Signal Processing Lab Project-2023

REPORT

Group name: Power On

Team member 1: Masumi Desai

Roll no: 2022102057

Team member 2: Kripi Singla

Roll no: 2022102063

Team member 3: Shravan Gadbail

Roll no: 2022102025

Introduction

The Project aims to deal with echoes, how to generate it and remove it and classifying different noises playing in the background in a music audio file.

The methods used in this project are convolution with a weighted decaying impulse-train, cross correlation, etc.

PART 1 : Echo Creation

1.1 Aim/Objective

In signal processing terms, an "echo" refers to a delayed and attenuated version of a sound or signal that is heard after the original sound. It is a time-delayed replication of the original signal, where the delayed version is often quieter than the original due to the

signal being reflected off surfaces or traveling a longer path. Echoes are commonly encountered in various acoustic environments, such as rooms, hallways, and outdoor spaces.

In this particular part, we are required to create an echo effect for a given audio file.

1.2 Input

The input provided in this part is an echo free signal.

1.3 Problem Solving Approach

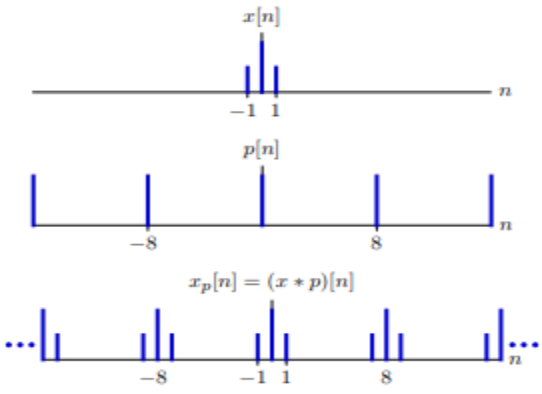
Signal Processing Technique Applied:

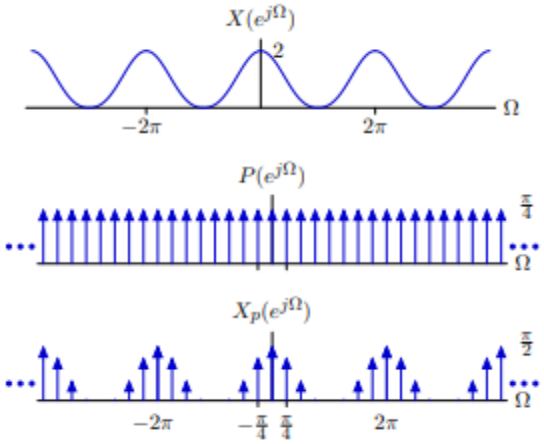
The signal processing technique applied in this particular part is Convolution with an Impulse train.

For the given audio signal, we perform its convolution with a decaying impulse train having a uniform delay ,which can be set as per our requirements.

The designed impulse train is something of the type a^n*u[n-k*delay_length] where the range of k is usually kept 0 to 1000.

The following images will make the technique more clear:-





Code Block:-

```
[x, Fs] = audioread('q1_hard.wav');

% sound(x,Fs);

% x_mono = mean(x,2);

% plot(x);

delay = 0.6;

D = round(delay * Fs);

a = 0.6;

impulse_response = [];

current_amplitude = 1;

while current_amplitude > 0.1

   t_impulse = (0:length(impulse_response) + D - 1) / Fs;

   current_impulse = a.^t_impulse;

   impulse_response = [impulse_response, zeros(1, D), current_amplitude *
current_impulse];

   current_amplitude = current_amplitude * a;
```

```
end

y= conv(x_mono, impulse_response);

y = y / max(abs(y));

subplot(2, 1, 1);

plot(x);

title('Original Signal');

subplot(2, 1, 2);

plot(y);

title('Echoed Signal');

sound(y, Fs);
```

Code Implementation

This MATLAB code implements a basic echo effect on an audio signal. It reads an audio file and stores the audio signal in x and the sampling frequency in Fs. It sets a particular delay time and calculates the corresponding number of samples (D) based on the sampling frequency.

It generates an echo impulse response. The loop creates an exponentially decaying impulse response with decreasing amplitudes over time. The impulse response is constructed by appending each new impulse to the previous response with a delay of D samples.

It convolves the original audio signal with the generated impulse response, creating the echoed signal y.

It normalizes the echoed signal to ensure that its maximum absolute amplitude is 1.

In summary, this code implements a simple echo effect on an audio signal by convolving it with an exponentially decaying impulse response. The resulting echoed signal is then normalized and both the original and echoed signals are plotted in a subplot. Finally, the echoed signal is played back.

Difficulties Faced

Initially, we started with the code that provided just a single echo for a particular note present in the audio file by implementing the feedback equation y[n] = x[n] + alpha*x[n-D]. It was not resembling the natural echo behaviour which decays with time and is no longer heard.

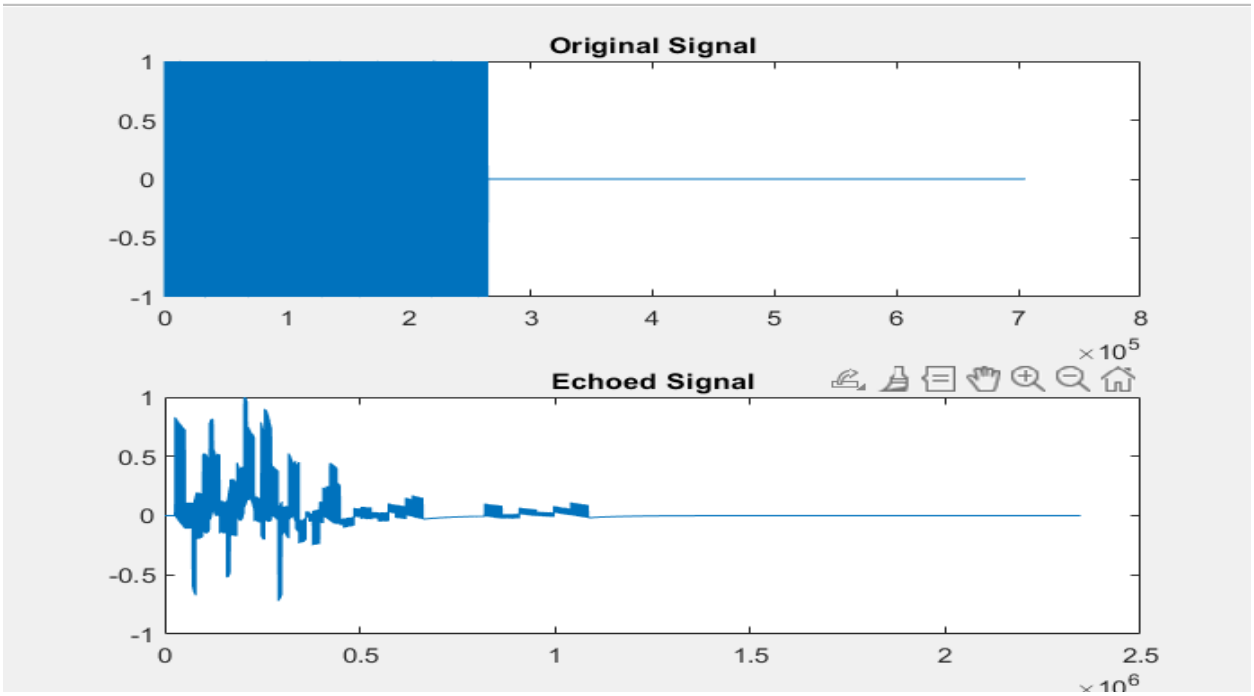
To overcome this problem, we thought of convolving the input signal with a decaying impulse train so that we can get multiple echoes that decay naturally over time.

Since, the amplitude should nearly equal to zero, it was hard for us to determine how to terminate the program exactly. Eventually, we just set a current amplitude threshold which is close to zero,about 0.05 or 0.1 as per our requirements.

