Music 320
Autumn 2010–2011
Homework #3
Complex Sinusoids, Geometric Signal Theory, DFT
125 points
Due in one week (10/14/2010) by 11:59pm

Theory Problems

1. (10 pts) [Orthogonality of sinusoids] Show that if two length N sampled sinusoids $s_k(n)$ and $s_l(n)$ are orthogonal, *i.e.*, $s_k \perp s_l$, then $\mathcal{A}s_k \perp \mathcal{B}s_l$ for all complex constants $\mathcal{A}$ and $\mathcal{B}$ where $\mathcal{A} = Ae^{j\alpha}$ and $\mathcal{B} = Be^{j\beta}$. That is, the orthogonality of two sinusoids is independent of their phases and (nonzero) amplitudes.

   Solution:

   $$s_k \perp s_l \Longrightarrow < s_k, s_l >= 0$$

   Then,

   $$\begin{aligned} < \mathcal{A}s_k, \mathcal{B}s_l > &= \mathcal{A} < s_k, \mathcal{B}s_l > \\ &= \mathcal{A}\overline{\mathcal{B}} < s_k, s_l > \\ &= 0 \end{aligned}$$

   Therefore,

   $$\mathcal{A}s_k \perp \mathcal{B}s_l$$

2. (10pt) [Inner Products] For the complex vectors $x = (1, j, 1-j)$ and $y = (1+j, -1+j, -j)$ in the 3-D complex space $\mathbf{C}^3$, find the inner products

   (a) $\langle x, x\rangle$
   (b) $\langle y, y\rangle$
   (c) $\langle x, y\rangle$
   (d) $\langle y, x\rangle$

   Solution: Since $x = (1, j, 1-j)$ and $y = (1+j, -1+j, -j)$,
   $\overline{x} = (1, -j, 1+j)$ and $\overline{y} = (1-j, -1-j, j)$. Therefore,

   (a) $\langle x, x\rangle = \sum_{n=0}^{N-1} x(n)\overline{x(n)} = 1\cdot 1 + j\cdot(-j) + (1-j)\cdot(1+j) = 4$
   (b) $\langle y, y\rangle = \sum_{n=0}^{N-1} y(n)\overline{y(n)} = (1+j)\cdot(1-j) + (-1+j)\cdot(-1-j) + (-j)\cdot(j) = 5$

(c) $\langle x, y\rangle = \sum_{n=0}^{N-1} x(n)\overline{y(n)} = 1\cdot(1-j) + j\cdot(-1-j) + (1-j)\cdot j = 3-j$

(d) $\langle y, x\rangle = \sum_{n=0}^{N-1} y(n)\overline{x(n)} = (1+j)\cdot 1 + (-1+j)\cdot(-j) + (-j)\cdot(1+)j = 3+j$

3. (20 pts) [Projections] For the complex vectors $x = (1, j, 1-j)$ and $y = (1+j, -1+j, -j)$, find the

   (a) the projection of $y \in C^N$ onto $x \in C^N$, and the coefficient of projection
   Solution:

$$\mathbf{P}_x(y) = \frac{\langle y, x\rangle}{\|x\|^2}x = \frac{3+j}{4}x = \frac{3+j}{4}(1, j, 1-j) = \frac{1}{4}(3+j, -1+3j, 4-2j)$$

$$\text{projection coefficient : } \frac{3+j}{4}$$

   (b) the projection of $x \in C^N$ onto $y \in C^N$, and the coefficient of projection
   Solution:

$$\mathbf{P}_y(x) = \frac{\langle x, y\rangle}{\|y\|^2}y = \frac{3-j}{5}y = \frac{3-j}{5}(1+j, -1+j, -j) = \frac{1}{5}(4+2j, -2+4j, -1-3j)$$

$$\text{projection coefficient : } \frac{3-j}{5}$$

4. (15 pts) [Inner Products] For the set of N-sample-long complex sinusoids defined as:

$$S_k[n] = e^{j\frac{2\pi kn}{N}}$$

   Compute the following inner products for $N = 6$:

   (a) $\langle S_0, S_1\rangle$
   Solution:

$$\begin{aligned}\langle S_0, S_1\rangle &= \sum_{n=0}^{5} e^0 e^{-j\frac{2\pi n}{6}} = \sum_{n=0}^{5} e^{-j\frac{2\pi n}{6}} \\ &= 1 + e^{-j\frac{2\pi}{6}} + e^{-j\frac{4\pi}{6}} + e^{-j\frac{6\pi}{6}} + e^{-j\frac{8\pi}{6}} + e^{-j\frac{10\pi}{6}}\end{aligned}$$

   Note that this is the sum of the 6 roots of unity, which we know sum to 0. Another way of thinking of this is to realize that the first three terms are cancelled out by the last three terms (they form three pairs of complex sinusoids with a phase difference of π). Therefore

$$\langle S_0, S_1\rangle = 0$$

(b) $\langle S_1, S_1 \rangle$

Solution:

$$\begin{aligned}\langle S_1, S_1 \rangle &= \sum_{n=0}^{5} e^{j\frac{2\pi n}{6}} e^{-j\frac{2\pi n}{6}} = \sum_{n=0}^{5} e^{0} = \sum_{n=0}^{5} 1 \\ &= 6\end{aligned}$$

(c) $\langle S_1, S_2 \rangle$

Solution:

$$\langle S_1, S_2 \rangle = \sum_{n=0}^{5} e^{j\frac{2\pi n}{6}} e^{-j\frac{4\pi n}{6}} = \sum_{n=0}^{5} e^{-j\frac{2\pi n}{6}}$$

which is the same *sum* we solved for $\langle S_0, S_1 \rangle$. Therefore:

$$\langle S_1, S_2 \rangle = 0$$

5. (10 pts) [DFT of an impulse]

(a) Find the length 16 DFT of the unit-amplitude, zero-centered impulse:

$$x(n) = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

Solution:

$$\begin{aligned}X(k) &= \sum_{n=0}^{15} x(n) e^{-j2\pi nk/16} \\ &= 1\end{aligned}$$

So the DFT of an impulse is a constant:

$$X(k) = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$$

(b) Find the length 16 DFT of the unit-amplitude impulse centered at $n = 3$:

$$x(n) = [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

Solution:

$$\begin{aligned}X(k) &= \sum_{n=0}^{15} x(n) e^{-j2\pi nk/N} \\ &= e^{-j2\pi 3k/16} = e^{-j3\pi k/8}\end{aligned}$$

So the magnitude of the DFT is still a constant, but now there's an extra linear phase term:

$$
\begin{aligned}
X(k) = [1, &e^{-j3\pi/8}, e^{-j6\pi/8}, e^{-j9\pi/8}, e^{-j12\pi/8}, e^{-j15\pi/8}, e^{-j18\pi/8}, e^{-j21\pi/8}, e^{-j24\pi/8},\\
&e^{-j27\pi/8}, e^{-j30\pi/8}, e^{-j33\pi/8}, e^{-j36\pi/8}, e^{-j39\pi/8}, e^{-j42\pi/8}, e^{-j45\pi/8}]\\
= [1, &e^{-j3\pi/8}, e^{-j3\pi/4}, e^{-j9\pi/8}, e^{-j3\pi/2}, e^{-j15\pi/8}, e^{-j9\pi/4}, e^{-j21\pi/8}, e^{-j3\pi},\\
&e^{-j27\pi/8}, e^{-j15\pi/4}, e^{-j33\pi/8}, e^{-j9\pi/2}, e^{-j39\pi/8}, e^{-j21\pi/4}, e^{j3\pi/8}]\\
= [1, &e^{-j3\pi/8}, e^{-j3\pi/4}, e^{j7\pi/8}, j, e^{j\pi/8}, e^{-j\pi/4}, e^{-j5\pi/8}, -1,\\
&e^{j5\pi/8}, e^{j\pi/4}, e^{-j\pi/8}, -j, e^{-j7\pi/8}, e^{j3\pi/4}, e^{3\pi/8}]
\end{aligned}
$$

6. (15 pts) [DFT of a cosine]

   (a) Find the length 16 DFT $X[k]$ of the unit-amplitude cosine defined as:

$$x(n) = \cos\left(\frac{\pi}{4}n\right)$$

Solution:

$$x(n) = \cos\left(\frac{\pi}{4}n\right) = \frac{1}{2}(e^{jn\pi/4} + e^{-jn\pi/4})$$

Then

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{15} \frac{1}{2}(e^{jn\pi/4} + e^{-jn\pi/4})e^{-j2\pi nk/16}\\
&= \frac{1}{2}\sum_{n=0}^{15} e^{jn\pi/4-j2\pi nk/16} + e^{-jn\pi/4-j2\pi nk/16}\\
&= \frac{1}{2}\sum_{n=0}^{15} e^{jn\pi(1/4-2k/16)} + e^{-jn\pi(1/4+2k/16)}\\
&= \frac{1}{2}\sum_{n=0}^{15} e^{jn\pi(2-k)/8} + e^{-jn\pi(2+k)/8}
\end{aligned}
$$

Note than when $k = 2$, then $e^{jn\pi(2-k)/8} = 1$ and $e^{jn\pi(2+k)/8} = e^{jn\pi/2}$. Therefore

$$
\begin{aligned}
X(2) &= \frac{1}{2}\sum_{n=0}^{15} 1 + e^{-jn\pi/2} = \frac{1}{2}\left(16 + \sum_{n=0}^{15} e^{-jn\pi/2}\right)\\
&= 8 + \frac{1}{2}(1 - j - 1 + j + 1 - j - 1 + j + 1 - j - 1 + j + 1 - j - 1 + j)\\
&= 8
\end{aligned}
$$

Similarly, for $k = 14$

$$\begin{aligned} X(2) &= \frac{1}{2}\sum_{n=0}^{15} e^{-jn3\pi/2} + 1 = \frac{1}{2}\Big(16 + \sum_{n=0}^{15} e^{-jn3\pi/2}\Big) \\ &= 8 + \frac{1}{2}(1+j-1-j+1+j-1-j+1+j-1-j+1+j-1-j) \\ &= 8 \end{aligned}$$

For $k \neq 2, 14$ remember the geometric series:

$$\sum_{i=A}^{B} r^i = \frac{r^A - r^{B+1}}{1-r}$$

Then

$$\begin{aligned} X(k) &= \frac{1}{2}\left(\frac{1-e^{j\pi(2-k)16/8}}{1-e^{j\pi(2-k)/8}} + \frac{1-e^{-j\pi(2+k)16/8}}{1-e^{-j\pi(2+k)/8}}\right) \\ &= \frac{1}{2}\left(\frac{1-e^{j2\pi(2-k)}}{1-e^{j\pi(2-k)/8}} + \frac{1-e^{-j2\pi(2+k)}}{1-e^{-j\pi(2+k)/8}}\right) \end{aligned}$$

Note that for $k \neq 2, 14$ both numerators are 0 and both denominators are defined ($\neq 0$), so $X(k) = 0$ for $k \neq 2, 14$.

Therefore

$$X(k) = [0,0,8,0,0,0,0,0,0,0,0,0,0,0,8,0]$$

(b) Find the length 16 DFT $X[k]$ of the unit-amplitude cosine defined as:

$$x[n] = \cos\left(\frac{\pi}{4}n - \pi/4\right)$$

Solution:
If we add a phase ϕ to the original cosine:

$$x(n) = \cos\frac{\pi}{4}n + \phi$$

Is easy to prove that (follow the same train of thought used in the previous part)

$$X(k) = \frac{1}{2}\sum_{n=0}^{15} e^{j\phi}e^{jn\pi(2-k)/8} + e^{-j\phi}e^{-jn\pi(2+k)/8}$$

Note that the new terms do not depend on n or k. Therefore, the previous derivation holds true, but now we need to multiply the new terms when corresponding. For $\phi = -\pi/4$ we end up with

$$X(k) = [0, 0, 8e^{-j\pi/4}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8e^{j\pi/4}, 0]$$

(c) Find the length 16 DFT $X[k]$ of the unit-amplitude cosine defined as:

$$x[n] = \cos\left(\frac{\pi}{4}n - \pi/2\right)$$

Solution:
Using the results discussed in the previous part:

$$X(k) = [0, 0, -8j, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8j, 0]$$

Comment on the differences between the three. You might find useful to plot the magnitude response ($|X[k]|$) and the phase response ($\angle X[k]$)

Solution: A constant time delay maps to a linear phase term in the DFT. Note that if the delay is $-\pi/2$, then to delayed cosine becomes a sine, so the answer to the third part is the DFT of $\sin\left(\frac{\pi}{4}n\right)$

Lab Assignments

For all lab assignments, submit your M-file scripts, functions, and figures in one zip file through coursework[1]. Within coursework, upload the zip file using the Drop Box menu.

The zip file should be named with your last name, first name and homework number. Each problem should be named with your last name, first name, homework number, and the problem number. So, for John Doe's zip file, the file should be titled `doe_john_hw2.zip`. For John Doe's answer to problem 2 on homework 2, the file would be titled `doe_john_hw2_q2.m`. Also, at the beginning of each script, include the following comment:

```
% Your Name / Lab # - Question #
```

For problems with question(s), include your answer(s) in the body of the script files as comments.

[1] http://coursework.stanford.edu

1. (30 pts) [Additive synthesis] Using your `additive` function (from previous homework), try to produce the following sounds:

   (a) Square wave (`square.wav`)

   (b) Triangle wave (`triangle.wav`)

   (c) Sawtooth wave (`sawtooth.wav`)

   Write a script which

   (a) prepares parameters (`f` and `Z`) for each sound,

   (b) utilizes your additive function to save the results as wave files (with names given above), and

   (c) plots every waveform for the first five periods.

   Please note:

   (a) Your sounds should be 2 seconds long.

   (b) Use 441 `[Hz]` for f or fundamental frequency and 44100 `[Hz]` for your sampling frequency f_s.

   (c) Make sure your function works fast enough, and your sounds are not clipped. If necessary, modify your function.

   (d) Verify your result with plot. Put four waveforms in one figure, and name them appropriately.

   (e) Submit your script and function (no plots). Include your name, lab and problem number. For each sound, how many partials do you need to obtain a faithful production of the waveform? How do they sound? Also, describe the characteristics of your frequency components.

   Solution:

```
% duration
dur = 2;        % sec
% frequency of waveforms
freq = 440;     % Hz
% sampling rate
fs = 44100;     % Hz
% number of harmonics, bandlimited by Nyquist frequency(fs/2) to avoid aliasing
nharm = floor(fs/2/freq);

% harmonic numbers of sawtooth, square and triangle
% square and triangle have only odd harmonics
sqr_harm = [1:2:nharm];
```

```
tri_harm = [1:2:nharm];
saw_harm = [1:nharm];

% ampltidues of harmonics
amp = 0.9;
sqr_Z = amp./sqr_harm*exp(-j*pi/2);     % adding sine waves
tri_Z = amp./(tri_harm.^2);             % adding cosine waves
saw_Z = amp./saw_harm*exp(-j*pi/2);     % adding sine waves

% synthesize waveforms
square = additive(freq*sqr_harm, sqr_Z, fs, dur, 'square.wav');
triangle = additive(freq*tri_harm, tri_Z, fs, dur, 'triangle.wav');
sawtooth = additive(freq*saw_harm, saw_Z, fs, dur, 'sawtooth.wav');

% number of periods to plot
nPeriod = 5;
samplesPerPeriod = fs/freq;
% sample index to plot
idx_sample = [1:ceil(samplesPerPeriod*nPeriod)];
% scaling in periods
peroid =(idx_sample-1)/samplesPerPeriod;

figure(1);

subplot(311);
plot(peroid, square(idx_sample)); grid;
title('sqaure wave'); xlabel('period'); ylabel('amplitude');
subplot(312);
plot(peroid, triangle(idx_sample)); grid;
title('triangle wave'); xlabel('period'); ylabel('amplitude');
subplot(313);
plot(peroid, sawtooth(idx_sample)); grid;
title('sawtooth wave'); xlabel('period'); ylabel('amplitude');
```

2. (15 pts) [ADSR envelope] ADSR ("Attack, Decay, Sustain, Release") envelopes are most commonly used in computer music to make a sound more realistic. In this case, we will be implementing a piece-wise linear ADSR envelope generator (there is also piece-wise exponential ADSR, but we won't be worrying about that in this lab). The envelope is broken into 4 linear sections: Attack, Decay, Sustain, and Release.

   - *Attack*: Amplitude increases linearly from 0 to a_1 during the time interval from 0 to t_1, which is usually very quick.

- *Decay*: Amplitude decreases linearly from a_1 to a_2 during the interval from t_1 to t_2.
- *Sustain*: Amplitude decreases from a_2 to a_3, but this decrease is typically much slower than the first two stages (so $t_3 - t_2$ is longer than the first two intervals).
- *Release*: Amplitude decreases to a_4, which is usually 0, over the interval from t_3 to t_4.

Write a Matlab function that implements an ADSR envelope generator. The syntax of your function should be:

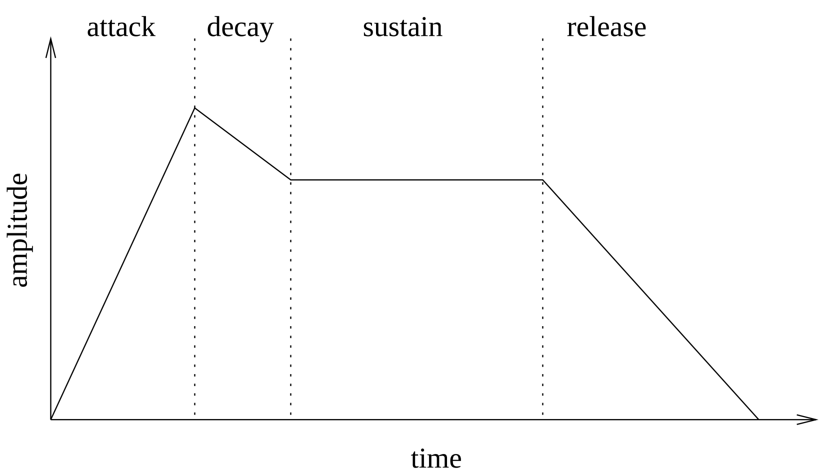
```
function A = ADSR (amps, durs, fs)

% function A = ADSR (amps, durs, fs)
% amps: vector of amplitude values at each transition point [0~1] (1x4)
% durs: vector of durations of each period [s] (1x4)
% fs: sampling rate in Hz
% Your Name / Lab#-Q#
```

(a) Elements of `amps` represent amplitude values at attack/decay, decay/sustain, sustain/release, and end point.

(b) Elements of `durs` represent durations of attack, decay, sustain, and release period (`durs=[t1,t2-t1,t3-t2,t4-t3]` in the figure).

Plot an envelope with `amps = [0.99 0.7 0.5 0]`, `durs = [0.3 0.7 3 1]`, and `fs = 200`. Submit your plot together with your function (or submit a script that creates the plot using your function). Note that you are only creating an amplitude envelope, not a sinusoid. From here, apply it to your *additive* function from last week and create an interesting sound example :)



Solution:

```
function A = ADSR (amps, durs, fs)

% function A = ADSR (amps, durs, fs)
% amps: vector of amplitude values at each transition point [0~1] (1x4)
% durs: vector of durations of each period [s] (1x4)
% fs: sampling rate in Hz

dur=sum(durs);
t = 0:1/fs:dur;

index_durs(1)=1+floor(durs(1)*fs);
index_durs(2)=1+floor(sum(durs(1:2))*fs);
index_durs(3)=1+floor(sum(durs(1:3))*fs);
%index_durs(4)=1+ceil(sum(durs(1:4))*fs);
index_durs(4)=length(t);

a1=amps(1)/durs(1);
A(1:index_durs(1))= a1*t(1:index_durs(1));

a2=(amps(2)-amps(1))/durs(2);
b2=amps(1)-a2*durs(1);
A(index_durs(1)+1:index_durs(2))= b2+a2*t(index_durs(1)+1:index_durs(2));

a3=(amps(3)-amps(2))/durs(3);
b3=amps(2)-a3*(sum(durs(1:2)));
A(index_durs(2)+1:index_durs(3))= b3+a3*t(index_durs(2)+1:index_durs(3));

a4=(amps(4)-amps(3))/durs(4);
b4=amps(3)-a4*(sum(durs(1:3)));
A(index_durs(3)+1:index_durs(4))= b4+a4*t(index_durs(3)+1:index_durs(4));

return
```

Example:

```
clear all
close all

%ADSR
amps = [0.99 0.7 0.5 0];
durs = [0.3 0.7 3 1];

A200 = ADSR (amps, durs, 200);
```

```
t200=0:1/200:sum(durs);
plot(t200,A200,'.');
hold
plot(t200,A200);
grid
title('ADSR envelope')
xlabel('time (sec)')
ylabel('amplitude')
```