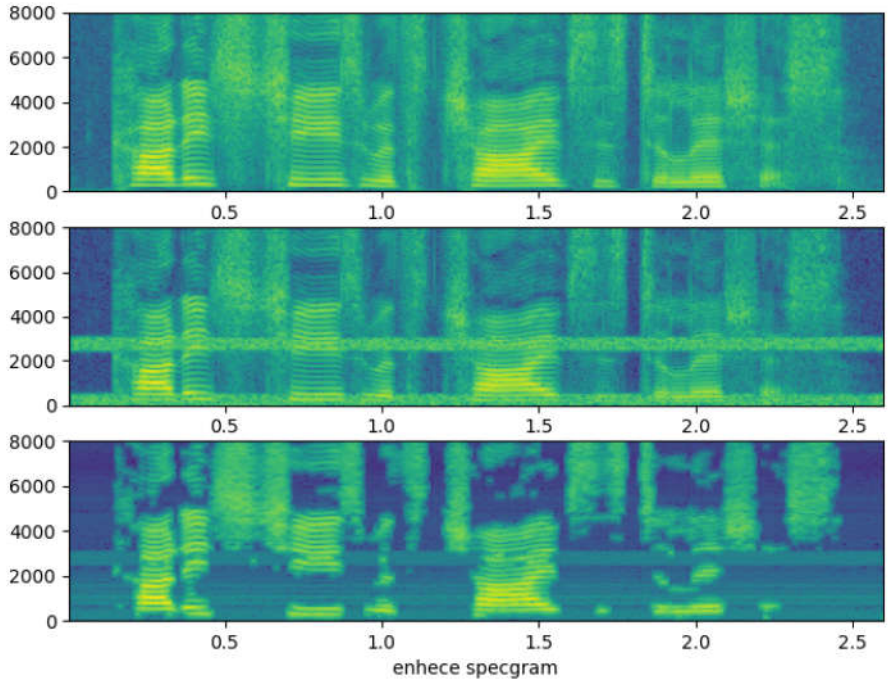
语音增强-谱减法

Speech Enhancement- Spectral Subtraction



于泓
鲁东大学
信息与电气工程学院
2021.6.28

语音增强（去噪）

- 消除语音中的噪声，增加语音听感与可懂度

谱减法的基本原理

$$y(n) = x(n) + e(n)$$

$$Y(\omega) = X(\omega) + E(\omega)$$

$$\left|\hat{X}(\omega)\right| = \left|Y(\omega)\right| - \left|E(\omega)\right|$$

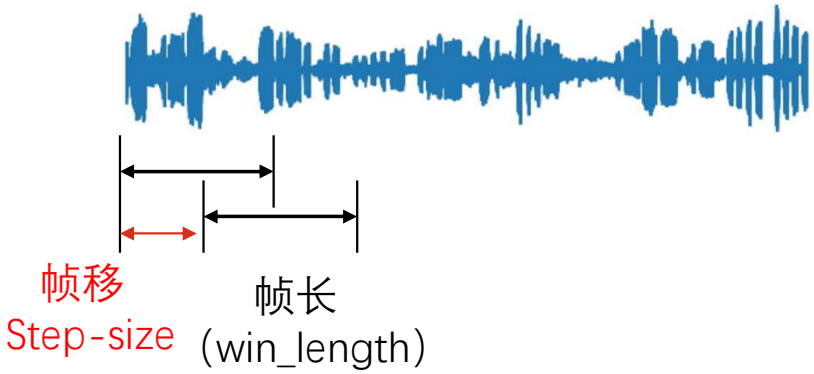
$$\hat{X}(\omega) = \left|\hat{X}(\omega)\right| e^{j\varphi_Y(\omega)}$$

$$\hat{x}(t) = \mathrm{IDFT}(\hat{X}(\omega))$$

$$\left|\hat{X}(\omega)\right| = \begin{cases} 0 & \left|\hat{X}(\omega)\right| < 0 \\ \left|\hat{X}(\omega)\right| & \text{其他} \end{cases}$$

实现过程

- 1 将输入语音分帧进行短时傅里叶变换（STFT）



- 2 取前N帧进行噪声估计（获取 $E(\omega)$）
- 3 谱减
- 4 ISTFT 还原获取干净语音$x(n)$

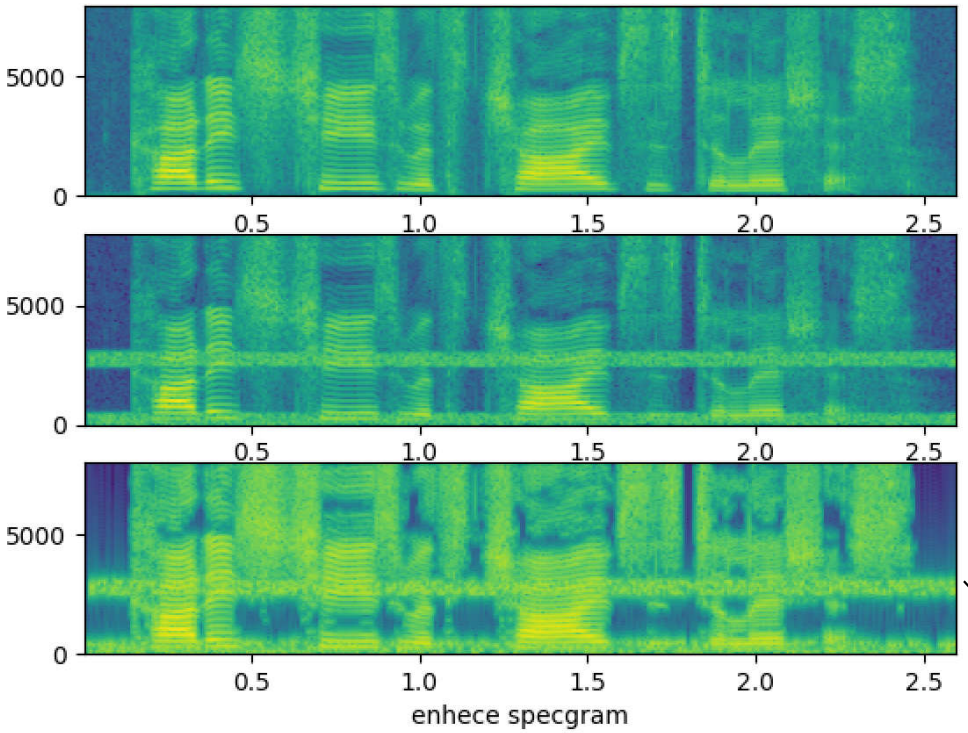
```python
import librosa
from librosa.core.spectrum import amplitude_to_db
import numpy as np
import soundfile as sf
import matplotlib.pyplot as plt

if __name__ == "__main__":
    clean_wav_file = "sf1_cln.wav"
    clean,fs = librosa.load(clean_wav_file,sr=None)
    print(fs)

    noisy_wav_file = "sf1_n0L.wav"
    noisy,fs = librosa.load(noisy_wav_file,sr=None)

    # 计算 nosiy 信号的频谱
    S_noisy = librosa.stft(noisy,n_fft=256, hop_length=128, win_length=256)  # D x T
    D,T = np.shape(S_noisy)
    Mag_noisy= np.abs(S_noisy)
    Phase_nosiy= np.angle(S_noisy)
    Power_nosiy = Mag_noisy**2
    print(fs)
    # 估计噪声信号的能量
    # 由于噪声信号未知 这里假设 含噪（noisy）信号的前30帧为噪声
    Mag_nosie = np.mean(np.abs(S_noisy[:,:30]),axis=1,keepdims=True)
    Power_nosie = Mag_nosie**2
    Power_nosie = np.tile(Power_nosie,[1,T])
```

干净 

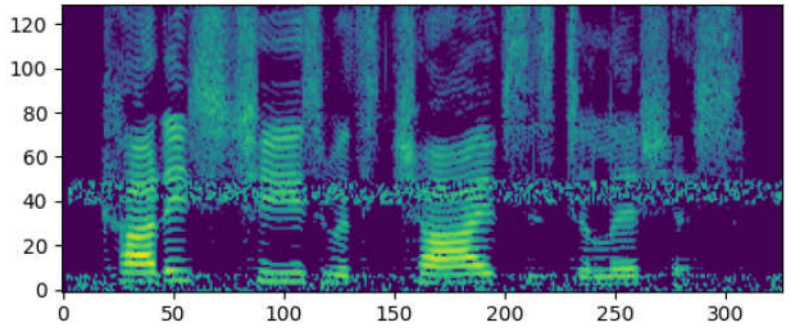
噪声 

```
# 能量减
Power_enhenc = Power_nosiy-Power_nosie
# 保证能量大于0
Power_enhenc[Power_enhenc<0]=0
Mag_enhenc = np.sqrt(Power_enhenc)

# 幅度减
# Mag_enhenc = np.sqrt(Power_nosiy) - np.sqrt(Power_nosie)
# Mag_enhenc[Mag_enhenc<0]=0

# 对信号进行恢复
S_enhec = Mag_enhenc*np.exp(1j*Phase_nosiy)
enhenc = librosa.istft(S_enhec, hop_length=128, win_length=256)
sf.write("enhce.wav",enhenc,fs)
print(fs)
# 绘制谱图

plt.subplot(3,1,1)
plt.specgram(clean,NFFT=256,Fs=fs)
plt.xlabel("clean specgram")
plt.subplot(3,1,2)
plt.specgram(noisy,NFFT=256,Fs=fs)
plt.xlabel("noisy specgram")
plt.subplot(3,1,3)
plt.specgram(enhenc,NFFT=256,Fs=fs)
plt.xlabel("enhece specgram")
plt.show()

plt.imshow(librosa.amplitude_to_db(Mag_enhenc,ref=np.max),origin='lower')
plt.show()
```

增强语音



音乐噪声



过减法

一般为4-6

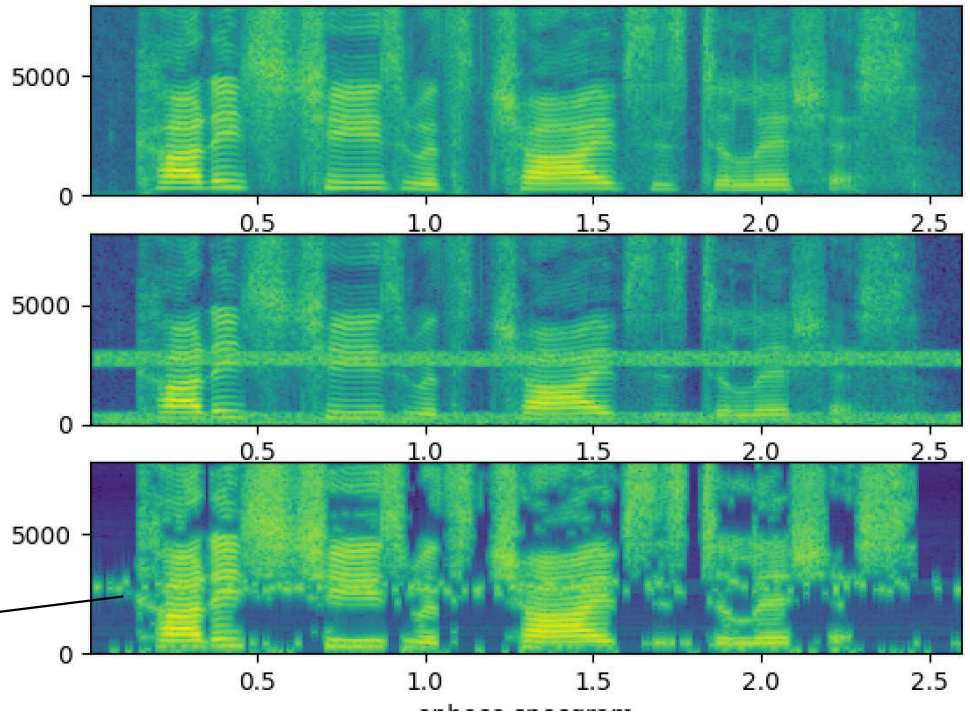
$$\mathrm{P_x}(\omega)=\left(P_Y(\omega)^{\gamma}-\alpha P_Y(\omega)^{\gamma}\right)^{\frac{1}{\gamma}}$$

较小 0.001~0.0001

$$\mathrm{P_x}(\omega)=\begin{cases}\beta P_e(\omega) & P_X(\omega)<\beta P_e(\omega)\\ P_x(\omega) & \text{其他}\end{cases}$$

$$\left|\mathrm{X}(\omega)\right|=\sqrt{\mathrm{P_x}(\omega)}$$

```
## 方法2 超减
# 引入参数
alpha = 4
gamma = 1

Power_enhenc = np.power(Power_nosiy,gamma) - alpha*np.power(Power_nosie,gamma)
Power_enhenc = np.power(Power_enhenc,1/gamma)

# 对于过小的值用 beta* Power_nosie 替代
beta = 0.0001
mask = (Power_enhenc>=beta*Power_nosie)-0
print(mask.shape)
Power_enhenc = mask*Power_enhenc + beta*(1-mask)*beta*Power_nosie

Mag_enhenc = np.sqrt(Power_enhenc)
```



谱线的不连续

引入平滑机制

最大噪声残差

$$\max(\omega) = \arg\max \sum_{t=0}^{T_{noise}} \mathrm{E}_t(\omega) - E(\omega)$$

估计的噪声（$E_t(w)$的均值）

对于过小的部分用相邻帧的最小值取代

$$|\mathrm{X}(\omega)| = \begin{cases} \arg\min \sum_{t-k}^{t+k} |\mathrm{X}_t(\omega)| & |\mathrm{X}(\omega)| < \max(\omega) \\ |\mathrm{X}(\omega)| & \text{其他} \end{cases}$$

```
## 方法3 引入平滑
Mag_noisy_new = np.copy(Mag_noisy)
k=1
for t in range(k,T-k):
    Mag_noisy_new[:,t] = np.mean(Mag_noisy[:,t-k:t+k+1],axis=1)

Power_nosiy = Mag_noisy_new**2

# 超减法去噪
alpha = 4
gamma = 1

Power_enhenc = np.power(Power_nosiy,gamma) - alpha*np.power(Power_nosie,gamma)
Power_enhenc = np.power(Power_enhenc,1/gamma)

# 对于过小的值用 beta* Power_nosie 替代
beta = 0.0001
mask = (Power_enhenc>=beta*Power_nosie)-0
Power_enhenc = mask*Power_enhenc + beta*(1-mask)*Power_nosie

Mag_enhenc = np.sqrt(Power_enhenc)
```

```
Mag_enhenc_new  = np.copy(Mag_enhenc)
# 计算最大噪声残差
maxnr = np.max(np.abs(S_noisy[:,:31])-Mag_nosie,axis =1)

k = 1
for t in range(k,T-k):
    index = np.where(Mag_enhenc[:,t]<maxnr)[0]
    temp = np.min(Mag_enhenc[:,t-k:t+k+1],axis=1)
    Mag_enhenc_new[index,t] = temp[index]


# 对信号进行恢复
S_enhec = Mag_enhenc_new*np.exp(1j*Phase_nosiy)
enhenc = librosa.istft(S_enhec, hop_length=128, win_length=256)
sf.write("enhce_3.wav",enhenc,fs)
print(fs)
```