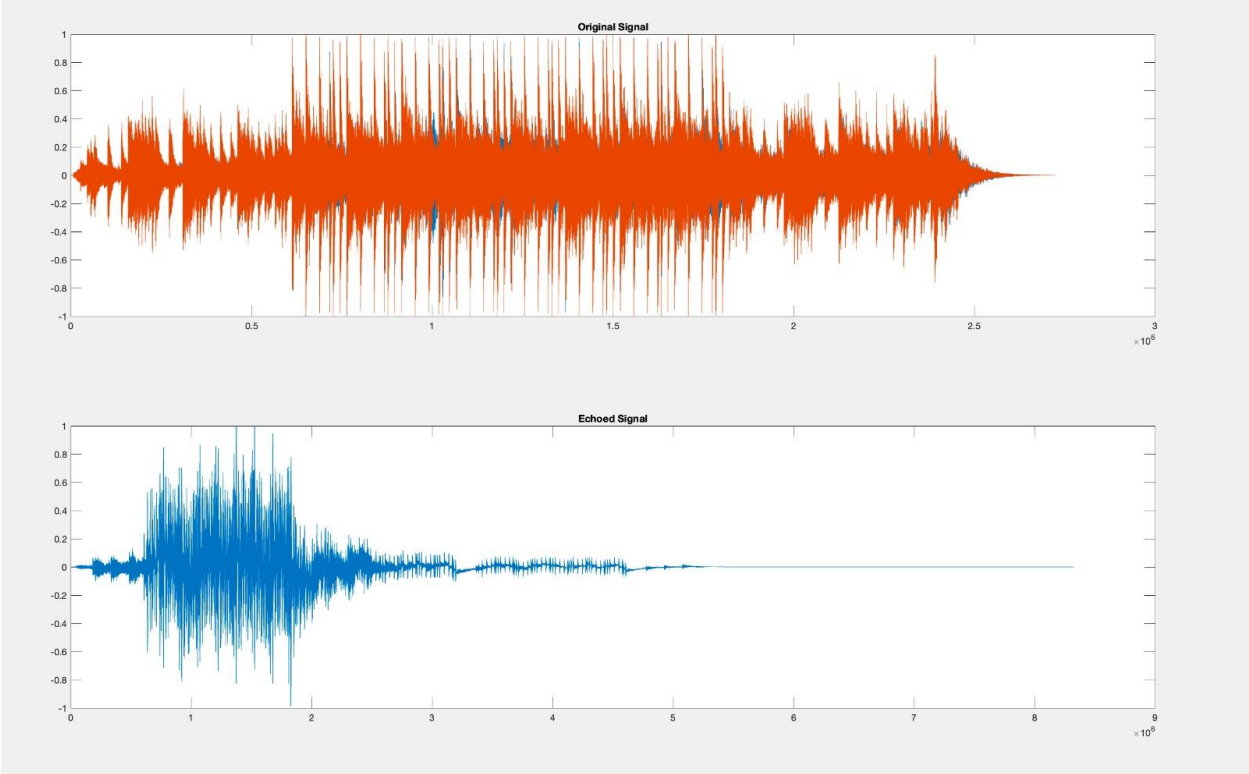
While figuring this out, we tried to do the same procedure in the frequency domain but finally switched back to our implementation in the time domain.

For the given q1_hard file, the conv function was giving out an error, on displaying and checking the input signal, we discovered that it is a stereo signal (the signal with two dimensions) and eventually the mean over both the columns was taken to make it one-dimensional.

1.4 Results


Original Signal
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
×10^5
Echoed Signal
1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
×10^6

q1_easy


Original Signal
Echoed Signal

q1_hard

'q1_hard' above is obtained using attenuation factor(a) = 0.7, this is the highest value of attenuation that we could achieve as when the attenuation factor exceeded 0.7, the final audio signal went out of bound.

1.5 Conclusion

In order to generate an echoed signal from the given original signal, convolution with a decaying impulse train is done.

PART 2: Cancel the echo

2.1 Objective/Aim:

We were given 2 audio signals which had echo in them, so out objective was to remove the echo.

2.2 Input:

The given audio files were:

- q2_easy.wav
- q2_not_so_easy.wav

And we were todd that we can use the parameter desired signal while implementing the LMS algorithm so the other 2 input files were

- q1_easy
- q1_hard

We had the freedom to add echo to these signals and then do the echo cancellation using the LMS or NLMS approach.

2.3 Problem solving approach:

We tried to use the LMS filter which is a type of adaptive filter, which changes its filter coefficients on every iteration so that the difference between the desired signal and the output of the filter becomes the least.

Auto Correlation:

Autocorrelation refers to the degree of correlation of the same variables between two successive time intervals.

It measures how the lagged version of the value of a variable is related to the original version of it in a time series.

It basically tells us about the relationship between the current sample of the signal and its past and future values.

Code:

```
function y = echo_cancellation(x, d, mu, order)

   N = length(x);

   w = zeros(order, 1);

   y = zeros(N, 1);


   for n = order:N

            x_n = d(n:-1:n-order+1);

            y(n) = w' * x_n;

        e = x(n) - y(n);

        w = w + mu * e * x_n;

   end

   disp(mean(e));

   % Apply LMS algorithm

   while mean(e)>0.001

        for n = order:N

            x_n = x(n:-1:n-order+1);

            y(n) = w' * x_n;

            e = d(n) - y(n);

            w = w + mu * e * x_n;

        end

   end

end
```

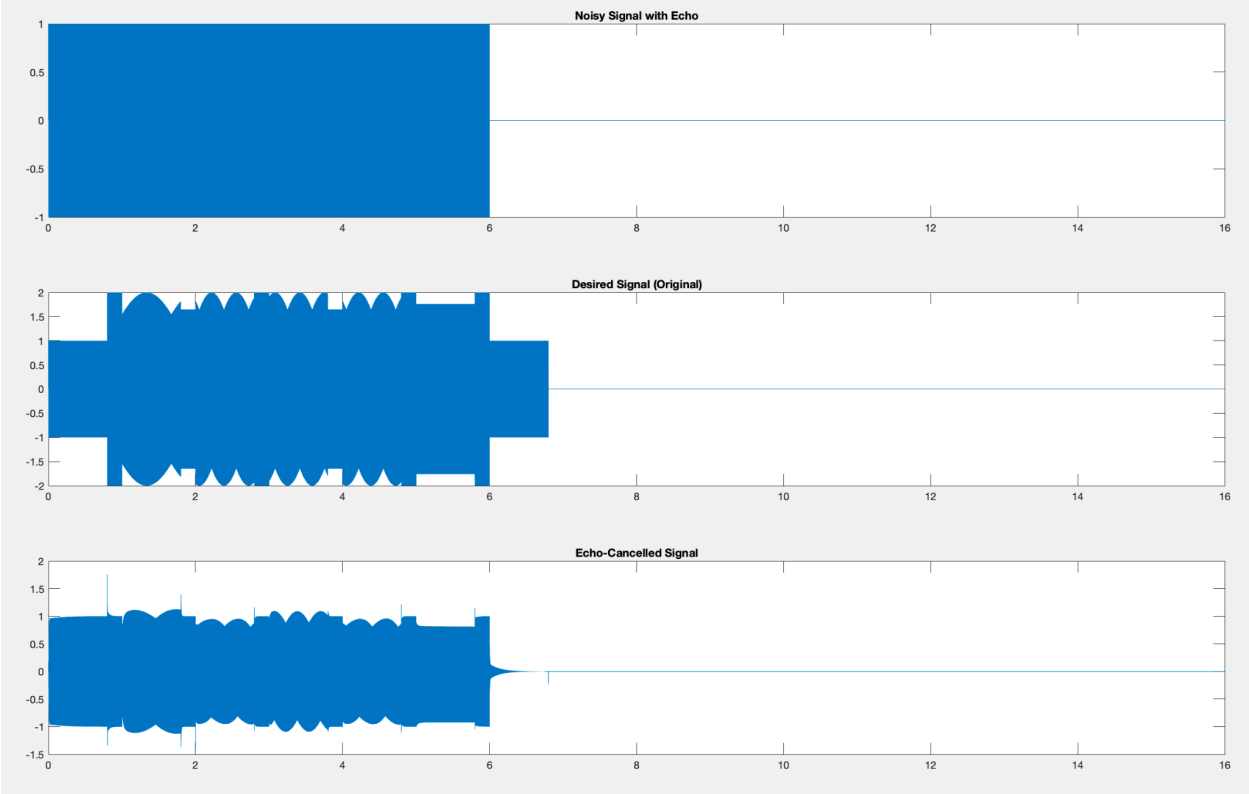
2.4 Difficulties faced:

We were unable to design the LMS filter in the start as we didn't have a desired signal and even after we got to know it was very challenging to design the filter and as a result, the filter isn't removing the echo completely but is sure attenuating the echo by a significant amount.

The file to run is q2_1.m

Note: The file q2_2 is the code for one of our tries.

2.5 Results:



This plot is for the q1_easy.wav file. We first added echo to it and then we used the LMS filtering method to remove it.

PART 3: What is this noise?

3.1: Objective/Aim:

We were to classify background noise in music recordings without removing the noise. The objective is to accurately identify and categorize the type of noise present in the recording, distinguishing source of origin.

3.2: Input:

The four different background noises that were given are:

- Music_ceiling-fan
- Music_city-traffic
- Music_pressure-cooker
- Music_water-pump

