Lab 3: Fun with Filters

Kimaru Boruett

May 4 2025

PART A - Echo

Time Delay

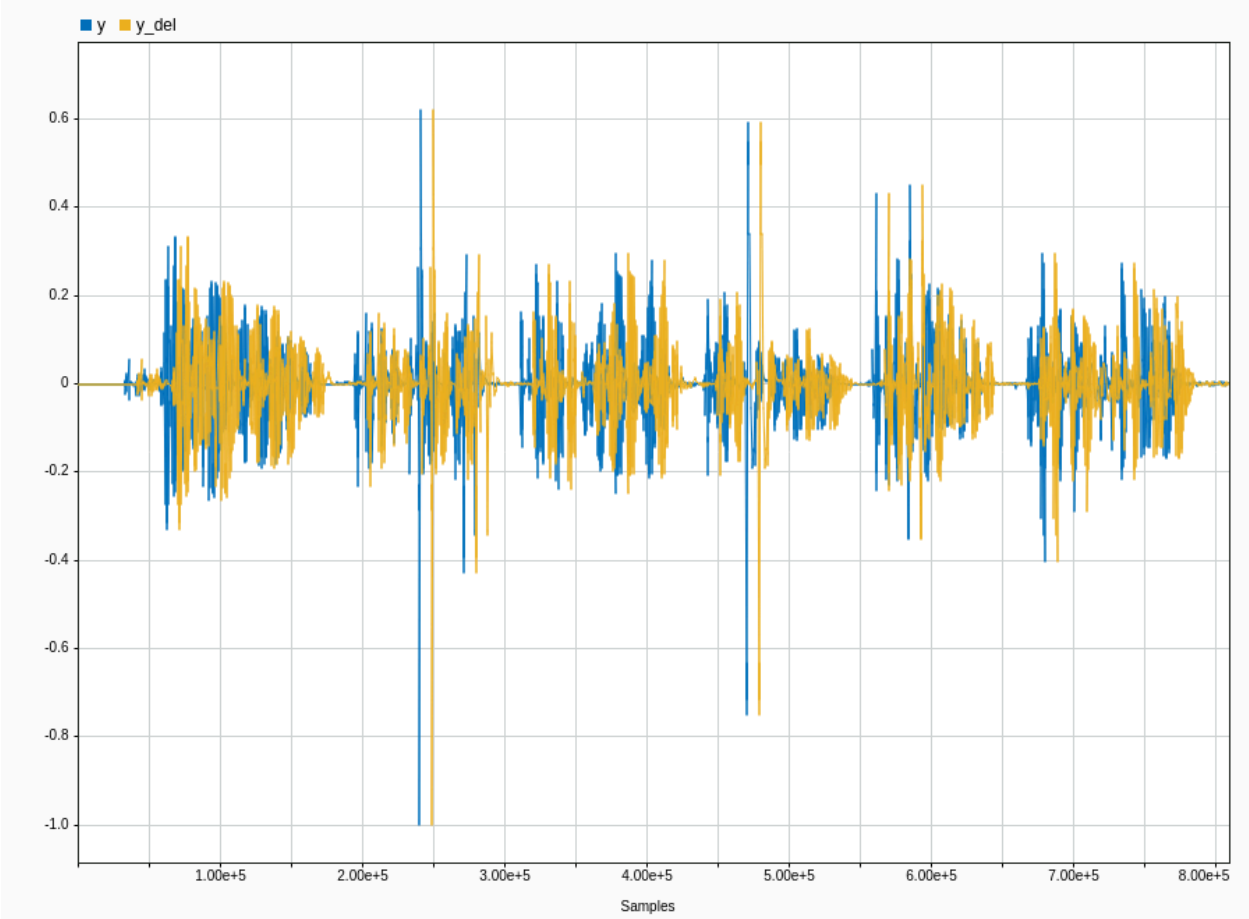


Figure 1. Time Delay Signal with Original

The blue trace is the unprocessed speech waveform, while the yellow trace (y_del) is a time-shifted copy created with the FIR [zeros(1,D) 1]. Throughout the record you can see the yellow signal starting D samples (~200 ms in this plot) after every blue feature. Because the filter is an ideal delay, the spectral content and amplitude envelope are identical; the only visible distinction is the horizontal offset. This plot makes the concept of group delay intuitive—every point is translated rightward by precisely the same amount.

Normal Echo

This code creates three distinct echoes at 50 ms, 100 ms, and 200 ms delays. For each delay, a FIR filter is designed with a numerator b_delay containing zeros followed by a 1 at the delayed sample position (e.g., D = round(delay_ms(k)*Fs/1000). The delayed signals are scaled by a fixed decay factor (0.6) and added cumulatively to the original. You will hear clear, discrete repetitions of the original audio at the specified intervals, each at reduced volume (60% of the original), creating a "hall-like" effect with distinct, non-overlapping echoes.

