PROBLEM SET 5

Issued: 9/28/17
Due: 10/4/17

Reminder: Quiz 1 will be Wednesday November 18 in class. You may bring one sheet of notes to this exam.

Introduction: This assignment consists mostly of machine problems on short-time Fourier analysis and synthesis. While the machine problems may seem long, they are for the most part variations on the same system that is re-used from problem to problem.

Reading: We have continued our discussion of short-time Fourier analysis (STFA) as discussed in class, which follows the presentation in Lim and Oppenheim Chapter 6, and especially Secs. 6.1-6.4 of LO, which is available on the Web. Some supplementary notes on STFA are also on the Web. A complementary view of STFA with somewhat different notation may be found in Sec. 10.3 of Oppenheim, Schafer, Yoder, and Padgett (OSYP).

The next problem set (Problem Set 6) will include two problems on phase vocoding. While phase vocoding is skimmed over in Chapter 6 of LO, it is covered to some degree in Sec. 6.7.2 of Digital Processing of Speech Signals, published in 1978 by L. R. Rabiner and R. W. Schafer. This section, along with the original paper on phase vocoding by J. L. Flanagan and R. M. Golden (“Phase Vocoder,” Bell Syst. Tech. J. 45: 1493-1509 (1966) are available online. (Be aware of the varying notational conventions that are used, though!) I will also publish a condensation of my class notes on the subject.

After we conclude our discussion of short-time Fourier analysis we will begin a review of the basics of probability theory, followed by an introduction to random processes.

Problem 5.1: (This is a problem from Quiz 2 of a previous year. It is long but not difficult.)

In this question we will consider some issues related to sampling in the time and frequency domains in short-time Fourier transforms.

As you will recall, with discrete frequencies, the short-time Fourier transform (STFT) can be defined as
where \( N \) represents in this case the number of separate frequency channels used in the representation. In general we do not retain \( X[n, k] \) for every value of \( n \); we save the STFT only for values of \( n = rL \), or every \( L \) samples. In other words, we downsample \( X[n, k] \) in time by a factor of \( L \).

In class we discussed two ways of recovering an estimate of \( x[n] \) from \( X[n, k] \). Using the Filterbank Synthesis approach (FBS), we form an estimate of \( x[n] \) by summing the channel outputs over frequency.

\[
\hat{x}[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[n, k] e^{j2\pi nk/N}
\]

where \( N \) represents the number of frequency channels summed. This calculation is performed for every value of \( n \).

Using the Overlap Add method (OLA), we obtain \( x[n] w[rL - m] \) by computing the inverse DFT of \( X[n, k] \) at the frames for which \( n = rL \), and adding the (generally overlapping) inverse transforms over time.

\[
\hat{x}[n] = \sum_{r=-\infty}^{\infty} \frac{1}{N} \left[ \sum_{k=0}^{N-1} X[rL, k] e^{j2\pi nk/N} \right]
\]

In this question we will focus on sampling requirements in time and frequency using different window types, along with constraints needed to resynthesize.

Consider the three standard FIR windows and the two standard IIR windows specified on the next page. Note that the FIR windows have the free parameter \( M \), which is related to the length of the window. The IIR windows have the free parameter \( \omega_c \), which represents their cutoff frequency in the frequency domain.

<table>
<thead>
<tr>
<th>Window type</th>
<th>Sample response</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FIR Rectangular</strong></td>
<td>( w[n] = \begin{cases} 1, &amp; 0 \leq n \leq M \ 0, &amp; \text{otherwise} \end{cases} )</td>
</tr>
<tr>
<td><strong>FIR Bartlett</strong></td>
<td>( w[n] = \begin{cases} 2n/M, &amp; 0 \leq n \leq M/2 \ 2 - 2n/M, &amp; M/2 \leq n \leq M \ 0, &amp; \text{otherwise} \end{cases} )</td>
</tr>
</tbody>
</table>

Table 2. FIR and IIR windows specified in time and frequency.
Now let’s consider a specific example. Assume that speech has been sampled at a frequency of 16 kHz.

