Lab 5 - Solution

November 20, 2019

1 Lab 5 - Solution

This lab will work through saving the top n magnitude values at each frame in the STFT to build a very basic sinusoidal encoder.

Download the Voices folder from Bb Learn and place in same directory as this notebook to get started

1.1 Import Libraries

```
In [3]: import librosa
        import numpy as np
        import IPython.display as ipd
```

1.2 Load Audio File

```
In [4]: filename = "Voices/male1.wav"
        y, sr = librosa.load(filename)

In [5]: ipd.Audio(y, rate = sr)

Out[5]: <IPython.lib.display.Audio object>
```

1.3 Compute STFT

```
In [6]: # set the hop size
        hop= 128

        #compute the stft of y and store it in D
        # take the transpose of D so that time frames is the first dimension
        D = librosa.stft(y, hop_length=hop).T
```

1.4 Create Encoded Array

```
In [7]: # n is the number of frequencies to save
        n = 50

        # create an empty array a to be our encoded stru
        a = np.zeros((D.shape[0],2,n))
```

1.5 Encode Audio

We will save the magnitude of the top n frequencies

```
In [8]: index = 0
        for dft in D:

          # get mag spec
          mag_spec = np.abs(dft)

          # create list of indices
          freq = (np.arange(len(dft))/len(dft)) * (sr//2)

          # sort magnitude spectrums and frequencies by descending magnitude
          mag_spec, freq = (list(t) for t in zip(*sorted(zip(mag_spec, freq),reverse = True))

          # get top n mags and freqs
          A = mag_spec[:n]
          F = freq[:n]

          # store them in a
          a[index][0] = A
          a[index][1] = F

          #increment index
          index += 1
```

1.6 Create Decoded Array

```
In [9]: #create output vector
        y_out = np.zeros((1,0))
```

1.7 Decode Audio

Reconstruct using sum of weighted sinusoids

```
In [10]: for frame in a:

           A = frame[0]
           F = frame[1]

           dur = hop / sr

           t = np.linspace(0,dur,int(dur*sr))

           F.shape = (n,1)
           A.shape = (1,n)
           t.shape = (1,len(t))
```

```
    # Generate matrix of sinusoids
    sine_mat = np.cos(2 * np.pi * F * t)

    # Weight by amplitude vector A and sum
    sine_sum = np.dot(A,sine_mat)

    # append sine sum onto y_out vector
    y_out = np.concatenate((y_out,sine_sum),axis=1)
```

1.8 Listen to the Decoded Signal

```
In [11]: ipd.Audio(y_out, rate = sr)
```

```
Out[11]: <IPython.lib.display.Audio object>
```