Problem 1:

(b)

(i) Write a MATLAB function implementing a single echo filter that takes the vector x of input values as well as the parameters and R as input arguments and that returns the output vector y.

Code:

function[y] = non\_recursive\_FIR(x,alpha, N, R)

y = x;

for n = 0:length(y)

for k = 1:N

shift = n-(k\*R);

if shift > 0

y(n) = y(n) + (alpha.^k)\*x(n-R\*k);

end

end

end

End

(ii) Set alpha= 1/2 and R = 3. Test your function using a discrete dirac pulse dirac[n] as

input. Use MATLAB to calculate the frequency response as the DFT of the impulse response h[n] and plot its magnitude and phase between 0 Hz and fs. Additionally choose alpha = 1 and explain the observed phase response in this case.

Code:

x = 1; %dirac

x = [x zeros(1, 5000)]; %zero padding

fs = 16000; %sampling frequency

dt = 1/fs; %Ts time between samples

t = 0:dt:(length(x)\*dt)-dt; %time axis

subplot(2,2,1)

stem(t,x); xlabel('Seconds'); ylabel('Amplitude');title('Input X');

xlim([0, 0.0031]); %limiting for better plot

h\_n = my\_FIR(x, 0.5, 3); %apha)1/2, R=3

subplot(2,2,2)

stem(t,h\_n); xlabel('Seconds'); ylabel('Amplitude');title('Output Y');

xlim([0, 0.0031]);

H\_z = fft(h\_n);

L = length(H\_z);

phase = angle(H\_z);

mag = abs(H\_z); % half is essential, rest is aliasing

fq =((0:L-1)/L)\*fs; % frequencies

subplot(2,2,3)

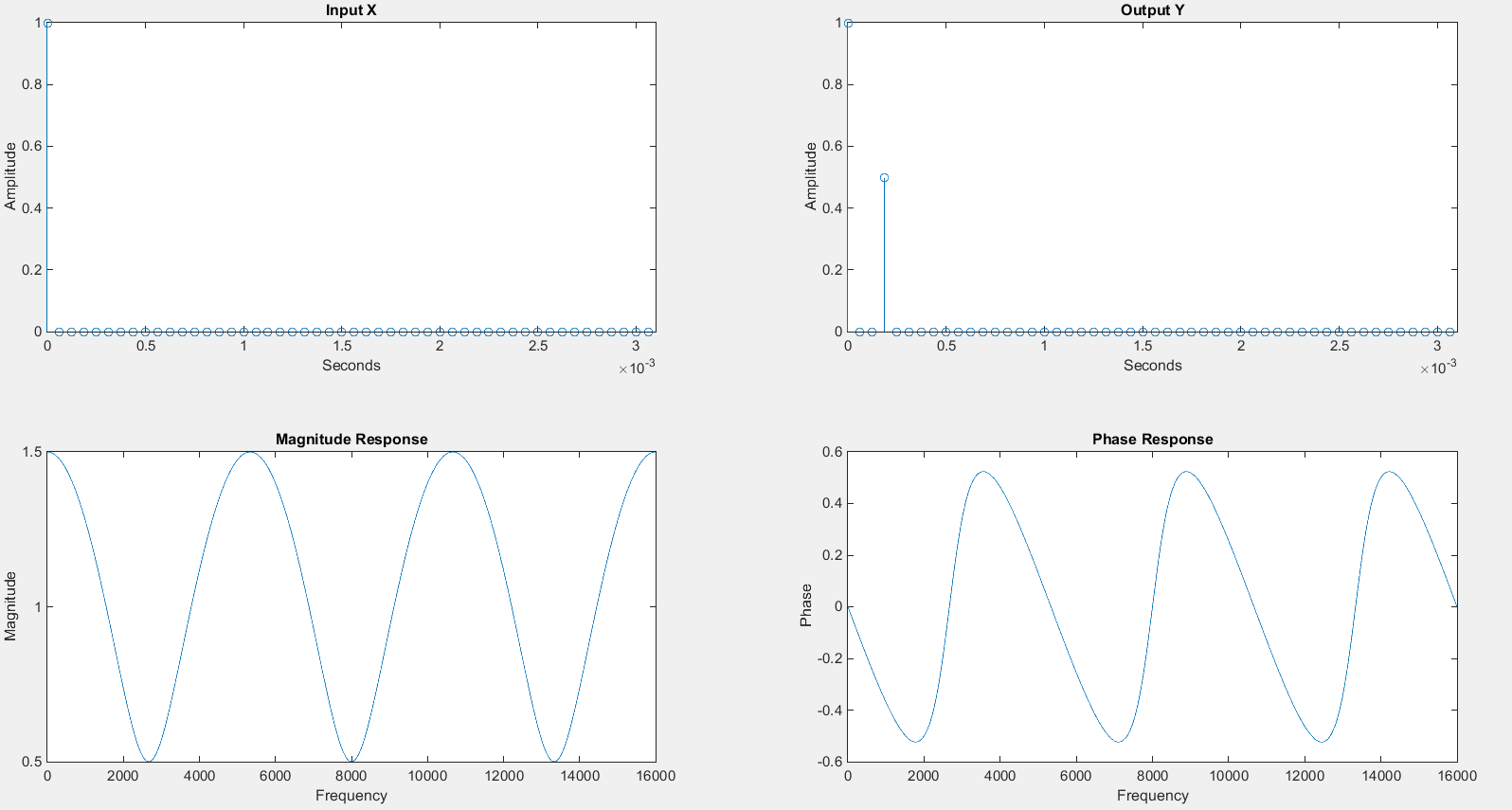
plot(fq,mag);xlabel('Frequency');ylabel('Magnitude');title('Magnitude Response');