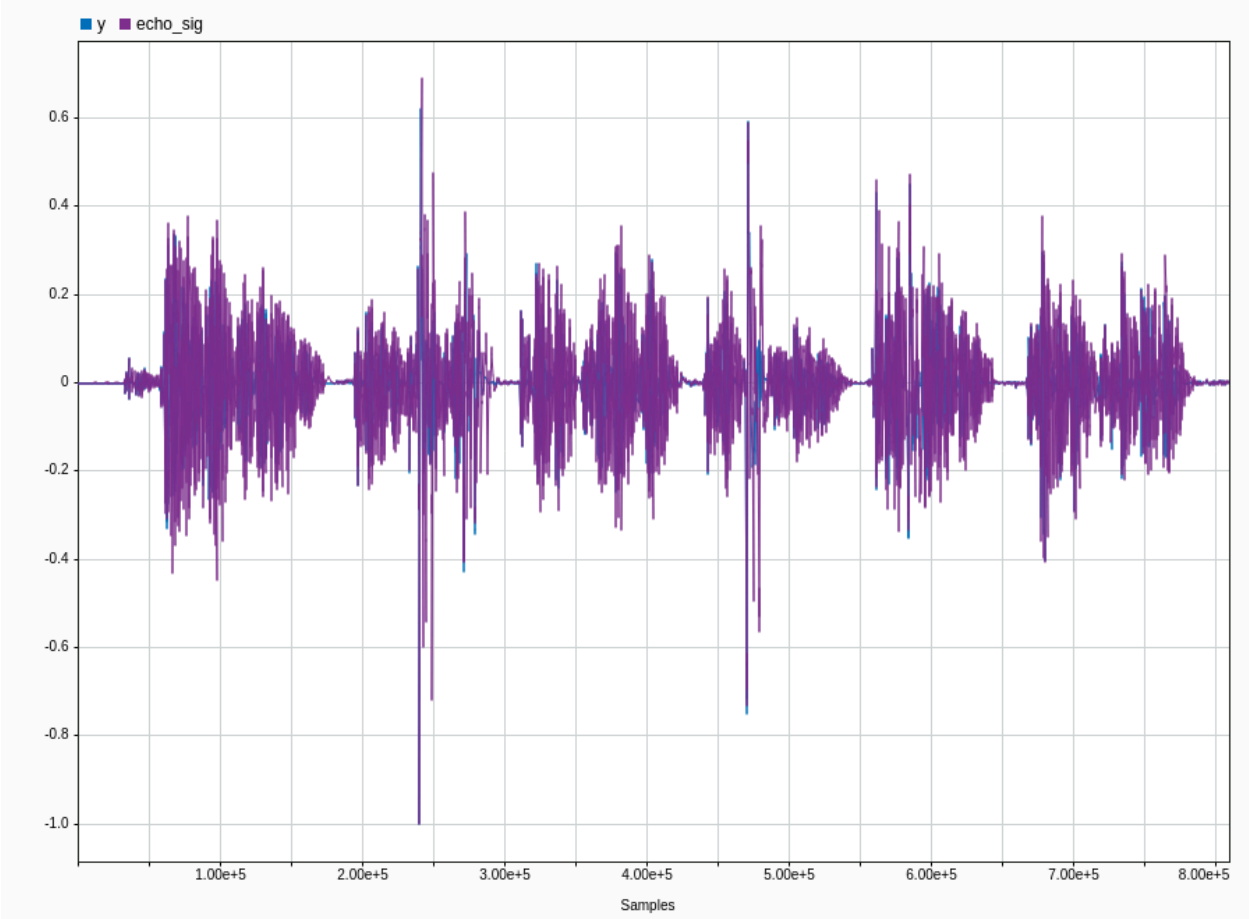


Figure 2. Echo Signal with Original

Here the purple trace (echo_sig) is the sum of the original plus three delayed replicas at 50 ms, 100 ms, and 200 ms, each scaled by 0.6. The waveform still hugs the outline of the original (blue), but transient peaks are “thicker” and low-energy regions never quite return to zero. That fill-in reflects overlapping echoes: earlier speech energy is still present when the next syllable arrives.

Bouncing Echo

This effect uses a single FIR filter with exponentially decaying echoes spaced at 120 ms intervals. The filter numerator b is constructed by appending delayed copies of the signal, each weighted by alpha^n (where alpha = 0.65 and n increases for each echo). This results in 8 progressively quieter echoes at 120 ms, 240 ms, ..., up to 960 ms. The decay factor ensures each subsequent echo is softer, mimicking a sound "bouncing" and losing energy. You will hear a smooth, decaying trail of echoes, resembling a reverberation in a long corridor.

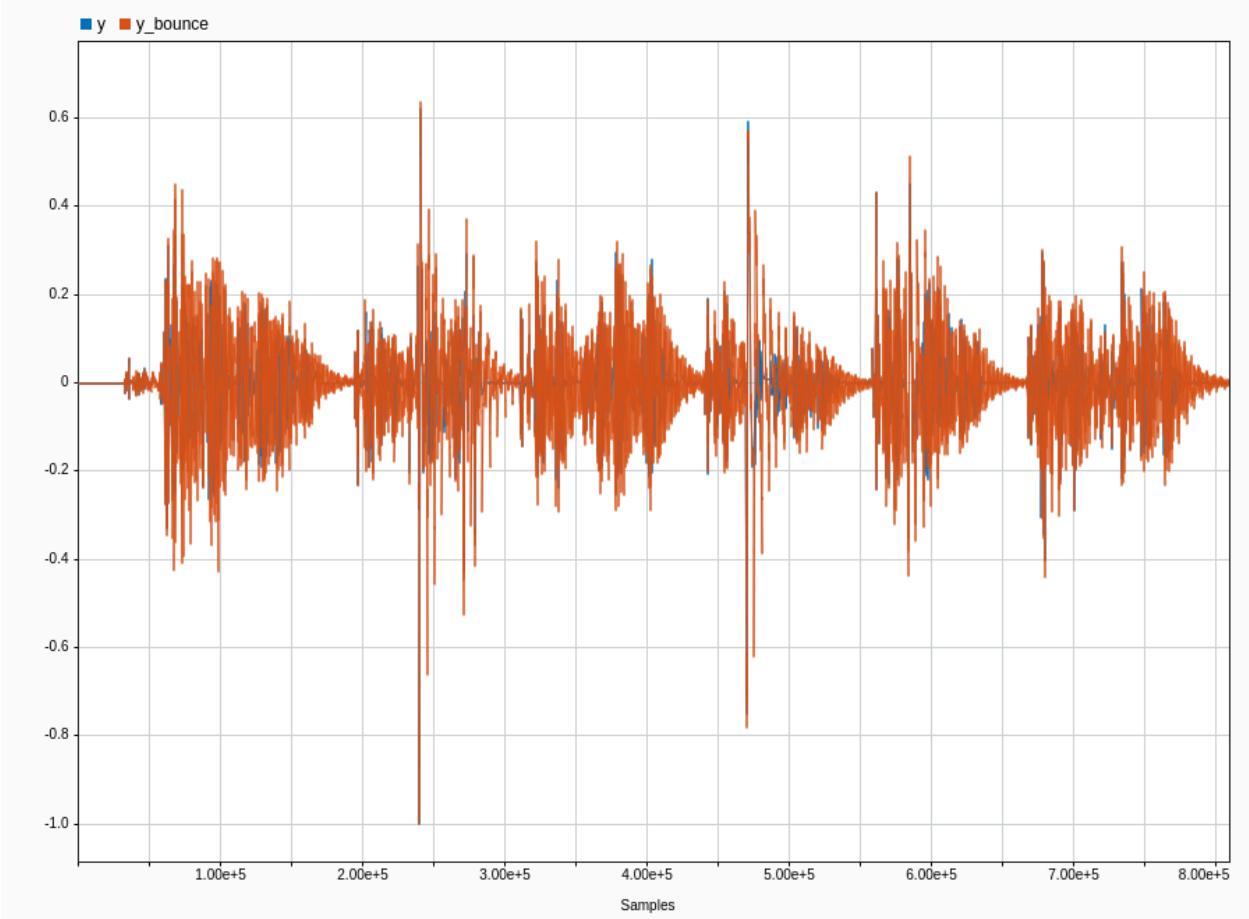


Figure 3. Bouncing Echo with Original Signal

The orange trace (y_bounce) overlays eight echoes spaced 120 ms apart with a geometric decay α = 0.65. Compared with the normal echo, you now see a trailing tail after each phrase: the first echo is prominent, then successive echoes diminish but are still visible as oscillations that taper toward zero. The sustained ringing is most obvious after large plosives (e.g., sample 4.8 × 10^5) where the orange envelope lingers well beyond the blue. This demonstrates how a simple FIR with regularly spaced taps can emulate a small room's multiple early reflections.

