ISTANBUL TECHNICAL UNIVERSITY

SIGNALS AND SYSTEMS

Homework Report

150160529

Nurettin Kağan Çocalak

Problem 1

a)

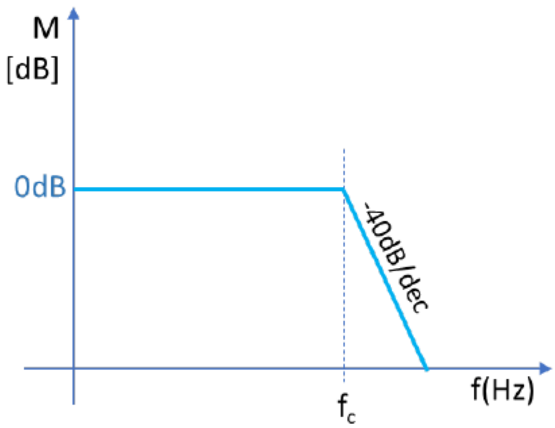

Figure 1: LPF with 0 gain

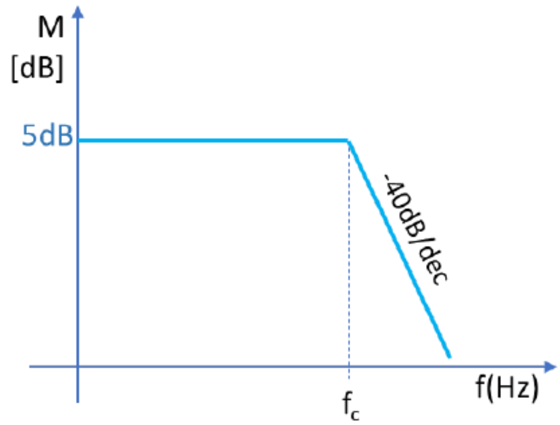

Figure 2: LPF with 5dB gain

In Figure 1, there is no gain and its order is second order. Magnitude Low pass response formula is $M_{LP} = 1/\sqrt{(1+(W/W_c)^{2n})}$. So our response is $1/\sqrt{(1+(W^4/W_c^4))}$.

In Figure 2, LPF with 5dB gain and magnitude response is $\sqrt[4]{10}/\sqrt{(1+(W^4/W_c^4))}$. I used this $\sqrt[4]{10}$ value for gain in our code.

Pseudocode

1: procedure Second Order Low Pass Filter

2: Input X

3: Input fc

4: Input Δt

5: Input gain

6: α = Δt / (Δt + 1/(2* pi * fc))*gain

7: Y[0] ← αX[0]

8: Y'[0] ← αY[0]

9: for i ← 0 to X.length - 1 do

10: Y[i+1] ← Y[i](1 - α) + αX[i+1]

11: Y'[i+1] ← Y'[i](1 - α) + αY[i+1]

return Y'

b) In this part, I implement the pseudo code to python. Firstly with this lpf function created.

```
def lpf(samples,fc,delta,gain):                          #low pass filter function definition with samples, cut off, delta and gain inputs
    alpha = delta/(delta + 1 /(2*np.pi*fc))*gain         #calculation of alpha with gain for filters
    resultFirstOrder = samples.copy()                    #resultFirstOrder arrayz
    resultSecondOrder = samples.copy()                   #resultSecondOrder array

    resultFirstOrder[0] = alpha * samples[0]             #samples multiply with alpha for the first order filter
    resultSecondOrder[0] = alpha * resultFirstOrder[0]   #created first order filter multiply with alpha for the creation of second order filter

    for i in range(len(samples) - 1):

        resultFirstOrder[i+1] = resultFirstOrder[i]*(1-alpha) + alpha*samples[i+1]            #calculation for first order low pass filter for len(samples) inputs
        resultSecondOrder[i+1] = resultSecondOrder[i]*(1-alpha) + alpha*resultFirstOrder[i+1]  #calculation for second order low pass filter thanks to first order

    return resultSecondOrder
```

