今天声学了吗

音频信号处理
及基于音频的深度学习教程

Audio Signal Processing
Audio-based Deep Learning Tutorials

b站：今天声学了吗

公众号：今天声学了吗

邮箱：1319560779@qq.com

信号分析简介

什么是信号分析?

1. 信号分析，**将一个复杂信号分解成若干简单信号分量之和**，或者用有限的一组参量去表示一个复杂波形的信号，从这些简单的分量组成情况去考察复杂信号的特性；
2. **特征提取**的过程，从一段复杂的波形中提取出我们需要的信息。

为什么要分析信号?

1. 信号分析是**获取信号特征信息的重要手段**，通过对信号分析，得到复杂信号的基本特征。
2. 信号分析是**获取信号源特征信息的重要手段**，人们往往可以通过对信号特征的详细了解，得到信号源特性、运行状况等信息。
3. 深度学习的第一步都是要收集数据集的特征(feature)。所以需要信号分析，提取出信号数据集的特征集。

CHAPTER 1

信号分析简介

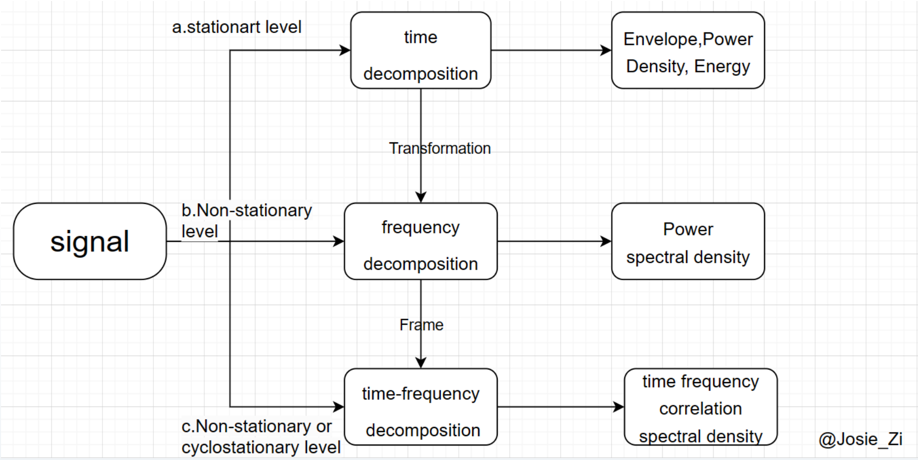
● **怎样进行信号分析的方式**

信号分析的方式——根据信号的类型选择

1. 信号平稳——由于信号不随时变化，可以在时域和频域直接分析；
2. 信号不平稳——由于信号会随时变化，需要**将信号拆解分析**

   1. 时间维度拆解，利用平均瞬时功率信号等，可以知道信号传播的规律
   2. 频域维度拆解，按照不同频带的滤波器进行滤波处理，平均瞬时功率谱信号
   3. 时频域维度拆解，同时包含信号随时间和频率变化的规律。

   因此，信号分析的方式可分为时域、频域、时频域。



CHAPTER 1--信号的频域拆解

多个信号的叠加

信号的叠加: https://teropa.info/harmonics-explorer/

一个复杂信号分解成若干简单信号分量之和。不同个频率信号的叠加：由于和差化积，会形成包络结构与精细结构。

发现: 低频信号决定了信号的包络形状,高频信号决定其精细结构

结论: 在语音识别中,主要通过信号的包络结构来区分不同音频信号,因此在识别领域更关注低频作用.

$$\sin\alpha+\sin\beta=2\sin\frac{\alpha+\beta}{2}\cos\frac{\alpha-\beta}{2}\cdots\cdots(1)$$

$$\sin\alpha-\sin\beta=2\cos\frac{\alpha+\beta}{2}\sin\frac{\alpha-\beta}{2}\cdots\cdots(2)$$

$$\cos\alpha+\cos\beta=2\cos\frac{\alpha+\beta}{2}\cos\frac{\alpha-\beta}{2}\cdots\cdots(3)$$

$$\cos\alpha-\cos\beta=-2\sin\frac{\alpha+\beta}{2}\sin\frac{\alpha-\beta}{2}\cdots\cdots(4)$$