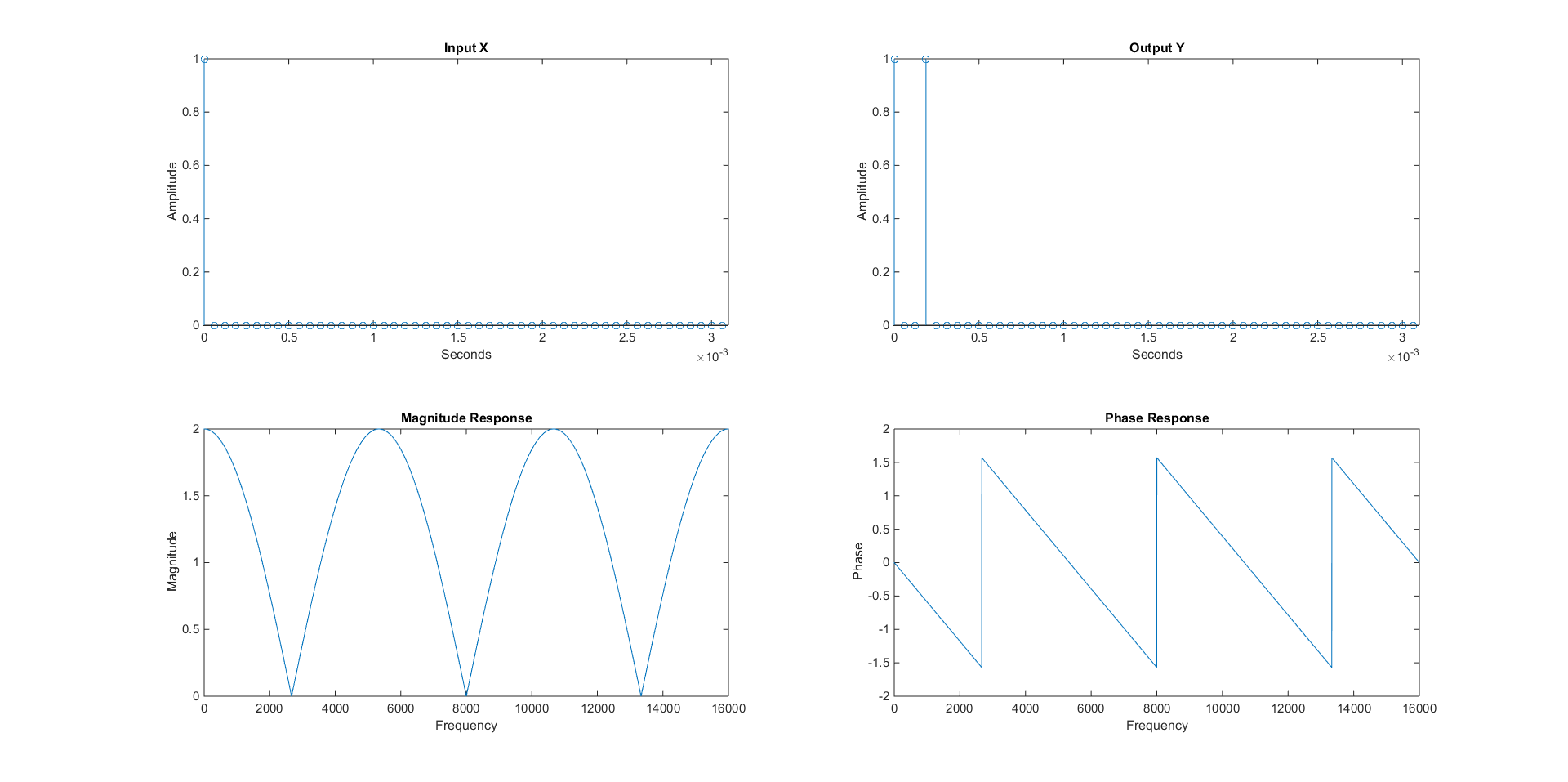
subplot(2,2,4)

plot(fq,phase);xlabel('Frequency'); ylabel('Phase');title('Phase Response');

Output:

Alpha = ½, R = 3;

Alpha = 1, R = 3;



Changing alpha affects both the magnitude and the phase response. With alpha = 1, the phase response is piece-wise linear.

(iii) Apply the echo filter to a sound file (in this case we used best\_weapon.wav). Choose R so that Td =100ms as a starting point. Vary alpha and R and observe the respective effects.

Td = R Ts

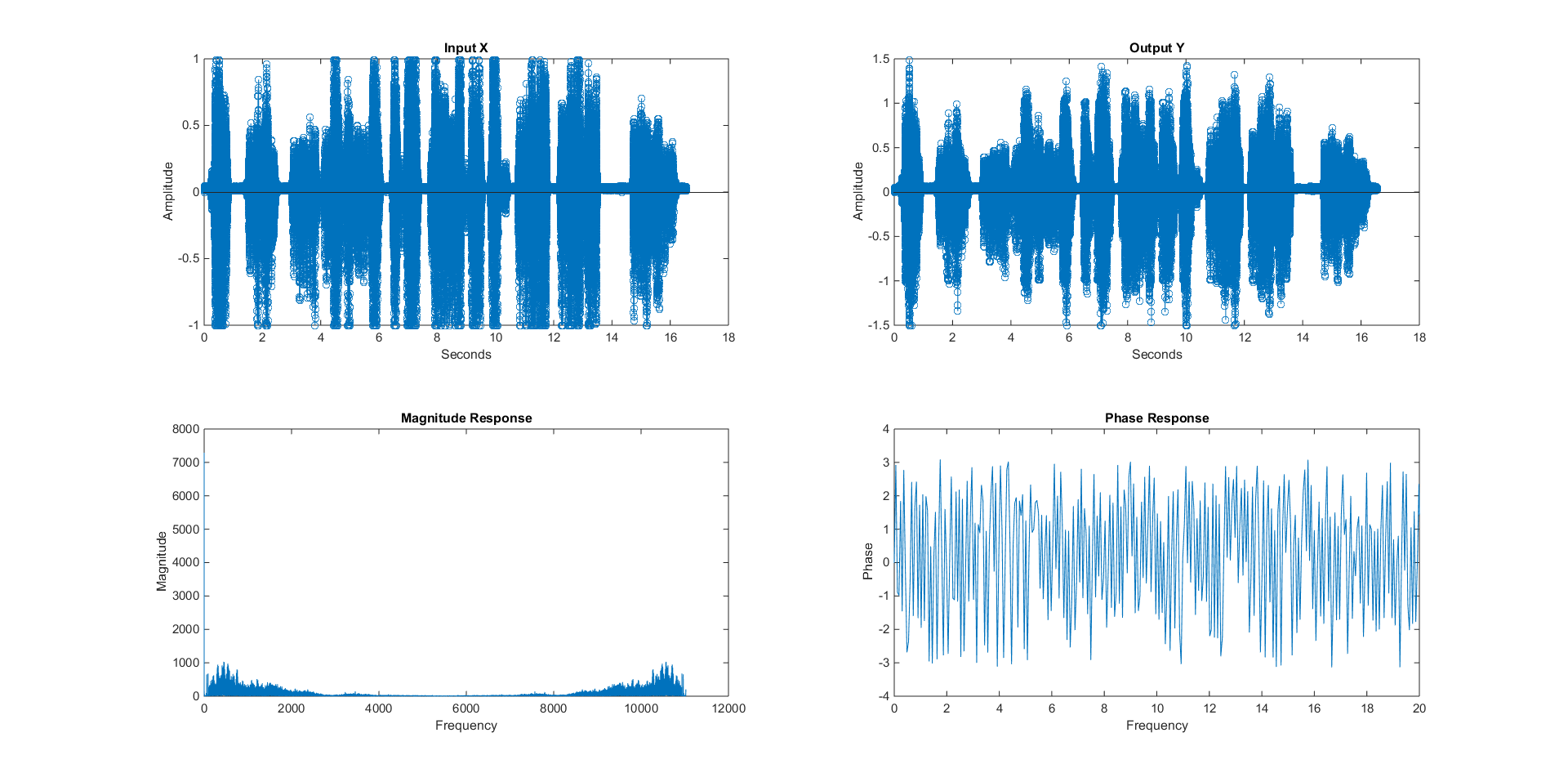
With fs = 11025 (from the sound file)

