HuYuDataInsight LLC

Zhaowei Cai 20240520

Part 1 (a)

```r
library(quantmod)
```

```
## Warning: package 'quantmod' was built under R version 4.2.3

## Loading required package: xts

## Warning: package 'xts' was built under R version 4.2.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 4.2.3

##
## 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 4.2.3

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

```r
library(urca)
```

```
## Warning: package 'urca' was built under R version 4.2.3
```

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.2.3
```

```r
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.2.3
```

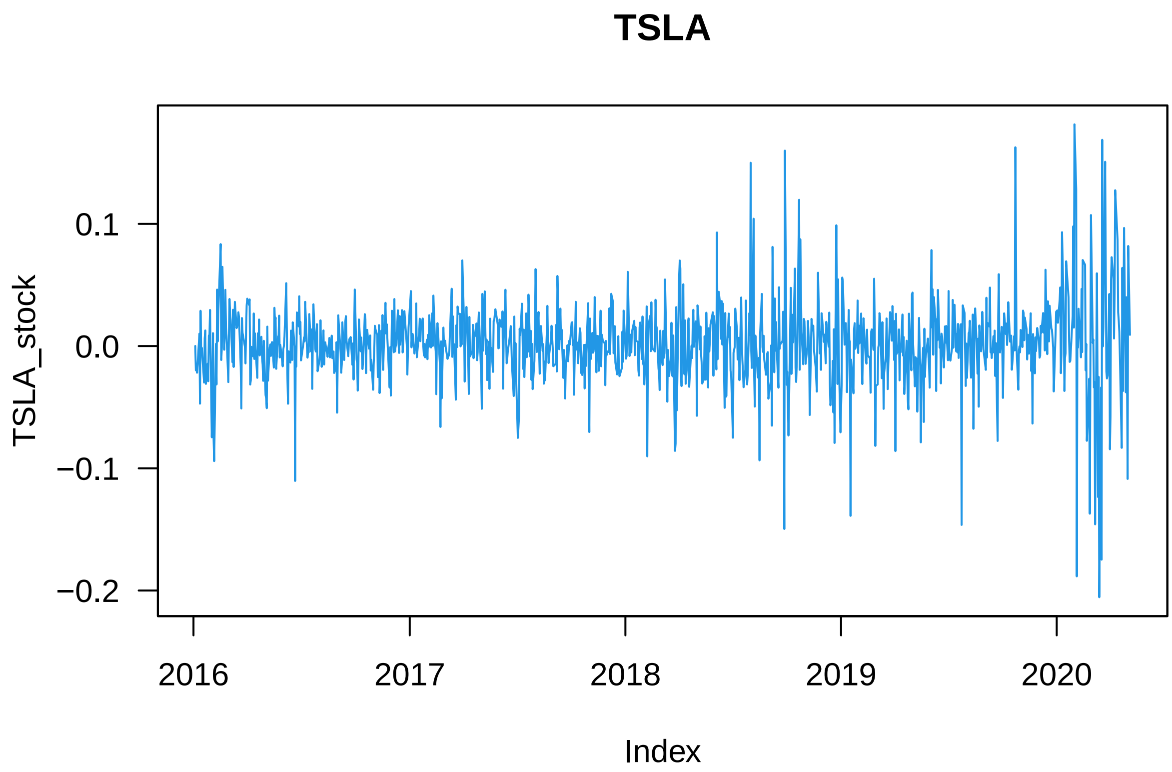
```r
library(fGarch)
```

```
## Warning: package 'fGarch' was built under R version 4.2.3

## NOTE: Packages 'fBasics', 'timeDate', and 'timeSeries' are no longer
## attached to the search() path when 'fGarch' is attached.
##
## If needed attach them yourself in your R script by e.g.,
##         require("timeSeries")

##
## Attaching package: 'fGarch'

## The following object is masked from 'package:TTR':
##
##     volatility
```

```r
library(zoo)
library(tseries)
library(rugarch)
```

```
## Warning: package 'rugarch' was built under R version 4.2.3

## Loading required package: parallel

##
## Attaching package: 'rugarch'

## The following object is masked from 'package:stats':
##
##     sigma
```

```r
data = read.csv('TSLA2.csv')
closing = data$Close # closing price
log_closing = log(data$Close) # log closing price
log_return = na.omit(diff(log(data$Close))) # log return

# Visualize the data
time = as.Date(data$Date, format = '%m/%d/%y')
time <- time[1:1091]
df = data.frame(datefield = time, TSLA = log_return)
TSLA_stock = with(df, zoo(TSLA, order.by = time))
plot.zoo(TSLA_stock, col=4, las=1, main="TSLA")
```

