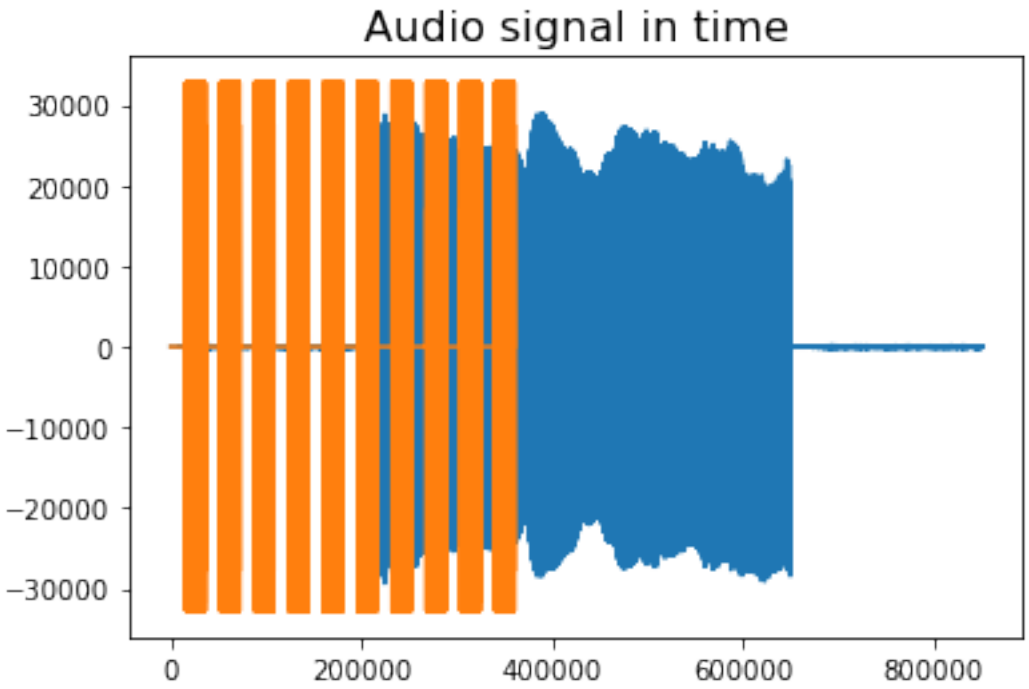
PSD_Zhengkun Ye

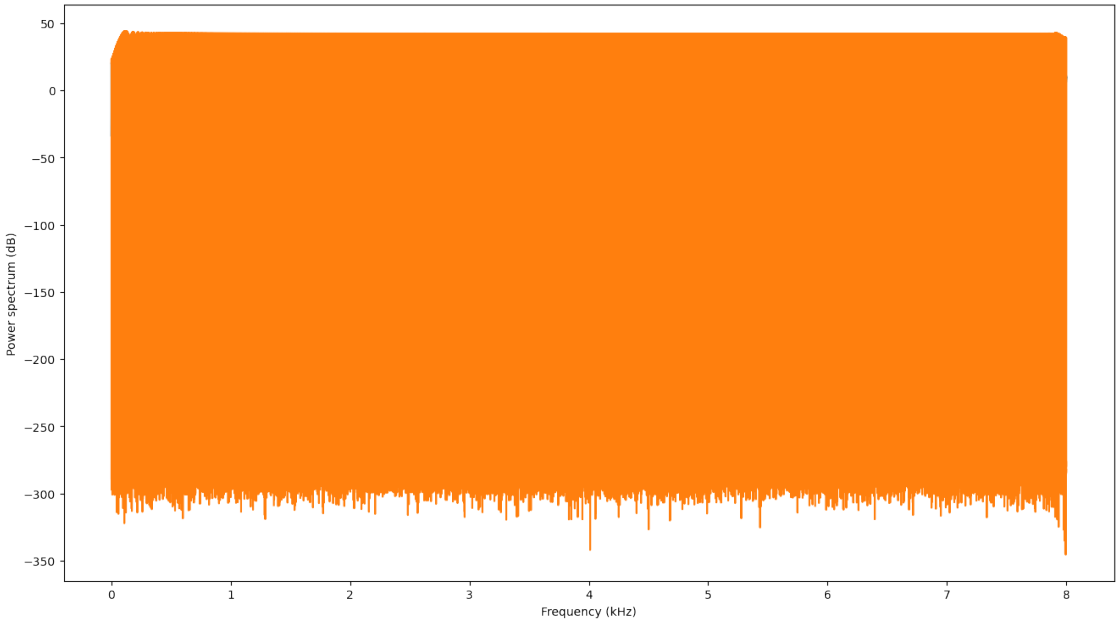
December 3, 2021

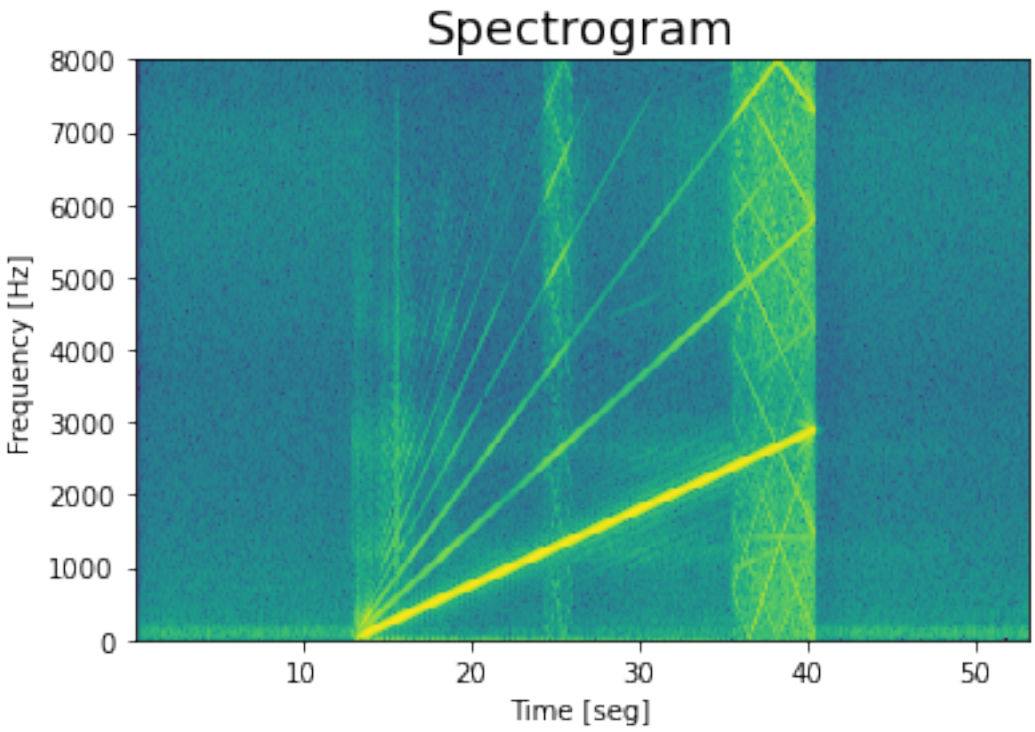
[2]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

AudioName2 = "C://Users/yzhengk/Desktop/
 ↪chirp_50hzto8khz_16khzfs_125ms_100ms_repeat10.wav" # Audio File
AudioName1 = "C://Users/yzhengk/Desktop/
 ↪chirp_50hzto8khz_16khzf_125ms_100ms_repeat10_Cell Phone Non Pressure.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)

n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2
```

```
MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');


#Spectrogram
from scipy import signal
N = 512 #Number of point in the fft
f, t, Sxx = signal.spectrogram(Audiodata1, fs,window = signal.
 ↪blackman(N),nfft=N)
plt.figure()
plt.pcolormesh(t, f,10*np.log10(Sxx)) # dB spectrogram
#plt.pcolormesh(t, f,Sxx) # Lineal spectrogram
plt.ylabel('Frequency [Hz]')
plt.xlabel('Time [seg]')
plt.title('Spectrogram',size=18);

