Homework 7 Elias Washor

Elias Washor

2024-10-17

Q1)

```
sdev <- rep(NA, 10000)
for (i in 1:nrow(Xlist)) {
  sdev[i] <- sqrt(var(Xlist[i,]))
}

se <- qt(alp/2, n-1, lower.tail = FALSE) * sdev / sqrt(n)
```

```
## [1] 0.9504
```

```
sdev <- rep(NA, 10000)
for (i in 1:nrow(Xlist)) { sdev[i] <- sqrt(var(Xlist[i,])) }

moe <- qnorm(alp/2, lower.tail = FALSE) * sdev / sqrt(n)
```

```
## [1] 0.9008
```

```
sdlist2 <- apply(X2list, 1, sd)
moe.2 <- qnorm(alp/2, lower.tail = FALSE) * sdlist2 / sqrt(n)
```

```
set.seed(5400)

CovProb <- function(n, mu=2, alp=0.05) {
  X2list <- matrix(rexp(10000 * n, rate=1/mu), 10000, n)
  Xbar2list <- rowMeans(X2list)

  sdlist2 <- apply(X2list, 1, sd)
  moe.2 <- qnorm(alp/2, lower.tail = FALSE) * sdlist2 / sqrt(n)

  CI <- cbind(Xbar2list - moe.2, Xbar2list + moe.2)

  covered <- apply(CI, 1, function(x) mu > x[1] & mu < x[2])
  mean(covered)
}

CovProb(10) # [1] 0.8687
```

```
## [1] 0.8687
```

```r
CovProb(30) # [1] 0.9187
```

```
## [1] 0.9187
```

```r
CovProb(50) # [1] 0.9293
```

```
## [1] 0.9293
```

```r
CovProb(200) # [1] 0.9449
```

```
## [1] 0.9449
```

Q2)

```r
set.seed(5400)

CovProb <- function(n, mu=2, alp=0.05, dist, ...) {
  str_func <- (paste0("r", dist))
  rand <- do.call(str_func, args = list(10000 * n, ...))

  Xlist3 <- matrix(rand, 10000, n)
  Xbarlist3 <- rowMeans(Xlist3)

  sdlist3 <- apply(Xlist3, 1, sd)

  moe <- qnorm(alp/2, lower.tail = FALSE) * sdlist3 / sqrt(n)

  CI <- cbind(Xbarlist3 - moe, Xbarlist3 + moe)

  covered <- apply(CI, 1, function(x) mu > x[1] & mu < x[2])
  mean(covered)
}

## all should match with 1 - alpha = 0.90
CovProb(100, alp = 0.10, mu=2, dist = "exp", rate= 1/2)
```

```
## [1] 0.8916
```

```r
CovProb(100, alp = 0.10, mu=6, dist = "gamma", shape = 3, scale = 2)
```

```
## [1] 0.8899
```

```r
CovProb(100, alp = 0.10, mu = 5, dist = "unif", min = 0, max = 10)
```

```
## [1] 0.8911
```

```r
CovProb(100, alp = 0.10, mu = 6, dist = "pois", lambda = 6)
```

```
## [1] 0.8943
```

```r
CovProb(100, alp = 0.10, mu = 5/7, dist = "beta", shape1 = 5, shape2 = 2)
```

```
## [1] 0.8965
```

Q3)

```r
set.seed(5400)

### Estimate Bias, Variance, MSE ###
M <- matrix(rt(15 * 10000, 4), 15, 10000)
N <- apply(M, 2, FUN = sort)
O <- N[-c(1,15),]

theta_list <- colMeans(O)
th <- mean(theta_list)

MSE_list <- theta_list * theta_list ## - 0 + 0
MSE <- mean(MSE_list)
variance <- var(theta_list) ###

cat("Bias:", th,"\n")
```

```
## Bias: -0.001707364
```

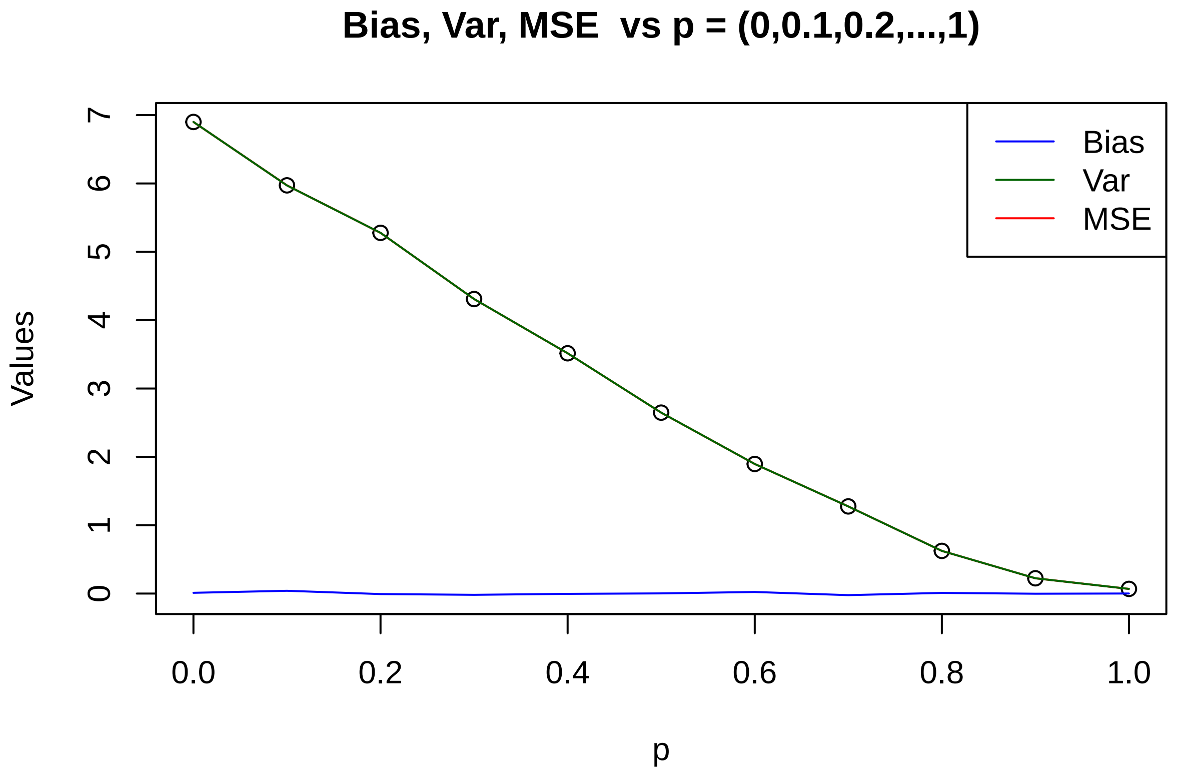
```r
cat("MSE:", MSE, "\n")
```

```
## MSE: 0.1028865
```

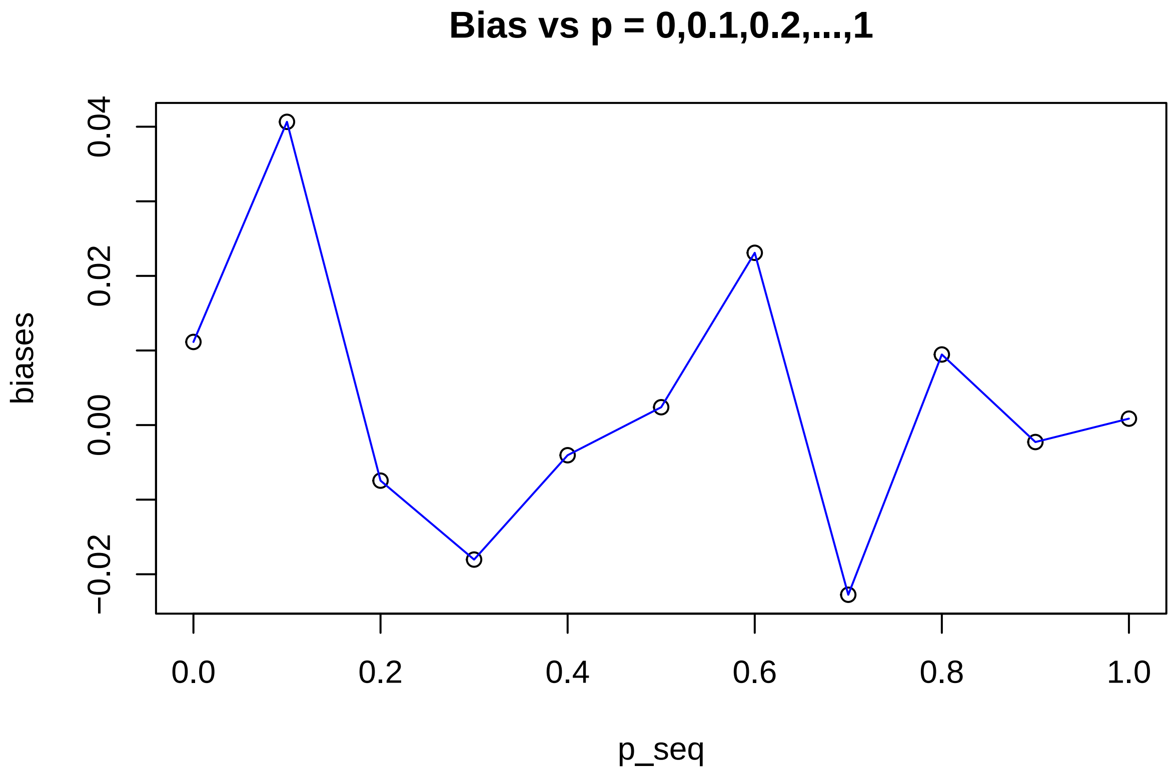
```r
cat("variance:", variance, "\n")
```

```
## variance: 0.1028939
```

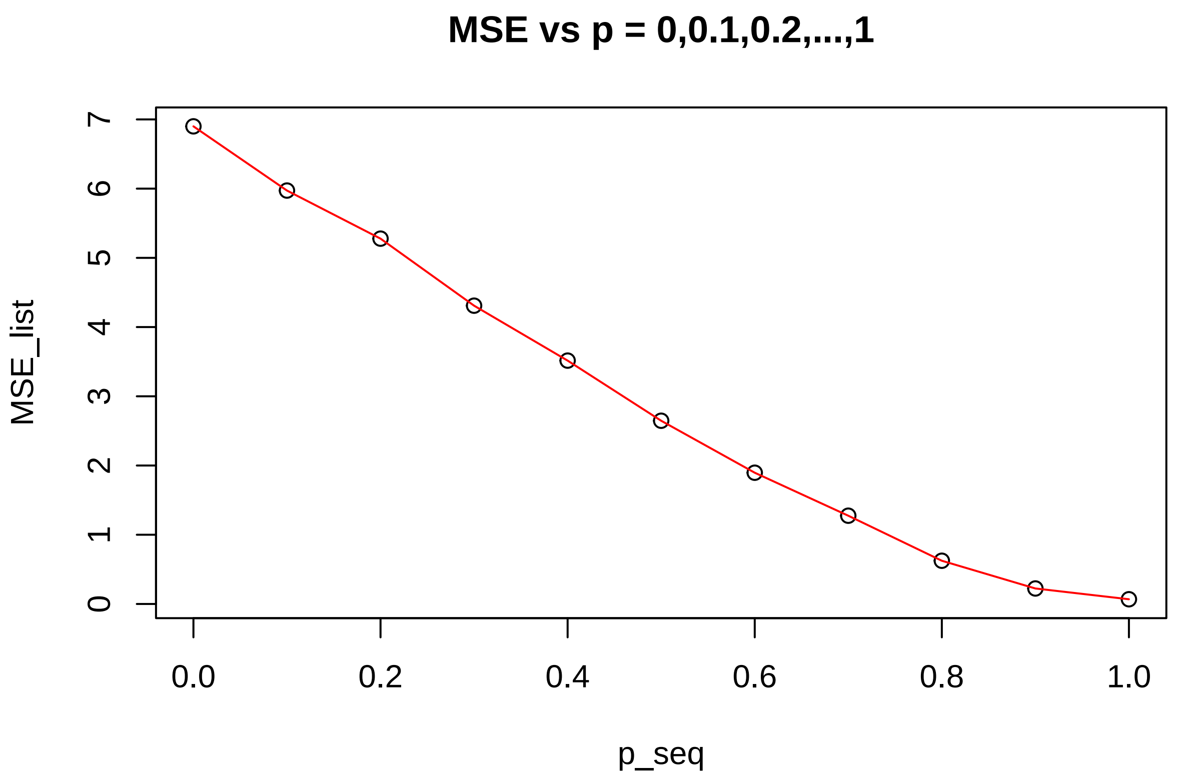
```r
### Mixture Normal Dist ###
biases <- rep(NA, 11)
MSE_list <- rep(NA, 11)
vars_list <- rep(NA, 11)

p_seq <- seq(0,1, 0.1)

for (i in 1:length(p_seq)) {
  p <- p_seq[i]
  runiform <- runif(15 * 10000)

  samp <- rep(NA, 15*10000)
  for (j in 1:(15*10000)) {
    if (runiform[j] <= p) {
      samp[j] <- rnorm(1,0,1)
    }
    else {samp[j] <- rnorm(1,0,10)}
  }
  X <- matrix(samp,15,10000)
  Y <- apply(X, 2, FUN = sort)
  Z <- Y[-c(1,15),]

  theta_hat2 <- (colMeans(Z))
  bias1<- mean(theta_hat2) - 0 ## 0 true mean
  MSE1 <- mean(theta_hat2 * theta_hat2)

  biases[i] <- bias1
  MSE_list[i] <- MSE1
  vars_list[i] <- var(theta_hat2)
}

#par(mfrow = c(2,2))
plot(p_seq, MSE_list, main = "Bias, Var, MSE  vs p = (0,0.1,0.2,...,1)",
     ylim = range(biases,MSE_list, vars_list), xlab="p", ylab="Values")
lines(p_seq, MSE_list, col="red")
lines(p_seq, biases, col="blue")
lines(p_seq, vars_list, col="darkgreen")

legend("topright", c("Bias", "Var", "MSE"),
       col = c("blue","darkgreen","red"), lty = 1)
```



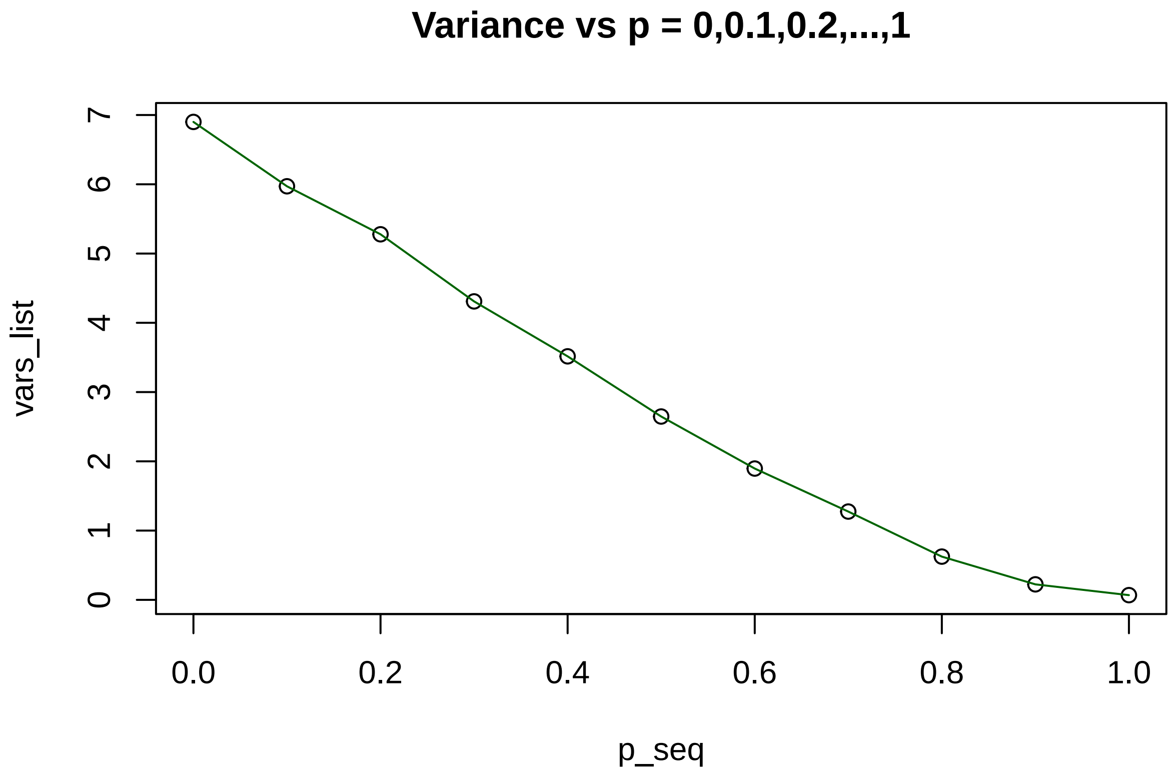
```
plot(p_seq, biases, main = "Bias vs p = 0,0.1,0.2,...,1")
lines(p_seq, biases, col="blue")
```



```
plot(p_seq, MSE_list, main = "MSE vs p = 0,0.1,0.2,...,1")
lines(p_seq, MSE_list, col="red")
```



```
plot(p_seq, vars_list, main = "Variance vs p = 0,0.1,0.2,...,1")
lines(p_seq, vars_list, col="darkgreen")
```



a) Bias, MSE, Variance: -0.0017, 0.1029, 0.1029

b) The Variance and MSE steadily decrease from apx. 7 –> 0 while p increases from 0 -> 1 . The Bias stays relatively close to 0 throughout, but is at its highest (0.04) when p = ~0.1 and lowest (-0.02) when p = ~0.7. Also, note that the MSE and variance are very close together, which means that the MSE from of estimating the trimmed mean is mostly due to variance since bias is relatively low.

Q4)

```r
set.seed(5400)
dat <- rpois(20,5)


## 95% conf int
alp <- 0.05
Xbar = mean(dat)
sdev <- sd(dat)
Z <- qnorm(alp/2, lower.tail = FALSE)

se <- Z * sdev / sqrt(20)

ConfInt <- Xbar + c(-se,se)
cat("Conf Int 95%: [", ConfInt, "]\n")
```

```
## Conf Int 95%: [ 4.32291 6.77709 ]
```

```
### coverage probability
CovProb2 <- function(n, mu=2, alp=0.05, dist, ...) {
  str_func <- (paste0("r", dist))
  rand <- do.call(str_func, args = list(100000 * n, ...))

  Xlist3 <- matrix(rand, 100000, n)
  Xbarlist3 <- rowMeans(Xlist3)
  sdlist3 <- apply(Xlist3, 1, sd)

  moe <- qnorm(alp/2, lower.tail = FALSE) * sdlist3 / sqrt(n)
  CI <- cbind(Xbarlist3 - moe, Xbarlist3 + moe)
  covered <- apply(CI, 1, function(x) mu > x[1] & mu < x[2])

  mean(covered)
}

cp <- CovProb2(20, alp = 0.05, mu = 5, dist = "pois", lambda = 5)
cat("Coverage Probability for interval: ", cp, "\n\n")
```

```
## Coverage Probability for interval:  0.93369
```

```
### Exact interval ###
n <- 20
s <- sum(dat)
lb <- qchisq(alp/2, 2*s) / (2*n)
ub <- qchisq(1- alp/2, 2*(s+1)) / (2*n)

Exact_Interval <- c(lb, ub)
cat("Theoretical Interval 95%: [", Exact_Interval, "]\n")
```

```
## Theoretical Interval 95%: [ 4.565666 6.683632 ]
```

```
### 10^5 replications
covered <- rep(NA, 100000)

## simulate exact conf int 10000 times
for (i in 1:100000) {
  data <- rpois(20,5)
  lambda <- 5
  n <- 20
  s <- sum(data)
  lb <- qchisq(alp/2, 2*s) / (2*n)
  ub <- qchisq(1- alp/2, 2*(s+1)) / (2*n)
  covered[i] <- (lb <= lambda & ub >= lambda)
}

pct  <- mean(covered)
cat("Coverage Probability for Theoretical Exact Interval: ", pct)
```

```
## Coverage Probability for Theoretical Exact Interval:  0.95431
```

The approximate conf int was slightly wider than the theoretical exact interval. However, the approximated interval only had a coverage probability of 0.934 and the theoretical had a coverage probability of 0.954.

Q5) Prop Conf Int

```r
set.seed(5400)

n <- 40
theta <- 0.6; Sim <- 10000; alp <- 0.05
Xbars <- rbinom(Sim, n, theta) / n

#WaldCI(sim = 10000, 100, 0.6, 0.05)
#n <- 100; th <- 0.6; alpha <- 0.05

## Wald interval
wald_se <- qnorm(1-alp/2) * sqrt(Xbars * (1 - Xbars)/n)
CI_Wald <- cbind(Xbars - wald_se, Xbars + wald_se)

## simple  interval
simple_se <- qnorm(1-alp/2) * sqrt(0.5 * (1 - 0.5)/n)
CI_simple <- cbind(Xbars - simple_se, Xbars + simple_se)

## Initialize empty CI Matrices
CI_score <- CI_CP <- CI_AC <- CI_Bayes <- matrix(NA, Sim, 2)

## other 4 CIs
for (i in 1:Sim) {
  CI_score[i, ] <- prop.test(x=(n*Xbars[i]), n=n,correct=FALSE,
                             conf.level=(1-alp))$conf.int

  CI_CP[i, ] <- binom.test(x=(n*Xbars[i]), n=n,conf.level=(1-alp))$conf.int

  ret <- binom::binom.confint(x=(n*Xbars[i]), n=n, conf.level=(1-alp),
                              methods="ac")

  CI_AC[i,] <- c(ret$lower, ret$upper)
  ret2 <- binom::binom.bayes(x=(n*Xbars[i]), n=n,type="central",
                             conf.level=(1-alp))

  CI_Bayes[i,] <- c(ret2$lower, ret2$upper)
}


metrics <- function(CI) {
  covered <- function(x) theta > x[1] & theta < x[2]
  se <- function(x) sd(x)/sqrt(nrow(CI))
  widths <- CI[,2] - CI[,1]
  coverages <- apply(CI, 1, covered)

  c(mean(widths), se(widths), mean(coverages), se(coverages))
}
```

```r
cnames <- paste0("CI_", c("simple", "Wald", "score", "CP", "AC", "Bayes"))
res <- sapply(lapply(cnames, get), metrics)
colnames(res) <- cnames
rownames(res) <- c("width", "se", "cov.prob", "se")
round(res, 3)
```

```
##          CI_simple CI_Wald CI_score CI_CP CI_AC CI_Bayes
## width        0.310   0.300    0.287 0.314 0.288    0.291
## se           0.000   0.000    0.000 0.000 0.000    0.000
## cov.prob     0.964   0.946    0.964 0.964 0.964    0.946
## se           0.002   0.002    0.002 0.002 0.002    0.002
```

Comparison of 6 Prop. Confidence Intervals)

The Wald Conf Int and CI_Bayes have slightly lower coverage probabilities of 0.946 compared to the goal of 0.95, regardless they are still close. Of the other 4 intervals, Agresti-Coull and the Wilson Score CI have respective widths of (0.288 and 0.287). Thus, both of them are optimal intervals to use since they have narrow widths and higher coverage probabilities.

Simply based on width, the simple conservative interval was wide (0.31) and so was Clopper-Pearson (0.314), but these are only marginally wider than the others by ~ 0.02. Further, The Wald Conf Interval was the widest interval (0.3) that had the lowest coverage (0.946) – which makes it less favorable to use, but again not a substantial difference.