Assignment: Homework 3

What to Hand In

1. Your code with appropriate comments and a **readme file**. You are responsible for making us understand and know how to run your code.
2. A report with answers, figures, and explanations.
3. Any sound files asked for in the problems.

How to Hand It In

1. Put all your files in one folder. Compress this folder and name it <firstname>_<lastname>_HW3.zip. For example, "Zhiyao_Duan_HW3.zip".
2. Submit to the corresponding entry on Blackboard**.**

When to Hand It In

It is due at 11:59 PM on the date specified on the course calendar. **Late assignments will receive a 20% deduction of the full grade each day.**

Problems (10 points in total)

1. (4 points) Onset Detection
    a. (0.5 points) Implement an energy-based onset detection function. The function syntax is below. Note that the estimated onsetStrength curve is normalized so that its maximum value is 1 and minimum value is 0. This will make it easier to set the threshold th. Onsets are chosen by thresholding and peak-picking, i.e., local maxima that are higher than the threshold are accepted as onsets.

```
function [onsets, onsetStrength] = onset_energy(x, win, hop, th)
% Energy-based onset detection
%
% Input
%   - x        : audio waveform
%   - win      : window function
%   - hop      : window hop size (in samples)
%   - th       : global threshold to determine onsets
% Output
%   - onsets              : frame indices of the onsets
%   - onsetsStrength      : normalized onset strength curve, one value per frame,
range in [0, 1]
```

    b. (0.5 points) Implement a function that takes an onset vector as input and synthesize one frame of white noise for each onset and add the noise to the original audio signal. In this way you can listen to the audio signal together with detected onsets. The syntax of the function is below.

```
function y = synth_onset(x, frameLen, frameHop, onsets)
% Synthesize each onset as a 1-frame long white noise signal, and add it to
% the original audio signal.
```

```
%
% Input
%   - x            : input audio waveform
%   - frameLen  : frame length (in samples)
%   - frameHop  : frame hop size (in samples)
%   - onsets     : detected onsets (in frames)
% Output
%   - y            : output audio waveform which is the mixture of x and
%   synthesized onset impulses.
```

c. (1 point) Run your “onset_energy” function on “Pop.wav” with appropriate parameters. You might need to tune the threshold. Plot the onset strength curve together with onsets displayed as vertical bars. Run “synth_onset” function to add sonication of detected onsets to the audio. Save the modified audio as “Pop_onset_energy.wav”. Listen to it and discuss your results.

d. (1 point) Implement a spectral-based onset detection function. Again, you might want to normalize your onsetStrength values to the range of [0, 1] to make it easier to choose the threshold. Onsets are again chosen through thresholding and peak-picking. Here you might want to use a local threshold together with the global threshold. **Hint:** In your onsetStrength curve, besides the large peaks corresponding to onsets, you may also find many relatively small peaks (but still pretty high) that are not corresponding to onsets. You can remove many of them by subtracting the local mean from the onsetStrength curve. The syntax of the function is below.

```
function [onsets, onsetStrength] = onset_spectral(x, win, hop, th, gamma)
% Spectral-based onset detection (spectral flux). The spectral flux calculation is
based on the compressed magnitude spectrogram: spectrogram_compressed = log(1
+ gamma * spectrogram). This is to reduce the dynamic range of the spectrogram.
%
% Input
%   - x         : audio waveform
%   - win       : window function
%   - hop       : window hop size (in samples)
%   - th        : threshold to determine onsets
%   - gamma : parameter for spectrogram compression
% Output
%   - onsets                : frame indices of the onsets
%   - onsetsStrength    : normalized onset strength curve, one value per frame,
range in [0, 1]
```

e. (0.5 points) Run your “onset_spectral” function on “Pop.wav” with appropriate parameters. Plot onsetStrength curve and onsets. Generate and save the modified audio that contains the onsets as “Pop_onset_spectral.wav”. Listen to it and discuss your results.

f. (0.5 points) Run your energy-based and spectral-based onset detection function on both “Shumann.wav” and “Haydn.wav” with appropriate parameters. Save the modified audio files as “[audiofilename]_onset_[detectionmehtod].wav”, and listen to them. These two files are more challenging and your results might not be satisfactory. Analyze your results and discuss why your results are good or not good. Suggest how to improve your results.

2. (2 points) Tempogram and tempo estimation.
    a. (1 point) Perform STFT on the onset strength curve calculated using your “onset_spectral” function on “Pop.wav” to get its tempogram. You can use “spectrogram” function to do so. What is the sampling rate of your onset strength curve? Choose your window size to be between 2 and 3 seconds, and 50% overlap. Use 4-times zero padding. Visualize your tempogram. The horizontal axis of your tempogram should be time in the unit of second, and the vertical axis should be frequency in the unit of BPM. Label your tempogram appropriately.
    b. (1 point) The “fundamental frequency” of the onset strength curve corresponds well to the tempo of the piece. By inspecting your tempogram, what is the fundamental frequency, i.e., tempo of the piece? Listen to the piece and verify your estimation result.

3. (4 points) Beat Tracking by dynamic programming.
    a. (3 points) Implement the dynamic-programming-based beat tracking approach introduced in class. The syntax of the function is below. The beat spacing penalty function is $P_{\hat{\delta}}(\delta) := -\left(\log_2(\delta/\hat{\delta})\right)^2$, where $\hat{\delta}$ is the beat spacing of the expected tempo.

```
function beats = beat_dp(wavData, onsetStrength, tempoExpected, lambda)
% beat tracking by dynamic programming.
%
% Input
%   - wavData           : the audio signal
%   - onsetStrength     : onset strength in each audio frame
%   - tempoExpected     : the expected tempo (in BPM)
%   - lambda            : tradeoff between the onset strength objective and beat
regularity objective
% Output
%   - beats             : the estimated beat sequence (in frame number)
```

    b. (1 point) Run your “beat_dp” function on the onset strength curve detected by your “onset_spectral” function on “Pop.wav”. Synthesize the beats and add them to the original audio and save it as “Pop_beats.wav”. Listen to it. Are your satisfied with your beat tracking results?