PABNA UNIVERSITY OF SCIENCE AND TECHNOLOGY

Faculty of Engineering & Technology
Department of Information and Communication Engineering

LAB REPORT

Engineering Statistics Sessional
STAT- 2202

Submitted To :

Dr. Md. Sarwar Hosain
PhD (Saitama University, Japan), MPhil (PUST)
Associate Professor of
Department of ICE, PUST

Submitted By :

Apurbo Sharma
Roll : 220604
Student of 2021-22 session
Department of ICE, PUST

Submission Date : 25-02-2025

Index

Lab 01 : Verification of Fisher's Lemma using simulated Data from Normal Distributions

Theory:

Fisher's Lemma states that the fisher information for a parameter in a Probability distribution is the negative expectation of the second derivative of the log-likelihood function. Mathematically, it is expressed as:

$$I(\theta) = -E\left[\frac{\partial^2}{\partial\theta^2} \log f(X;\theta)\right]$$

Where:

- $I(\theta)$ is the fisher Information
- $f(X;\theta)$ is the Probability density function of the data
- θ is the parameter of interest

For a normal distribution $X \sim N(\mu, \sigma^2)$, the fisher information for μ and σ^2 is given by.

$$I(\mu) = \frac{1}{\sigma^2} I(\sigma^2) = \frac{1}{2\sigma^4}$$

Our goal is to simulate normal data, compute the second derivative of the log-likelihood, a verify Fisher's Lemma empirically.

Objectives

The main objective of this experiment is to verify Fisher's Lemma by comparing the empirical estimation of Fisher's information with it's theoretical value.

Pseudo code:

1. set parameters: mean(), standard deviation(), sample size(), and number of simulations().
2. Initialize empty vectors for sample means and sample variance.
3. Repeat ~~for~~ to:
   - Generate a random sample from
   - Compute the sample mean and variance
   - store values in respective vectors.
4. Compute the correlation between sample means and sample variance.
5. Display correlation result
6. set up 2x2 graphical layout
7. Generate histograms for sample means and sample variances with theoretical curves.
8. Create a scatter plot of sample means vs. sample variances with a reference line.

R code:

```
mu <- 5
sigma <- 2
n <- 30
N_sim <- 1000
# Initialize vectors
sample_means <- numeric(N_sim)
sample_vars <- numeric(N_sim)
```

```
# Creats two empty numeric vectors of length N-sim to
store sample means and sample variances.

# simulation
set.seed(123)
for(i in 1:N_sim)
{
data <- norm(n,mean=mu,sd=sigma)
sample_means[i] <- mean(data)
sample_vars[i] <- var(data)
}

# check correlation
correlation <- cor(sample_means, sample_vars)
Print(Paste("correlation between sample mean and sample
variance:") correlation))

# computes the correlation between sample_means
and sample variance using pearson's correlation
coefficient.

# Displays the correlation value

# Graphical output
Par(mfrow=c(2,2))

# Divides the Plotting area into a 2x2 grid for four
Plots.

# Histogram and sample means
hist(sample_means, breaks=30, col="lightblue", main
="Distribution of sample means", xlab="Sample mean" border=
```

```
curve( dnorm( x, mean = mu, sd = sigma / sqrt(n)),
add = TRUE, col = "red", lwd = 2)

# Histogram of sample variances
hist( sample_vars, break = 30, col = "Light green",
main = "Distribution of sample variances", xlab =
"sample variance", border = "white")

curve( dchisq((x * (n-1)) / sigma^2, df = (n-1) / sigma^2,
add = TRUE, col = "blue", lwd = 2)

# Scatterplot of sample means vs. sample variances
plot( sample_means, sample_vars, pch = 19, col = rgb
(0, 0, 1, 0.5), main = "sample mean vs. sample
variance", xlab = "Sample mean", ylab = "sample
variance")

abline(h = sigma^2, col = "red", lwd = 2)

# Q-Q plot for sample means
qqnorm (sample_means, main = "Q-Q plot for
sample means", col = "blue")

qqline (sample_means, col = "red", lwd = 2)
```

Sample Input and output :

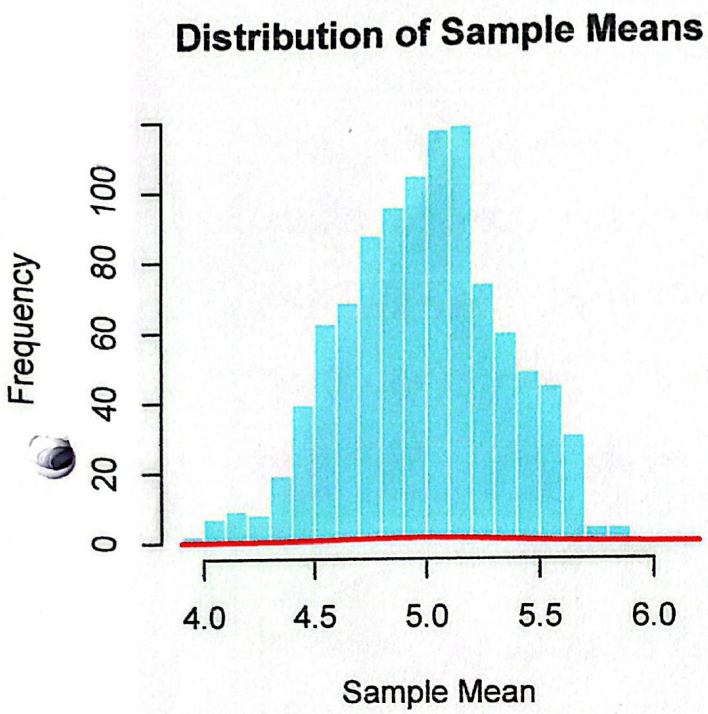
Input Parameters:

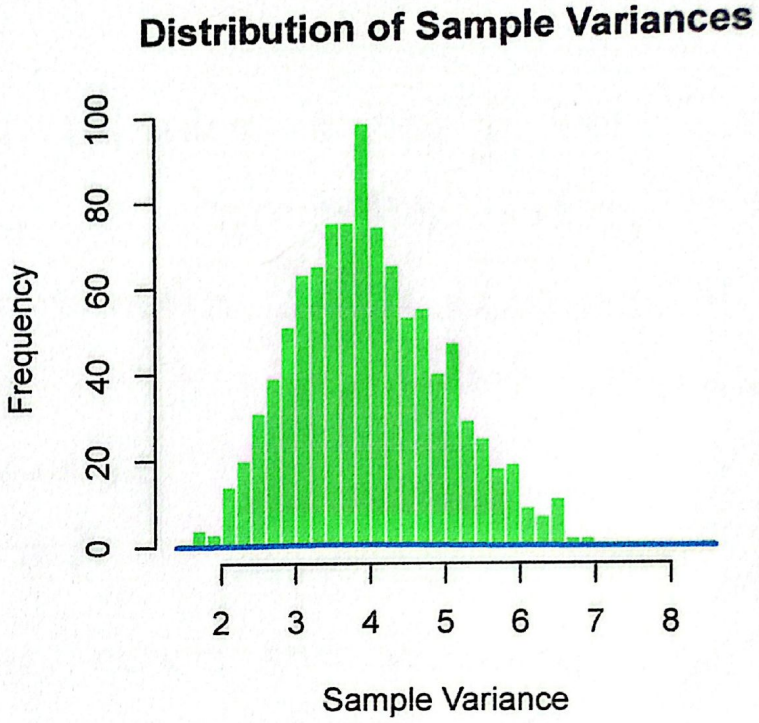
- Mean (): 5
- Standard deviation(): 2
- Sample size (): 30

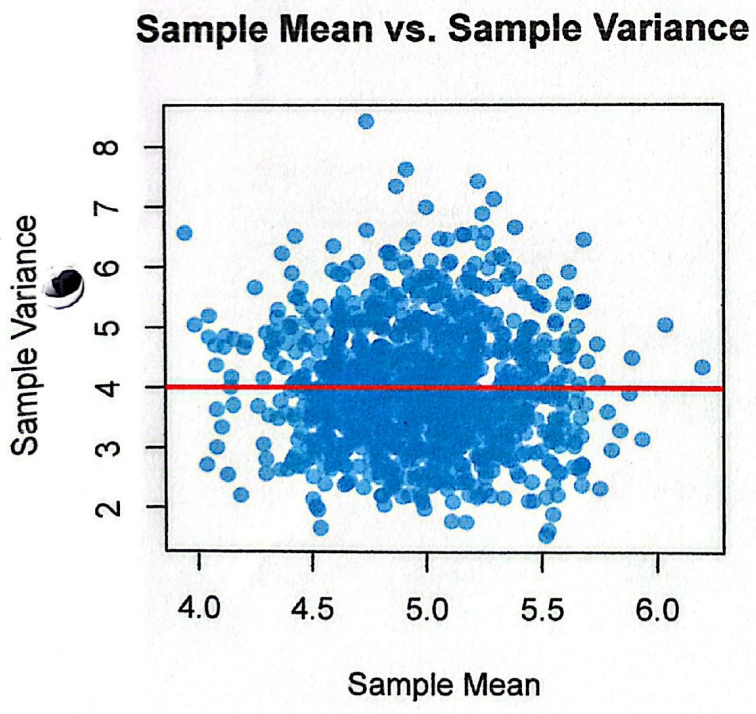
- Number of simulations() : 1000

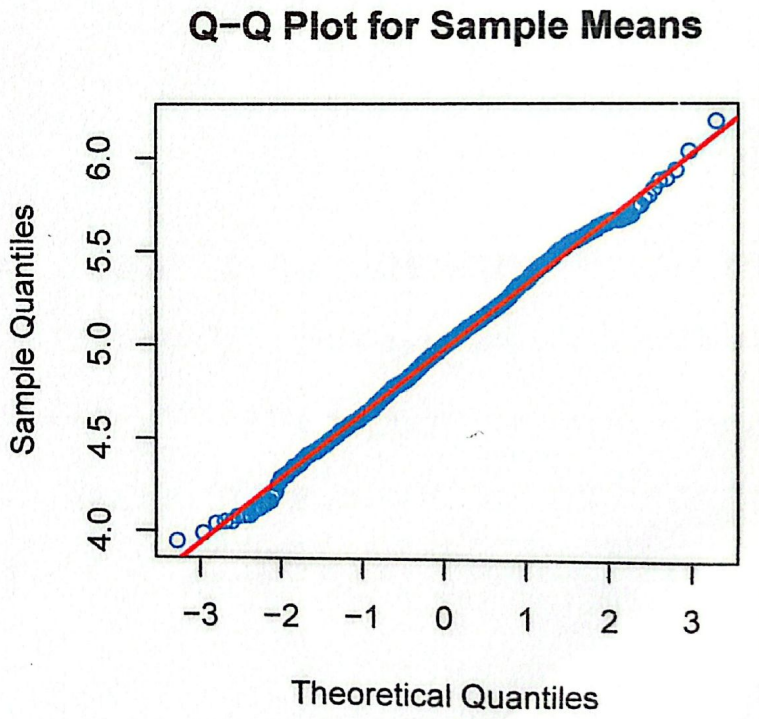
Output parameters:

Correlation between sample mean and sample variance : -0.028896015724436 8


Distribution of Sample Means
Frequency
0
20
40
60
80
100
4.0
4.5
5.0
5.5
6.0
Sample Mean


Distribution of Sample Variances
Frequency
0
20
40
60
80
100
2
3
4
5
6
7
8
Sample Variance


Sample Mean vs. Sample Variance
Sample Variance
2
3
4
5
6
7
8
4.0
4.5
5.0
5.5
6.0
Sample Mean


Q−Q Plot for Sample Means
Sample Quantiles
4.0
4.5
5.0
5.5
6.0
−3
−2
−1
0
1
2
3
Theoretical Quantiles

Lab 02: Generation and Analysis of X^2 - Distributed Data

Theory:

The chi-square (X^2) distribution is widely used in statistics, particularly in hypothesis testing and confidence interval estimation. It is defined as the sum of the squares of k independent standard normal random variables:

$$X = Z_1^2 + Z_2^2 + \dots + Z_k^2$$

Where $Z_i \sim N(0,1)$. The mean and variance of a X^2 distribution with k degrees of freedom are:

- $E[X] = k$
- $\text{Var}[X] = 2k$

Objective:

The objective of this lab is to generate X^2 - distribution data, compute empirical mean and variance, and compare them with theoretical values. Additionally, we visualize the distribution using histograms, density plots, and Q-Q plots.

Pseudo code:

1. Set Parameters: degrees of freedom and number of simulations.
2. Generate random samples from a X^2 distribution.

3. Compute empirical mean and variance.

4. Print computed mean and variance.

5. Generate and display:
   - Histogram with theoretical χ^2 density curve.
   - Density Plot with theoretical χ^2 curve
   - Q-Q plot comparing sample quantities to theoretical quantites.

R code:

```
K <- 5
N_sim <- 1000
set.seed(123)
chi2_data <- rchisq(N_sim, df = K)
r chisq(N_sim, df = K)
mean_chi2 <- mean(chi2_data)
var_chi2 <- var(chi2_data)

Print(Paste("mean:", mean_chi2))
Print(Paste("variance:", var_chi2))

Par(mfrow = c(1,3))
hist(Chi2_data, breaks = 30, col = "lightblue", Probability
= TRUE, main = "Chi-squared Distribution", xlab = "value",
 border = "white")
curve(dchisq(x, df = K), add = TRUE, col = "red", lwd = 2)
```

```
qqplot(qchisq(ppoints(N_sim), df = K), chi2_data, main
= "Q-Q plot for Chi-squared Data", col = "blue",
xlab = "Theoretical Quantities", ylab = "sample Quant-
iles")

abline(0,1, col = "red", lwd = 2)
```

Sample Input/output:

Input Parameters:

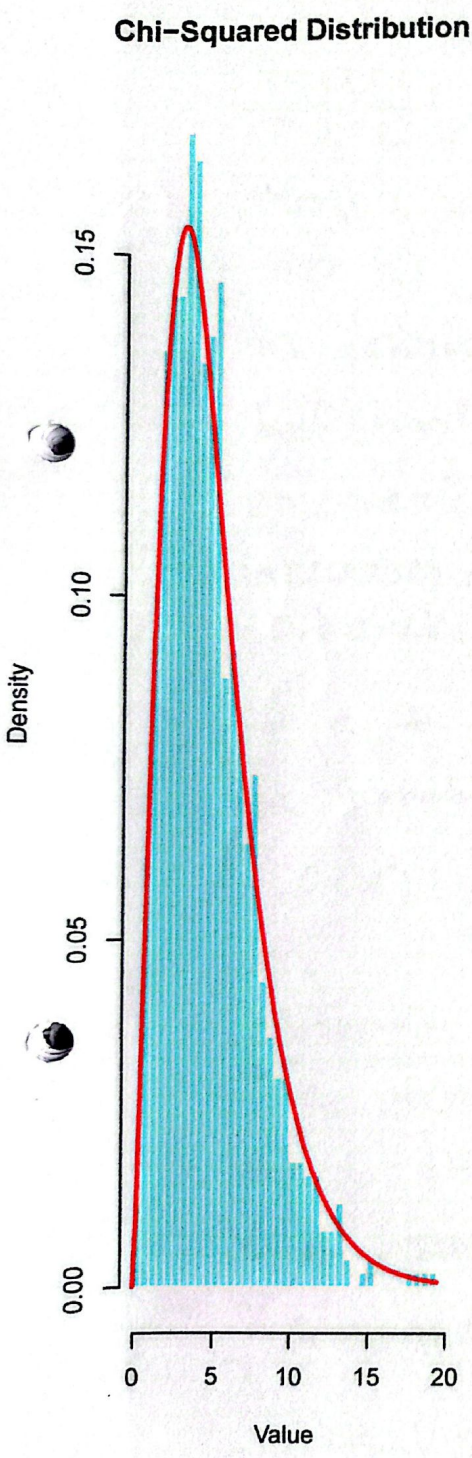
K = 5

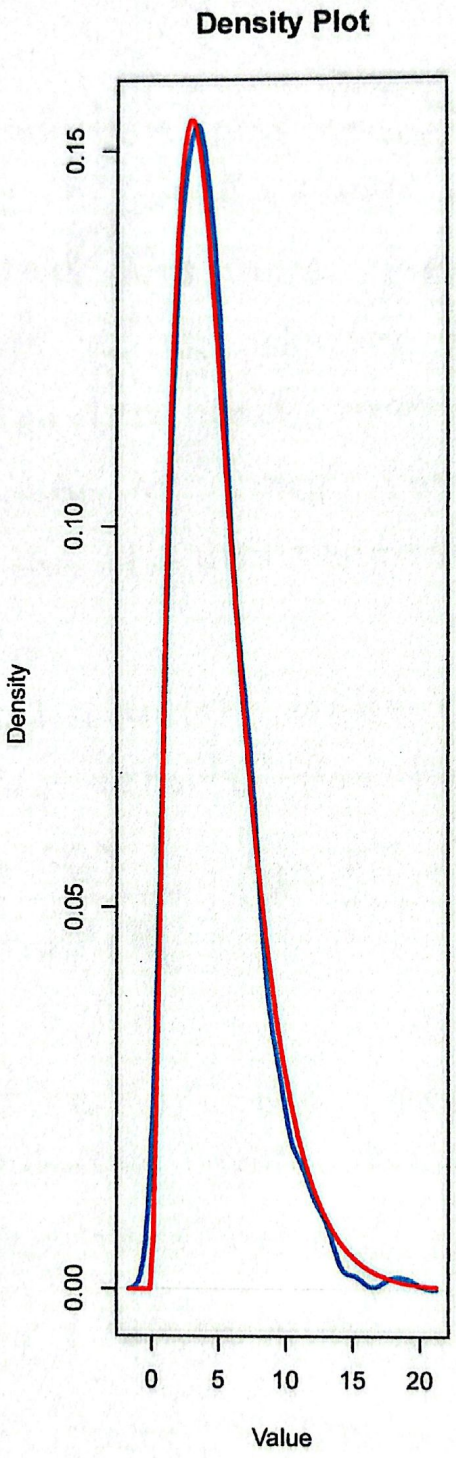
N_sim = 1000

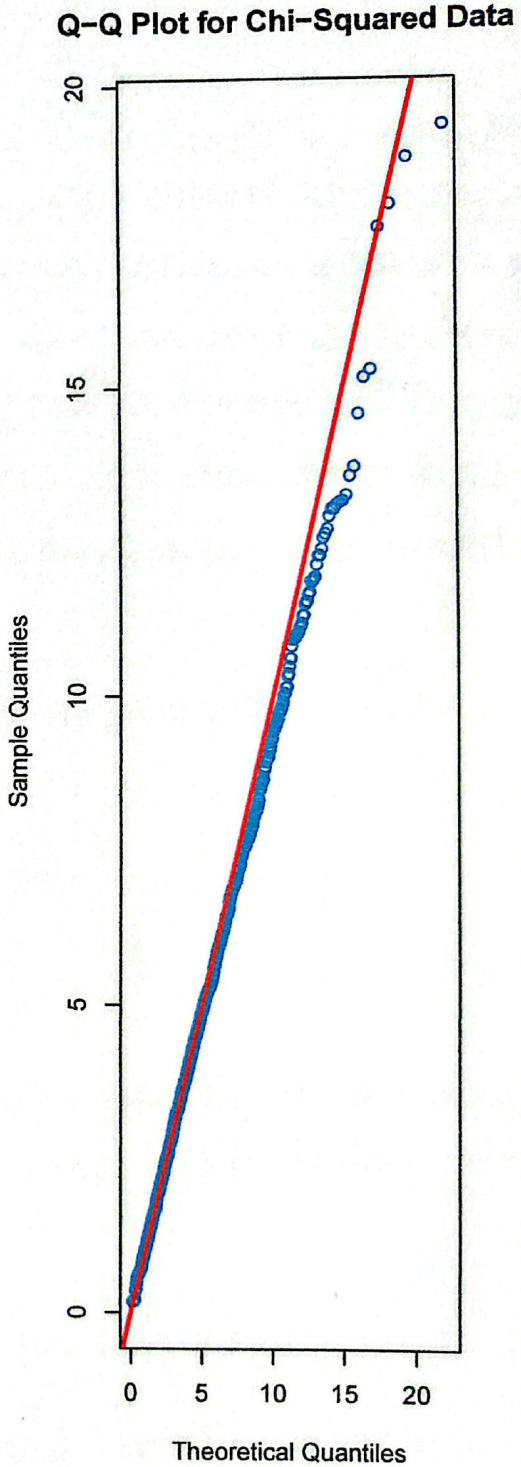
Output Parameters:

Mean: 4.793899857081 51

Variance: 8.401792490174 5


Chi-Squared Distribution
Density
0.00
0.05
0.10
0.15
0
5
10
15
20
Value


Density Plot
Density
0.00
0.05
0.10
0.15
0
5
10
15
20
Value


Q-Q Plot for Chi-Squared Data
Sample Quantiles
0
5
10
15
20
Theoretical Quantiles

Lab 03: Comparison of t-Distribution with Normal Distribution for small sample sizes.

Theory:

The t-distribution, also known as student's t-distribution, is used in statistical analysis when the sample size is small and the population standard deviation is unknown. It is similar to the normal distribution but has heavier tails, meaning it accounts for higher variability in small samples. As the sample size increases, the t-distribution approaches the normal distribution.

For a t-distribution with $n-1$ degrees of freedom, the probability density function is,

$$f(x) = \frac{\Gamma\left(\frac{n}{2}\right)}{\sqrt{n\pi}\,\Gamma\left(\frac{n-1}{2}\right)} \left(1 + \frac{x^2}{n-1}\right)^{-\frac{n}{2}}$$

Objective:

The objective of the experiment is to compare the t-distribution with the normal distribution for small sample sizes by:

- Generating t-distribution and normal data
- Visualizing their histograms and density functions.
- Comparing quantities using Q-Q plots.

