Name – aman kumar

Roll – no – EE22btech11006

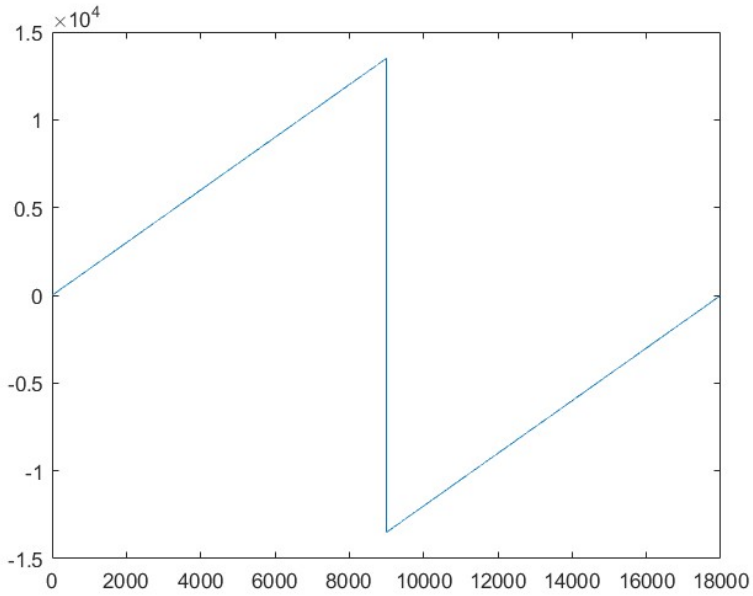
Code:

```
T = 3;
f_o = 500;
M=3;
L=3;
fs = 2*2*T*f_o*M;
Ts = 1/fs;
t= linspace(0,1,fs);
x = sin(2*pi*f_o*t);
for i = 1:T
    fi = 2*i*f_o;
    x = x + (i+1)*sin(2*pi*fi*t)/2;
end
%disp(x);
%plot(x(1:30));


down = down_sample(x,M);
up = up_sample(x,L);
x_u = up;
%disp(x_u);
%plot(x_u(1:30));
wc = pi/L;
T_1 = Ts/L;
f1= 1/T_1;

[xi,h] = lowpassfilter(x_u,wc);
%disp(xi);
%fvtool(xi);
%plot(xi(1:30));
%fvtool(h);
X_f = fft(x_i);
N = length(x_i);
f = (-N/2 : N/2-1) * (fs / N);
%X_if = fft(xi);
X_mag_i = abs(X_if);
figure;
plot(f, fftshift(X_mag_i));
plot(freqX);
xf = lowpassfilter(x,wc);
xd = down_sample(xf,M);
%plot(xd(1:30));
%fvtool(xd);
function [xi,h] = lowpassfilter(x,wc)
    N = 39;
    h_d= zeros(1,N);
    for n = -(N-1)/2:(N-1)/2
        if n == 0
            h_d(n+(N-1)/2 +1) = wc/pi;
        else
            h_d(n+(N-1)/2+1) = sin(wc*n)/(pi*n);
        end
```