Tempo-Synced Echo

Synced to 120 BPM, this code generates echoes at 0.5 s, 1 s, and 1.5 s intervals (aligned with musical beats). Each echo is created using a FIR filter with a delay matching the beat interval (delay_samples = round(Fs * beat_interval * [1, 2, 3]) and scaled to 50% amplitude. The delayed signals are summed with the original. You will hear rhythmic echoes that align with the tempo, creating a "musical" delay effect where echoes reinforce the beat structure, ideal for synchronizing with percussive elements.

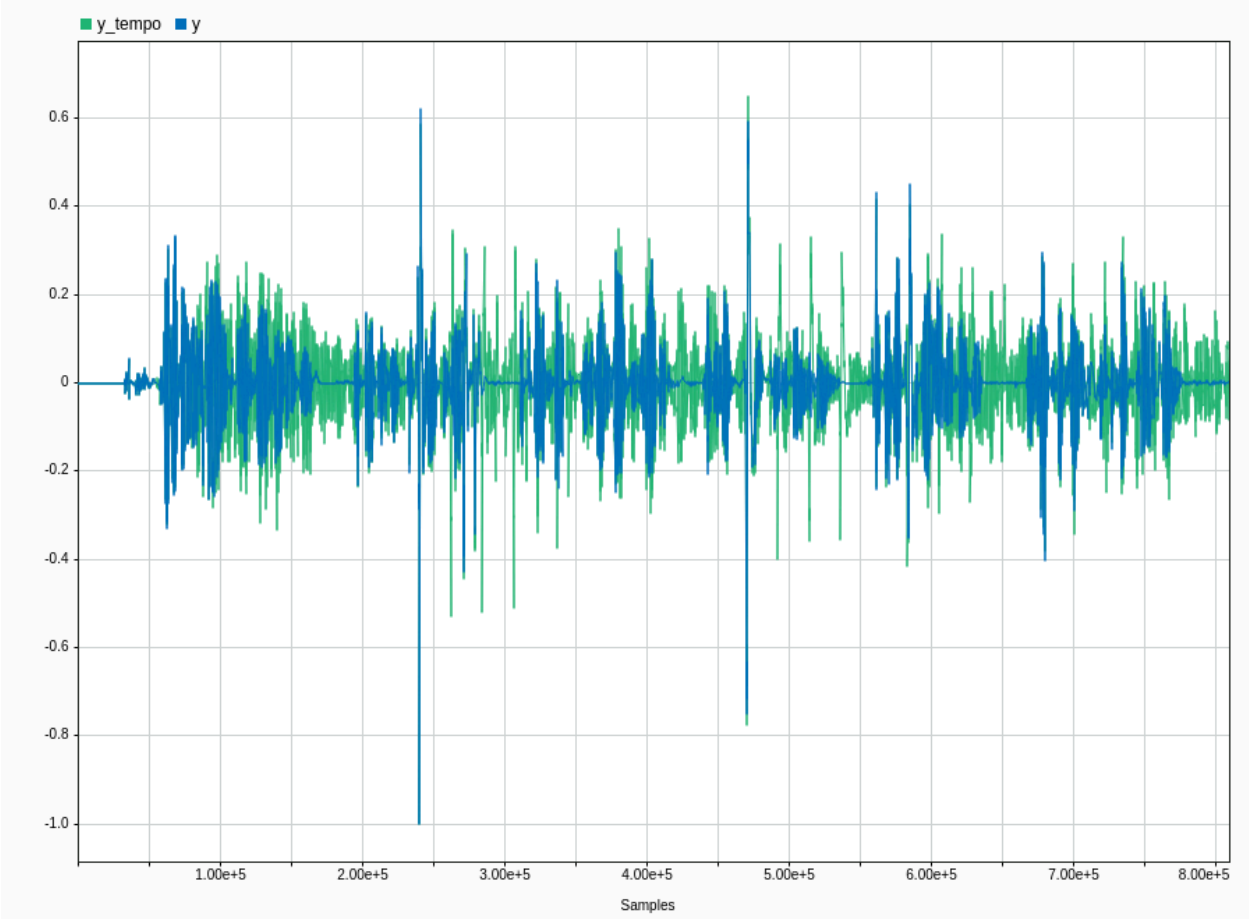


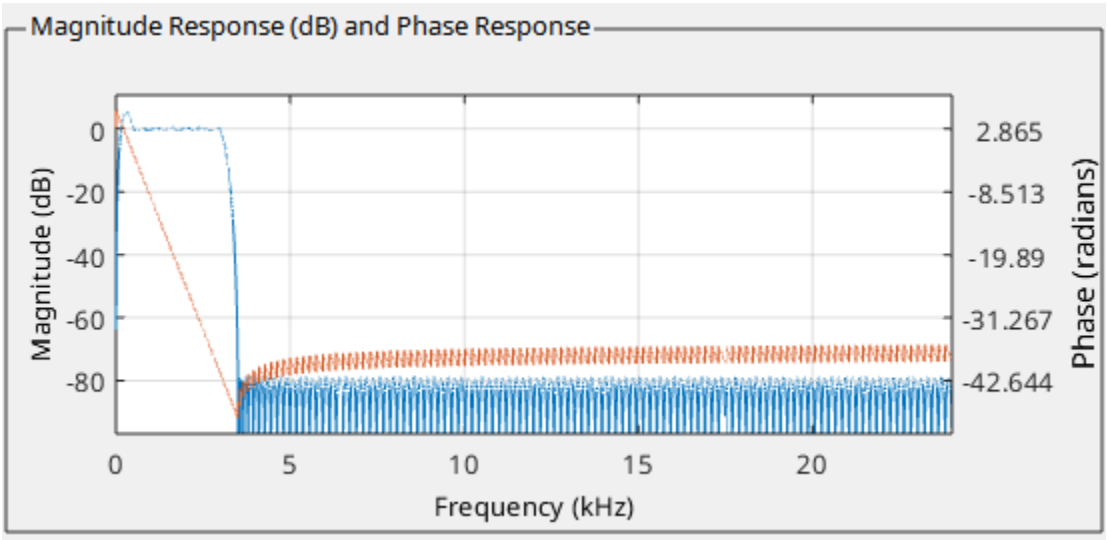
Figure 4. Tempo-Synced Echo with Original

