Visualisation: Assignment 1

Name: Adarsha Mondal | Roll Number: MDS202205

Dead Line : 03 Oct 2022

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 $\mathbf{R}$ 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:

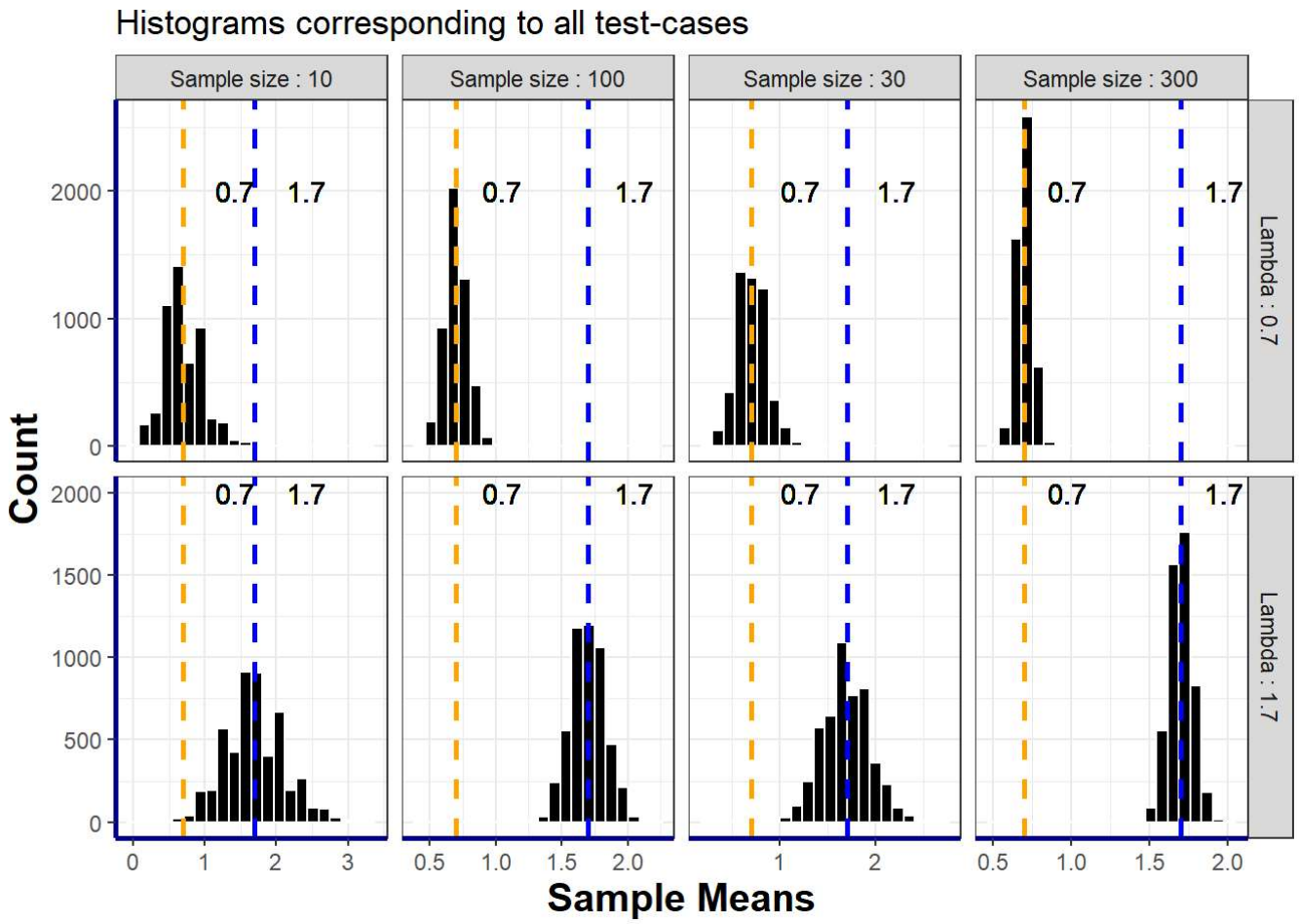
- Histogram of the sampling distribution using `ggplot`
- QQ-plot using `ggplot`

Test cases:

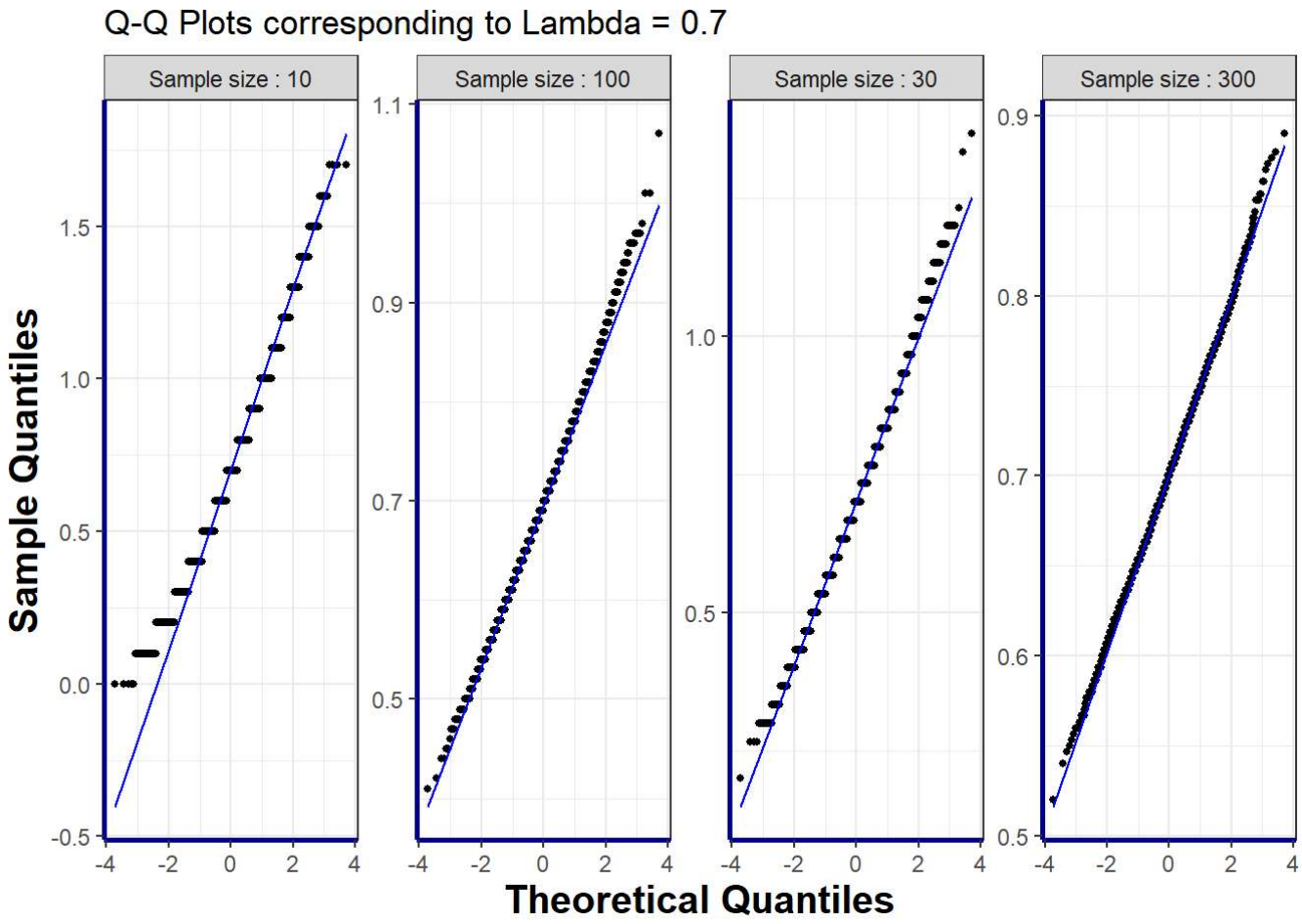
- case 1 a: $\lambda = 0.7$, n=10, N=5000
- case 1 b: $\lambda = 0.7$, n=30, N=5000
- case 1 c: $\lambda = 0.7$, n=100, N=5000
- case 1 c: $\lambda = 0.7$, n=300, N=5000
- case 2 a: $\lambda = 1.7$, n=10, N=5000
- case 2 b: $\lambda = 1.7$, n=30, N=5000
- case 2 c: $\lambda = 1.7$, n=100, N=5000

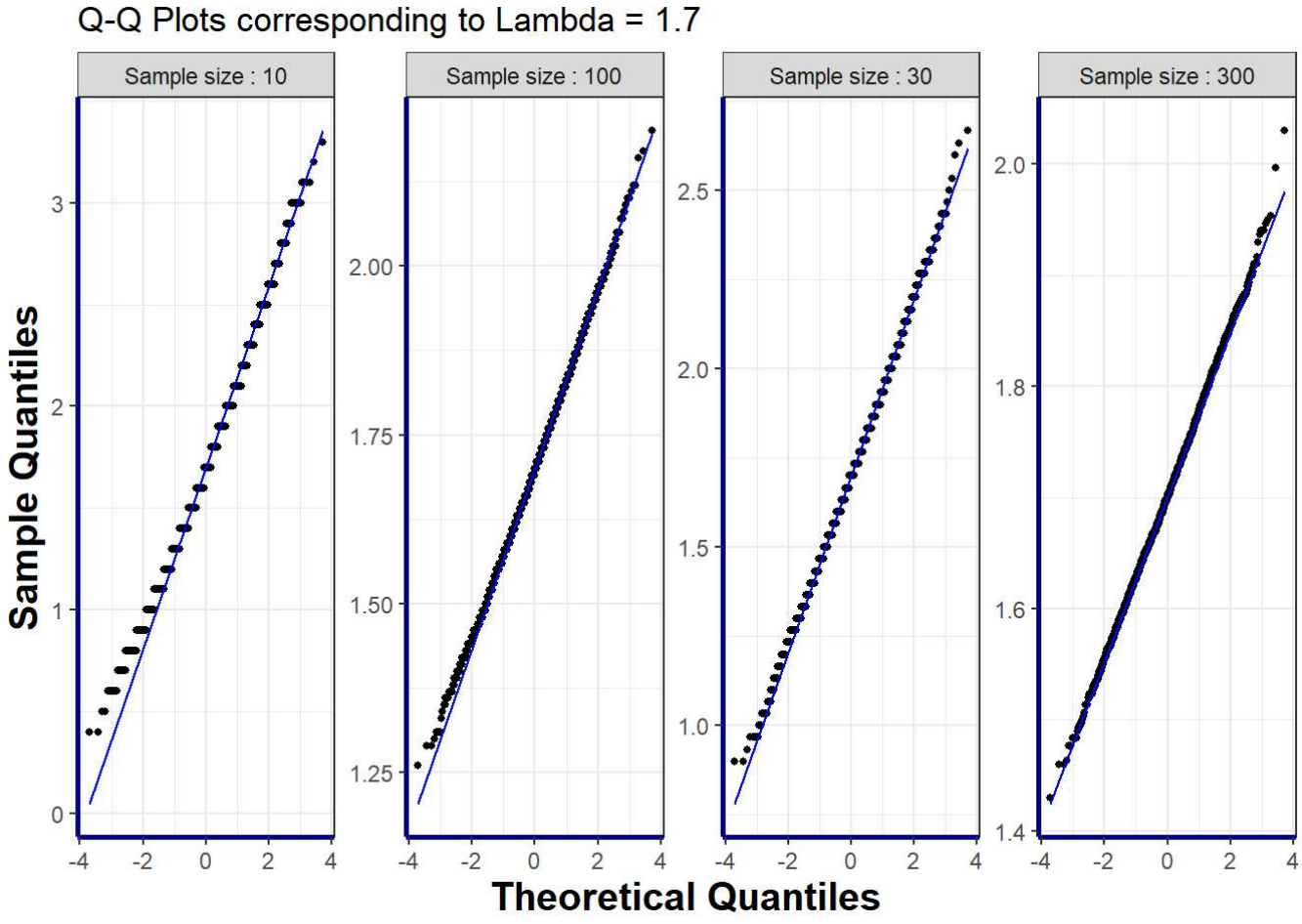
- case 2 c: $\lambda = 1.7$, n=300, N=5000

```
cumulative_df = data.frame(means = numeric(0),
                samp_size = character(0), lambda = character(0))

problem_1 = function(lambda, sample_size, sim_size){
  samplings_mean_vec = replicate(sim_size,mean(rpois(sample_size,lambda)))
  samplings_mean_df = data.frame(samplings_mean_vec,
        rep(paste("Sample size :", sample_size,sep = " "),sim_size),
        rep(paste("Lambda :",lambda,sep = " "),sim_size))
  return(samplings_mean_df)
}

lambdas = c(0.7, 1.7)

test_cases = c(010,030,100,300)

simulation_size = 5000

for (lambda in lambdas){
  for (test in test_cases){
    cumulative_df = rbind(cumulative_df, problem_1(lambda,test,simulation_size))
  }
}
colnames(cumulative_df) = c('means', 'samp_size', 'lambda')

# Histigrams
print(ggplot(cumulative_df)+aes(x=means)+
  geom_histogram(color="white", fill="black",bins=22)+
  geom_vline(xintercept = 0.7, linetype="dashed", color="orange",size=1)+
  geom_vline(xintercept = 1.7, linetype="dashed", color="blue",size=1)+
  #facet_wrap(~samp_size, ncol=4)+
  facet_grid(lambda ~ samp_size, scales = "free")+
  theme(axis.line = element_line(colour = "darkblue", size = 1,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=15),
        axis.title.y = element_text(face="bold",size=15))+
  geom_text(aes(0.6,2000,label = 0.7, hjust=-1))+
  geom_text(aes(1.6,2000,label = 1.7, hjust=-1))+
  labs(title = "Histograms corresponding to all test-cases",x = "Sample Means",
       y="Count"))
```




```
#head(cumulative_df[cumulative_df$lambda=='Lambda : 0.7',])
# Q-Q Plots
for (lambda in lambdas){

  df = cumulative_df[cumulative_df$lambda==paste('Lambda :',lambda,sep = ' '),]

  print(ggplot(data=df, aes(sample = means))+geom_qq(size=1)+
    geom_qq_line(color = "blue")+
    facet_wrap(vars(samp_size), scales = "free_y",nrow = 1)+
    theme(axis.line = element_line(colour = "darkblue", size = 1,
                                   linetype = "solid"),
          axis.title.x = element_text(face="bold",size=15),
          axis.title.y = element_text(face="bold",size=15))+
    labs(title = paste('Q-Q Plots corresponding to Lambda =',lambda,sep = ' '),
         x = "Theoretical Quantiles", y = "Sample Quantiles"))
}
```





Problem 2: (2 points)

Consider the `JohnsonJohnson` dataset. The datset 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
```

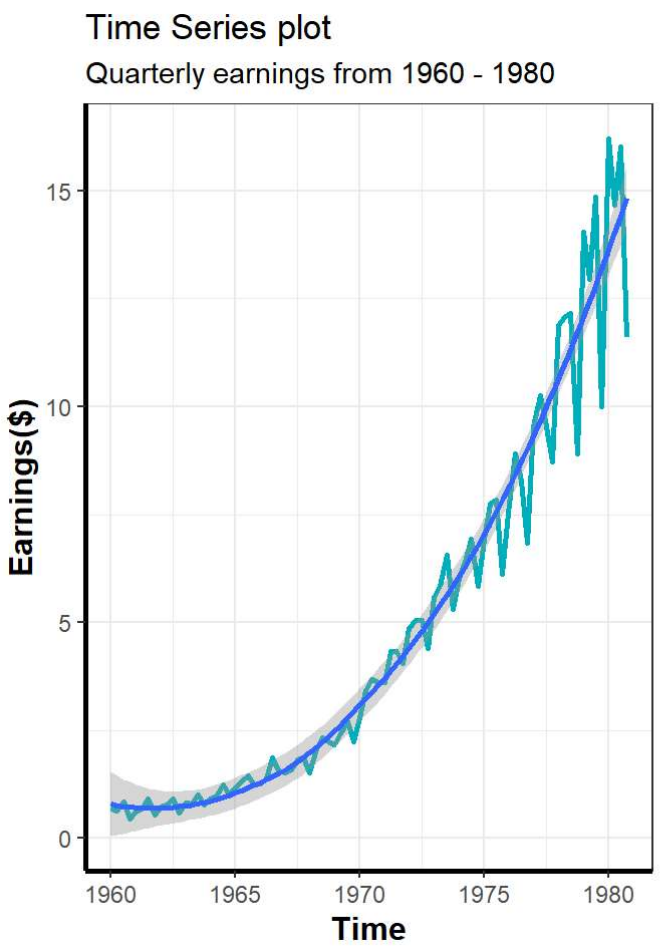
```
problem_2 = function(){
  jj_earnings_df = data.frame(time = time(JohnsonJohnson),
                              earnings = JohnsonJohnson)

  line = ggplot(data = jj_earnings_df, aes(x = time, y = earnings)) +
    geom_line(color = "#00AFBB", size = 1)+
    geom_smooth()+
    theme(axis.line = element_line(colour = "black", size = 1,
                                   linetype = "solid"),
          axis.title.x = element_text(face="bold",size=12),
          axis.title.y = element_text(face="bold",size=12))+
    labs(title = "Time Series plot",
         subtitle = "Quarterly earnings from 1960 - 1980",
         x = "Time",
         y = "Earnings($)")

  log = ggplot(data = jj_earnings_df, aes(x = time, y = earnings)) +
    geom_line(color = "red", size = 1) + scale_y_continuous(trans = "log")+
    geom_smooth(method = "lm")+
    theme(axis.line = element_line(colour = "black", size = 1,
                                   linetype = "solid"),
          axis.title.x = element_text(face="bold",size=12),
          axis.title.y = element_text(face="bold",size=12))+
    labs(title = "Log-scaled Time Series plot",
         subtitle = "Quarterly earnings from 1960 - 1980",
         x = "Time",
         y = "log(Earnings)")

  figure = ggarrange(line, log, ncol = 2)
  return(figure)
}
print(problem_2())
```

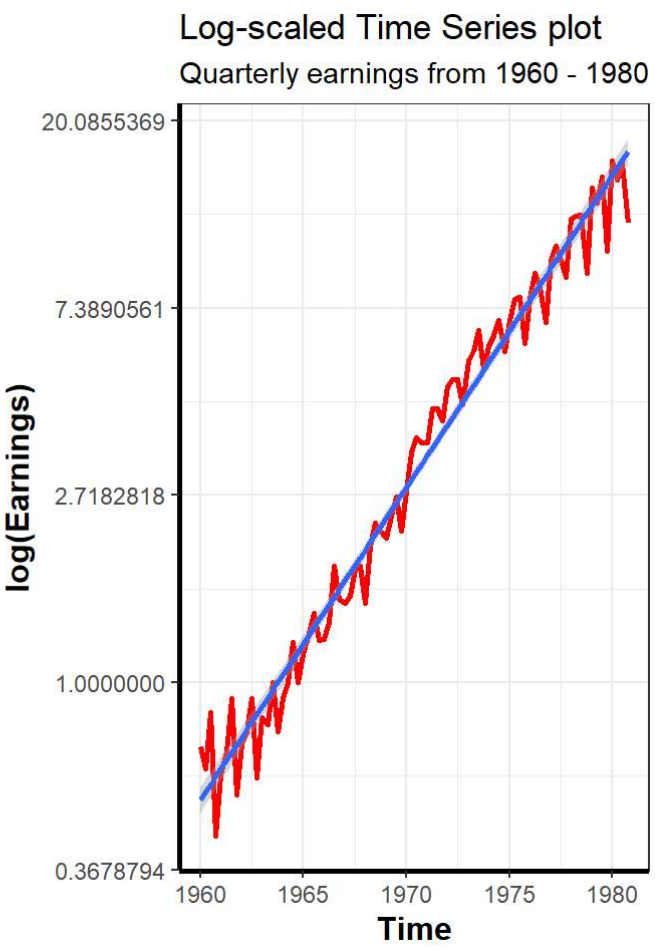
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuou
s.
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuou
s.
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

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

```
## `geom_smooth()` using formula 'y ~ x'
```

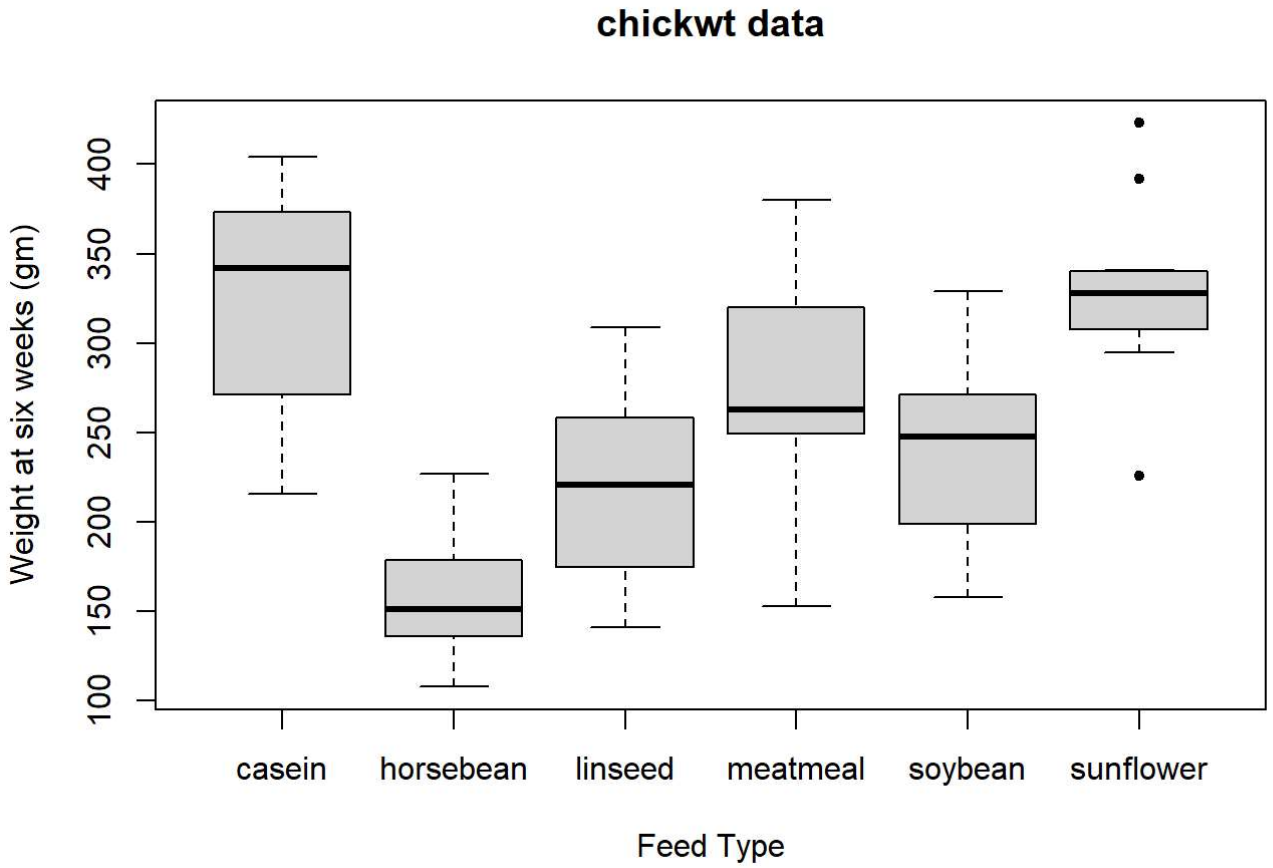




Problem 3: (2 points)

- 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.

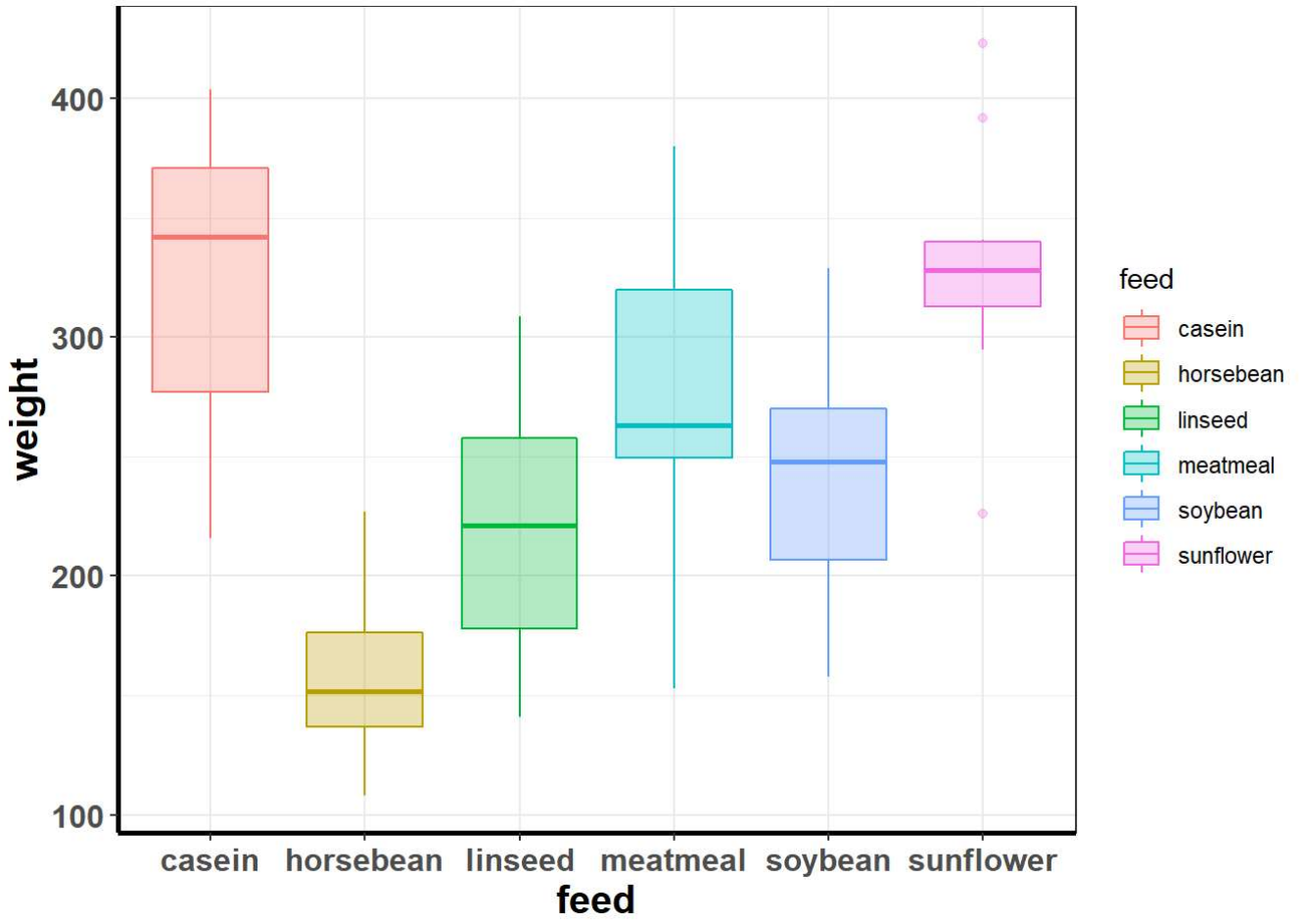
```
boxplot(weight~feed,data=chickwts,pch=20
        ,main = "chickwt data"
        ,ylab = "Weight at six weeks (gm)"
        ,xlab = "Feed Type")
```

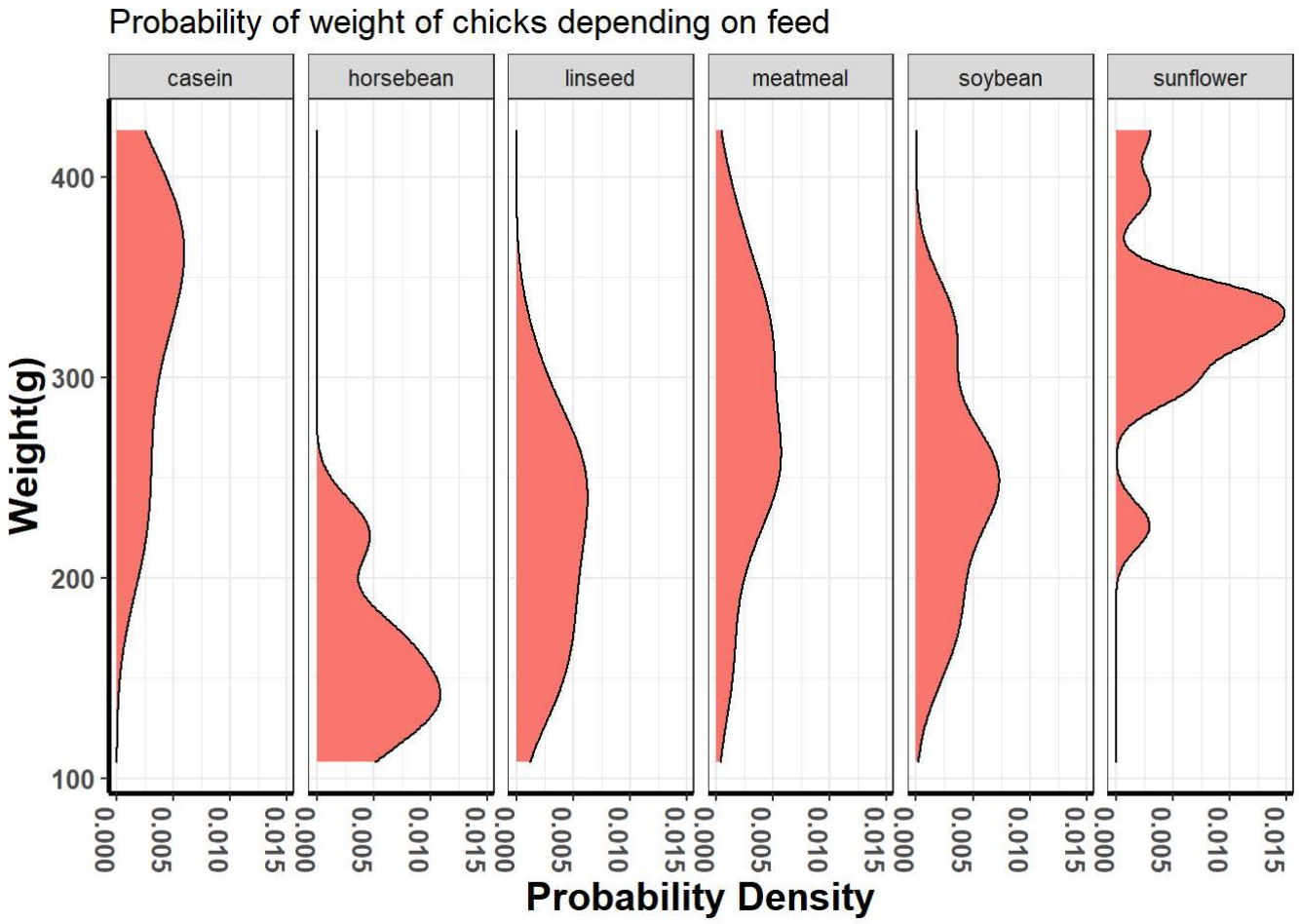


a. Reproduce the same plot using the `ggplot` ; while fill each boxes with different colour.

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

```
problem_3 = function(){
  chickwts_df = data.frame(chickwts)

  #
  print(ggplot(chickwts_df, aes(x = feed, y = weight, fill=feed))+
          aes(color = feed)+
          geom_boxplot(alpha=0.3)+
          theme(axis.line = element_line(colour = "black", size = 1,
                                         linetype = "solid"),
                axis.title.x = element_text(face="bold",size=15),
                axis.title.y = element_text(face="bold",size=15),
                axis.text.x = element_text(face = "bold",size = 12),
                axis.text.y = element_text(face = "bold",size = 12)))

  # Probability densities
  print(ggplot(chickwts_df)+
          geom_density(aes(y=weight, fill="f8766d"))+
          facet_grid(~feed)+
          theme(axis.line = element_line(colour = "black", size = 1,
                                         linetype = "solid"),
                axis.title.x = element_text(face="bold",size=15),
                axis.title.y = element_text(face="bold",size=15),
                axis.text.x = element_text(angle = -90,face = "bold",size = 10),
                axis.text.y = element_text(face = "bold",size = 10),
                legend.position="none")+
          labs(title = "Probability of weight of chicks depending on feed",
               x = "Probability Density",
               y = "Weight(g)"))
}
problem_3()
```





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?

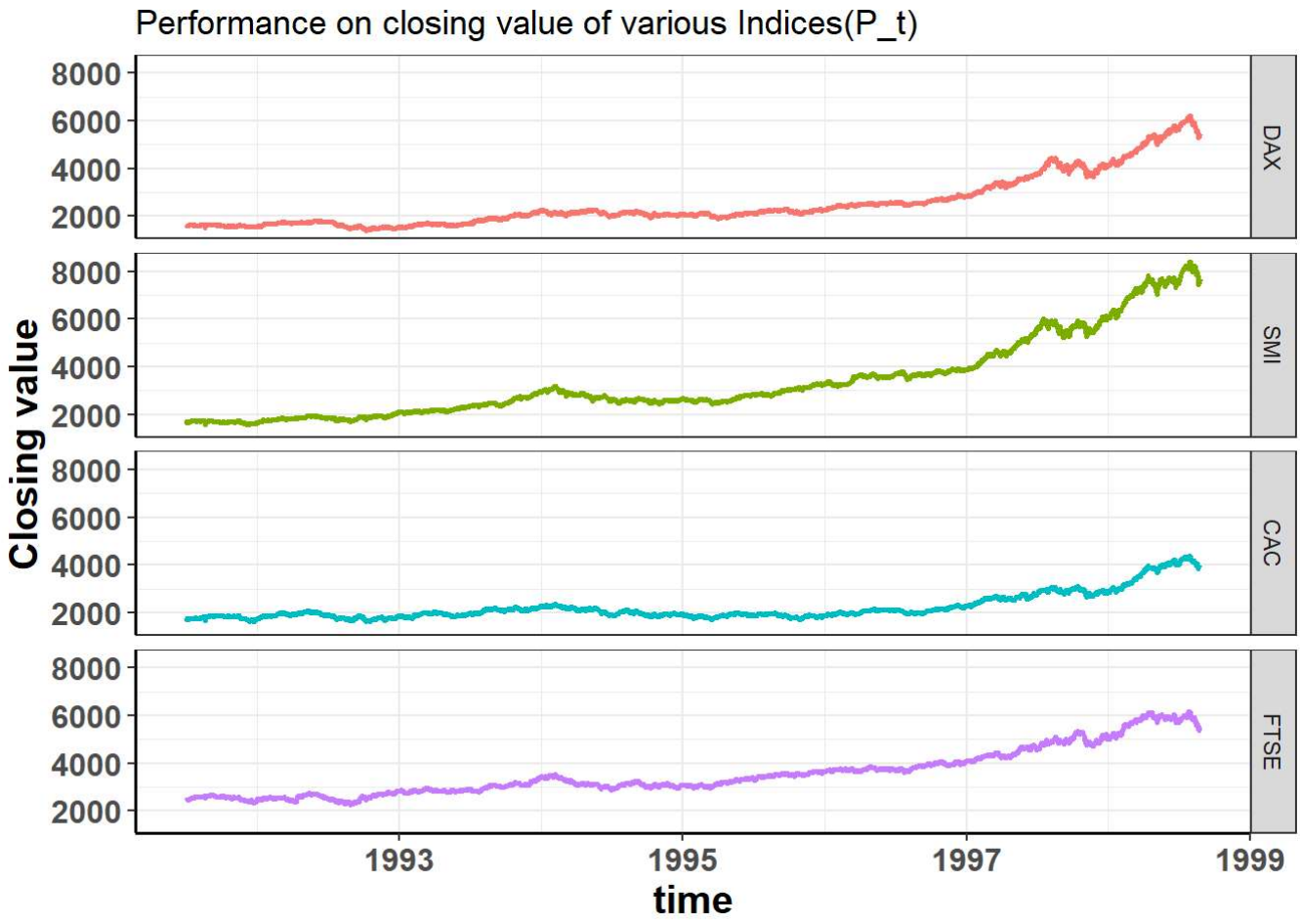
Make all your plots using `ggplot`.

```
time = data.frame(time = time(EuStockMarkets))
data = data.frame(EuStockMarkets)
n = nrow(data)

for(i in 1:4){
  temp_dr = c(0)
  temp_inv = c(1000)
  for(j in 2:n){
    k = log(data[j,i]/data[j-1,i])
    temp_dr = c(temp_dr,k)
    temp_inv = c(temp_inv,temp_inv[j-1]*exp(k))
  }

  if(i==1){
    dr_df = data.frame(IDX_dr = temp_dr)
    investment_df = data.frame(IDX_inv = temp_inv)
  }
  else{
    dr_df[paste(names(data)[i], 'dr', sep = '_')] = data.frame(temp_dr)
    investment_df[paste(names(data)[i], 'inv', sep = '_')] = data.frame(temp_inv)
  }
}

inv_melt=melt(cbind(time,investment_df),id=c('time'),value.name = "value")

pt_melt = melt(cbind(time, data), id = c("time"), value.name = "value")

dr_melt = melt(cbind(time, dr_df), id = c("time"), value.name = "value")
```

a.

```
print(ggplot(pt_melt)+aes(x=time, y=value)+
  geom_line(aes(color=variable),size=1)+
  facet_grid(vars(variable))+
  labs(title = "Performance on closing value of various Indices(P_t)",
       y = "Closing value")+
  theme(axis.line = element_line(colour = "black", size = 0.6,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=15),
        axis.title.y = element_text(face="bold",size=15),
        axis.text.x = element_text(face = "bold",size = 12),
        axis.text.y = element_text(face = "bold",size = 12),
        legend.position = "none"))
```

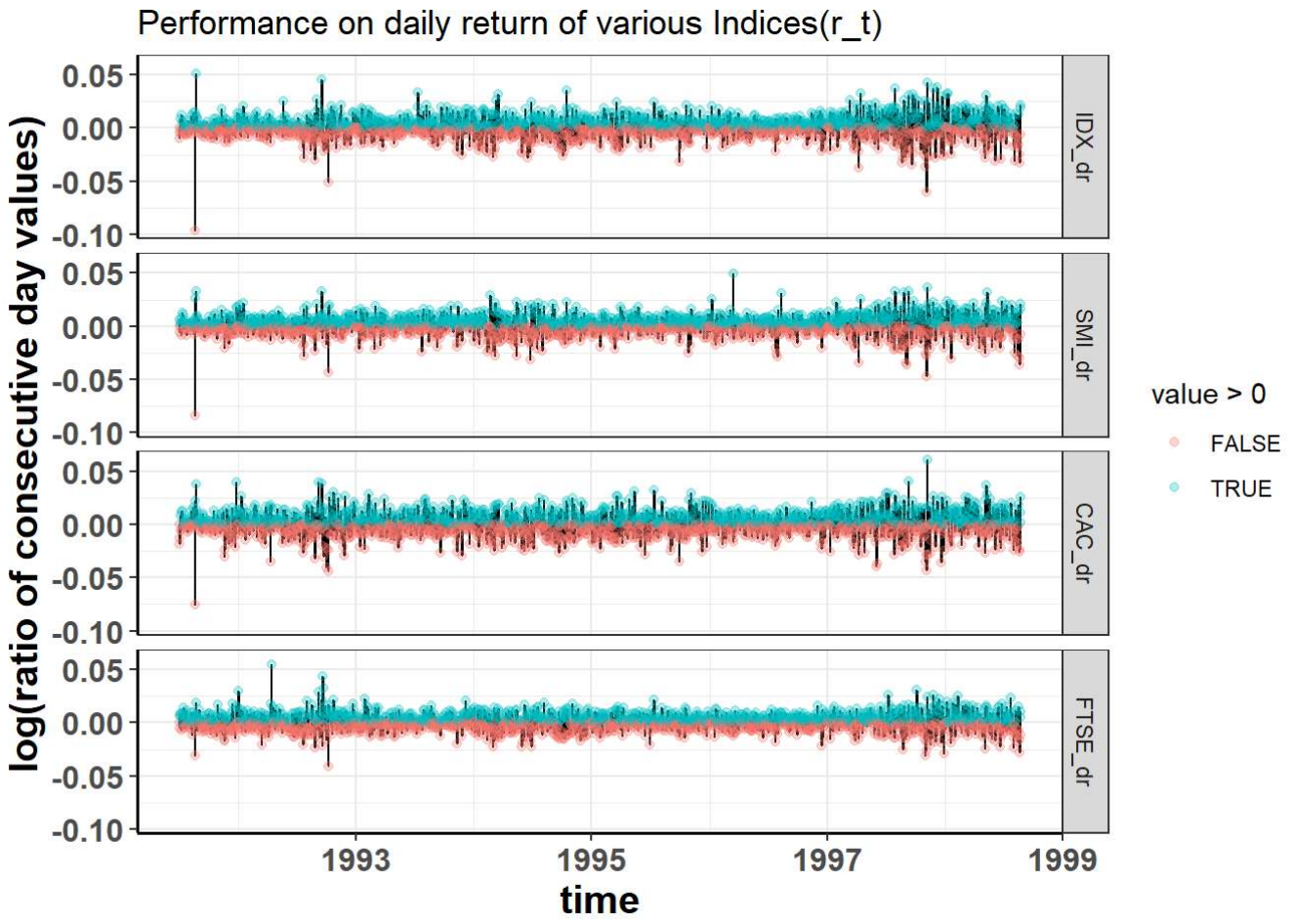
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuou
s.
```



b.

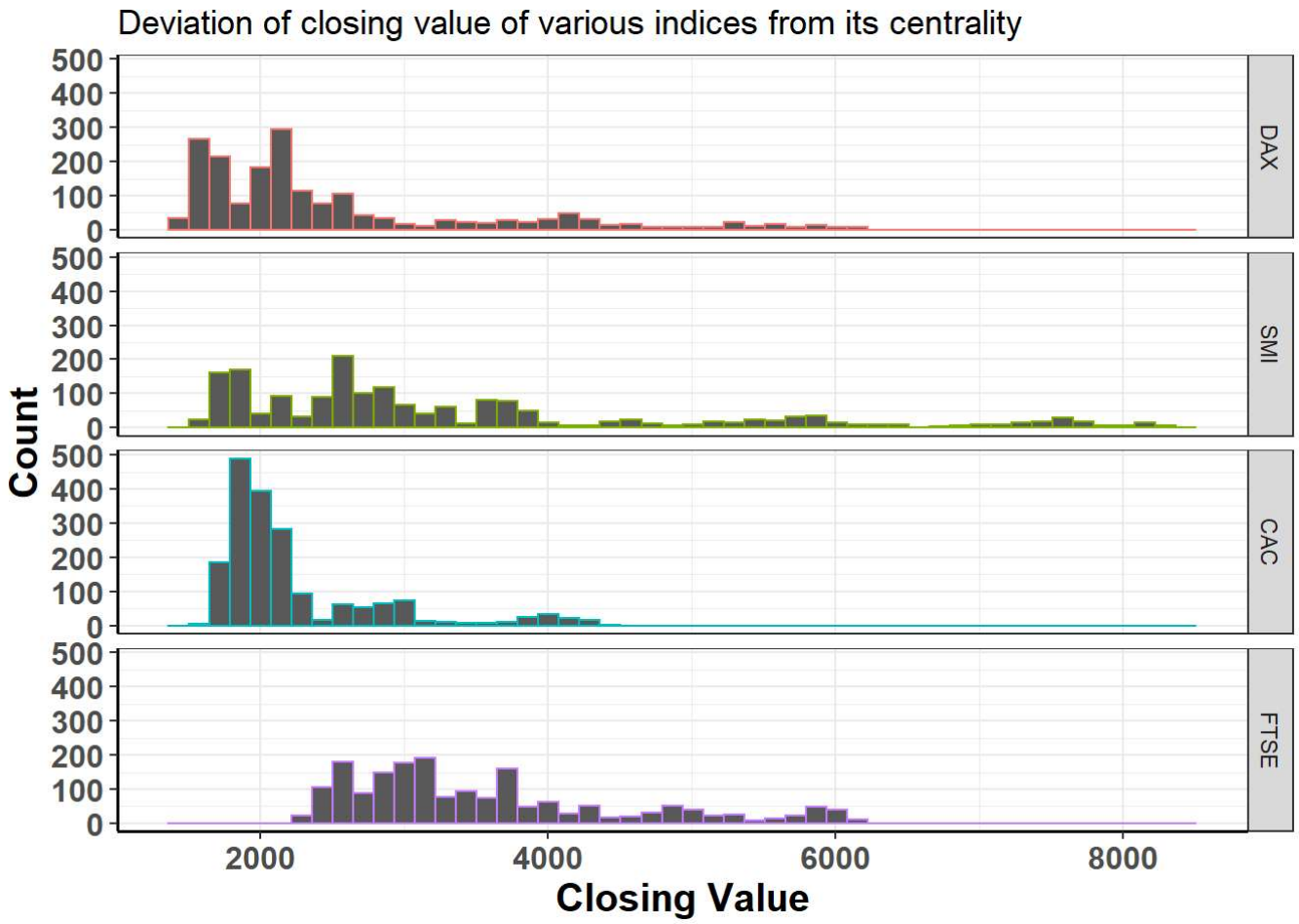
```
print(ggplot(dr_melt)+aes(x=time, y=value)+
  geom_line()+
  geom_point(alpha=0.3,aes(colour=value>0))+
  facet_grid(vars(variable))+
  labs(title = "Performance on daily return of various Indices(r_t)",
       y = "log(ratio of consecutive day values)")+
  theme(axis.line = element_line(colour = "black", size = 0.6,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=15),
        axis.title.y = element_text(face="bold",size=15),
        axis.text.x = element_text(face = "bold",size = 12),
        axis.text.y = element_text(face = "bold",size = 12)))
```

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



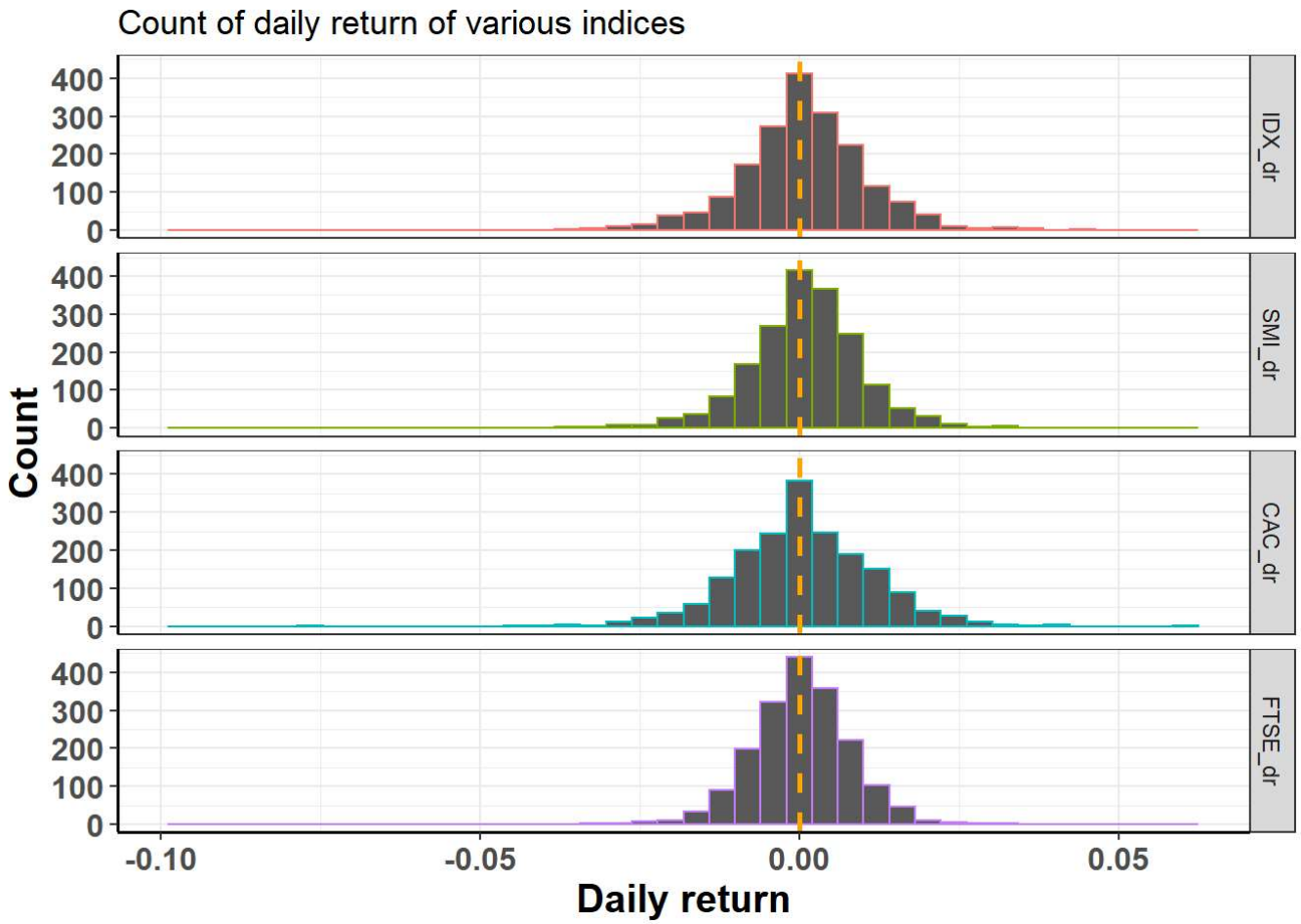
c.

```
print(ggplot(pt_melt)+aes(x=value)+
  geom_histogram(aes(color=variable),bins=50)+
  facet_grid(vars(variable))+
  labs(title =
         "Deviation of closing value of various indices from its centrality",
       y = "Count",x="Closing Value")+
  theme(axis.line = element_line(colour = "black", size = 0.6,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=15),
        axis.title.y = element_text(face="bold",size=15),
        axis.text.x = element_text(face = "bold",size = 12),
        axis.text.y = element_text(face = "bold",size = 12),
        legend.position = "none"))
```



d.

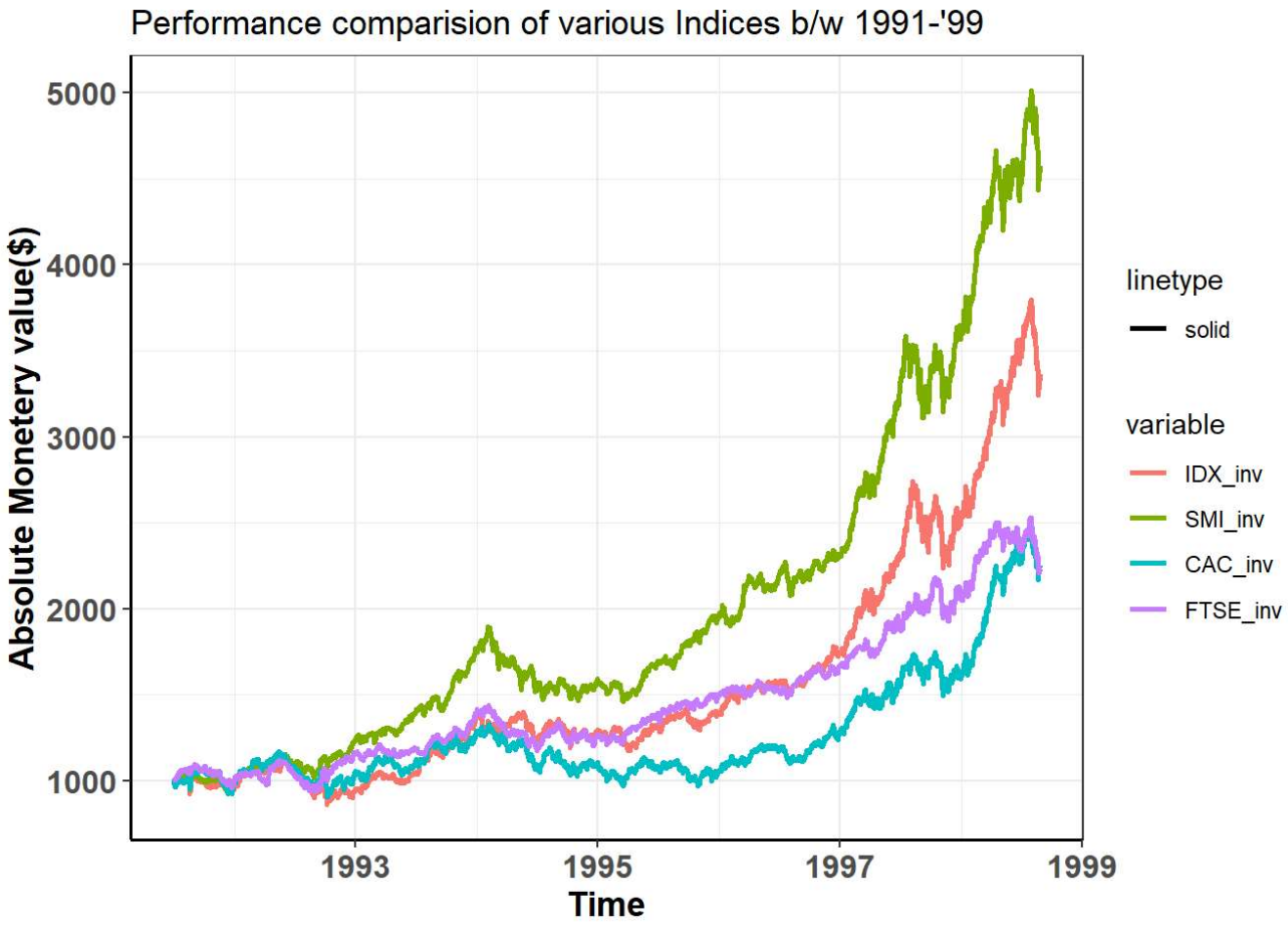
```
print(ggplot(dr_melt)+aes(x=value)+
  geom_histogram(aes(color=variable),bins=40)+
  facet_grid(vars(variable))+
  geom_vline(xintercept = 0, linetype="dashed", color="orange",size=1)+
  labs(title =
         "Count of daily return of various indices",
       y = "Count",x="Daily return")+
  theme(axis.line = element_line(colour = "black", size = 0.6,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=15),
        axis.title.y = element_text(face="bold",size=15),
        axis.text.x = element_text(face = "bold",size = 12),
        axis.text.y = element_text(face = "bold",size = 12),
        legend.position = "none"))
```



e.

```
print(ggplot(inv_melt, aes(x = time, y = value)) +
  geom_line(aes(color = variable, linetype = "solid"),size=1)+
  labs(title =
         "Performance comparision of various Indices b/w 1991-'99",
       y = "Absolute Monetery value($)",x="Time")+
  theme(axis.line = element_line(colour = "black", size = 0.6,
                                 linetype = "solid"),
        axis.title.x = element_text(face="bold",size=13),
        axis.title.y = element_text(face="bold",size=13),
        axis.text.x = element_text(face = "bold",size = 12),
        axis.text.y = element_text(face = "bold",size = 12)))
```

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



f. By analyzing the last graph(e), we get to know that, the SMI(Swiss Market Index) outperforms all the the other indices (DAX, CAC and FTSE) for almost the entire time-span given(1991-1999).