Stat361-HW3

Mert Göksel, 2429066 Bilge Özkır, 2218287

Q1

This experiment is of binomial distribution.

```
n <- 1000

total_ns <- c(6,12,18)
total_successes <- c(1,2,3)

for(j in 1:3){
  k <- 0 # counter for success
  for(i in 1:n){
    x <- sample(1:6, size = total_ns[j], replace=TRUE)
    num_six <- sum(x==6)

    if(num_six >= total_successes[j]){
    k <- k + 1
    }
  }
  est_prob <- k/n
  cat("When size is", total_ns[j],
      "and wanted success is",total_successes[j],
      ", estimated probability is",est_prob,"\n")
}
```

```
## When size is 6 and wanted success is 1 , estimated probability is 0.666
## When size is 12 and wanted success is 2 , estimated probability is 0.644
## When size is 18 and wanted success is 3 , estimated probability is 0.588
```

Q2

```
n <- 35; xbar <- 7.91; std <- sqrt(0.03); mu0 <- 8
N <- 10**5; a <- 0.01/2 #as two sided test
```

Part A

- H_0: $\mu = 8$
- H_1: μ != 8

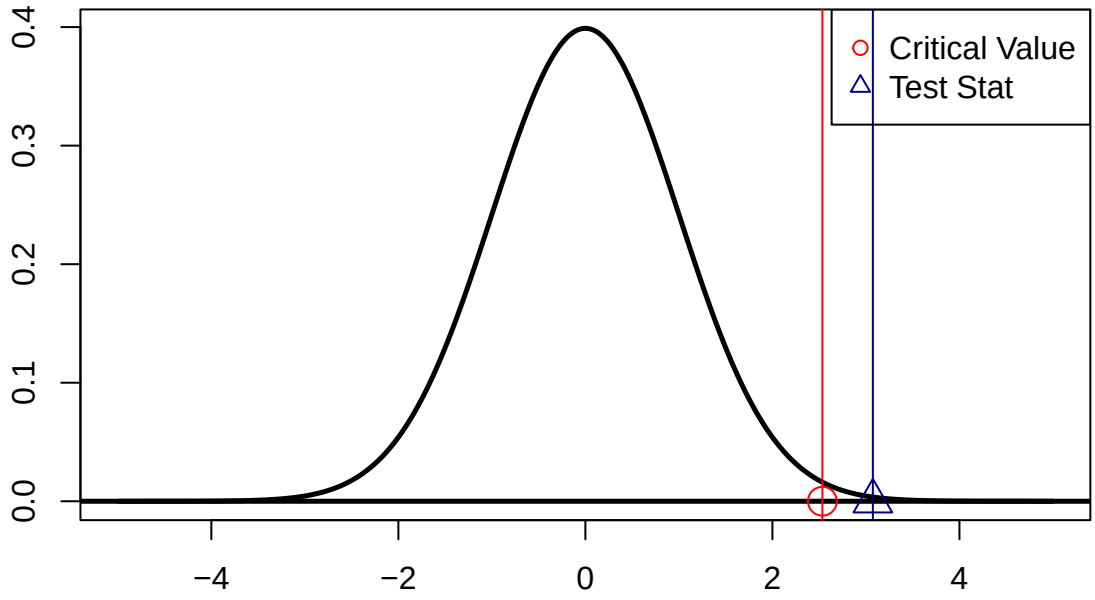
```
crit <- (xbar - 8)/(std/sqrt(n)) #-3.07
qnorm(0.025, lower.tail = 1) # -1.96
```

```
## [1] -1.959964
```

As z_{crit} is more extreme than $z_{tabulated}$ we will reject H_0 and say that the wanted average is not attained. But now lets use montecarlo and see for ourselves.

```
cv <- function(n,xbar, Mu0, sigma, M, alpha){

  S.Error <- (sigma / sqrt(n))
  test.stat <- abs((xbar - Mu0) / S.Error) #as this code only works for upper tail.

  TestScores <- numeric(M)

  for(i in 1:M){
    x <- rnorm(n, Mu0, sigma) # generate random sample under H0
    TestScores[i] <- (mean(x) - Mu0) / S.Error
  }
  # Get the critical value for alpha
  Critical.Value <- abs(quantile(TestScores,1-alpha))

  if(test.stat > Critical.Value){
    cat("From MC Simulation we get an estimated Critical Value of",
        round(Critical.Value,3),"\n", "Test Statistic is",
        round(test.stat,3),"\n","\n",
        "Therefore, we have enough evidence to reject the null hypothesis",
        "\n","\n")
  }else{
    cat("From MC Simulation we get an estimated Critical Value of",
        round(Critical.Value,3),"\n", "Test Statistic is",
        round(test.stat,3),"\n",
        "Therefore, we do not have enough evidence to reject the null hypothesis",
        "\n","\n")
  }
  a <- seq(-5,5,0.01)
  plot(a, dnorm(a), type = "l",lwd = 2.5, ylab = "", xlab = "")
  abline(h=0, lwd = 2.5)
  abline(v = Critical.Value, col = "Red")
  abline(v = test.stat, col = "Dark Blue")
  points(x = Critical.Value, y = 0, pch = 1, cex = 2, col = "Red")
  points(x = test.stat, y = 0, pch = 2, cex = 2, col = "Dark Blue")
  legend("topright",legend = c("Critical Value","Test Stat"),
         col = c("Red","Dark Blue"), pch = 1:2)

  return(list(test_stat = test.stat,
              Critical_Value = as.numeric(Critical.Value)))
}
cv(n, xbar, mu0, std, M = N, alpha = a)
```

```
## From MC Simulation we get an estimated Critical Value of 2.535
##  Test Statistic is 3.074
##
##  Therefore, we have enough evidence to reject the null hypothesis
##
```




```
## $test_stat
## [1] 3.074085
##
## $Critical_Value
## [1] 2.534732
```

Part B

```
pvalue <- function(n, xbar, Mu0, sigma, M, alpha){

  S.Error <- (sigma / sqrt(n))
  test.stat <- abs((xbar - Mu0) / S.Error)

  TestScores <- numeric(M)

  for(i in 1:M){
    x <- rnorm(n, Mu0, sigma) # generate random sample under H0
    TestScores[i] <- (mean(x) - Mu0) / S.Error
  }

  pvalue <- length(which(TestScores >= test.stat))/M

  if(pvalue < alpha){
    cat("Since P-value",pvalue,"which is less than the significance level",
        alpha, "\n",
        "Therefore, we can reject the null hypothesis.","\n","\n")
```

```
  }else{
    cat("Since P-value",pvalue,
        "which is greater than the significance level",
        alpha, "\n",
        "Therefore, we can not reject the null hypothesis.",
        "\n","\n")
  }
  return(list(P_Value = pvalue)) #return to p-value
}


pvalue(n=n, xbar=xbar, Mu0=mu0, sigma=std, M=N, alpha=a)
```

```
## Since P-value 0.00138 which is less than the significance level 0.005
##  Therefore, we can reject the null hypothesis.
##
```

```
## $P_Value
## [1] 0.00138
```

```
pnorm((xbar-mu0)/(std/sqrt(n)))
```

```
## [1] 0.001055746
```

Thus same result is achieved.

Part C

```
CL <- function(n, M, alpha, mu, sigma){

  # we are assuming that the population has a normal shape

  matrix <- matrix(0, nrow = M, ncol = 2, dimnames = list(c(1:M), c("Lower","Upper")))

  for(i in 1:M){
    x <- rnorm(n,mu,sigma)
    x_bar <- mean(x)
    x_sd <- sd(x)
    matrix[i,1] <- x_bar - qnorm(1-alpha/2) * (x_sd/sqrt(n))
    matrix[i,2] <- x_bar + qnorm(1-alpha/2) * (x_sd/sqrt(n))
    }
  out <- list(CI = apply(matrix,2,mean), conf_level = 1-alpha) #mean of rows
  return(out)
}

CL(n=n, M=N, alpha=0.05, mu = 8, sigma = std)
```

```
## $CI
##    Lower    Upper
## 7.943195 8.057059
##
## $conf_level
## [1] 0.95
```

Q3

```
set.seed(10403)
ss <- c(10, 50, 100, 500)
m <- 500
```

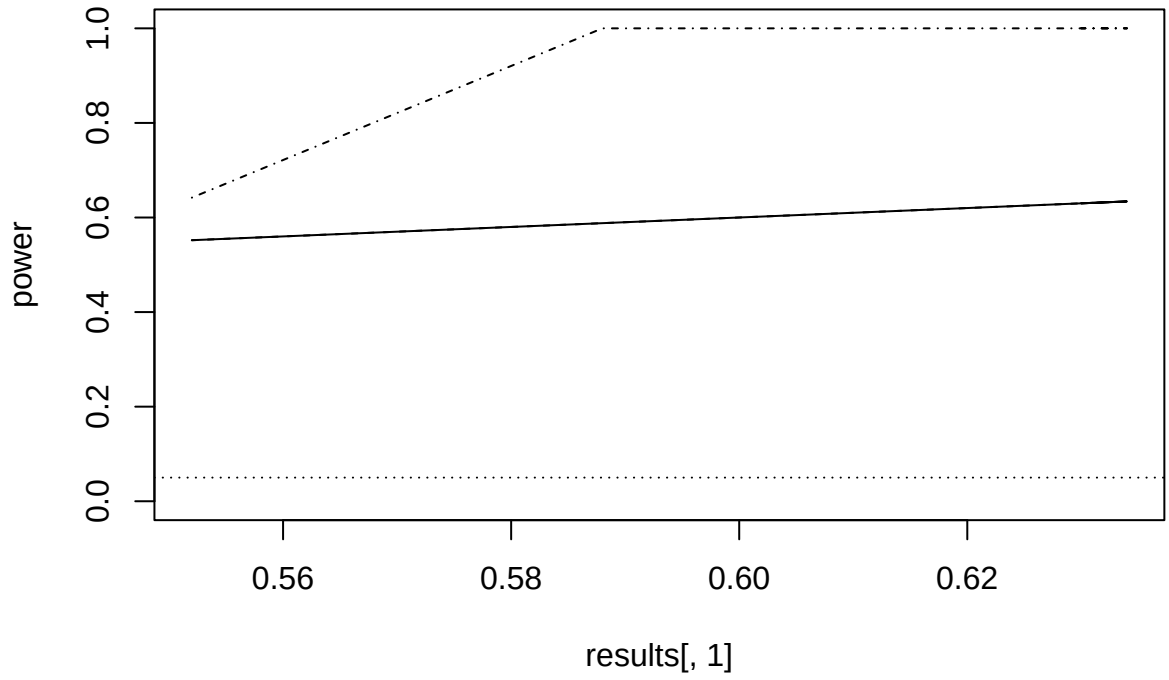
Part A

```
alpha <- 0.05
results <- matrix(1, nrow=4, ncol=3)

for(i in 1:length(ss)){
  test1 <- test2 <- test3 <- numeric(m)
  for (j in 1:m) {
    x <- rnorm(ss[i], 1.5, 0.07)
    y <- rcauchy(ss[i], 0, 0.5)
    test1[j] <- as.integer(t.test(x,y,paired=T,var.equal=F)$p.value <= alpha)
    test2[j] <- as.integer(t.test(x,y,paired=T,var.equal=T)$p.value <= alpha)
    test3[j] <- as.integer(wilcox.test(x,y,paired=T)$p.value <= alpha)
  }
  results[i,1] <- mean(test1)
  results[i,2] <- mean(test2)
  results[i,3] <- mean(test3)
}
results
```

```
##       [,1]  [,2]  [,3]
## [1,] 0.552 0.552 0.642
## [2,] 0.588 0.588 1.000
## [3,] 0.634 0.634 1.000
## [4,] 0.630 0.630 1.000
```

Plot

```
plot(results[,1], results[,2], ylim = c(0, 1), type = "l", ylab = "power")
lines(results[,1], results[,2], lty = 2)
lines(results[,1], results[,3], lty = 4)
abline(h = alpha, lty = 3)
```



Part B

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
results_p <- matrix(1, nrow=4, ncol = 3)
for(i in 1:length(ss)){
  test1 <- test2 <- test3 <- numeric(m)
  for (j in 1:m) {
    x <- rnorm(ss[i], 1.5, 0.07)
    y <- rcauchy(ss[i], 0, 0.5)
    test1[j] <- t.test(x,y,paired=T,var.equal=F)$p.value
    test2[j] <- t.test(x,y,paired=T,var.equal=T)$p.value
    test3[j] <- wilcox.test(x,y,paired=T)$p.value
  }
  results_p[i, 1] <- mean(test1)
  results_p[i, 2] <- mean(test2)
  results_p[i, 3] <- mean(test3)
}
```

```
results_p <- data.frame(results_p) %>% mutate(smallest = names(.)[max.col(.*-1)])
results_p
```

```
##          X1        X2           X3 smallest
## 1 0.1599248 0.1599248 7.129297e-02       X3
## 2 0.1473346 0.1473346 4.431856e-05       X3
## 3 0.1311931 0.1311931 2.620228e-08       X3
## 4 0.1379172 0.1379172 4.232036e-45       X3
```

Part C

```
results <- data.frame(results) %>% mutate(smallest = names(.)[max.col(.[1:3])])
results
```

```
##      X1    X2    X3 smallest
## 1 0.552 0.552 0.642       X3
## 2 0.588 0.588 1.000       X3
## 3 0.634 0.634 1.000       X3
## 4 0.630 0.630 1.000       X3
```

```
res <- data.frame(res_from_power = results$smallest, res_from_p = results_p$smallest)
res
```

```
##   res_from_power res_from_p
## 1             X3         X3
## 2             X3         X3
## 3             X3         X3
## 4             X3         X3
```

As we can see all goes to wilcoxon test as the best test of all!

Q4

Part A

```
M <- 100
success <- numeric(M)

for(i in 1:M){
  k <- sample(1:4, 1, prob=c(1/3,1/3,1/6,1/6))
  if (k %in% 1:2){
    success[i] <- sample(0:1, 1, prob=rep(1/2, 2))
  } else {
    success[i] <- sample(0:1, 1, prob=rep(1/2, 2))
  }
}

mean(success)
```

```
## [1] 0.47
```

Part B

```r
M <- 100
success <- numeric(M)

for(i in 1:M){
  success[i] <- sample(0:1, 1, prob=c(1/3, 2/3))
}

mean(success)
```

```
## [1] 0.67
```