plt.show()
```

```
<ipython-input-2-9c1d730de87f>:47: RuntimeWarning: divide by zero encountered in
log10
  plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
<ipython-input-2-9c1d730de87f>:56: RuntimeWarning: divide by zero encountered in
log10
  plt.pcolormesh(t, f,10*np.log10(Sxx)) # dB spectrogram
<ipython-input-2-9c1d730de87f>:56: MatplotlibDeprecationWarning: shading='flat'
when X and Y have the same dimensions as C is deprecated since 3.3.  Either
specify the corners of the quadrilaterals with X and Y, or pass shading='auto',
'nearest' or 'gouraud', or set rcParams['pcolor.shading'].  This will become an
error two minor releases later.
  plt.pcolormesh(t, f,10*np.log10(Sxx)) # dB spectrogram
```

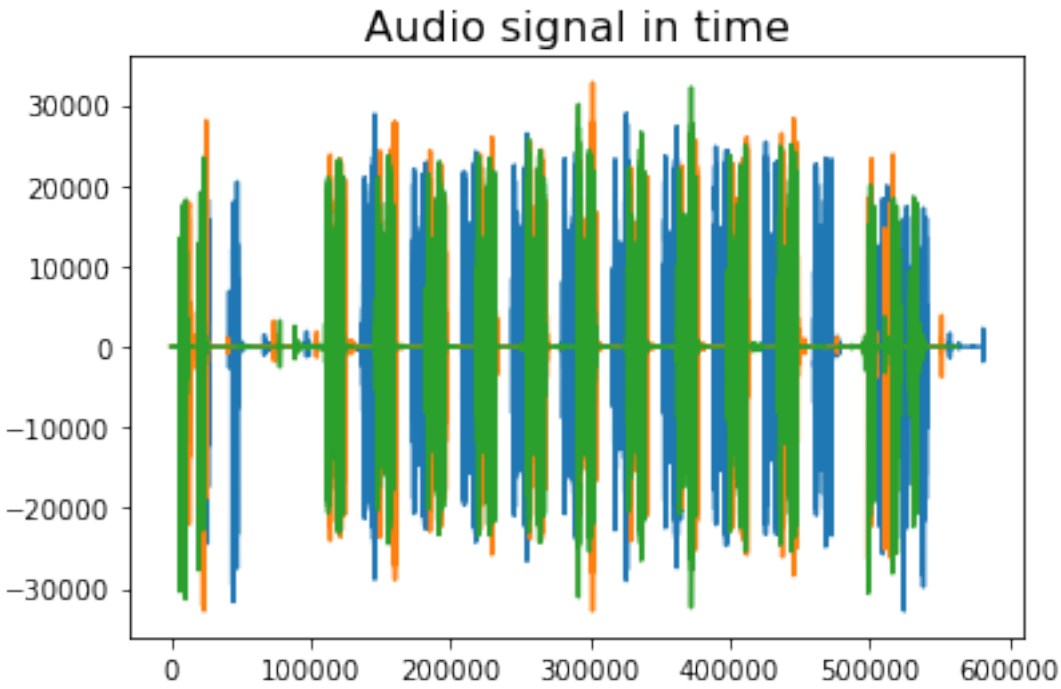

Audio signal in time
30000
20000
10000
0
−10000
−20000
−30000
0
200000
400000
600000
800000

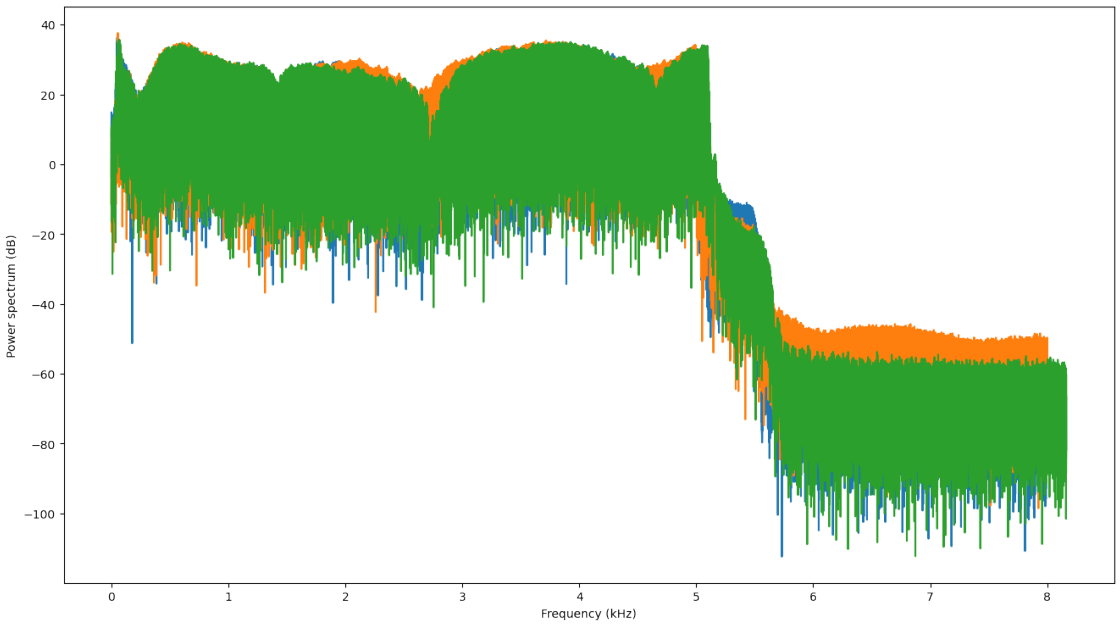

50
0
−50
−100
−150
