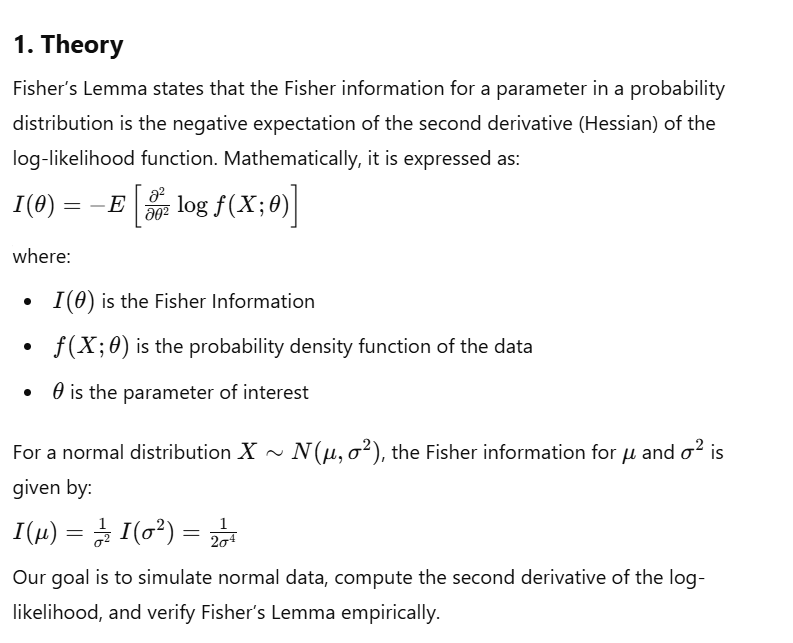
*# Lab 1: Verification of Fisher’s Lemma Using Simulated Data from Normal Distributions*



**2. Objective**

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

**3. Pseudocode**

1. Set parameters: mean (), standard deviation (), sample size (), and number of simulations ().
2. Initialize empty vectors for sample means and sample variances.
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 variances.
5. Display correlation result.
6. Set up a 2×2 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.
9. Generate a Q-Q plot for sample means 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)

*# Creates 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 <- rnorm(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\_vars using Pearson’s correlation coefficient.*

*# Displays the correlation value.*

*# Graphical Output*

par(*mfrow* = c(2, 2))

*# Divides the plotting area into a 2×2 grid for four plots.*

*# Histogram of sample means*

hist(sample\_means, *breaks* = 30, *col* = "lightblue", *main* = "Distribution of Sample Means", *xlab* = "Sample Mean", *border* = "white")

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

*# Histogram of sample variances*

hist(sample\_vars, *breaks* = 30, *col* = "lightgreen", *main* = "Distribution of Sample Variances", *xlab* = "Sample Variance", *border* = "white")

curve(dchisq((x \* (n-1)) / sigma^2, *df* = n-1) \* (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**

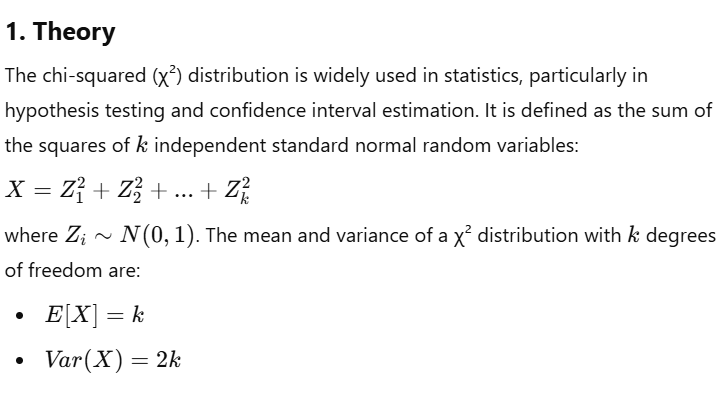
**Input Parameters:**

* Mean (): 5
* Standard deviation (): 2
* Sample size (): 30
* Number of simulations (): 1000

**Output Parameters:**

Correlation between sample mean and sample variance: -0.0288960157244368

*Lab 2: Generation and Analysis of χ2-Distributed Data*



**2. Objective**

The objective of this lab is to generate χ²-distributed 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.

**3. Pseudocode**

1. Set parameters: degrees of freedom and number of simulations .
2. Generate random samples from a χ² distribution.
3. Compute empirical mean and variance.
4. Print computed mean and variance.
5. Generate and display:
   * Histogram with theoretical χ² density curve.
   * Density plot with theoretical χ² curve.
   * Q-Q plot comparing sample quantiles to theoretical quantiles.

**R code**

*# Parameters*

k <- 5  *# Degrees of freedom*

N\_sim <- 1000  *# Number of simulations*

*# Generate chi-squared data*

set.seed(123)

chi2\_data <- rchisq(N\_sim, *df* = k)

*# set.seed(123) ensures reproducibility, meaning running the code multiple times will yield the same random numbers.*

rchisq(N\_sim, *df* = k)

*# generates 1000 random numbers from a 𝜒2 distribution with 5 degrees of freedom.*

*# Compute mean and variance*

mean\_chi2 <- mean(chi2\_data)  *# Compute mean*

var\_chi2 <- var(chi2\_data)  *# Compute variance*

print(paste("Mean:", mean\_chi2))

print(paste("Variance:", var\_chi2))

*# Graphical Output*

par(*mfrow* = c(1, 3))

*# Set plotting layout to 1 row, 3 columns*

*# Histogram*

hist(chi2\_data, *breaks* = 30, *col* = "lightblue", *probability* = TRUE, *main* = "Chi-Squared Distribution", *xlab* = "Value", *border* = "white")

*# lots a histogram with density on the y-axis.*

curve(dchisq(x, *df* = k), *add* = TRUE, *col* = "red", *lwd* = 2)

*# overlays the theoretical χ2 density curve in red.*

*# Density plot*

plot(density(chi2\_data), *col* = "blue", *lwd* = 2, *main* = "Density Plot", *xlab* = "Value")

*#  creates a smooth density plot of the simulated data.*

curve(dchisq(x, *df* = k), *add* = TRUE, *col* = "red", *lwd* = 2)

*# overlays the theoretical χ2 density function.*

*# Q-Q plot (Quantile-Quantile Plot)*

qqplot(qchisq(ppoints(N\_sim), *df* = k), chi2\_data, *main* = "Q-Q Plot for Chi-Squared Data", *col* = "blue", *xlab* = "Theoretical Quantiles", *ylab* = "Sample Quantiles")

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

**Sample Input/ Output**

**Input Parameters:**

k = 5

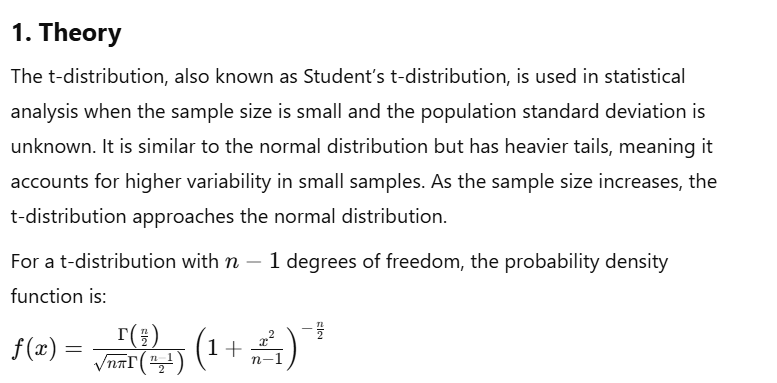
N\_sim = 1000

**Output Parameters:**

Mean: 4.79389985708151

Variance: 8.4017924901745

*Lab 3: Comparison of t-Distribution with Normal Distribution for Small Sample Sizes*



**Objective**

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

* Generating t-distributed and normal data.
* Visualizing their histograms and density functions.
* Comparing quantiles using Q-Q plots.

