Lab 5: Sampling and Discrete Fourier Transform

Table of Contents

Mai Abdelhameed & Omar Ahmad

Part A: Discrete Fourier Transform and Zero Padding

```
clear;
%A1
n1 = 0:9;   %10 samples,
fr = linspace(-0.5, 0.5-1/10, 10);

N1 = length(n1);    %length = 10
x1 = (exp(j*2*pi*n1*(30/100))) + (exp(j*2*pi*n1*(33/100)));
x2 = (cos(2*pi*n1*(30/100))) + (0.5*cos(2*pi*n1*(0.4)));

X1 = fft(x1, N1);
X2 = fft(x2, N1);

figure(1);
subplot(2,1,1);
stem(fr, abs(fftshift(X1)));
title('|X1(w)| with 10 samples');
xlabel('w');
grid;

subplot(2,1,2);
stem(fr, abs(fftshift(X2)));
title('|X2(w)| with 10 samples');
xlabel('w');
grid;

disp('A1. (i): |X2(w)| has a symmetric spectrum because x2[n] is a
 periodic signal.');
disp('A1. (ii): It is possible. As x1[n] is not a periodic signal, |
X1(w)| will have just one peak, unlike |X2(w)|, which has multiple.');
disp('A1. (iii): There is an exponential at w = 0.33, but as the
 sampling rate is not big enough, this frequency component is spread
 out at frequencies that taken into account.');

%A2
```

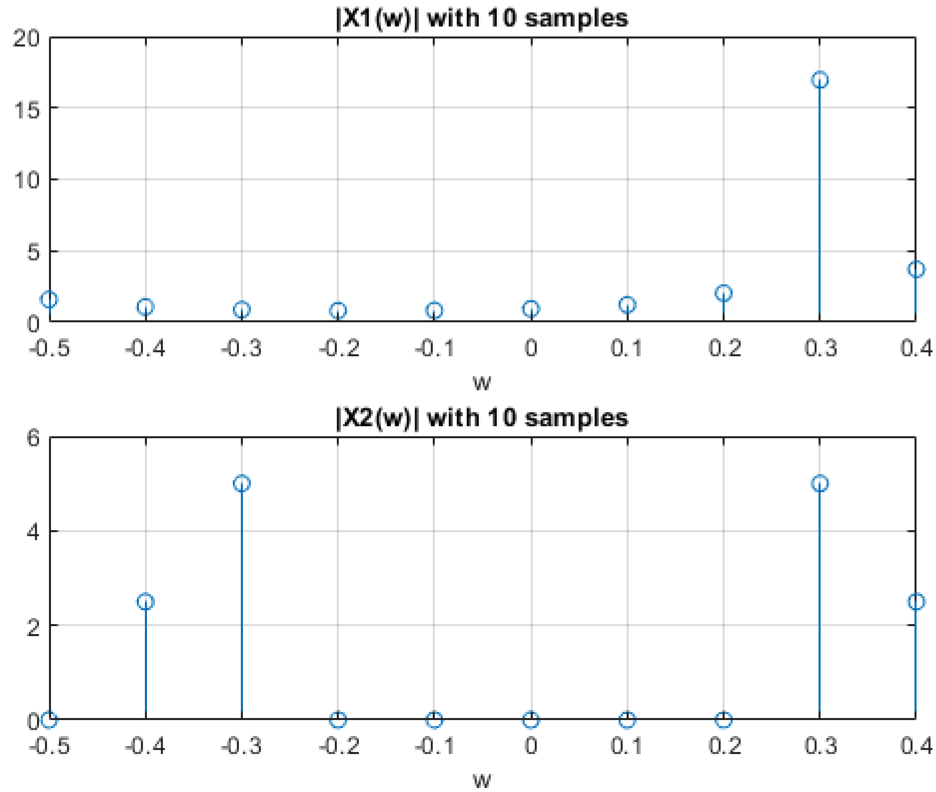
```
n2 = 0:499;     %500 samples
N2 = length(n2);    %length = 500

fr = linspace(-0.5, 0.5-1/500, 500);  %making freq vector same length
 as DFT vectors

x_1 = [zeros(1,245),x1, zeros(1,245)];  %490 zeros
X_1 = fft(x_1,N2);

x_2 = [zeros(1,245),x2, zeros(1,245)];  %490 zeros
X_2 = fft(x_2, N2);

figure(2);
subplot(2,1,1);
stem(fr, abs(fftshift(X_1)));
title('|X1(w)| with 10 samples but zero-padded with 490 zeros');
xlabel('w');
grid;

subplot(2,1,2);
stem(fr, abs(fftshift(X_2)));
title('|X2(w)| with 10 samples but zero-padded with 490 zeros')'
xlabel('w');
grid;

disp('A2: Yes. As we zero-pad the signals, we get a clearer image
 of what our spectrum is supposed to look like. It appears more
 continuous.');

%A3
n1 = 0:99;  %100 samples
fr = linspace(-0.5, 0.5-1/100, 100);  %making freq vector same length
 as DFT

N1 = length(n1);    %length = 100

x1 = (exp(j*2*pi*n1*(30/100))) + (exp(j*2*pi*n1*(33/100)));
x2 = (cos(2*pi*n1*(30/100))) + (0.5*cos(2*pi*n1*(0.4)));

X1 = fft(x1, N1);
X2 = fft(x2, N1);

figure(3);
subplot(2,1,1);
stem(fr, abs(fftshift(X1)));
title('|X1(w)| with 100 samples');
xlabel('w');
grid;

subplot(2,1,2);
stem(fr, abs(fftshift(X2)));
title('|X2(w)| with 100 samples');
xlabel('w');
grid;
```