−200
−250
−300
−350
Power spectrum (dB)
0
1
2
3
4
5
6
7
8
Frequency (kHz)



[4]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_1.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_2.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_3.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
```

```
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)

n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2

MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```

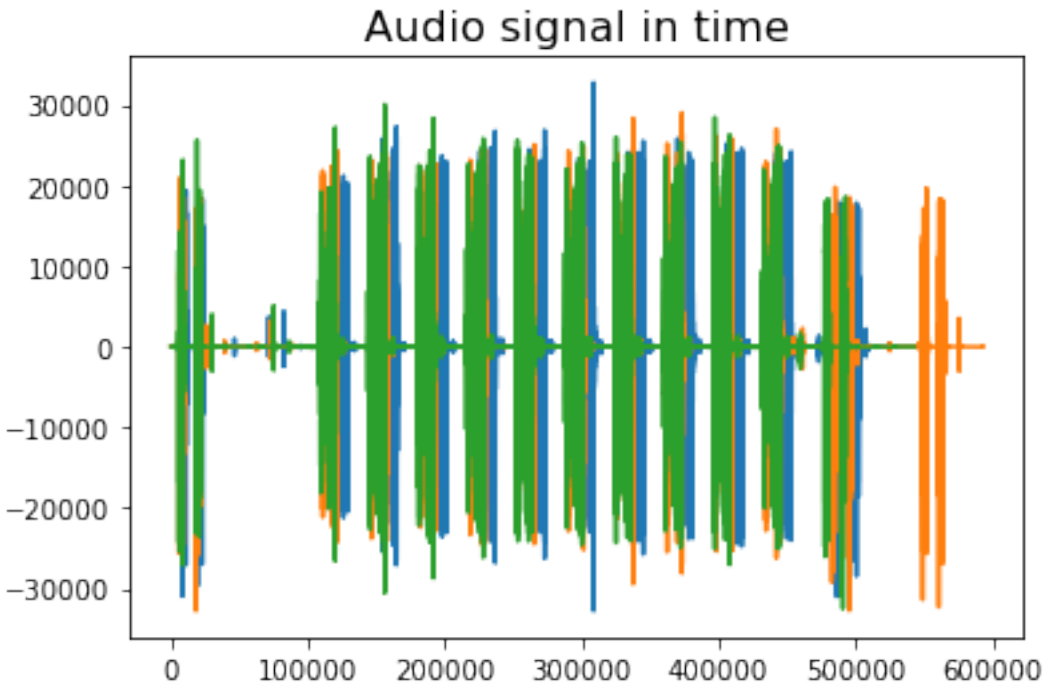


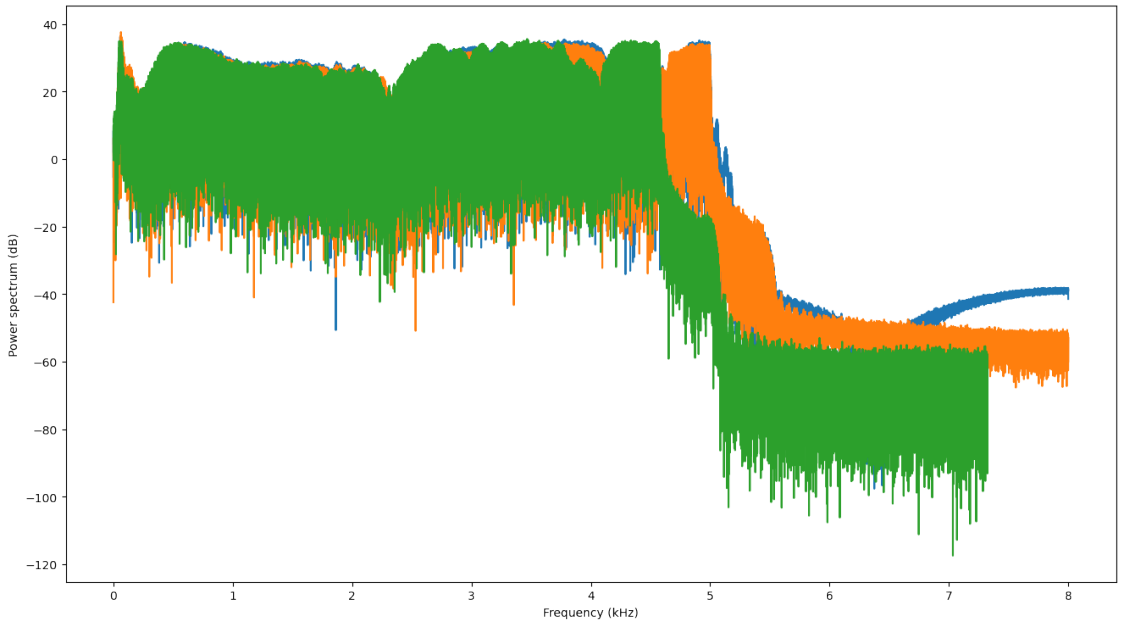


[8]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
```

```
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_1.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_2.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_3.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)

n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2
```

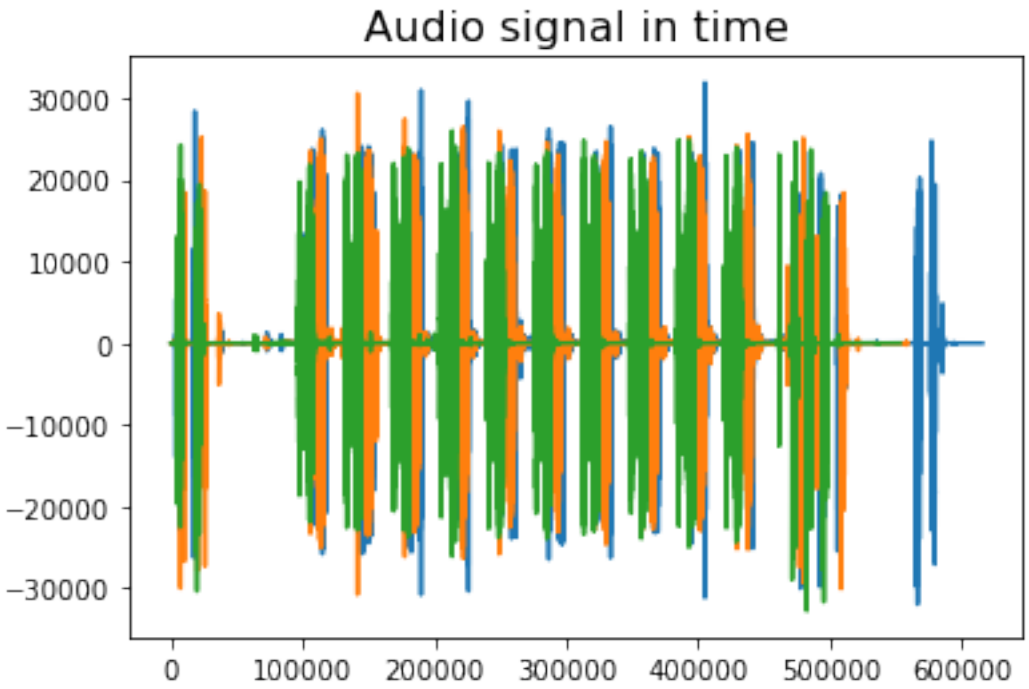
```
MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```

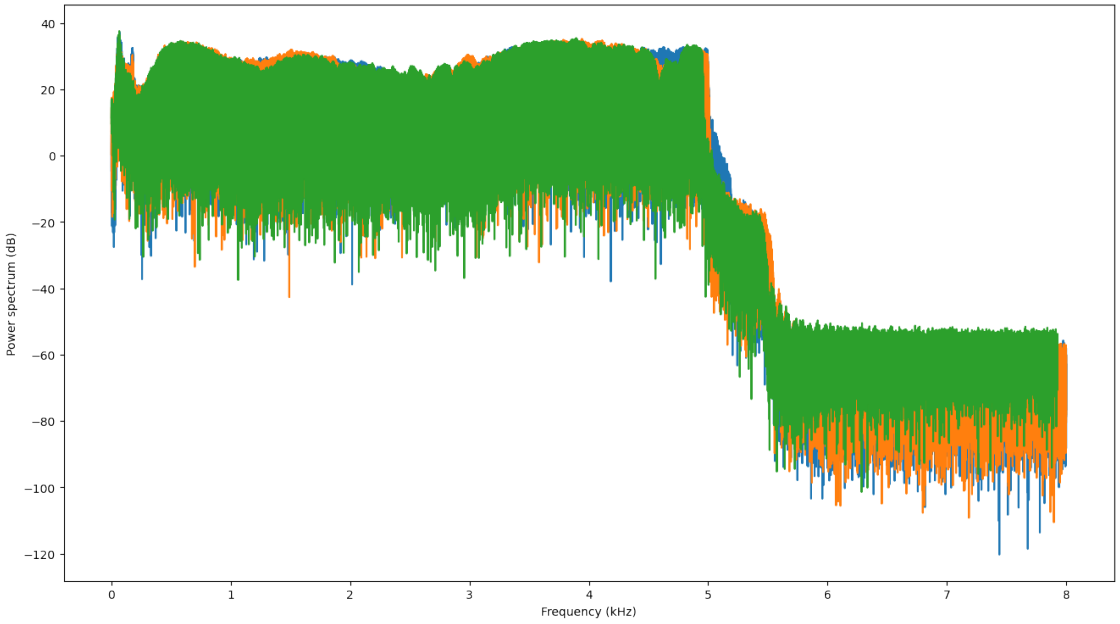




[9]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_1.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_2.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_3.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)
```

```
n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2

MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```

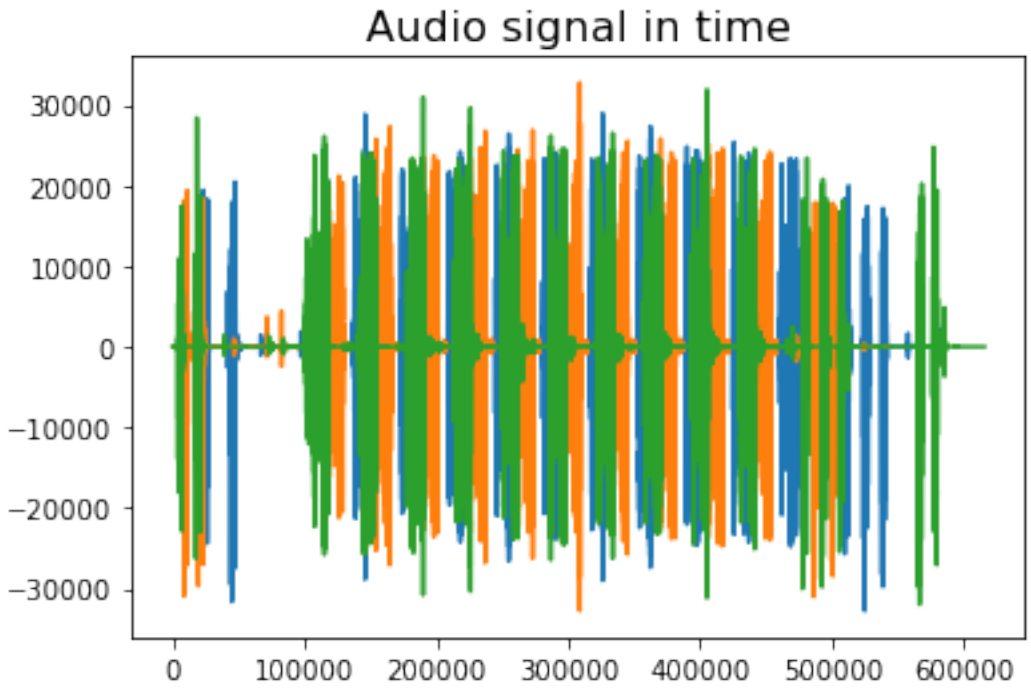


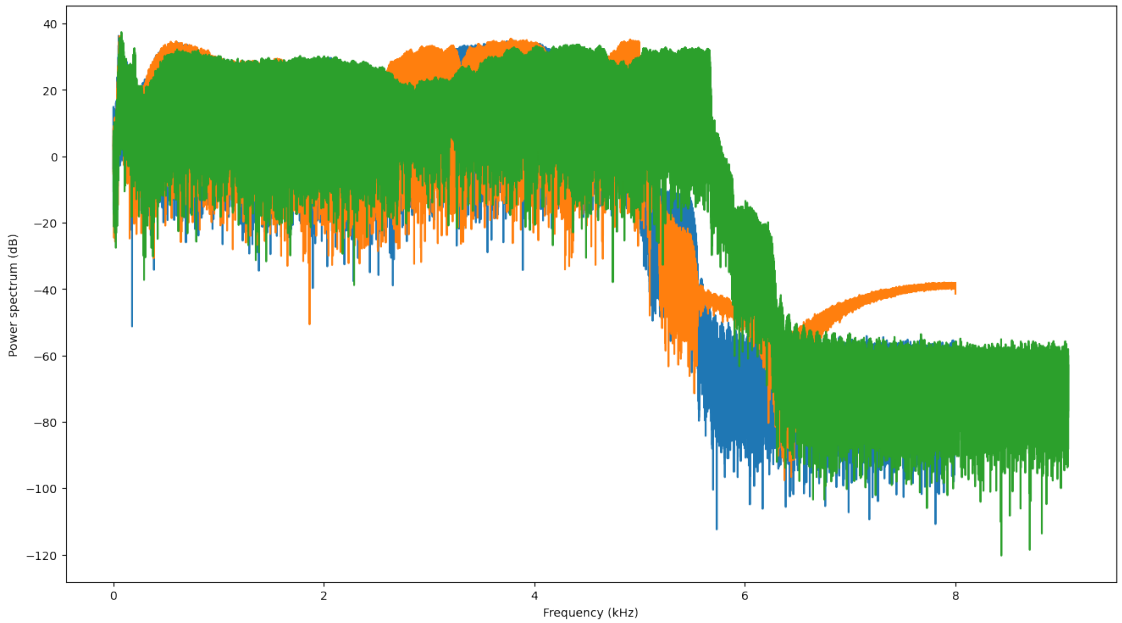


[5]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
```

```
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_1.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_1.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_1.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)

n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2
```

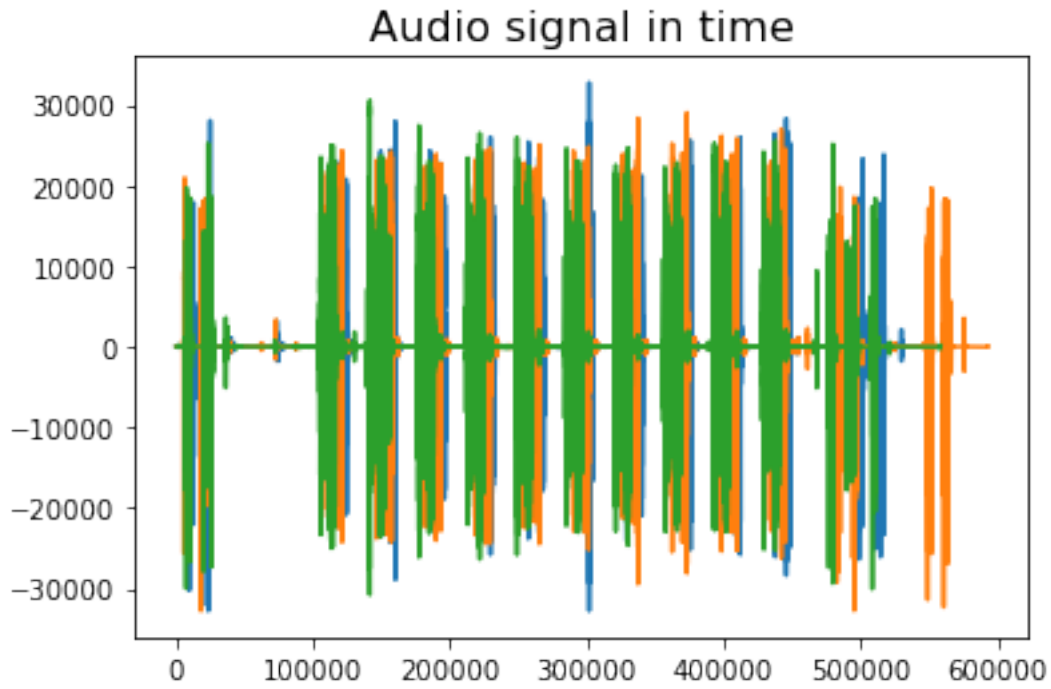
```
MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```

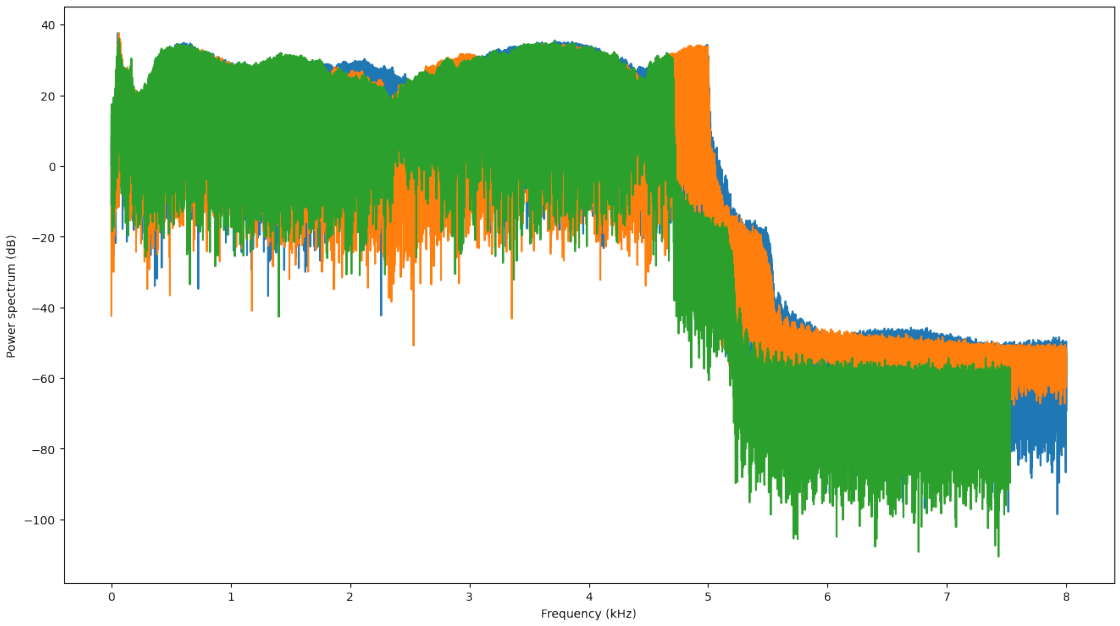




[6]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_2.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_2.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_2.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)