幅值包络Amplitude Envelope

操作：依次寻找每一帧中的幅值最大值，将每一帧中幅值最大值连起来就是幅值包络。

Last sample of frame *t*

Amplitude of *k*th sample

$$\boxed{AE_t} = \max_{\boxed{k=t\cdot K}}^{\boxed{(t+1)\cdot K-1}} \boxed{s(k)}$$

Amplitude envelope at frame *t*

First sample of frame *t*

现提取第t帧的AE值，其中k是采样点数， t是帧序列数， K每一帧的帧长，采样点k点在{t K ,(t+1) k-1}

代码：

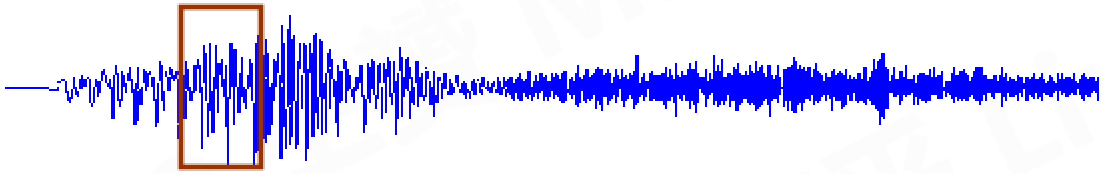
```
""" 提取信号的幅值包络
# 1. 加载信号 librosa.load()
# 2. 定义一个AE的函数，功能为取信号每一帧中幅值最值为该帧的包络
# 最值的获取方式：max(waveform[t*(frame_size-hop_size):t*frame_size])
# 3. 设置参数：每一帧长1024，以50%的重叠率分帧，调用该函数
# 4. 绘制信号的幅值包络信息
"""
```

```
"""
1.下载语音库librosa->conda activate py37->pip install librosa
2.下载数组处理库nunpy
3.加载路径区分：绝对路径和相对路径
4.构造函数需要知道：总的采样点数，帧的个数，每帧的采样点数,分帧补零
   if len(waveform)%hop_length != 0:
      n_pad = int((len(waveform)-frame_length)/hop_length)+1
      n_pad = n_pad*hop_length+frame_length-len(waveform)
      waveform = np.pad(waveform,(0,n_pad),'wrap')
   frame_num = int(np.ceil((len(waveform) - frame_length) / hop_length)) + 1
6.每一帧点数的选择waveform[t * (frame_size - hop_size):t * (frame_size - hop_size) +
frame_size]
7.return np.array(waveform_AE)
"""
```

应用：振幅包络可以给出响度soundness的大致信息，对突变信号特别敏感（outlier effect）

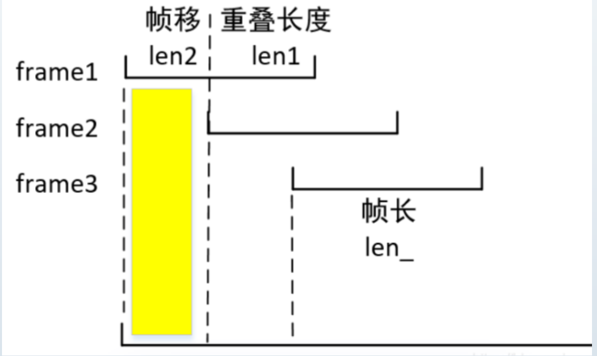
功能：音频检测、音频分类 onset detection/ music genre classification

