UVIC COMPUTER SCIENCE 575, Fall 2017
ASSIGNMENT #1

DUE Octobr 02, 2017

There are 2 question each worth 50 points. They are marked with stars as follows: **(*)** basic **(**)** expected **(***)** challenging. The assignment is worth 10% of the final grade. There is some variance in the amount of time each question probably will require. Therefore don't expect them to be equally difficult even though they are all worth the same number of points.

Please provide your answers in a **SINGLE PDF** file through the Connex website for the course. Also please answer the questions in order and explicitly mention if you have decided to skip a question. For questions that involve programming provide a listing of all the relevant code and associated plots with sufficient documentation and naming of variables so that the marker is able to understand what you have done.

Question #1. (50 points)

The goal of this question is to familiarize you with sine waves and audio as well as the basic operations from which the DFT is built i.e the vector inner product and sinusoidal basis functions. You can use any programming language for your implementation but it will probably be easier to do in an environment that supports plotting such as Matlab/Octave or Python/NumPy/SciPy/Matplotlib. Feel free to use any code we examined during class to inform your your submission but make sure you understand completely everything you submit.

- **Q1.1 (5pts) (*)** Write two functions *peakAmplitude* and *peakRMS* that take as input an array of audio samples and return the peak amplitude and the RMS amplitude of the input. Recall that if the input contains several periods of a sinusoid then the peak amplitude of a sinusoid is approximately a and the RMS amplitude is approximately $a/\sqrt{2}$. Confirm that you can correctly estimate the amplitude of sinusoids that way by using as input sinusoids of frequencies $200Hz, 440Hz, 500Hz$ with amplitudes $0.5, 1.0, 2.0, 3.0$. Consider all combinations of the previously given frequencies and amplitudes and check that your two estimates are close in value to the corresponding input amplitudes.

- **Q1.2 (5pts) (*)** Write a function that makes a mixture of three harmonically related sinusoids with frequencies $f, 2f, 3f$ with user provided amplitudes and phases. Show a time domain plot of 1000 samples of adding three sinusoids with amplitude $1.0, 0.5, 0.33$ corresponding respectively to $f, 2f, 3f$ all with 0 phases and $f = 440$, and another plot with random phases.

- **Q1.3 (5pts) (**)** Using the function from **Q1b** generate one second of audio in .wav format for a mixture with $f = 440Hz$. Listen to the generated mixture using an audio editor like *Audacity*. Provide a figure showing the spectrogram of your mixture using *Audacity*. Also show a plot of the first 1000 samples of the input sinusoids as well as the two mixtures (same phase and random phase). Describe if you observe a difference between these two plots and if you hear a difference between the corresponding audio files.

- **Q1.4 (5pts) (*)** Read about the concept of signal to noise ratio (SNR). Write a function that takes as input a SNR in decibels (dB) and a frequency in Hz in order to generate a mixture of white noise and a 440 sine wave. Generate 2 seconds of the corresponding audio and view/listen to it in *Audacity* for different SNR levels.

- **Q1.5 (5pts) (**)** Consider another way of estimating the amplitude of a sinusoid signal when you know the frequency. Try taking the inner product of a sinusoid with amplitude a with a sinusoid of the same frequency and phase and unit amplitude. Explore what happens when you change a. What do you observe about this inner product ? More specifically come up with a formula for estimating the amplitude of a sinusoid by taking the inner product of a sinusoid with the same frequency, phase and unit amplitude.

- **Q1.6 (5pts) (***)** Determine by listening 5 resonable values of SNR for a $440Hz$ sinusoid mixed with noise (from noise barely perceptible to sinusoid hard to hear in the noise) and create the corresponding signals. Now using the three amplitude estimation methods (peak, rms, and inner product if frequency is known) estimate the amplitude for these 5 configuration and create a plot showing all the estimates for all the configurations. Which of the three methods is more robust and how did you arrive to that conclusion ?

- **Q1.7 (5pts) (**)** Consider a mixture of 3 harmonically related sinusoids as the ones you created. What you would like to devise is a process to estimate the amplitudes of each sine wave assuming that you know the frequencies that the mixture is composed of. Try taking the inner product of a mixture with a unit amplitude sinusoid of frequency f, then take the inner product of the mixture with a unit amplitude sinusoid of frequency $2f$. Explore what happens when you take inner product with frequencies that are not present in the mixture. Make sure that the phases of the "probing" sinusoids are the same as the phases of the mixture sinusoids. What do you observe about these inner products ? More specifically come up with a formula for estimating the amplitude of a sinusoid by taking the inner product of a sinusoid with the same frequency and phase but unit amplitude. Describe how you could modify this procedure to estimate the amplitude of mixtures of 4 sinusoids mixed with noise.

- **Q1.8 (5pts) (***)** Now let's make the scenario a little bit more challenging. We still know the frequencies of the mixture but we want to estimate not only the amplitudes but also the phases. Observe what happens with the inner products when the phase is not zero. Plot the inner products of the input mixture for all possible phase shifts of the probing sinusoid. On that plot do you notice anything different about the "correct" phase ? Based on your observation describe and also implement an algorithm for estimating both amplitudes and phases for mixtures. The process of taking sliding inner products described above is called **cross-correlation**. Show some examples of how your method works with appropriate plots and estimates.

- **Q1.9 (5pts) (**)** Create a simple synthesizer for melodies using sine waves. The input should consist of a list of tuples containing a note name (like C# or A) and a numerical value corresponding to relative duration (4 for quarter notes, 2 for half notes, 1 for whole notes, 8 for eight notes) as well as a tempo in beats-per-minute (BPM). Your code should convert the relative durations to absolute times, the note names to frequencies, generate the corresponding sine waves and output the result to an audio file. Render a simple popular melody using your synthesizer for two different tempos and show the corresponding spectrograms.

- **Q1.10 (5 pts) (***)** Let's make the syntesizer sound a little bit better. Read about how to create a triangle wave using additive synthesis. Modify your code so that instead of sine waves, it generates triangle waves. Read about Attack-Decay-Sustain-Release (ADSR) envelopes used in synthesizers and add such envelopes to your synthesizer. Listen to the audio files for the trianlge wave rendering with and without ADSR envelopes and comment on the difference.

Question #2. (50 points)

The goal of this question is to explore one of the most important tools in music information retrievial, signal processing, and engineering in general the Discrete Fourier Transform (DFT) and its computationally efficient implementation the Fast Fourier Transform. In addition we will continue looking into audio programming and processing sound in buffers.

- **Q2.1 (10pt) (*)** Write code to read/write data from a .wav file (you can use a library) in buffers of 2048. Verify that your code can read, apply simple processing (like applying a simple gain) and write audio files correctly. Show the corresponding time domain waveform plots in Audacity.

- **Q2.2 (5pt) (*)** Using any library or implementation of the Fast Fourier Transform for your programming language calculate the frequency domain complex spectrum of the 3 component mixture signal from question 1. Plot the magnitude spectrum.

- **Q2.3 (10pt) (*)** Using a programming language of your choice write code to directly compute the Discrete Fourier Transform (DFT) of an input array. You should express everything directly at low level using array access, *for* loops, and arithmetic operations i.e do not use a complex number type if the language supports it, and do not use any matrix multiplication facilities. Provide a listing of your code and a plot showing that your algorithm produces the same magnitude response as a Fast Fourier Transform routine in your language of choice that is either built in or freely available. Verify that your output is identical with the one by the FFT implementation you used.

- **Q2.4 (5pt) (**)**

  Modify the code that reads/writes data in buffers so that each buffer is converted to a frequency domain complex spectrum. Compute magnitude and phase spectrum for each buffer and plot an example for each type. Convert the magnitude and phase spectra back to a complex spectrum and then using the inverse DFT return to the time domain and write the buffer to an audio file. If everything is working correctly you should get back the original audio. Once you have verified that's the case then you can perform some simple spectral processing. Replace the phase spectrum with appropriate random numbers uniformly distributed. How is the resulting audio affected ? Try to describe what you hear especially when processing pieces of music. Experiment with large windows corresponding to 3-5 seconds. What happens ?

- **Q2.5 (5pt) (**)** Consider a sinusoid with frequency equal to one of the DFT bins (the array indices of the mangitude spectrum). What does the magnitude spectrum looks like for this input ? How does it change when you change the amplitude of the input sinusoid ?How does it change when you change the phase ?

- **Q2.6 (5pt) (***)** Plot the time domain waveforms of the two basis signals (the cosine and sine corresponding to the frequency of the bin) as well as the time domain signal corresponding to the point-wise multiplication of the input to these two basis functions (the part of the inner-product before the summation). Choose a number of samples that allows you to observe the amplitude and cycles of the corresponding signals. Now change the phase of the input sinusoid and check what happens to these two point-wise product signals. What do you observe ? How does this relate to the real and imaginary part of the spectrum for that bin ? What happens if the input is a sinusoid of a different frequency ?

- **Q2.7 (5pt) (**)**

  Read about the overlap-add of computing the Short Time Fourier Transform. Basically the idea is to window each buffer and read the data in overlapping chunks so that when they are summed the two windows sum up to 1. First ensure that you can do proper overlap-add by reading buffer that overlap by half and windowing them appropriately. Once you have that working apply the same process as the previous question of randomizing the phase spectrum. What has changed compared to before ?

- **Q2.8 (5pt) (***)**

  Measure how long it takes to compute 100 DFT transforms of 256, 512, 1024, and 2048 using your direct DFT implementation and compare your measured times with the ones using some implementation of the Fast Fourier Transform. Using a plot show the results of this comparison.