ELEC-E7130 - Internet Traffic Measurements and Analysis

Assignment 5 Distributions and sampling

Haibi Peng 875552

Task 1: Sampling

Solution:

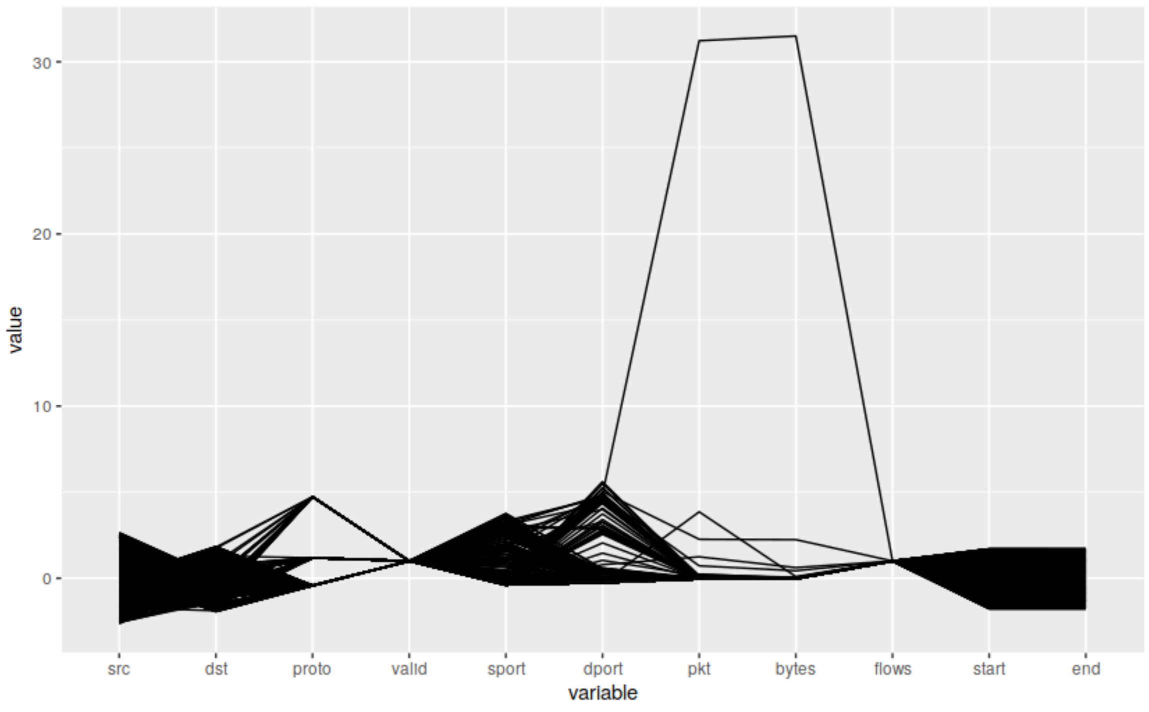
I. Select 1000 random sample data and produce a parallel plot to get an overview of the data.

At first, I imported the data into Rstudio and changed the column names using this command:

```
> colnames(flowdata)<-
c("src","dst","proto","valid","sport","dport","pkt","bytes","flows","start","end")
```

Then I selected 1000 random samples from the origin data and produced a parallel plot:

```
> index<-sample((1:dim(flowdata)[1]),1000)
> ggparcoord(flowdata[index,])
```

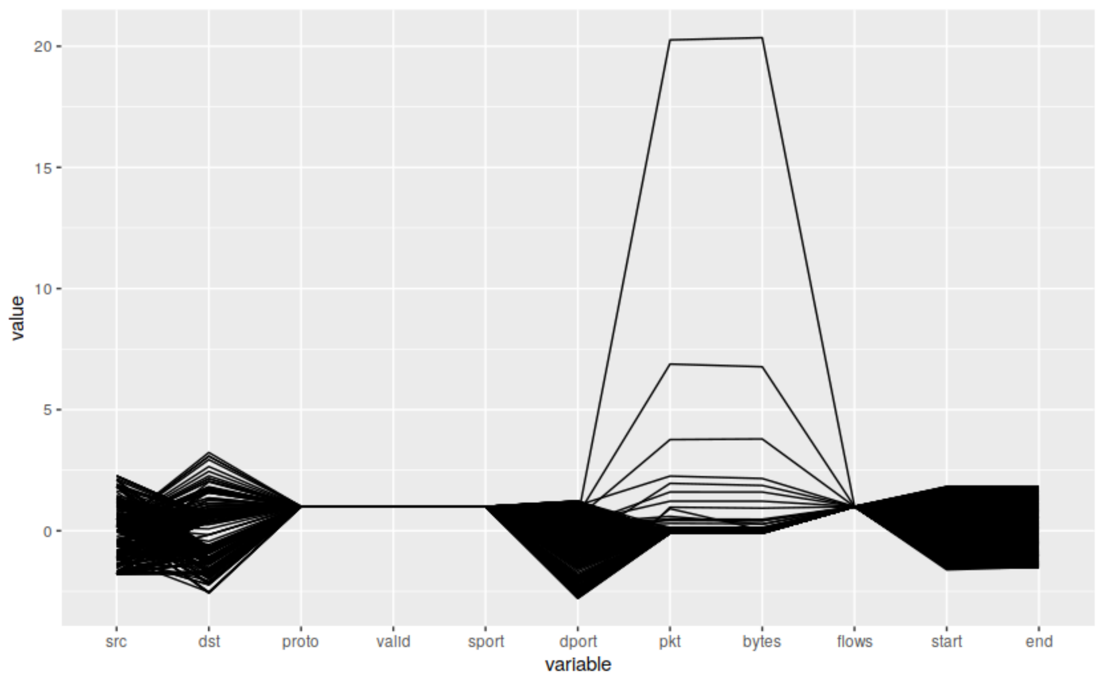


Parallel plot of 1000 random sample data

II. Repeat the same procedure for flows with source port 80 (WWW).

The commands used are as follows:

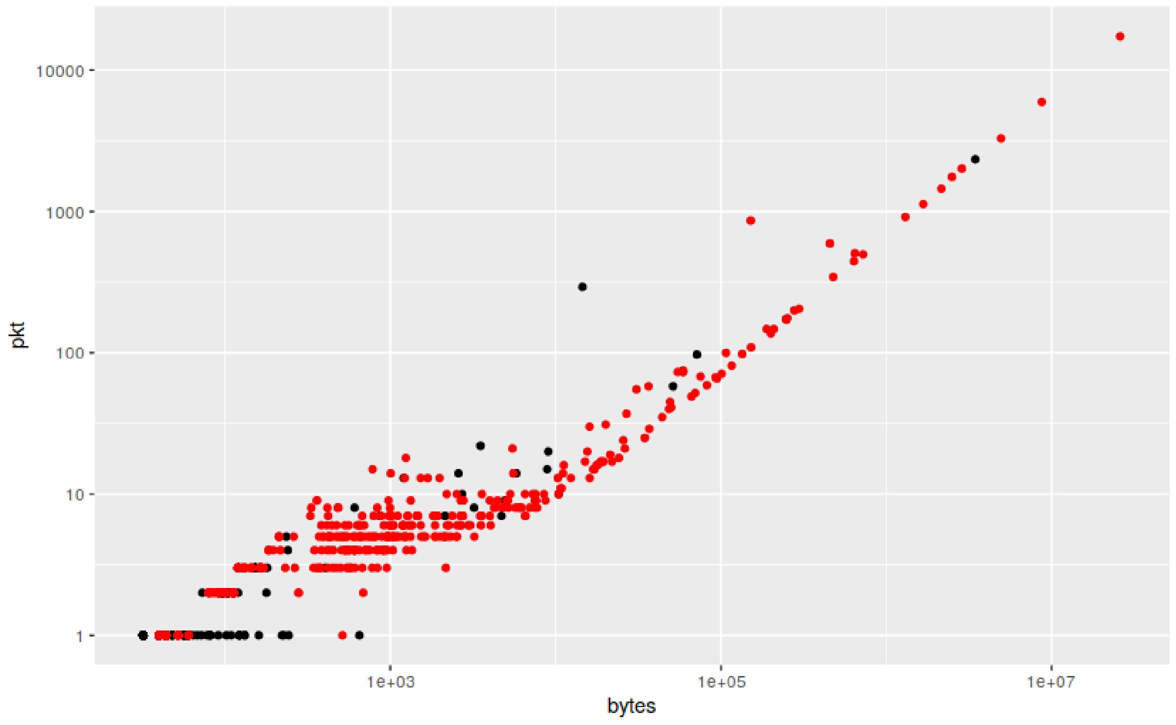
```
> www<-flowdata[flowdata$sport == 80,]
> ggparcoord(www)
```