**3. Pseudocode**

1. Set parameters: sample size and number of simulations .
2. Generate random values from a 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 quantiles.

R code

*# Parameters*

n <- 10  *# Sample size*

N\_sim <- 1000  *# Number of simulations*

*# Generate data*

set.seed(123)  *# Ensures reproducibility*

t\_data <- rt(N\_sim, *df* = n-1)  *# Generate t-distributed data with df = n-1*

normal\_data <- rnorm(N\_sim)  *# Generate standard normal data*

*# Graphical Output*

par(*mfrow* = c(2, 2)) *# Set layout: 2 rows, 2 columns*

*# Histogram of t-distribution*

hist(t\_data, *breaks* = 30, *col* = "lightblue", *probability* = TRUE, *main* = "t-Distribution", *xlab* = "Value", *border* = "white")

curve(dt(x, *df* = n-1), *add* = TRUE, *col* = "red", *lwd* = 2) *#overlays the theoretical t-distribution density.*

*# Histogram of normal distribution*

hist(normal\_data, *breaks* = 30, *col* = "lightgreen", *probability* = TRUE, *main* = "Normal Distribution", *xlab* = "Value", *border* = "white")

curve(dnorm(x), *add* = TRUE, *col* = "blue", *lwd* = 2)  *# overlays the theoretical normal density.*

*# Density plot comparison*

plot(density(t\_data), *col* = "red", *lwd* = 2, *main* = "Density Comparison", *xlab* = "Value", *ylim* = c(0, 0.4))

lines(density(normal\_data), *col* = "blue", *lwd* = 2)

legend("topright", *legend* = c("t-Distribution", "Normal Distribution"), *col* = c("red", "blue"), *lwd* = 2)

*# Q-Q plot for t-distribution*

qqplot(qt(ppoints(N\_sim), *df* = n-1), t\_data, *main* = "Q-Q Plot for t-Distribution", *col* = "red", *xlab* = "Theoretical Quantiles", *ylab* = "Sample Quantiles")

abline(0, 1, *col* = "blue", *lwd* = 2)

Sample Input/Output:

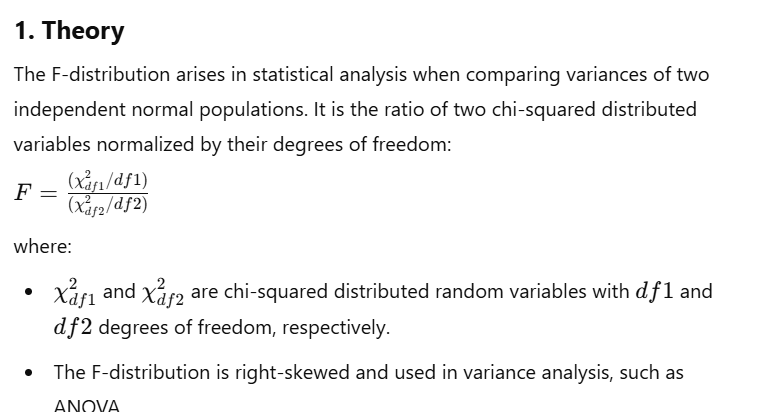
Input parameter:

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).

# Lab 4: Simulation of F-Distributed Data and Its Relationship with χ2-Distributions 

**Objective**

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

* Histogram and density plots.
* Q-Q plot for distribution verification.
* Boxplot to observe spread and skewness.

**3. Pseudocode**

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.

**4. R Code**

# Parameters

df1 <- 5 # Numerator degrees of freedom

df2 <- 10 # Denominator degrees of freedom

N\_sim <- 1000 # Number of simulations

# Generate F-distributed data

set.seed(123) # Ensures reproducibility

chi2\_1 <- rchisq(N\_sim, df = df1) # Generate Chi-Squared data with df1

chi2\_2 <- rchisq(N\_sim, df = df2) # Generate Chi-Squared data with df2

f\_data <- (chi2\_1 / df1) / (chi2\_2 / df2) # Compute F-distributed values

# Graphical Output

par(mfrow = c(2, 2)) # Set layout: 2 rows, 2 columns

# Histogram

hist(f\_data, breaks = 30, col = "lightblue", probability = TRUE, main = "F-Distribution", xlab = "Value", border = "white")

curve(df(x, df1 = df1, df2 = df2), add = TRUE, col = "red", lwd = 2)

# Density plot

plot(density(f\_data), col = "blue", lwd = 2, main = "Density Plot", xlab = "Value")

curve(df(x, df1 = df1, df2 = df2), add = TRUE, col = "red", lwd = 2)

# Q-Q plot

qqplot(qf(ppoints(N\_sim), df1 = df1, df2 = df2), f\_data, main = "Q-Q Plot for F-Distribution", col = "blue", xlab = "Theoretical Quantiles", ylab = "Sample Quantiles")

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

# Boxplot

boxplot(f\_data, col = "lightgreen", main = "Boxplot of F-Distributed Data", ylab = "Value")

# The boxplot shows the spread and skewness of the F-distributed data.

# Since the F-distribution is right-skewed, the upper whisker is typically longer.

**Sample Input and 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 distributions.
* Q-Q plot indicating the goodness of fit.
* Boxplot illustrating the skewness of the F-distribution.

*Lab 5: 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 distributions.

**Objective**

1. **Simulate 1000 random samples** from a normal distribution with mean (mu = 0) and standard deviation (sigma = 1), 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.

**Pseudocode**

1. **Set parameters**:
   * mu = 0 (mean of the normal distribution)
   * sigma = 1 (standard deviation of the normal distribution)
   * n = 20 (sample size for each simulation)
   * N\_sim = 1000 (number of simulations)
2. **Initialize vectors** to store the results of the medians and ranges for each simulation.
3. **Simulation Loop**:
   * For each of the 1000 simulations:
     + Generate a random sample of size n from the normal distribution.
     + Compute and store the **median** of the sample.
     + Compute and store the **range** of the sample (difference between max and min values).
4. **Set graphical layout** to a 2×2 grid.
5. **Create histograms** for the distribution of medians and ranges:
   * Overlay the theoretical normal distribution curve for medians.
6. **Create a density plot** for the distribution of medians.
   * Overlay the theoretical normal distribution curve.
7. **Create a boxplot** comparing the distributions of medians and ranges.

R code

*# Parameters*

mu <- 0

sigma <- 1

n <- 20

N\_sim <- 1000

*# Initialize vectors*

medians <- numeric(N\_sim)

ranges <- numeric(N\_sim)

*# Simulation*

set.seed(123)

*for* (i *in* 1:N\_sim) {

  data <- rnorm(n, *mean* = mu, *sd* = sigma)

  medians[i] <- median(data)

  ranges[i] <- max(data) - min(data)

}

*# Graphical Output*

par(*mfrow* = c(2, 2))

*# Histogram of medians*

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)

*# Histogram of ranges*

hist(ranges, *breaks* = 30, *col* = "lightgreen", *probability* = TRUE, *main* = "Distribution of Ranges", *xlab* = "Range", *border* = "white")

*# Density plot of medians*

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 of medians and ranges*

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

Sample Input & Output

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

*# Lab 6: 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, though it is biased, and needs to be adjusted when constructing confidence intervals.
* A **confidence interval** provides a range of values within which 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 (mean = 5, standard deviation = 2).
2. **Estimate the population parameters** (mean and variance) from the sample.
3. Calculate the **95% confidence interval** for both the sample mean and the sample variance.
4. **Visualize** the sample data with:
   * A histogram showing the sample data and highlighting the sample mean and the confidence interval for the mean.
   * A boxplot to summarize the distribution of the sample data.

