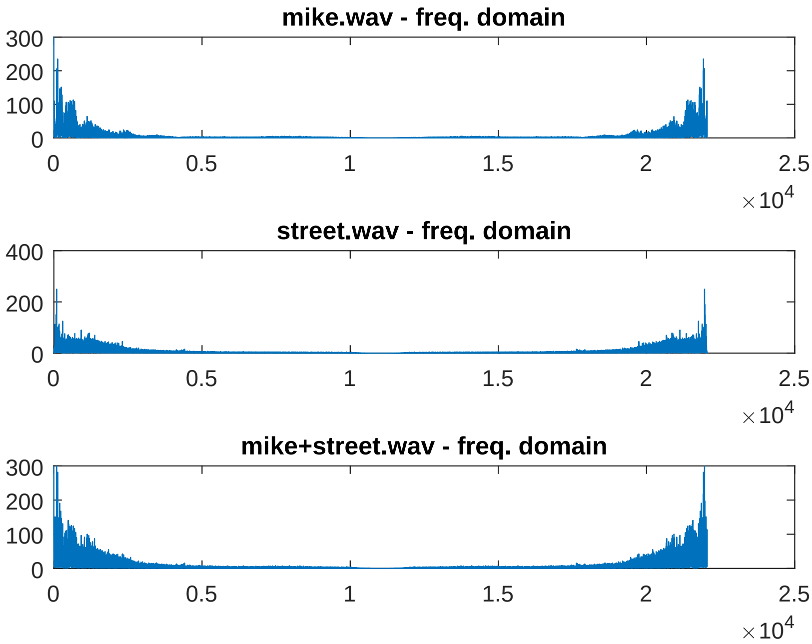
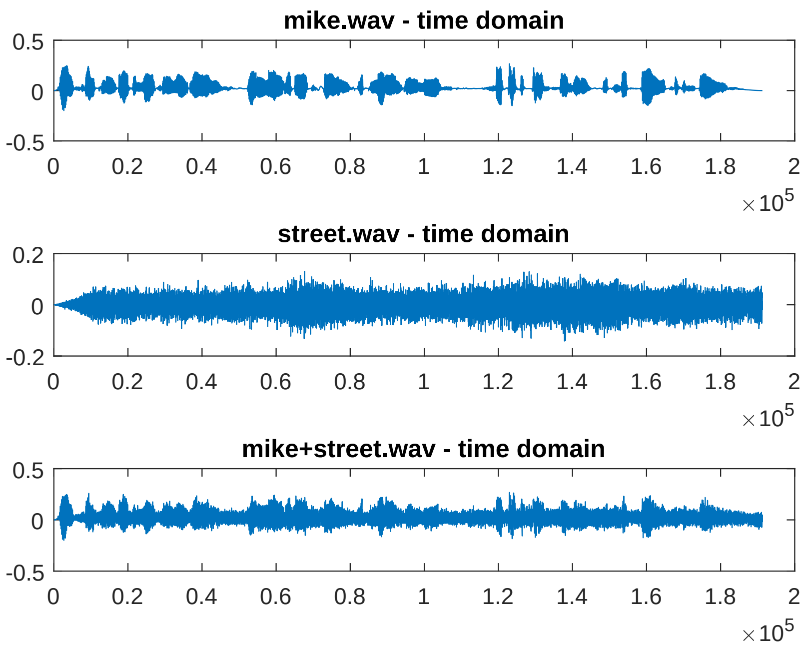
CmpE 362
2018 Spring
Homework 3

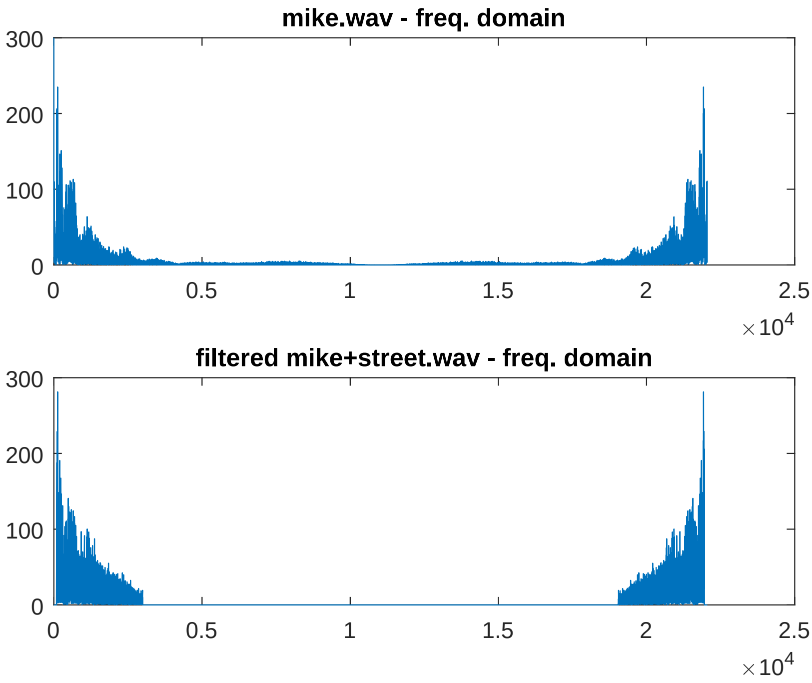
Baran Kılıç
2014400123

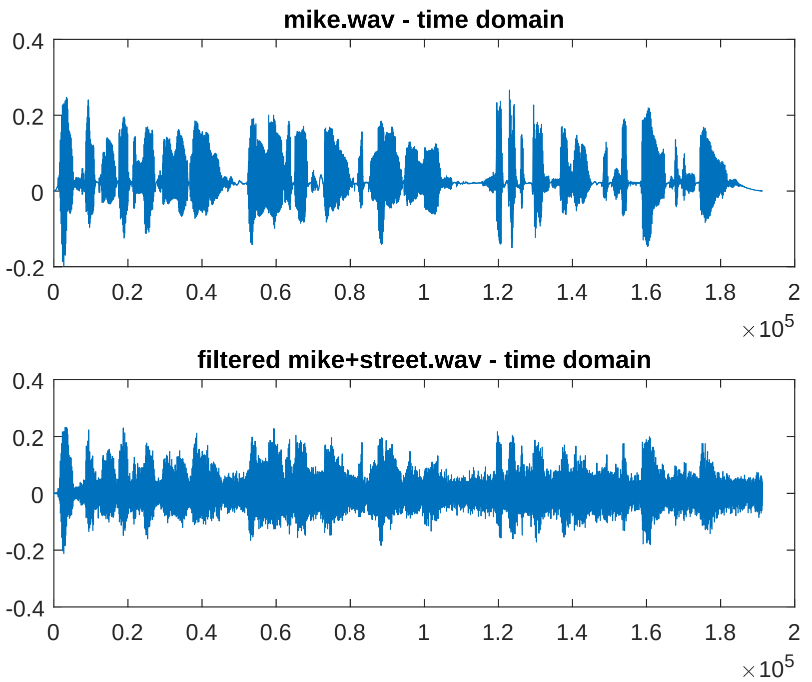
May 2, 2018

1 Clear Noise









SNR: 1.2155

mike.wav and street.wav have some parts that have same frequencies. Filtering outside of the speech band didn't deleted this frequency parts. Therefore, the resulting sound is not cristal clear. However, filtering deleted some part of the street noise because their frequency was outside of the human speech band. The noise is visible in filtered mike+street.wav in time domain where mike do not speak. In mike.wav, these parts have zero amplitude. But, in filtered mike+street.wav, the amplitude is never zero.

1.1 Code

```
[y1, Fs] = audioread('mike.wav');
[y2, Fs] = audioread('street.wav');

y3 = y1 + y2; %combined sound

NFFT = length(y3);
Y = fft(y3,NFFT);

Y1 = fft(y1,NFFT);
Y2 = fft(y2,NFFT);

lowerBoundFreq = 95;
upperBoundFreq = 3000;
lbfAdjusted = round( length(Y)*lowerBoundFreq/Fs );
ubfAdjusted = round( length(Y)*upperBoundFreq/Fs );
middleAdjusted = length(Y)/2;
endAdjusted = length(Y);

Ycleared = Y;

Ycleared(1:lbfAdjusted) = 0; % delete frequencies between 0 - 95 Hz
Ycleared(ubfAdjusted:middleAdjusted)=0; % delete frequencies between 3 - 11 KHz

Ycleared(endAdjusted-lbfAdjusted:endAdjusted) = 0;
% delete frequencies between -95 and 0 Hz
Ycleared(middleAdjusted:endAdjusted-ubfAdjusted) = 0;
% delete frequencies between -11 and -3 KHz

y3cleared = ifft(Ycleared,NFFT,'symmetric');

sound(y3cleared,Fs)

f = (0:length(Y)-1)*Fs/length(Y);

figure;
subplot(2,2,1);
plot(f,abs(Y1))
```

```
ax=gca;
ylim(ax, [0,300])
title('mike.wav - freq. domain')

subplot(2,2,2);
plot(f,abs(Y2))
title('street.wav - freq. domain')

subplot(2,2,3);
plot(f,abs(Y))
ax=gca;
ylim(ax, [0,300])
title('mike+street.wav - freq. domain')

figure;
subplot(2,2,1);
plot(y1)
title('mike.wav - time domain')

subplot(2,2,2);
plot(y2)
title('street.wav - time domain')

subplot(2,2,3);
plot(y3)
title('mike+street.wav - time domain')

figure;
subplot(2,1,1);
plot(f,abs(Y1))
ax=gca;
ylim(ax, [0,300])
title('mike.wav - freq. domain')

subplot(2,1,2);
plot(f,abs(Ycleared))
title('filtered mike+street.wav - freq. domain')

figure;
subplot(2,1,1);
plot(y1)
title('mike.wav - time domain')

subplot(2,1,2);
```

```
plot(y3cleared)
title('filtered mike+street.wav - time domain')

snr = 10*log10(sum(y1.^2) / sum((y3cleared-y1).^2));
disp(['SNR: ',num2str(snr)]);
```

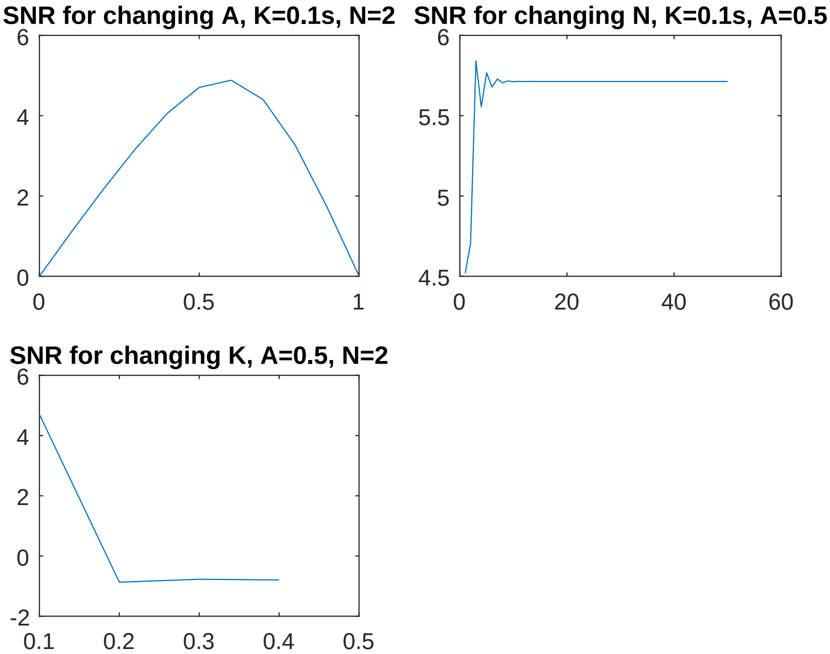
2 Clap Differentiate

I converted audio to frequency domain using fft. According to my observations, clap sounds have more frequency components between 500 and 1900 Hz. Snap sounds have more frequency components between 1900 and 9000 Hz. Therefore, I take the average of these ranges and compare them. If the average of 500-1900Hz is bigger than double the value of the average of 1900-9000Hz, I say that this sound is a clap. Otherwise, it is a snap.

2.1 Code

```
prompt = 'Enter name of the audio file in the working directory (clap.wav): ';
filename = input(prompt,'s');

[y, Fs] = audioread(filename);

Y = fft(y);

threshold = round(1900*length(Y)/Fs);
lowerBound = round(500*length(Y)/Fs);
upperBound = round(9000*length(Y)/Fs);

%mean of frequencies between 500 and 1900 Hz
m1 = mean(abs(Y(lowerBound:threshold)));
%mean of frequencies between 1900 and 9000 Hz
m2 = mean(abs(Y(threshold:upperBound)));

if m1 > 2*m2
    disp('clap sound detected')
else
    disp('snap sound detected')
end
```

3 N Tap Filter



We get the best filter if A has the value 0.5. If A goes to either 0 or 1, the filter becomes worse. When we increase N, the filter improves at first. However, after a point, increasing N does not have any effect on the filter. When we increase K (the delay), SNR value decreases. It becomes hard to filter the noise.

3.1 Code

```
K=0.1;
a=0.5;
N=1;

[y, Fs] = audioread('mike.wav');
yDelayed = [zeros(Fs*K, 1); y]; % 0.1 sec delay
yCombined = yDelayed(1:length(y)) + y;

snrForChangingA = [];
for a=0:0.1:1
filtered = nTap(yCombined,0.1,a,2,Fs);
snr = 10*log10(sum(y.^2) / sum((filtered-y).^2));
snrForChangingA(end+1) = snr;
end
```

```
snrForChangingN = [];
for N=1:1:50
filtered = nTap(yCombined,0.1,0.5,N,Fs);
snr = 10*log10(sum(y.^2) / sum((filtered-y).^2));
snrForChangingN(end+1) = snr;
end

snrForChangingK = [];
for K=0.1:0.1:0.4
filtered = nTap(yCombined,K,0.5,2,Fs);
snr = 10*log10(sum(y.^2) / sum((filtered-y).^2));
snrForChangingK(end+1) = snr;
end

figure;
subplot(2,2,1);
plot(0:0.1:1, snrForChangingA);
title('SNR for changing A, K=0.1s, N=2')

subplot(2,2,2);
plot(1:50, snrForChangingN);
title('SNR for changing N, K=0.1s, A=0.5')

subplot(2,2,3);
plot(0.1:0.1:0.4, snrForChangingK);
title('SNR for changing K, A=0.5, N=2')

function result = nTap(signal,K,a,N,Fs)
result=signal;
for i=1:N
delayedInput = [zeros(round(Fs*K*i), 1); signal]; % K second delay, i times
delayedInput = delayedInput.*((-a)^i);
result = result + delayedInput(1:length(signal));
end
end
```