```

```
n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2

MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```

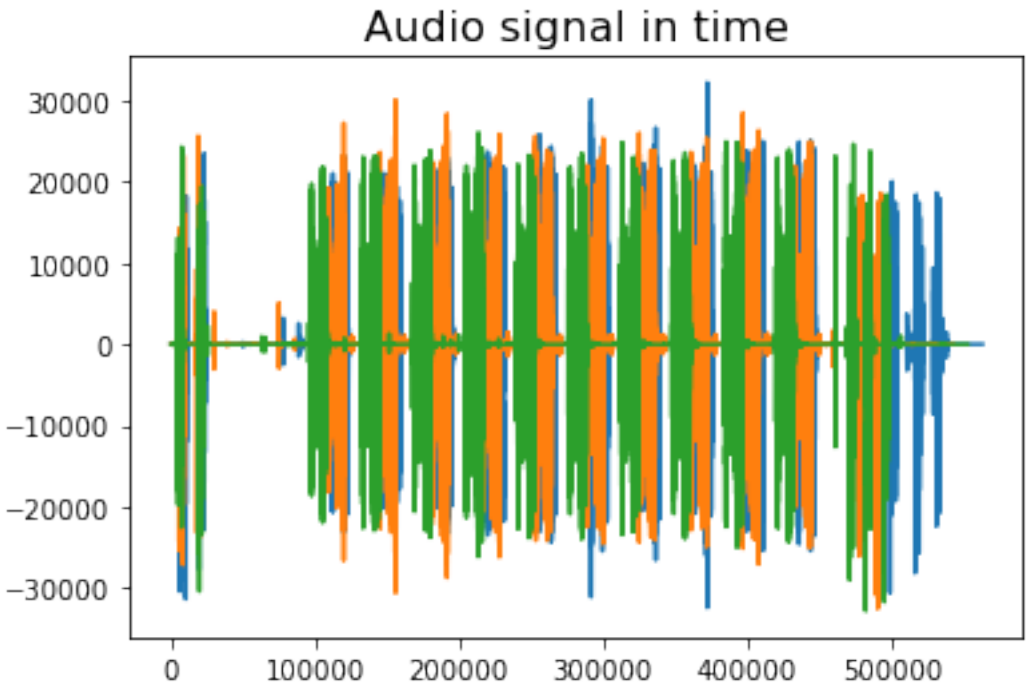


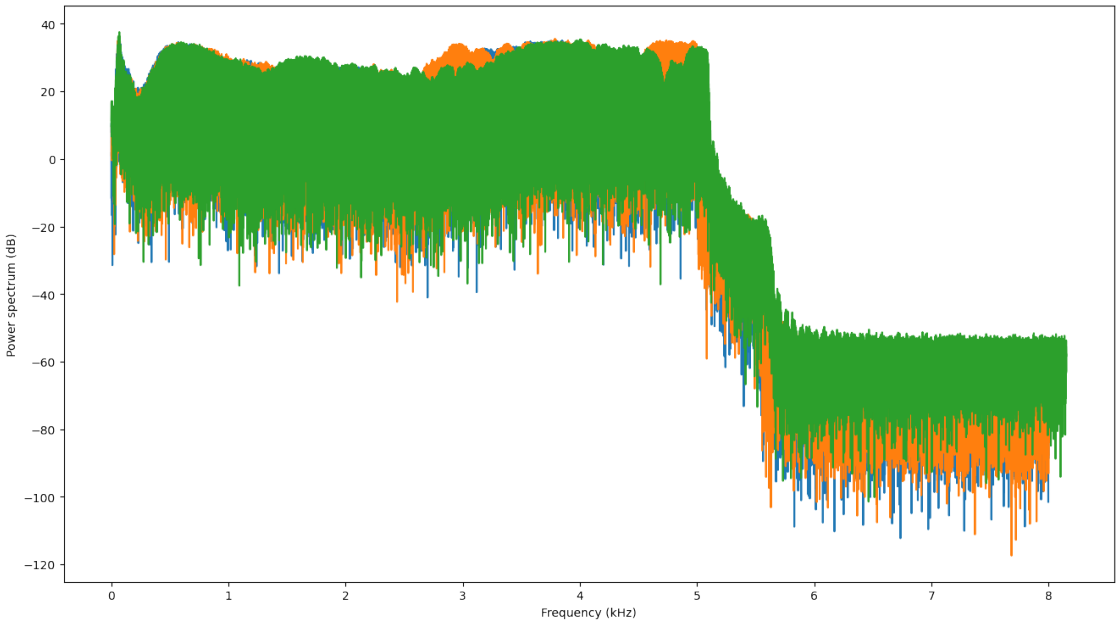


[7]:
```
from scipy.io import wavfile # scipy library to read wav files
import numpy as np

