LAB#03

A huge thanks to Ms. Mehri for her kind support thus far ☺

Shayan Vassef, UIN = 674579894, $N_{id} = 894$

1 Violin sound synthesis applying impulse train excitation to a finite-duration impulse response (FIR) filter

1.1 Creating a single note

Q1.

The sound of $vA_3$is not heard monotonously, instead, we can recognize the rise and fall at the beginning and the end of the voice. On the other hand, yA_3 is heard monotonously from the first to the end.

1.2 Creating a tune

Q2.

To create the given tune, I created a list of pitch frequencies and durations corresponding to the desired tune's existing notes. The pitch frequencies were obtained using the given table at the first page of the instructions. The durations are set wisely after listening to the song of the desired tune. For words like “doe”, and “a”, I considered a duration of 0.5; For the words “deer”, and “fe” in “female”, durations are set as 0.75 and 1. The only rest times were applied between “Doe” and “a”, and between “Deer” and “a”. As a procedure, for every single tone, the signal vA_3 is obtained using its corresponding duration, and pitch frequency. Finally, padding each tone with zeros as a rest time duration using the “rest durations” list, where index ith is associated with the rest time between $\boldsymbol{ith}$ and $\boldsymbol{ith+1}$ tone.

```
% Load the FIR filter impulse response
load('irviolin.mat');
a = 1; % denominator for FIR filter
b = impresp_violin;
% Sampling frequency
Fs = 44100;

% Define note frequencies and durations (in seconds)
note_freqs = [261.61, 293.66, 329.63, 261.61, 329.63, 261.61, 329.63]; % C4, D4, E4
note_durations = [0.5, 0.5, 0.75, 0.5, 1, 0.5, 0.5]; % durations of each note
rest_durations = [0.5,0,0.3,0,0,0];
% Initialize the tune as an empty array
tune = [];

% Create each note and concatenate to form the tune
for i = 1:length(note_freqs)
    Fpd = note_freqs(i); % Frequency of the current note
    Tdur = note_durations(i); % Duration of the current note in seconds

    % Rest (silence) between notes
    if i > 1
        tune = [tune, zeros(1, round(rest_durations(i-1) * Fs))];
    end
```

```
    % Generate the note using the previous code
    Ndur = round(Tdur*Fs); % # of samples in Tdur seconds
    n=[0:Ndur-1]; % index set [0 1 2 3 ... Ndur-1]
    scalefn = sin(pi*n/(Ndur));
    scalefn = scalefn.*scalefn; % modulating function for the note
    Np = round(Fs/Fpd); % # samples in pitch period, rounded.
    % Np is used in pulse train generation
    tem = n/Np - round(n/Np);
    % entry in tem is zero when corresponding entry in n is a multiple of Np
    pt = (tem == 0); % creates pulse train
    yA3 = filter(b,a,pt);% generates signal y_{A_3} for note A3 of duration Ndur s
    vA3 = yA3.*scalefn; % generates shaped signal v_{A_3}
    % ... (similar to the previous code)

    % Concatenate the note to the tune
    tune = [tune, vA3]; % Replace vA3 with the current note

end

% Play the tune
soundsc(tune, Fs);

% Save the tune as an audio file
audiowrite('tune_FIR.wav', tune, Fs);
```

2 Violin sound synthesis applying impulse train excitation to an infinite-duration impulse response (IIR) filter

2.1 Creating a single note

Q3.