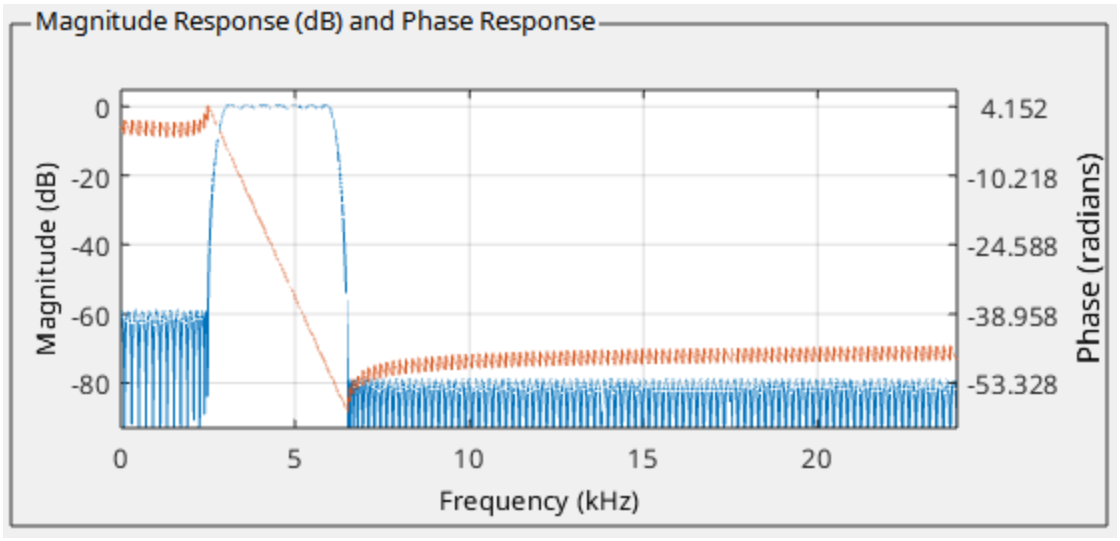
In the final plot the green trace (y_tempo) adds echoes aligned to musical beats at 120 BPM (500 ms, 1 s, 1.5 s). The resulting waveform shows periodic “ghosts” exactly every 0.5 s: peaks repeat at fixed rhythmic intervals rather than the dense smear of the previous echoes. Speech gaps are partially filled, but large silences still appear because the beat-aligned copies fall on discrete grid points. This style of delay is common in music production, tempo-sync preserves rhythmic structure while thickening the signal.

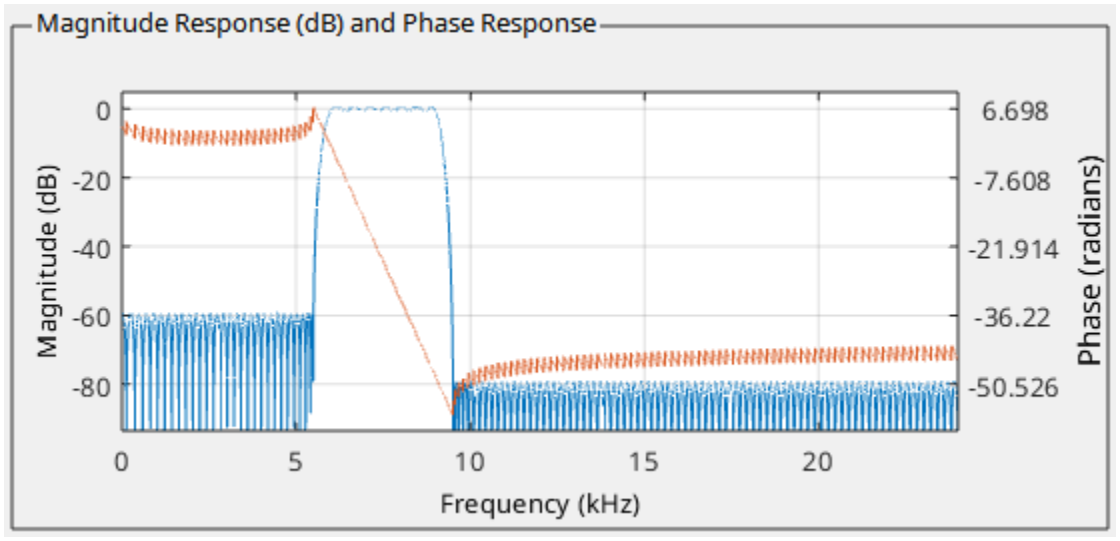
PART B - Equalizer

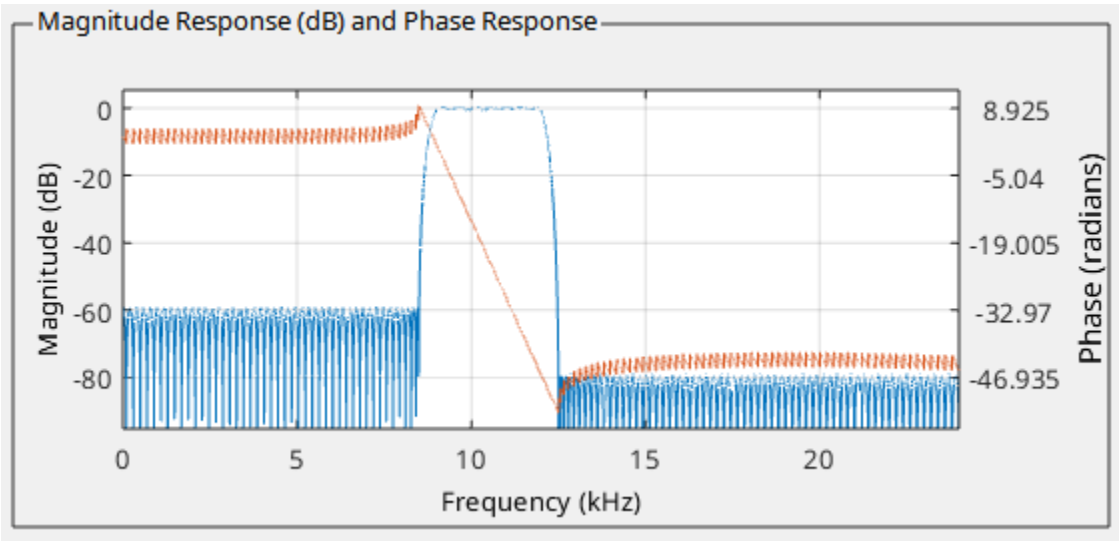
FIR Bandpass Filter Parameters Table			
Band #	Pass-band (kHz)	Lower stop-band (kHz)	Upper stop-band (kHz)
1	0 – 3	0.5	3.5
2	3 – 6	2.5	6.5
3	6 – 9	5.5	9.5
4	9 – 12	8.5	12.5
5	12 – 15	11.5	15.5
6	15 – 18	14	18.5
7	18 – 21	17.5	21.5
8	21 – 24	20.5	24

All filters were designed as FIR Equiripple Responses. This choice guarantees linear phase which is crucial for preserving stereo imaging when the bands are recombined.




