

Digital Signal Processing Lab Manual (MATLAB)

B.Tech III Sem
Electronics &
Communication
Engineering

1. Linear Convolution

```
% Input the sequences
x1n = input('Enter the first sequence: ');
x2n = input('Enter the second sequence: ');

% Perform convolution
yn = conv(x1n, x2n);

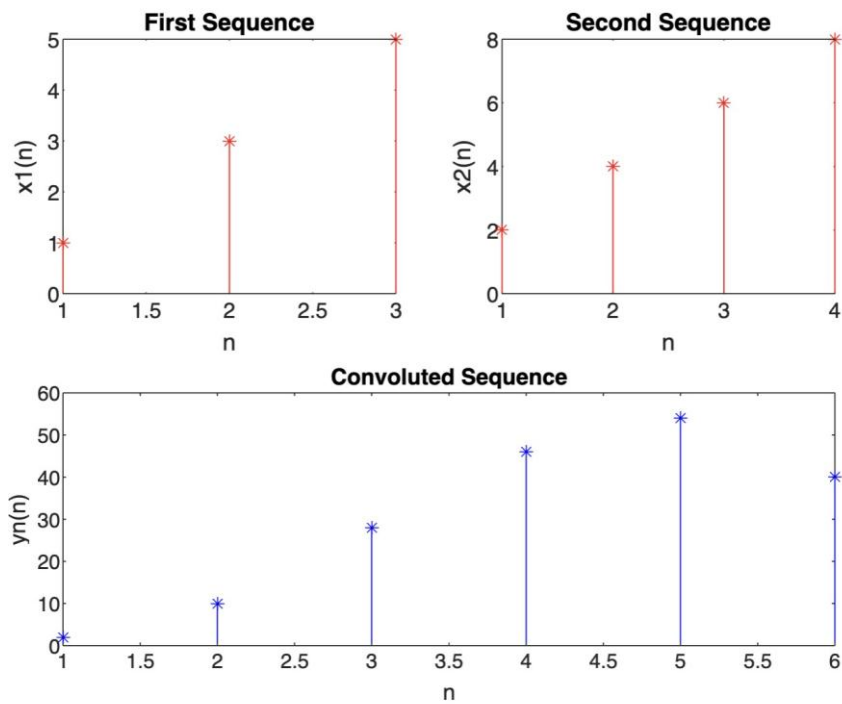
% Display the sequences and the convolution result
disp('Convolved output is: ');
disp(yn);

% Plot the sequences and the convolution result
subplot(2,2,1);
stem(x1n);
xlabel('n');
ylabel('x1(n)');
title('First Sequence');

subplot(2,2,2);
stem(x2n);
xlabel('n');
ylabel('x2(n)');
title('Second Sequence');

subplot(2,1,2);
stem(yn);
xlabel('n');
ylabel('yn(n)');
title('Convolved Sequence');
```

```
>> convol
Enter the first sequence (e.g., [1 2 3]):
[1 3 5]
Enter the second sequence (e.g., [4 5 6]):
[2 4 6 8]
First sequence is: 1 3 5
Second sequence is: 2 4 6 8
Convolved output is:
2 10 28 46 54 40
```



2. Circular Convolution

```
xn=input('Enter the first sequence x(n): ');
hn=input('Enter the second sequence h(n): ');
l1=length(xn);
l2=length(hn);
N=max(l1,l2);

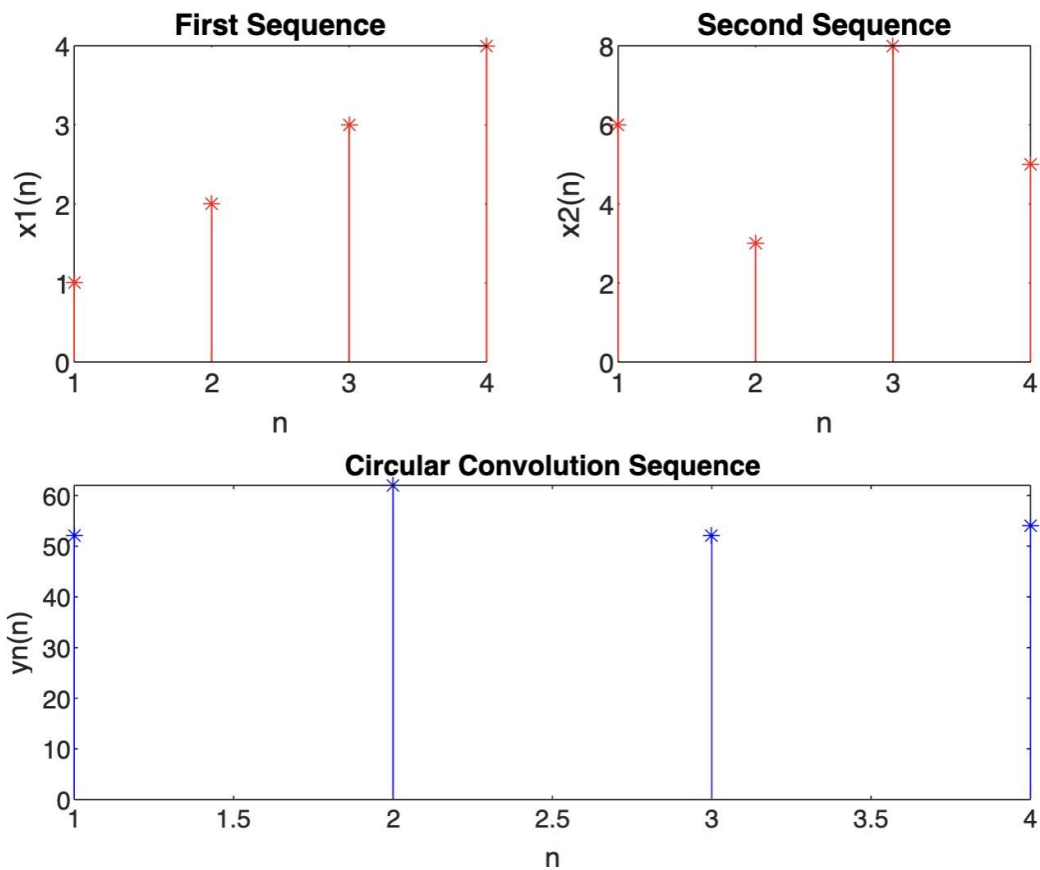
% Perform convolution
y = cconv(xn, hn, N);

% Padding Zeros
xn=[xn, zeros(1, N-l1)];
hn=[hn, zeros(1, N-l2)];

% Displaying Output
disp('circular convolution of the given sequence is: ');
disp(y);

% Plotting the Input & Output
subplot(2,2,1);
stem(xn);
xlabel('n');
ylabel('x(n)');
title('plot of first seq x(n)')
subplot(2,2,2);
stem(hn);
xlabel('n');
ylabel('h(n)');
title('plot of second seq h(n)')
subplot(2,1,2);
stem(y);
xlabel('n');
ylabel('y(n)');
title('plot of circular convolution y(n)');
```

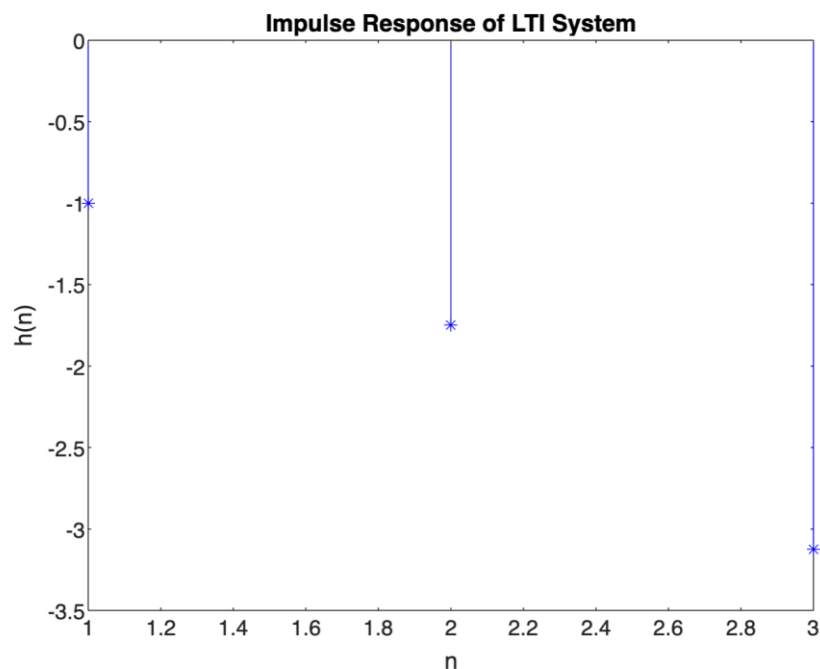
Enter the first sequence (e.g., [1 2 3]):
[1 2 3 4]
Enter the second sequence (e.g., [4 5 6]):
[6 3 8 5]
First sequence is: 1 2 3 4
Second sequence is: 6 3 8 5
Circularly Convolved output is:
52 62 52 54



3. Impulse Response

```
y = input('input sequence y(n) of the system: ');
x = input('input sequence x(n)of the system:');
N = input ('Enter the length of the impulse response sequence:');
h = impz(x, y, N);
disp ('Impulse response of system h(n)');
disp (h);
% Plotting
n = 0:1:N-1;
stem (n,h);
xlabel ('n');
ylabel ('h(n)');
title ('Impulse response');
```

```
>> impz
Enter input sequence:
[-1 2]
Enter output sequence:
[1 -1/4 -3/8]
Length of impulse response:
3
Input sequence: -1 2
Input sequence: 1 -0.25 -0.375
Impulse response
-1.0000
-1.7500
-3.1250
```

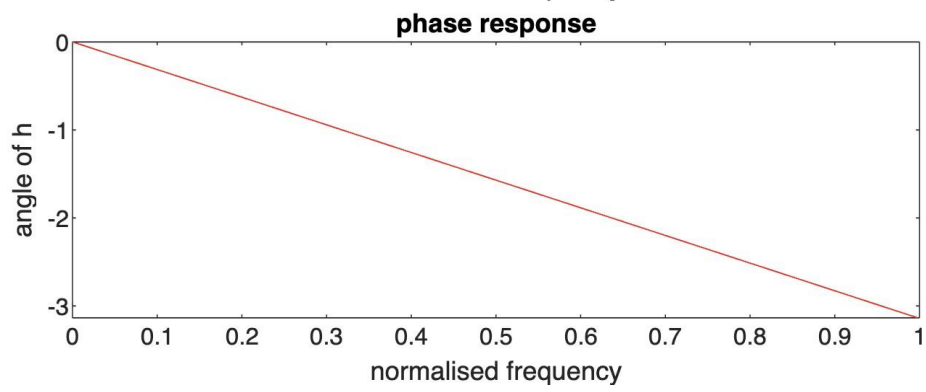
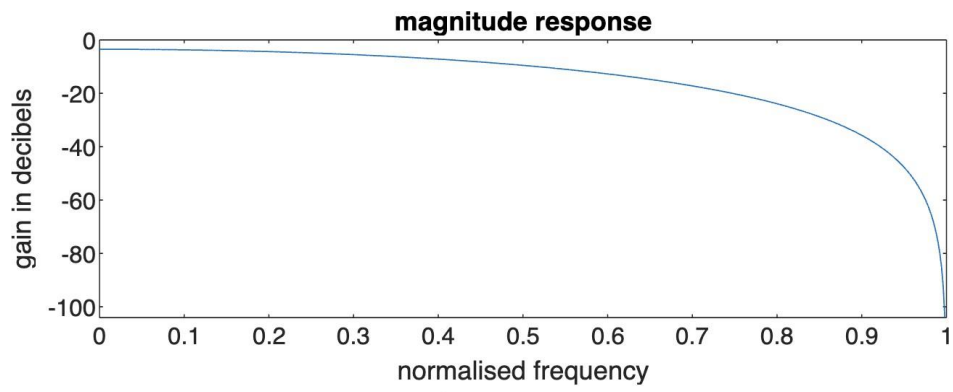


4. Frequency Response:

```
clc;
close all;
clear all;
num = input('Enter the numerator coefficients: ');
den = input('Enter the denominator coefficients: ');
[h, om] = freqz(num, den);
disp(h);

freqz(num, den);
```

```
enter the numerator coefficients:
[1/6 1/3 1/6]
enter the denominator coefficients:
[1]
0.6667 + 0.0000i
0.6666 - 0.0041i
0.6666 - 0.0082i
0.6665 - 0.0123i
0.6664 - 0.0164i
0.6662 - 0.0204i
0.6660 - 0.0245i
0.6657 - 0.0286i
```



5. N-Point DFT

```
clc;
close all;
clear all;
N = input('Enter the N point: ');
Xn = input('Enter the input sequence x(n):');
Xk = fft(Xn, N);
disp('N point DFT of x(n) is = ');
disp(Xk);

subplot(2,1,1);
n = 0:1:length(Xn)-1;
stem(n, Xn);
xlabel('n');
ylabel('x(n)');
title('original signal');

subplot(2,2,3);
k = 0:N-1;
stem(k, abs(Xk));
xlabel('k');
ylabel('|x(k)|');
title('Magnitude Spectrum');

subplot(2,2,4);
stem(k, angle(Xk));
xlabel('k');
ylabel('<x(k)');
title('Phase Spectrum');
```

Enter number of samples N:

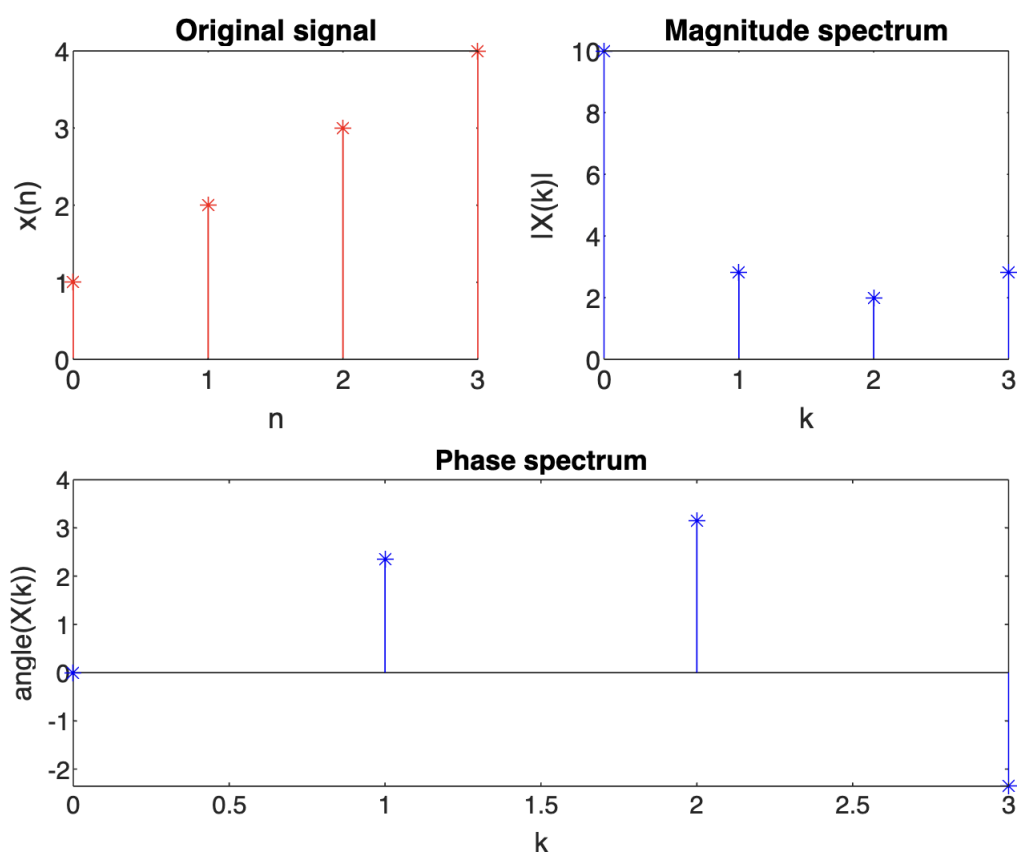
4

Enter the input sequence x(n):

[1 2 3 4]

N-point DFT is:

10.0000 + 0.0000i -2.0000 + 2.0000i -2.0000 + 0.0000i -2.0000 - 2.0000i



6. Linear Convolution using DFT & IDFT

```
clc;
close all;
clear all;
xn = input('Enter the sequence x(n): ');
hn = input('Enter the sequence h(n): ');
N = length(xn) + length(hn)-1;
xk = fft(xn, N);
hk = fft(hn, N);
yk = xk.*hk; % .* multiplies every element of xk to
hk_yn = ifft(yk, N);
disp('Linear Convolution of x(n) & h(n): ');
disp(hk_yn);

subplot(2,2,1);
stem(xn);
xlabel('n');
ylabel('x(n)');
title('plot of x(n)');

subplot(2,2,2);
stem(hn);
xlabel('n');
ylabel('h(n)');
title('plot of h(n)');

subplot(2,2,3);
stem(hk_yn);
xlabel('n');
ylabel('y(n)');
title('DFT IDFT convolution output');

% Verification
yv = conv(xn, hn);
disp(yv);
subplot(2,2,4);
stem(yv);
xlabel('n');
ylabel('yv(n)');
title('verified convolution');
```

7. Circular Convolution using DFT IDFT

```
clc;
close all;
clear all;
xn = input('Enter the sequence x(n): ');
hn = input('Enter the sequence h(n): ');
N = max(length(xn), length(hn));
xk = fft(xn, N);
hk = fft(hn, N);
yk = xk.*hk;
yn = ifft(yk, N);
disp('Circular Convolution of x(n) & h(n): ');
disp(yn);

subplot(2,2,1);
stem(xn);
xlabel('n');
ylabel('x(n)');
title('plot of x(n)');

subplot(2,2,2);
stem(hn);
xlabel('n');
ylabel('h(n)');
title('plot of h(n)');

subplot(2,1,2);
stem(yn);
xlabel('n');
ylabel('y(n)');
title('DFT IDFT circular convolution output');
```

8. Sampling Theorem

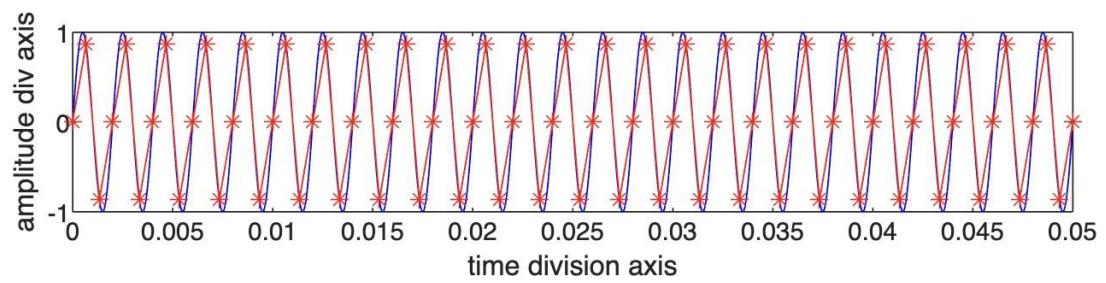
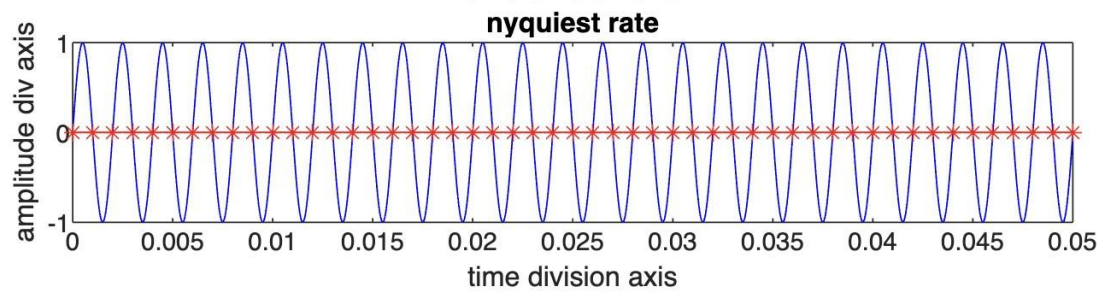
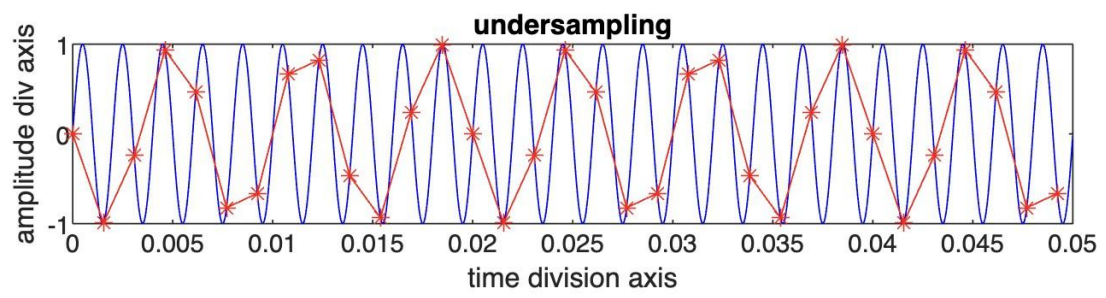
```
clc;
clear all;
close all;
final = 0.05;
t = 0:0.00005:final;
fd = input('Enter analog frequency: ');
xt = sin(2*pi*fd*t);

fs1 = 1.3 * fd;
n1 = 0:1/fs1:final;
xn = sin(2*pi*fd*n1);
subplot(3,1,1);
plot(t, xt, n1, xn, 'r*-');
xlabel('time division axis');
ylabel('amplitude div axis');
title('undersampling');

fs2 = 2.0 * fd;
n2 = 0:1/fs2:final;
xn = sin(2*pi*fd*n2);
subplot(3,1,2);
plot(t, xt, n2, xn, 'r*-');
xlabel('time division axis');
ylabel('amplitude div axis');
title('nyquist rate');

fs3 = 3 * fd;
n3 = 0:1/fs3:final;
xn = sin(2*pi*fd*n3);
subplot(3,1,3);
plot(t, xt, n3, xn, 'r*-');
xlabel('time division axis');
ylabel('amplitude div axis');
title('oversampling');
```

Enter analog frequency:
500



9. Z-Transform

```
clc;
close all;
clear all;
syms n;
a = 2;
x = a^n;
X = ztrans(x);
disp('Z Transform of a^n, a > 1');
disp(X);

syms n;
a = 0.5;
x = a^n;
X1 = ztrans(x);
disp('Z Transform of a^n, 1 > a > 0');
disp(X1);

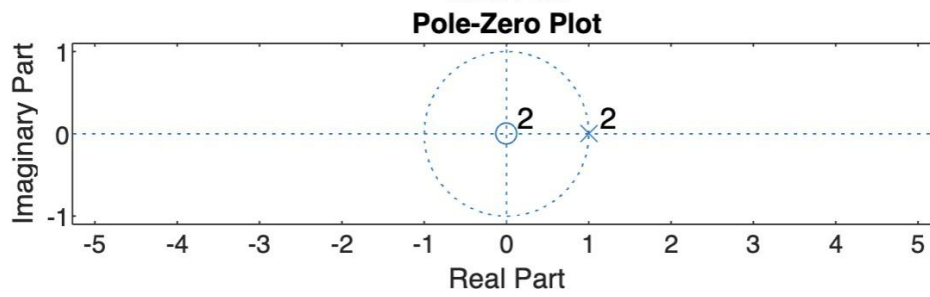
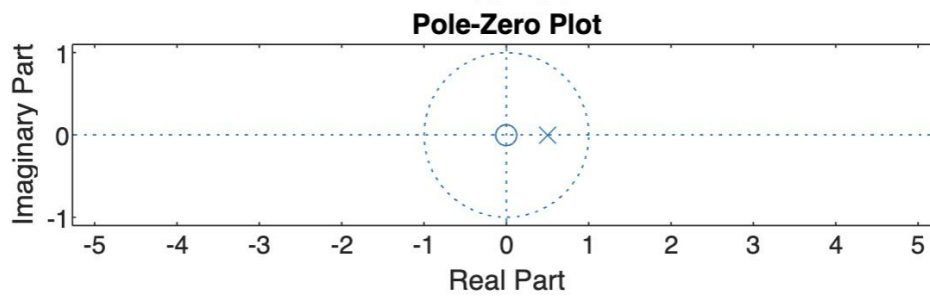
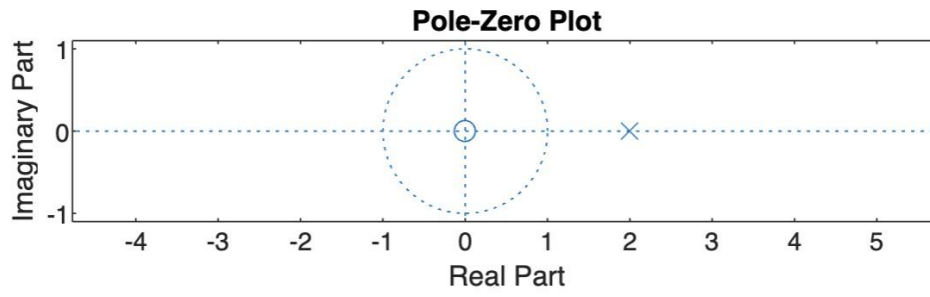
syms n;
x = n+1;
X2 = ztrans(x);
disp('Z transformation of 1+n');
disp(X2);

A = iztrans(X);
disp('Inverse Z Transform of a^n, a > 1: ');
disp(A);

B = iztrans(X1);
disp('Inverse Z Transform of a^n, 0 > a > 1: ');
disp(B);

C = iztrans(X2);
disp('Inverse Z Transform of 1+n: ');
disp(C);

subplot(3,1,1);
zplane([1 0], [1 -2]);
subplot(3,1,2);
zplane([1 0], [1 -1/2]);
subplot(3,1,3);
zplane([1 0 0], [1 -2 1]);
```



10. Design and Implementation of IIR Filter

10.1. Butterworth

```
clc;
clear all;
close all;

fp=input('Enter passband frequency in Hz: ');
fs=input('Enter stopband frequency in Hz: ');
ff=input('Enter sampling frequency in Hz: ');
ap = input('passband gain in dB: ');
as = input('stopband attenuat in dB: ');
wp = 2*pi*(fp/ff);
ws = 2*pi*(fs/ff);
up = 2*tan(wp/2);
us = 2*tan(ws/2);
[n, wc]=buttord(up, us, ap, as, 's');

disp('order of filter is: ');
disp(n);
disp('Normalised frequency is: ');
disp(wc);

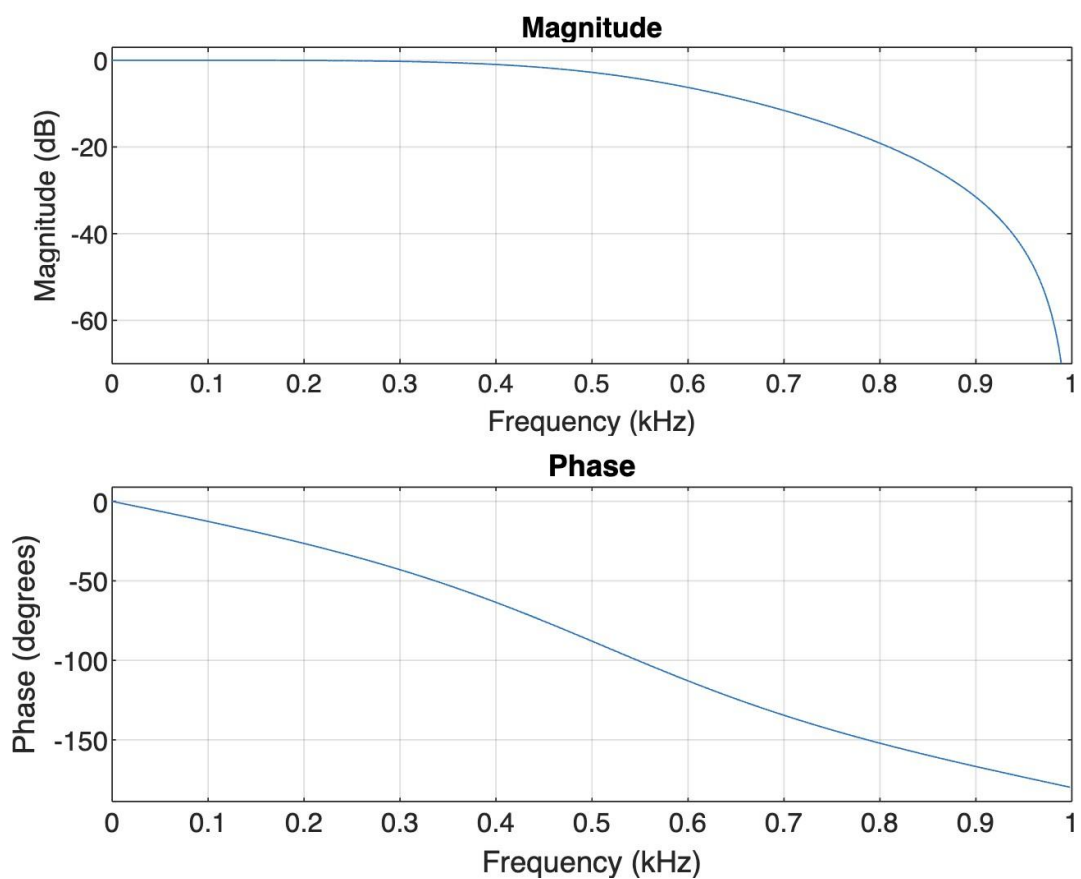
[num, den] = butter(n, wc, 's');
[b,a] = bilinear(num, den, 1);
freqz(b, a, 512, ff);
printsys(b,a,'z');
```

enter passband frequency in Hz:
500
enter stopband frequency in Hz:
750
enter sampling frequency in Hz:
2000
passband ripple in dB:
3.01
stopband ripple in dB:
15
order of filter is:
2

Normalised frequency is:
2.0526

num/den =

$$\frac{0.30053 z^2 + 0.60106 z + 0.30053}{z^2 + 0.030385 z + 0.17174}$$



10.2. Chebyshev

```
clc;
clear all;
close all;

fp=input('Enter passband frequency in Hz: ');
fs=input('Enter stopband frequency in Hz: ');
ff=input('Enter sampling frequency in Hz: ');
ap = input('passband ripple in dB: ');
as = input('stopband ripple in dB: ');
wp = 2*pi*(fp/ff);
ws = 2*pi*(fs/ff);
up = 2*tan(wp/2);
us = 2*tan(ws/2);

[n, wc]=cheb1ord(up, us, ap, as, 's');
disp('order of filter is: ');
disp(n);
disp('Normalised frequency is: ');
disp(wc);

[num, den] = cheby1(n, ap, wc, 's');

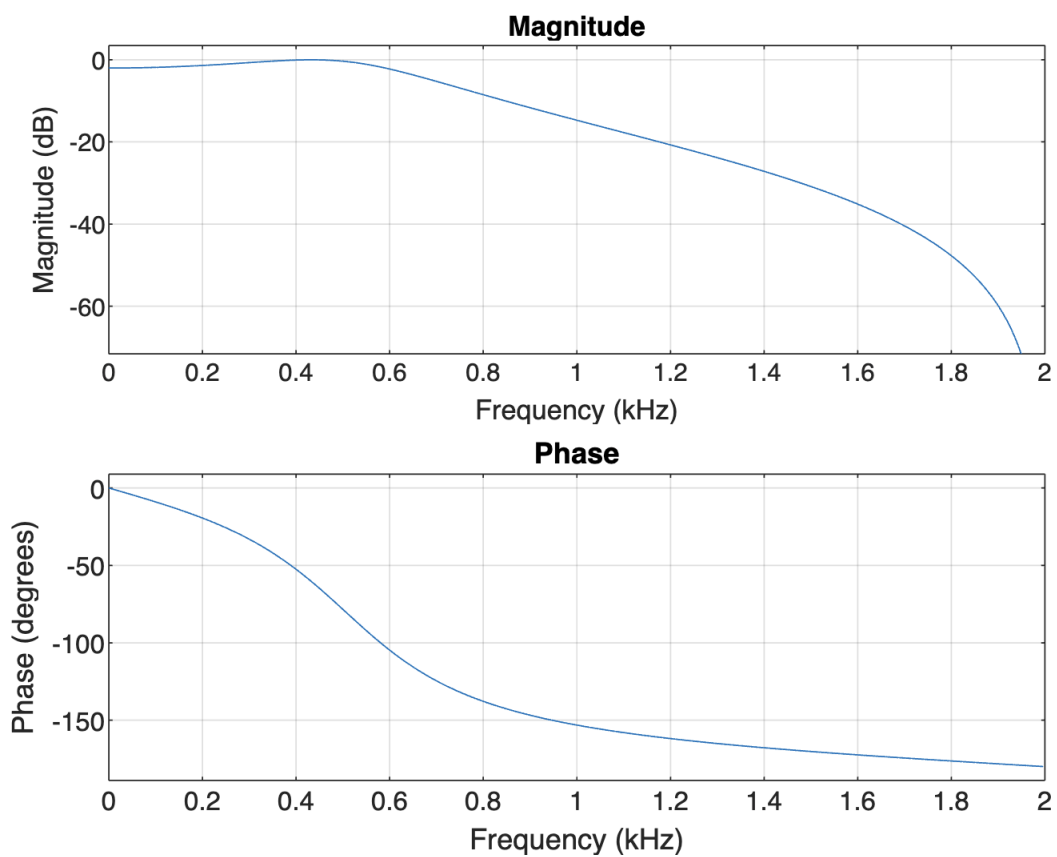
[b,a] = bilinear(num, den, 1);
freqz(b, a, 512, ff);
printsys(b,a,'z');
```

```
enter passband frequency in Hz:
100
enter stopband frequency in Hz:
500
enter sampling frequency in Hz:
4000
passband ripple in dB:
2
stopband ripple in dB:
20
order of filter is:
2
```

```
Normalised frequency is:
0.1574
```

```
num/den =
```

$$\frac{0.10167 z^2 + 0.20333 z + 0.10167}{z^2 - 0.98806 z + 0.50001}$$



11. Design and Implementation of FIR Filter

11.1. Hamming Window

```
clc;
clear all;
close all;
f = input('Enter frequency: ');
fs = input('Sampling Frequency: ');
N = input('Order of filter: ');
wc = 2*pi*(f/fs);
h= fir1(N-1, wc/pi, hamming (N));

disp('hamming window coefficients: ');
disp(hamming(N));
disp('impulse response of the FIR filter: ');
disp(h);

figure(1);
freqz(h);

figure(2);
n = 0:1:N-1;
stem(n,h);
xlabel('n');
ylabel('h(n)');
title('impulse response of the FIR filter');
```

```
Enter frequency:
1000
Sampling Frequency:
4000
Order of filter:
5
hamming window coefficients:
    0.0800
    0.5400
    1.0000
    0.5400
    0.0800

impulse response of the FIR filter:
    0.0000    0.2037    0.5926    0.2037    0.0000
```

