**Mini Project 3**

**Name :**

**Dhwaniben Kaneria (drk170130)**

**Utkarsh Gandhi (usg170030)**

**Contribution of team members:**

Dhwani:

* Wrote equation and calculation in word
* Drew a graph about simulation and found the way to distribute it
* Learned R Coding
* Tried different R codes
* Wrote narration for codes

Utkarsh:

* Did calculation on paper
* Explained the simulation
* Learned R coding
* Did debugging for R codes
* Derived conclusions from graphs and calculations

1. **(a)**

The mean square error of an estimator is defined as

MSE(θˆ) = E[(θ̂ − θ)2]

= Var( θ̂ ) + E[( θ̂ )- θ]

= Var( θ̂ ) + [Bias( θ̂ )]2

where θ̂ is an estimator of θ, an unknown population parameter.

The bias of an estimator θ̂ of a parameter θ is the difference between the expected value of θ̂ and θ; that is,

Bias( θ̂ ) = E( θ̂ ) − θ.

If E( θ̂ ) = θ for all θ then the estimator is unbiased.

* Thus, MSE has two components, one measures the variability of the estimator (precision) and the other measures the its bias (accuracy).
* An estimator that has good MSE properties has small combined variance and bias. To find an estimator with good MSE properties we need to find estimators that control both variance and bias.

For an unbiased estimator θ̂, we have

MSE( θ̂ ) =E(θ̂ − θ)2 =Var( θ̂ )

and so, if an estimator is unbiased, its MSE is equal to its variance.

If E(θ̂) is not equal to θ then the estimator has either a positive or negative bias. That is, on average the estimator tends to over (or under) estimate the population parameter.

Given the maximum likelihood estimator θ1̂=X(n) and the method of moments estimator, θ2̂= 2x̅, where x̅ is the sample mean.

To compare these two estimators, by Monte Carlo simulation for a specific n and θ:

1. Generate X1, ..., Xn ∼ Uniform(0, θ)

2. Calculate θ̂1 and θ2̂

3. Save (θ1̂ − θ)2 and (θ2̂ − θ)2

4. Repeat step 1-3 N times

5. Then the means of the (θ1̂ −θ)2 and (θ2̂ −θ)2, over the N replicates, are the monte carlo estimators of the MSEs of θ1̂ and θ2̂.

**(b)**

Let N= 1000

n=1 and θ=1 ,5,50,100

* First, we will generate numbers from 0 to θ and then we will calculate values of θ̂1 and θ2̂ and then we will compute mean square error for both the estimators. This step will be done for N times for performing Monte Carlo simulation.

**#R-code:**

1. theta <- c(1,5,50,100)
2. #Creating matrix to store the MSEs of both the estimators
3. mse <- matrix(0, length(theta), 2)
4. #Loop through values in theta
5. for(i in 1:length(theta))
6. {

#Generating 1000\*n random numbers which are uniformly distributed between 0 to theta. Here n=1

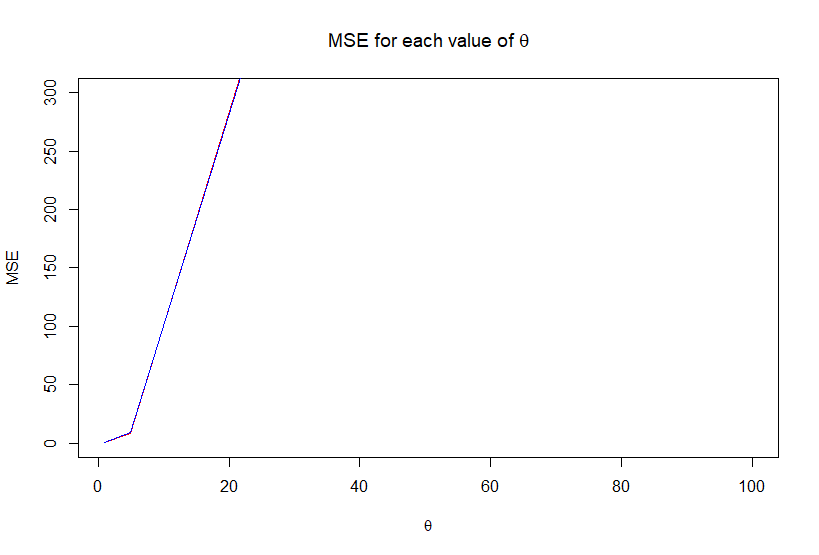
1. data <- matrix(runif(1000\*1,0,theta[i]),1000,1)
2. data
3. #Calculate theta hat\_1(Maximum Likelihood estimator) for each value of data
4. t\_Hat\_1 <- apply(data, 1, max)
5. t\_Hat\_1
6. # Calculate theta hat\_2 (Method of moment) for each value of data
7. t\_Hat\_2 <- 2\*apply(data, 1, mean)
8. t\_Hat\_2
9. # Computing the mse values
10. mse[i,1] <- mean((t\_Hat\_1 - theta[i])^2)
11. mse[i,2] <- mean((t\_Hat\_2 - theta[i])^2)

} # for loop ends

1. # Plot the results on the same axes
2. plot(theta, mse[,1], xlab=quote(theta), ylab="MSE",main=expression(paste("MSE for each value of ", theta)),type="l", col="red",ylim = c(0,300))
3. lines(theta, mse[,2], col="blue")

