BIOE 198MI Biomedical Data Analysis. Spring Semester 2019.
Lab6: Signal processing and filter design

Problem Statement: In this lab, we are considering the problem of designing a window-based digital filter (lowpass, highpass, bandpass) using MATLAB function fir1.

A. First let's load the bird chirping sound.

Load the MATLAB built-in audio file 'chirp'. look at the workspace and find out what's in the 'chirp' file.

```
load chirp;
whos; % Look at the workspace and find out what variables are loaded.
```

Q1: What is the sampling frequency(`Fs`) of the data? What is the meaning of sampling? Based on `y` and `Fs`, how can we compute the time duration of the `y`?

B. Record something of your own

Use MATLAB functions to record 3 seconds of your own voice data.

```
NumBit = 16; % bits per sample
NumChannel =1; % number of channels
Time = 3; %second
disp('get ready:')
pause(1);
disp('recording:')
recObj = audiorecorder(Fs,NumBit,NumChannel);
recordblocking(recObj, Time);
disp('finish recording.')
voice = getaudiodata(recObj)
```

First create an audio block with specific sampling frequency, number of bits per sample and number of channels. recObj = audiorecorder(Fs,NumBit,NumChannel);

Use function '`recordblocking`' to record certain time of voice data recordblocking(recObj, Time);

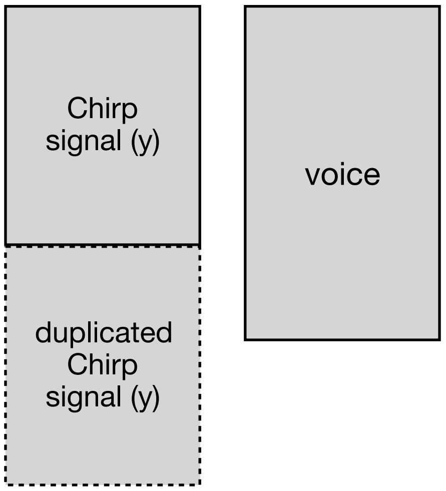
Use function '`getaudiodata`' to store the audio signal into an array.

Q2: `NumBit` can be 8, 16 or 24. play with recorded audio signal using function '`sound`' and describe the effect of `NumBit`(bit depth).

C. Mix the bird chirp and your voice together

Combine the chirp and recorded data together.

Notice that the chirp data '`y`' is a shorter variable than our `voice` data, therefore we can not directly add them together.



Duplicate the chirp signal first,

option 1: `chirp = [y ; y]`

Q3: Why do we use a *semicolon*?

option 2: `chirp = repmat(y,2,1);`

Then add same amount of elements together to create the combined signal.

```
combined = chirp(1:length(recorded))+recorded;
```

Q4: What if in Section 2 we recorded **4** seconds of audio signal instead of 3 seconds, how should we manipulate the `chirp` signal?

D. Play and display signal in T*ime Domain*

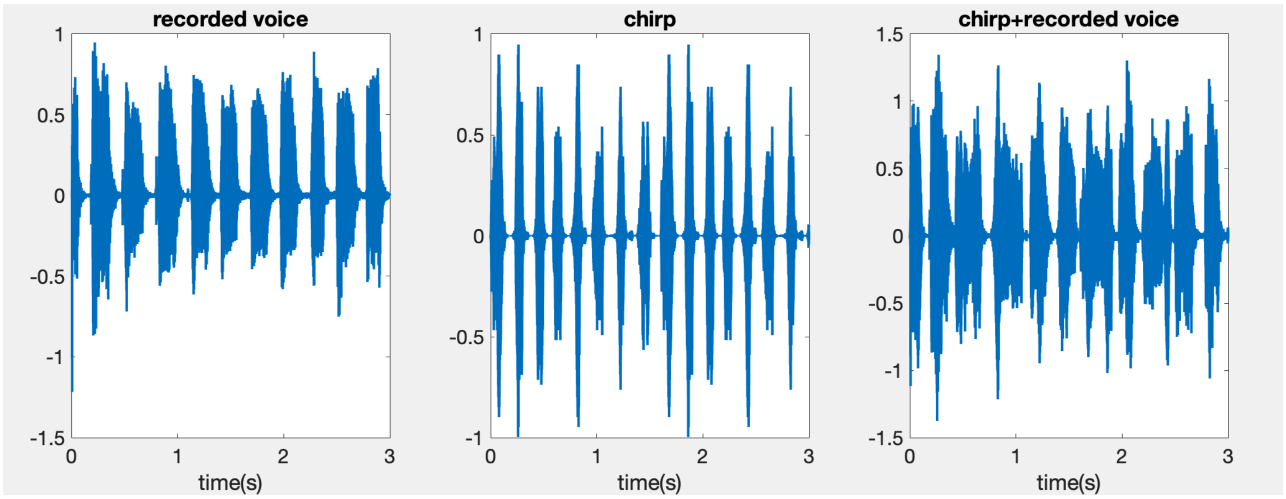
Play the recorded signal, the chirp signal and their combination. Plot all three signals in the time domain.

```
dt = 1/Fs; % second
L = length(voice); % size of voice
t = (0:L-1)*dt; % second

disp('Hit space to play voice sound:');
pause;
sound(voice,Fs);pause(Time);
disp('Hit space to play chirp sound:');
pause;
sound(chirp,Fs);pause(Time);
disp('Hit space to play combined sound:’);
pause;
sound(combined,Fs);pause(Time);

fh = figure(1);
set(fh, 'Position', [200, 500, 1200, 400]);
subplot(1,3,1)
plot(t,voice);
xlabel('time(s)');
title('recorded voice');
subplot(1,3,2)
plot(t,chirp(1:L));
xlabel('time(s)');
title('chirp');
subplot(1,3,3)
plot(t,combined);
xlabel('time(s)');
title('chirp+recorded voice');
```

In time domain, is there a way to separate each signal(the recorded and chirp signal) from the combined signal?