Magnitude Response (dB) and Phase Response
Magnitude (dB)
0
-20
-40
-60
-80
Phase (radians)
4.152
-10.218
-24.588
-38.958
-53.328
0
5
10
15
20
Frequency (kHz)


Magnitude Response (dB) and Phase Response
Magnitude (dB)
0
-20
-40
-60
-80
Phase (radians)
6.698
-7.608
-21.914
-36.22
-50.526
0
5
10
15
20
Frequency (kHz)


Magnitude Response (dB) and Phase Response
Magnitude (dB)
0
-20
-40
-60
-80
Phase (radians)
8.925
-5.04
-19.005
-32.97
-46.935
0
5
10
15
20
Frequency (kHz)

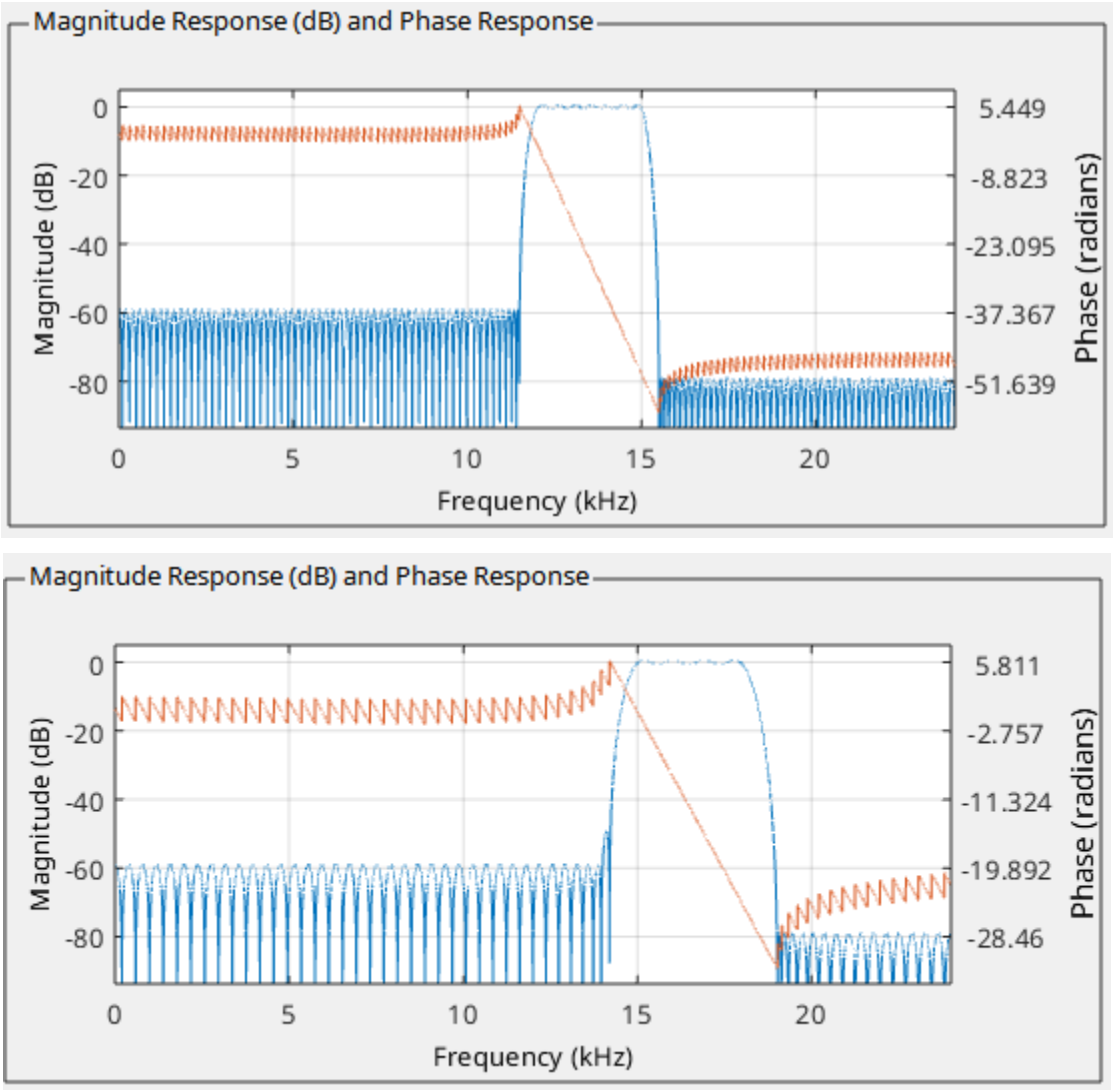


Figure 5. Magnitude and Phase Responses for FIR Bandpass Filters

When I turned the 0–3 kHz band all the way down, the song lost its body. The singer's voice and the bass almost disappeared, and the track sounded thin and hollow. Raising that band to about 1.4 brought the voice and bass back, making the mix feel warm and full again. Boosting the 3–6 kHz band made consonants like "t" and "k" pop out so the lyrics were clearer, but too much boost gave the vocal a nasal, sharp tone. Lowering it softened those edges yet made the words a little harder to catch. A small lift in the 6–9 kHz band added sparkle to guitars and gave the snare more snap; cutting this band dulled the song and took away some clarity. The 9–12 kHz band needed care: turning it up pushed "s" and "sh" sounds forward and made cymbals hissy, while turning it down smoothed things out. Keeping the 12–15 kHz and 15–18 kHz bands just below normal (around 0.8 and 0.7) kept hi-hats present without being harsh. Finally, a slight boost in the top 18–24 kHz range (about 1.2) added a light "air" to the mix, reverb tails and cymbal shimmer felt a bit more open.

The weights that felt best were:

Band (kHz)	0-3	3-6	6-9	9-12	12-15	15-18	18-21	21-24
Weight	1.4	1.2	1.0	0.9	0.8	0.7	0.8	1.2

With this curve, the vocals are clear and warm, the low end is solid, the highs are bright but not piercing, and the mix has a gentle, airy finish.

