Digital Signal Processing (18-491/18-691)
Spring Semester, 2024

The 18-691 Final Project Short-Time Fourier Analysis (Preliminary Version):

Issued: 4/19/24

Due: 4/28/23 at midnight via Gradescope

Note: This version is preliminary in that it is lacking data and infrastructure resources. These files will be provided tomorrow, April 20.

Background: At the beginning of the semester we noted that students enrolled in 18-691 will be required to complete one additional assignment in order to receive graduate credit for the course. This is necessary in order to comply with ECE Department rules that state that courses in multiple tiers with parallel numbers must have at least somewhat different content to justify the parallel numbering.

The final assignment for the 18-691 students will be an implementation of two algorithms that implement popular audio effects for musical performance. The project is intended to be easy and fun.

The final project will be given a grade equal to one homework assignment. Nevertheless, the grade for the final project will not be dropped from the final grade calculation, no matter how low it is. Each 18-691 student will work on his or her own on this project. The deadline for the final assignment will be April 28.

Introduction: In this assignment you will implement a system that performs short-time Fourier analysis, which is a key enabling technology for most practical processing of speech and audio. STFA was discussed in class in Lecture 25, and the techniques used for this assignment were introduced in that lecture. The two recommended references (in addition to the class notes and the Powerpoint presentation from Lecture 25) are the notes on short-time Fourier transforms (STFTs) from 18-792 (ADSP) and the chapter on STFTs by Nawab and Quatieri in the Lim and Oppenheim edited text used in ADSP. Both of these references are available online on the Lectures page of the 18-491/691 course website.

As described in the ADSP class notes, STFA is used to enable us to analyze the frequency components of a signal as they evolve over time. In Secs. 3.2 and 3.3 of the ADSP class notes we describe

three ways of implementing the STFT: the Fourier transform implementation, the lowpass filter implementation, and the bandpass filter implementation. The STFT $X[n'k]$ of a time function $x[n]$ can be thought of as a matrix of complex coefficients, indicating the instantaneous power of the signal in discrete time frames (indexed by the horizontal axis of the matrix) and frequency bins (indexed by the vertical axis of the matrix). In this assignment, we will implement different versions of the STFT for different purposes.

Problem C10.1: (Spectrographic display)

(You will work this problem by completing the main file `main_10_1.m` that we provide.)

The **spectrogram** is the major tool used to visualize signals as a simultaneous function of time and frequency. In essence it is a display of the log magnitude of the STFT of a signal, plotted as a function of time and frequency. In this problem you will complete the MATLAB function we provide called `spectrogram_ham_691.m` that implements a spectrographic representation of a speech waveform using short-time Fourier analysis, displaying it using MATLAB. (You also complete your work in Python if you wish.) You can compare your result to that produced by the MATLAB routines `specgram` or `spectrogram`. You will accomplish this by implementing the system shown in Fig. 6.29 of Lim & Oppenheim (LO), which you will recognize as a realization of the bandpass-filter implementation, retaining the magnitude only of the STFT. Use Hamming windows for your analysis function. After obtaining the square of the magnitude (omitting the square-root step at the end of the block diagram, since we will plot the output in decibels), create a 2-dimensional matrix `X` with rows representing frequency and columns representing time. Use Hamming windows for your analysis function. Create two vectors, `t` and `f` representing the nominal frame times and center analysis frequencies, respectively. Display your output using the MATLAB commands

```
IMAGESC(t,f,10*log10(x));
axis xy;
COLORMAP(JET);
```

The speech wave that you will use as input is the utterance "Welcome to DSP-I" that we have used in class from time to time, but downsampled to a sampling rate of 8 kHz. Obtain the waveform `welcome8k.wav` on the available in the `h691_extras` folder on the homework page on the Web.

Note: Turn in the MATLAB scripts that you used to obtain your answers in all of the following problems.

(a) Given that you are using Hamming windows, what is the most efficient choice of the time between frames given the window length that would enable you to completely reconstruct the original time function? Consider this to be a downsampling problem, with the downsampling ratio determined by the effective cutoff frequency of the window as a lowpass filter, as discussed in Sec. 3.4 of the ADSP class notes. Explain your reasoning.

(b) Using the procedures specified above, plot the spectrogram for the DSP utterance separately using effective filter bandwidths (from the windows) of 50 and 350 Hz. State how you achieve these bandwidths using Hamming windows.

What you should turn in on paper:

- A block diagram and description of your system as you implement it. Be sure to state the critical choices you made in your window length, time between analysis frames, and other design decisions. State how you arrived at these decisions.
- Plots of the two spectrograms of part (b) using MATLAB
- Hard copy of your MATLAB scripts

What you will submit electronically on Gradescope:

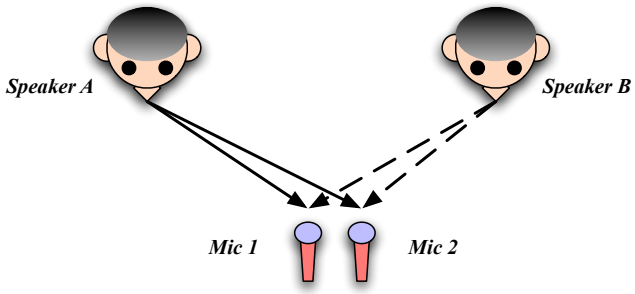
- Your spectrograms, in .pdf format. Label the files with your name and the suffix "`10_1a`" and "`10_1b`", as in "`rms10_1a.pdf`".

