Lab 1: Verification of Fisher's Lemma Using Simulated Data from Normal Distributions

```
# Parameters
mu <- 5
sigma <- 2
n <- 30
N_sim <- 1000

# Initialize vectors
sample_means <- numeric(N_sim)
sample_vars <- numeric(N_sim)

# 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))

# Graphical Output
par(mfrow = c(2, 2))

# Histogram of sample means
hist(sample_means, breaks = 30, col = "lightblue", main = "Distribution of Sample Means", xlab = "S
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 =
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 '
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)
```

Lab 2: Generation and Analysis of χ2-Distributed Data

```
    # Parameters
k <- 5
N_sim <- 1000

# Generate chi-squared data
set.seed(123)
chi2_data <- rchisq(N_sim, df = k)
```

```
# Compute mean and variance
mean_chi2 <- mean(chi2_data)
var_chi2 <- var(chi2_data)
print(paste("Mean:", mean_chi2))
print(paste("Variance:", var_chi2))

# Graphical Output
par(mfrow = c(1, 3))

# Histogram
hist(chi2_data, breaks = 30, col = "lightblue", probability = TRUE, main = "Chi-Squared Distribution
curve(dchisq(x, df = k), add = TRUE, col = "red", lwd = 2)

# Density plot
plot(density(chi2_data), col = "blue", lwd = 2, main = "Density Plot", xlab = "Value")
curve(dchisq(x, df = k), add = TRUE, col = "red", lwd = 2)

# Q-Q plot
qqplot(qchisq(ppoints(N_sim), df = k), chi2_data, main = "Q-Q Plot for Chi-Squared Data", col = "blu
abline(0, 1, col = "red", lwd = 2)
```

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

```
# Parameters
n <- 10
N_sim <- 1000

# Generate data
set.seed(123)
t_data <- rt(N_sim, df = n-1)
normal_data <- rnorm(N_sim)

# Graphical Output
par(mfrow = c(2, 2))

# Histogram of t-distribution
hist(t_data, breaks = 30, col = "lightblue", probability = TRUE, main = "t-Distribution", xlab = "Value
curve(dt(x, df = n-1), add = TRUE, col = "red", lwd = 2)

# Histogram of normal distribution
hist(normal_data, breaks = 30, col = "lightgreen", probability = TRUE, main = "Normal Distribution'
curve(dnorm(x), add = TRUE, col = "blue", lwd = 2)

# Density plot comparison
plot(density(t_data), col = "red", lwd = 2, main = "Density Comparison", xlab = "Value", ylim = c(0, 0
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 = '
abline(0, 1, col = "blue", lwd = 2)
```

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

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

# Generate F-distributed data
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)

# Graphical Output
par(mfrow = c(2, 2))

# Histogram
hist(f_data, breaks = 30, col = "lightblue", probability = TRUE, main = "F-Distribution", xlab = "Value
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 = "b
abline(0, 1, col = "red", lwd = 2)

# Boxplot
boxplot(f_data, col = "lightgreen", main = "Boxplot of F-Distributed Data", ylab = "Value")
```

Lab 5: Distribution of Medians and Ranges from Sampled Populations

```
# 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", x
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", xla

# 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 = "Boxplo
```

Lab 6: Estimate Population Parameters (Mean, Variance) from Sample Data

```
# 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", bo
abline(v = sample_mean, col = "red", lwd = 2)
abline(v = conf_int_mean, col = "blue", lty = 2)

# Boxplot
```

```
boxplot(sample_data, col = "lightgreen", main = "Boxplot of Sample Data", ylab = "Value")
```

Lab 7: Demonstrate Consistency by Increasing Sample Size

```
# Parameters
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 = "Sa
abline(h = mu, col = "red", lwd = 2)

# Plot sample variances
plot(sample_sizes, vars, type = "b", col = "green", main = "Convergence of Sample Variance", x
abline(h = sigma^2, col = "red", lwd = 2)
```

Lab 8: Compare Biased and Unbiased Estimators

```
# 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)
  biased_vars[i] <- sum((data - mean(data))^2) / n
```

```
}

# Graphical Output
par(mfrow = c(1, 2))

# Histogram of unbiased variances
hist(sample_vars, breaks = 30, col = "lightblue", main = "Unbiased Sample Variance", xlab = "Varian
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
abline(v = sigma^2, col = "red", lwd = 2)
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 =
hist(group2, breaks = 30, col = "lightgreen", main = "Histogram of Group 2", xlab = "Value", border =
```

Lab 9: Calculate Efficiency of Estimators

```
# 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", b

# Histogram of medians
hist(medians, breaks = 30, col = "lightgreen", main = "Distribution of Sample Medians", xlab = "Valu
```

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))
```

Lab 11: Simulate Decision-Making Processes Using Hypothesis Testing

```
# Parameters
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 = "wh
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)
```

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

```
# Parameters
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)
```

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

```
# Parameters
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")$
    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")$
    breaks = 30, col = "lightgreen", main = "P-values under H1", xlab = "P-value", border = "white")
abline(v = alpha, col = "red", lwd = 2)
```

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

```
# Parameters
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 = "wh
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)
```

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

```
# Parameters
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("ligl

# 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")
})
plot(sample_sizes, power_t, type = "b", col = "blue", main = "Power vs. Sample Size", xlab = "Sample
```

Lab 16: Apply Bartlett's Test to Compare Variances Across Multiple Groups

```
# Generate sample data for three groups
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 list
data_list <- list(group1, group2, group3)

# Perform Bartlett's test
bartlett_test_result <- bartlett.test(data_list)

# Output
print(bartlett_test_result)

# Graphical Output
par(mfrow = c(1, 2))

# Boxplot of groups
boxplot(data_list, col = c("lightblue", "lightgreen", "lightcoral"), main = "Boxplot of Groups", xlab = '

# 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)
legend("topright", legend = c("Group 1", "Group 2", "Group 3"), col = c("blue", "green", "red"), lwd =
```

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

```
# 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", xlal
legend("topright", legend = rownames(data), fill = c("lightblue", "lightgreen"))

# Mosaic plot
mosaicplot(data, main = "Mosaic Plot", color = TRUE)
```

Lab 18: Analyze Three-Way Contingency Tables Using Log-Linear Models

```
# 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("Su

# Fit a log-linear model
log_linear_model <- loglin(data, margin = list(c(1, 2, 3)), fit = TRUE, param = TRUE)

# Output
print(data)
print(log_linear_model)

# Graphical Output
par(mfrow = c(1, 2))

# Heatmap of the three-way table
heatmap(data, main = "Heatmap of Three-Way Table", col = heat.colors(100))

# Interaction plot
interaction.plot(data[, , 1], data[, , 2], data[, , 3], main = "Interaction Plot", xlab = "Gender", ylab = "C
```

Lab 19: Conduct Non-Parametric Tests

```
# 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 = "

# 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)
```

Lab 20: Perform z-Tests for Large Sample Sizes

```
# Generate sample data
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 =
hist(group2, breaks = 30, col = "lightgreen", main = "Histogram of Group 2", xlab = "Value", border =
```