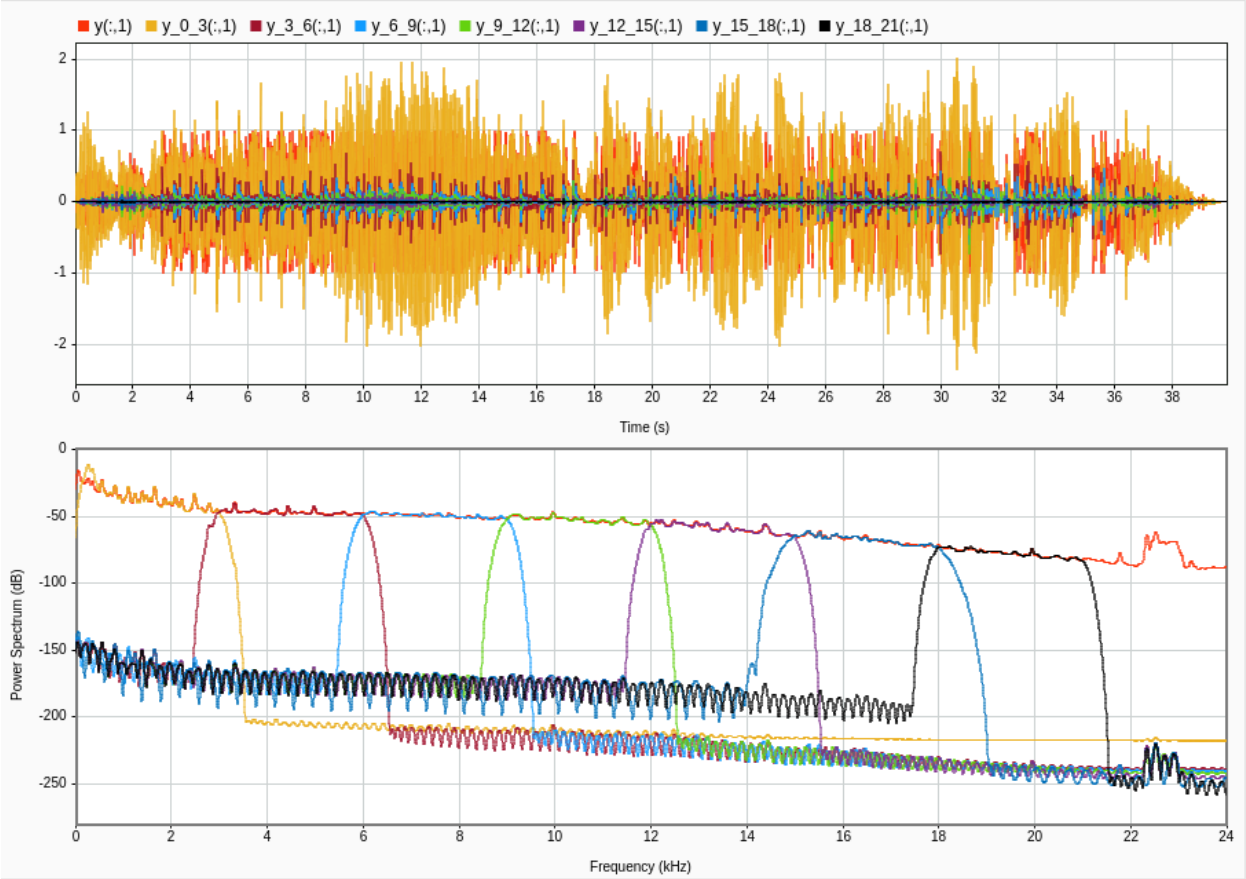


Figure 6. Frequency and Time Domain Plots of the Original and Filtered Signals

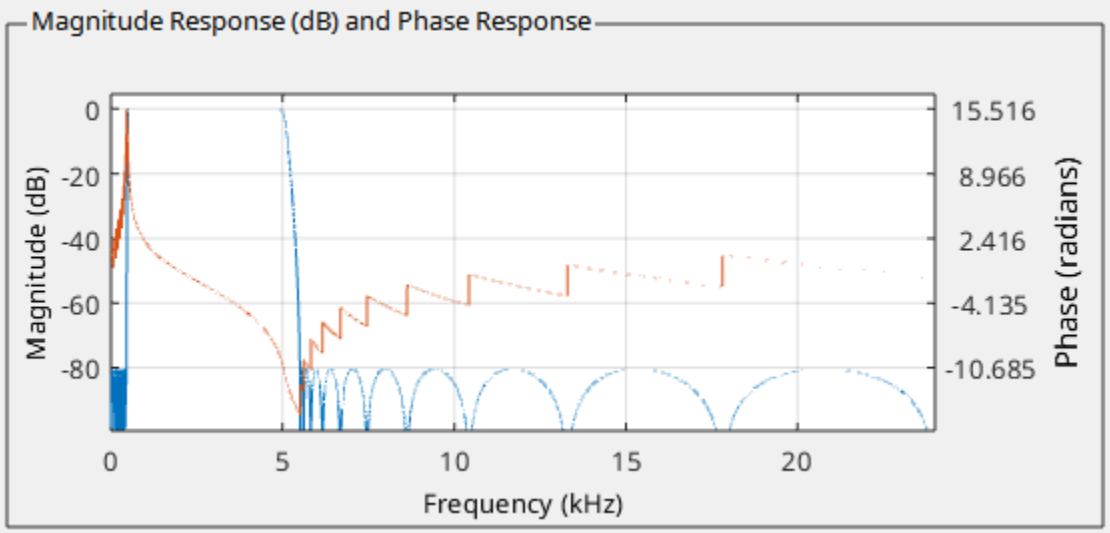
Figure 6 has two panels. The top panel shows the time waveform of the original music track (the solid orange trace) together with the eight band-pass copies produced by the equaliser filters. Because each coloured trace keeps only a narrow slice of the spectrum, their amplitudes are much smaller and cluster around the centre line, while the full-band signal spreads widely above and below zero. Visually this proves that the filters have separated the mix into eight versions that add up to the original when re-combined.

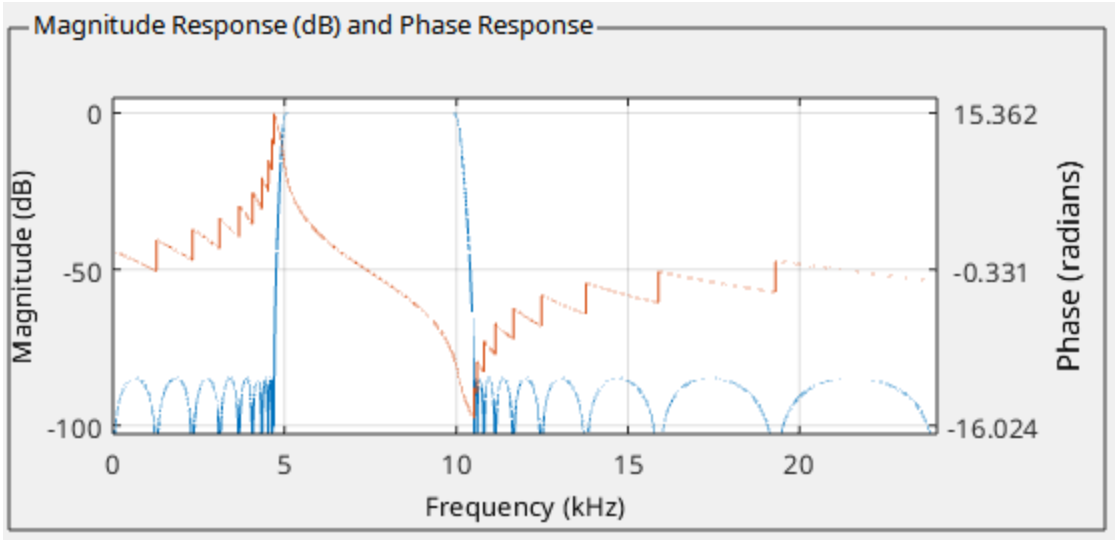
The bottom panel plots the power spectrum of every signal on a 0-to-24 kHz axis. Each coloured curve forms a tidy “window” that is flat (≈ 0 dB) inside its pass-band and quickly drops 60 dB or more outside it, exactly what we expect from the FIR equiripple design. The windows line up edge-to-edge: 0–3 kHz in red, 3–6 kHz in blue, 6–9 kHz in green, and so on up to 18–21 kHz in

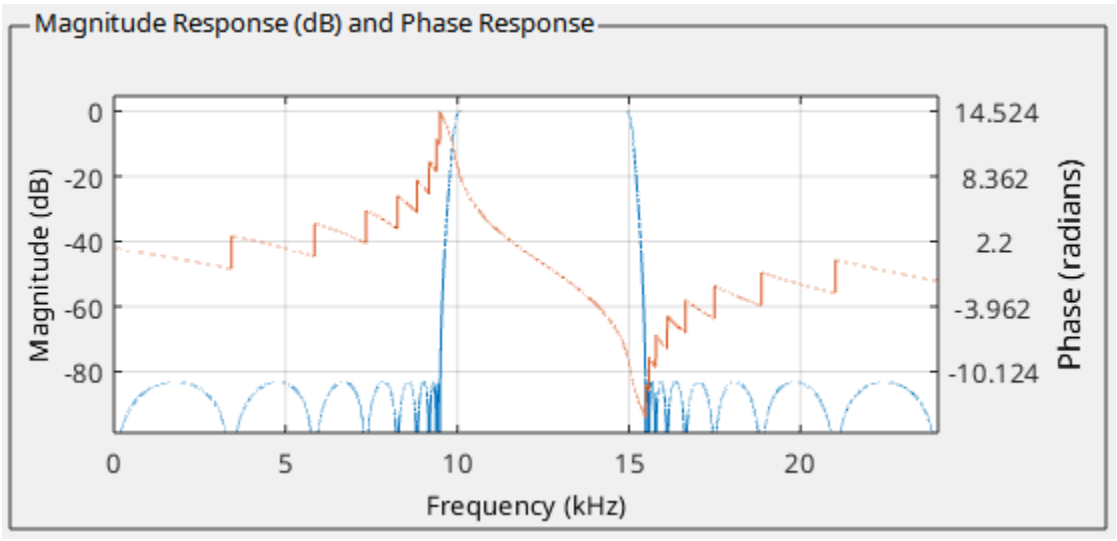
black. The orange curve spans the whole range, confirming it is the unfiltered track. Together the two plots demonstrate that the filters isolate each frequency band with steep skirts and little overlap, yet their sum faithfully reproduces the energy distribution of the original audio.

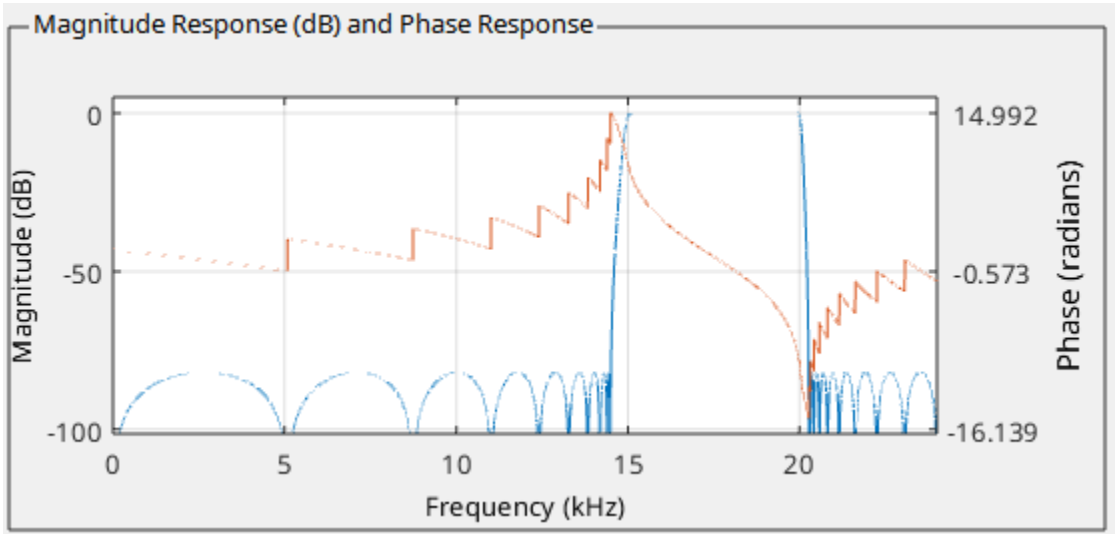
IIR Bandpass Filter Parameters Table			
Band #	Pass-band (kHz)	Lower stop-band (kHz)	Upper stop-band (kHz)
1	0 – 5	0.1	5.5
2	5 – 10	4.5	10.5
3	10 – 15	9.5	15.5
4	15 – 20	8.5	12.5
5	20 – 24	11.5	15.5

All the filters were designed as IIR Chebyshev Type II bandpass responses using the parameters in the table above.




