Department of Electronics and
Electrical Communications Engineering
Faculty of Engineering - Cairo University

ELC_3070

Project #1

Name	Section	Bench Number	ID
ايه اشرف محمد حسن احمد	1	48	9202359
محمد حسام محمد طه	3	27	9203204
محمد خالد محمد الاحمدى ابراهيم	3	31	9203226
محمد علاء الدين محمد الرفاعى ابراهيم عشماوى	3	41	9203300
يارا جوده احمد جوده	4	42	9203703
يارا محمد حسن احمد	4	43	9203706

Table of Contents

Table of Figures

1. Code Explanation

Generating the ensemble

For this part of the code, we generated the ensemble (500 realizations), and determined the number of bits per realization (100 bits) and the number of samples per bit (7 samples), so we ended up with 700 samples per bit. Then, we generated a zero matrix with these dimensions, and to produce the different line codes, we created another function with the name “Generate Random Data”.

```
A=4;                                                          % Amplitude Value
number_of_realizations=500;                                   % total number of Realizations
number_of_bits=100;                                           % number of bits in each Realization
number_of_samples_per_bit=7;                                  % number of samples for each bit
number_of_samples=number_of_bits*number_of_samples_per_bit;   % total number of samples for each Realization
Total_Transmitted_Data=zeros(number_of_realizations, number_of_samples-1);  % hold transmitted data
for i = 1 : number_of_realizations
    if i == 1
        [Total_Transmitted_Data] = Generate_Random_Data();
else
    Total_Transmitted_Data = [Total_Transmitted_Data; Generate_Random_Data()];
    end
end
```

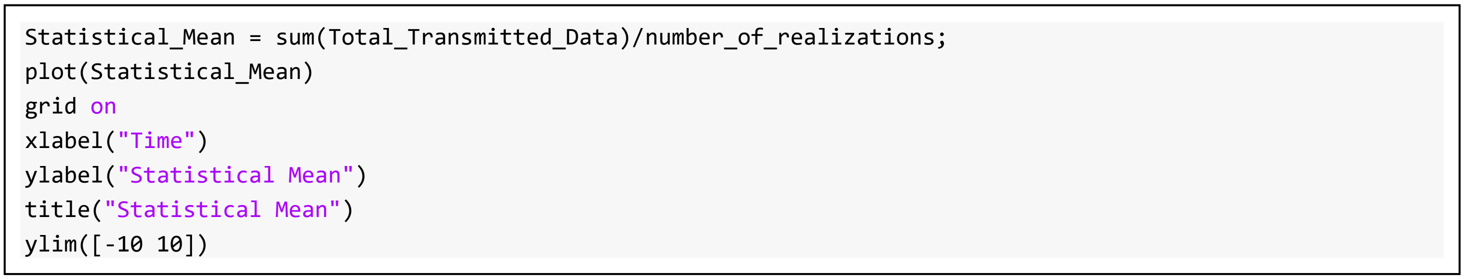
Generate Random Data

The function “Generate Random Data” is generic, with a variable “choose”, to be changed according to the needed line code. We used the function “randi” to generate random “0”and “1”, then we perform some operations on these values to obtain the right amplitude for each line code. Finally, we generated random values between 0 and 6 for the random start and we concatenated it with the data.

```
function data_transmitted = Generate_Random_Data()
A=4;                                        % Amplitude Value
number_of_bits=100;                         % number of bits in each realization
number_of_samples=7;                        % number of samples for each bit
data=randi(2,[1,number_of_bits+1])-1;       % generate 1x100 random numbers '0' or '1'
choose = 1;            % choose which line code to use (1. polar NRZ, 2. unipolar NRZ, 3. polar RZ)
switch choose
    case 1
        data=(2*data-1)*A;                        % mapping to A & -A (Polar NRZ)
        data=repmat(data, number_of_samples, 1);  % repeat each bit 7 times to sample DAC every 10ms
    case 2
        data=data*A;                             % mapping to A & 0  (Unipolar NRZ)
        data=repmat(data, number_of_samples, 1); % repeat each bit 7 times to sample DAC every 10ms
    case 3
        data   =  (2*data-1)*A;
        data = repmat(data, 4 ,1);
        data = [data; zeros(3, number_of_bits+1)];    %(Polar RZ)
    otherwise
        disp("invalid line code choose");
end
data=data(:);                               % convert data to column vector 700x1
td=randi(number_of_samples)-1;              % generate random number from 0 to 6
data_transmitted = (data(td+1:700+td))';    % window data from td to 700+td (700 samples)
end
```

- ## Calculate The Statistical Mean

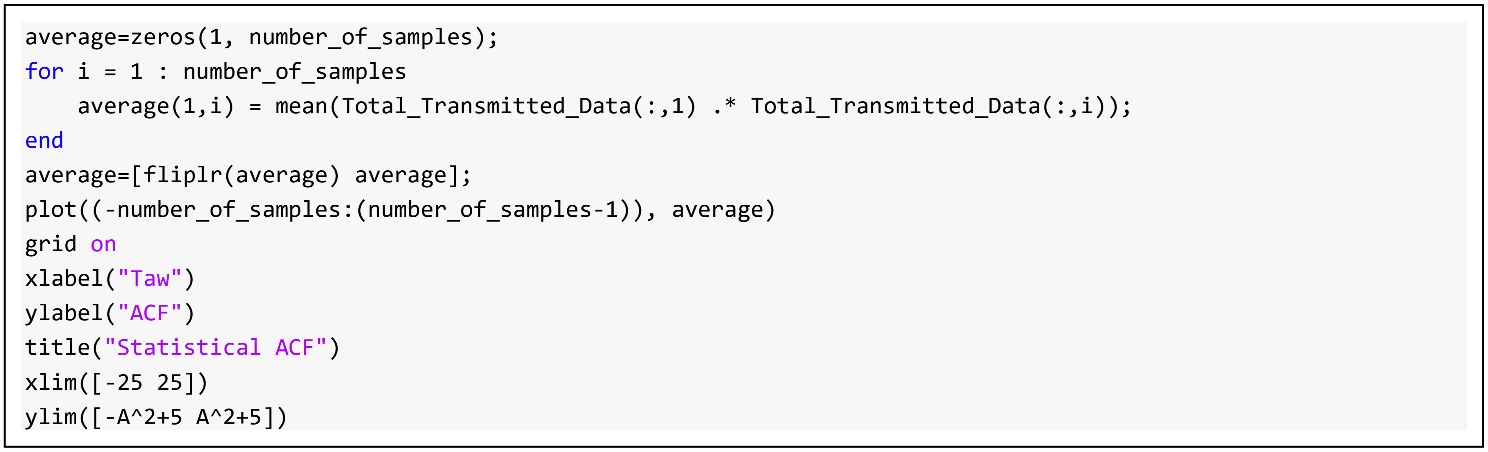
To compute the Statistical mean, we simply used the “sum” function, which sums the columns (different realizations) and divided by the number of realizations. Finally, we plotted the graph.



```
Statistical_Mean = sum(Total_Transmitted_Data)/number_of_realizations;
plot(Statistical_Mean)
grid on
xlabel("Time")
ylabel("Statistical Mean")
title("Statistical Mean")
ylim([-10 10])
```

