Probability and Statistics with R

Anuraj Kashyap, Akshay Thorave, Praveen Kumar

Nov 16-2022

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.

Answer: In order to define the probability distribution model using S, following necessary conditions must be satisfied.

1. $0 \leq P(X = i) \leq 1, \forall\ i.$
2. $\Sigma_{i=0}^{\infty} P(X = i) = 1.$

According to the given model $P(X = i) = ar^i$.

Therefore,

$$0 \leq a \leq 1 \quad \& \quad 0 \leq r < 1$$

Now,

$$\begin{aligned} &\Sigma_{i=0}^{\infty} P(X = i) = 1 \\ \implies &\Sigma_{i=0}^{\infty} ar^i = 1 \\ \implies &r = 1 - a \end{aligned}$$

Therefore, $P(X = i) = a(1 - a)^i$.

2.Check if mean and variance exists for the probability model.

Answer: $E(X) = \Sigma_{i=0}^{\infty} iP(X = i) = \Sigma_{i=0}^{\infty} a(1 - a)^i = \frac{1-a}{a}$.

Therefore, mean $= E(x) = \frac{1-a}{a}$.

variance $= Var(X) = \Sigma_{i=0}^{\infty}(i - E(X))^i = \frac{1-a}{a^2}$.

Therefore, Variance $= \frac{1-a}{a^2}$.

Hence, mean and variance exists for the probability model.

3. Can you find the analytically expression of mean and variance.

Answer: Yes, we can analytically find the expression of mean and variance as follows.

mean $= E(x) = \frac{1-a}{a}$.

Variance $= \frac{1-a}{a^2}$.

4.

From historical data we found the following summary statistics

mean	median	variance	total number of matches
1.5	1	2.25	380

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?

Answer: According to given historical data,

$$\begin{aligned} & mean = 1.5 \\ \implies & \frac{1-a}{a} = 1.5 \\ \implies & a = 0.4 \end{aligned}$$

Therefore, $X \sim geom(0.4)$.

a. probability that home team will score atleast one goal,

$= p(X \geq 1) = 1 - p(X = 0)$

```
p = 1 - dgeom(0,0.4)
print(p)
```

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

b. probability that home team will score at least one goal but less than four goal,

$= p(1 \leq X < 4) = p(X = 1) + p(X = 2) + p(X = 3)$

```
p = dgeom(1,0.4) + dgeom(2,0.4) + dgeom(3,0.4)
print(p)
```

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

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?

Answer: Now, we want to model above distribution with poisson probability model. Assume that, the underlying distribution is poisson probability distribution.

i.e Assume that, X~poisson(λ), where λ = average number of goals = mean.

According to summary statistics, mean = 1.5.

$\implies \lambda = 1.5$.

Thus, $X \sim poisson(1.5)$

a. probability that home team will score at least one goal,

$= p(X \geq 1) = 1 - p(X = 0)$

```
p = 1 - dpois(0,1.5)
print(p)
```

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

b. probability that home team will score at least one goal but less than four goal,

$= p(1 \leq X < 4) = p(X = 1) + p(X = 2) + p(X = 3)$

```
p = dpois(1,1.5) + dpois(2,1.5) + dpois(3,1.5)
print(p)
```

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

6. Which probability model you would prefer over another?

Answer: I prefer the poisson model over the newly defined probability model for this given data.

7. Write down the likelihood functions of your newly defined probability models and Poisson models. Clearly mention all the assumptions that you are making.

Answer:

i) Likelihood function for newly defined probability model.

Let $x_1, x_2, x_3, \cdots, x_n$ be a observed sample.

Then the likelihood function is given by,

$$f(x_1, x_2, x_3, \cdots, x_n|\hat{a}) = f(x_1|\hat{a})f(x_2|\hat{a})f(x_3|\hat{a})\cdots f(x_n|\hat{a}) \tag{1}$$
$$= \hat{a}(1-\hat{a})^{x_1}\hat{a}(1-\hat{a})^{x_2}\hat{a}(1-\hat{a})^{x_3}\cdots\hat{a}(1-\hat{a})^{x_n} \tag{2}$$
$$= \hat{a}^n(1-\hat{a})^{\Sigma_{i=1}^{\infty}x_i} \tag{3}$$

ii) Likelihood function for poisson model.

Let $x_1, x_2, x_3, \cdots, x_n$ be a observed sample.

Then the likelihood function is given by,

$$f(x_1, x_2, x_3, \cdots, x_n|\hat{\lambda}) = f(x_1|\hat{\lambda})f(x_2|\hat{\lambda})f(x_3|\hat{\lambda})\cdots f(x_n|\hat{\lambda}) \tag{4}$$
$$= \frac{e^{-\hat{\lambda}}\hat{\lambda}^{x_1}}{x_1!}\frac{e^{-\hat{\lambda}}\hat{\lambda}^{x_2}}{x_2!}\frac{e^{-\hat{\lambda}}\hat{\lambda}^{x_3}}{x_3!}\cdots\frac{e^{-\hat{\lambda}}\hat{\lambda}^{x_n}}{x_n!} \tag{5}$$
$$= \frac{e^{-n\hat{\lambda}}\hat{\lambda}^{\Sigma_{i=1}^{n}x_i}}{\Pi_{i=1}^{n}x_i!} \tag{6}$$

Assumptions:

- Number of goals scored in a match is independent of the number of goals scored in every other match.
- Number of goals in every match is identically distributed.
- Assume that no other factors is affecting the number of goals scored in a match.

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`

```
MyMLE=function(n,shape,scale)
{
  n<<-n

  Negloglike=function(data,theta)
  {
    alpha = exp(theta[1])
    sigma = exp(theta[2])
```

```
    l=0
    for(i in 1:n)
    {
      l=l+log(dgamma(data[i], shape = alpha, scale = sigma))

    }
    return(-l)
  }

  theta=c(0.1,0.1)


  sim=rgamma(n,shape,scale)
  data=sim
  fit = optim(par=theta,Negloglike,data=sim)

  alpha_hat = exp(fit$par[1])

  return(log(alpha_hat))
}
```

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)`

```
rgamma(20,1.5,scale=2.2)
```

```
##  [1]  8.1147300  8.2070249  1.6138993 10.0296050  2.1751505  2.0746930
##  [7] 10.3477266  2.0135825  1.1660005  4.2636061  2.2022863  6.0973293
## [13]  0.6355666  1.3058527  2.5514136  2.5932909  3.4632820 15.1452669
## [19]  3.9859409  2.9009841
```

(ii) Apply the 'MyMLE' to estimate θ and append the value in a vector

```
x=MyMLE(20,1.5,2.2)
x
```

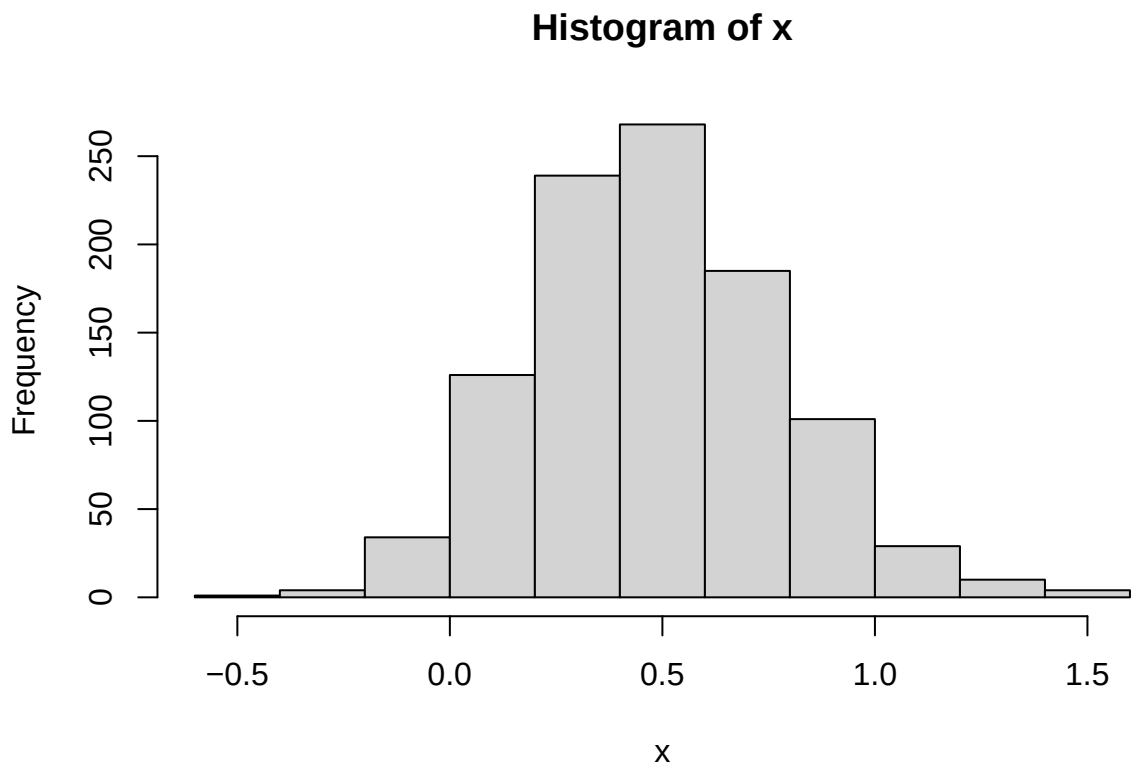
```
## [1] 0.3573635
```

(iii) Repeat the step (i) and (ii) 1000 times

```
for (i in 1:1000){

  x=append(x,MyMLE(20,1.5,2.2))
}
```

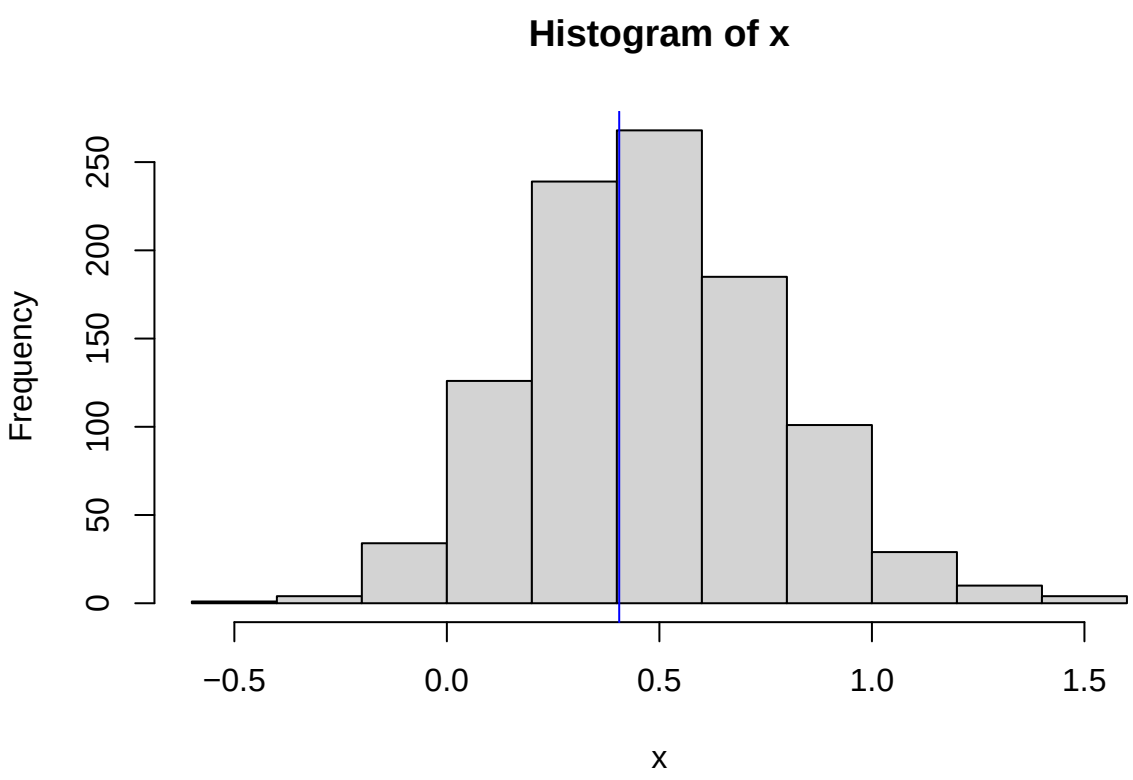
(iv) Draw histogram of the estimated MLEs of θ.

```
hist(x)
```



(v) Draw a vertical line using `abline` function at the true value of θ.

```
hist(x)
abline(v=log(1.5),col="blue")
```



(vi) Use ‘quantile‘ function on estimated θ’s to find the 2.5 and 97.5-percentile points.

```
y=quantile(x, probs = c(.025, .975))
y
```

```
##       2.5%      97.5%
## -0.0545285  1.0615413
```

3. Choose `n=40`, and `alpha=1.5` and repeat the (2). ##

   (i) Simulate $\{X_1, X_2, \cdots, X_n\}$ from `rgamma(n=40,shape=1.5,scale=2.2)`

```
rgamma(40,1.5,scale=2.2)
```

```
##  [1] 1.5267874 0.7370709 0.9912494 1.8759363 4.4724585 3.4085015 0.7716944
##  [8] 2.4638911 1.1927035 1.3672819 8.4320710 2.8972763 0.5113819 2.1346310
## [15] 2.7542156 3.0249056 2.9013095 9.9351367 3.2466457 9.1579653 2.5277719
## [22] 1.0208893 2.3540406 1.7999026 5.2918643 4.0167546 0.4694279 3.4635901
## [29] 3.7435920 7.3986797 0.6170984 2.7796745 0.7817986 0.2392237 1.0649369
## [36] 8.1578502 3.0951272 0.1074861 4.2642996 5.2316020
```

(ii) Apply the ‘MyMLE‘ to estimate θ and append the value in a vector

```
x=MyMLE(40,1.5,2.2)
x
```

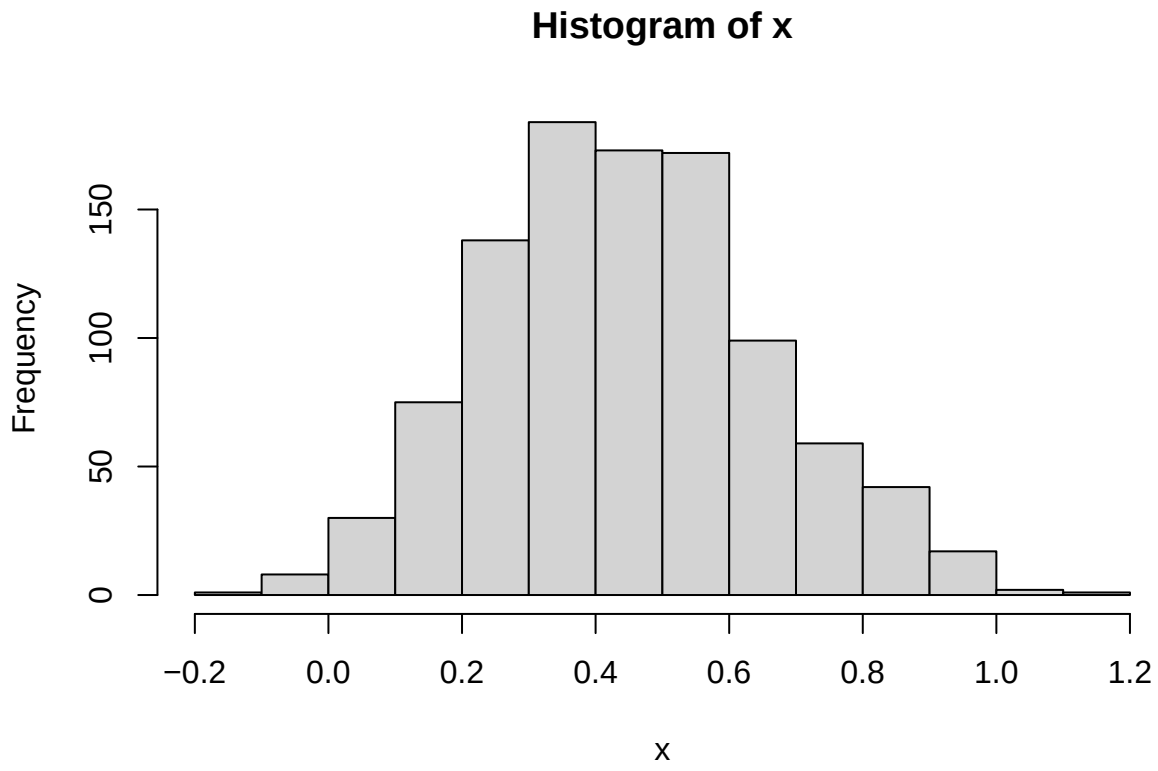
```
## [1] 0.91648
```

(iii) Repeat the step (i) and (ii) 1000 times

```
for (i in 1:1000){

  x=append(x,MyMLE(40,1.5,2.2))
}
```

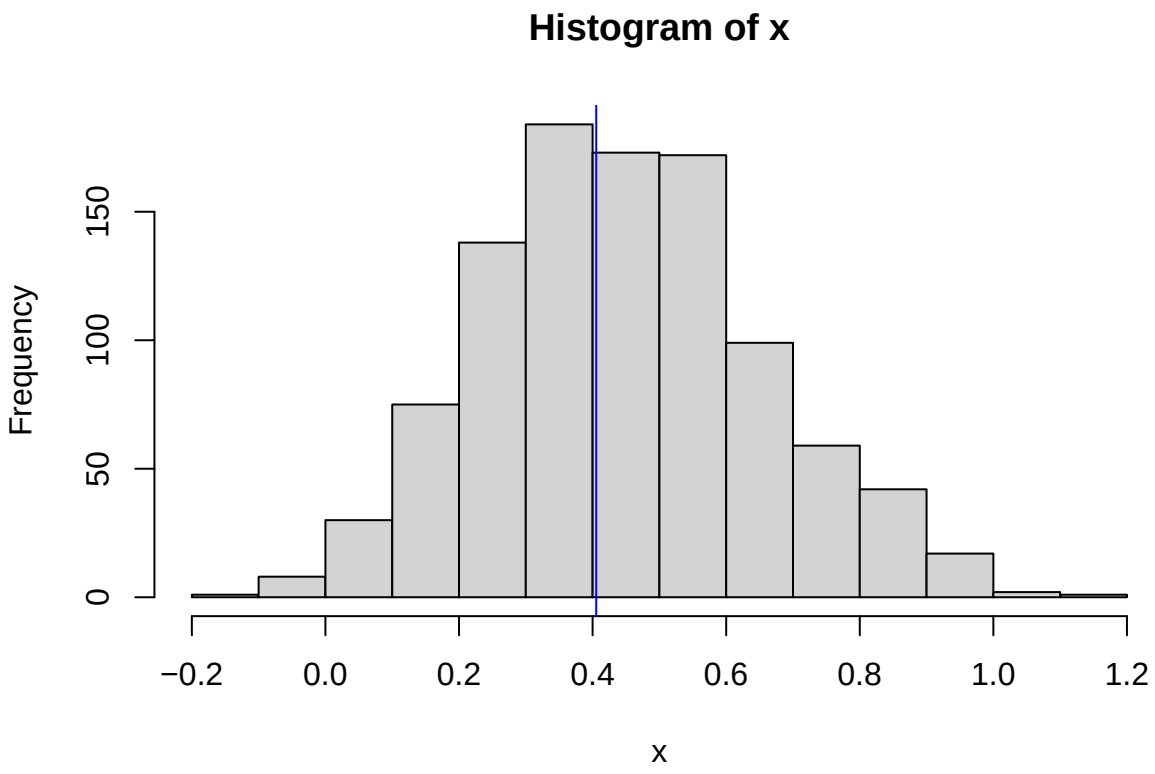
(iv) Draw histogram of the estimated MLEs of θ.

```
hist(x)
```



(v) Draw a vertical line using `abline` function at the true value of θ.

```
hist(x)
abline(v=log(1.5),col="blue")
```



(vi) Use 'quantile' function on estimated θ's to find the 2.5 and 97.5-percentile points.

```
y=quantile(x, probs = c(.025, .975))
y
```

```
##       2.5%      97.5%
## 0.06852554 0.88334044
```

4. Choose `n=100`, and `alpha=1.5` and repeat the (2).

   (i) Simulate $\{X_1, X_2, \cdots, X_n\}$ from `rgamma(n=100,shape=1.5,scale=2.2)`

```
rgamma(100,1.5,scale=2.2)
```

```
##   [1]  1.5031398  2.6626782  0.7451904  3.7803048  2.3761194  4.3897136
##   [7]  6.0972300  1.6920368  3.1377636  3.4667916  0.9605922  5.0877292
##  [13]  5.8403216  4.4609933  1.7498629  3.4724579  3.5511254  5.3019783
##  [19]  3.7906318  2.2957360 14.0058712  3.0158614  3.0282142  9.9098861
##  [25]  0.3385510  7.8130472  5.4231264  3.3178116  3.0931836  1.3929094
##  [31]  2.2177029  4.0118820  0.6370953  0.5484252  0.8037506  1.9901440
##  [37]  1.5688682  8.3979552  0.7803396  2.0230125  0.3694406  0.4948604
##  [43]  3.5418774  1.4063891  3.8938787  5.0644263  2.2218003  5.5426979
```

```
##  [49]  0.5190044  4.0099495  0.2256457  2.1616562  2.2977092  5.7894349
##  [55]  1.7811912  5.6521182  1.0451392  2.2679198  6.4390458  1.0803656
##  [61]  1.7250386  6.4279948  3.2641870  2.0848368 21.0448636  0.9845579
##  [67]  5.8785294  0.9333157  2.8072107  2.0145576  4.3454519  4.5750379
##  [73]  1.2000491  6.3076936  1.5998085  3.1349620  2.5962838  4.2306246
##  [79]  2.1942569  1.6539854  1.5126184  3.2954974  0.6777954  0.6677242
##  [85]  2.5895655  2.8750504  1.2079126 11.5093211  1.6774696  3.6467222
##  [91]  0.6091085  6.5365985  2.9548802  1.4681431  3.4010894  2.3073071
##  [97]  2.5353230  7.0948051  5.8538074  1.6018187
```

(ii) Apply the 'MyMLE' to estimate θ and append the value in a vector

```
x=MyMLE(100,1.5,2.2)
x
```

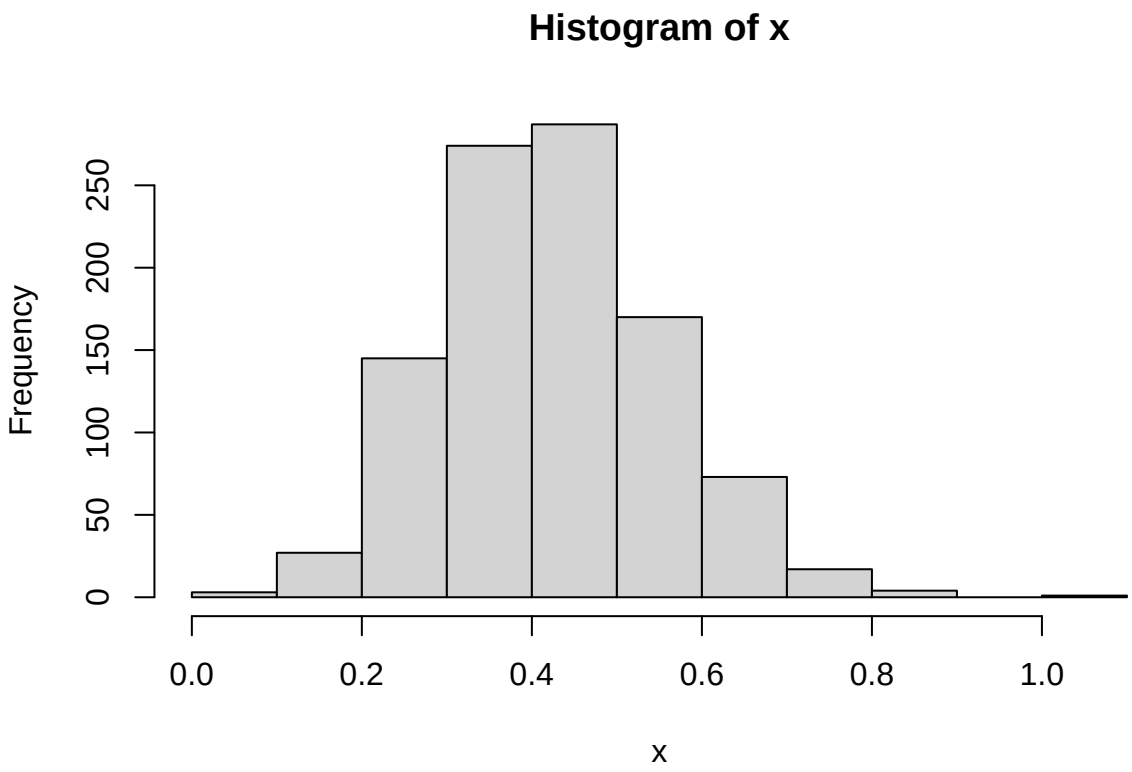
```
## [1] 0.4290939
```

(iii) Repeat the step (i) and (ii) 1000 times

```
for (i in 1:1000){

  x=append(x,MyMLE(100,1.5,2.2))
}
```

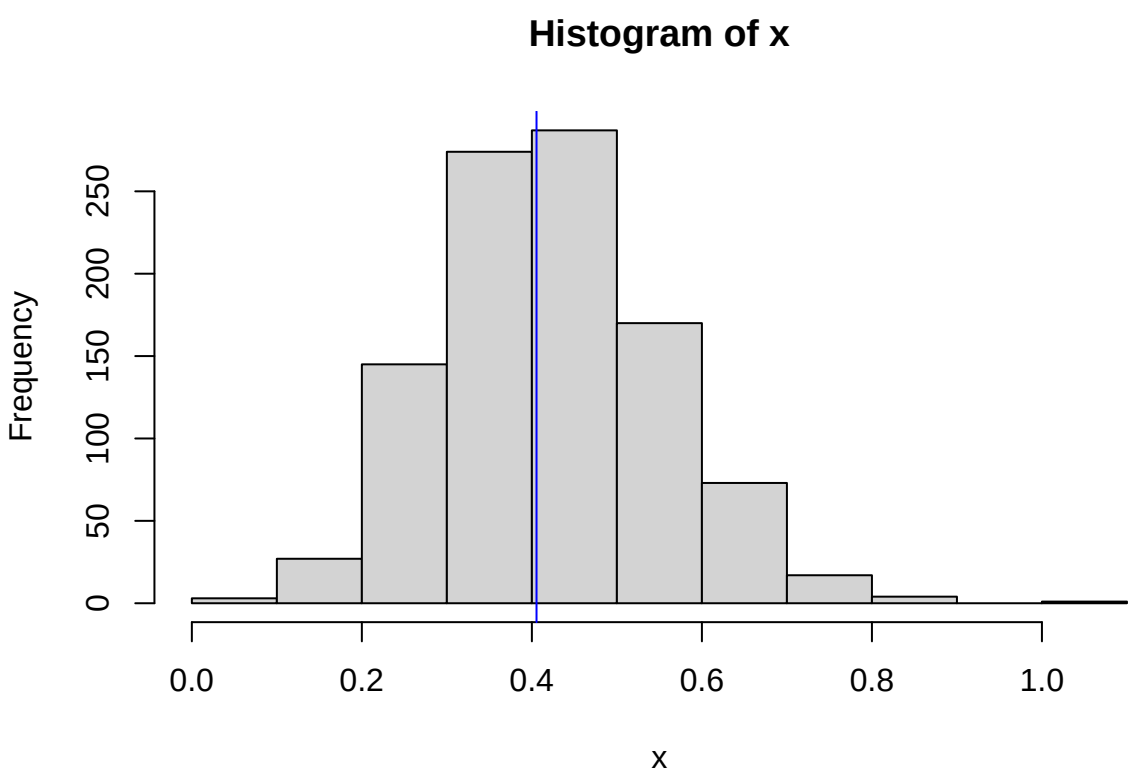
(iv) Draw histogram of the estimated MLEs of θ.

```
hist(x)
```



(v) Draw a vertical line using `abline` function at the true value of θ.

```
hist(x)
abline(v=log(1.5),col="blue")
```



(vi) Use 'quantile' function on estimated θ's to find the 2.5 and 97.5-percentile points.

```
y=quantile(x, probs = c(.025, .975))
y
```

```
##      2.5%     97.5%
## 0.1869650 0.6879458
```

5. Check if the gap between 2.5 and 97.5-percentile points are shrinking as sample size `n` is increasing?

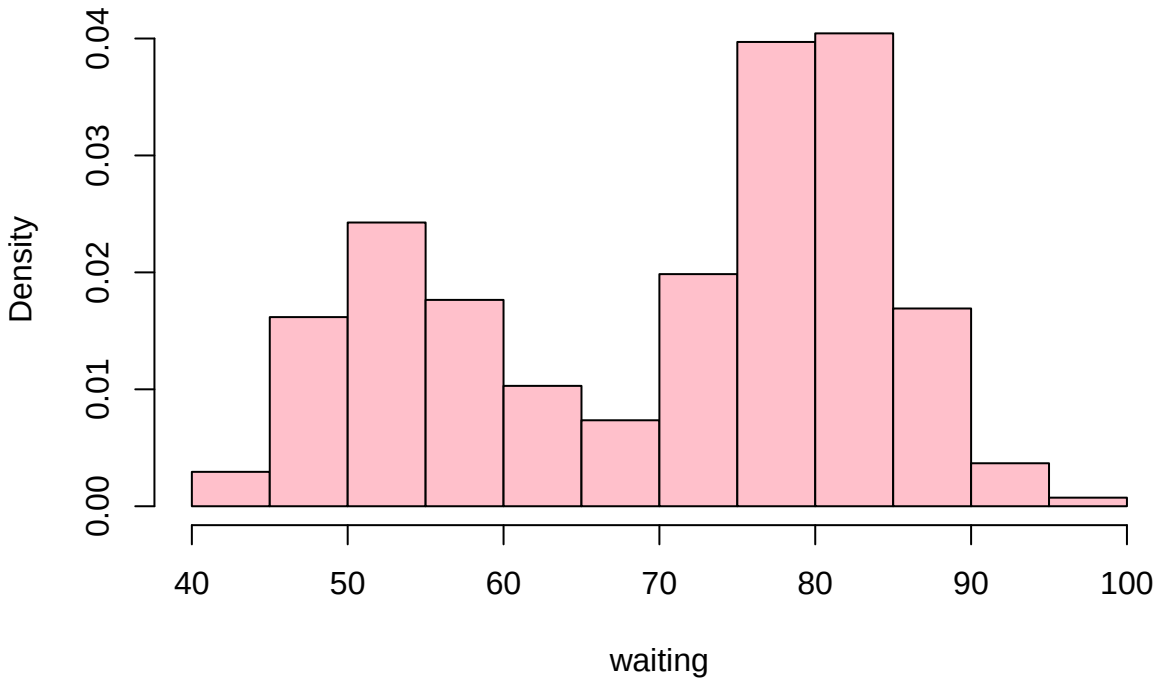
```
#Yes, It does.
```

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.

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

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

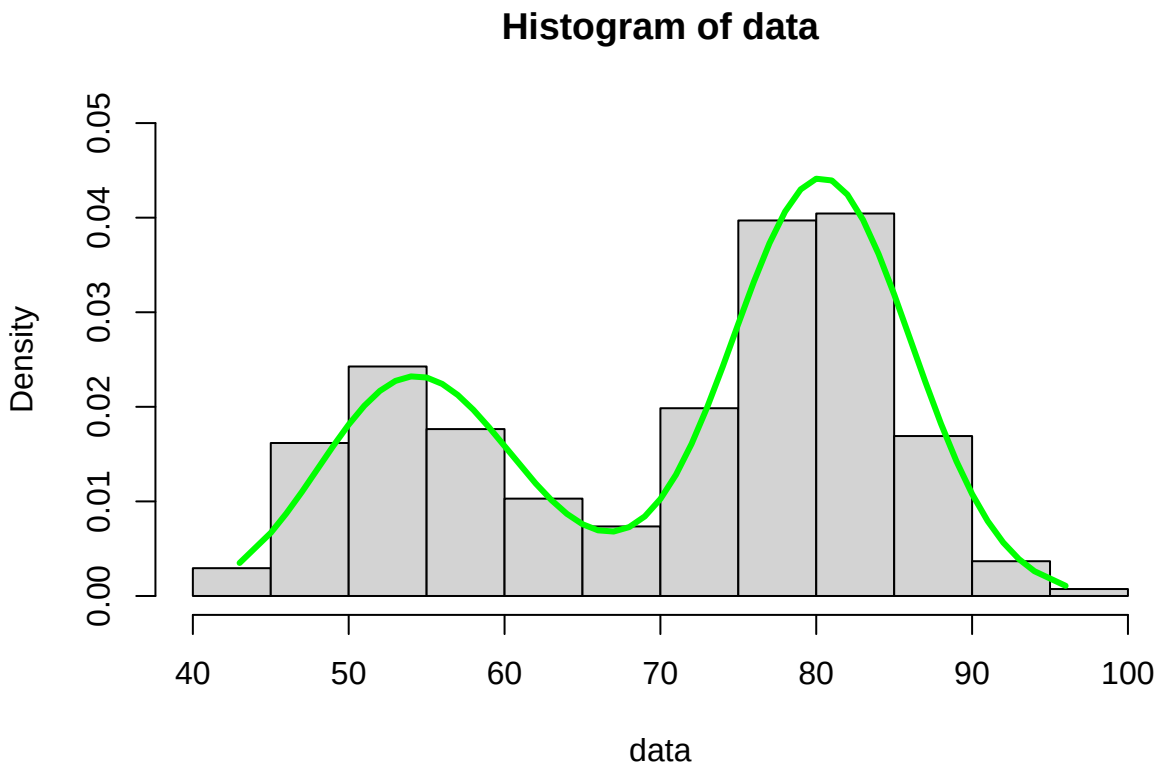
```
data = sort(faithful$waiting)
```

(i) Model 1:

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

```
loglike1 = function(theta,data){
   alpha = exp(theta[1])
   beta = exp(theta[2])
   mu = theta[3]
   sigma = exp(theta[4])
   p = exp(theta[5])/(1+exp(theta[5]))
   n = length(data)
   l=0
   for(i in 1:n){
      l = l + log(p*dgamma(data[i],shape = alpha, rate = beta)
                  +(1-p)*dnorm(data[i], mean = mu, sd = sigma))
   }
   return(-l)
```

```
}

theta_initial=c(4.4,0.47,75,8,0.35)

fit = optim(theta_initial, loglike1, data = data, control = list(maxit=2000))

theta_hat = fit$par
alpha_hat = exp(theta_hat[1])
beta_hat = exp(theta_hat[2])
mu_hat = theta_hat[3]
sigma_hat = exp(theta_hat[4])
p_hat = exp(theta_hat[5])/(1+exp(theta_hat[5]))

d_mle = p_hat*dgamma(data, shape = alpha_hat, rate = beta_hat)+
   (1-p_hat)*dnorm(data, mean = mu_hat,sd = sigma_hat)

hist(data, probability = T, ylim = c(0, 0.05))
lines(data, d_mle,lwd=3,col='green')
```




```
AIC = 2*length(fit$par) - 2*(-fit$value)
## AIC Value for model 1
AIC
```

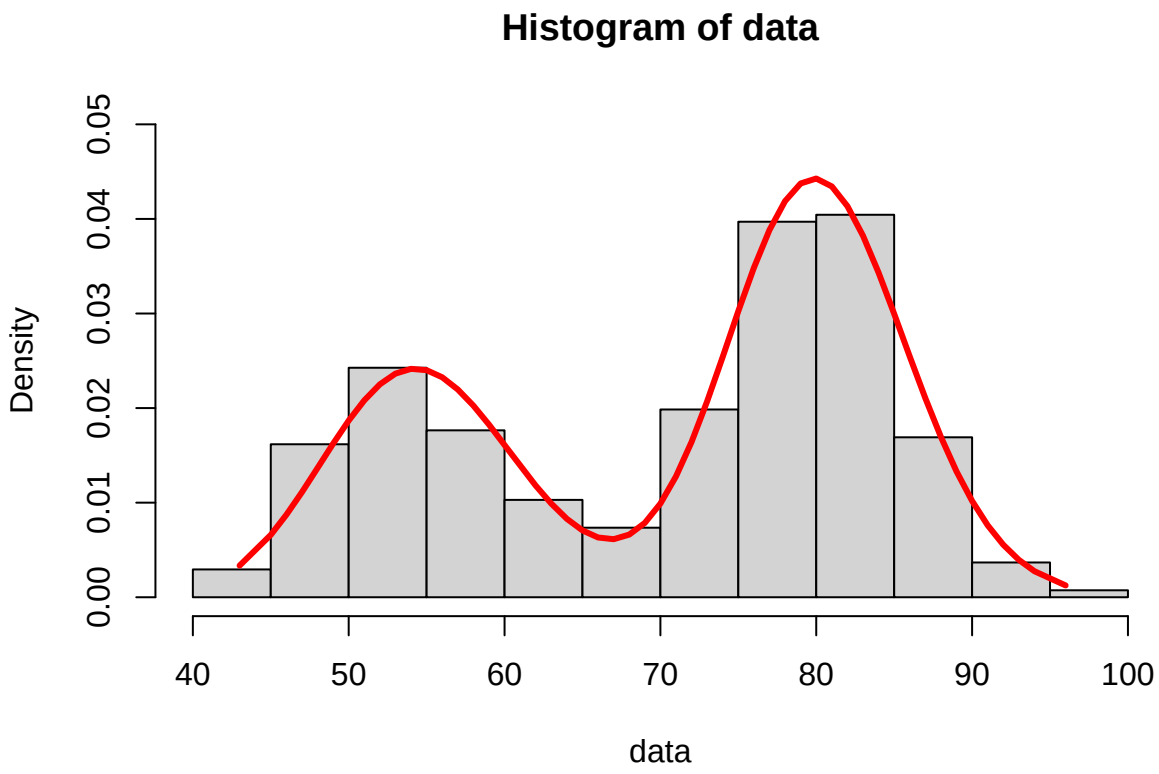
```
## [1] 2076.506
```

AIC value for model 1 is 2076.506

(ii) Model 2:

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

```
loglike2 = function(theta,data){
   alpha1 = exp(theta[1])
   beta1 = exp(theta[2])
   alpha2 = exp(theta[3])
   beta2 = exp(theta[4])
   p = exp(theta[5])/(1+exp(theta[5]))
   n = length(data)
   l=0
   for(i in 1:n){
      l = l + log(p*dgamma(data[i],shape = alpha1, rate = beta1)
                  +(1-p)*dgamma(data[i], shape = alpha2, rate = beta2))
   }
   return(-l)
}

theta_initial = c(4,0,4.4,0,0.35)

fit = optim(theta_initial, loglike2, data = data, control = list(maxit=2000))

theta_hat = fit$par
alpha1_hat = exp(theta_hat[1])
beta1_hat = exp(theta_hat[2])
alpha2_hat = exp(theta_hat[3])
beta2_hat = exp(theta_hat[4])
p_hat = exp(theta_hat[5])/(1+exp(theta_hat[5]))

d_mle = p_hat*dgamma(data, shape = alpha1_hat, rate = beta1_hat)+
   (1-p_hat)*dgamma(data, shape = alpha2_hat, rate = beta2_hat)

hist(data, probability = T, ylim = c(0, 0.05))
lines(data, d_mle,lwd=3,col='red')
```




```
AIC = 2*length(fit$par) - 2*(-fit$value)
## AIC Value for model 1
AIC
```

```
## [1] 2076.117
```

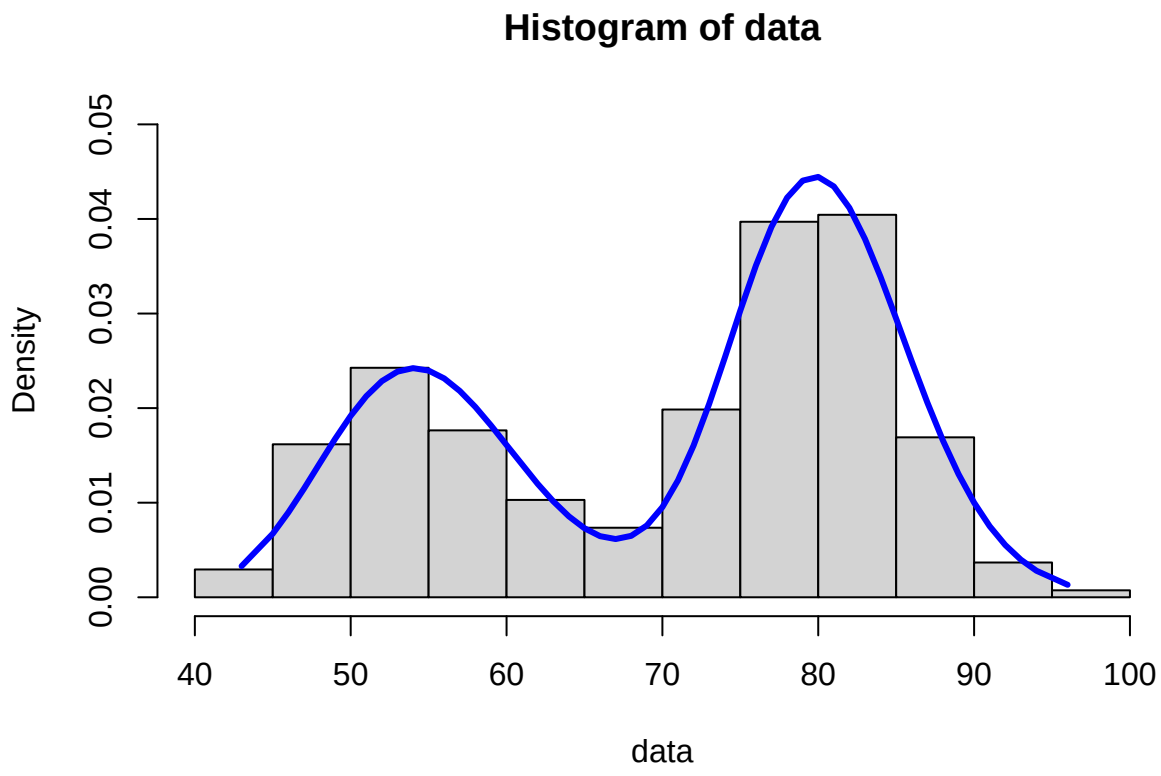
AIC value for model 2 is 2076.117

(iii) Model 3:

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

```
loglike3 = function(theta,data){
   mu1 = theta[1]
   sigma1 = exp(theta[2])
   mu2 = theta[3]
   sigma2 = exp(theta[4])
   p = exp(theta[5])/(1+exp(theta[5]))
   n = length(data)
   l=0
   for(i in 1:n){
      l = l + log(p*dlnorm(data[i],meanlog = mu1,sdlog = sigma1)
             +(1-p)*dlnorm(data[i],meanlog = mu2,sdlog = sigma2))
```

```
   }
   return(-l)
}

theta_initial = c(2.76,-2.25,4.4,-2.6,0.35)

fit = optim(theta_initial, loglike3, data = data, control = list(maxit=2000))

theta_hat = fit$par
mu1_hat = theta_hat[1]
sigma1_hat = exp(theta_hat[2])
mu2_hat = theta_hat[3]
sigma2_hat = exp(theta_hat[4])
p_hat = exp(theta_hat[5])/(1+exp(theta_hat[5]))

d_mle = p_hat*dlnorm(data,meanlog = mu1_hat,sdlog = sigma1_hat)+
      (1-p_hat)*dlnorm(data,meanlog = mu2_hat,sdlog = sigma2_hat)

hist(data, probability = T, ylim = c(0, 0.05))
lines(data, d_mle,lwd=3,col='blue')
```




```
AIC = 2*length(fit$par) - 2*(-fit$value)
## AIC Value for model 1
AIC
```

```
## [1] 2075.433
```

AIC value for model 3 is 2075.433

Akaike information criterion(AIC)

Suppose that we have a statistical model of some data. Let k be the number of estimated parameters in the model. Let $\hat{L}$ be the maximized value of the likelihood function for the model. Then the AIC value of the model is the following.

$$\text{AIC} = 2k - 2\ln(\hat{L})$$

Given a set of candidate models for the data, the preferred model is the one with the minimum AIC value.

Conclusion:

Comparing AIC values for the three given models, we can observe that AIC value of model 3 is minimum among them making it the best model for the given data.

Required probability using best model.

```
dMix = function(x,theta){
  mu1 = theta[1]
  sigma1 = theta[2]
  mu2 = theta[3]
  sigma2 = theta[4]
  p = theta[5]
  f = p*dlnorm(x,meanlog = mu1,sdlog = sigma1)+(1-p)*dlnorm(x, meanlog = mu2, sdlog = sigma2)
  return(f)
}

integrate(dMix,60,70,c(mu1_hat,sigma1_hat,mu2_hat,sigma2_hat,p_hat))
```

```
## 0.09112692 with absolute error < 1e-15
```

$$\mathbb{P}(60 < \texttt{waiting} < 70) = 0.09112692$$

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.

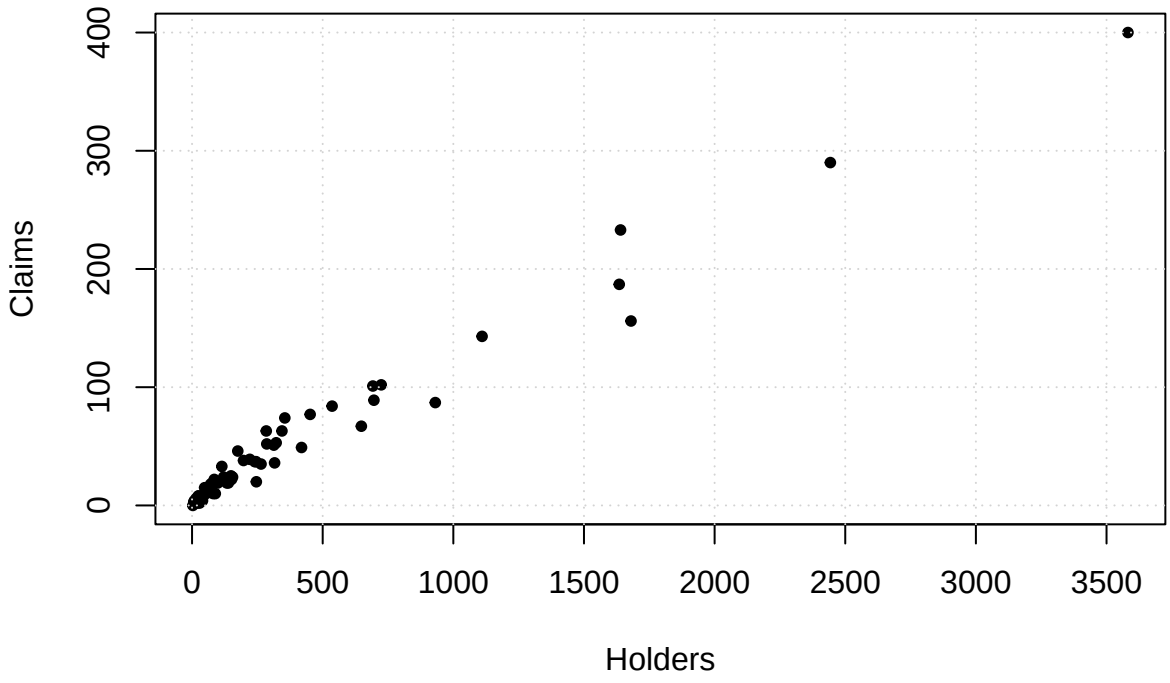
`Group` (an ordered factor): group of car with levels <1 litre, 1–1.5 litre, 1.5–2 litre, >2 litre.

`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,

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

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

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

```
library(SciViews)
library(MASS)
```

```
library(jmuOutlier)
Holders=Insurance$Holders
Claims=Insurance$Claims
data=data.frame(cbind(Claims,Holders))
data=data[-61,]
n=length(Holders)-1
y=data[,1]
x=data[,2]
```

```
Negloglike=function(data,theta)
{
  l=0
  for(i in 1:n)
  {
    l=l+log(dnorm(y[i], theta[1]+theta[2]*x[i],theta[3]))

  }
  return(-l)
}
theta=c(0.1,0.1,50)
fit=optim(theta,Negloglike,data=data)
##Estimated value of theta is:
c(fit$par[1],fit$par[2],fit$par[3])
```

```
## [1]  8.3084803  0.1125138 11.9133879
```

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

```
BIC_A=ln(n)*(length(fit$par))+2*fit$value
#BIC value is:
BIC_A
```

```
## [1] 503.405
```

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 $\theta = (\beta_0, \beta_1, \sigma)$

```
Negloglike=function(data,theta)
{
  l=0
  for(i in 1:n)
  {
```

```
    l=l+log(dlaplace(y[i], theta[1]+theta[2]*x[i],theta[3]))

  }
  return(-l)
}
theta=c(0.1,0.1,50)
fit=optim(theta,Negloglike,data=data)
##Estimated value of theta is:
c(fit$par[1],fit$par[2],fit$par[3])
```

```
## [1]  5.2021496  0.1165771 11.6746589
```

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

```
BIC_B=ln(n)*(length(fit$par))+2*fit$value
#BIC value is:
BIC_B
```

```
## [1] 491.7071
```

Part C: We want to fit the following models:

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

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

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 $\theta = (\alpha, \beta, \sigma)$

```
Negloglike=function(data,theta)
{
  l=0
  for(i in 1:n)
  {
    l=l+log(dlnorm(y[i], theta[1]+theta[2]*log(x[i]),theta[3]))

  }
  return(-l)
}

theta=c(0.1,0.1,1)
fit=optim(theta,Negloglike,data=data)
##Estimated value of theta is:

c(fit$par[1],fit$par[2],fit$par[3])
```

```
## [1] -1.0243551  0.8479072  0.3293700
```

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

```
BIC_C=ln(n)*(length(fit$par))+2*fit$value
#BIC value is:
BIC_C
```

```
## [1] 452.6034
```

Part D: We want to fit the following models:

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

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

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

```
e=2.718281828459045
Negloglike=function(data,theta)
{
  l=0
  for(i in 1:n)
  {
    l=l+log(dgamma(y[i], e^(theta[1]+theta[2]*log(x[i])),theta[3]))

  }
  return(-l)
}

theta=c(0.1,0.1,0.1)
fit=optim(theta,Negloglike,data=data)

##Estimated value of theta is:

c(fit$par[1],fit$par[2],fit$par[3])
```

```
## [1] -1.6430902  0.8371016  0.4858613
```

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

```
BIC_D=ln(n)*(length(fit$par))+2*fit$value
#BIC value is:
BIC_D
```

```
## [1] 437.3382
```

(iii) Compare the BIC of all three models

```
c(BIC_A,BIC_B,BIC_C,BIC_D)
```

```
## [1] 503.4050 491.7071 452.6034 437.3382
```

Problem 5: Computational Finance - Modelling Stock prices

Following piece of code download the prices of TCS since 2007

```
library(quantmod)
```

```
## Warning: package 'quantmod' was built under R version 4.2.2
```

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

```
## Warning: package 'xts' was built under R version 4.2.2
```

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

```
## Warning: package 'zoo' was built under R version 4.2.2
```

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

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

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

```
## Warning: package 'TTR' was built under R version 4.2.2
```

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

```
getSymbols('TCS.NS')
```

```
## 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"
```

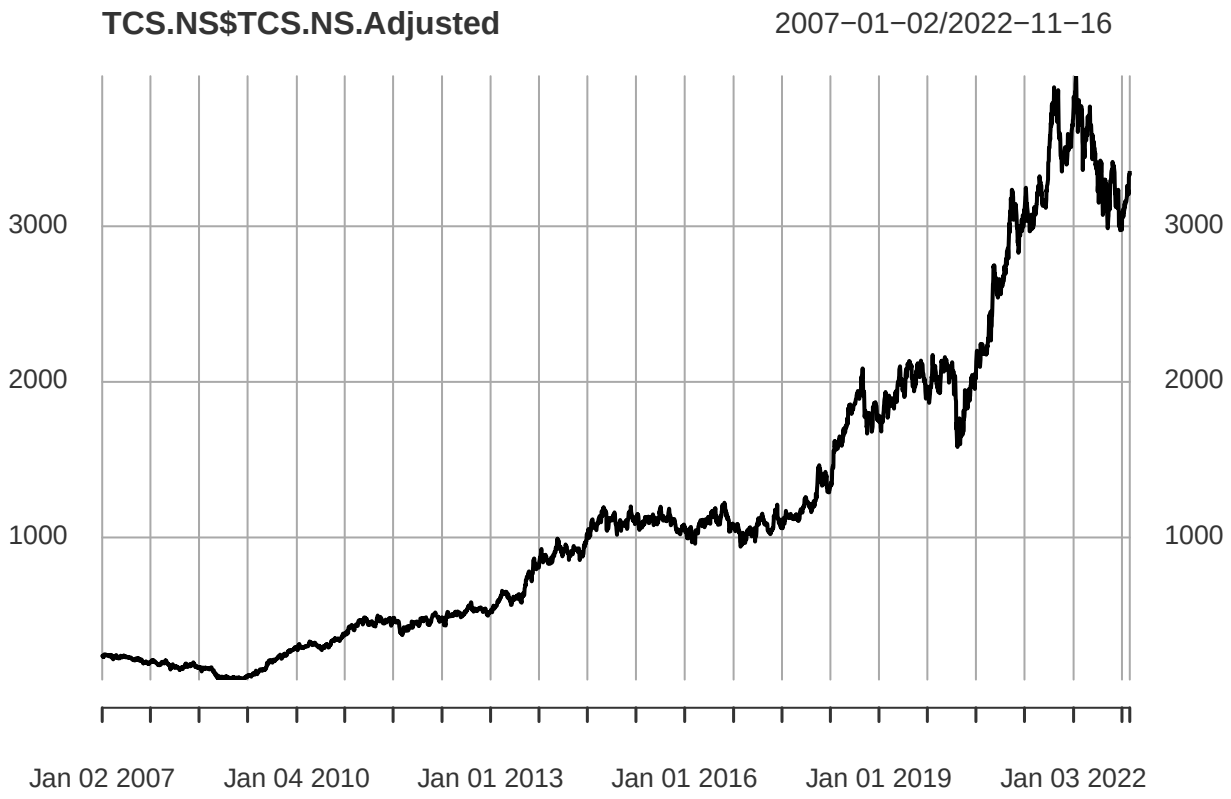
```
tail(TCS.NS)
```

```
##            TCS.NS.Open TCS.NS.High TCS.NS.Low TCS.NS.Close TCS.NS.Volume
## 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
```

```
## 2022-11-16      3338.9     3367.90    3321.45      3355.35       1748235
##            TCS.NS.Adjusted
## 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
## 2022-11-16         3355.35
```

Plot the adjusted close prices of TCS

```
plot(TCS.NS$TCS.NS.Adjusted)
```



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

```
getSymbols('^NSEI')
```

```
## 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"
```

```
tail(NSEI)
```

```
##            NSEI.Open NSEI.High NSEI.Low NSEI.Close NSEI.Volume NSEI.Adjusted
## 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
## 2022-11-16  18398.25  18442.15 18344.15   18409.65      219300      18409.65
```

Plot the adjusted close value of Nifty50

```
plot(NSEI$NSEI.Adjusted)
```

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.

```
TCS_rt = diff(log(TCS.NS$TCS.NS.Adjusted))
Nifty_rt = diff(log(NSEI$NSEI.Adjusted))
retrn = cbind.xts(TCS_rt,Nifty_rt)
retrn = na.omit(data.frame(retrn))
plot(retrn$NSEI.Adjusted,retrn$TCS.NS.Adjusted
     ,pch=20
     ,xlab='Market Return'
     ,ylab='TCS Return'
     ,xlim=c(-0.18,0.18)
     ,ylim=c(-0.18,0.18))
grid(col='grey',lty=1)
```

- 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.

Solution:

```
mu_y = mean(retrn$TCS.NS.Adjusted)
mu_x = mean(retrn$NSEI.Adjusted)
sigma_y = sd(retrn$TCS.NS.Adjusted)
sigma_x = sd(retrn$NSEI.Adjusted)
rho = cor(retrn$NSEI.Adjusted, retrn$TCS.NS.Adjusted)
alpha_0 = mu_y - rho * mu_x * (sigma_x/sigma_y)
beta_0 = rho *  (sigma_x/sigma_y)
retrn_mom = retrn
retrn_mom$mom_est_TCS.NS.Adjusted = alpha_0 + beta_0 * retrn_mom$NSEI.Adjusted
epsilon = retrn_mom$TCS.NS.Adjusted-retrn_mom$mom_est_TCS.NS.Adjusted
sigma_0 = sd(epsilon)
theta_0 = c(alpha_0, beta_0, sigma_0)
print( theta_0)
```

```
## [1] 0.0005872137 0.3904446996 0.0169110427
```

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

Solution:

```
lm_model = lm(TCS.NS.Adjusted~NSEI.Adjusted, data = retrn)
co = lm_model$coefficients
alpha_1 = matrix(co)[1,1]
beta_1 = matrix(co)[2,1]
sigma_1 = sd(lm_model$residuals)
theta_1 = c(alpha_1, beta_1, sigma_1)
print(theta_1)
```

```
## [1] 0.0004633048 0.7436615274 0.0161807372
```

3. Fill-up the following table

Parameters	Method of Moments	OLS
α		
β		
σ		

Solution:

Parameters	Method of Moments	OLS
α	0.0005848199	0.0004611208
β	0.3904739457	0.7436970826
σ	0.0169171740	0.0161865264

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?

Solution:

```
prediction_mom = function(Nifty_initial_value, Nifty_final_value, TCS_initial_value){
  beta = theta_0[2]
  alpha = theta_0[1]
  x = log(Nifty_final_value) - log(Nifty_initial_value)
  y = alpha + beta*x
  y1 = log(TCS_initial_value) + y
  y2 = exp(y1)
  return(y2)
}
prediction_ols = function(Nifty_initial_value, Nifty_final_value, TCS_initial_value){
  beta = theta_1[2]
  alpha = theta_1[1]
  x = log(Nifty_final_value) - log(Nifty_initial_value)
  y = alpha + beta*x
  y1 = log(TCS_initial_value) + y
  y2 = exp(y1)
  return(y2)
}
```

By the Method of Moments type method, we can say that we can expect TCS value to go up by

```
Nifty_initial_value = 18000
Nifty_final_value = 18200
TCS_initial_value = 3200
TCS_final_value = prediction_mom(Nifty_initial_value = Nifty_initial_value,
                                 Nifty_final_value = Nifty_final_value,
                                 TCS_initial_value = TCS_initial_value)
TCS_final_value - TCS_initial_value
```

```
## [1] 15.72351
```

By the OLS method, we can say that we can expect TCS value to go up by

```
TCS_final_value = prediction_ols(Nifty_initial_value = Nifty_initial_value,
                                 Nifty_final_value = Nifty_final_value,
                                 TCS_initial_value = TCS_initial_value)
TCS_final_value - TCS_initial_value
```

```
## [1] 27.89897
```

Thus we can expect TCS price to go up by some value around Rs. 15.72 and Rs. 27.89.