Coursera: Audio Signal Processing for Music Applications
Week 10, Peer-graded Assignment: A multi-resolution sinusoidal model, *by Omer Sayli*

In this assignment, we were asked to perform a multi-resolution sinusoidal model for sound analysis and synthesis. We were given the sine model version without tracking in the sms-tools package (namely sineModel.sineModel() function). Copying that function, following changes were added

- For each 'hop', three data frames were calculated for the given M1, M2, M3 sizes.
- For each 'hop', DFT analysis is done for the three data frames with the given N1, N2 and N3
- Peaks of the magnitude spectrum of the three DFTs for the three data frames are calculated
- Calculated peaks were considered if they fall in the desired frequency bandwidth ([0-B1] Hz, [B1-B2] Hz, [B2-B3] Hz, respectively for the three particular data frame width.
- Synthesis of the sound for the particular hop is done for the accepted magnitude peaks

First I have studied first the "**orchestra.wav**" sound file (fs=44100 Hz, mono) in the "**sms-tools-master\sounds**" folder as suggested by the question. First, I get the output for the pure sinusoidal model with the parameters blackman window with size 4096, M=N= 4096 and t=-100 ('**output_sin_orchestra.wav**'). Using SonicVisualizer, frequency content is rich for low frequencies and high frequencies, especially below 5000 Hz. *The sound has both percussive sounds and melodic content, so I choose parameters to capture both low and high frequency content.* The multi-resolution sinusoidal model was used with the following parameters : window type "blackman", M3=N3= 1024, M2=N2= 2048, M1=N1= 4096, B1= 500, B2= 3000, B3= 22050, t= -100 (**'multi_sin_orchestra_1.wav'**). Here, using lower frequency (B1= 500 Hz) for the longer window as well as using higher frequencies for the shorter window make it possible to capture ***low frequency sounds***, ***strokes-sudden sounds*** and ***high frequency content at the same time***. I especially noticed "low frequency drums" to be stronger than the analysis made with "sineModel()" function which uses only one window . The 'sine' model uses single window, hence 'same' frequency resolution.

However, for the parameters: M3=N3= 1024, M2=N2= 2048, M1=N1= 4096, B1= 100, B2= 1000, B3= 22050, t=-100 (**'output_multi_sin_orchestra_2.wav'**), I noticed some 'pulse noises' at the start of the output sound. Hence, selection of proper window lengths and bandwidth edges are important.

The 2nd sound I used I is this (https://freesound.org/people/dshoot85/sounds/331025/) Orchestral_Loop_with_Oriental_Touch_Mono.wav , fs=44100 Hz, mono). Again, I analyzed this sound first using SonicVisualizer. This sound also has 'sudden' percussive instruments as well as melodic content. First, I get the output for the pure sinusoidal model with the parameters blackman window with size 4096, N= 4096 and t=-100 ('**sin_orchestral_loop.wav**'). This sound has strong content below 1500 Hz, and 6000 Hz. Hence used parameters for multi-resolution sinusoidal model are blackman window with N3=M3 = 512, N2=M2 = 2048, N1=M1 = 4096, B1 = 1500,B2 = 5000, B3 = 22050, t = -100. The output sound has superior quality in both capturing sudden percussion sounds and melodic ones. The pure sinusoidal model output has 'less quality' compared to the multi-resolution sinusoidal model output as expected (i.e. even percussions have less magnitude)

Computational complexity of the multi-resolution sinusoidal model was more than the sinusoidal model. But this added burden was for the better time and frequency resolution obtained at the same time.

If I were to extend this multi-resolution sinusoidal model to HPR or HPS models, finding the harmonics would be harder since we have three bandwidths to consider which are computed for different window lengths. Algorithm to find harmonics should be able to deal with the peaks found from three distinct bandwidth analysis. Also these peaks are found from different window lengths, a harmonic found in a window may not be continued in longer window. The problem would be also persistent for F0 tracking. Tracking could be performed for the considering shortest window