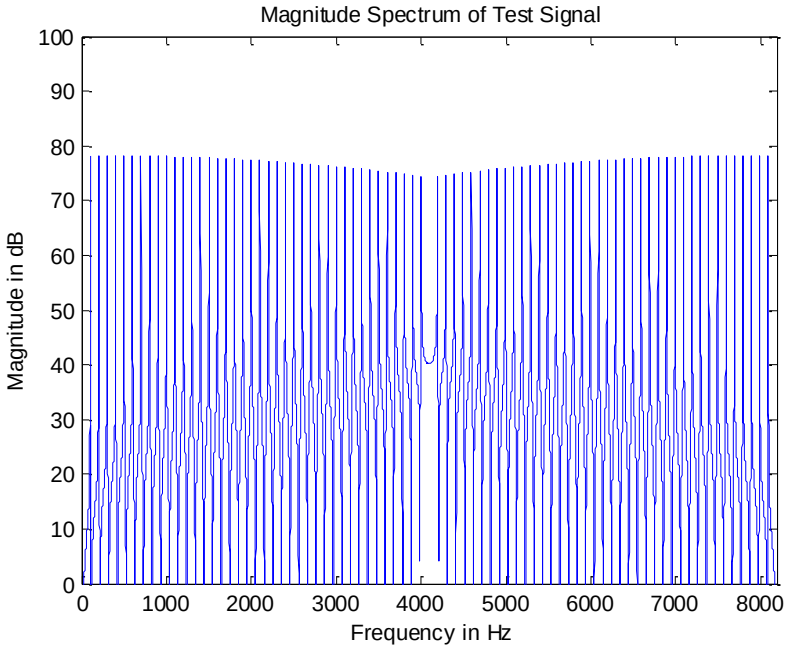
ECE 320
Matlab Project 3

Due electronically by April 30, 2014, 6PM

When you submit your projects, please include in your .zip file all scripts and functions necessary to replicate your results. I found your scripts and functions more informative than .mat files.

Part 1:
In this part, you will write a script: part1.m to create a test signal made up of the first 40 harmonics of 100 Hz, assuming the time between samples is 1/8192 s. You will then run the test signal through a lowpass filter and verify the effect of the filter.

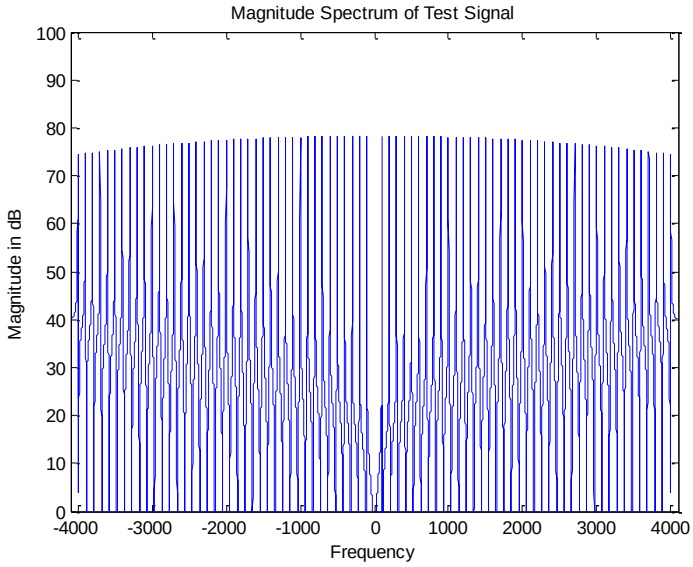
1) Generate a 2 second time vector (i.e. t = 0:1/8192:2) and, starting with x = zeros(size(t)), use a "for" loop with index k=1:40 to recursively modify x by adding sine waves at harmonic frequencies k*100 Hz to x.
2) Compute the magnitude spectrum of x using the fft command in Matlab (i.e. X = fft(x);). Plot magnitude in dB vs frequency. Your magnitude spectrum should look like:



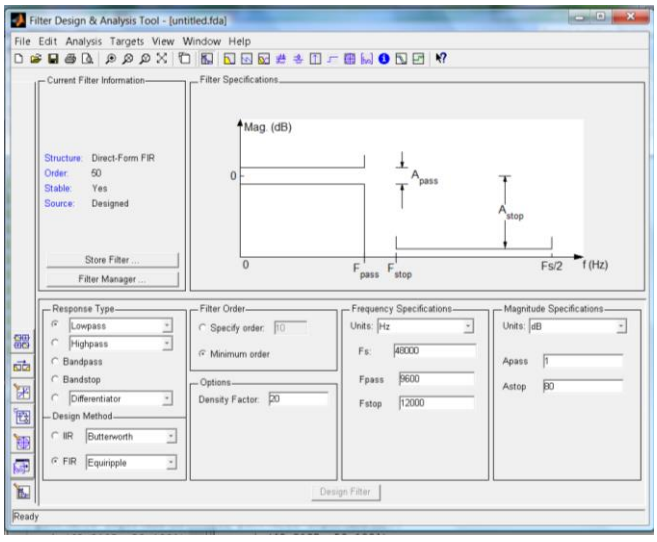
I used 'linspace' to generate the frequency vector and the 'axis' command to restrict the magnitude and frequency ranges for this plot.

3) A more appropriate analog frequency plot would display symmetrical positive and negative frequencies. To obtain such a plot, apply fftshift to X (it will shift the second half of the X values to the beginning of the vector, effectively shifting the frequencies

between 4096 and 8192 to appear from -4096 to 0) and subtract 4096 from freq so that the "corrected" magnitude plot looks like:

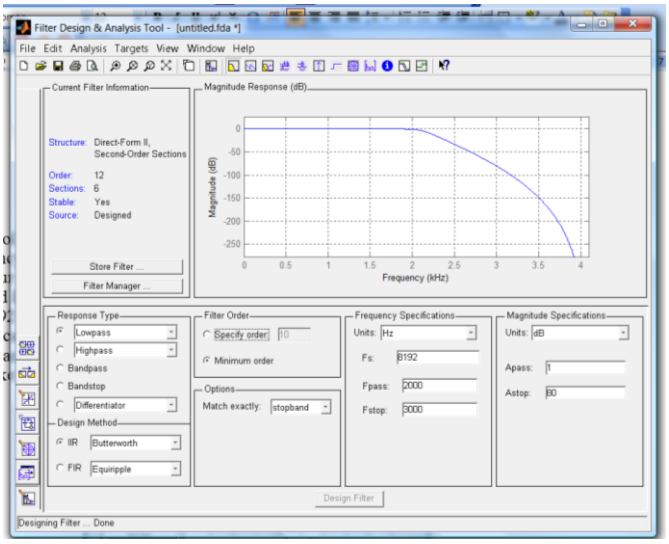


4) Use the command fdatool to run the Filter Design & Analysis Tool. The GUI initially looks like this:



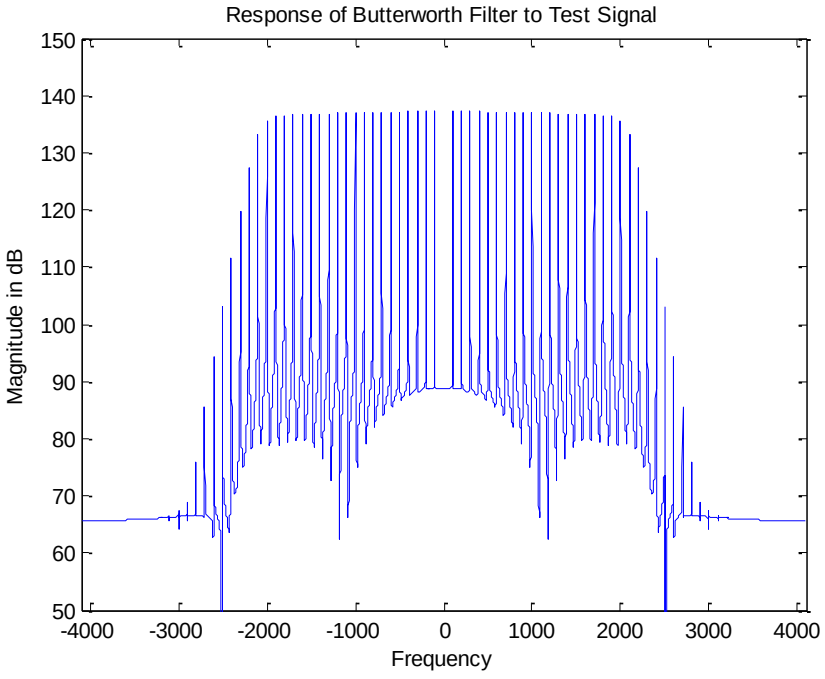
This tool will allow you to create/modify filter parameter settings and see the resulting frequency response.

For your initial filter, keep the Response Type: Lowpass. Then click on the Design Method: IIR (start with the default type of Butterworth). For Frequency Specifications, use 8192 Hz for Fs, 2000 Hz for Fpass, 3000 Hz for Fstop. You can leave the Magnitude Specifications at the defaults of Apass = 1 dB and Astop = 80 dB. Click on "Design Filter" and the frequency response of your resultant filter will be displayed. It should look like:



You can export your filter parameters to your workspace by clicking on "File" and pull down to "Export". Export as coefficients; add _LP to the default variable names SOS and G. Click "Override Variables" if you rerun the filter designer and want to export the updated parameters.

5) Obtain a filtered version of your signal using the sosfilt command: y = sosfilt(SOS_LP, x). This is necessary because the filter design tool creates 2nd order sections (SOS) that are cascaded to generate the overall result (12th order means 6 sections in this case). Write a script, lpfilt.m, that will generate the filtered signal and plot the magnitude spectrum of the filtered output on the same axis range as your original input (-4096 to 4096 Hz and a vertical dB range of 100). After applying fftshift, it should look like:



6) Note that while the magnitude range is still 100 dB, it is shifted up by 40 dB. The shift is because the gain terms that were exported with the SOS_LP coefficients were ignored. To correct the gain, you need to multiply all the G_LP terms together then multiply your output by the gain. I'll award 2 points extra credit if you include in your script code to do this and a plot of the adjusted spectrum.

In your .doc file, include magnitude plots of your test signal before and after filtering (and after gain adjustment if done). Also include a copy of the contents of the matrices SOS_LP and G.

7) Listen to your signal before and after filtering. What differences did you notice in the sound due to filtering?

8) Now go back to the fdatool and experiment. First, change the Design Method. Keep the IIR button selected but change from Butterworth to Chebyshev Type 1 then to Chebyshev Type 2 then to Elliptic. Keep the Options setting to Match exactly: stopband. What did you notice about the frequency responses of the resulting filters and the order required? What happens when you switch from IIR to FIR Equiripple? Using the Elliptic design, what happens when you change Apass from 1 dB to 10 dB? What happens when you change Astop from 80 dB to 40 dB? Note, you do not need to re-filter your test signal with any of these, just interpret the frequency responses displayed in the fdatool.

If you wish to see the effect of an FIR filter on a signal, when you export, you'll only have one coefficient: num. Then, the output is generated using the filter command: y = filter(num, 1, x).

Part 2:

1) Now, you are going to use a bandpass filter to filter your melody. Use the melody version from Project 2 that was designed to sound like a clarinet. The specifications for the bandpass filter are: pass frequencies between 350 and 3500 Hz with no more than 1 dB of amplitude distortion and attenuate frequencies below 50 and above 4000 Hz at least 40 dB. You get to select the type of filter and any other parameters.

2) When you've designed your filter, obtain a screen capture of the fdatool showing the parameters you've chosen along with the frequency response. Include the screen capture in your .doc file. Export the parameters to your workspace. Include a printout of your SOS and G (or num) parameters in your .doc file. Filter your melody. Include spectrograms of your initial melody and your filtered melody in your .doc file.

Part 3:

1) Obtain the score for a piece of music. There are several available for free at various internet sites or you can ask me and I'll give you one. The score doesn't have to be any longer than 6 lines. Try to look for one that's for voice or a single note

instrument like a horn or wind, otherwise you'll need to be able to synthesize chords (multiple notes that may vary in duration).

2) Synthesize the score using the tools you've developed over the term to make it sound natural and interesting (you don't need to use every tool and you can use others if you want). If you don't understand or recognize musical notation on your score, please ask.

3) For this part, turn in a .pdf of the score or a link to it if you can't save it as a .pdf, the script, titled part3.m, you use to generate your melody and any functions called by the script that you've written.

4) Explain in your .doc file why you chose the effects you did and comment on how successful you were in making the melody sound natural.

5) Did you find implementing convolution, Fourier series and filtering in these projects helpful in understanding the concepts as covered in class? What did you learn the most from?

6) What did you like most about these projects? What did you like the least? (I won't grade your answers to these questions, I'm just looking for feedback to possibly improve them for next year.)