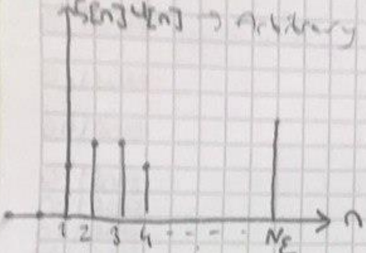
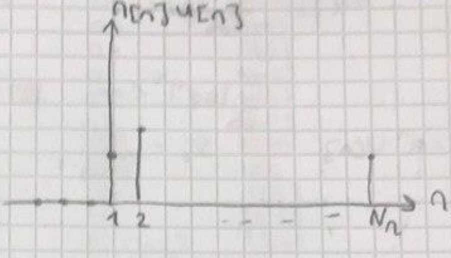
Part 1:

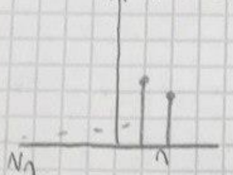
Part 1.1: we know that $u[n] = 1 \quad n > 0$
$u[n] = 0 \quad n < 0$

$\xi[n]u[n] \rightarrow$ Arbitrary





$$\psi[n] = \xi[n] * \eta[n] = \sum_{k=-\infty}^{\infty} u_k \xi[k] * \eta[n-k]\, u[n-k]$$



$\rightarrow$ This wave will shift on $\xi[k]u[k]$.

Overlap willbe exist untill $\eta[-k+n]u[-k+n]$ bigger than N_ξ. So just before the pass the value N_ξ, convolution result has an element. So the width of the convolution result is $N_\eta + N_\xi - 1$

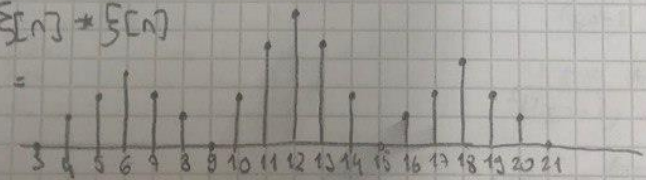
Just imagine that there is a graph which is moving accordingly index and there is a fixed graph also. Accordingly the index, the terms of graphs which are overlapping are multiplied then summed to each other.

Parb 1.2:

a)

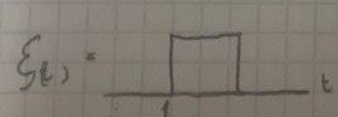
$y[n] = \xi[n] * \xi[n]$

$y[n] =$

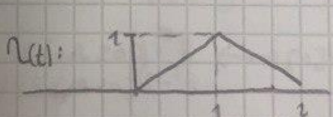


b)

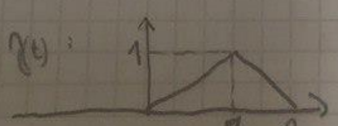
$\xi(t) =$



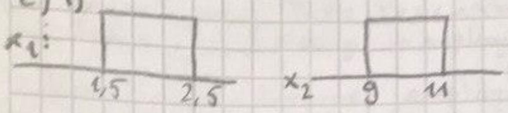
$\eta(t)$:



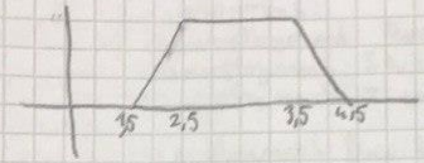
$\xi(t) * \eta(t)$:



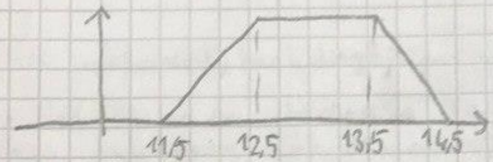
c) i)



$x_1 * x_2$



ii) same as i) just shifted 10 units



d) $f(t) = e^{-t^2/2}$ $\quad h(t) = 2e^{-t^2/2}$

$$y(t) = \int_{-\infty}^{\infty} f(t)h(t-\tau)\,d\tau = \int_{-\infty}^{\infty} e^{-t^2/2} \cdot 2\, e^{-(t-\tau)^2/2}\,d\tau$$

$$= 2\int_{-\infty}^{\infty} e^{-t^2/2 + t\tau - \tau^2}\,d\tau = 2e^{-t^2/4}\sqrt{2\pi}$$

Part 2:

For the requested functions result can be seen in Figure 2.