Parallel plot of data for flows with source port 80

III. Create a scatterplot based on number of bytes against packets and use logarithmic data if necessary. How are they related? What is the maximum average packet size?

```
> ggplot(data=flowdata[index,]) + geom_point(mapping = aes(x=bytes, y=pkt)) + scale_y_log10
() + scale_x_log10() + geom_point(data=www, colour='red',mapping=aes(x=bytes,y=pkt))
```



scatterplot based on number of bytes against packets in logarithmic scale

Based on the plot, we can see that basically most of flows have less than 10 packets and different packets have different sizes in bytes. When numbers of packets grow, the numbers of bytes grow accordingly. Note that this plot is in logarithmic scale both for packets and bytes, so basically there is a linear correlation between packets and bytes.

As for the maximum average packet size, the commands used are as follows:

```
> ps<-vector("double",100000)
> for(i in seq(1,100000)){ps[i]<-(flowdata[i,8]/flowdata[i,7])}
> summary(ps)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  28.00   32.00   32.00   42.95   32.00 1500.00
```

So the maximum average packet size should be 1500 bytes per packet.

IV. Study the average throughput of the connections. Average throughput is the number of bytes transferred divided by the transfer time. Clock resolution introduces some challenges, what can be said on the throughput of the flows that are transferred in zero time?

In order to calculate the average throughput of the connections, the commands used are as follows:

```
> throughput<-flowdata$bytes/(flowdata$end-flowdata$start)
> view(throughput)
```

But due to some zero transfer time, it gets many **Inf** values, like this:

1	Inf
2	Inf
3	Inf
4	Inf
5	Inf
6	Inf
7	32. 45394
8	Inf
9	Inf
10	Inf

So I took out the **Inf** values and made calculations again:

```
> INF = which(throughput==Inf)
> throughput0 = throughput[-INF]
> View(throughput0)
> summary(throughput0)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.      Max.
       1       32       87  1128592     1272 198180864
```

So the average throughput should be 1128592 bytes per second, which is 1.08 Mbytes per second. The throughput of the flows that are transferred in zero time is defined as peak measured throughput. Peak measured throughput is throughput measured by a real, implemented system, or a simulated system. The value is the throughput measured over a short period of time, which is mathematically the limit taken with respect to throughput as time approaches zero. This term is synonymous with ***instantaneous throughput***. This number is useful for systems that rely on burst data transmission; however, for systems with a high duty cycle this is less likely to be a useful measure of system performance.

V. State your own observations on the data.

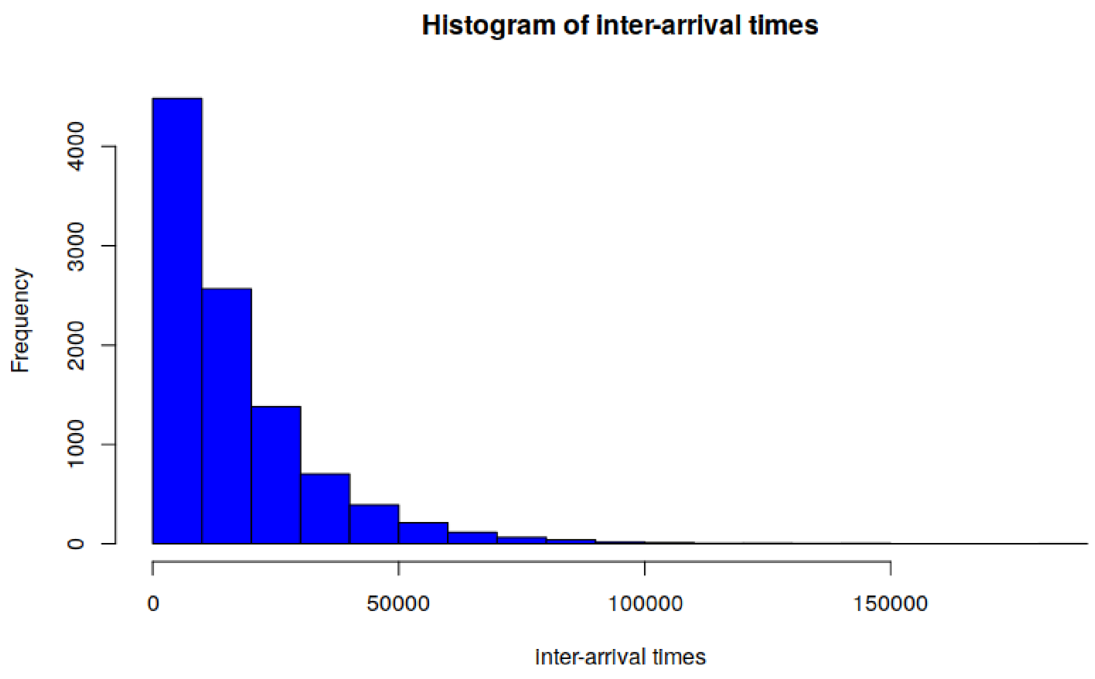
As we can see from the data, there are a lot of port numbers are 0, which is officially a reserved port in TCP/IP networking, meaning that it should not be used for any TCP or UDP network communications. However, port 0 sometimes takes on a special meaning in network programming, particularly Unix socket programming, where port 0 is a programming technique for specifying system-allocated (dynamic) ports. Configuring a new socket connection requires assigning a TCP or UDP port number. Instead of hard-coding a particular port number, or writing code that searches for an available port on the local system, network programmers can instead specify port 0 as a connection parameter. That triggers the operating system to automatically search for and return the next available port in the dynamic port number range. Basically, when the source port or destination port in 0, the throughput is considered as instantaneous throughput since the start and end is the same.

Task 2: Sampling and distributions

Solution:

1. Plot the histogram of the original data and compute the mean.

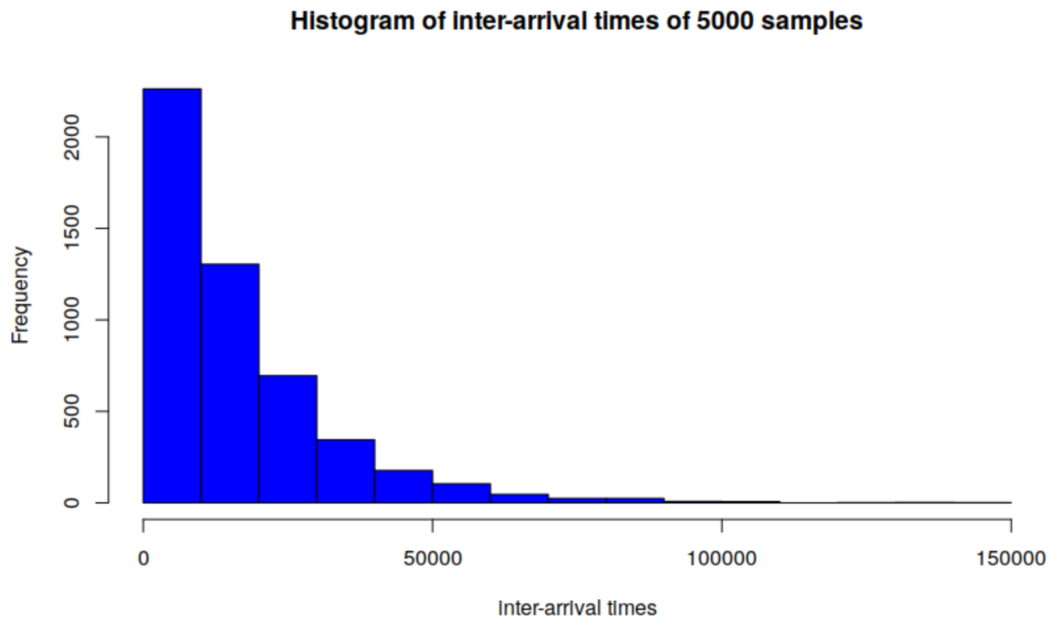
```
> hist(round(sampling[,]),col="blue", xlab = "inter-
arrival times", main = "Histogram of inter-arrival time")
> summary(sampling)
       V1
 Min.   :     1.39
 1st Qu.:  4731.55
 Median : 11503.40
 Mean   : 16432.25
 3rd Qu.: 22410.16
 Max.   :181717.71
```



And the mean value of the inter-arrival time should be ***16432.25***.