Pseudocode :

1. Set parameters : sample size and Number of simulations.

2. Generate random values from t-distribution with degrees of freedom

3. Generate random values from a standard normal distribution.

4. Plot :
   - Histogram of t-distribution with theoretical density.
   - Histogram of normal distribution with theoretical density.
   - Density plots of both distributions for comparison.
   - Q-Q Plot for t-distribution against theoretical quantities.

R-Code :

```
n <- 10
N_sim <- 1000
set.seed(123)
t_data <- rt(N_sim, df=n-1)
normal_data <-norm(N_sim)
Par (mfrow = c(2,2)
```

```
hist(t_data, breaks = 30, col = "light blue", Probability =
TRUE, main = "t-Distribution", xlab = "value", border =
"white")

curve(dt(x, df = n-1), add = TRUE, col = "red", lwd = 2)

hist(normal_data, breaks = 30, col = "light green", Proba-
bility = TRUE, main = "Normal Distribution", xlab = "Value",
border = "white")

curve(dnorm(x), add = TRUE, col = "blue", lwd = 2)
Plot(density(t_data), col = "red", lwd = 2, main = "Densi-
ty Comparison", xlab = "value", ylim = c(0, 0.4))
liner(density(normal_data), col = "blue", lwd = 2)
legend("topright", legend = c("t-Distribution", "Normal
Distribution"), col = c("red", "blue"), lwd = 2)

qq Plot(qt(ppoints(N_sim), df = n-1), t_data, main =
"Q-Q Plot for t-Distribution", col = "red", xlab = "Theo-
retical Quantities", ylab = "sample Quantities")
abline(0, 1, col = "blue", lwd = 2)
```

Sample Input / Output :

Input Parameter :

n = 10

N sim = 1000

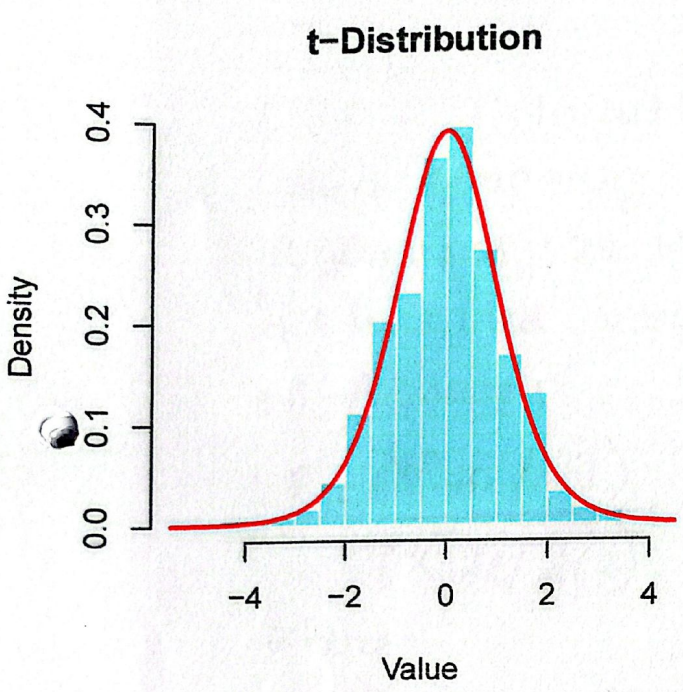
Output (Graphical Plots):

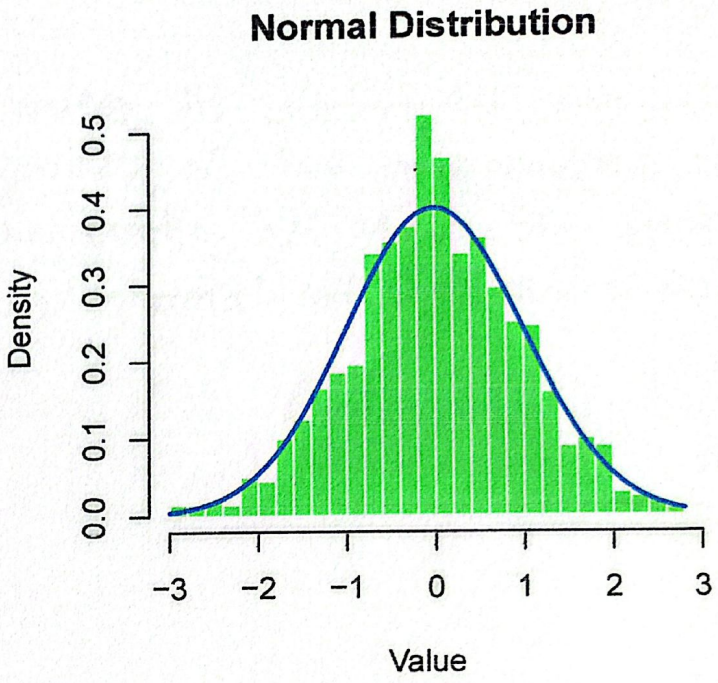
1. Histogram of t-Distribution (slightly heavier tails than normal)

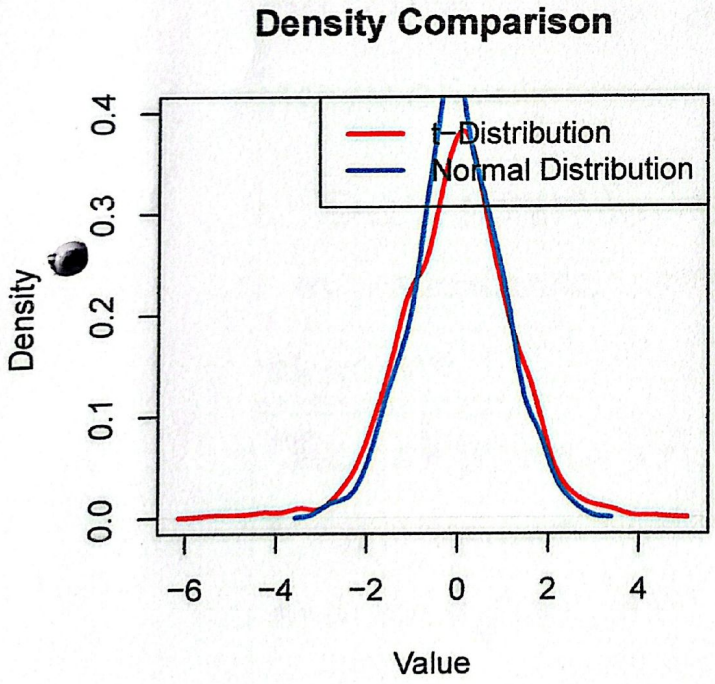
2. Histogram of Normal Distribution (Bell-shaped curve)

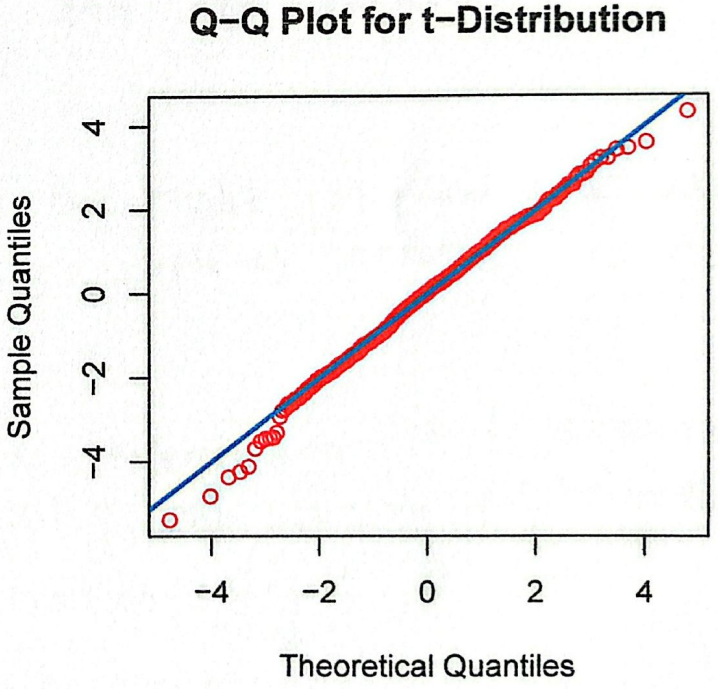
3. Density comparison plot (shows wider tails for t-Distribution)

4. Q-Q Plot (Compares sample t-data to theoretical quantiles)


t−Distribution
Density
0.0
0.1
0.2
0.3
0.4
−4
−2
0
2
4
Value


Normal Distribution
Density
0.0
0.1
0.2
0.3
0.4
0.5
−3
−2
−1
0
1
2
3
Value


Density Comparison
t−Distribution
Normal Distribution
Density
0.0
0.1
0.2
0.3
0.4
−6
−4
−2
0
2
4
Value


Q−Q Plot for t−Distribution
Sample Quantiles
−4
−2
0
2
4
−4
−2
0
2
4
Theoretical Quantiles

Lab 04 : Simulation of F-Distribution Data and it's Relationship with χ^2 - Distribution.

Theory :

The F- distribution arises in statistical analysis when comparing variances of two independent normal populations. It is the ratio of two chi-square distributed variables normalized by their degrees of freedom :

$$F = \frac{(\chi^2_{df1}/df1)}{(\chi^2_{df2}/df2)}$$

where,

- χ^2_{df1} and χ^2_{df2} are chi-square distributed random variables with df1 and df2 degrees of freedom, respectively.
- The F- distribution is right skewed and used in variance analysis, such as ANOVA

Objective :

The objective of this experiment is to simulate F-distributed data using chi-square distributions and analyze its properties using :

- Histogram and density plots
- Q-Q Plot for distribution verification
- Boxplot to observe spread and skewness

Pseudo code:

1. Set Parameters: numerator degrees of freedom, denominator degrees of freedom, and number of simulations.

2. Generate chi-squared distributed random numbers for both numerator and denominator.

3. Compute the F distributed values using their ratio

4. Generate graphical outputs:
   - Histogram with theoretical density curve
   - Density plot overlaying the theoretical curve
   - Q-Q Plot comparing sample quantiles to theoretical quantiles.
   - BOXPlot to visualize spread and skewness.

R-code:

```
df1 <- 5
df2 <- 10
N_sim <- 1000

set.seed(123)
chi2_1 <- rchisq(N_sim, df= df1)
chi2_2 <- rchisq(N_sim, df= df2)
f_data <- (chi2_1/df1)/(chi2_2/df2)

par(mfrow=c(2,2))
hist(f_data, breaks=30, col="lightblue", probability=
*TRUE, main="F-Distribution", xlab="value", border="white")
```

```
curve(df(x,df1, df2 = df2), add = TRUE, col = "red", lwd = 2)
plot(density(f_data), col = "blue", lwd = 2, main = "Density Plot", xlab = "value")
curve(df(x,df1, df2 = df2), add = TRUE, col = "red", lwd = 2)
qqplot(qf(ppoints(N_sim), df1, df2 = df2), f_data, main = "Q-Q plot for F-Distribution", col = "blue", xlab = "Theoretical Quantiles", ylab = "sample Quantiles")
boxplot(f_data, col = "lightgreen", main = "Boxplot of F-Distributed data", ylab = "Value")
```

sample Input/output :

Input parameter :

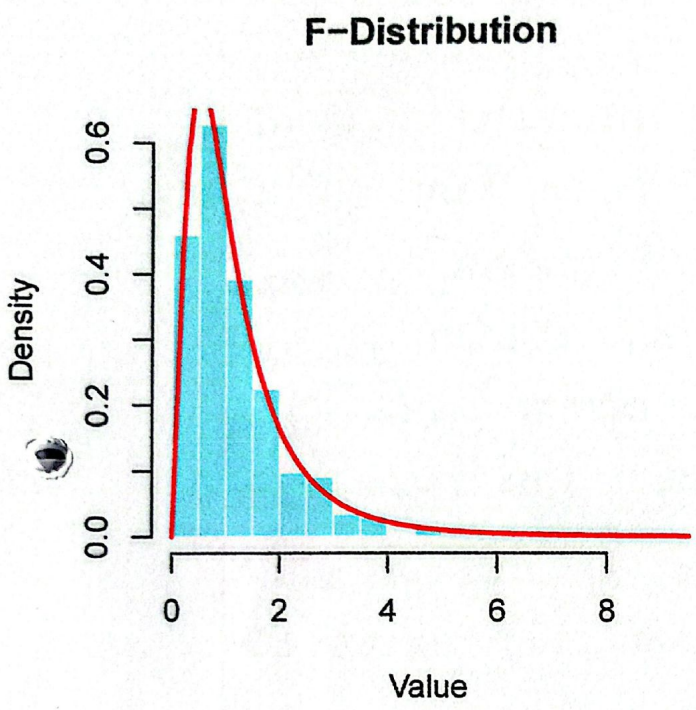
df1 = 5

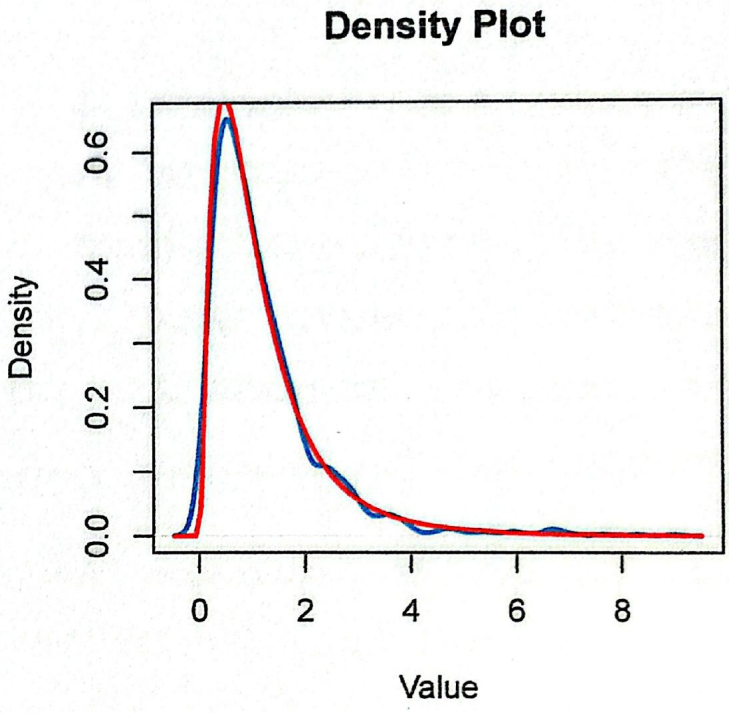
df2 = 10

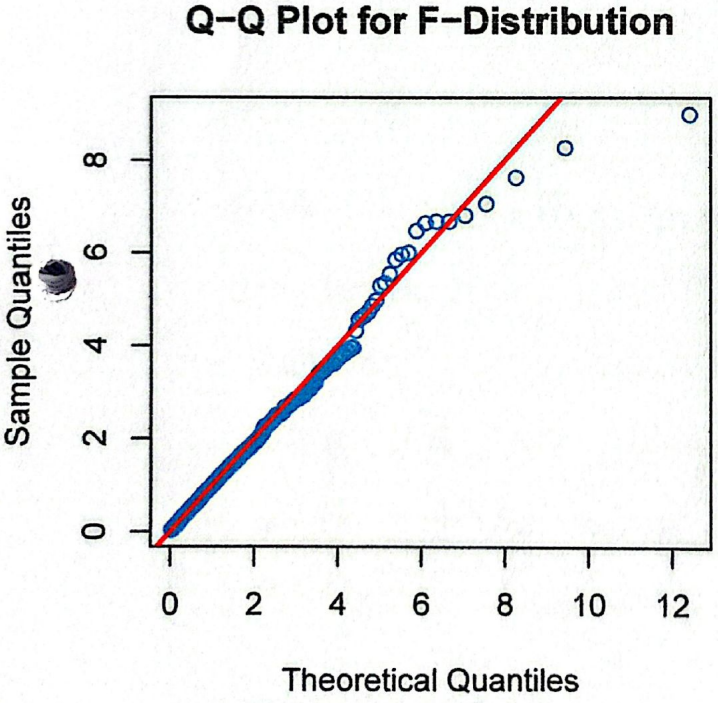
N_sim = 1000

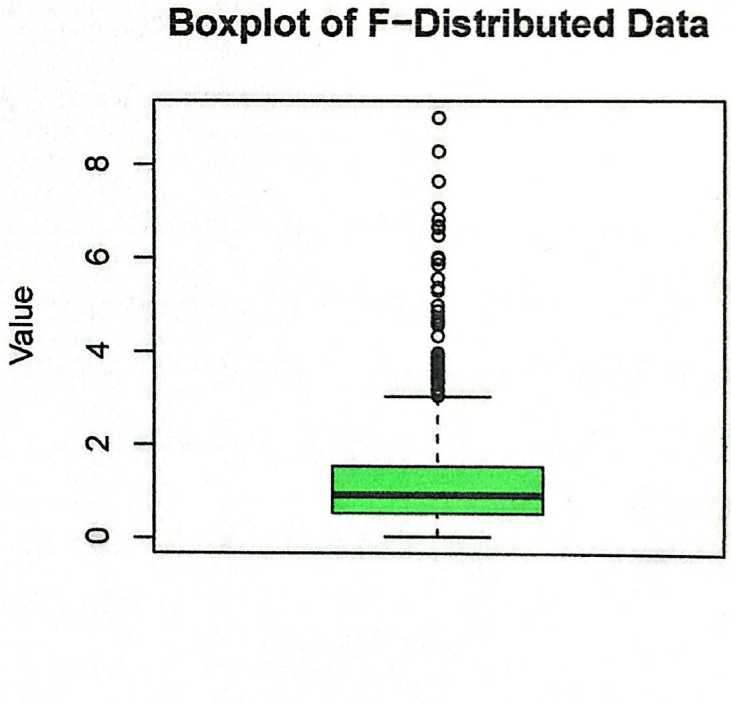
output :

- Histogram of f distribution with theoretical density curve.
- Density plot showing empirical and theoretical distribution.
- Q-Q plot indicating the goodness of fit.
- Boxplot illustrating the skewness of the F-Distribution.


F−Distribution
Density
0.0
0.2
0.4
0.6
0
2
4
6
8
Value


Density Plot
Density
0.0
0.2
0.4
0.6
0
2
4
6
8
Value


Q−Q Plot for F−Distribution
Sample Quantiles
0
2
4
6
8
0
2
4
6
8
10
12
Theoretical Quantiles


Boxplot of F−Distributed Data
Value
0
2
4
6
8

Lab 05: Distribution of Medians and Ranges from sampled Populations.

Theory:

In this lab, the goal is to examine the distributions of the medians and ranges computed from multiple samples drawn from a normal distribution. The median is the middle value when the data is sorted, and the range is the difference between the maximum and minimum values in the sample.

- The median is a robust measure of central tendency, especially in the presence of outliers.
- The range provides a measure of the spread or variability in the sample data, but it can be heavily influenced by outliers.

By simulating multiple samples from a normal distribution, we can investigate how the distributions of medians and ranges behave across repeated sampling. For each sample, we calculate both the median and the range, and then analyze their distribution.

Objective:

1. Simulate 1000 random samples from a normal distribution with mean and standard deviation, each containing 20 values.

2. Compute the median and range for each sample.

3. Visualize the distributions of medians and ranges through histograms, density plots and boxplots.

4. Compare the distributions of medians and ranges with the normal distribution and inspect their shapes.

Pseudo code:

1. Set Parameters:
   - mu = 0
   - sigma = 1
   - n = 20
   - N_sim = 1000

2. Initialize vectors to store the results of the medians and ranges for each simulation.

3. Set graphical layout to a 2x2 grid

4. Creat histograms for the distribution of medians and ranges:
   - overlay the theoretical normal distribution curve for medians.