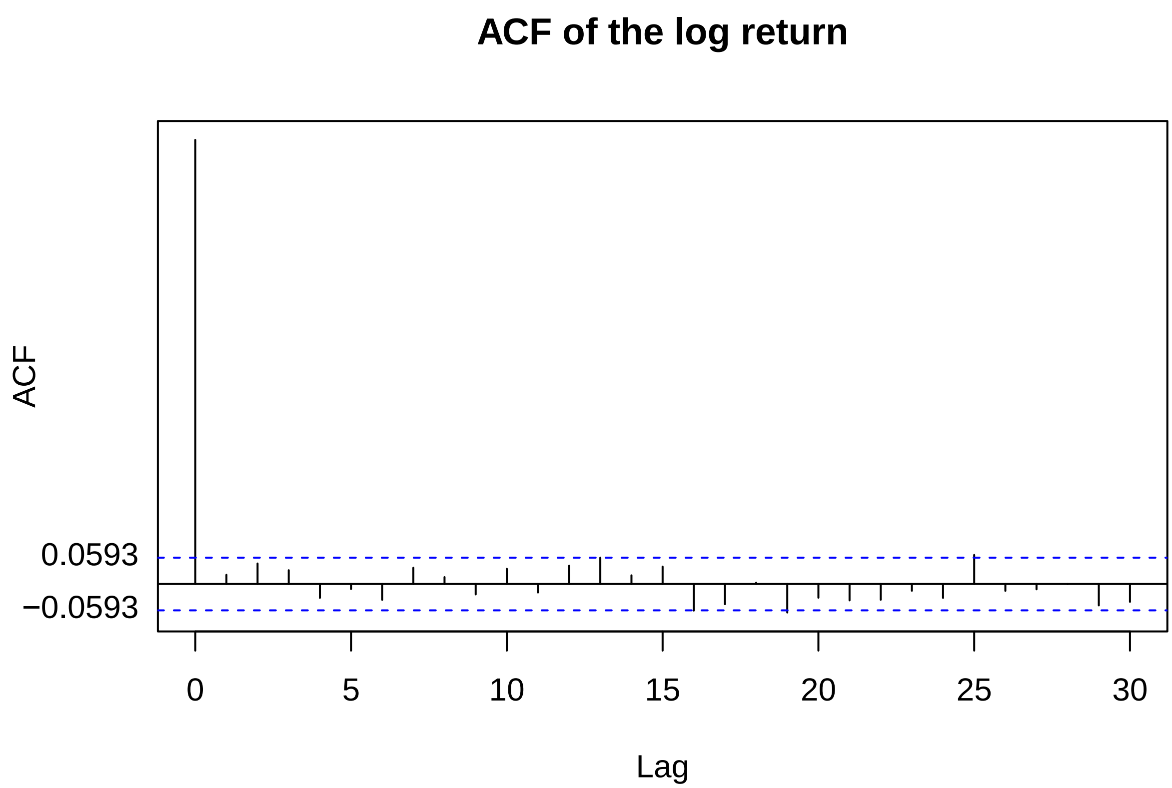



```r
##Check for the trend (the Augmented Dickey-Fuller (ADF) test)
summary(ur.df(log_return, type='trend', lags=20, selectlags="BIC"))
```

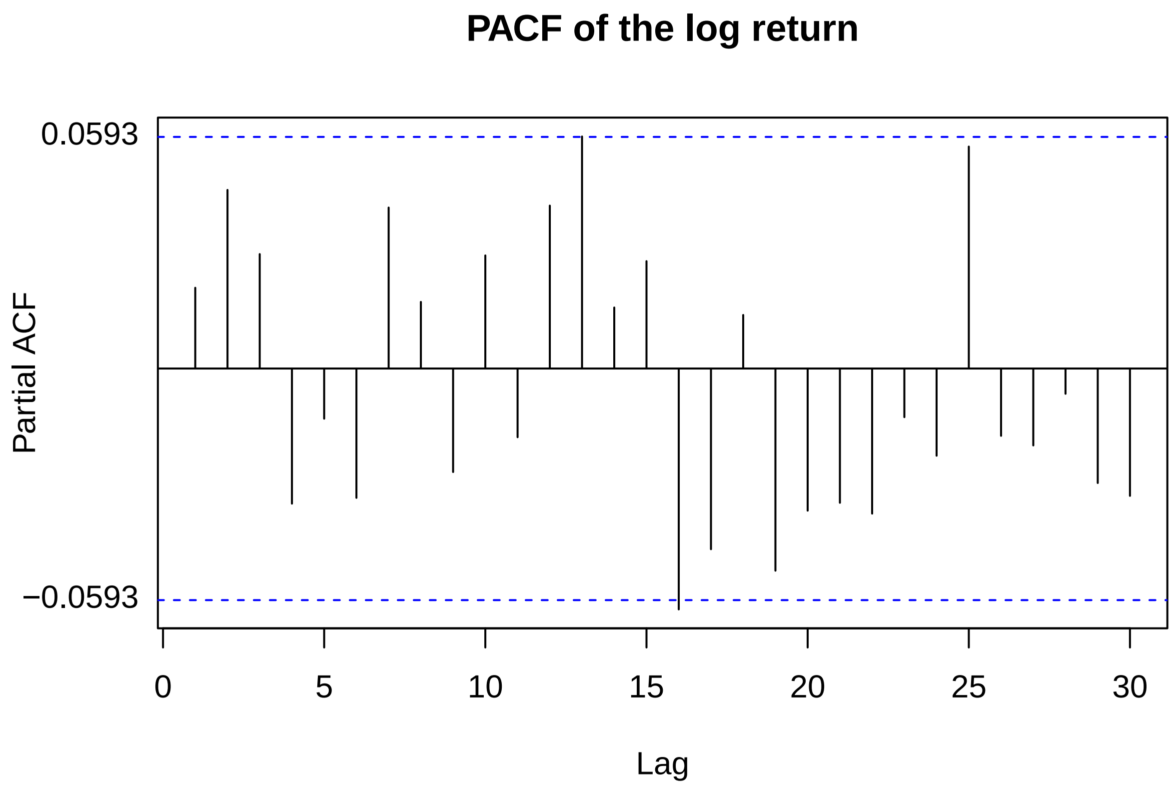
```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.201802 -0.014870 -0.000098  0.016060  0.174363
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.166e-04  2.169e-03   0.054    0.957
## z.lag.1     -9.360e-01  4.276e-02 -21.892   <2e-16 ***
## tt           2.152e-06  3.414e-06   0.630    0.529
## z.diff.lag  -4.573e-02  3.057e-02  -1.496    0.135
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03448 on 1066 degrees of freedom
## Multiple R-squared:  0.492,  Adjusted R-squared:  0.4905
## F-statistic: 344.1 on 3 and 1066 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -21.8922 159.7626 239.636
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

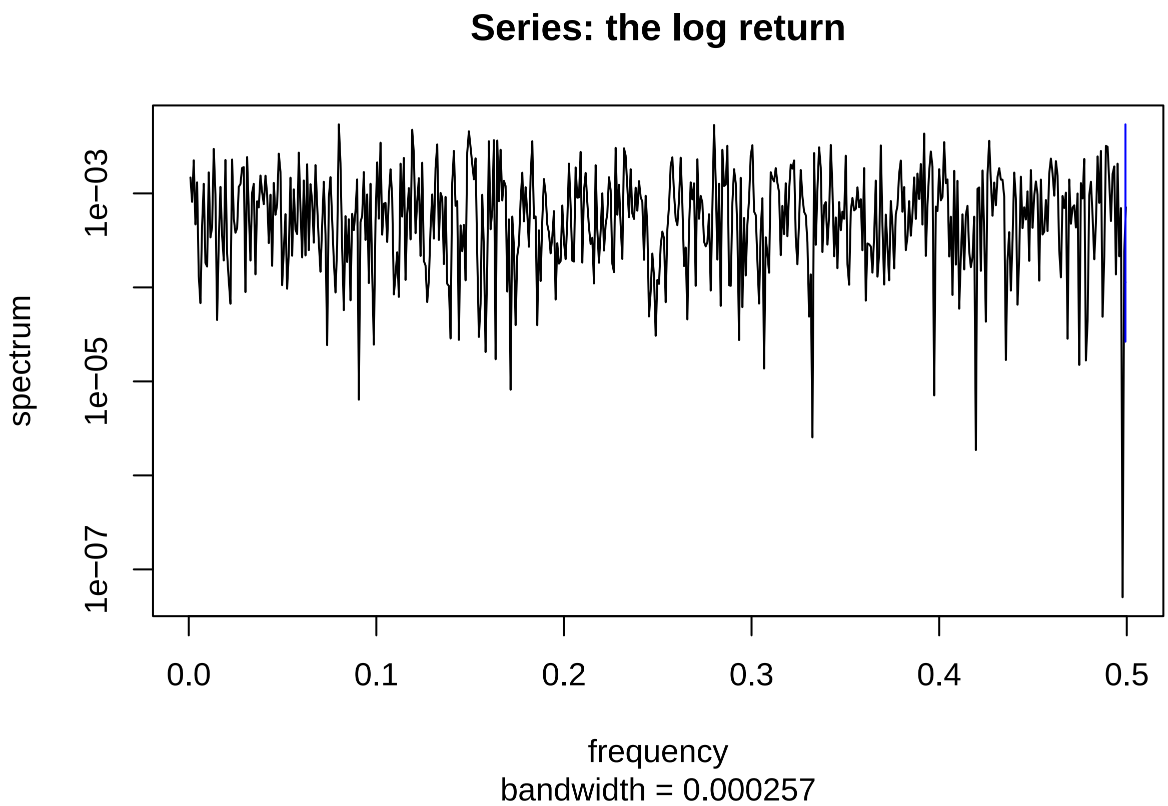
```
# From the result, we can see that the intercept is significantly 0. It means that the mean of the time
# Also, there is no linear trend for this time series because the coefficient for tt is not significant


##Check for the seasonality
n = length(log_return)
acf(log_return,main="ACF of the log return",yaxt="n")
ci=qnorm(c(0.025, 0.975))/sqrt(n)
text(y=ci,par("usr")[1],labels=round(ci,4),pos=2,xpd=TRUE)
```



```
pacf(log_return,main="PACF of the log return",yaxt="n")
text(y=ci,par("usr")[1],labels=round(ci,4),pos=2,xpd=TRUE)
```




```
spec.pgram(log_return,main="Series: the log return")
```




```r
# we cannot find any evidence for seasonality.

# also
adf.test(log_return)
```

```
## Warning in adf.test(log_return): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log_return
## Dickey-Fuller = -9.8415, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

```r
# accept the null hypothesis of non-stationary
# difference is needed.
# log_return = diff(log_closing)
```

(b)

```r
# Remove the drift
# Demean or Difference
adf.test(log_return)
```

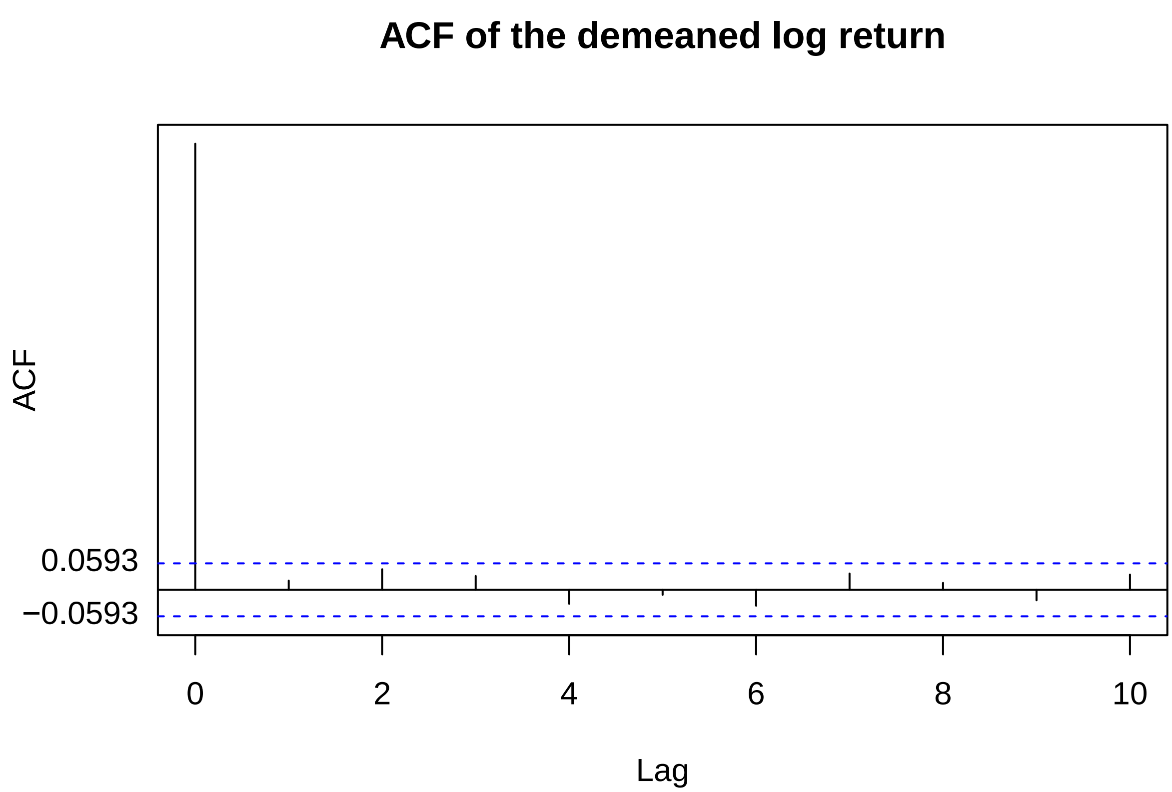
```
## Warning in adf.test(log_return): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log_return
## Dickey-Fuller = -9.8415, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

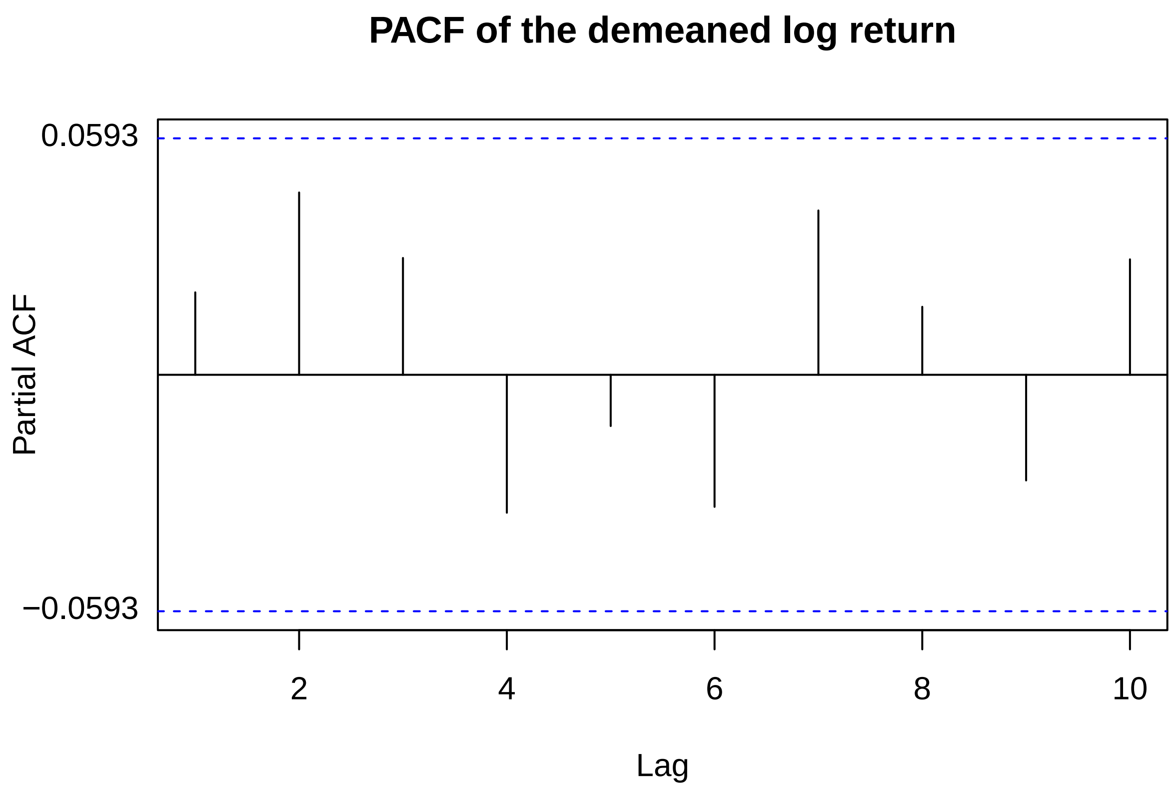
```r
# We know that difference is needed

# 1) demean:
mean(log_return)
```

```
## [1] 0.001132039
```

```r
log_return1=log_return-mean(log_return)

acf(log_return1,lag=10,main="ACF of the demeaned log return",yaxt="n")
text(y=ci,par("usr")[1],labels=round(ci,4),pos=2,xpd=TRUE)
```



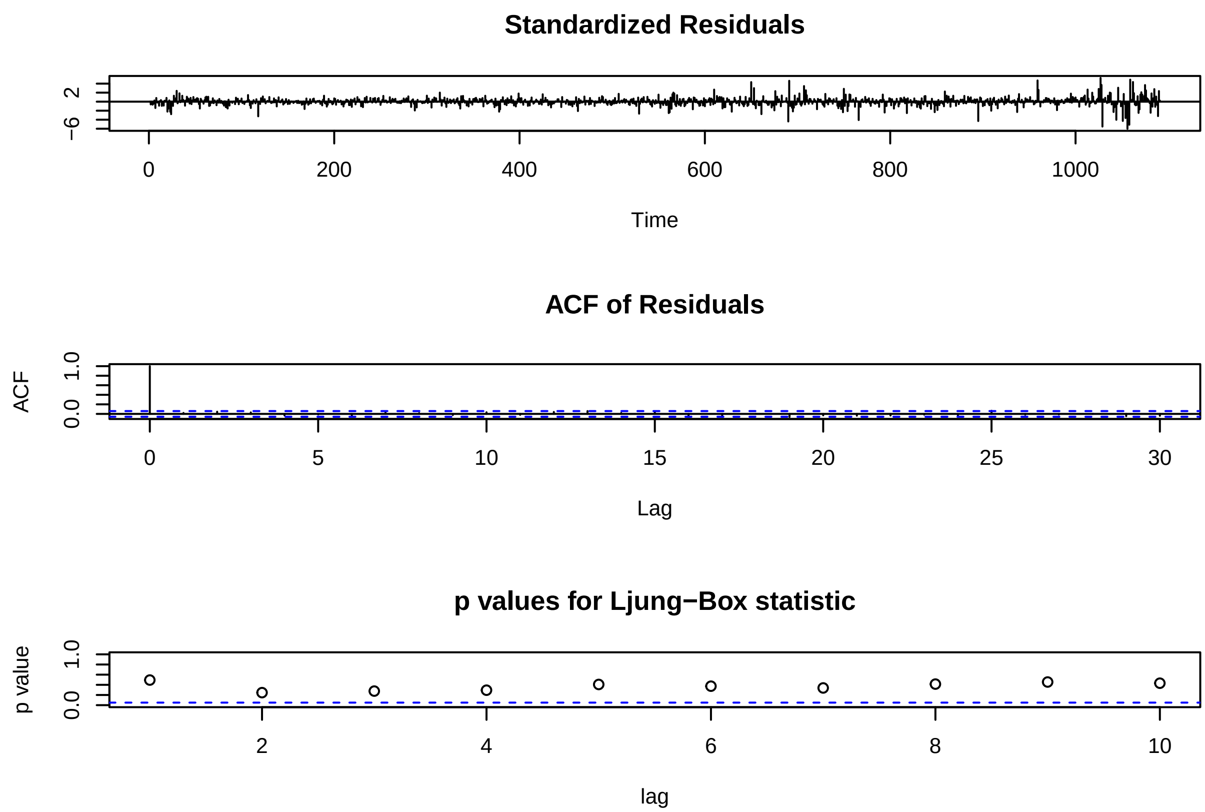

```r
pacf(log_return1,lag=10,main="PACF of the demeaned log return",yaxt="n")
text(y=ci,par("usr")[1],labels=round(ci,4),pos=2,xpd=TRUE)
```




```r
fit1 = auto.arima(log_return1, max.p=25, max.q=25, ic="bic",
                        seasonal=F, lambda=NULL,
                        stepwise=FALSE, approximation=FALSE
                        )
summary(fit1)
```

```
## Series: log_return1
## ARIMA(0,0,0) with zero mean
##
## sigma^2 = 0.001181:  log likelihood = 2129.44
## AIC=-4256.88   AICc=-4256.87   BIC=-4251.88
##
## Training set error measures:
##                          ME       RMSE        MAE MPE MAPE      MASE       ACF1
## Training set -8.334773e-20 0.03436346 0.02287349 100   100 0.6892961 0.02069019
```

```r
# ARIMA(0,1,0)
# also shows that difference is needed
tsdiag(fit1)
```




```
shapiro.test(fit1$residuals)
```

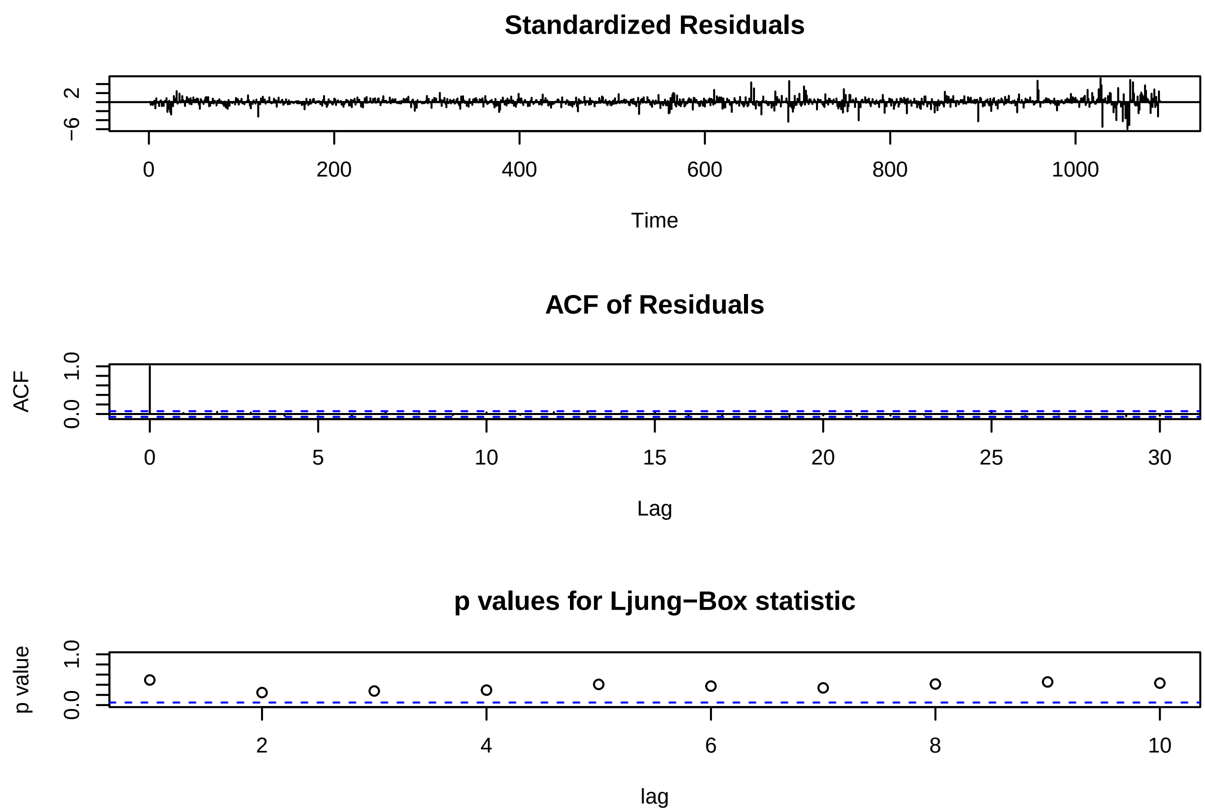
```
##
##  Shapiro-Wilk normality test
##
## data:  fit1$residuals
## W = 0.90385, p-value < 2.2e-16
```

```
# The null-hypothesis of this test is that the population is normally distributed.
# The null hypothesis is rejected and there is evidence that the residuals tested are not normally dist

# 2) difference
# log return = diff(log closing)
# we can difference the data first and fit the log return
fit2 = auto.arima(log_return, max.p=25, max.q=25, ic="bic",
                       seasonal=F, lambda=NULL,
                       stepwise=FALSE, approximation=FALSE
                       )
summary(fit2) # ARMA(0,0)
```

```
## Series: log_return
## ARIMA(0,0,0) with zero mean
##
## sigma^2 = 0.001182:  log likelihood = 2128.85
## AIC=-4255.69   AICc=-4255.69   BIC=-4250.7
```

```
##
## Training set error measures:
##                       ME       RMSE        MAE MPE MAPE      MASE       ACF1
## Training set 0.001132039 0.0343821 0.02288282 100  100 0.6895774 0.02069019
```

```
tsdiag(fit2)
```




```
# Check the normality
shapiro.test(fit2$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fit2$residuals
## W = 0.90385, p-value < 2.2e-16
```

(c)

```
prediction <- forecast(fit1, h=1, level=0.95)
(lower_interval <- as.numeric(exp(prediction$lower+mean(log_closing))))
```

```
## [1] 271.8002
```

```r
(price_forecast <- as.numeric(exp(prediction$mean+mean(log_closing))))
```

```
## [1] 290.7368
```

```r
(upper_interval <- as.numeric(exp(prediction$upper+mean(log_closing))))
```

```
## [1] 310.9928
```

```r
# Print the forecasted closing price and prediction interval
cat("1-day ahead closing price forecast:", price_forecast, "\n")
```

```
## 1-day ahead closing price forecast: 290.7368
```

```r
cat("95% Prediction Interval: (", lower_interval, ", ", upper_interval, ")\n")
```

```
## 95% Prediction Interval: ( 271.8002 ,  310.9928 )
```

(d)

```r
# using log return
summary(ur.df(log_return, type='trend', lags=20, selectlags="BIC"))
```

```
## 
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
## 
## Test regression trend 
## 
## 
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
## 
## Residuals:
##       Min        1Q    Median        3Q       Max 
## -0.201802 -0.014870 -0.000098  0.016060  0.174363 
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  1.166e-04  2.169e-03   0.054    0.957    
## z.lag.1     -9.360e-01  4.276e-02 -21.892   <2e-16 ***
## tt           2.152e-06  3.414e-06   0.630    0.529    
## z.diff.lag  -4.573e-02  3.057e-02  -1.496    0.135    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.03448 on 1066 degrees of freedom
## Multiple R-squared:  0.492,  Adjusted R-squared:  0.4905 
## F-statistic: 344.1 on 3 and 1066 DF,  p-value: < 2.2e-16
## 
```

```
##
## Value of test-statistic is: -21.8922 159.7626 239.636
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.96 -3.41 -3.12
## phi2  6.09  4.68  4.03
## phi3  8.27  6.25  5.34
```

```r
# No drift, no trend
# Stationary

# 1) default mean model of ARMA(1,1)
garch_spec <- ugarchspec()
garch_fit1 <- ugarchfit(spec = garch_spec, data = log_return)
garch_fit1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000683    0.000905   0.75460 0.450488
## ar1     0.460361    0.582350   0.79052 0.429222
## ma1    -0.431647    0.591775  -0.72941 0.465751
## omega   0.000003    0.000004   0.80780 0.419207
## alpha1  0.028888    0.006435   4.48907 0.000007
## beta1   0.970112    0.007653 126.76041 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000683    0.000944  0.72365  0.46928
## ar1     0.460361    0.392217  1.17374  0.24050
## ma1    -0.431647    0.397260 -1.08656  0.27723
## omega   0.000003    0.000016  0.21850  0.82704
## alpha1  0.028888    0.021310  1.35562  0.17522
## beta1   0.970112    0.026072 37.20901  0.00000
##
## LogLikelihood : 2250.086
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.1138
## Bayes        -4.0863
```

```
## Shibata      -4.1139
## Hannan-Quinn -4.1034
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.01948  0.8890
## Lag[2*(p+q)+(p+q)-1][5]   0.21352  1.0000
## Lag[4*(p+q)+(p+q)-1][9]   1.40099  0.9981
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      4.445 0.03500
## Lag[2*(p+q)+(p+q)-1][5]     9.781 0.01048
## Lag[4*(p+q)+(p+q)-1][9]    11.687 0.02135
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]     1.089 0.500 2.000  0.2967
## ARCH Lag[5]     1.303 1.440 1.667  0.6455
## ARCH Lag[7]     2.646 2.315 1.543  0.5828
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  11.3825
## Individual Statistics:
## mu     0.05831
## ar1    0.17110
## ma1    0.17488
## omega  1.86778
## alpha1 0.45898
## beta1  0.38700
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias          0.294031 0.7688
## Negative Sign Bias 1.356746 0.1751
## Positive Sign Bias 0.006052 0.9952
## Joint Effect       2.027274 0.5668
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
```

```
## 1    20     82.84    5.994e-10
## 2    30     91.37    2.234e-08
## 3    40     98.26    5.105e-07
## 4    50    117.30    1.578e-07
##
##
## Elapsed time : 0.278511
```

```r
# 2) Fit the mean model first
arma_model <- auto.arima(log_return)
arma_model
```

```
## Series: log_return
## ARIMA(0,0,0) with zero mean
##
## sigma^2 = 0.001182:  log likelihood = 2128.85
## AIC=-4255.69   AICc=-4255.69   BIC=-4250.7
```

```r
garch_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
                         mean.model = list(armaOrder = c(0,0)))
garch_fit2 <- ugarchfit(spec = garch_spec, data = arma_model$residuals)
garch_fit2
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000682    0.000862   0.79163 0.428577
## omega   0.000003    0.000004   0.80597 0.420261
## alpha1  0.028864    0.006461   4.46741 0.000008
## beta1   0.970136    0.007683 126.26416 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000682    0.000935  0.72954  0.46567
## omega   0.000003    0.000016  0.21749  0.82783
## alpha1  0.028864    0.021416  1.34779  0.17773
## beta1   0.970136    0.026212 37.01180  0.00000
##
## LogLikelihood : 2249.554
##
## Information Criteria
## ------------------------------------
```

```
##
## Akaike       -4.1165
## Bayes        -4.0982
## Shibata      -4.1165
## Hannan-Quinn -4.1096
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.120  0.2899
## Lag[2*(p+q)+(p+q)-1][2]     1.412  0.3820
## Lag[4*(p+q)+(p+q)-1][5]     1.705  0.6896
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      5.099 0.02394
## Lag[2*(p+q)+(p+q)-1][5]     9.845 0.01009
## Lag[4*(p+q)+(p+q)-1][9]    11.666 0.02159
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]     1.140 0.500 2.000  0.2857
## ARCH Lag[5]     1.346 1.440 1.667  0.6336
## ARCH Lag[7]     2.661 2.315 1.543  0.5798
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  10.9868
## Individual Statistics:
## mu     0.06385
## omega  1.85934
## alpha1 0.46376
## beta1  0.39080
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.07 1.24 1.6
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.1943 0.8460
## Negative Sign Bias  1.3061 0.1918
## Positive Sign Bias  0.1315 0.8954
## Joint Effect        1.9298 0.5871
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
```

```
##   group statistic p-value(g-1)
## 1    20     87.53    9.046e-11
## 2    30     99.35    1.244e-09
## 3    40    116.08    1.387e-09
## 4    50    127.56    6.369e-09
##
##
## Elapsed time : 0.119612
```

```r
# 3) If difference is needed (here no need)
arma_model <- auto.arima(diff(log_return))
arma_model
```

```
## Series: diff(log_return)
## ARIMA(5,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5
##       -0.8092  -0.5969  -0.4124  -0.2938  -0.1394
## s.e.   0.0300   0.0378   0.0400   0.0379   0.0302
##
## sigma^2 = 0.00139:  log likelihood = 2040.74
## AIC=-4069.48   AICc=-4069.4   BIC=-4039.52
```

```r
garch_spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
                         mean.model = list(armaOrder = c(5,0)))
garch_spec <- ugarchspec()
garch_fit3 <- ugarchfit(spec = garch_spec, data = arma_model$residuals)
garch_fit3
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error     t value Pr(>|t|)
## mu      0.000030    0.000004     7.11428  0.00000
## ar1     0.762116    0.017900    42.57717  0.00000
## ma1    -0.995392    0.000108 -9174.82624  0.00000
## omega   0.000004    0.000004     0.96313  0.33548
## alpha1  0.031056    0.005219     5.94993  0.00000
## beta1   0.967944    0.006054   159.89403  0.00000
##
## Robust Standard Errors:
##         Estimate  Std. Error     t value Pr(>|t|)
```

```
## mu      0.000030    0.000016     1.82668 0.067748
## ar1     0.762116    0.015674    48.62330 0.000000
## ma1    -0.995392    0.000128 -7746.77405 0.000000
## omega   0.000004    0.000011     0.33478 0.737789
## alpha1  0.031056    0.012005     2.58686 0.009685
## beta1   0.967944    0.014650    66.07003 0.000000
##
## LogLikelihood : 2231.27
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.0831
## Bayes        -4.0556
## Shibata      -4.0831
## Hannan-Quinn -4.0727
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic   p-value
## Lag[1]                      6.696 9.665e-03
## Lag[2*(p+q)+(p+q)-1][5]     9.771 1.103e-12
## Lag[4*(p+q)+(p+q)-1][9]    17.361 2.143e-06
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic  p-value
## Lag[1]                      5.319 0.021097
## Lag[2*(p+q)+(p+q)-1][5]    10.672 0.006182
## Lag[4*(p+q)+(p+q)-1][9]    12.266 0.015773
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]     1.245 0.500 2.000  0.2645
## ARCH Lag[5]     1.273 1.440 1.667  0.6540
## ARCH Lag[7]     2.234 2.315 1.543  0.6678
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  8.6948
## Individual Statistics:
## mu     0.1190
## ar1    0.1482
## ma1    0.1251
## omega  1.6487
## alpha1 0.4760
## beta1  0.4079
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
```

```
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.2214 0.8248
## Negative Sign Bias  0.9960 0.3195
## Positive Sign Bias  0.6703 0.5028
## Joint Effect        1.6736 0.6428
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     79.06    2.695e-09
## 2    30     98.88    1.477e-09
## 3    40    113.16    3.771e-09
## 4    50    128.90    4.146e-09
##
##
## Elapsed time : 0.2211659
```

(e)

```r
# Use the garch_fit2 from d)
forecasted_returns <- ugarchforecast(garch_fit2, n.ahead = 1)

# Assuming the last observed closing price is on December 31, 2019
# You may need to replace this with the actual closing price date
last_close_price <- closing[1510]

# Forecast one day ahead (May 6, 2020)
(price_forecast <- as.numeric(last_close_price*exp(forecasted_returns@forecast$seriesFor)))
```

```
## [1] NA
```

```r
# Calculate the 95% prediction interval
(lower_interval <- as.numeric(price_forecast * exp(qnorm(0.025) * forecasted_returns@forecast$sigmaFor)
```

```
## [1] NA
```

```r
(upper_interval <- as.numeric(price_forecast * exp(qnorm(0.975) * forecasted_returns@forecast$sigmaFor)
```

```
## [1] NA
```

```r
# Print the forecasted closing price and prediction interval
cat("1-day ahead closing price forecast:", price_forecast, "\n")
```

```
## 1-day ahead closing price forecast: NA
```

```r
cat("95% Prediction Interval: (", lower_interval, ", ", upper_interval, ")\n")
```

```
## 95% Prediction Interval: ( NA ,  NA )
```

```r
# wider than the interval in c)
```

The End