HW3

Hannah Zmuda

11/18/2020

Question 1

Problem

Use Monte Carlo simulation to investigate whether the empirical Type I error rate of the t-test is approximately equal to the nominal significance level α, when the sampled population is non-normal. The t-test is robust to mild departures from normality. Discuss the simulation results for the cases where the sampled population is (i) $\chi^2(1)$, (ii) Uniform(0,2), and (iii) Exponential(rate=1). In each case, test H0 : $\mu = \mu_0$ vs H0 : $\mu \neq \mu_0$, where μ_0 is the mean of $\chi^2(1)$, Uniform(0,2), and Exponential(1), respectively.

Solution

```
library(ggplot2)
#Part a: looking at a type I error
#all follow a similar algorithm as shown on page 193 in SCRR
n <- 100 #number of replicates
a <- 0.05 #significance level alpha
muA <- mean(rchisq(n, df = 1)) #mu0 in part a
muB <- mean(runif(n, 0, 2)) #mu0 in part b
muC <- mean(rexp(n, rate = 1)) #mu0 in part c
#become alternatives in the estimate power of a test (b).

m <- 1000 #number of replicates
pA <- numeric(m)
pB <- numeric(m)
pC <- numeric(m)

for(i in 1:m){
  xA <- rchisq(n, df = 1) #sample dist part a
  xB <- runif(n, 0, 2) #sample dist part b
  xC <- rexp(n, rate = 1) #sample dist part c

  ttestA <- t.test(xA, alternative = "two.sided",mu = muA)
  ttestB <- t.test(xB, alternative = "two.sided",mu = muB)
  ttestC <- t.test(xC, alternative = "two.sided",mu = muC)

  pA[i] <- ttestA$p.value
  pB[i] <- ttestB$p.value
  pC[i] <- ttestC$p.value
}
```

```r
pHatA <- mean(pA <= a)
pHatB <- mean(pB <= a)
pHatC <- mean(pC <= a)

seHatA <- sqrt(pHatA * (1- pHatA)/m)
seHatB <- sqrt(pHatB * (1- pHatB)/m)
seHatC <- sqrt(pHatC * (1- pHatC)/m)
#Means:
cat("The means (of the p-values) are: ", c(pHatA, pHatB, pHatC), "\n")
```

```
## The means (of the p-values) are:  0.172 0.457 0.078
```

```r
#SE:
cat("The standard errors (of the p-values) are: ", c(seHatA, seHatB, seHatC))
```

```
## The standard errors (of the p-values) are:  0.01193382 0.01575281 0.00848033
```

```r
#Part b: Estimating power of a test and outputting empirical power curves of the t-test from the three
#initial data array
mu1 <- c(seq(0,2,1/20))
MC <- length(mu1)
powA <- numeric(MC)
powB <- numeric(MC)
powC <- numeric(MC)
powNorm <- numeric(MC)

#select theta_1 from the parameter subspace
#muA, muB, and muC values
#set for loop
for(j in 1:MC){
  #Chi Squared Distribution
  pvalA <- replicate(MC, expr = {
    xA <- rchisq(n, df = 1) #sample dist part a
    ttestA <- t.test(xA, alternative = "two.sided",mu = mu1[j])
    ttestA$p.value
  })
  powA[j] <- mean(pvalA <= a)
  #Uniform Distribution
  pvalB <- replicate(MC, expr = {
    xB <- runif(n, 0, 2) #sample dist part b
    ttestB <- t.test(xB, alternative = "two.sided",mu = mu1[j])
    ttestB$p.value
  })
  powB[j] <- mean(pvalB <= a)
  #Exponential Distribution
  pvalC <- replicate(MC, expr = {
    xC <- rexp(n, rate = 1) #sample dist part c
    ttestC <- t.test(xC, alternative = "two.sided",mu = mu1[j])
    ttestC$p.value
  })
  powC[j] <- mean(pvalC <= a)
  #Normal Distribution using power.t.test function
```

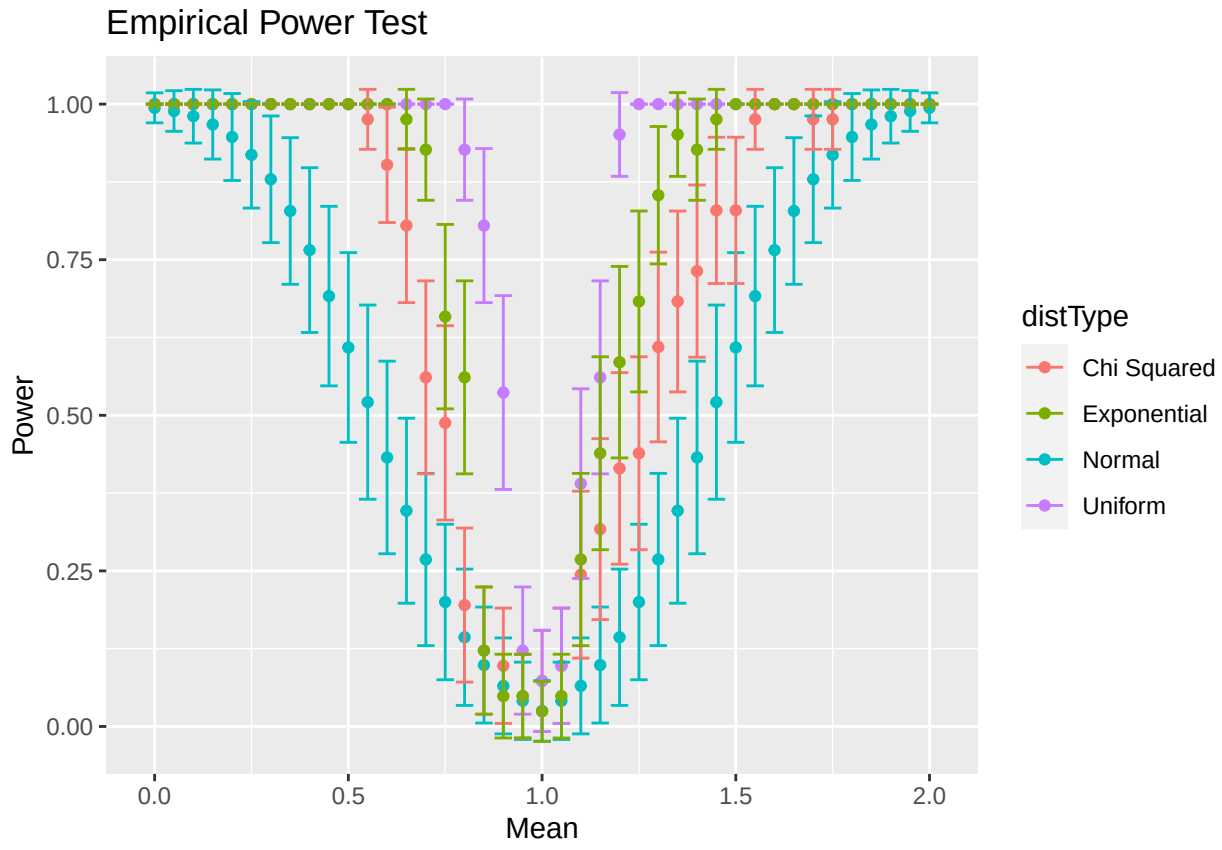
```r
  powerTest <- power.t.test(n = MC, delta = mu1[j]-1,sig.level = a,alternative = "two.sided")
  powNorm[j] <- powerTest$power

}

#making of the data frame
mean <- c(mu1,mu1,mu1,mu1)
powerR <- c(powNorm, powA, powB, powC)
distType <- c(replicate(MC, "Normal"),replicate(MC, "Chi Squared"),replicate(MC, "Uniform"),replicate(MC

data <- data.frame(mean, powerR,distType)

ggplot(data = data, aes(x = mean, y = powerR, color = distType)) +
  geom_point() +
  labs(x = 'Mean', y = 'Power', title = 'Empirical Power Test') +
  geom_errorbar(data = data, mapping = aes(x = mean, ymin = powerR - 2*(sqrt(powerR * (1- powerR)/MC)),
```



Question 2

Part a and b: MC and IS

For part (a) and (b), we are comparing Monte Carlo method and Importance Sampling (IS) to estimate α using the Z-test. In order to do Importance Sampling in part (b), we must first derive the weights: $f(X)/g(X)$. The problem statement tells us that the function $g(x) = Pois(1.5\lambda)$, which means the importance function, $f(x)$, is also based on the Poisson pdf. Therefore,

$$f(x) = \frac{e^{\lambda}\lambda^{x}}{x!}$$

and

$$g(x) = \frac{e^{-1.5\lambda}1.5\lambda^{x}}{x!}$$

```
set.seed(475)
#part a estimate alpha using MCEM
n <- 10 # distribution size
m <- 100 #Monte Carlo sample size
l <- 2 #lambda is equal to 2
#z test with Monte Carlo
ztest <- replicate(m, expr = {
  x <- rpois(n, l) #samples
  ztest <- (mean(x)-2)/(sd(x)/sqrt(n))
})
#alpha estimate
aMC <- mean(ztest > 2.326)
cat("Alpha estimate using MC:", aMC, "\n")
```

```
## Alpha estimate using MC: 0.01
```

```
#part b: estimate alpha using Importance Sampling
g <- function(x) (exp(-1.5*l)*((1.5*l)^x))/factorial(x)
f <- function(x) (exp(-l)*(l^x))/factorial(x)
phi <- function(x) as.numeric((mean(x)-2)/(sd(x)/sqrt(n))  > 2.326)
weight <- function(x) f(x) / g(x)
out <- numeric(m)

out <- replicate(m, expr = {
  x <- rpois(n, l) #samples
  out <- phi(x)*weight(x)
})

isOut <- mean(out)
cat("Alpha estimate using IS:", isOut,"\n")
```

```
## Alpha estimate using IS: 0.008561283
```

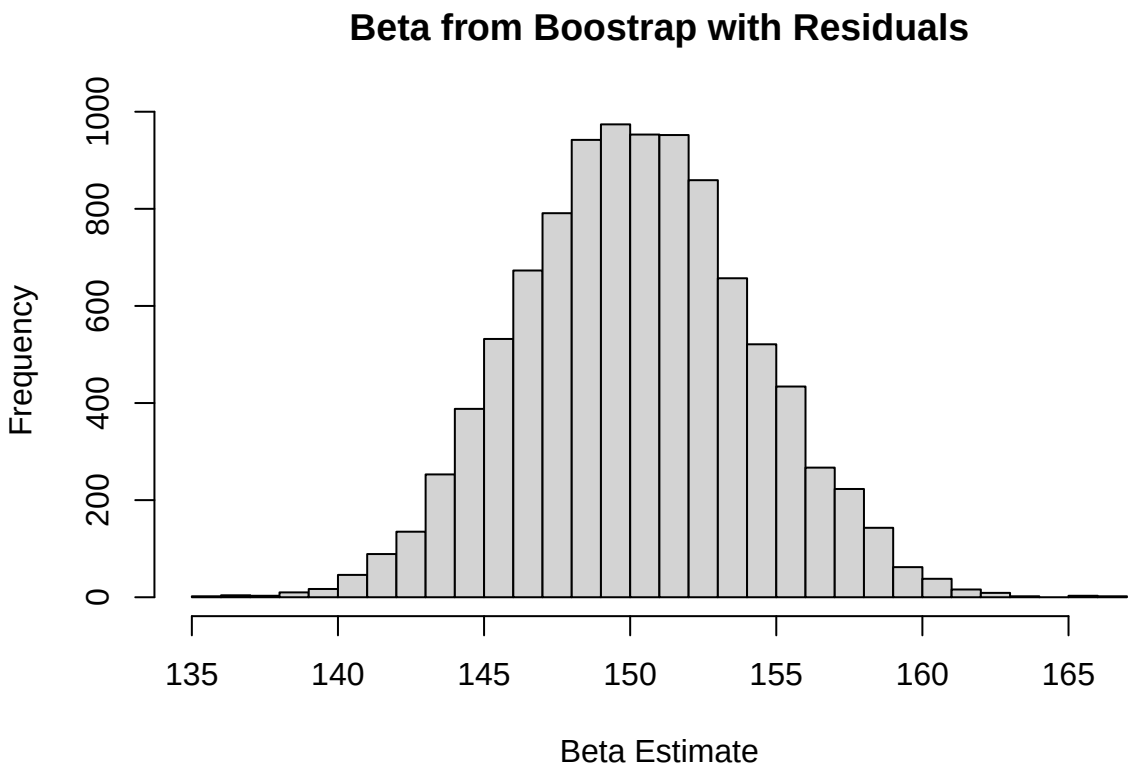
Part c: Which is better?

Both outputs of the function are off from the expected 0.05 estimate of α. I would argue that the Monte Carlo is easier to implement because the weights do not need to be derived. However, IS estimates the α value closer to the expected value, compared to using Monte Carlo by itself.

Question 3

```r
set.seed(475)
#get data
salmon <-read.table("salmon.dat", header = TRUE)
#initalize values
n <- length(salmon$recruits)
y <- 1/salmon$recruits
x <- 1/salmon$spawners
B <- 10000
beta.hat <- rep(0,B)
#get the linear model and coefficients and first set of residuals
linear.model <- lm(y~x)
#calculate coefficients (slope and intercept)
r <- linear.model$coefficients[1] + (linear.model$coefficients[2] * x)
beta <- (1-linear.model$coef[2])/linear.model$coef[1]
eps <- y - r#residual error

#Bootstrapping the residuals
for(b in 1:B){
  eps.new <- eps[sample(1:n,n,replace = TRUE)]
  yB <- eps.new + r
  fit <- lm(yB ~ x)
  beta.hat[b] <- (1-fit$coef[2])/fit$coef[1]
}
#output
hist(beta.hat,breaks = 25, main = "Beta from Boostrap with Residuals", xlab = "Beta Estimate")
```




```
print("BOOTSTRAP THE RESIDUALS")
```

```
## [1] "BOOTSTRAP THE RESIDUALS"
```

```
print("95% CI Beta estimate from Boostrap with Residuals")
```

```
## [1] "95% CI Beta estimate from Boostrap with Residuals"
```

```
quantile(beta.hat,probs = c(0.025,0.975),na.rm=T)
```

```
##     2.5%    97.5%
## 142.6342 158.1336
```

```
print("Standard error")
```

```
## [1] "Standard error"
```

```
inside <- sum(beta.hat - mean(beta.hat))
inside.sq <- inside^2
sqrt(inside.sq/(n-1))
```

```
## [1] 6.740203e-12
```

```r
print("Estimates Bias")
```

```
## [1] "Estimates Bias"
```

```r
bias = mean(beta.hat)-beta
bias
```

```
##         x
## 0.1047161
```

```r
print("Bias corrected estimate")
```

```
## [1] "Bias corrected estimate"
```

```r
cor.beta <- beta-bias
cor.beta
```

```
##        x
## 149.9929
```

```r
print("Standard Error for corrected estimator")
```

```
## [1] "Standard Error for corrected estimator"
```

```r
diff <- beta.hat - bias
inside <- sum(diff - mean(diff))
inside.sq <- inside^2
sqrt(inside.sq/(n-1))
```

```
## [1] 6.740203e-12
```

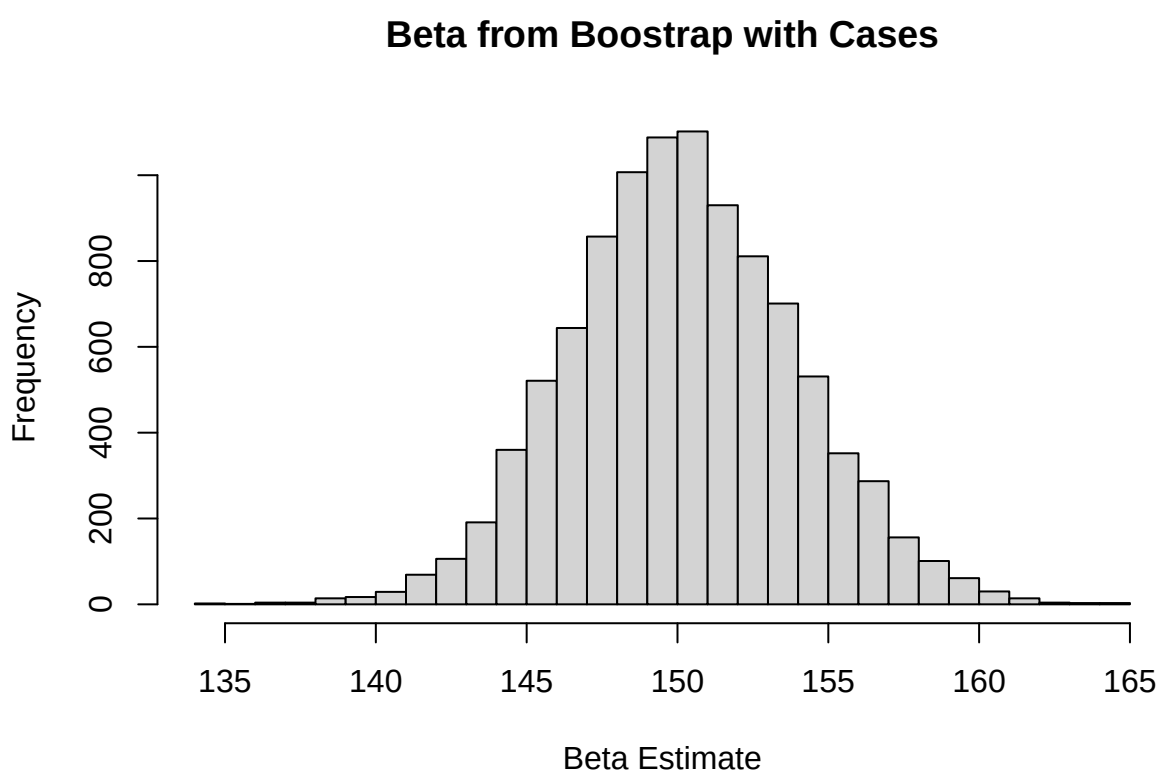
```r
#part b: Jackknife after bootstrap
se.beta <- rep(0,B)
for(b in 1:B){
  se.beta[b] <- sd(beta.hat[-b])
}
se.beta.bar <- mean(se.beta)
se.beta.jack <- sqrt(sum((se.beta-se.beta.bar)^2)/(n-1))
cat("Bias Corrected Estimate:", (se.beta.bar-sd(beta.hat)),"\n")
```

```
## Bias Corrected Estimate: -9.771516e-09
```

```r
cat("SE from Jackknife-after-bootstrap:", se.beta.jack)
```

```
## SE from Jackknife-after-bootstrap: 0.004470413
```

```r
#Bootstrapping the Cases
set.seed(475)
#get data
salmon <-read.table("salmon.dat", header = TRUE)
x <- 1/(salmon$spawners)
y <- 1/(salmon$recruits)
n <- length(x)
B <- 10000
beta.new <- rep(0,B)
model.new <- lm(y ~ x)
beta.true.c <- as.numeric((1-model.new$coef[2])/model.new$coef[1])
for(b in 1:B){
  j <- sample(1:n,n,replace = TRUE)
  x.new <- x[j]
  y.new <- y[j]
  model.new <- lm(y.new~x.new)
  beta.new[b] <- (1-model.new$coef[2])/model.new$coef[1]
}
hist(beta.new,breaks = 25, main = "Beta from Boostrap with Cases", xlab = "Beta Estimate")
```




```r
ci.95.case <- quantile(beta.new,probs = c(0.025,0.975))
print("95% CI Beta estimate from Boostrap with Cases")
```

```
## [1] "95% CI Beta estimate from Boostrap with Cases"
```

```r
ci.95.case
```

```
##     2.5%    97.5%
## 143.0399 157.7360
```

```r
print("Standard error")
```

```
## [1] "Standard error"
```

```r
inside <- sum(beta.new - mean(beta.new))
inside.sq <- inside^2
sqrt(inside.sq/(n-1))
```

```
## [1] 1.71122e-11
```

```r
print("Estimates Bias")
```

```
## [1] "Estimates Bias"
```

```r
bias = mean(beta.new)-beta.true.c
bias
```

```
## [1] 0.07331751
```

```r
print("Bias corrected estimate")
```

```
## [1] "Bias corrected estimate"
```

```r
cor.beta <- beta.true.c-bias
cor.beta
```

```
## [1] 150.0243
```

```r
print("Standard Error for corrected estimator")
```

```
## [1] "Standard Error for corrected estimator"
```

```r
diff <- beta.new - bias
inside <- sum(diff - mean(diff))
inside.sq <- inside^2
sqrt(inside.sq/(n-1))
```

```
## [1] 1.71122e-11
```

```r
#part b: Jackknife after bootstrap
se.beta <- rep(0,B)
m.beta <- rep(0,B)
for(b in 1:B){
  se.beta[b] <- sd(beta.new[-b])
}
se.beta.bar <- mean(se.beta)
se.beta.jack <- sqrt(sum((se.beta-se.beta.bar)^2)/(n-1))
cat("Bias Corrected Estimate:", (se.beta.bar-sd(beta.new)),"\n")
```

```
## Bias Corrected Estimate: -1.015036e-08
```

```r
cat("SE from Jackknife-after-bootstrap:", se.beta.jack)
```

```
## SE from Jackknife-after-bootstrap: 0.004426662
```

Question 4

part a

```
set.seed(475)
#get data
cancer <- read.table("cancersurvival.dat", header = TRUE) #1 is stomach cancer, 2 is breast cancer
cancer[,1] <- log(cancer[,1])
breast.cancer <- cancer[cancer[,2]==2,1]
stomach.cancer <- cancer[cancer[,2]==1,1]
#initialize
n <- length(breast.cancer)
B <- 10000
theta.b <- NULL
thetas.b <- rep(0,B)
theta.hat.b <- rep(0,n)
theta.s <- NULL
thetas.s <- rep(0,B)
theta.hat.s <- rep(0,n)
psi.b <- rep(0,n)
psi.s <- rep(0,n)

#Original estimates
theta.b <- mean(breast.cancer)
theta.s <- mean(stomach.cancer)

#Bootstrap!!
for(i in 1:B){
  rand.sample <- sample(1:n,n,replace = TRUE)
  cancer.new.s <- stomach.cancer[rand.sample]
  cancer.new.b <- breast.cancer[rand.sample]
  thetas.b[i] <- mean(cancer.new.b)
  thetas.s[i] <- mean(cancer.new.s)
}
ci.95.b <- quantile(thetas.b,c(0.025,0.975),na.rm=T)
ci.95.s <- quantile(thetas.s,c(0.025,0.975),na.rm=T)

#BCA
for(j in 1:n){
  theta.hat.b[j] <- mean(breast.cancer[-j])
  theta.hat.s[j] <- mean(stomach.cancer[-j])
}
for(k in 1:n){
  psi.b[k] <- mean(theta.hat.b[-k])-theta.hat.b[k]
  psi.s[k] <- mean(theta.hat.s[-k])-theta.hat.s[k]
}
a.b <- ((1/6)*sum(psi.b^3))/((sum(psi.b^2))^(3/2))
a.s <- ((1/6)*sum(psi.s^3))/((sum(psi.s^2))^(3/2))
b.b <- qnorm(mean(thetas.b<theta.b),0,1)
b.s <- qnorm(mean(thetas.s<theta.s),0,1)
beta1.b <- pnorm(b.b + (b.b+qnorm(.025,0,1))/(1-a.b*(b.b+qnorm(.025,0,1))))
beta2.b <- pnorm(b.b + (b.b+qnorm(.975,0,1))/(1-a.b*(b.b+qnorm(.975,0,1))))
beta1.s <- pnorm(b.s + (b.s+qnorm(.025,0,1))/(1-a.s*(b.s+qnorm(.025,0,1))))
```

```r
beta2.s <- pnorm(b.s + (b.s+qnorm(.975,0,1))/(1-a.s*(b.s+qnorm(.975,0,1))))
ci.95.bc.b = quantile(thetas.b,c(beta1.b,beta2.b),na.rm=T)
ci.95.bc.s = quantile(thetas.s,c(beta1.s,beta2.s),na.rm=T)
#output
theta.b #observed estimate theta
```

```
## [1] 6.558603
```

```r
print("95% Confidence Interval for Breast Cancer (Basic Bootstrap)")
```

```
## [1] "95% Confidence Interval for Breast Cancer (Basic Bootstrap)"
```

```r
ci.95.b #95% ci from basic bootstrap
```

```
##     2.5%    97.5%
## 5.537453 7.396224
```

```r
print("95% Confidence Interval for Breast Cancer (Basic BCa)")
```

```
## [1] "95% Confidence Interval for Breast Cancer (Basic BCa)"
```

```r
ci.95.bc.b #95% ci from BCA
```

```
## 0.9418057%  95.09835%
##   5.329144   7.286088
```

```r
print("95% Confidence Interval for Stomach Cancer (Basic Bootstrap)")
```

```
## [1] "95% Confidence Interval for Stomach Cancer (Basic Bootstrap)"
```

```r
theta.s #observed estimate theta
```

```
## [1] 4.96792
```

```r
ci.95.s #95% ci from basic bootstrap
```

```
##     2.5%    97.5%
## 4.064429 5.190881
```

```r
print("95% Confidence Interval for Stomach Cancer (Basic BCa)")
```

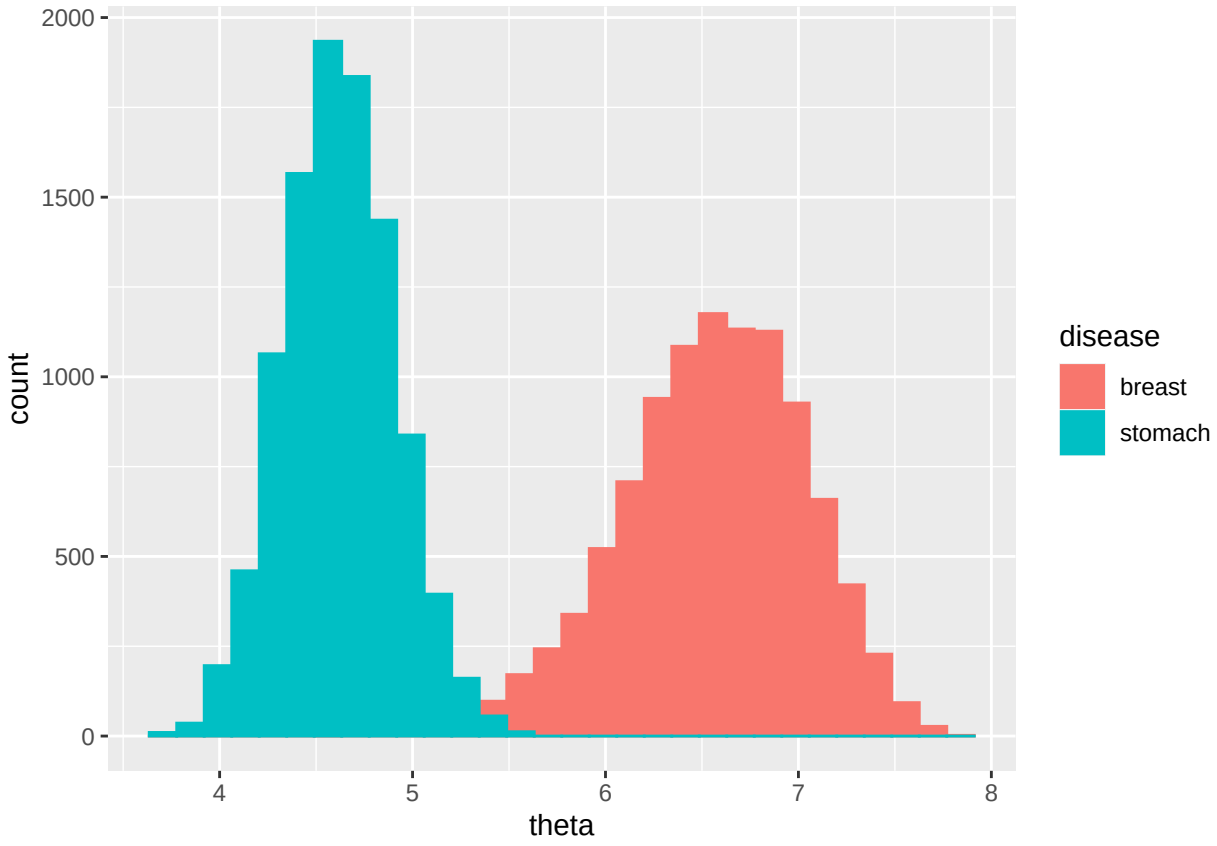
```
## [1] "95% Confidence Interval for Stomach Cancer (Basic BCa)"
```

```r
ci.95.bc.s #95% ci from BCA
```

```
## 67.88046% 99.99967%
##  4.750756  5.614601
```

```
cancer.output <- data.frame(
  theta = c(thetas.s,thetas.b),
  disease = c(rep('stomach',length(thetas.s)),rep('breast',length(thetas.b)))
)
ggplot(cancer.output,aes(x = theta, fill = disease, color = disease)) + geom_histogram(position = "iden
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
library(ggplot2)
#get data
cancer <- read.table("cancersurvival.dat", header = TRUE) #1 is stomach cancer, 2 is breast cancer
cancer[,1] <- log(cancer[,1])
breast.cancer <- cancer[cancer[,2]==2,1]
stomach.cancer <- cancer[cancer[,2]==1,1]
#initialize
n <- length(breast.cancer)
B <- 100000
#Bootstrap t
#initialize data
thetas.b <- rep(0,B)
t.b <- rep(0,B)
ci.95.t.b <- rep(0,2)
thetas.s <- rep(0,B)
t.s <- rep(0,B)
```

```r
ci.95.t.s <- rep(0,2)
#get original theta value from line 257, use to calculate covariance
theta.b <- mean(breast.cancer)
sigma.b <- sd(breast.cancer)/sqrt(length(breast.cancer))
theta.s <- mean(stomach.cancer)
sigma.s <- sd(stomach.cancer)/sqrt(length(stomach.cancer))
#Bootstrap loop
for(d in 1:B){
  rand.sample <- sample(1:n,n,replace = TRUE)
  cancer.new.s <- stomach.cancer[rand.sample]
  cancer.new.b <- breast.cancer[rand.sample]
  thetas.b[d] <- mean(cancer.new.b)
  thetas.s[d] <- mean(cancer.new.s)
  sigmas.b <- sd(cancer.new.b)/sqrt(length(breast.cancer))
  sigmas.s <- sd(cancer.new.s)/sqrt(length(stomach.cancer))
  t.b[d] = (thetas.b[d]-theta.b)/sigmas.b #reference distribution
  t.s[d] = (thetas.s[d]-theta.s)/sigmas.s #reference distribution
}
ci.95.t.b[1] = theta.b - sigma.b*quantile(t.b,.975,na.rm=T)
ci.95.t.b[2] = theta.b - sigma.b*quantile(t.b,.025,na.rm=T)
print("95% Confidence Interval for Breast Cancer")
```

```
## [1] "95% Confidence Interval for Breast Cancer"
```

```r
ci.95.t.b
```

```
## [1] 4.274203 7.399529
```

```r
ci.95.t.s[1] = theta.s - sigma.s*quantile(t.s,.975,na.rm=T)
ci.95.t.s[2] = theta.s - sigma.s*quantile(t.s,.025,na.rm=T)
print("95% Confidence Interval for Stomach Cancer")
```

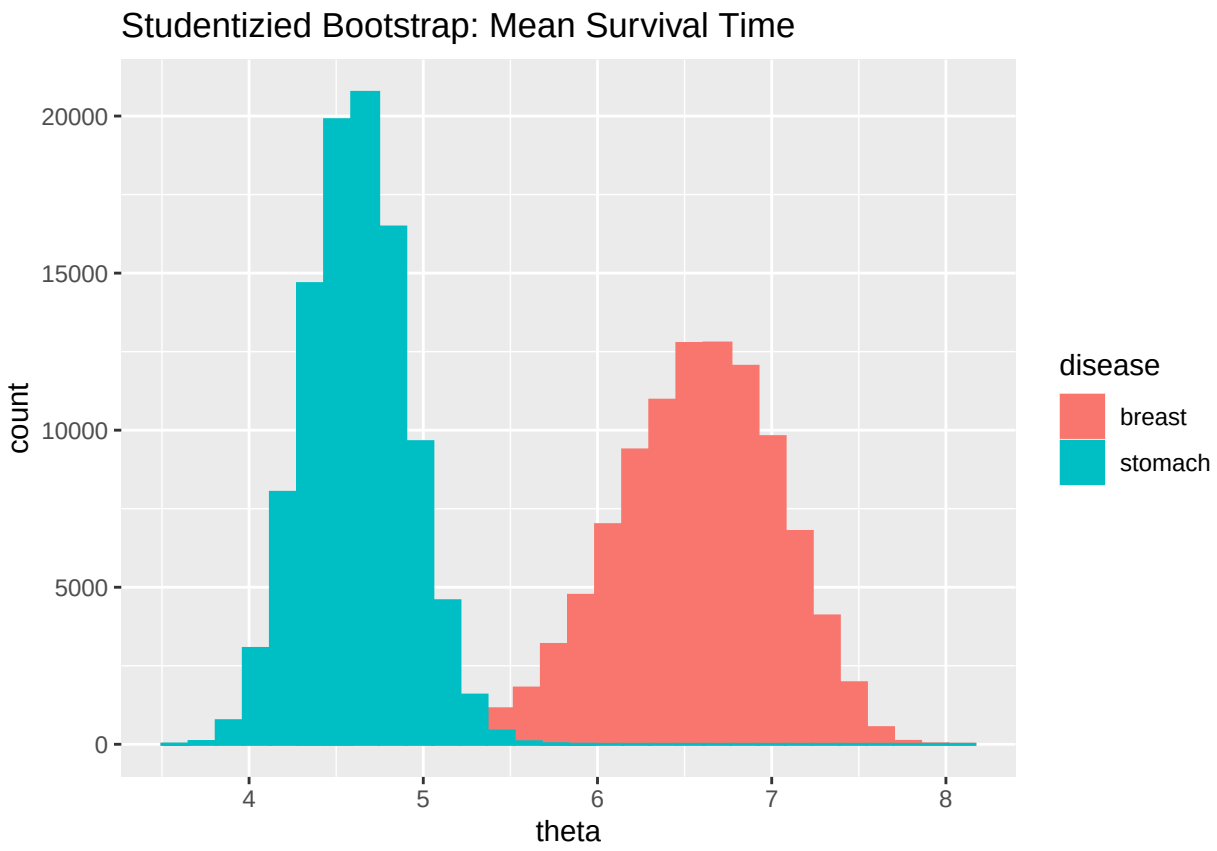
```
## [1] "95% Confidence Interval for Stomach Cancer"
```

```r
ci.95.t.s
```

```
## [1] 4.671640 6.564703
```

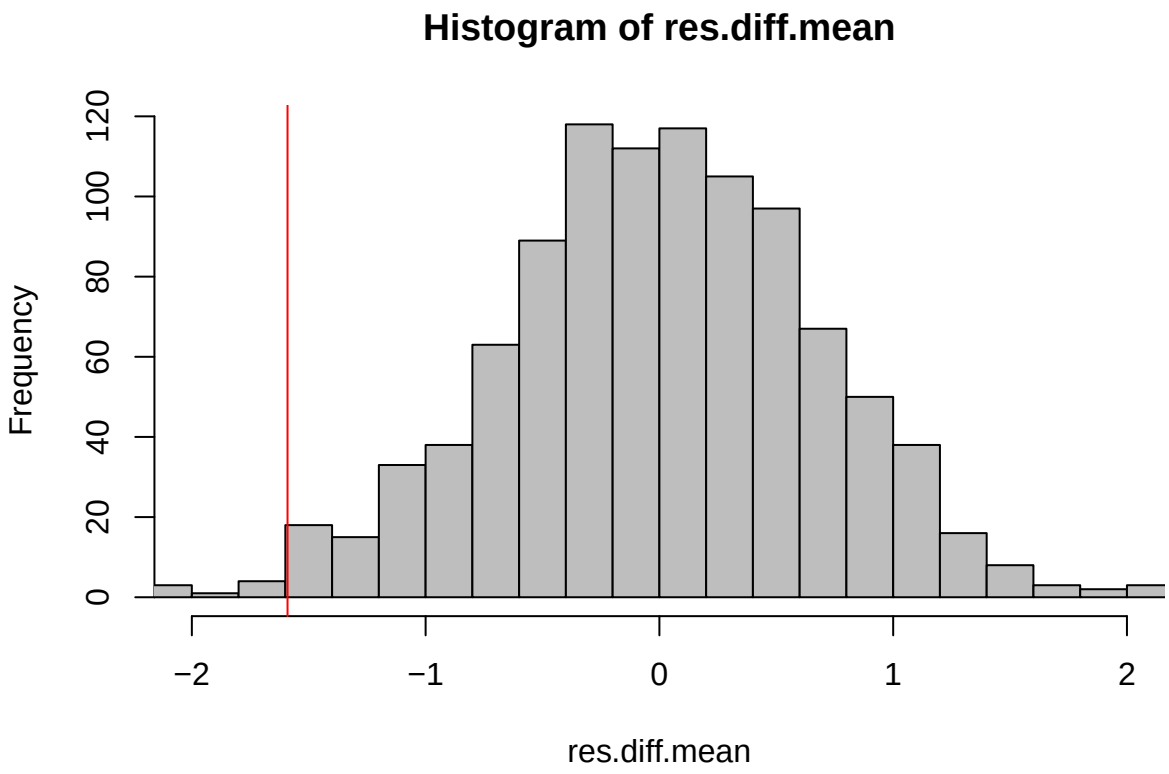
```r
cancer.output <- data.frame(
  theta = c(thetas.s,thetas.b),
  disease = c(rep('stomach',length(thetas.s)),rep('breast',length(thetas.b)))
)
ggplot(cancer.output,aes(x = theta, fill = disease, color = disease)) + geom_histogram(position = "iden
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



part b

```
set.seed(475)
#get data
cancer <- read.table("cancersurvival.dat", header = TRUE) #1 is stomach cancer, 2 is breast cancer
cancer[,1] <- log(cancer[,1])
#initialize
P <- 1000 #Number of permutation
theta.b <- mean(cancer[cancer$disease == 2,1])
theta.s <- mean(cancer[cancer$disease == 1,1])
res.diff.mean <- rep(0,P)
#Loop for permutation test
for(p in 1:P){
  perm <- sample(nrow(cancer))
  dat <- transform(cancer, disease = disease[perm])
  thetas.b <- mean(dat[dat$disease == 2,"survivaltime"])
  thetas.s <- mean(dat[dat$disease == 1,"survivaltime"])
  res.diff.mean[p] <- thetas.s - thetas.b
}
obs.diff.mean <- theta.s-theta.b
hist(res.diff.mean, breaks = 25, col = "gray", xlim = c(-2,2))
abline(v = obs.diff.mean, col = "red")
```




```
quantile(res.diff.mean,probs = c(0.025,0.975))
```

```
##      2.5%     97.5%
## -1.406922  1.244655
```

```
obs.diff.mean
```

```
## [1] -1.590684
```

Because the observed mean is outside of the 2.5% confidence interval, we can reject the null hypothesis.

part c

```
library(boot)
set.seed(475)
#get data
cancer <- read.table("cancersurvival.dat", header = TRUE) #1 is stomach cancer, 2 is breast cancer
cancer <- cancer[cancer$disease == 2,]
log.cancer <- log(cancer[,1])
thetas.b <- replicate(B,expr = {
  mean(log.cancer[sample(length(log.cancer),replace = TRUE)])
})
```

```r
true.thetas <- replicate(B,expr = {
  mean(cancer[sample(nrow(cancer),replace = TRUE),1])
})
quantile(exp(thetas.b), probs = c(0.025,0.975))
```

```
##      2.5%     97.5%
##  260.4049 1624.4914
```

```r
quantile(true.thetas, probs = c(0.025,0.975))
```

```
##      2.5%     97.5%
##  751.4545 2132.0932
```

Question 5

part i

```
library(boot)
set.seed (475)
```