CS306: Data Analysis and Visualization
Lab 1: Report

Rajdeep Pinge 201401103
Aditya Joglekar 201401086

Objective:

Testing data for normality and the possible effect on inference or decision making

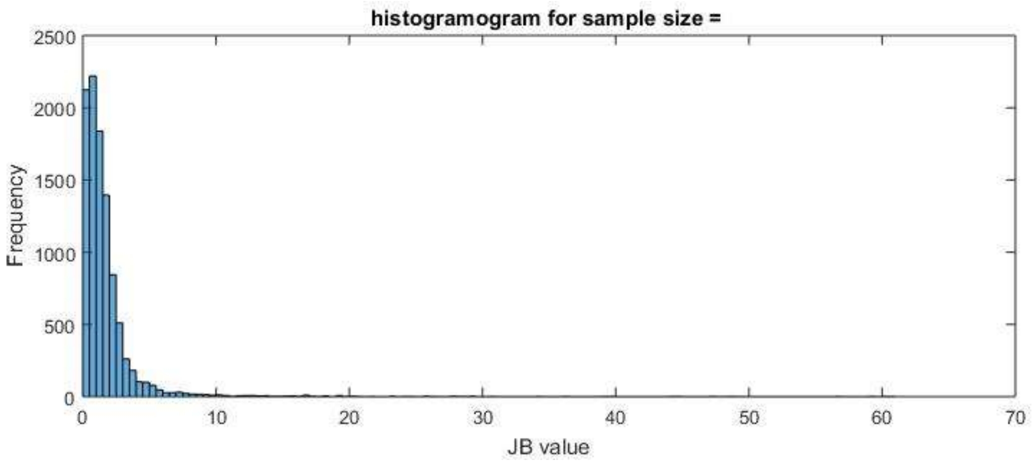
Experiment 1: For different data sets, perform JB test and verify the nature of the distribution of data.

JB test is a type of hypothesis testing which assumes that the JB test statistic follows a chi-squared distribution, if samples are drawn from a normal population. We aim to verify this assumption using experiments.

Q1 Load data_lab4.mat. ‘population_normal’ is a collection of 10 million observations drawn from a standard normal distribution. Assume this to be the population of interest. The Jarque-Bera (JB) test is a type of hypothesis testing which assumes that the JB test statistic follows a chi-squared distribution, if samples are drawn from a normal population. The aim of this experiment is to verify this assumption. Your solution should consider 3 different sample sizes: 50, 1500, and 2500. In which case is the assumption of chi-squared distribution more accurate? Based on the answer, use the corresponding sampling distribution and $\alpha = 0.05$, to ascertain the normality of the 5 samples provided. (note: for this experiment do not use the actual pdf but the experimental sampling distribution)

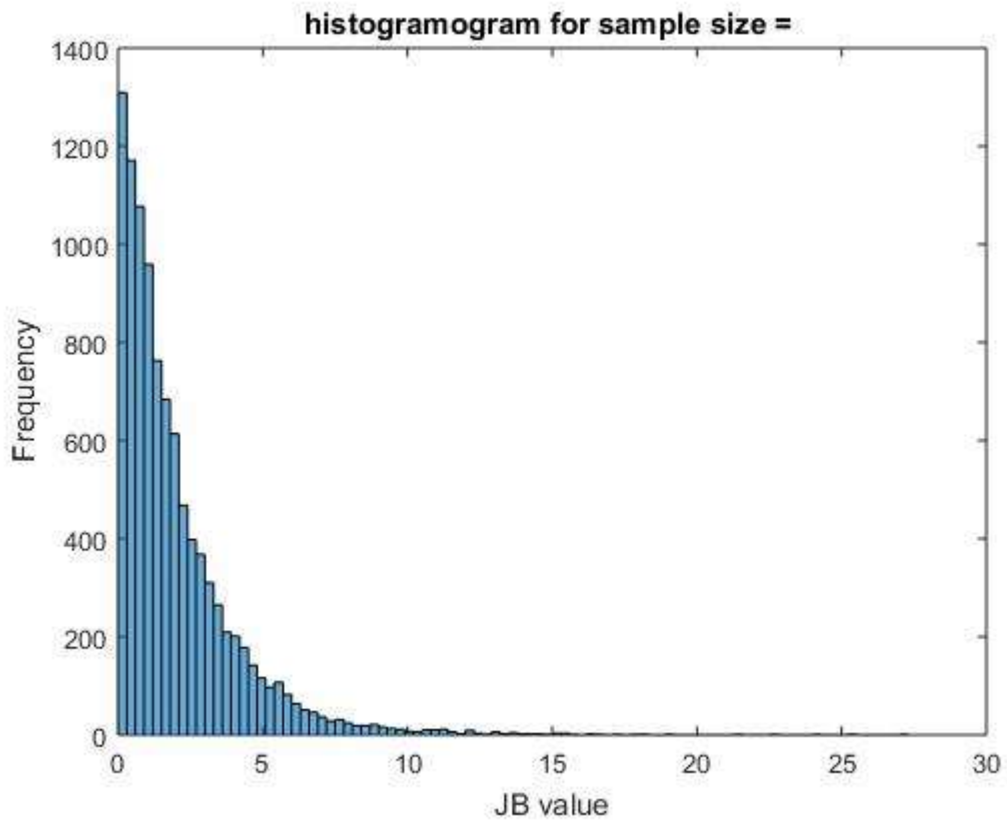
Data set: Population_normal

Sample size = 50



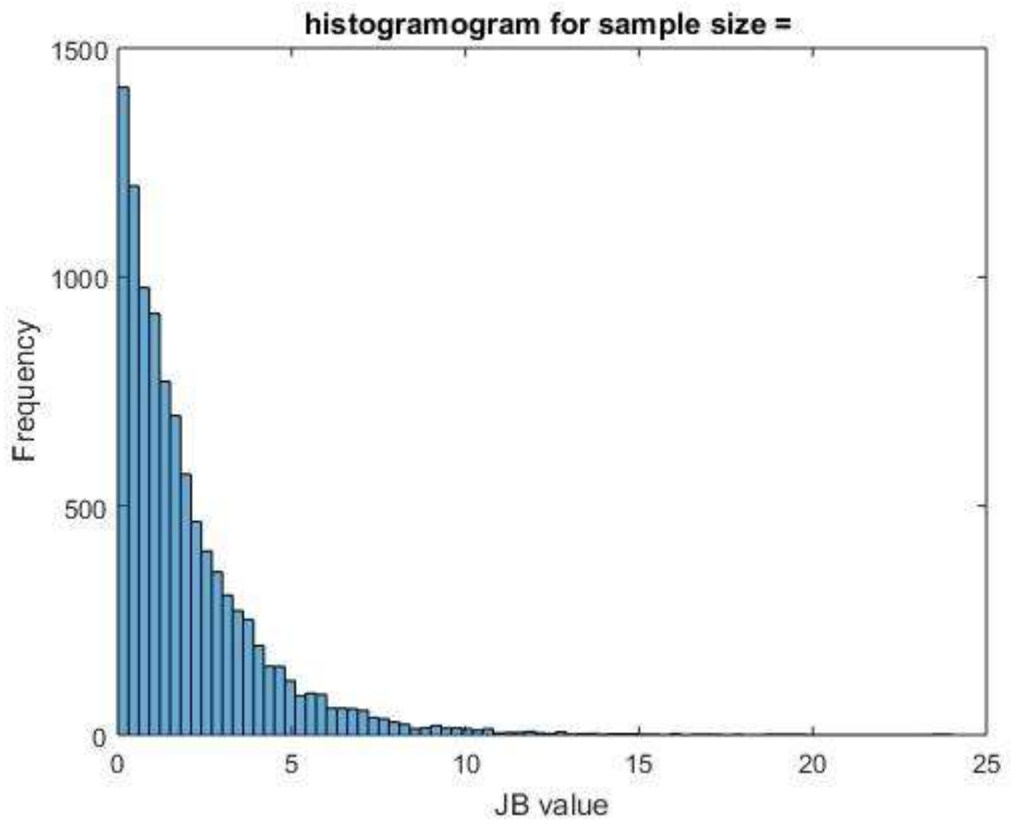
```
confidence_level =

    0.9704
```

Sample size = 1500



```
confidence_level =

    0.9516
```

Sample size = 2500

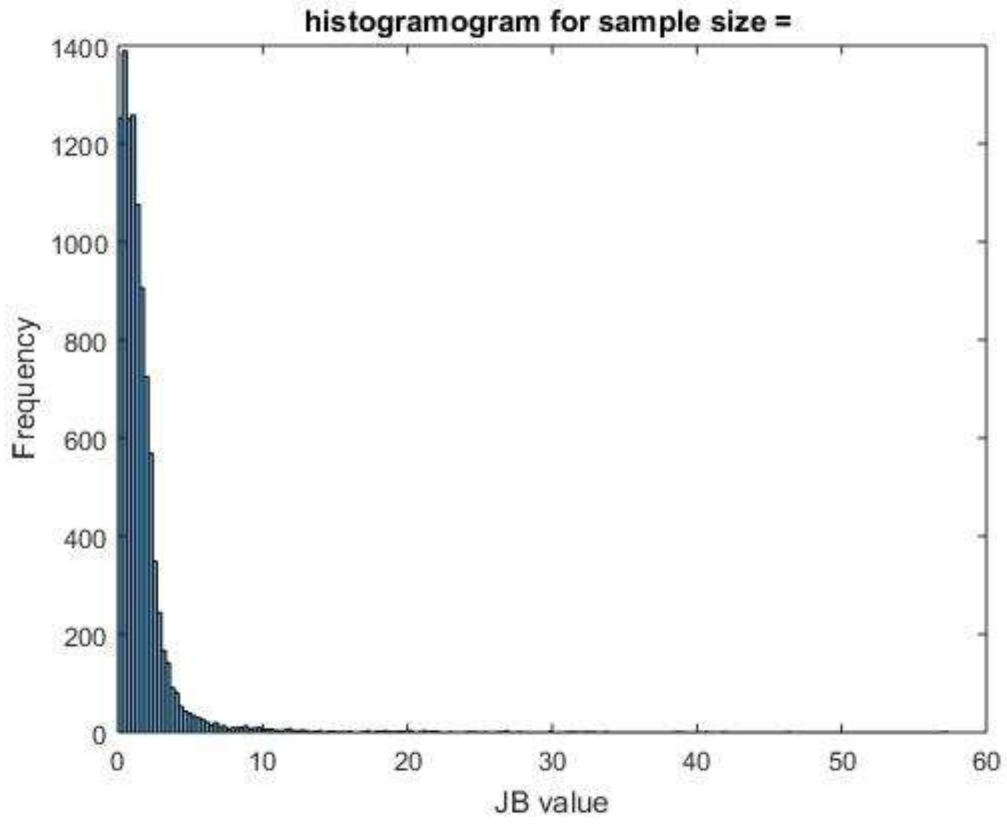


```
confidence_level =

    0.9473
```

Among above 3, the sample_size 50 gives the most accurate answer. It is very close to the normal distribution. On the whole, in 2 cases, we can say that the data is normally distributed.

