ST551: HOMEWORK 6

Brian Cervantes Alvarez
December 8, 2023

Question 1

Part A

```r
tableA <- matrix(c(0.02, 0.04, 0.06, 0.08,
                   0.03, 0.06, 0.09, 0.12,
                   0.05, 0.10, 0.15, 0.20),
                 nrow = 3,
                 byrow = TRUE)

# Perform Chi-squared test
chi2Result <- chisq.test(tableA, correct = FALSE)

# Results
pval <- chi2Result$p.value
print(paste0("p-val: ",pval, " Conclusion: ",
             ifelse(pval<0.5, "Reject Null", "Fail to reject Null")))
```

```
[1] "p-val: 1 Conclusion: Fail to reject Null"
```

In terms of performance, both tests are about the same. You can argue that Fisher's does the best at smaller sample sizes and Pearson with higher sample sizes.

```r
performTests <- function(sampleSize, table) {
  rejChi2 <-  numeric(0)
  rejFisher <- numeric(0)

  for (i in 1:10000) {
    # Simulate data
    simDatasets <- matrix(rmultinom(1, sampleSize, table), nrow = 3)

    # Pearson's Chi-squared test
    chi2Result <- chisq.test(simDatasets, correct = FALSE)
    rejChi2[i] <- chi2Result$p.value

    # Fisher's Exact test
    fisherResult <- fisher.test(simDatasets, simulate.p.value = TRUE)
    rejFisher[i] <- fisherResult$p.value
    }

  # Calculate rejection rates
  rejChi2 <- mean(rejChi2 < 0.05, na.rm = TRUE)
  rejFisher <- mean(rejFisher < 0.05, na.rm = TRUE)

  return(c(rejChi2,rejFisher))
}

# Sample sizes
sampleSizes <- c(50, 100, 500, 1000, 5000)

# Perform tests for each sample size
results <- t(sapply(sampleSizes, function(n) performTests(n, tableA)))

# Display results
colnames(results) <- c("Pearson's Rejection Rate", "Fisher's Rejection Rate")
rownames(results) <- paste("n =", sampleSizes)
print(results)
```

```
         Pearson's Rejection Rate Fisher's Rejection Rate
n = 50                 0.04540888                  0.0485
n = 100                0.04670000                  0.0461
n = 500                0.05010000                  0.0497
n = 1000               0.05100000                  0.0530
n = 5000               0.04810000                  0.0488
```

Part B

```r
tableB <- matrix(c(0.02, 0.04, 0.06, 0.08,
                   0.03, 0.06, 0.09, 0.12,
                   0.05, 0.10, 0.15, 0.20),
                 nrow = 3,
                 byrow = TRUE)

# Perform Chi-squared test on table B
chi2Result <- chisq.test(tableB, correct = FALSE)
# Results
pval <- chi2Result$p.value
print(paste0("p-val: ",pval, " Conclusion: ",
             ifelse(pval<0.5, "Reject Null", "Fail to reject Null")))
```

```
[1] "p-val: 1 Conclusion: Fail to reject Null"
```

In terms of performance, both tests are about the same. You can argue that Fisher's does the best at smaller sample sizes and Pearson with higher sample sizes.

```r
performTests <- function(sampleSize, table) {
  rejChi2 <-  numeric(0)
  rejFisher <- numeric(0)

  for (i in 1:10000) {
    # Simulate data
    simDatasets <- matrix(rmultinom(1, sampleSize, table), nrow = 3)

    # Pearson's Chi-squared test
    chi2Result <- chisq.test(simDatasets, correct = FALSE)
    rejChi2[i] <- chi2Result$p.value

    # Fisher's Exact test
    fisherResult <- fisher.test(simDatasets, simulate.p.value = TRUE)
    rejFisher[i] <- fisherResult$p.value
    }

  # Calculate rejection rates
  rejChi2 <- mean(rejChi2 < 0.05, na.rm = TRUE)
  rejFisher <- mean(rejFisher < 0.05, na.rm = TRUE)

  return(c(rejChi2,rejFisher))
}
```

```r
# Sample sizes
sampleSizes <- c(50, 100, 500, 1000, 5000)

# Perform tests for each sample size
results <- t(sapply(sampleSizes, function(n) performTests(n, tableA)))

# Display results
colnames(results) <- c("Pearson's Rejection Rate", "Fisher's Rejection Rate")
rownames(results) <- paste("n =", sampleSizes)
print(results)
```

```
         Pearson's Rejection Rate Fisher's Rejection Rate
n = 50                 0.04561897                  0.0509
n = 100                0.04890000                  0.0507
n = 500                0.05300000                  0.0523
n = 1000               0.05120000                  0.0518
n = 5000               0.04580000                  0.0467
```