CHAPTER 1--信号的时域分析



分帧概念

- 分帧：将信号按照时间长度分割，每一段的长度就是帧长frame_size，分出n段，则说明帧的个数frame_num，如果不考虑重叠分帧，那么该信号总的采样点数为frame_size* frame_num。
- 分帧重叠：为了让分帧后的信号更加平滑，需要重叠分帧，也就是下一帧中包含上一帧的采样点，那么包含的点数就是重叠长度hop_size。
- 分帧补零：帧的个数frame_num = 总样本数N / 重叠数hop_size (分帧不补零)，因为帧的个数frame_num是整数，为了不舍弃最后一帧不能凑成一个完整帧长的点，需要对信号补零。此时帧的个数frame_num = (总样本数N -帧长frame_size)/ 重叠数hop_size (分帧补零)




```
if len(waveform) % hop_length != 0:
    frame_num = int((len(waveform)-frame_length)/hop_length)+1
    n_pad = frame_num *hop_length+frame_length-len(waveform)
    waveform = np.pad(waveform,(0,n_pad),'wrap')
frame_num = int(np.ceil((len(waveform) - frame_length) / hop_length)) + 1
```

CHAPTER 1--信号的时域分析

均方根能量Root mean square energy

- 操作：依次寻找每一帧中的RMSE，其值为第t帧中每点幅值平方再取均值后开根号

$$RMS_t = \sqrt{\boxed{\frac{1}{K}\cdot\sum_{k=t\cdot K}^{(t+1)\cdot K-1} s(k)^2}}$$

Mean of sum of energy

- 对比：与时域包络相比，RMSE体现了每一帧的包络变化，适用于不平稳的信号。尤其是对于突变信号（outlier effect），RMSE得到的值较平稳，因为它利用每一帧的所有点幅值的平均值，而不像AE利用每一帧中的最大幅值。
- 应用：RMSE与响度有关，用于音频分段、分类 audio segmentation, music genre classification。
- 代码:

```
"""信号的均方根值RootMeanSquareEnergy
# 1.加载信号
# 2.定义函数，功能：计算每一帧的均方根能量，
公式=该帧信号的平方和，取帧长的平均值后，开根号后
# 3.设置参数：每一帧长1024，以50%的重叠率分帧，调用该函数
# 4.绘制图像
# 5.利用librosa.feature.rms绘制信号的RMSE
# 6.比较两者差异
"""
```

```
"""
0.公式 = np.sum(waveform[t*(frame_length - hop_length):t * (frame_length - hop_length) + frame_length] ** 2) / frame_length
1.平方的表达方式 **2
2.librosa得到的信号是二维数组，因此需要转置处理.T
3.画图过程，librosa.frames_to_time()
4.librosa.feature.rms(y=waveform,s = Spectrogram)
5.librosa.feature.rms会多计算一帧，
# 因此需要选择从第一帧开始的值.T[1:,0]
"""
```

CHAPTER 1--信号的时域分析

过零率Zero crossing rate

- 介绍：是一个信号符号变化的比率，即在每帧中语音信号从正变为负或从负变为正的次数。计算第t帧信号过零点数：

$$ZCR_t = \frac{1}{2} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K-1} \left| \boxed{sgn(s(k))} - sgn(s(k+1)) \right|$$

Sign function:

- s(k) > 0 → +1
- s(k) < 0 → -1
- s(k) = 0 → 0

其中Sgn是符号函数，前一个点与后一个点符号相反，则相减的绝对值为2；前一个点与后一个点符号相同，则相减的绝对值为0。前面乘以1/2, 还需要除帧的长度，也就是取均值。

- 功能：用于语音识别和音乐信息检索。通常对类似金属、摇滚等高冲击性的声音的具有更高的价值。一般情况下，过零率越大，频率近似越高。

- 代码：

```
# 1.加载信号
# 2.定义函数，功能：计算每一帧的均方根能量，
公式=该帧信号的平方和，开根号后，取帧长的平均值
np.sqrt(np.sum(signal[i:i + frame_length] ** 2) / frame_length)
# 3.设置参数：每一帧长1024，以50%的重叠率分帧，调用该函数
# 4.绘制图像
# 5.利用librosa.feature.rms绘制信号的RMSE
# 6.比较两者差异
```

```
"""
0.公式 = np.sum(waveform[t*(frame_length - hop_length):t * (frame_length - hop_length) + frame_length] ** 2) / frame_length
1.平方的表达方式 **2
2.librosa得到的信号是二维数组，因此需要转置处理.T
3.画图过程，librosa.frames_to_time()
4.librosa.feature.rms(y=waveform,s = Spectrogram)
5.librosa.feature.rms会多计算一帧，因此需要选择从第一帧开始的值.T[1:,0]
"""
```

