Professor Fearing EECS120/Python MiniLab 2 v1.01 (draft) Fall 2016

Due at 6 pm, Fri. Oct. 21 on BCourses

Up to 2 people may turn in a single iPython (Jupyter) notebook. Upload .ipynb and answers in pdf.

This exercise is worth 2% grade. (Python exercises will be 9% of grade).

Notes: Use Python 2.7, not 3.

Notebook file is: `www-inst.eecs.berkeley.edu/~ee120/fa16/hwk/digitalrcvr.ipynb`
Sound file is: `www-inst.eecs.berkeley.edu/~ee120/fa16/hwk/xmit-signal.wav`

In this Python exercise, you will model a digital quadrature amplitude modulation receiver. The received signal includes two channels, each with two signals. Similar to PS6-3, the received signal is:

$$r(t) = s_1(t)\cos(\omega_c t) + s_2(t)\sin(\omega_c t) + s_3(t)\cos(\omega_d t) + s_4(t)\sin(\omega_d t)$$

Here $\omega_c = (2\pi)300.0 \times 10^3 s^{-1}$ and $\omega_d = (2\pi)316.0 \times 10^3 s^{-1}$.

The discrete time signal $r[n] = r(nT_s)$ where $\frac{1}{T_s} = 16 \cdot 44.1$kHz. This file is provided as `xmit-signal.wav`. The signals $s_1(t)...s_4(t)$ are substantially bandlimited to 8 kHz.

The output of the Jupyter notebook should be the recovered signals $s_1[n], s_2[n], s_3[n], s_4[n]$ in 4 .wav files `sig1.wav, sig2.wav, sig3.wav, sig4.wav` which are downsampled to 44.1kHz.

The data file contains $2^{22} = 4,194304$ samples (a power of 2 is chosen for FFT efficiency). Efficient python code will be needed to have short run times. In particular `np.multiply()` with vectors is much quicker than a `for` loop. Use for speed, use the built-in numpy FFT `np.fft.fft(x)` and inverse FFT `np.fft.ifft(X)`.

1. (10 pts) Draw a block diagram for the digital receiver to recover $s_1[n], s_2[n], s_3[n], s_4[n]$ from $r[n]$.

2. (20 pts) Sketch approximate DFT spectra for $R[k], S_1[k]$ and spectra before filtering.

3. (10 pts) Find k_c and k_d which correspond to ω_c and ω_d respectively. What is the bandwidth taken by the modulated signals, $r(t)$? In the Jupyter notebook specify k_{min} and k_{max} to plot $R[k]$ in this range.

4. (10 pts) If a digital low pass filter has a cutoff frequency 8 kHz, what is the corresponding k_{cutoff}? Specify a digital LPF $H[k]$ with cutoff frequency 8 kHz. Be sure to specify H such that a real input will give a real output. Sketch $H[k]$.

5. (40 pts) Complete the functions necessary in the Jupyter notebook to recover $s_1[n], s_2[n], s_3[n], s_4[n]$ from $r[n]$ and store in the .wav files. If your algorithm is working correctly, each file should be a short playable segment.

6. (10 pts) Performance.
Briefly note any quality issues in the 4 recovered signals. How large can the cutoff frequency be before channels interfere? Is the interference audible with large cutoff frequency?