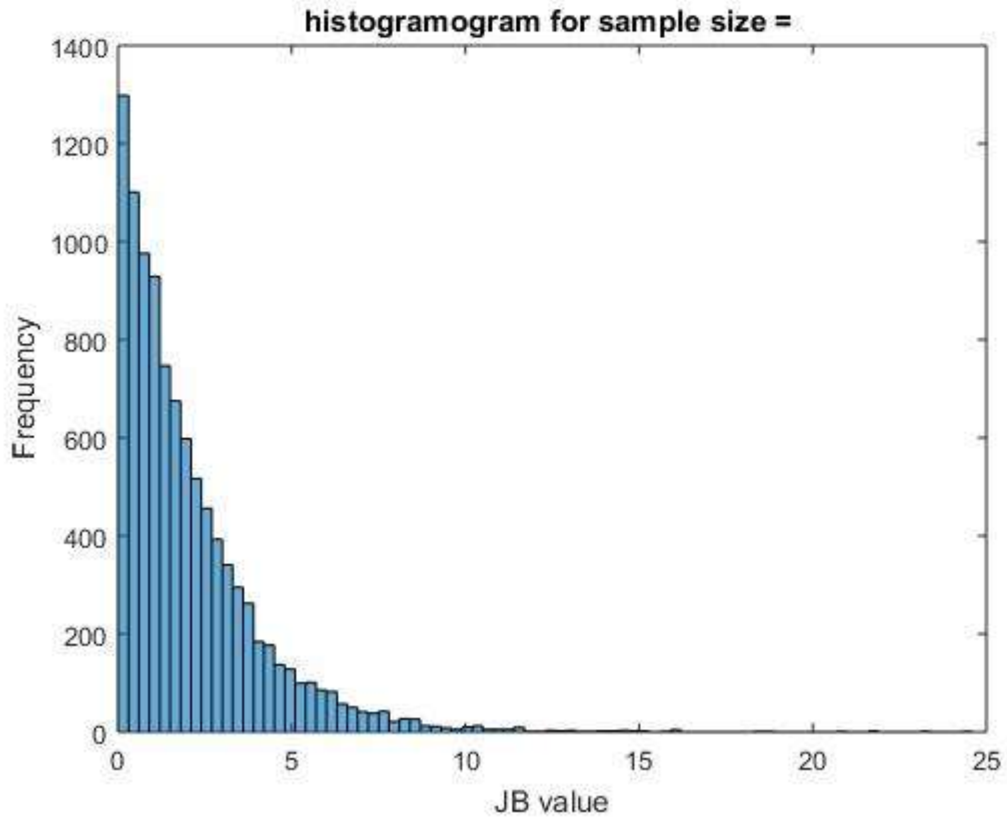
Data set : sample_50k

Sample size = 50



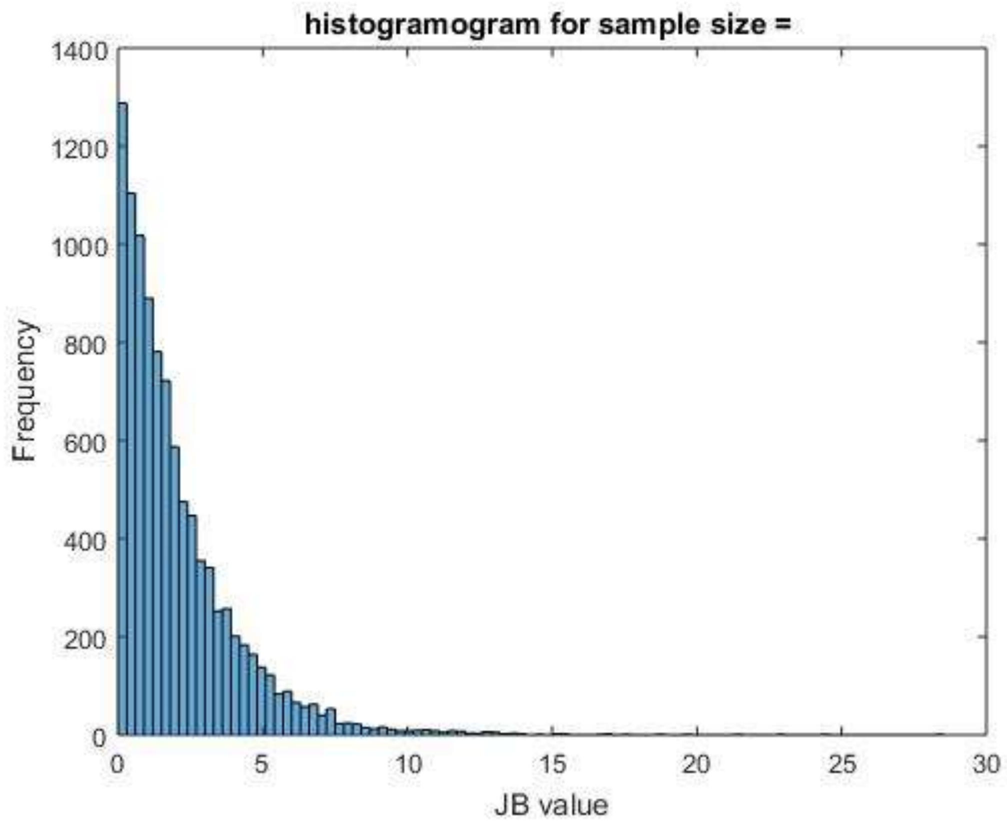
```
confidence_level =

    0.9723
```

Sample size = 1500



```
confidence_level =

    0.9495
```

Sample size = 2500

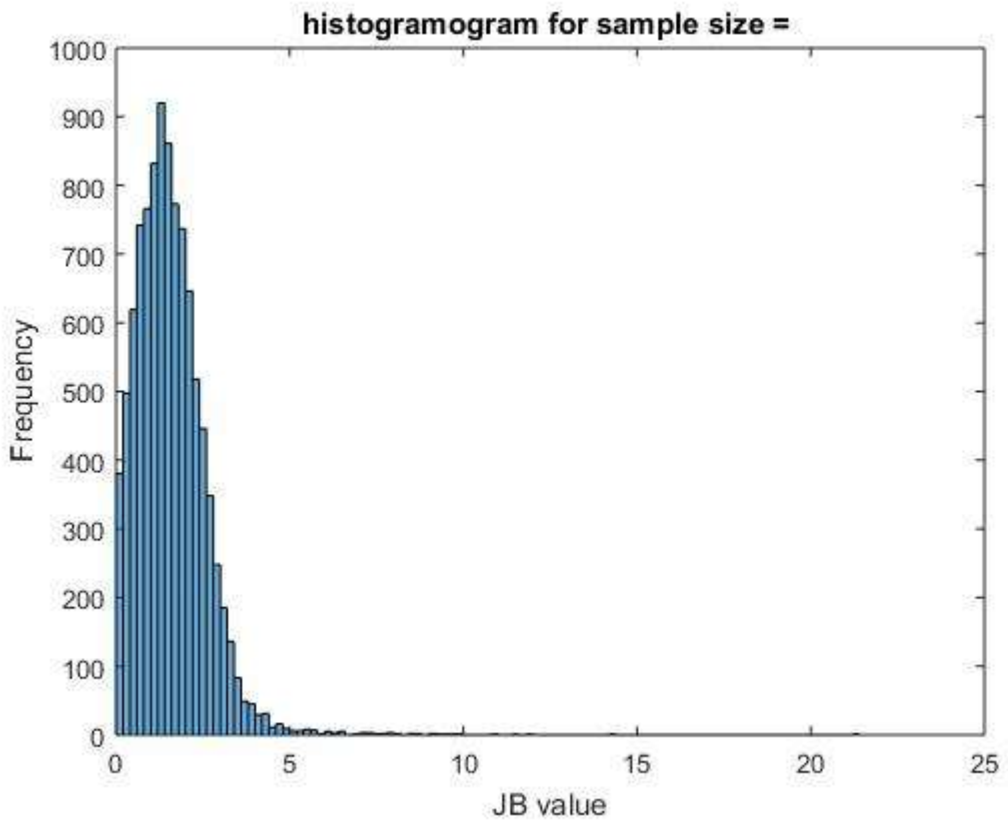


```
confidence_level =


    0.9495
```

Among above 3, the sample_size 50 gives the most accurate answer. It is very close to the normal distribution. On the whole, in first cases, we can say that the data is normally distributed.

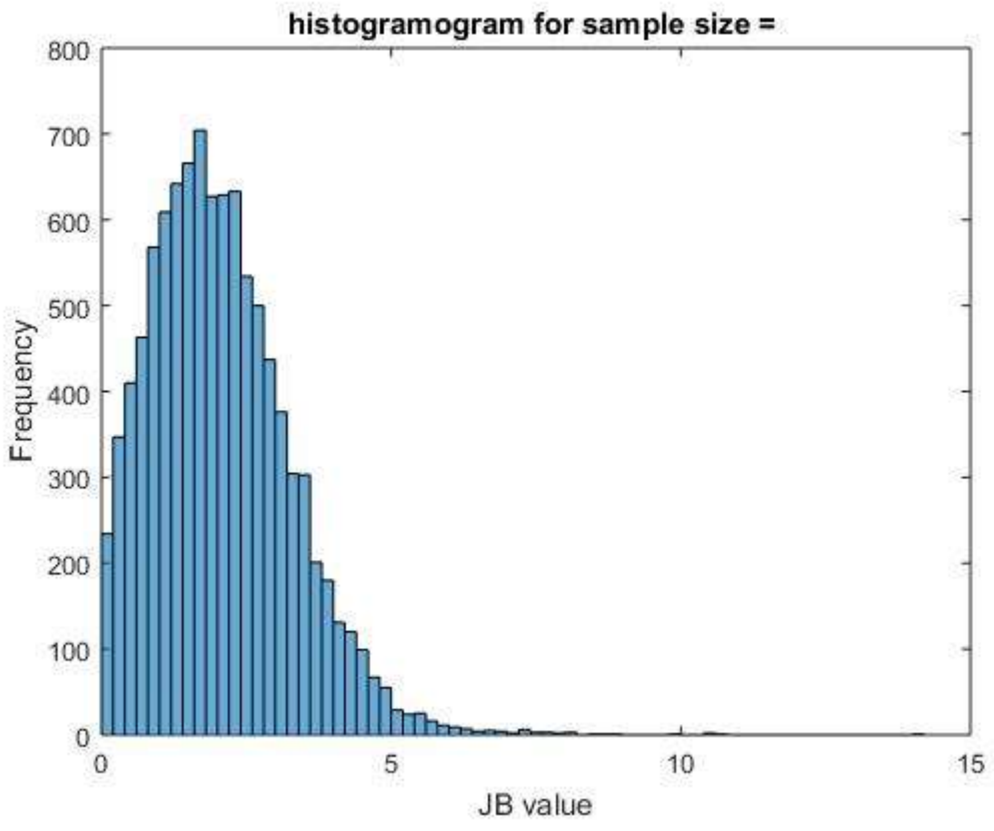
Data Set: Ammonia Concentration

Sample size = 50



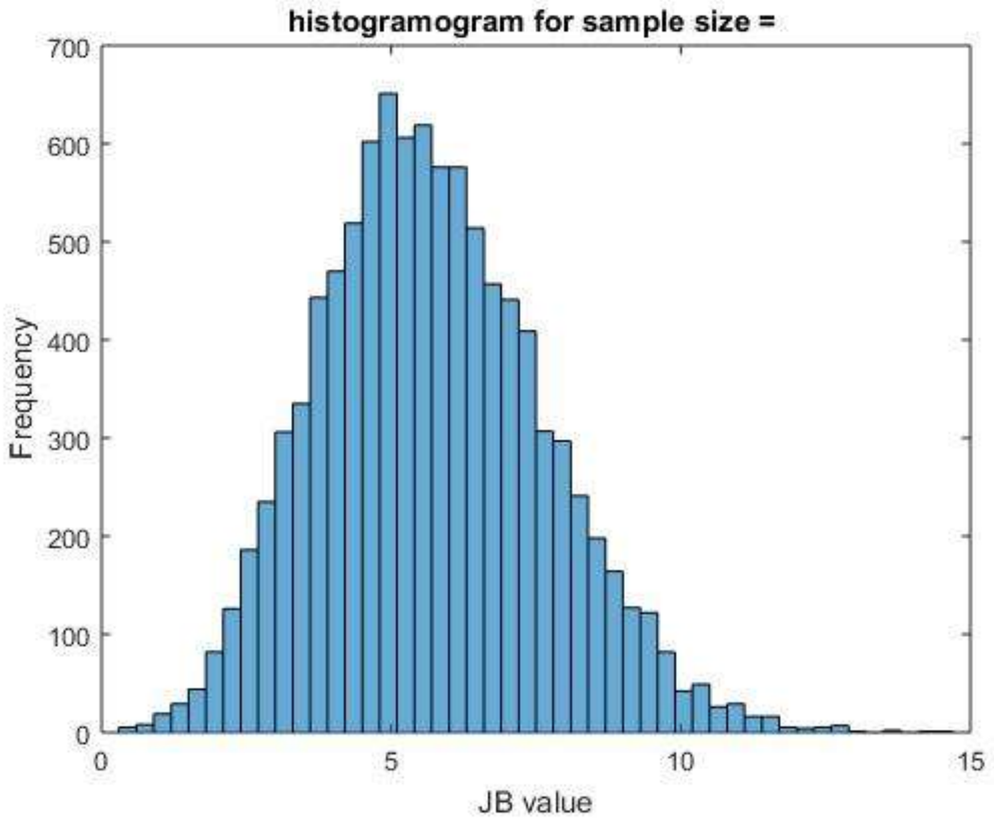
```
confidence_level =

   0.9956
```

Sample size = 100



```
confidence_level =

    0.9944
```

Sample size = 500



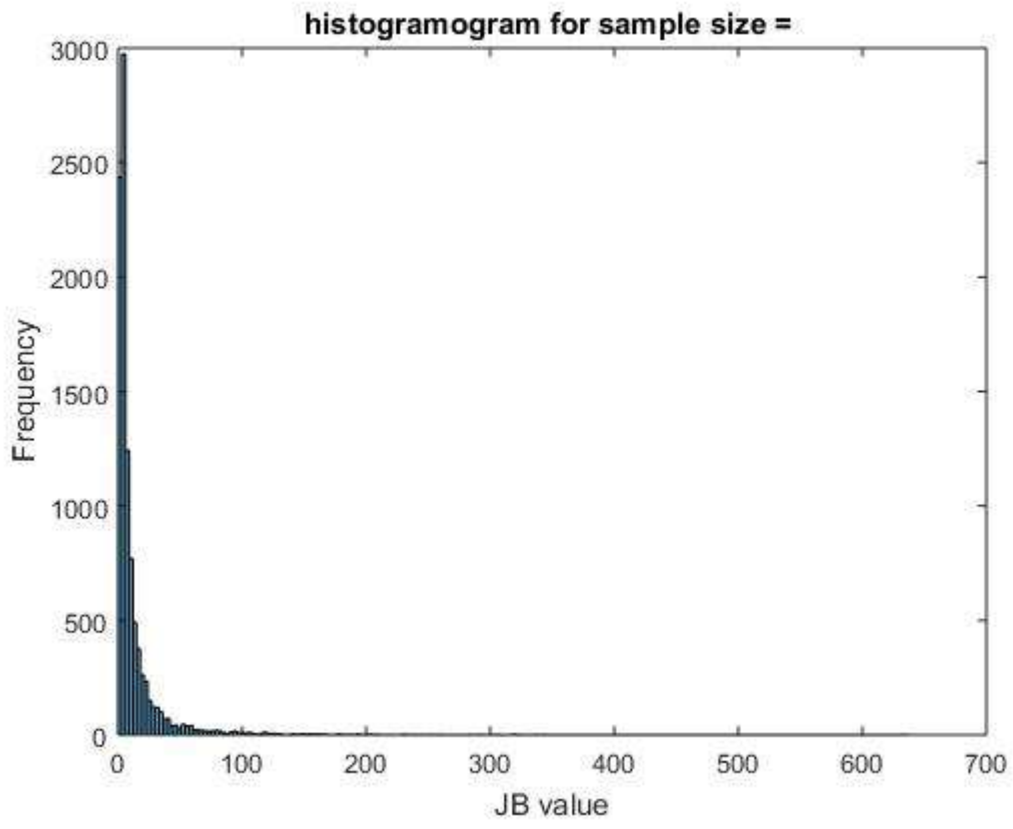
```
confidence_level =

    0.5850
```

Among above 3, the first and the second case give very good answers. It is very close to the normal distribution. On the whole, in 2 cases, we can say that the data is normally distributed.

The third sample size is gives poor result. May be because of large sample size.

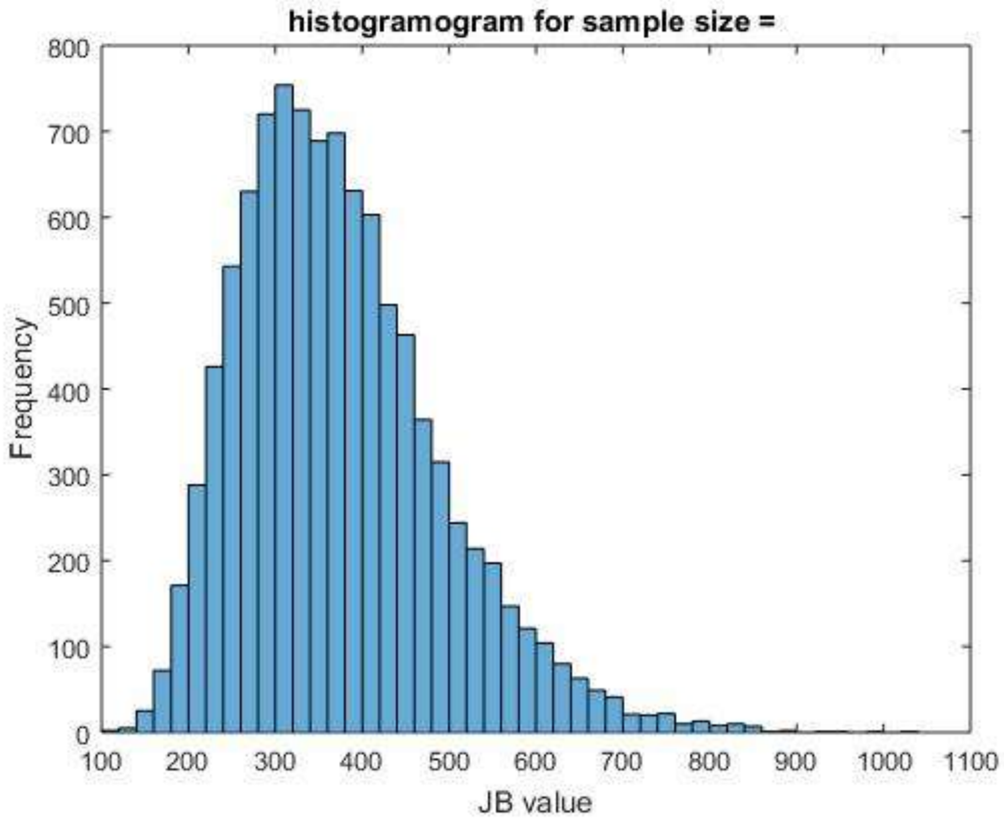
Data Set: score_natural_model

Sample size = 50



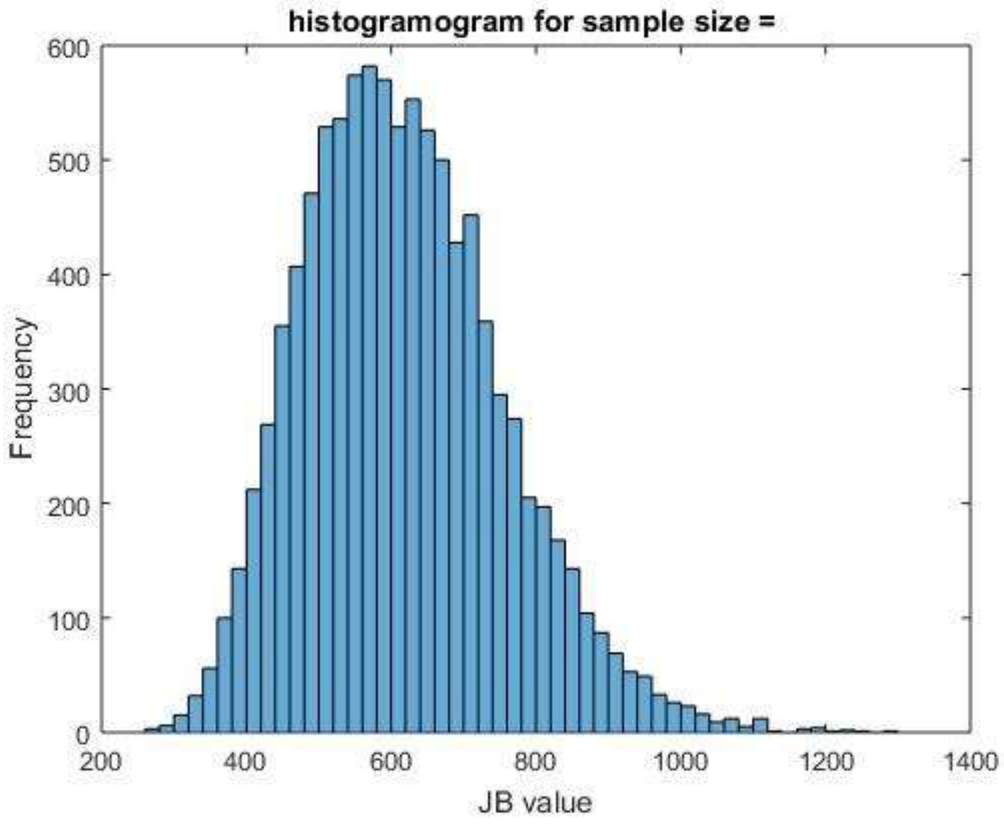
```
confidence_level =

    0.5402
```

Sample size = 1500



```
confidence_level =

     0
```

Sample size = 2500



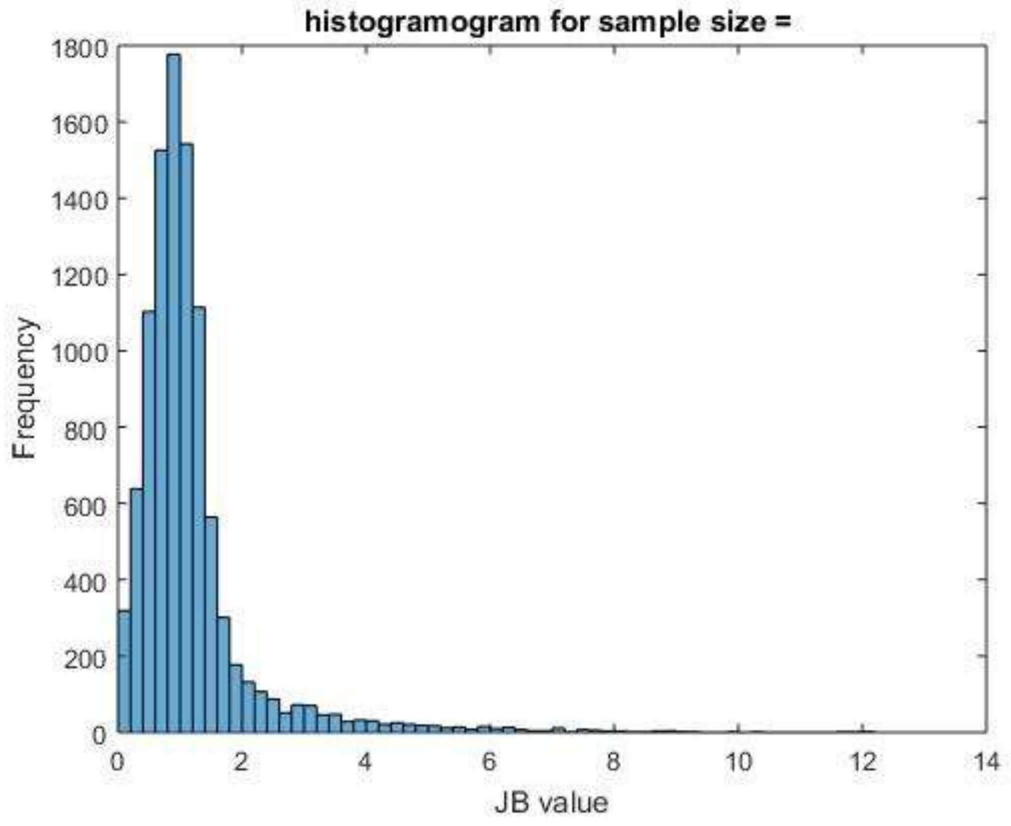
confidence_level =

    0

All the above 3 cases give poor result indicating the data is not normally distributed.

