Lab 2

Convolution and Aliasing

Michael Kwok (1548454)

ECE 340 Discrete Time Signals and Systems
Department of Electrical and Computer Engineering
University of Alberta
12 October 2020

1 Convolution

1.1 Pure Signal Convolution

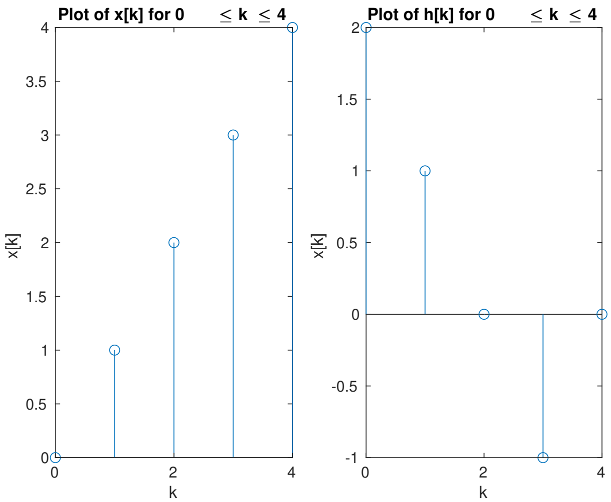


Figure 1

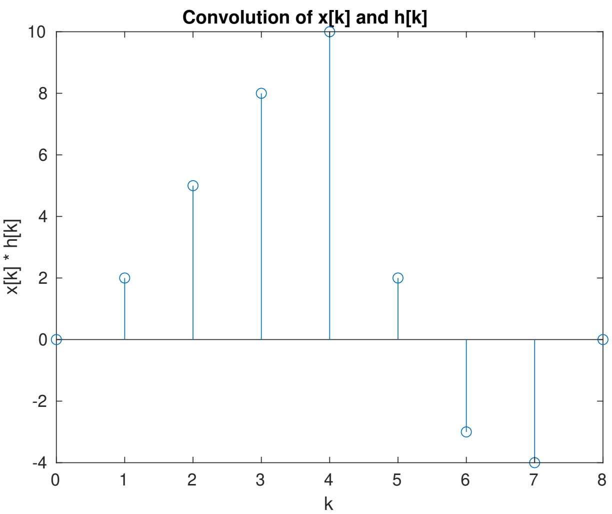


Figure 2

1.1.1 Convolution verification

$$\sum_{n=-\infty}^{\infty} h[n]x[k-n]$$

$$= h[0]x[k] + h[1]x[k-1] + h[2]x[k-2]$$
$$+ h[3]x[k-3] + h[4]x[k-4]$$

n	0	1	2	3	4	5	6	7	8
$2x(n)$	0	2	4	6	8	0	0	0	0
$x(n-1)$	0	0	1	2	3	4	0	0	0
$-x(n-3)$	0	0	0	0	-1	-2	-3	-4	0
$\sum$	0	2	5	8	10	2	-3	-4	0

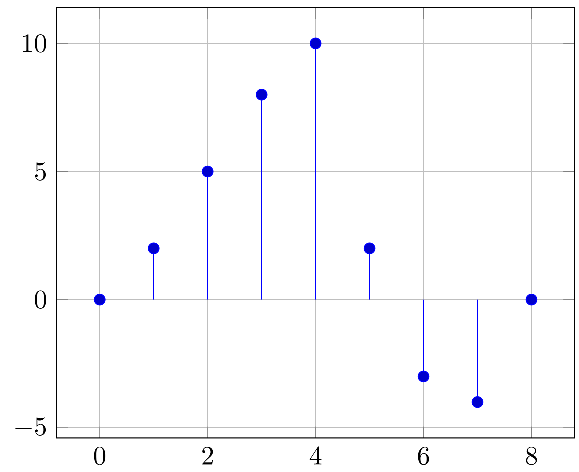


Figure 3: Result of manual convolution of $x[k]$ and $h[k]$

The two graphs match up, hence convolution verified.

