Visualisation: Assignment 1

Name: Roudranil Das Roll Number: MDS202227

2022-09-22

Instruction:

1) Work on the 'Assignment1.Rmd' file. Compile the file as pdf. Submit only the pdf file in moodle.
2) Make sure you write your name and roll number at the top of the 'Assignment1.Rmd' file.

Total Marks: 10 points

Problem 1 (3 points)

Problem Statement: Write an `R` function which will test Central Limit Theorem.

- Assume the underlying population distribution follow Poisson distribution with rate parameter λ
- We want to estimate the unknown λ with the sample mean

$$\hat{\lambda} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

- The exact sampling distribution of $\hat{\lambda}$ is unknown
- But CLT tells us that as sample size n increases the sampling distribution of $\hat{\lambda}$ can be approximated by Gaussian distribution.

Input in the function:

- n: sample size
- λ : rate parameter
- N: simulation size

Output from the function:

- Histogram of the sampling distribution using `ggplot`
- QQ-plot using `ggplot`

Test cases:

- case 1 a: $\lambda = 0.7$, n=10, N=5000
- case 1 b: $\lambda = 0.7$, n=30, N=5000
- case 1 c: $\lambda = 0.7$, n=100, N=5000
- case 1 d: $\lambda = 0.7$, n=300, N=5000
- case 2 a: $\lambda = 1.7$, n=10, N=5000
- case 2 b: $\lambda = 1.7$, n=30, N=5000
- case 2 c: $\lambda = 1.7$, n=100, N=5000

- case 2 d: $\lambda = 1.7$, n=300, N=5000

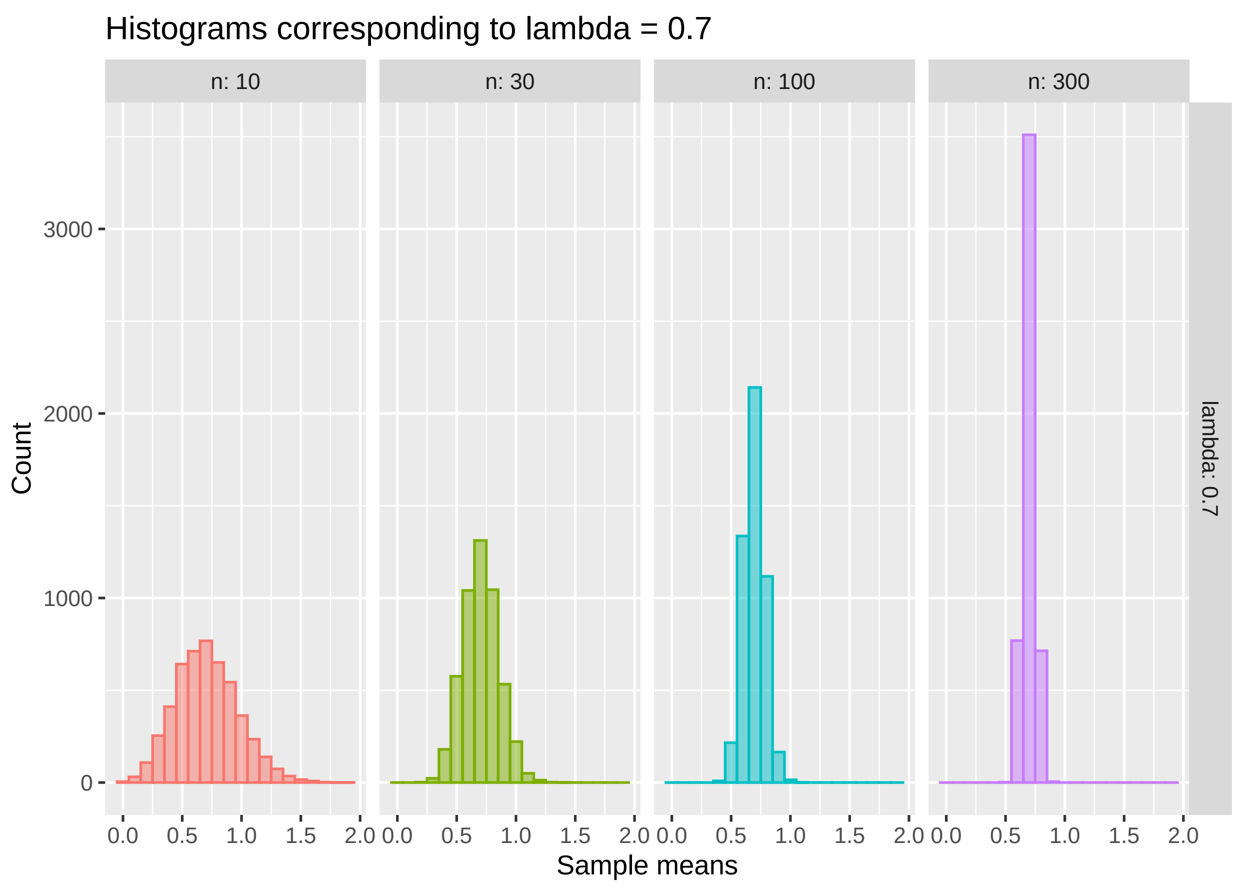
```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
simulate_clt = function(lambda, n, N){
    sim_means = c()
    for(i in 1:N){
        sample = rpois(n, lambda)
        sample_mean = mean(sample)
        sim_means = append(sim_means, sample_mean)
    }
    # we create a dataset with the columns being the sample means and
    # respective values of lambda and n
    data = data.frame(sim_means, rep(lambda, N), rep(n, N))
    colnames(data) = c("mean", "lambda", "n")
    return(data)
}

lambdas = rep(c(0.7, 1.7), times = 1, each = 4)
n = rep(c(10, 30, 100, 300), times = 2)
N = rep(5000, times = 8)
```

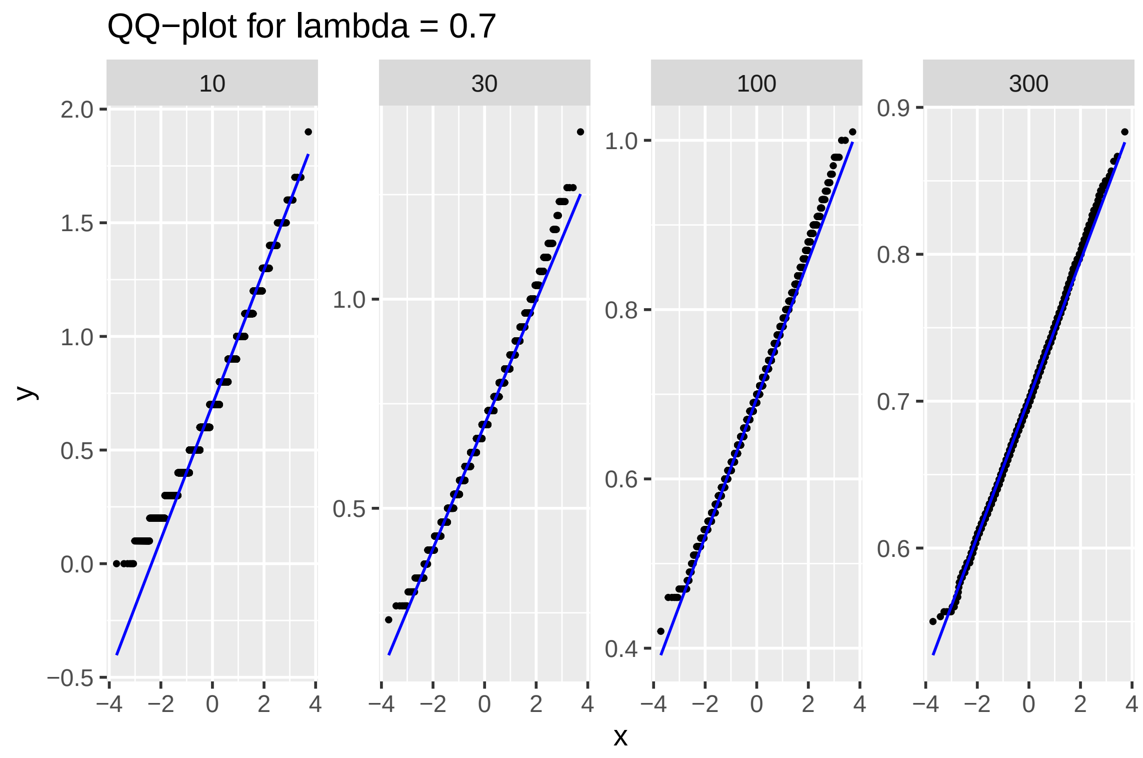
For $\lambda = 0.7$, we get the following 4 histograms:

```
dataset = data.frame()
for(i in 1:4){
    # the datasets all created are binded rowwise
    # this is done to help us to create the facet grids
    dataset = rbind(dataset, simulate_clt(lambdas[i], n[i], N[i]))
}
dataset %>%
    mutate(n = as.factor(n)) %>%
    ggplot(aes(x = mean)) +
    geom_histogram(bins = 20, aes(color = n, fill = n), alpha = 0.5) +
    facet_grid(vars(lambda), vars(n), scales = "free_y", labeller = label_both) +
    theme(legend.position = "none") +
    labs(title = paste("Histograms corresponding to lambda =", lambdas[1], sep = " "),
         x = "Sample means",
         y = "Count")
```



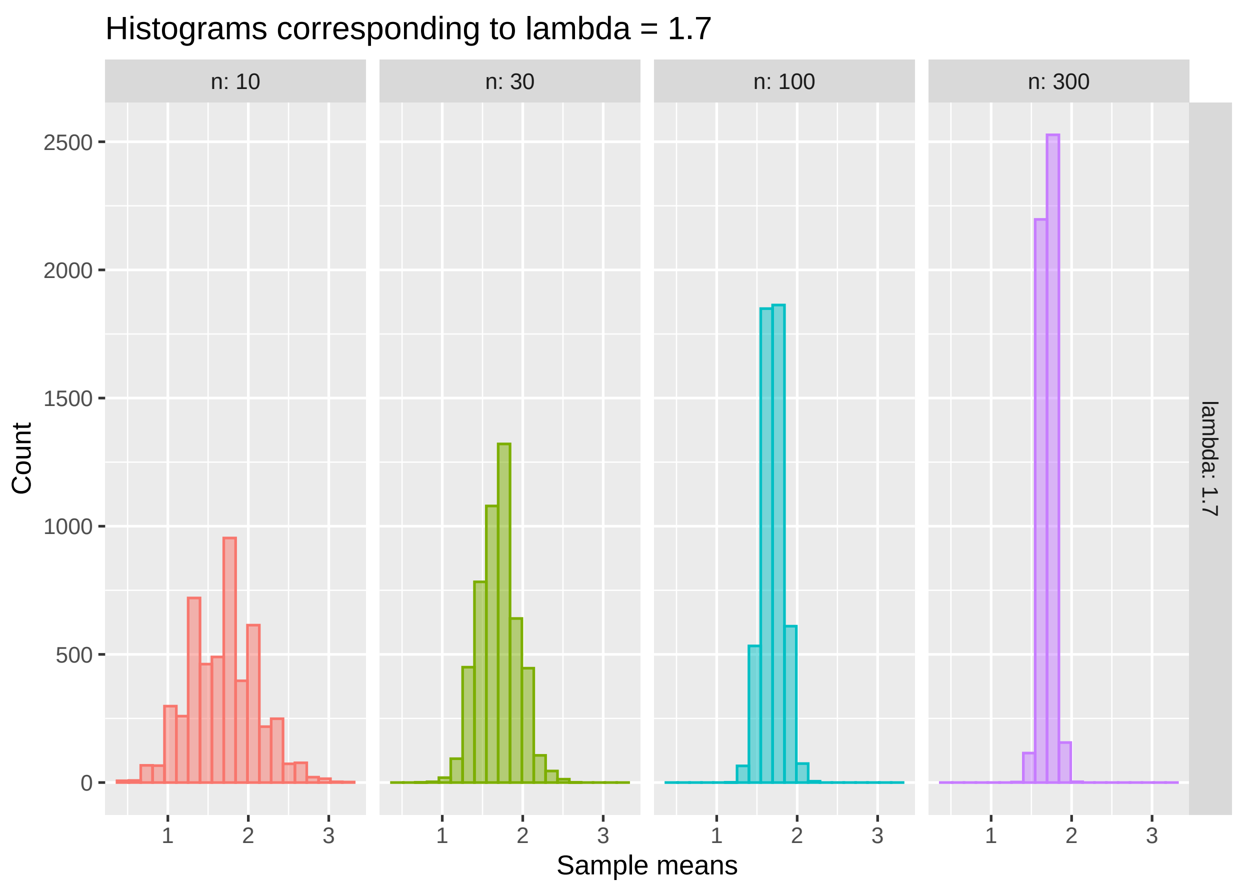
For the same value of λ, we get the following 4 QQ-plots.

```
dataset = data.frame()
for(i in 1:4){
    dataset = rbind(dataset, simulate_clt(lambdas[i], n[i], N[i]))
}
dataset %>%
    ggplot(aes(sample = mean)) +
    geom_qq(size = 0.6) +
    geom_qq_line(color = "blue") +
    facet_wrap(vars(n), scales = "free", nrow = 1) +
    labs(title = "QQ-plot for lambda = 0.7")
```



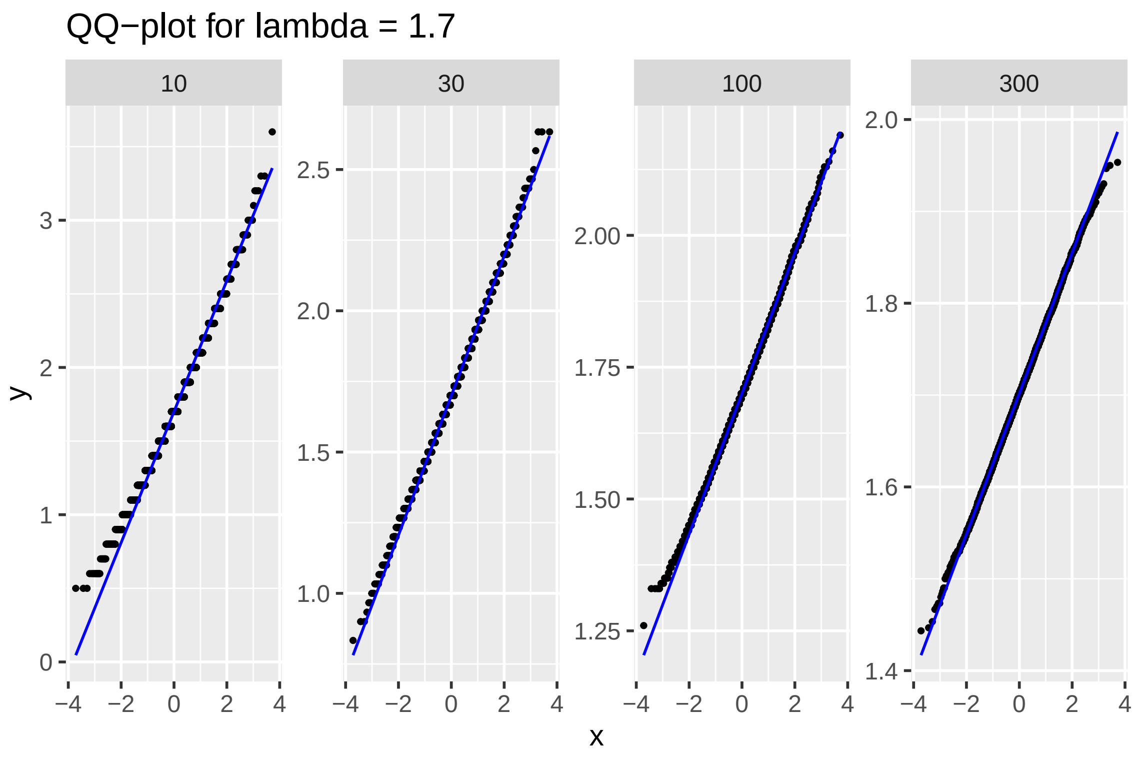
For $\lambda = 1.7$, we get the following 4 histograms:

```
dataset = data.frame()
for(i in 5:8){
    dataset = rbind(dataset, simulate_clt(lambdas[i], n[i], N[i]))
}
dataset %>%
    mutate(n = as.factor(n)) %>%
    ggplot(aes(x = mean)) +
    geom_histogram(bins = 20, aes(color = n, fill = n), alpha = 0.5) +
    facet_grid(vars(lambda), vars(n), scales = "free_y", labeller = label_both) +
    theme(legend.position = "none") +
    labs(title = paste("Histograms corresponding to lambda =", lambdas[5], sep = " "),
         x = "Sample means",
         y = "Count")
```



And we follow this the corresponding QQ-plots:

```r
dataset = data.frame()
for(i in 5:8){
    dataset = rbind(dataset, simulate_clt(lambdas[i], n[i], N[i]))
}
dataset %>%
    ggplot(aes(sample = mean)) +
    geom_qq(size = 0.6) +
    geom_qq_line(color = "blue") +
    facet_wrap(vars(n), scales = "free", nrow = 1) +
    labs(title = "QQ-plot for lambda = 1.7")
```



Problem 2: (2 points)

Consider the `JohnsonJohnson` dataset. The dataset contains the Quarterly earnings (dollars) per Johnson & Johnson share 1960–80.

a) Draw the time series plot of Quarterly earnings in regular scale and log-scale using the `ggplot` (1 point)

```
head(JohnsonJohnson)
```

```
## [1] 0.71 0.63 0.85 0.44 0.61 0.69
```

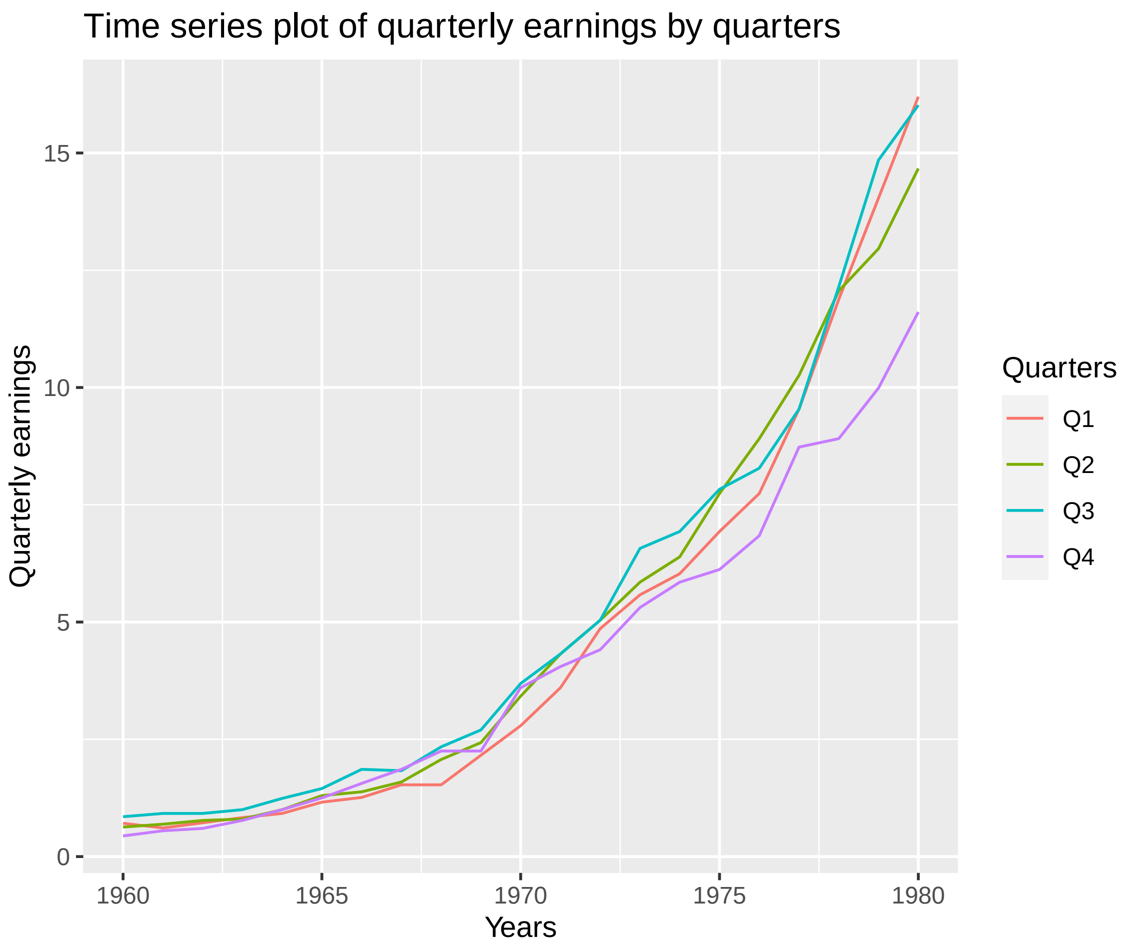
First we make a `tibble` out of this dataset, with the columns being the years, quarters and the earnings.

```
years = rep(1960:1980, each = 4)
quarters = rep(c("Q1", "Q2", "Q3", "Q4"), times = 21)
jj = tibble(years, quarters, JohnsonJohnson)
colnames(jj) = c("years", "qrtrs", "earnings")
```

Then we plot the following graphs:

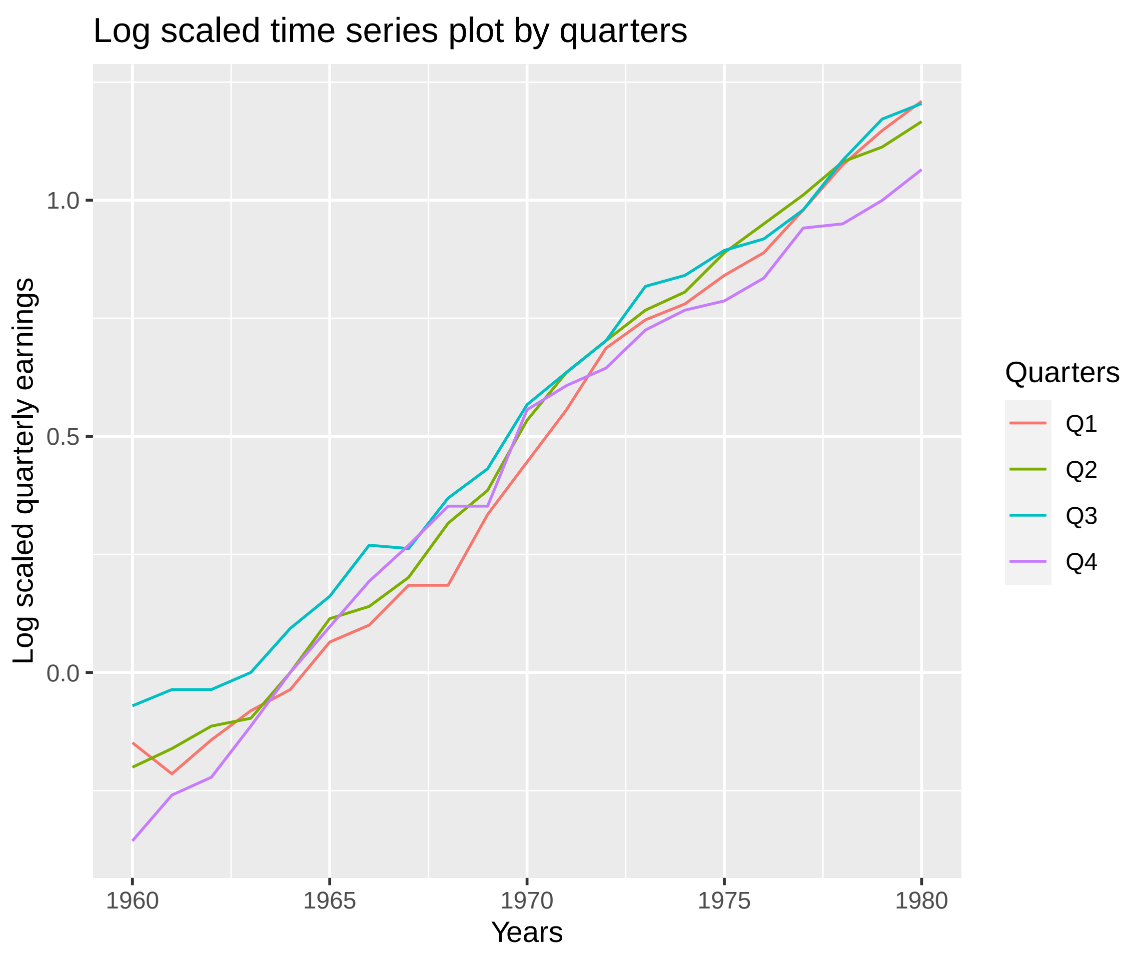
```
jj %>%
    ggplot(aes(x = years, y = earnings, color = qrtrs)) +
    geom_line() +
    labs(title = "Time series plot of quarterly earnings by quarters",
         x = "Years",
         y = "Quarterly earnings",
         color = "Quarters")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



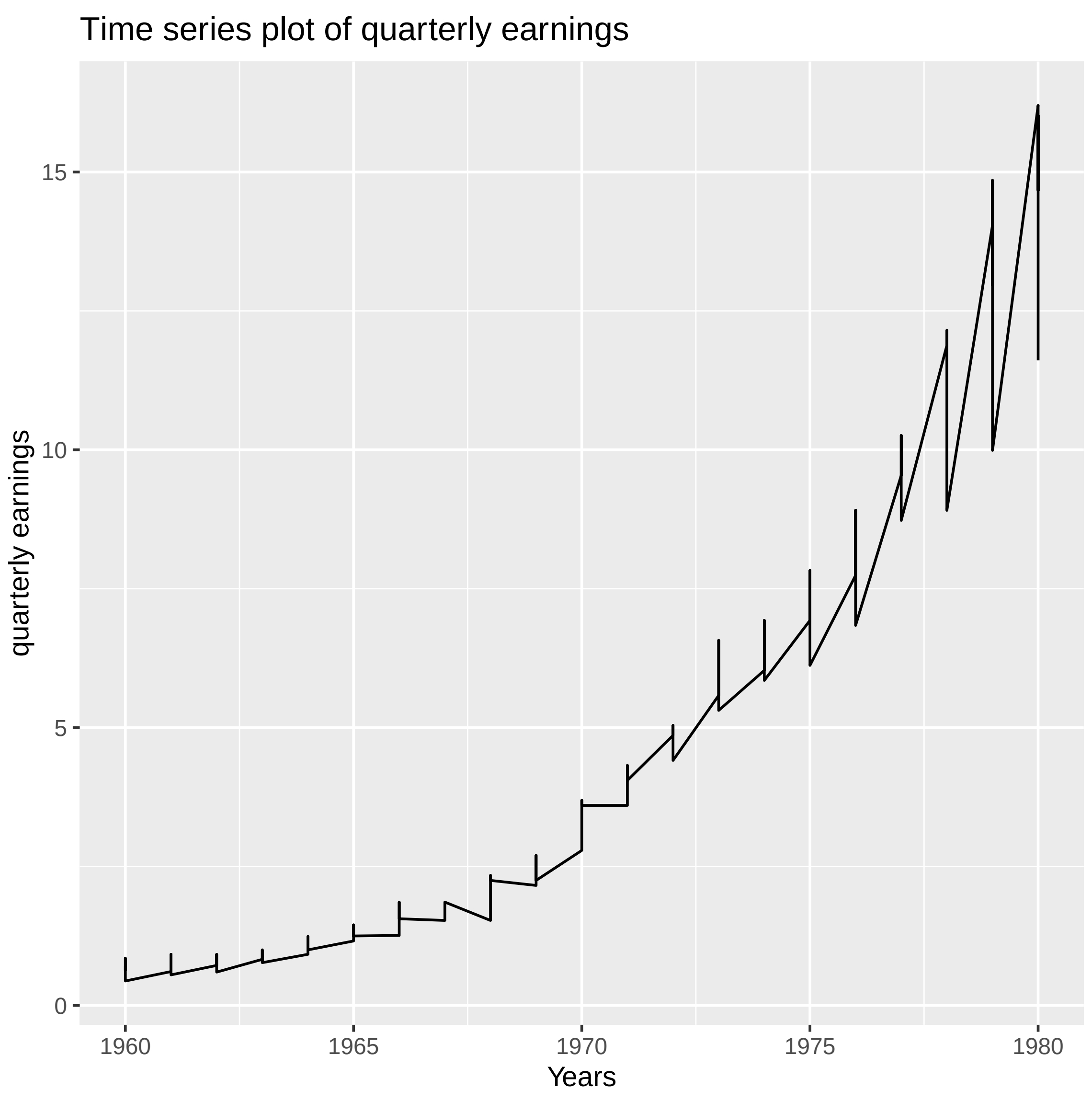
```
jj %>%
    ggplot(aes(x = years, y = log10(earnings), color = qrtrs)) +
    geom_line() +
    labs(title = "Log scaled time series plot by quarters",
         x = "Years",
         y = "Log scaled quarterly earnings",
         color = "Quarters")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



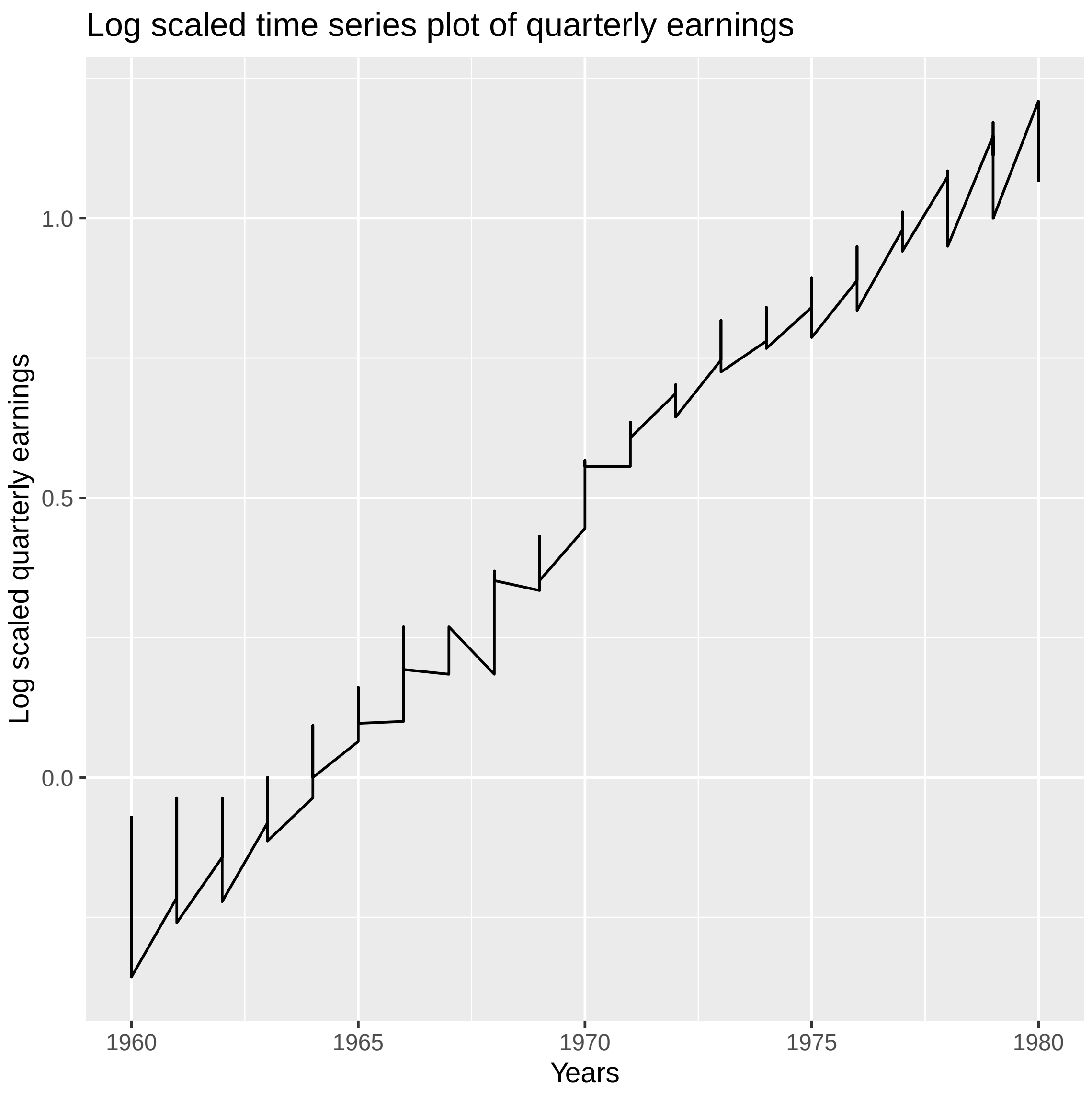
```
jj %>%
    ggplot(aes(x = years, y = earnings)) +
    geom_line() +
    labs(title = "Time series plot of quarterly earnings",
         x = "Years",
         y = "quarterly earnings")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



```r
jj %>%
    ggplot(aes(x = years, y = log10(earnings))) +
    geom_line() +
    labs(title = "Log scaled time series plot of quarterly earnings",
         x = "Years",
         y = "Log scaled quarterly earnings",
         color = "Quarters")
```

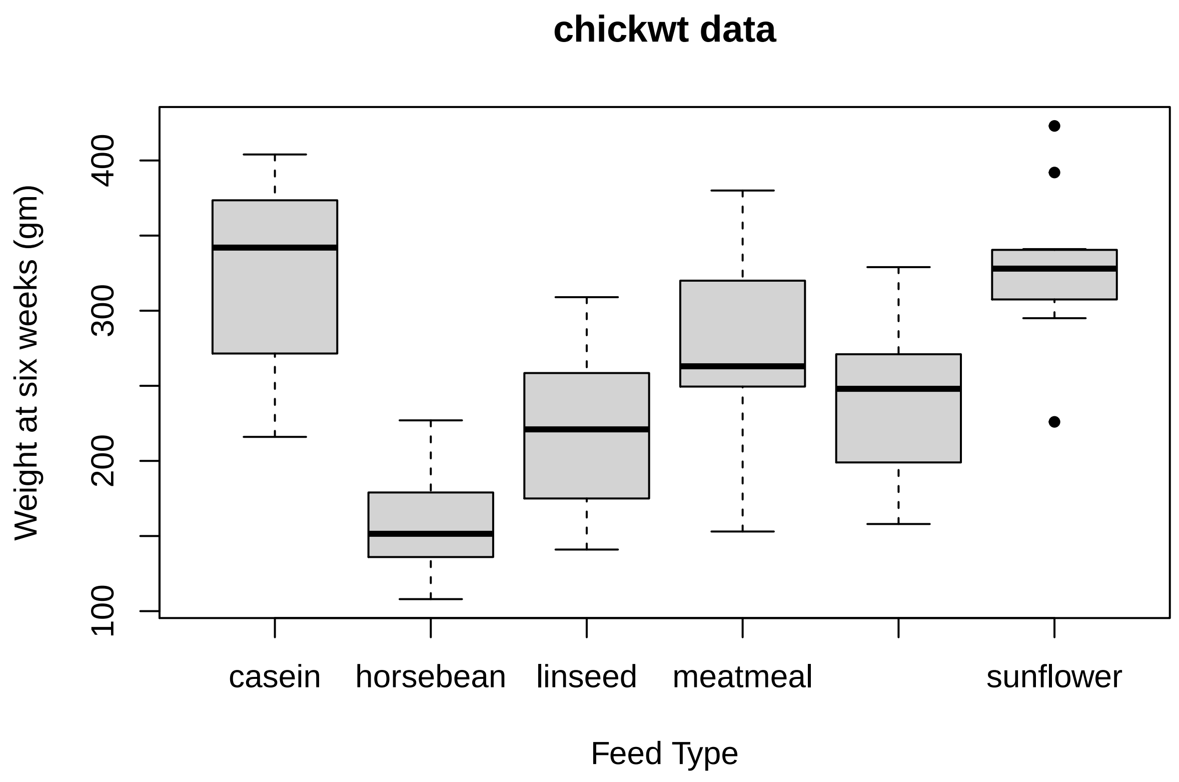
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



Problem 3: (2 points)

- An experiment was conducted to measure and compare the effectiveness of various feed supplements on the growth rate of chickens.
- Following R-code is a standard side-by-side boxplot showing effect of feed supplements on the growth rate of chickens.

```
boxplot(weight~feed,data=chickwts,pch=20
        ,main = "chickwt data"
        ,ylab = "Weight at six weeks (gm)"
        ,xlab = "Feed Type")
```



a) Reproduce the same plot using the `ggplot`; while fill each boxes with different colour.

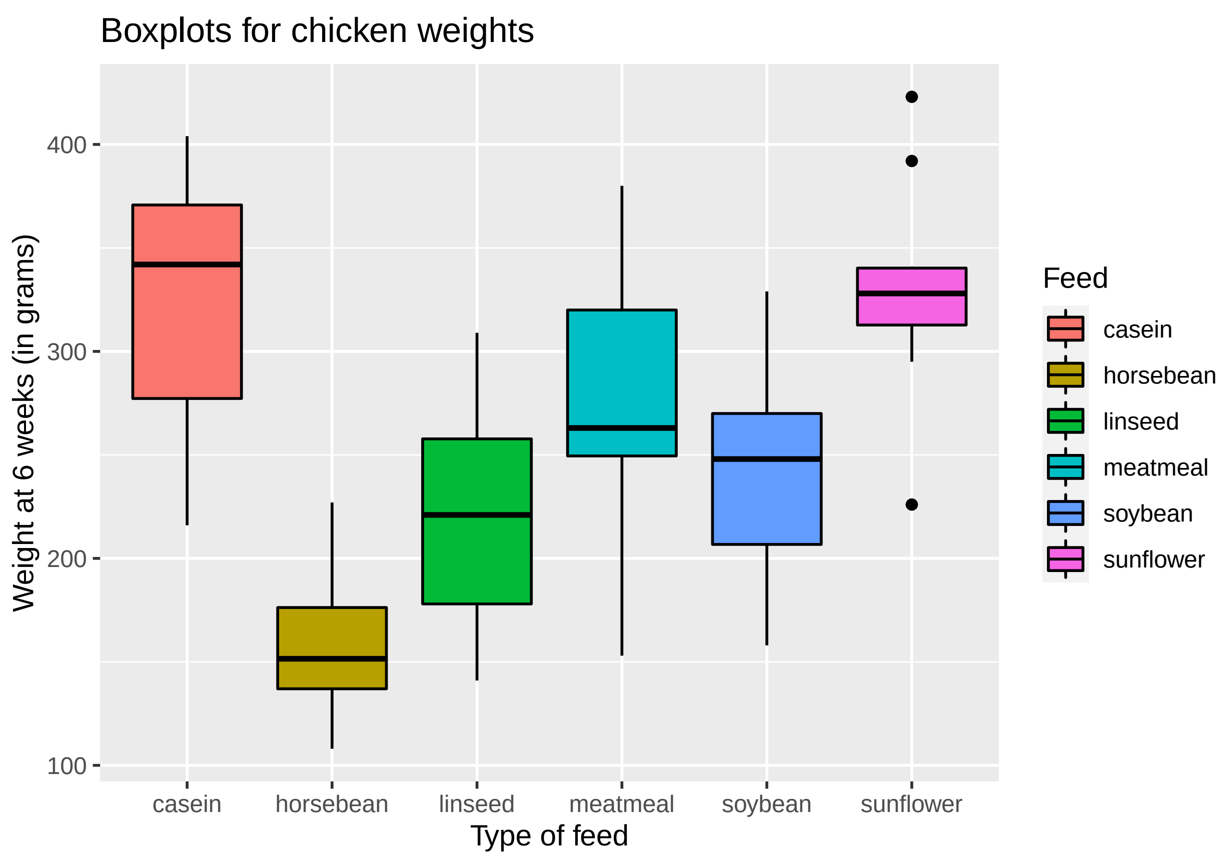
b) In addition draw probability density plot for weights of chicken's growth by each feed seperately using the `ggplot`. Draw this plot seperately.

First we convert the dateset to a tibble.

```r
data = tibble(chickwts)
```

a) The boxplot is as follows:

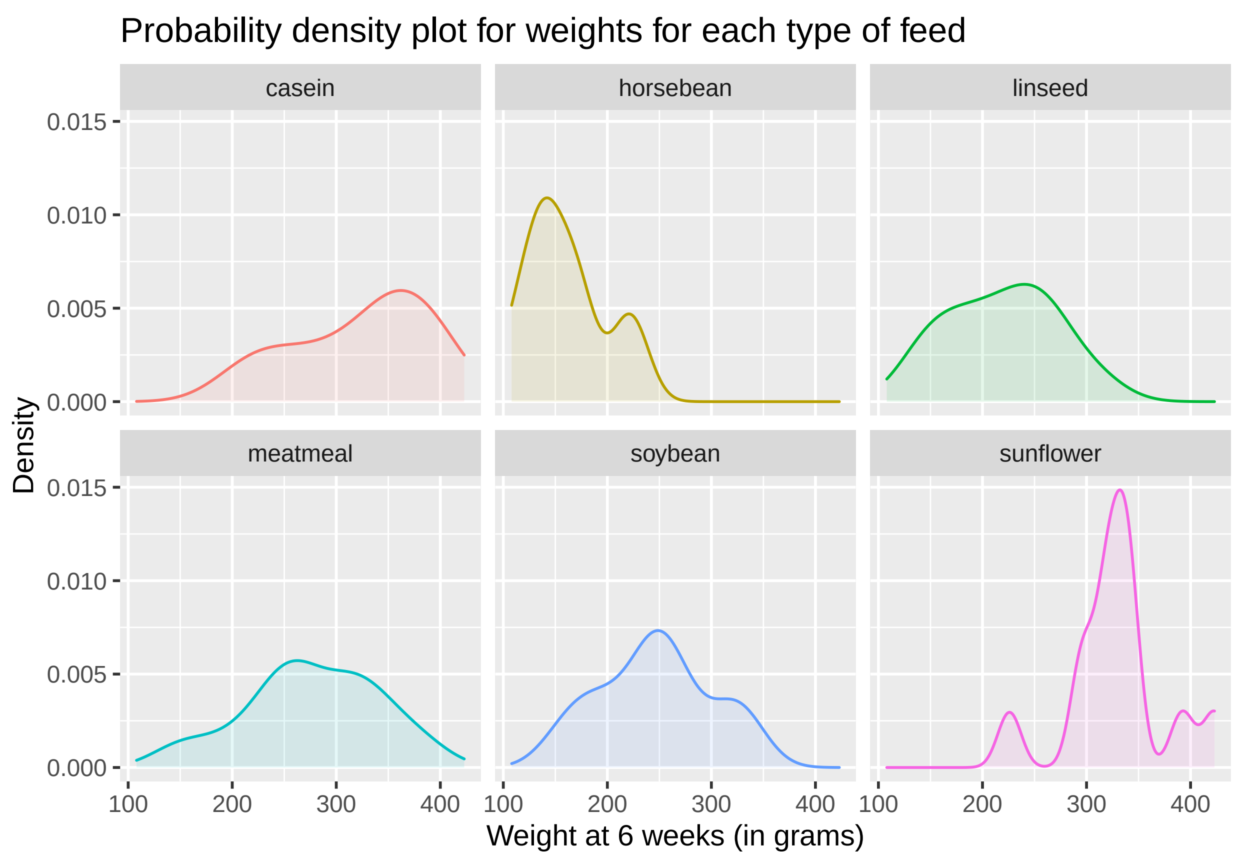
```r
data %>%
    ggplot(aes(x = feed, y = weight)) +
    geom_boxplot(aes(fill = feed), color = 'black') +
    labs(title = "Boxplots for chicken weights",
         x = "Type of feed",
         y = "Weight at 6 weeks (in grams)",
         fill = "Feed")
```



The probability density plot is as follows:

```
data %>%
    ggplot(aes(x = weight, color = feed, fill = feed)) +
    geom_density(bins = 25,
                 alpha = 0.1) +
    facet_wrap(vars(feed)) +
    theme(legend.position = "none") +
    labs(title = "Probability density plot for weights for each type of feed",
         x = "Weight at 6 weeks (in grams)",
         y = "Density")
```

```
## Warning: Ignoring unknown parameters: bins
```



Problem 4: (3 points)

Consider the `EuStockMarkets` data available in `R`. Contains the daily closing prices of major European stock indices: Germany DAX (Ibis), Switzerland SMI, France CAC, and UK FTSE. The data are sampled in business time, i.e., weekends and holidays are omitted.

```
head(EuStockMarkets)
```

```
##          DAX    SMI    CAC   FTSE
## [1,] 1628.75 1678.1 1772.8 2443.6
## [2,] 1613.63 1688.5 1750.5 2460.2
## [3,] 1606.51 1678.6 1718.0 2448.2
## [4,] 1621.04 1684.1 1708.1 2470.4
## [5,] 1618.16 1686.6 1723.1 2484.7
## [6,] 1610.61 1671.6 1714.3 2466.8
```

- Suppose P_t is the closing price of a stock indices on day t.
- The daily return r_t is defined as

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

a) Draw time-series plot of P_t for all four markets
b) Draw time-series plot of r_t for all four markets
c) Draw histogram of P_t for all four markets
d) Draw histogram of r_t for all four markets
e) Suppose you invested \$ 1000 in each market indices on day 1. Plot how your investment grows on the same plot for all four markets. Make your plot using `ggplot`.
f) Check which market outperform others during the same time?

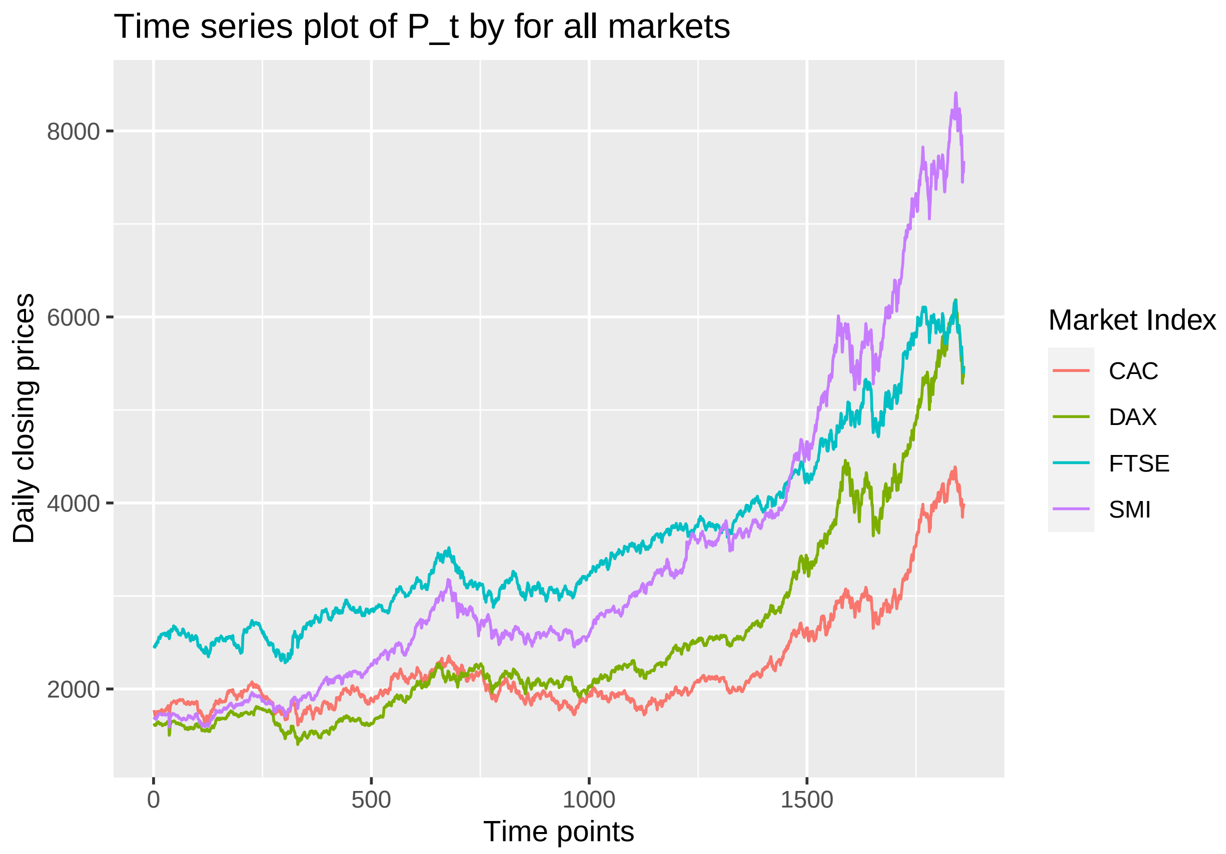
Make all your plots using `ggplot`.

Answer: We first load the dataset and convert it to a dataframe. Then we find the index of the data and the number of observations.

```
stockdata = data.frame(EuStockMarkets)
stockdata_idx = as.numeric(rownames(stockdata))
N = length(stockdata_idx)
```

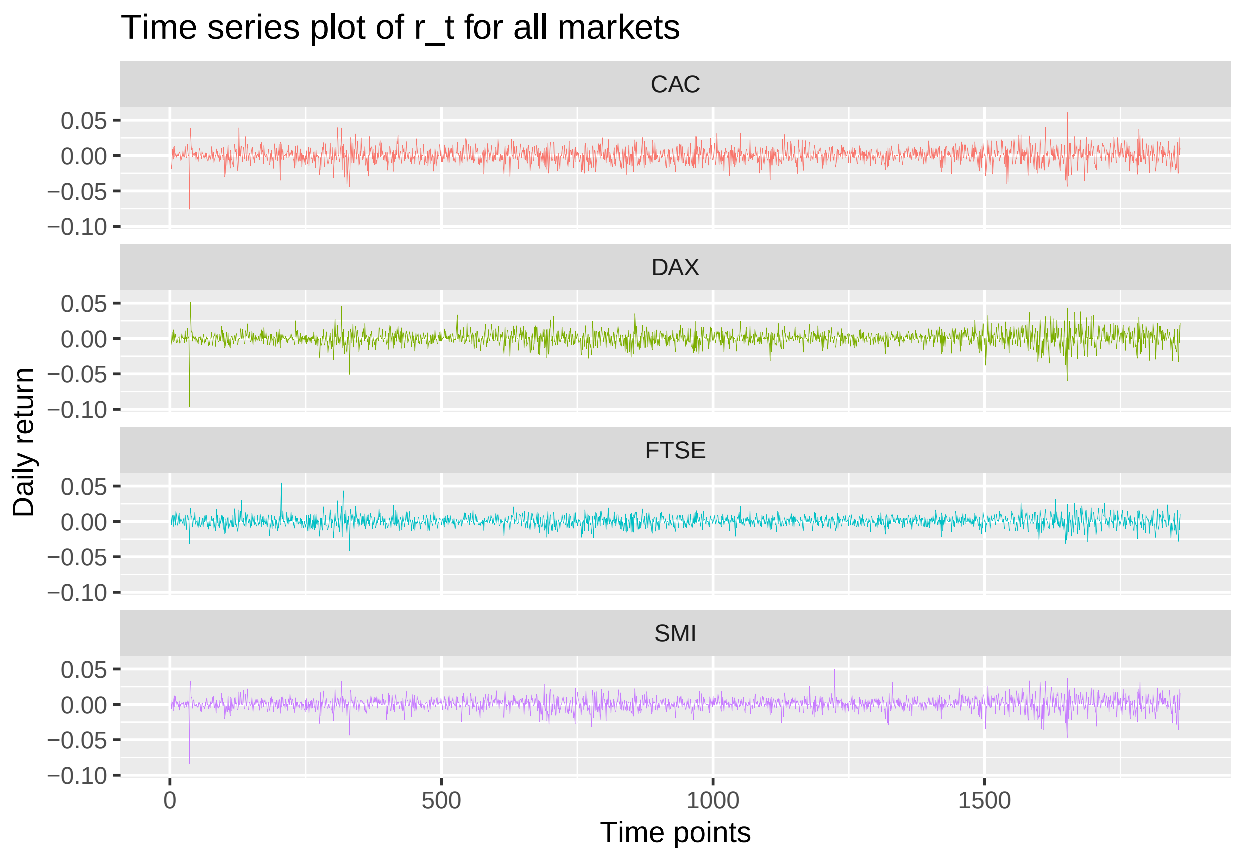
a)

```
stockdata %>%
    pivot_longer(1:4, names_to = c("index"), values_to = "closing_values") %>%
    mutate(idx = rep(stockdata_idx, each = 4)) %>%
    ggplot(aes(x = idx, y = closing_values, color = index)) +
    geom_line(size = 0.5) +
    labs(title = "Time series plot of P_t by for all markets",
         x = "Time points",
         y = "Daily closing prices",
         color = "Market Index")
```



b)

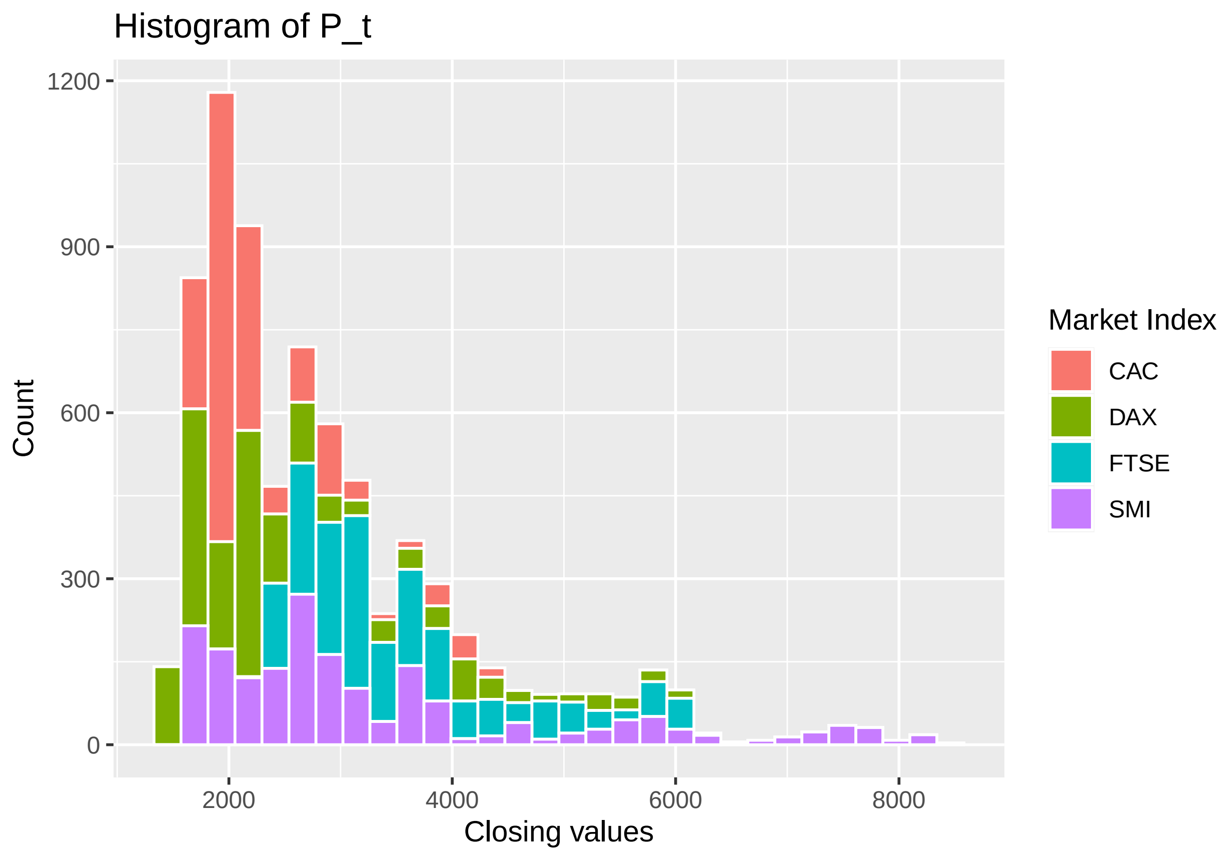
```
returndata = data.frame("idx" = stockdata_idx[2:N])
for(column in colnames(stockdata)){
    returndata[column] = log(stockdata[2:N, column]) - log(stockdata[1:(N-1),column])
}

returndata %>%
    pivot_longer(2:5, names_to = c("index"), values_to = "closing_values") %>%
    mutate(idx = rep(stockdata_idx[2:N], each = 4)) %>%
    ggplot(aes(x = idx, y = closing_values, color = index)) +
    geom_line(size = 0.1) +
    facet_wrap(vars(index), ncol = 1, nrow = 4) +
    labs(title = "Time series plot of r_t for all markets",
         x = "Time points",
         y = "Daily return",
         color = "Market Index") +
    theme(legend.position = "none")
```



c)

```
stockdata %>%
    pivot_longer(1:4, names_to = c("index"), values_to = "closing_values") %>%
    ggplot(aes(x = closing_values, fill = index)) +
    geom_histogram(color = "white") +
    labs(title = "Histogram of P_t",
         x = "Closing values",
         y = "Count",
         fill = "Market Index")
```

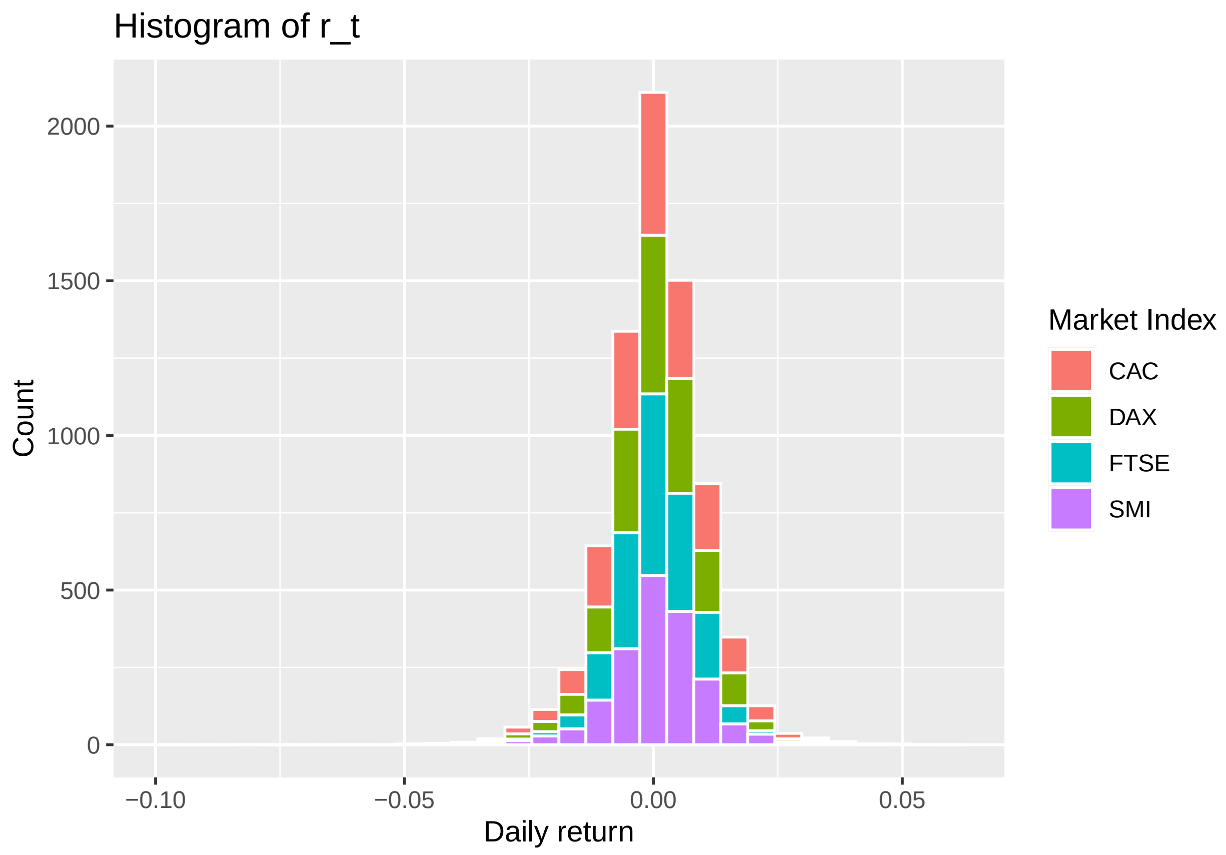
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



d)

```
returndata %>%
    pivot_longer(2:5, names_to = c("index"), values_to = "closing_values") %>%
    mutate(idx = rep(stockdata_idx[2:N], each = 4)) %>%
    ggplot(aes(x = closing_values, fill = index)) +
    geom_histogram(color = "white") +
    labs(title = "Histogram of r_t",
         y = "Count",
         x = "Daily return",
         fill = "Market Index")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


Histogram of r_t
Count
2000
1500
1000
500
0
−0.10
−0.05
0.00
0.05
Daily return
Market Index
CAC
DAX
FTSE
SMI

e)