UFABC - ESTI019 - Codificação de Sinais Multimídia

Laboratório 6 - PDS da Voz

Prof. Mário Minami

OBJETIVOS:

1. Gravar Arquivos de Áudio com dígitos, números gerais, texto lido e poesia
2. Leitura de Arquivos de Áudio e janelamento
3. Cálculo das Energias de Tempo Curto
4. Cálculo dos Espectrogramas
5. Determinação do Pitch, da Frequência Fundamental e das Formantes (f_1 a f_4) de Algumas Vogais
6. Determinação de fonemas surdos, sonoros, consoantes gerais e plosivos

1. Gravar Arquivos de Áudio com dígitos, números gerais, texto lido e poesia

Usando o Audacity, ou outro programa de áudio, grave arquivos com:

- Dígitos
- Números Gerais
- Texto lido
- Poesia declamada

2. Leitura de Arquivos de Áudio e janelamento

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
import numpy as np
import matplotlib.pyplot as plt
import librosa
import librosa.display
import IPython.display
#OBS.: insira nome do arquivo WAV
audio1 = '/content/drive/My Drive/Colab Notebooks/Entre_Leva_Catia_Falada.wav'
```

```
print(audio1)
v1 , sr1 = librosa.load(audio1)
print(type(v1), type(sr1))
print(v1.shape, sr1)
# Player será aberto! AGUARDE até abrir!
IPython.display.Audio(data=v1, rate=sr1)


plt.figure()
fig, ax = plt.subplots(figsize=(15, 5))
librosa.display.waveplot(v1, sr=sr1)
plt.title('Voz da Catia' + audio1)
```

2.1 Definição dos Parâmetros da Análise

```
print('Frequência de Amostragem', sr1)


fa = sr1
print(fa)


Ts = 0.04   # Tempo de duração do segmento em segundos
Nj = int(Ts*fa) # Número de pontos da Janela
print('Tamanho do Segmento', Nj)


Nseg = int(len(v1)/Nj)
Nover = int(Nj*0.5)
print('Número de Segmentos no Arquivo', Nseg, '. Pontos Soprepostos', Nover )
```

2.2 Obtendo uma Janela de Hamming

```
from scipy import signal
hm = signal.get_window('hamming', Nj)
plt.plot(hm); plt.title('Janela de Hamming')
```

3. Energia de Tempo Curto

3.1 Cálculo para arquivo de teste

```
Nover = int(Nj*0.5)
Nseg = int(len(v1)/Nj)
E = []
for l in range(1, Nseg):
    xjan = v1[(l-1)*Nj+Nover:l*Nj+Nover]*hm
    x2 = list(np.array(xjan**2))
    aux = sum(x2)/Nj
```

```
    E.append(aux)
E = 10*np.log10(E)
Emin = np.min(E)  # calcula nível de ruído de fundo
plt.figure
fig, ax = plt.subplots(figsize=(15, 3))
plt.plot(E - Emin)
plt.title('Energia da Voz' + audio1)
plt.ylabel('Energia[db]'); plt.xlabel('Segmento')
```

3.2 Agora leia os seus arquivos gravados e:

Determine o Contorno de Energia para três arquivos que vocês gravaram:

1. *dígitos*
2. *voz falada*
3. *voz declamada (poema)*

4. Espectrogramas

4.1 Visualização do Espectrograma para Arquivo de Teste

```
import matplotlib.pyplot as plt
plt.figure(figsize=(12, 8))
D = librosa.amplitude_to_db(np.abs(librosa.stft(v1)), ref=np.max)
fig, ax = plt.subplots(figsize=(15, 10))
librosa.display.specshow(D, x_axis='time',y_axis='linear')
plt.colorbar(format='%+2.0f dB')
plt.title('Potência e Espectrograma Linear na Frequência'+ audio1)
```

4.2 Agora com os seus arquivos

Faça os espectrogramas para os arquivos que vocês gravaram e calculem o contorno de energia:

- Dígitos
- Texto Lido
- Poesia

5. Determinação do Pitch e da F0

5.1 Pitch do Arquivo Teste1

```
# Segmente um fonema que tenha Pitch, p.ex "En" de "Entre"
v1En = v1[4000:16000]
IPython.display.Audio(data=v1En, rate=sr1)
```

```
from matplotlib.ticker import (MultipleLocator, FormatStrFormatter, AutoMinorLocator)
acEn = librosa.autocorrelate(v1En, max_size= sr1/32)
fig, ax = plt.subplots(figsize=(15, 5))
ax.xaxis.set_major_locator(MultipleLocator(1))
y1 = acEn[1:400]
x1 = range(len(y1))
xx = [i*1000/sr1 for i in x1]
plt.grid(True)
ax.plot(xx,y1)

plt.title('Auto-correlação 1')
plt.xlabel('tempo em [ms]')
plt.show()
```

O Período de Pitch (T_0) será o intervalo entre picos sucessivos.

A Frequência Fundamental (f_0) será o inverso do periódo de Pitch

5.2 Pitch do Arquivo Teste2

```
# Segmente outro fonema que tenha Pitch, p.ex "Ag" de "Agora"
v1A = v1[48000:56000]
IPython.display.Audio(data=v1A, rate=sr1)
```

```
acA = librosa.autocorrelate(v1A, max_size= sr1/32)
fig, ax2 = plt.subplots(figsize=(15, 5))
ax2.xaxis.set_major_locator(MultipleLocator(1))
y2 = acA[1:500]
x2 = range(len(y2))
xx2 = [i*1000/sr1 for i in x2]
plt.grid(True)
ax2.plot(xx2,y2)

plt.title('Auto-correlação 2')
plt.xlabel('tempo em [ms]')
plt.show()
```

Determine o Período de Pitch e a Fundamental desta vogal.

Agora com cada um dos seus arquivos:

Determine o Pitch e a f_0 da vogais que desejarem dos seus arquivos, para as versões:

1. Dígitos
2. Falada
3. Declamada

5.3 Determinação das Formantes

5.3.a Formantes do trecho de teste

```
# AUDIO DE "En" em v1En
f, Pxx_spec = signal.periodogram(v1En, fa, 'flattop', scaling='spectrum')
lf = len(f)
fig, AX = plt.subplots(figsize=(15, 5))
AX.xaxis.set_major_locator(MultipleLocator(100))
AX.plot(f[:int(lf/4)], 10*np.log10(np.sqrt(Pxx_spec[:int(lf/4)])))
plt.xlabel('Frequência [Hz]')
plt.ylabel('Log-spectro [dB]')
plt.title('Espectro da Vogal /En/ de <Entre> de '+audio1)
plt.grid(True)
plt.show()
```

5.3.b Os quatro primeros picos no espectro são as formantes f_1 a f_4

5.4 Agora com seus arquivos

Determine as formantes das vogais dos arquivos que vocês calcularam os espectrogramas

6. Determinação de Consoantes e Plosivos

Usando alguns de seus arquivos, no espectrograma, determine algumas consoantes e em especial as plosivas