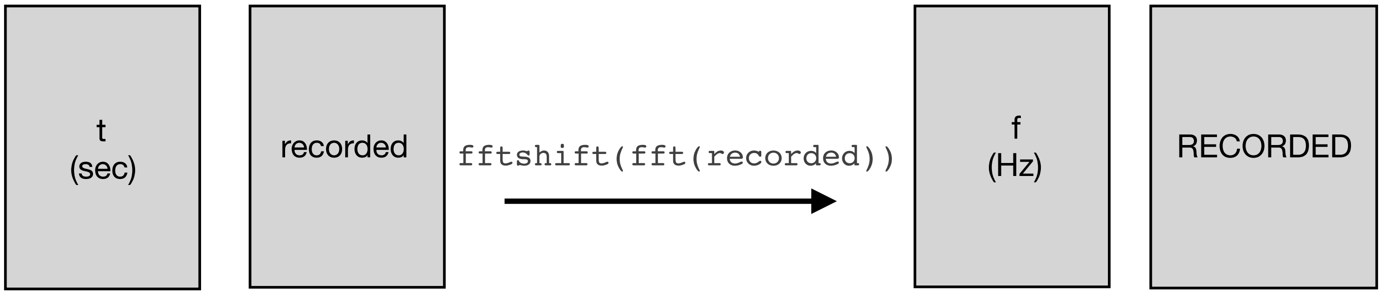


Obviously no. Because both signals are on top of each other. In other words, they’re overlapped with each other. That’s why we need to introduce the Fourier transform.

E. Fourier spectrum of each signal

Generally speaking, Fourier transform transfers a time domain signal into frequency domain, which allows you to tell what frequencies are present in your signal.

The Fourier transforms (Matlab 1D Fourier transform function ‘`fft`’) take a signal and express it in terms of the frequencies of the waves.



```
f = Fs*(-(L/2)+1:(L/2))/L; % Hz
Nyquist =Fs/2; %Hz

VOICE = fftshift(fft(voice));
CHIRP = fftshift(fft(chirp(1:L)));
COMBINED = fftshift(fft(combined));
peak = max(abs(COMBINED)+10);

fh =figure(2);
set(fh, 'Position', [200, 500, 1200, 400]);
subplot(1,3,1)
plot(f(end/2:end),abs(VOICE(end/2:end)));ylim([0,peak])
xlabel('frequency(Hz)');
ylabel('magnitude');
title('recorded Voice');
subplot(1,3,2)
plot(f(end/2:end),abs(CHIRP(end/2:end)));ylim([0,peak])
xlabel('frequency(Hz)');
ylabel('magnitude');
title('Chirp')
subplot(1,3,3)
plot(f(end/2:end),abs(COMBINED(end/2:end)));ylim([0,peak])
xlabel('frequency(Hz)');
ylabel('magnitude');
title('Combined signal'
```

Q5: What is Nyquist frequency? What does the function `fftshift` do?

We only plot positive side of the frequency spectrum. Look at the frequency spectrum of the signals and find out which one has higher frequency. If we want to separate the chirp and recorded data in frequency domain, what is the cutoff frequency we should use?

F. Design a low pass filter

Design a window-based lowpass filter to filter *out* the chirp sound.

```
fir1(Order, CutOff/Nyquist, 'low');
```

To use the MATLAB built-in window filter, we need at least 3 inputs, the 1) order of the filter, 2) cutoff ratio, and 3) filter type(*low pass, high pass* or *bandpass*).

The cutoff ratio can be calculated with equation

$$\text{Cutoff ratio} = \frac{\text{Cutoff frequency}}{\text{Nyquist frequency}} = \frac{\text{Cutoff frequency}}{\text{Sampling frequency}/2}$$

Q6: figure(3) display the magnitude the response of the designed filter; Use 10, 20, 50, 100, 500 as order number and describe the magnitude response of the filter. Think about what an ideal filter would look like.

G. Section 7 Design a high pass filter

Similarly to Section 6, now design a high pass filter. Design a window-based highpass filter (‘`bhi`’) to filter out the recorded sound.

Assignment:

You are given a MATLAB generated chord signal ‘`chord_sig.mat`’; The variable sig contains multiple pitches in it. However, they are all summed together in time domain.

- Appropriate format of the report with Introduction, method, result, discussion and conclusion. (1 pt)
- Use what you learned in lab to separate each pitch. Describe how you extract each note from the signal and justify your answer. Describe what order to you use and why. How long (in second) does each pitch last? (3 pt)2
- Create a small piece of audio song, for example ‘Twinkle, Twinkle, Little Star’, ‘Mary had a little lamb’, ‘The wheels on the bus’, etc. After creating the song, use MATLAB function ‘`audiowrite`’ to export the song. (eg. `audiowrite(‘TwinkleStar.wav’,song,Fs)`) Briefly describe how you generate the song. Upload the song to Compass as well as the script. (2 pt)
- Answer all the in-lab questions. Please clearly indicate the location of your answers. (3 pt)
- Clear figures/tables with proper caption. (1 pt)

For your reference, here is a table of the frequencies for several notes.

Note		Frequency (Hz)
C_3		131
D_3		147
E_3		165
F_3		175
G_3		196
A_3		220
B_3		247
C_4 (middle C)	do	262
D_4	re	294
E_4	mi	330
F_4	fa	349
G_4	so	392
A_4	la	440
B_4	ti	494
C_5		523
D_5		587
E_5		659
F_5		698
G_5		784
A_5		880
B_5		988
C_6 (high C)		1047