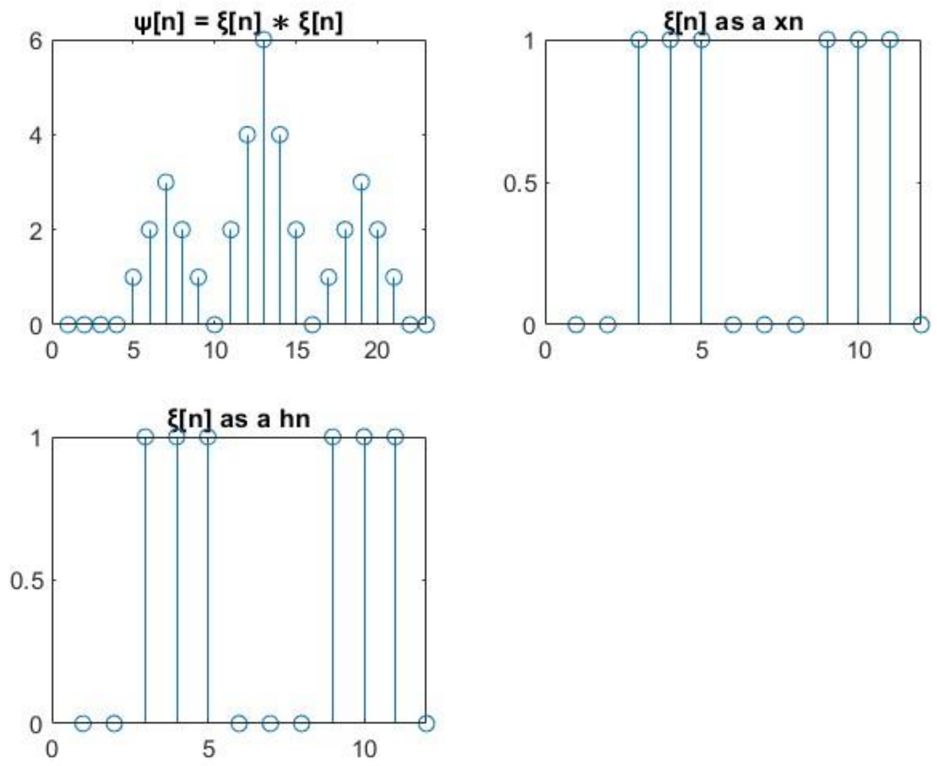


Figure 2 : Convolution Results

Part 3:

To create a convolution animation, two sequences with matching time intervals but different durations are needed. The process involves generating a time array and sampling interval, representing signals as x_n and h_n, obtaining convolution results using the ConvFUNC function, and plotting an animation. During the animation, x(n) remains stable while h(n) undergoes shifting and flipping, visualizing the convolution outcome and h(n). After the convolution is complete and the animation stops, the resulting figure is observed.

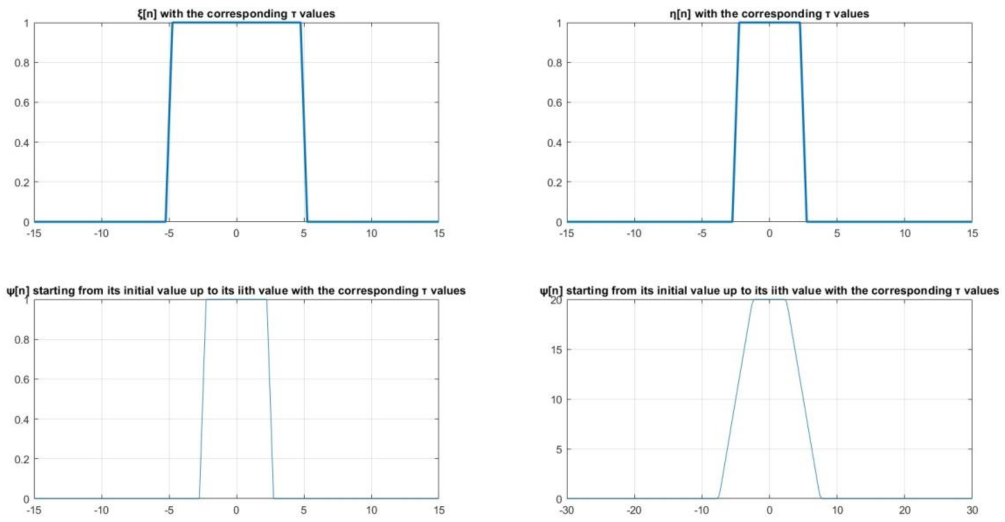


Figure 3: Convolution Animation

Part 4:

Part 4.1

For the sequences ξ[n]u[n] and η[n]u[n], a pre-processing method to obtain the cross-correlation using convolution is to flip one of the sequences. Specifically, if you flip η[n]u[n], you can perform a convolution with ξ[n]u[n], and the result will be equivalent to the cross-correlation of the original sequences. In discrete convolution, the output length is $N\xi + N\eta - 1$. In discrete cross-correlation, the output length is also $N\xi + N\eta - 1$.

Part 4.2

My ID number is 22001799. From here I calculated n1 =0 and n2=4 and sampling rate is 8192 as given.

For naked n1:

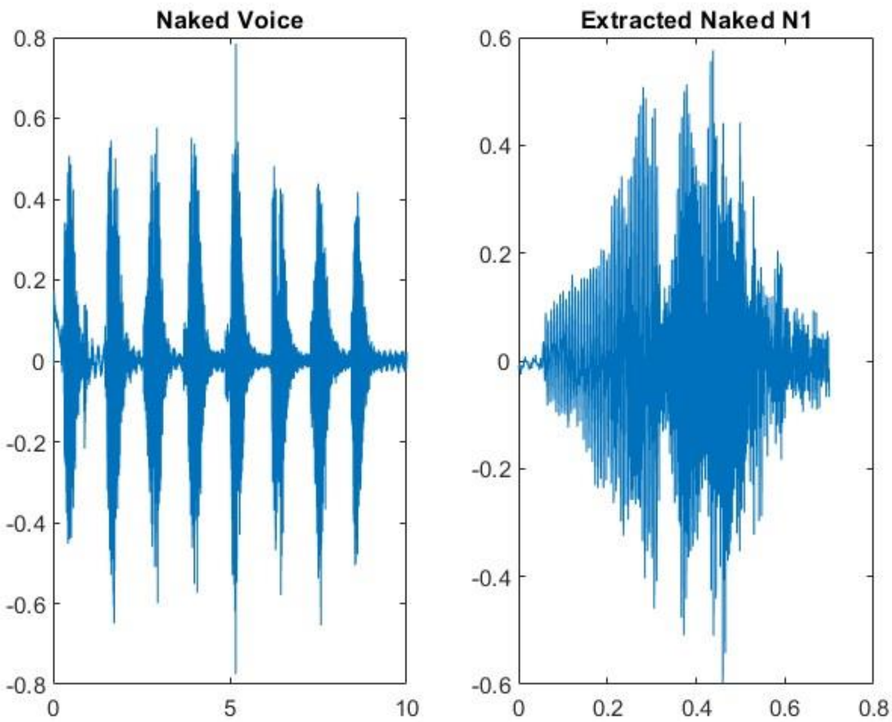


Figure 4: Graph of Robot Voice and Extracted Naked N1

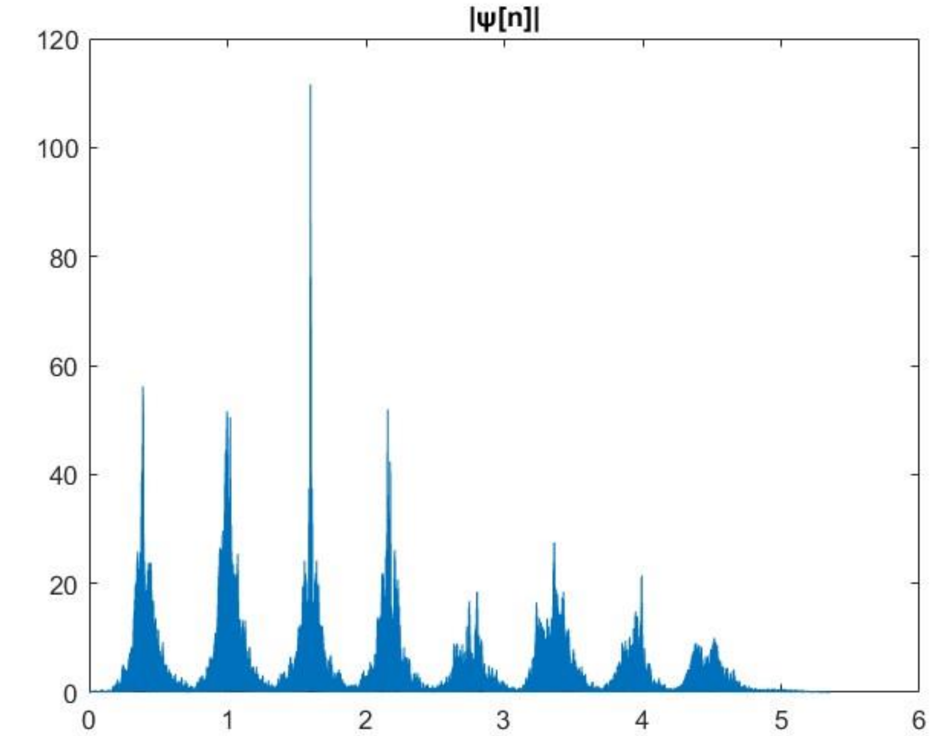


Figure 5: Graph of W[n]

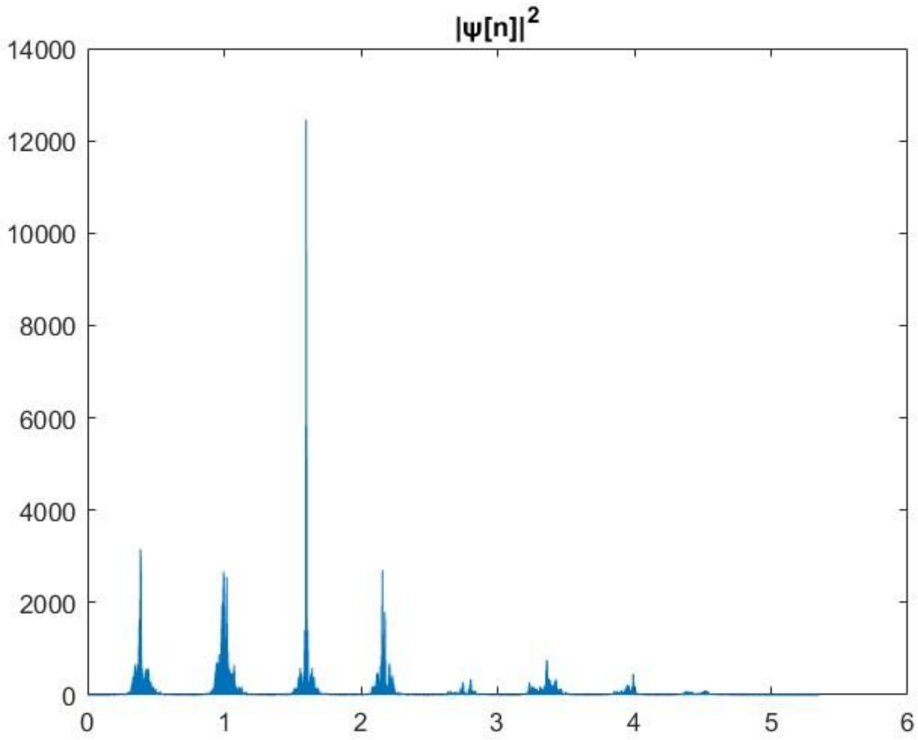


Figure 6: Graph of W[n]^2

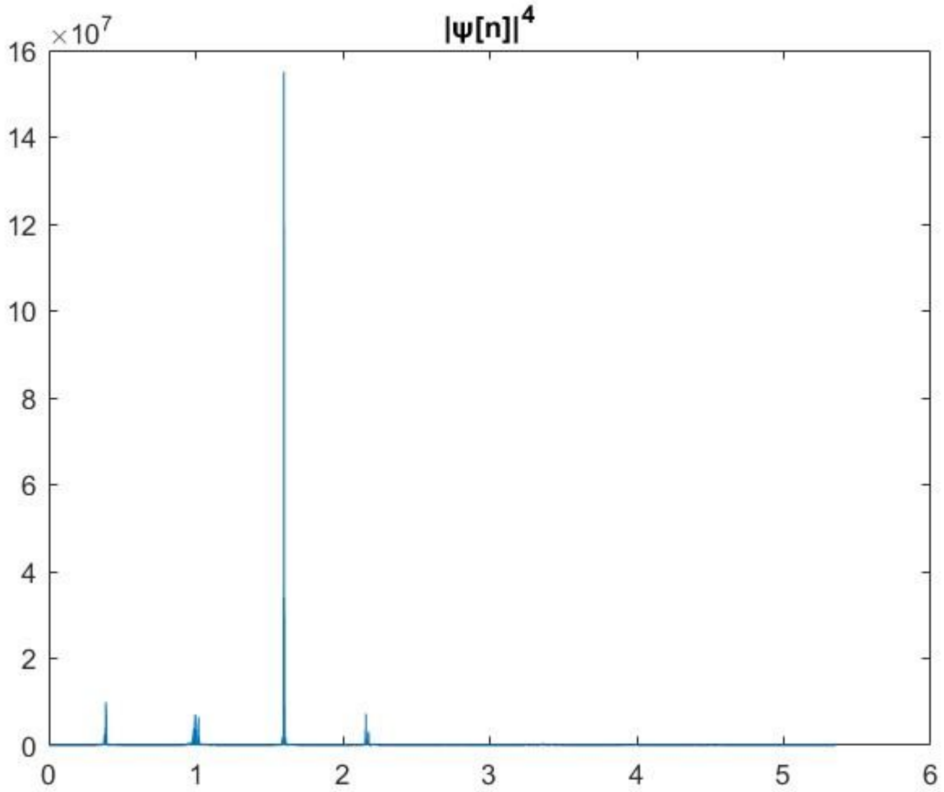


Figure 7: Graph of W[n]^4

Examining the data, it becomes evident that the most prominent peak corresponds to the n1 which is 0 of the speaker's ID, indicating a self-correlation of the signal. The second most significant peak aligns with the n2 (4) of the ID, albeit with a reduced amplitude, attributed to slight differences in the pronunciation.

For python based n1:

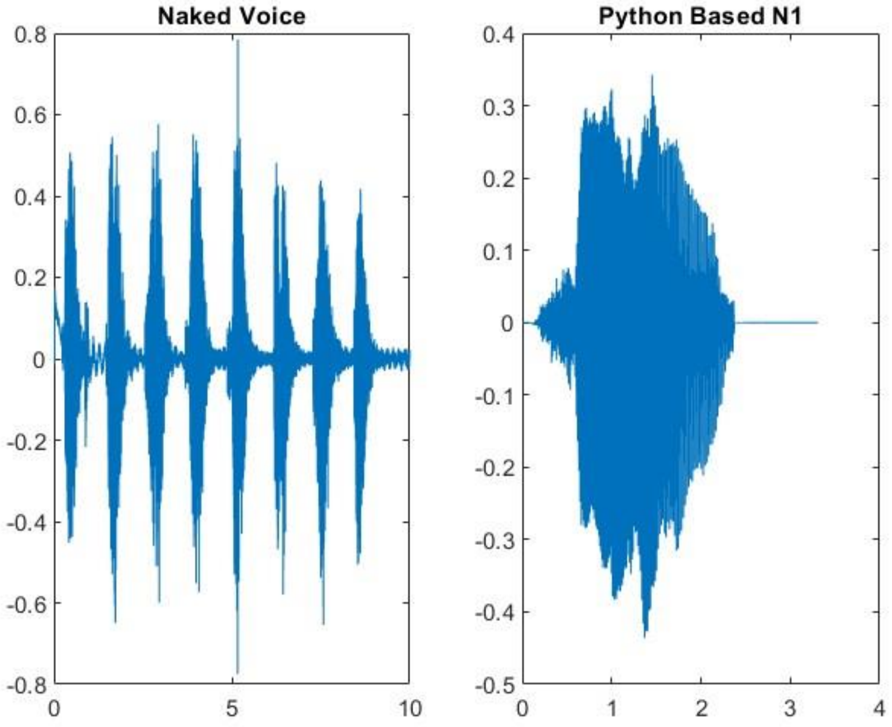


Figure 8: Graph of Robot Voice and Python Based N1

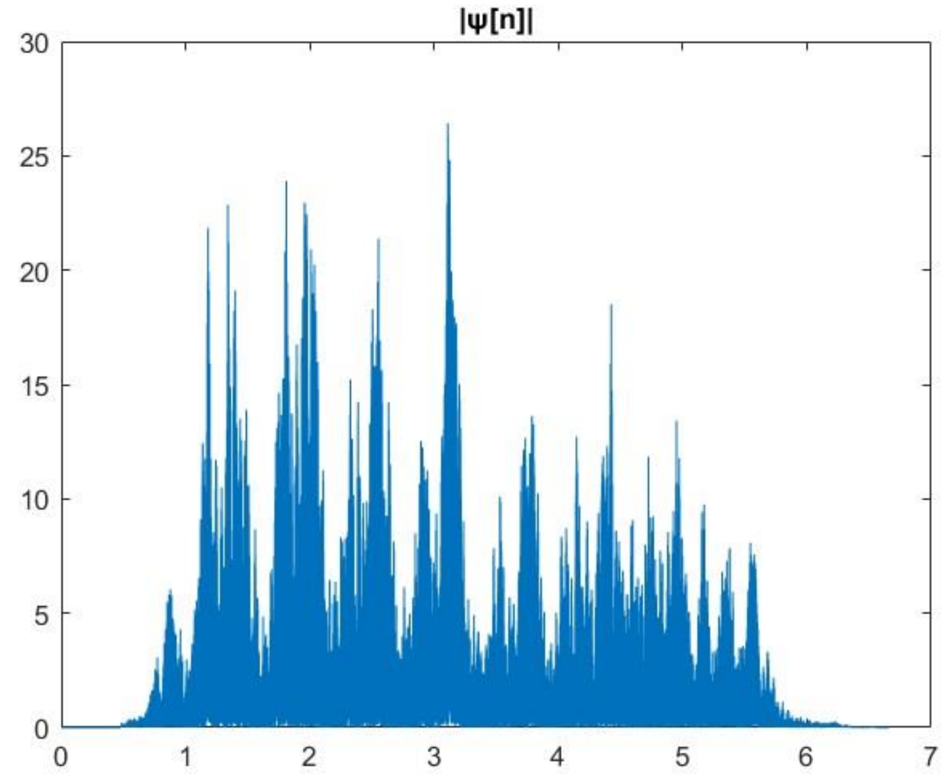


Figure 9: Graph of W[n]

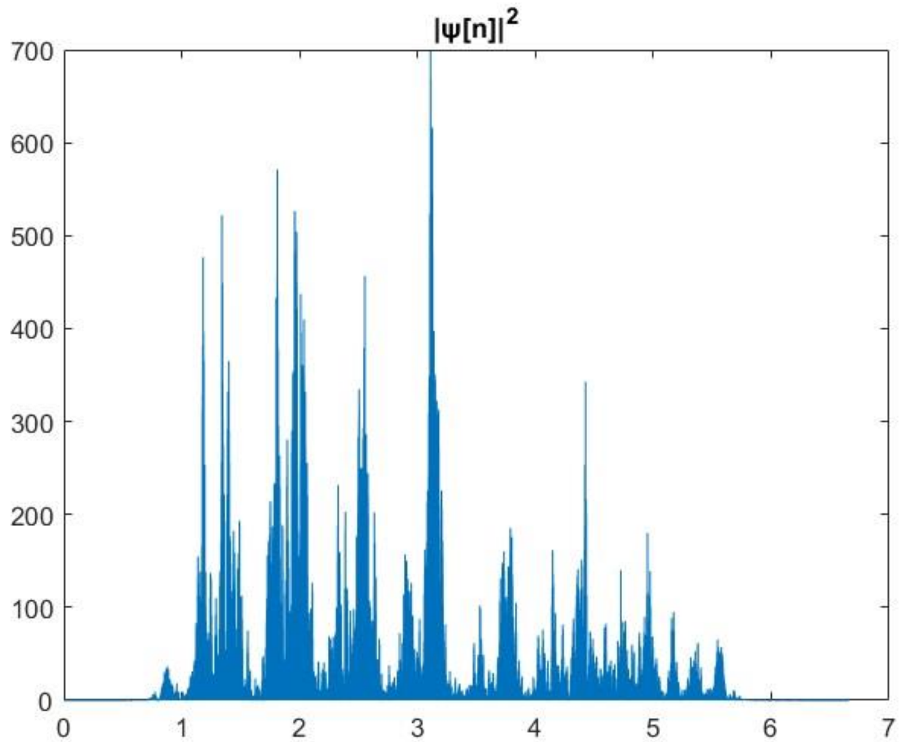


Figure 10: Graph of W[n]^2

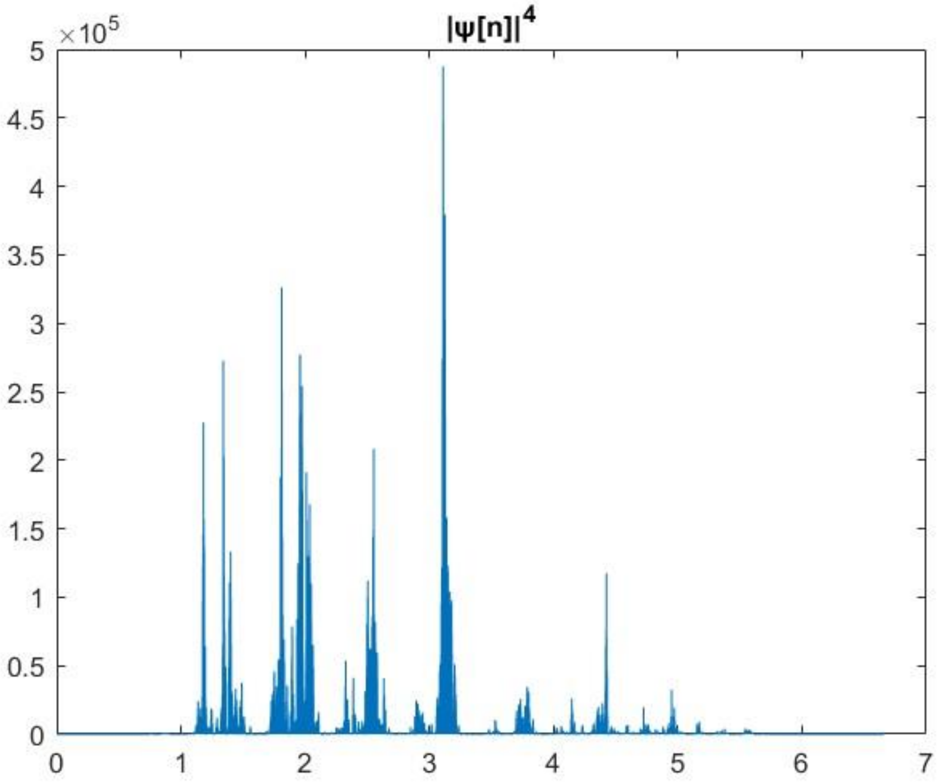


Figure 11: Graph of W[n]^4

For python based n2:

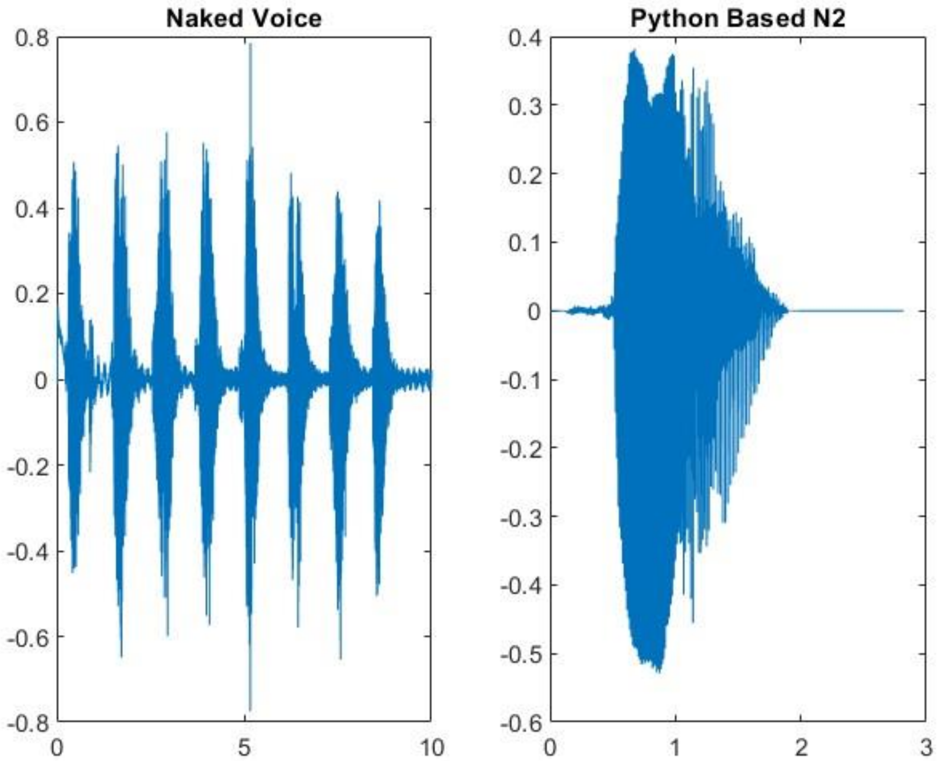


Figure 12: Graph of Robot Voice and Python Based N2

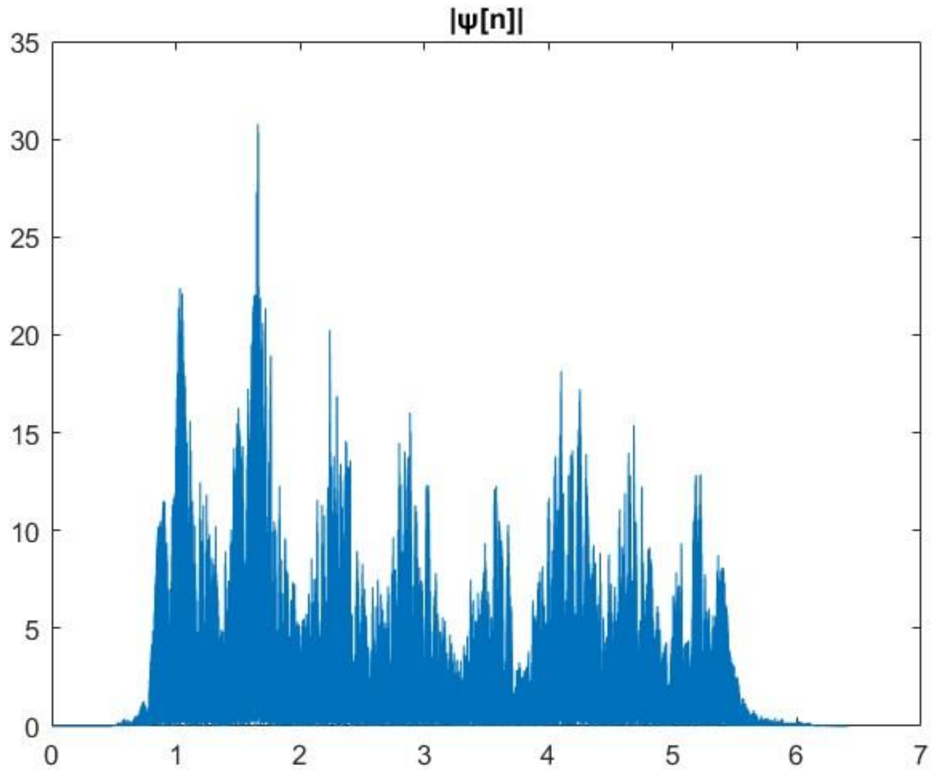


Figure 13: Graph of W[n]

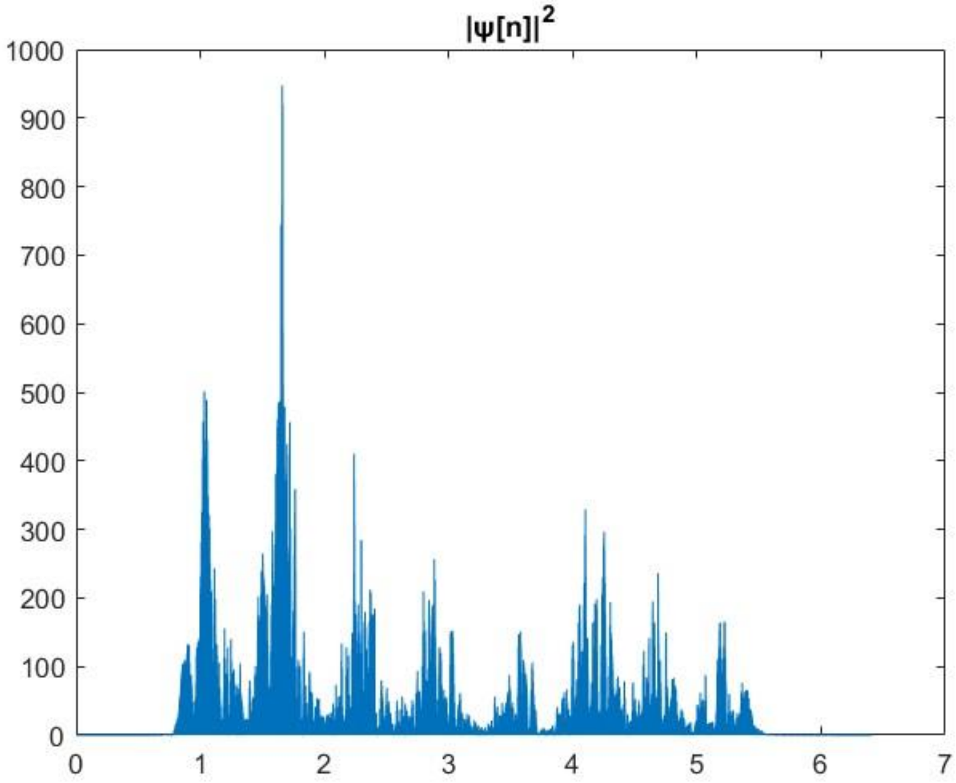


Figure 14: Graph of W[n]^2

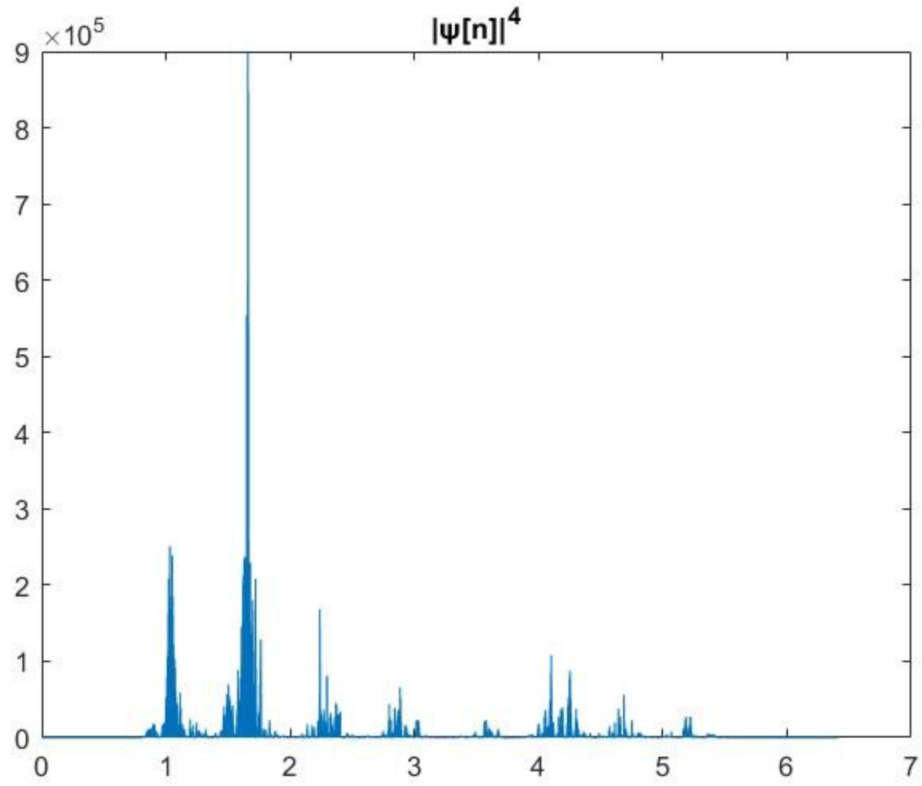


Figure 15: Graph of W[n]^4

Part 5:

SNR can be calculated with the formula below, and the outcome of the necessary processes is as shown in Figure 16.

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}$$

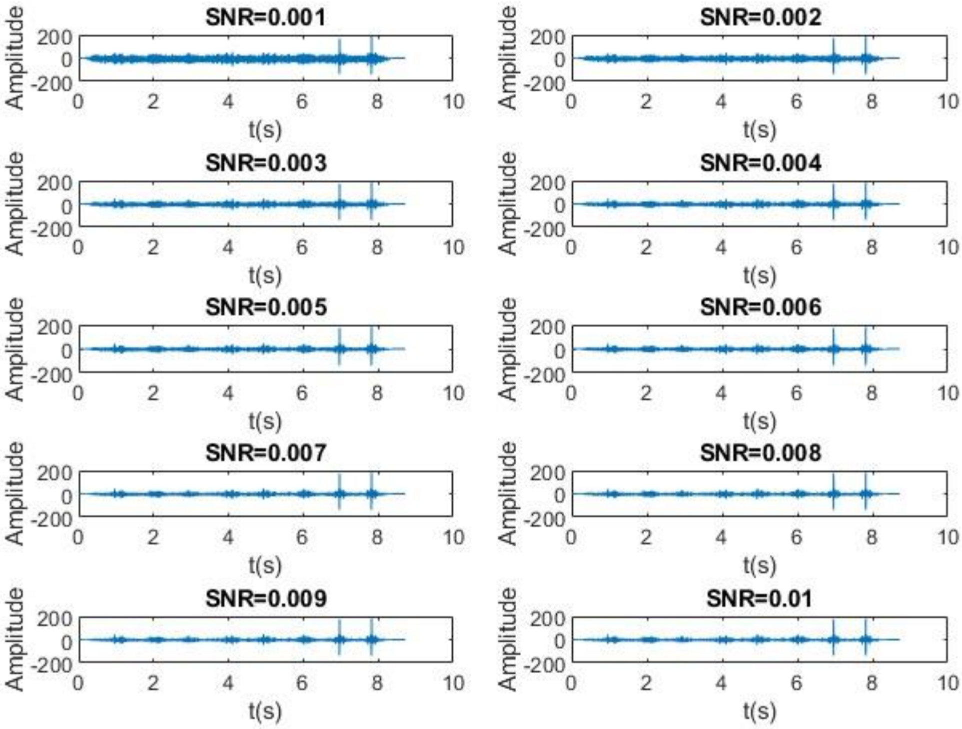


Figure 16: SNR

Appendices:

Part 2:

```
x = [0 0 1 1 1 0 0 0 1 1 1 0]
n = 1:1:length(x)
result = ConvFUNC(x,x)
figure;
subplot(2,2,1)
stem(result)
title ('ψ[n] = ξ[n] ∗ ξ[n]')

subplot (2,2,2)
stem(x)
title ('ξ[n] as a xn');
subplot (2,2,3)
```

```
stem(x)
title ('ξ[n] as a hn')


function [y] = ConvFUNC(x,h)
Xy = length(x) + length(h) -1
h = fliplr(h)
y = zeros(1,Xy)
trf = conv(x,h)

    for n = 1:Xy
        kmin = max(1, n - length(h) + 1);
        kmax = min(n, length(x));
        y(n) = sum(x(kmin:kmax) .* h(1:(kmax - kmin + 1)));
        y = trf

    end
end
```

Part 3:

```
tao = -15:0.25:15
et = heaviside(tao + 5) - heaviside(tao-5);
nt = heaviside(tao + 2.5) - heaviside(tao -2.5)
seytan= ConvFUNC(et,nt)
flippednt = fliplr(nt)
sizeconv = -30:0.25:30

subplot(2,2,1)
plot(tao, et, 'LineWidth', 2);
title('ξ[n] with the corresponding τ values');
grid on;


subplot(2,2,2)
plot(tao, nt, 'LineWidth', 2);
title('η[n] with the corresponding τ values');
grid on;

subplot(2,2,4)
plot(sizeconv,seytan)
title('ψ[n] starting from its initial value up to its iith value with the
corresponding τ values');
grid on;
%,
%• η[n] flipped and shifted right before ξ[n] with the corresponding τ values,
for ii = 1:(length(et)+length(nt)-1)
    flippednt = circshift(flippednt, ii)
    subplot(2,2,3)
    plot(tao, flippednt)
    pause(0.1)
    title('ψ[n] starting from its initial value up to its iith value with the
corresponding τ values');
```

```
    grid on;
end
```

Part 4:

For naked n1:

```
SR = 8192;
ooversr = 1/SR
twoversr = ooversr/2
voiceFull = transpose(audioread('/TotalNumber.flac'));
mynakedvoice = transpose(audioread('/voice.flac'));
mynakedvoice(length(mynakedvoice)+1)=0
sab = mynakedvoice
mynakedvoice = fliplr(mynakedvoice)
sasa = mynakedvoice
exn1starttime = 2.5
exn1endtime = 3.2
extractedn1 = sab(exn1starttime * SR :exn1endtime * SR);
ccof = fliplr(ConvFUNC(extractedn1, sasa));

figure();
Fs = 0:ooversr:10;
subplot(1,2,1);
plot(Fs, sab);

title('Naked Voice');

subplot(1,2,2);
plot(Fs(1:length(extractedn1)), extractedn1);

title('Extracted Naked N1');


figure();
FsCross = 0:twoversr:10;
plot(FsCross(1:length(ccof)), abs(ccof));

title('|ψ[n]|');

figure();
plot(FsCross(1:length(ccof)), abs(ccof).^2);

title('|ψ[n]|^2');


figure();
plot(FsCross(1:length(ccof)), abs(ccof).^4);

title('|ψ[n]|^4');
```

For python based n1:

Just extractedn1 changed with python based voice. That's why changed part of code is added.(Body part)

```
SR = 8192;
ooversr = 1/SR
twoversr = ooversr/2
voiceFull = transpose(audioread('/TotalNumber.flac'));
mynakedvoice = transpose(audioread('/voice.flac'));
mynakedvoice(length(mynakedvoice)+1)=0
sab = mynakedvoice
mynakedvoice = fliplr(mynakedvoice)
sasa = mynakedvoice
exn1starttime = 2.5
exn1endtime = 3.2
extractedn1 = sab(exn1starttime * SR :exn1endtime * SR);
robon1 = transpose(audioread('/0.flac'));
ccof = fliplr(ConvFUNC(robon1, sasa));
```

For python based n2:

```
SR = 8192;
ooversr = 1/SR
twoversr = ooversr/2
voiceFull = transpose(audioread('/TotalNumber.flac'));
mynakedvoice = transpose(audioread('/voice.flac'));
mynakedvoice(length(mynakedvoice)+1)=0
sab = mynakedvoice
mynakedvoice = fliplr(mynakedvoice)
sasa = mynakedvoice
exn1starttime = 2.5
exn1endtime = 3.2
extractedn1 = sab(exn1starttime * SR :exn1endtime * SR);
robon2 = transpose(audioread('/4.flac'));
ccof = fliplr(ConvFUNC(robon2, sasa));
```