جامعة العلوم والتكنولوجيا
University of Science and Technology

Communications and Information Engineering Program

Probability and Stochastic Processes (CIE 327)

Omar Wael Abdelmohsen Ayyad (202000806)

Project link:

https://github.com/Omar3yyad25/ANALYSIS-OF-STOCHASTIC-PROCESSES

1. Introduction

This project aims to analyze a stochastic process by importing the data and performing various statistical analyses. The ensemble mean and autocorrelation function (ACF) will be calculated for the entire dataset, as well as the time mean and time ACF for individual segments of the data. The power spectral density (PSD) will also be calculated, along with the total average power of the process. Additionally, a graphical user interface (GUI) has been developed to facilitate user interaction and data visualization. The results of these analyses will provide insight into the underlying properties and behavior of the stochastic process, and the GUI allows for easy access and interpretation of the results.

2. Code structure

- The code imports various libraries such as matplotlib, numpy, and scipy.io, which are used for visualization, mathematical calculations, and file input/output operations.
- A number of global variables are defined at the beginning of the code, such as ens_mean, time_mean, ACF_arr, ACF_matrix, ACF, result, time_ACF, X, t, and x. These variables are used to store intermediate results and final output throughout the code.
- The read_file() function is used to read a .mat file, which contains the signal and time data, and assigns the data to the X, t, and x variables.
- A Tkinter window is created, titled "Probabiblty project 2", with dimensions 900x750.
- Four matplotlib figures, fig1, fig2, fig3, and fig4 are created with subplots ax1, ax2, and ax3, respectively, to display the sample functions, ensemble mean, and 3D ACF.
- Tkinter labels and textboxes are created for user input, such as the file name, number of sample functions, and values for i and j for the ACF calculations.
- The samples_plot() function is defined and is used to plot the sample functions from the X matrix.
- The ensemble_mean() function is defined and is used to calculate and plot the ensemble mean of the sample functions.

- The ACF() function is defined and is used to calculate the ACF of the ensemble mean, and plot the results in 3D.
- The time_mean_nth() function is defined and is used to calculate the time mean of the nth sample function.
- The time_ACF_nth() function is defined and is used to calculate the time ACF between the i-th and j-th sample functions.
- The PSD() function is defined and is used to calculate the PSD of the ensemble mean.
- The Tkinter window is configured to display the figures using FigureCanvasTkAgg, and the buttons are created to call the relevant functions upon being clicked.

3. Probability

- The samples_plot() function:
  This function is used to plot the sample functions from the X matrix. It reads the number of sample functions entered by the user and plots the corresponding number of sample functions from the X matrix.
- The ensemble_mean() function:
  This function is used to calculate and plot the ensemble mean of the sample functions. The ensemble mean is calculated as the mean of all the sample functions in the X matrix. The equation for ensemble mean is:
  ens_mean = (1/N) * sum(X[i,:]) where N is the number of sample functions.
- The ACF() function:
  This function is used to calculate the Auto-Correlation Function (ACF) of the ensemble mean and plot the results in 3D. The ACF is a measure of similarity between a signal and a time-shifted version of itself, and is used to determine the time lag between the two signals. The equation for the ACF of a signal x(t) is:

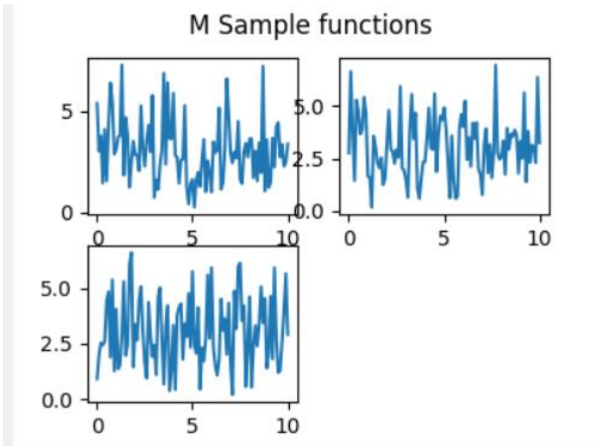
  ACF(tau) = (1/T) * Integral(x(t)*x(t-tau) dt) where T is the total time.

- The time_mean_nth() function:
  This function is used to calculate the time mean of the nth sample function. The time mean is calculated as the mean of the signal's amplitude over time. The equation for the time mean is:
  time_mean = (1/T) * Integral(X[n,i] dt) where T is the total time and n is the sample function number.
- The time_ACF_nth() function:
  This function is used to calculate the time Auto-Correlation Function (ACF) between the i-th and j-th sample functions. The time-ACF is a measure of similarity between two signals, and is used to determine the time lag between the two signals. The equation for the time-ACF of two signals x(t) and y(t) is:
  time_ACF = (1/T) * Integral(x(t)*y(t-tau) dt) where T is the total time and tau is the time lag.
- The PSD() function:
  This function is used to calculate the Power Spectral Density (PSD) of the ensemble mean. The PSD is a measure of the distribution of power in a signal as a function of frequency. The equation for the PSD of a signal x(t) is:
  PSD(f) = |FFT(x(t))|^2 where FFT is the Fast Fourier Transform of the signal.