5. Creat a boxplot comparing the distributions of medians and ranges.

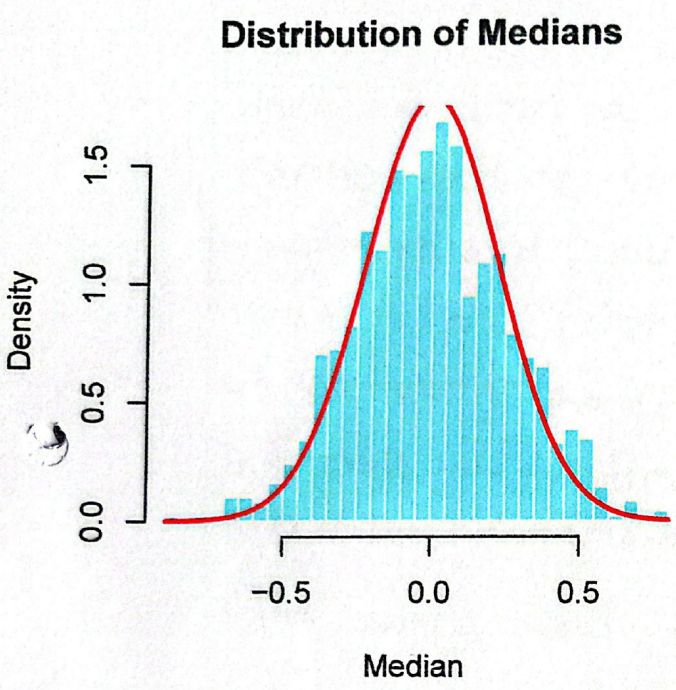
R code :

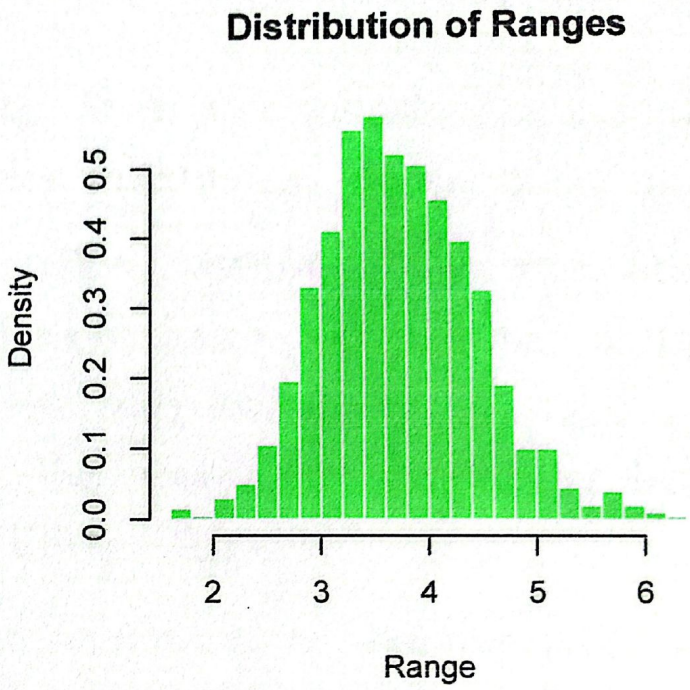
```
mu<-0
sigma <-1
n <-20
N_sim <-1000
medians <- numeric(N_sim)
ranges  <- numeric(N_sim)

set.seed(123)
for(i in 1:N_sim){
 data <- norm(n, mean=mu, sd=sigma)
medians[i] <- median(data)
ranges[i] <- max(data) - min(data)
}

par(mfrow = c(2,2))
hist(medians, breaks = 30, col = "lightblue", Probability = TRUE, main = "Distribution of Medians", xlab = "median", border = "white")
curve(dnorm(x, mean = mu, sd = sigma/sqrt(n)), add = TRUE, col = "red", lwd = 2)

hist(ranges, break = 30, col = "lightgreen", Probability = TRUE, main = "Distribution of Ranges", xlab = "Range", border = "white")

Plot(density(medians), col = "blue", lwd = 2, main = "Density Plot of Medians", xlab = "median")
```

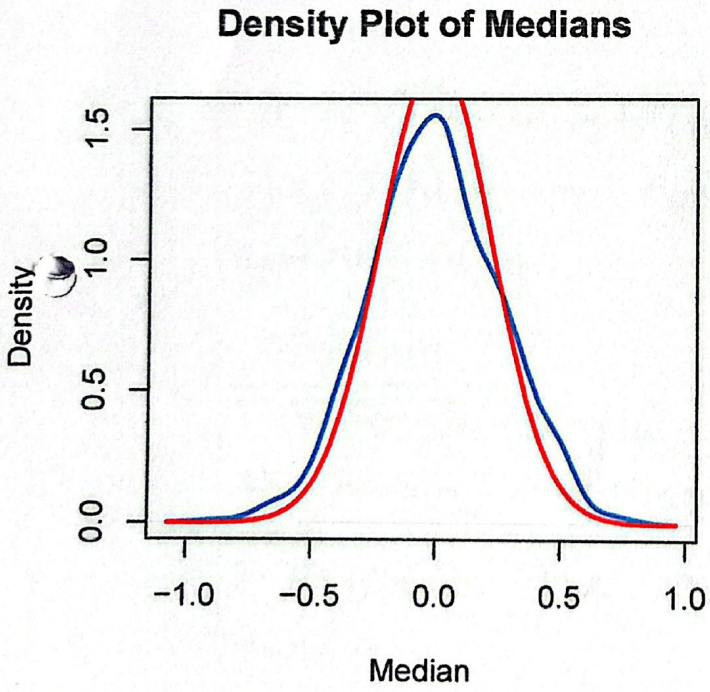
```
curve(dnorm(x, mean = mu, sd = sigma/sqrt(n)),
add = TRUE, col = "red", lwd = 2)

boxplot(list(Medians = medians, Ranges = ranges), col
= ("lightblue", "lightgreen"), main = "Boxplot of
Medians and Ranges", ylab = "Value")
```

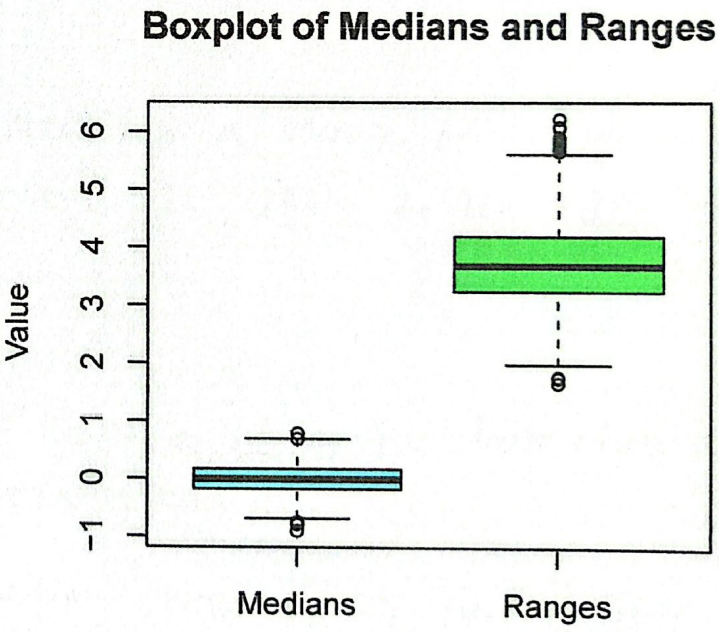
Sample Input and output:

Since this code generates random data, the exact numerical results will vary. However, the graphical outputs will have a clear pattern.


Distribution of Medians
Density
0.0
0.5
1.0
1.5
−0.5
0.0
0.5
Median


Distribution of Ranges
Density
0.0
0.1
0.2
0.3
0.4
0.5
2
3
4
5
6
Range


Density Plot of Medians
Density
0.0
0.5
1.0
1.5
−1.0
−0.5
0.0
0.5
1.0
Median


Boxplot of Medians and Ranges
Value
−1
0
1
2
3
4
5
6
Medians
Ranges

Lab 06 : Estimate Population Parameters (Mean, Variance) from Sample data

Theory:

When we draw a sample from a population, we use the sample data to estimate population parameters such as the mean and variance. The sample mean and sample variance are point estimates, but they come with some uncertainty. Therefore, we can calculate confidence intervals to estimate the range in which the true population parameters likely fall.

- The sample mean is an estimate of the population mean.
- The sample variance is an estimate of the population variance.
- A confidence interval provides a range of values within the population parameter is likely to lie, given the sample data.

Objective :

1. Generate sample data from a normal distribution using the known population parameters.
2. Estimate the population parameters from sample data.
3. Calculate the 95% confidence interval for both the sample mean and the sample variance
4. Visualize the sample data.

Pseudo code :

1. set parameters:
   - mu = 5
   - sigma = 2
   - n = 30
   - N_sim = 1000
2. Generate sample data:
   - Draw n = 30 random samples from a normal distribution with mean mu = 5 and standard deviation sigma = 2.
3. Point Estimates:
   - calculate the sample mean: sample mean = mean (sample_data)
   - calculate the sample variance: sample_var = (sample_data)
4. Confidence Intervals:
   - calculate the 95% confidence interval for the mean using the t-test function.
   - compute the 95% interval for the variance using chi-sq distribution formula.
5. Output Results:
   - Print the sample mean, sample variance, confidence Interval for the mean and confidence Interval for the variance.
6. Graphical output:

   Display histogram and Boxplot of the sample data.

R-Code :

```
mu <- 5
sigma <- 2
n <- 30
N_sim <- 1000
set.seed (123)
sample_data <- rnorm (n, mean = mu, sd = sigma)
sample_mean <- mean( sample_data)
sample_var <- mean (sample_data)
conf_int_mean <- t.test ( sample_data) $conf. int
conf_int_var <- c((n-1)* sample_var /qchisq (0.975, df = n-1),
(n-1) * sample_var / qchisq (0.025, df = n-1))
print ( Paste ( "sample Mean:", sample_mean))
print (Paste ("sample variance:", sample_var))
print ( Paste("95% CI for Mean:", conf_int_mean))
Print(Paste ("95% CI for variance:", conf_int_var ))
Par (mfrow = c(1,2))
hist (sample_data , breaks = 30, col = "lightblue", main =
sample Data with mean") xlab = "value", border =
"white")
abline ( v = sample_ data mean , col= "red", lwd = 2)
abline ( v = conf_int_mean , col= "blue", lty = 2)
boxplot(sample_data, col = "light green", main = "Boxplot of
sample Data", ylab = "Value")
```

Sample Input/out :

Input parameter:

Sample mean : 4.90579248793639

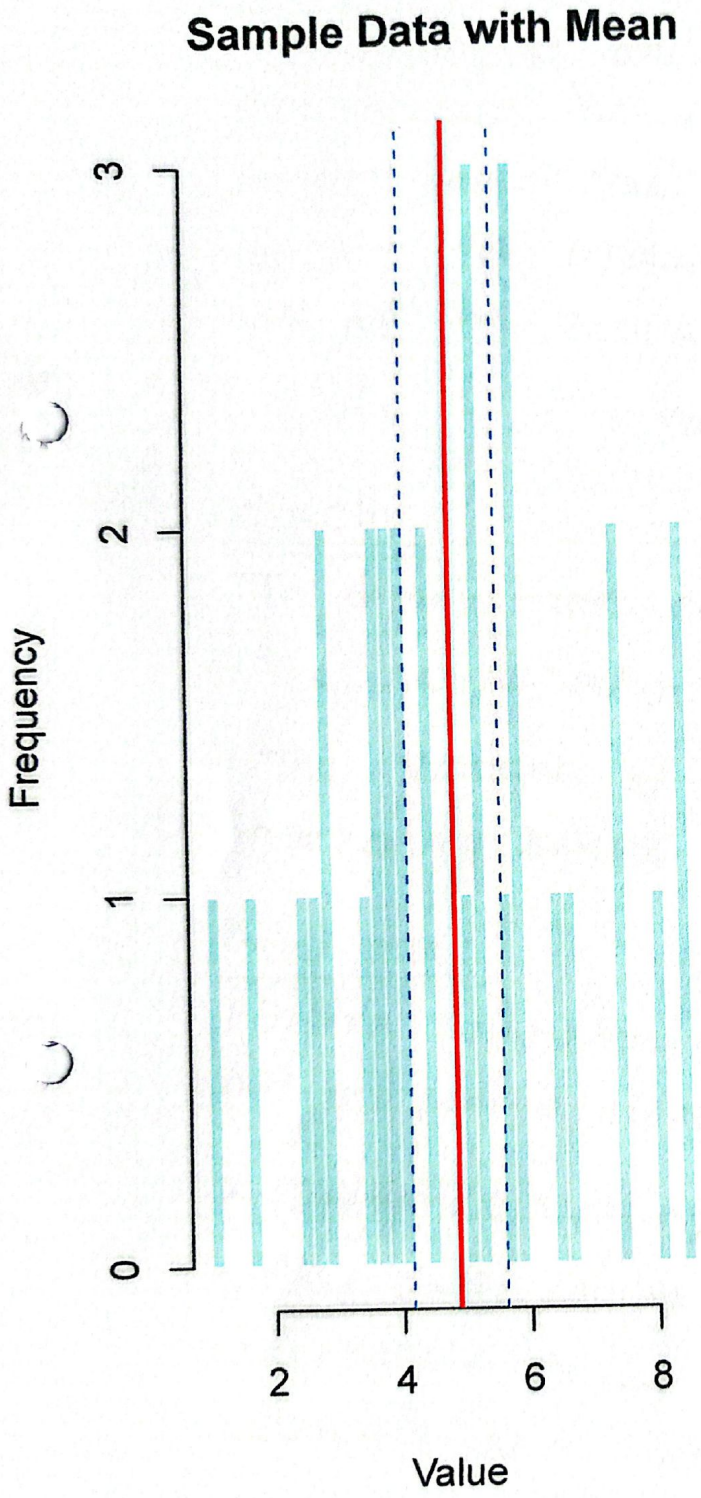
Sample variance : 3.84968493055034

95% CI for mean: (4.173146713, 5.638438595014 7)

95% CI for variance: (2.441716684329 67, 6.95 7086411 48571)

Output (Graphical Plots):

A histogram of the sample data with mean.


Sample Data with Mean
Frequency
0
1
2
3
2
4
6
8
Value

Lab 07: Demonstrate consistency by increasing sample size.

Theory:

Consistency of an estimator means that as the sample size increases, the estimator converges to the true parameter value. The sample mean $\bar{X}$ is a consistent estimator of the population mean μ, and its variance decreases as the sample size increases, following,

$$\text{var}(\bar{X}) = \frac{\sigma^2}{n}$$

Objective:

To demonstrate consistency by showing that:

- The sample mean converges to the population mean (μ)
- The variance of the sample mean decreases with increasing sample size.

Pseudo code:

1. Define parameters: Population mean (μ), standard deviation, sample sizes, and number of simulations (N_{sim})
2. Initialize vectors for sample means and variances.
3. For each sample size:
   - Generate N_{sim} sample means.
   - Compute and store the average sample mean and variance.
4. Plot the sample means and variances against sample sizes with theoretical reference lines.

R code:

```
mu <- 5
sigma <- 2
sample_sizes <- c(10, 30, 100, 500, 1000)
N_sim <- 1000
means <- numeric(length(sample_sizes))
vars <- numeric(length(sample_sizes))
set.seed(123)
for (i in 1:length(sample_sizes)){
n <- sample_sizes[i]
sample_means <- replicate(N_sim, mean(norm(n, mean
= mu, sd = sigma)))
means[i] <- mean(sample_means)
vars[i] <- var(sample_means)
}
par(mfrow = c(1,2))
plot(sample_sizes, means, type = "b", col = "blue", main =
"convergence of sample mean", xlab = "sample size",
ylab = "sample mean")
abline(h = mu, col = "red", lwd = 2)
plot(sample_sizes, vars, type = "b", col = "green", main
= "convergence of sample variance", xlab = "sample size",
ylab = "sample variance")
abline(h = sigma^2, col = "red", lwd = 2)
```

sample Input and output:

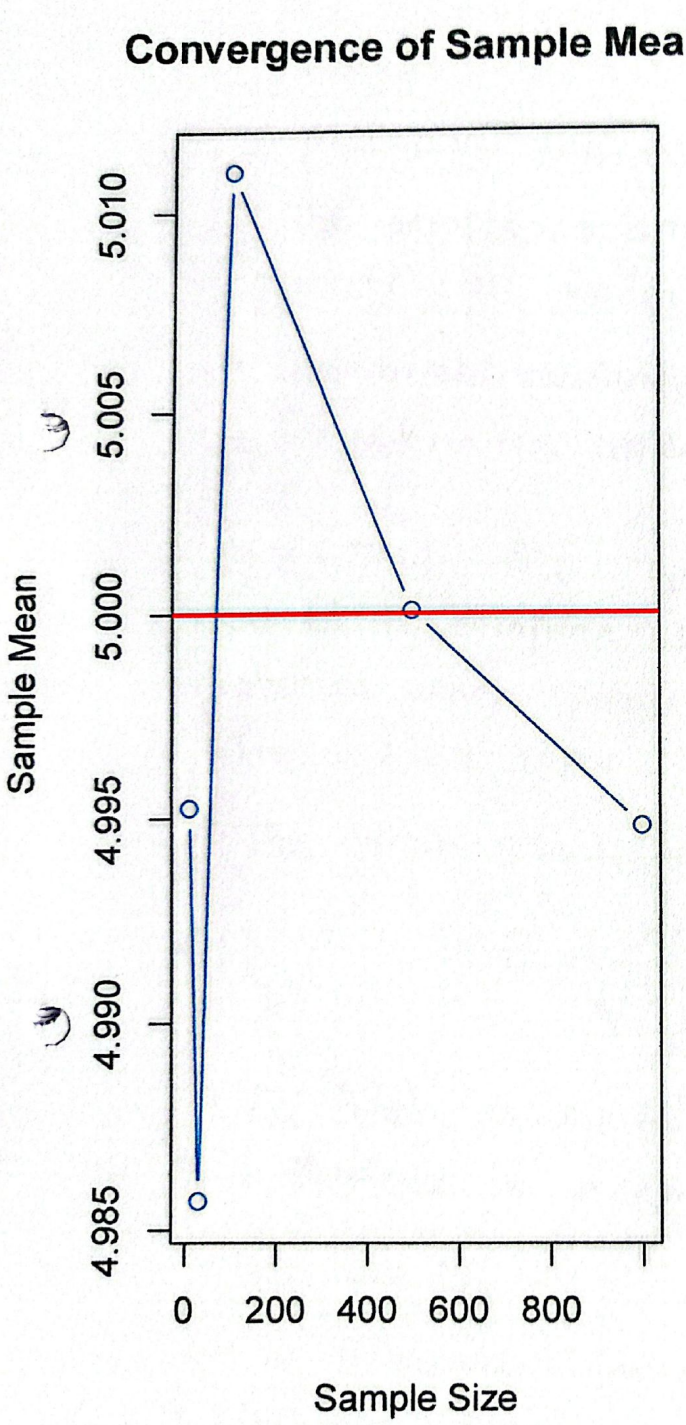
Input :

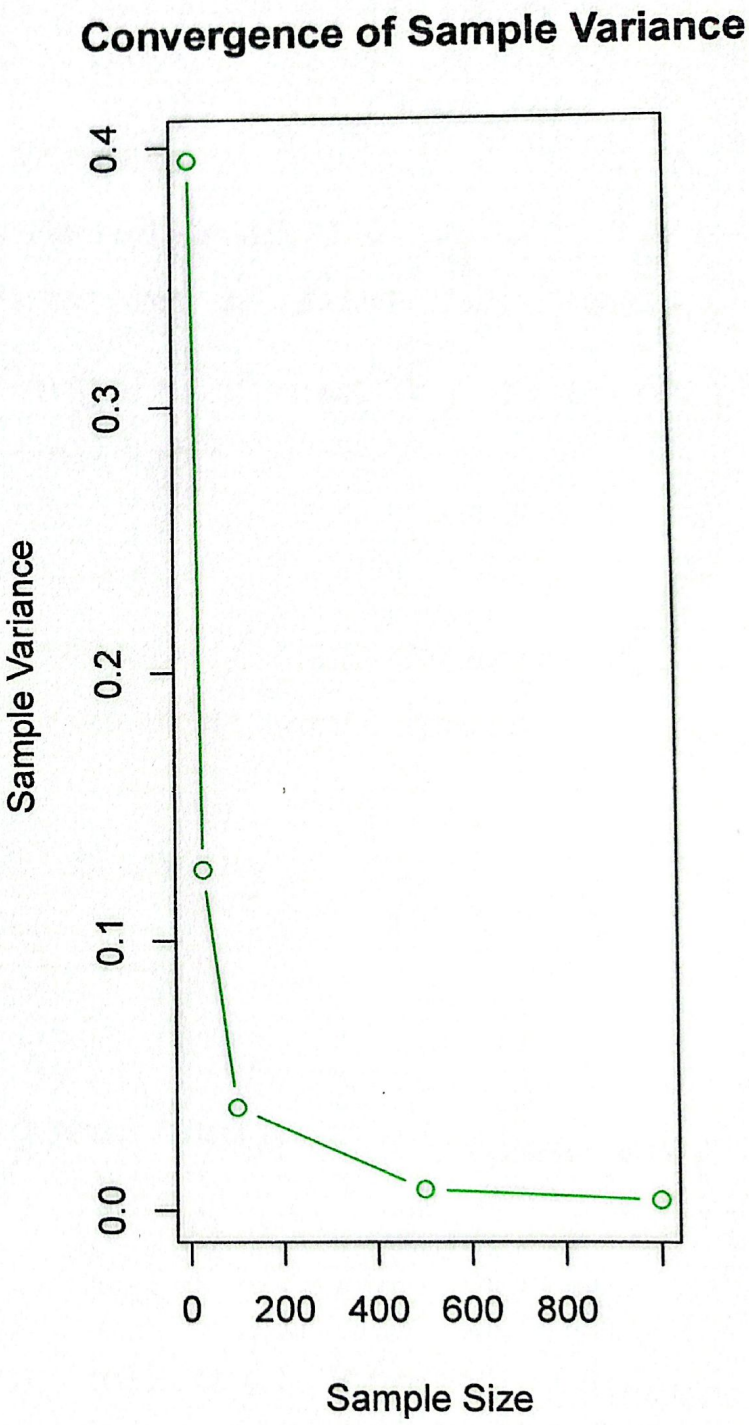
mu = 5

sigma = 2

sample_sizes = [10, 30, 100, 500, 1000]

N_sim = 1000


Convergence of Sample Mean
Sample Mean
4.985
4.990
4.995
5.000
5.005
5.010
0
200
400
600
800
Sample Size


Convergence of Sample Variance
Sample Variance
0.0
0.1
0.2
0.3
0.4
0
200
400
600
800
Sample Size

Lab 08: Comparison of Biased and Unbiased Variance and Two sample z-Test.

Theory:

1. Biased vs. Unbiased Variance:

- The sample variance formula divides by $n-1$ to correct for underestimation of population variance.
- The biased variance formula divides by n, leading to systematic underestimation.

2. z-Test for Two Means:

- Used to compare the means of two independent samples when population variances are known or large sample sizes apply.
- The test statistic is given by:

$$Z = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$$

- A high absolute z score suggests a significant difference in means.

objective:

1. To compare biased vs. unbiased sample variance using simulation.
2. To Perform a two-sample z-Test to determine if there is a significant difference between two independent groups.

Pseudocode:

1. Variance comparison;
   - set Parameters : μ, σ, n, Nsim \mu, \sigma, n, N_{sim} μ, σ, m, Nsim.
   - Initialize vectors for sample variance and biased variance.
   - for each simulation;
     - Generate a normal sample
     - Unbiased and variance computation
   - plot histogram of both variances.
2. Two-sample Z-Test;
   - Generate two normal samples with different means.
   - Perform a z-test using z test ().
   - Plot histogram of both groups.

