Librosa的使用方法：

一：特征提取：

Chroma特征：色度特征是色度向量（Chroma Vector）和色度图谱（Chromagram）的统称。色度向量是一个含有12个元素的向量，这些元素分别代表一段时间（如1帧）内12个音级中的能量，不同八度的同一音级能量累加，色度图谱则是色度向量的序列。

librosa.feature.chroma\_stft(y=None, sr=22050, S=None, norm=inf, n\_fft=2048, hop\_length=512, tuning=None, \*\*kwargs)

计算波形和功率谱色谱图，可获得每个帧中每个色度箱的归一化能量。

y:音频的时间序列

sr：采样频率

S：功率谱图

norm：可选择float或者none

n\_fft:傅里叶变换窗口大小

N\_chroma:产生色度的数量

librosa.feature.chroma\_cqt(y=None, sr=22050, C=None, hop\_length=512, fmin=None, tuning=None, n\_chroma=12, n\_octaves=7, bins\_per\_octave=None, cqt\_mode='full', window=None, norm=2, win\_len\_smooth=41)

色度能量归一化

C：一个预先计算的Q值谱

Norm:在chromagram列归一化

Fmin:最小的频率

N\_chroma:产生色度的数量

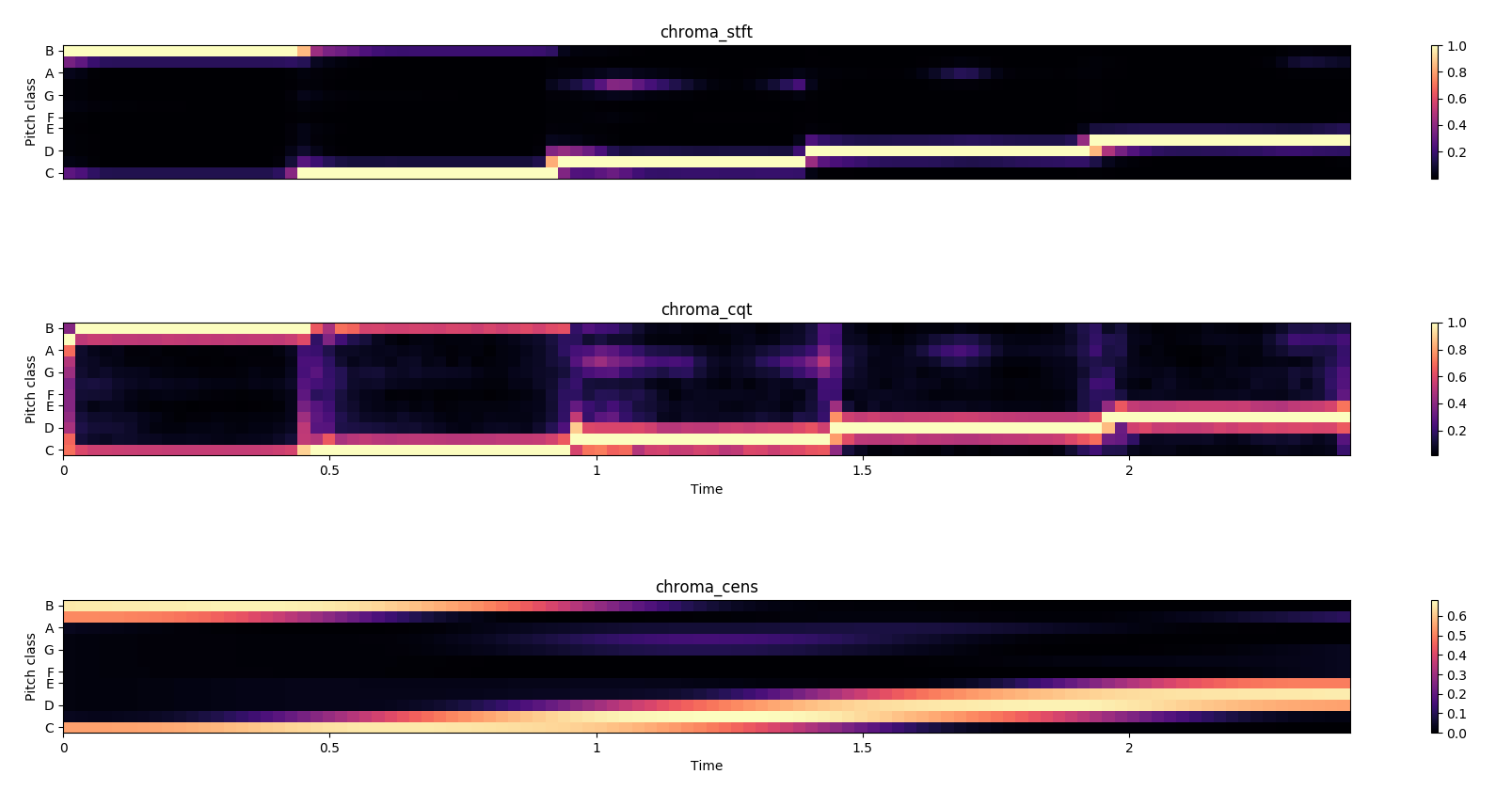
librosa.feature.chroma\_cens(y=None, sr=22050, C=None, hop\_length=512, fmin=None, tuning=None, n\_chroma=12, n\_octaves=7, bins\_per\_octave=None, cqt\_mode='full', window=None, norm=2, win\_len\_smooth=41)