CHAPTER 1--信号的频域分析

常见方法：傅里叶频谱分析、功率谱分析、倒频谱分析、共振解调技术等。

结果：得到不同频率的幅值和相位，幅值表示了原始信号和sin的相似度

谱质心Spectral centroid

- 介绍：是频率成分的重心，是在一定频率范围内通过能量加权平均的频率，其单位是Hz。

$$SC = \frac{\sum_{n=1}^{N} f(n) \cdot E(n)}{\sum_{1}^{N} E(n)} = \sum_{n=1}^{N} f(n) \cdot P(E(n))$$

SC= 每一帧的幅值*对应点的频率的和/每帧的幅值之和 centroid[t] = sum_k S[k, t] * freq[k] / (sum_j S[j, t])

- 应用：谱质心描述了声音的明亮度，具有阴暗、低沉品质的声音倾向有较多低频内容，谱质心相对较低，具有明亮、欢快品质的多数集中在高频，谱质心相对较高。该参数常用于对乐器声色的分析研究。

子带带宽Bandwidth

- 操作：在Spectral centroid的频谱范围，计算距离每一帧中心点的平均值

BW=每个采样点减去谱质心的绝对值*对应点的权重值/总的权重之和

(sum_k S[k, t] * (freq[k, t] centroid[t])**p)**(1/p)

$$BW_t = \frac{\sum_{n=1}^{N} \underbrace{\boxed{|n - SC_t|}}_{\text{Distance of frequency band from spectral centroid}} \cdot \underbrace{\boxed{m_t(n)}}_{\text{Weight for } n}}{\sum_{n=1}^{N} \boxed{m_t(n)}}$$

- 应用：用于音频识别和主观听音感受，如果音频的能量谱密度函数下降快，那么BW也下降，类似于频谱的变化速度。
- 代码

```
# 1. 加载信号
# 2. 获得信号的centroid和bandwidth
librosa.feature.spectral_centroid(y=waveform)
librosa.feature.spectral_centroid(y=waveform)
# 3. 绘制信号
```

常见方法：短时傅里叶变换、小波变换、Wigner-Ville时频分布及HHT等。

结果：得到不同时间、频率处信号的幅值和相位，很好地分析了实际音频信号的非平稳非线性特点。

短时傅里叶分析法STFT

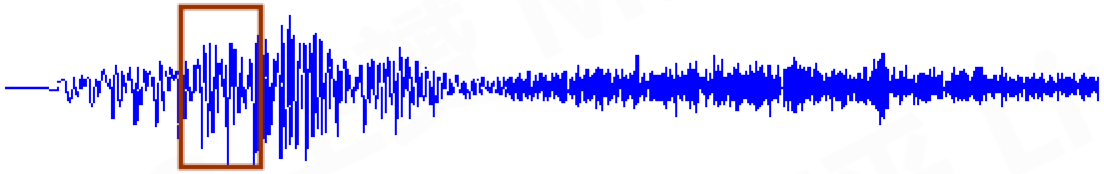
- 介绍：由于声信号往往是随时间变化的，在短时间内可以近似看做平稳（对于语音来说是几十毫秒的量级），所以我们希望把长的声音切短，来观察其随时间的变化情况，由此产生STFT分析方式。
- 结果：得到不同时刻，不同频率的频谱图（能量分布情况）
- FFT与STFT对比：