Rcode:

```
mu <- 5
sigma <- 2
n <- 30
N_sim <- 1000
sample_vars <- numeric(N_sim)
biased_vars <- numeric(N_sim)

set.seed(123)
for(i in 1:N_sim){
data <- norm(n, mean = mu, sd = sigma)
```

```
sample_vars [i] <- var (data)
biased_vars [ i] <- sum ((data - mean (data))^2) /n
}
par (mfrow = c(1,2))
hist ( sample_vars , breaks = 30, col="light blue", main
= " Unbiased sample Variance ", Xlab = " Variance ", border
  ="white")
 abline (v = sigma^2, col = "red" lwd=2)
hist ( biased_vars , breaks = 30, col = " lightgreen",
main = " Biased sample variance ", Xlab = " Variance ",
border = "White")
abline ( v = sigma ^ 2, col = " red ", lwd = 2)
set.seed (123)
group 1 <- norm (30, mean = 50, sd = 10)
group 2 <- norm (30, mean = 55, Sd = 10)
Z_test_result <- Z.test (group 1, group 2, sigma.x =
Sd(group 1), sigma.y = sd (group 2))
Print ( Z_test_result)
Par (mfrow = c(1,2))
hist ( group1, breaks = 30, col = " light blue", main
= " Histogram of Group 1 ", Xlab = "value", border =
 "white")
```

hist(group2, breaks = 30, col = "lightgreen", main = "Histogram of Group 2", xlab = "Value", border = "white")

Sample Input and output:

Input:

mu = 5

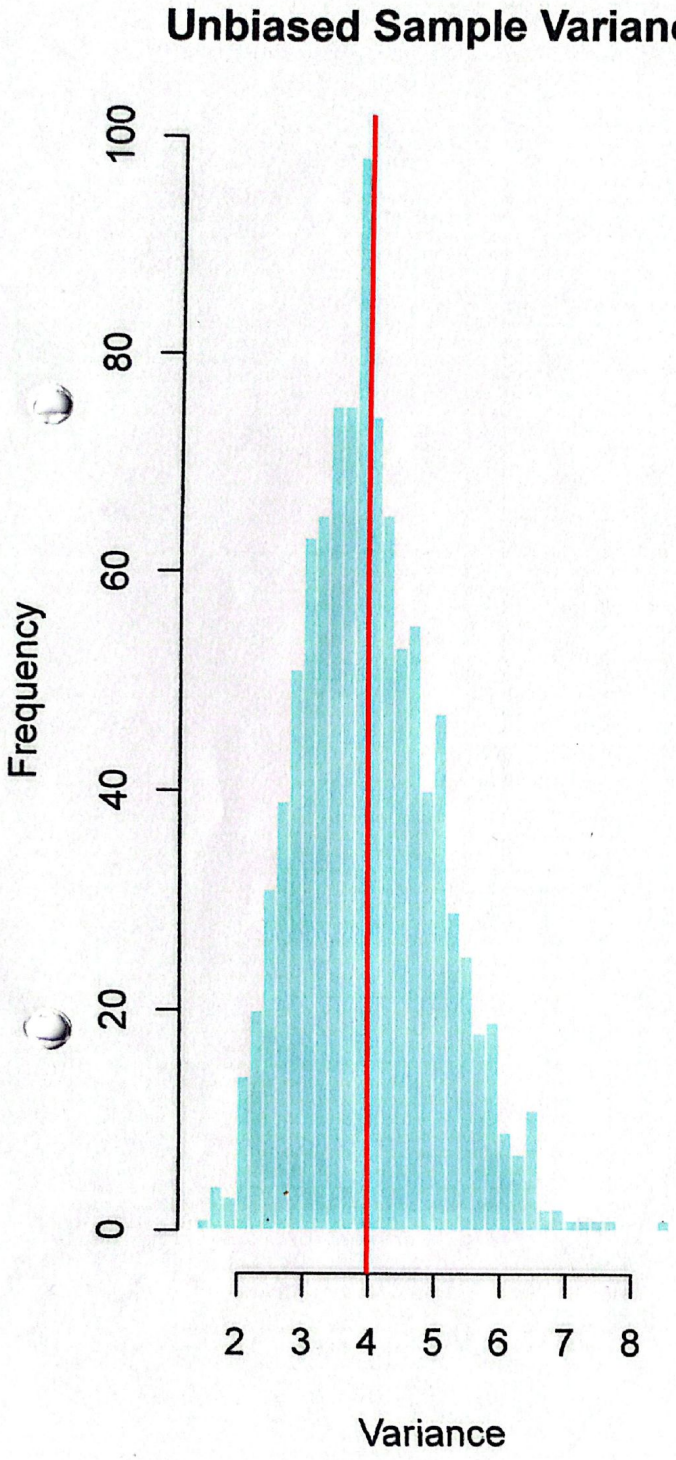
sigma = 2

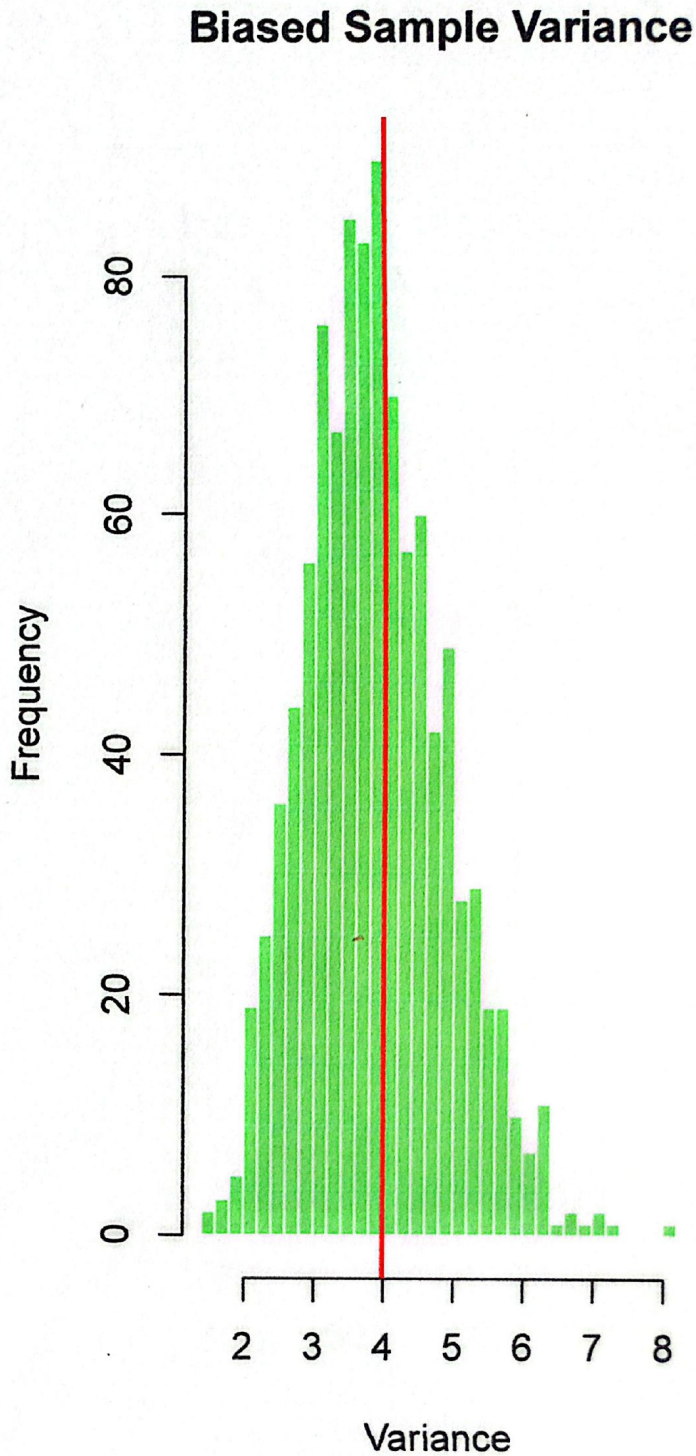
n = 30

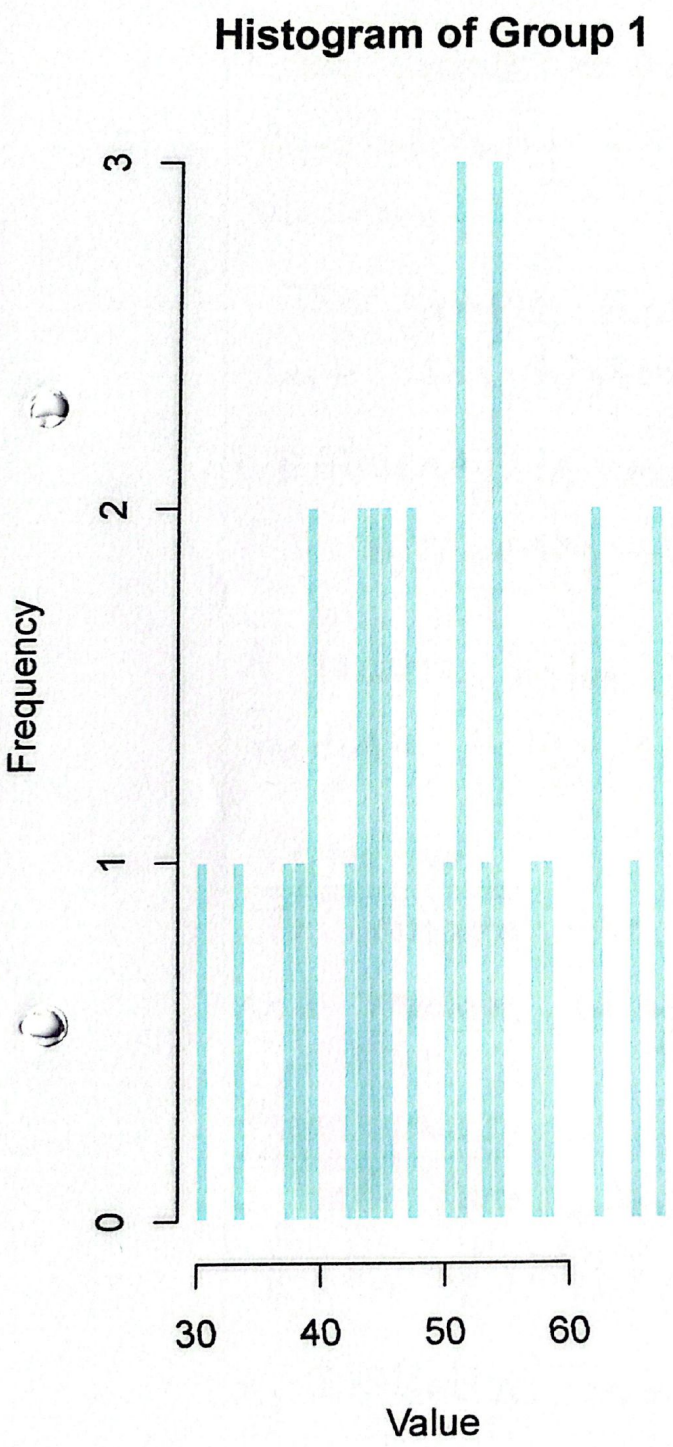
N_sim = 1000

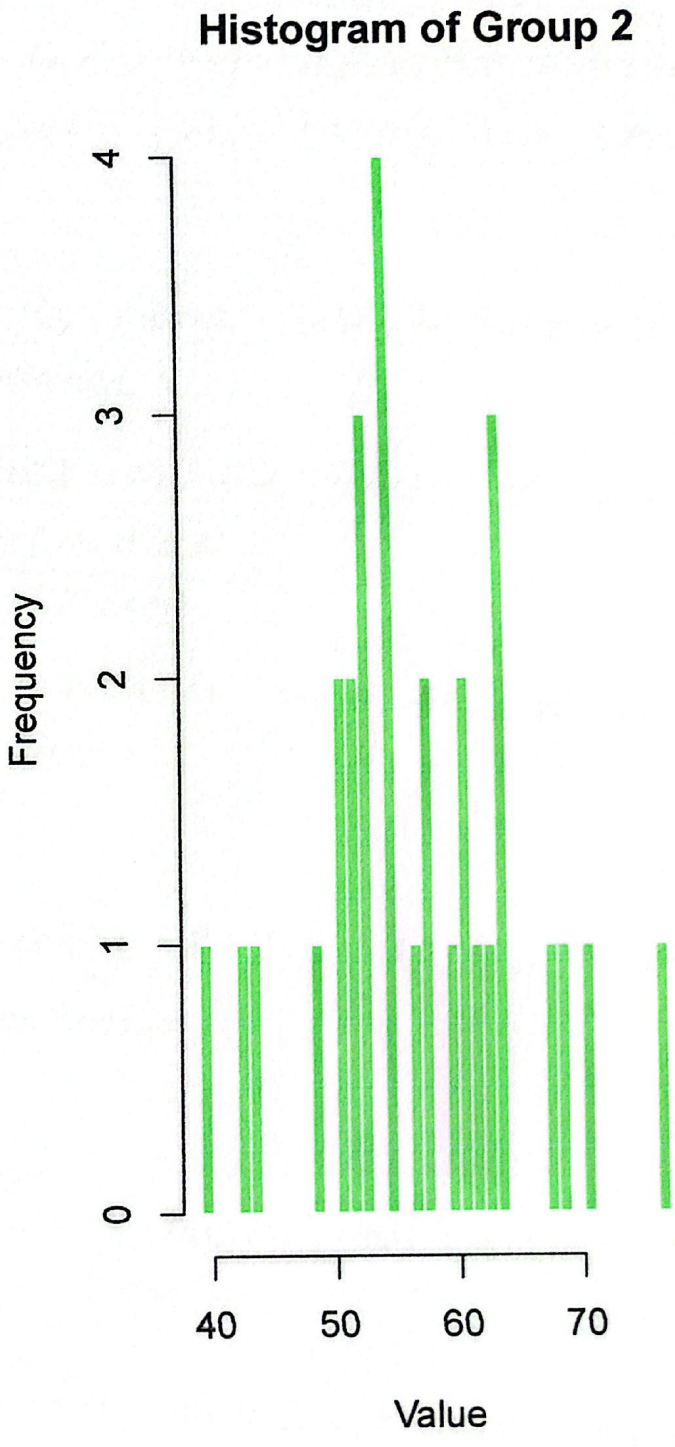
group1 <- norm(30, mean = 50, sd = 10)

group2 <- norm(30, mean = 55, sd = 10)


Unbiased Sample Variance
Frequency
0
20
40
60
80
100
2
3
4
5
6
7
8
Variance


Biased Sample Variance
Frequency
0
20
40
60
80
2
3
4
5
6
7
8
Variance


Histogram of Group 1
Frequency
0
1
2
3
30
40
50
60
Value


Histogram of Group 2
Frequency
0
1
2
3
4
40
50
60
70
Value

Lab 09 : Efficiency of Mean vs. Median

Theory :

- The sample mean is the most efficient estimator for normally distributed data, having the smallest variance.
- The sample median is more robust to outliers but has higher variance.
- Efficiency is measured as the ratio of variances.

$$\text{Efficiency} = \frac{\text{Var (median)}}{\text{Var (Mean)}}$$

A lower value (< 1) indicates that the mean and median using simulation.

Objective :

To compare the efficiency of the sample mean and median using simulation.

Pseudo code :

1. Set Parameters μ, σ, n, N_{sim} \mu, \sigma, n, N_{\text{sim}} μ, σ, n, N_{sim}
2. Initialize vectors for sample means and medians
3. for each simulations :
   - Generate normal data
   - Compute sample mean and median
4. Compute efficiency as the ratio of variances.
5. Plot histogram of means and medians.

R Code:

```
mu <- 5
sigma <- 2
n <- 30
N_sim <- 1000
means <- numeric(N_sim)
medians <- numeric(N_sim)
set.seed(123)
for (i in 1:N_sim) {
data <- norm(n, mean = mu, sd = sigma)
means[i] <- mean(data)
medians[i] <- median(data)
efficiency <- var(medians) / var(means)
print(paste("Efficiency (median/mean):", efficiency))
par(mfrow = c(1,2))
hist(means, breaks = 30, col = "lightblue", main =
"Distribution of sample means", xlab = "value", border
="white")
hist(medians, breaks = 30, col = "light green", main=
"Distribution of sample medians", xlab = "value",
border = "white")
```

sample Input and Output:

Input:

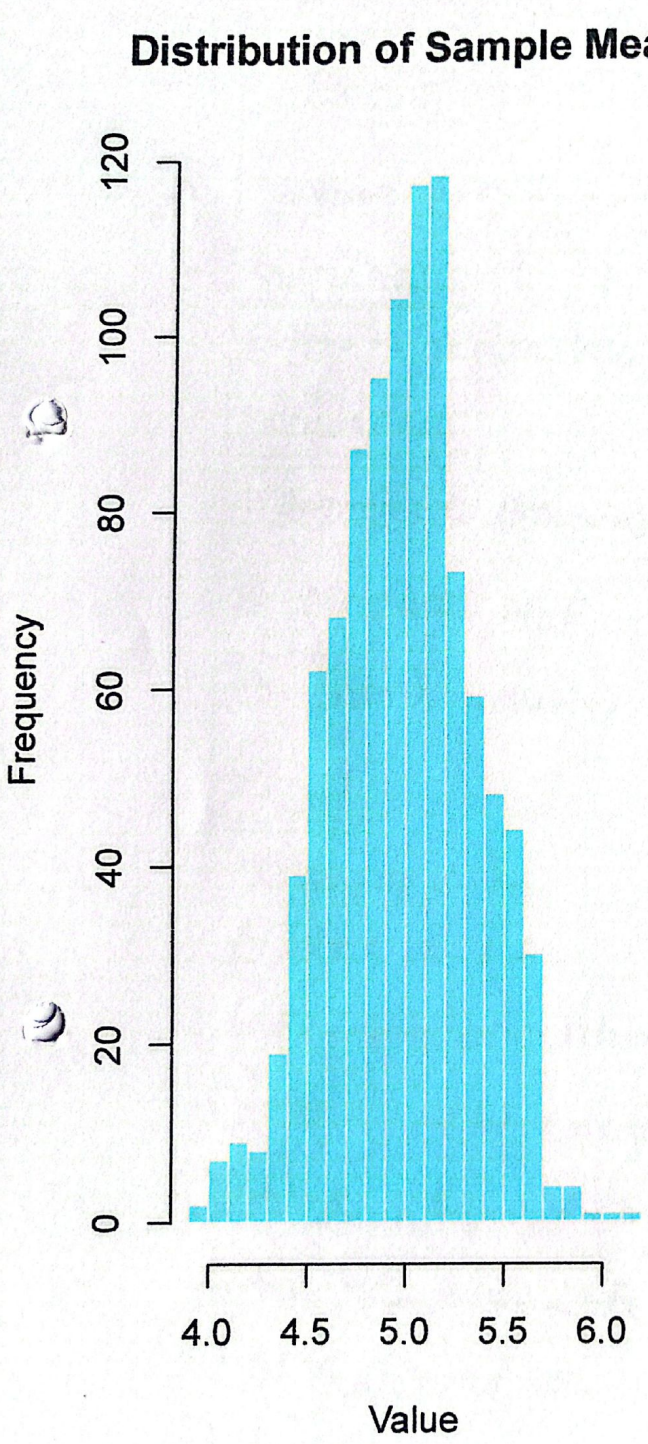
mu = 5

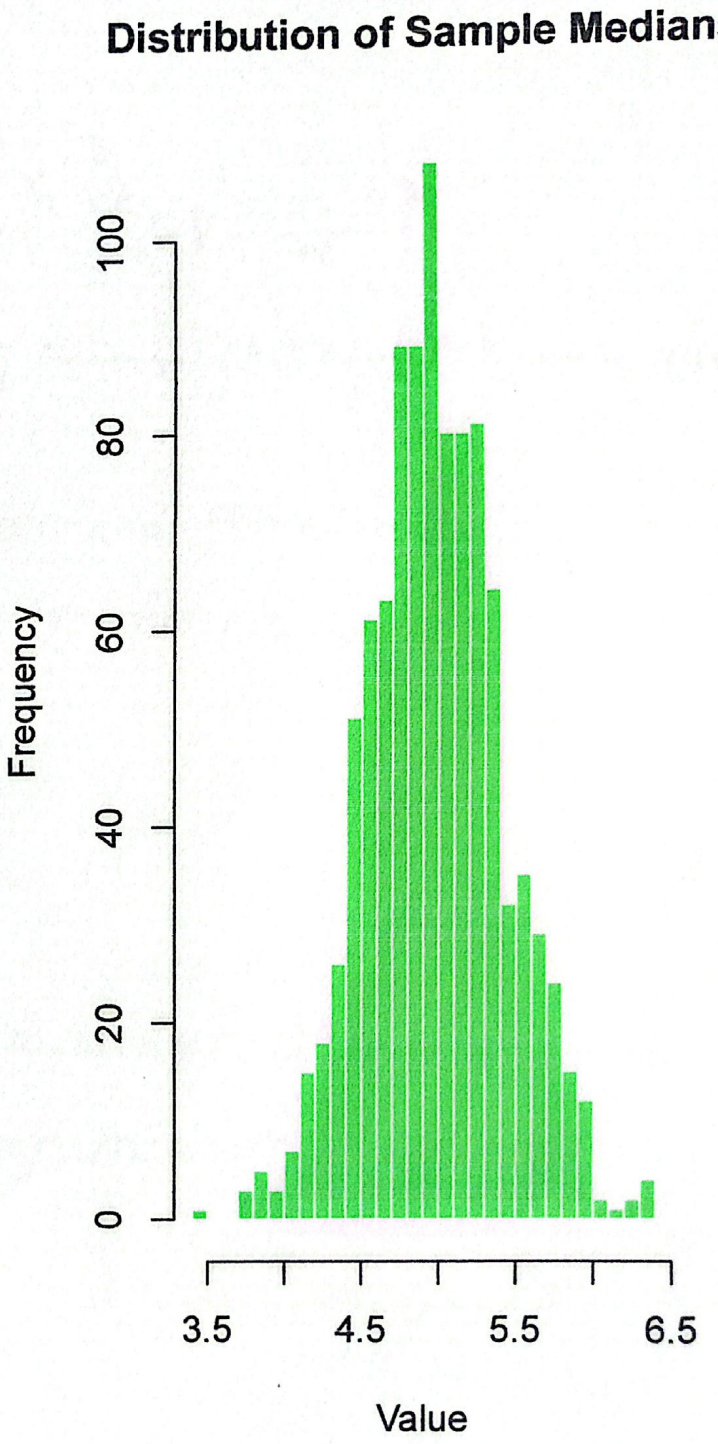
sigma = 2

n = 30

N_sim = 1000

output: Efficiency (median/mean): 1.508576093687


Distribution of Sample Means
Frequency
0
20
40
60
80
100
120
4.0
4.5
5.0
5.5
6.0
Value


