DSP First

Mini-Project 05: Note Detection

Work alone for this mini project. You are free to discuss ideas with others.

Background:

For this mini project you will write a simple MATLAB function that listens to a tone and identifies what note it is. The files **`Note0N.wav`** (where *N* is a number from 2 to 9) were zipped and are right below this file on the CD-ROM. Your job is to decide what note it is. Use `specgram` to find the frequency of the lowest note. Call it f_0.The highest note should be one octave above ($2f_0$).

Approach

You are to design 13 filters. Each filter will pass just one note. The first filter will be tuned to pass f_0; the next filter will pass $f_0 * 2^{1/12}$ Hz, and so on. The 13th filter will pass $2f_0$. See Lab 1 if you don't understand the $2^{1/12}$. You will then pass the unknown signal through each of the filters, i.e., the input signal is the input to each filter. You then measure the power coming out of each filter and the filter with the largest power corresponds to the note that was played.

Sample Code

The following MATLAB code is a good starting point:

```
function maxNote = toneDetect(xx, fs, DEBUG)
%toneDetect Returns a number indicating what note was played.
%   toneDetect(xx, fs) xx in the input sound, fs is the sample rate.
%
%   toneDetect(xx, fs, 1)   Like above, but turns DEBUGing on.
%

%   Copyright 2003 Mark A. Yoder, Rose-Hulman
%

if (nargin < 3) % Turn off debugging if last argument not present.
    DEBUG = 0;
end
f0 =                            Fill in your value of f0
all = xx';
max = 0;
maxNote = -1;

for note = 0:12;
       freq = f0 * 2.^(note/12);

       what = Insert code to compute ω̂ (what) based on freq and the sampling rate.

       aa = Insert code to make a bandpass filter with center at ω̂ (what).
       bb = 1;
       yy = filter(bb, aa, xx);

       pow = Insert code to find the total power in the signal yy


       if pow > max            % Is this the loudest note?
              max = pow;           % Yes, remember it.
```

```
                maxNote = note;
        end
        all = [all yy'];        % Save output for debugging.
end

if(DEBUG)       % If debuging in on, play outputs of all the filters.
    soundsc(all, fs)
end
```

Test your code with something like:

```
[xx, fs] = wavread('note1');
noteNum = toneDetect(xx,fs);
```

Filter the Frequencies

You should know enough to pick the *a* coefficients for your filter. Be sure the *a* coefficients are real so your output will be real. Also be sure your poles aren't on the unit circle; it could become unstable. Think in terms of poles and zeros and where you would like to place them. Don't use any more poles/zeros than needed. Hint: MATLAB's `poly` command may come in handy. In your memo, note the pole/zero locations you are using.

Due Date:

What is due:

1. One page memo describing what you did. Be sure to have a sentence or two intro and conclusion. Have a table listing the wav file name (e.g `Note01.wav`), and the note you think is in the file for each wav file. Highlight any **extras** you did.
2. Include a pole/zero plot (use `zplane`) and a frequency response plot (use `freqz`, but have linear, not dB, scales) of one of your filters. Place these inline in your memo, not as separate pages. Be sure to give them captions and refer to them in your memo.
3. Your MATLAB code. Attach this to your memo.