Digital Signal Processing
Exercise 11

Submission deadline: Thursday, 14.1.2015

I suggest going over the matlab files located beneath Recitation 9 pdf.

In this exercise we will analyze a sound file and create a filter which changes with time for separating a recorder from the background.

1. We will be using 3 test files in this work. You can use the command *wavread* to load them and *sound/wavplay* to play them. The files are:
   (a) dScale.wav - A diatonic scale played by a recorder.
   (b) dScaleMix.wav - A diatonic scale with background sounds.
   (c) littleJohnnyMix.wav - Our favorite song with background sounds.
2. Each sound we produce is a signal containing a wave frequency f (the main frequency of the sound, which we will call the first harmonic) and a number of frequencies which are integer multiples of the main frequency - 2f,3f etc. These waves are called the harmonics of the sound. Sounds of the same main frequency, produced by different instruments differ in the ratios of the amplitudes of their harmonics.
3. The end goal of this assignment is to design the following function:

   [y]=getMusic(x,fs,pitch,harmonicsNumber)

   The function has four inputs:
   x - a vector with the signal
   fs - the sample rate (the first two inputs are similar to the output of *wavread*)
   pitch - a 2 dimensional vector. The first column contains a timestamp (in seconds) while the second one is the first harmonic frequency (in Hz). For example:
   pitch = [0,440;0.3,800;0.6,600];
   Meaning: from 0 seconds to 0.3 seconds the first harmonic is is 440 Hz. From 0.3 seconds to 0.6 seconds the first harmonic is 800 Hz, and from 0.6 to the end of the file the first harmonic is 600 Hz.
   harmonicsNumber - the number of harmonics that should be taken for each note. Meaning if harmonicsNumber=3 and the pitch is 500 Hz. Frequencies 500,1000 and 1500 will be used.
   The output y is the filtered signal.

4. As a utility function you should make a function that makes a multi-band FIR filter.

   [Hd]=makeFilter(f,fs,deltaF,harmonicsNumber)

   Inputs:
   f - The first harmonic frequency. (which will be the value taken from the pitch vector)
   fs - sampling rate (which is one of the outputs of *wavread*)
   deltaF - the width of the pass band. (normalized between 0 to 1)
   harmonicsNumber - number of harmonics used.
   You can add optional inputs to this function if needed (such as N, the filters order) but they should all have default values. (the function exCheck.m should work as is)

   Outputs:
   Hd - is a FIR filter. It can be an object or the vector b of a FIR filter.
   You can use the function *freqz* to view your filter.
   Using these FIR filters we will separate the harmonics from the background.
   You can look at the code on the web (what we used in the tirgul) for an example of making a multi-band FIR filter.

   Note: as we saw in class there is a delay once you use a filter. In order overcome this issue you should overlap the sections when you filter them. Meaning if you have a pitch value for 2 to 3 seconds, you should probably filter for a bit more e.g. from 1.9 to 3.1 seconds, and cut the extra.

5. Since the signals main frequencies change with time we will want to view the Short-Time Fourier Transform (STFT) of the signal. Examine the signals using the function *specgram/spectrogram* (spectrogram is the newer function in Matlab). Read the help of the function in order to understand the different parameters.
   Although automatic methods can be developed, for this assignment we will use this function to manually create a pitch matrix for dScale.wav and dScaleMix.wav (it is the same pitch matrix).The matrix size(pitch) should be 8x2. Use the different function parameters (window size, window type ...) of the function for examining the STFT. The pitch matrix for littleJohnnyMix.wav is given (in littleJohnnyPitch.mat).

6. Save as jpg files figures that you find most useful (1-3 plots for each of the signals dScale.wav, dScaleMix.wav and littleJohnnyMix.wav). Add to the title the parameters you used. You can include zooming images.

Usefull Matlab functions: *fir1*, *filter*, *fvtool*.

On the web you can find the following files:

1. dScale.wav - A diatonic scale played by a recorder.
2. dScaleMix.wav - A diatonic scale with background sounds.
3. littleJohnnyMix.wav - Our favorite song with background sounds.

4. littleJohnnyPitch.mat - pitch file for littleJohnnyMix.wav.
5. dScaleMixExp.wav - an example of the filtered signal.
6. littleJohnnyMixExp.wav - an example of the filtered signal.
7. exCheck.m - a code that checks some features of your exercise.
8. myUname.m - a template code to be filled with your names and id's.

What to hand in:
This is a computer exercise and it should be zipped and submitted by email to dsp@cs with subject "ex9". In the zip you should have the following files:

1. You should create about 3 to 8 specgram jpeg's for dScale.wav, dScaleMix.wav and littleJohnnyMix.wav. Add to the title the parameters you used.
2. A readme.txt file with no more than 10 lines giving your basic guidelines for using the spectram images, finding the pitch for dScale, and implementing the filter.
3. The file dScalePitch.mat. The file should have the variable pitch which is the pitch for dScale. The pitch for littleJohnnyMix is given to you.
4. makeFilter.m
5. getMusic.m
6. myUname.m - filled with your names and id's.
7. Any other functions you wrote.

Before you hand in your exercise, make sure that the function 'exCheck.m' works properly with your code.