**Pseudocode**

1. **Set Parameters**:
   * mu = 5 (population mean)
   * sigma = 2 (population standard deviation)
   * n = 30 (sample size)
   * N\_sim = 1000 (number of simulations, though not used directly here)
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 = var(sample\_data)
4. **Confidence Intervals**:
   * Compute the **95% confidence interval for the mean** using the t.test function.
   * Compute the **95% confidence interval for the variance** using the chi-squared 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**:
   * **Histogram**: Display a histogram of the sample data with vertical lines indicating the sample mean and the confidence interval for the mean.
   * **Boxplot**: Display a boxplot of the sample data.

R code:

*# Parameters*

mu <- 5

sigma <- 2

n <- 30

N\_sim <- 1000

*# Generate sample data*

set.seed(123)

sample\_data <- rnorm(n, *mean* = mu, *sd* = sigma)

*# Point estimates*

sample\_mean <- mean(sample\_data)

sample\_var <- var(sample\_data)

*# Confidence intervals*

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))

*# Output*

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))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram with mean and CI*

hist(sample\_data, *breaks* = 30, *col* = "lightblue", *main* = "Sample Data with Mean", *xlab* = "Value", *border* = "white")

abline(*v* = sample\_mean, *col* = "red", *lwd* = 2)

abline(*v* = conf\_int\_mean, *col* = "blue", *lty* = 2)

*# Boxplotboxplot(sample\_data, col = "lightgreen", main = "Boxplot of Sample Data", ylab = "Value")*

Sample Input/Output:

**Input parameter:**

Sample Mean: 4.90579248793639

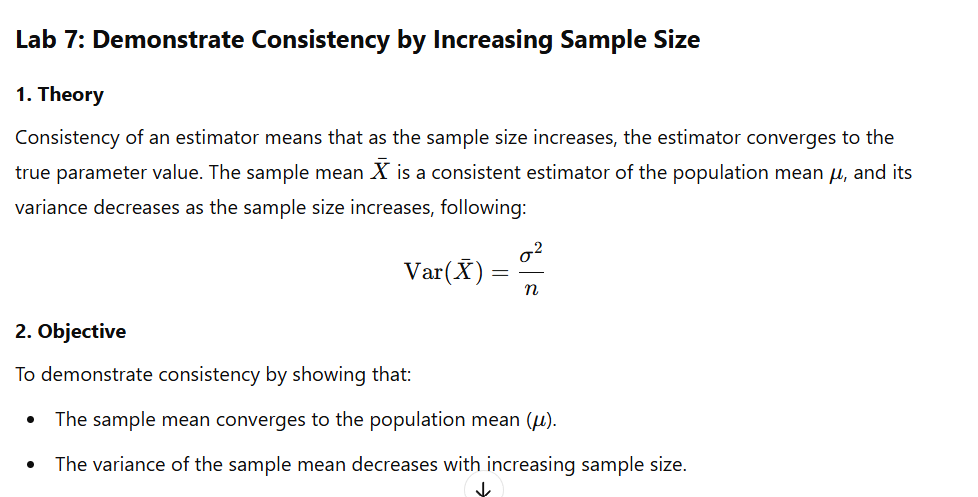
Sample Variance: 3.84968498055034

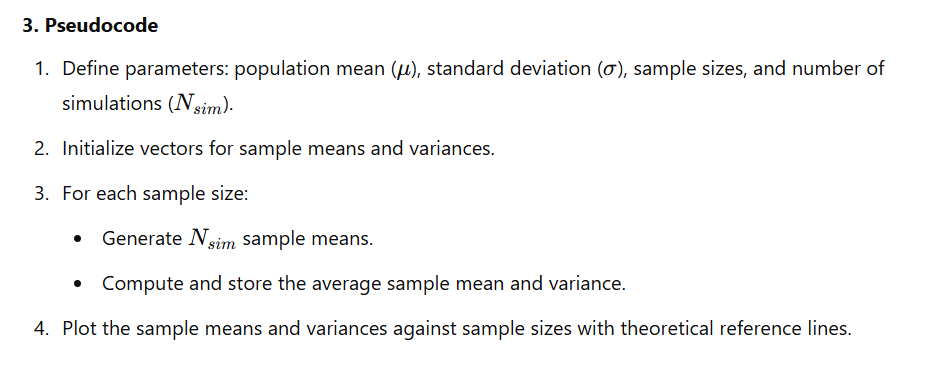
95% CI for Mean: (4.1731467163713, 5.63843825950147)

95% CI for Variance: (2.44171660432967, 6.95708641148571)

**Output (Graphical Plots):**

A histogram of the sample data with Mean





R code:

mu <- 5

sigma <- 2

sample\_sizes <- c(10, 30, 100, 500, 1000)

N\_sim <- 1000

*# Initialize vectors*

means <- numeric(length(sample\_sizes))

vars <- numeric(length(sample\_sizes))

*# Simulation*

set.seed(123)

*for* (i *in* 1:length(sample\_sizes)) {

 n <- sample\_sizes[i]

 sample\_means <- replicate(N\_sim, mean(rnorm(n, *mean* = mu, *sd* = sigma)))

 means[i] <- mean(sample\_means)

 vars[i] <- var(sample\_means)

}

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Plot sample means*

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 variances*

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:

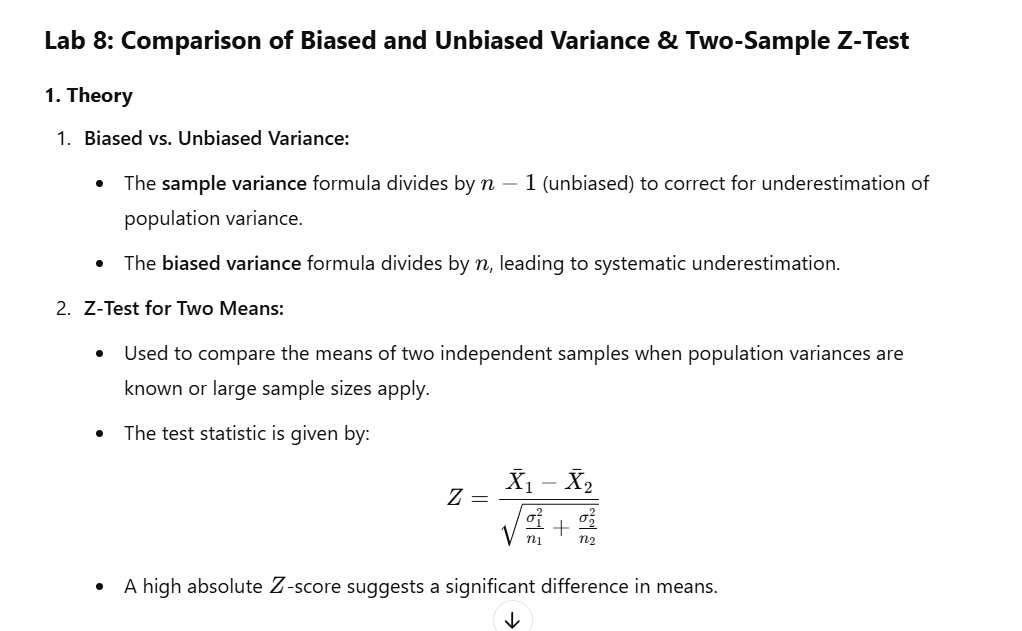
**Input:**

mu = 5

sigma = 2

sample\_sizes = [10, 30, 100, 500, 1000]

N\_sim = 1000



**2. 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.

**3. Pseudocode**

1. **Variance Comparison:**
   * Set parameters: μ,σ,n,Nsim\mu, \sigma, n, N\_{sim}μ,σ,n,Nsim​.
   * Initialize vectors for sample variance (unbiased) and biased variance.
   * For each simulation:
     + Generate a normal sample.
     + Compute unbiased and biased variance.
   * Plot histograms of both variances.
2. **Two-Sample Z-Test:**
   * Generate two normal samples with different means.
   * Perform a **Z-test** using z.test().
   * Plot histograms of both groups.

