# Assignment 2 , Q1

nsim <- 100000

# Generating Uniform random variate to generate the random variate with given F(x)

sunif <- runif(nsim,0,1)

rand\_var <- rep(0,nsim)

for (i in 1:nsim) {

if (sunif[i] >=0.5 ) {

if (sunif[i] == 0.5) {

rand\_var[i] <- 0}

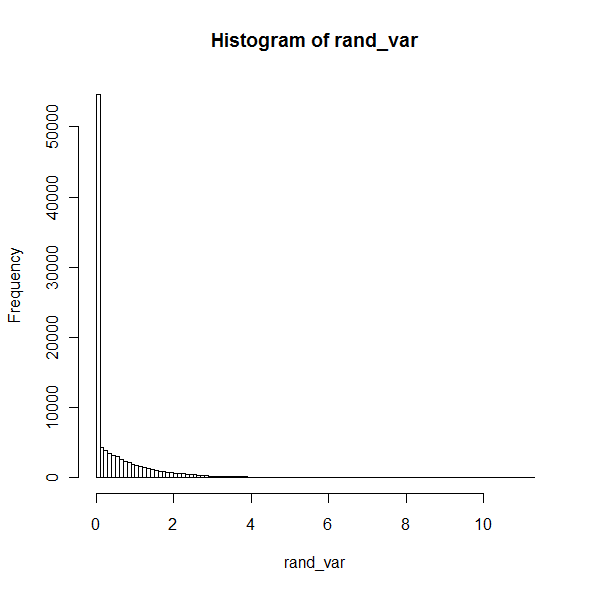
else { rand\_var[i] <- (log(1/(1-sunif[i])) -log(2)) }}}

hist(rand\_var,breaks=100)

cat ("Mean", mean(rand\_var))

cat ("Standard Dev", sd(rand\_var))

**# RESULT : Mean : 0.4976 , Standard Deviation : 0.8604**

****

**Q2 :**

# assigning p to 0.04

p<-0.4

nsim <-1000

alpha <-0.05

n<- c(10,30,50,100,250,500,1000)

n\_count <- rep(0,length(n))

for (i in 1:length(n)) {

# count reset to zero for each sample size to determine p^

count <- 0

# Level of significance alpha =.05

epsilon <- sqrt((1/(2\*n[i]))\*(log(2/alpha)))

for (j in 1:nsim) {

# resetting sample to n value

s<- rep(0,n[i])

# generating the bernouli variate

s<- rbinom(n[i],1,p)

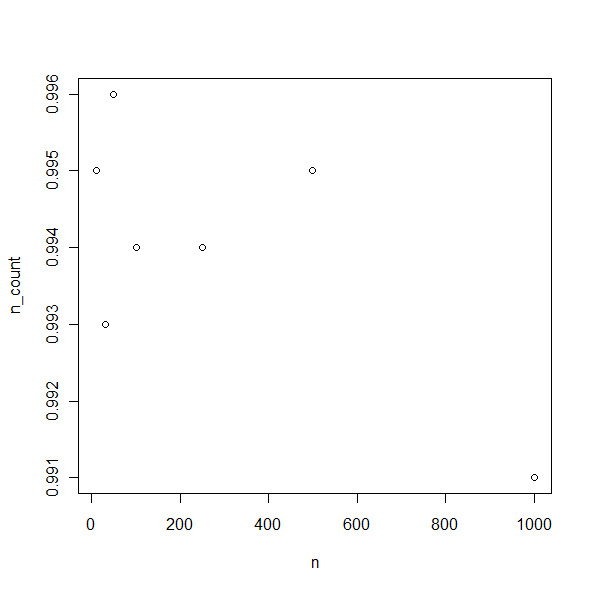
s\_mean <- mean(s)

if ( (s\_mean <= (p + epsilon)) & (s\_mean >= (p - epsilon)) ) {

count <- count +1 } }

n\_count[i] <- (count/nsim)}

plot(n,n\_count)