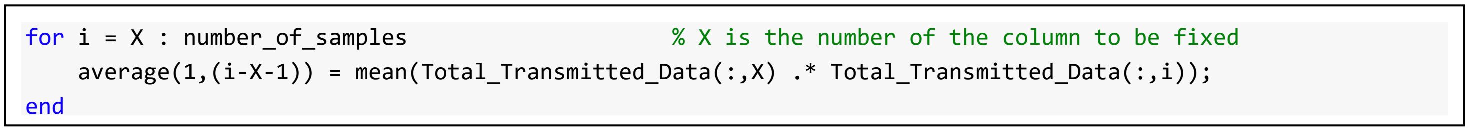
- ## Calculate Statistical Autocorrelation Function

To compute the statistical autocorrelation function, we got the average of the multiplication of the first column (same sample across all realizations) by all other columns, and we plotted it.



```
average=zeros(1, number_of_samples);
for i = 1 : number_of_samples
    average(1,i) = mean(Total_Transmitted_Data(:,1) .* Total_Transmitted_Data(:,i));
end
average=[fliplr(average) average];
plot((-number_of_samples:(number_of_samples-1)), average)
grid on
xlabel("Taw")
ylabel("ACF")
title("Statistical ACF")
xlim([-25 25])
ylim([-A^2+5 A^2+5])
```

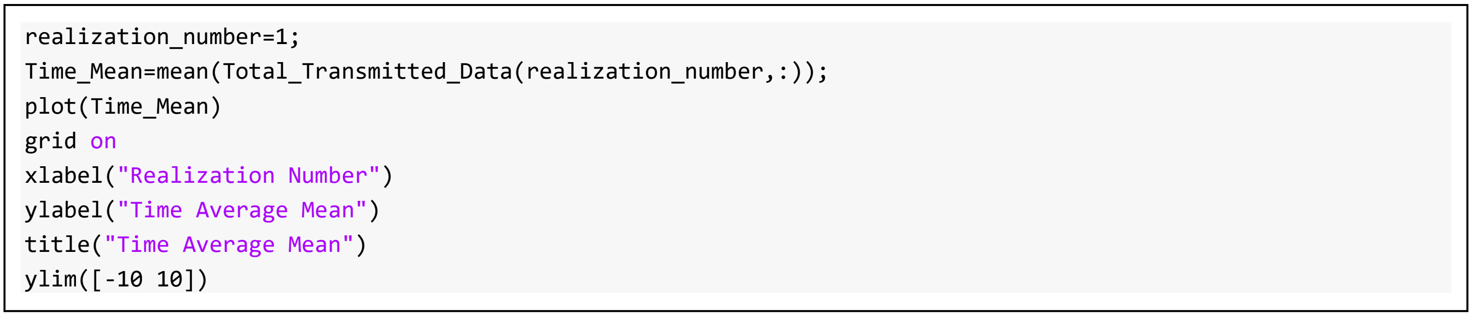
And to prove that the process is stationary, we changed the fixed column (first column in the above code), and perform the same computation as follows:



```
for i = X : number_of_samples                    % X is the number of the column to be fixed
    average(1,(i-X-1)) = mean(Total_Transmitted_Data(:,X) .* Total_Transmitted_Data(:,i));
end
```

- ## Calculate The Time Mean

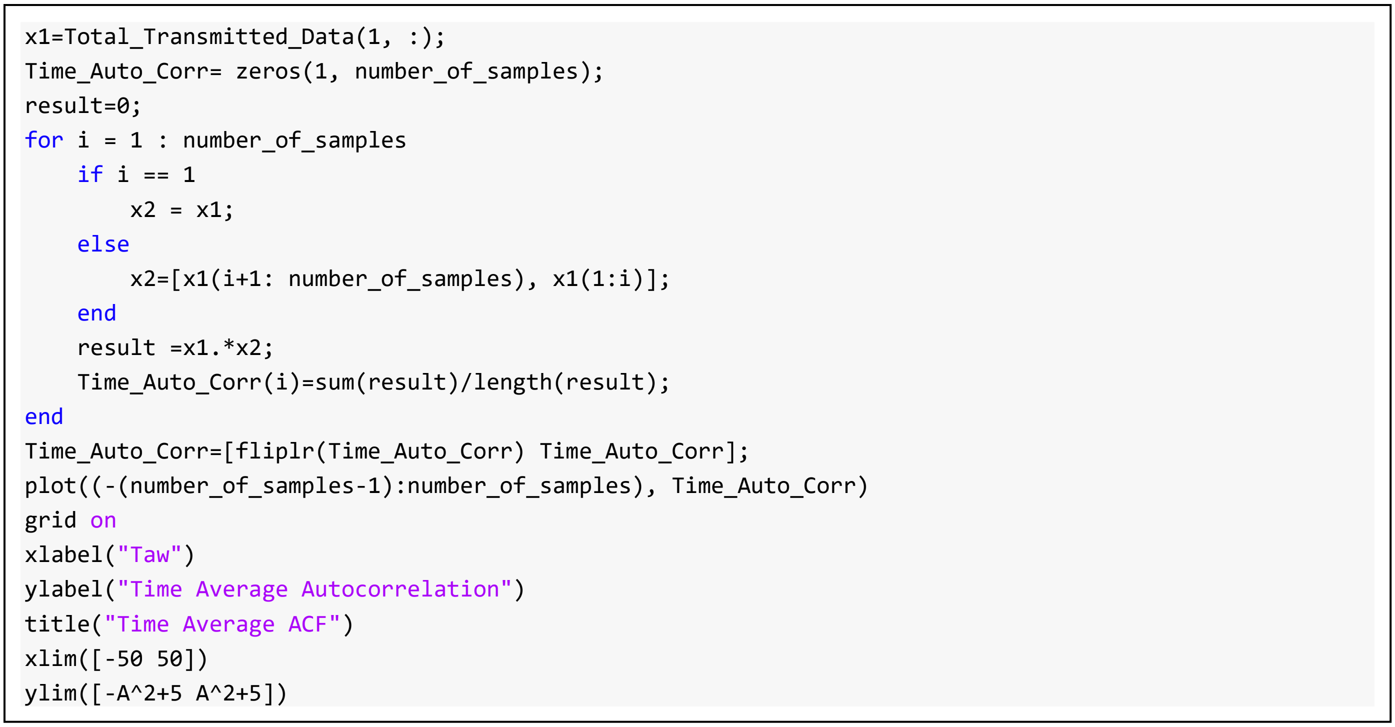
To compute the time mean, we obtained the average of one row (one realization) and then plotted it.



```
realization_number=1;
Time_Mean=mean(Total_Transmitted_Data(realization_number,:));
plot(Time_Mean)
grid on
xlabel("Realization Number")
ylabel("Time Average Mean")
title("Time Average Mean")
ylim([-10 10])
```