R code:

*# Load necessary library*

library(BSDA)

*# Parameters*

mu <- 5

sigma <- 2

n <- 30

N\_sim <- 1000

*# Initialize vectors*

sample\_vars <- numeric(N\_sim)

biased\_vars <- numeric(N\_sim)

*# Simulation*

set.seed(123)

*for* (i *in* 1:N\_sim) {

  data <- rnorm(n, *mean* = mu, *sd* = sigma)

  sample\_vars[i] <- var(data)  *# Unbiased variance*

  biased\_vars[i] <- sum((data - mean(data))^2) / n  *# Biased variance*

}

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram of unbiased variances*

hist(sample\_vars, *breaks* = 30, *col* = "lightblue",

*main* = "Unbiased Sample Variance", *xlab* = "Variance", *border* = "white")

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

*# Histogram of biased variances*

hist(biased\_vars, *breaks* = 30, *col* = "lightgreen",

*main* = "Biased Sample Variance", *xlab* = "Variance", *border* = "white")

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

*# Define groups for z-test*

set.seed(123)

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

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

*# Perform z-test*

z\_test\_result <- z.test(group1, group2, *sigma.x* = sd(group1), *sigma.y* = sd(group2))

*# Output test result*

print(z\_test\_result)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram of groups*

hist(group1, *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 and Output**

**Input:**

mu = 5

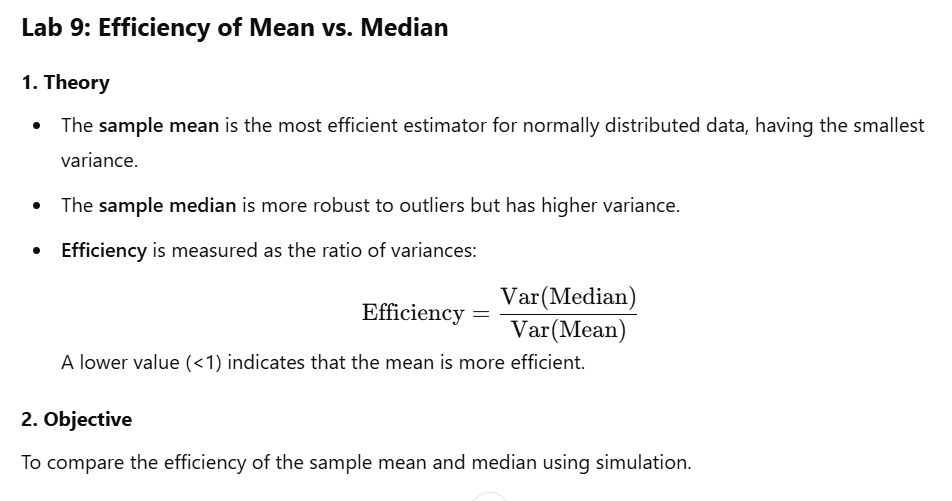
sigma = 2

n = 30

N\_sim = 1000

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

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



**3. Pseudocode**

1. Set parameters: μ,σ,n,Nsim\mu, \sigma, n, N\_{\text{sim}}μ,σ,n,Nsim​.
2. Initialize vectors for sample means and medians.
3. For each simulation:
   * Generate normal data.
   * Compute sample mean and median.
4. Compute efficiency as the ratio of variances.
5. Plot histograms of means and medians.

R code:

*# Parameters*

mu <- 5

sigma <- 2

n <- 30

N\_sim <- 1000

*# Initialize vectors*

means <- numeric(N\_sim)

medians <- numeric(N\_sim)

*# Simulation*

set.seed(123)

*for* (i *in* 1:N\_sim) {

  data <- rnorm(n, *mean* = mu, *sd* = sigma)

  means[i] <- mean(data)

  medians[i] <- median(data)

}

*# Efficiency (ratio of variances)*

efficiency <- var(medians) / var(means)

print(paste("Efficiency (Median/Mean):", efficiency))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram of means*

hist(means, *breaks* = 30, *col* = "lightblue",

*main* = "Distribution of Sample Means",

*xlab* = "Value", *border* = "white")

*# Histogram of medians*

hist(medians, *breaks* = 30, *col* = "lightgreen",

*main* = "Distribution of Sample Medians",

*xlab* = "Value", *border* = "white")

**Sample Input & Output**

**Input:**

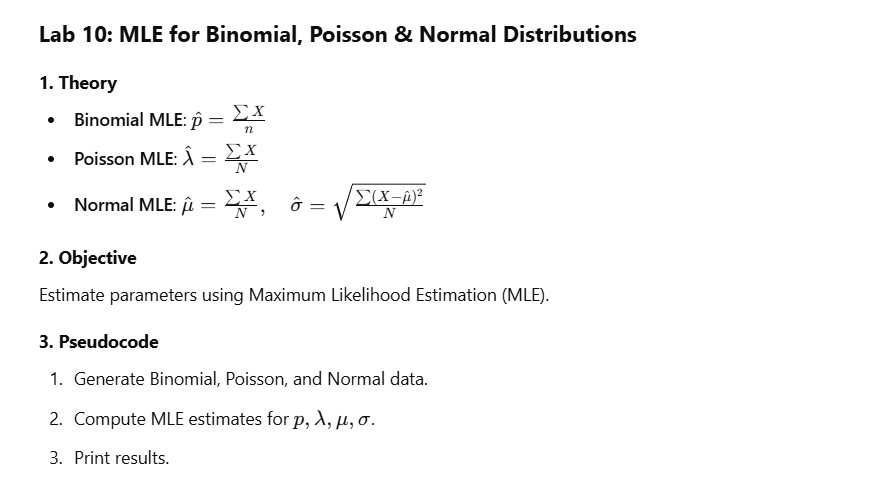
mu = 5

sigma = 2

n = 30

N\_sim = 1000

Output: Efficiency (Median/Mean): 1.5085760936887



R code:

*# Lab 10: Derive MLEs for Binomial, Poisson, and Normal Distributions*

*# Binomial MLE*

n\_binom <- 20

p\_true <- 0.6

data\_binom <- rbinom(100, *size* = n\_binom, *prob* = p\_true)

p\_mle <- mean(data\_binom) / n\_binom

*# Poisson MLE*

lambda\_true <- 3

data\_pois <- rpois(100, *lambda* = lambda\_true)

lambda\_mle <- mean(data\_pois)

*# Normal MLE*

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))

*# Output*

print(paste("Binomial MLE for p:", p\_mle))

print(paste("Poisson MLE for lambda:", lambda\_mle))

print(paste("Normal MLE for mu:", mu\_mle))

print(paste("Normal MLE for sigma:", sigma\_mle))