Distribution of Sample Medians
Frequency
0
20
40
60
80
100
3.5
4.5
5.5
6.5
Value

Lab 10: MLE for Binomial, Possion and Normal Distribution

Theory:

- Binomial MLE : $\hat{p} = \frac{\Sigma x}{n}$
- Poisson MLE : $\hat{\lambda} = \frac{\Sigma x}{N}$
- Normal MLE : $\hat{\mu} = \frac{\Sigma x}{N}$, $\hat{\sigma} = \sqrt{\frac{\Sigma (x - \hat{\mu})^2}{N}}$

Objective:

Estimate parameters using Maximum Likelihood Estimation (MLE)

Pseudo code:

1. Generate Binomial, Poisson and Normal data
2. Compute MLE estimates for p, λ, μ, σ
3. Print results.

R Code:

```
n_binom <- 20
p_true <- 0.6
data_binom <- rbinom(100, size = n_binom, prob = p_true)
p_mle <- mean(data_binom)/n_binom
lambda_true <- 3
~~data~~.mu_true <- 5
sigma_true <- 2
data_norm <- rnorm(100, mean = mu_true, sd = sigma_true)
mu_mle <- mean(data_norm)
sigma_mle <- sqrt(mean((data_norm - mu_mle)^2))
Print(Paste("Binomial MLE for p:", p_mle))
```

```
Print(Paste("Poisson MLE for lambda:", lambda_male))
Print(Paste("Normal MLE for mu:" mu_mle))
Print(Paste("Normal MLE for sigma:", sigma_mle))
```

Sample output:

```
Binomial MLE for P: 0.603
Poisson MLE for lambda: 2.89
Normal MLE for mu: 5.354205813930 28
Normal MLE for sigma: 1.857459785 2958
```

Lab 11: Simulate Decision-Making Processes using Hypothesis Testing

Theory:

The lab demonstrates the use of the hypothesis testing to simulate decision-making processes. It tests the hypothesis about the population mean by comparing the sample mean against hypothesis value.

objective:

- To simulate a decision making process using hypothesis testing.
- To perform a one-sample t-test and interpret the result.
- To visualize the data using a histogram and density plot.

Pseudocode:

1. Define Parameters:
   - Null hypothesis mean (mu0), true population mean (mu1), standard deviation (sigma), sample size (n), significant level (alpha).
2. Generate sample data based on the true population mean.
3. Perform a t-test to compare the sample mean with the null hypothesis mean (mu0)
4. Make a decision based on the P-value:
   - If the Pvalue < alpha, reject H0 (null hypothesis)
   - otherwise, fail to reject H0

5. Output the test statistic, P value and decision

6. Plot the histogram and density plot to visualize the critical region.

R code:

```
mu0<-5
mu1<-6
sigma<-2
n<-30
alpha<-0.05
set.seed(123)
sample_data<-rnorm(n,mean=mu1,sd=sigma)
t_test_result<-t.test(sample_data,mu=mu0,alternative
="greater")
if(t_test_result$p.value<alpha){
decision<-"Reject H0"
}
else{
decision<-"fails to reject H0".
}
Print(Paste("Test statistic:",t_test_result$statistic))
Print(Paste("P-value:",t_test_result$p.value))
Print(Paste("Decision:",decision))
```

```
par(mfrow = c(1,2))
hist(sample_data, breaks = 30, col = "light blue", main
= "Sample data", xlab = "value", border = "white")
abline(v = mu0, col = "red", lwd = 2)
abline(v = mean(sample_data), col = "blue", lwd = 2)
plot(density(sample_data), col = "blue", lwd = 2, main
= "Density Plot", xlab = "value")
abline(v = qt(1 - alpha, df = n - 1), col = "red", lty = 2)
```

Sample Input/output:

Input:

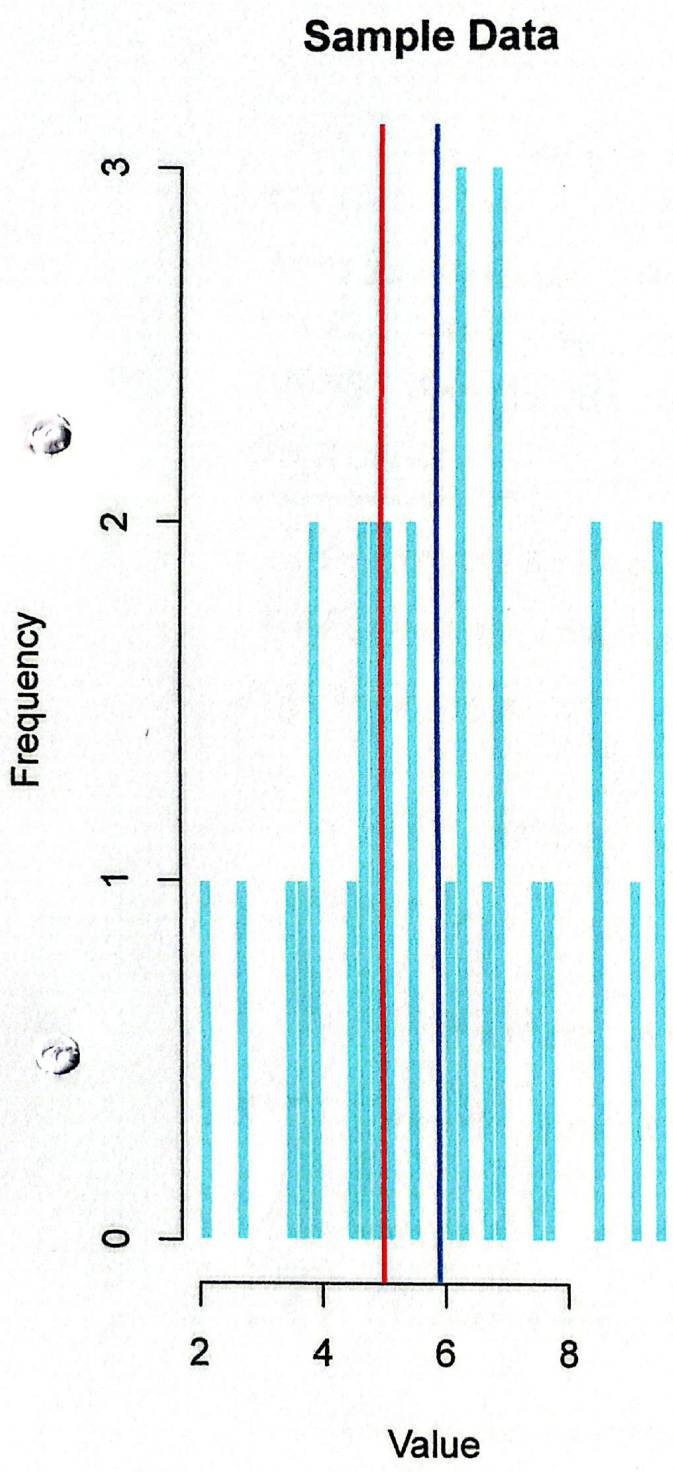
```
mu0 <- 5
mu1 <- 6
sigma <- 2
n <- 30
alpha <- 0.05
```

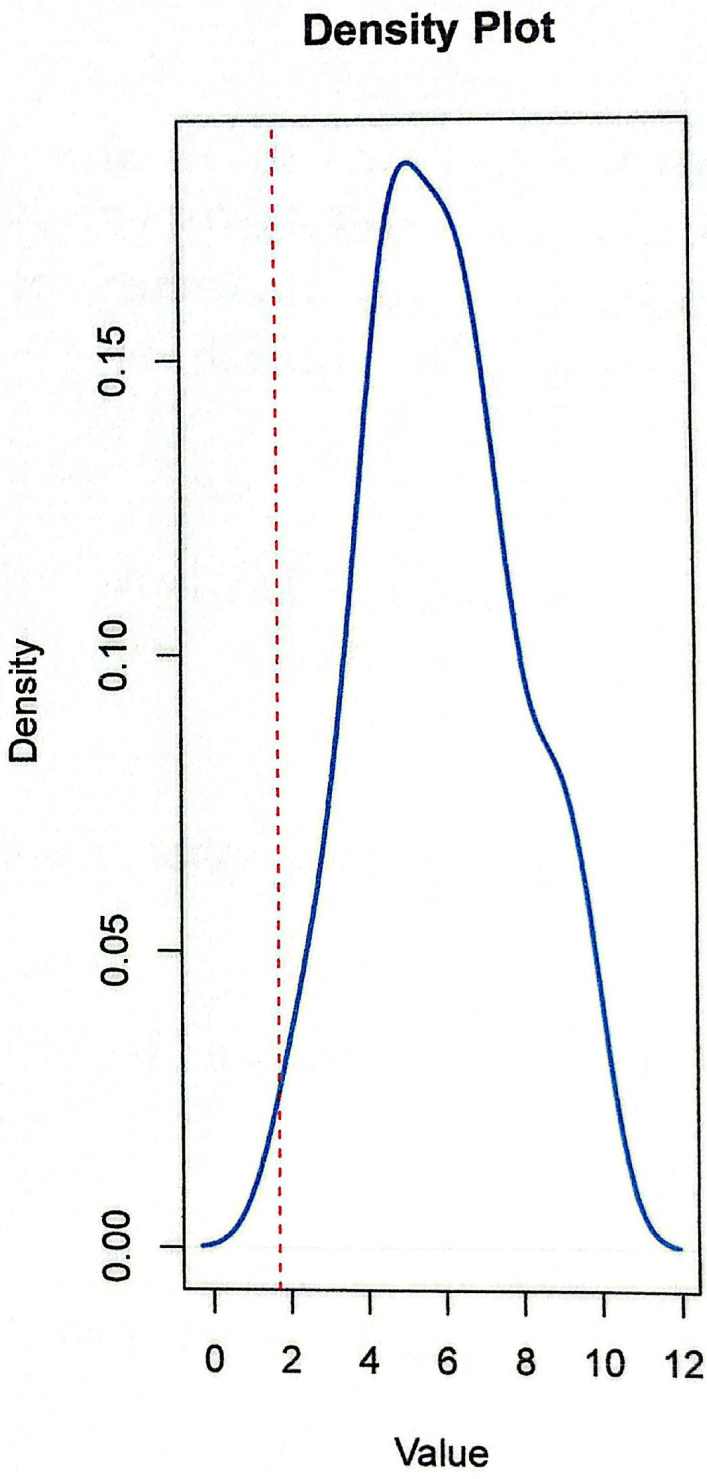
Sample output:

Test statistic: 2.53858027419059

P-value: 0.0085756602152024 2

Decision: Reject H0


Sample Data
Frequency
0
1
2
3
2
4
6
8
Value


Density Plot
Density
0.00
0.05
0.10
0.15
0
2
4
6
8
10
12
Value

Lab 12: Derive the Best Critical Region for simple vs. Composite Hypotheses.

Theory:

This lab demonstrates how to derive the best critical region for a hypothesis test, specifically for testing simple versus composite hypothesis using the likelihood ratio test. It explores the decision making process using sample data.

Objective:

- To derive the best critical region using the likelihood ratio test for simple vs. composite hypothesis.
- To Perform hypothesis testing using the critical region.
- To visualize the critical region in density plots for both H0 and H1

Pseudocode:

1) set Parameters: mu0, mu1, sigma, n, alpha

2. Generate Data:
   - sample_data_H0 from mu0
   - sample_data_H1 from mu1

3. Likelihood Ratio Test:

- calculate likelihood ratio for data.

4. critical value: critical_value = qnorm(1-alpha, mu0, sigma/sqrt(n))

5. Decision:

- decision_H0 = mean(sample_data_H0) > critical_value
- decision_H1 = mean(sample_data_H1) > critical_value

6. output: print critical value, decisions for H0 and H1.

7. Plot: plot density for sample_data H0 and sample_data H1 with critical value.

R code:

```
mu0 <- 5
mu1 <- 6
sigma <- 2
n <- 30
alpha <- 0.05
set.seed(123)
sample_data_H0 <- rnorm(n, mean = mu0, sd = sigma)
sample_data_H1 <- rnorm(n, mean = mu1, sd = sigma).
```

```
likelihood_ratio <- function(data, mu0, mu1, sigma){
exp(sum(dnorm(data, mean = mu1, sd = sigma, log = TRUE)
- sum(dnorm(data, mean = mu0, sd = sigma, log = TRUE)))
}

critical_value <- qnorm(1-alpha, mean = mu0, sd =
sigma / sqrt(n))

decision_H0 <- mean(sample_data_H0) > critical_value

decision_H1 <- mean(sample_data_H1) > critical_value.

Print(Paste("critical value:", critical_value))

Print(Paste("Decision Under H0:", decision_H0))

Print(Paste("Decision Under H1:", decision_H1))

Par(mfrow = c(1,2))

Plot(density(sample_data_H0), col = "blue", lwd = 2,
main = "Density Under H0", xlab = "Value")

abline(v = critical_value, col = "red", lty = 2)

Plot(density(sample_data_H1), col = "green", lwd = 2,
main = "Density Under H1", xlab = "Value").

abline(v = critical_value, col = "red", lty = 2)
```

Sample Input/output:

Input:

mu0 <-5

mu1 <-5

sigma <-2

n <- 30

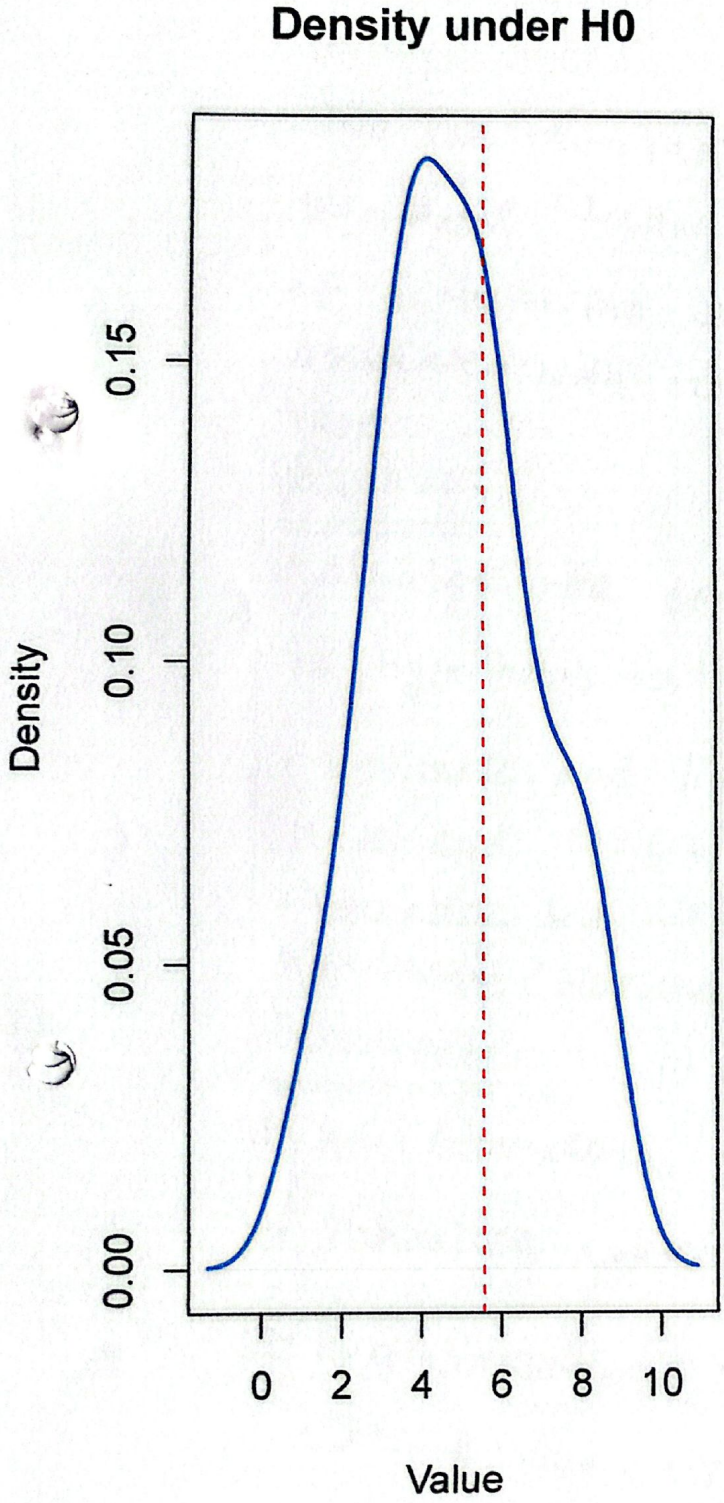
alpha <- 0.05

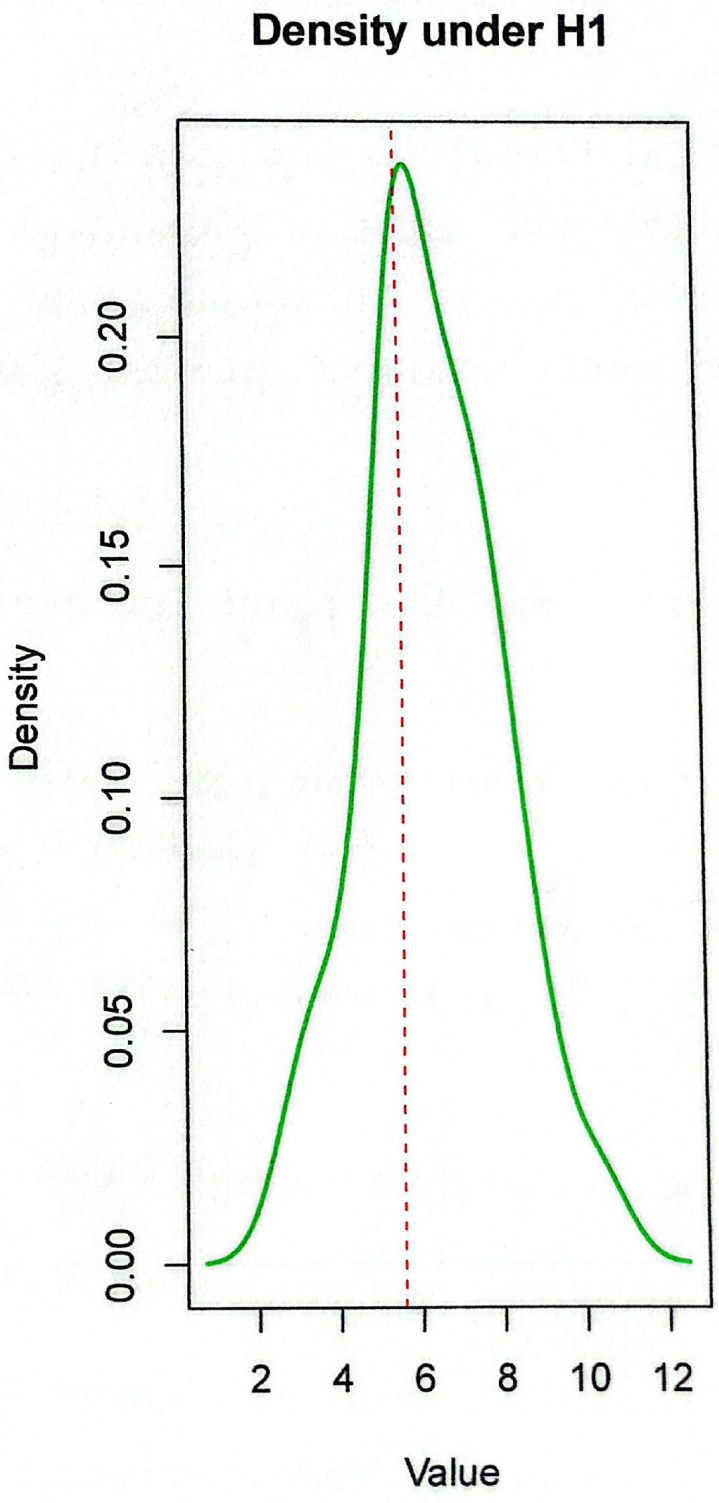
output:

Critical value : 5.643234

Decision Under H0: FALSE

Decision under H1 : TRUE


Density under H0
Density
0.00
0.05
0.10
0.15
0
2
4
6
8
10
Value


Density under H1
Density
0.00
0.05
0.10
0.15
0.20
2
4
6
8
10
12
Value

Lab 13: Simulate Type I and Type II Errors in Hypothesis Testing

Theory:

This lab simulates Type I and Type II errors in hypothesis testing by performing multiple simulations of t-test. The goal is to estimate the error rates associated with rejecting the null hypothesis when it is TRUE.