- ## Calculate The Time Autocorrelation Function

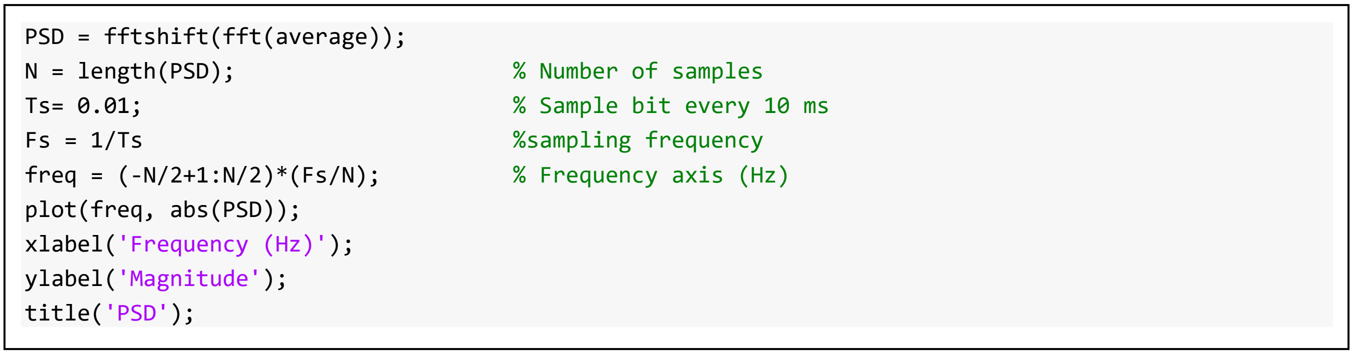
To compute the time autocorrelation function, we took the first row (first realization), then hold its values in a vector, then keep on shifting the same first row and multiply it by the fixed main version and then we obtained the average of this calculation.



```
x1=Total_Transmitted_Data(1, :);
Time_Auto_Corr= zeros(1, number_of_samples);
result=0;
for i = 1 : number_of_samples
    if i == 1
        x2 = x1;
    else
        x2=[x1(i+1: number_of_samples), x1(1:i)];
    end
    result =x1.*x2;
    Time_Auto_Corr(i)=sum(result)/length(result);
end
Time_Auto_Corr=[fliplr(Time_Auto_Corr) Time_Auto_Corr];
plot((-(number_of_samples-1):number_of_samples), Time_Auto_Corr)
grid on
xlabel("Taw")
ylabel("Time Average Autocorrelation")
title("Time Average ACF")
xlim([-50 50])
ylim([-A^2+5 A^2+5])
```

- ## Power Spectral Density

To be able to observe the bandwidth, we plotted the Fourier Transform of the Autocorrelation function.



```
PSD = fftshift(fft(average));
N = length(PSD);                        % Number of samples
Ts= 0.01;                               % Sample bit every 10 ms
Fs = 1/Ts                               %sampling frequency
freq = (-N/2+1:N/2)*(Fs/N);             % Frequency axis (Hz)
plot(freq, abs(PSD));
xlabel('Frequency (Hz)');
ylabel('Magnitude');
title('PSD');
```