STFT在时域中对信号进行加窗处理（分帧），所以最终结果是有关时域频域的信息，时域的信息是每一帧帧长（窗函数的长度）

FT: $$F(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-i\omega t}\,dt$$

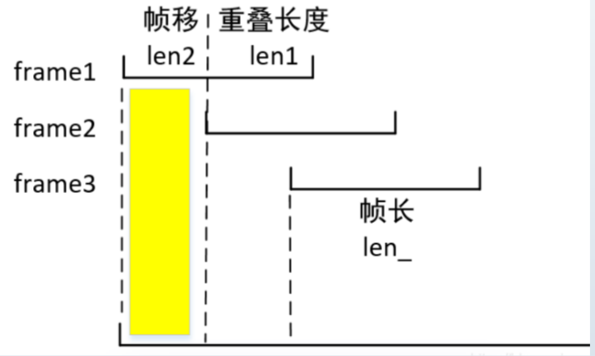
Time localization!

STFT: $$F(\boxed{\tau}, \omega) = \int_{-\infty}^{+\infty} f(t)\,w(t-\tau)e^{-i\omega t}\,dt$$



分帧概念

- 分帧：将信号按照时间长度分割，每一段的长度就是帧长frame_size，分出n段，则说明帧的个数frame_num，如果不考虑重叠分帧，那么该信号总的采样点数为frame_size* frame_num。
- 分帧重叠：为了让分帧后的信号更加平滑，需要重叠分帧，也就是下一帧中包含上一帧的采样点，那么包含的点数就是重叠长度hop_size。
- 分帧补零：帧的个数frame_num = 总样本数N / 重叠数hop_size (分帧不补零)，因为帧的个数frame_num是整数，为了不舍弃最后一帧不能凑成一个完整帧长的点，需要对信号补零。此时帧的个数frame_num = (总样本数N -帧长frame_size)/ 重叠数hop_size (分帧补零)




```
if len(waveform) % hop_length != 0:
    frame_num = int((len(waveform)-frame_length)/hop_length)+1
    n_pad = frame_num *hop_length+frame_length-len(waveform)
    waveform = np.pad(waveform,(0,n_pad),'wrap')
frame_num = int(np.ceil((len(waveform) - frame_length) / hop_length)) + 1
```

短时傅里叶分析法STFT

- 关系：如果窗函数带宽长，则包络中的精细结构较少，疏松，得到窄带语谱图，有较好的频域分辨率，但时域分辨率较差；如果窗函数带宽窄，则包络中的精细结构较多，密集，得到宽带语谱图，有较好的时域分辨率，但时域分辨率较差；
- 代码：

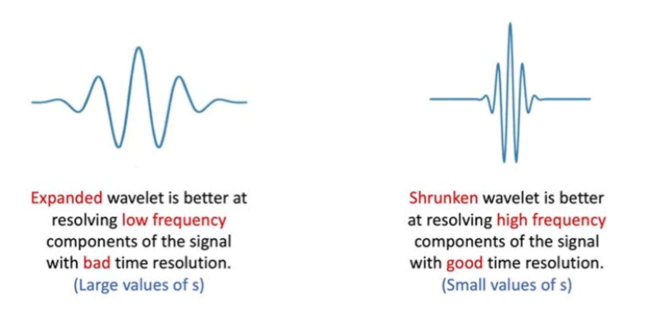
```
# 1.加载信号
# 2.信号分帧：补零->加窗->分帧
# 3.信号做傅里叶变换np.fft.rfft(waveform_frame,n_fft)
# waveform_fft = np.fft.rfft(waveform_win,n_fft)
# waveform_pow = np.abs(waveform_fft)**2/n_fft
# waveform_db = 20* np.log10(waveform_pow)
# 4.绘制波形
```

```
"""
1.分帧的具体步骤：补零->分帧的索引值->加窗
2.索引值的获取
row = np.tile(np.arange(0,frame_size),(frame_num,1))
column =
np.tile(np.arange(1,frame_num*hop_size,hop_size),(frame_size,
1))
3.rfft是对实数值做傅里叶变化，ifft是逆变换,fft是普通的傅里叶变换，nfft是n维的傅里叶变换
"""
```