Objective:

- Simulate the occurance of Type I and Type II errors in hypothesis testing.
- Estimate and display the Type I error rate. Type II error rate and Power of a hypothesis test.
- Visualize the distribution of P values under the null hypothesis (H0) and alternative hypothesis (H1)

Pseudocode:

1. Set Parameters: mu0, mu1, sigma, n, alpha, Nsim
2. Initialize counters: type_I errors = 0, type_II_errors = 0
3. Run Simulations (N-sim times):
   - If P-value < alpha, increment type_I_errors
   - If P value >= alpha, increment Type II error rate and Power.
4. Output: calculate and Print Type I error rate, Type II error rate and Power

5. Plot :

'Plot histograms for P-values under H0 (Type I error) and H1 (Type II error) with alpha threshold.

R Code :

```
mu0 <- 5
mu1 <- 6
sigma <- 2
n <- 30
alpha <- 0.05
N_sim <- 1000
type_I_errors <- 0
type_II_errors <- 0
set.seed(123)
for (i in 1:N_sim){
data_H0 <- rnorm(n, mean = mu0, sd = sigma)
t_test_H0 <- t.test(data_H0, mu=mu0, alternative = "greater")
if(t_test_H0$p.value < alpha){
type_I_errors <- type_I_errors + 1
}
data_H1 <- rnorm(n, mean=mu1, sd= sigma)
t_test_H1 <- t.test(data_H1, mu=mu0, alternative = "greater")
if (t_test_H1$p.value) = alpha){
type_II_errors) <- type_II_errors + 1
}
}
```

```
Print(Paste("Type I Error Rate:", type_I_errors/N_sim))
Print(Paste("Type II Error Rate:", type_II_errors/N_sim))
Print(Paste("Power:", 1-(type_II_errors/N_sim)))
Par(mfrow=c(1,2))
hist(replicate(N_sim, t.test(rnorm(n, mean=mu0, sd=sigma),
mu=mu0, alternative="greater")$p.value),
breaks=30, col="light blue", main="P-values under
H0", xlab="P-Value", border="white")
abline(v=alpha, col="red", lwd=2)
hist(replicate(N_sim, t.test(rnorm(n, mean=mu1, sd=
sigma), mu=mu0, alternative="greater")
$p.value),
breaks=30, col="lightgreen", main="P-values under H1",
Xlab="P-Value", border="White")
abline(v=alpha, col="red", lwd=2)
```

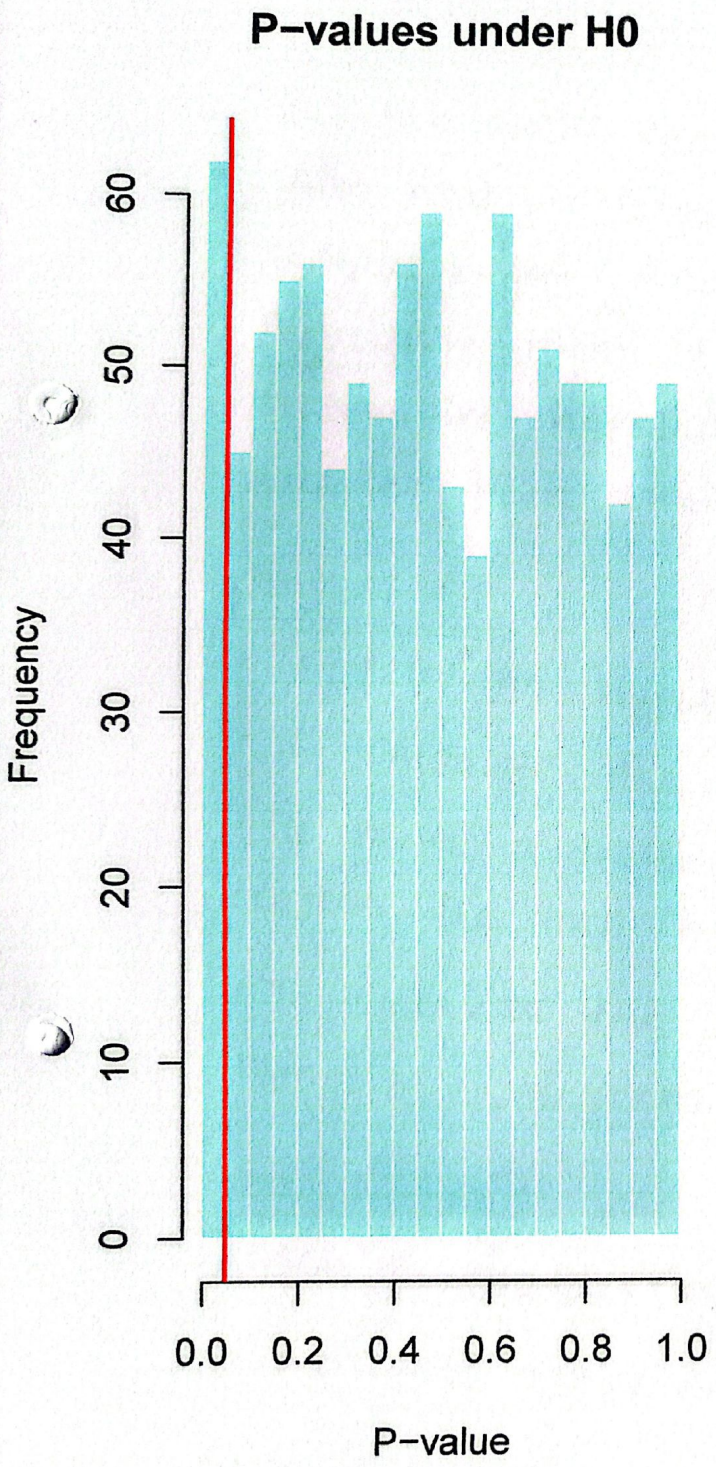
Sample Input/Output:

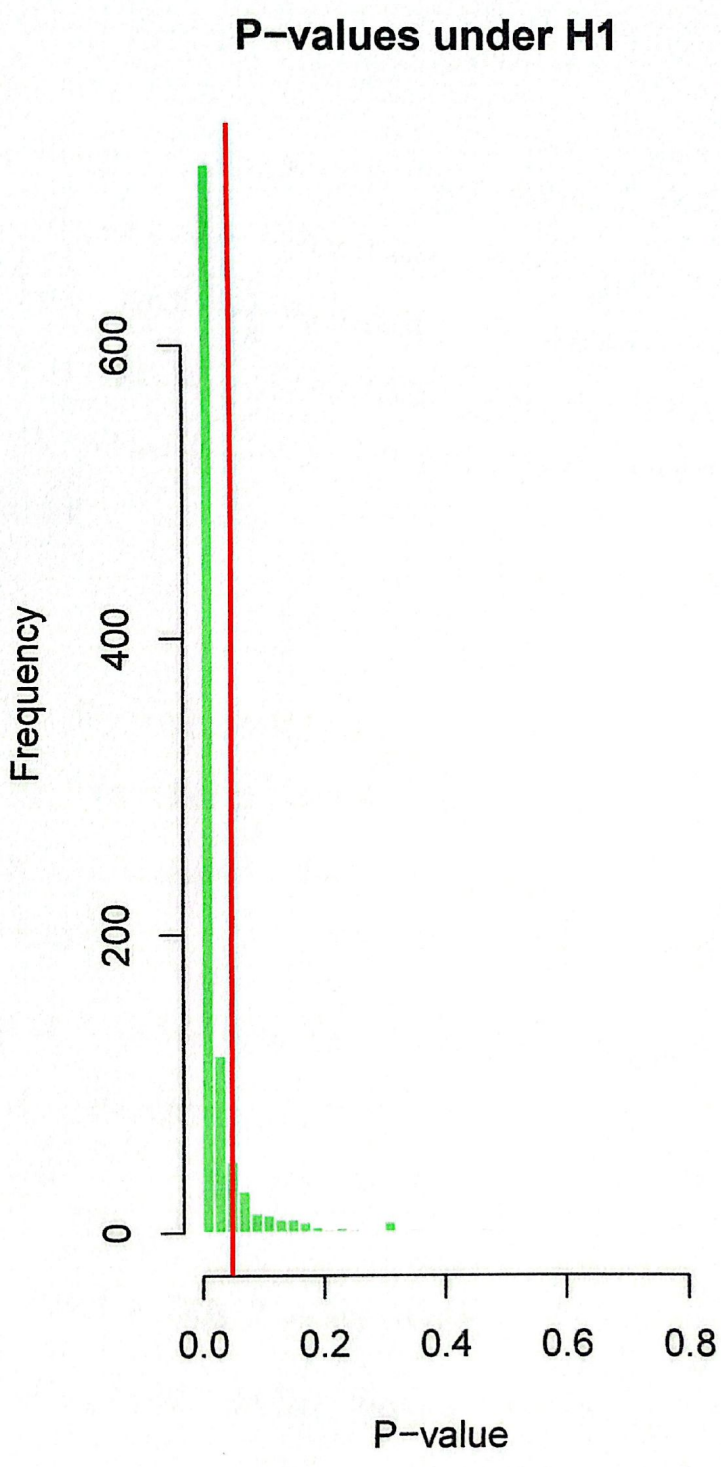
Input:

```
mu0<-5
mu1<-6
sigma<-2
n<-30
alpha<-0.05
Nsim<-1000
```

Output:

```
Type I Error Rate: 0.039
Type II Error Rate: 0.159
Power: 0.841
```


P−values under H0
Frequency
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0
P−value


P−values under H1
Frequency
0
200
400
600
0.0
0.2
0.4
0.6
0.8
P−value

Lab 14: Perform Hypothesis Testing step-By-step using Real or simulated Data.

Theory:

This lab walks through the process of hypothesis testing using real or simulated data. It involves testing a one-sample t-test, where the null hypothesis is that the population mean is equal to a specific value, and the alternative hypothesis is that the population mean is greater than that value.

Objective:

- To perform hypothesis testing step-by-step
- To calculate the test statistic and p-value
- To make a decision on whether to reject or fail to reject the null hypothesis
- To visualize the sample data with critical regions using a histogram and density plot.

Pseudocode:

1. Set Parameters: mu0, mu1, sigma, n, alpha
2. Generate sample data: simulate data based on mu1
3. State Hypothesis:
   - H0 : mu = 5
   - H1 : mu > 5
4. Choose significant level: set alpha = 0.05

5. Calculate Test statistic:

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$$

6. Determine critical value and P-value:
   - critical value: t_α, df
   - P-value: calculate using the Pt() function

7. Make Decision:
   - If t_stat > critical_value, reject H0.
   - otherwise, fail to reject H0.

8. Output: Display test statistic, P-value, critical value and decision.

9. Plot:
   - Histogram and density plot with critical region.

R code:

```
mu0 <- 5
mu1 <- 6
sigma <- 2
n <- 30
alpha <- 0.05
set.seed(123)
sample_data <- rnorm(n, mean = mu_1, sd = sigma)
Print("H0: mu = 5")
Print("H1: mu > 5")
print(Paste("Significance level (alpha):", alpha))
```

```
t_stat <-(mean(sample_data) - mu0)/(sd(sample_data)/
sqrt(n))
Print(Paste("Test Statistic (t):", t_stat))
critical_value <- qt(1-alpha, df = n-1)
P-value <- pt(t_stat, df=n-1, lower.tail = FALSE)
Print(Paste("critical value:", critical_value))
Print(Paste("P-value:", P_value))
if(t_stat > critical_value){
decision <- "Reject H0"}
else{
decision <- "fail to reject H0"
}
Print(Paste("Decision:", decision))
Par(mfrow = c(1,2))
hist(sample_data, breaks = 30, col = "light blue", main
="Sample Data", Xlab = "Value", border = "white")
abline(v=mu0, col = "red", lwd = 2)
abline(v=mean(sample_data), col= "blue", lwd = 2)
Plot(density(sample_data), col = "blue", lwd = 2, main =
"Density Plot", Xlab = "value")
abline(v = critical_value, col= "red", lty = 2)
```

Sample Input/Output:

Input:

mu0 <- 5

mu1 <- 6

sigma <- 2

n <- 30

alpha <- 0.05

Output:

Ho : mu=5

H1 : mu>5

significance level (alpha) : 0.05

Test statistic (t) = 2.528580274190 59

critical value : 1.699127026533 5

P - value : 0.008575660215 20242

Decision : Reject Ho

Lab 15: Compare the power of Different Tests for the same Hypothesis.

Theory:

This lab compares the power of two hypothesis tests: the t-test and the z-test for the same hypothesis. The objective is to simulate data under the alternative hypothesis (H1) and calculate the power of each test across multiple simulations. The power of a test is the probability that the test correctly rejects the null hypothesis when it is false.

Objective:

- To compare the power of the t-test and z-test for testing the same hypothesis.
- To investigate how the sample size affects the power of the t-test.
- To visualize the power comparison and the effect of sample size on the power of t-test.

Pseudocode:

1. Set Parameters: mu0, mu1, sigma, n, alpha
2. Initialize power counters: Power_t_test = 0, Power_z_test = 0

3. Run simulations (N_sim times):

'simulate data under H1

' Perform a t-test and check if P-value < alpha. If true, increment power_t_test.

' Perform a z-test and check if test statistic > critical value. If true, increment, power_z_test

4. Output: calculate and Print the power of the t-test and z-test.

5. Plot:

' Barplot comparing the power of both tests.

' Plot the effect of sample size on the power of the t-test.

R code:

```
mu0 <- 5
mu1 <- 6
sigma <- 2
n <- 30
alpha <- 0.05
N_sim <- 1000
Power_t_test <- 0
Power_z_test <- 0
set.seed(123)
for (i in 1:N_sim){
data <- rnorm(n, mean=mu1, sd = sigma)
```

```
t_test <- t.test(data, mu = mu0, alternative = "greater")
if (t_test$p.value < alpha) {
power_t_test <- power_t_test + 1
}
z_stat <- (mean(data) - mu0) / (sigma / sqrt(n))
z_critical <- qnorm(1 - alpha)
if (z_stat > z_critical) {
power_z_test <- power_z_test + 1
}
}
print(paste("Power of t_test:", power_t_test / N_sim))
print(paste("Power of z-test:", power_z_test / N_sim))
par(mfrow = c(1,2))
barplot(c(power_t_test/N_sim, power_z_test/N_sim),
names.arg = c("t_test", "z_test"), col = ("lightblue", "light
green"), main = "Power comparison", ylab = "Power")
sample_sizes <- seq(10, 100, by = 10)
power_t <- sapply(sample.sizes, function(n){
sum(replicate(N_sim, t.test(rnorm(n, mean = mu1, sd = sigma)
, mu = mu0, alternative = "greater")
$p.value < alpha))/N_sim
})
plot(sample_sizes, power_t, type = "b", col = "blue", main =
"Power vs. sample sizes", xlab = "sample")
```

Sample Input/Output:

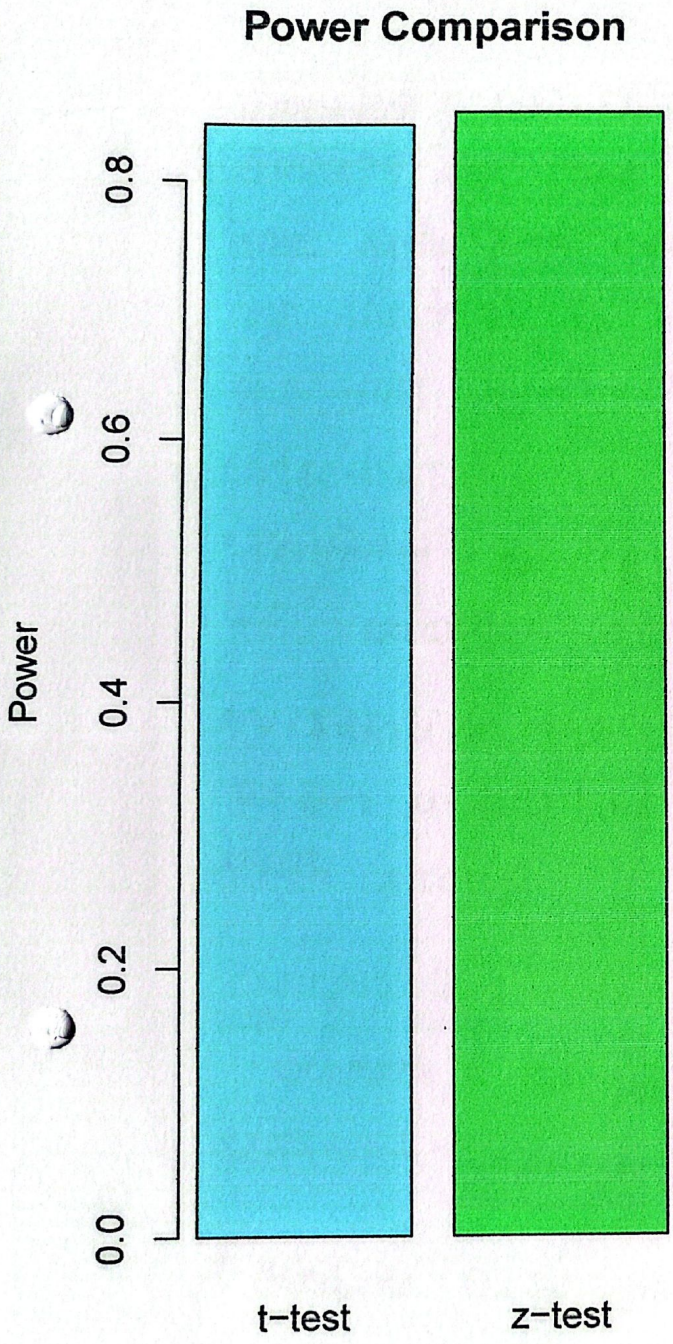
Input:

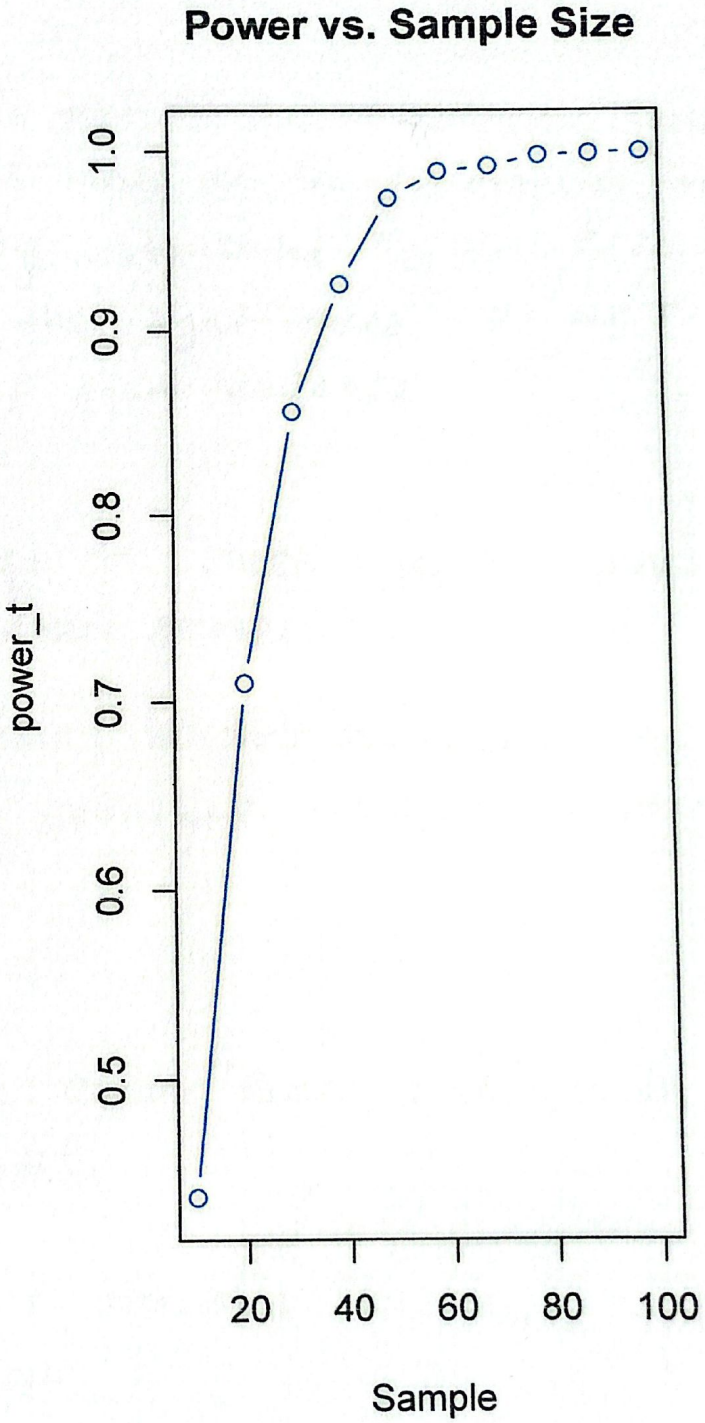
```
mu0 <- 5
mu1 <- 6
sigma <- 2
  n <- 30
alpha <- 0.05
N_sim <- 1000
```

Output:

Power of t-test : 0.844

Power of z-test : 0.854


Power Comparison
Power
0.0
0.2
0.4
0.6
0.8
t−test
z−test


Power vs. Sample Size
power_t
0.5
0.6
0.7
0.8
0.9
1.0
20
40
60
80
100
Sample

Lab 16: Apply Bartlett's test to compare variances across multiple groups.

Theory:

Bartlett's test is used to test if the variances of multiple groups are equal. It is based on the assumption that the data are normally distributed. If the P-value of the test is below the significance level, we reject the null hypothesis of equal variance.

Objective:

- Perform Bartlett's test to check the homogeneity of variances across three groups.
- Visualize the data using boxplots and density plots to better understand the distribution and variability of each group.