Then audio files reading part works.

```
wavFileNameFirst ='Africa.wav'
wavFileNameSecond = 'WinnerTakesAll.wav'
outputFileNameWithGain = '5dBGainLPF.wav'
outputFileNameWithoutGain = '0dBGainLPF.wav'

#read wav file
obj = wave.open(wavFileNameFirst,'rb')          #Africa.wav is opened and readed
amplitudeWidth = obj.getsampwidth()             #sample width to in bytes
frameRate = obj.getframerate()                  #sampling frequency
nTimesFrames = obj.getnframes()                 #number of audio frames
readFrames = obj.readframes(nTimesFrames)       #reads and returns at most n frames of audio, as a bytes object
samples = np.fromstring(readFrames, np.int16)   #create array with frames
obj.close()                                     #close the stream if it was opened by wave module

obj2 = wave.open(wavFileNameSecond,'rb')        #WinnerTakesAll.wav is opened and readed
amplitudeWidth2 = obj2.getsampwidth()           #sample width to in bytes
frameRate2 = obj2.getframerate()                #sampling frequency
nTimesFrames2 = obj2.getnframes()               #number of audio frames
readFrames2 = obj2.readframes(nTimesFrames2)    #reads and returns at most n frames of audio, as a bytes object
samples2 = np.fromstring(readFrames2, np.int16) #create array with frames
obj2.close()                                    #close the stream if it was opened by wave module
```

With fc, delta, gain and samples which come from read file part, lpf function called.

```
fc = 2000.0
delta = 1.0/44100.0
#calculate low pass filter
gain = 1
lpFilteredNoGain = lpf(samples,fc,delta,gain).astype(samples.dtype)       #lpf function is called for Africa.wav file with no gain
gain = pow(10,0.25)
lpFiltered = lpf(samples,fc,delta,gain).astype(samples.dtype)             #lpf function is called for Africa.wav file with 5dB gain
gain = 1
lpFilteredNoGain2 = lpf(samples2,fc,delta,gain).astype(samples2.dtype)    #lpf function is called for WinnerTakesAll.wav file with no gain
gain = pow(10,0.25)
lpFiltered2 = lpf(samples2,fc,delta,gain).astype(samples2.dtype)          #lpf function is called for WinnerTakesAll.wav file with 5dB gain
```

And finally results which are values of returning from lfp function, are written at write file part.

```
#write file with outputname for Africa0dBGainLPF
fileNameFirst = wavFileNameFirst.split(".",1)                   #write for outputfilename file name split for just Africa
outputFileNameNoGain = fileNameFirst[0] + outputFileNameWithoutGain     #and combine with 0dBGainLPF
outputName = wave.open(outputFileNameNoGain,'w')                #filtered file result open and write with new name which is created with outputFileNameWithoutGain
outputName.setparams((1,amplitudeWidth,frameRate,nTimesFrames,obj.getcomptype(),obj.getcompname()))  #accepts parameter tuple
outputName.writeframes(lpFiltered.tobytes('C'))                 #write audio frames and make sure they are correct
outputName.close()                                              #close the stream if it was opened by wave module

#write file with outputname for Africa5dBGainLPF
outputFileName = fileNameFirst[0] + outputFileNameWithGain      #combine with 5dBGainLPF
outputName = wave.open(outputFileName,'w')                      #filtered file result open and write with new name which is created with outputFileNameWithGain
outputName.setparams((1,amplitudeWidth,frameRate,nTimesFrames,obj.getcomptype(),obj.getcompname()))      #accepts parameter tuple
outputName.writeframes(lpFiltered.tobytes('C'))                 #write audio frames and make sure they are correct
outputName.close()                                              #close the stream if it was opened by wave module

#write file with outputname for WinnerTakesAll0dBGainLPF
fileNameSecond = wavFileNameSecond.split(".",1)                  #write for outputfilename file name split for just WinnerTakesAll
outputFileNameNoGain = fileNameSecond[0] + outputFileNameWithoutGain     #and combine with HPF.wav
outputName = wave.open(outputFileNameNoGain,'w')                 #filtered file result open and write with new name which is created with outputFileNameWithoutGain
outputName.setparams((1,amplitudeWidth2,frameRate2,nTimesFrames2,obj2.getcomptype(),obj2.getcompname()))    #accepts parameter tuple
outputName.writeframes(lpFiltered2.tobytes('C'))                 #write audio frames and make sure they are correct
outputName.close()                                               #close the stream if it was opened by wave module

#write file with outputname for WinnerTakesAll5dBGainLPF
outputFileName = fileNameSecond[0] + outputFileNameWithGain     #combine with 5dBGainLPF
outputName = wave.open(outputFileName,'w')                      #filtered file result open and write with new name which is created with outputFileNameWithGain
outputName.setparams((1,amplitudeWidth2,frameRate2,nTimesFrames2,obj.getcomptype(),obj.getcompname()))    #accepts parameter tuple
outputName.writeframes(lpFiltered2.tobytes('C'))                #write audio frames and make sure they are correct
outputName.close()                                              #close the stream if it was opened by wave module
```