Q3 : # Assignment 2 , Q3

nsim <- 10000

s<- c(4.8,4.2,5.4,4.1,4,4,4.8,4.4,4.7,4.3,4.4,4.6,4.4,4.4,6.1,4.3,6,4.5,4.4,4.4,4.5,4.2,4.4,4.7,5.4,4,4.6,5.2,4.5,4.4)

s\_mean <- mean(s)

# bootstrapping

s\_bt\_mean <- rep(0,nsim)

for (i in 1:nsim) {

s\_bt <- sample(s,replace=TRUE)

s\_bt\_mean[i] <- mean (s\_bt)}

hist(s\_bt\_mean)

cat ("Standard Error of mean:",s\_mean, "is:", sd(s\_bt\_mean),"\n")

**#RESULT : Standard Error of mean: 4.603333 is: 0.0951238**

# part b

cat ("95% confidence interval is :",quantile(s\_bt\_mean,.025),"::",quantile(s\_bt\_mean,.975),"\n")

**# RESULT : 95% confidence interval is : 4.43 :: 4.803333**

#part c

s\_prop <- 0

for (j in 1 : length(s)){

if ( s[j] > 5.5) {

s\_prop <- s\_prop +1}}

s\_prop <- s\_prop/length(s)

# bootstrapping

s\_bt\_prop <- rep(0,nsim)

for (i in 1:nsim) {

s\_bt\_p <- 0

s\_bt <- sample(s,replace=TRUE)

for (j in 1 : length(s\_bt)){

if ( s\_bt[j] > 5.5) {

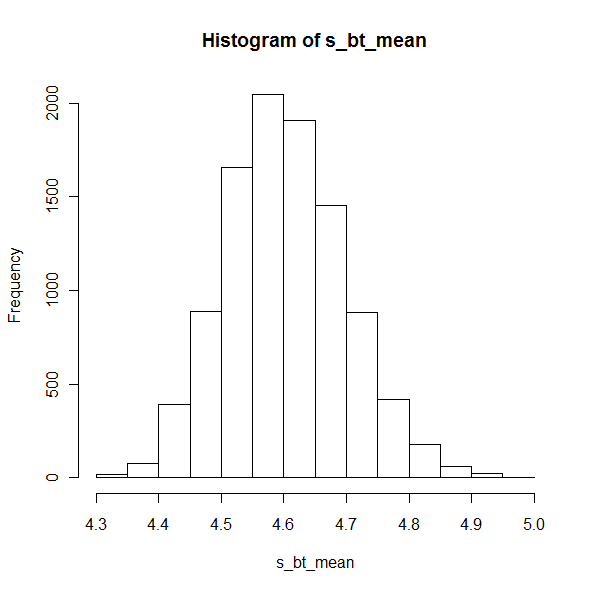
s\_bt\_p <- s\_bt\_p +1}}

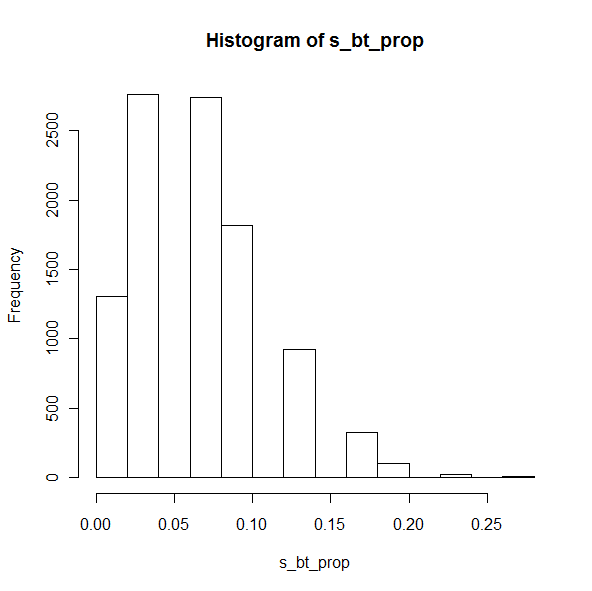
s\_bt\_prop[i] <- s\_bt\_p/length(s)}

hist(s\_bt\_prop)

cat ("Standard Error of mean:",s\_prop, "is:", sd(s\_bt\_prop),"\n")

**# RESULT : Standard Error of mean: 0.06666667 is: 0.04561**





Q4 nsim <- 10000

n<-15

test\_statistic <- rep(0,nsim)

for (i in 1 : nsim) {

# random variates

s <- rcauchy(n,0,1)

# neyman-pearson test\_statistic

l <- ((1+s^2)/(1+((s-1)^2)))

test\_statistic[i] <- prod(l)}

hist(test\_statistic,breaks=200)

cat ("cutoff is :",quantile(test\_statistic,.95),"\n")

# RESULT : cutoff is around 2.3

# part b

count <- 0

for (i in 1 : nsim) {

# random variates with mu =1

s <- rcauchy(n,1,1)

# neyman-pearson test\_statistic

l <- ((1+s^2)/(1+((s-1)^2)))

tstat <- prod(l)

if (tstat > 2.3) {

count <- count +1}}

power <- (count/nsim)\*100

cat ("power of test is:",power,"\n")

**#RESULT : power of the test is 83.6 % ; Since neyman-pearson's is the most powerful test, since the test statistic and threshold for rejection is specifically designed for cauchy (1,1) vs csauchy(0,1).**

# part c

nsim <- 10000

# sample size

n<-15

test\_statistic <- rep(0,nsim)

for (i in 1 : nsim) {

# random variates

s <- rnorm(n,0,1)

# neyman-pearson test\_statistic

l <- ((1+s^2)/(1+((s-1)^2)))

test\_statistic[i] <- prod(l)}

hist(test\_statistic,breaks=200)

cat ("cutoff is :",quantile(test\_statistic,.95),"\n")

# RESULT : cutoff is around 0.76

# part c

count <- 0

for (i in 1 : nsim) {

# random variates with mu =1

s <- rcauchy(n,1,1)

# neyman-pearson test\_statistic

l <- ((1+s^2)/(1+((s-1)^2)))

tstat <- prod(l)

if (tstat > 0.76) {

count <- count +1}}

power <- (count/nsim)\*100

cat ("power of test is:",power,"\n")

**#RESULT : power of the test is 92.04 % ; Since neyman-pearson's is the most powerful test and since cauchy distribution is quite close to normal distribution, with central tendency the test is still more powerful.**

**Q5.** # assignment 2 , q5

nsim <- 1000

# MLE of lamba is 1/x\_bar

wtime <- c(79,54,74,62,85,55,88,85,51,85,54,84,78,47,83,52,62,84,52,79,51,47,78,69,74,83,55,76,78,79)

n<- length(wtime)

s\_lambda <- 1/(mean(wtime))

# standard error se = sqrt(s\_lamdba^2/sample\_size)

# 95% confidence interval is s\_lamba+/- 1.96se

se <- sqrt(s\_lambda^2/n)

cat("95% confidence interval is:", s\_lambda-se, "::",s\_lambda+se,"\n")

cat("95% confidence interval is:", 1/(s\_lambda-se), "::",1/(s\_lambda+se),"\n")

# bootstrap

s\_bt\_mean <- rep(0,nsim)

for (j in 1 : nsim) {

s\_bt <- sample(wtime,replace=TRUE)

s\_bt\_mean[j] <- mean(s\_bt)}

hist(s\_bt\_mean,breaks=200)

cat ("95% confidence interval is:",quantile(s\_bt\_mean,.025),"::",quantile(s\_bt\_mean,.975),"\n")

**#RESULT IS 95% confidence interval is: 64.4 :: 74.26667**

**#part d**

**# COMMENT : previous interval 84.94145 :: 58.71372 , and bootstrap interval is 64.3 :: 74.535**

**# Bootstrapped interval looks tight, but since its non-parametric bootstrap, the actual 95% confidence interval might not be as tight as indicated by this bootstrap, and MLE based range arrived at using lambda, is much reliable.**

**Q6**