Tasks

(i) Obtain the time series using the correction code syntax making use of the code below

library(quantmod)

# Obj 1: Load stock prices by symbol

getSymbols(symbol) [5 marks]

```
getSymbols("^GSPC", from = '2009-01-01',to = "2020-12-31",warnings =
FALSE,auto.assign = TRUE)
```

From the ‘quantmod’ library, ‘getSymbols’ is used to load the “^GSPC” dataset within the data range ‘2009-01-01’ to ‘2020-12-31’. This dataset is the obtained time series.

(ii) Transform your series to log-returns. [5 marks]

The log return is a measure of the percentage change in the value of an investment over a given time period. It is calculated using the code:

```
returns = diff(log(GSPC$GSPC.Close))
```

The values for log returns are calculated using the formula:

$$R = (ln(Vf/Vi)/t)\times 100\%$$

where V_i is the initial investment

V_f is the final value

t is the number of time periods

Take the natural logarithm of Vf divided by Vi, and divide the result by t.

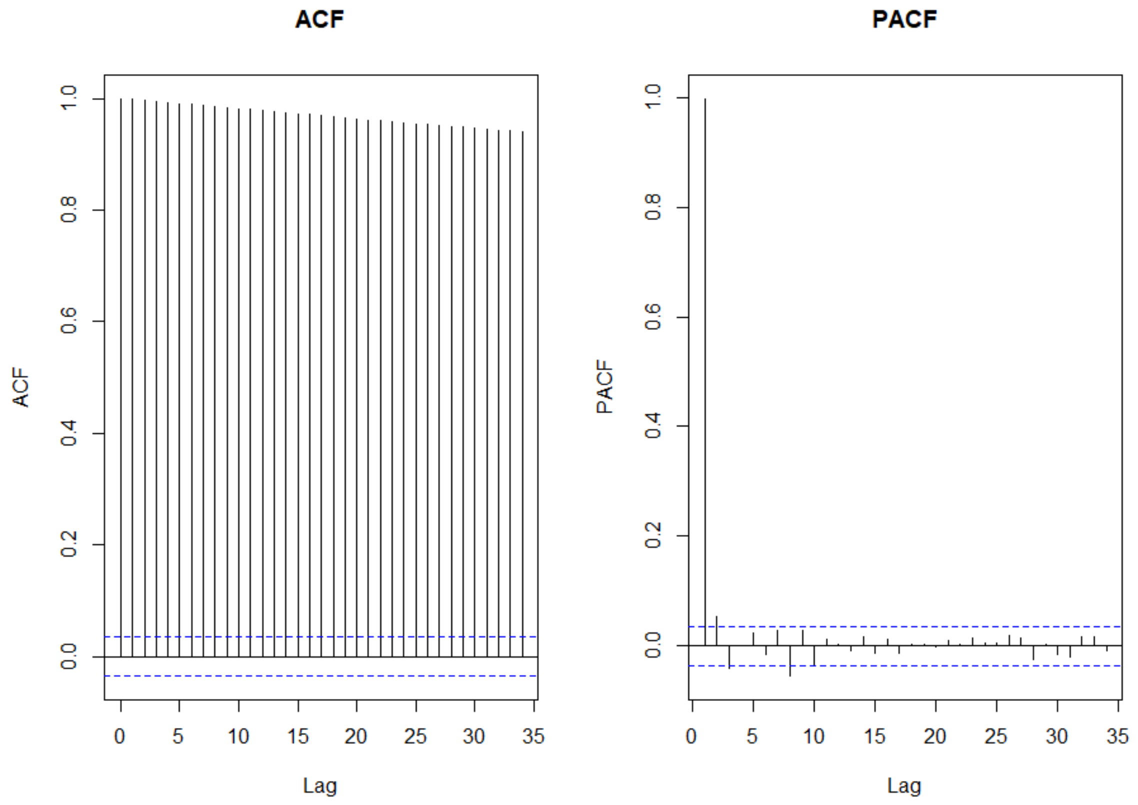
(iii) Examine the ACF and PACF functions. [10 marks]

An ACF is a type of statistical graph which plots the autocorrelation function. It displays the correlation between values of a time series at different points in time.

A PACF is a type of statistical graph which plots the partial autocorrelation function. It displays the correlation between values of a time series at different points in time, with one point excluded from consideration.

```
acf(GSPC$GSPC.Close)
pacf(GSPC$GSPC.Close)
```

Output:



ACF and PACF functions are used to predict the number of lags that are essential to predict the next outcome i.e the ‘p’ and ‘q’ values.

ACF will dampen exponentially and helps determine the q order for MA models.

PACF will dampen exponentially and helps determine the p order for AR models.

(iv) Perform the Ljung-Box test and describe the test-hypothesis and report/comment on the result. [10 marks]

H0 (Null Hypothesis): The no autocorrelation between the signal and its lagged version i.e white noise is present

H1 (Alternate Hypothesis): There is significant autocorrelation between the signal and its lagged version i.e no white noise.

```
Box.test(modelfit$residuals,lag=2,type="Ljung-Box")
```

Output:

```
data:  modelfit$residuals
X-squared = 0.024193, df = 2, p-value = 0.988
```

Since, the p value is greater than 0.05, and almost equal to 1 for lag=2, it can be concluded that Null Hypothesis is false and there is signifcant autocorrelation between the signal and its lagged version i.e no white noise.

(v) Check the data for stationarity using the correct test statistic and comment on the output. [10 marks]

Augmented Dicky Fuller(ADF) Test statistic is used to check the stationarity of dataset using the p value.

H0 (Null Hypothesis): The time series is regarded as non-stationary.
H1 (Alternate Hypothesis): The time series is regarded as stationary.

```
print(adf.test(GSPC$GSPC.Close))
```

Output:

```
data:  GSPC$GSPC.Close
Dickey-Fuller = -3.8165, Lag order = 14, p-value = 0.01815
alternative hypothesis: stationary
```

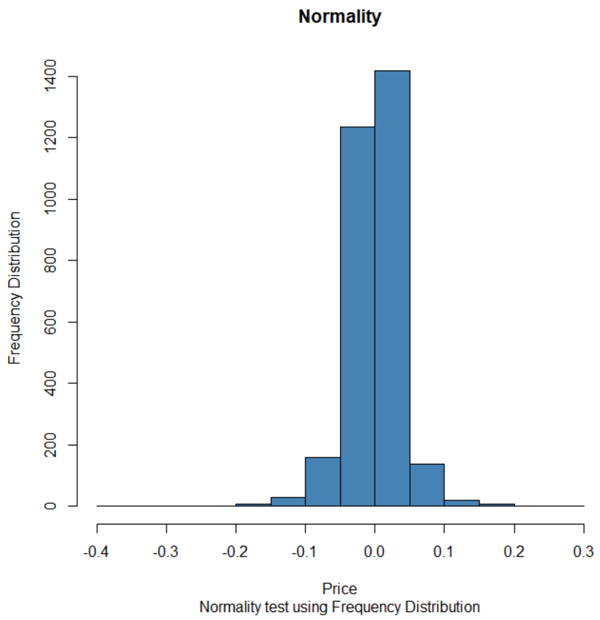
The test statistic and p-value come out to be equal to -3.8165 and 0.6925 respectively. Since the p-value is lesser than 0.05, hence null hypothesis can be rejected. It implies that the time series is stationary. In simple words, we can say that it does not possess some time-dependent structure and possesses constant variance over time.

(vi) Perform a normality test of your choice on the return series and report the output. Write down the hypothesis test and comment on the p-value. [10 marks]

Normality test is performed by Analysis of Graphs using Frequency Distribution plot.

When the series is plotted on a Frequency Distribution plot, the normal distribution can be seen as a bell-shaped curve. The majority of observations are being around the mean value which can be seen as the centre of the curve.
Hence, the data is sampled from a partially normal distribution.

```
hist(resid(modelfit),col='steelblue',main="Normality",sub="Normality test using Frequency Distribution",xlab="Price",ylab="Frequency Distribution")
```



Normality test is performed by Kolmogorov-Smirnov Test.
H0 (Null Hypothesis): Values are sampled from dataset that follows Normal Distribution.
H1 (Alternate Hypothesis): Values are not sampled from dataset that follows Normal Distribution.

```
ks.test(returns,'pnorm')
```

<u>Output:</u>

```
D = 0.51334, p-value < 2.2e-16
alternative hypothesis: two-sided
```

Since the p-value is lesser than 0.05, hence null hypothesis can be rejected. It implies that the time series is non-stationary. In simple words, the values are not sampled from dataset that follows Normal Distribution.

(vii) Fit an ARIMA model and determine the correct lag order: Show the 1-liner codes for output. [15 marks]

```
modelfit <-arima(GSPC$GSPC.Close, order = c(2,1,2))
```

Output:

```
Series: GSPC$GSPC.Close
ARIMA(2,1,2) with drift
Box Cox transformation: lambda= 0.1582214

Coefficients:
         ar1      ar2     ma1     ma2   drift
     -1.6896  -0.8592  1.5723  0.7438  0.0015
s.e.   0.0267   0.0271  0.0341  0.0361  0.0006

sigma^2 = 0.001382:  log likelihood = 5657.89
AIC=-11303.77   AICc=-11303.74   BIC=-11267.7
```

The correct lag order is determined using the ACF and PACF plots.
Since, there is a constant correlation between the lagsin the ACF plot, the ‘q’ value for ‘MA’ model can be a minimum value such as 1 or 2. In our case, it is considered as 2.
Also, as the PACF plot is exponentially decreasing at lag 2, the value of ‘p’ for AR model is considered as 2.
From ADF test, we have p-value less than 0.05. This means the dataset is stationary. Hence, the value for ‘d’ is 1.
Collectively, the lag order is (2,1,2) which represent (p,d,q).

(viii) Report the coefficients for the chosen ARIMA model and show the respective equation given these coefficients. [15 marks]

Coefficients chosen for ARIMA model are defined as:

- **p** is the number of autoregressive terms,
- **d** is the number of differences needed for stationarity, and
- **q** is the number of lagged forecast errors in the prediction equation.

The coefficients chosen for this ARIMA model are p=2, d=1, q=2 or equivalently ARIMA(2,1,2).

AR model equation is similar to that of linear regression. It has an intercept term (δ), regressors (y_{t-i}), parameters (ϕ_{t-i}) and an error term(ϵ). If lag up to p is included in the model like above, the AR process is said to be of order p.

For p=2: $yt = \delta + \phi 1yt - 1 + \phi 2yt - 2 + \cdots + \phi pyt - p + \epsilon t$

The equation for value of d. y denotes the dth difference of Y

For d=1: $yt = Yt - Yt - 1$

MA is another class of linear model. In MA, the output or the variable of interest is modeled via its own imperfectly predicted values of current and previous times. It can be written in terms of error terms:

The regressors are the imperfections (errors) in predicting previous terms. Here the model is specified with positive sign for the parameters. The model above included errors for q lags and said to have an order of q.

For q=2: $yt = \mu + \theta 1\epsilon t - 1 + \theta 2\epsilon t - 2 + \cdots + \theta q\epsilon t - q + \epsilon t$

ϵt represents the autoregressive term

θ represents the coefficient

μ represents a constant term

(ix) The residuals from an ARIMA fit require that:

a. The residuals have zero mean �[�!] = 0

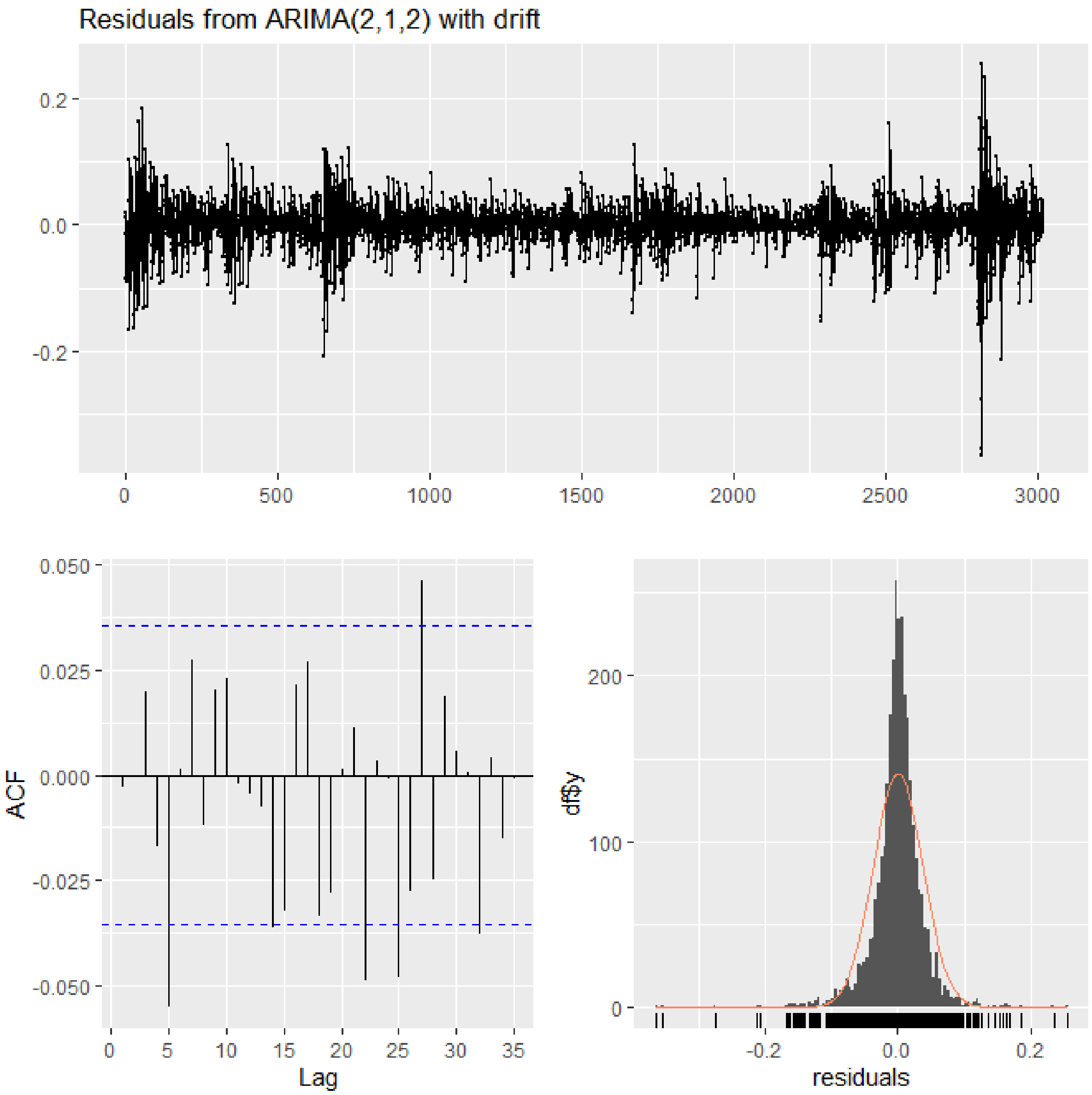
b. Have a finite variance ���[�!] = �"

c. Have zero autocovariance �[�!�#] = 0

Using the results from checkresiduals(fitted_model) function comment on the above. [20 marks]

checkresiduals(modelfit)

<u>Output:</u>



a. **mean(resid(modelfit))**

<u>Output:</u> 1.161908e-05

From the “Residuals from ARIMA(2,1,2)” graph, it can be observed that Residual value varies constantly on both positive and negative sides of the plot almost equally.

And mathematically, output of mean function of residuals is a close value to zero. Hence, the The residuals have zero mean.

b. var(resid(modelfit))

Output: 0.001379417

From the residuals plot, it can be inferred that the residuals on the plot varies on both the sides of zero. Hence, there should be a variance in the residuals. It is proved that variance is finite using the function.

c. acf(resid(modelfit),lag.max = 2,type = c("covariance"),plot = FALSE)

Output:

```
   0               1               2
 1.38e-03      -3.87e-06      -5.21e-07
```

The autocovariance is almost equivalent to zero. It is a very small integer varying on both sides of zero.