c) In this part, code read from file again.

```
wavFileNameFirst ='Africa.wav'
wavFileNameSecond = 'WinnerTakesAll.wav'
outputFileNameWithoutGain = 'HPF.wav'

#read wav file
obj = wave.open(wavFileNameFirst,'rb')              #Africa.wav is opened and readed
amplitudeWidth = obj.getsampwidth()                 #sample width to in bytes
frameRate = obj.getframerate()                      #sampling frequency
nTimesFrames = obj.getnframes()                     #number of audio frames
readFrames = obj.readframes(nTimesFrames)           #reads and returns at most n frames of audio, as a bytes object
samples = np.fromstring(readFrames, np.int16)       #create array with frames
obj.close()                                         #close the stream if it was opened by wave module

obj2 = wave.open(wavFileNameSecond,'rb')            #WinnerTakesAll.wav is opened and readed
amplitudeWidth2 = obj2.getsampwidth()               #sample width to in bytes
frameRate2 = obj2.getframerate()                    #sampling frequency
nTimesFrames2 = obj2.getnframes()                   #number of audio frames
readFrames2 = obj2.readframes(nTimesFrames2)        #reads and returns at most n frames of audio, as a bytes object
samples2 = np.fromstring(readFrames2, np.int16)     #create array with frames
obj2.close()                                        #close the stream if it was opened by wave module
```

I subtracted the low pass filter from our samples and I found the high pass filter results.

```
def hpf(samples,fc,delta,gain):                  #high pass filter function definition with samples, cut off, delta and gain inputs
    alpha = delta/(delta + 1 /(2*np.pi*fc))*gain #calculation of alpha with gain for filters
    resultFirstOrder = samples.copy()            #resultFirstOrder arrayz
    resultSecondOrder = samples.copy()           #resultSecondOrder array
    resulthpf = samples.copy()                   #resulthpf array

    resultFirstOrder[0] = alpha * samples[0]             #samples multiply with alpha for the first order filter
    resultSecondOrder[0] = alpha * resultFirstOrder[0]   #created first order filter multiply with alpha for the creation of second order filter
    resulthpf[0] = 0

    for i in range(len(samples) - 1):
        resultFirstOrder[i+1] = resultFirstOrder[i]*(1-alpha) + alpha*samples[i+1]              #calculation for first order low pass filter for len(samples) inputs
        resultSecondOrder[i+1] = resultSecondOrder[i]*(1-alpha) + alpha*resultFirstOrder[i+1]   #calculation for second order low pass filter thanks to first order
        resulthpf[i+1] = samples[i+1]-resultSecondOrder[i+1]                                    #calculation of high pass filter with subtruct low pass filter from samples

    return resulthpf
```

```
fc = 2000.0
delta = 1.0/44100.0
#calculate low pass filter
gain = 1
hpFiltered = hpf(samples,fc,delta,gain).astype(samples.dtype)          #hpf function is called for Africa.wav file
hpFiltered2 = hpf(samples2,fc,delta,gain).astype(samples2.dtype)       #hpf function is called for WinnerTakesAll.wav file
```

And finally write this results.

```
#write file with outputname
fileNameFirst = wavFileNameFirst.split(".",1)                       #write for outputfilename file name split for just Africa
outputFileNameNoGain = fileNameFirst[0] + outputFileNameWithoutGain    #and combine with HPF.wav
outputName = wave.open(outputFileNameNoGain,'w')                      #filtered file result open and write with new name which is created with outputFileNameNoGain
outputName.setparams((1,amplitudeWidth,frameRate,nTimesFrames,obj.getcomptype(),obj.getcompname()))  #accepts parameter tuple
outputName.writeframes(hpFiltered.tobytes('C'))                       #write audio frames and make sure they are correct
outputName.close()                                                    #close the stream if it was opened by wave module

fileNameSecond = wavFileNameSecond.split(".",1)                        #write for outputfilename file name split for just WinnerTakesAll
outputFileNameNoGain = fileNameSecond[0] + outputFileNameWithoutGain    #and combine with HPF.wav
outputName = wave.open(outputFileNameNoGain,'w')                       #filtered file result open and write with new name which is created with outputFileNameNoGain
outputName.setparams((1,amplitudeWidth2,frameRate2,nTimesFrames2,obj2.getcomptype(),obj2.getcompname()))    #accepts parameter tuple
outputName.writeframes(hpFiltered2.tobytes('C'))                       #write audio frames and make sure they are correct
outputName.close()                                                     #close the stream if it was opened by wave module
```

My output files in this drive address:
https://drive.google.com/drive/folders/18LZ7oAWSggy9RoUTUBe-6F40IqH8Y7o2?usp=sharing