UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 498MH Signal and Image Analysis

Homework 8

Fall 2013

Assigned: Friday, November 8, 2013 Due: Friday, November 15, 2013

Reading: SPF 12-3

Problem 8.1

Consider a pure tone at 9000Hz, $x(t) = \cos(2\pi 9000t)$.

(a) The signal $x(t)$ is sampled at $F_s = 10,000$ samples/second, resulting in a sampled signal $y[n]$. What is $y[n]$? Express your answer in the form $y[n] = \cos(\omega_a n)$, where $-\pi \leq \omega_a \leq \pi$.

(b) The signal $y[n]$ is passed through a sample-and-hold D/A converter, resulting in the piece-wise constant signal $z(t)$ given by

$$z(t) = y[n], \quad nT \leq t < (n+1)T$$

where $T = \frac{1}{F_s} = 10^{-4}$s. Because of the discontinuities, this signal has energy at many frequencies. At what frequencies does this signal have energy?

(c) Let $f(t) = h(t) * z(t)$. What is the value of $h(t)$ that will produce $f(t) = \cos(2\pi 1000t)$?

Matlab Exercises

Problem 8.2

This problem explores D/A aliasing. Unfortunately for this experiment, most D/A cards include an analog anti-aliasing filter, so we can't generate real aliasing from the D/A card. Instead, we will simulate it.

Create a 1-second, 3kHz tone, with a sampling frequency of $F_s = 8$kHz (thus the period is 8/3 of a sample: `n=1:8000; x=sin(2*pi*n*3000/8000);`). Play it at a sampling frequency of 8kHz (`soundsc(x,8000);`)

Play the same digital signal at a sampling frequency of 24kHz (`soundsc(x,24000);`). Notice that it's 1/3 the length, and 3 times the frequency!

Try upsampling the tone by inserting zeros (`y=zeros(1,24000);y(1:3:24000)=x;`) then play the result at a 24kHz sampling rate. You should hear a tone at 3kHz (just like the first tone), but with overtones at 5kHz and (if your sound card is good enough) also at 11kHz.

To better understand what you are hearing, compute spectra from the first 512 samples of each of these tones, and plot them with the correct frequency axes:

```
MX=abs(fft(x(1:512)));
fX= [0:511]*8000/512;
subplot(4,1,1);plot(fX(1:256),MX(1:256));
MY=abs(fft(y(1:512)));
fX= [0:511]*24000/512;
subplot(4,1,2);plot(fY(1:256),MY(1:256));
```

A real D/A card works by doing a sample-and-hold (piece-wise-constant) reconstruction of the digital signal, and by then passing the sample-and-hold signal through a lowpass filter. Let's simulate that by filtering $y[n]$ digitally. Since we upsampled by a factor of 3, we need to filter with a cutoff of $\frac{\pi}{3}$:

```
m=[-127.5:127.5];
hlpf=hamming(256) .* (1/3)*sin(pi*m/3)./(pi*m/3);
z=conv(hlpf,y);
```

Play back the signal $z[n]$ (`soundsc(z,24000);`—you should hear something more like a pure tone at 3kHz. Plot its spectrum (`MZ=abs(fft(z(1:512)));subplot(4,1,3);plot(fY(1:256),MZ(1:256));`)—you should see just one tone at 3kHz.

A fun thing to do is this: let's filter the upsampled signal in order to keep one of the aliased tones, while throwing away the original tone! You can do this by creating a bandpass filter. For example, in order to create a bandpass filter that passes everything between $\frac{\pi}{3}$ and $\frac{2\pi}{3}$ radians/sample, you can take an ideal lowpass filter at $\frac{2\pi}{3}$, and subtract from it an ideal lowpass filter at $\frac{\pi}{3}$:

```
hlpf2=hamming(256) .* (2/3)*sin(2*pi*m/3)./(2*pi*m/3);
hbpf = hlpf2 - hlpf;
f = conv(hbpf,y);
```

Play back the signal $f[n]$ (`soundsc(f,24000);`—you should hear a pure tone at 5kHz, which is the first aliased copy of the 3kHz tone. Plot its spectrum (`MF=abs(fft(f(1:512)));subplot(4,1,4);plot(fY(1:256),MF(1:256));`) you should see just one tone at 5kHz.

Turn in a copy of the plot with four subfigures, showing magnitude spectra of the four audio signals, with frequency axes labeled in Hertz.