2. Unipolar NRZ Line Code

a. Transmitted Waveform

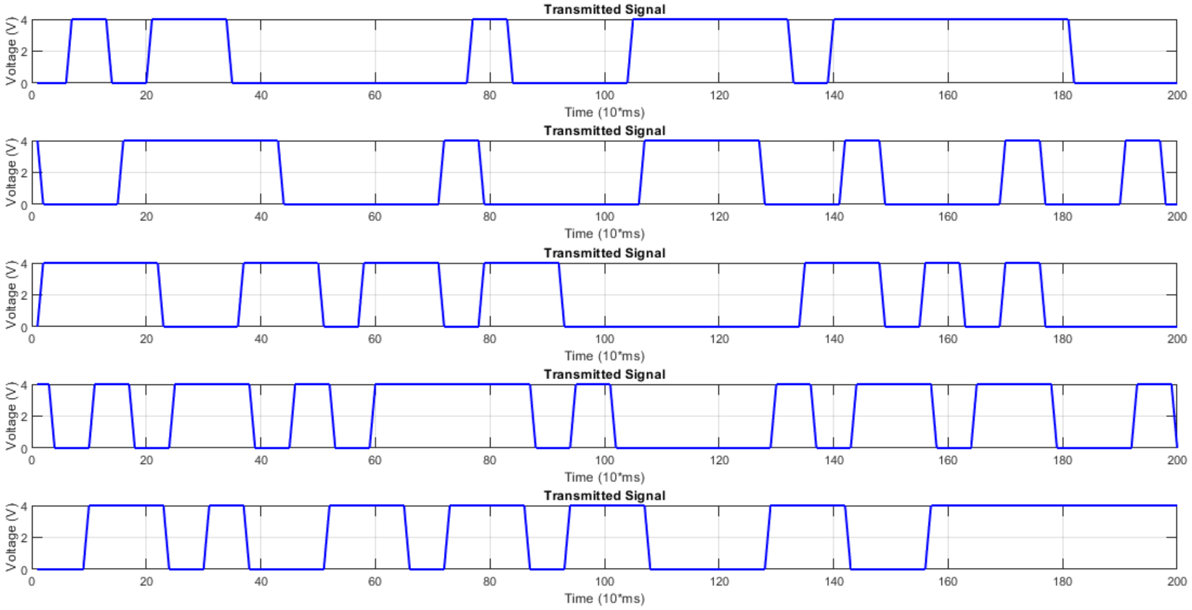


Figure 1 Transmitted Waveform for Unipolar line code

b. Figures

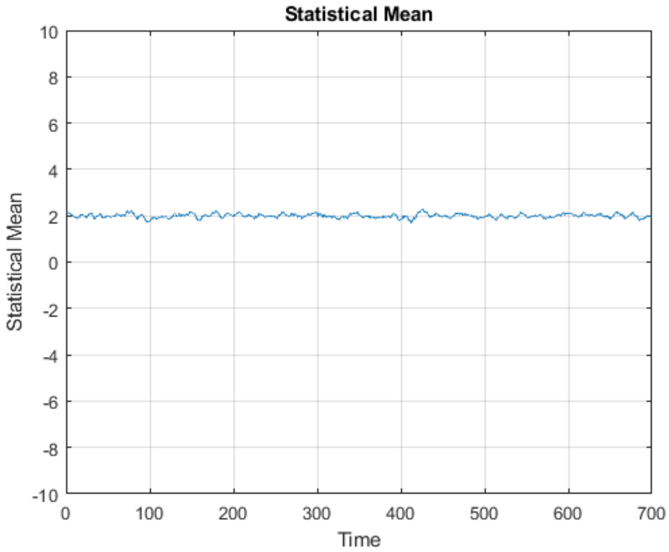


Figure 2 Unipolar NRZ Statistical Mean

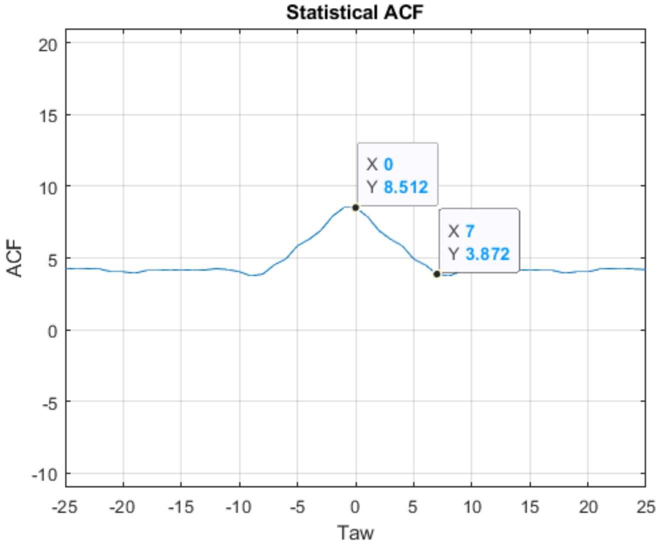


Figure 3 Unipolar NRZ Statistical ACF (computed while column 1 is fixed)

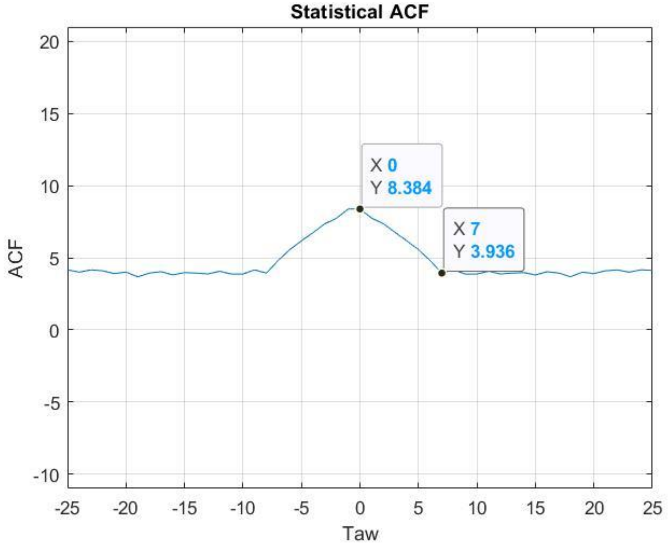
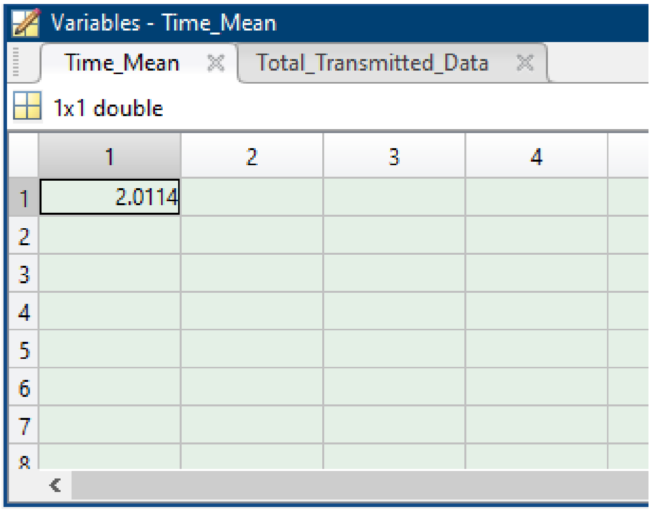


figure a 1 Unipolar NRZ Statistical ACF (computed while column 2 is fixed)



Variables - Time_Mean

Time_Mean | Total_Transmitted_Data

1x1 double

	1	2	3	4
1	2.0114			
2				
3				
4				
5				
6				
7				
8				

Figure 4 Unipolar NRZ Time Mean

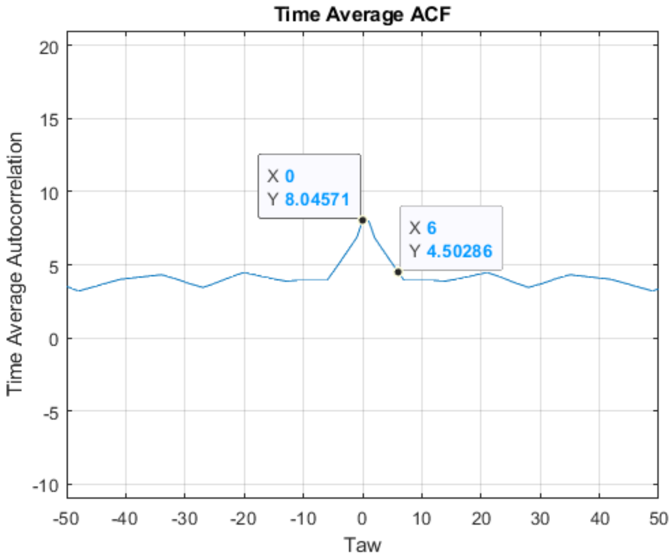


Figure 5 Unipolar NRZ Time ACF

c. Discussion

• Is the random process Stationary?

YES, after computing the statistical mean and the statistical autocorrelation, it turns out that the process is stationary, as the mean is constant across time (figure 1) , and the autocorrelation function is function in time difference only whenever the two samples taken lie in the same bit, otherwise the autocorrelation function output is zero (figure 2), and this was verified by fixing different columns (sample across all realizations) and computing the multiplication by all other columns and averaging the result and getting the same result (figure 2 and figure a 1).

• Is the random process ergodic?

YES, the process is ergodic, as the statistics across the time are the same as across the ensemble: the time mean (computed from one realization) is equal to the statistical mean (figure 3) , and the time autocorrelation function is the same as the statistical autocorrelation function (figure 4).