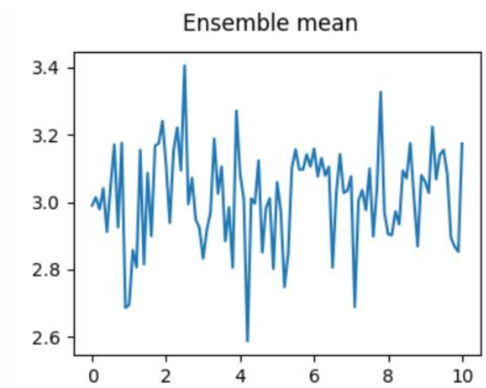
4. Tests

Process 1: which is provided by on classroom

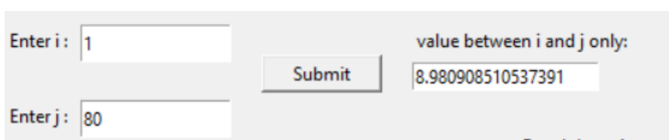
M sample functions:

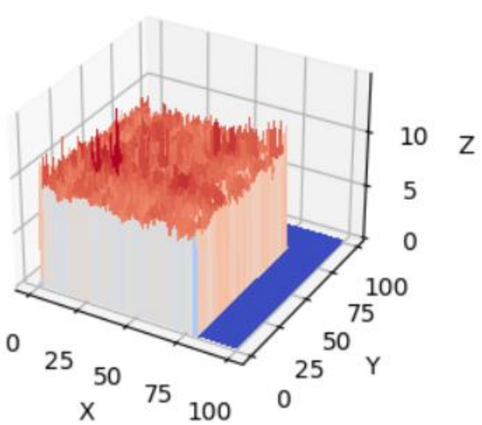


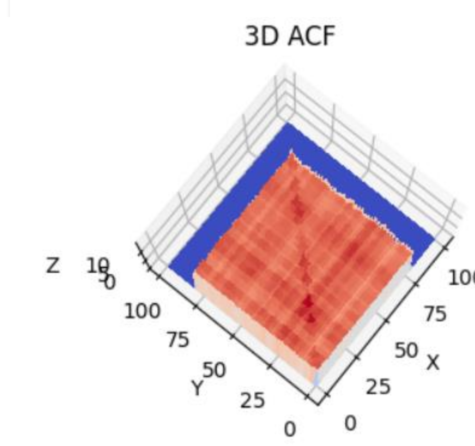
Ensemble mean:



3D ACF:



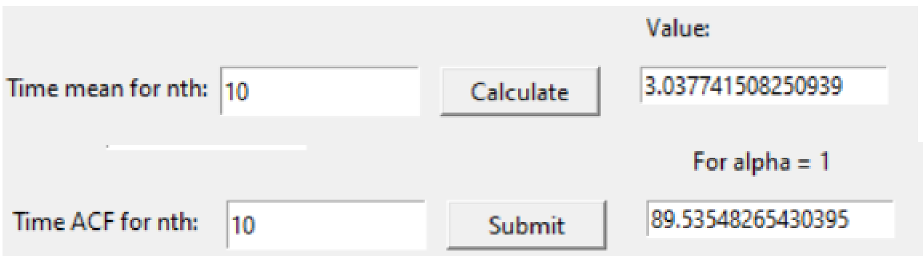




Comment:

In these figures it is shown that when ***i, j*** *are equal the have higher autocorrelation function as they are the same sample with difference. It would be much obvious in the top view as shown the right figure.*

Time mean and ACF:



Comment:

In the context of a time-series process, the statistical mean and the time mean refer to different things.

The statistical mean, also known as the population mean, is a measure of central tendency that is calculated by summing all the values of a variable and dividing by the total number of observations in the population. It is denoted by the symbol μ *and represents an estimate of the true mean of a population from which a sample is drawn.*

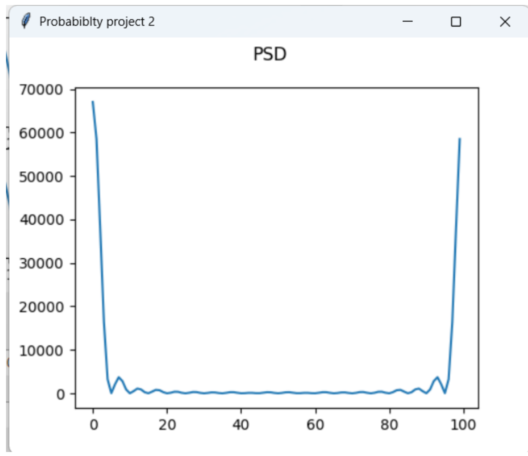
The time mean, also known as the time-average, is a measure of the long-term behavior of a time-series process, it is calculated by averaging the values of a variable over time.