Question 2

Part A

The Wilcoxon Rank-sum Test isn't a precise test for comparing medians over large sample sizes. I illustrate this using distributions F and G with identical medians. As a result, the test consistently rejects the null hypothesis, indicating a mismatch. My code reveals that this rejection rate increases with larger sample sizes. Reducing the sample size doesn't make the test suitable for distributions with smaller samples.

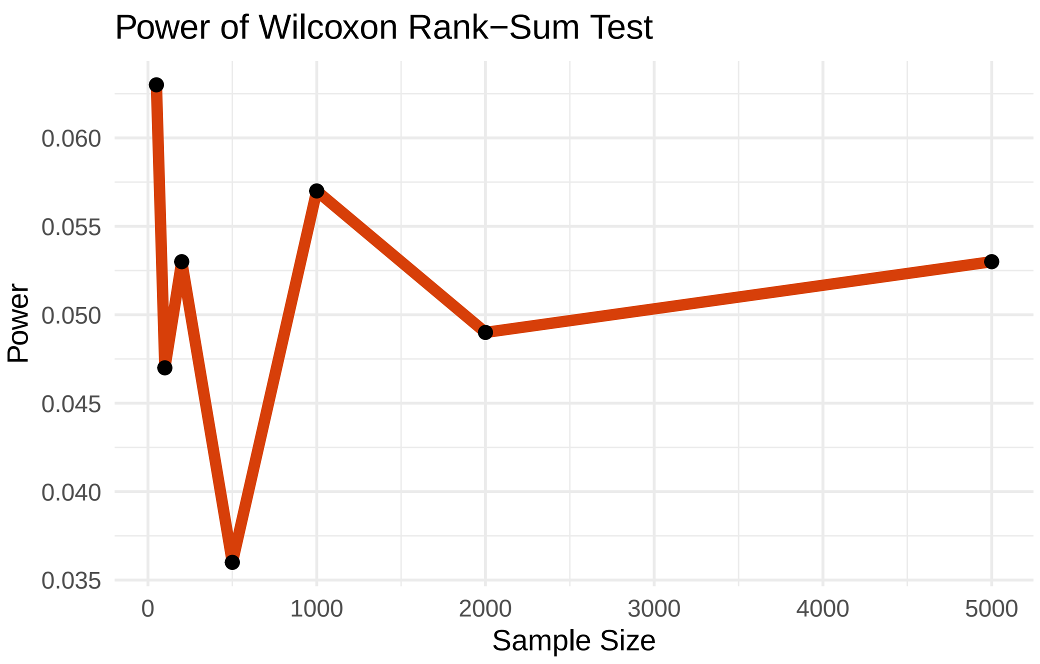
```
# Different sample sizes
sampSizes <- c(50, 100, 500, 1000)

for (n in sampSizes) {
  rejCount <- 0

  # Simulate the test for same median distributions
  for (i in 1:1000) {
    X <- rnorm(n, mean = 1)
    Y <- rexp(n, rate = 1)

    result <- wilcox.test(X, Y, alternative = "two.sided")

    if (result$p.value < 0.05) {
      rejCount <- rejCount + 1
    }
  }

  # Calculate rejection rate
  rejRate <- rejCount / 1000
  print(paste0("Rejection rate for equal medians (n = ", n, "): ",
               rejRate*100, "%"))
}
```

```
[1] "Rejection rate for equal medians (n = 50): 9.2%"
[1] "Rejection rate for equal medians (n = 100): 16.6%"
[1] "Rejection rate for equal medians (n = 500): 53.6%"
[1] "Rejection rate for equal medians (n = 1000): 84%"
```

Part B

The Rank-Sum test lacks consistency for the null hypothesis $H_0 : F = G$, meaning its power doesn't consistently approach 1 as the sample size increases. I showed the result in the graph below.