1.2 Audio Convolution

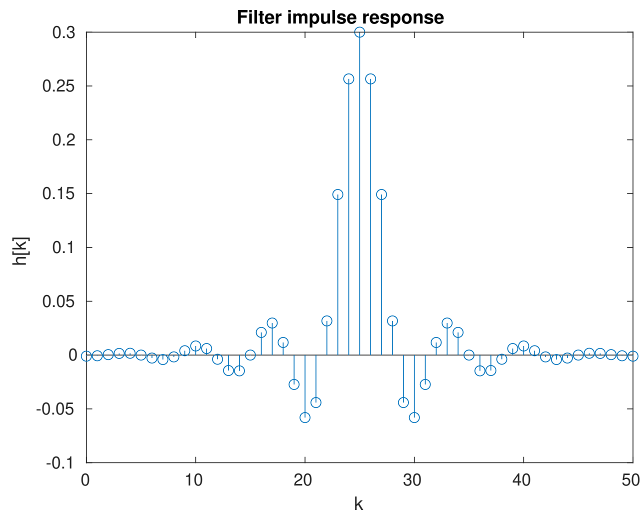


Figure 4

Audio sounds less clear, more muddy. This is due to the application of the filter. The resulting audio is around double the size, which means the sample rate for the saved file is about double too.

1.3 Code

```
% Set output domain
k1 = [0:4];

% Use lambda functions to make definition shorter
x1 = @(n) (n) .* ((0 <= n) & (n <= 4));
h1 = @(n) (2 - n) .* ((0 <= n) & (n <= 3));

% Use built in convolution function
y = conv(x1(k1), h1(k1));
```

2 Signal Aliasing

2.1 Pure Signal Aliasing

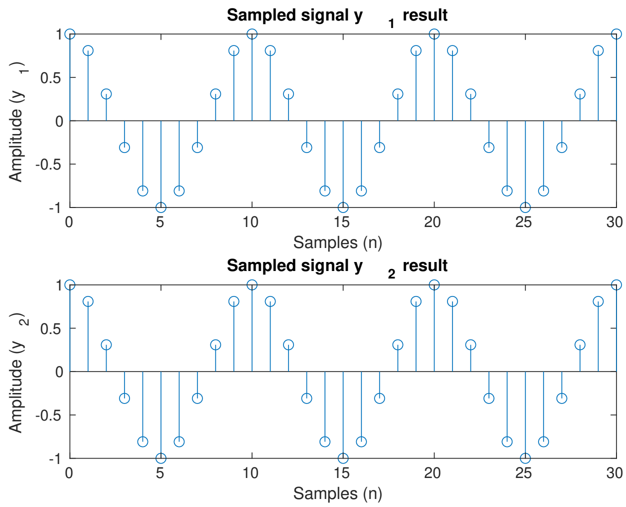


Figure 5: $\cos(20\pi t)$ (top) vs $\cos(180\pi t)$ (bottom) at 100 Hz

The plot of both signals look exactly the same. This is due to aliasing.

$$\begin{aligned} y_1[n] &= \cos\left(\frac{20\pi}{100}n\right) \\ &= \cos(0.2\pi n) \\ y_2[n] &= \cos(1.8\pi n) \\ &= \cos(2\pi n - 0.2\pi n) \\ &= \cos(-0.2\pi n) \end{aligned}$$

Using the fact that $\cos(-\theta) = \cos(\theta)$:

$$y_2[n] = \cos(-0.2\pi n) = \cos(0.2\pi n) = y_1[n]$$

Which shows that y_2, when sampled at the specified frequency of 100 Hz results in the same signal as y_1, which is the definition of aliasing.

Contrasted to the signals when they get sampled at 1000 Hz:

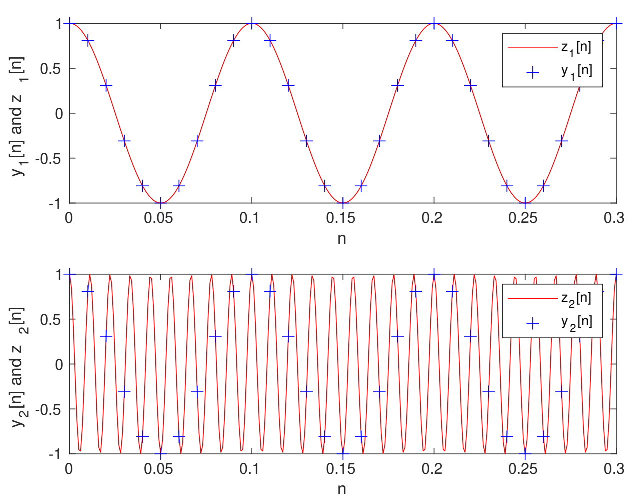


Figure 6: z_n are the respective y_n signals sampled faster

At 100 Hz, the higher frequency signal gets aliased as the lower frequency result, while at 1000 Hz, the higher frequency signal shows the more correct result.

Other coeffecients of t in the continuous cos can also be used to show aliasing. One of them is 380π as shown below:

$$\begin{aligned} y_1[n] &= \cos(0.2\pi n) \\ y_3[n] &= \cos(0.2\pi n + 2N\pi n) \end{aligned}$$

Substitute N with any integer such as -2 in this case:

$$\begin{aligned} y_3[n] &= \cos(0.2\pi n - 4\pi n) \\ &= \cos(3.8\pi n) \end{aligned}$$

Convert back into a continuous unsampled function with $n = F_s t$

$$\begin{aligned} x_3(t) &= \cos(3.8\pi \cdot 100 \cdot t) \\ &= \cos(380\pi t) \end{aligned}$$

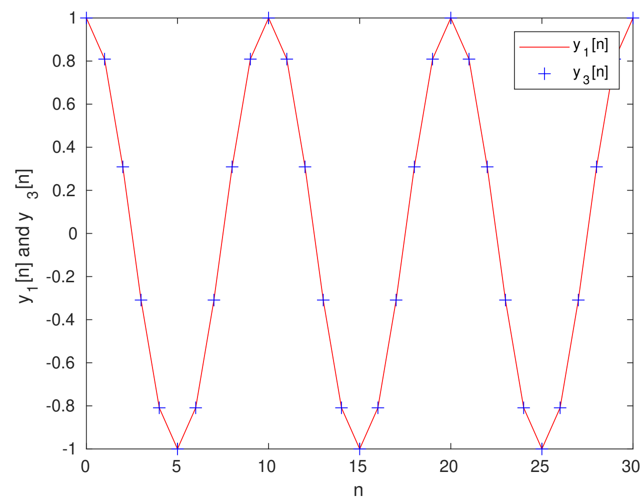


Figure 7: Another possible signal that can be aliased to y_1

2.1.1 Code

```
% Part a
n1 = [0:30]';
x1 = @(t) (cos(20 .* pi .* t));
x2 = @(t) (cos(180 .* pi .* t));

T1 = 1 / 100;

y1 = x1(T1 .* n1);
y2 = x2(T1 .* n1);

% Part b
n2 = [0:300];
T2 = 1 / 1000;
z1 = x1(T2 .* n2);
z2 = x2(T2 .* n2);

% Part c
x3 = @(t) (cos(380 .* pi .* t));
y3 = x3(T1 .* n1);
```

2.2 Image Aliasing

Figure 8: barbaraLarge.jpg with a brightness bar on the side

The floor behind the person stays unchanged despite all the downscaling, due to it being mostly “random noise” and not ordered lines. The tablecloth, pants and headscarf all have lines that due to the compression algorithm was aliased into different lines. The direction of the lines seem to be change in every image. Sharp edges such as the table leg’s line down also lose defintion as the image size is reduced. The frequency of which the image is sampled (“resolution”) is reduced as the image scale gets smaller, which explains all the above. The antialiased versions of every picture is smoother than the non-antialiasing versions, which could be due to application of a low pass filter, reducing the high frequency pixel data.

Downscaled images on next page.

2.2.1 Code

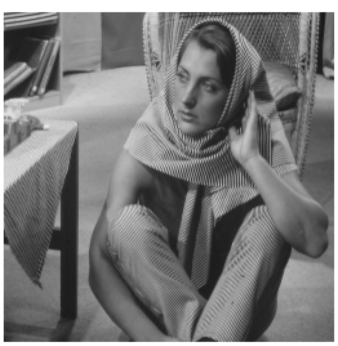
```
%% Question 4 Image Aliasing
% Part a
img = imread('barbaraLarge.jpg');

% Part c
img09aa = imresize(img, 0.9, 'nearest', 'antialiasing', 1);
img09noaa = imresize(img, 0.9, 'nearest', 'antialiasing', 0);

img07aa = imresize(img, 0.7, 'nearest', 'antialiasing', 1);
img07noaa = imresize(img, 0.7, 'nearest', 'antialiasing', 0);

img05aa = imresize(img, 0.5, 'nearest', 'antialiasing', 1);
img05noaa = imresize(img, 0.5, 'nearest', 'antialiasing', 0);
```

(a) barbaraLarge.jpg downscaled by 0.9 with antialiasing

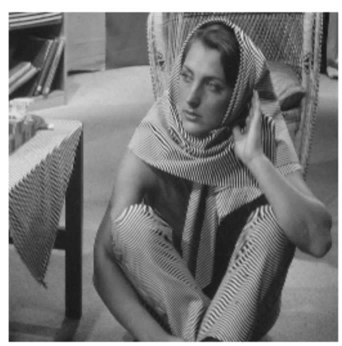
(b) barbaraLarge.jpg downscaled by 0.9 without antialiasing

(c) barbaraLarge.jpg downscaled by 0.7 with antialiasing

(d) barbaraLarge.jpg downscaled by 0.7 without antialiasing



(e) barbaraLarge.jpg downscaled by 0.5 with antialiasing



(f) barbaraLarge.jpg downscaled by 0.5 without antialiasing

Figure 9: All scaled figures

A Complete Code (lab2.m)

```
%% Question 1 Signal Convolution
k1 = [0:4];

% Part a
x1 = @(n) (n) .* ((0 <= n) & (n <= 4));
h1 = @(n) (2 - n) .* ((0 <= n) & (n <= 3));

% Part b
figure;
subplot(1, 2, 1);
stem(k1, x1(k1));
title('Plot of x[k] for 0\leq k\leq4');
xlabel('k');
ylabel('x[k]');

subplot(1, 2, 2);
stem(k1, h1(k1));
title('Plot of h[k] for 0 \leq k \leq 4');
xlabel('k');
ylabel('x[k]');

% Part C
y = conv(x1(k1), h1(k1));
figure;
stem([0:8], y);
title('Convolution of x[k] and h[k]');
xlabel('k');
ylabel('x[k] * h[k]');

%% Question 2 Audio Convolution
k = [0:50];

% Part a
h2 = @(n) (0.3 .* sinc(0.3 .* (n - 25)) .* (0.54 - (0.46 .* cos((2 .* pi .* n) ./ 50)))) .* ((0 <= n) & (n <= 50));

% Part b
figure;
stem(k, h2(k));
title('Filter impulse response');
xlabel('k');
ylabel('h[k]');

% Part c
[x3, fs] = audioread('baila.wav');
filt_x3 = conv(x3, h2(k));

% Part d
audiowrite('baila_filtered.wav', filt_x3, fs)

%% Question 3 Signal Aliasing
% Part a
n1 = [0:30]';
x1 = @(t) (cos(20 .* pi .* t));
x2 = @(t) (cos(180 .* pi .* t));

fs1 = 100;
T1 = 1 / fs1;
```

```
y1 = x1(T1 .* n1);
y2 = x2(T1 .* n1);

figure;
subplot(2, 1, 1);
stem(n1, y1);
title('Sampled signal y_1 result');
xlabel('Samples (n)');
ylabel('Amplitude (y_1)');

subplot(2, 1, 2);
stem(n1, y2);
title('Sampled signal y_2 result');
xlabel('Samples (n)');
ylabel('Amplitude (y_2)');

% Part b
n2 = [0:300];
fs2 = 1000;
T2 = 1 / fs2;
z1 = x1(T2 .* n2);
z2 = x2(T2 .* n2);

figure;
subplot(2, 1, 1);
plot(n2 / fs2, z1, 'r-', n1 / fs1, y1, 'b+');
xlabel('n');
ylabel('y_1[n] and z_1[n]');
legend('z_1[n]', 'y_1[n]');

subplot(2, 1, 2);
plot(n2 / fs2, z2, 'r-', n1 / fs1, y2, 'b+');
xlabel('n');
ylabel('y_2[n] and z_2[n]');
legend('z_2[n]', 'y_2[n]');

% Part c
x3 = @(t) (cos(380 .* pi .* t));
y3 = x3(T1 .* n1);

figure;
plot(n1, y1, 'r-', n1, y3, 'b+');
xlabel('n');
ylabel('y_1[n] and y_3[n]');
legend('y_1[n]', 'y_3[n]');

%% Question 4 Image Aliasing
% Part a
img = imread('barbaraLarge.jpg');

% Part b
figure;
imshow(img), colorbar;

% Part c
img09aa = imresize(img, 0.9, 'nearest', 'antialiasing', 1);
img09noaa = imresize(img, 0.9, 'nearest', 'antialiasing', 0);

img07aa = imresize(img, 0.7, 'nearest', 'antialiasing', 1);
```

```
img07noaa = imresize(img, 0.7, 'nearest', 'antialiasing', 0);

img05aa = imresize(img, 0.5, 'nearest', 'antialiasing', 1);
img05noaa = imresize(img, 0.5, 'nearest', 'antialiasing', 0);

figure;
imshow(img09aa);
figure;
imshow(img09noaa);
figure;
imshow(img07aa);
figure;
imshow(img07noaa);
figure;
imshow(img05aa);
figure;
imshow(img05noaa);
```