# code ends

For n=1



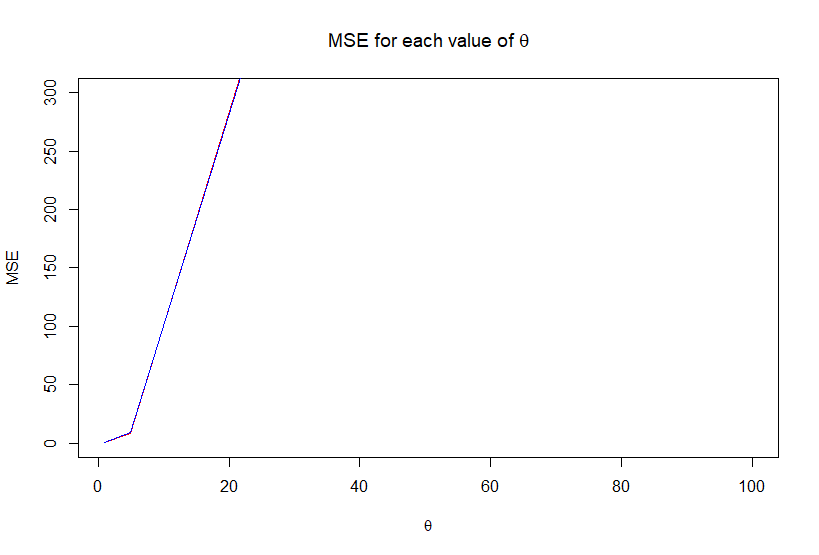
**(c)**

**#R-code:**

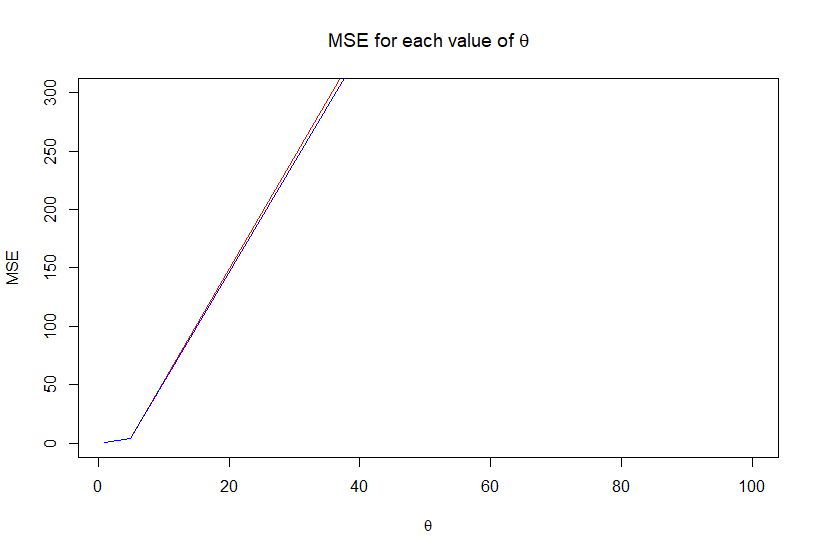
1. n<-c(1,2,3,5,10,30)
2. theta <- c(1,5,50,100)
3. #Creating matrix to store the MSEs of both the estimators
4. mse <- matrix(0, length(theta), 2)
5. #Loop through values in N
6. for (j in 1:length(n)) #for loop starts
7. {
8. #Loop through values in theta
9. for(i in 1:length(theta))
10. {

#Generating 1000\*n random numbers which are uniformly distributed between 0 to theta

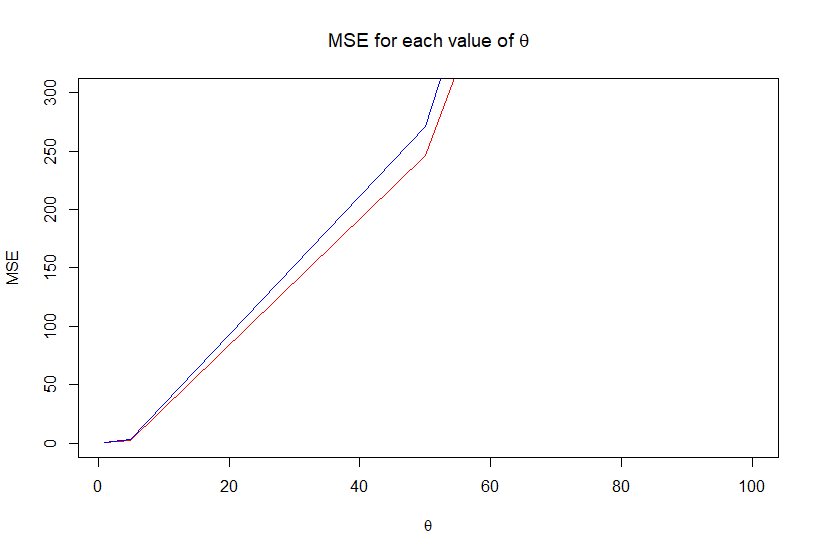
1. data <- matrix(runif(1000\*j,0,theta[i]),1000,j)
2. data
3. # Calculate theta hat\_1(Maximum Likelihood estimator) for each value of data
4. t\_Hat\_1 <- apply(data, 1, max)
5. t\_Hat\_1
6. # Calculate theta hat\_2 (Method of moment) for each value of data
7. t\_Hat\_2 <- 2\*apply(data, 1, mean)
8. t\_Hat\_2
9. # Save the MSEs
10. mse[i,1] <- mean((t\_Hat\_1 - theta[i])^2)
11. mse[i,2] <- mean((t\_Hat\_2 - theta[i])^2)
12. } # second for loop ends
13. # Plot the results on the same axes
14. plot(theta, mse[,1], xlab=quote(theta), ylab="MSE",main=expression(paste("MSE for each value of ", theta)),type="l", col="red",ylim = c(0,300))
15. lines(theta, mse[,2], col="blue")
16. } #first for loop ends
17. **For n=1**

****

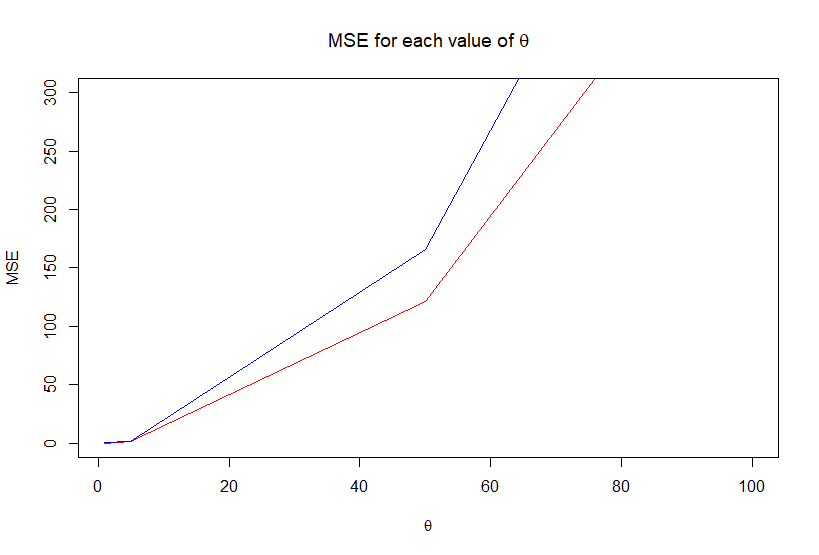
1. **For n=2**



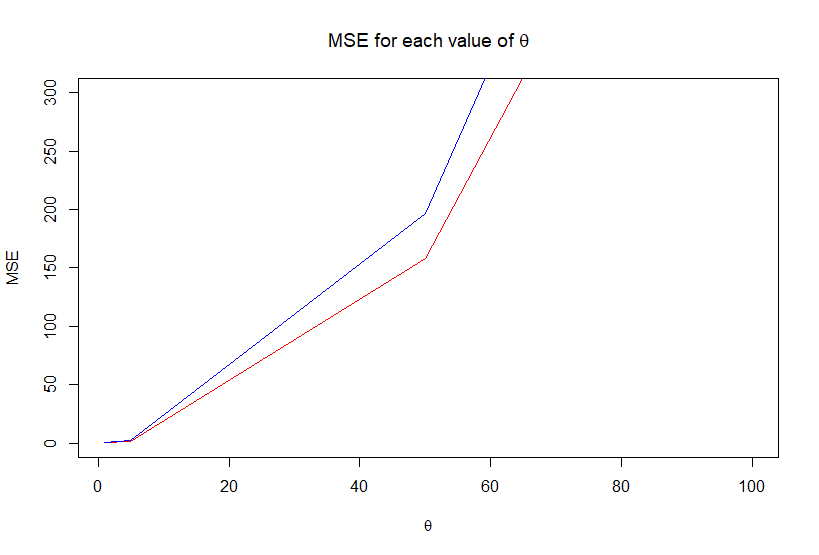
1. **For n=3**



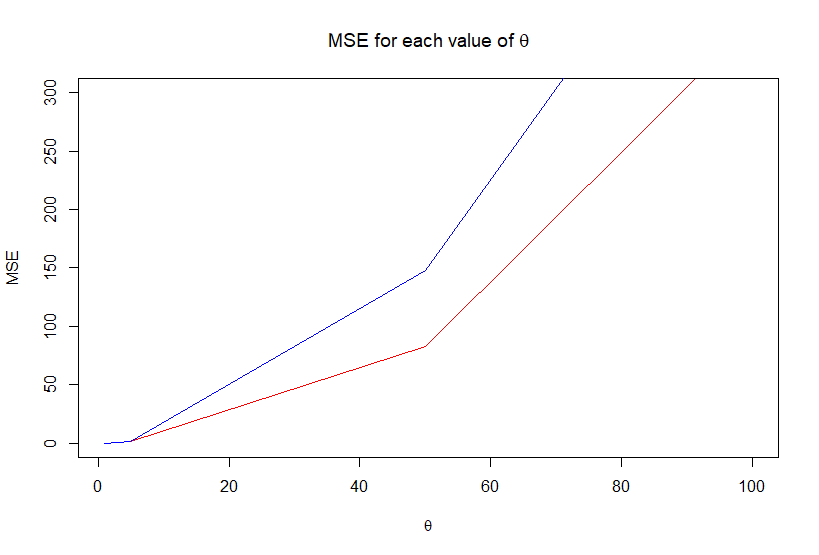
1. **For n=5**



1. **For n=10**

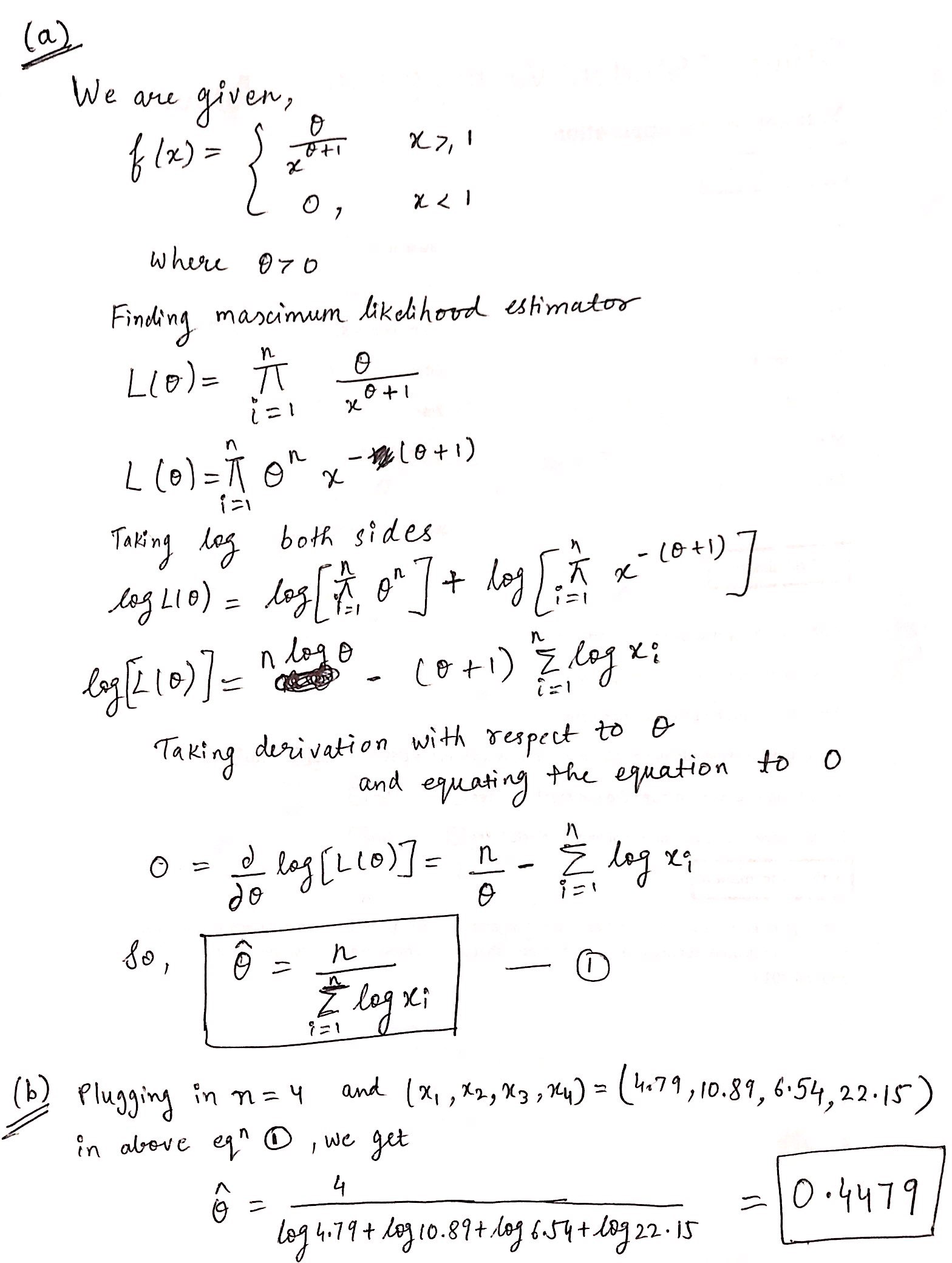


1. **For n=30**



**(d)**

* From 1.(b) we can see that for n=1, MSE values for θ̂1  and θ2̂ are almost same. So we can say that for small value of n, the efficiency of both the maximum likelihood estimator and method of moments estimator are the same. As the θ increase with same n value, MSE values of both estimators are also increasing with MSE of θ2̂ greater than MSE of θ̂1. As θ increase MLE becomes more efficient than MoM estimator.
* Moreover, from 1.(c) we can see that, for n=2, MSE value of both θ̂1 and θ2̂ are lower for θ=1 compare to θ=100. So we can say that as θ increase for fixed N value, MSE values of both estimators are also increasing with MSE of θ2̂ greater than MSE of θ̂1 . If we compare this result to n=1, we can see that MSE values of both θ̂1 and θ2̂ are less in the case of n=2. This implies that as n increase, MSE decreases and also for larger values of n, θ1̂ and θ2̂ gives more accurate estimation than for smaller values of n.
* Similarly, we can see that n=30 and θ=1, the MSE values of both θ̂1 and θ2̂ are lower compared to the θ̂1 and θ2̂ values when θ = 100. So we can see that as θ increases for a fixed n,ie here n=30, the MSE values of both the estimators are increasing with MSE of θ2̂ greater than MSE of θ̂1.
* So we can see that for small value of θ, the efficiency of both the maximum likelihood estimator and the method of moments estimator are the same. As θ increases MLE becomes more efficient than Method Of Moments estimator. As when we compared with n=1,2,3,5,10 MSE of both θ̂1 and θ2̂ with n=30, we observed that both estimator are less in n=30. This shows that as n increases, MSE decreases and also for larger values of n, θ̂1 and θ2̂ gives more accurate estimation than for smaller values of n.
* In conclusion from the graph, we can say that θ̂1 has low MSE than θ2̂ for higher values of n and θ. So, θ̂1 is better estimator than θ2̂ . For smaller values of n and θ, the efficiency of both estimators are the same.
* As θ is increasing, MSE values for both estimator are increasing and for higher θ, MSE for Maximum likelihood estimator is less than MSE of Method of Moments estimator.

**2) **

**(c)**

1. #input the data
2. x=c(4.79,10.89,6.54,22.15) # taking values of x
3. # Negative of log-likelihood function
4. fnMLE<- function(theta,data)
5. {
6. n=length(data)
7. # computing f(x) for value of parameter theta and data x
8. res<- (n\*log(theta))-((theta+1)\*sum(log(data)))
9. return(-res)
10. }
11. # estimate theta by MLE method
12. ml.est<-optim(par=1,fn=fnMLE,method="BFGS",hessian = TRUE,data=x)
13. # NOTE: par=1 is being used as starting point for optimization
14. ml.est$par # gives the estimate of theta

**Output**

> ml.est$par

[1] 0.4479208

Yes, the answers do match as we got same by computing

**(d)**

# calculating standard error by hessian

se<-sqrt(diag(solve(ml.est$hessian)))

> se

[1] 0.2239593

So **standard error is 0.223**

# As n is not normal and n is not large ,taking t

n <- length(x)

df <- n - 1

ml.est$par + c(-1,1) \* qt(1-(1-0.95)/2, df) \* se

> ml.est$par + c(-1,1) \* qt(1-(1-0.95)/2, df) \* se

[1] -0.2648176 1.1606592

Here upper bound is 1.1606592 and lower bound is -0.2648176.

Yes the approximations are good, because our estimated parameter value in the given range of confidence interval.