Magnitude Response (dB) and Phase Response
Magnitude (dB)
Phase (radians)
Frequency (kHz)
0
-50
-100
15.362
-0.331
-16.024
0
5
10
15
20


Magnitude Response (dB) and Phase Response
Magnitude (dB)
Phase (radians)
Frequency (kHz)
0
-20
-40
-60
-80
14.524
8.362
2.2
-3.962
-10.124
0
5
10
15
20


Magnitude Response (dB) and Phase Response
Magnitude (dB)
Phase (radians)
Frequency (kHz)
0
-50
-100
14.992
-0.573
-16.139
0
5
10
15
20

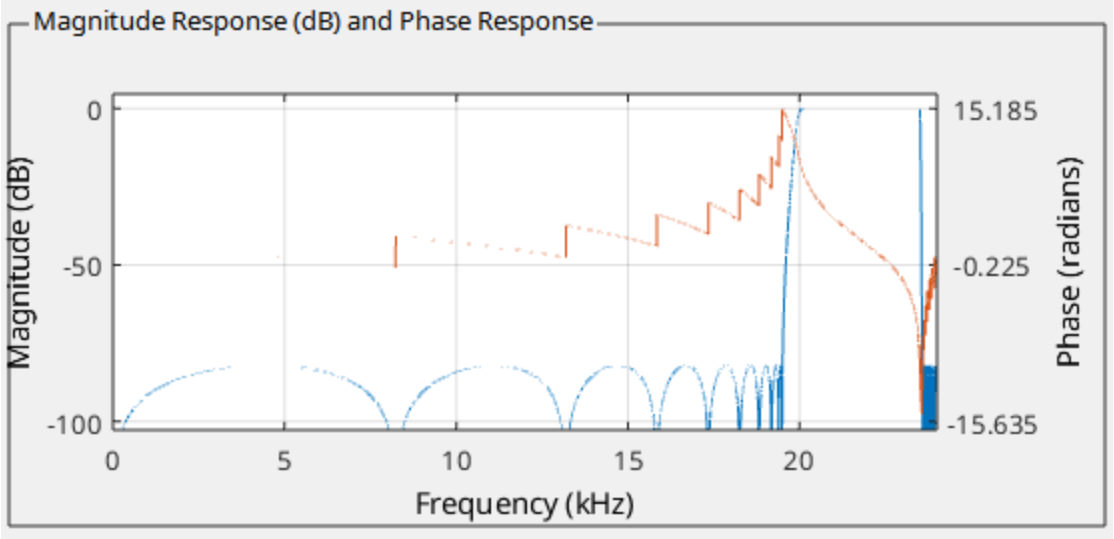


Figure 7. Magnitude and Frequency Plots for IIR Chebyshev Bandpass Filters

Running the equaliser with Chebyshev Type II IIR bandpass filters produced a mix whose overall tone matched what I heard with the FIR equiripple design: muting the 0–5 kHz band hollowed out the vocal and bass, while modest boosts in the middle and upper bands kept guitars, snare, and cymbals bright and present. One noticeable difference, however, was extra ringing in the IIR result, sharp drum hits and plosive consonants carried faint pre- and post-echo tails that were absent, or at least far weaker, in the FIR version. This extra ringing stems from the Chebyshev filter's pole-zero structure and nonlinear phase: energy "stores up" in the filter sections and is released a little before and after fast transients, whereas the linear-phase FIR passes them with minimal smearing. Apart from that added ringing and a slightly steeper cutoff at each band edge, the two approaches delivered similar spectral balance, with the IIR achieving the task using far fewer coefficients and lower computation.

Matlab Code

Part A

```
[y_f, Fs] = audioread('harvard.wav');
y = y_f(:,1);
%Normal Echo
delay_ms = [50 100 200];
echo_sig = y;
decay    = 0.6;
for k = 1:numel(delay_ms)
   D       = round(delay_ms(k)*1e-3*Fs);
   b_delay = [zeros(1,D) 1];
```

```
   y_del   = filter(b_delay,1,y);
   echo_sig = echo_sig + decay*y_del;
end
echo_sig = echo_sig / max(abs(echo_sig));
% sound(echo_sig,Fs);
%Bouncing Echo
D0      = round(120e-3*Fs);
nEchoes = 8;
alpha   = 0.65;
b = 1;
for n = 1:nEchoes
   b = [b zeros(1,D0-1) alpha^n];
end
y_bounce = filter(b,1,y);
y_bounce = y_bounce / max(abs(y_bounce));
% sound(y_bounce,Fs);
%Tempo Echo
bpm = 120;
beat_interval = 60 / bpm;
delay_times = beat_interval * [1, 2, 3];
y_total = y;
for d = delay_times
   delay_samples = round(Fs * d);
   b = [zeros(1, delay_samples), 0.5];
   y_total = y_total + filter(b, 1, y);
end
y_tempo = y_total / max(abs(y_total(:)));
sound(y_tempo,Fs);
```

Part B

```
[y, Fs] = audioread("BeeMoved.flac");
y_filt = zeros(size(y,1), 8, size(y,2));
% % FIR Filters - Equiripple
% All filters designed using filterDesigner and exported to workspace as
% variable
y_filt(:,1,:) = filtfilt(bp_0_3 , 1 , y);
y_filt(:,2,:) = filtfilt(bp_3_6 , 1 , y);
y_filt(:,3,:) = filtfilt(bp_6_9 , 1 , y);
y_filt(:,4,:) = filtfilt(bp_9_12, 1 , y);
y_filt(:,5,:) = filtfilt(bp_12_15,1 , y);
y_filt(:,6,:) = filtfilt(bp_15_18,1 , y);
```

```
y_filt(:,7,:) = filtfilt(bp_18_21,1 , y);
y_filt(:,8,:) = filtfilt(bp_21_24,1 , y);
weights = [1.4 1.2 1.0 0.9 0.8 0.7 0.8 1.2].';
y_sum = reshape(y_filt,[],8) * weights;
y_sum = reshape(y_sum,[],size(y,2));
y_sum = y_sum ./ max(abs(y_sum),[],'all');
% sound(y_sum, Fs);
%IIR Filters - Chebyshev
y_filt_iir = zeros(size(y,1), 5, size(y,2));
y_filt_iir(:,1,:) = sosfilt(sos_0_5 , y);
y_filt_iir(:,2,:) = sosfilt(sos_5_10 , y);
y_filt_iir(:,3,:) = sosfilt(sos_10_15 , y);
y_filt_iir(:,4,:) = sosfilt(sos_15_20 , y);
y_filt_iir(:,5,:) = sosfilt(sos_20_25 , y);
weights = [1.4 1.2 1.0 0.9 0.8].';
ysum = reshape(y_filt_iir,[],5) * weights;
ysum = reshape(ysum,[],size(y,2));
ysum = ysum ./ max(abs(ysum),[],'all');
sound(ysum,Fs);
```