bn – 1.

For 2 vectors of the same length, the circular convolution is equal to the inverse DFT of the product of the vector’s DFTs. The size of the circular convolution is the same size as the biggest vector that is being circularly convolved.

Or can be computed using the inbuilt cconv(a, b, n) function.

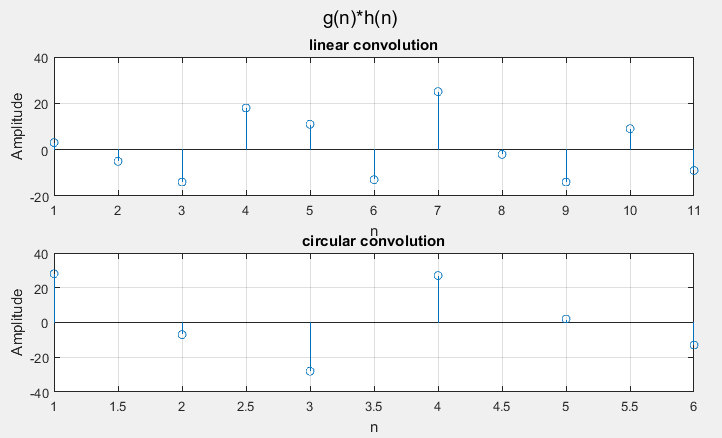
The convolution of g[n] and h[n] can be seen in figure 7. As expected, the size of the output array for the linear convolution followed the above principles and outputted 11 results (An+Bn -1).

Figure 7 - convolution of gn and hn

Complex numbers are difficult to represent in a graph format; therefore, convolutions of both exercises can be seen in Table 2.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Linear A | 3.0 | -5.0 | -14.0 | 18.0 | 11.0 | -13.0 | 25.0 | -2.0 | -14.0 | 9.0 | -9.0 |
| Circular A | 28.0 | -7.0 | -28.0 | 27.0 | 2.0 | -13.0 |  |  |  |  |  |
| Linear B | -13i | -21+8i | 10+14i | 3+15i | -1-2i | -26+27i | -20+23i | -8+29i | -6+8i |  |  |
| Circular B | -26+14i | 41+31i | 2+43i | -3+23i | -1-2i |  |  |  |  |  |  |

Table 2 – Linear and circular convolutions of and b exercises

# M4.6) Causal IIR digital Filter

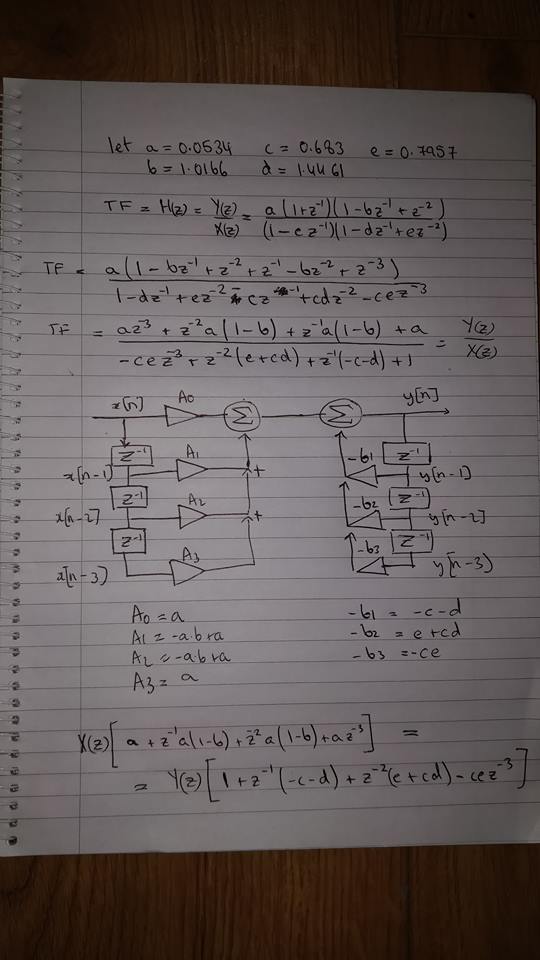


Figure 8 - Filter diagram, values and working out

Figure 8 shows the diagrammatic representation of the given transfer function H(z). It is **3rd order** **filter** as can be seen by the existence of **3rd order** delays in IIR and FIR components of the filter.

The alphabetical variables are used to simplify the visualisation of the diagram.