Sample output:

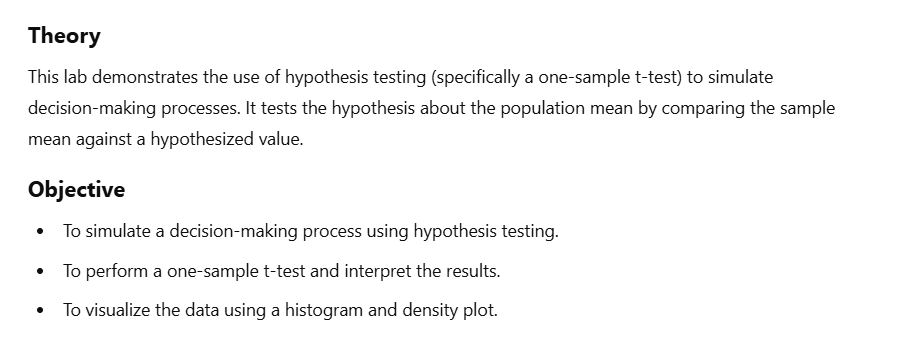
Binomial MLE for p: 0.603

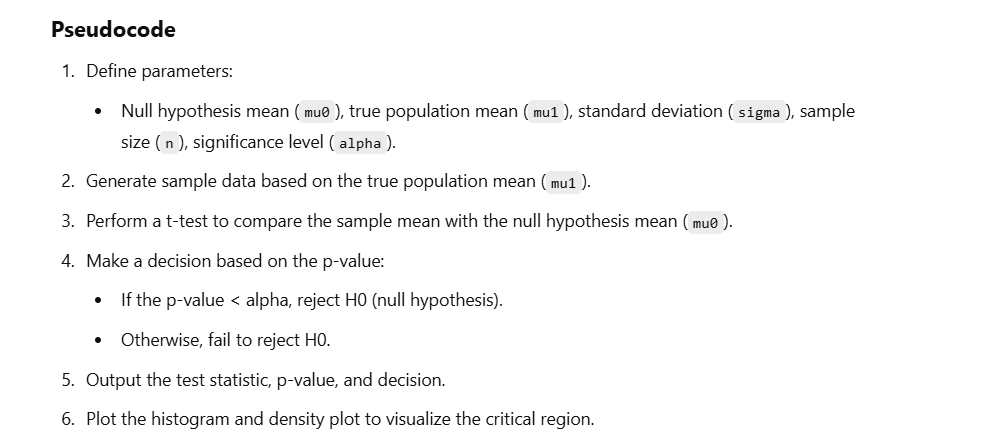
Poisson MLE for lambda: 2.89

Normal MLE for mu: 5.35420581393028

Normal MLE for sigma: 1.8574597852958

*# Lab 11: Simulate Decision-Making Processes Using Hypothesis Testing*





R code:

mu0 <- 5 *# Null hypothesis mean*

mu1 <- 6 *# True population mean*

sigma <- 2 *# Population standard deviation*

n <- 30 *# Sample size*

alpha <- 0.05 *# Significance level*

*# Generate sample data*

set.seed(123)

sample\_data <- rnorm(n, *mean* = mu1, *sd* = sigma)

*# Perform t-test*

t\_test\_result <- t.test(sample\_data, *mu* = mu0, *alternative* = "greater")

*# Decision*

*if* (t\_test\_result$p.value < alpha) {

 decision <- "Reject H0"

} *else* {

 decision <- "Fail to reject H0"

}

*# Output*

print(paste("Test Statistic:", t\_test\_result$statistic))

print(paste("P-value:", t\_test\_result$p.value))

print(paste("Decision:", decision))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram with critical region*

hist(sample\_data, *breaks* = 30, *col* = "lightblue", *main* = "Sample Data", *xlab* = "Value", *border* = "white")

abline(*v* = mu0, *col* = "red", *lwd* = 2)

abline(*v* = mean(sample\_data), *col* = "blue", *lwd* = 2)

*# Density plot with critical region*

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 # Null hypothesis mean

mu1 <- 6 # True population mean

sigma <- 2 # Population standard deviation

n <- 30 # Sample size

alpha <- 0.05 # Significance level

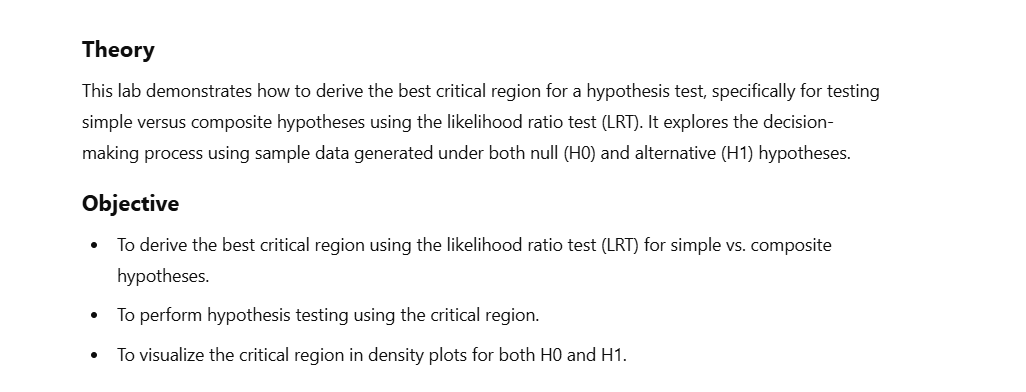
Sample Output:

Test Statistic: 2.52858027419059

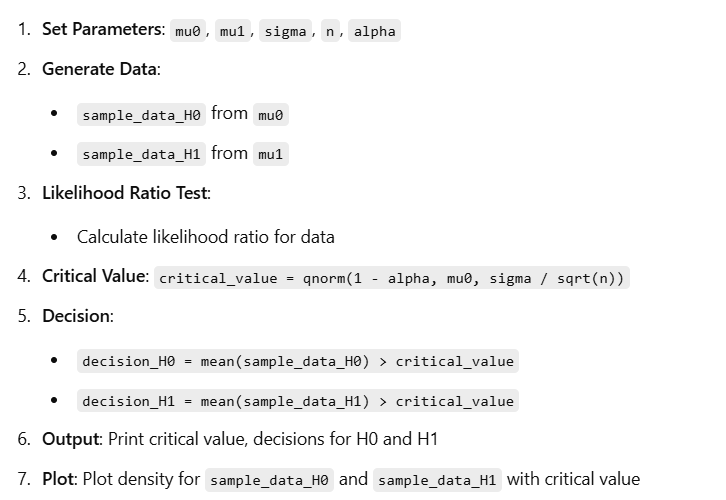
P-value: 0.00857566021520242

Decision: Reject H0

*# Lab 12: Derive the Best Critical Region for Simple vs. Composite Hypotheses*



Pseudocode:



R code:

mu0 <- 5 *# Null hypothesis mean*

mu1 <- 6 *# Alternative hypothesis mean*

sigma <- 2 *# Population standard deviation*

n <- 30 *# Sample size*

alpha <- 0.05 *# Significance level*

*# Generate sample data under H0*

set.seed(123)

sample\_data\_H0 <- rnorm(n, *mean* = mu0, *sd* = sigma)

*# Generate sample data under H1*

sample\_data\_H1 <- rnorm(n, *mean* = mu1, *sd* = sigma)

*# Likelihood ratio test*

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 region*

critical\_value <- qnorm(1 - alpha, *mean* = mu0, *sd* = sigma / sqrt(n))

*# Decision*

decision\_H0 <- mean(sample\_data\_H0) > critical\_value

decision\_H1 <- mean(sample\_data\_H1) > critical\_value

*# Output*

print(paste("Critical Value:", critical\_value))

print(paste("Decision under H0:", decision\_H0))

print(paste("Decision under H1:", decision\_H1))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Density plot under H0*

plot(density(sample\_data\_H0), *col* = "blue", *lwd* = 2, *main* = "Density under H0", *xlab* = "Value")

abline(*v* = critical\_value, *col* = "red", *lty* = 2)

*# Density plot under H1*

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 # Null hypothesis mean

mu1 <- 6 # Alternative hypothesis mean

sigma <- 2 # Population standard deviation

n <- 30 # Sample size

alpha <- 0.05 # Significance level

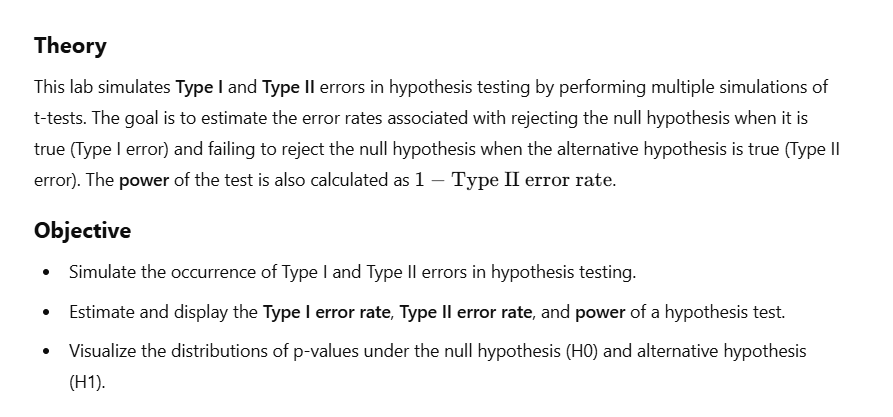
**output:**

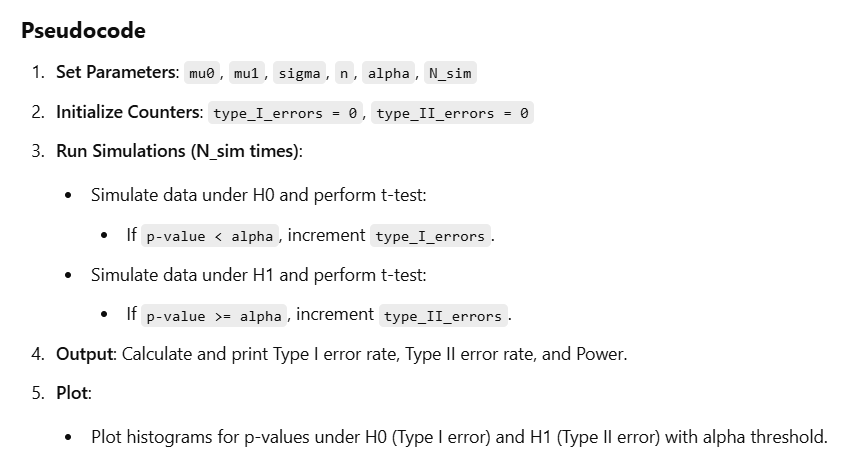
Critical Value: 5.643234

Decision under H0: FALSE

Decision under H1: TRUE

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





R code:

mu0 <- 5 *# Null hypothesis mean*

mu1 <- 6 *# Alternative hypothesis mean*

sigma <- 2 *# Population standard deviation*

n <- 30 *# Sample size*

alpha <- 0.05 *# Significance level*

N\_sim <- 1000 *# Number of simulations*

*# Initialize counters*

type\_I\_errors <- 0

type\_II\_errors <- 0

*# Simulation*

set.seed(123)

*for* (i *in* 1:N\_sim) {

*# Simulate data under H0*

 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

 }

*# Simulate data under H1*

 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

 }

}

*# Output*

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)))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Type I Error Distribution*

hist(replicate(N\_sim, t.test(rnorm(n, *mean* = mu0, *sd* = sigma), *mu* = mu0, *alternative* = "greater")$p.value),

*breaks* = 30, *col* = "lightblue", *main* = "P-values under H0", *xlab* = "P-value", *border* = "white")

abline(*v* = alpha, *col* = "red", *lwd* = 2)

*# Type II Error Distribution*

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 # Null hypothesis mean

mu1 <- 6 # Alternative hypothesis mean

sigma <- 2 # Population standard deviation

n <- 30 # Sample size

alpha <- 0.05 # Significance level

N\_sim <- 1000 # Number of simulations

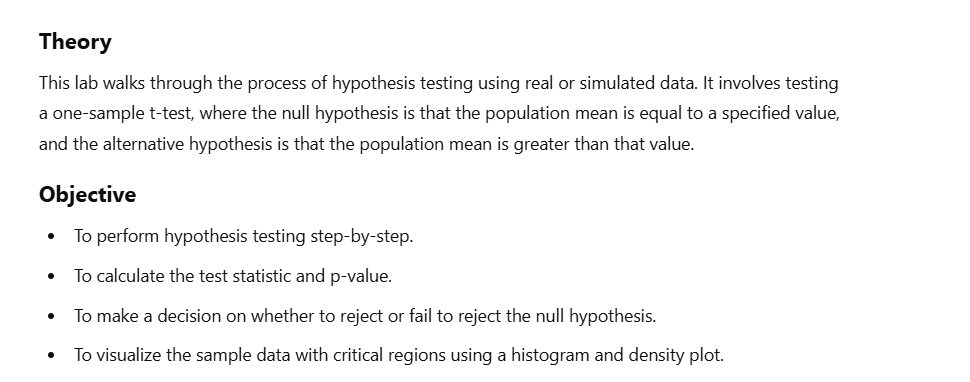
**Output:**

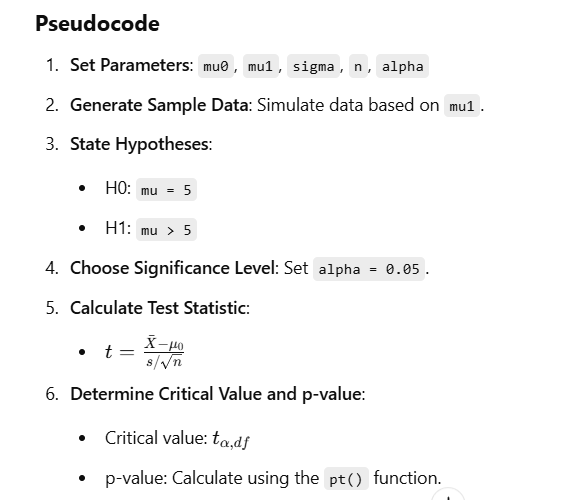
Type I Error Rate: 0.039

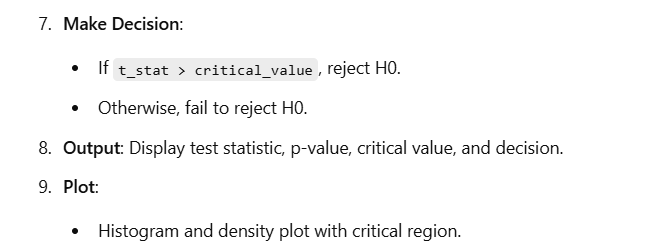
Type II Error Rate: 0.154

Power: 0.846

*# Lab 14: Perform Hypothesis Testing Step-by-Step Using Real or Simulated Data*







R code:

mu0 <- 5 *# Null hypothesis mean*

mu1 <- 6 *# True population mean*

sigma <- 2 *# Population standard deviation*

n <- 30 *# Sample size*

alpha <- 0.05 *# Significance level*

*# Generate sample data*

set.seed(123)

sample\_data <- rnorm(n, *mean* = mu1, *sd* = sigma)