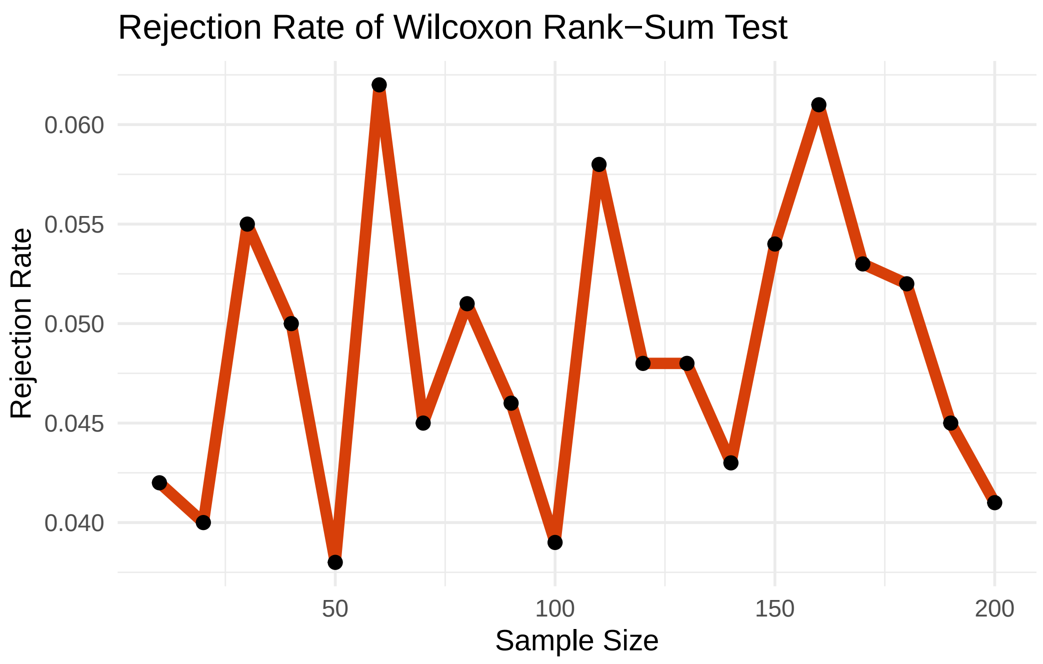
```r
library(ggplot2)

wilcoxonPowerSimulation <- function(sampleSize, numSimulations = 1000) {
  power <- numeric(numSimulations)

  for (i in 1:numSimulations) {
    # Generate samples from two different distributions
    X <- rnorm(sampleSize)
    Y <- rt(sampleSize, df = 2)

    # Perform Test
    result <- wilcox.test(X, Y, alternative = "two.sided")

    # Store the power of the test
    power[i] <- result$p.value < 0.05
  }

  # Calculate power
  return(mean(power))
}

# Simulate power for different sample sizes
sampleSizes <- c(50, 100, 200, 500, 1000, 2000, 5000)
powerResults <- sapply(sampleSizes, wilcoxonPowerSimulation)

ds <- data.frame(sampleSize = sampleSizes, power = powerResults)
ggplot(ds, aes(x = sampleSize, y = power)) +
  geom_line(color = "#D73F09", size = 2) +
  geom_point(color = "black", size = 2) +
  labs(x = "Sample Size", y = "Power",
       title = "Power of Wilcoxon Rank-Sum Test") +
  theme_minimal()
```


Power of Wilcoxon Rank−Sum Test
Power
0.060
0.055
0.050
0.045
0.040
0.035
0
1000
2000
3000
4000
5000
Sample Size