The difference equation is therefore as follows:

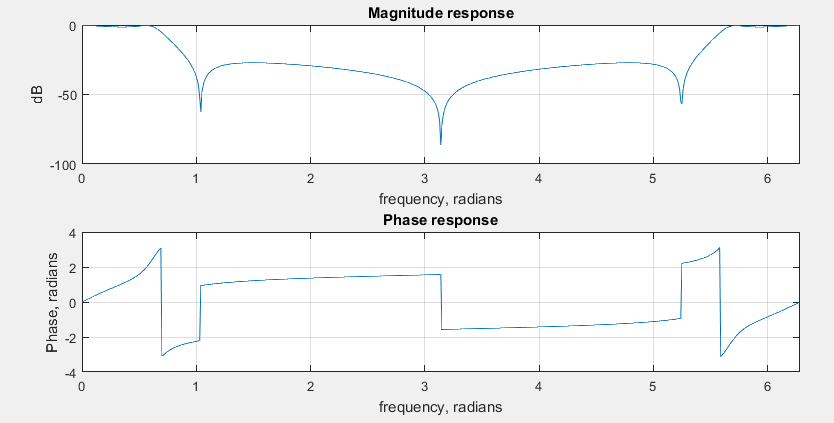


Figure 9- Magnitude and Phase response of the filter

By looking at the magnitude response of the filter in figure 9, it seems that the function of the filter is to act as a **Band Stop filter** at pi and pi/3 radians.

Appendix – MATLAB CODE

M2.2)

% M2.2)

%A) Sawtooth-------------------------------------

close all

clear all

A=7; %Peak amplitude

N=13; %Period

l=100; %Sample size

DC=60; %Duty cycle percentage

t=0:1:l-1; %discrete time

S=sawtooth(2\*pi\*(1/(N-1))\*(t));

S=S\*7;

figure

stem(t,S); title('Sawtooth Wave (A=7, N=13)');

ylabel('Amplitude, A'); xlabel('Number of samples, L');

grid on

%%

%B) Square -------------------------------

figure(2)

sq= A\*square(2\*pi\*(1/N)\*t,DC);

stem(t,sq); title('Square Wave (A=7, N=13, DC=60)');

ylabel('Amplitude, A'); xlabel('Number of samples, L');

grid on

%%

% subplot of both --------------------------

figure(3)

subplot(2,1,1);

stem(t,S);

title('Sawtooth Wave(A=7, N=13)');

ylabel('Amplitude, A'); xlabel('Number of samples, L');

subplot(2,1,2);

stem(t,sq);

title('Square Wave (A=7, N=13, DC=60)');

ylabel('Amplitude, A'); xlabel('Number of samples, L');

grid

M2.4)

%M2.4) a)generating sinusoidal sequence-------------

clear all

close all

A=3; %amplitude

L=150;

wo=pi/4; %angular frequency (between 0 and pi)

p=0; %phase angle (between 0 and 2pi)

figure(1)

n = 0:0.5:L;

x=A\*cos(wo\*n+p);

stem(n, x);

title('Generated sequence');

ylabel('Amplitude, A'); xlabel('Length');

%{

%M2.4) b) varying angular frequencies--------------

% figure(2)

% x1=A\*cos(0.14\*pi\*n+p); %0.14

% G1=stem(n,x1);

% title('0.14\*pi Angular Frequency');

%

% figure(3)

% x2=A\*cos(0.24\*pi\*n+p); %0.24

% G2=stem(n,x2);

% title('0.24\*pi Angular Frequency');

%

% figure(4)

% x3=A\*cos(0.34\*pi\*n+p); %0.34

% G3=stem(n,x3);

% title('0.34\*pi Angular Frequency');

%

% figure(5)

% x4=A\*cos(0.68\*pi\*n+p); %0.68

% G4=stem(n,x4);

% title('0.68\*pi Angular Frequency');

%

% figure(6)

% x5=A\*cos(0.75\*pi\*n+p); %0.75

% G5=stem(n,x5);

% title('0.75\*pi Angular Frequency');

%}

%%

figure(7)

subplot(5,1,1)

x1=A\*cos(0.14\*pi\*n+p); %0.14

stem(n,x1); title('0.14\*pi Angular Frequency');

ylabel('Amplitude, A'); xlabel('Length');

subplot(5,1,2)

x2=A\*cos(0.24\*pi\*n+p); %0.24

stem(n,x2); title('0.24\*pi Angular Frequency');

ylabel('Amplitude, A'); xlabel('Length');

subplot(5,1,3)

x3=A\*cos(0.34\*pi\*n+p); %0.34

stem(n,x3); title('0.34\*pi Angular Frequency');

ylabel('Amplitude, A'); xlabel('Length');

subplot(5,1,4)

x4=A\*cos(0.68\*pi\*n+p); %0.68

stem(n,x4); title('0.68\*pi Angular Frequency');

ylabel('Amplitude, A'); xlabel('Length');

subplot(5,1,5)

x5=A\*cos(0.75\*pi\*n+p); %0.75

stem(n,x5); title('0.75\*pi Angular Frequency');

ylabel('Amplitude, A'); xlabel('Length');

M3.1)

%Real+Imaginary parts, magnitude and phase spectra of DTFt

%M3.1-------------

close all; clear all; clc

%parameters

r1=0.6;

p1=0;

w=0:0.01:4\*pi;

%DTFT equation

G1=1./(1-2\*r1\*(cos(p1))\*exp(-1i\*w)+r1^(2)\*exp(-1i\*2\*w));

%Points of interest

Re1=real(G1);

Im1=imag(G1);

Magnitude1=abs(G1);

Phase1=angle(G1);

