AUDIO EQUALIZER

Ahmed Khaled Abdel Sayed Abdel Aal 6
Abdelrahman Ahmed Mohamed Abdelfattah Omran 37
Muhammad Elsayed Sharaf Eldeen 53

Approach

For designing the IIR filter we used the Elliptic method as It has a sharp cut-off and narrow transition width and the frequency response specifications can be satisfied by a lower order filter.

Two IIR filters were used:

1. A low pass filter

```
function Hd = IIR1(fs,fpass)
%IIR1 Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 8.4 and the Signal Processing Toolbox 6.22.
% Generated on: 21-Dec-2018 03:04:06

% Elliptic Lowpass filter designed using FDESIGN.LOWPASS.

% All frequency values are in Hz.
Fs = fs;  % Sampling Frequency

N     = 16;    % Order
Fpass = fpass;  % Passband Frequency
Apass = 1;     % Passband Ripple (dB)
Astop = 80;    % Stopband Attenuation (dB)

% Construct an FDESIGN object and call its ELLIP method.
h  = fdesign.lowpass('N,Fp,Ap,Ast', N, Fpass, Apass, Astop, Fs);
Hd = design(h, 'ellip');
```

2. A band pass filter

```
function Hd = IIR2(fs,fpass1,fpass2)
%IIR2 Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 8.4 and the Signal Processing Toolbox 6.22.
% Generated on: 21-Dec-2018 03:02:15

% Elliptic Bandpass filter designed using FDESIGN.BANDPASS.
```

```
% All frequency values are in Hz.
Fs = fs;  % Sampling Frequency

N     = 32;     % Order
Fpass1 = fpass1;   % First Passband Frequency
Fpass2 = fpass2;  % Second Passband Frequency
Apass  = 1;      % Passband Ripple (dB)
Astop  = 80;     % Stopband Attenuation (dB)

% Construct an FDESIGN object and call its ELLIP method.
h  = fdesign.bandpass('N,Fp1,Fp2,Ast1,Ap,Ast2', N, Fpass1, Fpass2, ...
                      Astop, Apass, Astop, Fs);
Hd = design(h, 'ellip');
```

For designing the FIR filter we used the window kaiser method. the major advantages of using window method is their relative simplicity as compared to other methods and ease of use also well defined equations are often available for calculating the window coefficients.And kaiser method which gives better widths at same approximation error as the passband and stopband ripple size are not affected much by the change in the window length(when compared to other windows.

Two FIR filters were used:

1. A low pass filter

```
function Hd = fir_low(Fs, Fc)
%FIR_LOW Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 9.2 and the Signal Processing Toolbox 7.4.
% Generated on: 21-Dec-2018 02:40:15

% FIR Window Lowpass filter designed using the FIR1 function.

% All frequency values are in Hz.

N    = 16;       % Order
```

```
flag = 'scale';  % Sampling Flag
Beta = 0.5;      % Window Parameter

% Create the window vector for the design algorithm.
win = kaiser(N+1, Beta);

% Calculate the coefficients using the FIR1 function.
b  = fir1(N, Fc/(Fs/2), 'low', win, flag);
Hd = dfilt.dffir(b);
```

2. A band pass filter

```
function Hd = fir_band(Fs, Fc1, Fc2)
%FIR_BAND Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 9.2 and the Signal Processing Toolbox 7.4.
% Generated on: 21-Dec-2018 02:41:58

% FIR Window Bandpass filter designed using the FIR1 function.

% All frequency values are in Hz.