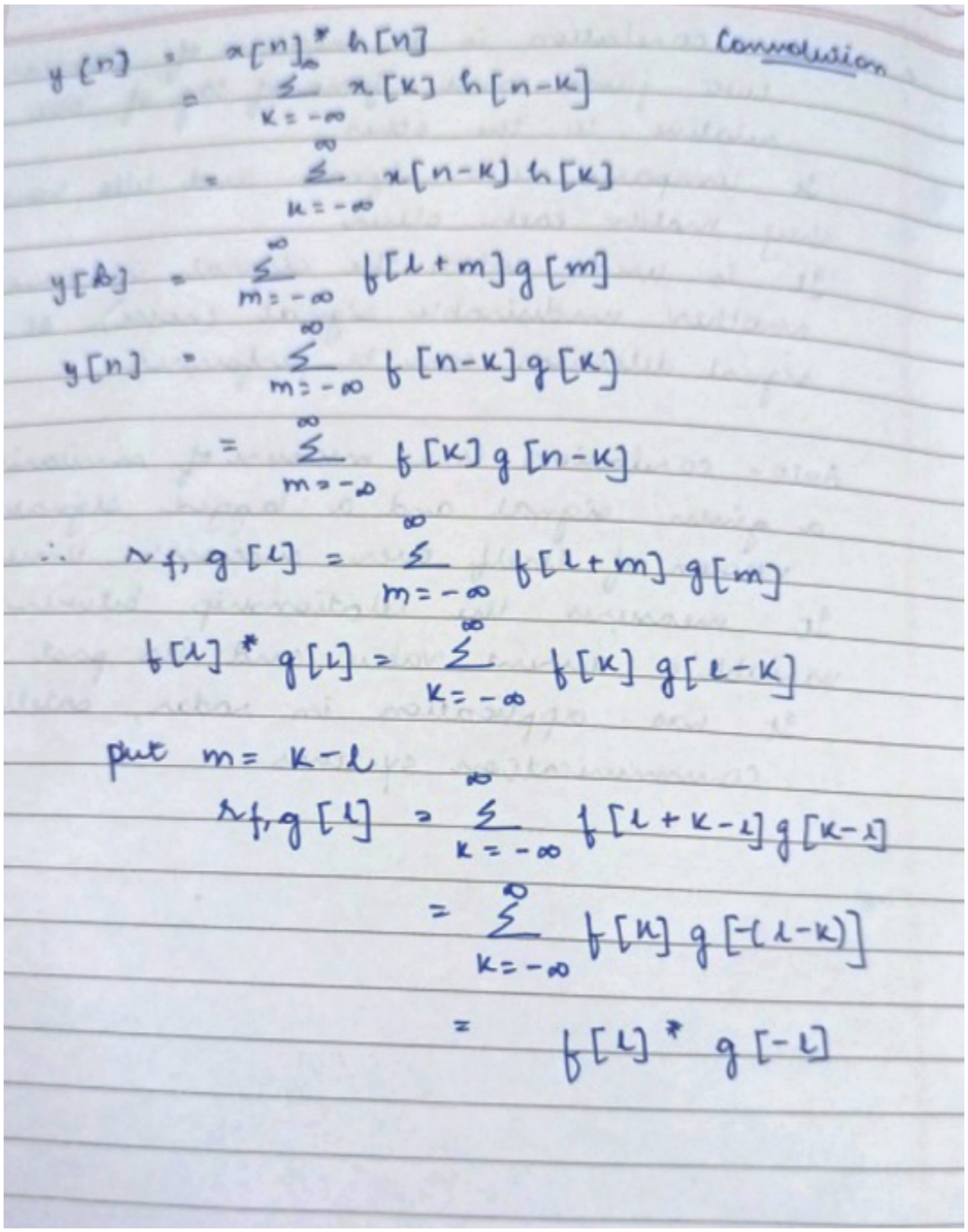
3.3 Problem-Solving Approach:

Signal Processing Techniques applied:

We used cross-correlation of a particular noise with all the noisy audio signals given.

As we know cross-correlation is a measure of similarity between two signals and thus one of the four plots with the highest peak for a particular audio signal is the one with that type of noise.

Cross Correlation function:-



Code Block:

```
[inputAudioFile = "music_city-traffic.wav";

f1 = "traffic.mp3";
```

```
    f2 = "pressure.mp3";
    f3 = "ceiling_fan.wav";
    f4 = "water_pump.mp3";

    [y,~] = loadAudioFile(inputAudioFile);
    [y1,~] = loadAudioFile(f1);
    [y2,~] = loadAudioFile(f2);
    [y3,~] = loadAudioFile(f3);
    [y4,~] = loadAudioFile(f4);
noiseType = classifyNoise(y,y1,y2,y3,y4);
disp(['The detected noise type is: ' noiseType]);
% Function 1: Load the input audio file
function [y,fs] = loadAudioFile(filename)
    % Load the audio file
    [y,fs] = audioread(filename);
end
% Function 2: Noise classification
function noiseType = classifyNoise(y,y1,y2,y3,y4)
    minLen = min(length(y), length(y1));
    y1 = y1(1:minLen);
    ytemp = y(1:minLen);
    lags = -minLen+1:minLen-1;
    crossCorr = xcorr(ytemp, y1);
    maximum = max(crossCorr);
    noiseType = "Traffic";
```

```
figure;
plot(lags, crossCorr);
title('Cross-Correlation with Traffic');
xlabel('Lag');
ylabel('Cross-Correlation');
minLen = min(length(y), length(y2));
y2 = y2(1:minLen);
ytemp = y(1:minLen);
lags = -minLen+1:minLen-1;
crossCorr = xcorr(ytemp, y2);
if maximum<max(crossCorr)
    maximum = max(crossCorr);
    noiseType = "Pressure Cooker";
end
figure;
plot(lags, crossCorr);
title('Cross-Correlation with Pressure Cooker');
xlabel('Lag');
ylabel('Cross-Correlation');
minLen = min(length(y), length(y3));
y3 = y3(1:minLen);
ytemp = y(1:minLen);
lags = -minLen+1:minLen-1;
crossCorr = xcorr(ytemp, y3);
if maximum<max(crossCorr)
    maximum = max(crossCorr);
```

```
        noiseType = "Ceiling fan";
    end


    figure;
    plot(lags, crossCorr);
    title('Cross-Correlation with Ceiling fan');
    xlabel('Lag');
    ylabel('Cross-Correlation');
    minLen = min(length(y), length(y4));
    y4 = y4(1:minLen);
    ytemp = y(1:minLen);
    lags = -minLen+1:minLen-1;
    crossCorr = xcorr(ytemp, y4);
    maximum = max(crossCorr);
    if maximum<max(crossCorr)
        maximum = max(crossCorr);
        noiseType = "Water pump";
    end
    figure;
    plot(lags, crossCorr);
    title('Cross-Correlation Water Pump');
    xlabel('Lag');
    ylabel('Cross-Correlation');
end
```

Code Implementation

The code loads one audio signal x1 and four noise signals(x2, x3, x4, x5) and their respective sampling rates (fs1, fs2, fs3, fs4, fs5) using the audioread function.

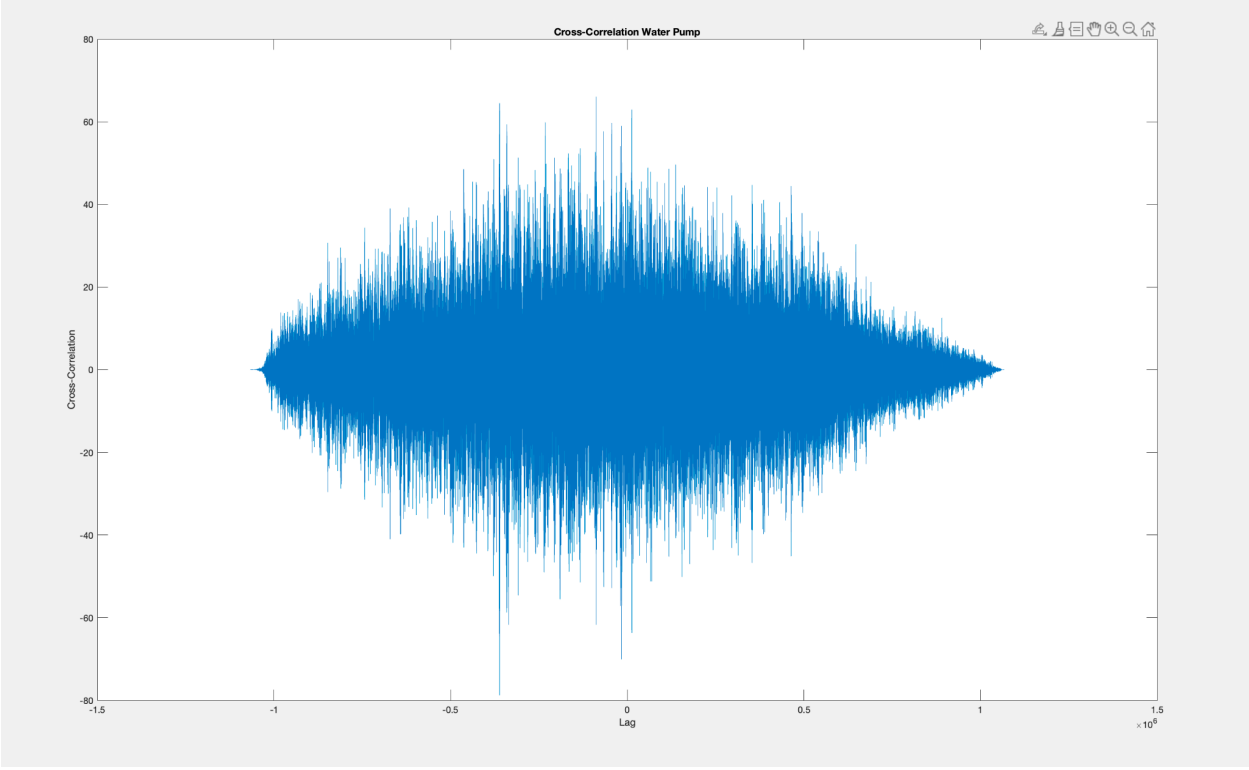
The code determines the minimum length among all the audio signals and truncates them to this common length (minLen).