%plotting af r=0.6; p=0;

figure()

subplot(2,2,1)

plot(w,Re1); title('Real component'); grid on

subplot(2,2,2)

plot(w,Im1); title('Imaginary component'); grid on

subplot(2,2,3)

plot(w,Magnitude1); title('Magnitude'); grid on

subplot(2,2,4)

plot(w,Phase1); title('Phase'); grid on

suptitle('r=0.6; p=0')

%%

%plotting af r=0.9; p=pi;

%parameters

r2=0.9;

p2=pi;

%DTFT equation

G2=1./(1-2\*r2\*(cos(p2))\*exp(-1i\*w)+r2^(2)\*exp(-1i\*2\*w));

%Points of interest

Re2=real(G2);

Im2=imag(G2);

Magnitude2=abs(G2);

Phase2=angle(G2);

figure()

subplot(2,2,1)

plot(w,Re2); title('Real component'); grid on

subplot(2,2,2)

plot(w,Im2); title('Imaginary component'); grid on

subplot(2,2,3)

plot(w,Magnitude2); title('Magnitude'); grid on

subplot(2,2,4)

plot(w,Phase2); title('Phase'); grid on

suptitle('r=0.9; p=pi')

%%

%plotting af r=0.99; p=pi/4;

%parameters

r3=0.99;

p3=pi/4;

%DTFT equation

G3=1./(1-2\*r3\*(cos(p3))\*exp(-1i\*w)+r3^(2)\*exp(-1i\*2\*w));

%Points of interest

Re3=real(G3);

Im3=imag(G3);

Magnitude3=abs(G3);

Phase3=angle(G3);

figure()

subplot(2,2,1)

plot(w,Re3); title('Real component'); grid on

subplot(2,2,2)

plot(w,Im3); title('Imaginary component'); grid on

subplot(2,2,3)

plot(w,Magnitude3); title('Magnitude'); grid on

subplot(2,2,4)

plot(w,Phase3); title('Phase'); grid on

suptitle('r=0.99; p=pi/4')

M3.9)

%circular convolution of a sequence

close all; clear all; clc;

%length of linear conv. is gn+hn-1

% a)

g=[3, 4, -2, 0, 1, -3];

h=[1, -3, 0, 4, -2, 3];

% gfft=fft(g);

% hfft=fft(h);

%Conv=gfft.\*hfft %time convolution

linearA = conv(g, h); %linear convolution

circularA = ifft(fft(g).\*fft(h)); %circular convolution

new1=cconv(g, h, 6);

%%

% b)

x=[2+3\*1i, 3-1i, -1+2\*1i, 3\*1i, 2+4\*1i];

v=[-3-2\*1i, 1+4\*1i, 1+2\*1i, 5+3\*1i, 1+2\*1i];

linearB = conv(x, v); %linear convolution

circularB = ifft(fft(x).\*fft(v)); %cincular convolution

new2=cconv(x, v, 5);

%%

%plotting

figure(1)

subplot(2,1,1)

stem(linearA); title('linear convolution');

ylabel('Amplitude'); xlabel('n'); grid on;grid on;

subplot(2,1,2)

stem(circularA); title('circular convolution');

ylabel('Amplitude');xlabel('n'); grid on; grid on;

suptitle('g(n)\*h(n)')

figure(2)

subplot(2,1,1)

stem(linearB); title('linear convolution');

ylabel('Amplitude'); grid on;

subplot(2,1,2)

stem(circularB); title('circular convolution');

ylabel('Amplitude'); grid on;

suptitle('b)')

%%

figure(3)

ReCA=real(circularA);

ImCA=imag(circularA);

subplot(2,1,1)

stem(ReCA);

subplot(2,1,2);

stem(ImCA);

%%

figure(3)

ReCB=real(circularB);

ImCB=imag(circularB);

subplot(2,1,1)

stem(ReCB);

subplot(2,1,2);

stem(ImCB);

M4.6)

%M4.6) Causal IIR filter

% https://ece.uwaterloo.ca/~ece413/HW/Solution1.pdf

% https://uk.mathworks.com/help/ident/ref/bode.html TF

close all;

clear all;

k = 1;

for w = 0:0.01:2\*pi %frequency range

z=exp(-1i\*w);

%Transfer function

H(k)=(0.0534.\*(1+z.^(-1)).\*(1-1.0166.\*z.^(-1)+z.^(-2)))./...

((1-0.683.\*z.^(-1)).\*(1-1.4461.\*z.^(-1)+0.7957.\*z.^(-2)));

k = (k + 1);

end

Magnitude=abs(H);

Amp=20\*log10(Magnitude);

Phase=angle(H);

figure()

subplot(2,1,1)

plot(0:0.01:2\*pi, Amp); title('Magnitude response'); grid on

xlim([0, 2\*pi]); ylabel('dB'); xlabel('frequency, radians')

subplot(2,1,2)

plot(0:0.01:2\*pi, Phase); title('Phase response'); grid on

xlim([0, 2\*pi]); ylabel('Phase, radians'); xlabel('frequency, radians')