N    = 16;       % Order
flag = 'scale';  % Sampling Flag
Beta = 0.5;      % Window Parameter
% Create the window vector for the design algorithm.
win = kaiser(N+1, Beta);

% Calculate the coefficients using the FIR1 function.
b  = fir1(N, [Fc1 Fc2]/(Fs/2), 'bandpass', win, flag);
Hd = dfilt.dffir(b);
```

Main Code

```
function play_equalizer(hObject, handles,isFir,sampleRate)

global player;
%contains all the frequency intervals
global yArr;
yArr=[170 310 600 1000 3000 6000 12000 14000 16000];
global handl;

%the output signal in time domain for each interval
global y1;
global y2;
global y3;
global y4;
global y5;
global y6;
global y7;
global y8;
global y9;

%the total output signal in time domain
global y10;

%the output signal in frequency domain for each interval
global yf1;
global yf2;
global yf3;
global yf4;
global yf5;
global yf6;
global yf7;
global yf8;
global yf9;

%the total output signal in frequency domain
global yf10;

%the original signal
global yOriginal;
[handles.y,handles.Fs] = audioread(handles.fullpathname);
yOriginal = handles.y;
plot(handles.y);
```

```
%getting all the amplification parameters
handl(1)=get(handles.b1,'value');
handl(2)=get(handles.b2,'value');
handl(3)=get(handles.b3,'value');
handl(4)=get(handles.b4,'value');
handl(5)=get(handles.b5,'value');
handl(6)=get(handles.b6,'value');
handl(7)=get(handles.b7,'value');
handl(8)=get(handles.b8,'value');
handl(9)=get(handles.b9,'value');


%lowpass filter for the first interval (0 -> 170 HZ)
 if(isFir)
    a=fir_low(handles.Fs,yArr(1));
 else
    a=IIR1(handles.Fs,yArr(1));
 end
 y1=handl(1)*filter(a,handles.y,1);



 %bandpass1 filter for the second interval (170 -> 310 HZ)
if(isFir)
     a=fir_band(handles.Fs,yArr(1),yArr(2));
else
       a=IIR2(handles.Fs,yArr(1),yArr(2));
end
   y2=handl( 2 )*filter(a,handles.y,1);

%bandpass filter for the third domain (310 -> 600 HZ)
if(isFir)
     a=fir_band(handles.Fs,yArr(2),yArr(3));
else
       a=IIR2(handles.Fs,yArr(2),yArr(3));
end
   y3=handl( 3 )*filter(a,handles.y,1);

%bandpass filter for the fourth domain (600 -> 1000 HZ)
if(isFir)
     a=fir_band(handles.Fs,yArr(3),yArr(4));
else
       a=IIR2(handles.Fs,yArr(3),yArr(4));
end
   y4=handl( 4 )*filter(a,handles.y,1);