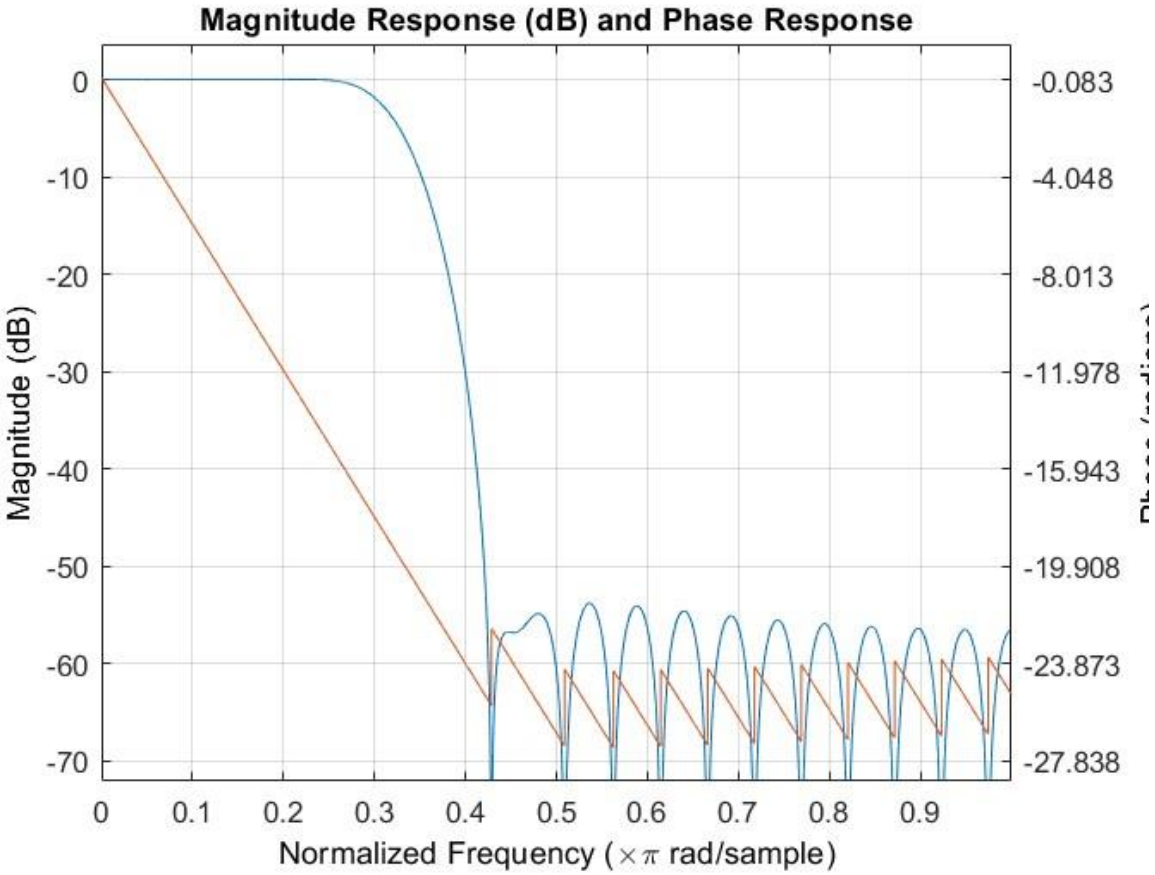
```
    end
    w_h = zeros(1,N);
    for k= 0:N-1
        w_h(k+1) = 0.54-0.46*cos(2*pi*k/(N-1));
    end
    h = h_d.*w_h;
    xi = convolution_sum(x,h);
end
function xi = convolution_sum(x,h)
    n = length(x);
    m = length(h);
    N = [x, zeros(1, m - 1)];
    M = [h, zeros(1, n - 1)];
    Y = zeros(1, m + n - 1);
    for i = 1:m + n - 1
        for j = 1:n
            if (i - j + 1 > 0 && i - j + 1 <= m)
                Y(i) = Y(i) + N(j) * M(i - j + 1);
            end
        end
    end
    xi = Y;
end
function down = down_sample(x,M)
    m = length(x);
    down = zeros(1,floor(m/M));
    for i=1:M:m
        down((i+M-1)/M)=x(i);
    end
end
function up = up_sample(x,L)
    n = length(x);
    up = zeros(1,n*L);
    for i = 1:n
        up((i-1)*L+1) = x(i);
    end
end
```

Fig:

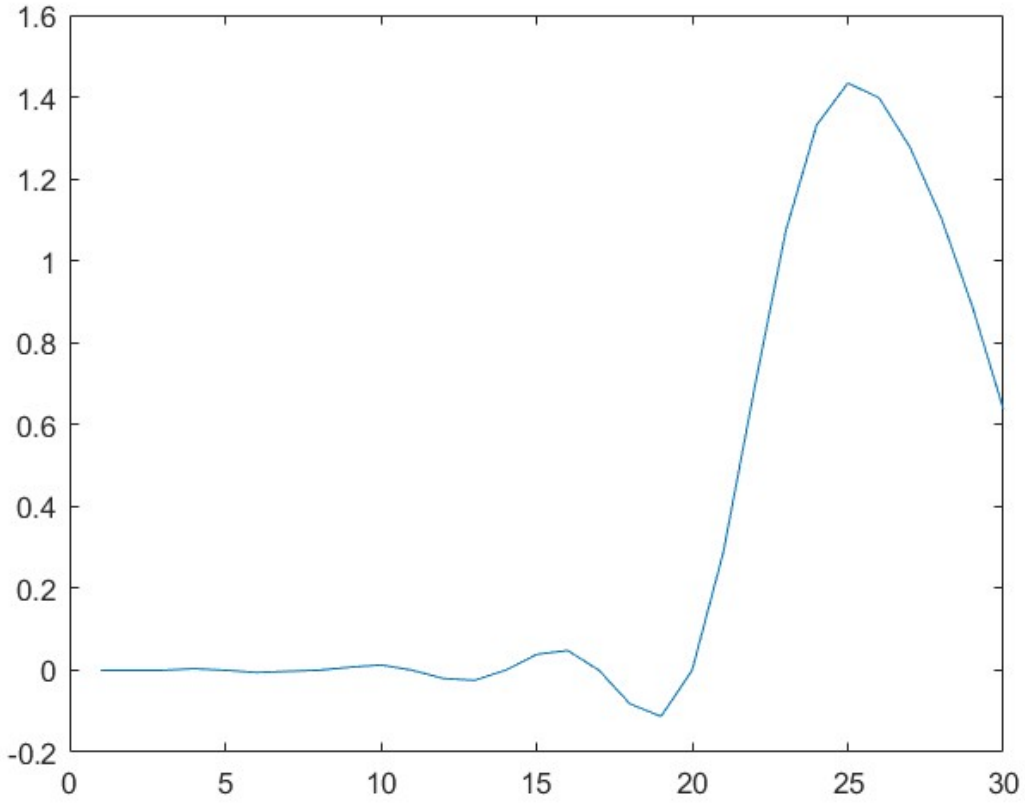
X_f(i) vs f

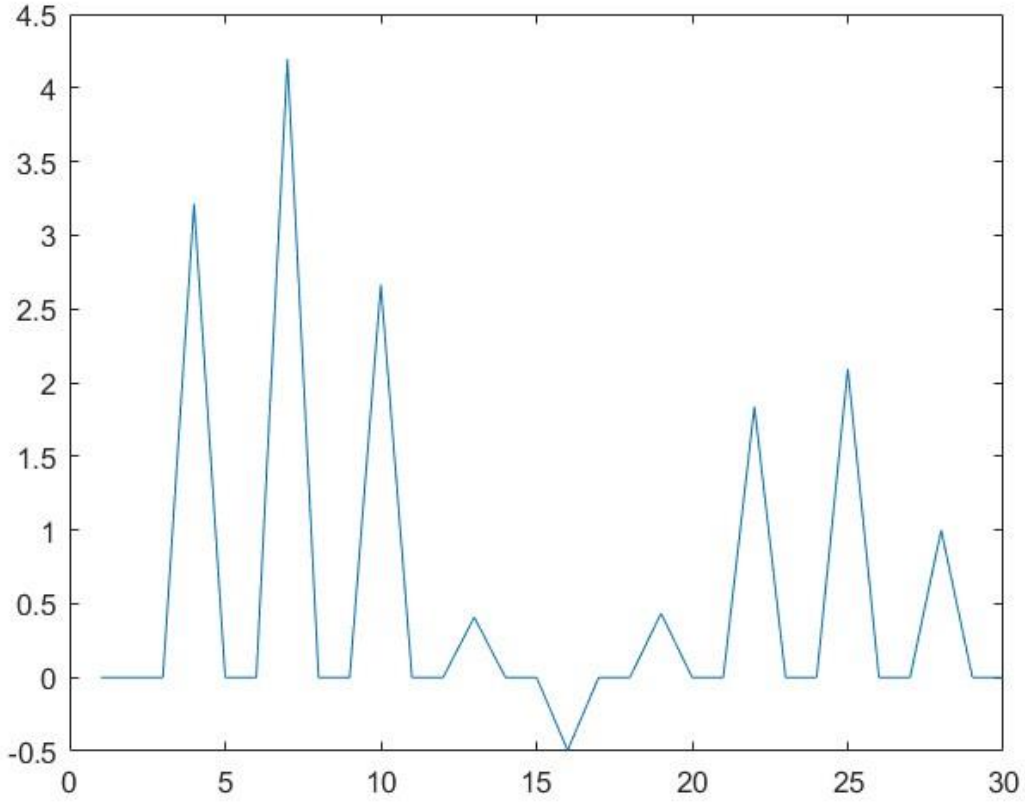


Impulse response of low pass filter:

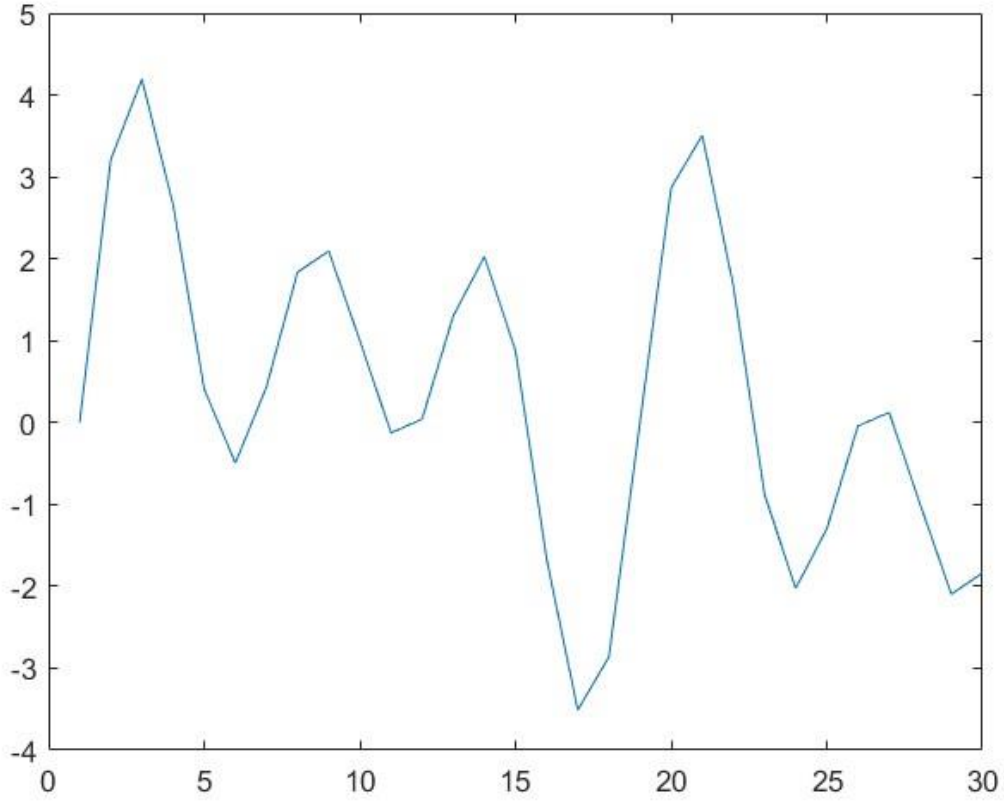

Magnitude Response (dB) and Phase Response
Magnitude (dB)
0
-10
-20
-30
-40
-50
-60
-70
-0.083
-4.048
-8.013
-11.978
-15.943
-19.908
-23.873
-27.838
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Normalized Frequency (×π rad/sample)

x_i vs n


1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
5
10
15
20
25
30


4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
0
5
10
15
20
25
30

x_d vs n


5
4
3
2
1
0
-1
-2
-3
-4
0
5
10
15
20
25
30