Pseudocode:

1. Generate sample data: creat three groups with different standard deviations.

2. Combine Data: creat a combined data frame for performing Bartlett's test.

3. Perform Bartlett's test:
   - Test the null hypothesis that the variances are equal across the groups.

4. Output: Print the result of Bartlett's test.

5. Plot :

- Creat a Boxplot to visualize the spread and variability across groups.
- Creat density plots to compare the distributions of the groups
- Add a legend to distinguish between groups.

R code:

```
set.seed(123)
group1 <- rnorm(30, mean=5, sd=2)
group2 <- rnorm(30, mean =5, sd = 3)
group3 <- rnorm(30, mean=5, sd=4)
data_values <- c(group1, group2, group3)
group_labels <- rep(c("Group 1", "Group 2", "Group 3"), each =30)
bartlett_test_result <- bartlett.test(data_values ~ group_labels)
print(bartlett_test_result)
par(mfrow=c(1,2))
boxplot(group1, group2, group3, col = c("lightblue", "lightgreen", "lightcoral"), main = "Boxplot of Groups", names= c("Group 1", "Group 2", "Group 3"), xlab = "Group", ylab = "value")
plot(density(group1), col="blue", lwd=2, main="Density Plots", xlab="value", ylim=c(0, 0.25))
```

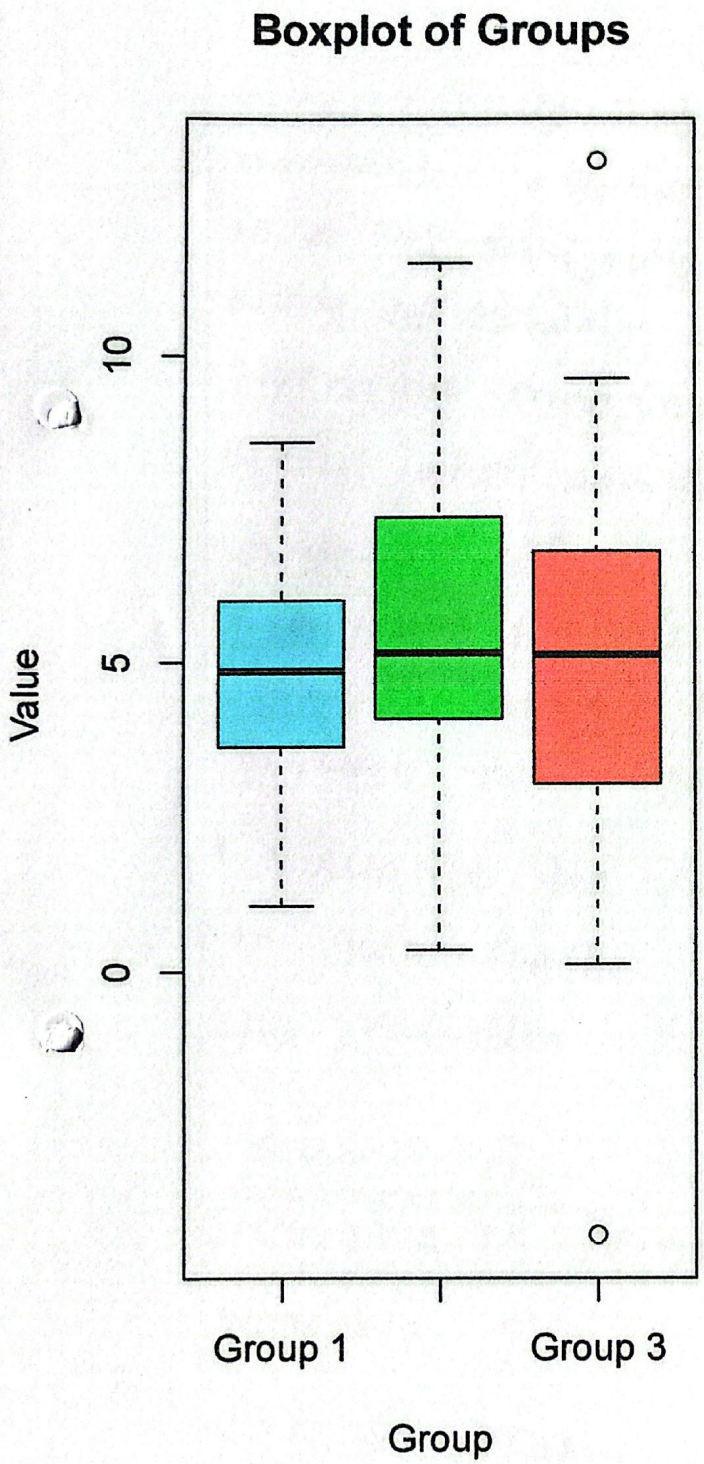
```
lines(density(group2), col="green", lwd=2)
lines(density(group3), col="red", lwd=2)
legend("topright", legend=c("Group 1", "Group 2", "Group 3"), col=c("blue","green","red"), lwd=2)
```

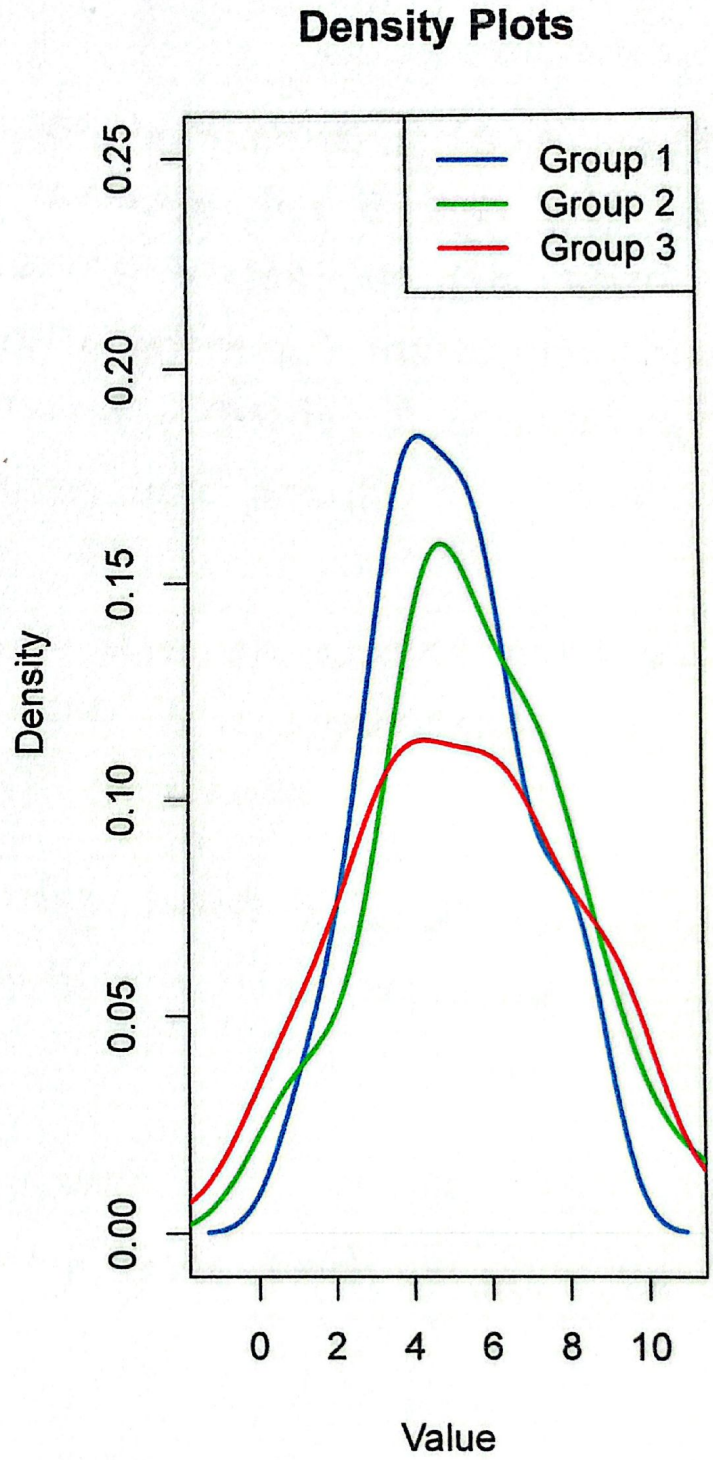
Sample Input:

```
set.seed(123)
group1 <- rnorm(30, mean=5, sd=2)
group2 <- rnorm(30, mean=5, sd=3)
group3 <- rnorm(30, mean=5, sd=4)
```

Output:

```
Bartlett test of homogeneity of variances
data: data_values by group_lables
Bartlett's K-squared = 9.4313, df = 2, p-value = 0.008954
```


Boxplot of Groups
Value
0
5
10
Group 1
Group 3
Group


Density Plots
Group 1
Group 2
Group 3
Density
0.00
0.05
0.10
0.15
0.20
0.25
0
2
4
6
8
10
Value

Lab 17: Perform Fisher's Exact Test on 2x2 Contigency Tables.

Theory:

Fisher's Exact Test is used to determine if there are nonrandom associations between two categorical variables in a 2x2 contigency table. It calculates the exact Probability of obtaining a distribution of values in a table given the marginal sums, making it particularly useful when sample sizes are small.

Objective:

- Perform Fisher's Exact Test on a 2x2 contigency table to assess if there is a significant relationship between two categorical variables.
- Visualize the data using barplots and mosaic plots to gain further insights into the distribution of the variables.

Pseudocode:

1. Creat a 2x2 contingeney Table with values for two variables

2. Perform Fisher's Exact Test on the contingency table to test for associations.

3. Output:

   Print the contingency table and fishers test result.

5. Graphical output:

- Creat a barplot to visualize the counts for each outcome.
- creat a mosaic plot to visualize the relationship between the two categorical variables.

R code:

```
data <-matrix(c(10,5,2,8), nrow=2, byrow=TRUE)
rownames(data) <- c("Group A", "Group B")
colnames(data) <- c("success", "faqlure")
fisher_test_result <-fisher.test(data)
Print(data)
Print(fisher_test_result)
Par(mfrow=c(1,2))
barplot(data, beside=TRUE, col=c("lightblue", "lightgreen"),
main="2 x 2 contingency Table", xlab="Outcome", ylab=
"count")
legend("topright", legend=rownames(data), fill=c("light.
blue", "lightgreen"))
mosaicplot(data, main="mosaicPlot", color=TRUE)
```

Sample Input:

```
data <-matrix(c(10,5,2,8), nrow=2, byrow=TRUE)
rownames(data) <-c("Group A", "Group B")
colnames(data) <-c("success", "failure")
```

Sample output:

	success	failure
Group A	10	5
Group B	2	8

Fisher's Exact Test for count Data

data: data

P-value = 0.04141

alternative hypothesis: true odds ratio is not equal to 1.

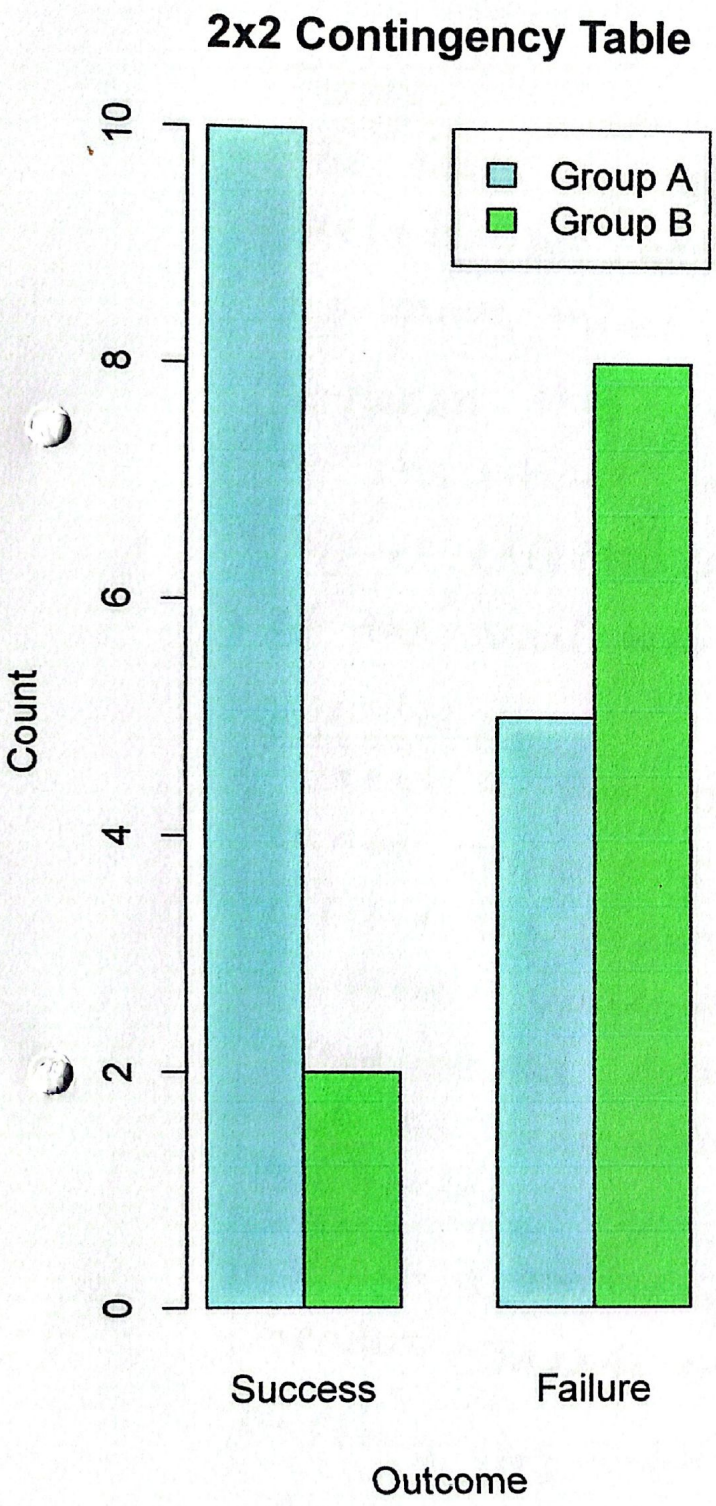
95 percent confidence interval:

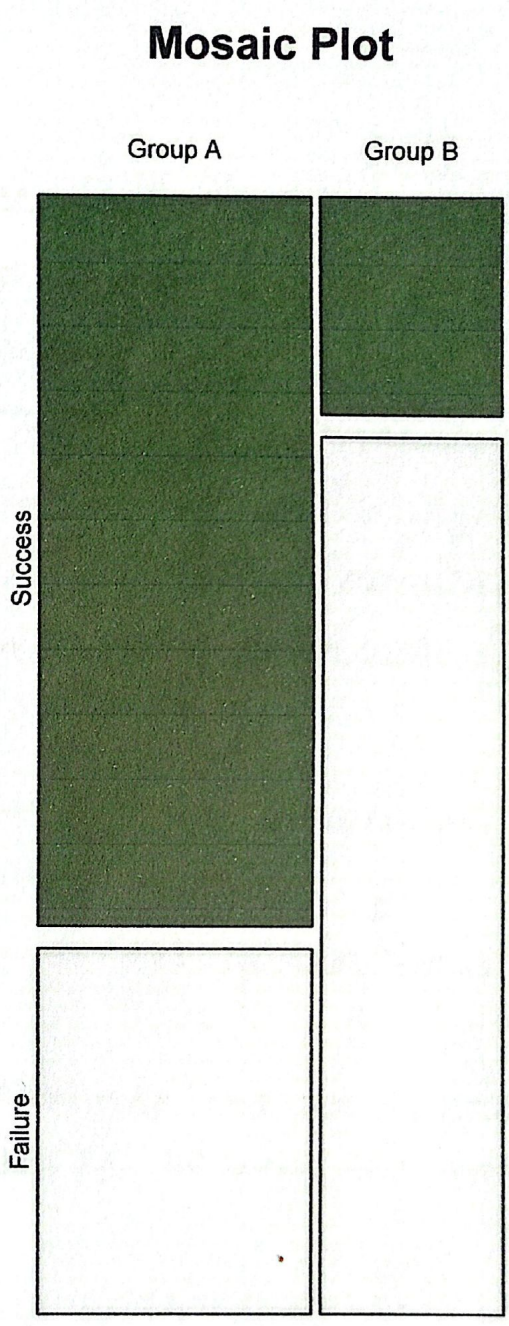
0.9540368 96.2686947

sample estimates:

Odds ratio

7.281573


2x2 Contingency Table
Group A
Group B
Count
0
2
4
6
8
10
Success
Failure
Outcome


Mosaic Plot
Group A
Group B
Success
Failure

Lab 18 : Analyze three way contingency tables using log-linear models.

Theory:

The three way contingency table used to examine the relationship between three categorical variables. A log-linear model is often used to access interactions between these variables. The log-linear model fits a statistical model to the data by modeling the log of expected cell counts as a linear combination of the variables effects and their interactions.

Objective:

- Creat a 3Way contingency table with dimensions of Gender, Treatment and outcome.
- Fit a log-linear model to access interactions betwee n these variables.
- Visualize the data using a heatmap and an interaction plot to understand relationships between the variables.

Pseudocode:

1. Creat a 3D contingency table with dimensions for Gender, Treatment and outcome.
2. Fit a log linear model to the table to identify the relationship between the variables.
3. Output:
    - Print the three way contingency table and the results of the log-linear model

4. Graphical Output:

- Creat a heatmap to visualize the joint distribution of Gender and Treatment, summing over outcome.
- Creat an interaction plot to visualize the counts of outcomes by Treatment and Gender.

R code:

```
data <- array(c(10,5,2,8,3,6,4,7), dim=c(2,2,2))
dimnames(data) <- list(Gender = c("Male","Female"),
                       Treatment = c("Yes","No"),
                       outcome = c("Success","Failure"))
log_linear_model <- loglin(data, margin = list(1,2,3),
fit = TRUE)
Print("Three-way Contingency Table:")
Print(data)
Print("Log-linear model output:")
Print(log_linear_model)
Par(mfrow=c(1,2))
heatmap_matrix <- apply(data, c(1,2), sum).
heatmap(heatmap_matrix, main = "Heatmap of Gender
Vs. Treatment", col= heat.colors(100))
Gender <- rep(c("Male","Female"), times = 4)
Treatment <- rep(c("Yes","No"), each=2, times=2)
outcome <- rep(c("Success","Failure"), each=4)
counts <- as.vector(data)
```

```
interaction.plot(Treatment, Gender, counts,
        main = "Interaction Plot : Treatment and Gender",
        Xlab="Treatment", ylab = "counts", col=c("red",
        blue"), lwd=2)
```

Sample Input:

```
data <- array(c(10, 5, 2, 8, 3, 6, 4, 7), dim=c(2,2,2))
dimnames(data) <- list(Gender = c("Male", "Female"),
                       Treatment = c("Yes", "No"),
                       Outcome = c("success", "failure"))
```

Output:

Three way contigency Table:

Outcome = success

Treatment:

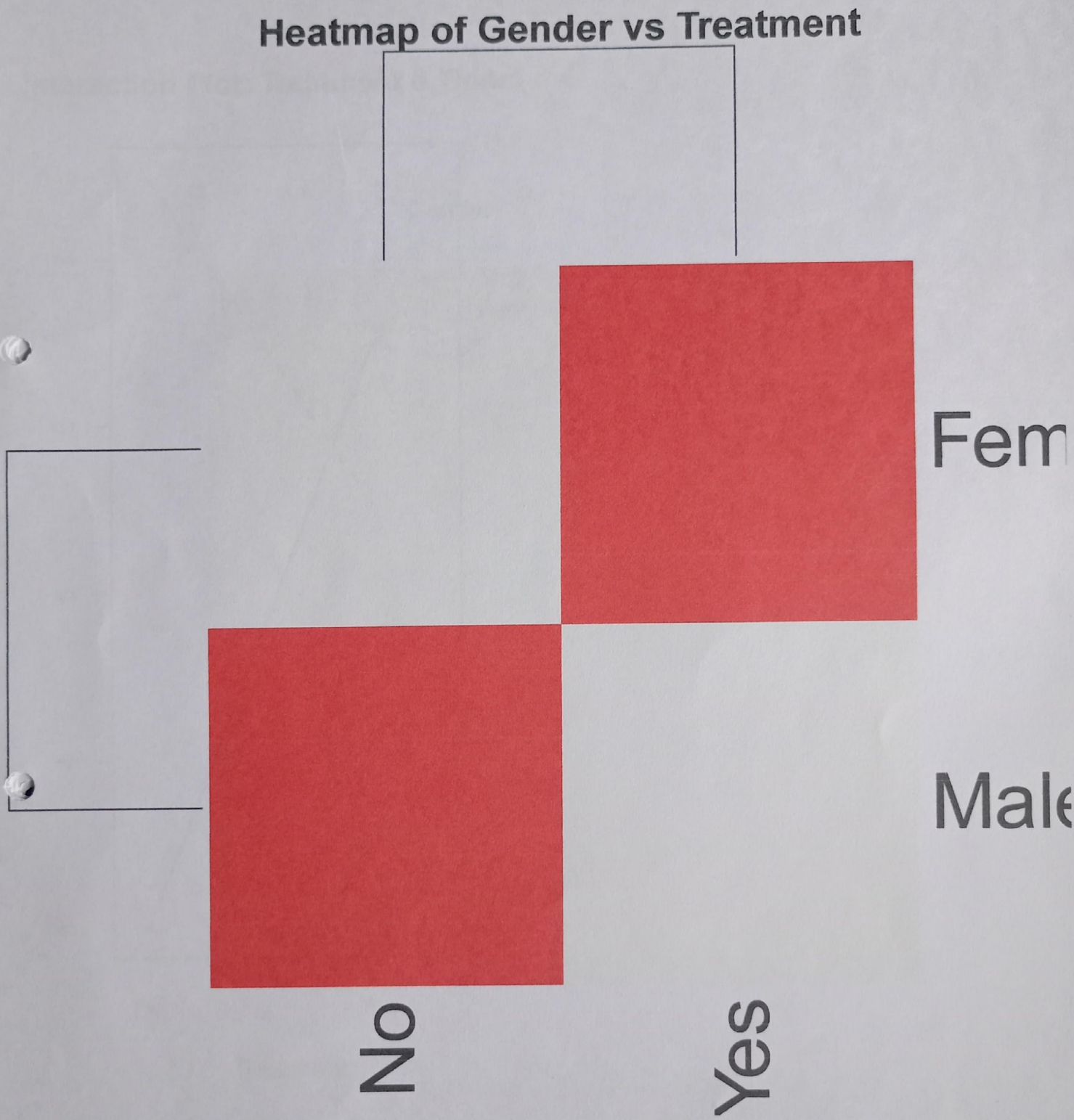
Gender	Yes	No
Male	10	5
Female	2	8

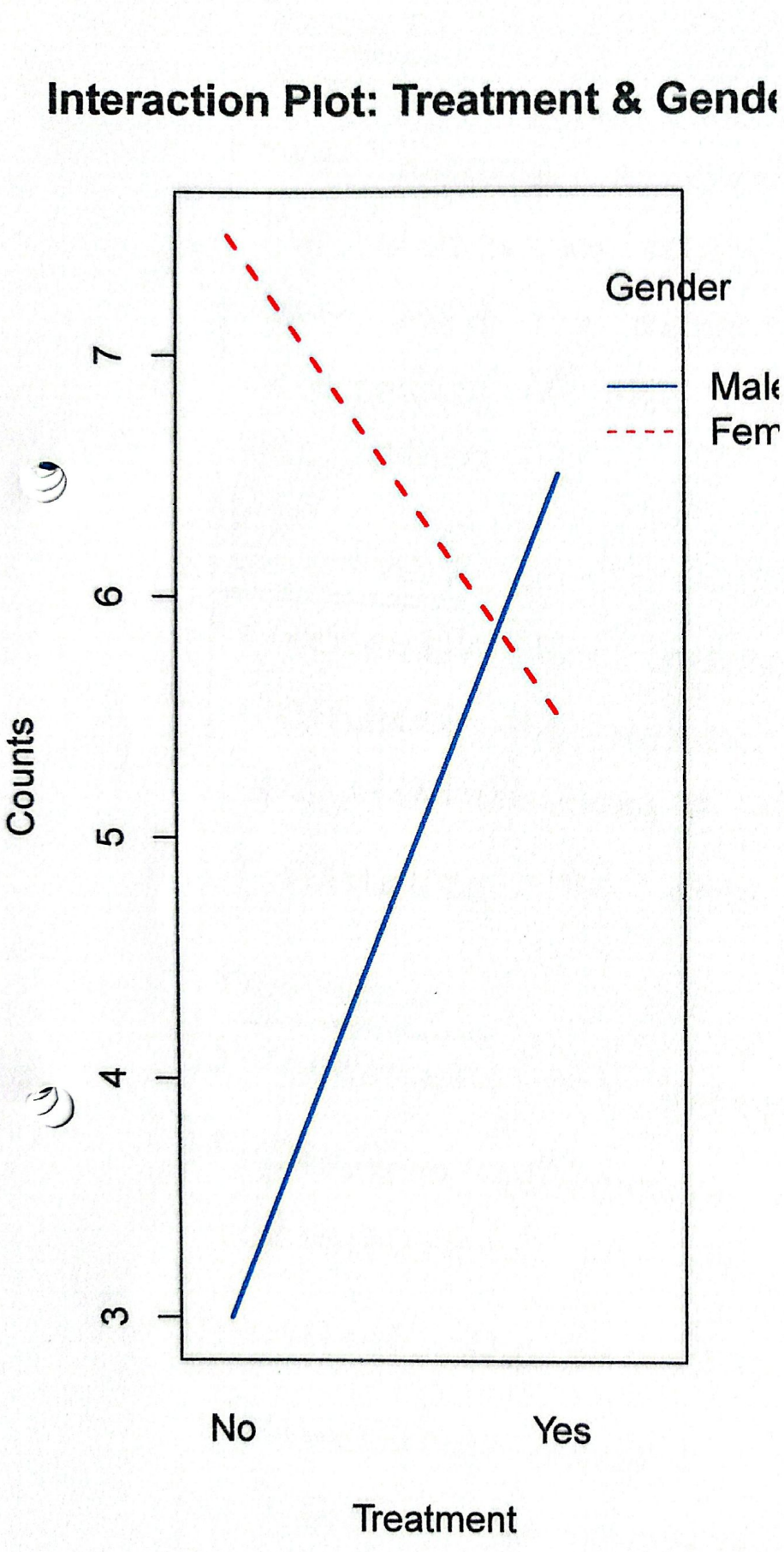
outcome = failure

Treatment:

Gender	Yes	NO
Male	3	6
female	9	7

এই চিত্রটা কিছুটা অস্পষ্ট, একটা কালার দেখা যাচ্ছে না। এটা এমনই ছিলো।




Interaction Plot: Treatment & Gende
Gender
Male
Fem
Counts
7
6
5
4
3
No
Yes
Treatment

Lab 19: Conduct Non-Parametic Tests

Theory:

Non-Parametic tests are statistical methods that do not assume a specific distribution for the data. The willcoxon rank-sum test is a non-Parametic test used to compare two independent samples to determine if one tends to have higher values than the other.

Objective:

- Generate two independent groups of data
- conduct the willcoxon rank-sum test to compare the distributions of the two groups.
- Visualize the data using boxplots and density plots.

Pseudocode:

1. Generate two independent groups of data
2. Perform Willcoxon rank-sum test to compare the distributions of the two groups.
3. Output the test results.
4. Graphical output:
   - Boxplot to compare distributions of the two groups
   - Density plot to visualize the Probability distributions of the two groups.

R code:

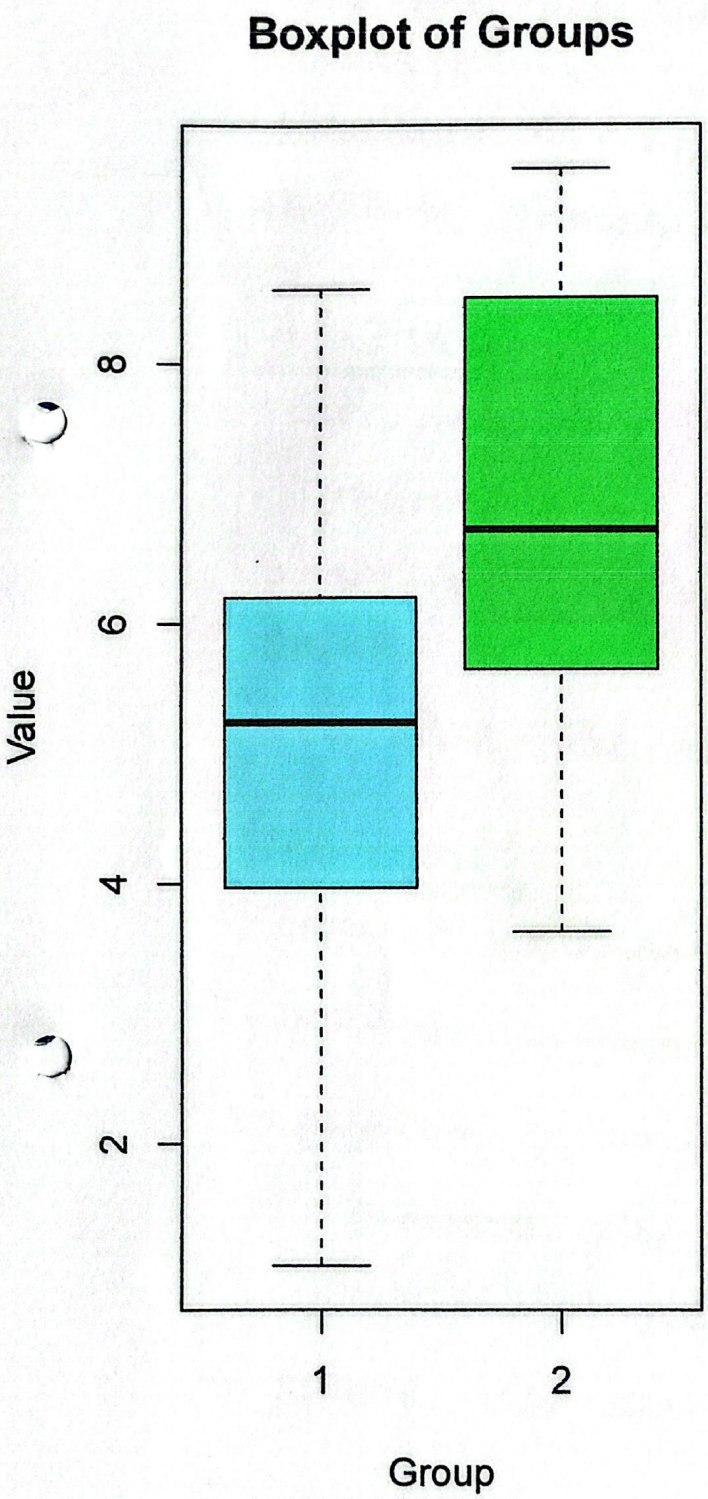
```
set.seed(123)
group1 <- rnorm(20, mean = 5, sd = 2)
group2 <- rnorm(20, mean = 7, sd = 2)

wilcox_test_result <- wilcox.test(group1, group2)
Print(wilcox_test_result)
Par(mfrow = c(1,2))
boxplot(list(group1, group2), col=c("lightblue", lightgreen"), main = "Boxplot of Groups", xlab = "Groups", ylab = "value")
Plot(density(group1), col="blue", lwd=2, main="Density Plots", xlab = "value", ylim = c(0, 0.25))
lines(density(group2), col ="green", lwd = 2)
legend("topright", legend = c("Group 1", "Group 2"), col=c("blue", "green"), lwd = 2)
```

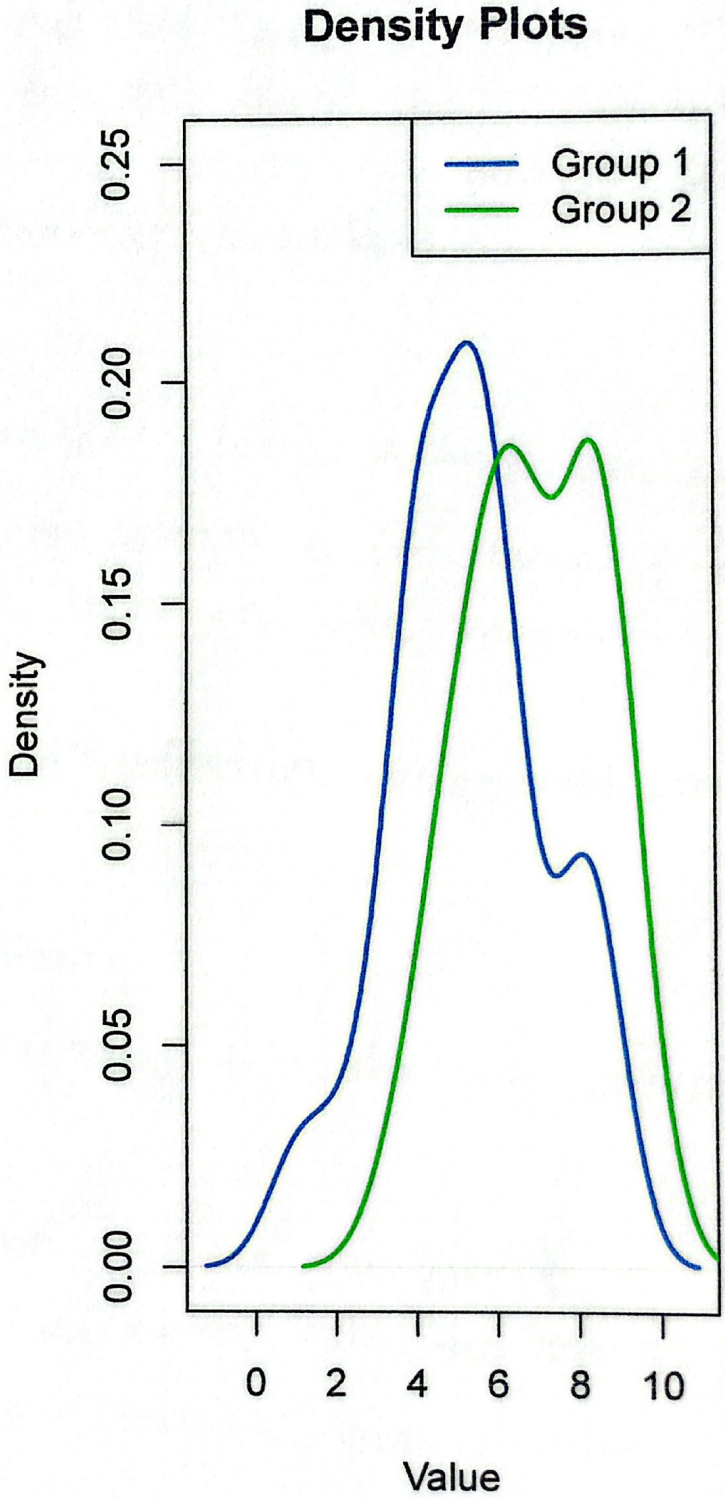
Sample Input:

```
group1 <- rnorm(20, mean =5, sd =2)
group2 <- rnorm(20, mean = 7, sd= 2)
```

Sample output:

```
Wilcoxon rank sum exact test
data: group 1 and group 2
W = 104, p-value = 0.008712
alternative hypothesis: true location shift is not equal to 0.
```


Boxplot of Groups
Value
2
4
6
8
1
2
Group


Density Plots
Group 1
Group 2
Density
0.00
0.05
0.10
0.15
0.20
0.25
0
2
4
6
8
10
Value

Lab 20 : Perform Z-Test for Large sample sizes

Theory :

A z test is a statistical test used to compare the means of two large samples when the population standard deviations are known. It assumes that the data follows a normal distribution.

Objective :

- Generate two independent large sample groups.
- Perform a z-test to determine if there is a significant difference between the means of the groups.
- Visualize the data distribution using histograms.

Pseudocode :

1. Import necessary library
2. Generate random sample data for two groups
3. perform z test :
    - compare the means of the two groups
    - use the standard deviation for two groups
4. Output test results (z-statistic and p-value)
5. Graphical output :
    - Histogram for Group 1
    - Histogram for Group 2

R code:

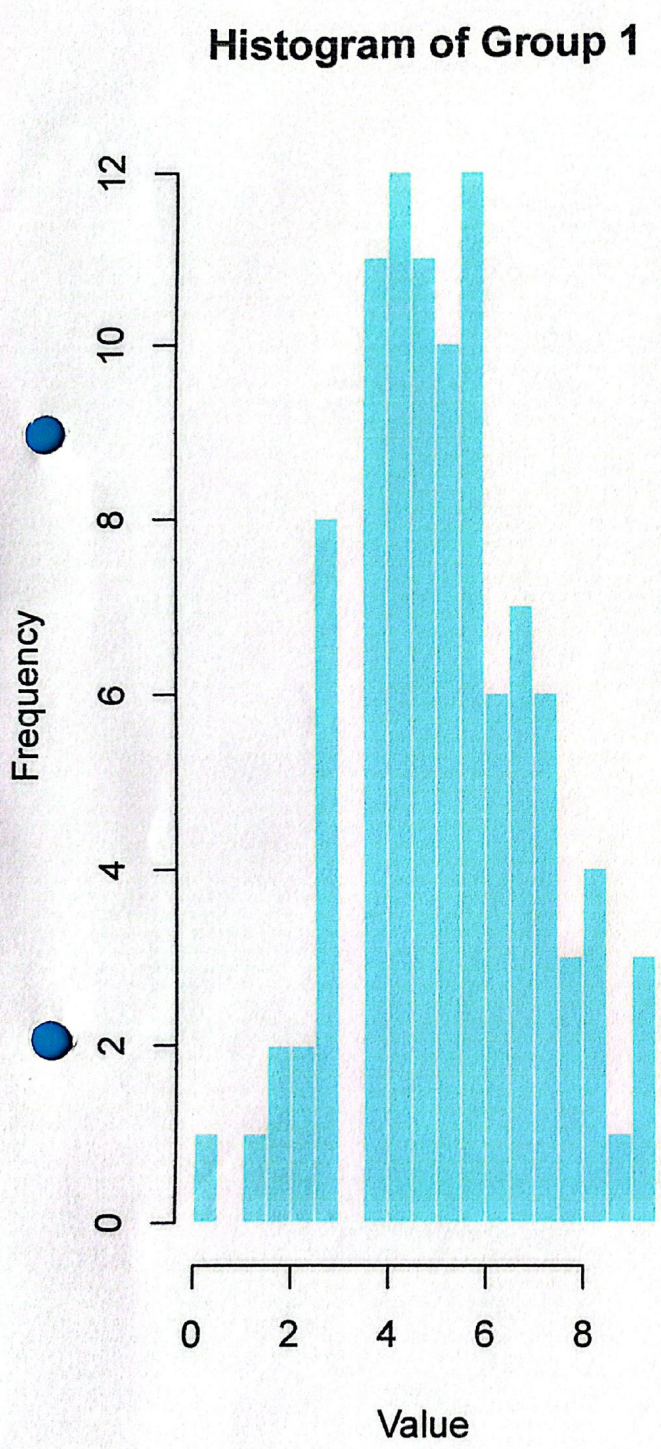
```
library(BSDA)
set.seed(123)
group1 <- rnorm(100, mean=5, sd=2)
group2 <- rnorm(100, mean=6, sd=2)
z_test_result <- z.test(group1, group2, sigma.x=sd(group1), sigma.y=sd(group2))
print(z_test_result)
par(mfrow=c(1,2))
hist(group_1, breaks=30, col="lightblue", main="Histogram of Group 1", xlab="value", border="white")
hist(group2, breaks=30, col="lightgreen", main="Histogram of Group 2", xlab="value", border="white")
```

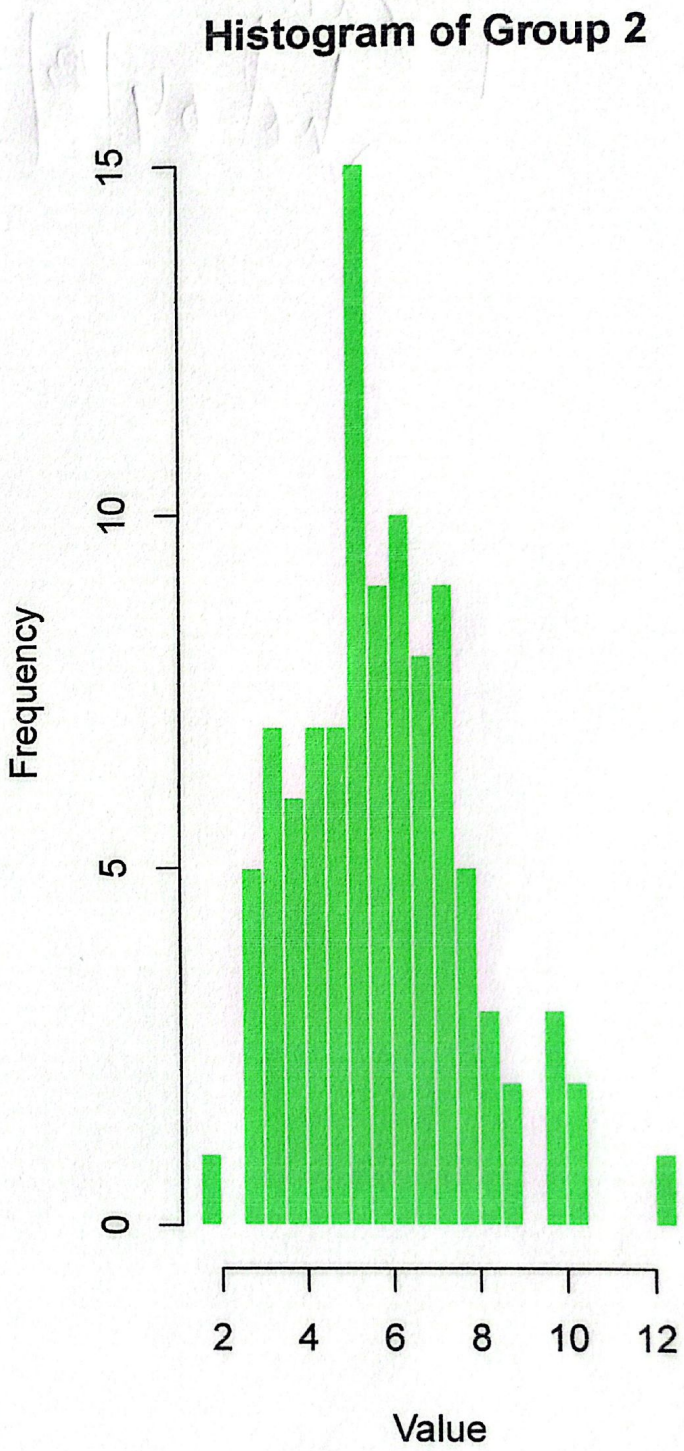
sample Input:

```
group_1 <- rnorm(100, mean=5, sd=2)
group2 <- rnorm(100, mean=6, sd=2)
```

sample output:

```
        Two sample z-Test

data:  group 1 and group 2
z = -2.2714, p-value = 0.02312
alternative hypothesis: true difference in means is not equal to 0.
95 percent confidence interval: -1.12535598 -0.08283319
sample estimates:
   mean of x    mean of y
5.180812 5.784906
```


Histogram of Group 1
Frequency
0
2
4
6
8
10
12
0
2
4
6
8
Value


Histogram of Group 2
Frequency
0
5
10
15
2
4
6
8
10
12
Value