```
#*************************************************************************
# Date: 04/01/2021
# Comment: simulated execrise
# Used for: MATH185 - HW2
# Data File: N/A
# Data Source: Monte-Carlo, simulated as given in homework instruction
#*************************************************************************

# Clear all variables and prior sessions
rm(list=ls(all=TRUE))

# Load packages and set macro parameters
library(data.table)
p = 0.3
exp.p = 0.3


#Q1: observed change in moments at given size for binomial----
  #simPar ~ size of binomial draw
simPar = c(1,2,5,10)
for (par in simPar) {
  m=n=par;
  set.seed(simPar^2+1);
  sim = rbinom(m,n,p)/n
  simMean = mean(sim)
  simVar = var(sim)       #var() formula 1/n-1, n = 1
  var.p = (1-p)*p/n
  print(c(simMean,simVar,var.p))
}

#helper method: normal approximated 95 confidence interval----
width.95confi <- function(phat,n){
  w <- 1.96 * sqrt((1-phat)*phat/n);
  return(w)
}

#Q2: Actual and theoretical coverage prob ----
require(plotrix)
  #m: No. of iterations
  #n: binomial parameter for size
m = 100
n = 10
set.seed(1999)
sim2 <- rbinom(m,n,p)/n
  #observed probability
obs1<-data.table(data.frame(phat=sim2))
obs1[,Upper:=phat+width.95confi(phat,n)]
obs1[,Lower:=phat-width.95confi(phat,n)]
obs1[,covered:=(Lower<p&Upper>p)]
obs1[covered==TRUE,.N]/100    #coverage probability with n = 10
  #plot actual CI's for each iteration
plotCI(obs1$phat,ui=obs1$Upper,li=obs1$Lower,ylab="",pch=20)

  #check when binomial parameter n = 100
n = 100;
sim3 = rbinom(m,n,p)/n
obs2 <- data.table(data.frame(phat=sim3))
obs2[,Upper:=phat+width.95confi(phat,n)]
```

```
obs2[,Lower:=phat-width.95confi(phat,n)]
obs2[,covered:=(Lower<p&Upper>p)]
obs2[covered==TRUE,.N]/100  #coverage probability with n = 100
  #plot actual CI
plotCI(obs2$phat,ui=obs2$Upper,li=obs2$Lower,ylab="",pch=20)
#

#Q3: observed power curve ----
  #grid: probability parameter, in vector
grid <- seq(0.1,0.9,by=0.1)
  #bionomial distribution parameter
n = 100;
m = 100;
p = 0.3;
  #simulate Bino observations
success <- rbinom(m,n,p)
sapply(success, prop.test, n=100, p=0.3)
total <- rep(100,100)
p.true <- rep(.3,100)
  #instantiate result datframe
powerCurve <- data.table(data.frame(grid))
powerCurve[,power := -1]

n.total <- rep(n,times=100);
p.true <- rep(.3,100);

  #simulate for changing probability
for (p.sim in grid) {
  success <- rbinom(m,n,p.sim);
  id <- (1:100);
  dt.sim <- data.table(data.frame(success,n.total,p.true,id))
    #run hypothesis test
  dt.sim[,p.test:=(prop.test(success,n.total,p.true))$p.value , by = id]
    #compute actual power
  power.sim = dt.sim[ p.test < 0.05,.N]/100
    #store observed power
  powerCurve[grid == p.sim,power<-power.sim]
}

  #record and plot power curve
#1 0.56 0.04 0.52 1 1 1 1
power1<-c(1,0.56,.04,.52,1,1,1,1,1)
power1C <- data.frame(grid,power1)
plot(power1C,type="l",ylab="power",xlab="p",main="Power Curve:n=100")

  #set sample size to 200 ----
n=200
n.total <- rep(200,times =100)
for (p.sim in grid) {
  #p.sim = 0.1;
  success <- rbinom(m,n,p.sim);
  id <- (1:100);
  dt.sim <- data.table(data.frame(success,n.total,p.true,id))
  dt.sim[,p.test:=(prop.test(success,n.total,p.true))$p.value , by = id]
  power.sim = dt.sim[ p.test < 0.05,.N]/100
  print(power.sim)
  powerCurve[grid == p.sim,power<-power.sim]
}
```

```
  #record and visualize
power2C<- data.frame(grid,c(1,.86,.07,.87,1,1,1,1,1))
plot(power2C,type="l",ylab="power",xlab="p",main="Power Curve:n=200")

  #set sample size n = 50 ----
n=50
n.total <- rep(50,times =100)
for (p.sim in grid) {
  #p.sim = 0.1;
  success <- rbinom(m,n,p.sim);
  id <- (1:100);
  dt.sim <- data.table(data.frame(success,n.total,p.true,id))
  dt.sim[,p.test:=(prop.test(success,n.total,p.true))$p.value , by = id]
  dt.sim
  power.sim = dt.sim[ p.test < 0.05,.N]/100
  print(power.sim)
  #powerCurve[grid == p.sim,power<-power.sim]
}
  #record and visualize
power3C<- data.frame(grid,c(.92,.3,.07,.32,.82,.99,1,1,1))
plot(power3C,type="l",ylab="power",xlab="p",main="Power Curve:n=50")

#helper method: compute CI width----
wid <- function(mid,n){
  return (1.96*sqrt(mid/n))
}

#Q4: MLE for Pois rate----
n=50
set.seed(2020)
sim.pois1 <-rpois(n,lambda = 10)
width.pois <- 1.96*sqrt(mean(sim.pois1)/n)
  #normal approximation
ci1.pois <- rep(mean(sim.pois1),2)+c(-width.pois,+width.pois)
  #variance stablized
ci2.pois <- rep((mean(sim.pois1)+1.96^2/(4*n)),2)+c(-width.pois,
+width.pois)
pois.dt <- data.table(data.frame(idx=rep(1:500)))

pois.dt[,ctr:=mean(rpois(n=50,lambda = 10)),by=idx]
pois.dt[,ctr2:=(ctr+1.96^2/(4*n)),by=idx]
pois.dt[,lb1:=ctr-wid(ctr,50),by=idx]
pois.dt[,ub1:=ctr+wid(ctr,50),by=idx]
pois.dt[,lb2:=ctr2-wid(ctr,50),by=idx]
pois.dt[,ub2:=ctr2+wid(ctr,50),by=idx]
pois.dt[,cover1:=(lb1<10&ub1>10),by=idx]
pois.dt[,cover2:=(lb2<10&ub2>10),by=idx]
mean(pois.dt[,cover1])
mean(pois.dt[,cover2])
```