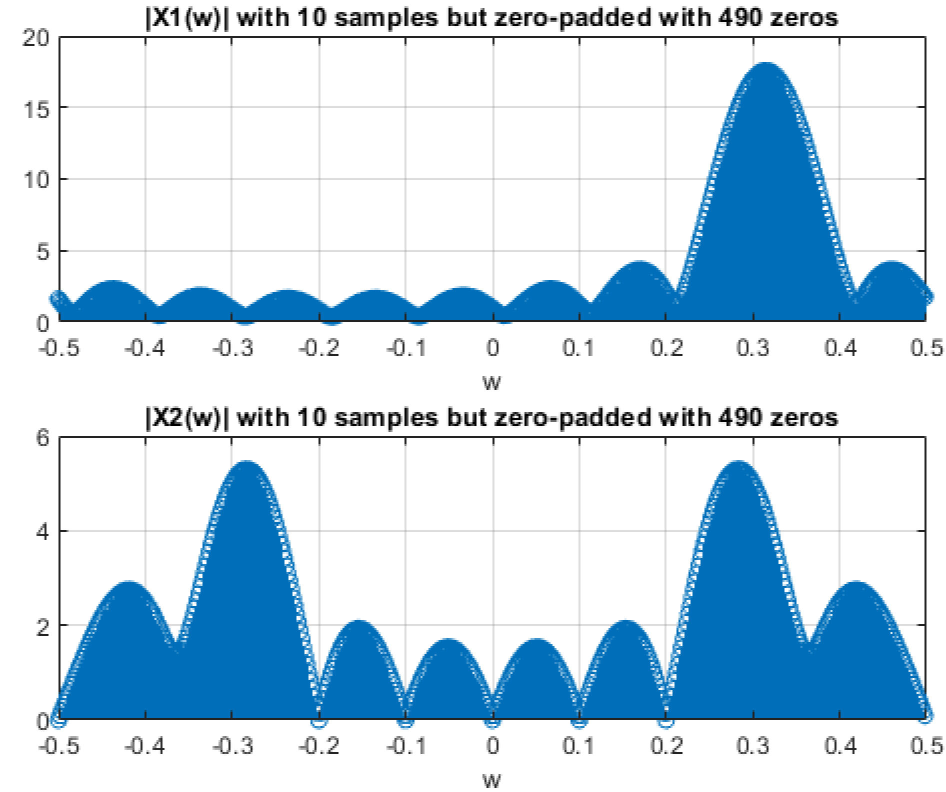
```
disp('A3: |X2(w)| has a symmetric spectrum since x2[n] is a cos
 function, which is periodic.');

%A4
fr = linspace(-0.5, 0.5-1/500, 500);

n2 = 0:499;      %500 samples
N2 = length(n2);    %length = 500

x_1 = [zeros(1,200),x1, zeros(1,200)];  %400 zeros
X1 = fft(x_1,N2);

x_2 = [zeros(1,200),x2, zeros(1,200)];  %400 zeros
X2 = fft(x_2, N2);

figure(4);
subplot(2, 1, 1);
stem(fr, abs(fftshift(X1)));
title('|X1(w)| with 100 samples but zero-padded with 400 zeros');
xlabel('w');
grid;

subplot(2, 1, 2);
stem(fr, abs(fftshift(X2)));
title('|X2(w)| with 100 samples but zero-padded with 400 zeros');
xlabel('w');
grid;

disp('A4: Yes. With more samples taken, and zero-padding, there is a
 much more accurate representation of the signal.')
```

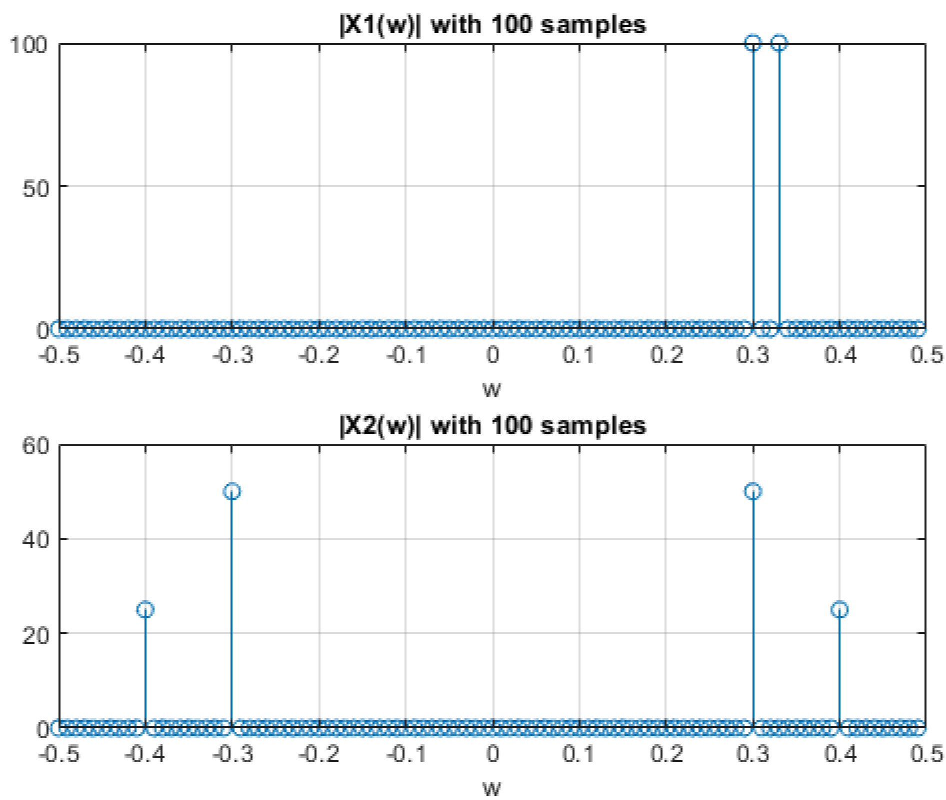
```
A1. (i): |X2(w)| has a symmetric spectrum because x2[n] is a periodic
 signal.
A1. (ii): It is possible. As x1[n] is not a periodic signal, |X1(w)|
 will have just one peak, unlike |X2(w)|, which has multiple.
A1. (iii): There is an exponential at w = 0.33, but as the sampling
 rate is not big enough, this frequency component is spread out at
 frequencies that taken into account.

ans =

  Text (|X2(w)| with 10 samples but zero-padded with 490…) with
 properties:

                 String: '|X2(w)| with 10 samples but zero-padded with
 490 zeros'
               FontSize: 9.9000
             FontWeight: 'bold'
               FontName: 'Helvetica'
                  Color: [0 0 0]
    HorizontalAlignment: 'center'
               Position: [6.3835e-07 6.0517 1.4211e-14]
                  Units: 'data'
```

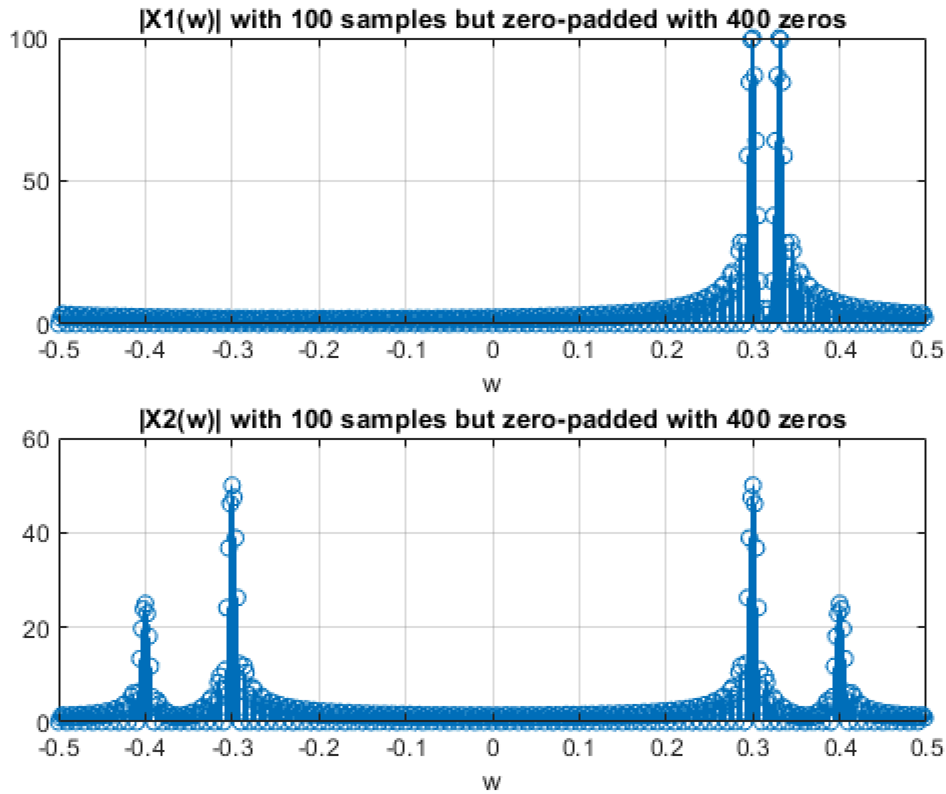
Use GET to show all properties

A2: Yes. As we zero-pad the signals, we get a clearer image of what our spectrum is supposed to look like. It appears more continuous.
A3: |X2(w)| has a symmetric spectrum since x2[n] is a cos function, which is periodic.
A4: Yes. With more samples taken, and zero-padding, there is a much more accurate representation of the signal.




|X1(w)| with 10 samples but zero-padded with 490 zeros
20
15
10
5
0
-0.5 -0.4 -0.3 -0.2 -0.1 0 0.1 0.2 0.3 0.4 0.5
w
|X2(w)| with 10 samples but zero-padded with 490 zeros
6
4
2
0
-0.5 -0.4 -0.3 -0.2 -0.1 0 0.1 0.2 0.3 0.4 0.5
w


|X1(w)| with 100 samples
100
50
0
-0.5 -0.4 -0.3 -0.2 -0.1 0 0.1 0.2 0.3 0.4 0.5
w
|X2(w)| with 100 samples
60
40
20
0
-0.5 -0.4 -0.3 -0.2 -0.1 0 0.1 0.2 0.3 0.4 0.5
w



Part B: Sampling

```
clear;
load chirp.mat;
filename = 'chirp.wav';
audiowrite(filename,y,Fs);
clear y Fs
[y,fs] = audioread('chirp.wav');

%B.1
N0 = length(y)     %Number of samples
T0 = N0/fs     %Duration of the signal
T = 1/fs   %Sampling interval

%B.2
t = linspace(0, T0, N0);

figure(5);
plot(t, y);
title('B.2: y with respect to time');
xlabel('t');
grid;

%B.3
omega = linspace(-(fs/2), (fs/2), N0);
```

```
Y = fft(y);

figure(6);
plot(omega, fftshift(abs(Y)));
title('B.3: DFT(y(t))');
xlabel('w');
grid;

%B.4
y1 = y(1:2:N0);

N1 = length(y1)
T01 = N1/fs
T1 = 2*fs

%B.5
t1 = t(1:2:N0);

figure(7);
plot(t1, y1);
title('B.5: y1 with respect to time');
xlabel('t');
grid;

%B.6
omega1 = linspace(-(fs/4), (fs/4), N1); %because sampling by rate of 2
Y1 = fft(y1);

figure(8);
plot(omega1, fftshift(abs(Y1)));
title('B.6: DFT of y1');
xlabel('w');
grid;

disp('The signal has subsampled by a factor of 2. On the spectrum, the
 frequency was halved (1/2).')

%B.7
%sound(y, fs)
%sound(y1, fs)

%B.8
y5 = y(1:5:N0);

N5 = length(y5)
T05 = N5/fs
T5 = 2*fs

t5 = t(1:5:N0);

figure(9);
plot(t5, y5);
title('B.8: y5 with respect to time')
xlabel('t');
```

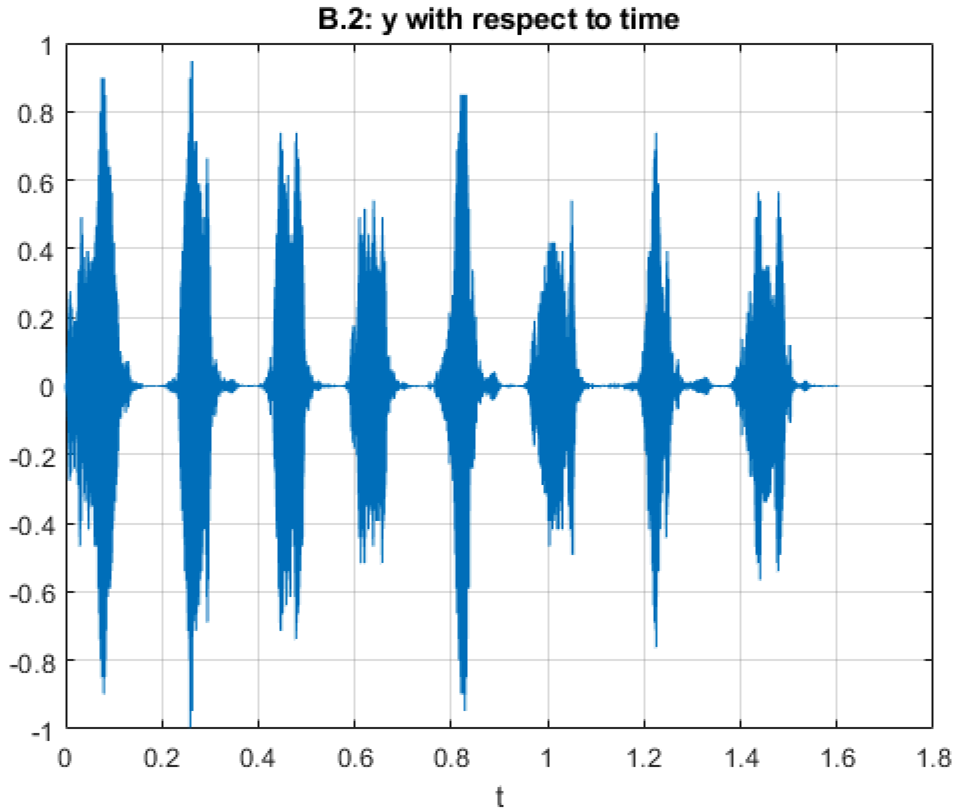
```
grid;

%sound(y5, fs)

omega5 = linspace(-(fs/10), (fs/10), N5);
Y5 = fft(y5);

figure(10);
plot(omega5, fftshift(abs(Y5)));
title('B.8: DFT of y5');
xlabel('w');
grid;
```

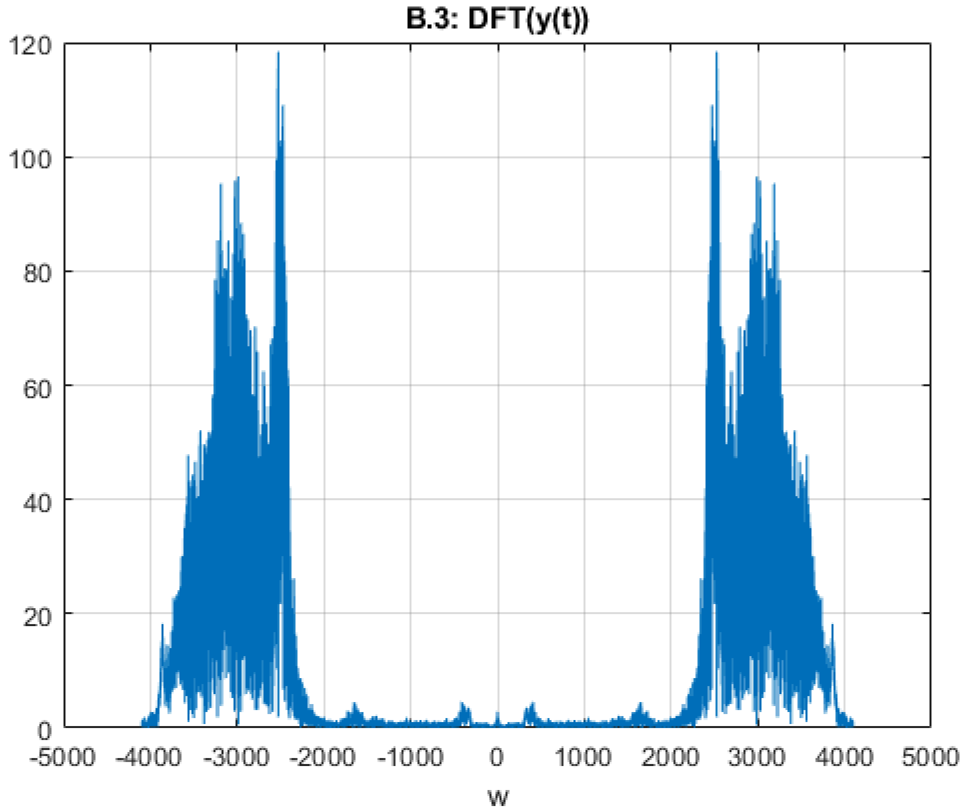
```
N0 =

       13129


T0 =

    1.6027


T =

   1.2207e-04


N1 =

        6565


T01 =

    0.8014


T1 =

       16384
```

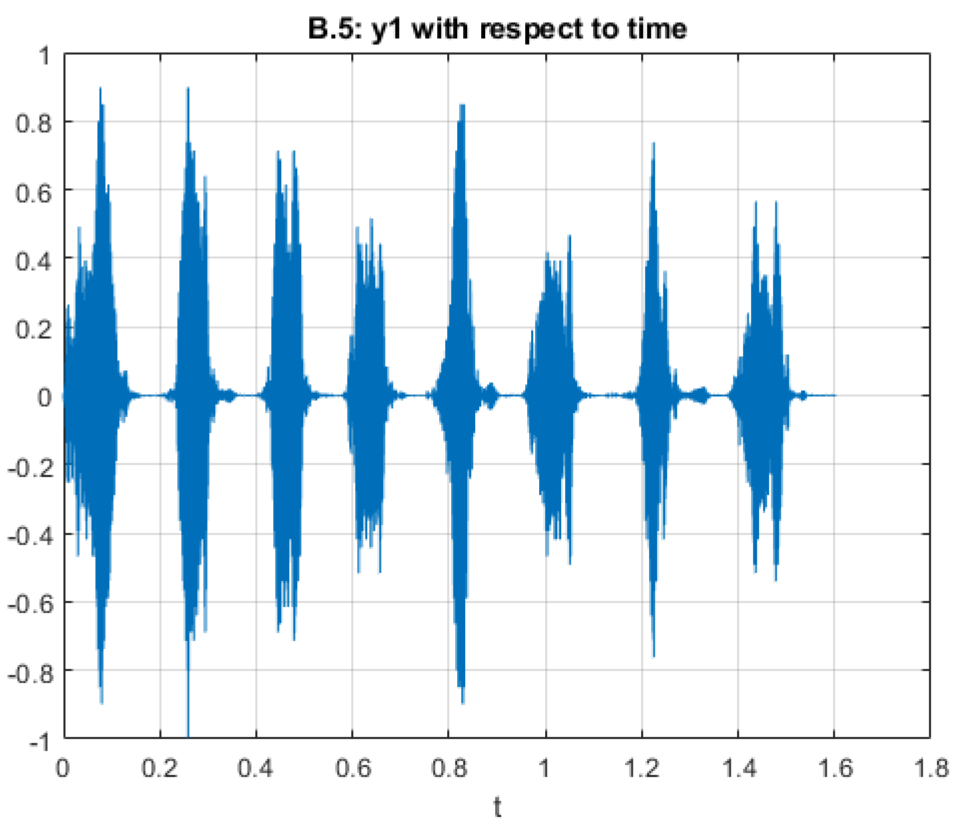
The signal has subsampled by a factor of 2. On the spectrum, the frequency was halved (1/2).

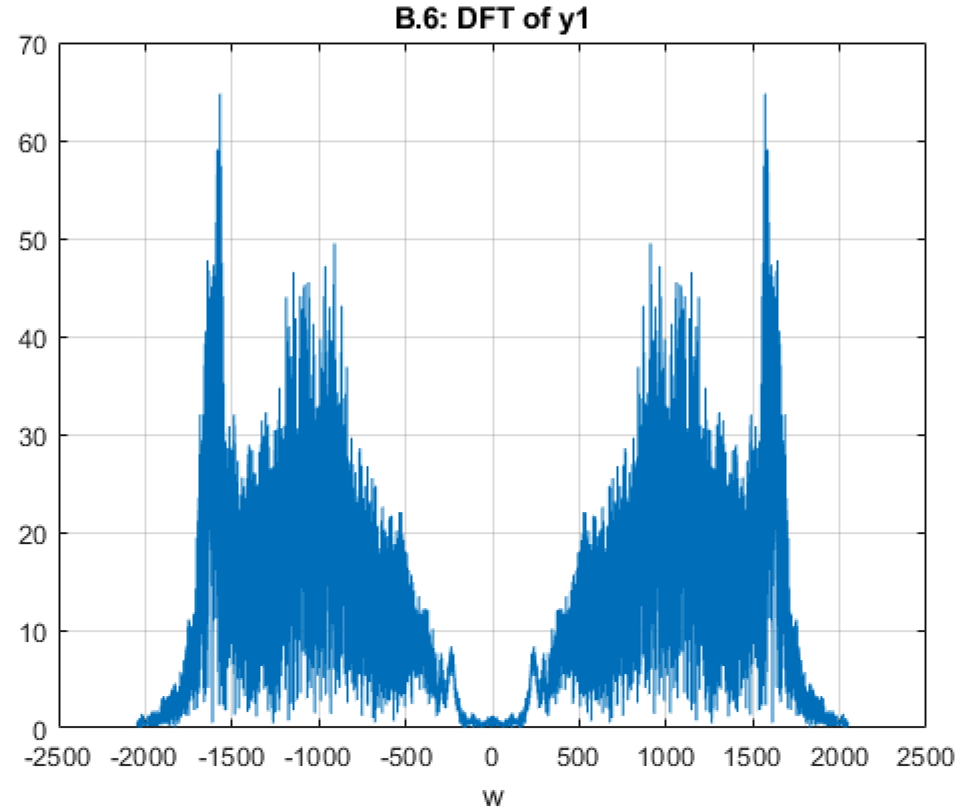
```
N5 =

        2626


T05 =

    0.3206
```

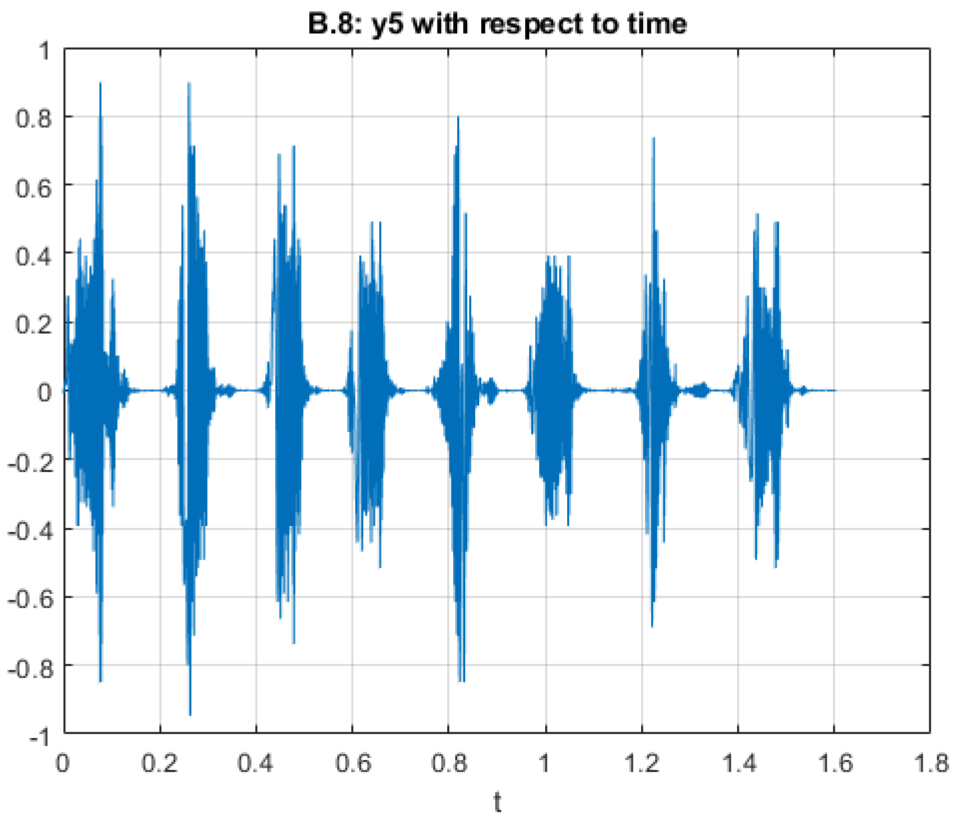
```
T5 =

        16384
```

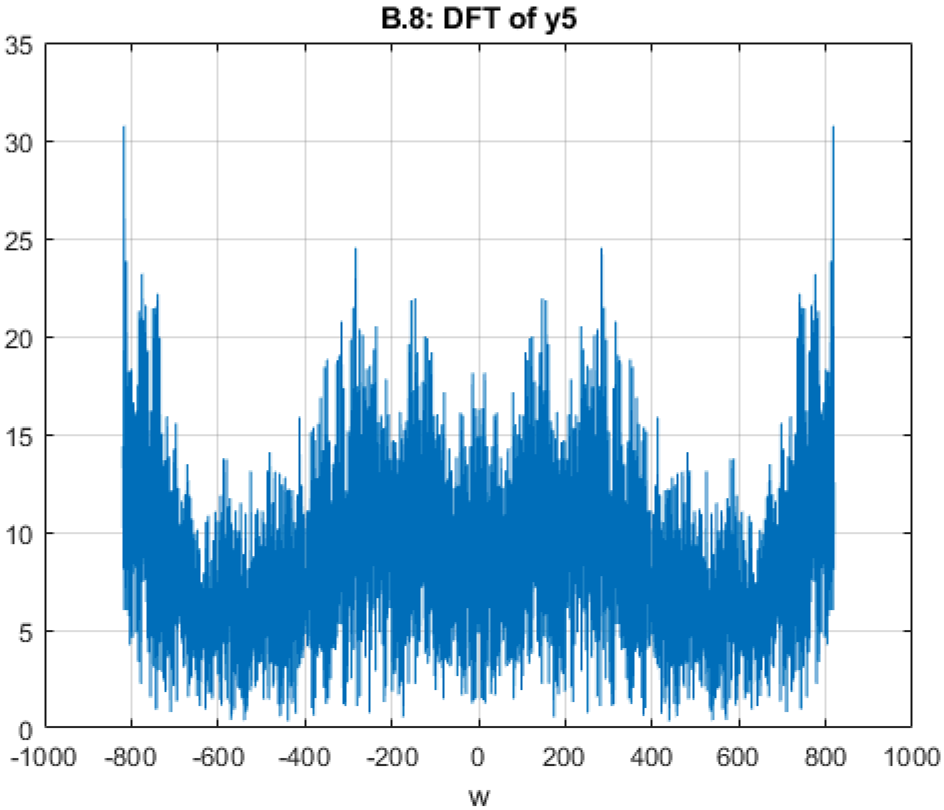

B.2: y with respect to time
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
t


B.3: DFT(y(t))
120
100
80
60
40
20
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
w


B.5: y1 with respect to time
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8
t


B.6: DFT of y1
70
60
50
40
30
20
10
0
-2500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
w


B.8: y5 with respect to time
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
t



Part C: Filter Design

```
%C.1
clear;

% file is already loaded in previous section, so just read the .wav
 file
file = 'handel.wav'
[y,Fs] = audioread(file);

audio = y;
DFT_audio = fftshift(fft(audio)); %DFT of time domain audio signal,
 shifted
half = Fs/2;
t = 0:1:length(DFT_audio)-1; % Time X axis
t = t/10000;
f = linspace(0, Fs, length(y)); % Frequency X axis
f = f-half;

% Lowpass filter @2kHz
H = abs(f) < 2000;
H = transpose(H);

filtered_audio = H.*DFT_audio; % filtered system @2kHz
```

```
figure;
subplot(2,1,1);
    plot(t,audio);
    title('Audio in Time Domain');
    xlabel('Time (s)');
    grid on;
subplot(2,1,2);
    plot(f,abs(DFT_audio));
    title('Audio in Frequency Domain');
    xlabel('Frequency (Hz)');
    grid on;

figure;
plot(f,abs(H));
    title('2kHz Lowpass Filter');
    xlabel('Frequency (Hz)');
    grid on;

figure;
subplot(2,1,1);
    plot(f,abs(filtered_audio));
    title('Audio in Frequency Domain from -2kHz to 2kHz');
    xlabel('Frequency (Hz)');
    grid on;
subplot(2,1,2);
    plot(t,real(ifft(fftshift(filtered_audio))));
    title('Filtered Audio in Time Domain');
    xlabel('Time (s)');
    grid on;

% C.2

sound(real(ifft(fftshift(filtered_audio))),Fs);
disp('The frequencies higher than +-2kHz got removed, so parts of the
 song');
disp('that contained those frequencies went silent.')
% C.3

% Bandpass bass filtering out between 16-256
H2 = ~(abs(f) >= 16 & abs(f) <= 256);
H2 = transpose(H2);

% Bass frequencies filtered out
filtered_audio2 = DFT_audio.*H2;

figure;
plot(f,abs(H2));
    title('Bass Filter between 16-256 Hz');
    xlabel('Frequency (Hz)');
    grid on;

figure;
subplot(2,1,1);
    plot(f,abs(filtered_audio2));
```

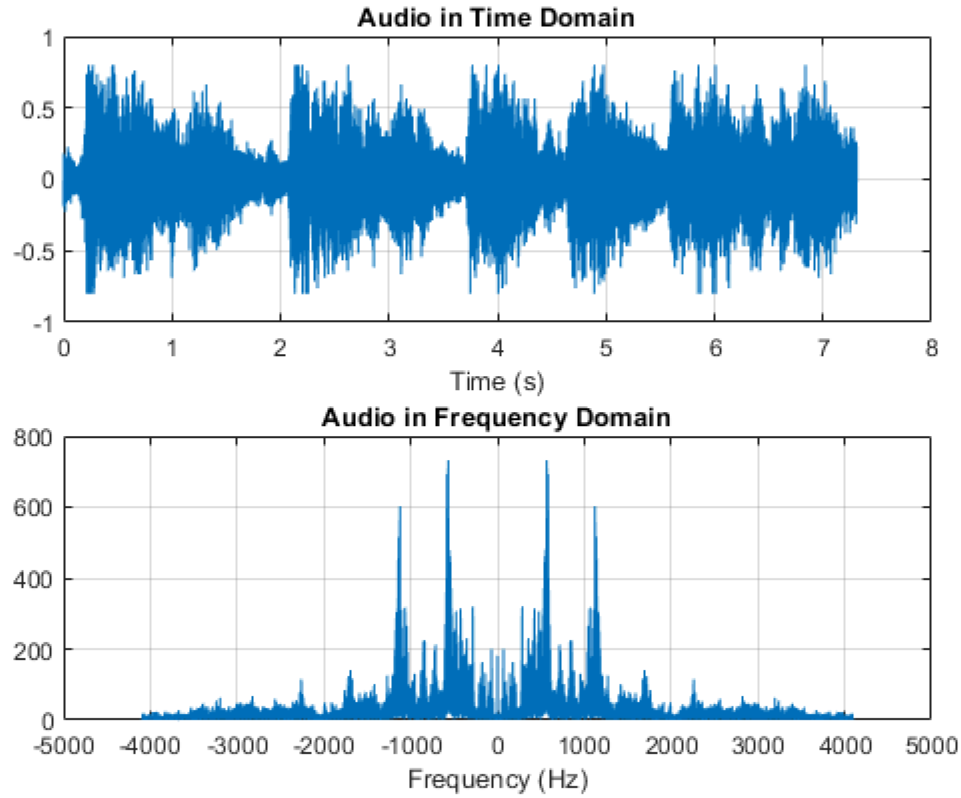
```
    title('Filtered Audio in the Frequency Domain');
    xlabel('Frequency (Hz)');
    grid on;
subplot(2,1,2);
    plot(t,real(ifft(fftshift(filtered_audio2))));
    title('Filtered Audio in the Time Domain');
    xlabel('Time (s)');
    grid on;

sound(real(ifft(fftshift(filtered_audio2))),Fs);
disp('The low frequencies/bass sounds were removed, and replaced with
 silence');
% C.4

% Bandpass filter between 2048-16384
H3 = abs(f) >= 2048 & abs(f) <= 16384;
H3 = transpose(H3);

% Amplitude of bandpass is going to be 0.25 to reduce the frequencies
% passing though
H3 = H3.*0.25;

%grab the 25% to add to the original audio
filtered_audio3 = DFT_audio+(DFT_audio.*H3);

figure;
plot(f,real(H3));
    title('Treble Filter between 2048-16384 Hz');
    xlabel('Frequency (Hz)');
    grid on;

figure;
subplot(2,1,1);
    plot(f,abs(filtered_audio3));
    title('Amplified Audio in the Frequency Domain');
    xlabel('Frequency (Hz)');
    grid on;
subplot(2,1,2);
    plot(t, real(ifft(filtered_audio3)));
    title('Amplified Audio in the Time Domain');
    xlabel('Time (s)');
    grid on;

sound(real(ifft(fftshift(filtered_audio3))),Fs);
disp('The higher frequencies of the music were amplified and got
 louder');
```