```

```
%bandpass filter for the fifth domain (1 -> 3 KHZ)
if(isFir)
    a=fir_band(handles.Fs,yArr(4),yArr(5));
else
      a=IIR2(handles.Fs,yArr(4),yArr(5));
end
   y5=handl( 5 )*filter(a,handles.y,1);

%bandpass filter for the sixth domain (3 -> 6 KHZ)
if(isFir)
    a=fir_band(handles.Fs,yArr(5),yArr(6));
else
      a=IIR2(handles.Fs,yArr(5),yArr(6));
end
   y6=handl( 6 )*filter(a,handles.y,1);

%bandpass filter for the seventh domain (6 -> 12 KHZ)
if(isFir)
    a=fir_band(handles.Fs,yArr(6),yArr(7));
else
      a=IIR2(handles.Fs,yArr(6),yArr(7));
end
   y7=handl( 7 )*filter(a,handles.y,1);

%bandpass filter for the eighth domain (12 -> 14 KHZ)
if(isFir)
    a=fir_band(handles.Fs,yArr(7),yArr(8));
else
      a=IIR2(handles.Fs,yArr(7),yArr(8));
end
   y8=handl( 8 )*filter(a,handles.y,1);

%bandpass filter for the ninth domain (14 -> 16 KHZ)
if(isFir)
    a=fir_band(handles.Fs,yArr(8),yArr(9));
else
      a=IIR2(handles.Fs,yArr(8),yArr(9));
end
   y9=handl( 9 )*filter(a,handles.y,1);


 %calculating the total signal
 handles.yT=y1+y2+y3+y4+y5+y6+y7+y8+y9;

 %check if the output sample rate is not 1
 if(strcmp(sampleRate,'0.5'))
```

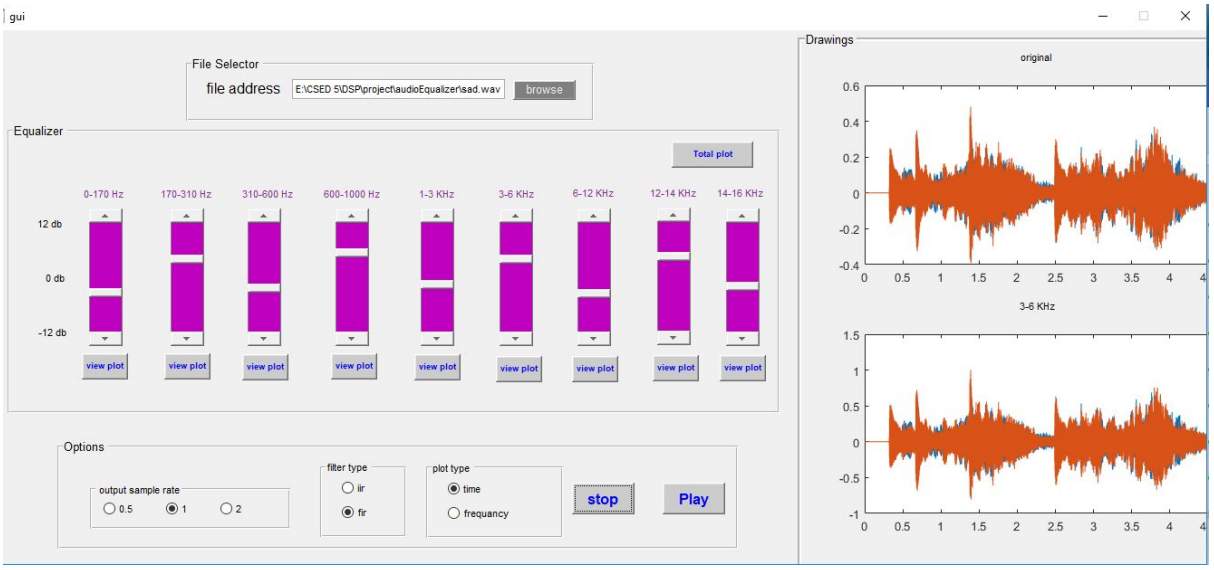
```
    handles.yT= resample(handles.yT,1,2);
else
    if(strcmp(sampleRate,'2'))
        handles.yT= resample(handles.yT,2,1);
    end
end

y10=handles.yT;
% calculating the signals in frequency domain.
yf1=abs(fft(y1));
yf2=abs(fft(y2));
yf3=abs(fft(y3));
yf4=abs(fft(y4));
yf5=abs(fft(y5));
yf6=abs(fft(y6));
yf7=abs(fft(y7));
yf8=abs(fft(y8));
yf9=abs(fft(y9));

yf10=abs(fft( handles.yT));

player = audioplayer(handles.yT, handles.Fs);

%plotting the signal in time domain
subplot(2,1,1);
plot(handles.y);
subplot(2,1,2);
plot(handles.yT);
guidata(hObject,handles)

%==========================================================

% --- Executes on button press in browse.
function browse_Callback(hObject, eventdata, handles)
% hObject    handle to browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[filename, pathname] = uigetfile({'*.wav'},'File Selector');
handles.fullpathname = strcat(pathname,filename);
set(handles.address,'String',handles.fullpathname)
guidata(hObject,handles)

% --- Executes on button press in Play.
function Play_Callback(hObject, eventdata, handles)
% hObject    handle to Play (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
global player;
selectedOsr=get(handles.osr,'SelectedObject');
Osr=get(selectedOsr,'String');
selected=get(handles.filterPanel,'SelectedObject');
filterType=get(selected,'String');
play_equalizer(hObject,handles,strcmp(filterType,'fir'),Osr); % osr = 1 ,2 , 0.5
play(player);
guidata(hObject,handles)

% --- Executes on button press in pause.
function pause_Callback(hObject, eventdata, handles)
% hObject    handle to pause (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global player;
%play_equalizer(hObject,handles,strcmp(filterType,fir));
stop(player);
guidata(hObject,handles)

% --- Executes on button press in bpT.
function bpT_Callback(hObject, eventdata, handles)
% hObject    handle to bpT (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global y10;
global yf10;
global yOriginal;
set(handles.secPlotLbl,'String','filtered Tolal')
subplot(2,1,2);
selected=get(handles.plotType,'SelectedObject');
plotType=get(selected,'String');
if strcmp(plotType,'frequancy')
 plot(yf10);
 subplot(2,1,1);
 plot(abs(fft(yOriginal)));
else
 plot(y10);
 subplot(2,1,1);
 plot((yOriginal));
end

function plotBoth(handles,y10,yf10,handlN,str)
global handl
