EE3025 ASSIGNMENT- 1

S.sahithi - ES18BTECH11015

Download all codes from

https://github.com/GuttaRamaneesh/EE3025

1 Problem

Modify the following code given in problem 2.3 with different input parameters to get the best possible output.

```
import soundfile as sf
from scipy import signal

#read .wav file
input_signal,fs = sf.read('Sound_Noise.wav')

#sampling frequency of Input signal
sampl_freq=fs

#order of the filter
order = 3

#cutoff frequency 4kHz
cutoff_freq=4000.0

#digital frequency
Wn=2*cutoff_freq/sampl_freq

# b and a are numerator and denominator
    polynomials respectively
b, a = signal.butter(order,Wn,'low')

#filter the input signal with butterworth filter
output_signal = signal.filtfilt(b, a, input_signal)
#output_signal = signal.lfilter(b, a, input_signal)

#write the output signal into .wav file
sf.write('Sound_With_ReducedNoise.wav',
    output_signal, fs)
```

2 Solution

The main parameters for a standard Butterworth filter are

- Cutoff frequency
- Order of the filter
- function for applying the filter

2.1 *Cutoff Frequency*

Cutoff frequency for a Butterworth filter is the $-3dB$ point after which all the frequency components roll off down to zero.
To select a good cutoff frequency, we observe the spectogram from Problem 2.2 and conclude that most of the frequencies corresponding to piano notes are around mid point of 440Hz and 4700Hz, so let the cutoff frequency be $(440 + 4700)/2 = 2570Hz$

2.2 *Order of the filter*

The main disadvantage of the Butterworth filter is that it achieves the pass band flatness at the expense of a wide transition band as the filter changes from the pass band to the stop band.
We can fix this by increasing the order, but having a very high order would create numerical instabilities while simulating. So we will stick to the general order 4.

2.3 *function for applying the filter*

The latter function just passes the ray through the butterworth filter this introduces a phase shit of -/2 whereas the signal.filtfilt does more than that .It is a forward or backward filter .This helps to cancel the phase shift earlier . For getting the good results ,we can apply signal.filtfilt mora than once . But the only backdrop we can see in this filter is it can only be used on recorded signals that are already completed.

3 Results

After applying the filter with above parameters we observe that we still have noise. The frequency response of signal before and after applying the filter shows that the transition region is too wide,

there are too many non zero frequency components after the cutoff frequency.

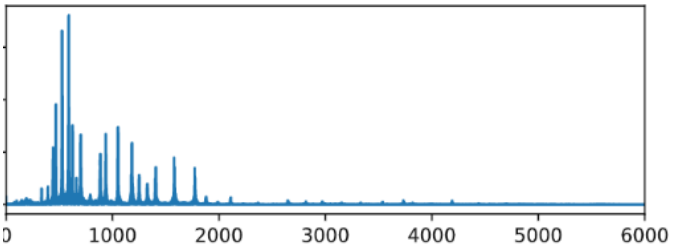

Fig. 0: Before Filtering

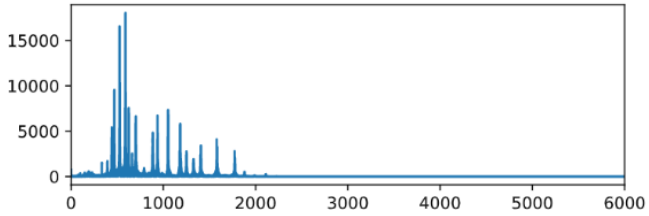

Fig. 0: After Filtering

To overcome this we would need a steeper transition. Since we cant increase the order, one solution is to cascade multiple better worth filters. The transfer function rolls off to zero from pass band to stop band, as we cascade more and more filters the gain values< 1 would get multiplied and reach closer to zero.

Parameter	Original	Filtered
Cutoff Frequency	2570Hz	2570Hz
Order	-	4
No. of Cascades	-	19
Fraction of signal before cutoff	82.35%	99.51%
Fraction of signal after cutoff	17.64%	0.485%

The clear information for understanding about the noise present in the signal we sum up all the non zero frequency components before and after the cutoff frequency to define the metric fraction which tells us how much non zero frequency component is present before and after the cutoff

The input and output wav files can be found at - input:

https://github.com/GuttaRamaneesh/EE3025/blob/main/Sound_Noise.wav

output:

https://github.com/GuttaRamaneesh/EE3025/blob/main/Reduced_Noise.wav