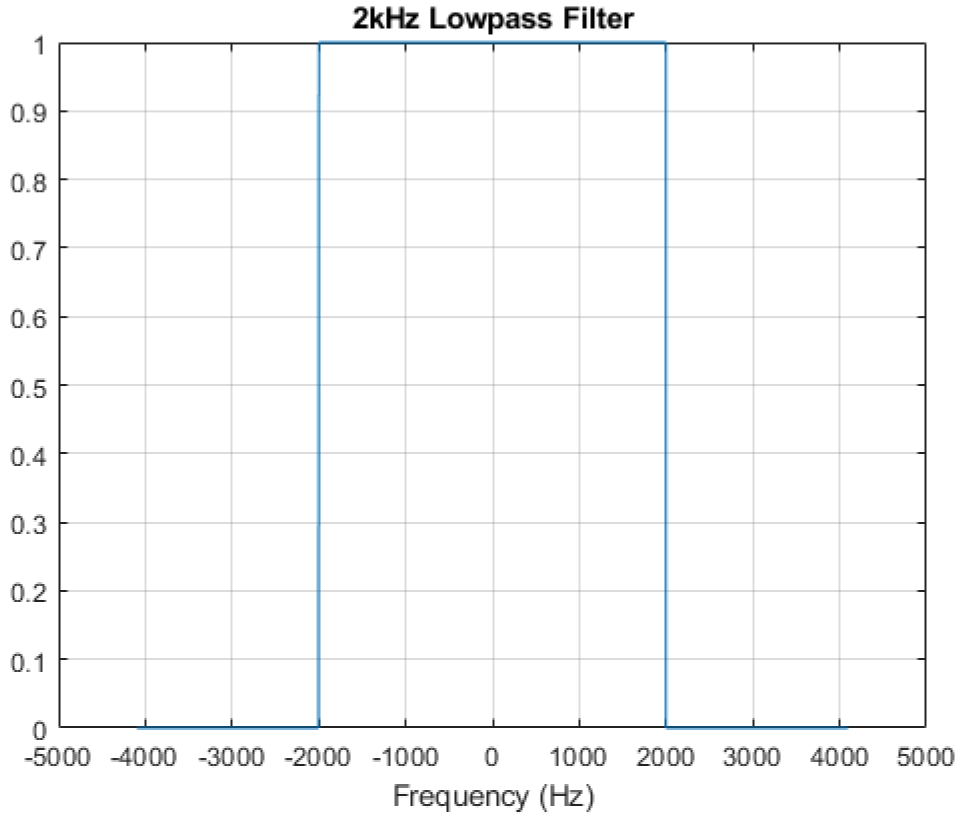
file =

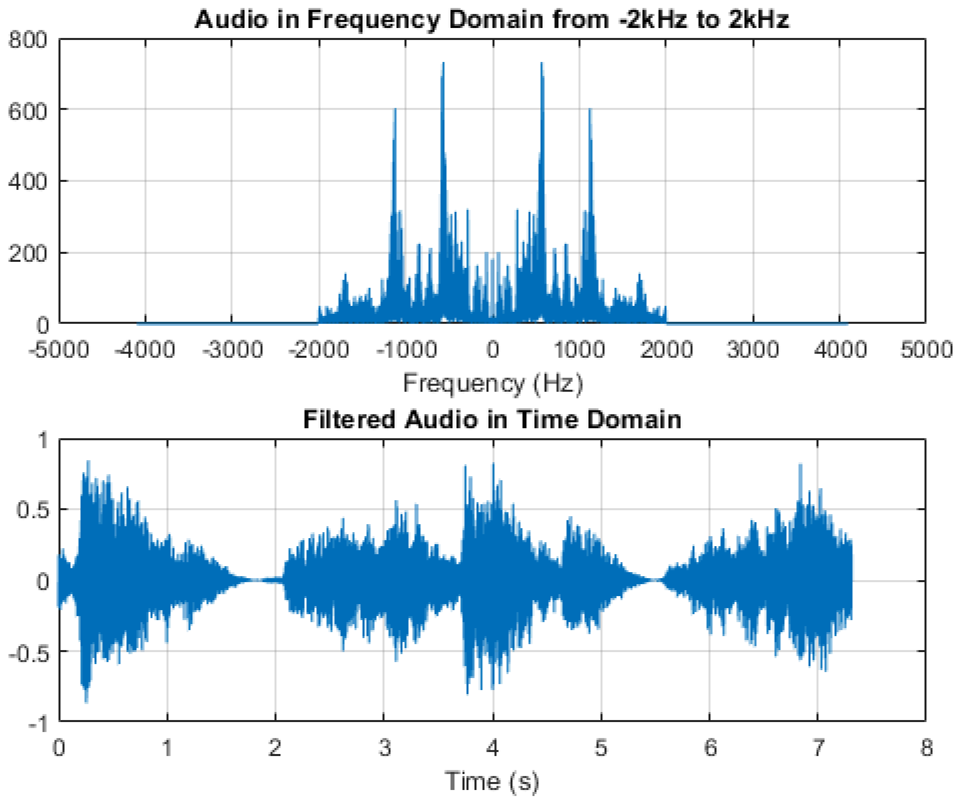
'handel.wav'

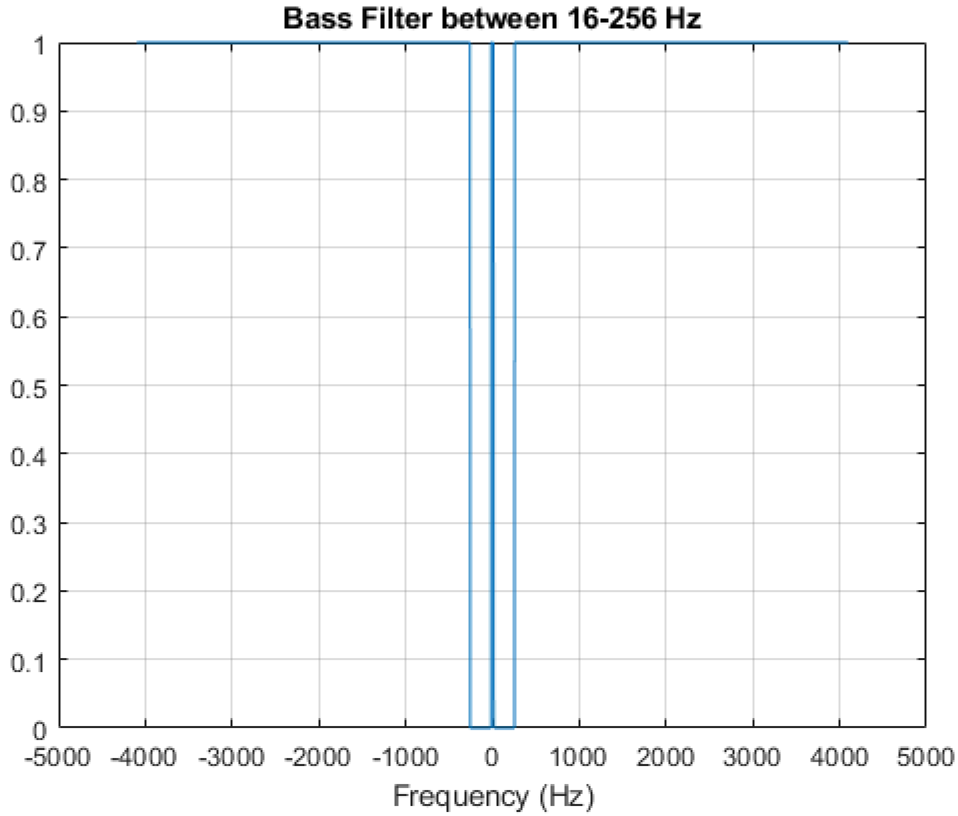
The frequencies higher than +-2kHz got removed, so parts of the song that contained those frequencies went silent.

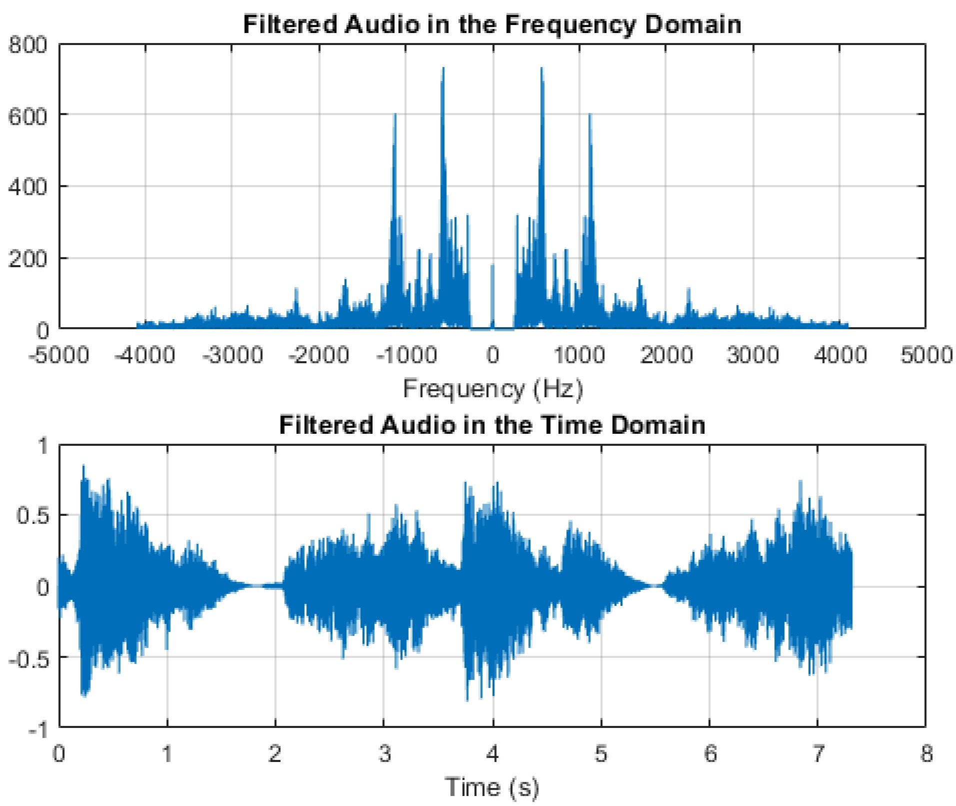
The low frequencies/bass sounds were removed, and replaced with silence
The higher frequencies of the music were amplified and got louder

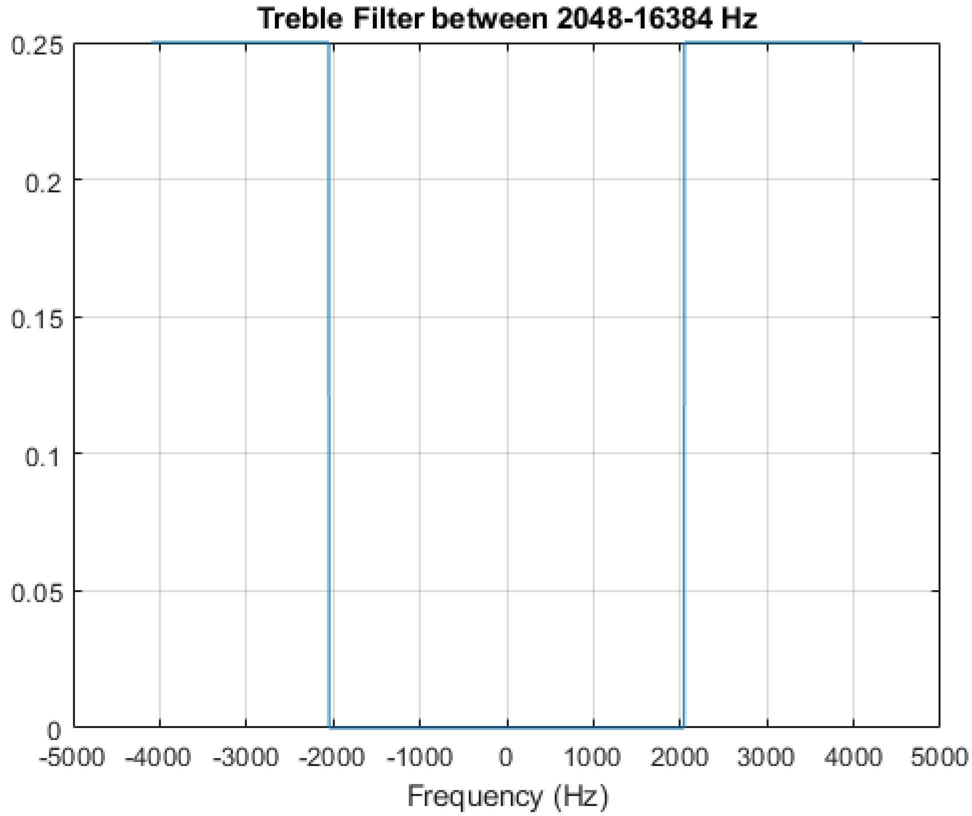


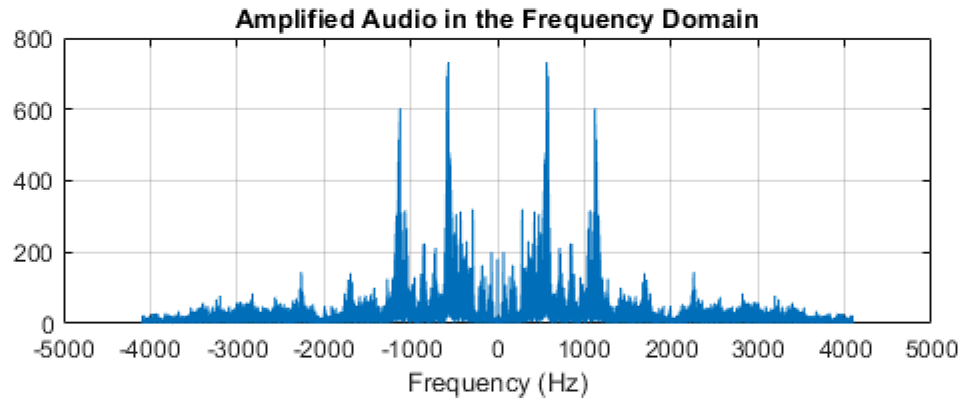

2kHz Lowpass Filter
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)

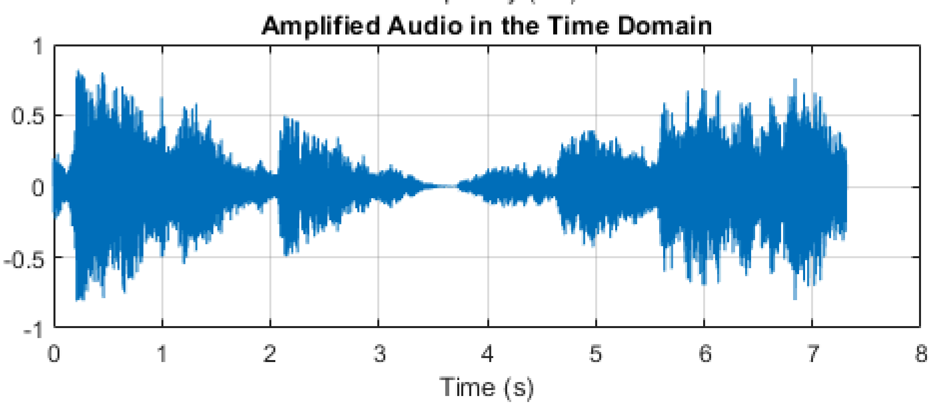

Audio in Frequency Domain from -2kHz to 2kHz
800
600
400
200
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)
Filtered Audio in Time Domain
1
0.5
0
-0.5
-1
0 1 2 3 4 5 6 7 8
Time (s)


Bass Filter between 16-256 Hz
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)


Filtered Audio in the Frequency Domain
800
600
400
200
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)
Filtered Audio in the Time Domain
1
0.5
0
-0.5
-1
0 1 2 3 4 5 6 7 8
Time (s)


Treble Filter between 2048-16384 Hz
0.25
0.2
0.15
0.1
0.05
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)


Amplified Audio in the Frequency Domain
800
600
400
200
0
-5000 -4000 -3000 -2000 -1000 0 1000 2000 3000 4000 5000
Frequency (Hz)


Amplified Audio in the Time Domain
1
0.5
0
-0.5
-1
0 1 2 3 4 5 6 7 8
Time (s)

Published with MATLAB® R2018b