2. Select 5000 random samples from original data (i.e., you should have a vector of length 5000 values). Plot its histogram and compute the mean.

```
> index<-sample((1:dim(sampling)[1]),5000)
> hist(round(sampling[index,]),col="blue", xlab = "inter-
arrival times", main = "Histogram of inter-arrival times of 5000 samples")
> summary(sampling[index,])
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
    2.89   4743.42  11395.47  16008.87  21781.81 147635.67
```



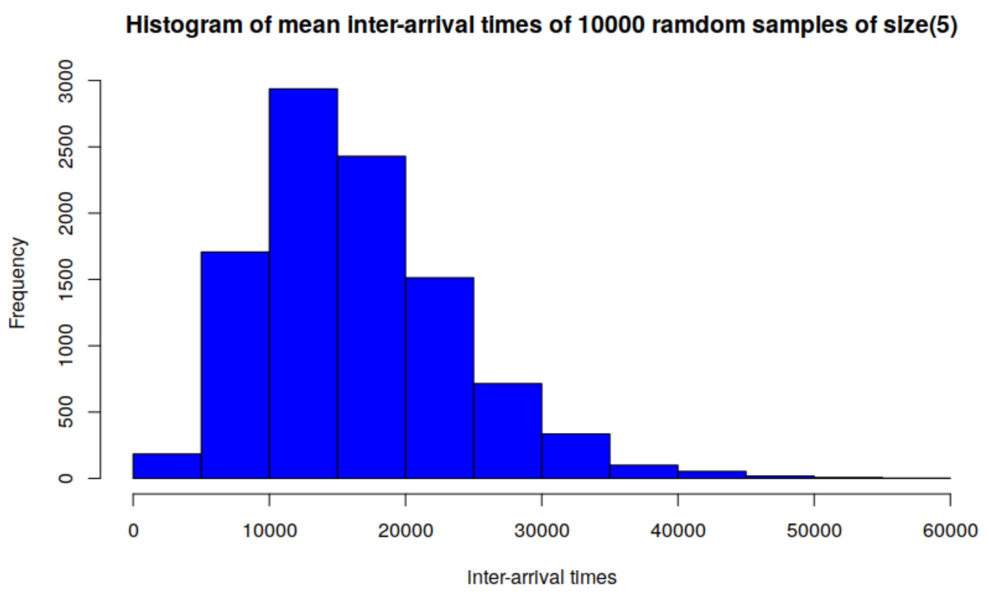
And the mean value of the inter-arrival time of 5000 random samples should be ***16008.87***.

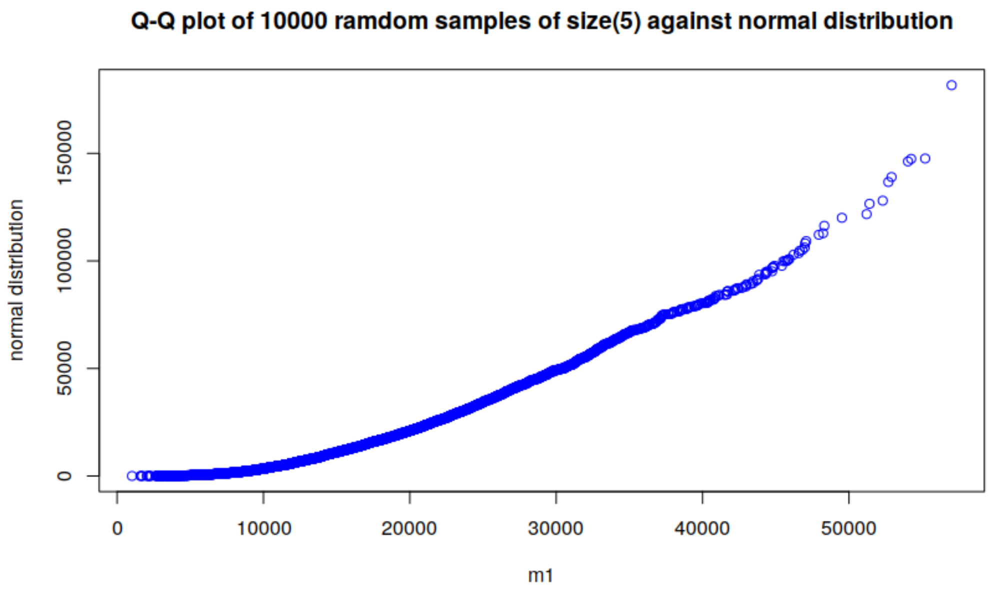
Following we generate 10000 random samples of different sizes (**n**) from the data. Select 10000 times n random elements from the data and compute mean of these n values. As a result, you should have a vector of 10000 values, each of them is mean value of n random elements. These values represent different results you could get for your statistic in a random sample and can be seen as samples from the sampling distribution of the sample mean statistic for n samples.

In addition to histogram of these 10000 values, study the values in a Q-Q plot against normal distribution and compute the mean and standard deviations of these 10000 values.

3. n=5

```
> m1<-vector("double",10000)
> for(i in seq(1,10000)){m1[i]<-mean(sampling[sample(1:dim(sampling)[1],5),])}
> hist(round(m1),col="blue", xlab = "inter-arrival times", main = "Histogram of mean inter-arrival times of 10000 ramdom samples of size(5)")
> summary(m1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1019   11090   15294   16434   20644   57014
> qqnorm(sampling[,], ylab = "normal distribution", main="Q-Q plot of 10000 ramdom samples of size(5) against normal distribution", col="blue")
> sd(m1)
[1] 7333.584
```



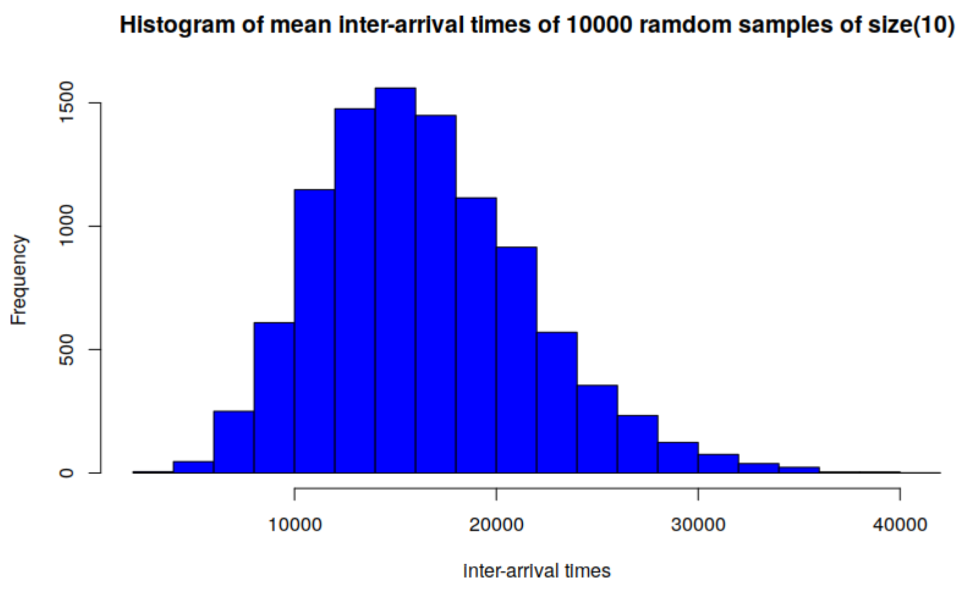


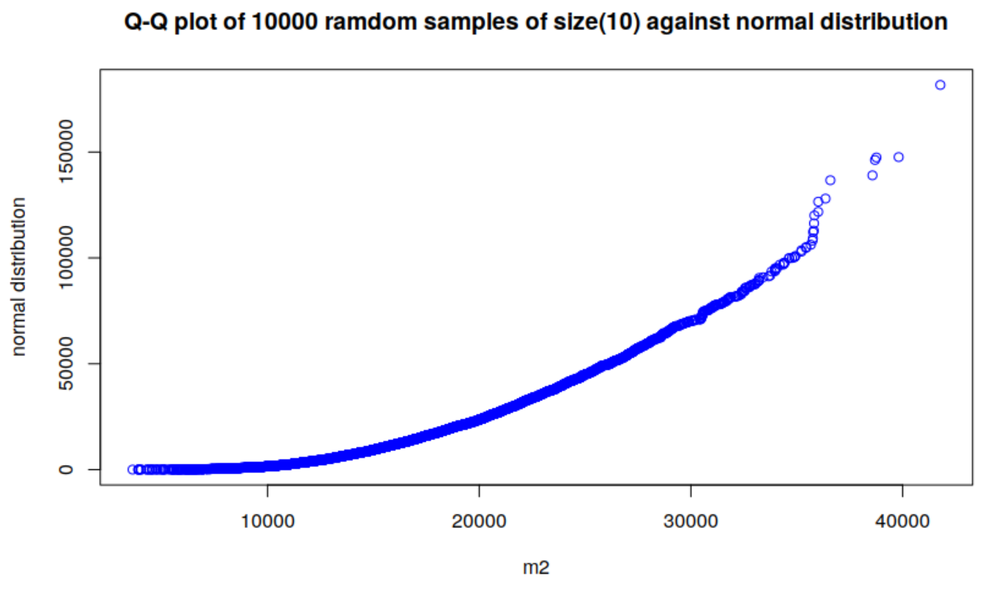
And the mean value is ***16434***, standard deviations is ***7333.584***.

4. n=10

```
> m2<-vector("double",10000)
> for(i in seq(1,10000)){m2[i]<-mean(sampling[sample(1:dim(sampling)[1],10),])}
> hist(round(m2),col="blue", xlab = "inter-arrival times", main = "Histogram of mean inter-
arrival times of 10000 ramdom samples of size(10)")
> summary(m2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   3628   12648   15890   16432   19669   41784
```

```
> qqnorm(sampling[,], ylab = "normal distribution", main="Q-
Q plot of 10000 ramdom samples of size(10) against normal distribution", col="blue")> sd(m1
)
> sd(m2)
[1] 5255.351
```

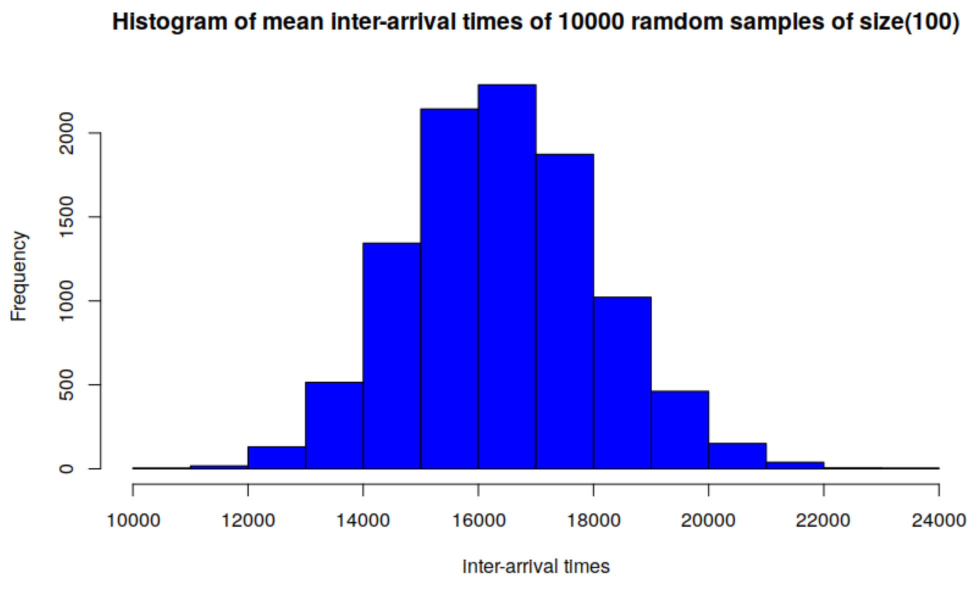


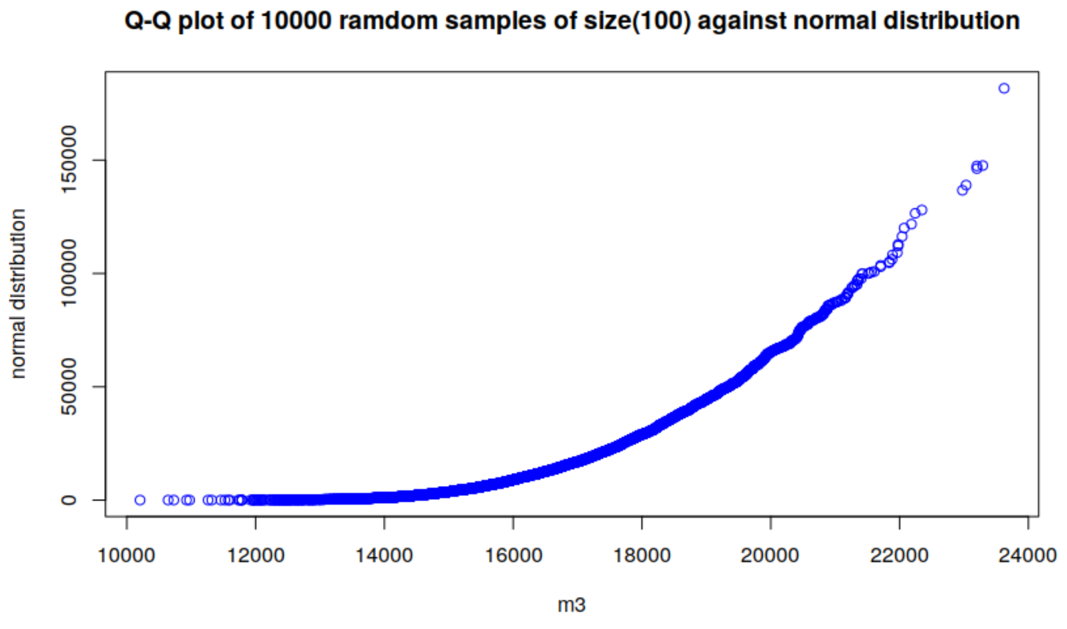


And the mean value is ***16432***, standard deviations is ***5255.351***.

5. n=100

```
> m3<-vector("double",10000)
> for(i in seq(1,10000)){m3[i]<-mean(sampling[sample(1:dim(sampling)[1],100),])}
> hist(round(m3),col="blue", xlab = "inter-arrival times", main = "Histogram of mean inter-arrival times of 10000 ramdom samples of size(100)")
> summary(m3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  10206   15242   16361   16416   17505   23624
> qqnorm(sampling[,], ylab = "normal distribution", main="Q-Q plot of 10000 ramdom samples of size(100) against normal distribution", col="blue")
> sd(m3)
[1] 1673.198
```





And the mean value is ***16416***, standard deviations is ***1673.198***.

6. Discuss the effects of sample size to the sampling distribution and to the accuracy of the estimate.

As we can see from the plots, when the sample size increases, the sampling distribution becomes closer to normal distribution. The standard deviation decreases as the sample size increases. Therefore, the accuracy of the estimate will be improved as the sample size grows.

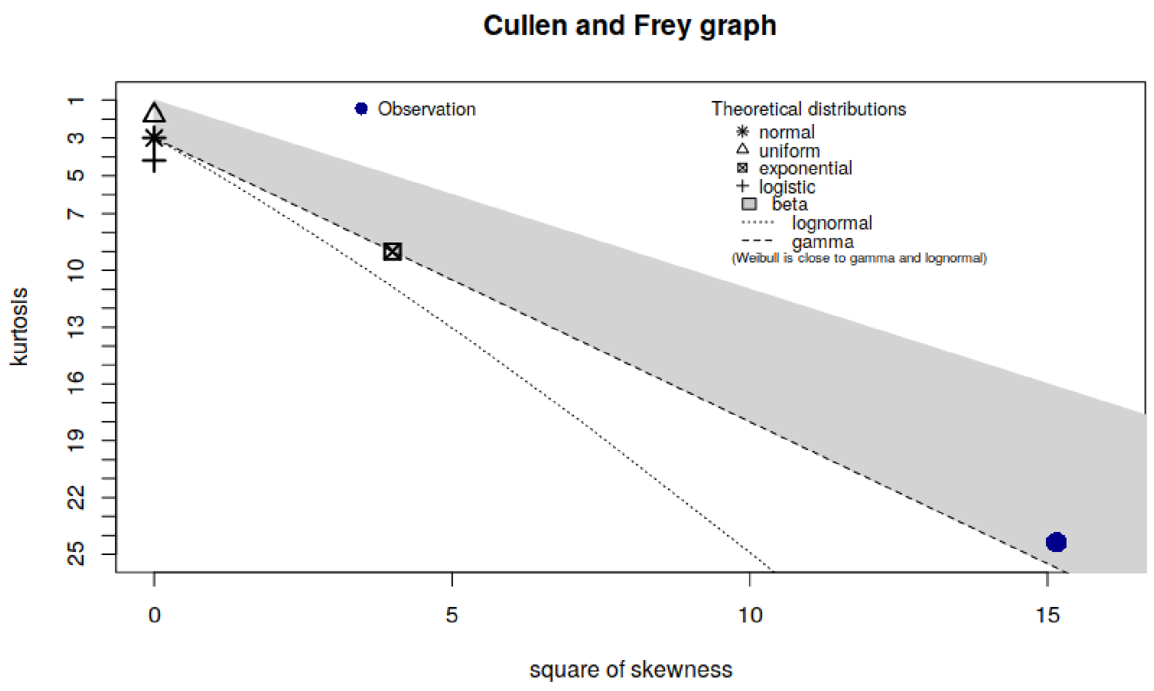
Task 3: Distributions

Solution:

1. For dataset distr_a.txt

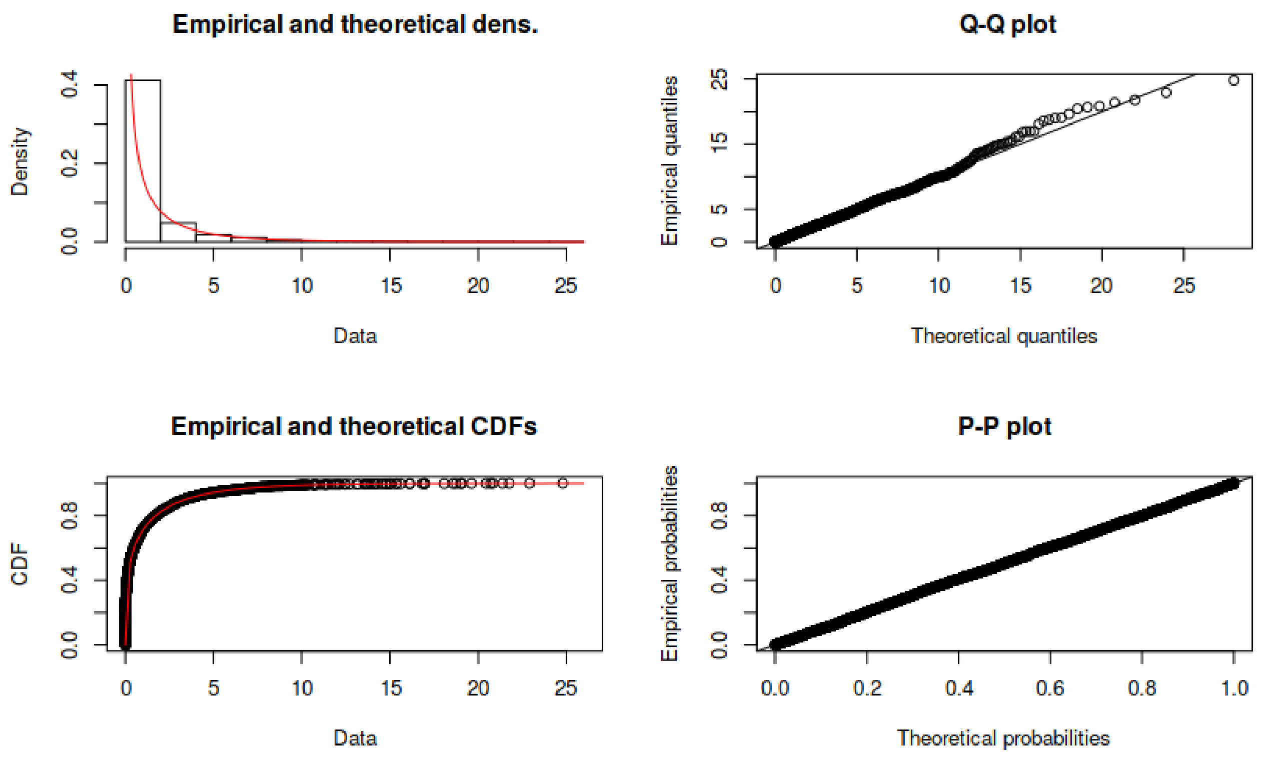
First, I used the ***descdist()*** function in the ***fitdistplus*** package in R to preliminarily check the data set is closest to what distribution.

```
> descdist(distr_a[,1], discrete = FALSE, boot = NULL, method = "unbiased",
+           graph = TRUE, obs.col = "darkblue", obs.pch = 16, boot.col = "orange")
summary statistics
------
min:  8.334182e-13   max:  24.78845
median:  0.2364361
mean:  1.146677
estimated sd:  2.220498
estimated skewness:  3.89215
estimated kurtosis:  24.37036
```



As we can see, the Observation point is in the beta distribution area but it is most closed to the line for gamma distribution. So basically we can say the data set follows the gamma distribution. And the validation is done below:

```
> fit.gamma<-fitdist(distr_a[,1],"gamma")
> plot(fit.gamma)
```



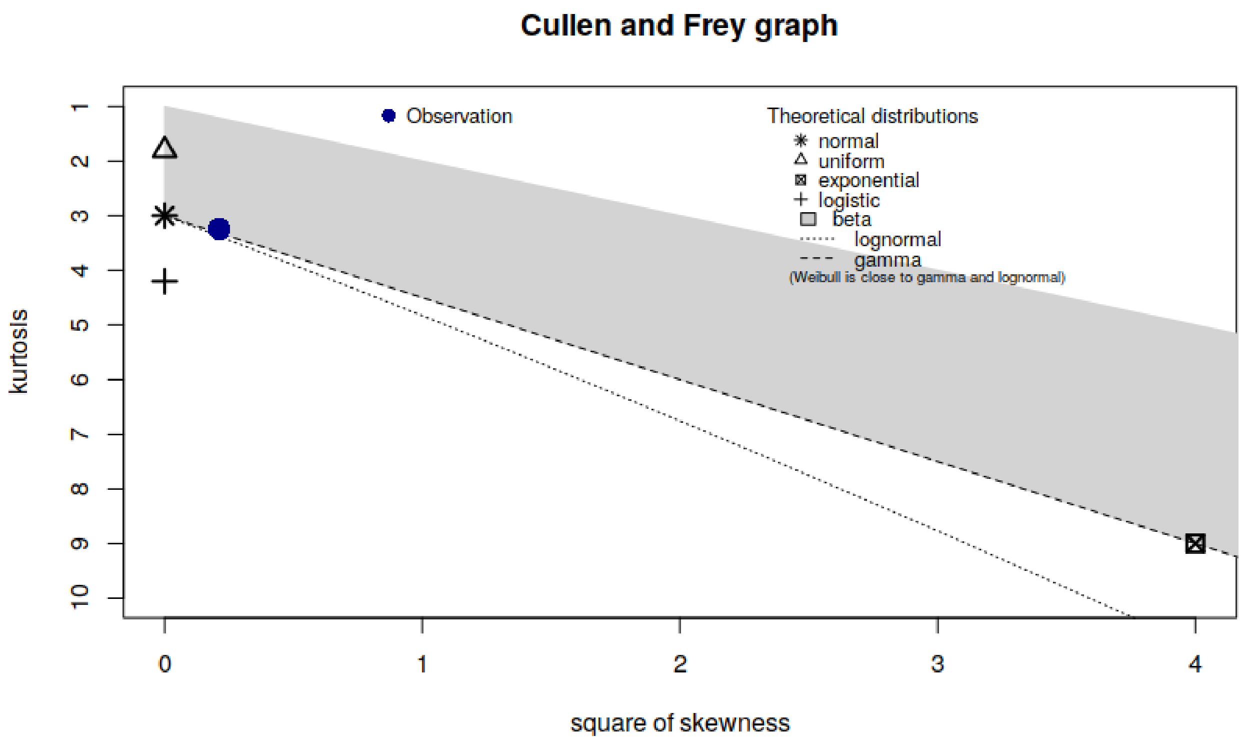
As we can see, the data set fits the gamma distribution quite well. And the estimated parameters of the distribution are as follows:

```
> fitdist(distr_a[,1],"gamma")
Fitting of the distribution ' gamma ' by maximum likelihood
Parameters:
       estimate  Std. Error
shape 0.2759024 0.003917206
rate  0.2406022 0.006785372
```

2. For dataset distr_b.txt

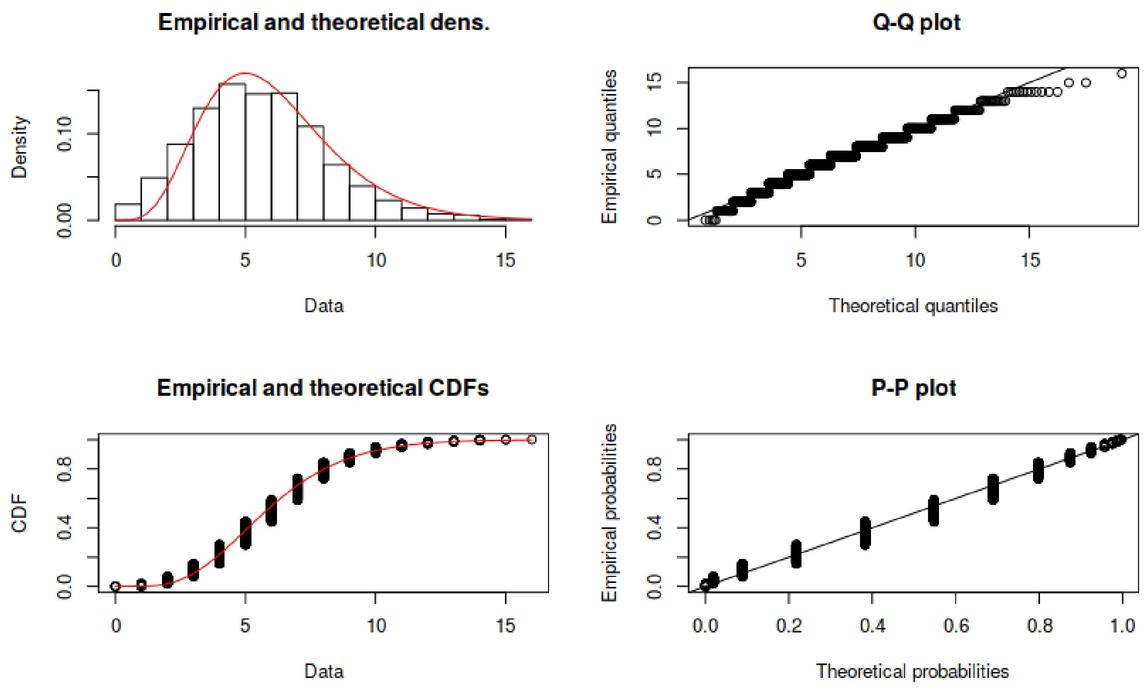
We can do the same check as in previous part:

```
> descdist(distr_b[,1], discrete = FALSE, boot = NULL, method = "unbiased",
+           graph = TRUE, obs.col = "darkblue", obs.pch = 16, boot.col = "orange")
summary statistics
------
min:  0   max:  16
median:  6
mean:  6.052381
estimated sd:  2.543405
estimated skewness:  0.4604355
estimated kurtosis:  3.245417
```



Again, the Observation point is most closed to gamma distribution. So the validation is done below:

```
> fit.gamma<-fitdist(distr_b[,1],"gamma","mme")
> plot(fit.gamma)
```



And the distr_b.txt data set also fits well into the gamma distribution. The estimated parameters of the distribution are as follows:

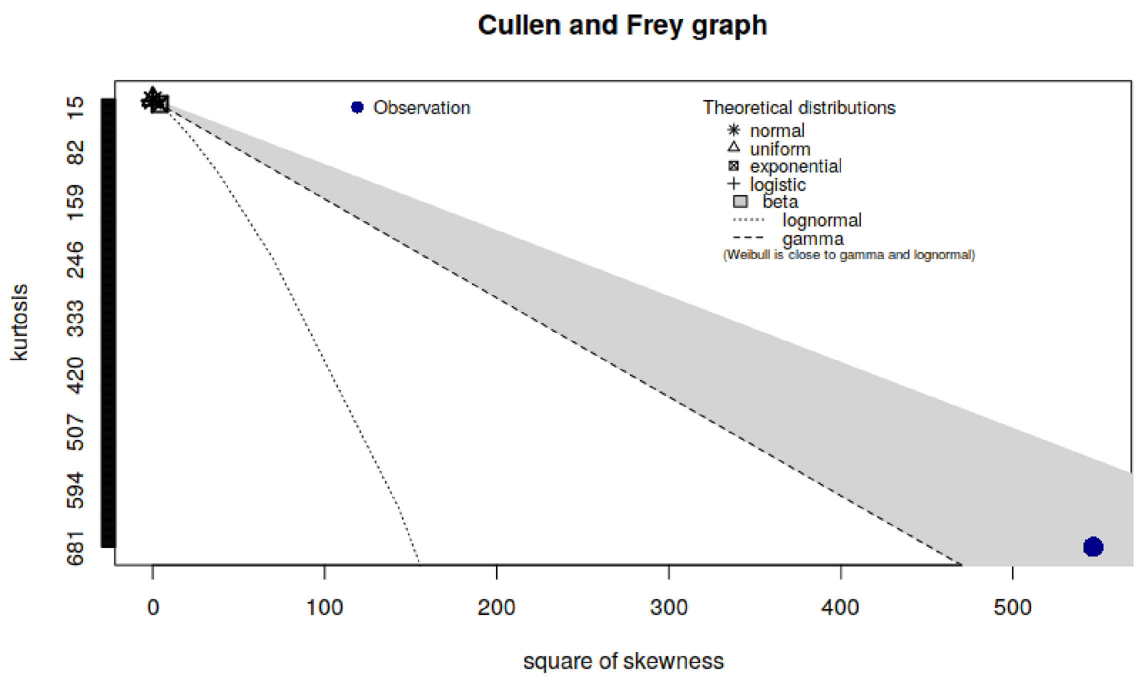
```
> fitdist(distr_b[,1],"gamma", 'mme')
Fitting of the distribution ' gamma ' by matching moments
Parameters:
         estimate
shape 5.6653714
rate  0.9360566
```

However, this time the estimation method is 'mme'(moment method estimations), while in the distr_a data set it is 'mle'(maximum likelihood estimation, default method for ***fitdist()*** function).

3. For dataset distr_c.txt

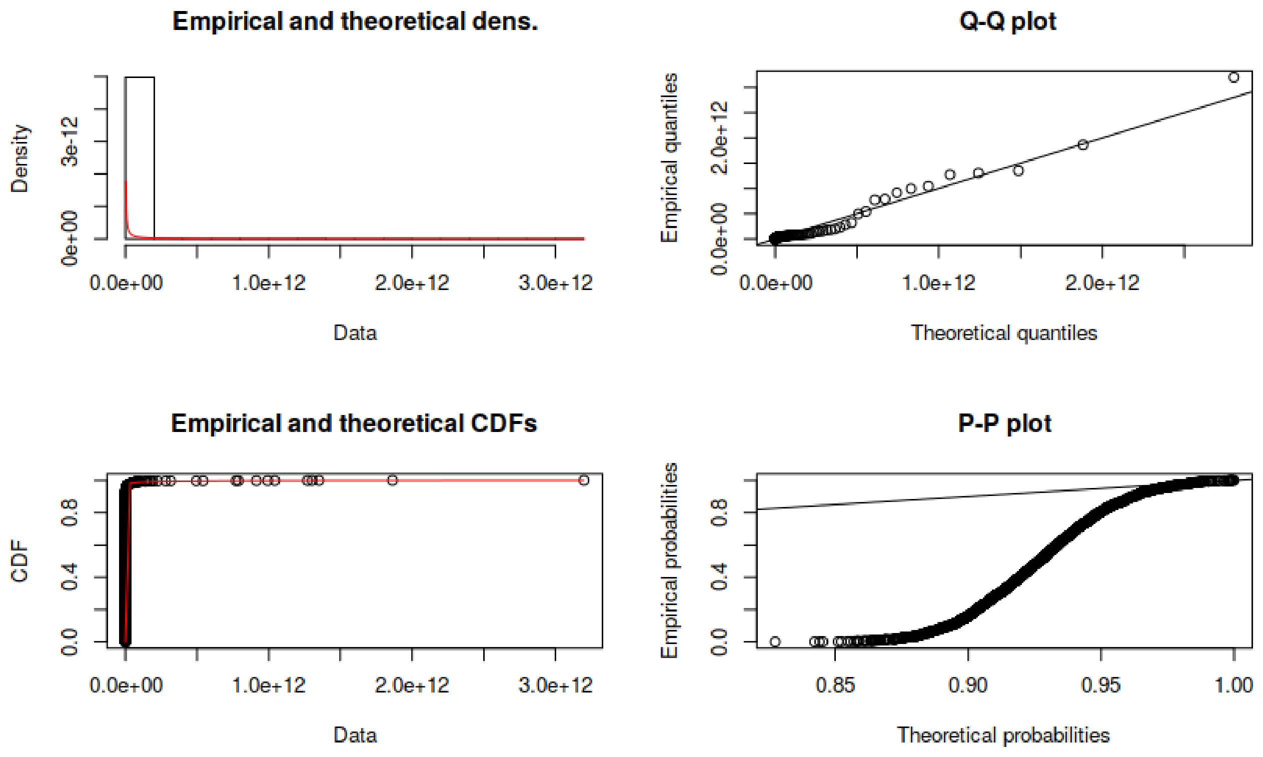
The method is the same as previous ones:

```
> descdist(distr_c[,1], discrete = FALSE, boot = NULL, method = "unbiased",
+           graph = TRUE, obs.col = "darkblue", obs.pch = 16, boot.col = "orange")
summary statistics
------
min:  0.005968803   max:  3.199112e+12
median:  1498015
mean:  6785925289
estimated sd:  88664087008
estimated skewness:  23.37977
estimated kurtosis:  680.6618
```



We can see that the Observation point again is most closed to gamma distribution, but this time it is not as close as in the previous cases. Se let's do the validation:

```
> fit.gamma<-fitdist(distr_c[,1],"gamma","mme")
> plot(fit.gamma)
```



As we can see from the plots, the P-P plot is not fitted as good as that in the previous cases but other plots seem quite well fitted. The estimated parameters of the distribution are as follows:

```
> fitdist(distr_c[,1],"gamma", 'mme')
Fitting of the distribution ' gamma ' by matching moments
Parameters:
          estimate
shape 5.859561e-03
rate  8.634874e-13
```

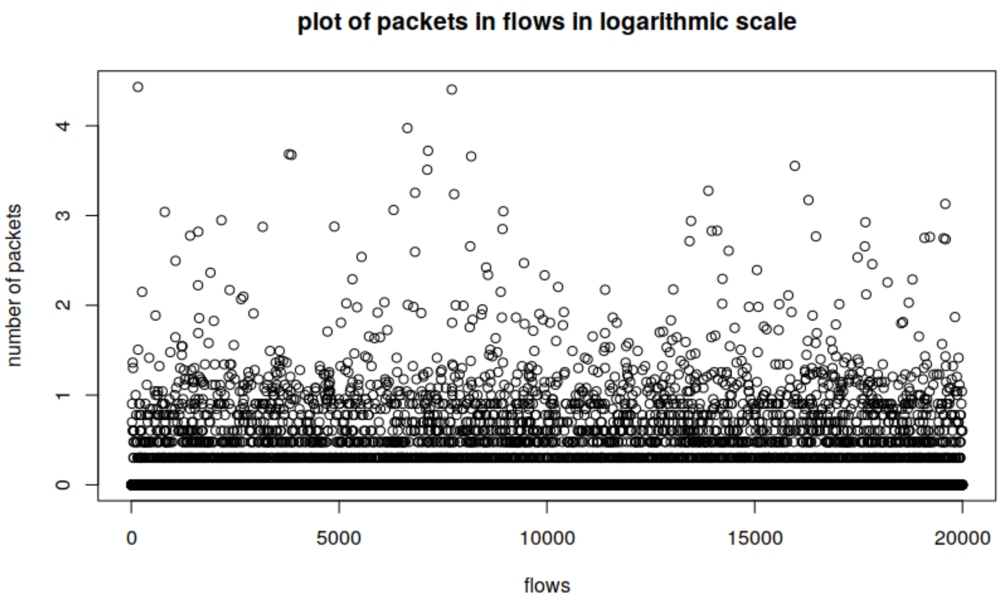
It should be noted that I also plotted some other distributions but in comparison the chosen distribution, which is gamma distribution, is the best fitted. So that is also the reason I chose the gamma distribution.

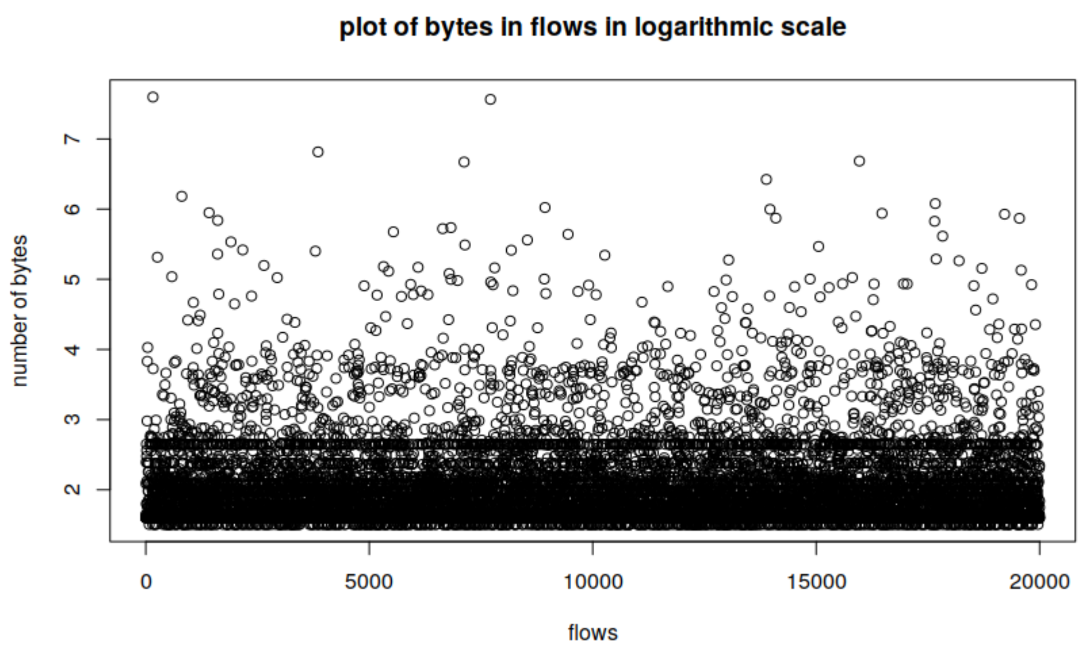
Task 4: High variability

Solution:

1. Plot the data and compute its mean and median for both packets and bytes.

```
> summary(flows)
    packets                bytes
 Min.    :    1.000   Min.    :       32
 1st Qu.:    1.000   1st Qu.:       40
 Median :    1.000   Median :       40
 Mean   :    7.761   Mean   :     6016
 3rd Qu.:    1.000   3rd Qu.:       88
 Max.   :27037.000   Max.   :39563775
> plot(log10(flows[,1]), xlab = "flows", ylab = "number of packets") + title(main = "plot o
f packets in flows in logarithmic scale")
integer(0)
> plot(log10(flows[,2]), xlab = "flows", ylab = "number of bytes") + title(main = "plot of
bytes in flows in logarithmic scale")
integer(0)
```





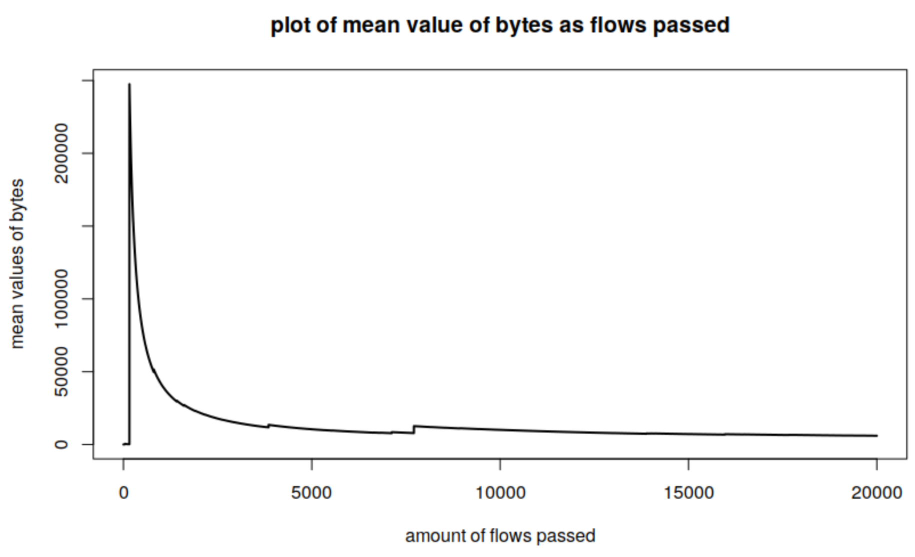
2. Expression for running mean.

As is described in the instruction, the expression for running mean should be like:

$$\overline{Bytes}(n) = {\sum_{i=1}^{n} Bytes(i)}\Big/{n}$$

3. Plot of mean estimate. Explain your observations.

```
> m<-vector("double",20000)
> for(i in seq(1,20000)){m[i]<-mean(flows[1:i,2])}
> plot(m, xlab = "amount of flows passed", ylab = "mean values of bytes", type = "l", lwd =
 2) + title(main = "plot of mean value of bytes as flows passed")
integer(0)
```



As we can see from the plot, the mean values of bytes increase sharply from the beginning of flows to about 160 flows. Then they decrease quickly from peak to a normal values and then decrease slowly until the end. I believe the reason for that is there are some extremely big values in bytes, which increase the mean values sharply around the peak mean value, as we can see from the data plot in precious part.

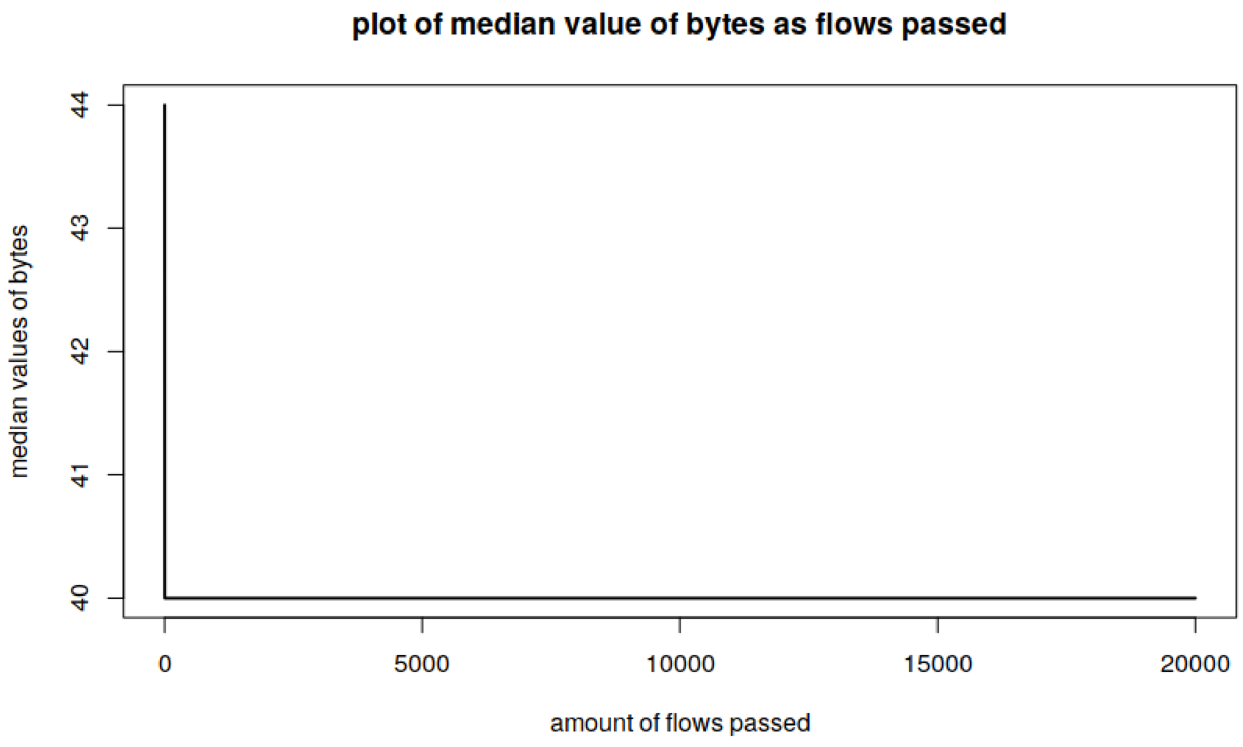
4. Computing median instead of mean. Derive expression.

As is described in the instruction, the expression for running median should be like:

$$Bytes^M(n) = Median([Bytes[1], Bytes[2], \dots, Bytes[n])$$

And the plot should be like:

```
> md<-vector("double",20000)
> for(i in seq(1,20000)){md[i]<-median(flows[1:i,2])}
> plot(md, xlab = "amount of flows passed", ylab = "median values of bytes", type = "l", lw
d = 2) + title(main = "plot of median value of bytes as flows passed")
integer(0)
```


plot of median value of bytes as flows passed
median values of bytes
amount of flows passed