色度能量归一化

这三种应该都是chroma特征提取的方法，但是所选用的变换方式不同，第一种所选用的是短时傅里叶变换，第二种是CQT变换，第三种不太清楚

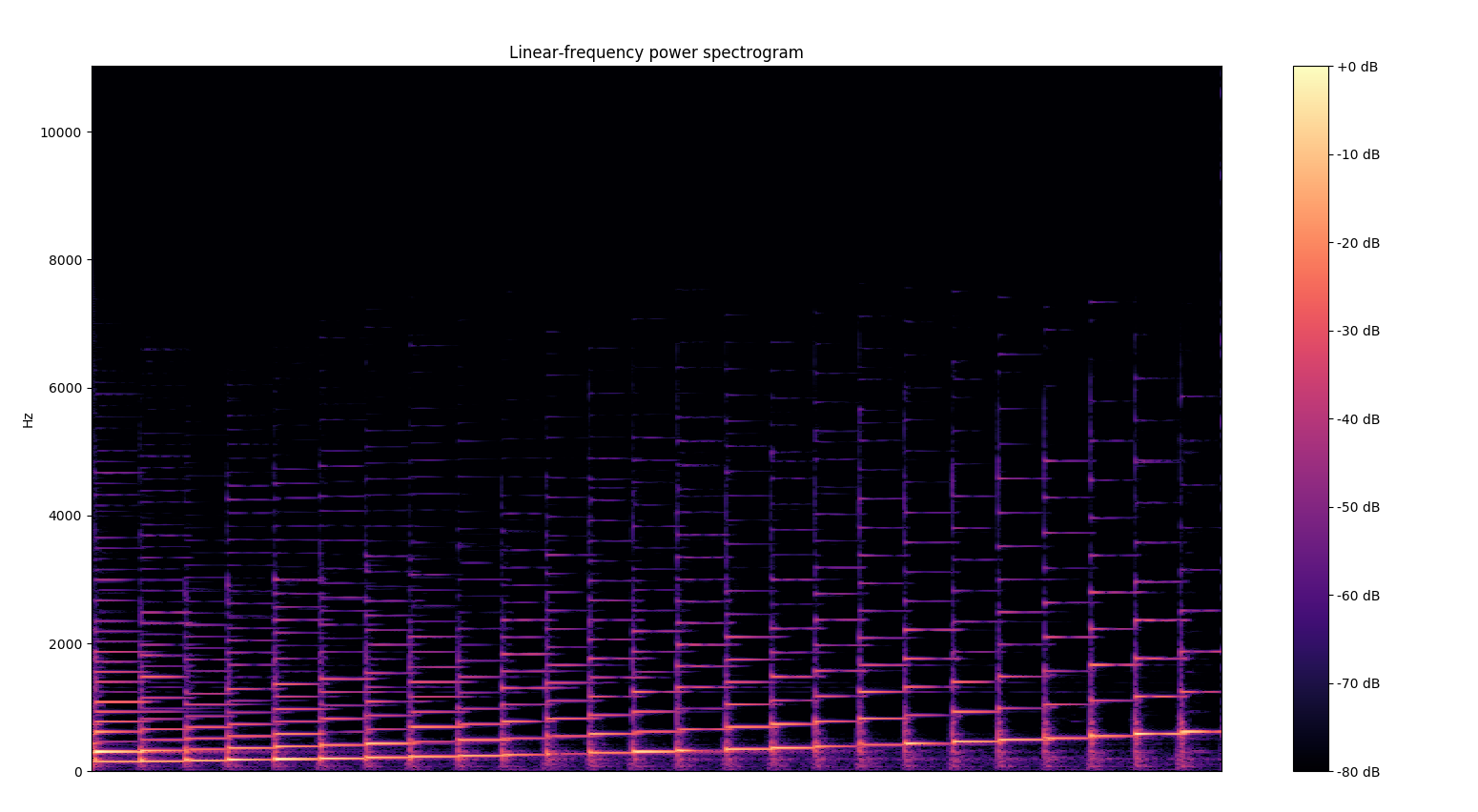
比较三种特征的差别：

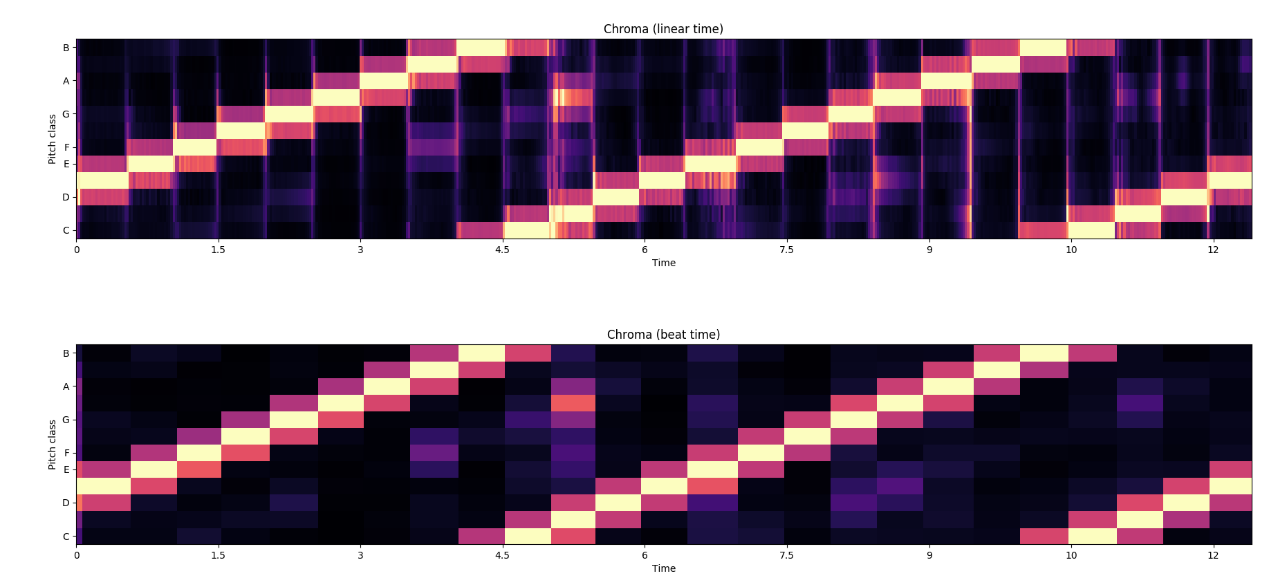
**import** numpy  
**import** librosa  
**import** librosa.display  
**import** matplotlib.pyplot **as** plt  
y, sr = librosa.load(**'piano-chrom.wav'**,offset=10, duration=15)  
chroma\_stft = librosa.feature.chroma\_stft(y=y, sr=sr, n\_fft=4096)  
chroma\_cq = librosa.feature.chroma\_cqt(y=y, sr=sr)  
chroma\_cens = librosa.feature.chroma\_cens(y=y, sr=sr)  
  
**import** matplotlib.pyplot **as** plt  
plt.figure()  
plt.subplot(3,1,1)  
librosa.display.specshow(chroma\_stft, y\_axis=**'chroma'**)  
plt.title(**'chroma\_stft'**)  
plt.colorbar()  
plt.subplot(3,1,2)  
librosa.display.specshow(chroma\_cq, y\_axis=**'chroma'**, x\_axis=**'time'**)  
plt.title(**'chroma\_cqt'**)  
plt.colorbar()  
plt.subplot(3,1,3)  
librosa.display.specshow(chroma\_cens, y\_axis=**'chroma'**, x\_axis=**'time'**)  
plt.title(**'chroma\_cens'**)  
plt.colorbar()  
plt.tight\_layout()  
plt.show()



而对于色度的其他谱的特征图（第一张图为线性频率功谱图，第二张为以线性时间和节奏时间的音阶色度图）

*#!/usr/bin/env python  
#-\*- coding:UTF-8 -\*-***import** matplotlib.pyplot **as** plt  
**import** librosa  
**import** librosa.display  
**import** numpy **as** np  
y, sr = librosa.load(**'piano-chrom.wav'**)*#加载piano-Chrome.wav文件*plt.figure(figsize=(20, 16)) *#定义色度图的大小*D = librosa.amplitude\_to\_db(librosa.stft(y), ref=np.max) *#将文件时域图像转化为频域图像，并转化成以db为单位  
#plt.subplot(4, 2, 1)*librosa.display.specshow(D, y\_axis=**'linear'**) *#画出该图*plt.colorbar(format=**'%+2.0f dB'**)  
plt.title(**'Linear-frequency power spectrogram'**)*#定义标题：线性频率功率谱图*C = librosa.feature.chroma\_cqt(y=y, sr=sr)*#获取cqt变换*plt.figure()*#定义色度图*tempo, beat\_f = librosa.beat.beat\_track(y=y, sr=sr, trim=False)  
beat\_f = librosa.util.fix\_frames(beat\_f, x\_max=C.shape[1])  
Csync = librosa.util.sync(C, beat\_f, aggregate=np.median)  
beat\_t = librosa.frames\_to\_time(beat\_f, sr=sr)  
ax1 = plt.subplot(2,1,1)  
librosa.display.specshow(C, y\_axis=**'chroma'**, x\_axis=**'time'**)  
plt.title(**'Chroma (linear time)'**)  
ax2 = plt.subplot(2,1,2, sharex=ax1)  
librosa.display.specshow(Csync, y\_axis=**'chroma'**, x\_axis=**'time'**,x\_coords=beat\_t)  
plt.title(**'Chroma (beat time)'**)  
plt.tight\_layout()  
plt.show()





节奏和节拍：

librosa.beat.tempo(y=None, sr=22050, onset\_envelope=None, hop\_length=512, start\_bpm=120, std\_bpm=1.0, ac\_size=8.0, max\_tempo=320.0, aggregate=<function mean>)

返回值为tempo（每分钟的拍子数）

部分参数：

onset\_enxelope: 预先计算的起始强度包络

max\_tempo: 如果提供，只能估算低于此阈值的速度

aggregate：若不设值，则默认为估算总体的速度，若等于None，则估算的是每一帧的速度。

librosa.beat.beat\_track(y=None, sr=22050, onset\_envelope=None, hop\_length=512, start\_bpm=120.0, tightness=100, trim=True, bpm=None, units='frames')

返回值为tempo：估算总体速度

beats：在每个事件中估算beats事件的位置

部分参数：

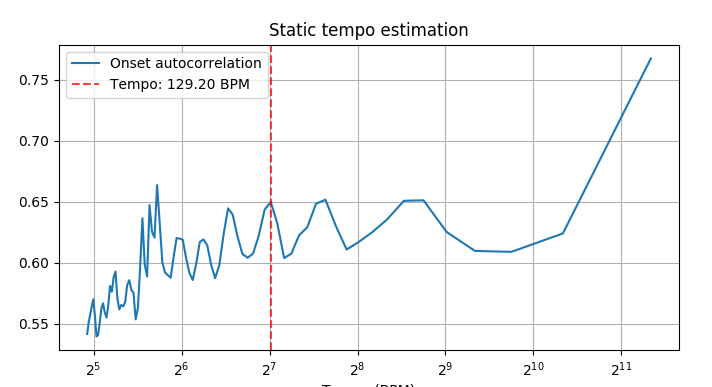
start\_bpm: 速度估计器的初始猜测（以分钟为单位）

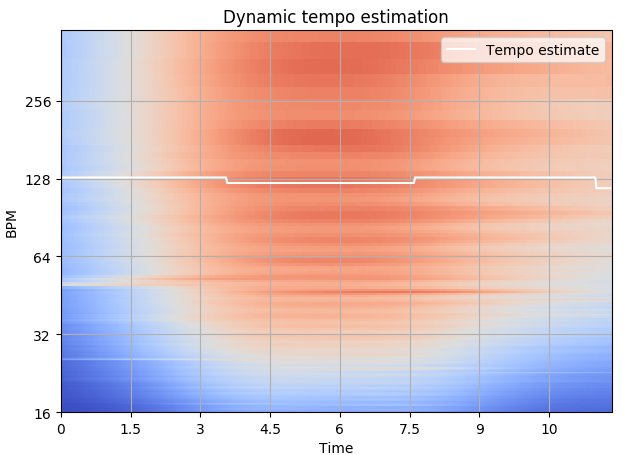
bpm：如果进行设置，请使用bpm作为速度，而不是从onsets估计它。

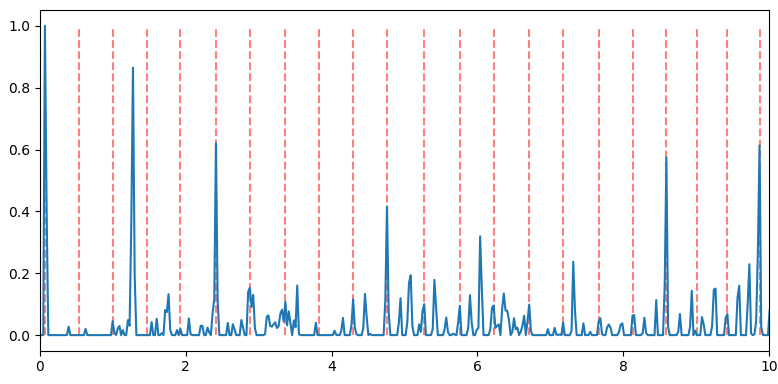
units：检测beats事件的单位。默认情况下，使用'frames'。可选择：frames，sampls，time；

librosa.beat,tempo:

*#!/usr/bin/env python  
#-\*- coding:UTF-8 -\*-***import** matplotlib.pyplot **as** plt  
**import** librosa  
**import** librosa.display  
**import** numpy **as** np  
  
y, sr = librosa.load(**'piano-chrom.wav'**)  
onset\_env = librosa.onset.onset\_strength(y=y, sr=sr)  
tempo = librosa.beat.tempo(onset\_envelope=onset\_env, sr=sr)  
dtempo = librosa.beat.tempo(onset\_envelope=onset\_env, sr=sr,aggregate=None)  
*# Convert to scalar*tempo = np.asscalar(tempo)  
*# Compute 2-second windowed autocorrelation*hop\_length = 512  
ac = librosa.autocorrelate(onset\_env, 2 \* sr // hop\_length)  
freqs = librosa.tempo\_frequencies(len(ac), sr=sr,hop\_length=hop\_length)  
*# Plot on a BPM axis. We skip the first (0-lag) bin.*plt.figure(figsize=(8,4))  
plt.semilogx(freqs[1:], librosa.util.normalize(ac)[1:],label=**'Onset autocorrelation'**, basex=2)  
plt.axvline(tempo, 0, 1, color=**'r'**, alpha=0.75, linestyle=**'--'**,label=**'Tempo: {:.2f} BPM'**.format(tempo))  
plt.xlabel(**'Tempo (BPM)'**)  
plt.grid()  
plt.title(**'Static tempo estimation'**)  
plt.legend(frameon=True)  
plt.axis(**'tight'**)  
  
plt.figure()  
tg = librosa.feature.tempogram(onset\_envelope=onset\_env, sr=sr,hop\_length=hop\_length)  
librosa.display.specshow(tg, x\_axis=**'time'**, y\_axis=**'tempo'**)  
plt.plot(librosa.frames\_to\_time(np.arange(len(dtempo))), dtempo,color=**'w'**, linewidth=1.5, label=**'Tempo estimate'**)  
plt.title(**'Dynamic tempo estimation'**)  
plt.legend(frameon=True, framealpha=0.75)  
  
plt.grid()  
plt.tight\_layout()  
plt.show()





*#!/usr/bin/env python  
#-\*- coding:UTF-8 -\*-***import** matplotlib.pyplot **as** plt  
**import** librosa  
**import** librosa.display  
**import** numpy **as** np  
  
y, sr = librosa.load(**'2.wav'**) *#加载视频，将波形赋值给y，采样频率给sr*onset\_env = librosa.onset.onset\_strength(y, sr=sr, aggregate=np.median)  
tempo, beats = librosa.beat.beat\_track(y=y,sr=sr)  
beats[:20]  
**print**(tempo)  
tempo=librosa.beat.tempo(y=y,sr=sr,aggregate=None)*#获得每一帧的速度*librosa.frames\_to\_time(beats[:20], sr=sr)  
**print**(tempo)  
hop\_length = 512  
plt.figure(figsize=(8, 4))  
times = librosa.frames\_to\_time(np.arange(len(onset\_env)),sr=sr, hop\_length=hop\_length)  
plt.plot(times, librosa.util.normalize(onset\_env),label=**'Onset strength'**)  
plt.vlines(times[beats], 0, 1, alpha=0.5, color=**'r'**,linestyle=**'--'**, label=**'Beats'**)  
*# Limit the plot to a 15-second window*plt.xlim(0,10)  
plt.gca().xaxis.set\_major\_formatter(librosa.display.TimeFormatter())  
plt.tight\_layout()  
plt.show()  


Librosa网址：http://librosa.github.io/librosa/index.html