Statistical Methods for Data Science

Mini Project 3

Members:

SHIVA RANGA CHAWALA	sxc167630
KRISHNA SINDHU KOTA	kxk171030

Contribution:

Shiva Ranga Chawala – Equally contributed

Krishna Sindhu Kota – Equally contributed

1. **Suppose we would like to estimate the parameter ϴ(> 0) of a Uniform (0, ϴ) population based on a random sample X1,X2,…Xn from the population. In the class, we have discussed two estimators for ϴ -- the maximum likelihood estimator, ϴ1 = X(n), where X(n) is the maximum of the sample, and the method of moments estimator, ϴ2 = 2X, where X is the sample mean. The goal of this exercise is to compare the mean squared errors of the two estimators to determine which estimator is better.**
   Recall that the mean squared error of an estimator ϴ_cap of a parameter ϴ is defined as E{(ϴ_cap - ϴ)^2}. For the comparison, we will focus on n = 1; 2; 3; 5; 10; 30 and ϴ = 1; 5; 50; 100.

   a. **Explain how you will compute the mean squared error of an estimator using Monte Carlo simulation.**

      MLE Method:
      Step 1: Generate random values by using runif command
      Step 2: By using MLE method, we find the maximum value of the random values generated which gives theta_hat.
      Step 3: We compute (theta_hat – theta)^2
      Step 4: Repeat/replicate above steps 1000 times
      Step 5: Calculate the mean of all 1000 values to compute MSE.

      MoM Method:
      Step 1: Generate random values by using runif command
      Step 2: By using MoM method, we find 2*(mean) of the random values generated which gives theta_hat.
      Step 3: We compute (theta_hat – theta)^2
      Step 4: Repeat/replicate above steps 1000 times
      Step 5: Calculate the mean of all 1000 values to compute MSE

   b. **For a given combination of (n, ϴ), compute the mean squared errors of both ϴ1_hat and ϴ2_hat using Monte Carlo simulation with N = 1000 replications. Be sure to compute both estimates from the same data.**

      R Code: Using combination of n=30 and ϴ=100
      re1=replicate(1000,(((max(runif(30, min = 0, max = 100))) - 100)^2))
      #generating 30 values from 0 to 100 using runif. Computing max of these 30 values.
      #Calculating the squared error and replicating it 1000 times
      print(paste("MSE by using MLE method",mean(re1))) #calculating mean of squared
      #errors

      *re2=replicate(1000,((2*mean(runif(30, min = 0, max = 100))) - 100)^2)*
      #generating 30 values from 0 to 100 using runif. Computing 2 times mean of these 30
      #values. Calculating the squared error and replicating it 1000 times
      print(paste("MSE by using MoM method",mean(re2))) #calculating mean of squared
      #errors

O/p:
"MSE by using MLE method 19.1162104730239"
"MSE by using MoM method 108.236879531222"

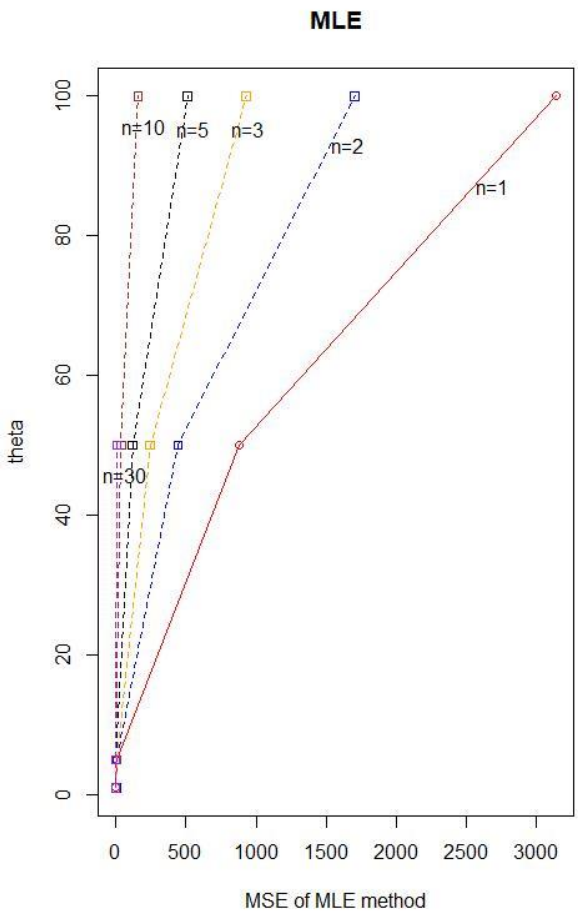
c. **Repeat (b) for the remaining combinations of (n, ϴ). Summarize your results graphically.**

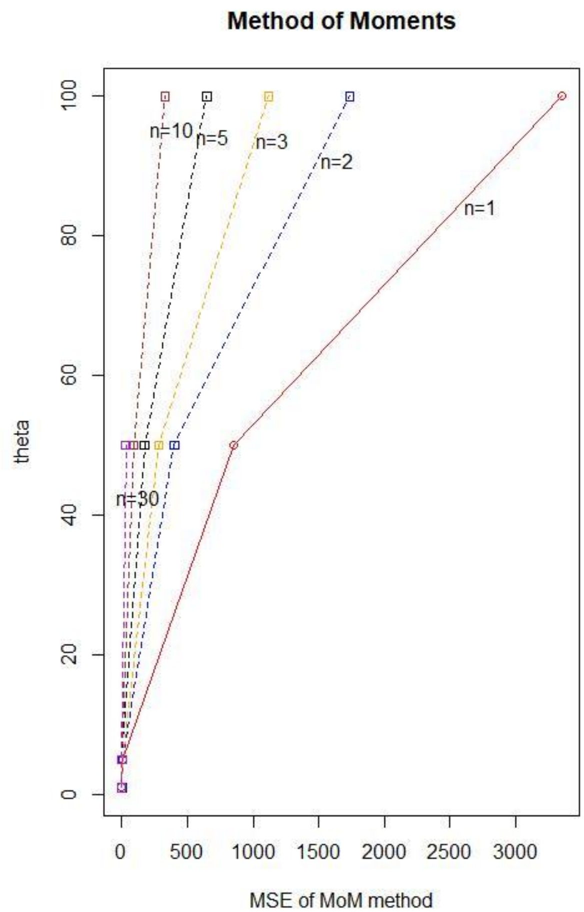
R Code: For remaining combinations

```
theta=c(1,5,50,100) #Storing values of theta
n=c(1,2,3,5,10,30) #Storing values of n
for(i in n){
  for(j in theta){
    if(i!=30 || j!=100) #Excluding combination n = 30, ϴ =100
    {
      re3=replicate(1000,((max(runif(i, min = 0, max =j ))) - j)^2)
      mse1=c(mse1,mean(re3))
      re4=replicate(1000,((2*mean(runif(i, min = 0, max = j))) - j)^2)
      mse2=c(mse2,mean(re4))
    }
  }
}
par(mfrow=c(1,2)) #diving plot frame into 1 by 2 frames
theta_plot = c(1,5,50,100,1,5,50,100,1,5,50,100,1,5,50,100,1,5,50,100)

plot(mse1[1:4], theta_plot[1:4], type = "o", col="red", axes = TRUE, ann =
TRUE,main="MLE", xlab = "MSE of MLE method", ylab = "theta")
#plotting MSE of MLE on X-axis vs ϴ on Y-axis for n=1

lines(mse1[5:8], theta_plot[5:8],type="o", pch=22, lty=2, col="blue")
#Adding line to same plot for n=2
lines(mse1[9:12], theta_plot[5:8],type="o", pch=22, lty=2, col="orange")
#Adding line to same plot for n=3
lines(mse1[13:16], theta_plot[5:8],type="o", pch=22, lty=2, col="black")
#Adding line to same plot for n=5
lines(mse1[17:20], theta_plot[5:8],type="o", pch=22, lty=2, col="brown")
#Adding line to same plot for n=10
lines(mse1[21:23], theta_plot[5:7],type="o", pch=22, lty=2, col="purple")
#Adding line to same plot for n=30
text(locator(), labels = c("n=1", "n=2","n=3","n=5","n=10","n=30"))
#Used for adding labels to the lines in the plot

plot(mse2[1:4], theta_plot[1:4], type = "o", col="red", axes = TRUE, ann =
TRUE,main="Method of Moments",  xlab = "MSE of MoM method", ylab = "theta")
#plotting MSE of MoM on X-axis vs ϴ on Y-axis for n=1

lines(mse2[5:8], theta_plot[5:8],type="o", pch=22, lty=2, col="blue")
```

```
lines(mse2[9:12], theta_plot[5:8],type="o", pch=22, lty=2, col="orange")
lines(mse2[13:16], theta_plot[5:8],type="o", pch=22, lty=2, col="black")
lines(mse2[17:20], theta_plot[5:8],type="o", pch=22, lty=2, col="brown")
lines(mse2[21:23], theta_plot[5:7],type="o", pch=22, lty=2, col="purple")
text(locator(), labels = c("n=1", "n=2","n=3","n=5","n=10","n=30"))
```

O/p:





d. Based on (c), which estimator is better? Does the answer depend on n or ϴ? Explain. Provide justification for all your conclusions.

From the above plot, we have observed that the MSE of either of the estimators is directly proportional to theta and is indirectly proportional to n. With the increase of n, the MSE decreases since there is more accuracy. MLE seems more accurate since MSE of MLE is lesser than that of MoM. As theta increases the difference between them also increases.

2. **f(x) =(Ө/x^(Ө + 1) → x>=1 ; 0 → x < 1;)**

   a. **Derive an expression for maximum likelihood estimator of Ө**

   b. **Suppose n = 4 and the sample values are x1 = 4:79; x2 = 10:89; x3 = 6:54; x4 = 22:15. Use the expression in (a) to provide the maximum likelihood estimate for Ө based on these data.**

2.

$$f(x) = \begin{cases} \frac{\theta}{x^{\theta+1}} & x \geq 1, \\ 0 & x < 1 \end{cases}$$

a) $L(\theta) = \prod_{i=1}^{n} f_\theta(x_i)$

$= \prod_{i=1}^{n} \frac{\theta}{x^{\theta+1}}$

$= \theta^n \left(\prod_{i=1}^{n} \frac{1}{x_i} \right)^{\theta+1}$

Taking logarithm of above gives log-likelihood function as

$\log\{L(\theta)\} = n\log(\theta) + (\theta+1)\sum_{i=1}^{n} \log\left(\frac{1}{x_i}\right)$

Differentiating the above with respect to θ and setting the derivative to zero gives likelihood equation

$0 = \frac{\partial}{\partial\theta} \log\{L(\theta)\} = \frac{n}{\theta} + \sum_{i=1}^{n} \log\left(\frac{1}{x_i}\right)$

Solving this equation gives MLE of θ

$\frac{n}{\theta} = -\sum_{i=1}^{n} \log\left(\frac{1}{n_i}\right)$

$$\boxed{\hat{\theta} = \frac{-n}{\sum_{i=1}^{n} \log\left(\frac{1}{x_i}\right)}}$$

b) Plugging $n=4$ and $x_1 = 4.79$, $x_2 = 10.89$, $x_3 = 6.54$, $x_4 = 22.15$

$\hat{\theta} = \frac{-4}{\log\left(\frac{1}{4.79}\right) + \log\left(\frac{1}{10.89}\right) + \log\left(\frac{1}{6.54}\right) + \log\left(\frac{1}{22.15}\right)}$

$= \frac{-4}{-1.5664 - 2.3881 - 1.8779 - 3.0989}$

$= \frac{-4}{-8.9313}$

$\hat{\theta} = 0.4479$

c. **Estimate by numerically maximizing the log-likelihood function using optim function in R. Do your answers match?**

R Code:

```
data = c(4.79, 10.89, 6.54,22.15)  #storing the values of x1,x2,x3,x4 in data.
# Negative of log-likelihood function
fun = function(theta, x) {
  result = sum(log(theta / (x^(theta + 1))))
  return(-result)
}
#Estimate theta by the MLE method
ml.est = optim(par = 1, fn = fun, method = "BFGS", hessian = TRUE, x = data)
#optim function is used for maximizing the log-likelihood function
#BFGS is a quasi-Newton method method and uses function values and gradients to
#build up a picture of the surface to be optimized.
mle = ml.est$par #$par variable will contain the best value of theta_cap where our
#function maximizes
print(paste("Maximum Likelihood Estimator for theta is = ",mle))
```

Yes, MLE values of theoretical and by using optim function are matching.

O/p:

Maximum Likelihood Estimator for theta is = 0.44792081662746

d. **Use the output of numerical maximization in (c) to provide approximate standard error of the maximum likelihood estimate and an approximate 95% confidence interval for ϴ. Are these approximations going to be good? Justify your answer.**

R Code:

```
se.mle = sqrt(1/ml.est$hessian) #Computing & storing estimated standard error in
#se.mle
print(paste("Standard Error of Maximum Likelihood Estimator for theta is =
",se.mle))

alpha <- 1-0.95 #α value for 95% Confidence Interval
n <- length(data) #Storing length of data in 'n'
upperCI <- mle +qt(1-(alpha/2),(n-1)) * se.mle
lowerCI <- mle - qt(1-(alpha/2),(n-1)) * se.mle #Calculating upper and lower limits
#of 95% CI using qt
print(paste("Upper Limit is = ",upperCI))
print(paste("Lower Limit is = ",lowerCI))
```

O/p:

"Standard Error of Maximum Likelihood Estimator for theta is = 0.22395929203646"
"Upper Limit is = 1.16065923810285"
"Lower Limit is = -0.264817604847927"

Yes, the approximations are going to be good.
Because Maximum Likelihood Estimator for theta is 0.448 and falls in 95% CI.