*# Step 1: State hypotheses*

print("H0: mu = 5")

print("H1: mu > 5")

*# Step 2: Choose significance level*

print(paste("Significance level (alpha):", alpha))

*# Step 3: Calculate test statistic*

t\_stat <- (mean(sample\_data) - mu0) / (sd(sample\_data) / sqrt(n))

print(paste("Test Statistic (t):", t\_stat))

*# Step 4: Determine critical value or p-value*

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))

*# Step 5: Make a decision*

*if* (t\_stat > critical\_value) {

 decision <- "Reject H0"

} *else* {

 decision <- "Fail to reject H0"

}

print(paste("Decision:", decision))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram with critical region*

hist(sample\_data, *breaks* = 30, *col* = "lightblue", *main* = "Sample Data", *xlab* = "Value", *border* = "white")

abline(*v* = mu0, *col* = "red", *lwd* = 2)

abline(*v* = mean(sample\_data), *col* = "blue", *lwd* = 2)

*# Density plot with critical region*

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 # Null hypothesis mean

mu1 <- 6 # True population mean

sigma <- 2 # Population standard deviation

n <- 30 # Sample size

alpha <- 0.05 # Significance level

output:

H0: mu = 5

H1: mu > 5

Significance level (alpha): 0.05

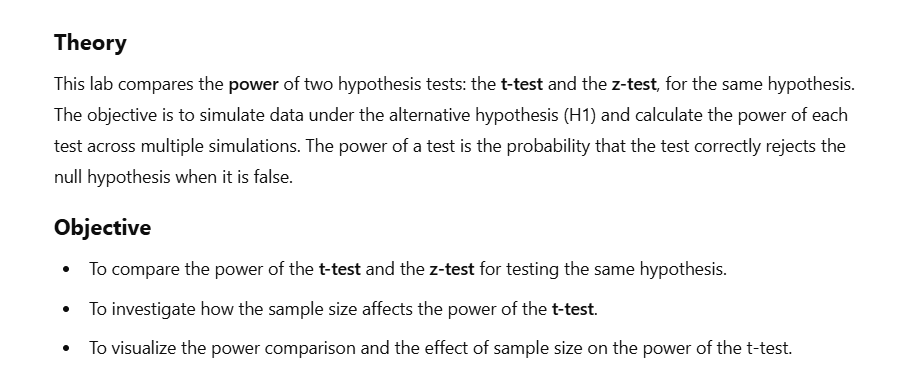
Test Statistic (t): 2.52858027419059

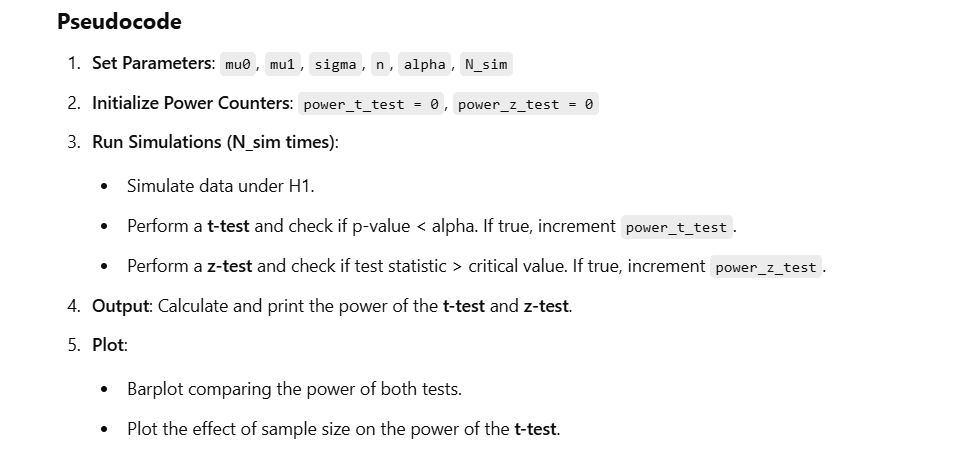
Critical Value: 1.6991270265335

P-value: 0.00857566021520242

Decision: Reject H0

*# Lab 15: Compare the Power of Different Tests for the Same Hypothesis*





R code:

mu0 <- 5 *# Null hypothesis mean*

mu1 <- 6 *# Alternative hypothesis mean*

sigma <- 2 *# Population standard deviation*

n <- 30 *# Sample size*

alpha <- 0.05 *# Significance level*

N\_sim <- 1000 *# Number of simulations*

*# Initialize power counters*

power\_t\_test <- 0

power\_z\_test <- 0

*# Simulation*

set.seed(123)

*for* (i *in* 1:N\_sim) {

*# Simulate data under H1*

 data <- rnorm(n, *mean* = mu1, *sd* = sigma)

*# Perform t-test*

 t\_test <- t.test(data, *mu* = mu0, *alternative* = "greater")

*if* (t\_test$p.value < alpha) {

 power\_t\_test <- power\_t\_test + 1

 }

*# Perform z-test*

 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

 }

}

*# Output*

print(paste("Power of t-test:", power\_t\_test / N\_sim))

print(paste("Power of z-test:", power\_z\_test / N\_sim))

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Power comparison*

barplot(c(power\_t\_test / N\_sim, power\_z\_test / N\_sim), *names.arg* = c("t-test", "z-test"), *col* = c("lightblue", "lightgreen"), *main* = "Power Comparison", *ylab* = "Power")

*# Effect of sample size on 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 Size", *xlab* = "Sample")

**sample input/output:**

**input:**

mu0 <- 5 # Null hypothesis mean

mu1 <- 6 # Alternative hypothesis mean

sigma <- 2 # Population standard deviation

n <- 30 # Sample size

alpha <- 0.05 # Significance level

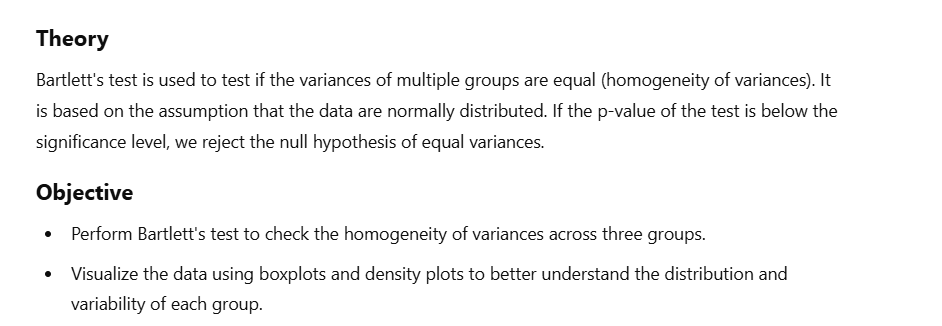
N\_sim <- 1000 # Number of simulations

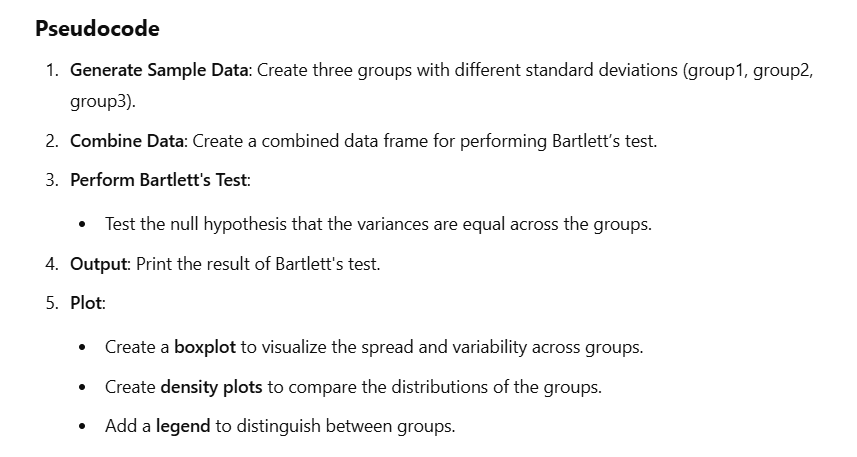
**Output:**

Power of t-test: 0.844

Power of z-test: 0.854

*Lab 16. Apply Bartlett’s test to compare variances across multiple 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)

*# Combine data into a data frame for Bartlett's test*

data\_values <- c(group1, group2, group3)

group\_labels <- rep(c("Group 1", "Group 2", "Group 3"), *each* = 30)

*# Perform Bartlett's test*

bartlett\_test\_result <- bartlett.test(data\_values ~ group\_labels)

*# Output test result*

print(bartlett\_test\_result)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Boxplot of groups*

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")

*# Density plots of groups*

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)

*# Add legend*

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\_labels

Bartlett's K-squared = 9.4313, df = 2, p-value = 0.008954

*# Lab 17: Perform Fisher's Exact Test on 2×2 Contingency Tables*

**Theory**

Fisher's Exact Test is used to determine if there are nonrandom associations between two categorical variables in a **2x2 contingency 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 contingency 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. **Create a 2x2 Contingency Table** with values for two variables (e.g., Group A vs Group B, Success vs Failure).
2. **Perform Fisher's Exact Test** on the contingency table to test for associations.
3. **Output**:
   * Print the contingency table and Fisher's test result.
4. **Graphical Output**:
   * Create a **barplot** to visualize the counts for each outcome.
   * Create a **mosaic plot** to visualize the relationship between the two categorical variables.

R code:

*# Create a 2x2 contingency table*

data <- matrix(c(10, 5, 2, 8), *nrow* = 2, *byrow* = TRUE)

rownames(data) <- c("Group A", "Group B")

colnames(data) <- c("Success", "Failure")

*# Perform Fisher's exact test*

fisher\_test\_result <- fisher.test(data)

*# Output*

print(data)

print(fisher\_test\_result)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Barplot of the contingency table*

barplot(data, *beside* = TRUE, *col* = c("lightblue", "lightgreen"), *main* = "2x2 Contingency Table", *xlab* = "Outcome", *ylab* = "Count")

legend("topright", *legend* = rownames(data), *fill* = c("lightblue", "lightgreen"))

*# Mosaic plot*

mosaicplot(data, *main* = "Mosaic Plot", *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

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

**Theory**

A **three-way contingency table** is used to examine the relationship between three categorical variables. A **log-linear model** is often used to assess 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**

* Create a **three-way contingency table** with categorical variables (Gender, Treatment, Outcome).
* Fit a **log-linear model** to assess interactions between these variables.
* Visualize the data using a **heatmap** and an **interaction plot** to understand relationships between the variables.

**Pseudocode**

1. **Create a 3D contingency table** with dimensions for Gender, Treatment, and Outcome.
2. **Fit a log-linear model** to the table to identify the relationships between the variables.
3. **Output**:
   * Print the three-way contingency table and the results of the log-linear model.
4. **Graphical Output**:
   * Create a **heatmap** to visualize the joint distribution of Gender and Treatment, summing over Outcome.
   * Create an **interaction plot** to visualize the counts of outcomes by Treatment and Gender.

R code:

*# Create a three-way contingency table*

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"))

*# Fit a log-linear model*

log\_linear\_model <- loglin(data, *margin* = list(1, 2, 3), *fit* = TRUE)

*# Output*

print("Three-Way Contingency Table:")

print(data)

print("Log-Linear Model Output:")

print(log\_linear\_model)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Convert to 2D matrix for heatmap*

heatmap\_matrix <- apply(data, c(1, 2), sum)  *# Summing over Outcome*

heatmap(heatmap\_matrix, *main* = "Heatmap of Gender vs Treatment", *col* = heat.colors(100))

*# Convert data into vectors for interaction plot*

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)

*# Create an interaction plot*

interaction.plot(Treatment, Gender, Counts,

*main* = "Interaction Plot: Treatment & 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 Contingency Table:

, , Outcome = Success

Treatment

Gender Yes No

Male 10 5

Female 2 8

, , Outcome = Failure

Treatment

Gender Yes No

Male 3 6

Female 4 7

Lab 19: Conduct Non-Parametric Tests

**Theory**

Non-parametric tests are statistical methods that do not assume a specific distribution for the data. The **Wilcoxon rank-sum test** (also known as the **Mann-Whitney U test**) is a non-parametric 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 **Wilcoxon rank-sum test** to compare the distributions of the two groups.
* Visualize the data using **boxplots** and **density plots**.

**Pseudocode**

1. **Generate random sample data** for two independent groups.
2. **Perform Wilcoxon rank-sum test** to check if there is a significant difference between 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:

*# Generate sample data*

set.seed(123)

group1 <- rnorm(20, *mean* = 5, *sd* = 2)

group2 <- rnorm(20, *mean* = 7, *sd* = 2)

*# Perform Wilcoxon rank-sum test*

wilcox\_test\_result <- wilcox.test(group1, group2)

*# Output*

print(wilcox\_test\_result)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Boxplot of groups*

boxplot(list(group1, group2), *col* = c("lightblue", "lightgreen"), *main* = "Boxplot of Groups", *xlab* = "Group", *ylab* = "Value")

*# Density plots of groups*

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: group1 and group2**

**W = 104, p-value = 0.008712**

**alternative hypothesis: true location shift is not equal to 0**

*# Lab 20: Perform z-Tests for Large Sample Sizes*

**Theory**

A **z-test** is a statistical test used to compare the means of two large samples (typically n>30n > 30n>30) 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 two groups.
* Visualize the data distribution using histograms.

**Pseudocode**

1. **Import necessary library** (BSDA for z-test).
2. **Generate random sample data** for two groups.
3. **Perform z-test**:
   * Compare the means of the two groups.
   * Use the **standard deviation** of each group.
4. **Output test results** (z-statistic and p-value).
5. **Graphical Output**:
   * **Histogram** for Group 1.
   * **Histogram** for Group 2.

R code:

*# Generate sample data*

library(BSDA)

set.seed(123)

group1 <- rnorm(100, *mean* = 5, *sd* = 2)

group2 <- rnorm(100, *mean* = 6, *sd* = 2)

*# Perform z-test*

z\_test\_result <- z.test(group1, group2, *sigma.x* = sd(group1), *sigma.y* = sd(group2))

*# Output*

print(z\_test\_result)

*# Graphical Output*

par(*mfrow* = c(1, 2))

*# Histogram of groups*

hist(group1, *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**

group1 <- rnorm(100, mean = 5, sd = 2)

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

**Sample Output:  
 Two-sample z-Test**

**data: group1 and group2**

**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**