# Audio File
```

```
AudioName1 = "X://Ye/Smart Watch Authentication/12.03.2021/NonPress_3.wav"
AudioName2 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Top_3.wav"
AudioName3 = "X://Ye/Smart Watch Authentication/12.03.2021/Press Side_3.wav"
fs, Audiodata1 = wavfile.read(AudioName1)
fs, Audiodata2 = wavfile.read(AudioName2)
fs, Audiodata3 = wavfile.read(AudioName3)

# Plot the audio signal in time
import matplotlib.pyplot as plt
plt.plot(Audiodata1)
plt.plot(Audiodata2)
plt.plot(Audiodata3)
plt.title('Audio signal in time',size=16)

# spectrum
from scipy.fftpack import fft # fourier transform

n1 = len(Audiodata1)
AudioFreq1 = fft(Audiodata1)
AudioFreq1 = AudioFreq1[0:int(np.ceil((n1+1)/2.0))] #Half of the spectrum
MagFreq1 = np.abs(AudioFreq1) # Magnitude
MagFreq1 = MagFreq1 / float(n1)

n2 = len(Audiodata2)
AudioFreq2 = fft(Audiodata2)
AudioFreq2 = AudioFreq2[0:int(np.ceil((n2+1)/2.0))] #Half of the spectrum
MagFreq2 = np.abs(AudioFreq2) # Magnitude
MagFreq2 = MagFreq2 / float(n2)

n3 = len(Audiodata3)
AudioFreq3 = fft(Audiodata3)
AudioFreq3 = AudioFreq3[0:int(np.ceil((n3+1)/2.0))] #Half of the spectrum
MagFreq3 = np.abs(AudioFreq3) # Magnitude
MagFreq3 = MagFreq3 / float(n3)

# power spectrum
MagFreq1 = MagFreq1**2
if n1 % 2 > 0: # ffte odd
    MagFreq1[1:len(MagFreq1)] = MagFreq1[1:len(MagFreq1)] * 2
else:# fft even
    MagFreq1[1:len(MagFreq1) -1] = MagFreq1[1:len(MagFreq1) - 1] * 2

MagFreq2 = MagFreq2**2
if n2 % 2 > 0: # ffte odd
    MagFreq2[1:len(MagFreq2)] = MagFreq2[1:len(MagFreq2)] * 2
else:# fft even
    MagFreq2[1:len(MagFreq2) -1] = MagFreq2[1:len(MagFreq2) - 1] * 2
```

```
MagFreq3 = MagFreq3**2
if n3 % 2 > 0: # ffte odd
    MagFreq3[1:len(MagFreq3)] = MagFreq3[1:len(MagFreq3)] * 2
else:# fft even
    MagFreq3[1:len(MagFreq3) -1] = MagFreq3[1:len(MagFreq3) - 1] * 2

plt.figure(figsize=(16, 9), dpi=100)
freqAxis1 = np.arange(0,int(np.ceil((n1+1)/2.0)), 1.0) * (fs / n1)
freqAxis2 = np.arange(0,int(np.ceil((n2+1)/2.0)), 1.0) * (fs / n2);
freqAxis3 = np.arange(0,int(np.ceil((n3+1)/2.0)), 1.0) * (fs / n2);
plt.plot(freqAxis1/1000.0, 10*np.log10(MagFreq1)) #Power spectrum
plt.plot(freqAxis2/1000.0, 10*np.log10(MagFreq2)) #Power spectrum
plt.plot(freqAxis3/1000.0, 10*np.log10(MagFreq3)) #Power spectrum
plt.xlabel('Frequency (kHz)'); plt.ylabel('Power spectrum (dB)');
```




Power spectrum (dB)
40
20
0
−20
−40
−60
−80
−100
−120
0
1
2
3
4
5
6
7
8
Frequency (kHz)

[ ]: