In [3]:

```
# importing libraries
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.style as ms
ms.use('seaborn-muted')
%matplotlib inline
import IPython.display
import librosa
import librosa.display
import essentia
import essentia.standard as es
```

In [208]:

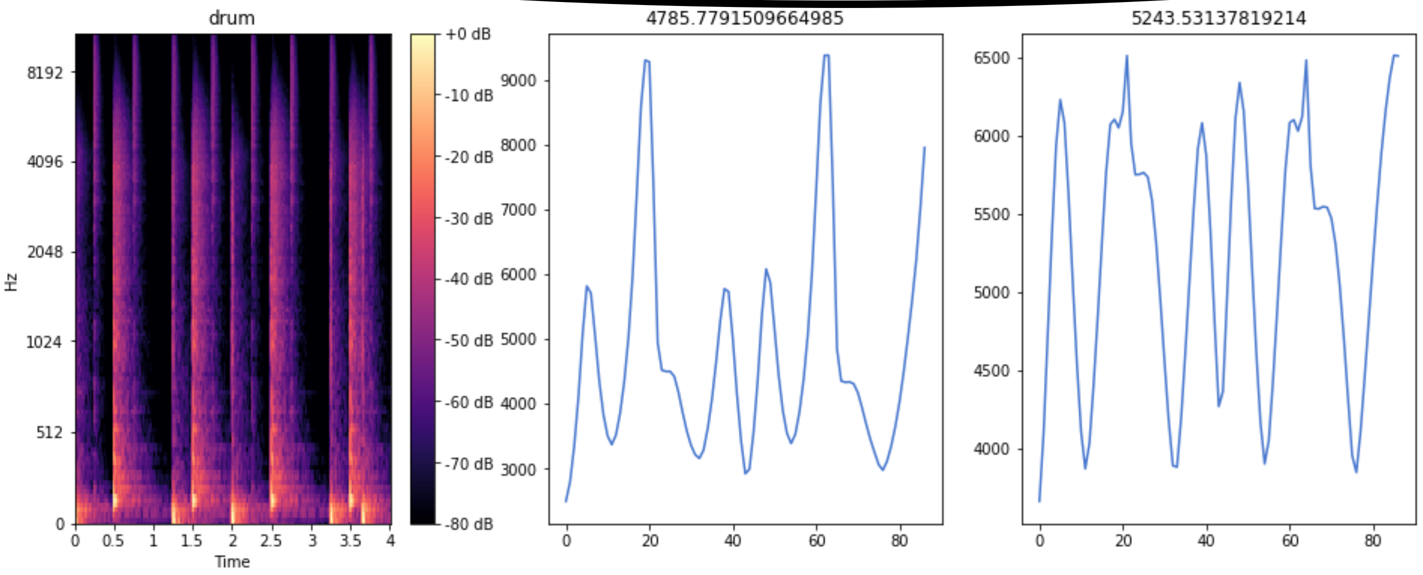
```
sr = 44100
#figure out dynamic array, here it works since same size
track = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
sc = np.zeros((13,87))
sca = np.zeros(np.shape(tracks))
sb = np.zeros((13,87))
sba = np.zeros(np.shape(tracks))
sb_ = np.zeros((13,87))
sba_ = np.zeros(np.shape(tracks))
tracks = ['drum','beat','bass','organ', 'epiano','piano','gui_ac','
gui_st','clean','str','voc_f','voc_m','voc_h']
cen = es.SpectralCentroidTime()
for t in range(0,len(tracks)):

    track[t],sr = librosa.load('../samples/phrases1/'+tracks[t]+'.w
av', sr = sr, mono = 'True')

    plt.figure(figsize=(16,6))
#     plt.subplot(1,3,1)
#     librosa.display.waveplot(track[t])
#     plt.title(tracks[t])
#     plt.title(tracks[t])


    S = librosa.feature.melspectrogram(track[t], sr=sr, n_mels=128)
    log_S = librosa.power_to_db(S, ref=np.max)
    plt.subplot(1,3,1)
    librosa.display.specshow(log_S, sr=sr, x_axis='time', y_axis='m
el')
    plt.title(tracks[t])
    plt.colorbar(format='%+02.0f dB')

    sc[t] = librosa.feature.spectral_centroid(y=track[t], sr=sr, S=
None, n_fft=2**15, hop_length=2**11, freq=None)
    sca[t] = np.average(sc[t])
    sb[t] = librosa.feature.spectral_bandwidth(y=track[t], sr=sr, S
=None, n_fft=2**15, hop_length=2**11, freq=None,centroid=None, norm
=True, p=2)
    sba[t] = np.average(sb[t])
    p=1
    sb_[t] = librosa.feature.spectral_bandwidth(y=track[t], sr=sr,
S=None, n_fft=2**15, hop_length=2**11, freq=None,centroid=None, nor
m=True, p=p)
    sba_[t] = np.average(sb_[t])
    print tracks[t],'\t\t',sca[t],'\t',sba[t],'\t',sba_[t],'\t',cen
(track[t])
    print '\n'
    plt.subplot(1,3,2)
    plt.plot(sc[t])
    plt.title(sca[t])

    plt.subplot(1,3,3)
    plt.plot(sb[t])
    plt.title(sba[t])
```

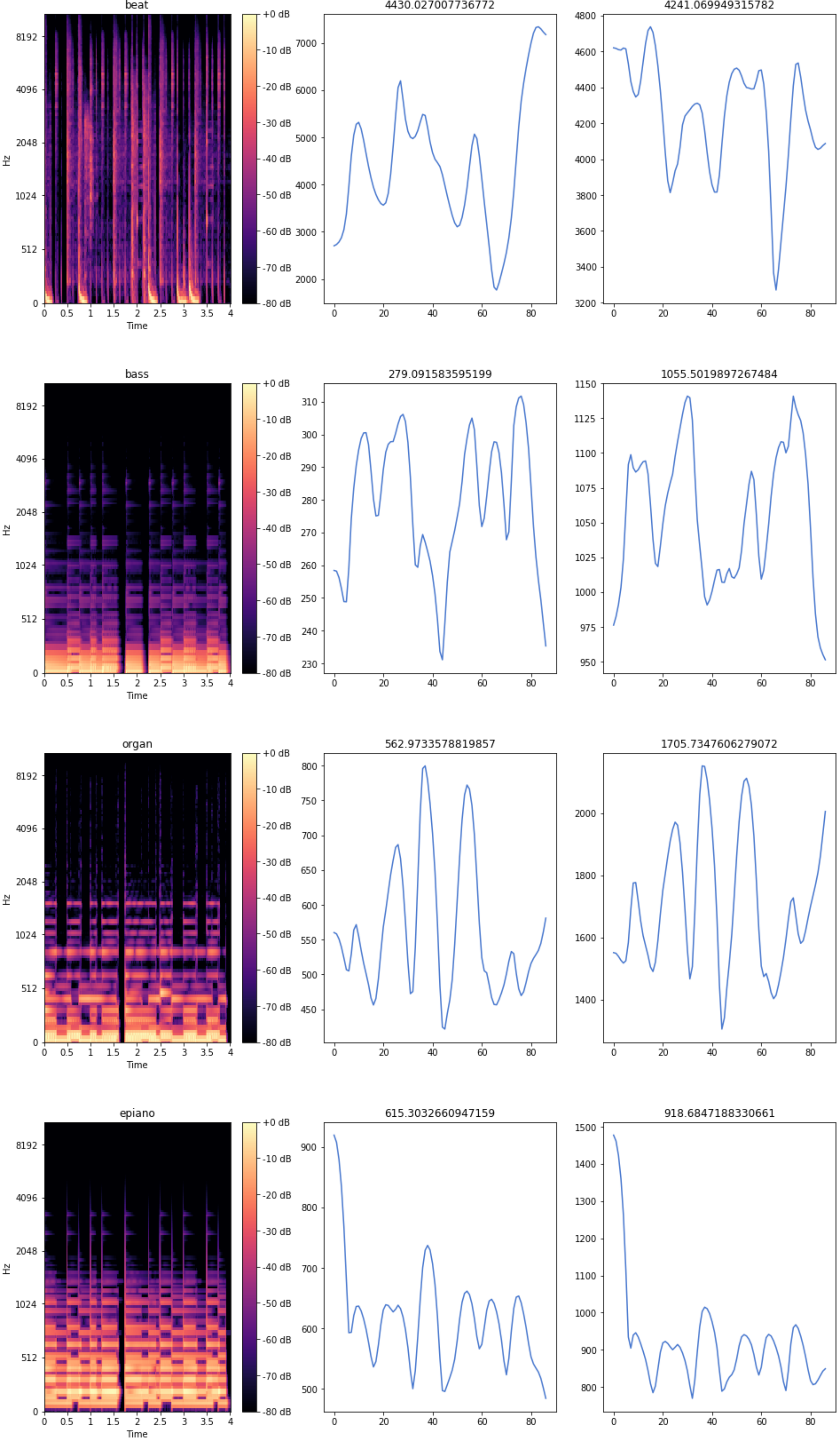
```
drum            4785.7791509664985
5243.53137819214        4248.831469865706
1051.32580566
```

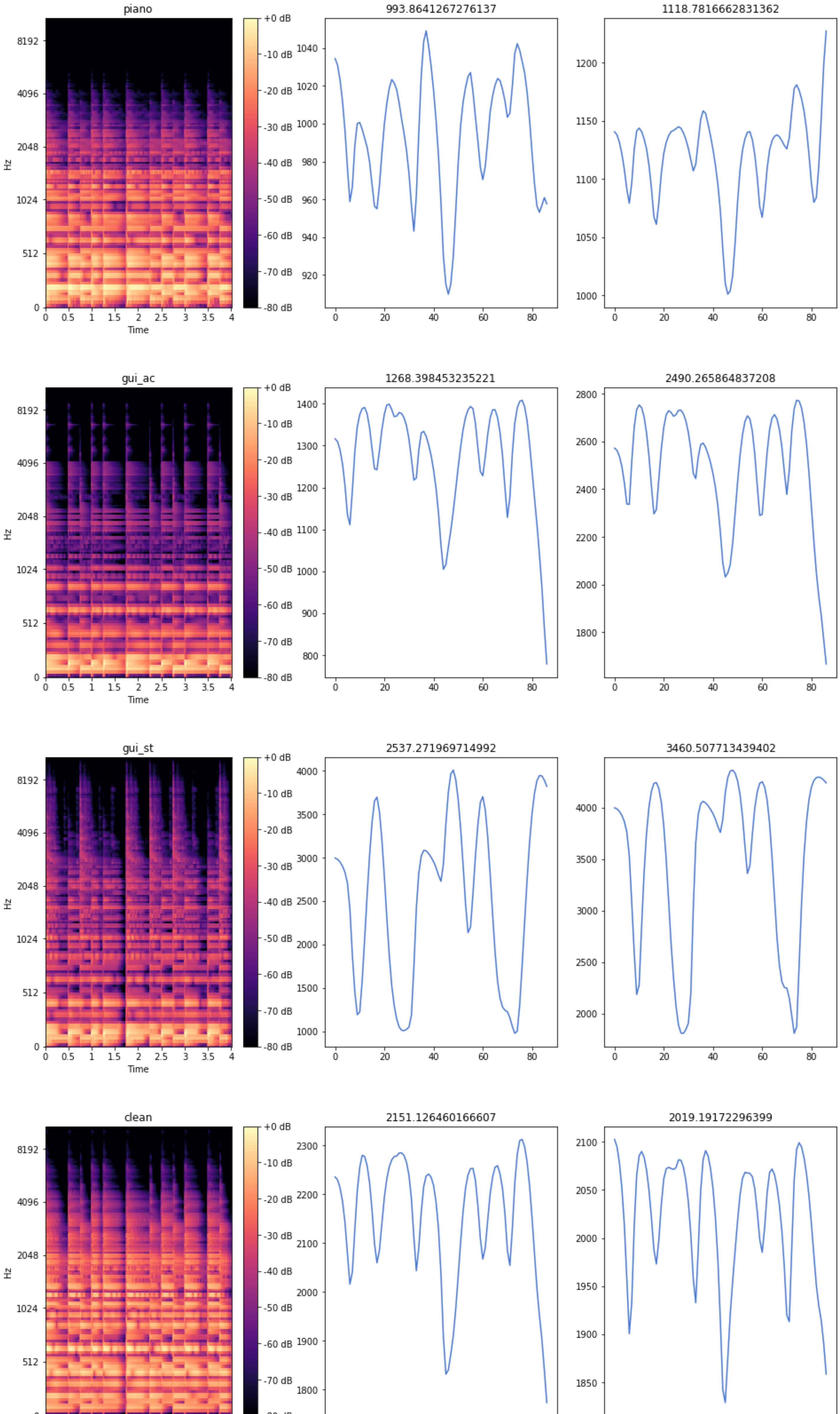
```
beat            4430.027007736772
4241.069949315782       3355.7721603902473
1267.13818359



bass            279.091583595199
1055.5019897267484      311.9865198288378
104.82459259



organ           562.9733578819857
1705.7347606279072      686.7386678436246
173.24861145



epiano          615.3032660947159
918.6847188330661       429.75688001503806

355.127929688



piano           993.8641267276137
1118.7816662831362      778.78008852033
591.545715332



gui_ac          1268.398453235221
2490.265864837208       1494.4163622738388
361.122375488



gui_st          2537.271969714992
3460.507713439402       2549.199143521468
710.729553223
```

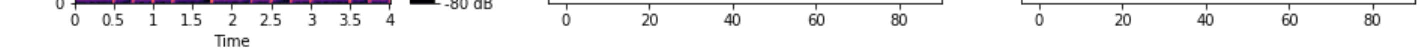
```
clean            2151.126460166607
2019.19172296399        1436.7991689984572
1391.41589355



str              1333.0583504574831
2063.383165332822       1307.4335106440647
320.568908691



voc_f            3753.915359391293
4145.100972937142       3112.2522977035615
1695.33508301



voc_m            4570.797097454433
4261.18059841652        3277.5207783013407
2228.99023438



voc_h            3929.553271733003
4157.492324864263       3392.16865404368
1322.15539551
```

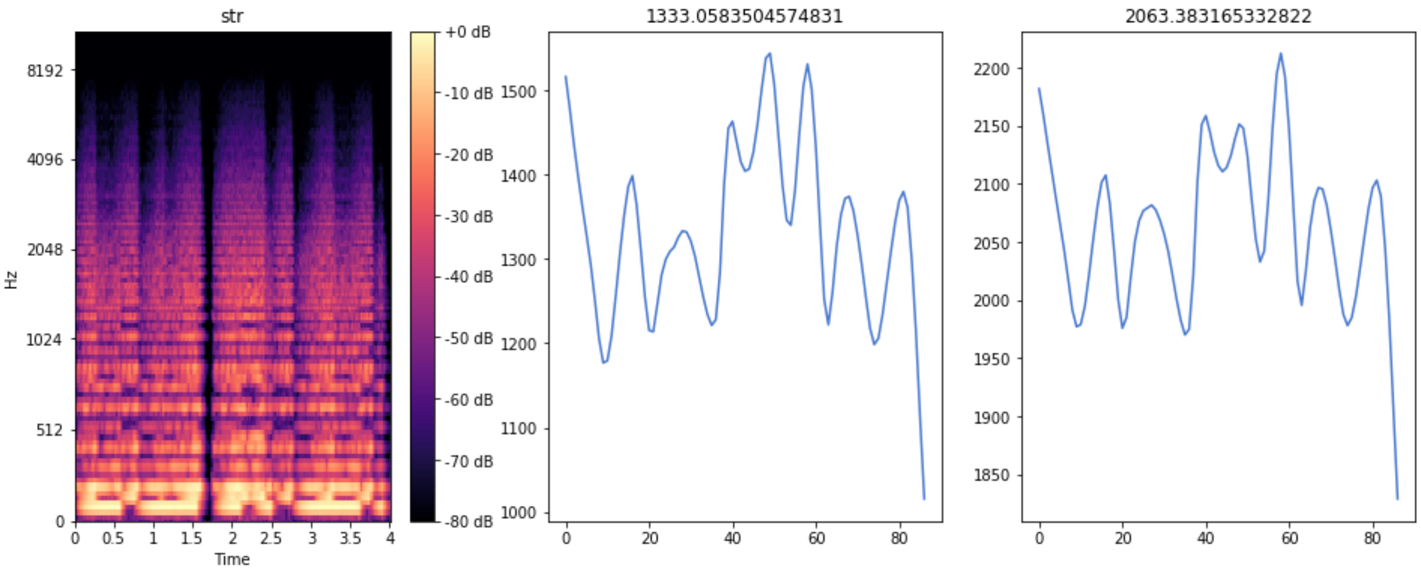
1st plot is spectrogram, 2nd one spectral centroid across STFT hops, 3rd on is spectral bandwidth across hops, on top of each graph is the average across all hops.
X axis is time/hops, Y axis is frequency bins. On top of spectrogram I labelled the track

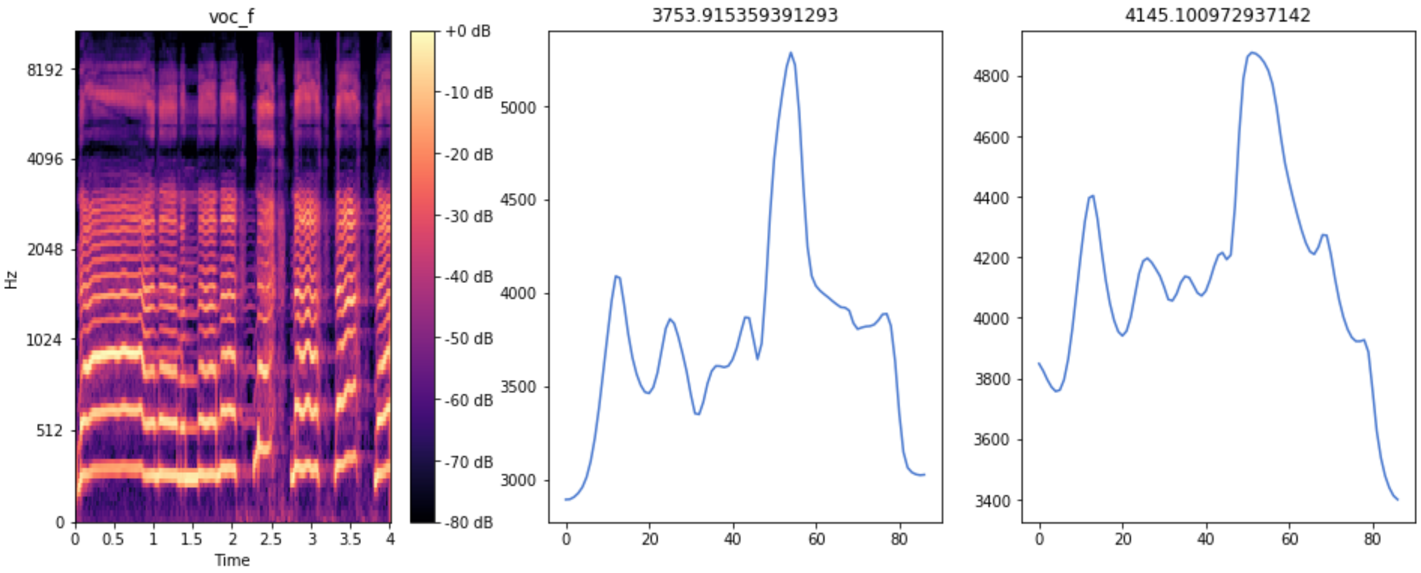


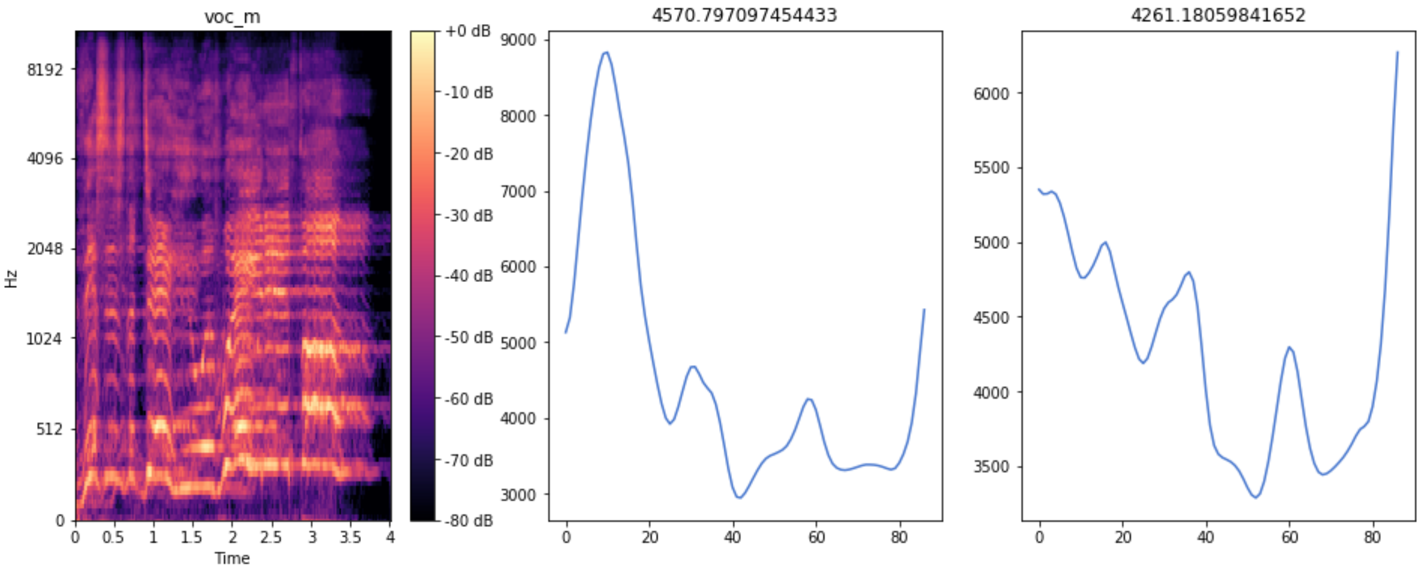

beat
4430.027007736772
4241.069949315782
bass
279.091583595199
1055.5019897267484
organ
562.9733578819857
1705.7347606279072
epiano
615.3032660947159
918.6847188330661
Hz
Time
8192
4096
2048
1024
512
0
0 0.5 1 1.5 2 2.5 3 3.5 4
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB
0 20 40 60 80
7000
6000
5000
4000
3000
2000
4800
4600
4400
4200
4000
3800
3600
3400
3200
310
300
290
280
270
260
250
240
230
1150
1125
1100
1075
1050
1025
1000
975
950
800
750
700
650
600
550
500
450
2000
1800
1600
1400
900
800
700
600
500
1500
1400
1300
1200
1100
1000
900
800

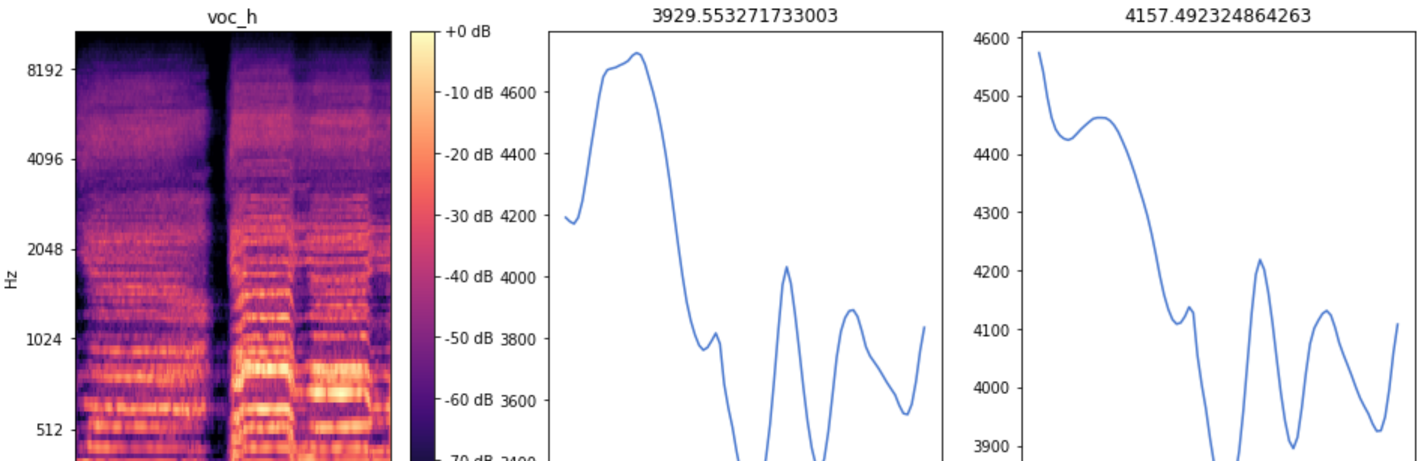

piano
993.8641267276137
1118.7816662831362
gui_ac
1268.398453235221
2490.265864837208
gui_st
2537.271969714992
3460.507713439402
clean
2151.126460166607
2019.19172296399
Hz
Time
8192
4096
2048
1024
512
0
0 0.5 1 1.5 2 2.5 3 3.5 4
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB
0 20 40 60 80

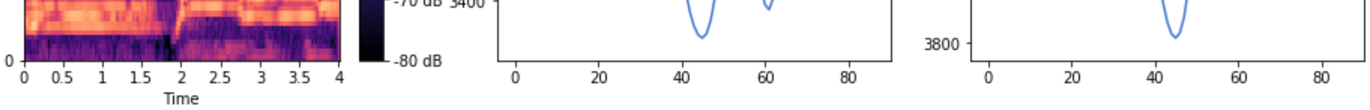

0
0 0.5 1 1.5 2 2.5 3 3.5 4
Time
-80 dB
0 20 40 60 80
0 20 40 60 80


str
8192
4096
2048
1024
512
0
Hz
0 0.5 1 1.5 2 2.5 3 3.5 4
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB
1333.0583504574831
1500
1400
1300
1200
1100
1000
0 20 40 60 80
2063.383165332822
2200
2150
2100
2050
2000
1950
1900
1850
0 20 40 60 80


voc_f
8192
4096
2048
1024
512
0
Hz
0 0.5 1 1.5 2 2.5 3 3.5 4
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB
3753.915359391293
5000
4500
4000
3500
3000
0 20 40 60 80
4145.100972937142
4800
4600
4400
4200
4000
3800
3600
3400
0 20 40 60 80


voc_m
8192
4096
2048
1024
512
0
Hz
0 0.5 1 1.5 2 2.5 3 3.5 4
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB
4570.797097454433
9000
8000
7000
6000
5000
4000
3000
0 20 40 60 80
4261.18059841652
6000
5500
5000
4500
4000
3500
0 20 40 60 80


voc_h
8192
4096
2048
1024
512
Hz
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
3929.553271733003
4600
4400
4200
4000
3800
3600
4157.492324864263
4600
4500
4400
4300
4200
4100
4000
3900

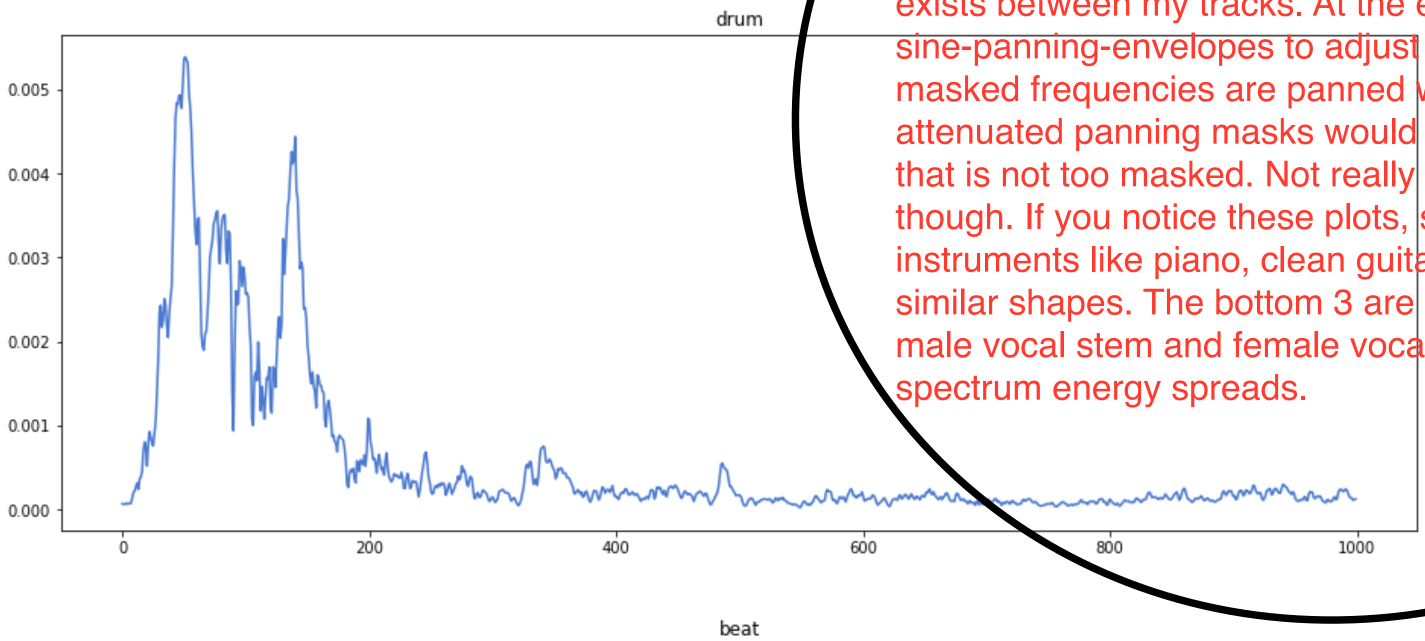


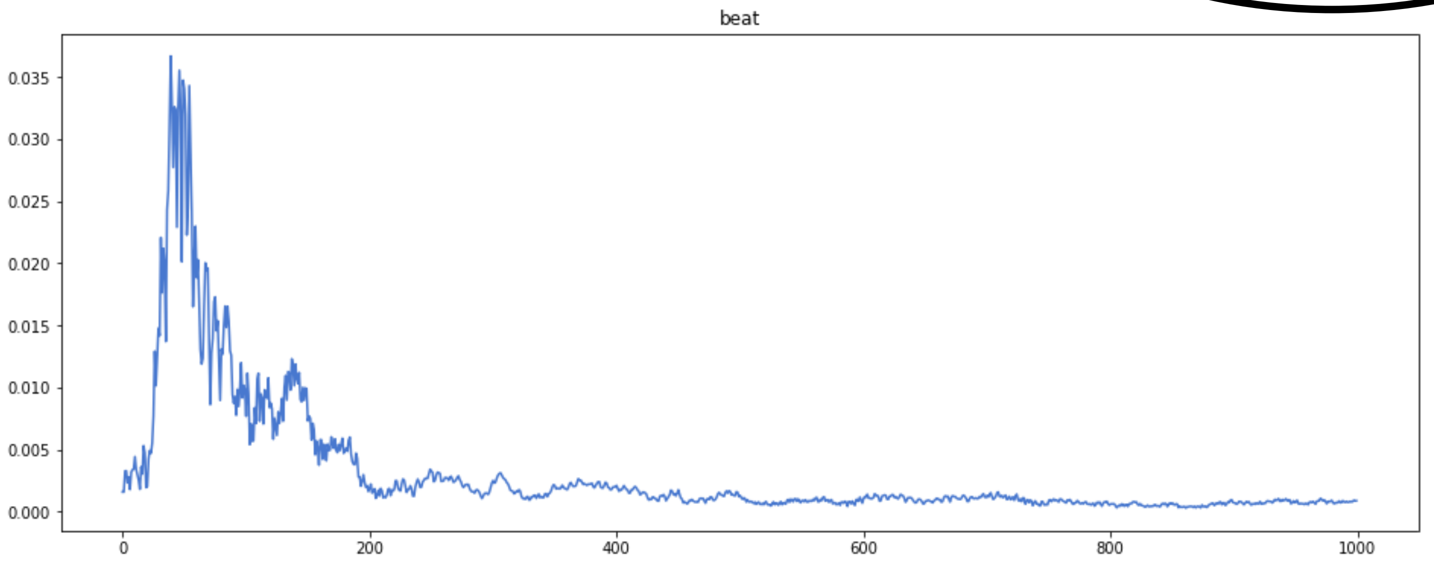
In [154]:

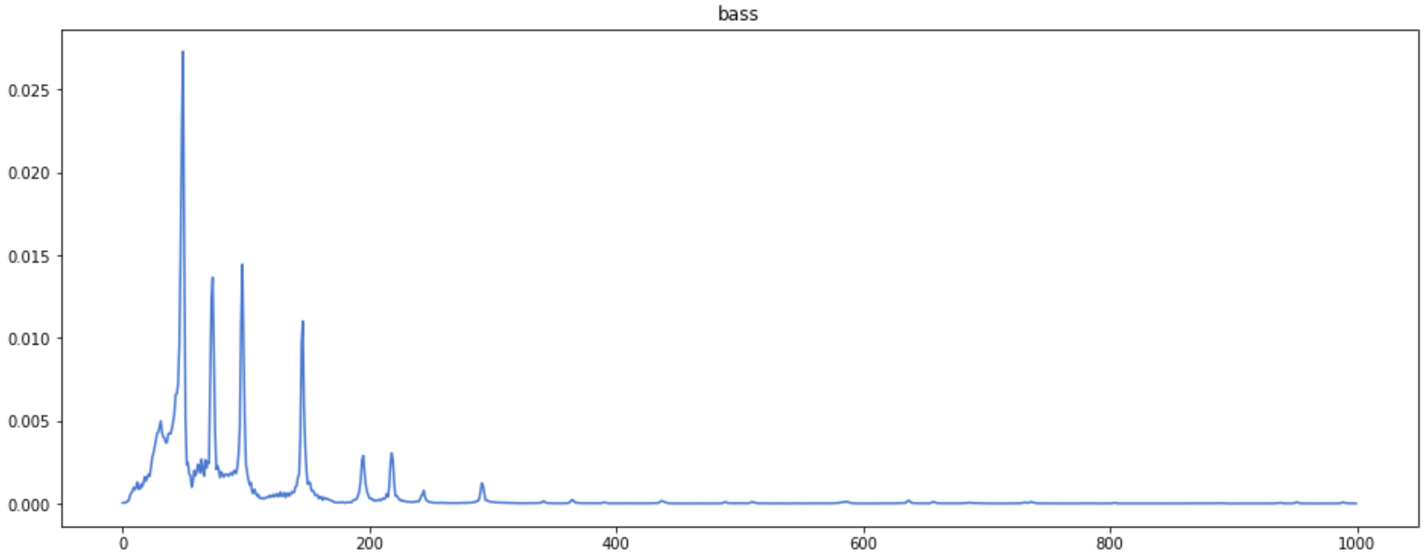
```
# initialising STFT params
N = 2 ** 15 #FFT size
M = N     #window size
H = N/16  #hop size
W = np.hanning(N)#'hann'
sr = 44100
track = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
stft = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
mX = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
pX = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
spec_avg = [[],[],[],[],[],[],[],[],[],[],[],[],[]]
# m_sim=[np.zeros(1+N/2), np.zeros(1+N/2), np.zeros(1+N/2)]
# p_sim=[np.zeros(1+N/2), np.zeros(1+N/2), np.zeros(1+N/2)]

plt.figure(figsize=(16,6))

tracks = ['drum','beat','bass','organ', 'epiano','piano','gui_ac','gui_st','clean','str','voc_f','voc_m','voc_h']
for t in range(0,len(tracks)):

    track[t],sr = librosa.load('../samples/phrases1/'+tracks[t]+'.wav', sr = sr, mono = 'True')
    stft[t] = librosa.stft(y = track[t], n_fft = 2**15,win_length=2**15, hop_length=2**15, window = 'hann')
    mX[t], pX[t] = librosa.magphase(stft[t])
    mX[t] =  mX[t] / np.sum(W) #normlaising STFT output
    spec_avg[t] = np.average(mX[t],axis=1)
    plt.figure(figsize=(16,6))
    plt.plot(spec_avg[t][0:1000])
    plt.title(tracks[t])
    plt.show()
print np.shape(stft[0])
```

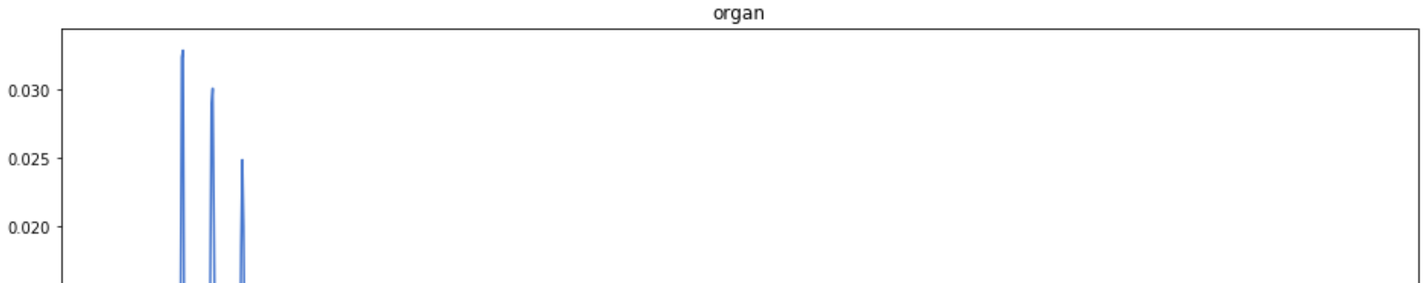
```
<Figure size 1152x432 with 0 Axes>
```

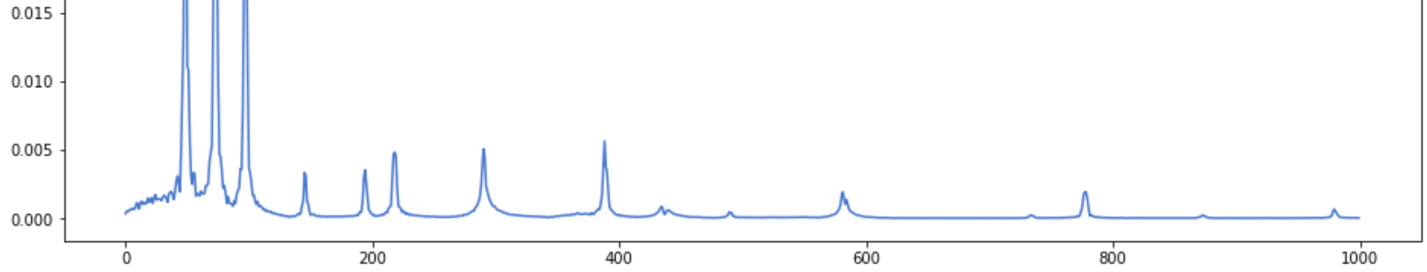
This is STFT average curve across time. Y axis real magnitudes, X axis - 1st 1000 frequency bins. I feel this is the best metric for me to decide if there is spectral masking exists between my tracks. At the end I somehow want my sine-panning-envelopes to adjust in such a way that the masked frequencies are panned well enough and the attenuated panning masks would be on top of the spectrum that is not too masked. Not really sure how to go about this though. If you notice these plots, similar sounding instruments like piano, clean guitar and acoustic guitar have similar shapes. The bottom 3 are stems of vocal harmony, male vocal stem and female vocal which have very similar spectrum energy spreads.



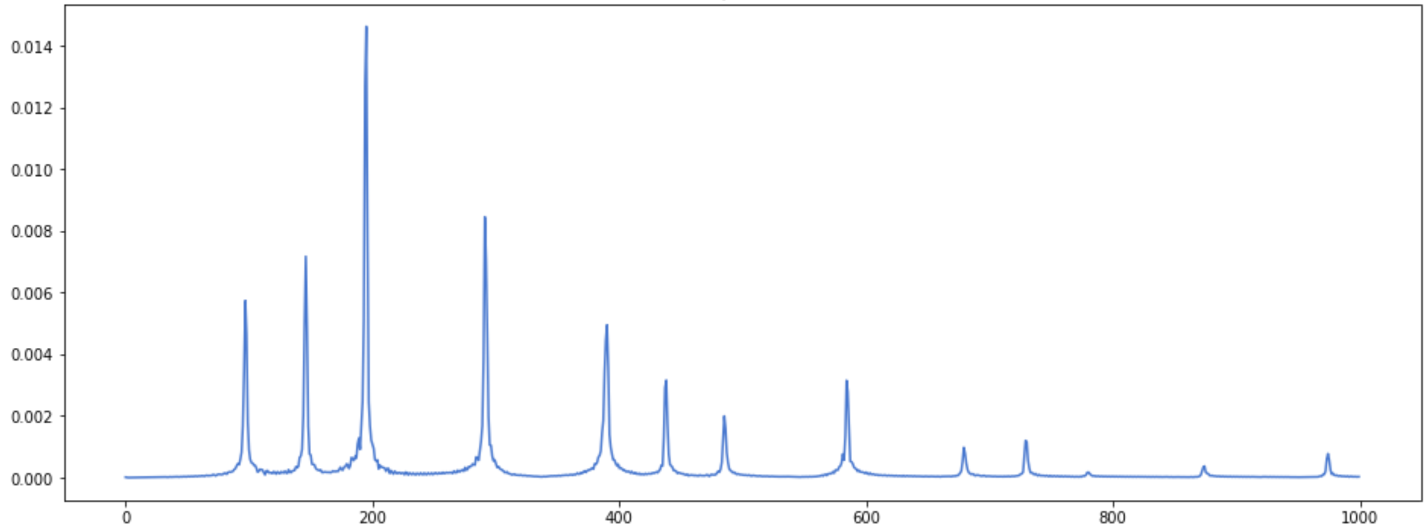




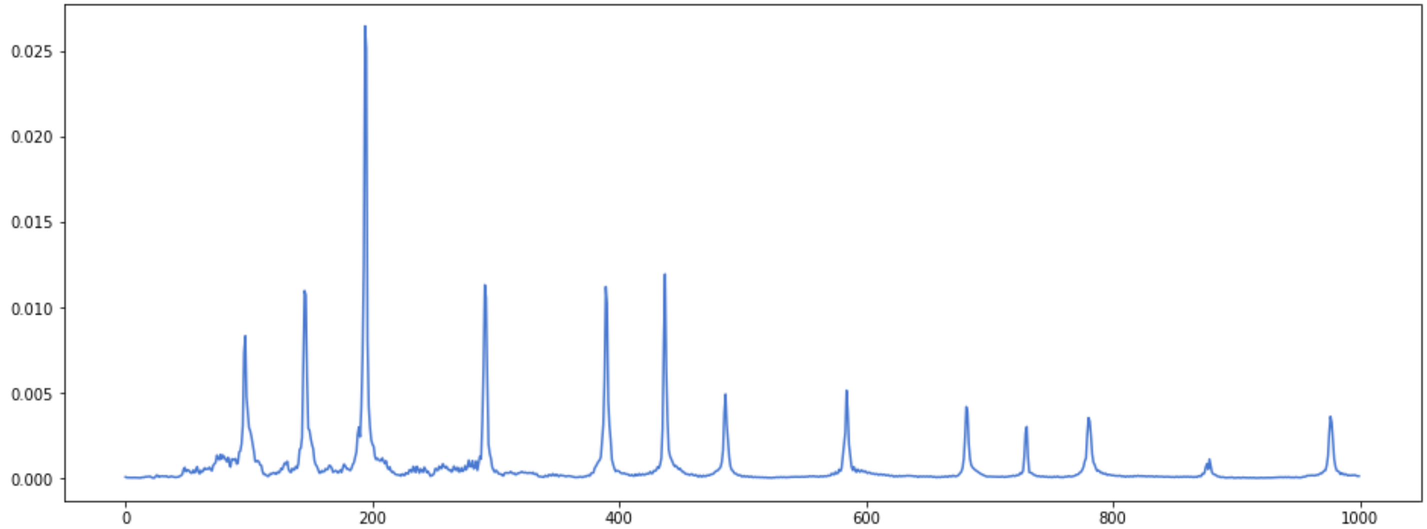




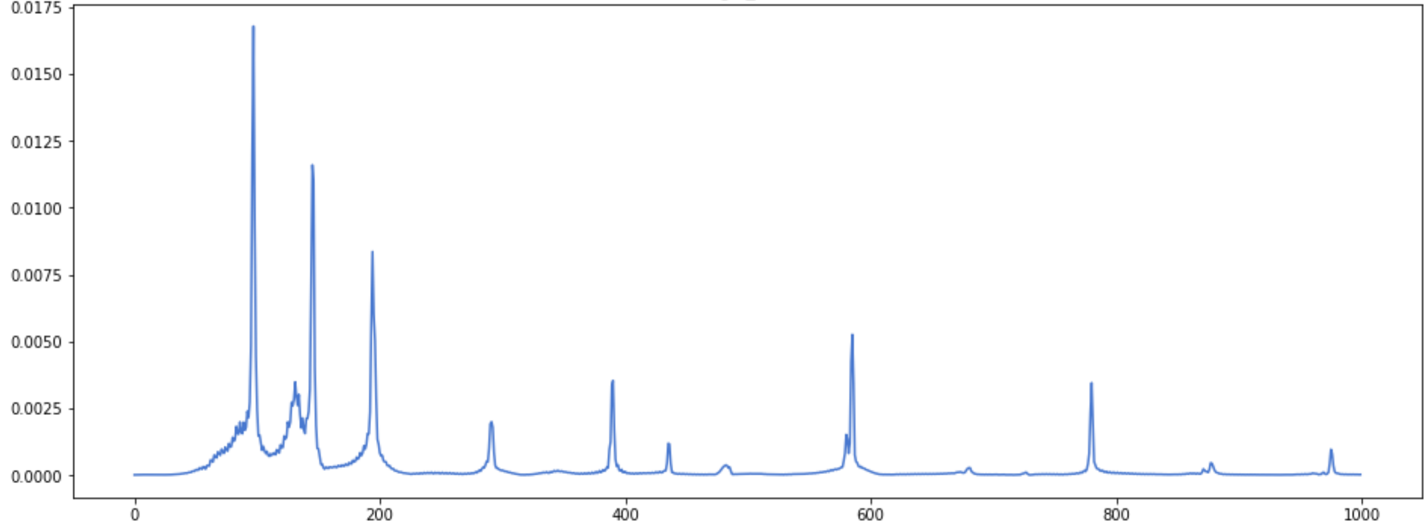
epiano



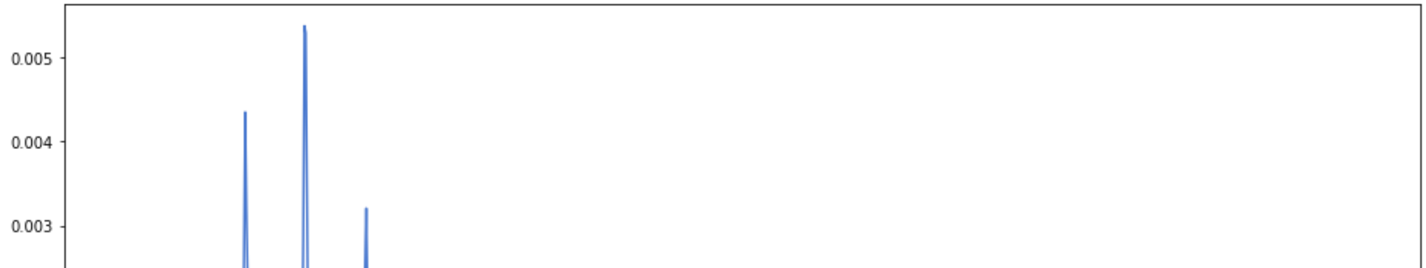
piano

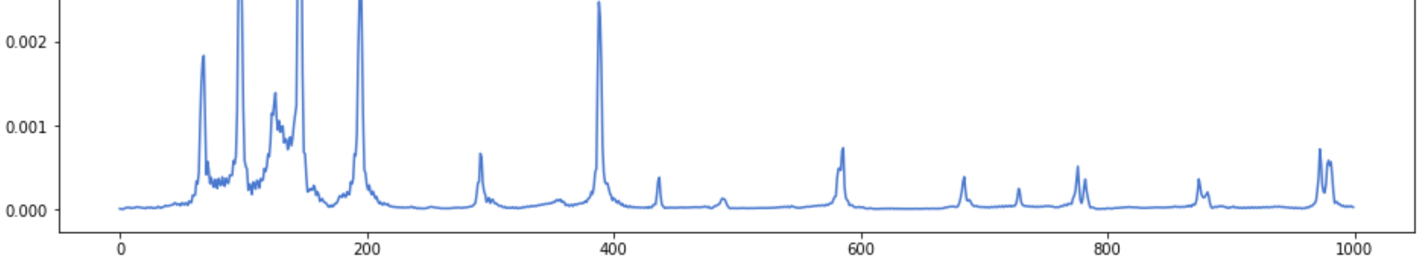


gui_ac

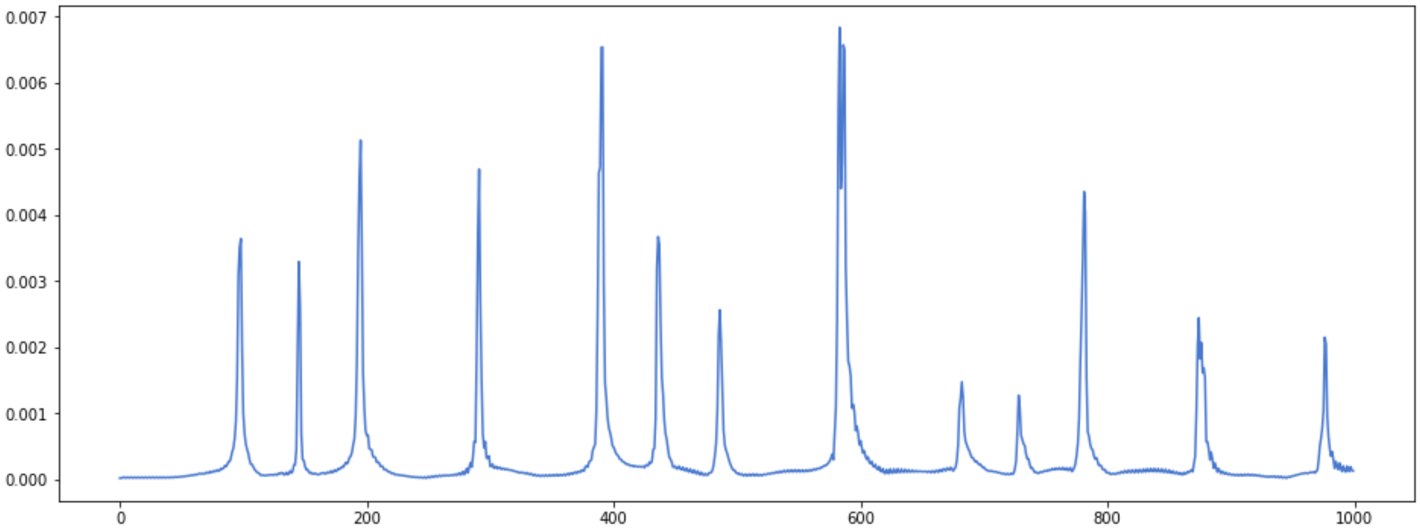


gui_st

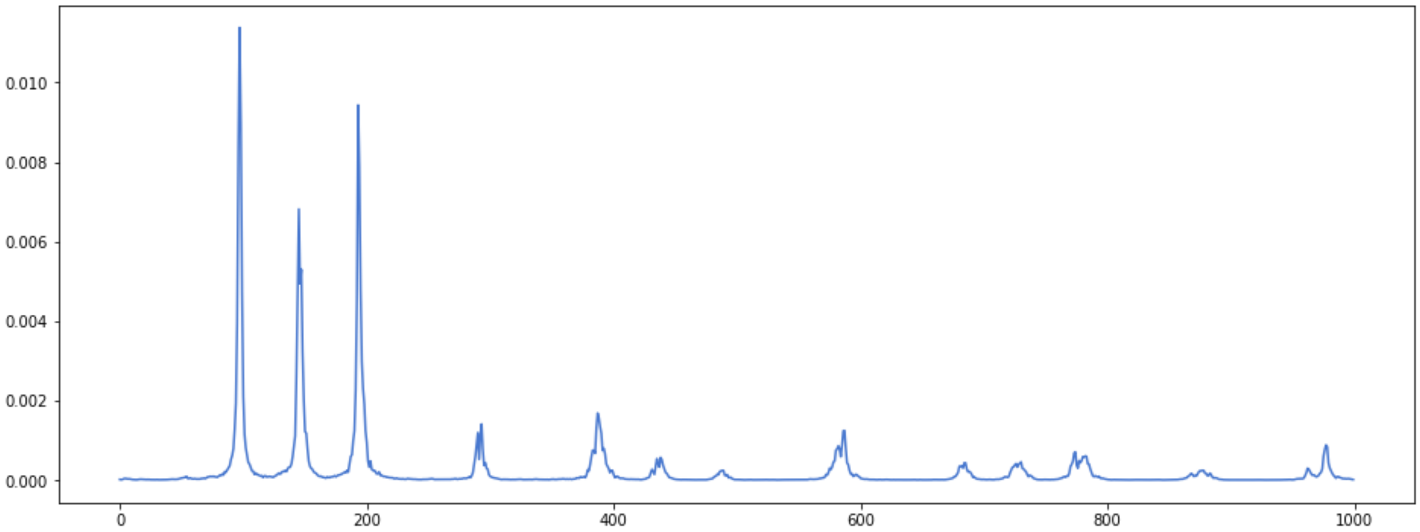




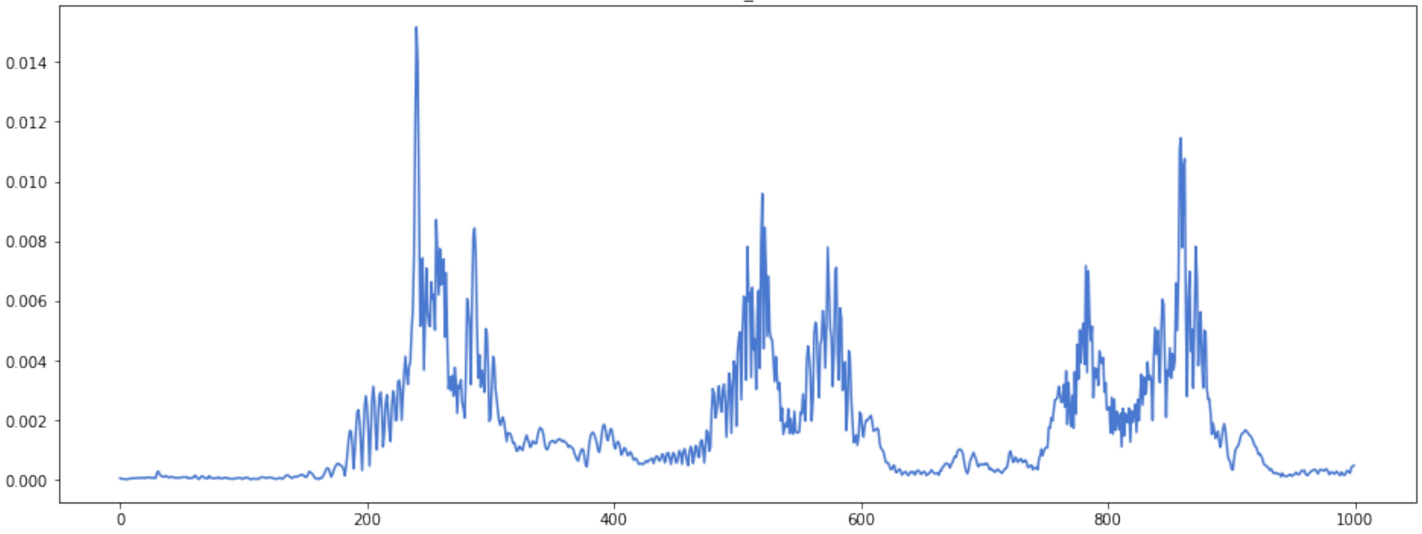
clean



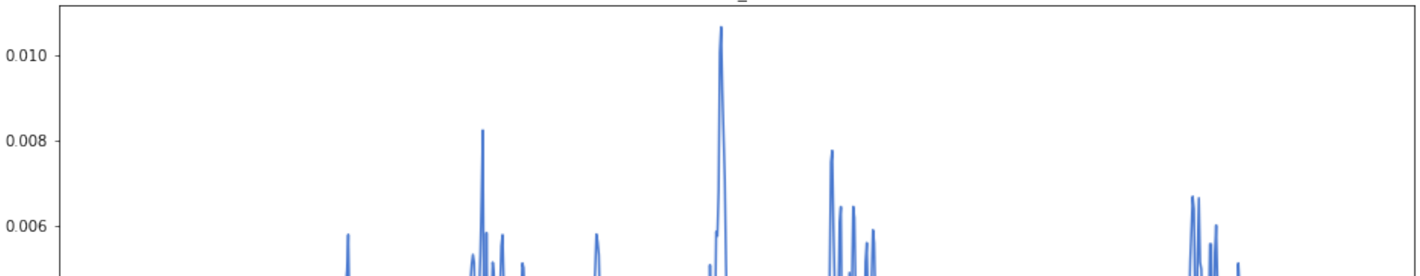
str

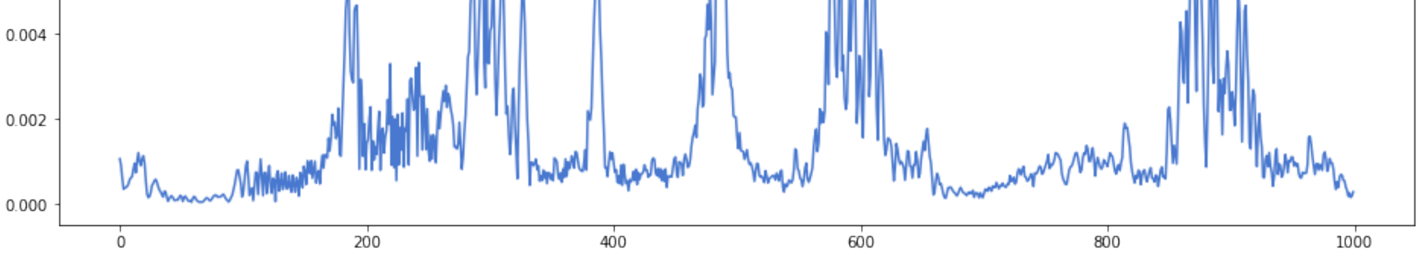


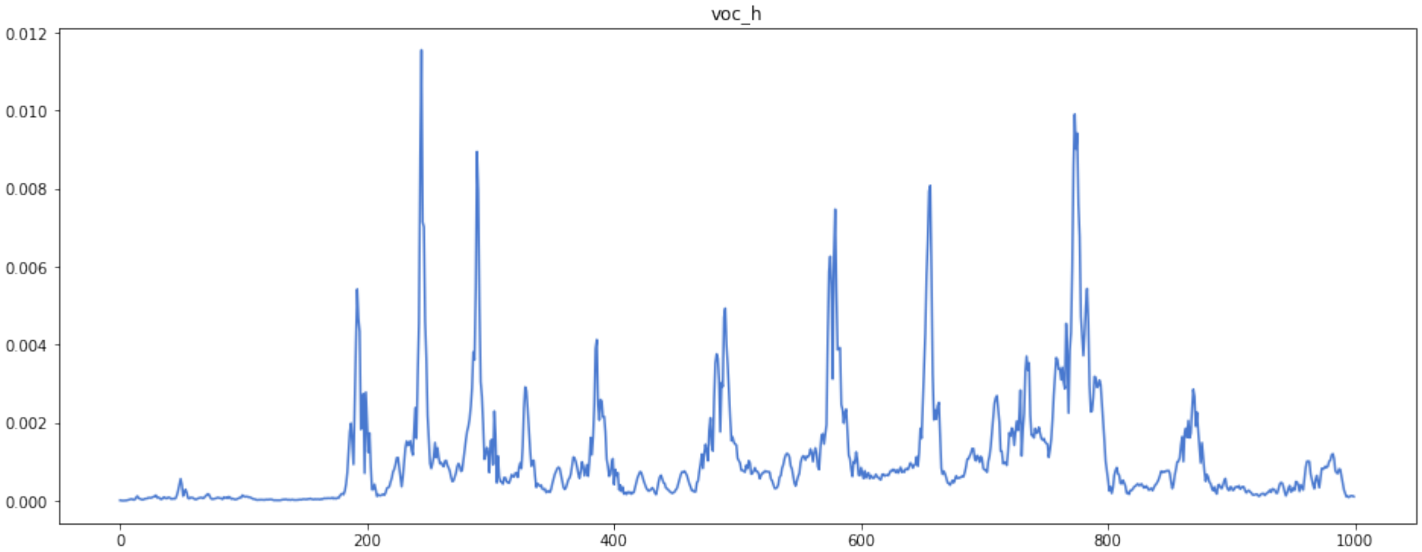
voc_f



voc_m







```
(16385, 6)
```

In [127]:

```
corr = es.CrossCorrelation(maxLag=100)
print tracks
cor_avg = np.zeros((len(tracks),len(tracks)))
for i in range(0,len(tracks)):
    for j in range(0,len(tracks)):
        corr_arr =  corr(track[i],track[j])
        cor_avg[i][j] = np.average(corr_arr)
```

```
(16385,)
```

In [155]:

```
corr = es.CrossCorrelation(maxLag=0)
print tracks
cor_avg = np.zeros((len(tracks),len(tracks)))
for i in range(0,len(tracks)):
    for j in range(0,len(tracks)):
        corr_arr =  corr(spec_avg[i],spec_avg[j])
        cor_avg[i][j] = np.average(corr_arr)
```

```
['drum', 'beat', 'bass', 'organ', 'epiano', 'piano',
'gui_ac', 'gui_st', 'clean', 'str', 'voc_f', 'voc_m'
, 'voc_h']
```