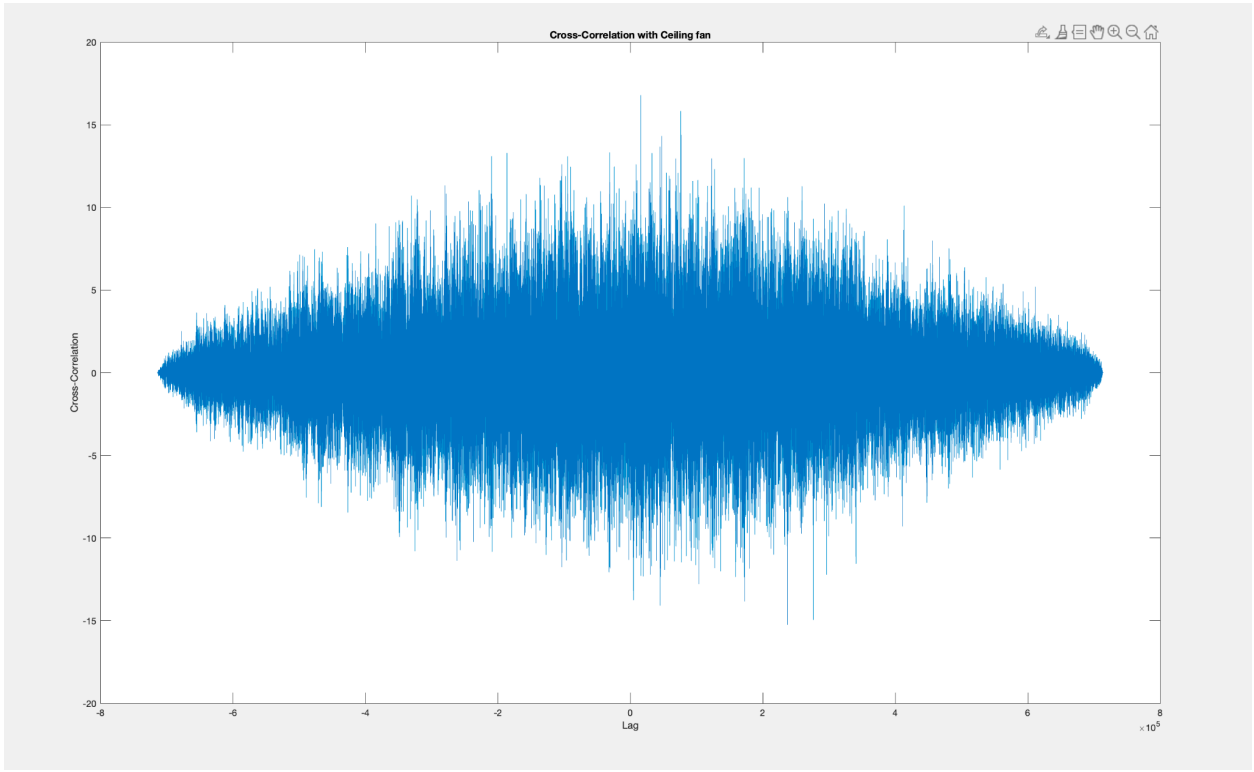
The cross-correlation is calculated between x1 and each of the other signals (x2, x3, x4, x5) using the xcorr function. The results are stored in variables (crossCorr1, crossCorr2, crossCorr3, crossCorr4).

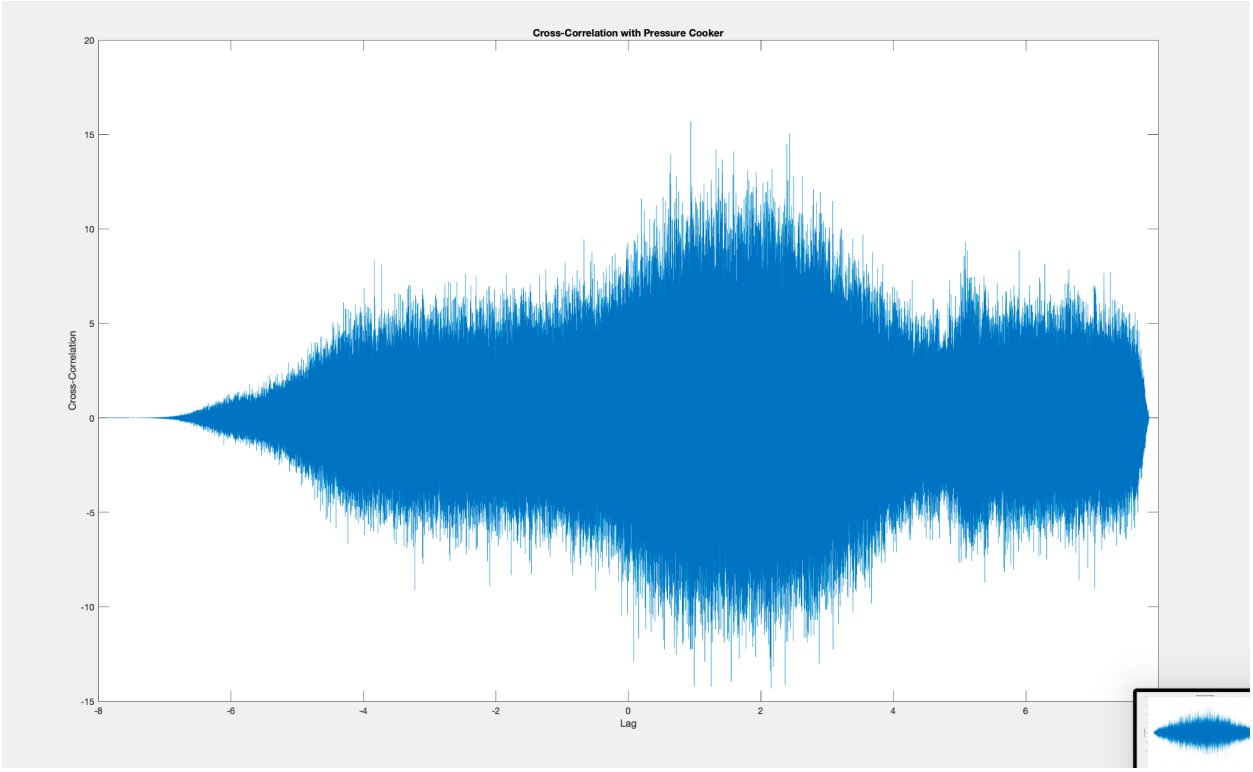
The cross-correlation outputs are plotted and the one giving the maximum peak is declared as the noise present in that particular signal.

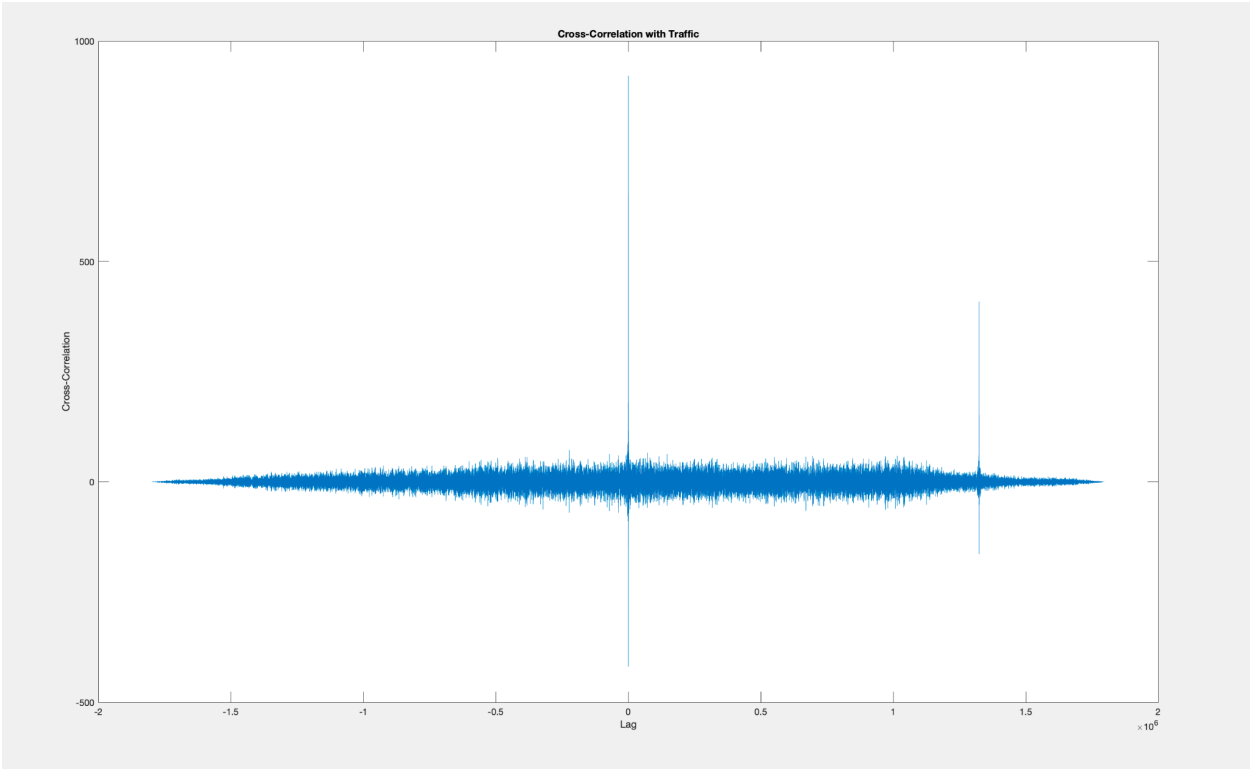
3.4 Results:

City traffic classification




Cross-Correlation with Ceiling fan
Cross-Correlation
Lag
20
15
10
5
0
-5
-10
-15
-20
-8
-6
-4
-2
0
2
4
6
8
×10^5

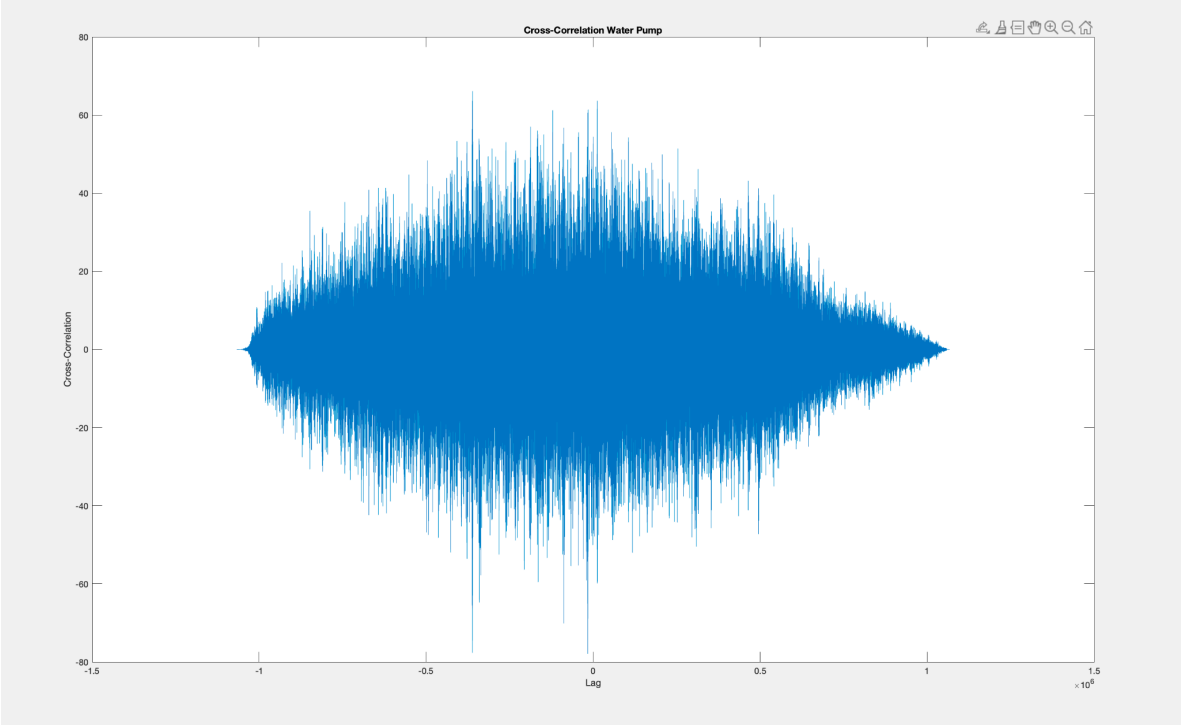

Cross-Correlation with Pressure Cooker
Cross-Correlation
Lag
20
15
10
5
0
-5
-10
-15
-8
-6
-4
-2
0
2
4
6

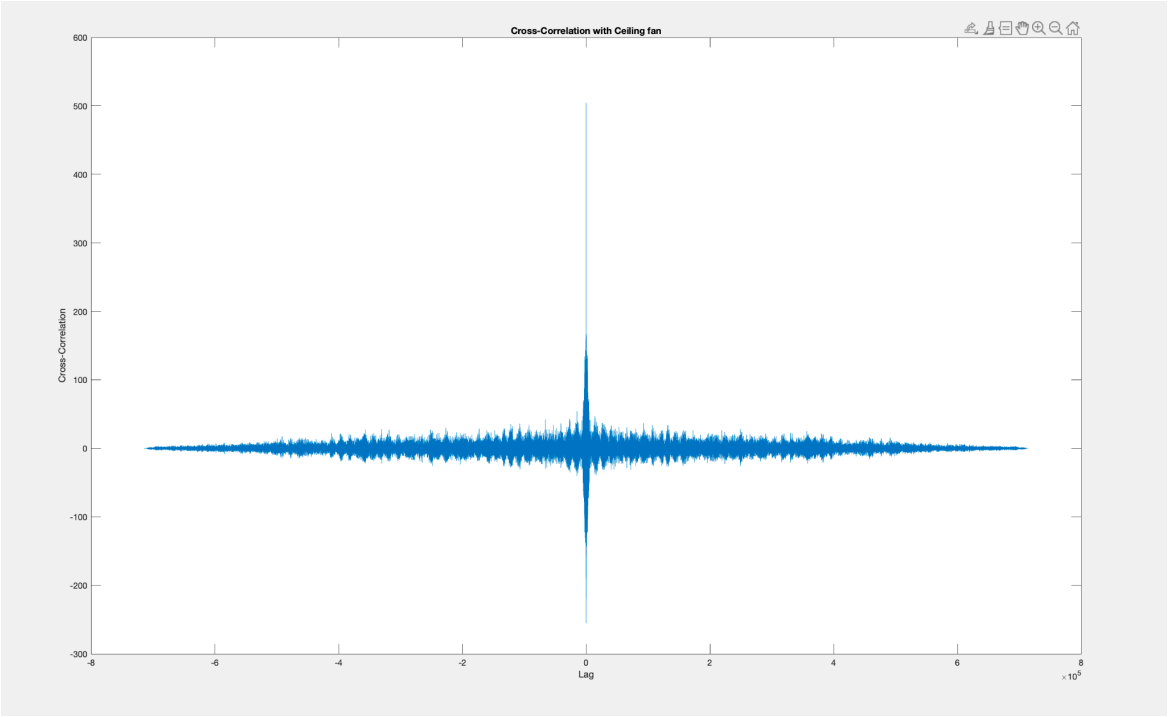

Cross-Correlation with Traffic
Cross-Correlation
Lag
1000
500
0
-500
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
×10^6

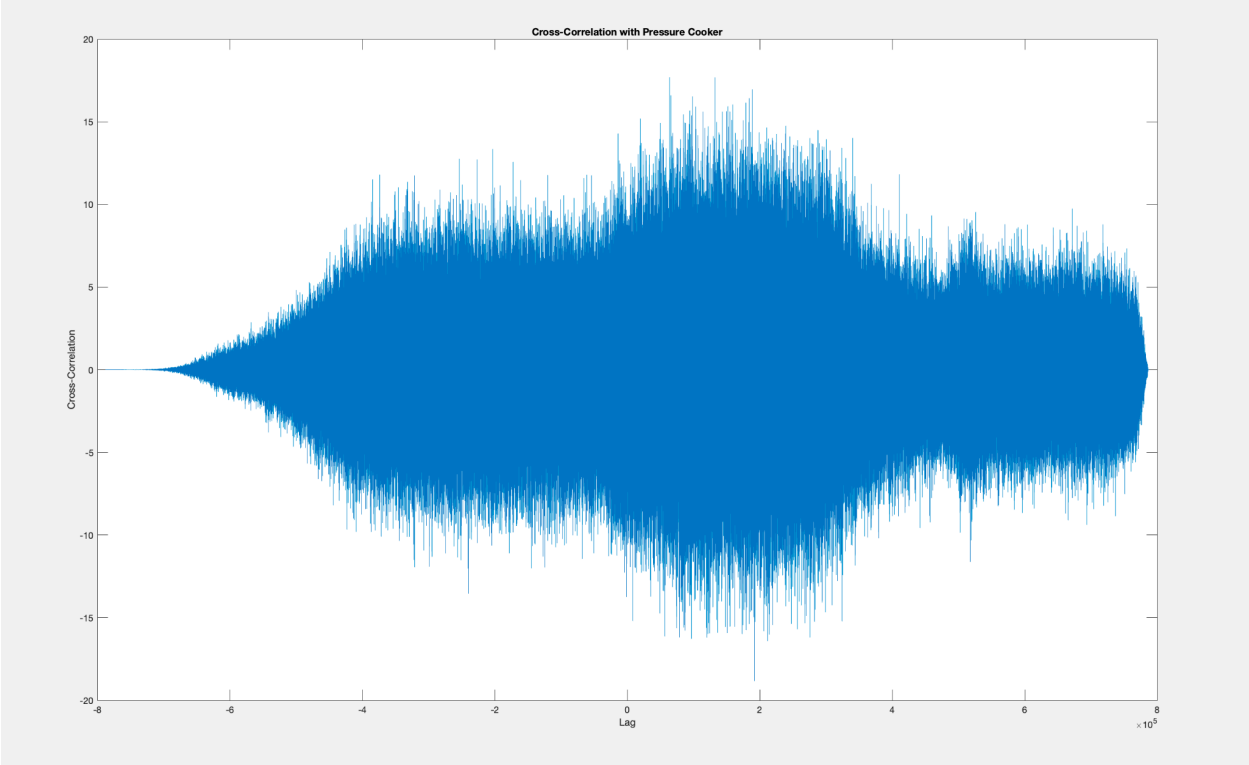
Output :

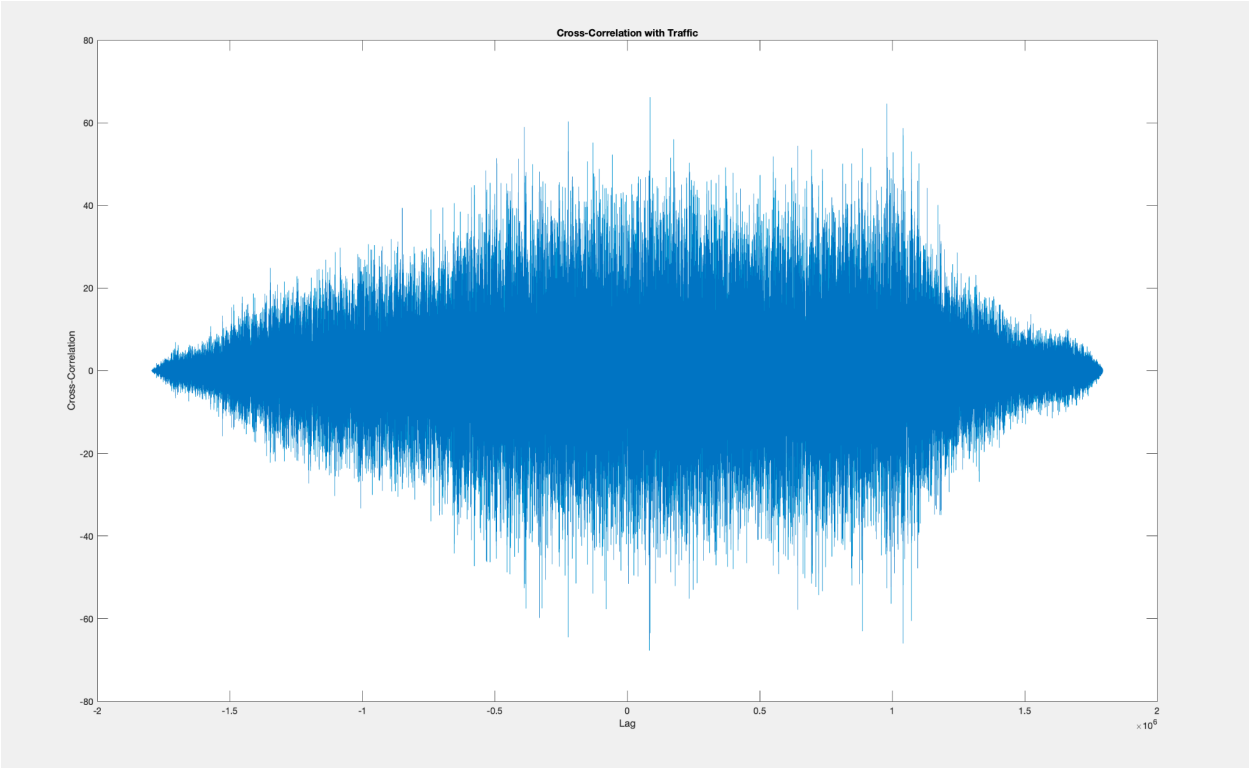
```
>> q3
    "The detected noise type is: "    "Traffic"
```

Ceiling fan classification




Cross-Correlation with Ceiling fan
Cross-Correlation
Lag
×10^5
600
500
400
300
200
100
0
-100
-200
-300
-8
-6
-4
-2
0
2
4
6
8

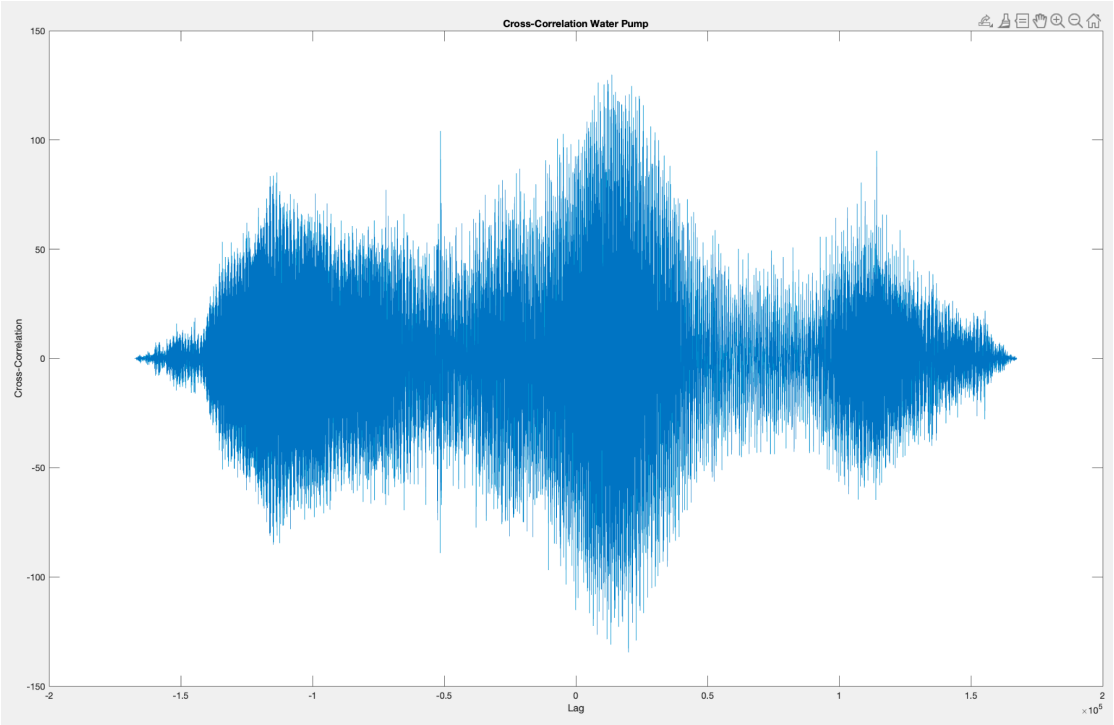

Cross-Correlation with Pressure Cooker
Cross-Correlation
Lag
×10^5
20
15
10
5
0
-5
-10
-15
-20
-8
-6
-4
-2
0
2
4
6
8

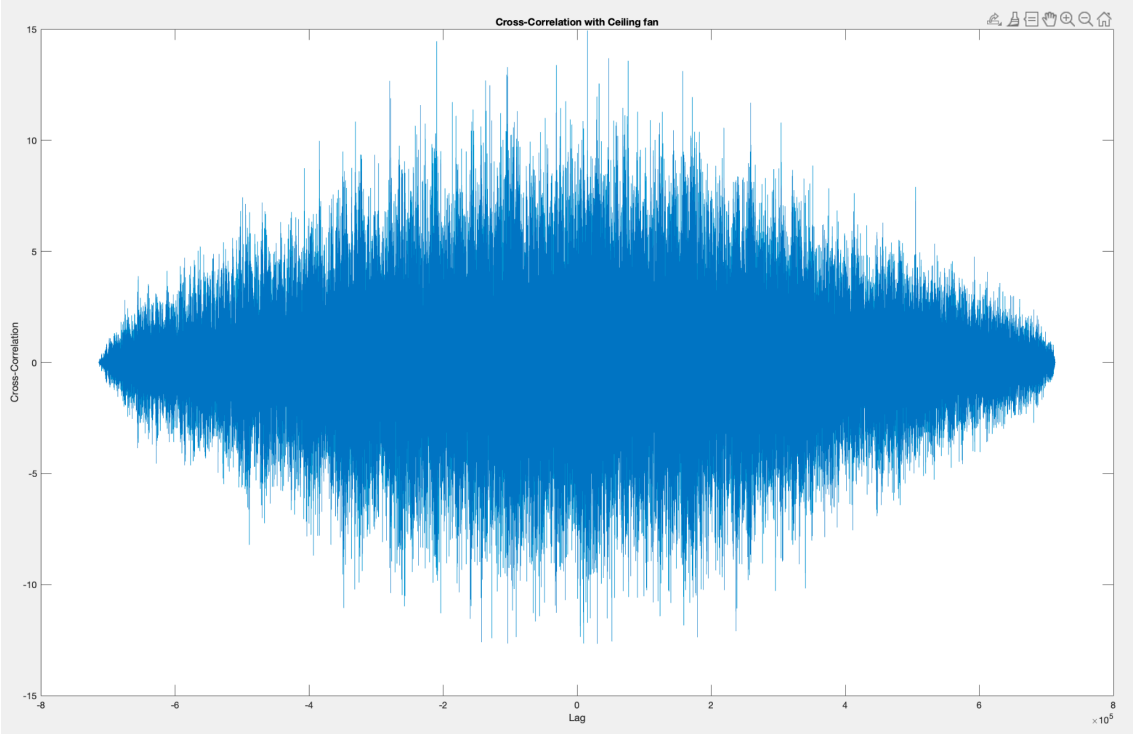


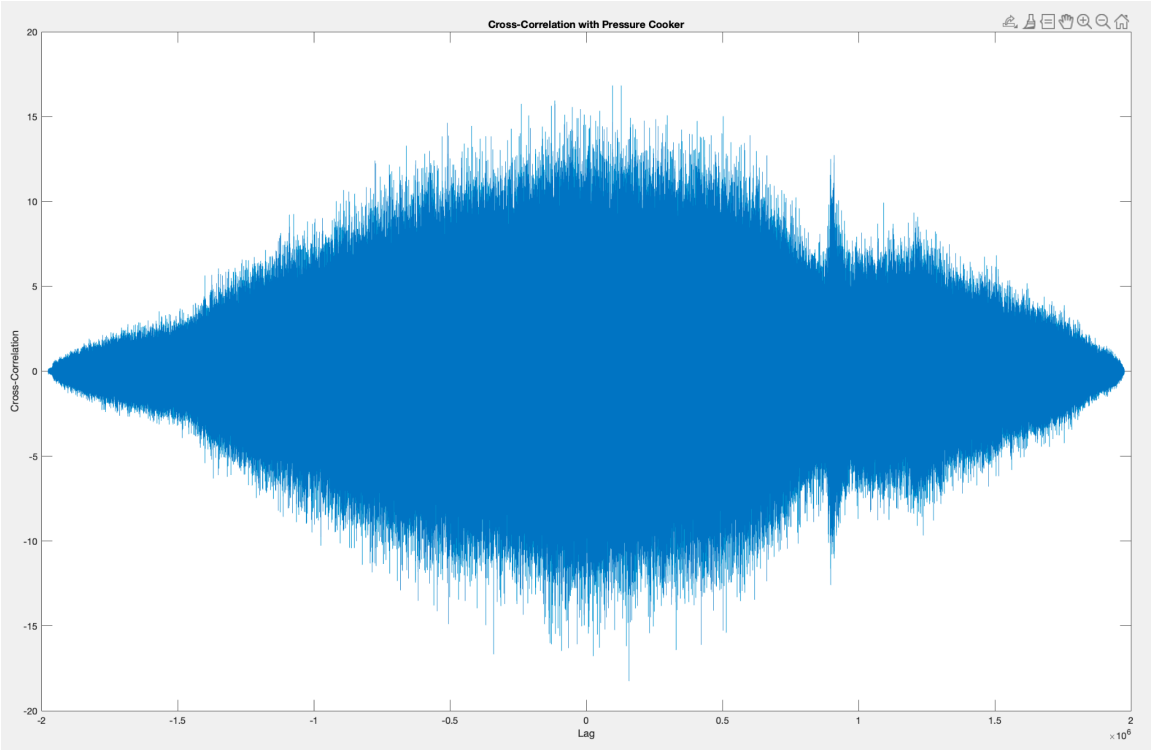
Output:

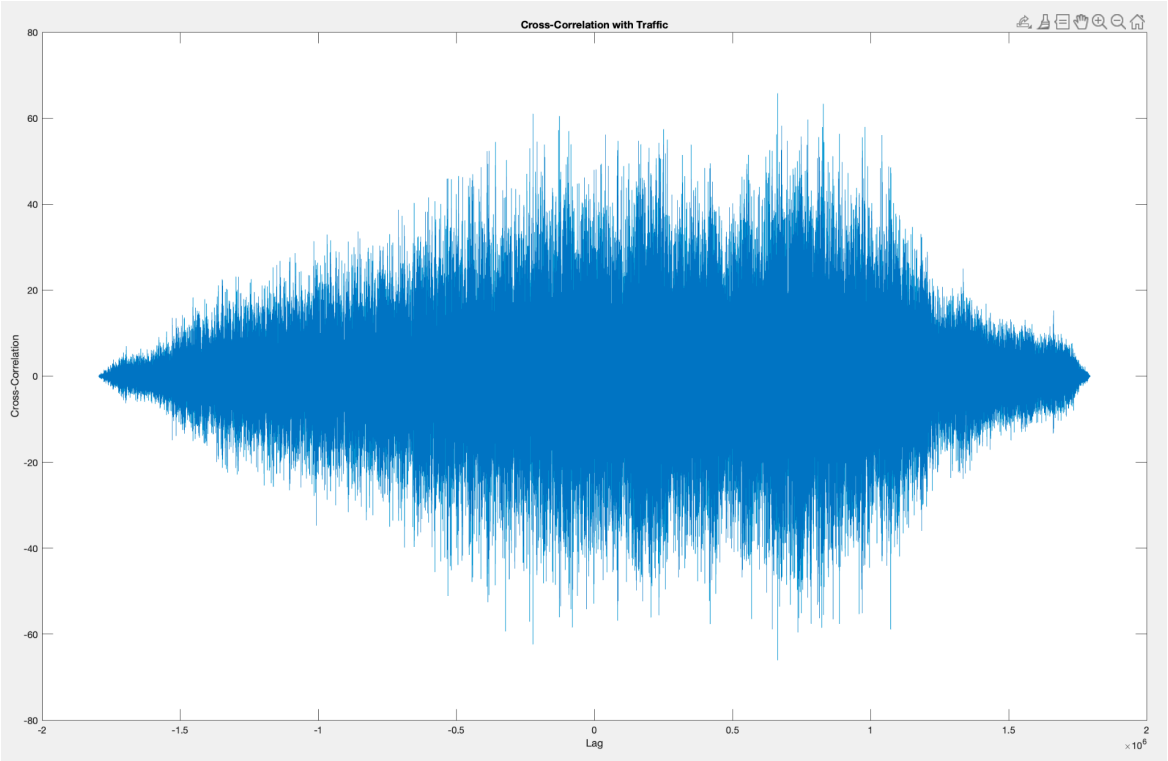
```
>> q3
    "The detected noise type is: "    "Ceiling fan"
```

Water pump classification


Cross-Correlation Water Pump
Cross-Correlation
Lag
150
100
50
0
-50
-100
-150
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
×10^5


Cross-Correlation with Ceiling fan
Cross-Correlation
Lag
15
10
5
0
-5
-10
-15
-8
-6
-4
-2
0
2
4
6
8
×10^5

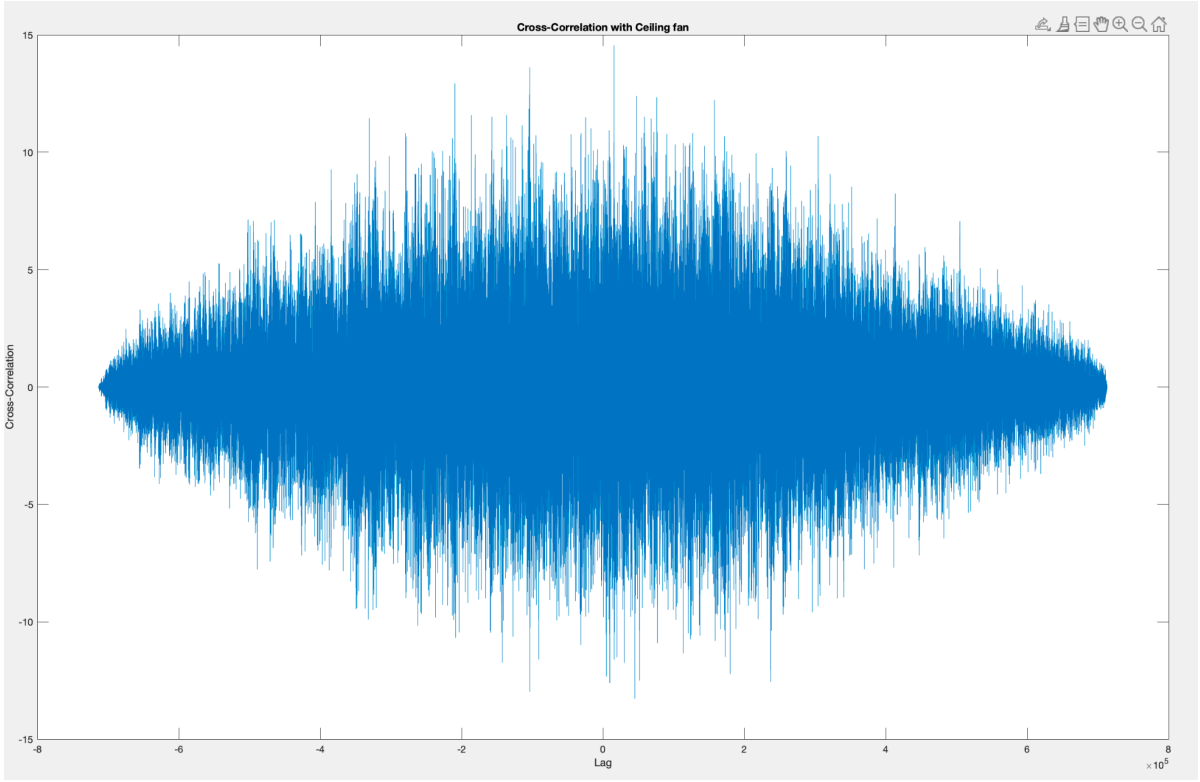

Cross-Correlation with Pressure Cooker
Cross-Correlation
Lag
20
15
10
5
0
-5
-10
-15
-20
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
×10^6

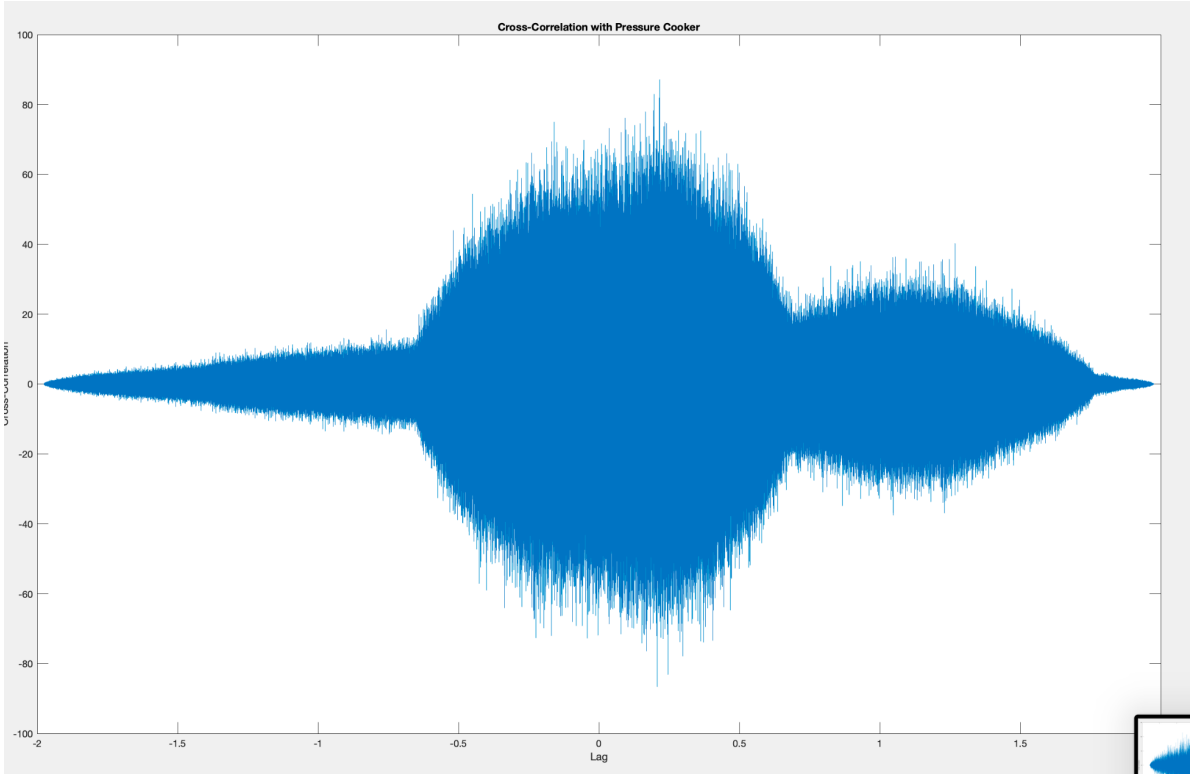


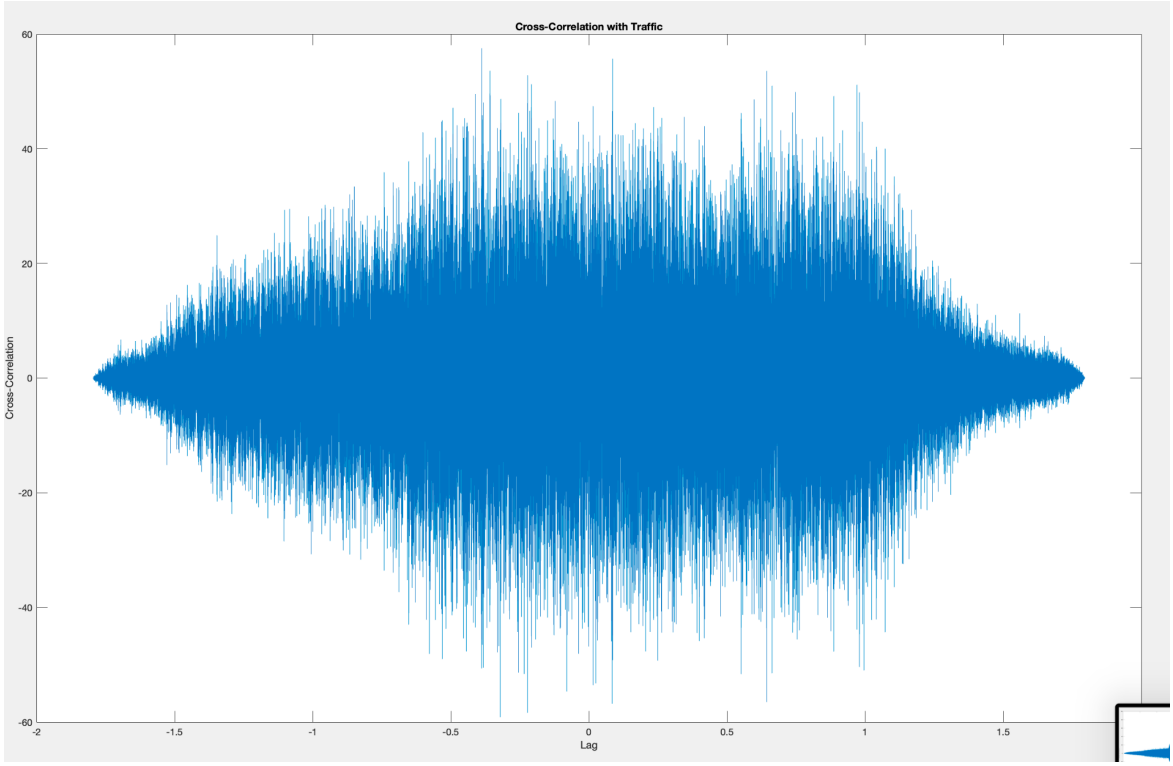
Output:

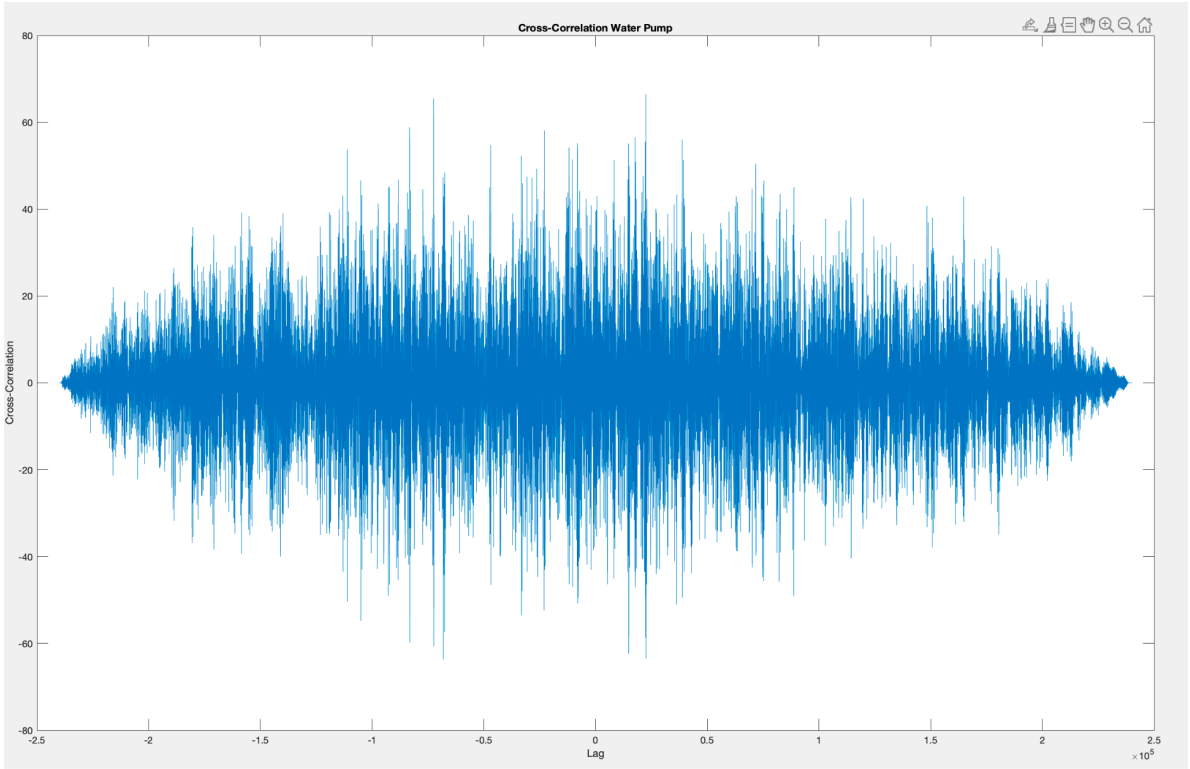
```
>> q3
    "The detected noise type is: "    "Water pump"
```

Pressure cooker classification


Cross-Correlation with Ceiling fan
Cross-Correlation
Lag
×10^5


Cross-Correlation with Pressure Cooker
Lag


Cross-Correlation with Traffic
Cross-Correlation
Lag
60
40
20
0
-20
-40
-60
-2
-1.5
-1
-0.5
0
0.5
1
1.5


Cross-Correlation Water Pump
Cross-Correlation
Lag
80
60
40
20
0
-20
-40
-60
-80
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
×10^5

Output:

```
>> q3
    "The detected noise type is: "    "Pressure Cooker"
```

3.5 Conclusion:

Thus, cross-correlating a particular audio-signal with the 4 noises and declaring the one with the maximum peak as the noise present in the audio-signal works and thus the noise is classified.

3.6 Difficulties Faced:

In this part, the major difficulty that we faced is that for classifying the sound we tried to isolate the noise which posed immense difficulties because most of the sources online suggested to use principles of Machine Learning.

End of report