(a) Sketch the unit sample response of the three FIR windows for $M = 19$.

(b) Sketch the magnitude of the DTFT of the two IIR windows for $\omega_c = \pi/6$.

(c) **Sampling in time and frequency for FIR windows:**

1. For each of the three FIR windows, determine the maximum value of $L$, i.e. the maximum interval between successive values of $n$ that must be retained to enable $x[n]$ to be recovered from $X[n, k]$.
   (This answer may vary for different window shapes.) Express your answer in terms of $N_{win}$, the total length of the window. (The parameter $N_{win}$ equals $M + 1$.)

2. For each of the FIR windows, determine the value of $N$, the number of frequency channels needed to recover $x[n]$. Explain your reasoning.

3. For each of the FIR windows, determine the total number of values of $X[n, k]$ that must be retained for each second of the original speech waveform to enable $x[n]$ to be recovered.

(d) **Sampling in time and frequency for IIR windows:**

1. For each of the two IIR windows, determine the maximum value of $L$, i.e. the maximum interval between successive values of $n$ that must be retained to enable $x[n]$ to be recovered from $X[n, k]$.
   (Again, this answer may vary for different window shapes.) Express your answer in terms of the parameter $\omega_c$.

2. For each of the IIR windows, determine the value of $N$, the number of frequency channels needed to recover $x[n]$. Explain your reasoning.

3. For each of the IIR windows, determine the total number of values of $X[n, k]$ that must be retained.
for each second of the original speech waveform to enable \( x[n] \) to be recovered.

4. (e) Recovery using FBS and OLA

1. For each of the five window shapes, determine whether or not the FBS method can be used to recover the original signal \( x[n] \), assuming your answers to parts (c) and (d) to specify the sampling intervals in time and frequency for the STFT. Be sure to explain your reasoning.

2. For each of the five window shapes, determine whether or not the OLA method can be used to recover the original signal \( x[n] \), assuming your answers to parts (c) and (d) to specify the sampling intervals in time and frequency for the STFT. Again, be sure to explain your reasoning.

Problem 5.2:

Consider the analysis and synthesis of the signal \( x[n] = \cos(\omega_0 n) \). The analysis and synthesis networks using the lowpass implementation of STFA with real signals are shown in the figure above for the \( k^{th} \) channel where (as usual) \( \omega_k = \frac{2\pi k}{N} \).

(a) Determine \( a[n, k] \) and \( b[n, k] \) for the given input signal.

(b) Assuming that \( h[n] \) is a narrowband lowpass filter, simplify your expressions for \( a[n, k] \) and \( b[n, k] \) assuming that \( (\omega_0 - \omega_k) \) falls within the band of the filter, and that \( H(e^{j\omega}) \approx 1 \) for such frequencies.

(c) The signals \( a[n, k] \) and \( b[n, k] \) are combined to produce the magnitude \( M[n, k] \) and the phase derivative with respect to time \( \dot{\phi}[n, k] \). Obtain expressions for \( M[n, k] \) and \( \dot{\phi}[n, k] \) in terms of \( a[n, k] \) and \( b[n, k] \).

(d) Show that using the synthesis network in the right half of the figure at the beginning of the problem that the output signal is essentially identical to the input signal.

(e) The phase derivative signal \( \dot{\phi}[n, k] \) can be computed using the relation

\[
\dot{\phi}[n, k] = \frac{b[n, k]\dot{a}[n, k] - a[n, k]\dot{b}[n, k]}{(a[n, k])^2 + (b[n, k])^2}
\]
where the symbols $\dot{a}[n, k]$ and $\dot{b}[n, k]$ represent (ideally) sampled versions of the derivatives with respect to time of the continuous-time functions that would be obtained if $a[n, k]$ and $b[n, k]$ were converted to continuous time by an ideal D/C converter as discussed earlier in class.

