Convolution, Sampling and Aliasing in MATLAB

ECE 340 Lab D51

Riley Dixon - 1425251

Lab performed: October 27th, 2017

**Analytic Solutions:**

Lab2A Answers:

See the attached page A1 in the back for hand written calculations for the convolution of x[k]\*h[k] to verify that MATLAB's conv function is correct for this input & response signals. The calculations do match up with the graph that MATLAB produced after convolving the input & response signals.

----------------------------------------------Lab2A-----------------------------------------

clear;

clc;

%Setup variables.

xk = 4;

hk = 3;

x = [0:xk];

h = 2 - [0:hk];

%Plot x[k].

subplot(2,2,1);

stem([0:length(x)-1],x);

title('x[k] response');

ylabel('x[k]');

xlabel('k');

%Plot h[k].

subplot(2,2,2);

stem([0:length(h)-1],h);

title('h[k] response');

ylabel('h[k]');

xlabel('k');

y = conv(x, h);

%Plot y[k].

subplot(2,2,3);

stem([0:length(y)-1],y);

title('y[k] response');

ylabel('y[k]');

xlabel('k');

----------------------------------------------Lab2A End-----------------------------------------

Lab1B Answers:

The balia\_filtered.wav is the song after the original balia.wav song is run through the filter. The resulting song sounds terrible and should never be played again. The quality of the filtered song was significantly reduced, however the song could still be identified.

----------------------------------------------Lab2B-----------------------------------------

clear;

clc;

%Setup variables.

k = [0:50];

hk = (0.3\*sin(0.3\*(k - 25)\*pi)./(k\*pi)) .\* (0.54 - 0.46\*cos(2\*pi\*k/50));

%Plot h[k].

stem(hk);

title('h[k] response');

ylabel('h[k]');

xlabel('k');

%Load audiofile and convolve it with h[k].

balia = audioread('baila.wav');

balia\_filtered = conv(hk, balia);

%Output the resulting audiofile.

audiowrite('balia\_filtered.wav', balia\_filtered, 44100);

----------------------------------------------Lab2B End-----------------------------------------

Lab2C Answers:

Y1[k] and y2[k] appear to model the same function even though they are modelled against two different continuous time signals x1(t) and x2(t). The signals looking identical is to be expected if for θ1 = θ2 + 2\*pi\*n for some integer n. In this case n = -100.

Z1[k] is sampled at a greater rate however it still matches up with y1[k] perfectly and can be used to reconstruct x1(t). Y2[k] however is sampled too infrequently which is evident when it is superimposed onto a graph alongside z2[k]. Z2[k] shows the accurate representation on what the original x2(t) signal is. Y2[k] however shows a much slower oscillating wave, even though all of y2[k]'s points can be mapped to a point on x2(t) and z2[k].

----------------------------------------------Lab2C-----------------------------------------

clear;

clc;

%Setup variables.

T1 = 1/100; %Sampling frequency

n1 = [0:30];

t1 = n1 .\* T1;

y1k = cos(20\*pi\*t1);

y2k = cos(180\*pi\*t1);

%Plot y1[k].

subplot(2,1,1);

stem(y1k);

title('cos(20\*pi\*t) sampled at 100Hz');

ylabel('y1[k]');

xlabel('k');

%Plot y2[k].

subplot(2,1,2);

stem(y2k);

title('cos(180\*pi\*t) sampeld at 100Hz');

ylabel('y2[k]');

xlabel('k');

disp('Press any key to continue.');

pause();

close();

%2nd part of Lab2C

%Setup variables

T2 = 1/1000;

n2 = [0:300];

t2 = n2 .\* T2;

z1k = cos(20\*pi\*t2);

z2k = cos(180\*pi\*t2);

%Code copied and modified out of the lab manual.

subplot(2,1,1);

plot(n2\*T2,z1k,'r-', n1\*T1,y1k,'b+');

xlabel('n');

ylabel('y\_1[n] and z\_1[n]');

legend('z\_1[n]','y\_1[n]');

subplot(2,1,2);

plot(n2\*T2,z2k,'r-', n1\*T1,y2k,'b+');

xlabel('n');

ylabel('y\_2[n] and z\_2[n]');

legend('z\_2[n]','y\_2[n]');

disp('Press any key to continue.');

pause();

close();

%Close the figure to setup for the 2nd part

%3rd part of Lab2C

%Let omega = 220, thus n = -1.

%Trick: Add or subtract by a multiple of 200 from 20.

y3k = cos(220\*pi\*t1);

%Plot the original y1[k].

subplot(2,1,1);

stem(y1k);

title('cos(20\*pi\*t) sampled at 100Hz');

ylabel('y1[k]');

xlabel('k');

%Plot the new y3[k].

subplot(2,1,2);

stem(y3k);

title('cos(220\*pi\*t) sampeld at 100Hz');

ylabel('y3[k]');

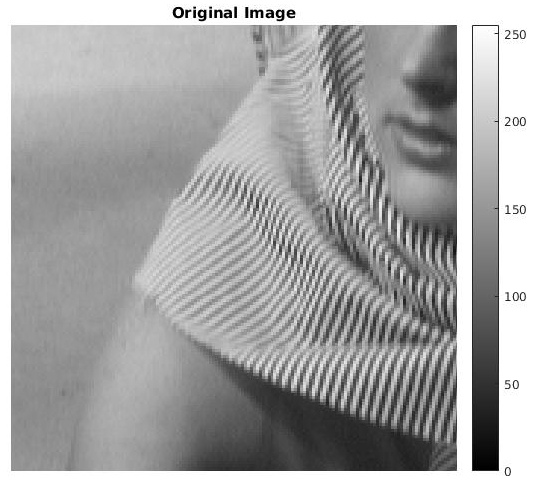
xlabel('k');

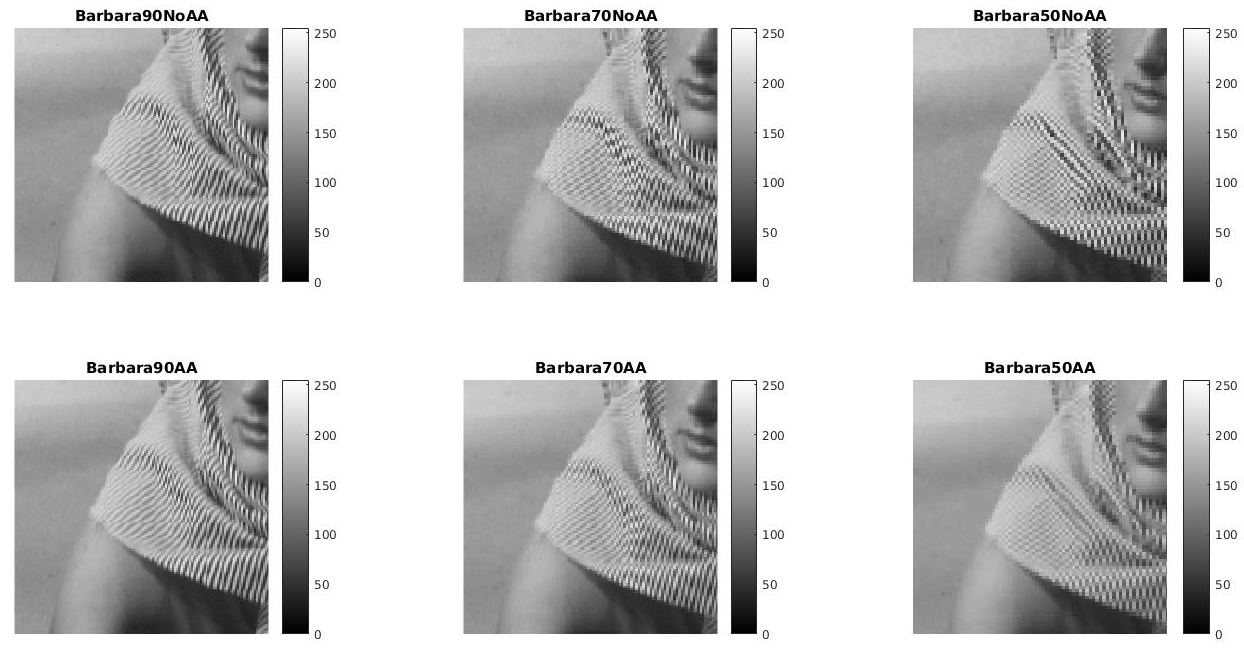
----------------------------------------------Lab2C End-----------------------------------------

Lab2D Answers:

Low frequency parts of the image are where there is not a large variation in colour in an area of the image. Conversely high frequency parts are where the colour varies a lot within a given area.

Antialiasing effectively blurs the high frequency parts of the image so that it doesn't appeear pixelated. There is information lost from the original image when resized, the antialiasing tries to limit the amount of information lost. Without antialiasing the image loses it's 'natural flow' that the original contained.

When the sample size is reduced the high frequency portions of the image are greatly affected as there is a large amount of variation between pixels which can lead to information loss. The computer has to choose which pixels to throw away and which to keepfor the new image. If you imagine a frequency graph the frequency cutoff gets smaller and smaller as the image is reduced further and further which will cutoff more and more of the original signal. In the form of this image, that means that the lines on Barbara's dress become checkered patterns.

****

----------------------------------------------Lab2D-----------------------------------------

%Load image.

barbaraLarge = imread('barbaraLarge.jpg');

resizeFactor = [0.9, 0.7, 0.5];

%Display the original image.

figure(1);

imshow(barbaraLarge), colorbar;

title('Original Image');

%Create the resized images with both aliasing and nonaliasing.

barbara90NoAA = imresize(barbaraLarge, resizeFactor(1), 'nearest',...

'Antialiasing', 0);

barbara70NoAA = imresize(barbaraLarge, resizeFactor(2), 'nearest',...

'Antialiasing', 0);

barbara50NoAA = imresize(barbaraLarge, resizeFactor(3), 'nearest',...

'Antialiasing', 0);

barbara90AA = imresize(barbaraLarge, resizeFactor(1), 'nearest',...

'Antialiasing', 1);

barbara70AA = imresize(barbaraLarge, resizeFactor(2), 'nearest',...

'Antialiasing', 1);

barbara50AA = imresize(barbaraLarge, resizeFactor(3), 'nearest',...

'Antialiasing',1);

%Display the images in a subplot to comapre

figure(2);

subplot(2,3,1);

imshow(barbara90NoAA), colorbar;

title('Barbara90NoAA');

subplot(2,3,2);

imshow(barbara70NoAA), colorbar;

title('Barbara70NoAA');

subplot(2,3,3);

imshow(barbara50NoAA), colorbar;

title('Barbara50NoAA');

subplot(2,3,4);

imshow(barbara90AA), colorbar;

title('Barbara90AA');

subplot(2,3,5);

imshow(barbara70AA), colorbar;

title('Barbara70AA');

subplot(2,3,6);

imshow(barbara50AA), colorbar;

title('Barbara50AA');

%Save the images

imwrite(barbara90NoAA, 'barbara90NoAA.jpg', 'jpg', 'Quality', 100);

imwrite(barbara70NoAA, 'barbara70NoAA.jpg', 'jpg', 'Quality', 100);

imwrite(barbara50NoAA, 'barbara50NoAA.jpg', 'jpg', 'Quality', 100);

imwrite(barbara90AA, 'barbara90AA.jpg', 'jpg', 'Quality', 100);

imwrite(barbara70AA, 'barbara70AA.jpg', 'jpg', 'Quality', 100);

imwrite(barbara50AA, 'barbara50AA.jpg', 'jpg', 'Quality', 100);

----------------------------------------------Lab2D End-----------------------------------------

Lab2Test3:

The low pass filter reduces the contrast of the original image before resizing in order to reduce the frequency of the higher frequency areas (the low frequency areas aren’t impacted as much as the higher frequency areas). Then when actually resizing the image the amount of aliasing or apparent choppiness in the image is reduced.

----------------------------------------------Lab2test3----------------------------------------

% EE338 Lab 2

% test3.m

clear

clc

close all

I=imread('barbaraLarge.jpg');

% Low pass filtering before downsampling

filt=fspecial('average',[3 3]); % creates a 3x3 low pass filter kernel

filt\_img=imfilter(I,filt,'conv'); % applies the lpf by convolving the image with the filter kernel

B\_LPF=imresize(filt\_img, 0.7 , 'nearest');

B=imresize(I, 0.7, 'nearest');

figure, imshow(I); title('Original Barbara Image');

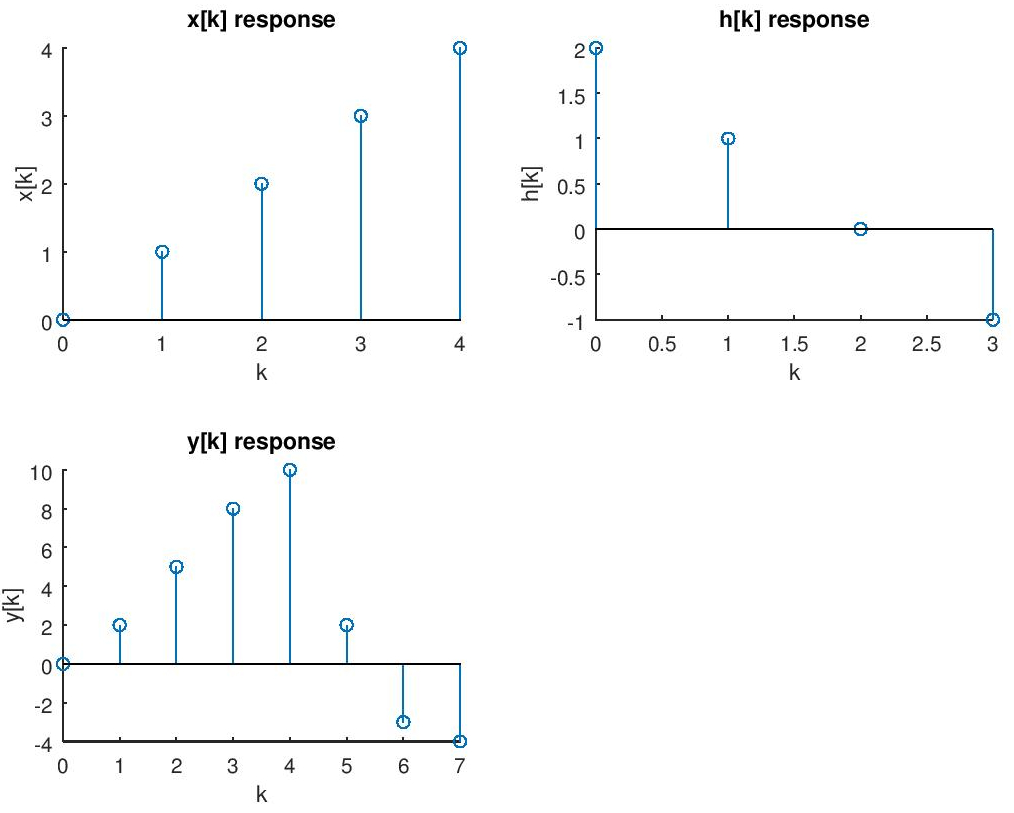
figure, imshow(B); title('Barbara Image resized by 70% of original size');

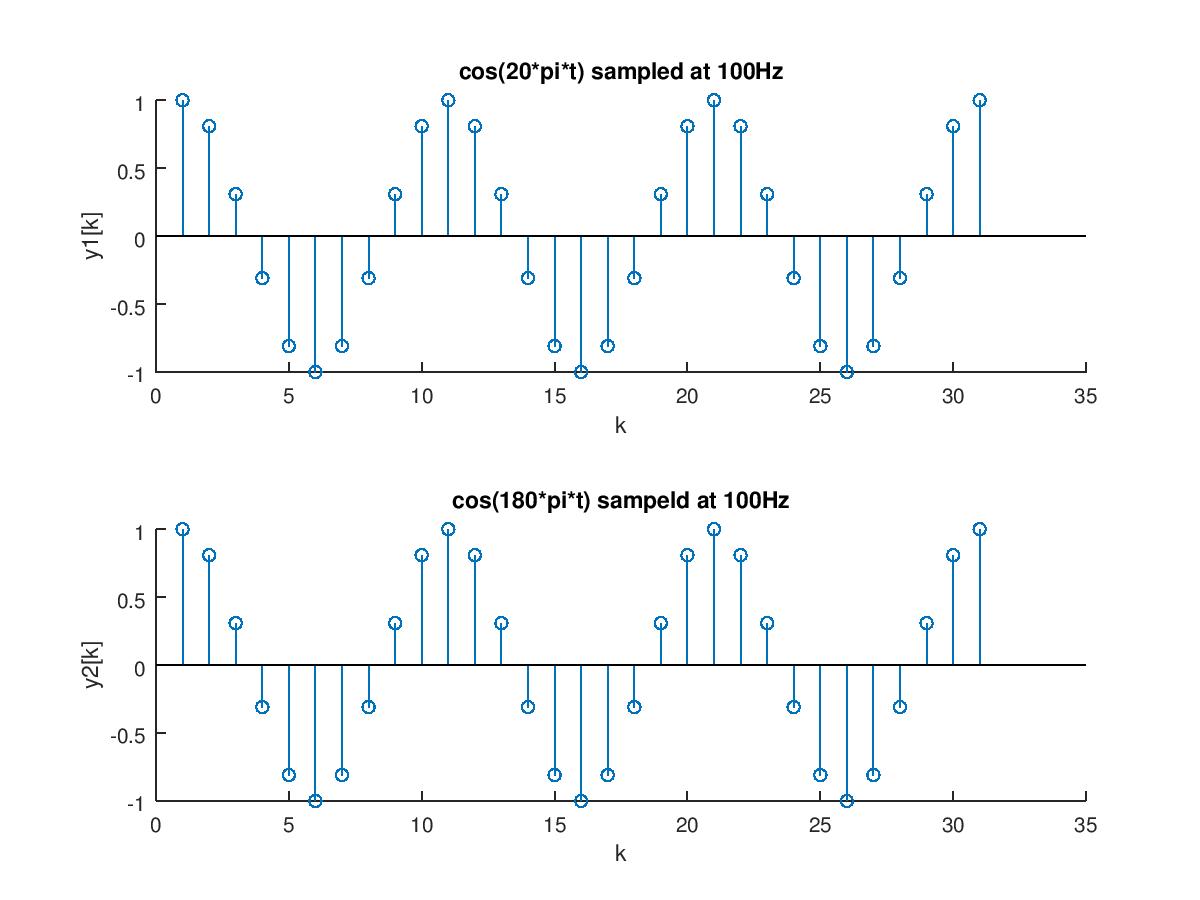
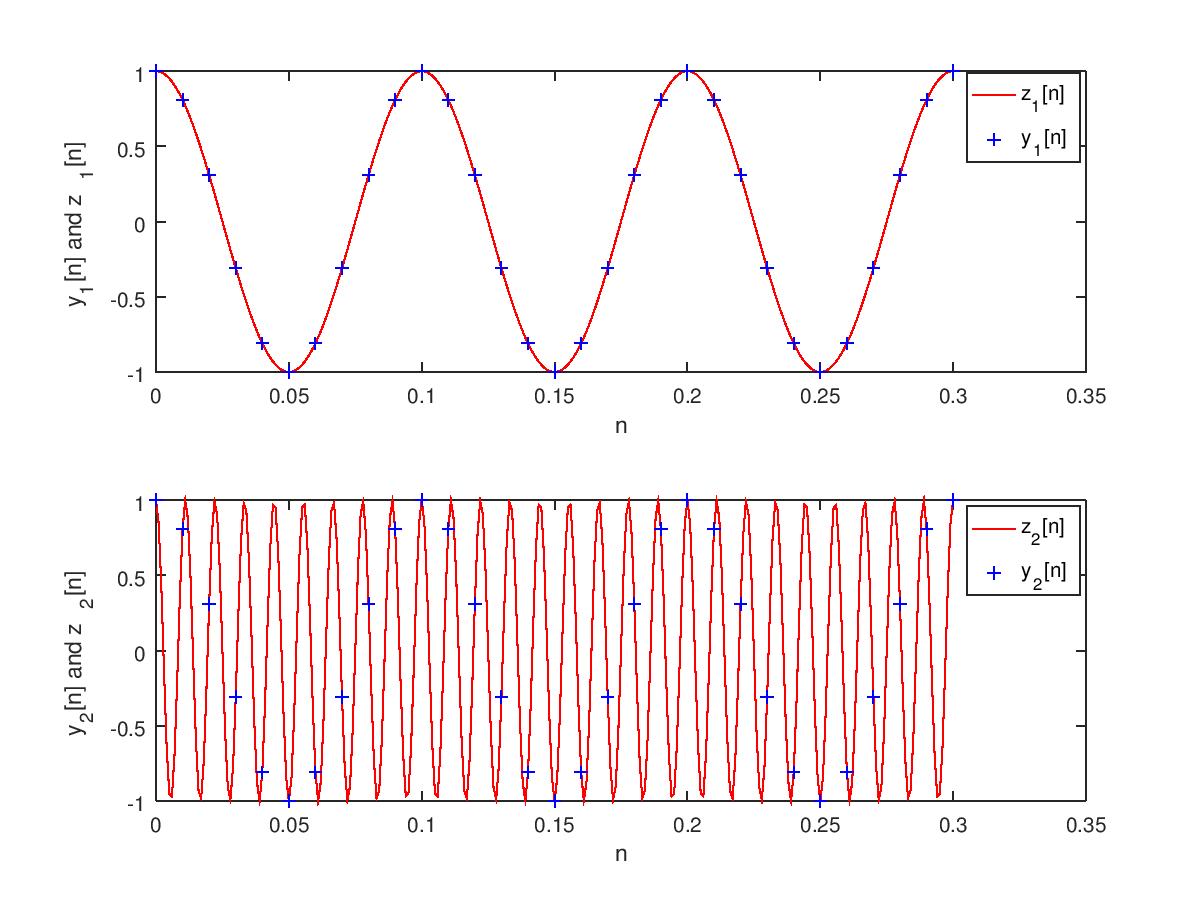
figure, imshow(B\_LPF); title('Low pass filter applied before Resizing to 70% of original size');

----------------------------------------------Lab2test3 End-----------------------------------------

**Plots**

Lab2A:

****

Lab2C:

