Problem Set #6 – Solutions
Due: Friday May 25, 2018 at 5 PM.

1. Overlap-and-add with latency requirements

Consider
\[ h[n] = \begin{cases} 
\frac{1}{6} & \text{if } 0 \leq n \leq 5, \\
0 & \text{otherwise}, 
\end{cases} \]
and another sequence \( x[n] \) satisfying
\[ x[n] = \begin{cases} 
\text{nonzero} & \text{if } 0 \leq n \leq 9999, \\
0 & \text{otherwise}. 
\end{cases} \]

Consider the following two ways of evaluating \( y[n] = x[n] * h[n] \). Assume that all linear convolution operations are performed with zero padding and circular convolution via DFT as described in Pages 88-89 of the reader.

- (direct) Compute the linear convolution \( y[n] = h[n] * x[n] \) directly.
- (block20) Use the overlap add method as described in page 91 of the course reader with \( L = 20 \).

(a) Assuming that the DFT operations take \( 0.5N \log_2 N \) multiplications for a blocklength of \( N \), evaluate the complexity (i.e., number of multiplications performed) of both methods.

(b) The latency is defined as the time between when an input sample first affects the output and the time the output is independent of that sample. That is, in the overlap-add method with \( L > P \) the latency equals \( L \) sample times, or equivalently it equals \( LT_s \) seconds where \( T_s \) is the sample time. Find the complexity as a function of latency (in seconds) for \( 1 \leq L \leq 10000 \) and \( T_s = 10\mu \text{sec} \). You may only consider values of \( L \) which divide 10000. Plot the complexity (number of multiplications) with respect to \( L \).

Solution:

(a) We have:

- (direct) We have \( N = 6 + 10000 - 1 \). Thus number of multiplications is
\[
2\text{ (mults. in DFT)} + \text{ (mults. in IDFT)} + \text{ (pointwise mult. of 2 sequence)}
\]
\[
= 1.5N \log_2 N + N.
\]

The factor of two comes from the fact that there are two sequences.
(block20) For \( L = 20 \), we have the number of multiplications equal to 
\[
\frac{10000}{L} \times (1.5(M) \log_2(M) + M),
\]
where \( M = 20 + 6 - 1 = 25 \).

(b) The number of multiplications is given by
\[
\frac{10000}{L} \times (1.5(L + 5) \log_2(L + 5) + (L + 5)).
\]

The plot is presented below.

2. Computing the Laplace Transform

Find the Laplace transform \( X(s) \), region of convergence (ROC), and the zero-pole plot of the following signals. Use Laplace transform tables and properties (OWN pp. 691-692, and posted to class website to minimize your effort.

(a) 
\[ x(t) = e^{-4t} u(t) + e^{-5t} \sin(5t) u(t) \]

Solution:
Using the transform table:
\[ e^{-4t} u(t) \leftrightarrow \frac{1}{s+4}, \quad \Re(s) > -4 \]
\[ e^{-5t} \sin(5t) u(t) \leftrightarrow \frac{5}{(s+5)^2 + 25} = \frac{5}{s^2 + 10s + 50}, \quad \Re(s) > -5. \]

The sum of the two terms is
\[
\frac{1}{s+4} + \frac{5}{s^2 + 10s + 50}, \quad \text{(1)}
\]

In order to obtain the pole-zero map we decompose the numerator and denominator of \((1)\) as follows:
\[
\frac{1}{s+4} + \frac{5}{s^2 + 10s + 50} = \frac{s^2 + 15s + 70}{(s+4)(s^2 + 10s + 50)} = \frac{(s - \beta_1)(s - \beta_2)}{(s - \gamma_1)(s - \gamma_2)(s - \gamma_3)}.
\]
We find that \( \beta_{1,2} = -\frac{15 \pm j \sqrt{55}}{2} \), \( \gamma_1 = -4 \), and \( \gamma_{1,2} = -5 \pm j 5 \). Since no poles disappeared when the two signals were combined, the ROC is the intersection of the two ROC above, which equals \( \Re(s) > -4 \).

(b) 
\[ x(t) = |t|e^{-2|t|} \]

*Hint:* Express \( x(t) \) as the sum of left- and right-sided signals.

*Solution:*
\[ x(t) = -te^{2t}u(-t) + te^{-2t}u(t). \]

From the transform table:
\[ -te^{2t}u(-t) \leftrightarrow \frac{1}{(s-2)^2}, \quad \Re(s) < 2, \]
\[ te^{-2t}u(t) \leftrightarrow \frac{1}{(s+2)^2}, \quad \Re(s) > -2. \]

Combining the two, finding a common denominator, and factoring to identify the poles and zeros:
\[
X(s) = \frac{1}{(s-2)^2} + \frac{1}{(s+2)^2} \\
= \frac{2(s^2 + 4)}{(s-2)^2(s+2)^2} \\
= \frac{2(s+2j)(s-2j)}{(s-2)^2(s+2)^2}.
\]

No poles disappear in combining the two terms so the ROC is the intersection of the two ROCs above:
\[
ROC = \{ s \in \mathbb{C}, \ -2 < \Re s < 2 \}.
\]

3. **Inverse Laplace Transform**

Determine the signal \( x(t) \) for each of the following Laplace transforms and associated region of convergence. Use the OWN Laplace transform tables on the class website and properties whenever possible to minimize your effort. Each time signal is real, and should be expressed in real form.

(a) 
\[
X(s) = \frac{1}{s^2 + 9}, \quad \Re(s) > 0
\]

*Solution:*
\[
X(s) = \frac{3}{3s^2 + 3^2}, \quad \Re(s) > 0
\]

From the transform table we obtain
\[ x(t) = \frac{1}{3} \sin(3t)u(t). \]
Note that we can also find \( x(t) \) by partial-fraction expansion:

\[
X(s) = \frac{1}{(s - 3j)(s + 3j)} = \frac{1}{3} \cdot \frac{1}{2j} \left( \frac{1}{s + 3j} - \frac{1}{s - 3j} \right), \quad \Re(s) > 0.
\]

This leads to

\[
x(t) = \frac{1}{3} u(t) \frac{e^{3t} + e^{-3t}}{2j} = \frac{1}{3} u(t) \sin(3t).
\]

(b) \[ X(s) = \frac{e^{-s}}{s + 2}, \quad \Re(s) > -2 \]

**Solution:**

The ROC indicates that \( x(t) \) is a right-sided signal. Since \( e^{-2t} u(t) \leftrightarrow \frac{1}{s+2} \) and from the time-shift property we obtain

\[
x(t) = e^{-2(t-1)} u(t-1).
\]

(c) \[ X(s) = \frac{s + 1}{(s + 1)^2 + 9}, \quad \Re(s) < -1 \]

**Solution:**

Since \( X(s) \) is a rational function it is required to find its partial-fraction expansion representation

\[
X(s) = \frac{A_1}{s - \gamma_1} + \frac{A_2}{s - \gamma_2},
\]

where \( \gamma_1, \gamma_2 \in \mathbb{C} \). By expanding the denominator and matching coefficients of \( s \) we find

\[
X(s) = \frac{1}{2} \left( \frac{1}{s - (-1 + 3j)} + \frac{1}{s - (-1 - 3j)} \right).
\]

There are two complex-conjugate poles and the ROC indicates that \( x(t) \) is a left-sided signal. We conclude that

\[
x(t) = -\frac{1}{2} \left( e^{(-1+3j)t} + e^{(-1-3j)t} \right) u(-t) \\
= -\frac{1}{2} \left( e^{-t} e^{3jt} + e^{-t} e^{-3jt} \right) u(-t) \\
= -e^{-t} \cos(3t) u(-t)
\]

4. **Time domain analysis using Laplace** (15 pts)

A causal continuous-time system with input \( x(t) \) and output \( y(t) \) is described by the following differential equation:

\[
2y''(t) + 3y'(t) + y(t) = 2x'(t)
\]

(a) Find the system impulse response \( h(t) \). (5 pts)

**Solution:**

Taking the Laplace transform of both sides of the differential equation leads to
\[(2s^2 + 3s + 1)Y(s) = 2sX(s).\]

Solving for the transfer function:

\[H(s) = \frac{Y(s)}{X(s)} = \frac{2s}{(2s^2 + 3s + 1)} = \frac{s}{(s + \frac{1}{2})(s + 1)}\]

Performing the partial-fraction expansion:

\[H(s) = \frac{A_1}{s + \frac{1}{2}} + \frac{A_2}{s + 1}\]

where \(A_1 = -1, A_2 = 2\). Inverting using the transform table and the fact that the system is causal leads to \(h(t) = (-e^{-\frac{t}{2}} + 2e^{-t})u(t)\).

(b) Find the output signal \(y(t)\) assuming

\[x(t) = \cos(t) + u(t + 1), -\infty < t < \infty.\]

Hint: denote \(x_1(t) = \cos(t)\) and \(x_2(t) = u(t + 1)\). Analyze separately the responses \(y_1(t)\) and \(y_2(t)\) due to input \(x_1(t)\) and \(x_2(t)\), respectively, and then use linearity to say that \(y(t) = y_1(t) + y_2(t)\).

Recall that the output of a linear system to a periodic signal is easily obtained from the frequency response of the system. (10 pts)

**Solution:**

Using the notation introduced in the hint: First term: Since the system is causal, the ROC is \(\text{Re}(s) > -\frac{1}{2}\) which includes the \(j\omega\) axis. This implies that the frequency response exists and is given by:

\[H(j\omega) = \frac{2j\omega}{(1 - 2\omega^2) + 3j\omega}\]

For a system with conjugate symmetric frequency response \(H(j\omega) = H^*(j\omega)\), an input \(x_1(t) = \cos(\omega_0 t)\) induces an output \(y_1(t) = |H(j\omega_0)|\cos(\omega_0 t + \angle H(j\omega_0))\). In our case we have:

\[|H(j\omega)| = \frac{2|\omega|}{\sqrt{(1 - 2\omega^2)^2 + (3\omega)^2}} = \frac{2|\omega|}{\sqrt{(1 - 2\omega^2)^2 + 9\omega^2}}\]

\[\angle H(j\omega) = \angle(2j\omega) - \angle((1 - 2\omega^2) + 3j\omega) = \frac{\pi}{2} \text{sgn}(\omega) - \tan^{-1}(\frac{3\omega}{1 - 2\omega^2})\]

Hence,

\[y_1(t) = |H(j1)|\cos(t + \angle H(j1)) = \frac{2}{\sqrt{10}} \cos(t + \frac{\pi}{2} - \tan^{-1}(-3))\]

\[= \frac{2}{\sqrt{10}} \cos(t + \frac{\pi}{2} + \tan^{-1}(3)) = -\frac{2}{\sqrt{10}} \sin(t + \tan^{-1}(3))\]

Note to Grader: there’s a different approach (not shown here) which will give

\[y_1(t) = \frac{2}{\sqrt{10}} \cos\left(t + \frac{\pi}{4} - \arctan(2)\right)\].

This are mathematically equivalent, even though it looks super different, and should be given full credit.

Second Term: The laplace transform of \(Y_2(s)\) is given by

\[Y_2(s) = H(s)X_2(s) = H(s)\frac{e^s}{s} = \frac{e^s}{(s + \frac{1}{2})(s + 1)}\]
From the transform table, the term $e^s$ in the Laplace domain implies a delay of $t_0 = 1$ in the time domain, so we only need to invert the rational part of $Y_1(s)$ which is $H(s)/s$. Partial-fraction expansion of the latter leads to

$$\frac{H(s)}{s} = \frac{2}{s + \frac{1}{2}} - \frac{2}{s + 1}$$

Inverting using the transform table and the shifting property of the transform we get:

$$y(t) = (2e^{-\frac{1}{2}(t+1)} - 2e^{-(t+1)})u(t+1).$$

Then combining both terms:

$$y(t) = -\frac{2}{\sqrt{10}}\sin(t + \tan^{-1}(3)) + (2e^{-\frac{1}{2}(t+1)} - 2e^{-(t+1)})u(t+1).$$

5. **Laplace Transform ROC**

For each of the following statements about $x(t)$, and for each of the pole-zero plots of $X(s)$ in the figure below, determine the corresponding constraint on the ROC (so there should be three answers for each part, one answer for each of the 3 figures):

(a) $x(t)e^{-3t}$ is absolutely integrable

(b) $x(t) = 0$, $t > 1$

(c) $x(t) = 0$, $t < -1$

(d) **Extra credit**: $x(t) * (e^{-t}u(t))$ is absolutely integrable

---

![Pole-Zero Plot Diagram](image-url)
Solution:
There are only three possibilities for the ROC: either to the left of $-2$, in between $-2$ and 2 and to the right of 2.

(a) Multiplication by $e^{-3t}$ shifts the transform to the left by 3. $x(t)e^{-3t}$ being absolutely integrable implies that $X(s + 3)$ must contain the $j\omega$ axis. This is only possible if the original ROC is to the right of 2 for top two plots. For the bottom plot, the argument gives that ROC must be to the right of $s = -2$.

(b) $x(t)$ cannot be a right hand signal. This also means that the ROC must be to the left of the leftmost pole. This uniquely restricts the ROC for all the figures shown.

(c) $x(t)$ cannot be a left sided signal. This means that the ROC must be to the right of the rightmost pole.

(d) Convolution with $e^{-t}u(t)$ is equivalent to multiplication by $1/(s + 1)$, i.e., introduction of a pole at $s = -1$ and restricting the ROC to a subset of $s > -1$. Absolute integrability requires that the ROC of the convolved signal must contain the $j\omega$ axis. For the top left figure this restricts the ROC of $X(s)$ to be between $s = -2$ and $s = 2$. For the top right figure the ROC is restricted to the left of $s = 2$. For the bottom figure, the ROC of $X(s)$ must be to the right of $s = -2$. 
MATLAB Assignment

General Instructions

Answer all questions asked. Your submission should include all m-file listings and plots requested. All plots should have a title and x- and y-axes properly labeled.

This week we will look at estimating signals whose spectrum change over time.

The basic problem is that we have a long segment of a signal \( x[n] \), where \( n = 1, N \). Since the signal is long, the spectrum of different segments can vary a lot. As a result, we break the signal up into short segments and compute the spectrum of each of these separately.

A sampled bird song is shown in Fig. 1. Obviously you can see that the amplitude is changing as a function of time. When you examine the signal closely, you can also see that the frequency is changing. Since the frequency is changing, we want to break the signal into segments over which the frequency is relatively constant. Then we will analyze each segment using the DFT. The result of analyzing the first 256 samples, and another block of 256 samples taken 1024 samples later is shown in Fig. 2. In Fig. 2, the lowest frequency on the x-axis corresponds to zero frequency, and the highest frequency corresponds to a frequency equal to the sampling rate, since we have not centered the spectrum around 0 as it is typically displayed (to do so we would shift the spectrum down by half the sampling rate; there is a command in Matlab called fftshift to do this but it was not applied here). Observe that the frequency goes up by about 300 Hz from block 1 to block 2.

The presentation format of Fig. 2 is not very suitable for displaying time-varying spectra. Given a long signal, a large number of plots would need to be displayed, and they dont convey the information very effectively. A more effective way to display time-varying spectra is in an image format, with frequency on the vertical axis, time on the horizontal axis, and the brightness of a particular pixel representing the magnitude of the spectrum at the particular time and frequency. Since we are assuming that the signal is real, the magnitude of the spectrum is an even function of frequency, and it is only necessary to display positive frequencies. This presentation is known as a spectrogram. An example of this type of plot for the signal that Fig. 1 was taken from is shown in Fig. 3.
A MATLAB routine that takes a two-dimensional array y and makes an image of the log-magnitude is given below:

```matlab
function sgplot(tl,fl,y)

% Plot an image of the spectrogram y, with axes labeled with time tl and frequency
% tl -- time axis label, nt samples
% fl -- frequency axis label, nf samples
% y -- spectrogram, nf by nt array

% find the maximum
mx = max(max(abs(y)));
% compute 20*log magnitude normalized by the maximum
```

Figure 2: Spectra from two 256-sample blocks of the signal from Fig. 1

Figure 3: Spectrogram of a segment of a sampled bird song.
yl = 20*log10(abs(y/mx)+eps); % show 60 dB of dynamic range
dbf = 60;
image(tl,fl,64*(yl+dbf)/dbf);
axis xy; colormap('gray'); brighten(0.2); colorbar;
xlabel('Time (s)');
ylabel('Frequency (Hz)');
end

The only parameter of note is dbf, which determines the dynamic range in dB that will be presented. It is hard-coded to 60 dB, corresponding to an amplitude factor of 1000. This allows you to see small spectral components without seeing all of the background noise, and is reasonably suitable for our purposes here. For other applications with higher or lower noise levels, you may want to change this.

Several different sound files are available on the class web site to for you to work with in the MATLAB file hw7_sounds.mat (http://web.stanford.edu/class/ee102b/contents/restricted/hw7_sounds.mat). Specifically, there are 4 sound signals stored in the MATLAB variables s1, s2, s3, and s4. These are: s1 A bird call, s2 A creaking door, s3 An orca call, s4 A sound effect.

Play each of these with the MATLAB sound() function, and try to visualize what you would expect the spectrum to look like as a function of time. In each case, the sampling rate is 8192 Hz.

1. Computing a simple spectrogram. (10 pts)

For our first attempt at making a spectrogram, we will simply take the original signal of length N, and split it into blocks of length M. We then compute the DFT of each block. This corresponds to using a rectangular window to extract each block from the signal, as is illustrated in Fig. 4.

Write a MATLAB m-file called myspectogram that computes the spectrogram for a signal. If the original signal is a vector x, It should:

Break the signal up into M-sample blocks, stored in the columns of a 2D matrix \( \mathbf{x}_m \). This will require padding the signal with zeros, so that the length is a multiple of the block size. Apply the \( \mathbb{F} \) to the matrix. Compute vectors for the time \( t_l \) and frequency \( f_l \) labels for each row and column. Call the \texttt{sgplot}(t_l,f_l,xmf(1:m/2,:)), where we are only plotting the positive frequencies.

Your routine should be invoked with:

\[
\text{>> myspectrogram(x,m)}
\]

where \( x \) is the signal, and \( m \) is the block size. Assume a sampling rate of 8192 Hz. One MATLAB trick that can help you here is the \texttt{reshape()} command. This command can split up a vector and store the values in a matrix. In our example, if you want to split up your original signal vector \( x \) into \( nt \) time windows of length \( m \), you would use the following command:

\[
\text{>> % use reshape to make it an m by nt matrix} \\
\text{\% use reshape to make it an m by nt matrix} \\
\text{xm = reshape(xp,m,nt);}
\]

where \( xp \) is your vector \( x \) that is zero-padded to have length \( m \times nt \).

To compute the time and frequency vectors, recall that the DFT frequencies go from 0 to the sampling frequency 8192 Hz in steps of \( \frac{8192}{M} \) Hz and the time of a particular block is the period of one
sample, 1/8192 seconds, multiplied by the number of samples in the block. You can assume we are only going to plot the positive frequencies.

Try your spectrogram routine with the bird call sound file, s1, with a block size of 256 samples and include this plot in your submitted homework.

![Fig 4: Sequential Box-Car](image1)

2. A better spectrogram (10 pts)

The problem with the spectrogram from Task 1 is that we have used a rectangular window to extract blocks of the signal. In the spectral domain this corresponds to convolving a sinc function with the spectra. The sinc sidelobes are high, and fall off slowly. We need to do something a little gentler with the data.

To improve our spectrogram, we will extract each block of data with a Hann window. We can do this by multiplying our $x_m$ matrix with a matrix that has a Hann window along its columns. Try your m-file on the bird song again. Include a copy of this spectrogram in your submitted homework.

![Fig 5: Sequential Hann](image2)

3. Extra Credit: An even better spectrogram (10 pts)

As a final enhancement, we will overlap the Hann windows, as shown in Fig. 6. Each block of the signal, and column of the $x_m$ matrix overlaps the adjacent blocks by 50 percent. There are a number of ways to do this in MATLAB. One is to replicate each half block in $x_p$ before the reshape. Another direct approach would be to program it in a for loop. Note that you also have to modify the time vector, since you have twice as many time samples.
Try this on the bird song. It should look smoother than the previous case. Include a copy of this spectrogram in your report.

![Overlapping Hann](image)

4. **Extra Credit: Time and Frequency Resolution** *(5 pts)*

Thus far, we have only set the block size to 256 samples, which represents about 31 ms of data. The frequency resolution is then $8192/256 = 32$ Hz. Ideally, we would like to good resolution in both time and frequency. However, one comes at the expense of the other. For example if we wanted to improve our frequency resolution to 16 Hz, we would need to increase the time between samples to 62 ms. The optimal tradeoff between the two factors depends on the nature of the spectrum you are analyzing. A good example is the orca call, signal s3. This has segments that are very rapidly changing, and segments where the changes are slow. Try using `myspectrogram(s3,m)` for different block sizes, m = 128, 256, 512, and 1024 samples. Notice the tradeoff between time and frequency resolution. Include plots.
Matlab Pset 6

Table of Contents

Part 1 .................................................................................................................. 1
Part 2 .................................................................................................................. 1
Part 3 .................................................................................................................. 2
Part 4 .................................................................................................................. 3

Part 1

close all
load hw7_sounds.mat
myspectrogram(s1, 256)
title('Spectrogram with Rectangular Window')

Part 2

x = s1; m = 256;
lx = length(x); nt = ceil(lx/m);
xp = zeros(1, nt*m); xp(1:lx) = x;
xm = reshape(xp, m, nt);
Part 3

if m is odd, round it the nearest integer greater than m/2. When m is odd, we need to be careful. In the case of odd m, let's replicate (m-1)/2 (=floor(m/2)) samples.

```matlab
m_half_c = ceil(m/2);
m_half_f = floor(m/2); nt_rep = ceil(lx/m_half_c);
xp_rep = zeros(1, nt_rep*m_half_c);
xp_rep(1:lx) = x;

xm3_upper_row = reshape(xp_rep, m_half_c, nt_rep);
xm3_lower_row = [xm3_upper_row(1:m_half_f, 2:nt_rep) zeros(m_half_f, 1)];
xm3 = [xm3_upper_row; xm3_lower_row];
xm3 = xm3.*(hann(m)*ones(1, nt_rep)); % hanning windowing

xmf3 = fft(xm3);
tl3 = [1:nt_rep]*m_half_c/8192+m_half_f/8192;
figure
sgplot(tl3, fl, xmf3(1:m/2,:))
title('Spectrogram with Overlapping Hanning Window')
```
% Implementation with for loop (For Task 3)
xp4=[];
for i = 1:nt_rep,
    if ((i-1)*m_half_c+m>lx)
        tmp = zeros((i-1)*m_half_c+m-lx,1);
        xp4((i-1)*m+1:m*i) = [x((i-1)*m_half_c+1:end); tmp];
    else
        xp4((i-1)*m+1:m*i) = x((i-1)*m_half_c:(i-1)*m_half_c+m);
    end
end
xm4 = reshape(xp4,m,nt_rep);
xm4 = xm4.*(hann(m)*ones(1,nt_rep));
xmf4 = fft(xm4);
% figure
% sgplot(tl3,fl,xmf4(1:m/2,:))

Part 4

x=s3;
m = [128 256 512 1024];
for i = 1: length(m)
    myspectrogram(x,m(i))
    title(m(i))
end
function myspectrogram(x,m)
% % Take the original signal x and split it into blocks of length m
% This corresponds to using a rectangular window
% Pad x up to a multiple of m
lx = length(x);
nt = ceil(lx/m);
xp = zeros(1,nt*m);
xp(1:lx) = x;
% use reshape to make it an m by nt matrix
xm = reshape(xp,m,nt);
% frequency index
fl = [0:(m/2-1)]*8192/m;
tl = [1:nt]*m/8192;
xmf = fft(xm);
figure
sgplot(tl,fl,xmf(1:m/2,:))
end

Published with MATLAB® R2018a