Assignment: Homework 1

How to Hand It In

1. Put all your solutions (report, code, audio, etc.) in one folder. Compress this folder and name it <firstname>_<lastname>_HW1.zip. For example, "Zhiyao_Duan_HW1.zip".
2. Submit it 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. (2 points) Name ten computer audition applications. Briefly describe each application with a couple of sentences.

2. (5 points) Practice basic audio signal processing techniques using Matlab. You should submit all your code and audio files. **Your code should be well commented.**
   a. (1 point) Download Audacity (http://audacity.sourceforge.net/). Read the following sentence and record it with Audacity into a single-channel (mono) audio file with bit depth of 16 and sample rate of 44100Hz. You can specify the parameters in Edit->Preferences. Speak clearly and not too fast. After recording, delete the extra silence before and after your voice if needed. (Having a little silence before and after is good though.) Save the file as "myvoice.wav".
   "My name is (Your full name), and I love computer audition."
   b. (1 point) Read your "myvoice.wav" into Matlab (using the **audioread** function). Zero mean and calculate its RMS value. Also read "noise.wav" into Matlab. Take the same length as your voice. Zero mean and calculate its RMS value. Now scale your voice data to make the signal-to-noise ratio (SNR) to 5dB. What is the RMS value of your scaled voice? Add your scaled voice with the noise and save the mixture as "myvoice_noisy.wav" (using the **audiowrite** function). Scale the mixture if needed to avoid clipping.

   $\sqrt[4]{10} \cdot rms\_noise = 0.0933$

   2084

   c. (1 point) Downsample your clean voice data to 16kHz (using the **resample** function). A low-pass anti-aliasing filter is already implemented in this function. Take a 512-point non-silent segment of your data. Apply a 512-point hamming window. Then perform FFT with 4-times zero padding. Plot the magnitude spectrum (only the positive frequencies, should contain 1025 points). Label the horizontal axis in frequency (Hz) and vertical axis in amplitude (dB). How long is this window in milliseconds? What is the (real) frequency resolution?

   ??? ① $\frac{512}{16k} = 32ms$

   ② $\frac{16k}{512} = 31.25$ Hz

   d. (1 point) Segment your downsampled clean voice with Hamming window with length of 512 points and hop size of 256 points. Perform STFT in each segment. Stack all the magnitude spectra into a spectrogram. Show the magnitude spectrogram (using the **imagesc** function). Label the horizontal axis in time (second)

   35-36

and vertical axis in frequency (Hz). You can also use the **spectrogram** function for this question.

e. (1 point) Zero all the **linear-amplitude spectra** outside the frequency range [300Hz, 3400Hz]. This is the frequency range that is used in telephone signals. Convert the new spectrogram back to the time domain using Overlap-Add technique with the original phase. Save the time domain signal as “myvoice_telephone.wav”. Does it sound like a voice coming out of a telephone? What if you use zero phase for the entire spectrogram? Save the time domain signal as “myvoice_telephone_zerophase.wav”.

3. (3 points) Make a Shepard tone to practice the additive-synthesis technique.
   a. (1 point) Synthesize a complex tone with six sinusoids, whose frequencies are 110, 220, 440, 880, 1760, and 3520Hz, respectively. Their peak amplitudes are the same. The sampling rate should be 44,100Hz. The length of this tone is 3 seconds. Save this sound as “complextone.wav”. Scale it if needed to avoid clipping. Calculate the MIDI number of these frequencies using the following formula:

$$\text{MIDI number} = 69 + 12\log_2 \frac{Hz}{440}$$

   Each MIDI number corresponds to a piano key (i.e. a music half step). The MIDI number of 440Hz is 69.
   b. (1 point) Make a Gaussian function in the MIDI number scale whose center is at MIDI number 81 and standard deviation is 12. Apply this function to the six sinusoids to shape their peak amplitudes, i.e. the peak amplitudes of the sinusoids follow this Gaussian shape. Save this sound as “complextone_Gaussian.wav”.
   c. (1 point) Make a series of complex tones. Each tone is 0.5 second long, followed by a 0.5 second silence. The first tone is what you made in b (but shorter). In the second tone, move the frequency of all the six sinusoids a half step upward. That is, their MIDI numbers are all increased by 1. Their amplitudes are determined by the same Gaussian function. Note that the mean and variance of the Gaussian function is fixed. In the third tone, the frequencies of all the sinusoids are again moved a half step upward. Do this until the 12^{th} tone. Then copy all the 12 complex tones you have made and paste them for 4 times. You now have a series of 12*5 complex tones (spaced with silence). Save this sound as “Shepardtone.wav”. Play it back and listen. It sounds like the pitch is always going up.

amp → Gauss (σ 12, ε = 82)

0.5s b → MIDI + 1 →

~~frequency~~

tone : 0.5s + 0.5s silence