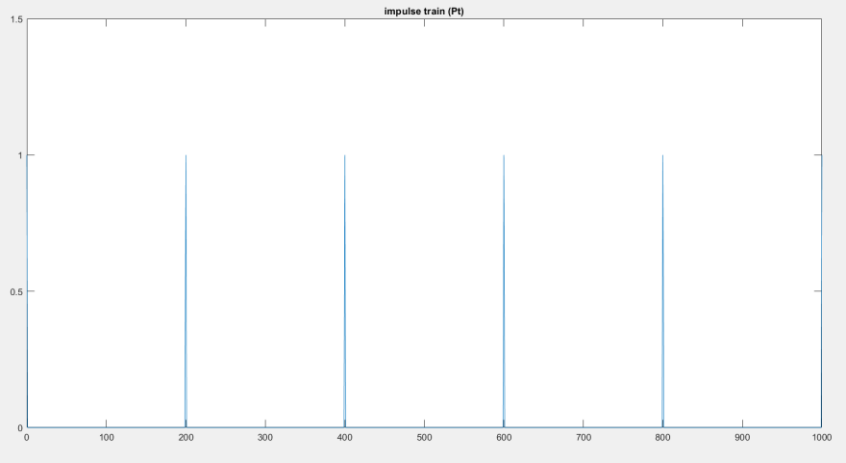

Fig 1. Impulse train with 200 samples in pitch period

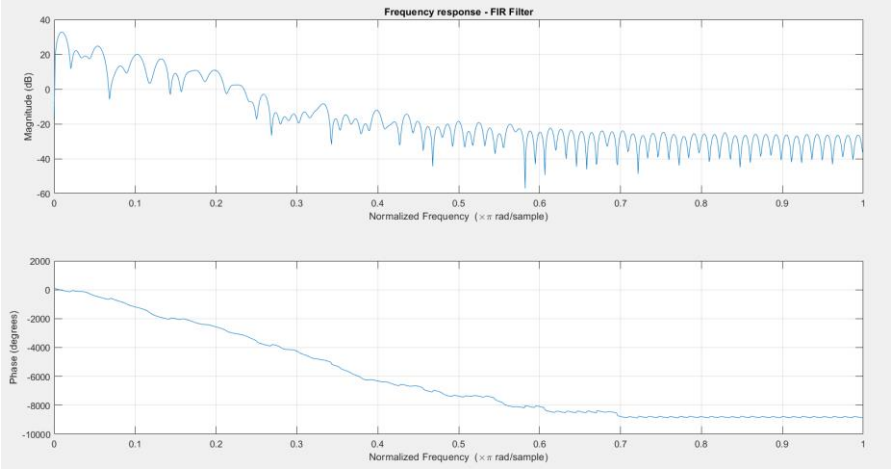

Fig 2. Frequency response of FIR filter

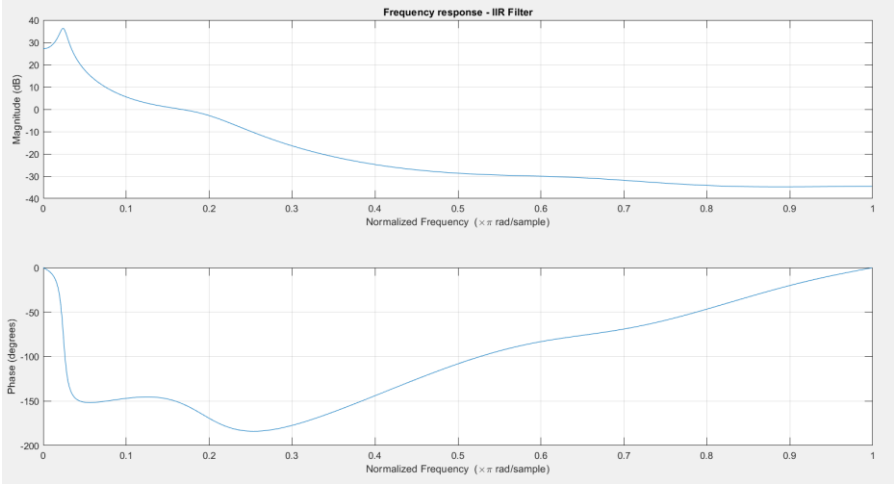

Fig 3. Frequency response of IIR filter

As can be seen in the two above figures, the phase of the IIR filter is purely non-linear, causing this to be the main difference with the FIR filter. Besides, there are too many ups and downs in the magnitude response of the FIR filter whereas the magnitude response of the IIR filter is much smoother, meaning that the same functionality in the FIR filter can be reached by fewer parameters in the IIR filter. These frequent ups and downs affect the filtered signal, yA_3, where there are too many high-frequency components in the resulting signal of the FIR filter. On the other hand, there are fewer low-frequency components in the resulting signal of the IIR filter. As we know, lower frequencies tend to appear in "bass tone", where the generated sound is heard "lower" to the human ear. By listening to the generated tones of the IIR and FIR filter, we can confirm that the filtered signal by the FIR filter sounds "higher" than the filtered signal by the IIR filter.

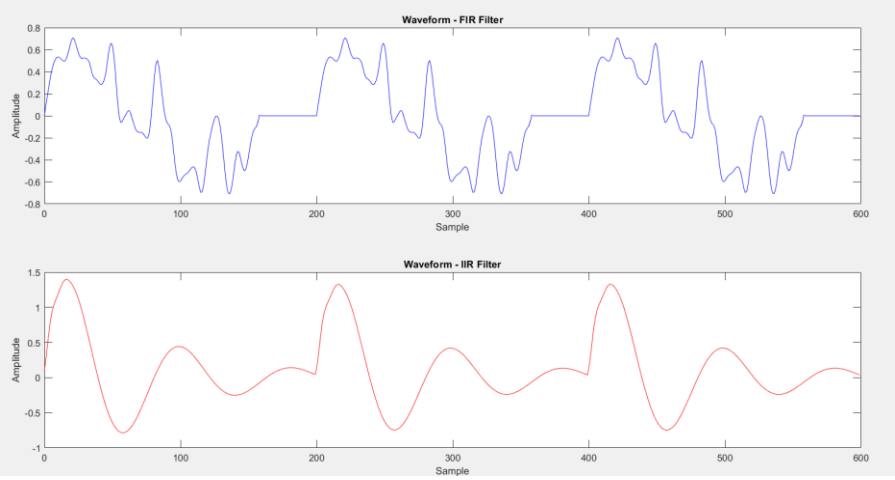

Fig 4. Difference in waveform generated by the FIR and IIR filters

2.2 Creating a tune

Q4.

The same procedure for Q2. was applied, however, the filter using in this section is an IIR filter instead of FIR filter.

Q5.

As the duration and rest times for both of the Q2. and Q5. happened to be the same, the single tone comparison in Q3. can be generalized to this question as well. For each tone in the desired tune, the tone generated with the FIR filter sounds “louder”/” higher” as opposed to the tone generated by the IIR filter. Thus, in general, the generated tune by the IIR filter sounds “lower”, but both tunes are heard clearly.

**** All the generated tones/tunes are attached to the .zip file.**