Therefore Ts = 1/fs = 9.07 e-5

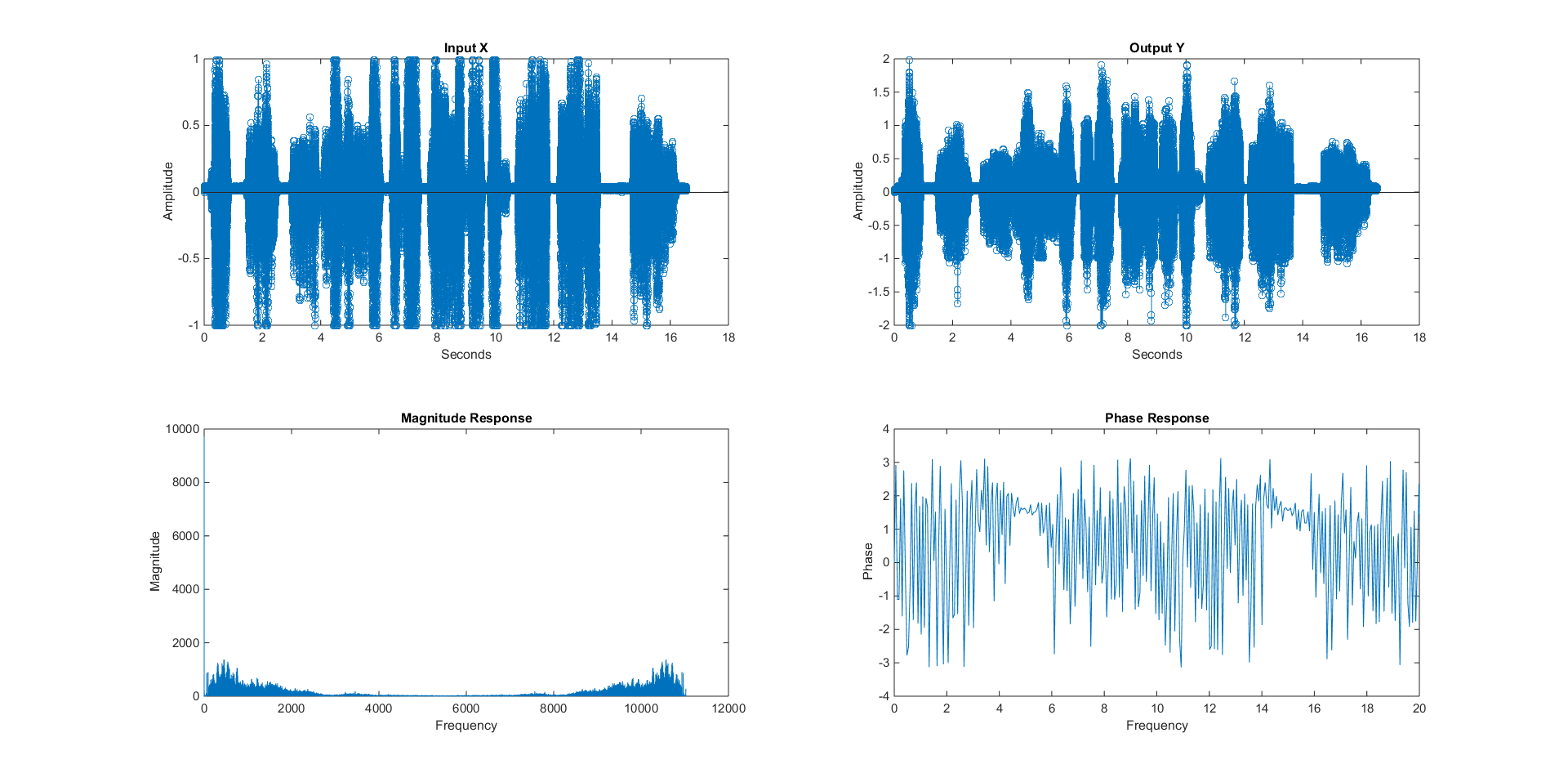
R = Td/Ts = 100ms/9.07 e-5 = 1103

Changing alpha:

R = 1103; alpha = 1/2

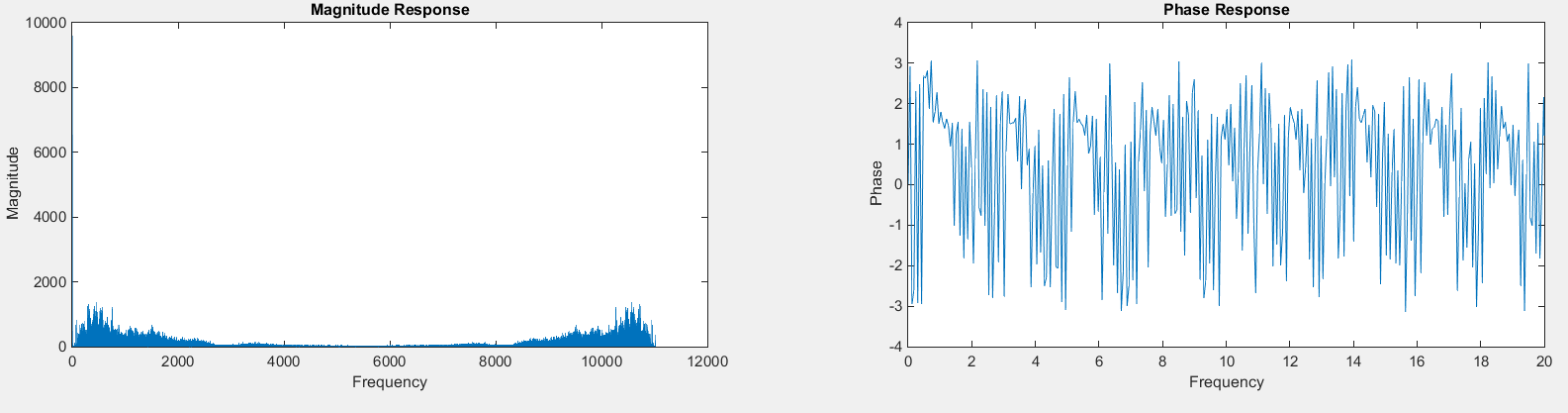


R = 1103; alpha = 1

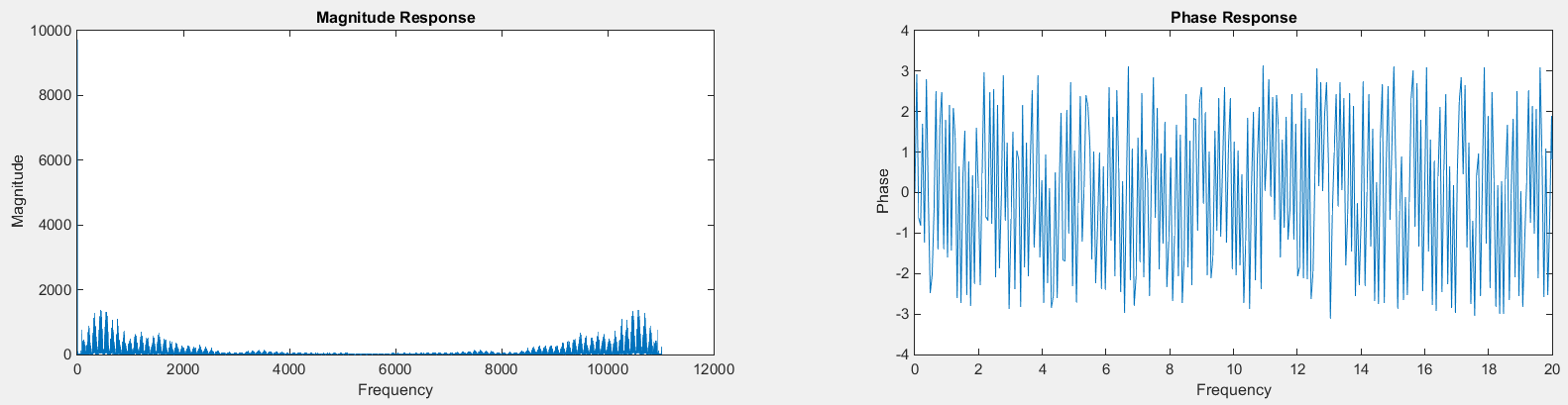


Changing R:

R=5000; alpha = 1;



R=100; alpha = 1;



Changing R affects both the magnitude as well as the phase response.

Problem (2):

(b) MATLAB implementation:

(i) Implement both the recursive and non-recursive filters for N = 6 and verify

that they yield the same impulse response.

Code: Non-recursive:

function[y] = non\_recursive\_FIR(x,alpha, N, R)

% x = [x zeros(1,R)];

y = x;

for n = 0:length(y)

for k = 1:N

shift = n-(k\*R);

if shift > 0

y(n) = y(n) + (alpha.^k)\*x(n-R\*k);

end

end

end

Code: recursive:

function [y] = recursive(x ,alpha,N, R)

y = x;

for n = 1:length(x)

value\_x = 0;

value\_y = 0;

shift\_y = n - R;

shift\_x = n - (R\*(N+1));

if shift\_y > 0

value\_y = y(n - R);

end

if shift\_x > 0

value\_x = x(n - (R\*(N+1)));

end

y(n) = x(n) - (alpha ^ (N+1) \* value\_x)+ (alpha \* value\_y);

end

end

(ii) • Use MATLAB to calculate the DFT of the impulse response and plot the amplitude

and phase responses (Hint: If you do this correctly, you should be able to see why this type of filter is also called comb filter).

• Use MATLAB to create arrays representing the coefficients of the recursive filter and plot the pole/zero map using MATLAB’s function zplane contained in the Signal Processing toolbox. Is the filter stable for alpha > 1?

General Test code for part 2:

%x = 1; %dirac

%x = [x zeros(1, 5000)]; %zero padding for dirac

%fs = 16000; %sampling frequency

[x,fs] = audioread('best\_weapon.wav');

dt = 1/fs; %Ts time between samples

t = 0:dt:(length(x)\*dt)-dt; %time axis

subplot(2,2,1)

%stem(t,x); xlabel('Seconds'); ylabel('Amplitude');

plot(t,x);

xlabel('Seconds'); ylabel('Amplitude'); title('Input X');

%xlim([0, 0.0031]);

h\_n = non\_recursive\_FIR(x,10,6, 5000); %nonrecursive

%h\_n = recursive(x,10,6, 5000); %recursive

subplot(2,2,2)

%stem(t,h\_n); xlabel('Seconds'); ylabel('Amplitude');

plot(t,h\_n);

xlabel('Seconds'); ylabel('Amplitude');title('Output Y');

%xlim([0, 0.0031]);

H\_z = fft(h\_n);

L = length(H\_z);

phase = angle(H\_z);

mag = abs(H\_z);

fq =((0:L-1)/L)\*fs; % frequencies

subplot(2,2,3)

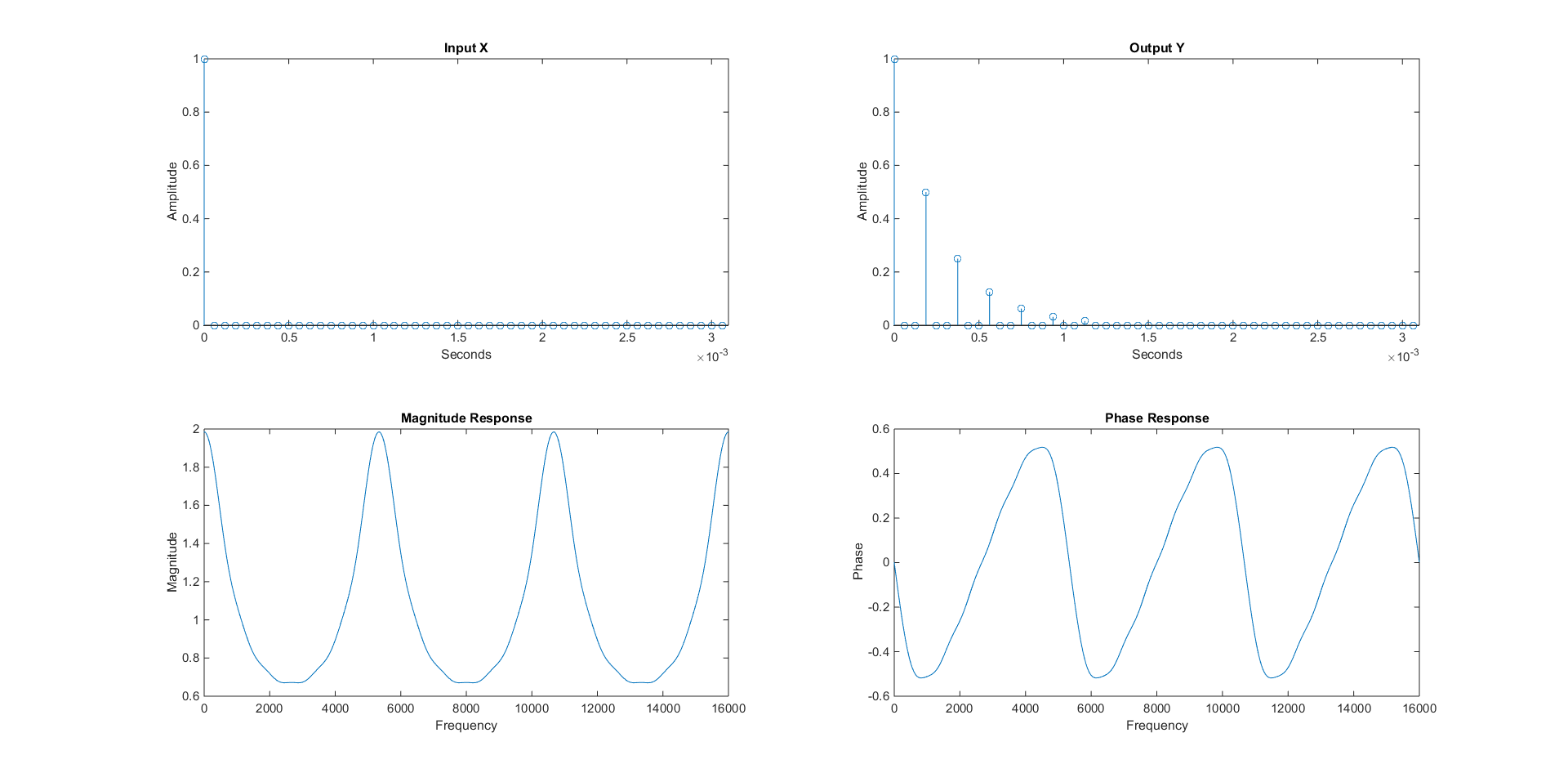
plot(fq,mag);xlabel('Frequency');ylabel('Magnitude');title('Magnitude Response');

subplot(2,2,4)