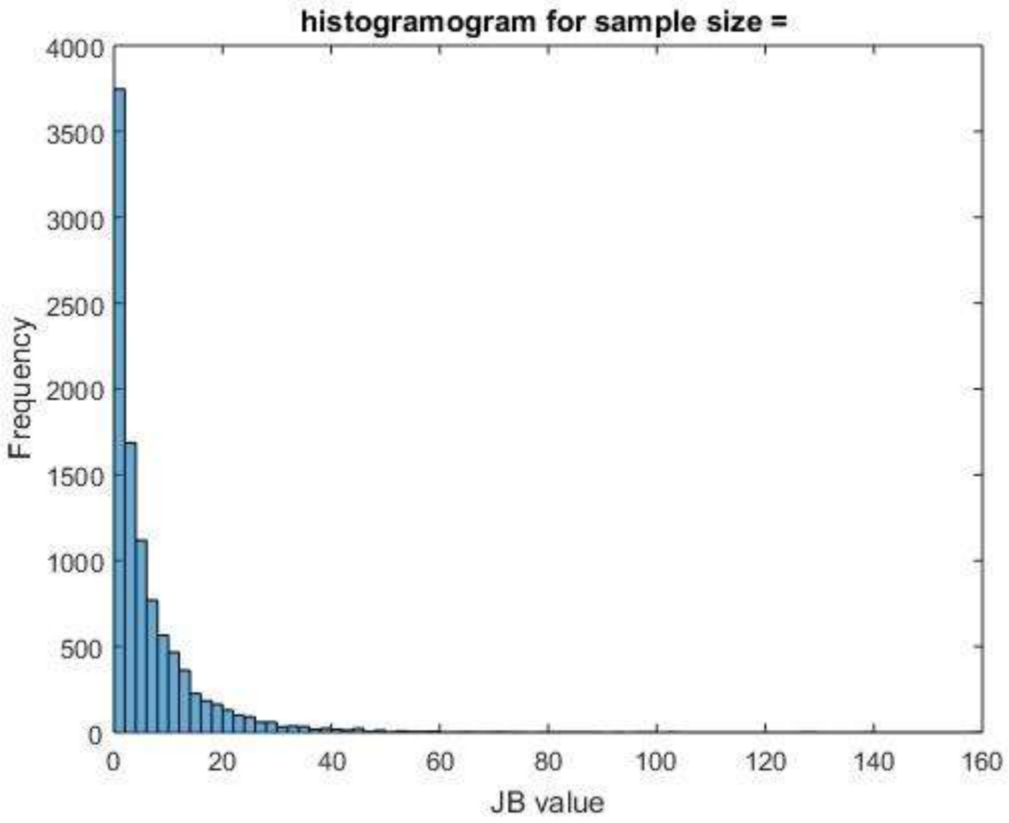
Data Set: Data 4

Sample size = 10



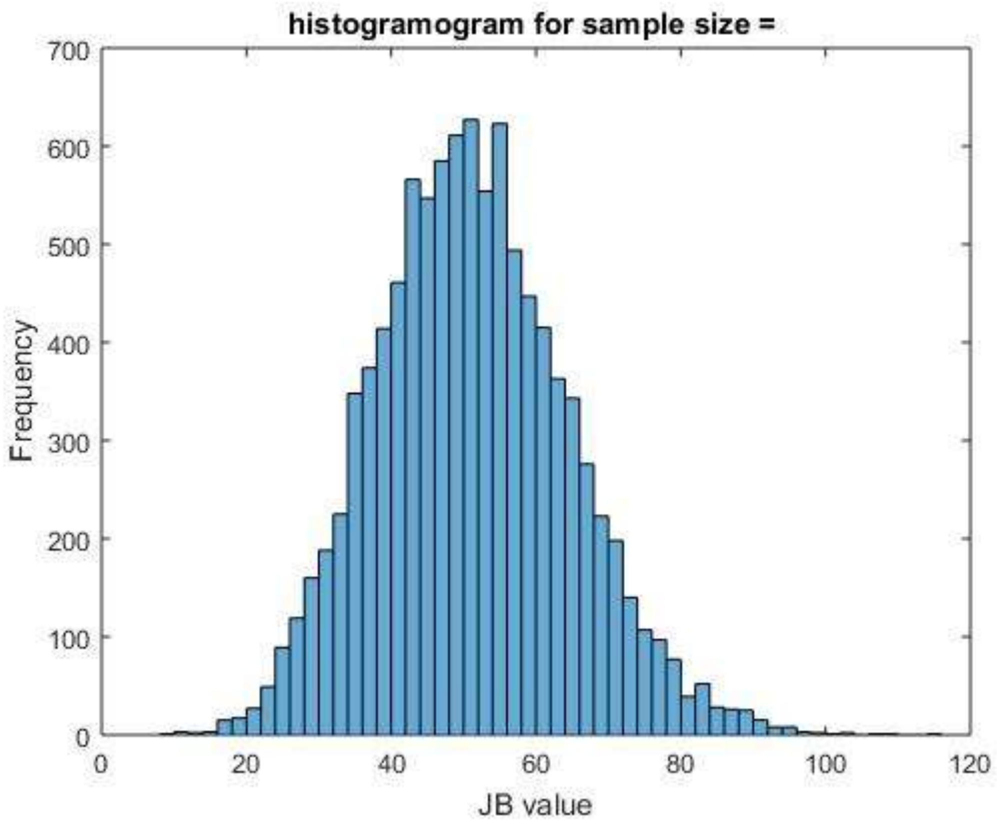
```
confidence_level =

    0.9914
```

Sample size = 50



```
confidence_level =

    0.6549
```

Sample size = 500



confidence_level =

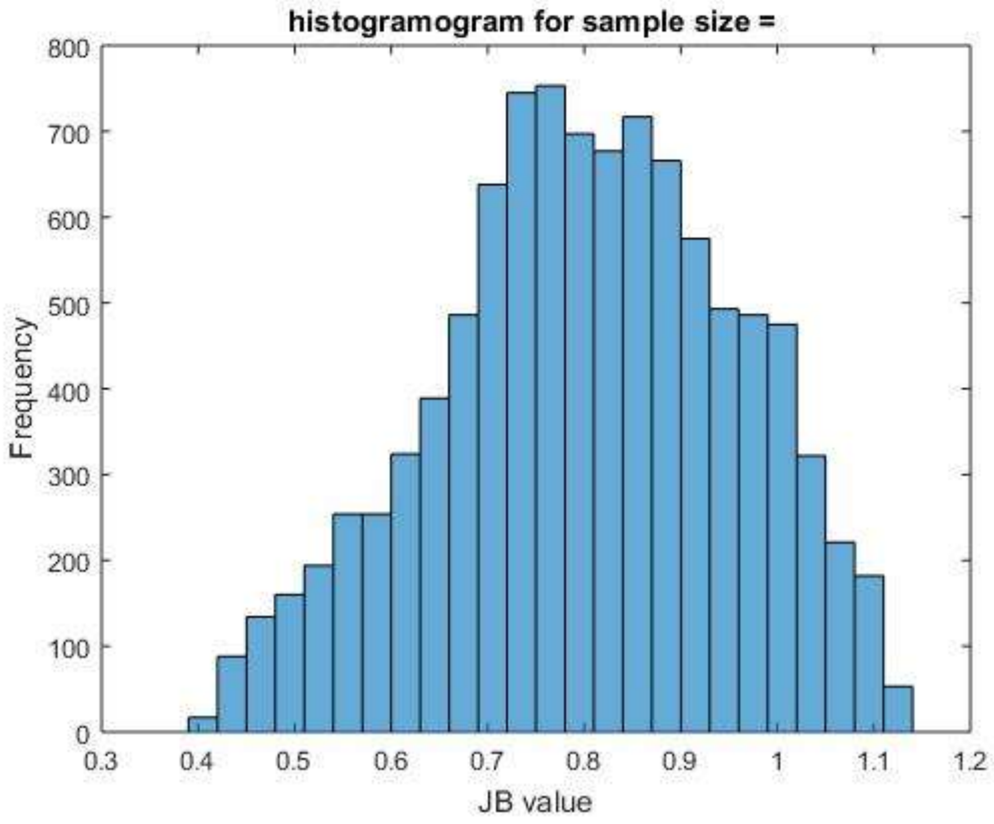
```
    0
```

Among above 3, the first case gives very good answers. It is very close to the normal distribution. So here the problem must be of sample size. On the whole, in 1st case, we can say that the data is normally distributed.

Rest 2 give poor result may be because of large sample size.

Data set: sample_50

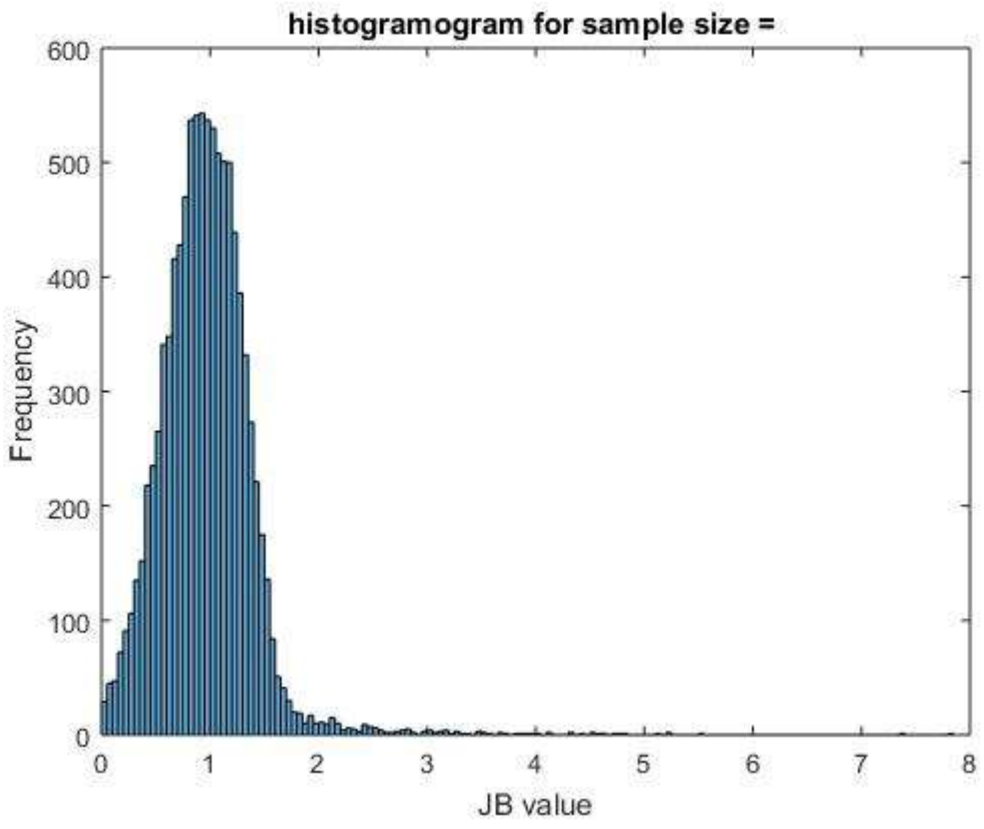
Sample size = 5



confidence_level =

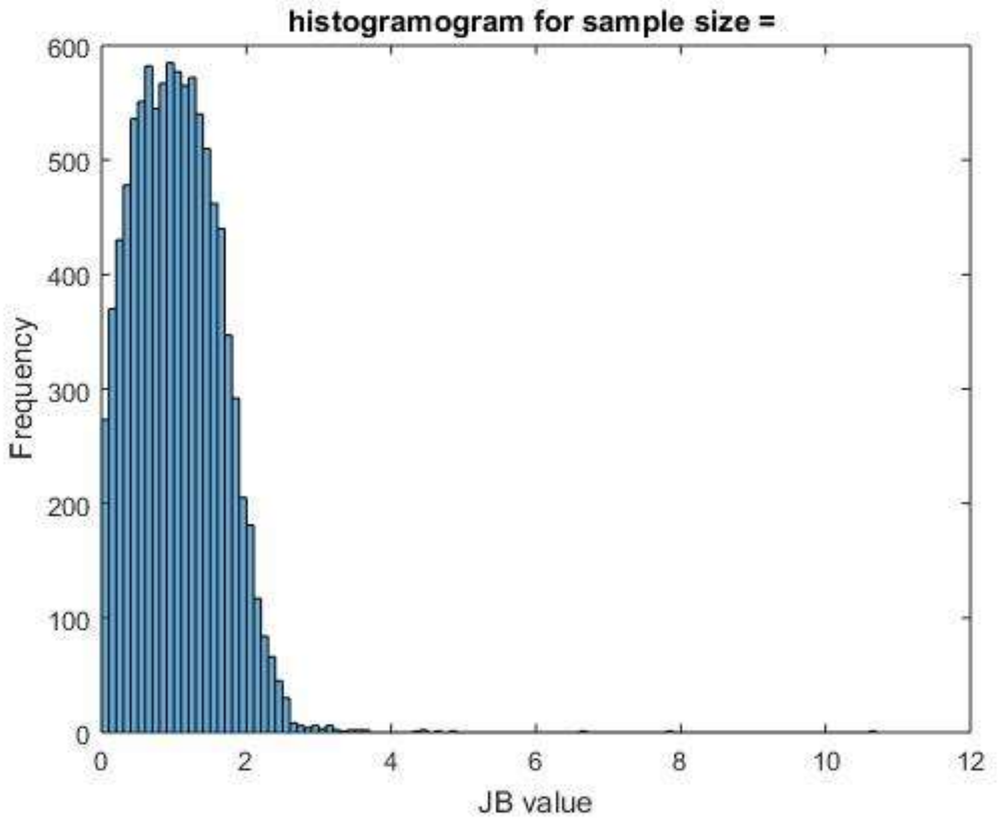
```
    1
```

Sample size = 10



```
confidence_level =

    0.9998
```

Sample size = 25



```
confidence_level =

    0.9997
```

All the above 3 cases indicate that this data set is normally distributed.

Codes

```matlab
clear
close all

load('data_lab4.mat')

%%%%%%%%%%%%%%%%%% data set : population_normal %%%%%%%%%%%%%

population = population_normal;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;

% number of iterations
N = 10000;

% array of sample sizes
sample_size = [50 1500 2500];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_pop = zeros(length(sample_size), 1);

% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end

    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_pop(j) = 1;
    end

    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end
```

```
%%%%%%%%%%%%%%%%%% data set : sample_50k %%%%%%%%%%%%%%

population = sample_50k;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;

% number of iterations
N = 10000;

% array of sample sizes
sample_size = [50 1500 2500];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_50k = zeros(length(sample_size), 1);
```

```
% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end

    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_50k(j) = 1;
    end
    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end
```

```
%%%%%%%%%%%%%%%%%% data set : ammonia_concentration %%%%%%%%%%%%%%

population = ammonia_concentration;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;

% number of iterations
N = 10000;

% array of sample sizes
sample_size = [50 100 500];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_ammo = zeros(length(sample_size), 1);

% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end
    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_ammo(j) = 1;
    end

    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end







%%%%%%%%%%%%%%%%%% data set : score_natural_model %%%%%%%%%%%%%

population = score_natural_model;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;
```

```
% number of iterations
N = 10000;

% array of sample sizes
sample_size = [50 1500 2500];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_sco = zeros(length(sample_size), 1);

% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end

    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
```

```
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_sco(j) = 1;
    end

    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end
```

```
%%%%%%%%%%%%%%%%%% data set : data4 %%%%%%%%%%%%%%

population = data4;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;

% number of iterations
N = 10000;

% array of sample sizes
sample_size = [10 50 500];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_d4 = zeros(length(sample_size), 1);

% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end

    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_d4(j) = 1;
    end

    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end
```

```
%%%%%%%%%%%%%%%%%% data set : sample_50 %%%%%%%%%%%%%%

population = sample_50;

% normalize the data
mu = mean(population);
sigma = std(population);
population= (population - mu) / sigma;

% number of iterations
N = 10000;

% array of sample sizes
sample_size = [5 10 25];

% array of JB test values
jb_arr= zeros(N,(length(sample_size)));

% prediction array 1 = Normal Dist, 0 = Not Normal
prediction_50 = zeros(length(sample_size), 1);

% loop over different sample sizes
for j = 1:length(sample_size)

    % iterate a large number of times
    for i = 1:N

        % this is valid in MATLAB 2015 or higher versions
        permindex= randperm(length(population),sample_size(j));
        sample = population(permindex);

        % for MATLAB 2010
        % sample the given data
        %sample = randsample(population, sample_size(j));

        mu = mean(sample);
        sigma = std(sample);

        skew = sum( ( (sample-mu)./sigma).^3 ) /sample_size(j);
        kurtosis = sum( ( (sample-mu)./sigma).^4 ) /sample_size(j);

        % calc the jb values
        jb_arr(i, j)= sample_size(j)/6*(skew*skew + (kurtosis-3)*(kurtosis-
3)/4);
    end

    %%%%%plot histogramogram

    % matlab 2015
    %histogramogram(jb_arr);

    figure
    histogram(jb_arr(:, j));
    title('histogramogram for sample size = ')
    xlabel('JB value')
    ylabel('Frequency')


    % check from JB values whether the data follows normal distribution
    alpha = 0.05;

    chi2DistVal = 5.991;

    confidence = 0;

    for k = 1:N
        if jb_arr(k, j) <= chi2DistVal
            confidence = confidence + 1;
        end
    end

    confidence_level = confidence / N

    % if confidence level more, prediction = Normal Dist.
    if confidence_level >= (1-alpha)*100
        prediction_50(j) = 1;
    end

    % plot CDF to determine whether sample is normally distributed or not
    figure
    cdfplot(jb_arr(:, j));

end
```

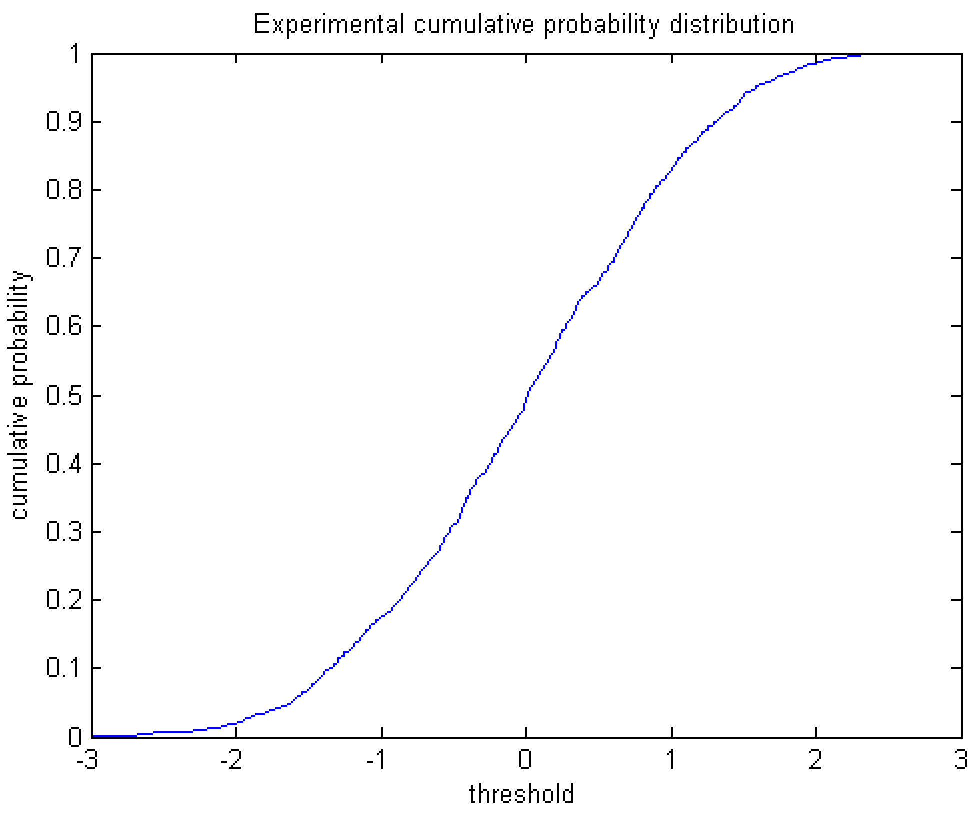
Experiment 2:

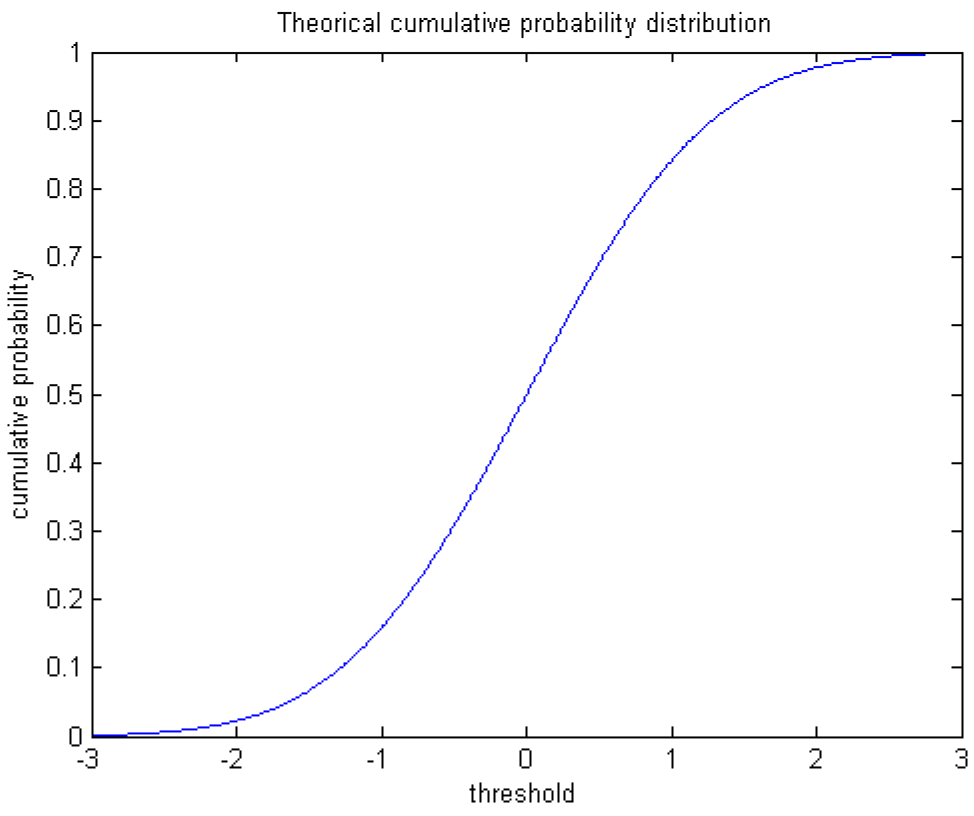
The goal of this experiment is to compare the inferences made based on the given data and those from theoretical model. To that end, obtain the probability that the given data is less than a threshold value in two cases: (a) from the given data, (b) using the theoretical normal distribution. You should repeat this for a large number of threshold values to generate a set of probabilities for the two cases.

1. Compute the mean squared error (MSE) between theoretical and observed probabilities (across all threshold values), for all the 5 sample sets. Analyze the resultant MSE values in the light of the normality as determined by the JB test in the previous experiment.

Ammonia data-

The plots are as follows:

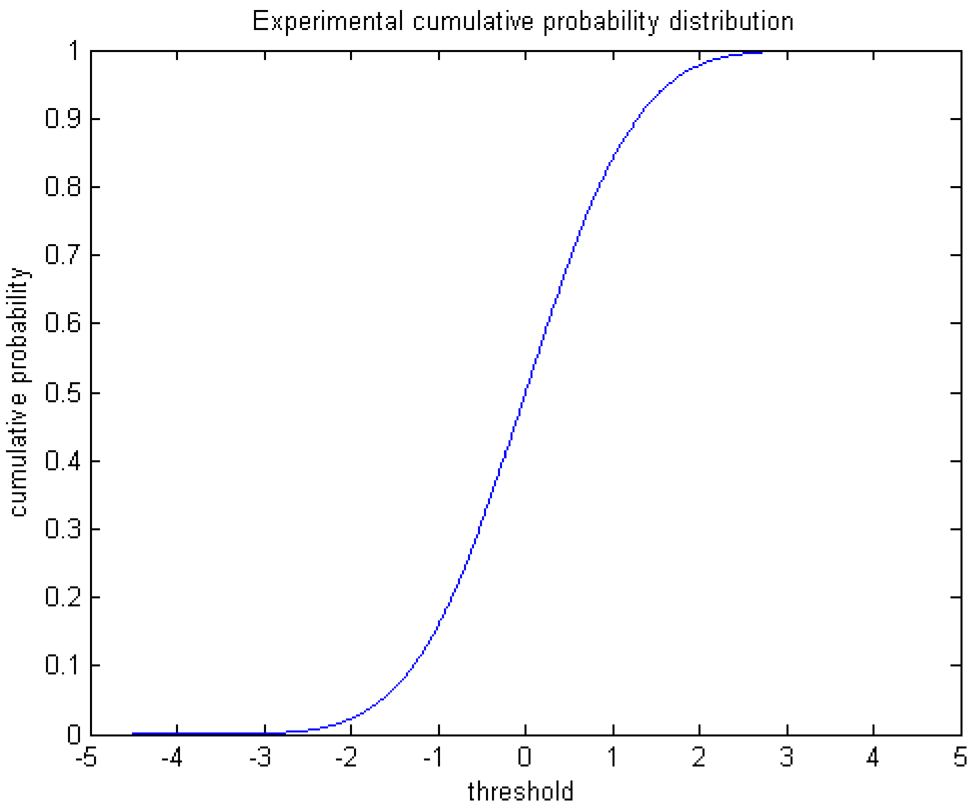


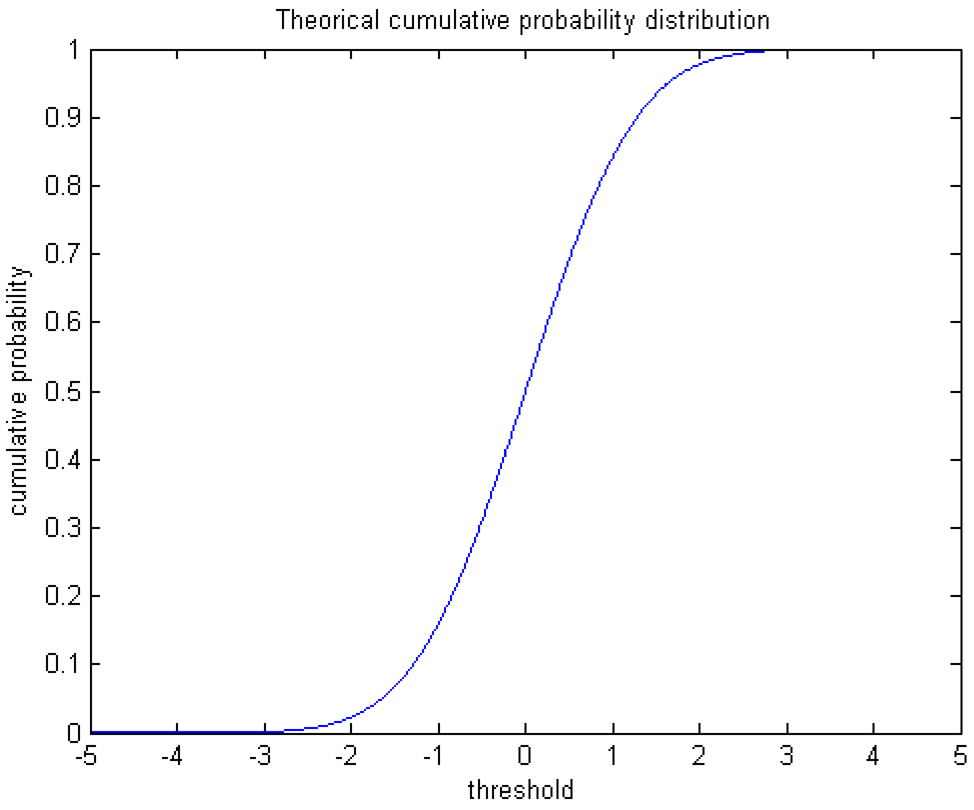


MSE= 8.5952e-005

The MSE is very less, this means that the assumption of normality is correct. The cdf plots too are very similar. So Ammonia dataset is normal.

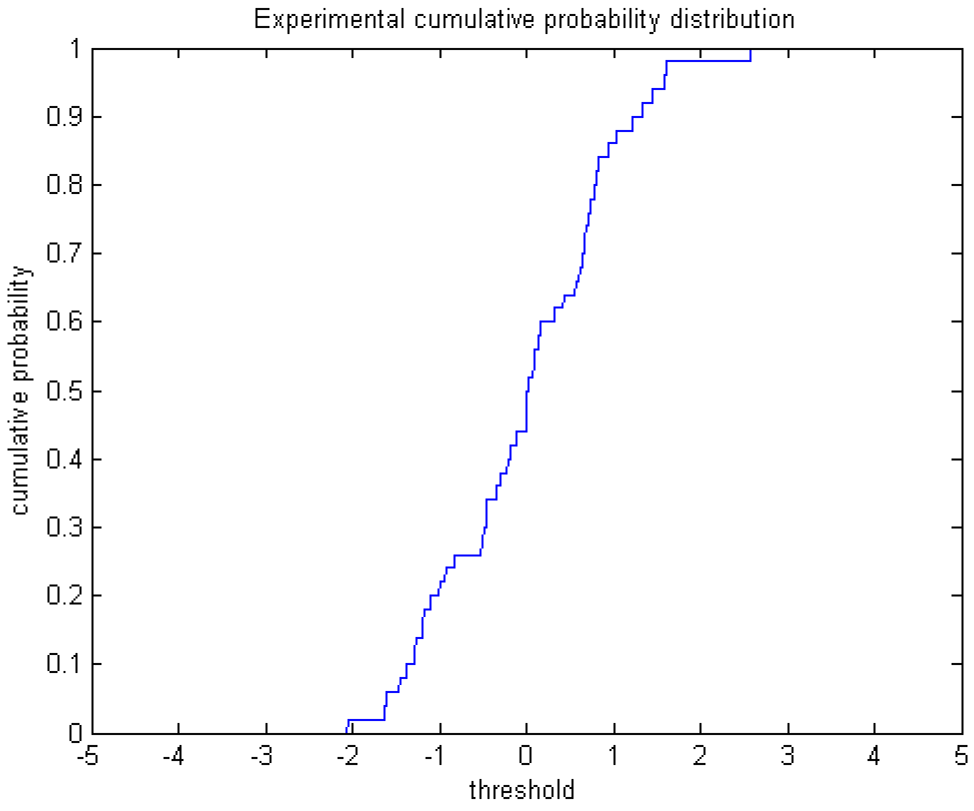
Sample_50k dataset

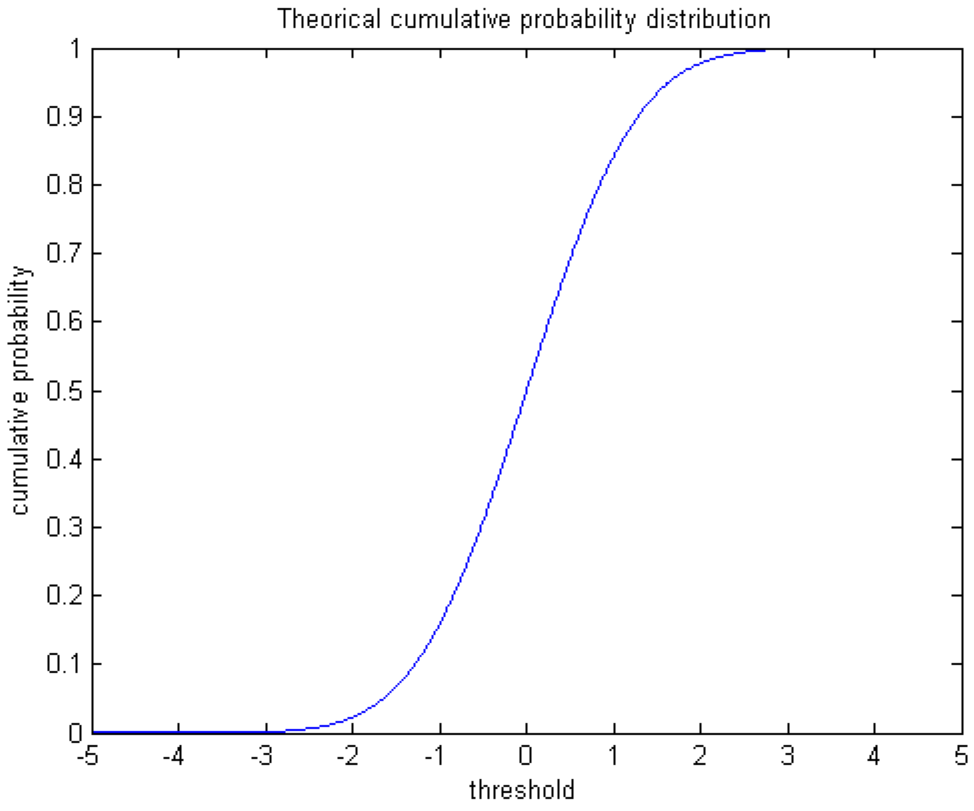




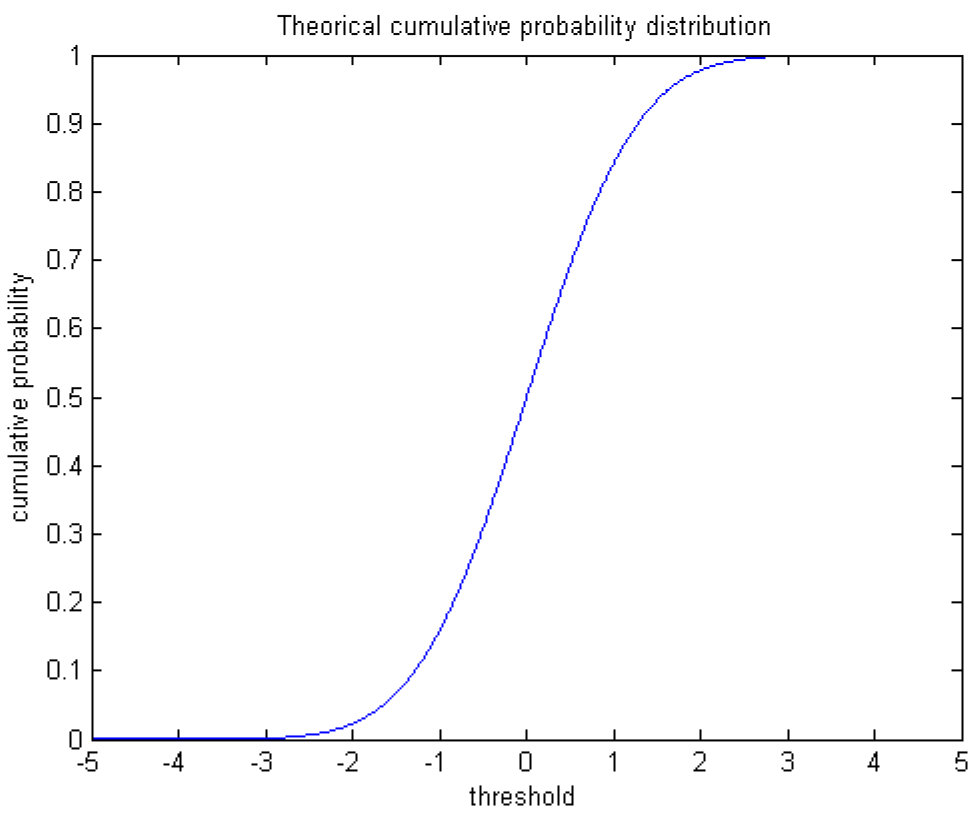
MSE- 2.9982e-007

Dataset Sample_50

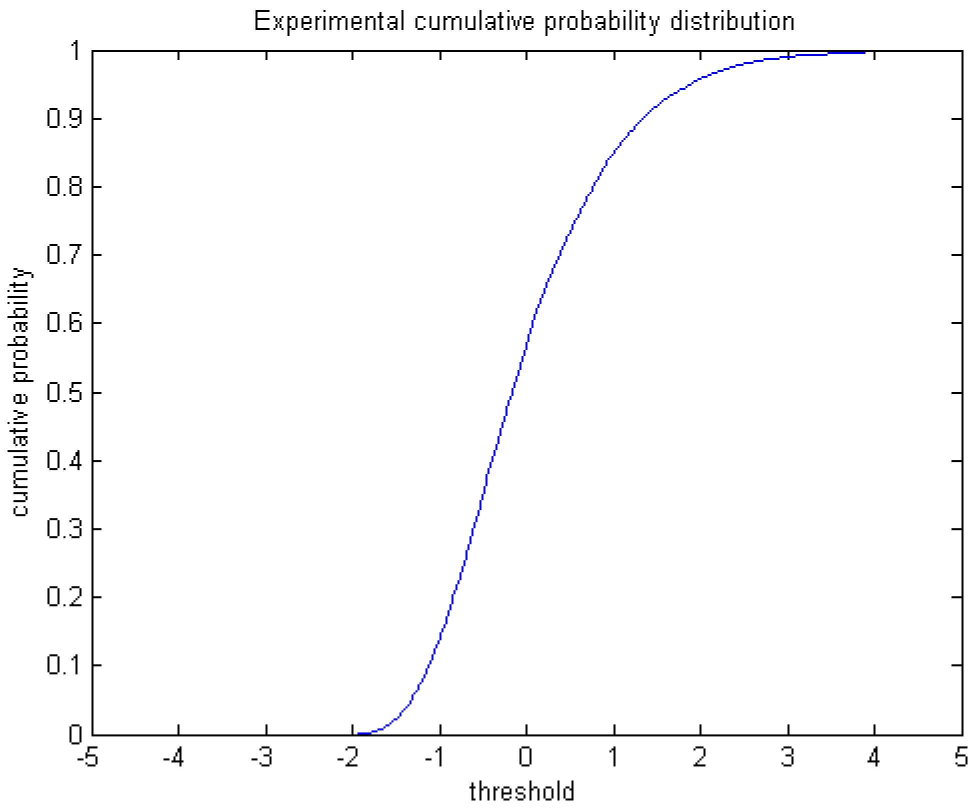




MSE- 3.1698e-004



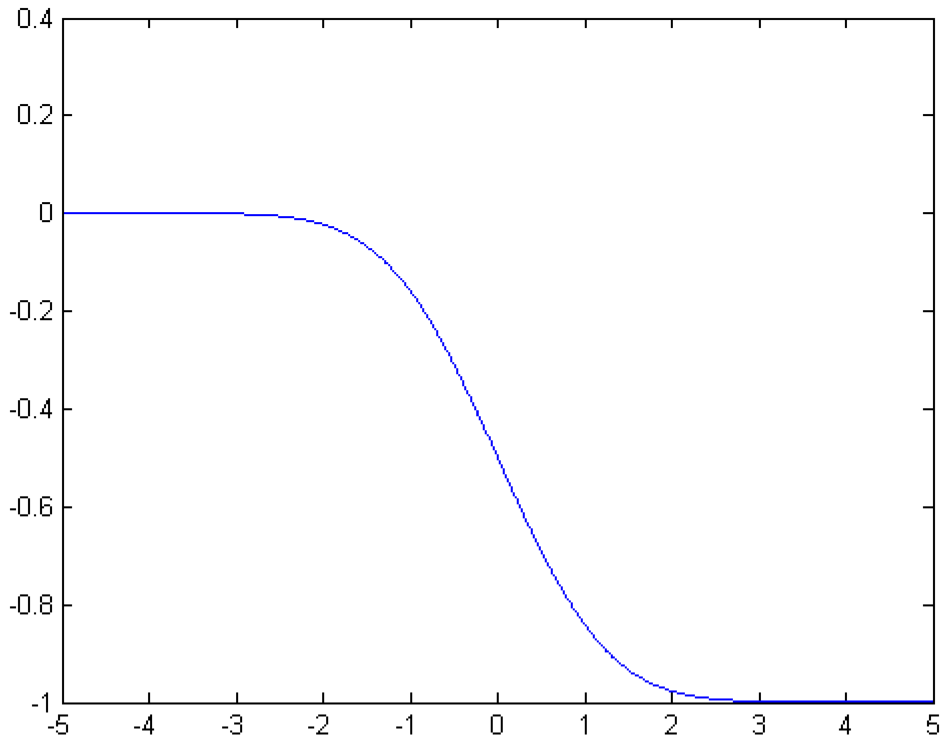
Theorical cumulative probability distribution
cumulative probability
threshold
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
-5
-4
-3
-2
-1
0
1
2
3
4
5



MSE- 6.2393e-004

Thus, most of these datasets are approximately normally distributed. The natural score model shows a slightly more MSE and its CDF is a little skewed to the left.

Are the differences between the theoretical and observed probability values same for all the chosen threshold values, yes/no? Give possible reason(s) for your answer.



Plot of

Plot of errot between theoritical and experimental probability across the range of threshold considered. Clearly the error is not the same for all values. Error depends on the pdf values of the distributions. We have asssumed that the data is normally distributed. But the actual experimental behaviour can be completely different. Hence the error at each point too will be different.

Codes-

Q2:

```
clear;
close all;

load('data_lab4.mat');

ammonia = ammonia_concentration;
N = 60;


mu = mean(ammonia);
sigma = std(ammonia);

dx= 1e-2;

points= 6/dx;

experi_thresprob= zeros(points,1);
theori_thresprob= zeros(points,1);
mse=zeros(points,1);
% for a threshold compute theorti and experi probs

stan_ammo = ammonia;

stan_ammo = (stan_ammo - mu)/sigma;



i=1;
length= length(stan_ammo);

for thres=-5:dx:5

    % experimental prob less than thres
    %exam=find(stan_ammo<=0);
    %size(exam,1)
   experi_thresprob(i)= size((find(stan_ammo<=thres)),2) / length;

   p = normcdf([-10 thres]);
   theori_thresprob(i)= p(2)-p(1);

   mse(i)= experi_thresprob(i)- theori_thresprob(i);
   i= i+1;
end

%remember to reset i;
```

```
thres_range= -5:dx:5;
figure
plot(thres_range,experi_thresprob);
title('Experimental cumulative probability distribution');
xlabel('threshold');
ylabel('cumulative probability');

figure
plot(thres_range,theori_thresprob);

title('Theorical cumulative probability distribution');
xlabel('threshold');
ylabel('cumulative probability');

figure
plot(thres_range,mse);

disp('mse')
N= size(theori_thresprob,1);
sum((theori_thresprob-experi_thresprob).^2)/N
```