• What is the Bandwidth of the transmitted signal?

The bandwidth of the transmitted signal is 14.2 Hz. In this case, due to the DC shift in the Autocorrelation function, we have a delta at zero.

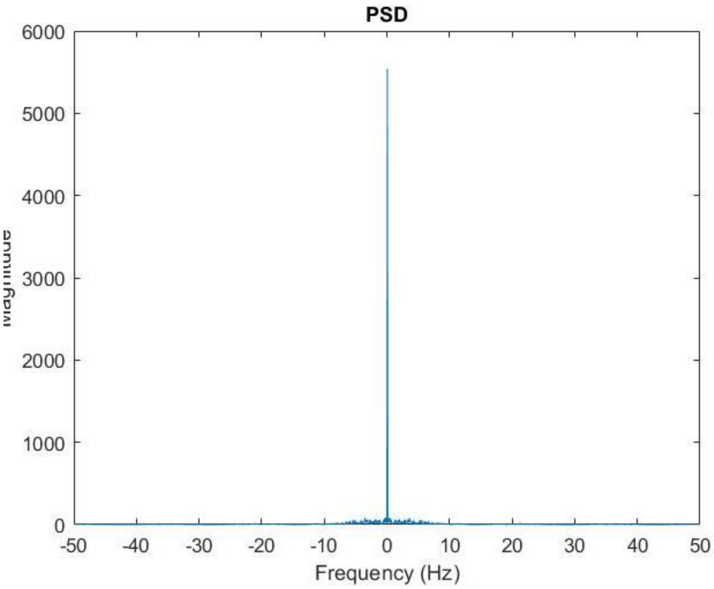


Figure 6 Unipolar PSD

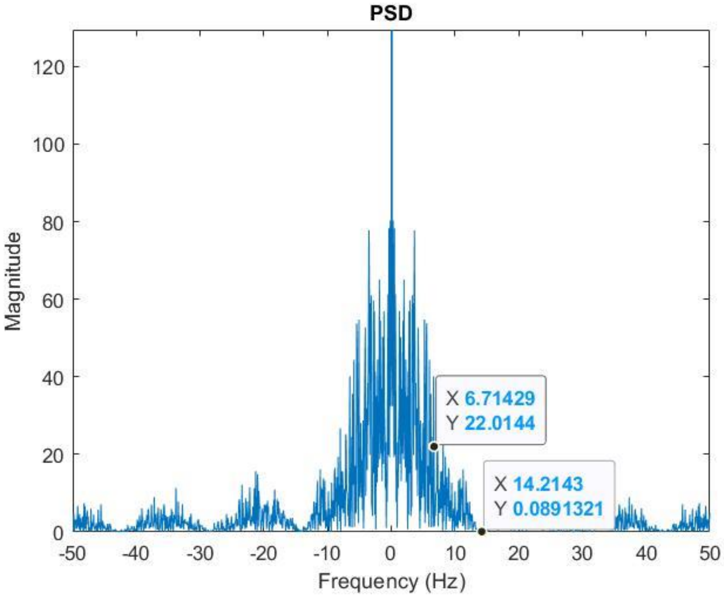


Figure 7 Unipolar PSD (Vertically Zoomed)

3. Polar NRZ Line Code

a. Transmitted Waveform

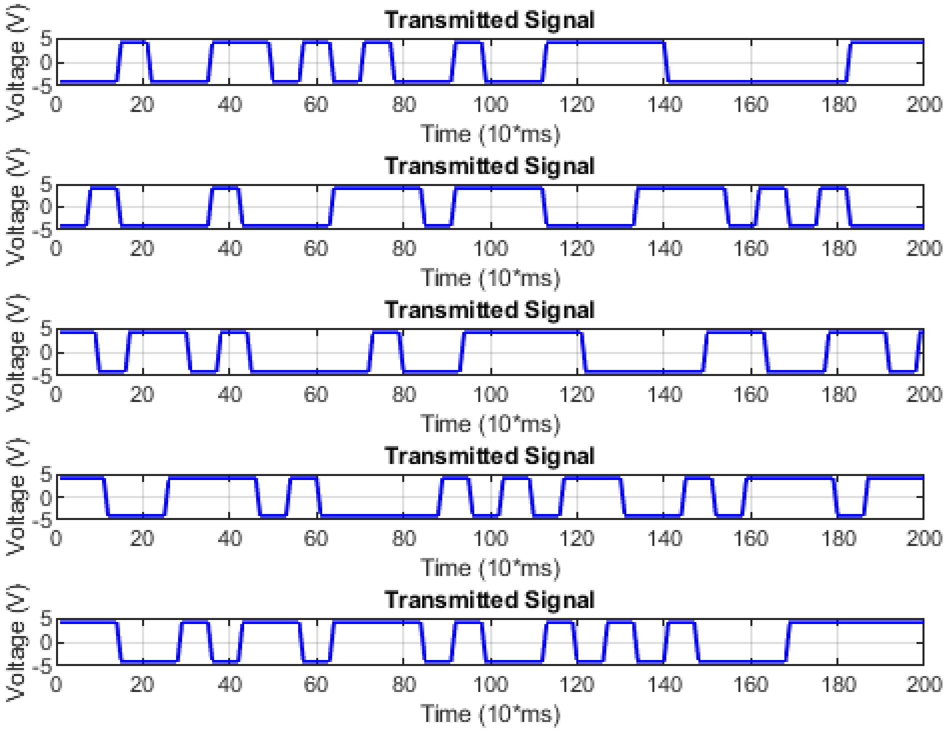


Figure 8 Transmitted Waveform Polar NRZ line code

b. Figures

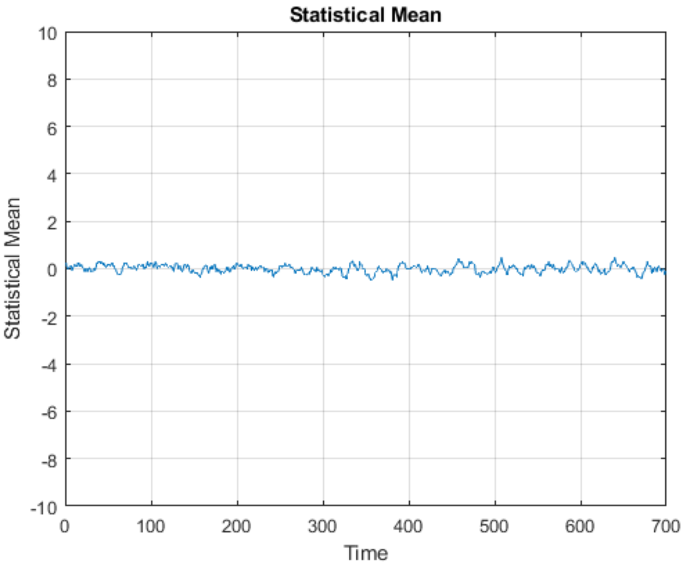


Figure 10 Polar NRZ statistical Mean

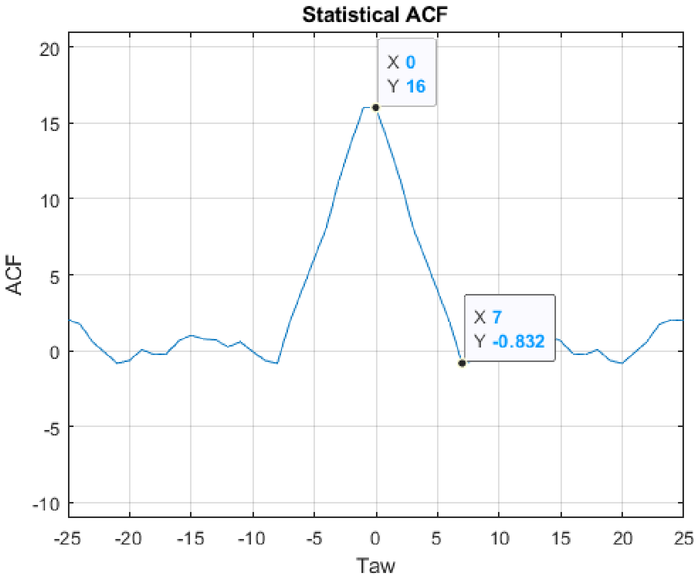


Figure 9 Polar NRZ Statistical ACF (computed while column 1 is fixed)

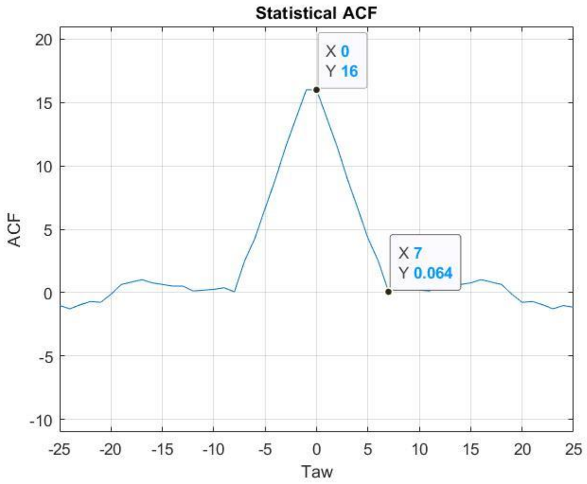
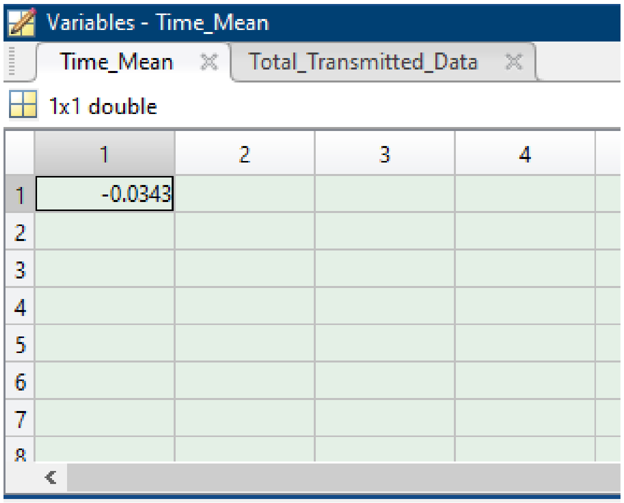


figure a 2 Polar NRZ Statistical ACF (computed while column 2 is fixed)



Variables - Time_Mean

Time_Mean | Total_Transmitted_Data

1x1 double

	1	2	3	4
1	-0.0343			
2				
3				
4				
5				
6				
7				
8				

Figure 11 Polar NRZ Time Mean

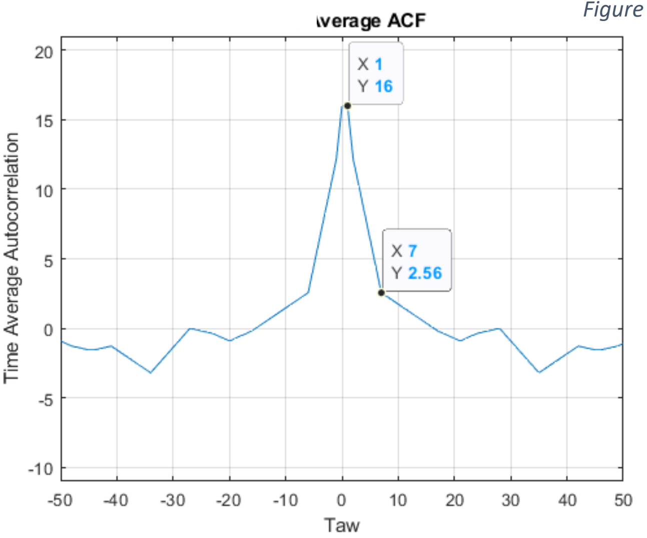


Figure 12 Polar NRZ Time ACF

c. Discussion

- ### Is the random process Stationary?

YES, after computing the statistical mean and the statistical autocorrelation, it turns out that the process is stationary, as the mean is constant across time (figure 7) , and the autocorrelation function is function in time difference only whenever the two samples taken lie in the same bit, otherwise the autocorrelation function output is zero (figure 8). And this was verified by fixing different columns (sample across all realizations) and computing the multiplication by all other columns and averaging the result and getting the same result (figure 8 and figure a 2).

- ### Is the random process ergodic?

YES, the process is ergodic, as the statistics across the time are the same as across the ensemble: the time mean (computed from one realization) is equal to the statistical mean (figure 9) , and the time autocorrelation function is the same as the statistical autocorrelation function (figure 10).

- ### What is the Bandwidth of the transmitted signal?

The bandwidth of the transmitted signal is 14.1 Hz.

