BONUS Filter Design Project

CSE 3313 – 001 Spring 2023

Due: Wednesday, May 10

Submission Format: .zip file

Description

In this project, you will be designing a filter to remove unwanted noise from an audio file. You will accomplish with the paper design of a Butterworth lowpass filter. The designed filter will then be implemented in MATLAB and used to filter the audio and compare to the unfiltered audio.

1. **Audio Frequency Analysis**

    a. Use the *audioread()* MATLAB command to provide the sampling frequency Fs of the file and read the contents of 'noisyaudio.wav' into an array.
    b. Calculate and plot the magnitude of the DFT of the audio sample vs frequency.
    c. Use the DFT to analyze the noisy signal. The DFT should show energy in the speech frequencies from about 100 Hz to 2 kHz. The plot should also show significant high-frequency noise from about 2.5 kHz to 5.5 kHz. The goal will be to design a filter that will remove this high-frequency noise occurring above 2.5 kHz, while having minimal impact on the speech frequencies up to around 2 KHz.

2. **Filter Design**

    a. Based on the DFT of the signal, decide on a digital frequency ω_p for the end of the passband and a digital frequency ω_s for the beginning of the stop band.
    b. Assume no more than 1dB attenuation in the passband. Using your analysis in 1c, decide on the appropriate minimum attenuation for the stopband so that the noise level is much lower than the desired speech.
    c. Use the Butterworth filter design procedure in the lecture for the impulse invariance method to design an analog filter according to the specification on the digital filter that you found in 2a and 2b above. This will involve finding the filter order, N, and the Butterworth cutoff frequency, Ω_c. Describe where you are meeting filter requirements and where you are exceeding filter requirements in choosing Ω_c.
    d. Find the equivalent ω_c for the digital filter based on 2c above.
    e. Plot the logarithmic gain of the frequency response of your analog filter using the equation below.

$$|H_a(s)| = 20\log_{10}\left(\sqrt{\frac{1}{1+\left(\frac{j\Omega}{j\Omega_c}\right)^{2N}}}\right)$$

3. Filter Implementation

Manual filter implementation would normally involve finding $H(s)$ for the analog filter designed in step 2, converting this to $H(z)$ using the bilinear transformation, and implementing this transfer function using a difference equation or signal flow graph. For higher-order filters, this can take a lot of work on paper! Luckily, we now have computers to do all of this for us.

a. The MATLAB *butter()* command will design a digital filter for us based on our desired digital cutoff frequency. Use the MATLAB *butter()* command to find the numerator and denominator (b and a) polynomial coefficients for a digital Butterworth lowpass filter with the final digital cutoff frequency ω_c and order N you found in step 2. In this command, the cutoff frequency Wn is specified as a percentage of $F_s/2$ in the continuous domain or a percentage of π in the digital domain. So, if your ω_c is 0.25π rad/s, you would calculate Wn as:

$$\mathrm{Wn} = \frac{\omega_c}{\pi} = 0.25$$

in the *butter()* command for cutoff frequency.

b. Use the a and b filter coefficients you found in 3a and the MATLAB *filter()* command to create a filtered version of the original noisy audio sample.
c. Calculate and plot the magnitude of the DFT of the filtered audio sample.
d. Compare the unfiltered audio and filtered audio DFT plots. Did the filter reduce the noise in the stop band frequencies and keep the desired speech in the passband frequencies?
e. Use the *sound()* MATLAB command to listen to the original audio and filtered audio. What difference do you hear?
f. Write the new filtered audio file to 'filteredaudio.wav' using the *audiowrite()* MATLAB command. Be sure to use the same sampling frequency as the original file.

4. Report

Write a report to summarize your design, show your plots, and describe your findings. Upload a .zip file with your report, MATLAB code, audio files, and plots.

5. BONUS BONUS

Write your own code to implement this filter using a difference equation. You can use the a and b coefficients to form a difference equation and solve for the output recursively. Assume $y = 0$ for $n < 0$ as an initial condition. Plot the magnitude of the DFT before and after applying the filter to show that your difference equation works.