set(handles.secPlotLbl,'String',str)
selected=get(handles.plotType,'SelectedObject');
```

```
plotType=get(selected,'String');
if strcmp(plotType,'frequancy')
 subplot(2,1,2);
 plot(yf10);
 subplot(2,1,1);
 plot(abs(fft(y10/handl(handlN))));
else
  subplot(2,1,2);
 plot(y10);
 subplot(2,1,1);
 plot((y10/handl(handlN)));
end


% --- Executes during object creation, after setting all properties.
function plotType_CreateFcn(hObject, eventdata, handles)
% hObject    handle to plotType (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

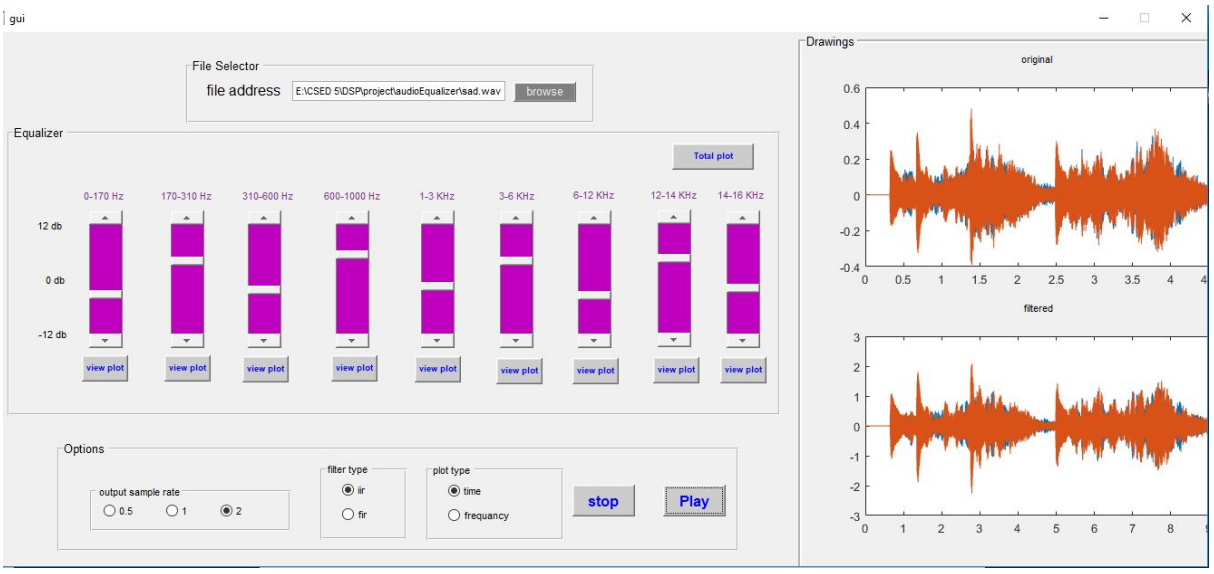
Sample Runs

1. Using FIR filter

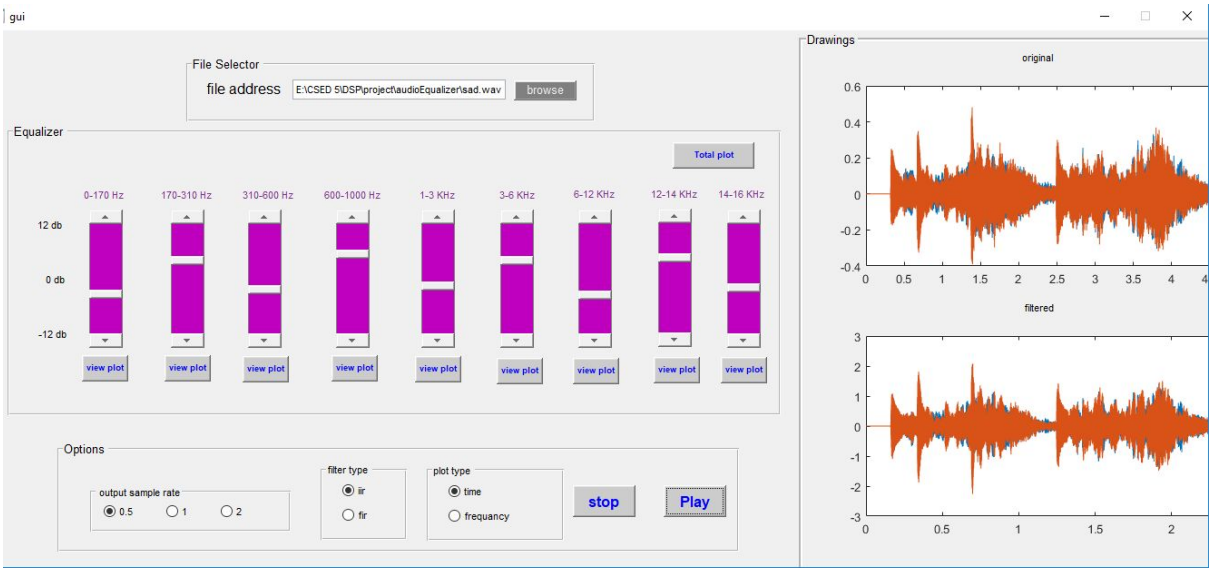


2. Using IIR filter