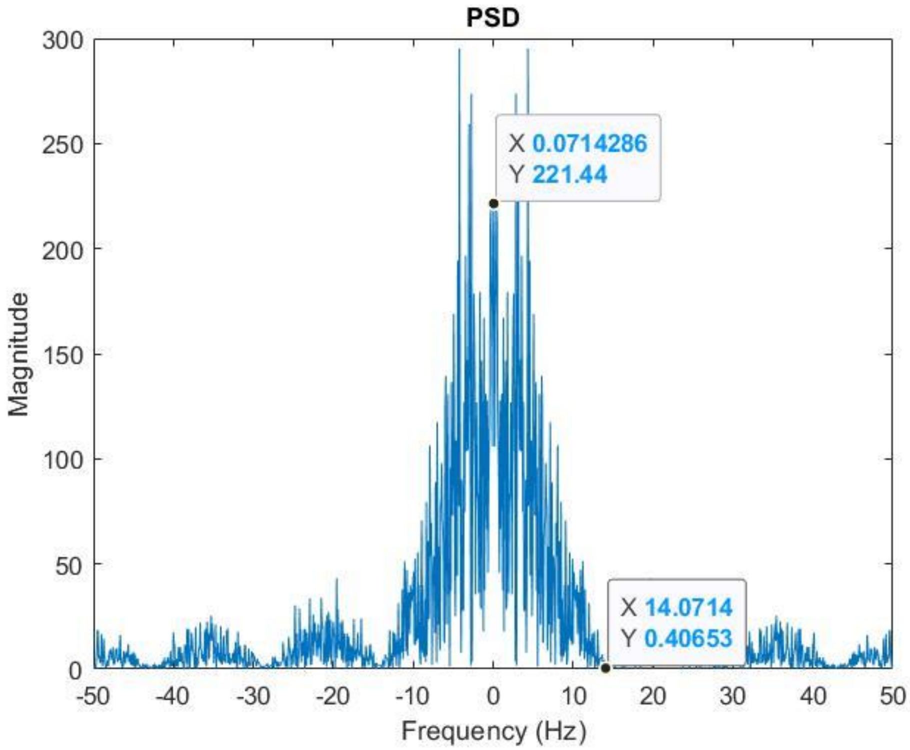


Figure 13 Polar NRZ PSD

4. Polar RZ Line Code

a. Transmitted Waveform

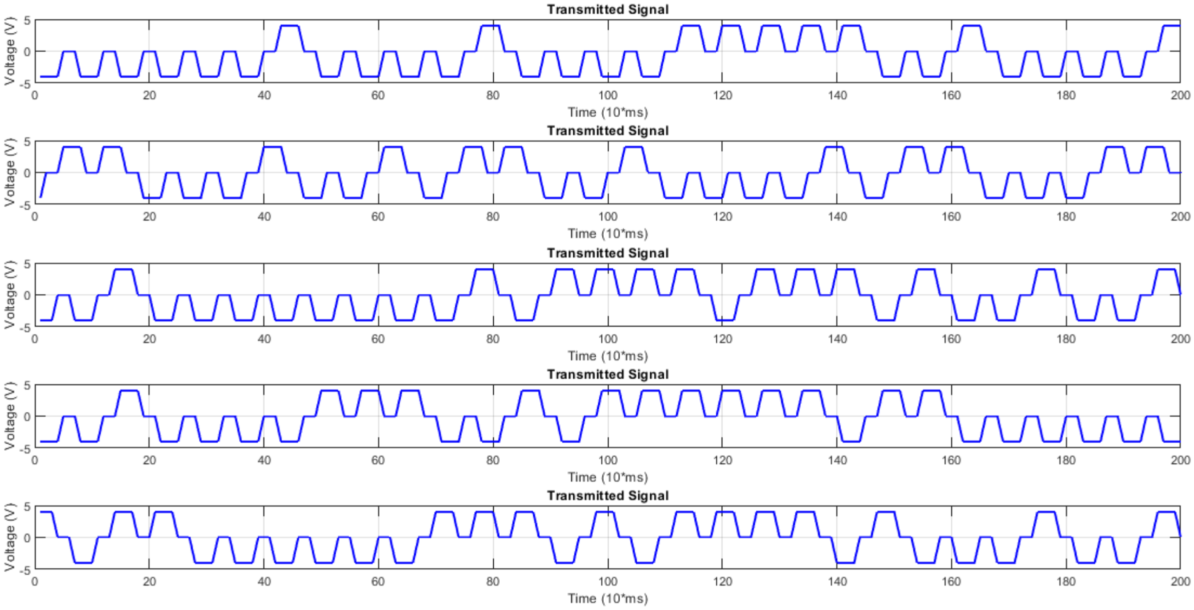


Figure 14 Transmitted Waveform Polar RZ

b. Figures

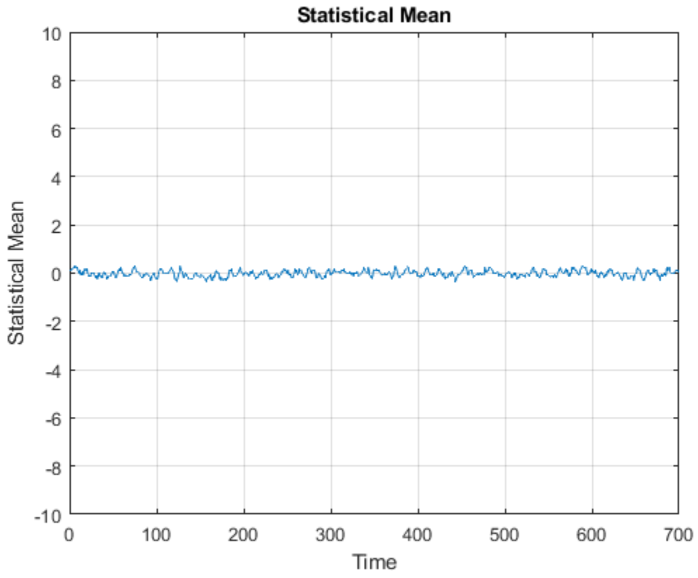


Figure 16 Polar RZ Statistical Mean

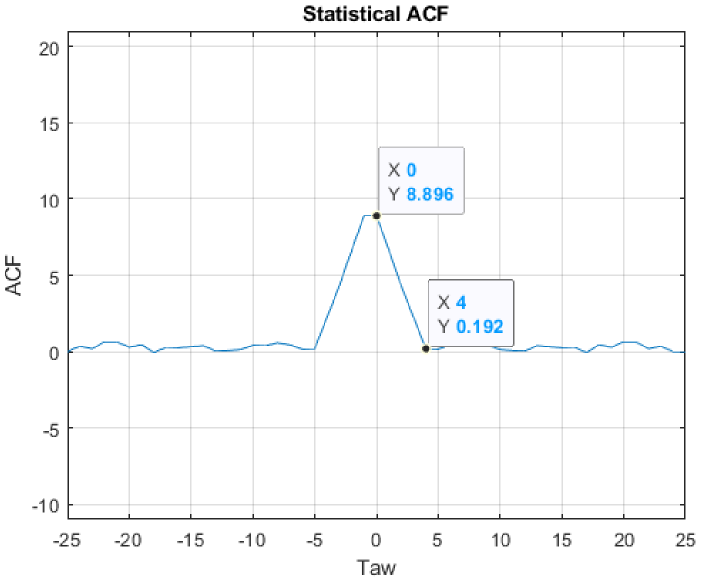


Figure 15 Polar RZ Statistical ACF (computed while column 1 is fixed)

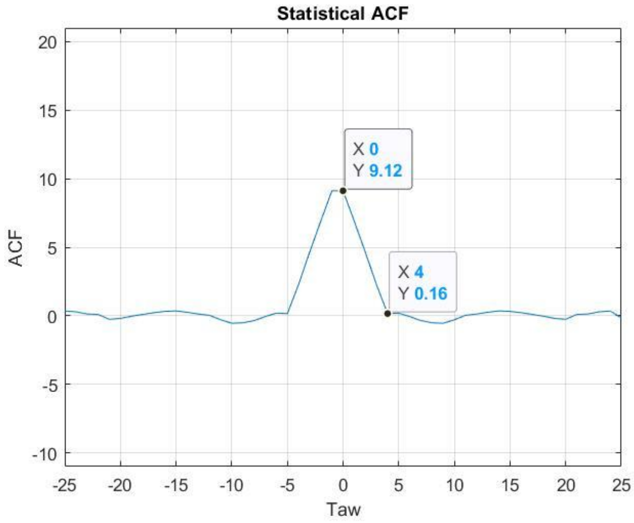
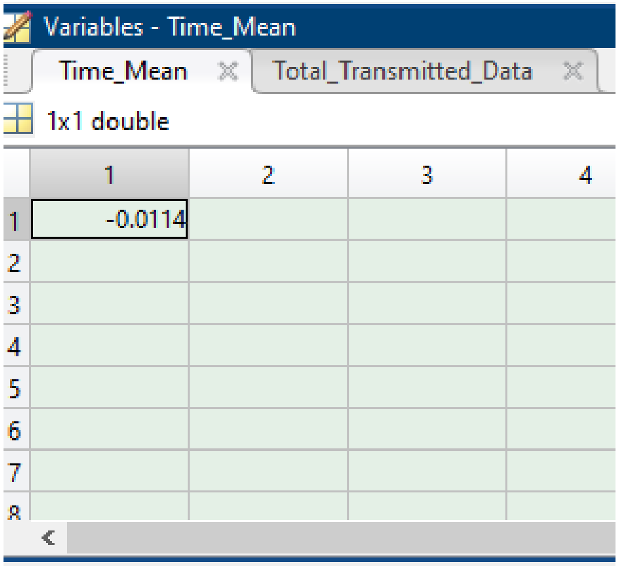


figure a 3 Polar RZ Statistical ACF (computed while column 1 is fixed)



Variables - Time_Mean

1x1 double

	1	2	3	4
1	-0.0114			
2				
3				
4				
5				
6				
7				
8				

Figure 17 Polar RZ Time Mean

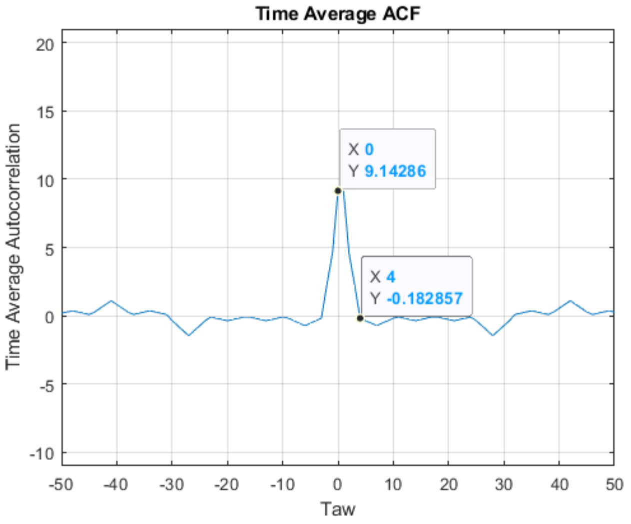


Figure 18 Polar RZ Time ACF

c. Discussion

- Is the random process Stationary?

YES, after computing the statistical mean and the statistical autocorrelation, it turns out that the process is stationary, as the mean is constant across time (figure 12) , and the autocorrelation function is function in time difference only whenever the two samples taken lie in the same bit and exactly in the first half of it (within 4 samples) , otherwise the autocorrelation function output is zero (figure 13), And this was verified by fixing different columns (sample across all realizations) and computing the multiplication by all other columns and averaging the result and getting the same result (figure 13 and figure a 3).

- Is the random process ergodic?

YES, the process is ergodic, as the statistics across the time are the same as across the ensemble: the time mean (computed from one realization) is equal to the statistical mean (figure 14) , and the time autocorrelation function is the same as the statistical autocorrelation function (figure 15).

- What is the Bandwidth of the transmitted signal?

The bandwidth of the transmitted signal is 25 Hz.

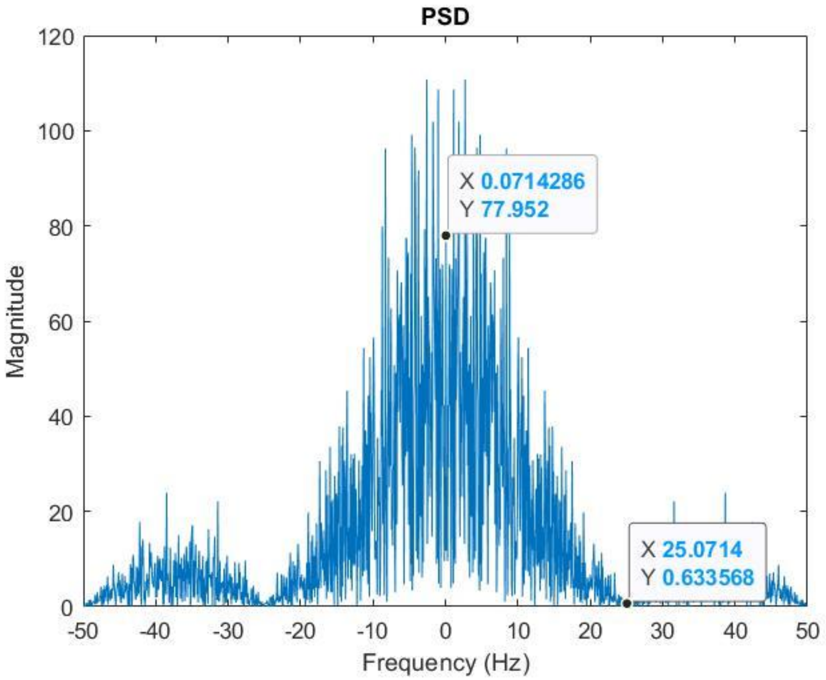


Figure 19 Polar RZ PSD