小波变换

- 介绍：对于不稳定的信号，难以用普通的FFT分析出频域随时变化的信息。所以对于不同时间段的频域函数进行分析，**利用小波作为基函数，各个小波函数按照不同比例系数展开得到F，其中小波函数可以更改中心频率和带宽**。
- 操作：将欧拉函数置换成小波函数，小波函数随着原信号会发生变化。

$$F(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{+\infty} f(t)\, \psi^{*}\left(\frac{t-\tau}{s}\right) dt$$



- STFT与小波变换对比：

STFT分帧的窗函数是固定时长，相同窗长分帧的过程使频率分辨率就会降低。因为分帧也就是加窗的过程，由于窗函数长度有限的，会造成截断。窗函数长，则时间分辨率低，而频率分辨率高；窗函数短，则频率分辨率低，而时间分辨率高

小波变换通过更改小波函数的带宽，设置不同的基频，因此不同频段利用不同的分辨率。

在低频成分用高的频率分辨率；高频成分用高的时间分辨率。

Mel-Filter-Banks & MFCC

- 介绍：正常语音频谱的频率是线性分布的，但人的耳朵对频率的感觉是对数的。因此出现了具有对数分布频率的梅尔谱。
- 公式：

$$m = 2595 \log_{10}(1 + \frac{f}{700})$$

$$f = 700(10^{m/2595} - 1)$$

f是频率，m是Mel频率。当处于高频时，Mel频率变化缓慢。

- 特性：低频段变化迅速，高频段变化缓慢。
- 梅尔滤波器组：在梅尔频谱的滤波器。在高频区域，滤波器的数量变得相对较少，根据人耳的功能，分布非常稀疏。
- MFCC：对于Meil谱，做离散余弦变换（DCT，类似于傅里叶变换的线性变换），

然后可以取一部分系数作为MFCC。

CHAPTER 1--信号的时频域分析

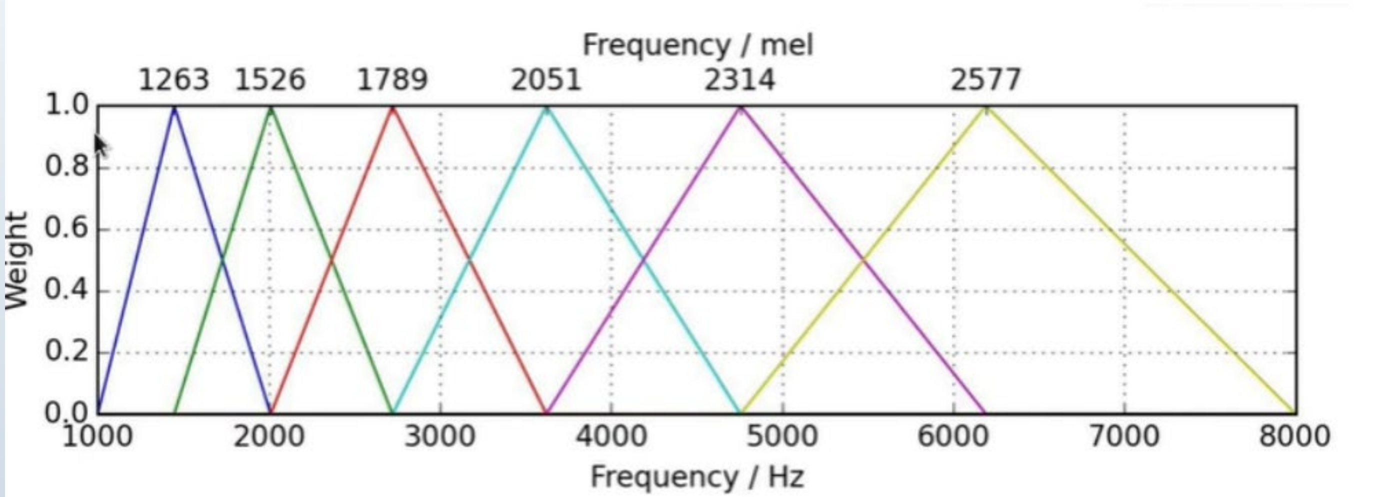
Mel-Filter-Banks

- 滤波器组：一系列等高三角形滤波器，每个起点位于前一滤波器的中点。其对应的频率在Meil尺度上是线性的，并且可以在Meir标尺上线性地划分为几个频带，然后转换回实际的频率尺度。