Comment:

The statistical ACF is a measure of the correlation between the values of a variable at different lag values, computed based on a sample of the population. It is denoted by the symbol $\rho(k)$ *and it is a property of the underlying probability distribution of the population.*

The time ACF, also known as the sample ACF, is a measure of the correlation between the values of a variable at different lag values, computed based on a sample of the time series. It is denoted by the symbol r(k) and it is a property of the sample time series.

PSD:

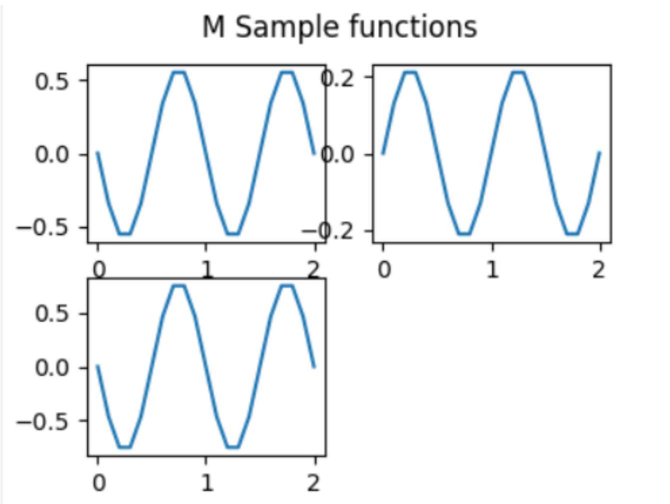


<u>Process 2: generated by matlab</u>

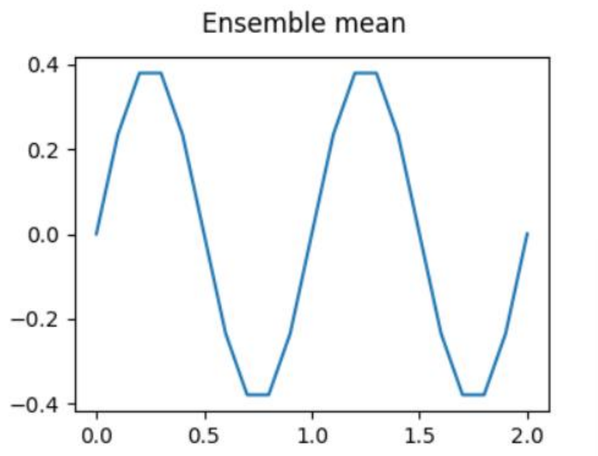
First we generated the process as the following code which has its boundaries and normal distribution for its Beta. 100 sample function were generated randomly and the results came as following

```
dt = 0.1; % time step
t = 0:dt:2; % time vector from 0 to 2 with time step 0.1

for i = 1:100
    beta = normrnd(0,1); % generate random variable from normal distribution N(0,1)
    X(i,:) = beta*sin(2*pi*t); % compute X(t)
end
```

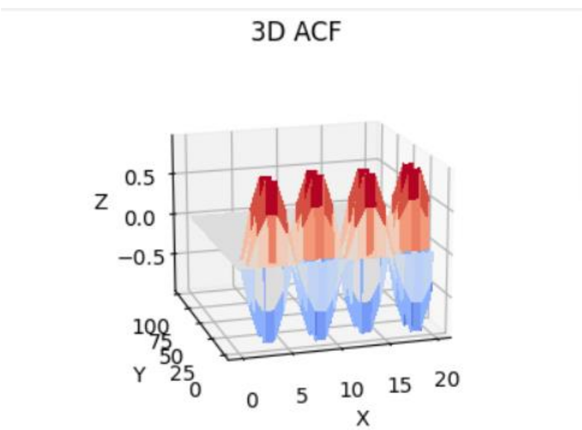
M sample functions:

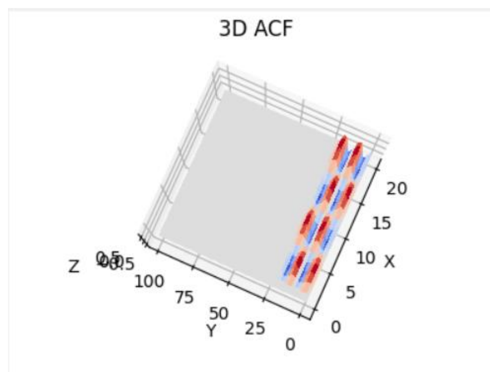


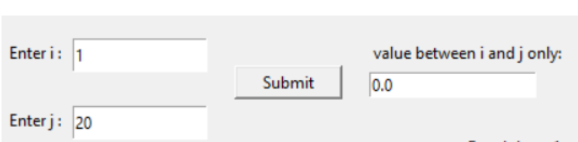
Ensemble mean:



3D ACF:

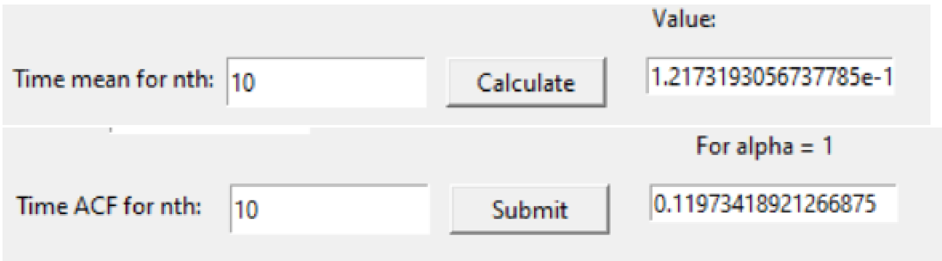




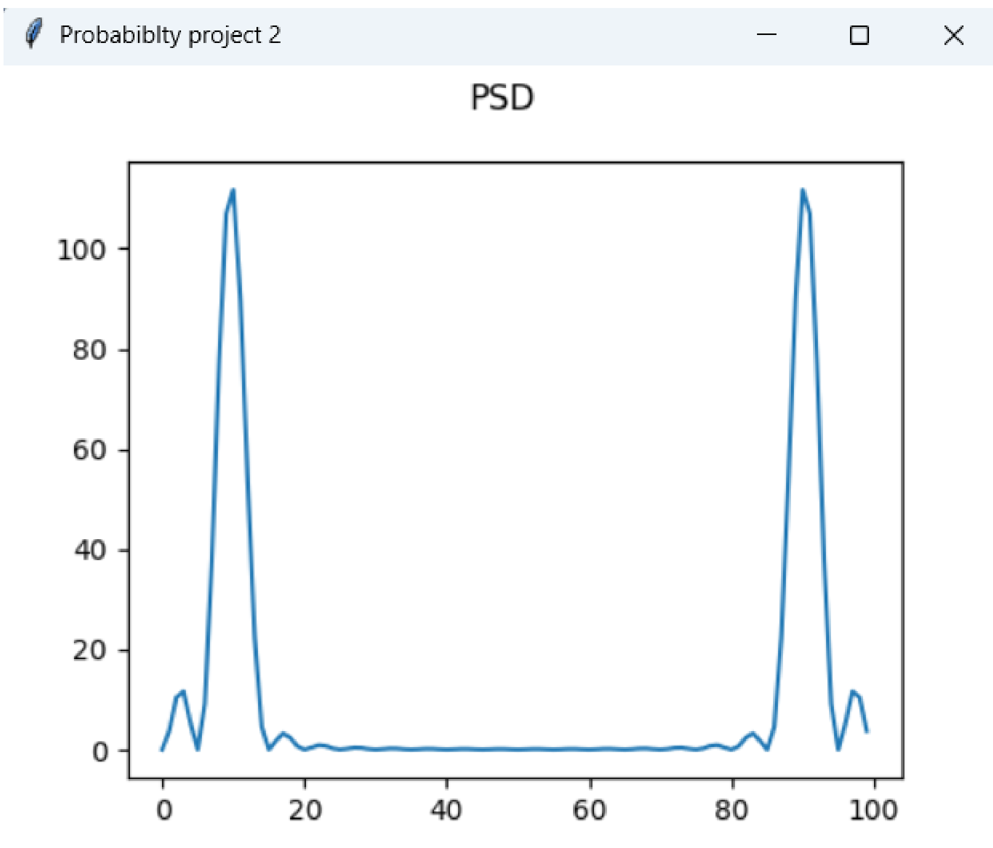


The 3D ACF is from i = 1, and j = 20

Time mean and ACF:



PSD:



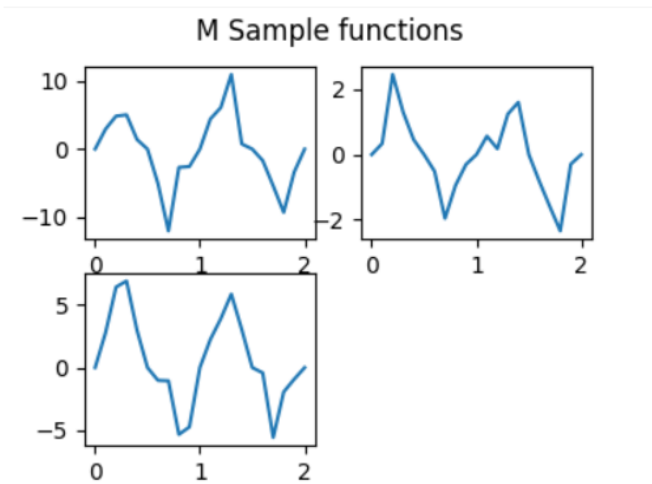
Process 3: which are multiplication of X(t) and Y(t)

It's generated by the following code

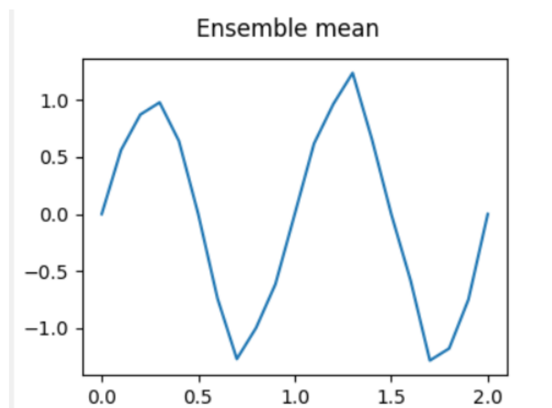
```
data1 = load('Sample_Process_2022.mat');
data2 = load('y_t.mat');
X1 = data1.X;
X2 = data2.X;
t1 = data1.t;
t2 = data2.t;

X1 = X1(:,1:21);

if size(X1) == size(X2)
    X = X1 .* X2;
else
    error('the size of X1 and X2 are not equal');
end

```

The results of testing the code came as following:

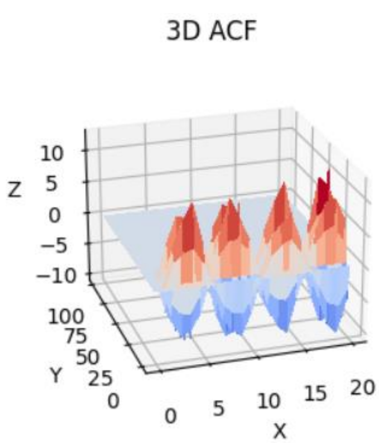
M sample functions

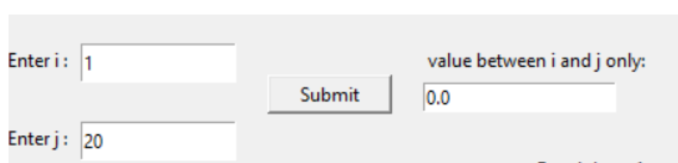


Ensemble mean:

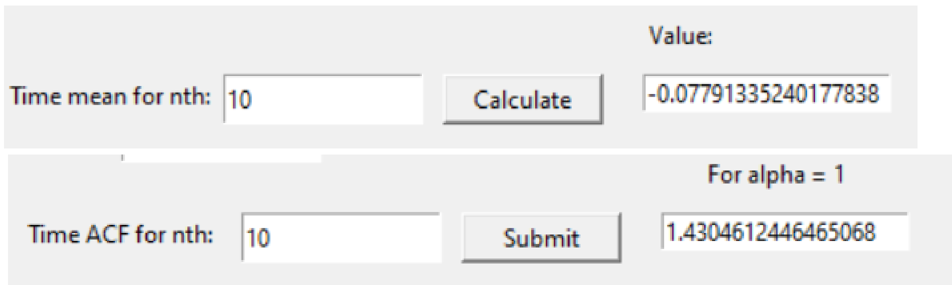


3D ACF:

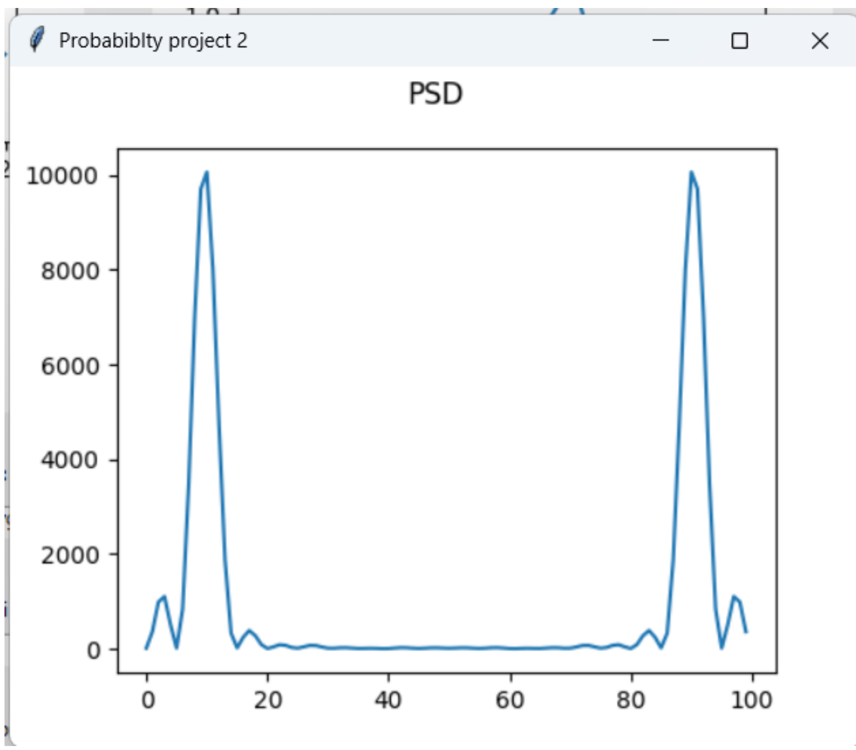




Time mean and ACF:



PSD:



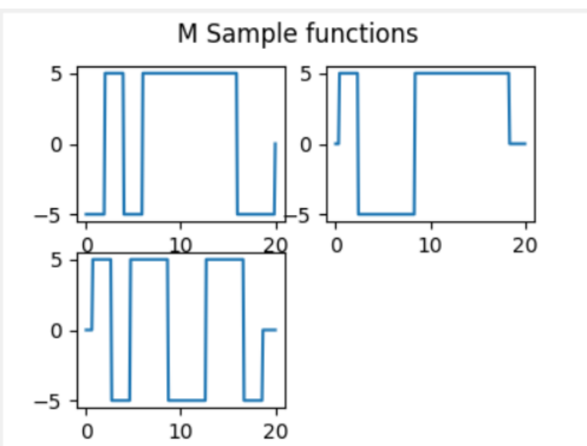
Process 4: generated by matlab for Polar NRZ code

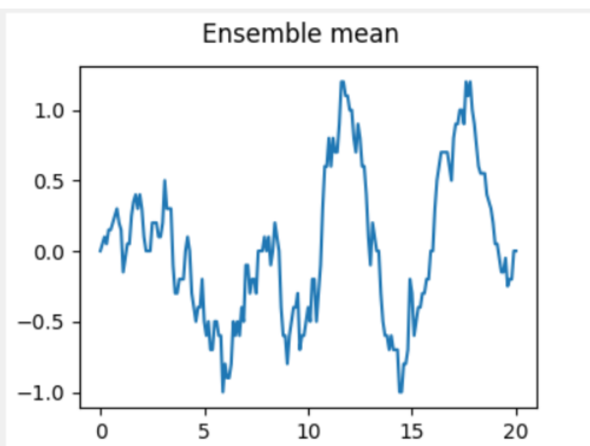
It's generated by the following code

```matlab
% Parameters
A = 5;
Tb = 2;
num_of_bits = 10;
X = zeros(100, numel(t)); % Initialize signal with zeros
alphas = zeros(100,1);
bits_matrix = zeros(100,num_of_bits);

for k = 1:100
    % Generate random bits and alpha for each iteration
    bits = randi([0 1],1,num_of_bits);
    alpha = rand(1)*Tb;
    alphas(k) = alpha;
    bits_matrix(k,:) = bits;

    start_index = ceil(alpha*10); % Starting index of bit in P_rawpl

    for i = 1:num_of_bits
        if bits(i) == 1
            X(k, start_index:start_index+(Tb/0.1)-1) = A;
        else
            X(k, start_index:start_index+(Tb/0.1)-1) = -A;
        end
        start_index = start_index+(Tb/0.1);
        % Ensure all bits stop at t = 20
        if start_index+Tb/0.1 > numel(t)
            break;
        end
    end
end
```

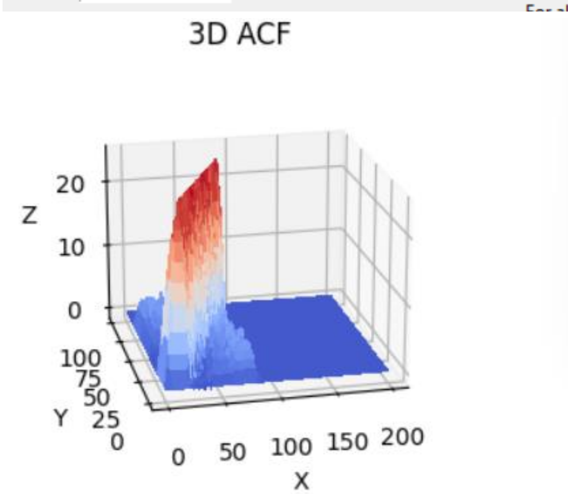
The results of testing the code came as following:

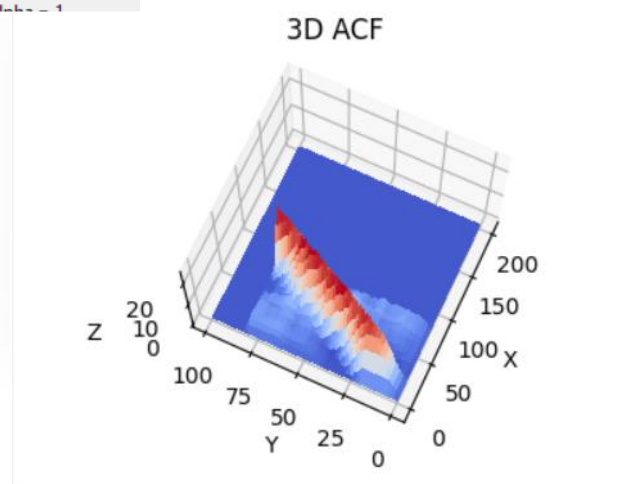
M sample function ***Ensemble mean***



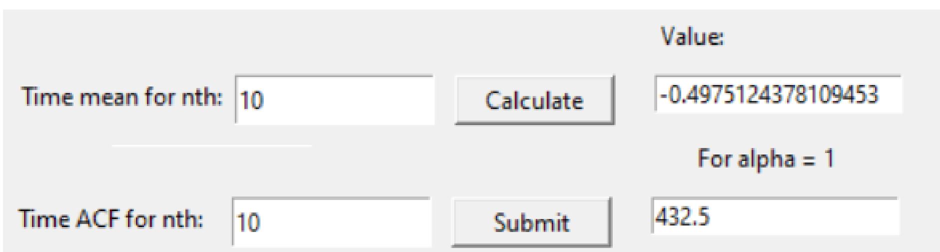


3D ACF:

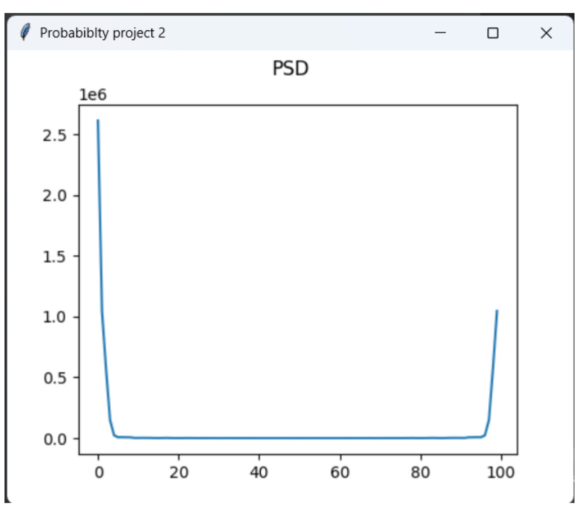




Time mean and ACF:



PSD:



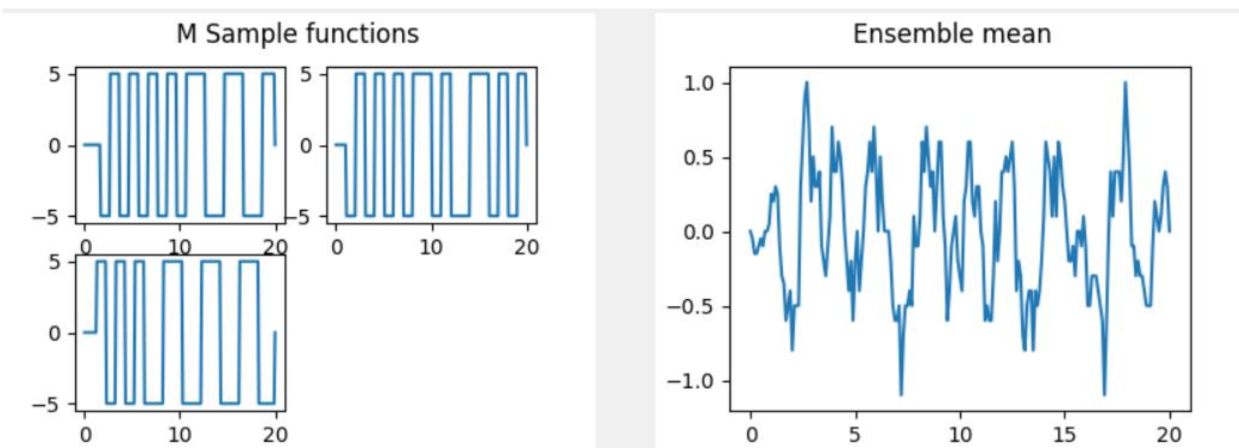
<u>Process 5: which generated by matlab for Manchester Code</u>