Problem C10.2: (Basic STFT analysis and resynthesis)

In this problem you will analyze a brief musical excerpt and subsequently resynthesize it using short-time Fourier analysis and synthesis. In this problem you will explore two analysis-synthesis methods, and consider the impact of reducing the number of numbers used in the representation on the signal distortion in the reconstructed signal.

[Detailed problem statement to be added.]

Problem C10.3: (Signal separation using STFTs)



In this problem we consider one way in which short-time Fourier analysis can be used in signal separation systems. With two microphones we can separate two incoming sound sources by *direction of arrival*, which can be estimated by comparing the instantaneous phase for each time-frequency segment of the two signals. For example, in the diagram above, speech from Speaker A will arrive at Mic 1 a few hundreds of microseconds before it arrives at Mic 2. For Speaker B, the reverse is true, and the sound will arrive at Mic 2 before Mic 1.

The file `brian_and_stef_2chan16K.wav` is a stereo recording of Brian and Stef, who had been placed in two symmetric locations relative to the two mics, as in the figure above.[1] You are asked

[1]Brian and Stef were students in the CMU Language Technologies Institute when these utterances were recorded. They graduated some time ago and began their careers (separately) at Microsoft Research. The utterances are part of the CMU Arctic Database, which has been used primarily for training text-to-speech systems. The CMU Arctic database was the primary resource used around the world for training text-to-speech (TTS) systems for the better part of a decade, although it has since been superseded by other resources. You might recognize one or two of the voices used in the database.

to separate their voices based on time of arrival. In case you are interested, one way of doing this is described in a paper from our group by the 18-792 TA from 2012, which may be downloaded at

`http://www.cs.cmu.edu/~robust/Papers/KimStern09.pdf`

The senior author (Chanwoo Kim) was a former student of mine who is presently an Executive Vice-President of Samsung. In this problem you will implement a *much* simpler algorithm that has the same goals using OLA analysis-synthesis techniques. The basic principle is that you will estimate the time delay between the sensors from the phase angle of arrival of the two signals, and you will segregate the signals by re-synthesizing them from only the subset of time-frequency segments that appear to be dominated from signals that come from a particular direction of arrival.

Please complete this problem by filling in the main file named `main_10_3.m` or `main_10_3.py` that we provide.

(a) Using Hamming windows of 20-ms duration and the minimum frame sampling rate in time that satisfies the OLA constraint, complete the MATLAB function `STFT_DFT.m` or the Python function `STFT_DFT.py` that we provide to obtain the 320-point DFT for each 20-ms segment for the signal in the left channel and the right channel of the stereo.wav file `brian_and_stef_2chan16K.wav`.

(b) For each coefficient in time and frequency, compare the phase angles from the two signals. Since the signal from Speaker A arrives at Mic 1 sooner than it does at Mic 2, what does that imply for the difference in phase of the STFT coefficients for the two signals? Consider what might happen as the delay exceeds half a period at higher frequencies. You may want to also consider whether phase unwrapping is helpful (or even necessary).

(c) Produce a signal that has Speaker A only (or at least an approximation to that result), by completing the MATLAB function `iSTFT_OLA.m` or Python function `iSTFT_OLA.py` that we provide to resynthesize the time-domain waveform using OLA techniques, using only the components of $X[n,k]$ for which the signal to Mic 1 appears to lead in time relative to the signal to Mic 2. (You can do this by setting the "other" components to zero and then applying the usual inverse transform/overlap/add approaches.) If you are curious, you can compare your separated signal to the "ideal" separated signals `brian.wav` and `stef.wav`.

Hints and comments: Remember that the coefficients $X[k]$ of any DFT of size N from $0 \le k \le (N/2)-1$ represent positive frequencies and the coefficients $X[k]$ for $N/2 \le k \le N-1$ represent negative frequencies. If the time function is real, the DFT coefficients will be $X[n,N-k] = X^*[n'k]$ because $X[n,k]$ is Hermitian symmetric with respect to k and periodic in k with period N. This means (among other things) that if Speaker A is obtained by selecting only those spectro-temporal segments for which the difference of phase is *positive*, the corresponding spectro-temporal segments for negative frequencies would have a phase difference that is *negative*. To be safe, you could observe the inter-mic phase difference of $X[n,k]$ only for channels $0 \le k \le N/2$ and calculate the coefficients of $X[n,k]$ for the remaining values of k by invoking Hermitian symmetry.

(d) Using similar techniques, produce a signal that is dominated by Speaker B only. Note that these techniques do not make use of any "oracle" information other than the putative locations of the two sound sources. This information is relatively easy to obtain, at least in "clean" conditions without very much noise or reverberation.

What you should turn in on paper:

- A brief description of what you did to get your results.

- Plots of the MATLAB or Python spectrograms of the two separated and reconstituted speech waveforms. Compare these with the corresponding spectrogams of the original utterances using the same analysis parameters.
- Hard copy of the MATLAB or Python scripts you used

What you will submit electronically:

- The two separated and resynthesized speech waveforms, in `.wav` format, similar to the input file. Label the files with your name and the suffix `10_3a` and `10_3b`, as in `rms10_3a.wav`
- The actual MATLAB or Python code that you used to achieve your results:
  - The completed main file `main_10_3` that we provided
  - The completed functions `STFT_DFT` and `iSTFT_OLA` that we provided