Laboratory 3 – Fun with Filters

Objective

This lab explores filtering of sound files. Specifically, we will create an echo of a spoken word sound file and an equalizer for a music file.

This lab will provide useful exercises for using filterDesigner, signalAnalyzer, and Matlab functions to apply filters.

Instructions

Part A – Echo

Read a spoken word sound file into the Matlab workspace using the command:

```
[y,Fs] = audioread('file_name');
```

You can find or create your own spoken file, or use the one provided: `havard.wav`.

Create a FIR filter that delays the input by some number of samples without modifying the frequency content. (Note that if your numerator coefficients are in a vector `b`, you will want to use the filter command: `yfilt = filter(b,1,y);` rather than `filtfilt`, since `filtfilt` would remove the time delay.

To create an echo, experiment with trying delays that correspond to 50 msec, 100 msec and 200 msec. Add your delayed file to the original file and renormalize the maximum amplitude to 1. Play your files and listen to what the echo sounds like.

Experiment with some different types of echos. For example, create multiple delayed files, each weighted by an exponentially decreasing factor, and add them together to create a "bouncing back-and-forth" echo. You can adjust the delay factor and amplitude decay to achieve the echo effect you want.

Example of delay filter: If the sample rate is 48000 Hz, and you want to create a 0.100 msec delay filter, you would create a FIR filter of length 4801, with 4800 zeros and 1 as the last element. This would correspond to a filter: $H(z) = z^{-4800}$.

Part B – Equalizer

Read a music sound file into the Matlab workspace using the command:

```
[y,Fs] = audioread('file_name');
```

You can find your own music file or use the one provided: `BeeMoved.flac`

Create a series of bandpass filters, for example 0-3 kHz, 3-6 kHz, 6-9 kHz, 9-12 kHz, 12-15 kHz, 15-18 kHz, 18-21 kHz, 21-24 kHz. Filter your original signal, saving each filtered band in a separate array. For example, the array for the band from 15 to 18 kHz could be created with the command:

```
y_15_18 = filtfilt(b_15_18,a_15_18,y);
```

where `a_15_18` and `b_15_18` are the filter coefficients.

I suggest using `filterDesigner` and exporting the filter coefficients to the Matlab workspace. Or, you can use Matlab commands for creating the filters. Experiment with various kinds of FIR and IIR filters.

Listen to each sound file you create to see what they sound like.
For example use: `sound(y_15_18,Fs)` to listen to the 15 to 18 kHz band.

Recombine the sound files weighting different frequency bands.

```
ysum = weight_0_3*y_0_3 + weight_3_6*y_3_6 + …
```

You will need to normalize the sum so the largest amplitude is 1, i.e.

```
ysum(:,1) = ysum(:,1)/max(ysum(:,1));
ysum(:,2) = ysum(:,2)/max(ysum(:,2));
```

If you have a stereo file (2 channels).

Experiment with different equalization weights.

You could write more concise Matlab code using cell arrays if you wish. For example

```
Bands = [0,3,6,9,12,15,18,21,24];
% Get coefficients for filters and store them in cell arrays
b{1}, b{2}, …, etc. for each band
% b{1} contains the numerator filter values for band 0 – 3 kHz.
ysum = zeros(size(y));
For k = 1:length(bands)-1
     yfilt{k} = filtfilt(b{k},a{k},y)
     ysum = ysum + weight(k)*yfilt{k};
end
ysum = ysum/max(ysum);
```