It's generated by the following code

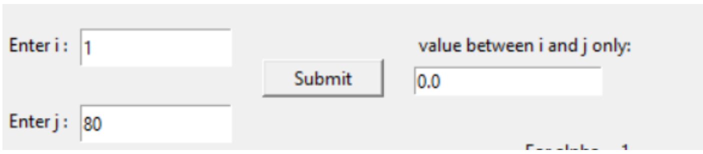
```matlab
        bits_matrix(j,:) = data;
        % Initialize the manchester code
        manchester =[];

        for i = 1:n
            if data(i) == 0
                manchester =[manchester , -A*ones(1,(Tb/2)/dt) , A*ones(1,(Tb/2)/dt)];
            else
                manchester =[manchester , A*ones(1,(Tb/2)/dt) , -A*ones(1,(Tb/2)/dt)];
            end
        end

        % Append zero values at the beginning of the signal to represent the alpha time shift
        alpha_vector = A*linspace(0,0,ceil(alpha/dt)+1);
        alpha_vector = alpha_vector(2:end);

        alpha_length = length(alpha_vector);
        manchester_length = ceil(T/dt) - alpha_length;
        X(j, alpha_length+1: alpha_length+manchester_length) = manchester(1:manchester_length);
        alpha_matrix(j) = alpha;

    end

    % Plot the signal matrix
    figure;
    for j = 1:100
        plot(t,X(j,:));
        hold on;
        xlabel('Time (s)')
        ylabel('Amplitude (V)')
        title('Manchester Code')
    end
```

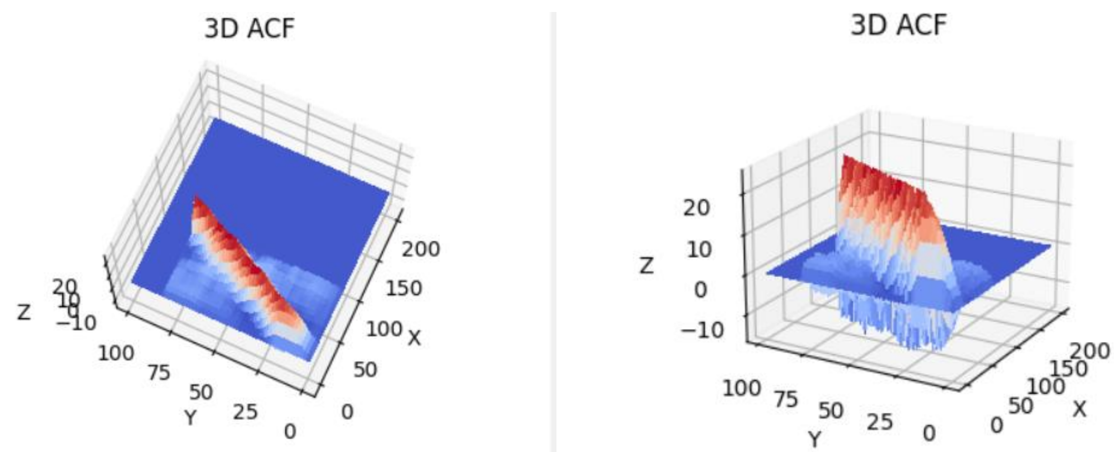
The results of testing the code came as following:

M samples functions and Ensemble mean:

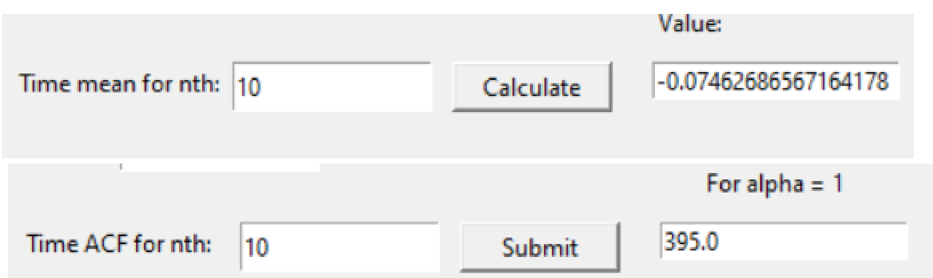


3D ACF:





Time mean and ACF:



PSD: