Probability and Statistics with R

Assignment 2

Submission Nov 16-2022 (Wednesday)

Note: Below I explain how you collaborate on `GitHub` .

1. It will be group assignment.
2. A group would be of size at most 3. If you want to create a group size more than 3, you must take permission.
3. Decide among yourself and one of you create a GitHub repository for Probability Statistics Assignments.
4. In that repository add your group members as collaborator
5. Once you add your collaborator (or group members), create a folder and name it as `Assignment_2`
6. In that folder you should have 2 folders `code` and `report` . And one `README.md` file. Write a brief report in `README.md` file.
7. For each problem, you should create a separate GitHub `issue` . All your discussion should be documented in the `issue` .
8. In the issue mention clearly, which group member is taking ownership of what problem?
9. The other member should `fork` the repository in their GitHub account.
10. Once you have your forked the main repository in your GitHub account - you should clone the repository in you local laptop or just download it as zip.
11. Once you develop the code - you should `commit` the code first in your repository and then `push` it.
12. Finally you make the `pull-request` in the final repository.
13. Once a member make a pull request, the other members have to review the code.
14. While reviewing the code the reviewer may have to download the code and run the code in his or her system and reproduce the result.
15. If the result is reproduced then she or he would accept and merge the code in final repository.
16. At the end you submit the link of the repository in the moodle.
17. The entire process will be evaluated.

Problem 1

Suppose X denote the number of goals scored by home team in premier league. We can assume X is a random variable. Then we have to build the probability distribution to model the probability of number of goals. Since X takes value in $\mathbb{N} = \{0, 1, 2, \cdots\}$, we can consider the geometric progression sequence as possible candidate model, i.e.,

$$S = \{a, ar, ar^2, ar^3, \cdots\}.$$

But we have to be careful and put proper conditions in place and modify S in such a way so that it becomes proper probability distributions.

1. Figure out the necesary conditions and define the probability distribution model using S.
2. Check if mean and variance exists for the probability model.
3. Can you find the analytically expression of mean and variance.
4. From historical data we found the following summary statistics Using the summary statistics and your newly defined probability distribution model find the following:

a. What is the probability that home team will score at least one goal?
b. What is the probability that home team will score at least one goal but less than four goal?

5. Suppose on another thought you want to model it with off-the shelf Poisson probability models. Under the assumption that underlying distribution is Poisson probability find the above probabilities, i.e.,

a. What is the probability that home team will score at least one goal?
b. What is the probability that home team will score at least one goal but less than four goal?

6. Which probability model you would prefer over another?
7. Write down the likelihood functions of your newly defined probability models and Poisson models. Clearly mention all the assumptions that you are making.

1.We are modelling the number of goals scored by the home team in a premier league match using S the geometric progression sequence.We have made the following conditions and definitions regarding the model. Let a to associated with an hypothetical situation after which no goal is going to be scored and r that a goal happens.We assume $r > 0$ and $a > 0$ For example: consider outcome of a match to be ra, this means that in that match after the home team scored a goal it didn't score any more goals.The outcome only a would represent no goal was scored in the match by the home team. The distribution would be

$$f(X = n) = r^n a$$

For every n the the value is positive.For this to be a probability function sum of all the probabilities should be 1.

$$\sum_{n=0}^{inf} f(X = n) = 1$$

$$\sum_{n=0}^{inf} r^n a = 1$$

$$a \sum_{n=0}^{inf} r^n = 1$$

This series converges only when $|r| < 1$. Therefore we have $0 < r < 1$ (associated to probability of scoring a goal)

$$\frac{a}{1-r} = 1$$
$$a = 1 - r$$

Therefore the probability distribution would be

$$P(X = n) = r^n(1 - r) \quad n = 0, 1, 2, \cdots$$

where n is the number of goals scored by the home team in that match.

2. We need to show that mean and variance exists for the new distribution we have defined.To show mean exists we need to show that $E[|X|]$ exist, since X takes positive values E[|X|] is nothing but E[X].

$$E[X] = \sum_{n=0}^{inf} nP(X=n)$$
$$= \sum_{n=0}^{inf} nr^n(1-r)$$
$$= (1-r)\sum_{n=0}^{inf} nr^n$$
$$(\ \ wkt \quad \sum_{n=0}^{inf} nr^n = \frac{r}{(1-r)^2}\ \ )$$
$$= (1-r)\frac{r}{(1-r)^2} = \frac{r}{(1-r)}$$

Therefore mean exists for the new distribution, we can similarly show that $E[|X^2|]$ (which is equal to the E[X^{2}]) exists and is equal to

$$E[X^2] = \frac{r^2+r}{(1-r)^2}$$

Hence variance also exists as

$$var[X] = E[X^2] - (E[X])^2$$

and it is equal to

$$Var[X] = \frac{r^2+r}{(1-r)^2} \ - \ (\frac{r}{(1-r)})^2$$
$$= \quad \frac{r}{(1-r)^2}$$

Therefore mean and variance exists for the new distribution.

3. The analytical form of mean and variance is

$$E[X] = \frac{r}{(1-r)} \quad Var[X] = \frac{r}{(1-r)^2}$$

4. Historical data gives us that the mean number of goals scored by the home team (based on past 380 matches) is 1.5. We equate this to our theoretical mean and calculate the value.

$$\frac{r}{1-r} = \frac{3}{2}$$
$$2r = 3 - 3r$$
$$r = \frac{3}{5}$$

Therefore our probability distribution is

$$P(X=n) = \frac{2}{5}(\frac{3}{5})^n$$

5.

a. Probability of at least one goal.

$$P(X \geq 1) = \sum_{n=1}^{inf} \frac{2}{5}(\frac{3}{5})^n$$

$$= \frac{2}{5}\sum_{n=1}^{inf}(\frac{3}{5})^n = \frac{2}{5}\frac{\frac{3}{5}}{1-\frac{3}{5}} = \frac{3}{5}$$

This is similar to $ 1- P(X=0) = 1- = $ b. The probability that home team will score at least one goal but less than four goal

$$P(1 \leq X < 4) = \sum_{n=1}^{3} \frac{2}{5}(\frac{3}{5})^n$$

$$= \frac{2}{5}\sum_{n=1}^{3}(\frac{3}{5})^n = \frac{2}{5}(\frac{3}{5} + \frac{9}{25} + \frac{27}{125}) = 0.4704$$

We know that this new model is geometric distribution with parameter $p = \frac{3}{5}$. We can use inbuilt functions to calculate these probabilities.

```
## a. probability of atleast one goal
P_a= 1- dgeom(0,2/5)
## a. probability of atleast one goal and less than 4 goals
P_b= sum(dgeom(c(1,2,3),2/5))
P_a
```

```
## [1] 0.6
```

```
P_b
```

```
## [1] 0.4704
```

5. We assume that the underlying distribution for the number of goals scored by the home team is poisson distribution.Let random variable $Y \sim Pois(\lambda)$. We know that $E[Y] = \lambda$ and from historical data we know the mean is 1.5. Hence $Y \sim Pois(\lambda = 1.5)$. We calculate the probability using inbuilt function.

```
## a. probability of at least one goal
P_a= 1- dpois(0,1.5)

## b. probability of at least one goal and less than 4 goals
P_b= sum(dpois(c(1,2,3),1.5))


P_a
```

```
## [1] 0.7768698
```

```
P_b
```

```
## [1] 0.7112274
```

6. Poisson distribution is generally used for counting the occurrence of certain event in a fixed amount of time. (Ex. Poisson distribution is used to model number of cars passing the tollgate etc). Therefore we would prefer poisson distribution as it is a better model for fitting the number of goals scored by the home team in a match (fixed time - 90 minutes) compared to geometric distribution.

7. Maximum Likelihood function: Consider $x_1, x_2 \cdots, x_n$ be a sample drawn from the theoretical distribution

a. New distribution - $P(X = n) = (1 - r)(r)^n$ The likelihood function for the following distribution is :-

$$\begin{aligned} L(r_\theta|x_1, x_2 \cdots, x_n) &= P(x_1, x_2 \cdots, x_n|r_\theta) \\ &= P(x_1|r_\theta).\, P(x_2|r_\theta) \cdots P(x_n|r_\theta) \\ &= (1 - r_\theta)(r_\theta)^{x_1}.\, (1 - r_\theta)(r_\theta)^{x_1} \cdots (1 - r_\theta)(r_\theta)^{x_n} \\ &= (1 - r_\theta)^n.\, (r_\theta)^{\sum_{i=1}^n x_i} \end{aligned}$$

b. Poisson Distribution - $P(X = n) = \frac{e^{-\lambda}\lambda^n}{n!}$ The likelihood function of the following distribution is

$$\begin{aligned} L(\lambda_\theta|x_1, x_2 \cdots, x_n) &= P(x_1, x_2 \cdots, x_n|\lambda_\theta) \\ &= P(x_1|\lambda_\theta).\, P(x_2|\lambda_\theta) \cdots P(x_n|\lambda_\theta) \\ &= \frac{e^{-\lambda}\lambda^{x_1}}{x_1!} . \frac{e^{-\lambda}\lambda^{x_2}}{x_2!} \cdots \frac{e^{-\lambda}\lambda^{x_n}}{x_n!} \\ &= \frac{e^{-n\lambda}\lambda^{\sum_{i=1}^n x_i}}{x_1!.\, x_2! \cdots x_n!} \end{aligned}$$

One main assumption that we make in both the likelihood functions is that the sample is independently and identically drawn from their respective distributions. Here independence means that the goals scored in one match doesn't affect the goals scored in another match. We assume that each match is identical to the other matches (that is there is no knockout matches or a club rivalry match), this ensures that the distribution followed in each match is identical.

Problem 2 : Simulation Study to Understand Sampling Distribution

Part A Suppose $X_1, X_2, \cdots, X_n \overset{iid}{\sim} Gamma(\alpha, \sigma)$, with pdf as

$$f(x|\alpha, \sigma) = \frac{1}{\sigma^\alpha \Gamma(\alpha)} e^{-x/\sigma} x^{\alpha-1}, \quad 0 < x < \infty,$$

The mean and variance are $E(X) = \alpha\sigma$ and $Var(X) = \alpha\sigma^2$. Note that `shape =` α and `scale =` σ.

1. Write a `function` in `R` which will compute the MLE of $\theta = \log(\alpha)$ using `optim` function in `R`. You can name it `MyMLE`
2. Choose `n=20`, and `alpha=1.5` and `sigma=2.2`
   i. Simulate $\{X_1, X_2, \cdots, X_n\}$ from `rgamma(n=20,shape=1.5,scale=2.2)`
   ii. Apply the `MyMLE` to estimate θ and append the value in a vector
   iii. Repeat the step (i) and (ii) 1000 times
   iv. Draw histogram of the estimated MLEs of θ.
   v. Draw a vertical line using `abline` function at the true value of θ.
   vi. Use `quantile` function on estimated θ's to find the 2.5 and 97.5-percentile points.
3. Choose `n=40`, and `alpha=1.5` and repeat the (2).
4. Choose `n=100`, and `alpha=1.5` and repeat the (2).
5. Check if the gap between 2.5 and 97.5-percentile points are shrinking as sample size `n` is increasing?

Hint: Perhaps you should think of writing a single `function` where you will provide the values of `n`, `sim_size`, `alpha` and `sigma`; and it will return the desired output.

```
MyMLE = function(data_obtained,sigma)
{
  LogLike = function(initial = 1.5 ,data_obtained)
  {
    alphatemp = exp(initial)
    n = length(data_obtained)
    logl = sum(dgamma(data_obtained,
                  shape = alphatemp,scale=sigma,log = TRUE))
    return(logl)
  }
  fit = optimise(f = LogLike,interval = c(0,5),
                 maximum = T,data_obtained=data_obtained)
  fit$maximum
}
```

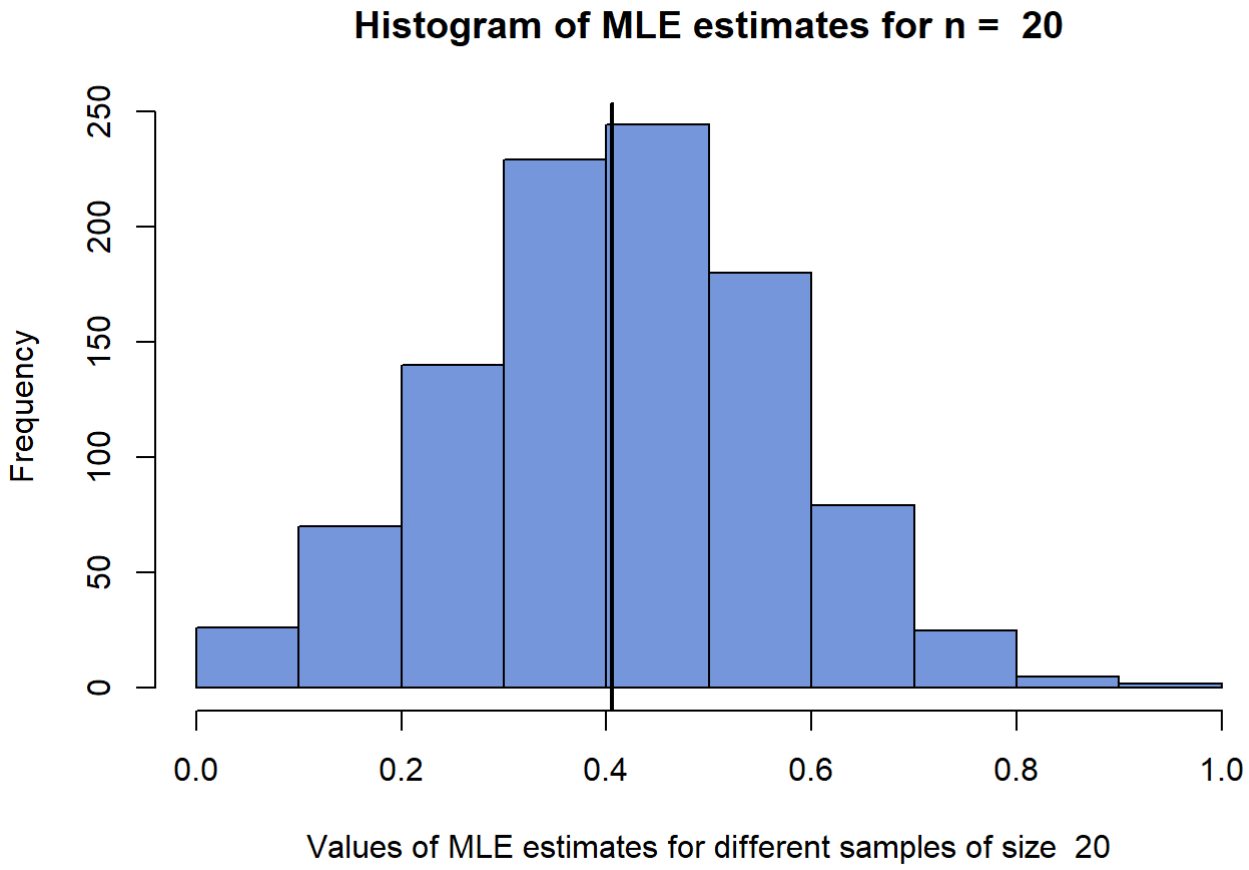
We are now going to write a function called ‘result function’ which takes values for alpha, sigma, n and simulation size as input and outputs the histogram of MLE estimates as well as the Interquartile range for the selected value of n.

```
resultfunction = function(alpha,sigma,n,sim_size)
{
  mles = c(rep(0,sim_size))
  for(i in 1:sim_size)
  {
    dat = rgamma(n,shape=alpha,scale = sigma)
    mles[i] = MyMLE(data_obtained = dat,sigma = sigma)
  }
  library(ggplot2)
  library(randomcoloR)

  hist(mles,col=randomColor(),
       main=paste('Histogram of MLE estimates for n = ',n),
       xlab = paste('Values of MLE estimates for different samples of size ',n),ylab = 'Frequ
ency')

  abline(v=log(alpha),lwd=2,col='black')

  IQR = quantile(mles,probs=0.975) - quantile(mles,probs=0.025)

  print(paste('Inter quantile range for n = ',n,'is:',round(IQR,2)))
}
```

Further we are going to examine the sampling distribution of MLEs for different values of n.

Choose n = 20:

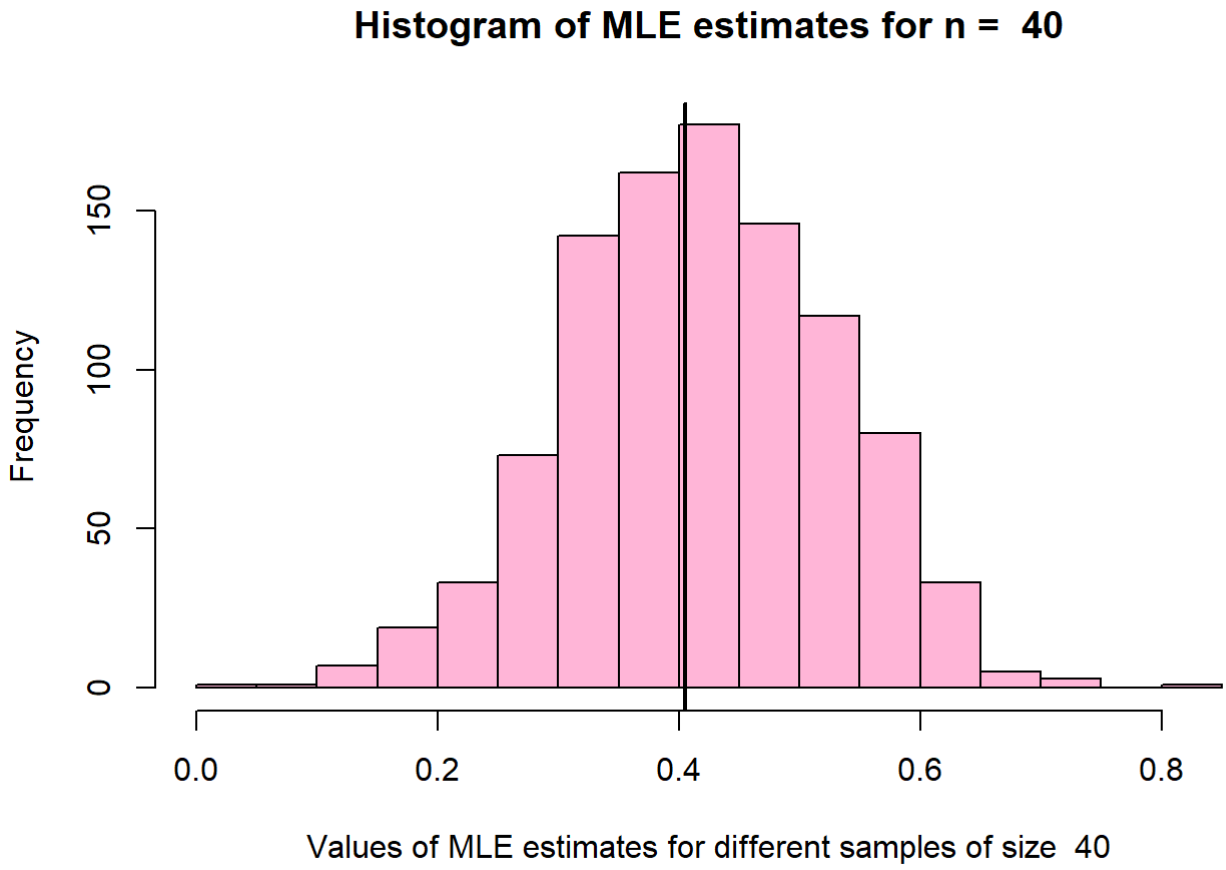
```
resultfunction(alpha = 1.5,sigma = 2.2,
               n = 20,sim_size = 1000)
```




```
## [1] "Inter quantile range for n =  20 is: 0.62"
```

Choose n = 40

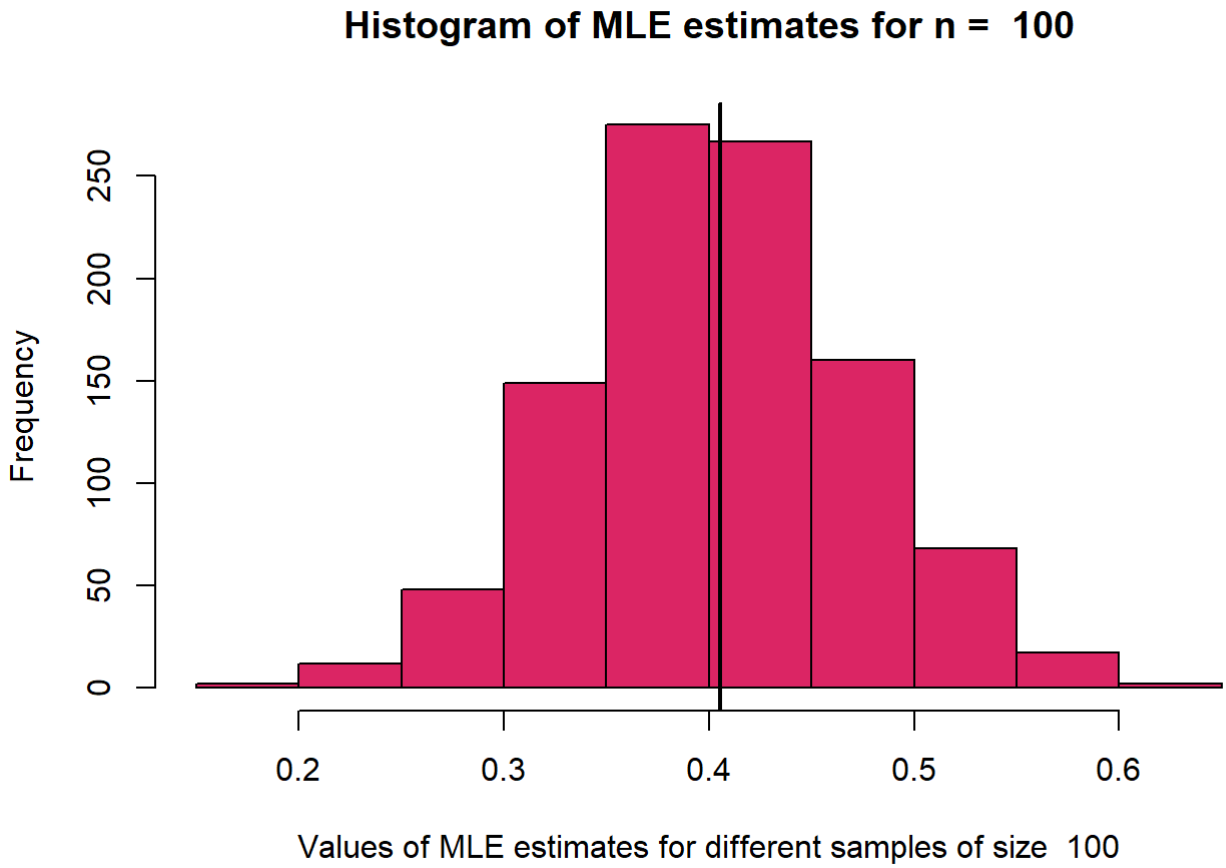
```
resultfunction(alpha = 1.5,sigma = 2.2,
               n = 40,sim_size = 1000)
```




```
## [1] "Inter quantile range for n =  40 is: 0.43"
```

Choose n = 100

```
resultfunction(alpha = 1.5,sigma = 2.2,
               n = 100,sim_size = 1000)
```



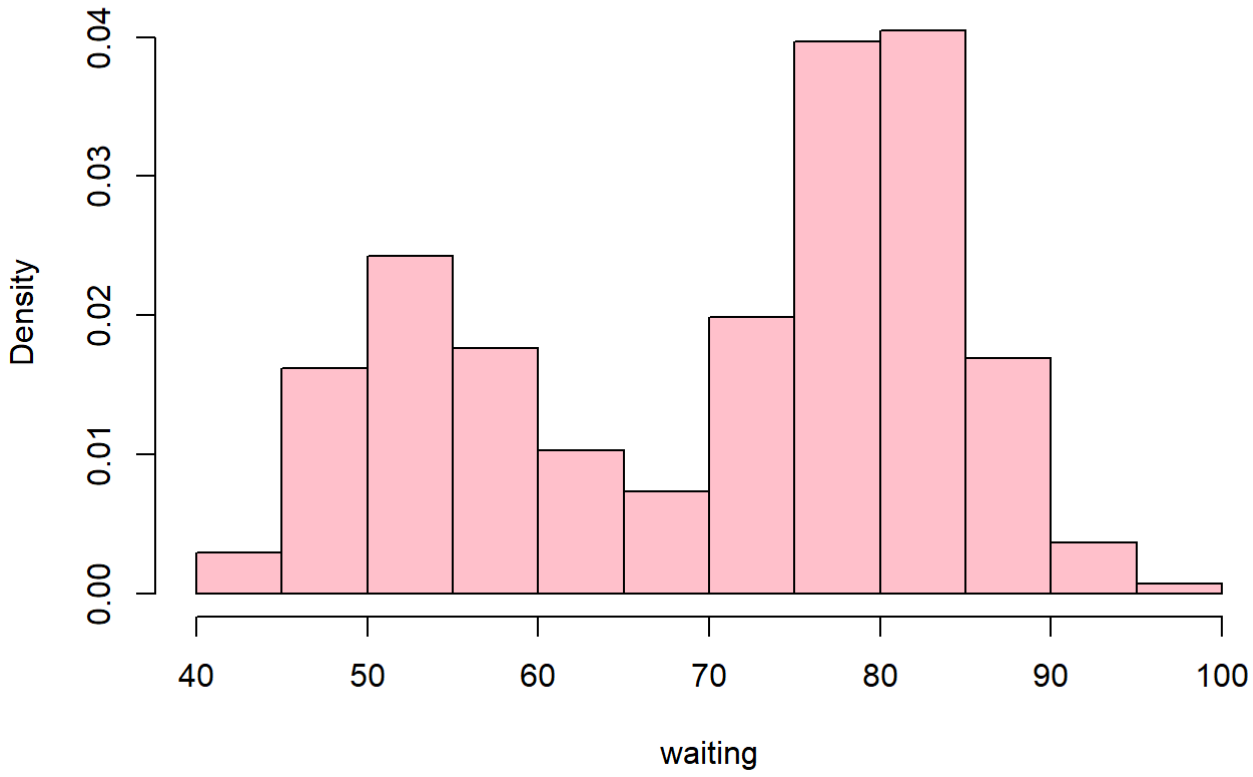

```
## [1] "Inter quantile range for n =  100 is: 0.27"
```

Thus we can observe that the as the value of n increases, the interquartile range decreases.

Problem 3: Analysis of `faithful` datasets.

Consider the `faithful` datasets:

```
attach(faithful)
hist(faithful$waiting,xlab = 'waiting',probability = T,col='pink',main='')
```



Fit following three models using MLE method and calculate **Akaike information criterion** (aka., AIC) for each fitted model. Based on AIC decides which model is the best model? Based on the best model calculate the following probability

$$\mathbb{P}(60 < \texttt{waiting} < 70)$$

i. **Model 1**:

$$f(x) = p * Gamma(x|\alpha, \sigma_1) + (1-p)N(x|\mu, \sigma_2^2), \;\; 0 < p < 1$$

ii. **Model 2**:

$$f(x) = p * Gamma(x|\alpha_1, \sigma_1) + (1-p)Gamma(x|\alpha_2, \sigma_2), \;\; 0 < p < 1$$

iii. **Model 3**:

$$f(x) = p * logNormal(x|\mu_1, \sigma_1^2) + (1-p)logNormal(x|\mu_1, \sigma_1^2), \;\; 0 < p < 1$$

We know that we need to give suitable initial values for our model to get the correct Maximum Likelihood Estimates. Hence, we can use method of moments estimation to get the suitable initial values.

```
mean1 = mean(waiting[waiting<65])
variance1 = (sd(waiting[waiting<65]))^2
mean2 = mean(waiting[waiting>=65])
variance2 = (sd(waiting[waiting>=65]))^2
```

Model 1:

Model one is a mixture distribution of a Gamma distribution and a Normal distribution.

- For waiting time < 65:

mean = 54.05319 $\implies$ $\frac{\alpha}{\sigma_1}$ = 54.05319 and variance = 28.78209 $\implies$ $\frac{\alpha}{{\sigma_1}^2}$ = 28.78209. Hence, σ_1 = 1.8780 and α = 101.5127

- For waiting time >= 65:

mean = 79.79213 $\implies$ μ = 79.79213 and standard deviation = 6.132856 $\implies$ σ_2 = 6.132856

Model 2:

Model two is a mixture distribution of 2 Gamma distributions.

- For waiting time < 65:

mean = 54.05319 $\implies$ $\frac{\alpha}{\sigma_1}$ = 54.05319 and variance = 28.78209 $\implies$ $\frac{\alpha}{{\sigma_1}^2}$ = 28.78209. Hence, σ_1 = 1.8780 and α = 101.5127

- For waiting time >= 65:

mean = 79.79213 $\implies$ $\frac{\alpha_2}{\sigma_2}$ = 79.79213 and variance = 37.61192 $\implies$ $\frac{\alpha_2}{{\sigma_2}^2}$ = 37.61192. Hence, σ_2 = 2.1215 and α = 169.2757

Model 3:

Model three is a mixture distribution of 2 Log normal distributions.

- For waiting time < 65:

mean = 54.05319 and variance = 28.78209 then using formulas of mean and variance for log normal distribution we can find ${\sigma_1}^2$ = log(1 + $\frac{variance}{mean^2}$) = 0.0098. From this we can get μ_1 = 3.9851

- For waiting time >= 65:

mean = 79.79213 and variance = 37.61192 then using formulas of mean and variance for log normal distribution we can find ${\sigma_2}^2$ = log(1 + $\frac{variance}{mean^2}$) = 0.00589. From this we can get μ_2 = 4.3765

For all the 3 models:

p = P(waiting time < 65) = 0.3456

Creating a function named ‘MyMLE’ which will return a vector containing the MLE estimates of the parameters and the value of the likelihood function where ever it has been maximized, according to the model number.

```
MyMLE = function(data_obtained,modelnumber,initial)
{
  if(modelnumber==1)
{
    NegLogLike = function(initial,data_obtained)
    {
      alpha = exp(initial[1])
      sigma1 = exp(initial[2])
      mu = initial[3]
      sigma2 = exp(initial[4])
      p = exp(initial[5])/(1 + exp(initial[5]))
      n = length(data_obtained)
      logl = 0
      logl = sum(log(p*dgamma(data_obtained,shape = alpha,
                                    scale = sigma1) + (1-p)*dnorm(data_obtained, mean = mu,sd=
sigma2)))
      return(-logl)
    }
    fit = optim(initial,NegLogLike,data_obtained=data_obtained)
    alphar = exp(fit$par[1])
    sigma1r = exp(fit$par[2])
    mur = fit$par[3]
    sigma2r = exp(fit$par[4])
    pr = exp(fit$par[5])/(1 + exp(fit$par[5]))
    toreturn = c(length(fit$par),alphar,sigma1r,mur,sigma2r,pr,-fit$value)
    return(toreturn)
  }
else if(modelnumber == 2)
  {
    NegLogLike = function(initial,data_obtained)
    {
      alpha1 = exp(initial[1])
      sigma1 = exp(initial[2])
      alpha2 = exp(initial[3])
      sigma2 = exp(initial[4])
      p = exp(initial[5])/(1 + exp(initial[5]))
      n = length(data_obtained)
      logl = sum(log(p*dgamma(data_obtained,shape = alpha1,
                              scale = sigma1) + (1-p)*dgamma(data_obtained,
                                                             shape = alpha2,scale=sigma2)))
      return(-logl)
    }
    fit = optim(initial,NegLogLike,data_obtained=data_obtained)
    alpha1r = exp(fit$par[1])
    sigma1r = exp(fit$par[2])
    alpha2r = exp(fit$par[3])
    sigma2r = exp(fit$par[4])
    pr = exp(fit$par[5])/(1 + exp(fit$par[5]))
    toreturn = c(length(fit$par),alpha1r,sigma1r,alpha2r,sigma2r,pr,-fit$value)
    return(toreturn)
  }
else{
    NegLogLike = function(initial,data_obtained)
    {
      mu1 = initial[1]
```

```
    sigma1 = exp(initial[2])
    mu2 = initial[3]
    sigma2 = exp(initial[4])
    p = exp(initial[5])/(1 + exp(initial[5]))
    n = length(data_obtained)
    logl = sum(log(p*dlnorm(data_obtained,mu1,
                            sigma1) + (1-p)*dlnorm(data_obtained,
                                                   mean = mu2,sd=sigma2)))
    return(-logl)
  }
  fit = optim(initial,NegLogLike,data_obtained=data_obtained)
  mu1r = fit$par[1]
  sigma1r = exp(fit$par[2])
  mu2r = fit$par[3]
  sigma2r = exp(fit$par[4])
  pr = exp(fit$par[5])/(1 + exp(fit$par[5]))
  toreturn = c(length(fit$par),mu1r,sigma1r,mu2r,sigma2r,pr,-fit$value)
  return(toreturn)
}}
```

Akaike's Information Criterion (AIC) :

Akaike's Information Criterion also called AIC is used for comparing different statistical models. It's defined as follows:

AIC = $2\times$ *No. of parameters in the fitted model* - $2\times$ *Value of the maximized log likelihood function*

The **lower** the AIC for a model, the better the model.

We have written a function $MyMLE$. Along with the MLE estimates of the parameters, it will also return the value of the likelihood function where it has been maximized. Using that we are now going to write a function which is going to calculate the AIC for the 3 given models.

```
    aicfunction = function(modelnumber,data_obtained)
    {
      if(modelnumber == 1)
      {vals = MyMLE(data_obtained,modelnumber,
                    initial = c(log(101.5127),log(1.8780),79.79213,log(6.132856),-log(1/0.35
 - 1)))
      answer = 2*vals[1] - 2*vals[7]
      parameters = vals[2:6]
      }
      else if(modelnumber == 2)
      {
        vals = MyMLE(data_obtained,modelnumber,initial = c(log(101.5127),log(1.8780),log(169.
2757),log(2.1215),-log(1/0.35 - 1)))
        answer = 2*vals[1] - 2*vals[7]
        parameters = vals[2:6]
      }
      else
      {
        vals = MyMLE(data_obtained,modelnumber,initial = c(3.9851,log(sqrt(0.0098)),4.3765,lo
g(sqrt(0.00589)),-log(1/0.35 - 1)))
        answer = 2*vals[1] - 2*vals[7]
        parameters = vals[2:6]
      }
      return(list(answer,parameters))
    }
```

AIC for model 1:

```
    x1 <- aicfunction(modelnumber = 1,data_obtained = faithful$waiting)
    cat('AIC for model 1: ',x1[[1]])
```

```
## AIC for model 1:  2200.578
```

AIC for model 2:

```
    x2 <- aicfunction(modelnumber = 2,data_obtained = faithful$waiting)
    cat('AIC for model 2: ',x2[[1]])
```

```
## AIC for model 2:  2215.851
```

AIC for model 3:

```
   x3 <- aicfunction(modelnumber = 3,data_obtained = faithful$waiting)
    cat('AIC for model 3: ',x3[[1]])
```

```
## AIC for model 3:  2075.42
```

From above we can see that the third model has the lowest AIC, hence, the **3rd** model is the best among the 3 models.

Calculating probability using the 3rd model:

The parameters for the 3rd model are:

```
    Parameters = c("Mu1_hat","Sigma1_hat","Mu2_hat","Sigma2_hat","p_hat")
    theta = x3[[2]]
    params = data.frame(Parameters,'Parameter_estimates' = theta)
    print(params)
```

```
##   Parameters Parameter_estimates
## 1    Mu1_hat          4.00389385
## 2 Sigma1_hat          0.11484437
## 3    Mu2_hat          4.38434747
## 4 Sigma2_hat          0.06973538
## 5      p_hat          0.37617486
```

Defining the probability density function for the 3rd model.

```
    theta = x3[[2]]

    dMix = function(x,theta)
    {
      mu1_hat = theta[1]
      sigma1_hat = theta[2]
      mu2_hat = theta[3]
      sigma2_hat = theta[4]
      p_hat = theta[5]
      f = p_hat*dlnorm(x,mu1_hat,sigma1_hat)+(1-p_hat)*dlnorm(x,mu2_hat,sigma2_hat)
      return(f)
    }
```

Integrating the probability density function defined above for the calculated MLE estimates:

```
    probability = as.vector(integrate(dMix,60,70,theta))

    cat('P(60 < waiting time < 70) = ',probability[[1]],"\n","Absolute error = ",
        probability[[2]])
```

```
## P(60 < waiting time < 70) =  0.09083472
##  Absolute error =  1.008468e-15
```

Hence, the probability for waiting time of eruptions to be between 60 and 70 minutes is approximately **9%**.

Problem 4: Modelling Insurance Claims

Consider the `Insurance` datasets in the `MASS` package. The data given in data frame `Insurance` consist of the numbers of policyholders of an insurance company who were exposed to risk, and the numbers of car insurance claims made by those policyholders in the third quarter of 1973.

This data frame contains the following columns:

`District` (factor): district of residence of policyholder (1 to 4): 4 is major cities.

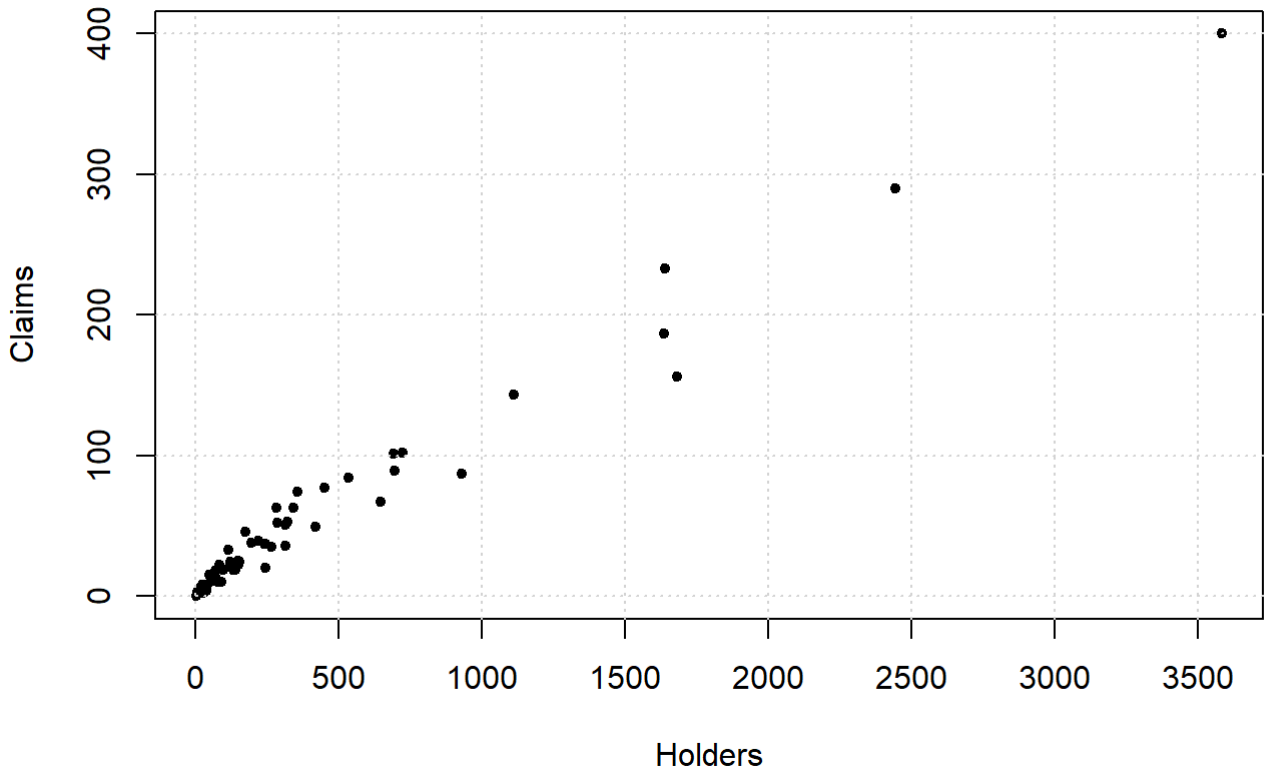
`Group` (an ordered factor): group of car with levels <1 litre, 1–1.5 litre, 1.5–2 litre, >2 litre.

`Age` (an ordered factor): the age of the insured in 4 groups labelled <25, 25–29, 30–35, >35.

`Holders` : numbers of policyholders.

`Claims` : numbers of claims

```
library(MASS)
plot(Insurance$Holders,Insurance$Claims
     ,xlab = 'Holders',ylab='Claims',pch=20)
grid()
```



Note: If you use built-in function like `lm` or any packages then no points will be awarded.

Part A: We want to predict the `Claims` as function of `Holders` . So we want to fit the following models:

$$\texttt{Claims}_i = \beta_0 + \beta_1 \, \texttt{Holders}_i + \varepsilon_i, \quad i = 1, 2, \cdots, n$$

Assume : $\varepsilon_i \sim N(0, \sigma^2)$. Note that $\beta_0, \beta_1 \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$.

The above model can alse be re-expressed as,

$$\texttt{Claims}_i \sim N(\mu_i, \sigma^2), \quad where$$

$$\mu_i = \beta_0 + \beta_1 \, \texttt{Holders}_i + \varepsilon_i, \quad i = 1, 2, \cdots, n$$

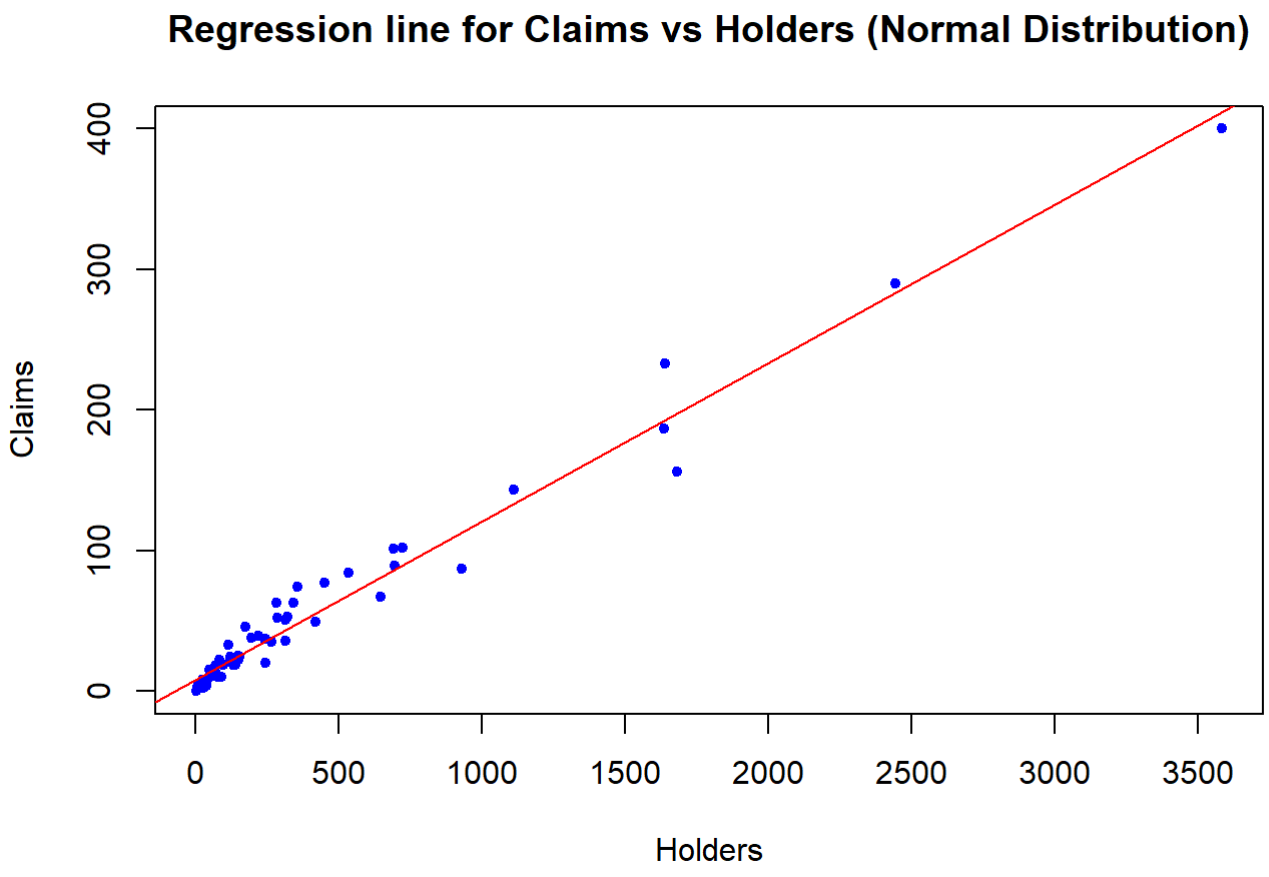
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

i. Clearly write down the negative-log-likelihood function in `R`. Then use `optim` function to estimate MLE of $\theta = (\beta_0, \beta_1, \sigma)$
ii. Calculate **Bayesian Information Criterion** (BIC) for the model.

```
## [1] "the MLE of parameter are"
```

```
##     beta_0     beta_1      sigma
##  8.1251542  0.1126412 11.8711080
```

```
## Bayesian Information criterion for this model is 510.7587
```



Part B: Now we want to fit the same model with change in distribution:

$$\texttt{Claims}_i = \beta_0 + \beta_1 \, \texttt{Holders}_i + \varepsilon_i, \quad i = 1, 2, \cdots, n$$

Assume : $\varepsilon_i \sim Laplace(0, \sigma^2)$. Note that $\beta_0, \beta_1 \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$.

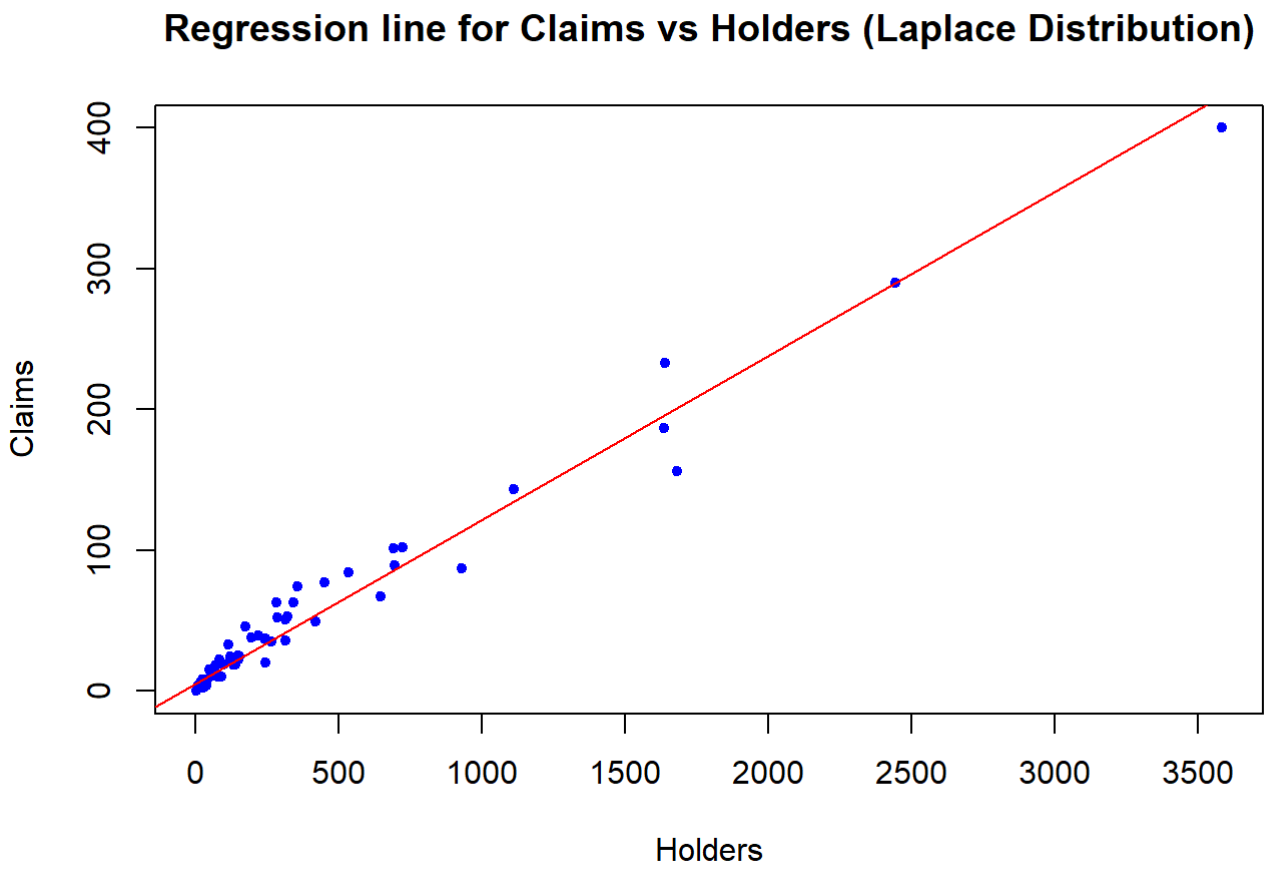
i. Clearly write down the negative-log-likelihood function in `R`. Then use `optim` function to estimate MLE of $\theta = (\beta_0, \beta_1, \sigma)$

ii. Calculate **Bayesian Information Criterion** (BIC) for the model.

```
## [1] "the MLE of parameter are"
```

```
##    beta_0    beta_1     sigma
## 5.0845434 0.1166252 8.1977382
```

```
## Bayesian Information criterion for this model is 498.6871
```



Part C: We want to fit the following models:

$$\texttt{Claims}_i \sim LogNormal(\mu_i, \sigma^2), where$$

$$\mu_i = \beta_0 + \beta_1 \log(\texttt{Holders}_i), \;\; i = 1, 2, \ldots, n$$

Note that $\beta_0, \beta_1 \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$.

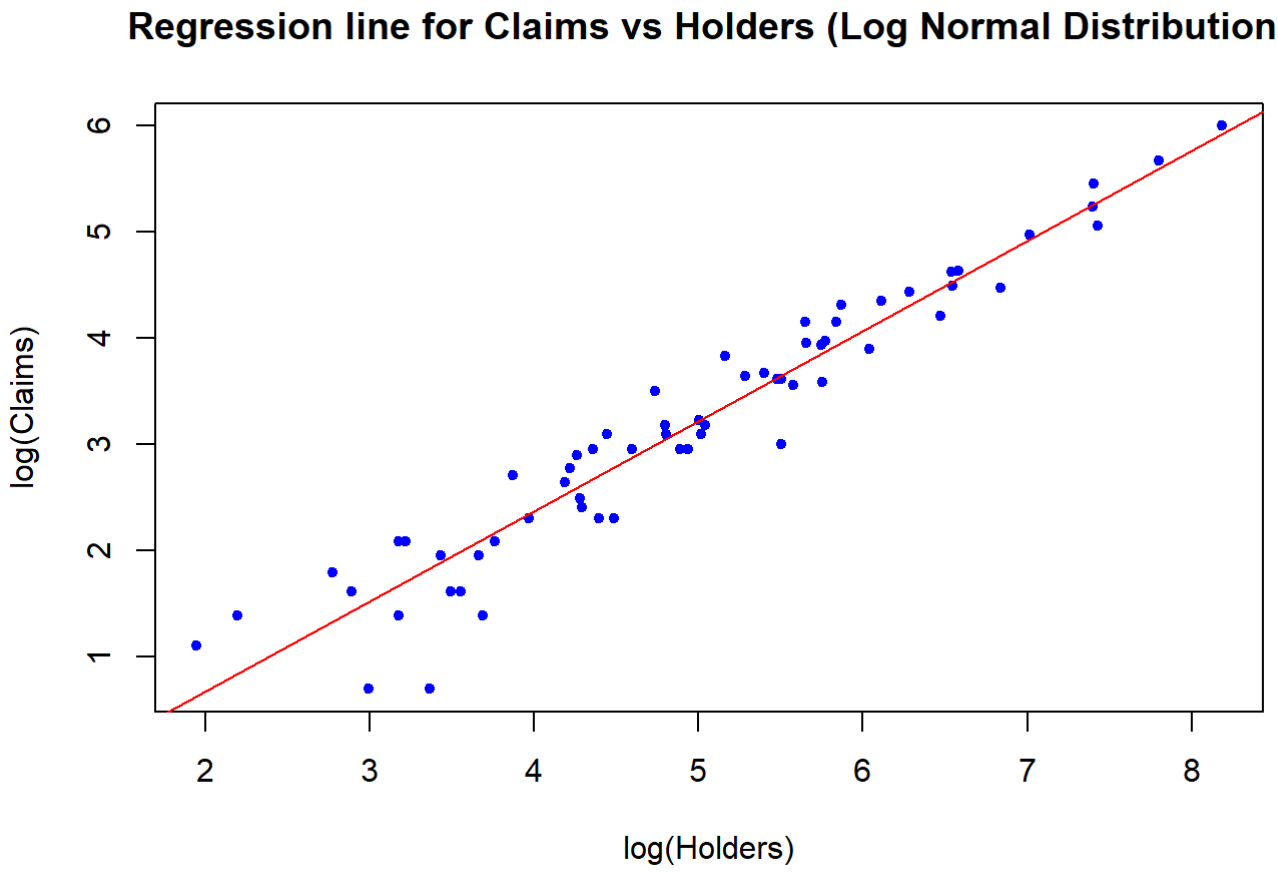
i. Clearly write down the negative-log-likelihood function in `R`. Then use `optim` function to estimate MLE of $\theta = (\alpha, \beta, \sigma)$

ii. Calculate **Bayesian Information Criterion** (BIC) for the model.

```
## [1] "the MLE of parameter are"
```

```
##     beta_0     beta_1      sigma
## -1.0245813  0.8479253  0.3293131
```

```
## Bayesian Information criterion for this model is 452.6034
```



Part D: We want to fit the following models:

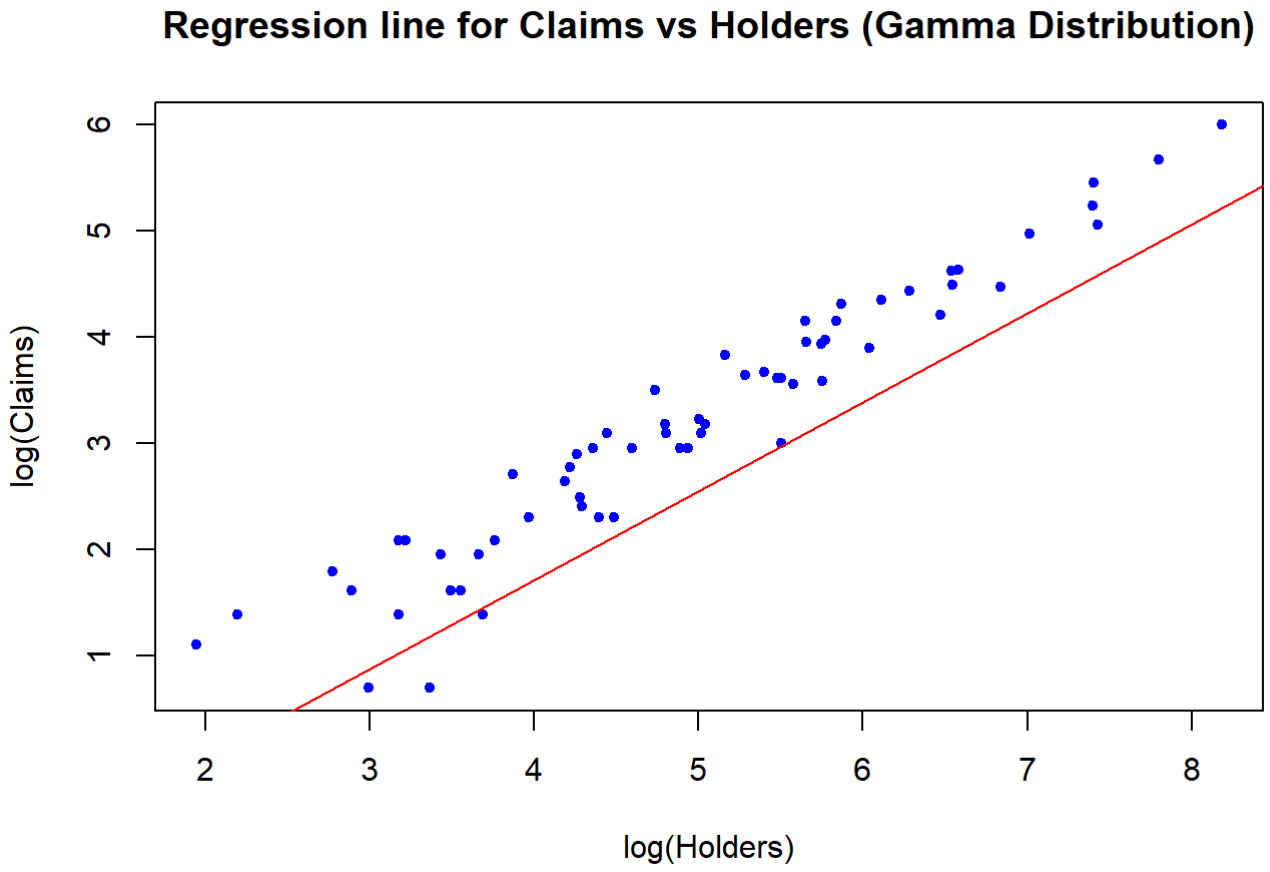
$$\texttt{Claims}_i \sim Gamma(\alpha_i, \sigma), where$$

$$log(\alpha_i) = \beta_0 + \beta_1 \log(\texttt{Holders}_i), \;\; i = 1, 2, \ldots, n$$

```
## [1] "the MLE of parameter are"
```

```
##     beta_0     beta_1      sigma
## -1.6421109  0.8371004  0.7208980
```

```
## Bayesian Information criterion for this model is 437.3382
```



iii. Compare the BIC of all three models

```
## [1] "The MLE comparison for 4 models"
```

```
##   distribution_type  Beta_0 Beta_1   Sigma
## 1            Normal  8.1252 0.1126 11.8711
## 2           Laplace  5.0845 0.1166  8.1977
## 3        Log Normal -1.0246 0.8479  0.3293
## 4             Gamma -1.6421 0.8371  0.7209
```

```
##   distribution_type BIC_Value
## 1            Normal  510.7587
## 2           Laplace  498.6871
## 3        Log Normal  452.6034
## 4             Gamma  437.3382
```

Bayesian Information Criterion Comparison:

```
     Bayesian Information criterion is useful for selecting the best model among the finite
set of models.In general the models with lower BIC are preferred. Hence from the above 4 mode
ls Gamma and Log Normal distributions are preferred comapred to Normal and Laplace Distributi
on.
```

Problem 5: Computational Finance - Modelling Stock prices

Following piece of code download the prices of TCS since 2007

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
## Loading required package: TTR
```

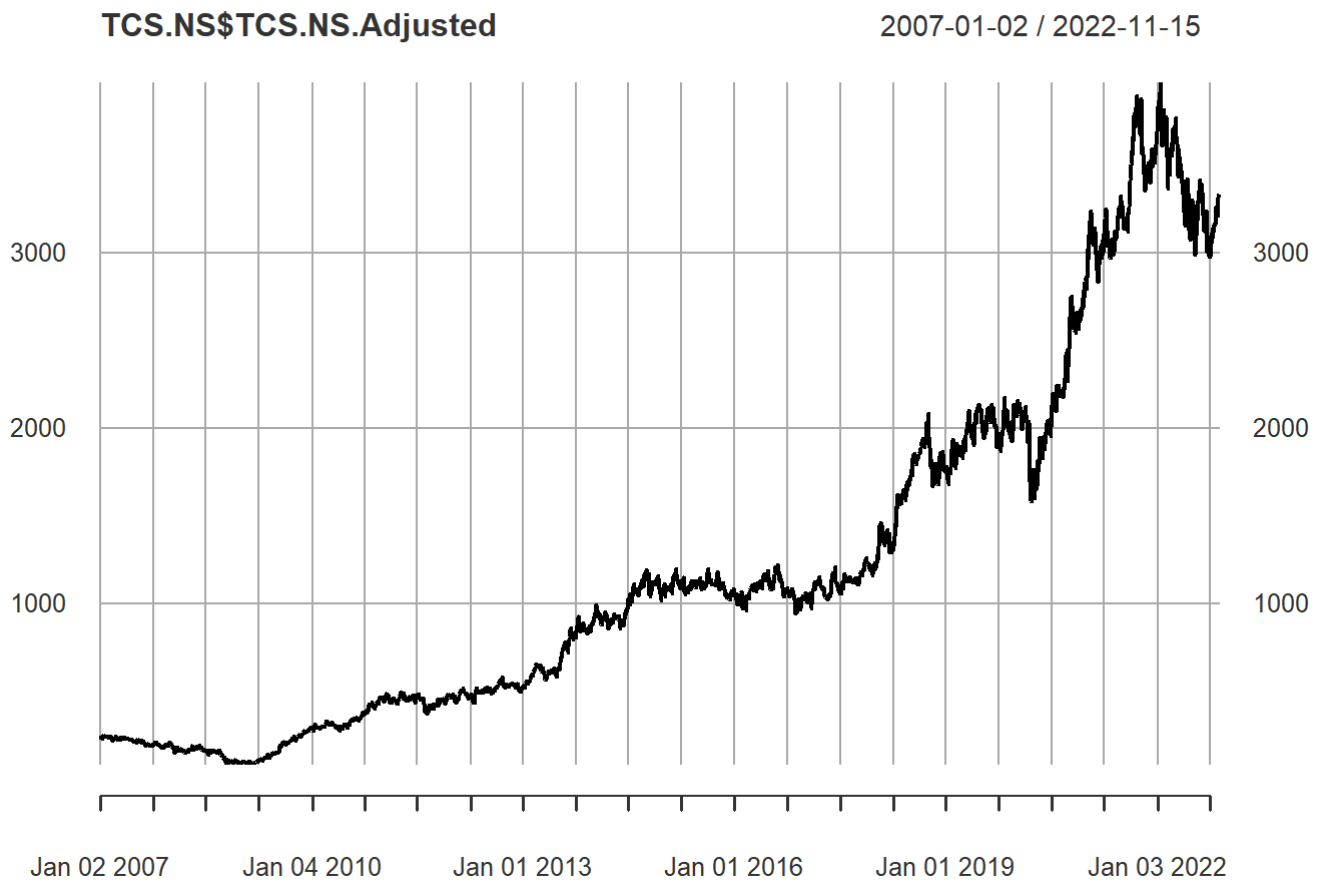
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Warning: TCS.NS contains missing values. Some functions will not work if objects
## contain missing values in the middle of the series. Consider using na.omit(),
## na.approx(), na.fill(), etc to remove or replace them.
```

```
## [1] "TCS.NS"
```

```
##            TCS.NS.Open TCS.NS.High TCS.NS.Low TCS.NS.Close TCS.NS.Volume
## 2022-11-07      3229.0     3242.80    3195.10      3233.70       1474498
## 2022-11-09      3249.8     3249.80    3201.65      3216.05       1162267
## 2022-11-10      3170.0     3225.00    3170.00      3205.65       1573092
## 2022-11-11      3269.6     3341.60    3255.05      3315.95       3265394
## 2022-11-14      3324.0     3349.00    3309.00      3335.50       1342074
## 2022-11-15      3321.0     3339.95    3292.00      3332.60       1400708
##            TCS.NS.Adjusted
## 2022-11-07         3233.70
## 2022-11-09         3216.05
## 2022-11-10         3205.65
## 2022-11-11         3315.95
## 2022-11-14         3335.50
## 2022-11-15         3332.60
```

Plot the adjusted close prices of TCS



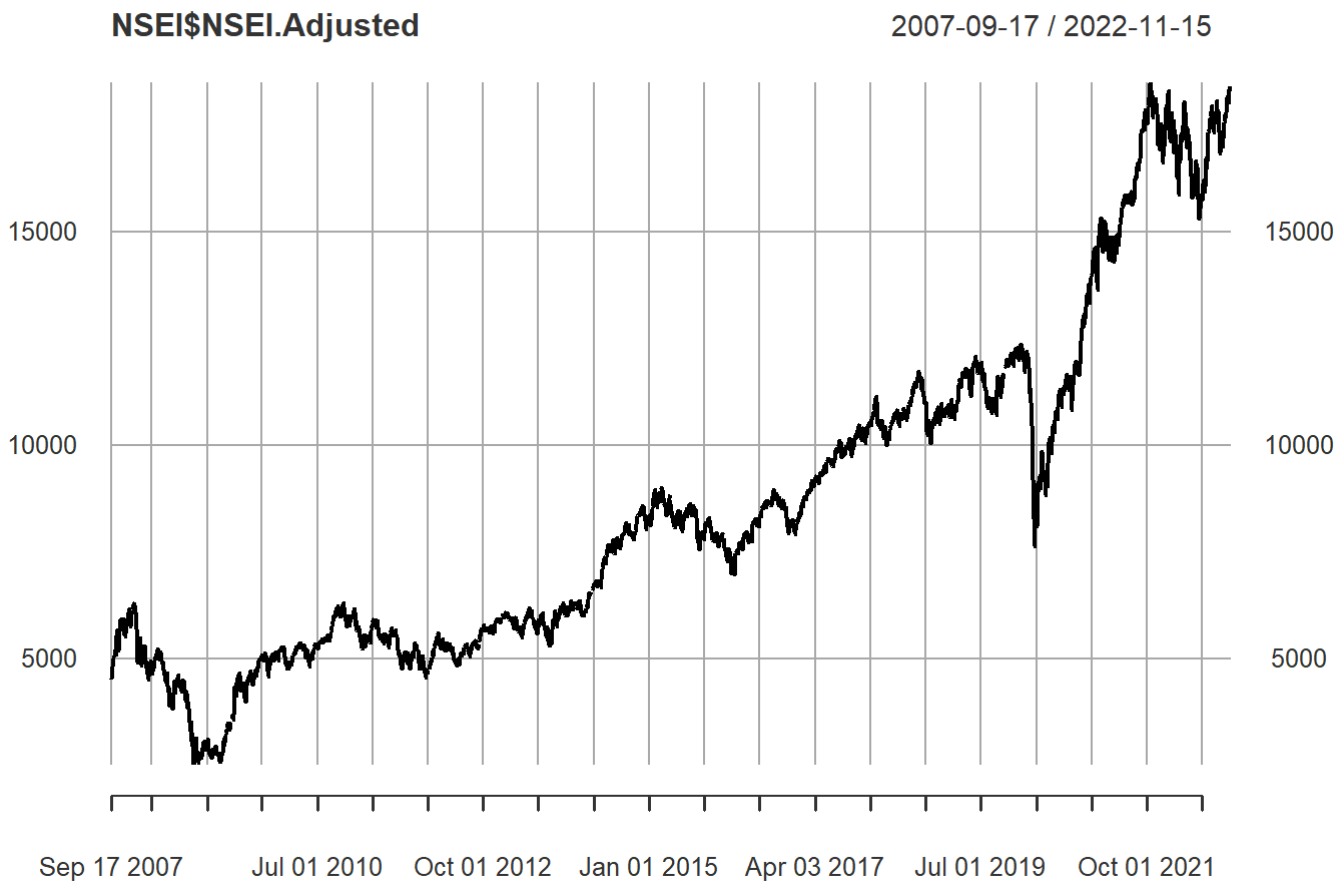
Download the data of market index Nifty50. The Nifty 50 index indicates how the over all market has done over the similar period.

```
## Warning: ^NSEI contains missing values. Some functions will not work if objects
## contain missing values in the middle of the series. Consider using na.omit(),
## na.approx(), na.fill(), etc to remove or replace them.
```

```
## [1] "^NSEI"
```

```
##            NSEI.Open NSEI.High NSEI.Low NSEI.Close NSEI.Volume NSEI.Adjusted
## 2022-11-07  18211.75  18255.50 18064.75   18202.80      314800      18202.80
## 2022-11-09  18288.25  18296.40 18117.50   18157.00      307200      18157.00
## 2022-11-10  18044.35  18103.10 17969.40   18028.20      256500      18028.20
## 2022-11-11  18272.35  18362.30 18259.35   18349.70      378500      18349.70
## 2022-11-14  18376.40  18399.45 18311.40   18329.15      301400      18329.15
## 2022-11-15  18362.75  18427.95 18282.00   18403.40      250900      18403.40
```

Plot the adjusted close value of Nifty50



Log-Return

We calculate the daily log-return, where log-return is defined as

$$r_t = \log(P_t) - \log(P_{t-1}) = \Delta \log(P_t),$$

where P_t is the closing price of the stock on t^{th} day.

- Consider the following model:

$$r_t^{TCS} = \alpha + \beta r_t^{Nifty} + \varepsilon,$$

where $\mathbb{E}(\varepsilon) = 0$ and $\mathbb{V}ar(\varepsilon) = \sigma^2$. 1. Estimate the parameters of the models $\theta = (\alpha, \beta, \sigma)$ using the method of moments type plug-in estimator discussed in the class.

The three moments that we are going to use to estimate the parameter values are

$$E[\epsilon_i] = 0 \quad E[\epsilon_i r_{t_i}^{Nifty}] = 0 \quad E[\epsilon_i^2] = \sigma^2$$

The equivalent equations are:-

$$\frac{1}{n}\sum_{i=1}^{n}[r_{t_i}^{TCS}-\hat{\alpha}-\hat{\beta}r_{t_i}^{Nifty}]=0$$

$$\frac{1}{n}\sum_{i=1}^{n}r_{t_i}^{TCS}-\frac{1}{n}\sum_{i=1}^{n}\hat{\alpha}-\frac{1}{n}\sum_{i=1}^{n}\hat{\beta}r_{t_i}^{Nifty}=0$$

$$\bar{r}_{t_i}^{TCS}-\hat{\alpha}-\hat{\beta}\bar{r}_{t_i}^{Nifty}=0 \quad ----equation\ 1$$

$$\frac{1}{n}\sum_{i=1}^{n}[(r_{t_i}^{TCS}-\hat{\alpha}-\hat{\beta}r_{t_i}^{Nifty})r_{t_i}^{Nifty}]=0$$

$$\frac{1}{n}\sum_{i=1}^{n}[(r_{t_i}^{TCS}r_{t_i}^{Nifty}-\hat{\alpha}r_{t_i}^{Nifty}-\hat{\beta}(r_{t_i}^{Nifty})^2]=0$$

$$\frac{1}{n}\sum_{i=1}^{n}(r_{t_i}^{TCS}r_{t_i}^{Nifty}-\frac{1}{n}\sum_{i=1}^{n}\hat{\alpha}r_{t_i}^{Nifty}-\frac{1}{n}\sum_{i=1}^{n}\hat{\beta}(r_{t_i}^{Nifty})^2=0$$

$$\frac{1}{n}\sum_{i=1}^{n}(r_{t_i}^{TCS}r_{t_i}^{Nifty})-\hat{\alpha}\bar{r}_{t_i}^{Nifty}-\hat{\beta}\frac{1}{n}\sum_{i=1}^{n}(r_{t_i}^{Nifty})^2=0 \quad ----equation\ 2$$

$$\frac{1}{n}\sum_{i=1}^{n}[r_{t_i}^{TCS}-\hat{\alpha}-\hat{\beta}r_{t_i}^{Nifty}]^2=\hat{\sigma}^2 \quad ----equation\ 3$$

The estimated values of $\hat{\alpha}$,$\hat{\beta}$ and σ^2 respectively is

```
## [1] 0.000461652 0.743661787 0.016184817
```

2. Estimate the parameters using the `lm` built-in function of `R`. Note that `lm` using the OLS method.

```
##
## Call:
## lm(formula = retrn$TCS.NS.Adjusted ~ retrn$NSEI.Adjusted)
##
## Coefficients:
##         (Intercept)  retrn$NSEI.Adjusted
##           0.0004617            0.7436618
```

Therefore the linear model fit is

$$r_t^{TCS}=0.0004611+0.7436969r_t^{Nifty}$$

The estimated parameter values are

```
##         (Intercept) retrn$NSEI.Adjusted
##         0.000461652         0.743661787         0.016184817
```

3. Fill-up the following table

Parameters	Method of Moments	OLS
α	0.0004611207	0.0004611207
β	0.7436969330	0.7436969330
σ	0.0161848154	0.0161848154

We got the same estimated parameter values using Method of Moments and the $lm()$ function which uses Ordinary least square method.

4. If the current value of Nifty is 18000 and it goes up to 18200. The current value of TCS is Rs. 3200/-. How much you can expect TCS price to go up?

First we calculate the log returns for nifty and substitute it in the model we fit

```
## [1] 3227.893
```

Therefore we can expect the TCS price to go up 27.893 Rupees.