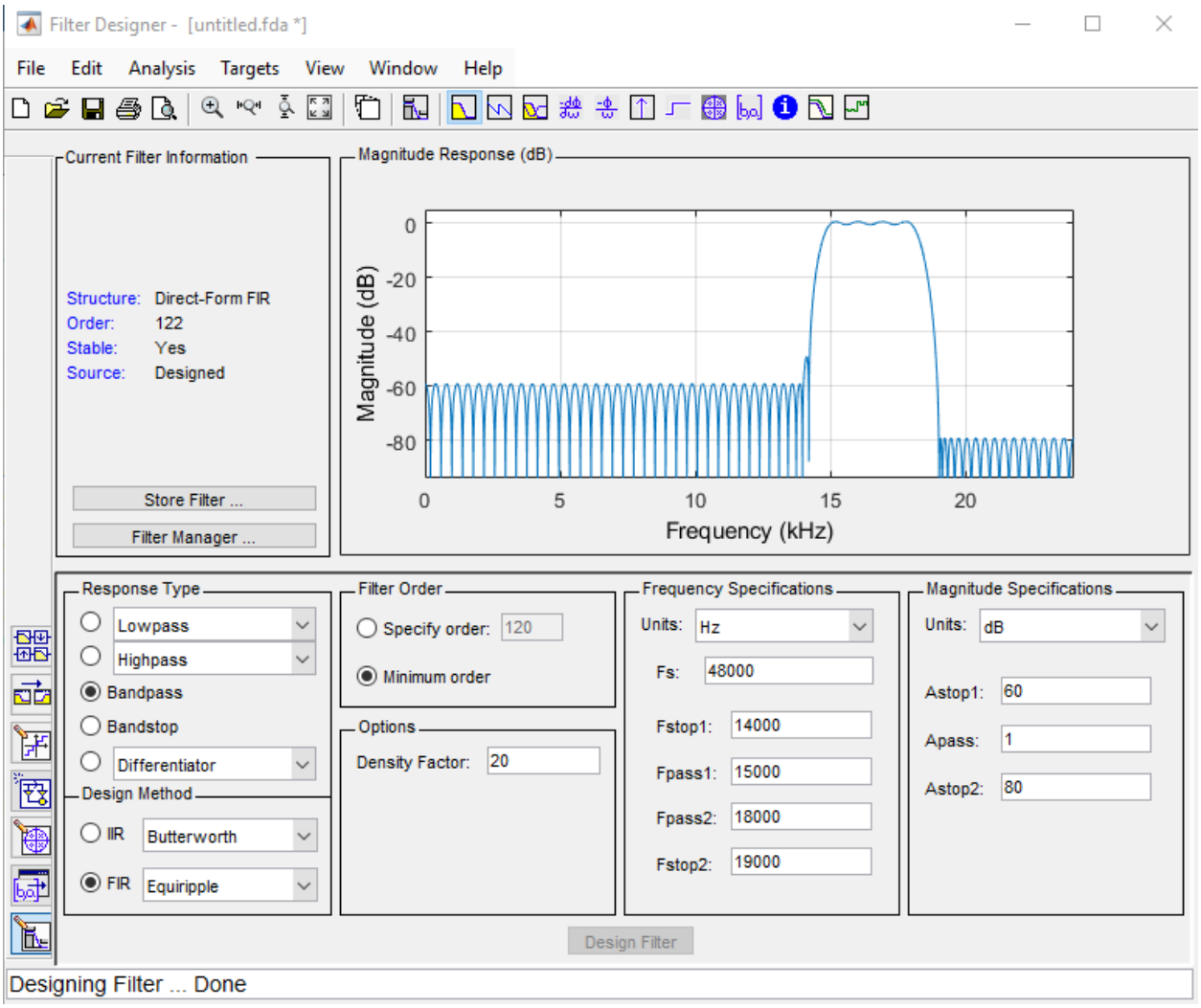
(This code isn't guarantee to work as is. Modify it as necessary.)

If you create your filters in filterDesigner, you can export your filter code. For example, to create a 15 to 18 kHz filter, the filterDesigner window is shown on the following page.
The parameters for the filter are:
Fs = 48000 Hz
FIR equiripple filter
Edge of lower stop-band: 14000 Hz
Edge of lower pass-band: 15000 Hz
Edge of upper pass-band: 18000 Hz
Edge of upper stop-band: 19000 Hz
Attenuation in lower stop-band: 60 dB
Attenuation in upper stop-band: 80 dB

This created a 122 order FIR filter.
The code created is:

```
function Hd = bp_15_18
%BP_15_18 Returns a discrete-time filter object.
% MATLAB Code
% Generated by MATLAB(R) 24.2 and Signal Processing Toolbox 24.2.
% Generated on: 14-Apr-2025 15:46:47
% Equiripple Bandpass filter designed using the FIRPM function.
% All frequency values are in Hz.
Fs = 48000;  % Sampling Frequency
Fstop1 = 14000;           % First Stopband Frequency
Fpass1 = 15000;           % First Passband Frequency
Fpass2 = 18000;           % Second Passband Frequency
Fstop2 = 19000;           % Second Stopband Frequency
Dstop1 = 0.001;           % First Stopband Attenuation
Dpass  = 0.057501127785;  % Passband Ripple
Dstop2 = 0.0001;          % Second Stopband Attenuation
dens   = 20;              % Density Factor
% Calculate the order from the parameters using FIRPMORD.
[N, Fo, Ao, W] = firpmord([Fstop1 Fpass1 Fpass2 Fstop2]/(Fs/2), [0 1 ...
                          0], [Dstop1 Dpass Dstop2]);
% Calculate the coefficients using the FIRPM function.
b  = firpm(N, Fo, Ao, W, {dens});
Hd = dfilt.dffir(b);
```

Read the Matlab documentation to understand what functions and parameters are used in this code.



What to Report

Much of what you report will be your qualitative descriptions of what you did and what you hear. You should also include a description of the filters you use, and explanation of how you constructed your echo or equalized outputs. For the echo files, you could include zoomed in regions of the original and delays signals that would show the delay. For the equalizer part, describe the filters and parameters used and plots of the amplitude and phase response some filters. (You don't need to include each band's filters.) Also show some spectra of the original signal and different bands