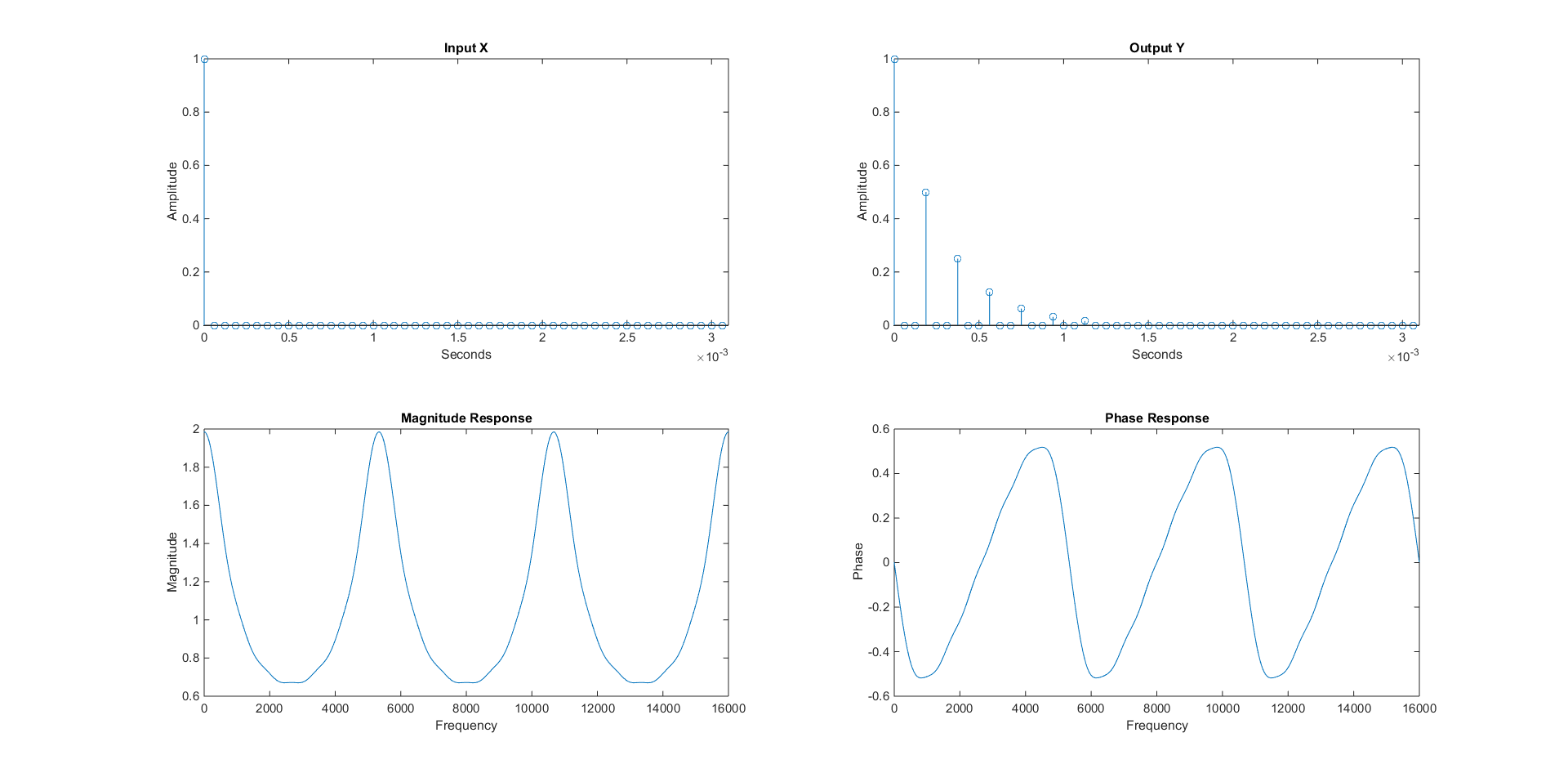
plot(fq,phase);xlabel('Frequency'); ylabel('Phase');title('Phase Response');

xlim([0, 5]);

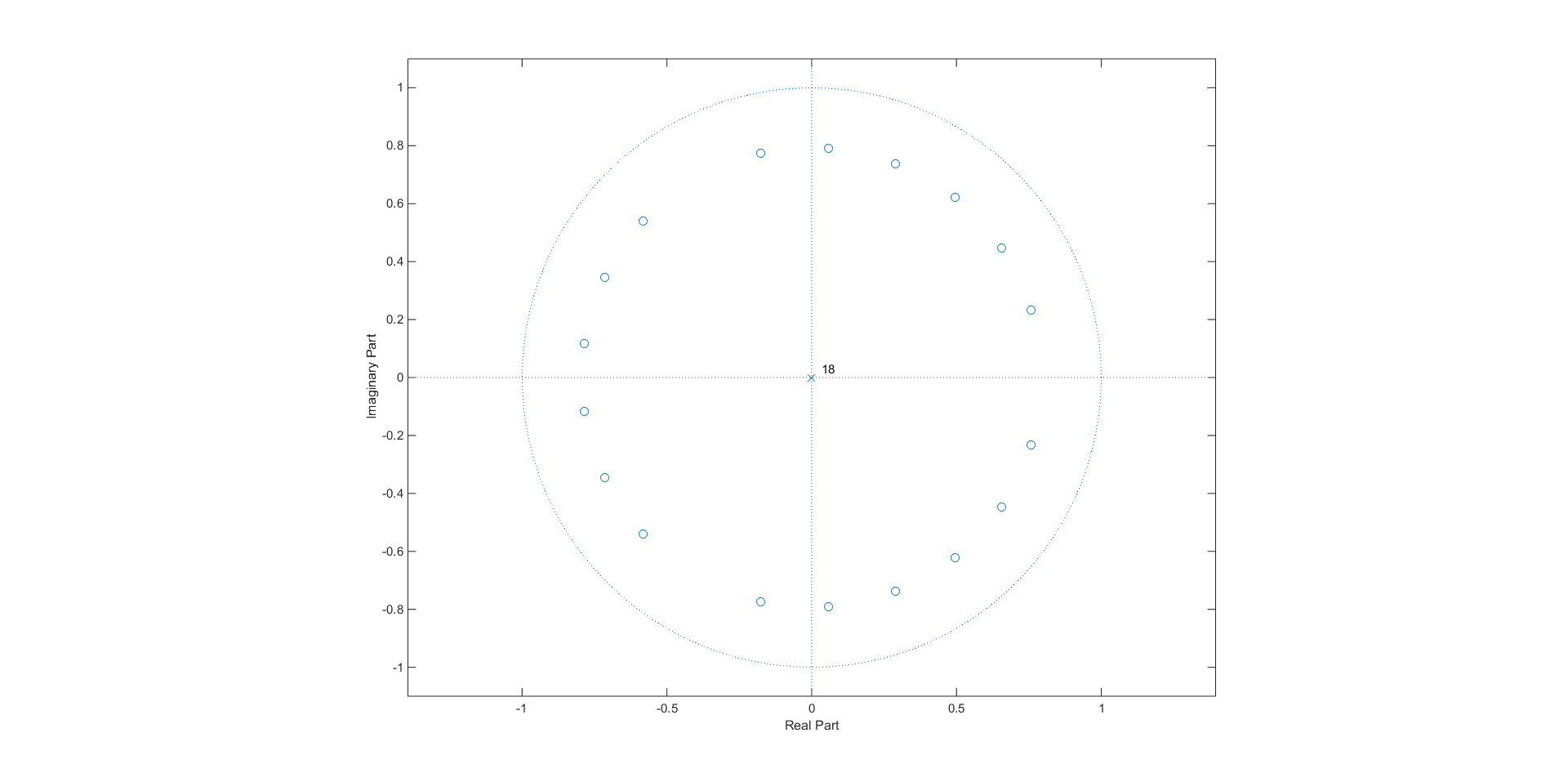
Graph from non-recursive function:



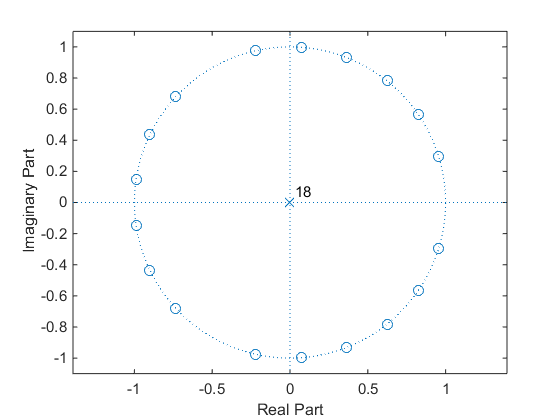
Graph from recursive function:



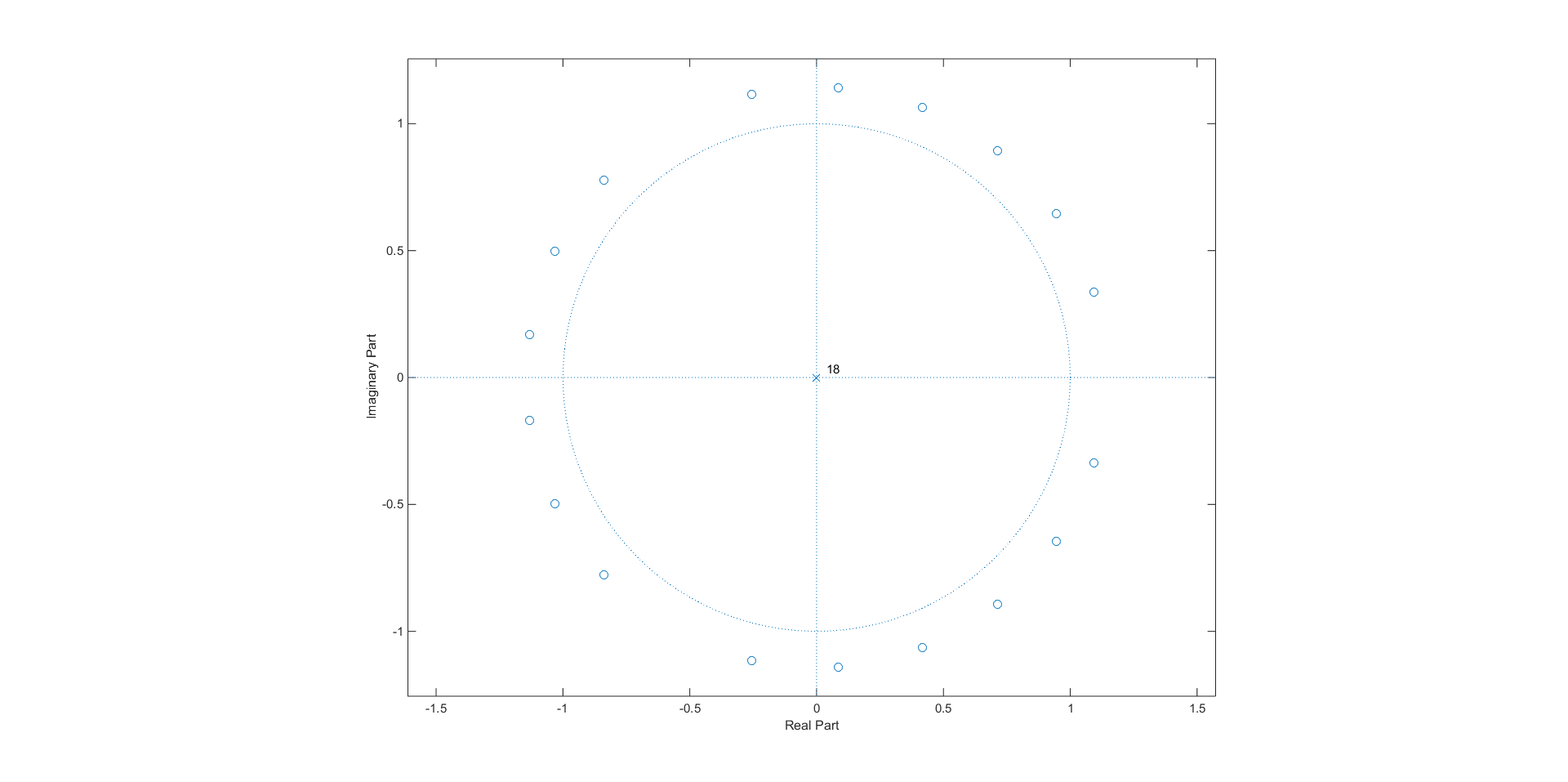
Zplane for alpha = 0.5. System is stable since all the poles are inside the unit circle.



Zplane for alpha = 1;



Zplane for alpha = 1.5; System is unstable because the poles are outside of the unit circle

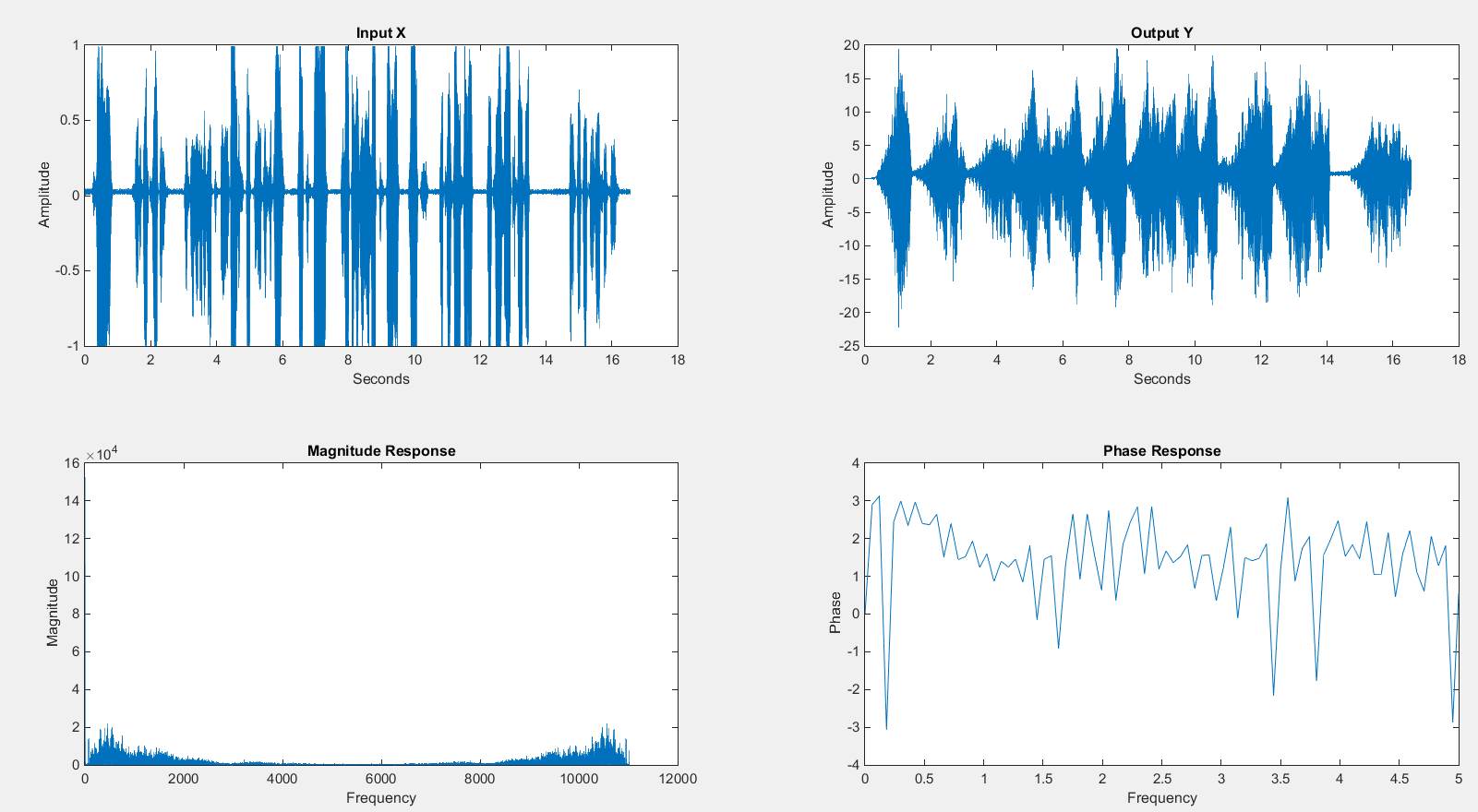


(iii) Apply both the recursive and non-recursive filters to a sound file. Choose R so that Td 100ms as a starting point. Vary alpha and R, particularly testing the

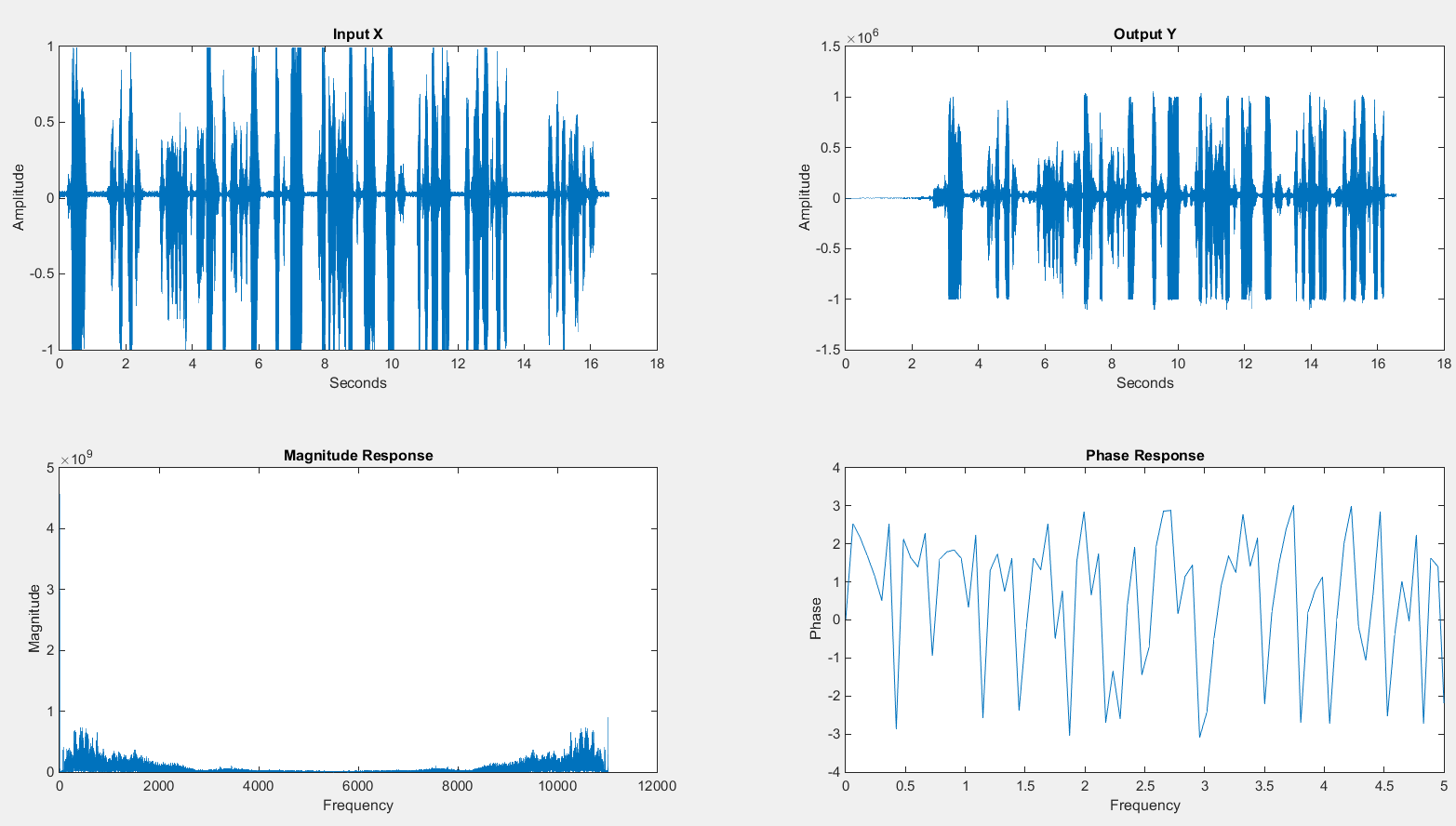
behaviour for alpha > 1. Explain the different behaviours of the recursive and non-recursive implementations in this case.

