Introduction to Digital Signal Processing 67567 **2019-2020**

Exercise 10 - MATLAB Assignment

Due: 26/01/19 23:59

In this exercise we will analyze and filter sound files in order to separate a recorder (חלילית) from unwanted background sounds. This will be achieved by creating an FIR filter which changes with time.

The Sound Files

We will be using 3 test files in this exercise. You can use the command `audioread` for loading the files and `sound` for playing them on your speaker. The files are:

1. `dScale.wav` - A diatonic scale played by a recorder.
2. `dScaleMix.wav` - A diatonic scale with background sounds.
3. `littleJohnnyMix.wav` - יונתן הקטן with background sounds.

Each sound in the recorder is a wave with frequency f, the main frequency of the sound, and a number of frequencies which are integer multiples of the main frequency $2f, 3f$ etc. These waves are called harmonics and f is called the first harmonic. Sounds of the same main frequency, produced by different instruments differ in the ratios of the amplitudes of their harmonics.

Part 1 - STFT

The main frequency in the signals above is not constant. We will use Short-time Fourier Transform (STFT) in order to find the frequencies as a function of time.

Spectrogram Analysis

Examine the signals using the command `spectrogram`. Read the documentation of this function in order to understand its input arguments. Use different parameters such as window type, window size, overlap etc. for better visualization of the main frequencies and their harmonics.
For each of the 3 files, save 2-3 figures of useful (or interesting) spectrograms with different parameters. Give informative file names and explain the figures in the README file.

Pitch Matrix

A pitch matrix used in this exercise is defined as follows: an $N \times 2$ matrix where the first column contain timestamps (in seconds) and the second column contains the first harmonic frequency (in Hz). For example:

$$\text{pitch} = \begin{bmatrix} 0 & 440 \\ 0.3 & 800 \\ 0.6 & 600 \end{bmatrix}$$

The meaning of this matrix is that from 0 to 0.3 seconds the first harmonic is 440 Hz, from 0.3 to 0.6 seconds the first harmonic is 800 Hz and from 0.6 seconds until the end of the sequence the first harmonic is 600 Hz. Although automatic methods can be developed, for the simplicity of this assignment we will **manually** create the pitch matrix using spectrograms.
Generate a pitch matrix of size 8×2 for `dScale.wav` and `dScaleMix.wav` and save it as `dScalePitch.mat`. The pitch matrix for `littleJohnnyMix.wav` is given to you in the file `littleJohnnyPitch.mat`.

Part 2 - Creating an FIR Filter

The next step is to create a multi-band FIR filter for the desired frequency bands. Write the following function:

```
[H] = makeFilter(f, fs, deltaF, numHarmonics)
```

- **Input:**
  - `f`: The frequency of the first harmonic. The values for this argument are taken from the pitch matrix.
  - `fs`: The sampling rate.
  - `deltaF`: The width of the pass band (normalized between 0 to 1).
  - `numHarmonics`: The number of harmonics to use. For example if this value is 3 and $f = 500$ then pass bands will be used for the frequencies $500, 1000, 1500$.
- **Output:**
  - `H`: The FIR filter. It can be object or the vector `b` of the coefficients of the FIR filter.
- We will study methods for FIR filter design. In the meantime, use the supplied example for learning how to create a multi-band FIR filter using `fir1` command. Further reading about FIR filter design in MATLAB can be found here:

  https://www.mathworks.com/help/signal/ug/fir-filter-design.html
- You can use the commands `freqz` and `zplane` to view your filter.
- Note: You can add optional arguments to this function but they should also have default values. The file `test.m` should work as is.

Part 3 - Filtering the Sequences

The final goal of this assignment is to filter the provided sequences and remove the background sounds. Write the following function:

```
[y] = getMusic(x, fs, pitch, numHarmonics)
```

- **Input:**
  - `x`: A vector of the input signal.
  - `fs`: The sampling rate (from `audioread`).
  - `pitch`: An $N \times 2$ pitch matrix as described above.
  - `numHarmonics`: The number of harmonics to use.
- **Output:**
  - `y`: The filtered signal.
- Note: As we discussed in class, the filter causes a delay in the output signal. In order to overcome this issue it is recommended to overlap the sections when you filter them. For example, if you have a certain pitch value from 2 to 3 seconds, you should probably filter a longer range such as 1.9 to 3.1 seconds (depends on your filter order) and take only the last one second from the output.

List of Provided Files

1. `dScale.wav`, `dScaleMix.wav`, `littleJohnnyMix.wav`: The signals you should work with.
2. `dScaleMixExp.wav`, `littleJohnnyMixExp.wav`: Examples of the expected filtered signals.
3. `littleJohnnyPitch.mat`: Pitch matrix for `littleJohnnyMix.wav`.
4. `test.m`: A code with basic testing of your implementation.

Submission Guidelines:

1. You should submit a zip/tar file with your MATLAB code.
2. You can submit the exercise **in pairs or alone**.

   ⇒ Don't forget to remind your partner (and yourself) to fill the Teaching Evaluation Survey (available here: http://students.huji.ac.il/)
3. Submit the following files:
   - Spectrogram figures for each signal (2-3 each) with informative file names and titles. Save the figures in an image format (`jpeg`, `png`) and not as `fig` files.
   - `dScalePitch.mat`: this file should have a matrix named `pitch` as in the provided `littleJohnnyPitch.mat`.
   - `makeFilter.m`
   - `getMusic.m`
   - `README.txt`: A file with explanations regarding the spectrogram figures, the pitch matrix and considerations taken in your implementation. Add your names and IDS at the beginning of the file.
   - Any additional MATLAB files you need for you code.