Lab 5

Window

0. Abstract

Fast Fourier transform (FFT) calculations are essential to signal processing. The mathematics of the FFT assumes signals have an infinite length, but computer can not deal with something that is infinite. So we need to sample the signal, but it may cause the another problem that is spectrum leakage. In order to deal with this problem, window is applied to help fix the it. In this lab, we will try to use four different windows to processing the signal and find the different property of each window.

1. Sections

1.1. Read Zeropad the Sample to a total length of 1024 samples

Using 32 samples of a sinusoid with frequency 220 Hz, sampled at a constant rate of 2,048 Hz, zeropad it to a total length of 1024 samples and calculate the DFT of the result. The code is as follows:

```
function exe1_1()
fs=2048;
Ndata=32;
N=1024;
n=0:Ndata-1;t=n/fs;
for n = 1:32
x(n)=sin(2*pi*220*n/2048);
end
y=fft(x,N);
y1 = fft(x,Ndata);
mag=abs(y);
f=(0:N-1)*fs/N;
f1 = (0:Ndata-1)*fs/Ndata;
mag1 = abs(y1);
plot(f(1:N/2),mag(1:N/2));
hold on
stem(f1(1:Ndata/2),mag1(1:Ndata/2));
ylabel('|x|');
xlabel('Frequency(Hz)')
```

end

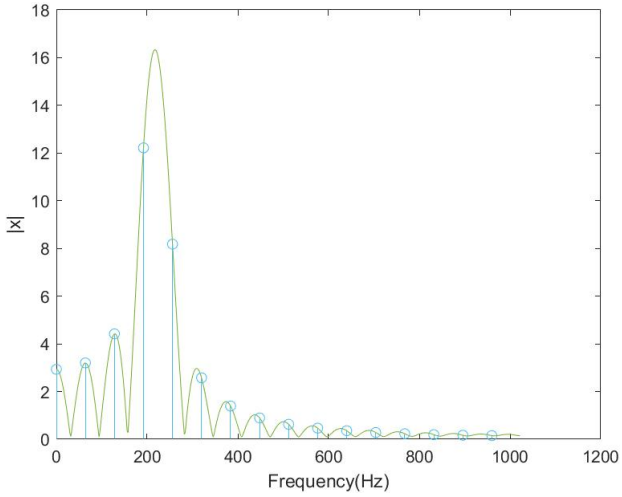

Figure 1: The DFT of the length-32 sinusoid with frequency 220 Hz (stem), and with zeropadding to length-1024 (line).

1.2. Run again in 128Hz

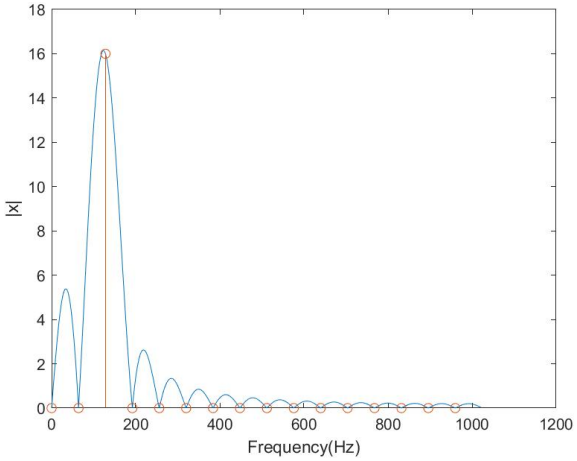

Figure 2: The DFT of the length-32 sinusoid with frequency 128 Hz (stem), and with zeropadding to length-1024 (line).

Zero padding occurs when we add a border of pixels all with value zero around the edges of the input images. This adds kind of a padding of zeros around the outside of the image, hence the name zero padding. When using 128Hz instead of 220Hz, besides a peek in the 128Hz, there is also a high peak at around 32Hz, which means the real componment at 128Hz becomes several components. This causes some spurious frequencies.

1.3. Evaluate the DFT of three sinusoids

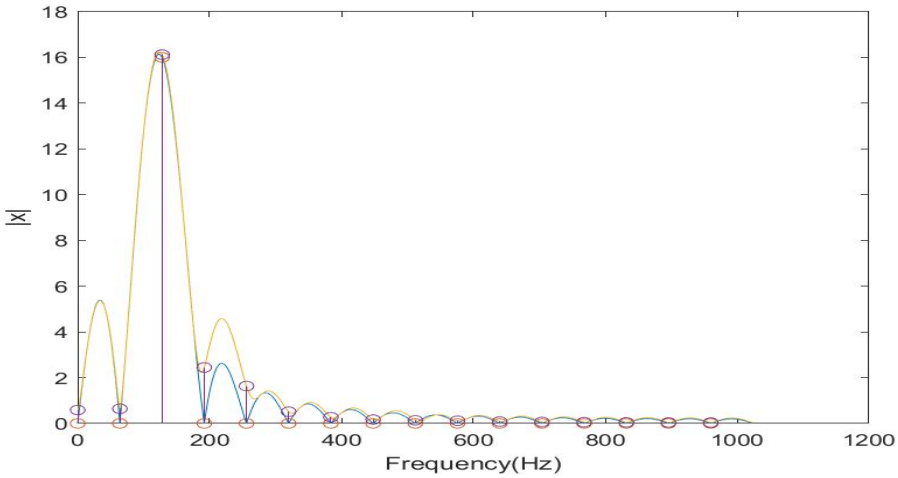

Figure 3: The DFT of the length-32 with three sinusoids, and with zeropadding to length-2048 (line).

1.4. Pick out the frequency components

From Figure 3, it is easy to pick out the 128Hz and 220Hz. The peak at around 32Hz is very misleading. And it is hard to distinguish the 525Hz since the magnitude of this frequency is 0.01.

2.1.The plot of four different windows

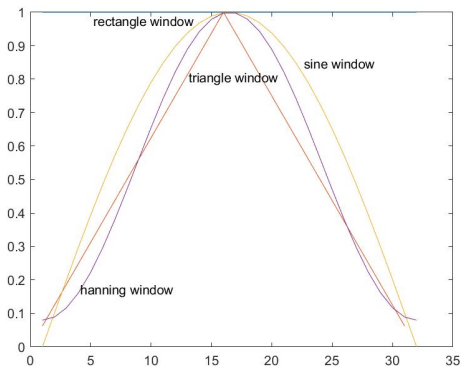

Figure 4: The plot of four windows

2.2 The DFT of each window

DFT is applied to each of the window at 1024 points. The code is as follows:

```
function exe2_2()
rec = boxcar(32);
tri = triang(31);
for n = 0 : 31
sine(n+1) = sin(pi*n/31);
end
han = hamming(32);

recfft = fft(rec,1024);
recfft = abs(recfft);
recfft = fftshift(recfft);
n = linspace(-0.5,0.5,1024);
subplot(2,2,1), plot(n,recfft),title('Rectangular');

trifft = fft(tri,1024);
trifft = abs(trifft);
trifft = fftshift(trifft);
n = linspace(-0.5,0.5,1024);
subplot(2,2,2), plot(n,trifft),title('Triangular');

sinefft = fft(sine,1024);
sinefft = abs(sinefft);
sinefft = fftshift(sinefft);
n = linspace(-0.5,0.5,1024);
subplot(2,2,3),plot(n,sinefft),title('Sine');

hanfft = fft(han,1024);
hanfft = abs(hanfft);
hanfft = fftshift(hanfft);
n = linspace(-0.5,0.5,1024);
subplot(2,2,4),plot(n,hanfft),title('Hann');

xlabel('f');
end
```

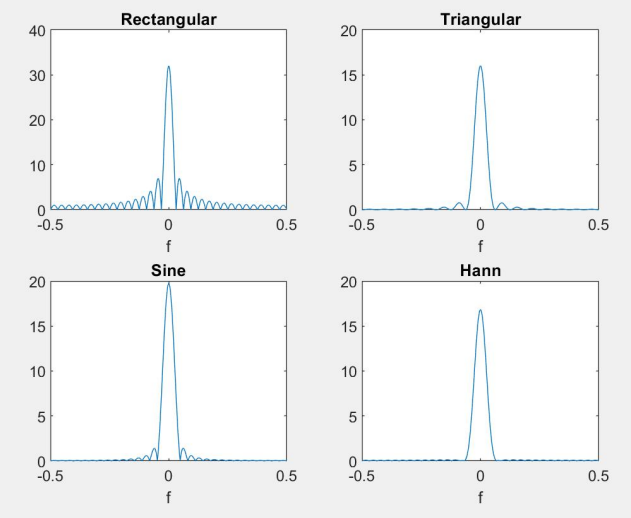

Figure 5: The DFT of each window

2.3Plot the normalized dB spectrum of the rectangular and triangular

The code of the normalized dB spectrum of the rectangular and triangular window is as follows:

```
function exe2_3()
rec = boxcar(32);
tri = triang(31);


recfft = fft(rec,1024);
recfft = abs(recfft);
recfft = fftshift(recfft);
recfft = 20*log10(abs(recfft)./max(abs(recfft)));
n = linspace(-0.5,0.5,1024);
plot(n,recfft),title('Rectangular');
hold on

trifft = fft(tri,1024);
trifft = abs(trifft);
trifft = fftshift(trifft);
trifft = 20*log10(abs(trifft)./max(abs(trifft)));
n = linspace(-0.5,0.5,1024);
plot(n,trifft,'--'),title('Triangular');
axis([-0.2 0.2 -80 5]);
xlabel('f');
legend('Rectangular','Triangular');
end
```

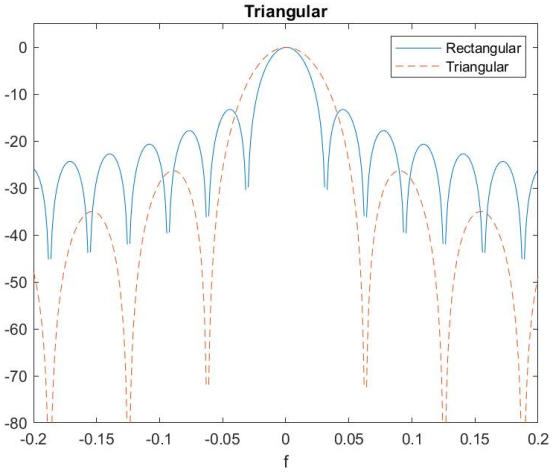

Figure 6: The normalized magnitude spectrum in different frequencies

2.4 Find the main-lobe

The width of the rectangle window is approximate to 0.1 and the width of the triangular window is approximate to 0.2. The triangular has twice as much as the width of rectangle window. Because the expression of the width of the rectangle window is 4 π /N, while the rectangular window is 8 π /N.

2.5 Answer the question

The code and comments are as follow:

The magnitude is -13.29dB for the first side-lobe for the rectangular window.And the magnitude is -26.40db for the first side-lobe for the triangular window.

2.6 Repeat for Sine and Hann window

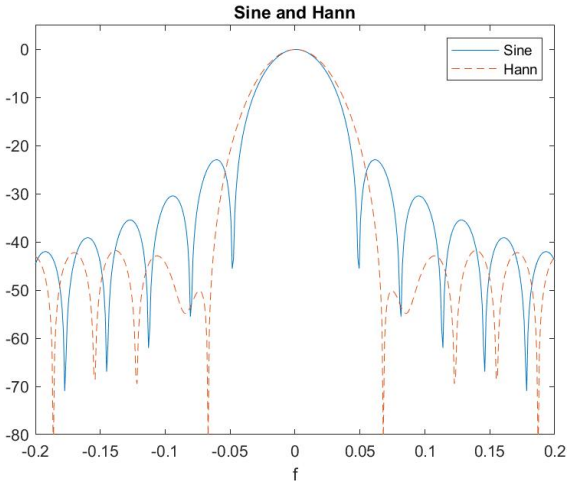

Figure 6: The normalized magnitude spectrum in different frequencies

2.7 The width of the main-lobe for the sine and Hann window and the magnitude of the first side-lobe

The width of the main-lobe for the sine and Hann are 0.12 and 0.21 respectively. The magnitude of the height of the first side-lobe for the Sine and Hann window are -22.9 and -42.8 respectively.

3.1Plot the normalized manitude of each window

The code is as follows:

```
function exe3_1()
fs=2048;
Ndata=32;
N=2048;
n=0:Ndata-1;t=n/fs;
for n = 1:32
x(n)=sin(2*pi*128*n/2048) + 0.2*sin(2*pi*220*n/2048) +
0.01*cos(2*pi*525*n/2048);
end
rec = boxcar(32);
tri = triang(32);
for n = 0 : 31
sine(n+1) = sin(pi*n/31);
```

```
end
han = hamming(32);

for m = 1 :32
y1(m) = x(m) * rec(m);
y2(m) = x(m) * tri(m);
y3(m) = x(m) * sine(m);
y4(m) = x(m) * han(m);
end
y1 = fft(y1);
y2 = fft(y2);
y3 = fft(y3);
y4 = fft(y4);
mag1=abs(y1);
mag2=abs(y2);
mag3=abs(y3);
mag4=abs(y4);

f=(0:Ndata-1)*fs/Ndata;

subplot(2,2,1),
stem(f(1:Ndata/2),mag1(1:Ndata/2)./max(mag1)),title('Rectangular') ;
subplot(2,2,2),
stem(f(1:Ndata/2),mag2(1:Ndata/2)./max(mag2)),title('Triangular') ;
subplot(2,2,3), stem(f(1:Ndata/2),mag3(1:Ndata/2)./max(mag3)),title('Sine');
subplot(2,2,4), stem(f(1:Ndata/2),mag4(1:Ndata/2)./max(mag4)),title('Hamm') ;
end
```

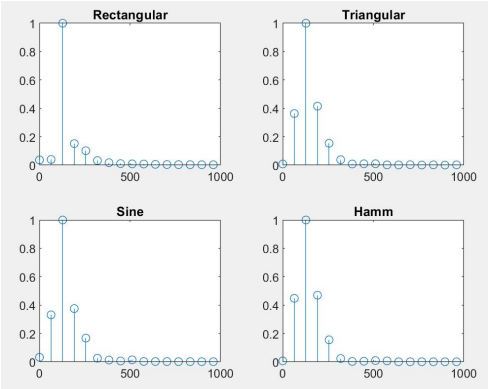

Figure 7: Winow the 32 sa-sample sequence created in 1.3 with the rectangular, triangular, Sine, and Hann windows

3.2 Zeropad each sequence to length 2,048

The code is as follows:

```
function exe3_2()
fs=2048;
Ndata=32;
N=2048;
```

```
n=0:Ndata-1;t=n/fs;
for n = 1:32
x(n)=sin(2*pi*128*n/2048) + 0.2*sin(2*pi*220*n/2048) +
0.01*cos(2*pi*525*n/2048);
end
rec = boxcar(32);
tri = triang(32);
for n = 0 : 31
sine(n+1) = sin(pi*n/31);
end
han = hamming(32);

for m = 1 :32
y1(m) = x(m) * rec(m);
y2(m) = x(m) * tri(m);
y3(m) = x(m) * sine(m);
y4(m) = x(m) * han(m);
end
y1 = fft(y1,2048);
y2 = fft(y2,2048);
y3 = fft(y3,2048);
y4 = fft(y4,2048);


mag1=20 * log10(abs(y1)./max(abs(y1)));
mag2=20 * log10(abs(y2)./max(abs(y2)));
mag3=20 * log10(abs(y3)./max(abs(y3)));
mag4=20 * log10(abs(y4)./max(abs(y4)));

f=(0:Ndata-1)*fs/Ndata;

subplot(2,2,1), plot(f(1:Ndata/2),mag1(1:Ndata/2)),title('Rectangular') ;
subplot(2,2,2), plot(f(1:Ndata/2),mag2(1:Ndata/2)),title('Triangular') ;
subplot(2,2,3), plot(f(1:Ndata/2),mag3(1:Ndata/2)),title('Sine');
subplot(2,2,4), plot(f(1:Ndata/2),mag4(1:Ndata/2)),title('Hamm') ;
end
```

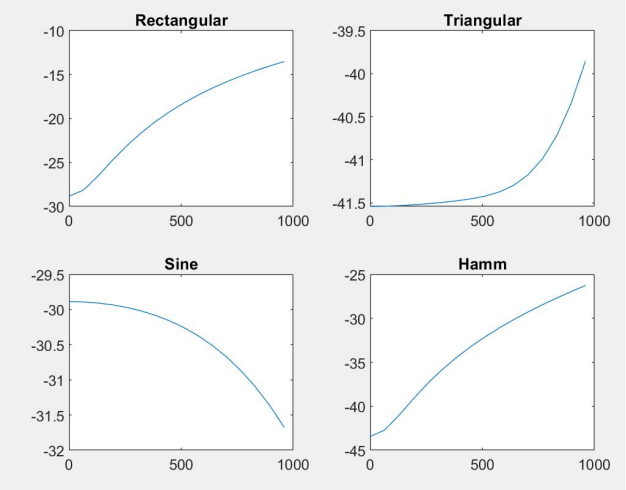

Figure 8: Winow the 32 sa-sample sequence created in 1.3 with the rectangular, triangular, Sine, and Hann windows zeropad to 2048

3.3 Answer to the question

From the Hanning window , it can be say this signal has three components. The Triangular window can have the accurate frequency of the middle amplitude sinusoid.
The Hann and Sine window the lowest magnitude side-lobes of all these four windows, so it can revealing the lowest amplitude component. Because the magnitude of the height of the side-lobe is very small , so it can not reveal the middle amplitude component.
The lowest amplitude component have a normalized dB level of -40dB, because the coefficient of the 128Hz is 1, and the coefficient of the 525Hz is 0.01. Using the dB equation, the outcome is -40dB.

3.4 The DFT of the first 1.4 seconds of this signal

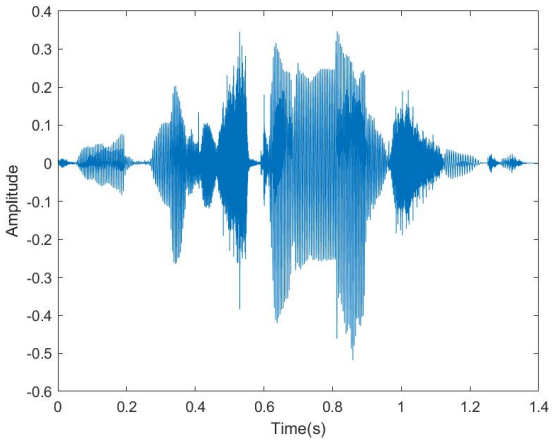

Figure 8: The DFT of the first 1.4 seconds of this signal

3.5 Plot the normalized dB magnitude spectrum of the windowed speech

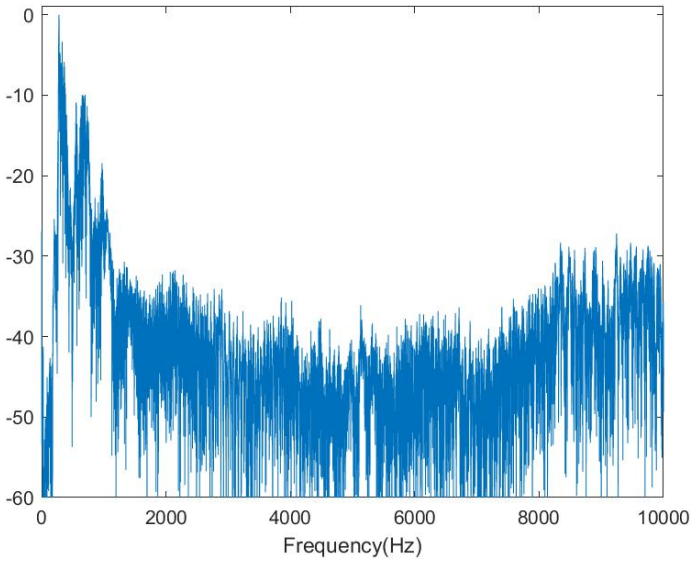

Figure 9: The normalized dB magnitude spectrum of the window speech

3.6 Take a 2048-sample segment starting at 0.15 seconds

The code is as follows:

```
function exe3_6()
Ndata = 2048;
[x,Fs] = audioread('speech_female.wav');
y = x(Fs * 0.15 : Fs * 0.15 + 2047);
han = hamming(2048)
for i = 1 : 2048
y_win(i) = y(i) * han(i)
end
y_win = fft(y_win);
y_win = 20 * log10(abs(y_win)./max(abs(y_win)));
f=(0:Ndata-1)*Fs/Ndata;
plot(f(1:Ndata),y_win(1:Ndata))
ylabel('dB');
end
```

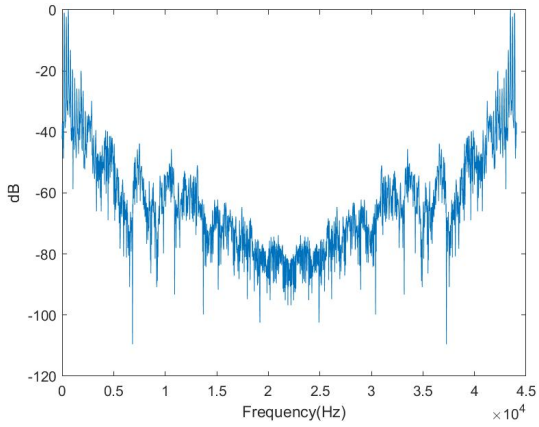

Figure 10: The DFT of this windowed segment and plot its normalized magnitude dB

3.7 Take a 2048-sample segment

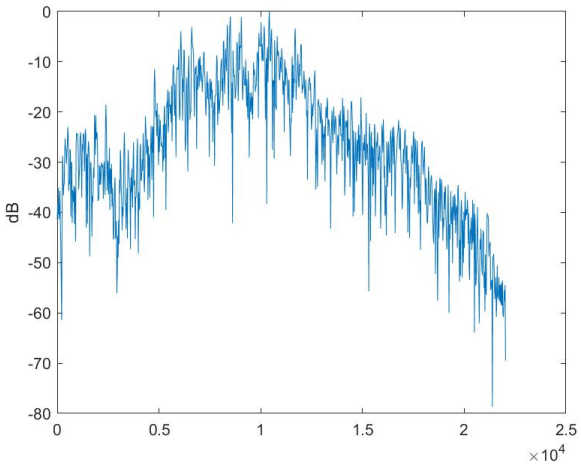

Figure 11: The DFT of this windowed segment and plot its normalized magnitude dB

3.8 The duration of each of these windowed segments

The the duration of each of these windowed segments is 0.0464s by applying the formula :`a = 2048*Duration/length(x).`

2. Conclusion

In zero padding, add zeros to the end of the input sequence so that the total number of samples is equal to the next higher power of two. But zeropadding is not always good, since it may break some sequence into several sequence.

Window functions are added to a signal processing algorithm to address the discontinuity problem but they do not totally overcome it. These functions operate by multiplying the time waveform by a finite-length window with an amplitude that varies smoothly and gradually toward zero at the edges.

However, multiplication in time does result in a distortion in frequency. So, while the FFT of the windowed waveform will be similar to the FFT of the original, it will have been altered. It needs to be careful when choosing the window since there is always a trade-off.

3. Acknowledgments

Thanks for the TA gives us the illustration explain why the rectangular window is usually not ideal when processing the signal.