Non-recursive:

R = 1103; alpha = 1.5

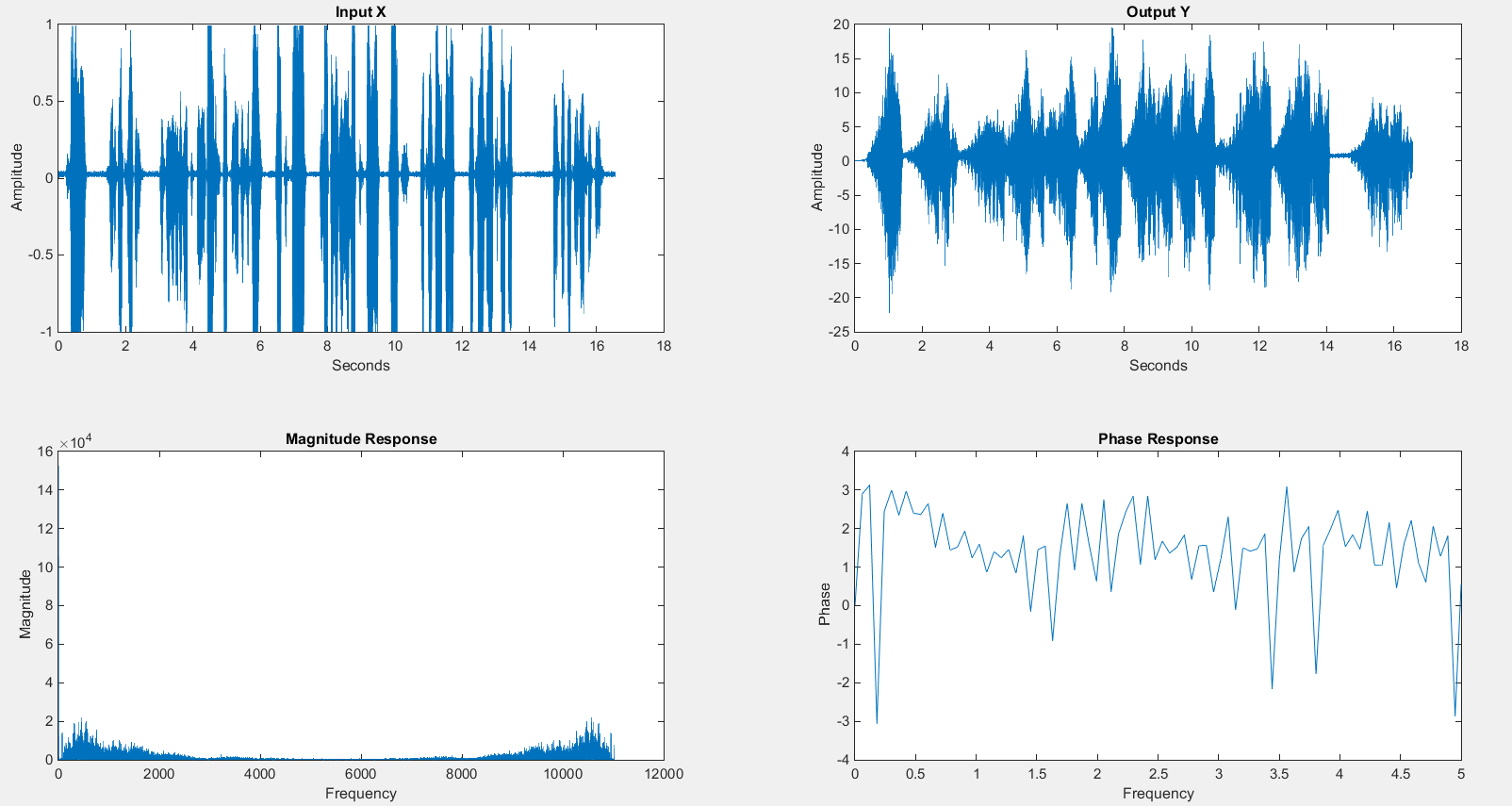


R = 5000; alpha = 10;

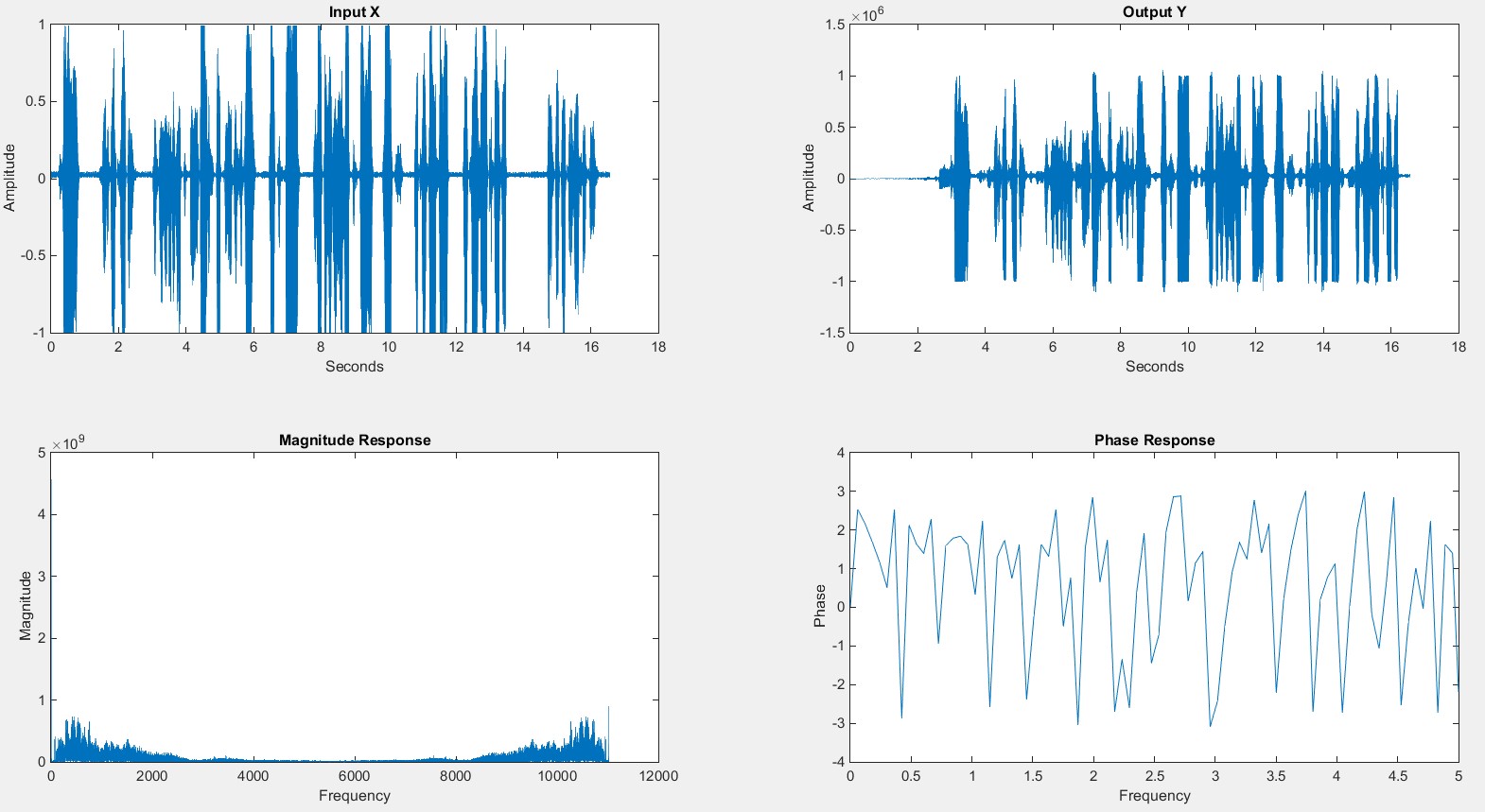


Recursive:

R = 1103; alpha = 1.5



R = 5000; alpha 10;



Recursive and non-recursive filter approach give the same output for same values of coefficients. Increasing R increases the delay of the filter resulting in echo, and decreasing alpha < 1 decreases the echo exponentially.