3. doubling output sample rate

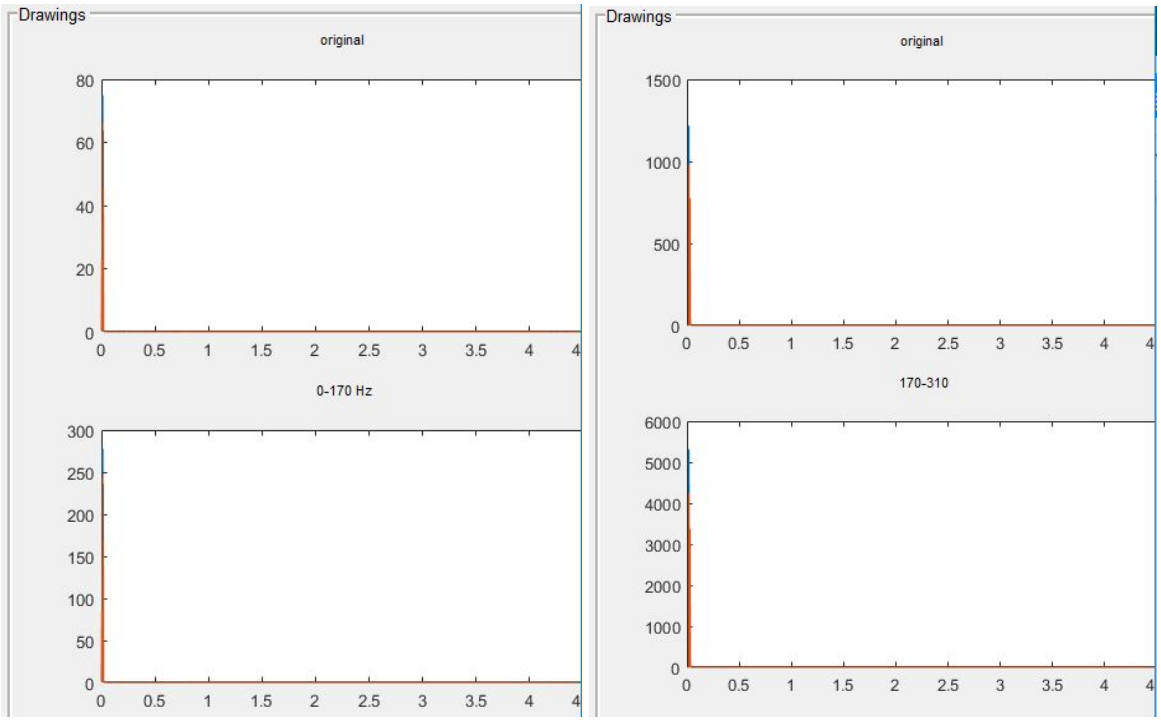


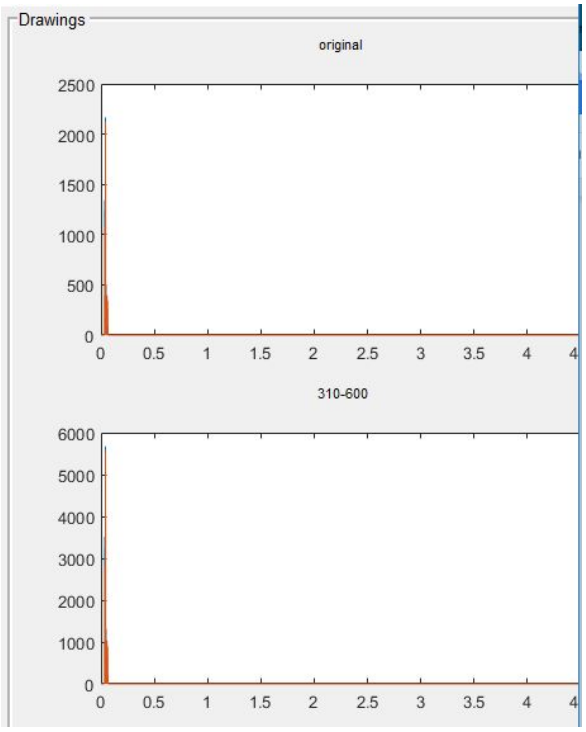
4. decreasing output sample rate to half

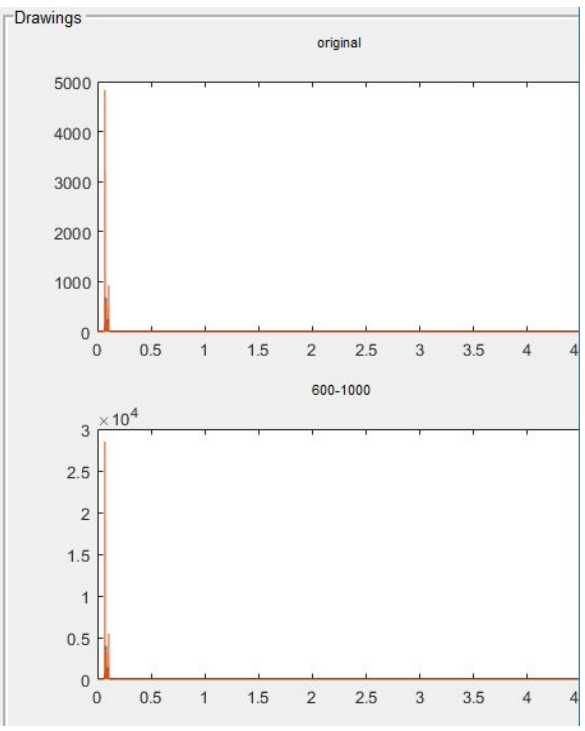


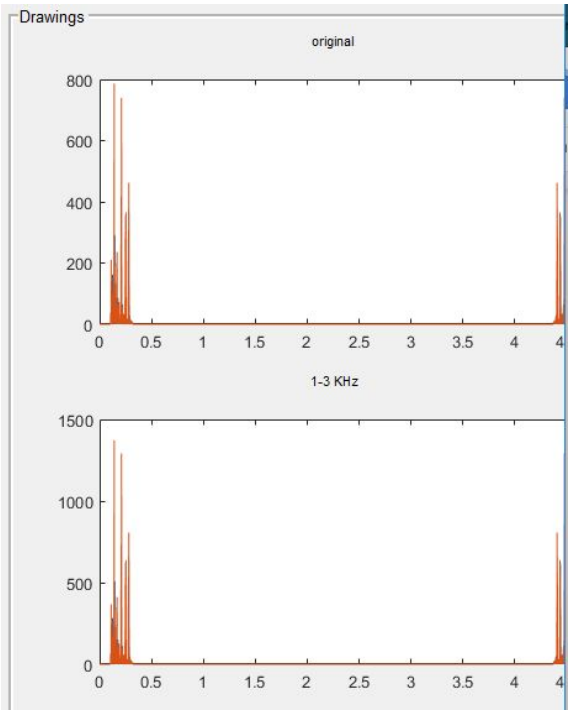
Frequency Plots Sample

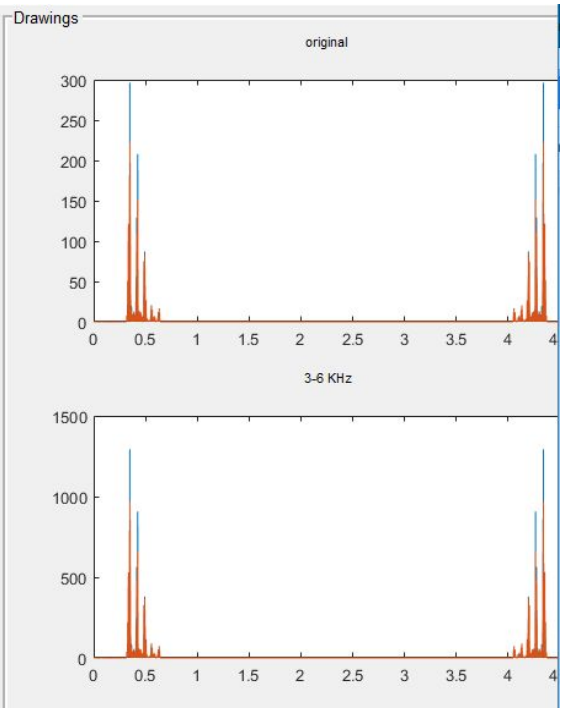
Comment: Low frequencies dosen’t appear because of the very big scale

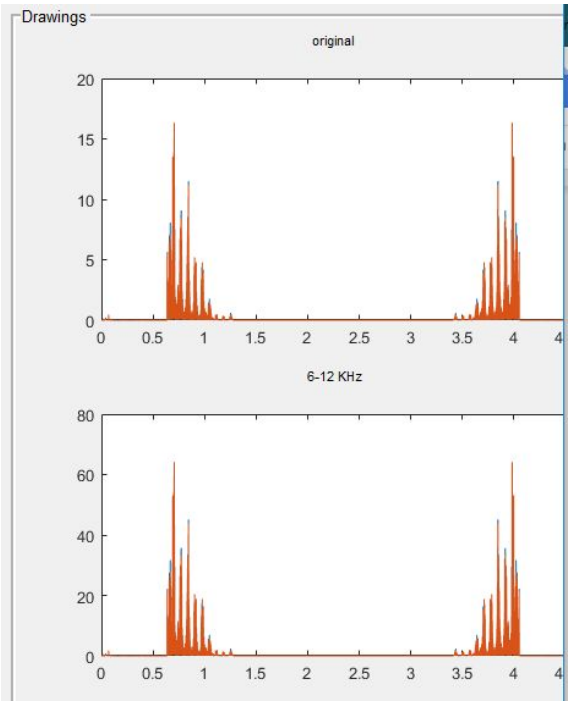


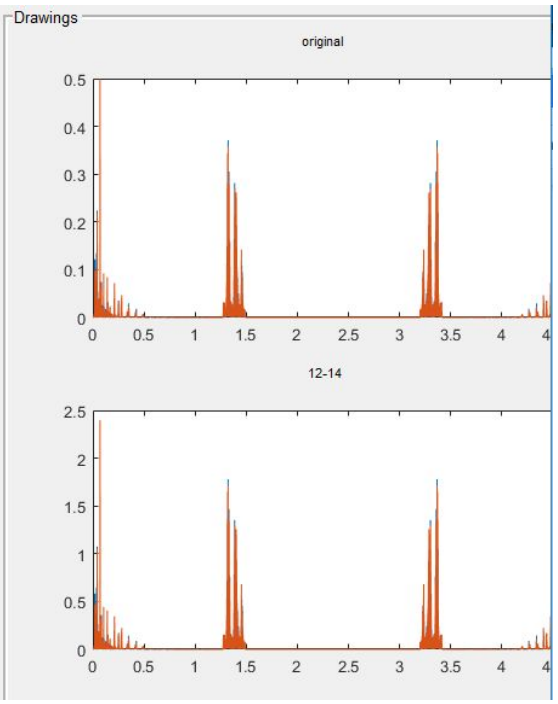

Drawings
original
2500
2000
1500
1000
500
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4
310-600
6000
5000
4000
3000
2000
1000
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4

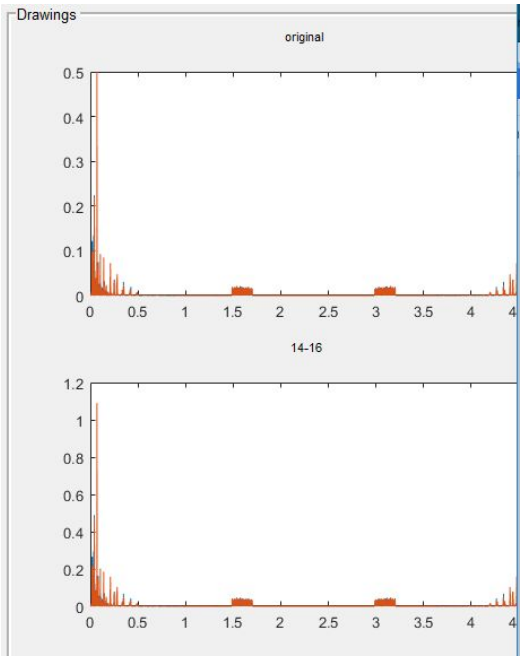

Drawings
original
5000
4000
3000
2000
1000
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4
600-1000
×10^4
3
2.5
2
1.5
1
0.5
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4


Drawings
original
800
600
400
200
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4
1-3 KHz
1500
1000
500
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4


Drawings
original
300
250
200
150
100
50
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4
3-6 KHz
1500
1000
500
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4


Drawings
original
6-12 KHz


Drawings
original
12-14


Drawings
original
14-16