Part C

The rejection rates vary across different sample sizes, demonstrating that the test may not have sufficient power to consistently detect the true difference in this scenario. This behavior is shows that the Wilcoxon Rank-sum Test is not guaranteed to be asymptotically exact, especially under specific conditions or when sample sizes are not large enough.

```r
set.seed(42)

# Set up parameters
numSimulations <- 1000
sampleSizes <- seq(10, 200, by = 10)

# Perform simulations
rejRates <- numeric(length(sampleSizes))

for (i in seq_along(sampleSizes)) {
  nX <- sampleSizes[i]
  nY <- round(nX / 2)

  rejectCount <- 0

  for (j in 1:numSimulations) {
    X <- rexp(nX, rate = 1)
    Y <- rexp(nY, rate = 1)

    # Perform Test
    result <- wilcox.test(X, Y, alternative = "two.sided")
    if (result$p.value < 0.05) {
      rejectCount <- rejectCount + 1
    }
  }

  # Calculate rejection rate
  rejRates[i] <- rejectCount / numSimulations
}

ds <- data.frame(sampleSize = sampleSizes, rejectionRate = rejRates)
ggplot(ds, aes(x = sampleSize, y = rejectionRate)) +
  geom_line(color = "#D73F09", size = 2) +
  geom_point(color = "black", size = 2) +
  labs(x = "Sample Size", y = "Rejection Rate",
       title = "Rejection Rate of Wilcoxon Rank-Sum Test") +
  theme_minimal()
```



Rejection Rate of Wilcoxon Rank−Sum Test
Rejection Rate
0.060
0.055
0.050
0.045
0.040
50
100
150
200
Sample Size

Part D

```r
# Params
n <- 100000
const <- 1
X <- rnorm(n, sd = 4)
Y <- rchisq(n, df = 3) + const
Z <- -rchisq(n, df = 3) + const

# Calculate probs
probXY <- mean(X > Y)
probYZ <- mean(Y > Z)
probZX <- mean(Z > X)

print(paste("P(X > Y):", probXY))
```

```
[1] "P(X > Y): 0.19509"
```

```r
print(paste("P(Y > Z):", probYZ))
```

```
[1] "P(Y > Z): 1"
```

```r
print(paste("P(Z > X):", probZX))
```

```
[1] "P(Z > X): 0.34321"
```

Question 3

Part A

Test stat => $Z = 0.54002$. What does this mean? Given the p-value of 0.5892 and significance level of 0.05, there is not enough evidence to reject the null hypothesis. Despite having test statistic of 0.54, which measures the difference in variances, we cannot go with the alternative given the high pval.

```r
# Diet Data
mediterranean <- c(-6.7, -1.9, -0.6,
                   0.3, 0.9, 1.0, 2.4,
                   4.7, 7.8, 9.7, 10.3, 12.0)
lowCarb <- c(-5.6, 0.3, 2.0, 2.0, 4.6, 5.0, 6.5, 7.5, 11.8, 17.0)

# Perform Mood's test
result <- mood.test(mediterranean, lowCarb)
print(result)
```

```
    Mood two-sample test of scale

data:  mediterranean and lowCarb
Z = 0.54002, p-value = 0.5892
alternative hypothesis: two.sided
```

Part B

In this case, there is no evidence to reject the null hypothesis where there is a difference in diets, where one is greater than the other. In other words, the shift is not greater than 0 in this case.

```r
# Perform Wilcoxon Signed-Rank test
result <- wilcox.test(mediterranean,
                      lowCarb,
                      alternative = "greater")
print(result)
```

```

    Wilcoxon rank sum test with continuity correction

data:  mediterranean and lowCarb
W = 49.5, p-value = 0.766
alternative hypothesis: true location shift is greater than 0
```

Part C

Again, we fail to reject the null hypothesis given the p-val of 0.9686 in my case.

```r
permTStat <- function(x, y) {
  return(abs(mean(x) - mean(y)))
}

obsStat <- permTStat(mediterranean, lowCarb)

nPerm <- 10000

results <- replicate(nPerm, {
  samp <- sample(c(mediterranean, lowCarb))
  permTStat(samp[1:length(mediterranean)],
                samp[(length(mediterranean)+1):(length(mediterranean)
                                                +length(lowCarb))])
})

pvalPerm <- mean(results >= obsStat) * 2

print(paste0("p-val = ", pvalPerm))
```

```
[1] "p-val = 0.9834"
```

Part D

We fail to reject the null hypothesis, indicating that there is no significant evidence to suggest a difference in variances between the two groups. Again, very similar results compared to the other 3 tests.

```
library(car)
# Perform Levene's test
result <- leveneTest(c(mediterranean, lowCarb),
                            group = rep(c("Mediterranean", "LowCarb"),
                                        times = c(length(mediterranean),
                                                  length(lowCarb))))
print(result)
```

```
Levene's Test for Homogeneity of Variance (center = median)
      Df  F value Pr(>F)
group  1   6e-04   0.98
      20
```