1. Show that the above relationship is valid.

2. Compare the expression obtained for $\dot{\phi}[n, k]$ that you obtain with the above relationship with the corresponding one that you obtained in part (c) for the specific input signal $x[n] = \cos(\omega_0 n)$.

(f) Now assume that the derivatives of part (e) are computed using a simple first difference, i.e.

$$\dot{a}[n, k] \approx \frac{1}{T}(a[n, k] - a[n - 1, k])$$

where $T$ is the sampling period in the seconds. Now solve for $\dot{\phi}[n, k]$ and compare your results with part (c). Under what conditions are they approximately the same?

**MATLAB Problems**

**Problem C5.1:**

In this problem we will implement short-time analysis and synthesis using overlap-add (OLA) and filter-bank summation (FBS) techniques. We will use system parameters that would be typical of real analysis of speech.

Specifically, consider the “Welcome to DSP-I” utterance (welcome16k.wav), sampled at the usual 16 kHz. We will analyze the signal using Hamming windows of 20-ms, or 320 samples, in duration, a window size that is common for the implementation of automatic speech recognition systems that operate on 16-kHz speech waveforms.

(a) Let us consider first the OLA method. As you will recall, the STFT $X[n, k]$ is formed by multiplying the input waveform $x[m]$ by the window function $w[n - m]$ and computing the DFT of the product. As we discussed, we do not need to update the window function for every value of $n$, provided that the sum of the windows adds to a constant. This will occur, for example, with Hamming windows if the distance from frame to frame is half the duration of the Hamming window. Hence, with a 20-ms window (320 samples), we will calculate a new vector of $X[n, k]$ every 10 ms (160 samples).

1. Plot the magnitude of the values of the DTFT of $x[m]w[n - m]$, $X_1[n, k]$ you calculated as a spectrogram, using the techniques that you developed in Problem C4.2.

2. Reconstitute the time function $y_1[n]$ by computing the inverse DFT for all of the vectors of $X[n, k]$ and adding these time functions together delaying each one by successive increments of 10 ms. The sum of the time segments should be proportional to the original input.
(b) Now we will do the analysis and synthesis using the FBS method. The FBS constraint developed in class tells us that in order for the time function to be reconstructed using FBS synthesis the window function \( w[n] \) must equal zero for \( n = \pm N, \pm 2N, \pm 3N \ldots \) etc. This will happen trivially if \( N = 320 \). Hence we will implement the FBS method using 320 parallel channels. As you will recall, the Hamming window (like the other windows commonly used for STFA) is lowpass in nature, with a nominal cutoff frequency of \( 4\pi/N \). For \( N = 320 \) this becomes

\[
\omega_c = \frac{4\pi}{320} = \frac{\pi}{80}
\]

which suggests that the output of each channel can be downsampled by a factor of 80. We will consider the result of the analysis/synthesis process without and with this downsampling.

1. Implement the filterbank analysis of the input \( \text{welcome16k.wav} \) using the “lowpass” implementation of the STFT with 320 channels. This should result in a huge number of STFT values of \( X_2[n, k] \), 320 times the original number of samples in \( \text{welcome16k.wav} \).

2. Reconstitute the time function using filterbank summation by multiplying each output signal by \( e^{j2\pi nk/320} \) and summing the product across channels. The resulting time function \( y_2[n] \) should be proportional to the original input.

3. Now consider for each value of \( k \) the sequence of coefficients \( X_2[n, k] \). As noted above, these sequences are highly over sampled, and should be nominally downsampled by a factor of 80. Using the MATLAB command \texttt{decimate} or \texttt{resample}, obtain downsampled versions of \( X_2[n, k] \) for each of the 320 values of \( k \). Plot the downsampled sequences of \( X_2[n, k] \) as a spectrum as you had done previously.

4. Now we will reconstruct a time function from the downsampled sequences of \( X_2[n, k] \). First upsample the downsampled \( X_2[n, k] \) using the MATLAB commands \texttt{interp} or \texttt{resample}. Then for each channel \( k \), multiply each sequence of \( X_2[n, k] \) by \( e^{j2\pi nk/320} \) and sum the products across channels, as you did in part (b2) of this problem. The resulting function should in principle be proportional to the input, but it will not be because the Hamming window is far from a perfect lowpass filter. Listen to the input and output and comment on the extent to which the output from the downsampled FBS representation adequately describes the individual input. If downsampling and upampling by a factor 80 introduces too much distortion, what downsampling ratio provides an adequate degree of fidelity in your opinion?

(c) Compare the total number of values of \( X[n, k] \) per second that was needed to represent the time function using OLA analysis/synthesis in part (a) with the number that was needed to represent the time function using FBS analysis/synthesis in part (b) with downsampling by the factor of 80. Which representation is more efficient in terms of storage space?

What you should turn in on paper:

- A block diagram and description of your systems from parts (a) and (b) as you implement them.
Problem C5.2:

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. You are asked 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/KimStern08.pdf

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.

(a) Using Hamming windows of 20-ms duration and the minimum frame sampling rate in time that satisfies the OLA constraint, 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).

**Comment:** Remember that the coefficients $X[k]$ of any DFT of size $N$ from $0 \leq k \leq N/2$ represent positive frequencies and the coefficients $X[k]$ for $(N/2) + 1 \leq k \leq N − 1$ represent negative frequencies. If the time function is real, the DFT coefficients will be Hermitian symmetric and $X[N − k] = X^*[N − k]$ because $X[k]$ is periodic in $k$ with period $N$.

(c) Produce a signal that has Speaker A only (or at least an approximation to that result), by resynthesizing 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`.

(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 spectrograms of the two separated and reconstituted speech waveforms. Compare these with the corresponding MATLAB spectrograms of the original utterances using the same analysis parameters.
- Hard copy of your MATLAB 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 “5_2a” and “5_2b”, as in “rms5_2a.wav”
- The actual MATLAB code that you used to achieve your results.
Problem C5.3:

In this problem you will develop and analyze a phase vocoder for speech processing. Recall that in phase vocoding you will perform the following steps:

- Perform short-time Fourier analysis as usual using the lowpass or bandpass filter implementation.
- Compute the magnitude and phase-derivative signals as in Problem 5.2 above.
- Downsample and transmit the magnitude and phase-derivative signals.
- Decode and upsample the magnitude and phase-derivative signals, and use them to resynthesize the original waveform according to the formula

\[
y[n] = \sum_{k=0}^{N/2-1} M[n, k] \cos(\omega_k n + \hat{\phi}[n, k] + \phi_k)
\]

where \(\omega_k = 2\pi k/N\), \(\hat{\phi}[n, k] = T \sum_{l=0}^{n} \hat{\phi}[l, k]\), and \(\phi_k = \phi[0, k]\).

In the equation above \(T\) refers to the time between the samples (or the sampling period, the reciprocal of the sampling frequency). Note that only \(N/2\) components are used because we are not paying attention to negative frequencies in the synthesis.

(a) First we will perform a basic analysis and resynthesis of the familiar “Welcome to DSP-I” utterance.

1. Using the 8-kHz sampled version of “Welcome to DSP-I” available on the Web as `welcome8k.wav`, develop an analysis system for phase vocoding. You may wish to start by using a number of channels and other design parameters based on the implementations in Flanagan and Golden (1966), but understand that this was an implementation for a continuous-time system. For example, you will want to use FIR Hamming windows rather than the IIR Bessel filters in the Flanagan and Golden paper. Plot the spectrum of typical magnitude signals \(M[n, k]\) and phase-derivative signals \(\hat{\phi}[n, k]\), and estimate the upper cutoff frequencies for these two signals in Hz. How do these bandwidths depend on channel number \(k\)?

2. Resynthesize the speech signal directly using the method of sinewave synthesis described above and evaluate the quality of the result. The question of how to estimate the phase parameters \(\{\phi_k\}\) blindly has been the object of a great deal of research, as the correct choice is important in reducing artifacts in resynthesis (which you will observe). We will avoid this issue for now simply by using the empirically-observed initial phase values \(\phi_k = \phi[0, k]\), or you could try the initial actual phase values of the DFT coefficients as a plausible alternative.

3. Now downsample the magnitude and phase-derivative signals, \(M[n, k]\) and \(\hat{\phi}[n, k]\), based on your estimates of what we need from part (a). (You may used different downsampling ratios from channel to channel.) Then upsample by the same amount and resynthesize. What is the minimum number of
total real numbers per second \((M[n, k] \text{ and } \phi[n, k])\) in the phase vocoding representation that you need to achieve intelligible speech of reasonably high quality? How does this compare with the number of numbers you used in the channel vocoding method that you developed in Problem C5.1? (Keep in mind that the initial sampling rate is only 8 kHz in this problem.)

(b) Now let us explore the use of the phase vocoder to slow down and speed up the speech. Recall that speech can be slowed down by a factor of \(L/M\) by multiplying all instantaneous frequency values by \(L/M\) and subsequently upsampling by \(L\) and downsampling by \(M\). Implement these transformations and listen to the resulting audio. What are the maximum ratios by which you can increase and decrease the speech rate without rendering it unintelligible?

What you should turn in on paper:

- A description of what you did to get your results, discussing the various design issues, along with your answers to the questions that are asked in parts 1 and 3 of (a) and part (b).
- Hard copy of your MATLAB scripts you used

What you will submit electronically:

- An example of the basic resynthesized speech from part 2 of (a)
- An example of the speech with maximum downsampling in the parametric representation from part 3 of (a)
- Examples of speech with the rate increased and decreased as much as possible from part (b).
- The actual MATLAB code that you used to achieve your results.

Label the speech files with your name and the suffixes such as “5_3a2” and “5_3a3”, etc., as in “rms5_3a2.wav”
Problem C5.4:

The file PSO_B1short.wav is a 16-kHz stereo recording of a short excerpt of chamber music. To save time and computation, you may perform the following computations in mono (by adding the signals in the two stereo channels together), if you wish.

(a) Using techniques similar to those in Problem C5.3, develop an analysis-synthesis without and with downsampling based on phase vocoding for this excerpt. Note that you will need to change some system parameters because the sampling frequency has been doubled. What is the smallest number of numbers per second that you believe that you need to obtain “good” fidelity of reproduction?

(b) Use the phase vocoder to transpose the music up and down by a major third (a ratio of 5:4 in frequency). This can be accomplished by multiplying or dividing all instantaneous frequencies by 5:4 while leaving the sampling frequency unchanged. In your opinion, is the sound quality adequate? How far can you go with transposing up and down before the sound quality degrades? (Consider increasing or decreasing the instantaneous frequencies by ratios of 6:5 (a minor third), 5:4 (a major third), 4:3 (a perfect fourth), and 3:2 (a perfect fifth).

What you should turn in on paper:

• A description of what you did to get your results, discussing the various design issues, along with your answers to the questions that are asked in parts (a) and (b).

• Hard copy of your MATLAB scripts you used

What you will submit electronically:

• Examples of the resynthesized music without and with downsampling in part (a).

• Examples of the musical excerpt transposed up and down by a major third from part(b).

• The actual MATLAB code that you used to achieve your results.

Label the speech files with your name and the suffixes such as “5_4a1” and “5_4a2”, etc., as in “rms5_4a1.wav”

1. It is the opening bars of Bach’s first Brandenburg concerto, a performance on the Pittsburgh Symphony Community Concert Series in CMU’s Kresge auditorium in the spring of 2006. I’m playing harpsichord.

Copyright 2017, Richard M. Stern