- 公式：

$$H_m(k) = \left\{ 0 \quad k < f(m-1) \quad \frac{k - f(m-1)}{f(m) - f(m-1)} \quad f(m-1) \leq k < f(m) \quad 1 \quad k = f(m) \quad \frac{f(m+1) - k}{f(m+1) - f(m)} \quad f(m) < k \leq f(m+1) \quad 0 \quad k > f(m+1) \right.$$

- 结果:



今天声学了吗

Mel-Filter-Banks

- 计算公式:

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k < f(m) \\ 1 & k = f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & f(m) < k \le f(m+1) \\ 0 & k > f(m+1) \end{cases}$$

$$m = 2595 \log_{10}(1 + \frac{f}{700})$$

$$f = 700(10^{m/2595} - 1)$$

- 流程:
  - 将最低和最高频率转换为Mel-freq
  - 分为几个频带，每个频带的间隔是相同的梅尔频率
  - 将Mel-freq转换为线性频率
  - 舍入到最近的频率点
  - 三角形滤波函数作为每个帧的窗口
  - 每个滤波器都与pow相乘，得到最后的滤波结果。

- 注意:
  - m是滤波器组数，f（m-1）/f（m）/f（m+1）是#m个滤波器的开始/中间/最终点的频率点。频率点不是频率，而是线性频率对应的采样点! 比如，这里最大频率是22050Hz，所以对应的是第512个样品即频率f所对应的值是f*NFFT/fs)

  - Mel-N不能太大。如果Mel-N太大，前几个滤波器的长度将为0

Mel-Spectrum

- **构建Mel滤波器组**

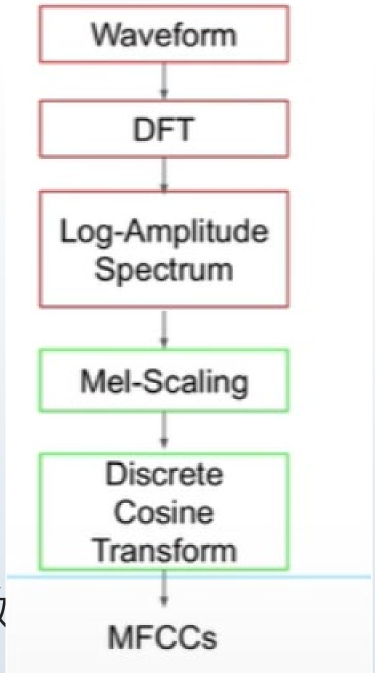
$$p(f) = \left|X(f)^2\right| = |\mathrm{FFT}(x(n))|^2$$

- **得到信号的能量谱密度**

$$E(m) = \sum_{k=0}^{N-1} (p(f) \cdot H_m(f))$$

- **Mel滤波器进行滤波**
- **取对数值**

$$E'(m) = \lg \sum_{k=0}^{N-1} (p(f) \cdot H_m(f))$$



MFCC

- 频谱继续做离散余弦变换(离散余弦变换) 以获得MFCC参数（实际上是傅里叶逆变换过程）取n=1,2……p，p是MFCC最高阶次，M是过滤器数量

$$C(n) = \sum_{k=1}^{M} E'(m) \cos\left[\frac{\pi(k-0.5)n}{M}\right]$$

将第2至第13系数作为MFCC。
- 为什么使用离散余弦变换而不是傅里叶变换?
- 因为余弦变换可以得到实数，而不是虚数，并且我们不需要对称结果，所以MFCC（delta窗函数不重叠。
- 代码

```
"""信号的Mel变换
# 0.构建Mel滤波器
# 1.加载信号
# 2.对信号做预处理，增强高频成分
# 3.做STFT（分帧，加窗，FFT）
# 4.对功率谱函数用Mel滤波器滤波
# 5.绘制图像
"""
```