FDDRates.R

Shankii

2023-04-24

```
library(tseries)
```

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

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(stats)
library(forecast)
```

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

```
library(TSA)
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
##
## Attaching package: 'TSA'
##
## The following object is masked from 'package:readr':
##
##     spec
##
## The following objects are masked from 'package:stats':
##
##     acf, arima
##
## The following object is masked from 'package:utils':
##
##     tar
```

```
library(urca)
```

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

```
library(FinTS)
```

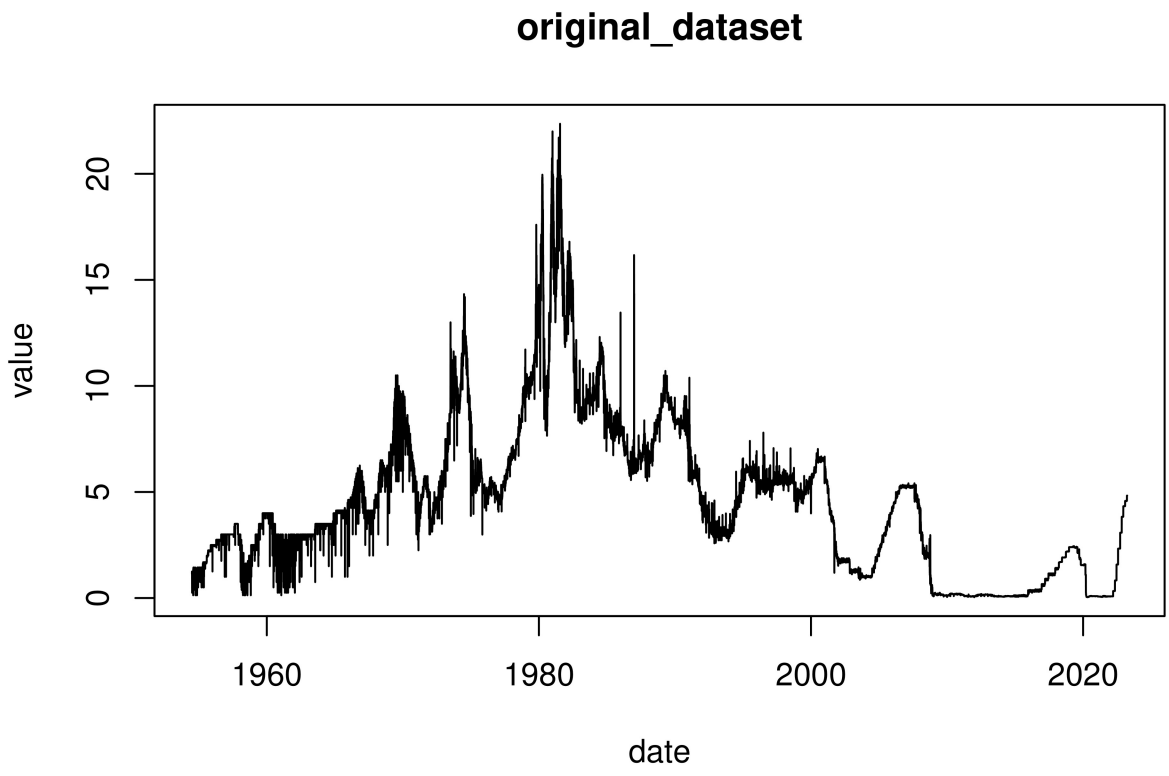
```
## Warning: package 'FinTS' was built under R version 4.2.3
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
##
## Attaching package: 'FinTS'
##
## The following object is masked from 'package:forecast':
##
##     Acf
```

```
df_main = read_csv("E:\\Stevens institute\\SecondSem\\MA641 Time Series\\Project\\archive\\FDDRates.csv
colnames(df_main) = c('date','value')
head(df_main)
```

```
## # A tibble: 6 x 2
##   date       value
##   <date>     <dbl>
## 1 1954-07-01  1.13
## 2 1954-07-02  1.25
## 3 1954-07-03  1.25
## 4 1954-07-04  1.25
## 5 1954-07-05  0.88
## 6 1954-07-06  0.25
```

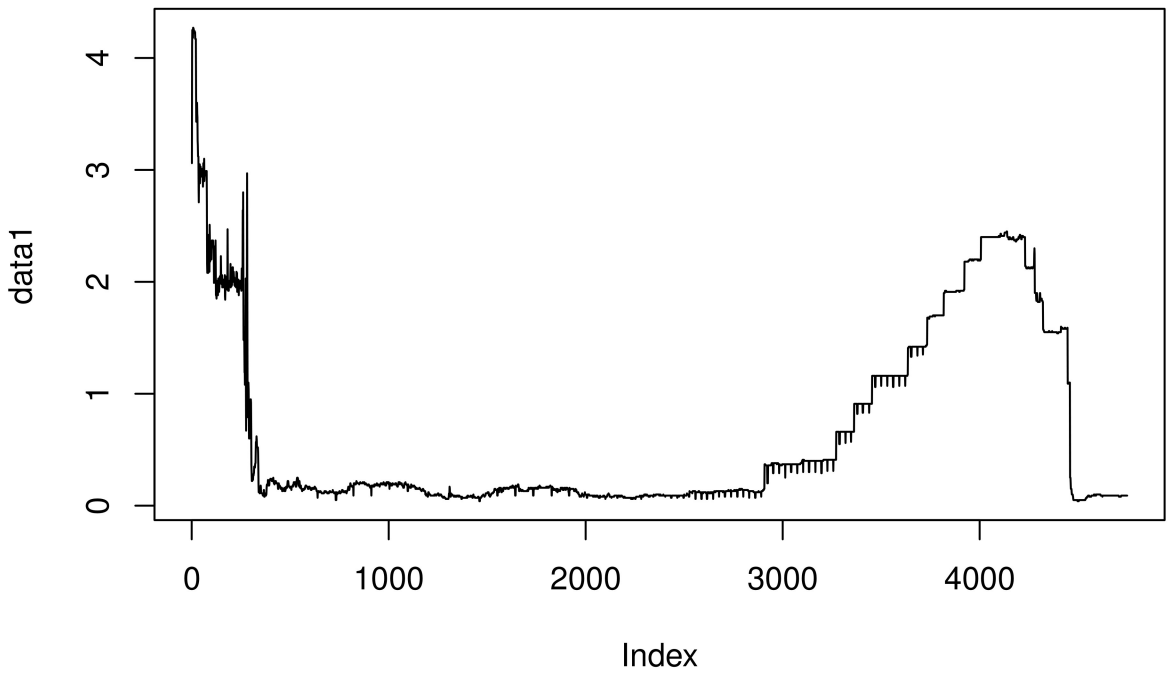
```
df_main$date <- as.Date(df_main$date, "%Y-%m-%d")
plot(df_main, main = 'original_dataset', type='l')
```



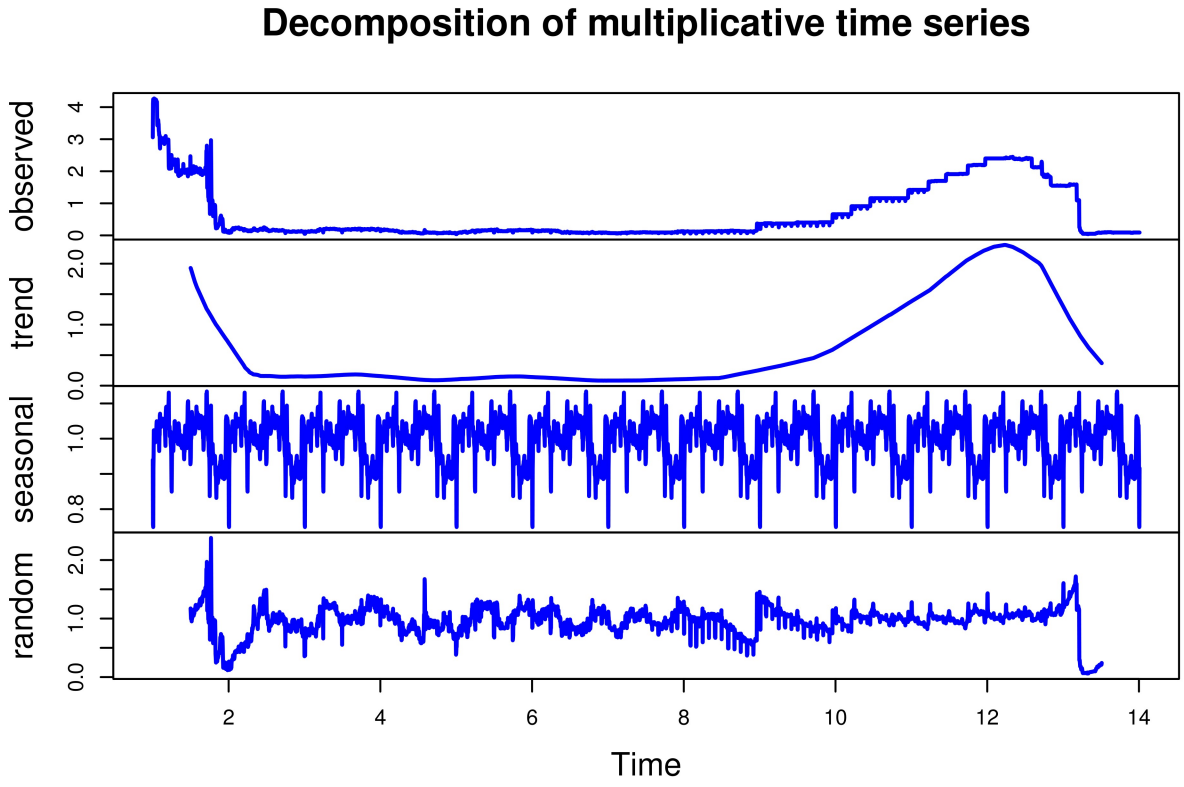

```
################################################################################

tail(df_main)
```

```
## # A tibble: 6 x 2
##   date       value
##   <date>     <dbl>
## 1 2023-03-25  4.83
## 2 2023-03-26  4.83
## 3 2023-03-27  4.83
## 4 2023-03-28  4.83
## 5 2023-03-29  4.83
## 6 2023-03-30  4.83
```

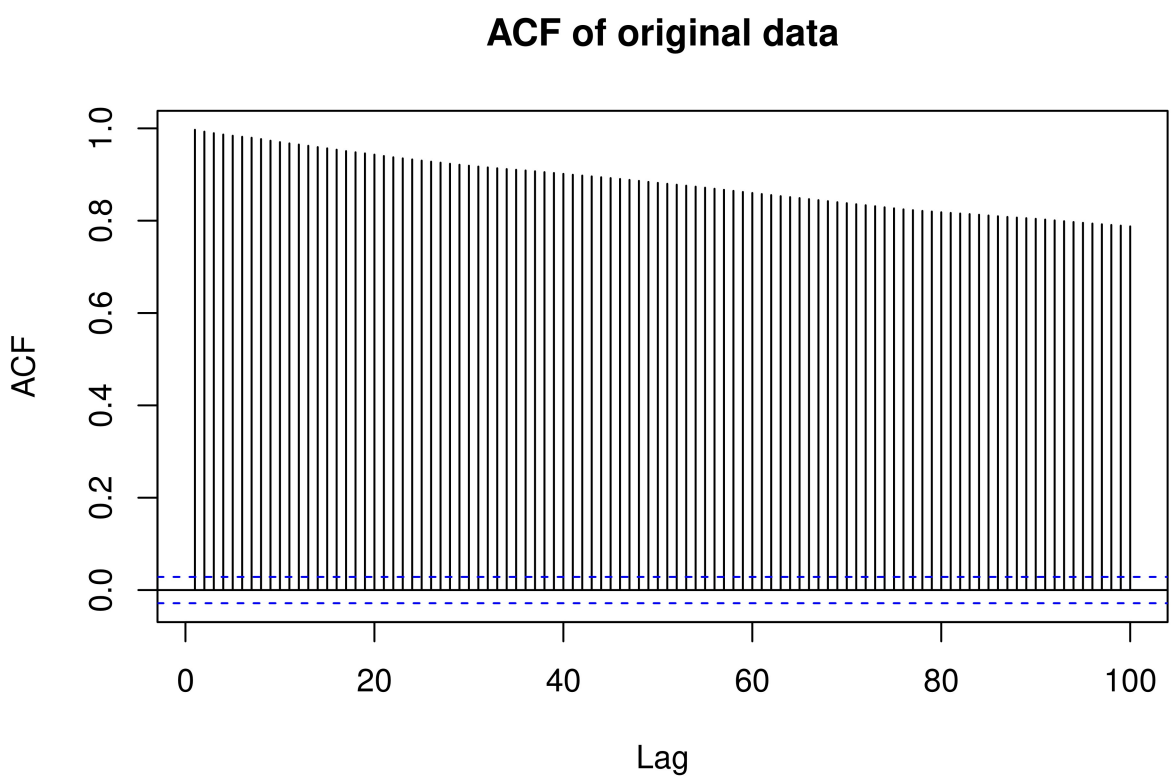
```
# select data between certain dates
start_date <- as.Date("2008-01-01")
end_date <- as.Date("2020-12-31")
df <- subset(df_main, date >= start_date & date <= end_date)
df_test = subset(df_main, date > end_date & date <= as.Date('2021-02-28'))

data1 = df$value
data_365 = ts(data1, frequency = 365)
decompose_data = decompose(data_365, "multiplicative")
plot(data1, type='l')
```



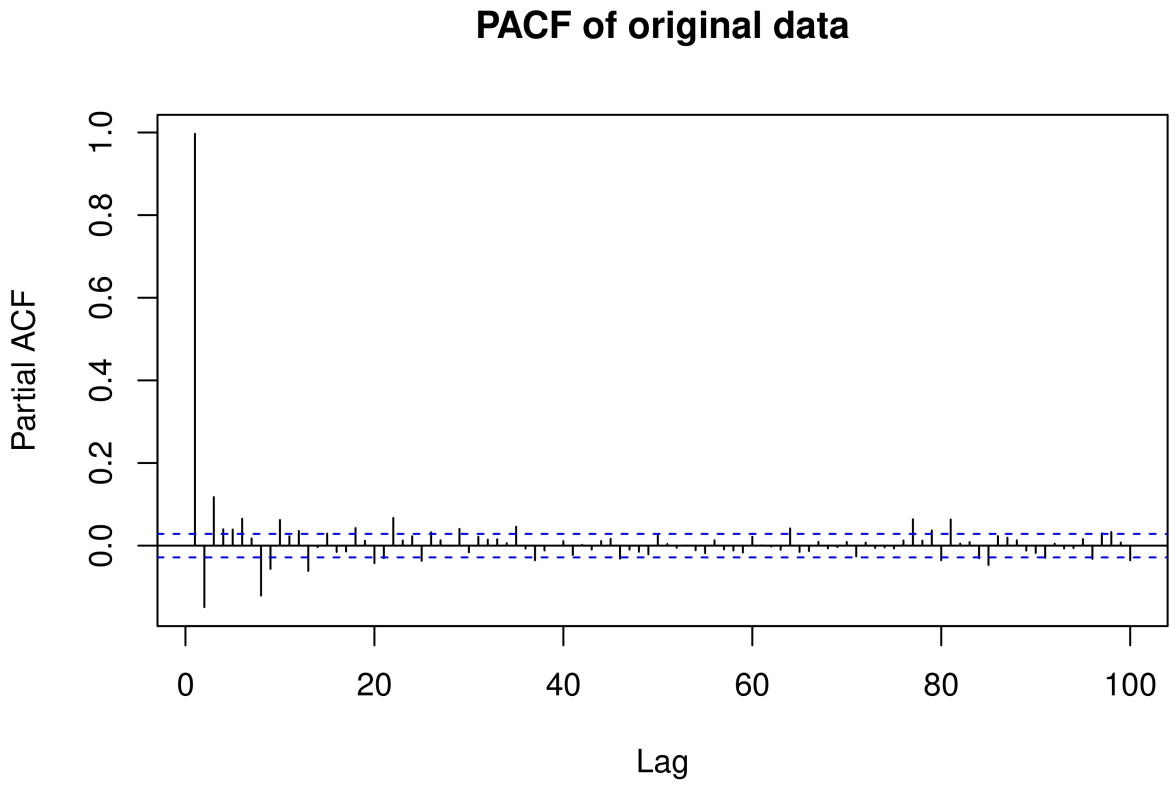

```
plot(decompose_data, type='l',lwd=2, col = 'blue')
```




```
##########################################################

data1 = df$value
acf(data1, main = 'ACF of original data', lag.max = 100)
```




```
## exponentially dying

pacf(data1,main = 'PACF of original data', lag.max = 100 )
```



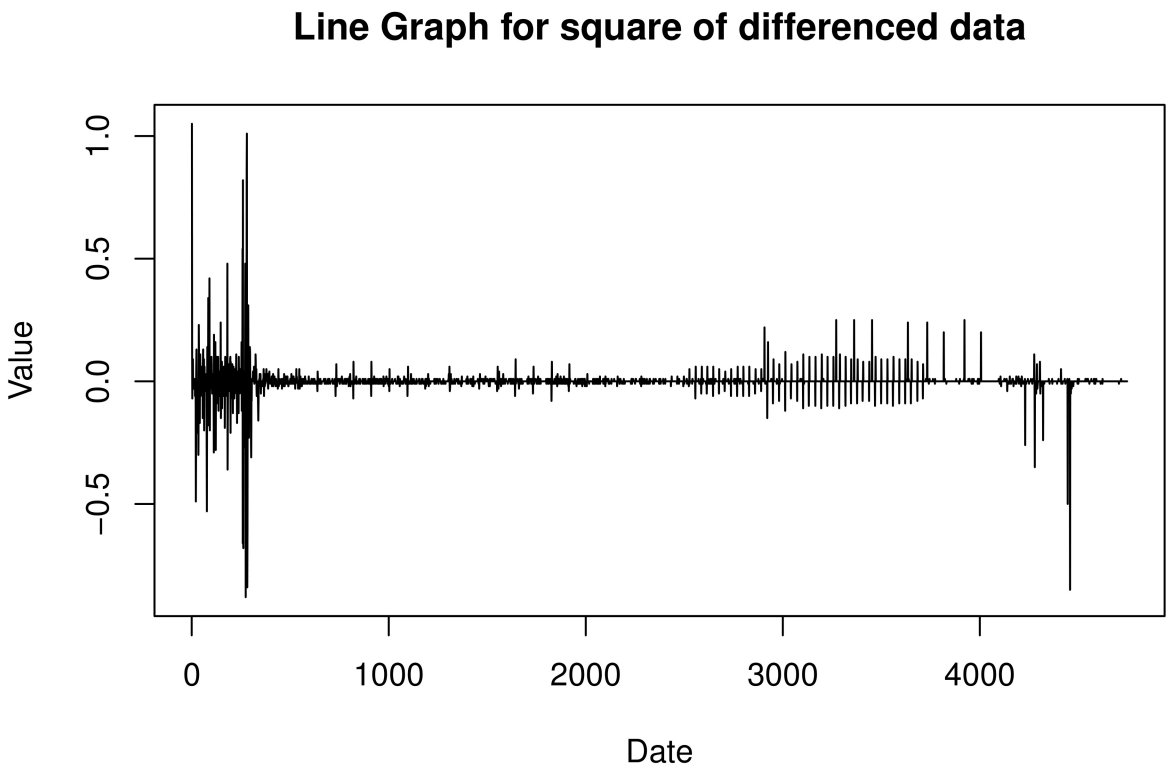

```
## spikes are there at regular intervals

## Performing dickey Fuller Test

adf_test = adf.test(data1)
```

```
## Warning in adf.test(data1): p-value smaller than printed p-value
```

```
print(adf_test)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  data1
## Dickey-Fuller = -5.6844, Lag order = 16, p-value = 0.01
## alternative hypothesis: stationary
```

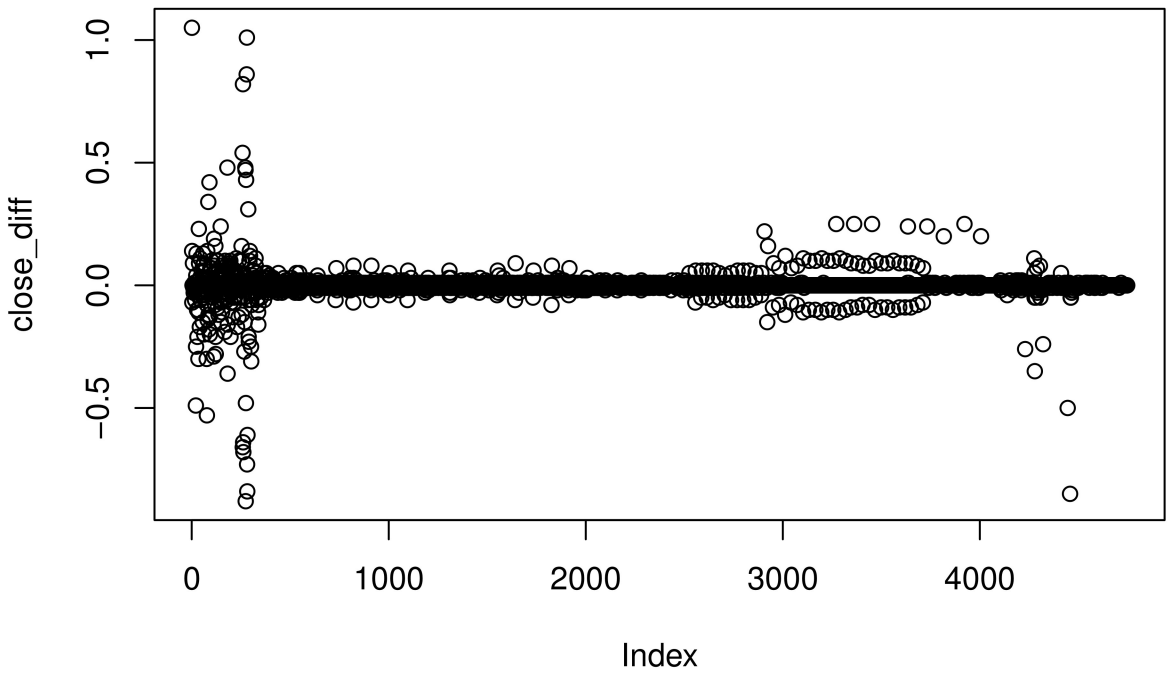
```
######################################################################

## differencing:


close_diff = (diff(data1))

plot(close_diff, type = "l", xlab = "Date", ylab = "Value", main = "Line Graph for square of difference
```



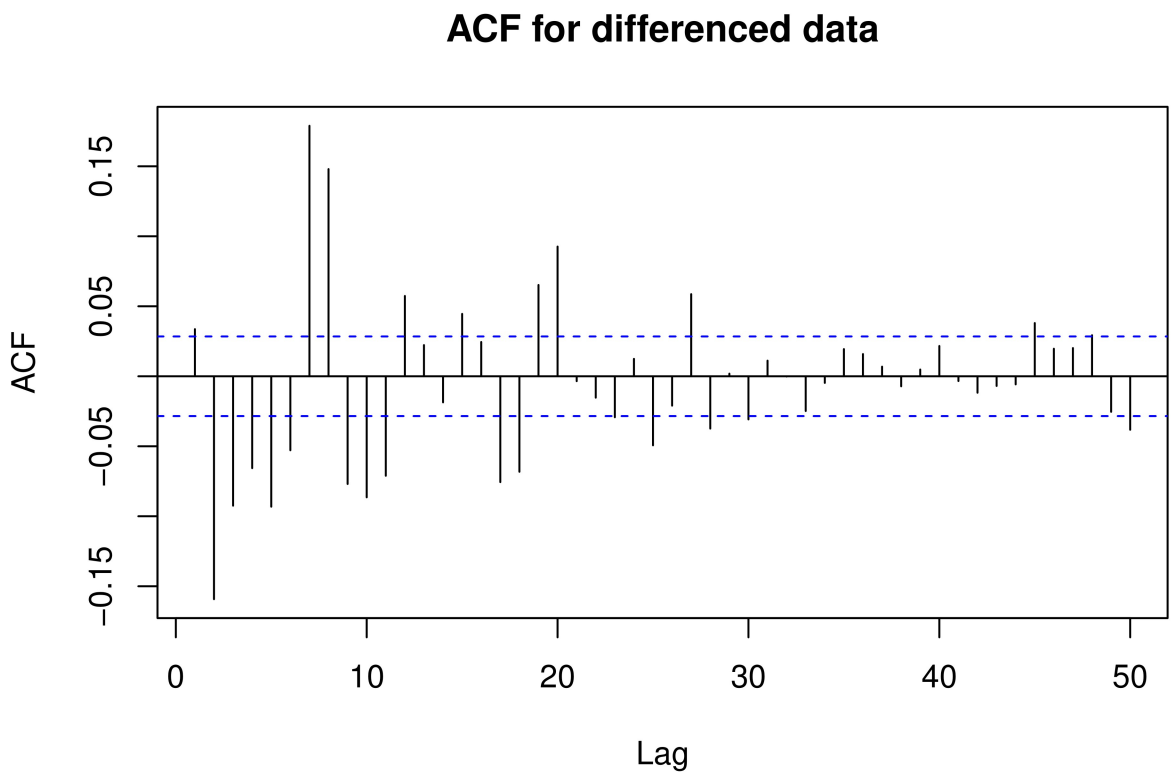

```
## still some blatant peaks are there

adf_test = adf.test(close_diff)
```

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

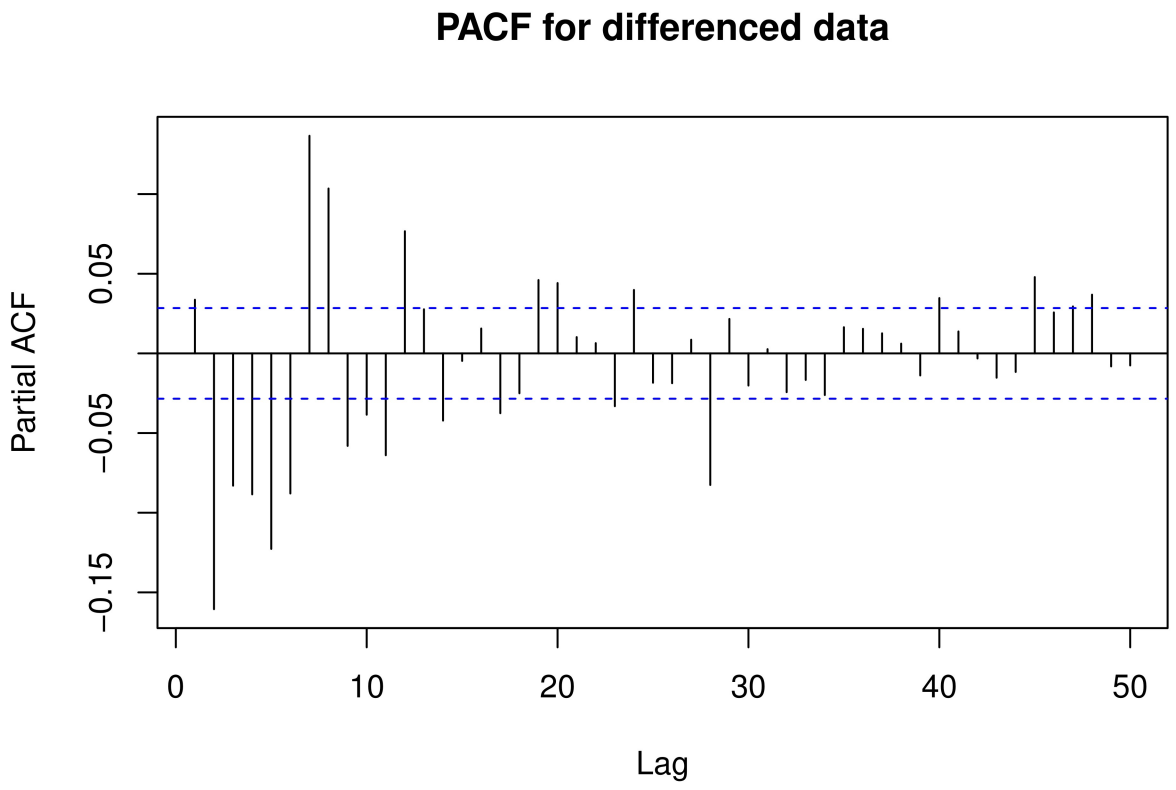
```
## Means the data has become stationary
plot(close_diff)
```




```
## acf and pacf plots

acf(close_diff, lag.max = 50, main = 'ACF for differenced data')
```



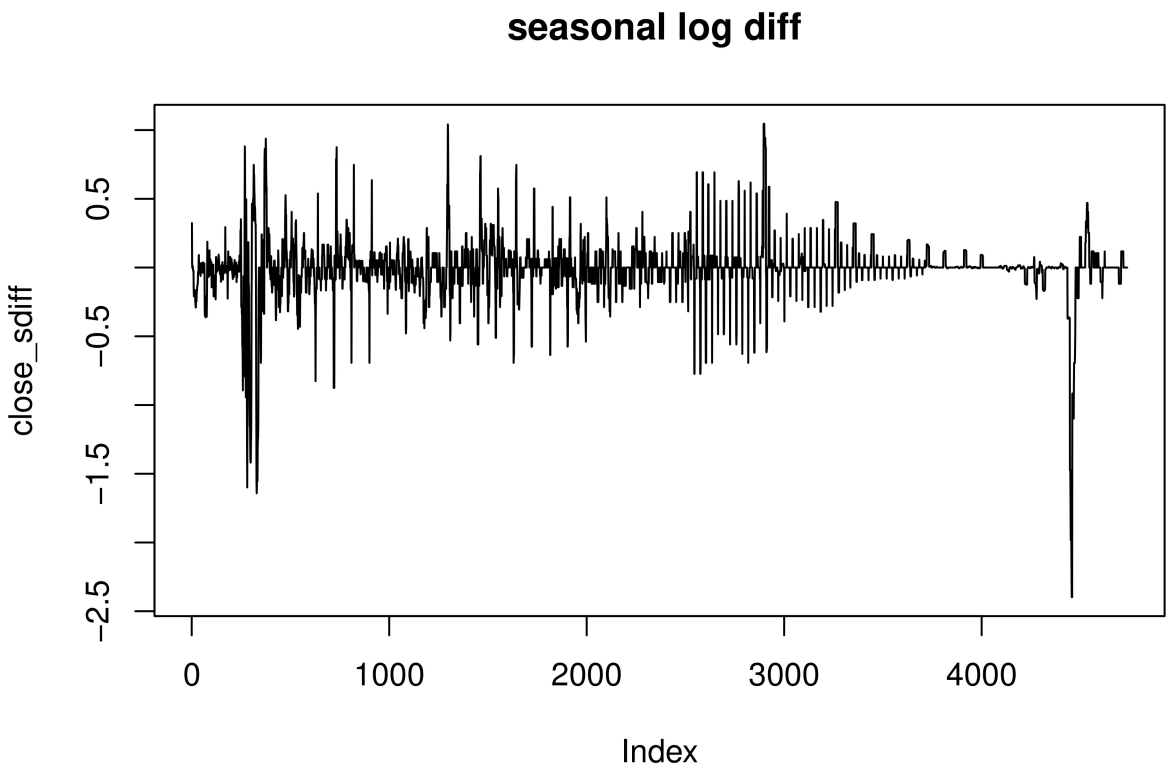

```
pacf(close_diff, lag.max = 50, main = 'PACF for differenced data')
```




```
eacf(close_diff)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x x x x x x x  x  o  o
## 1 x x o o x x x x x o x  x  x  o
## 2 x x o x o x x x x o x  x  x  o
## 3 x x x x o x x x x o x  x  x  x
## 4 x x x o o x o x x o x  x  x  o
## 5 x x x x x x o x x x x  x  x  o
## 6 x x x x x x x o x x x  o  x  o
## 7 x x x x x o o o o o x  o  x  o
```

```
########################################################################
########################################################################

## full data with seasonal and first difference

close_sdiff = diff(log(data1), lag = 12)
plot(close_sdiff, type='l', main = 'seasonal log diff ')
```



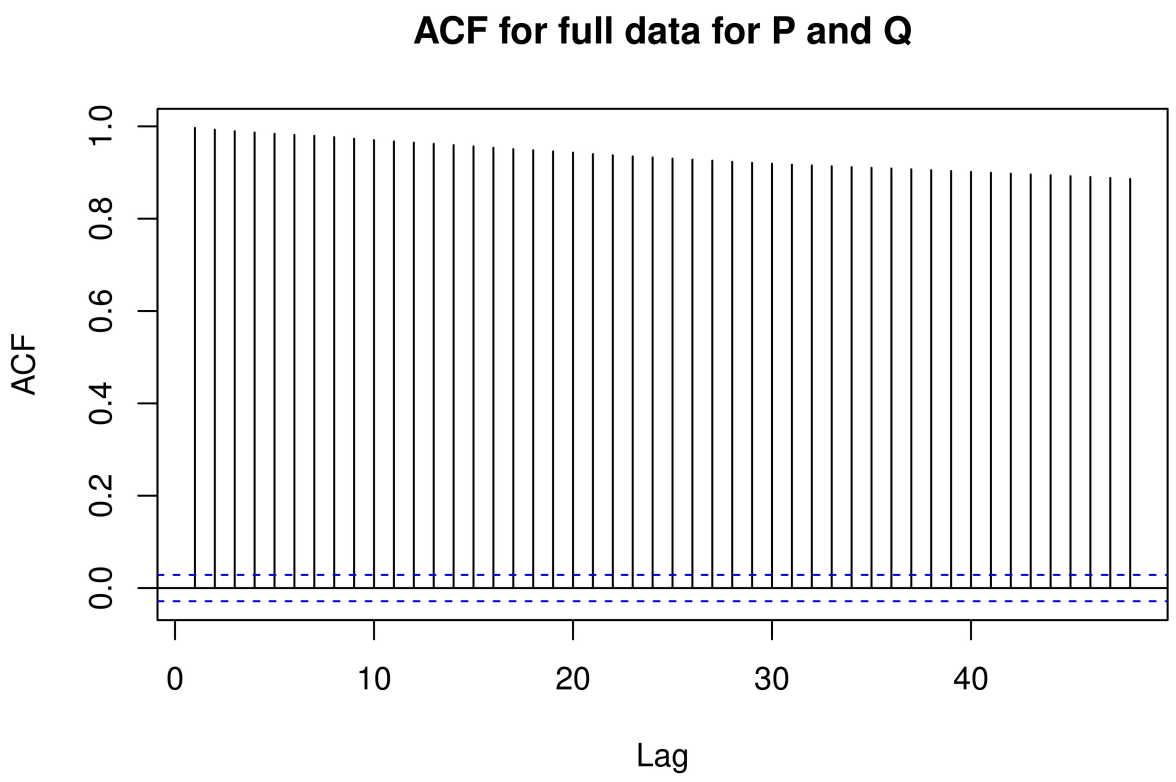

```
adf_test = adf.test(close_sdiff)
```

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

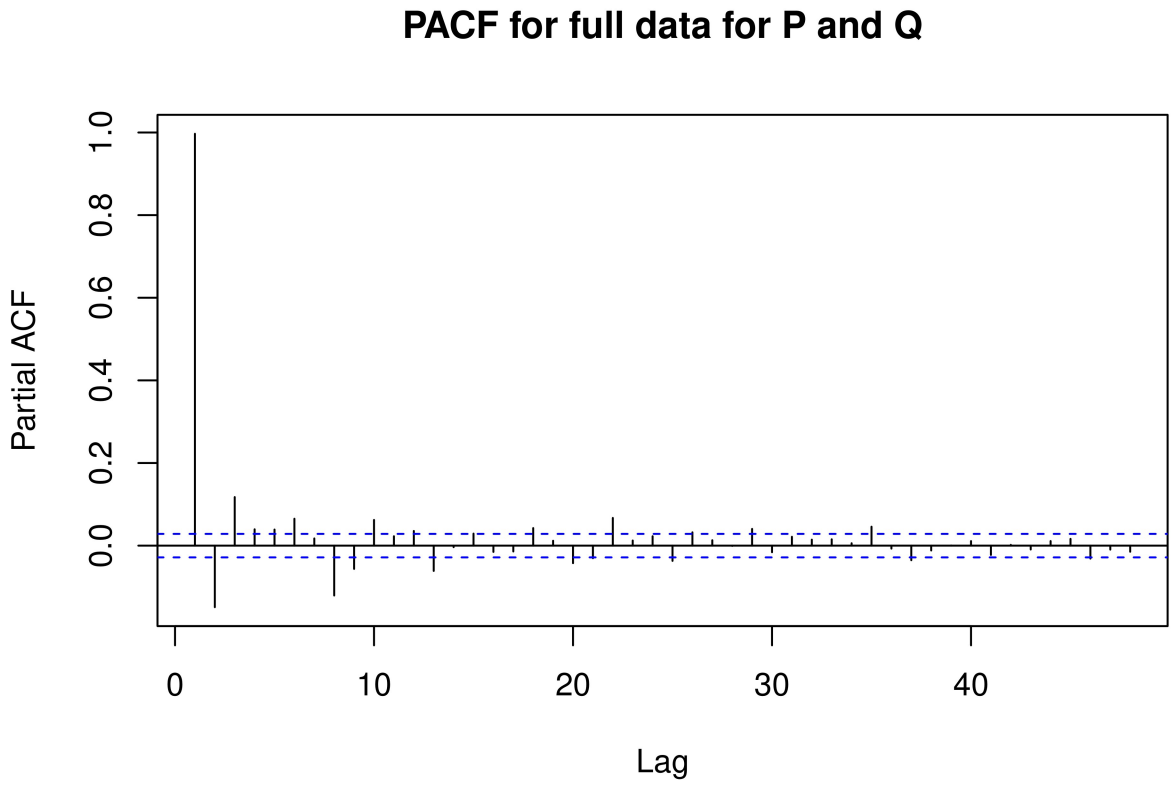
```
adf_test
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  close_sdiff
## Dickey-Fuller = -12.311, Lag order = 16, p-value = 0.01
## alternative hypothesis: stationary
```

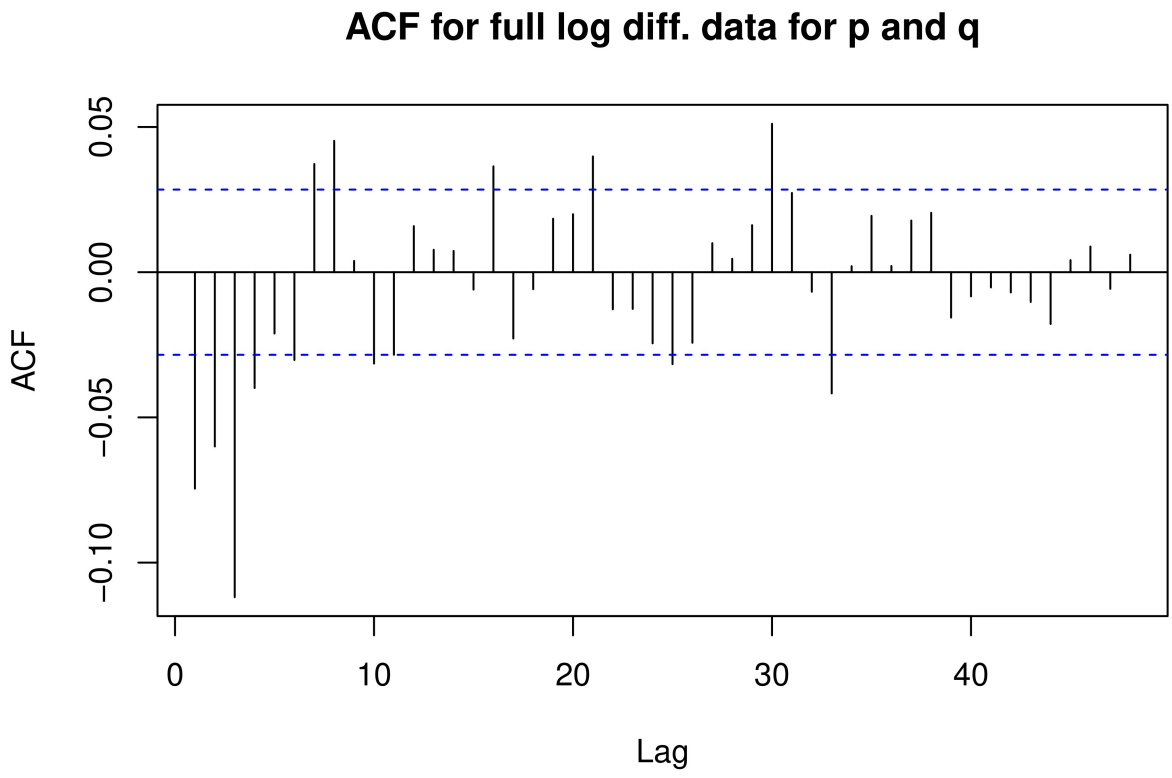
```
 ################################################

## Model Determination:


## for the SARMA Model
acf(as.vector(data1), lag.max = 48, main = 'ACF for full data for P and Q')
```



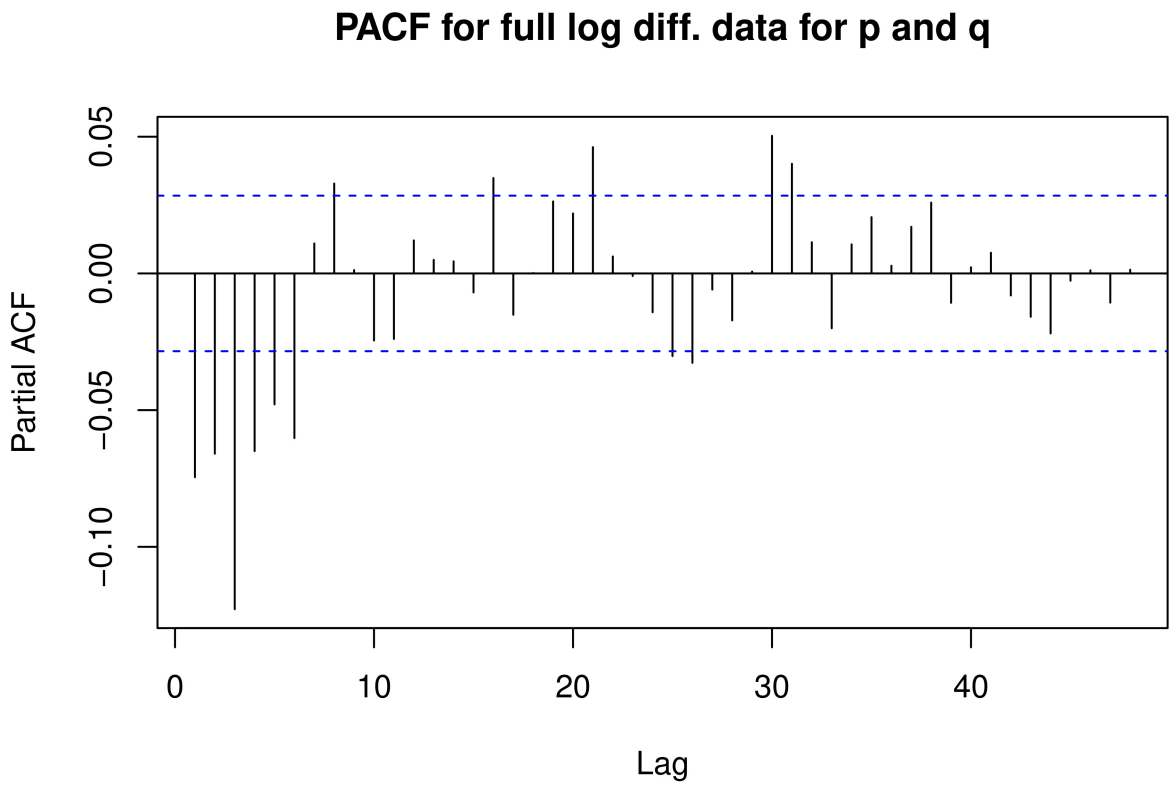

```
pacf(as.vector(data1), lag.max = 48, main = 'PACF for full data for P and Q')
```




```
## from the ACF: its dying: so SMA(Q) = 0
## from the PACF: its significant: so SAR(P) = 3


#########################################################

acf(as.vector(diff(log(data1))), lag.max = 48, main = 'ACF for full log diff. data for p and q')
```




```
pacf(as.vector(diff(log(data1))), lag.max = 48, main = 'PACF for full log diff. data for p and q')
```




```
eacf((diff(data1)))
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x x x x x x x  x  o  o
## 1 x x o o x x x x x o x  x  x  o
## 2 x x o x o x x x x o x  x  x  o
## 3 x x x x o x x x x o x  x  x  x
## 4 x x x o o x o x x o x  x  x  o
## 5 x x x x x x o x x x x  x  x  o
## 6 x x x x x x x o x x x  o  x  o
## 7 x x x x x o o o o o x  o  x  o
```

```
## Possible models:
## ARIMA(1,1,3)
## ARIMA(1,1,2)
## ARIMA(2,1,2)
## ARIMA(3,1,4)



## Model Evaluation
## use the log diff data with lag = 12
## close_sdiff = diff(log(data)) with lag = 12

## SARMA(1,1,3)
sarma113 = arima(close_sdiff,order = c(1,1,3), seasonal = list(order=c(3,1,0), period = 12))
```

```
sarma113
```

```
##
## Call:
## arima(x = close_sdiff, order = c(1, 1, 3), seasonal = list(order = c(3, 1, 0),
##     period = 12))
##
## Coefficients:
##          ar1      ma1      ma2      ma3     sar1     sar2     sar3
##       0.4997  -0.5972  -0.0304  -0.0843  -1.1709  -0.8843  -0.3977
## s.e.  0.1000   0.1005   0.0198   0.0226   0.0135   0.0175   0.0133
##
## sigma^2 estimated as 0.01607:  log likelihood = 3044.37,  aic = -6074.74
```

```
##################

## SARMA(1,1,2)
sarma112 = arima(close_sdiff,order = c(1,1,2), seasonal = list(order=c(3,1,0), period = 12))
sarma112
```

```
##
## Call:
## arima(x = close_sdiff, order = c(1, 1, 2), seasonal = list(order = c(3, 1, 0),
##     period = 12))
##
## Coefficients:
##          ar1      ma1     ma2     sar1     sar2     sar3
##       0.8834  -1.0111  0.0111  -1.1554  -0.8677  -0.3895
## s.e.  0.0079   0.0167  0.0167   0.0136   0.0176   0.0134
##
## sigma^2 estimated as 0.01566:  log likelihood = 3102.13,  aic = -6192.26
```

```
#################

## SARMA(2,1,2)
#sarma212 = arima(close_sdiff,order = c(2,1,2), seasonal = list(order=c(3,1,0), period = 12))
#sarma212

########################

## SARMA(3,1,4)
sarma314 = arima(close_sdiff,order = c(3,1,4), seasonal = list(order=c(3,1,0), period = 12))
```

```
## Warning in log(s2): NaNs produced
```

```
sarma314
```

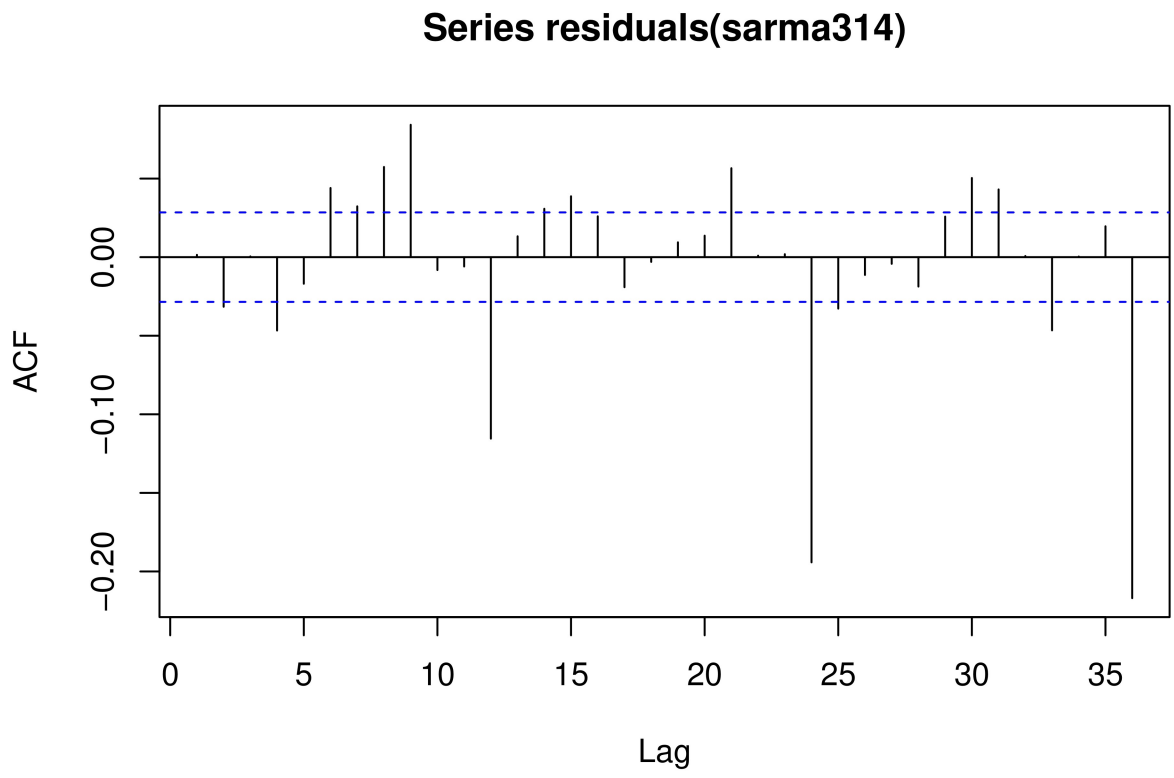
```
##
## Call:
## arima(x = close_sdiff, order = c(3, 1, 4), seasonal = list(order = c(3, 1, 0),
##     period = 12))
##
## Coefficients:
##           ar1      ar2     ar3      ma1      ma2     ma3     ma4     sar1
##       -0.0382  -0.0464  0.7573  -0.0574  -0.0005  -1.000  0.0579  -1.1278
## s.e.   0.0157   0.0117  0.0101   0.0224   0.0041   0.001  0.0212   0.0140
##          sar2     sar3
```

```
##       -0.8362  -0.3740
## s.e.   0.0180   0.0136
##
## sigma^2 estimated as 0.01458:  log likelihood = 3263.52,  aic = -6507.05
```

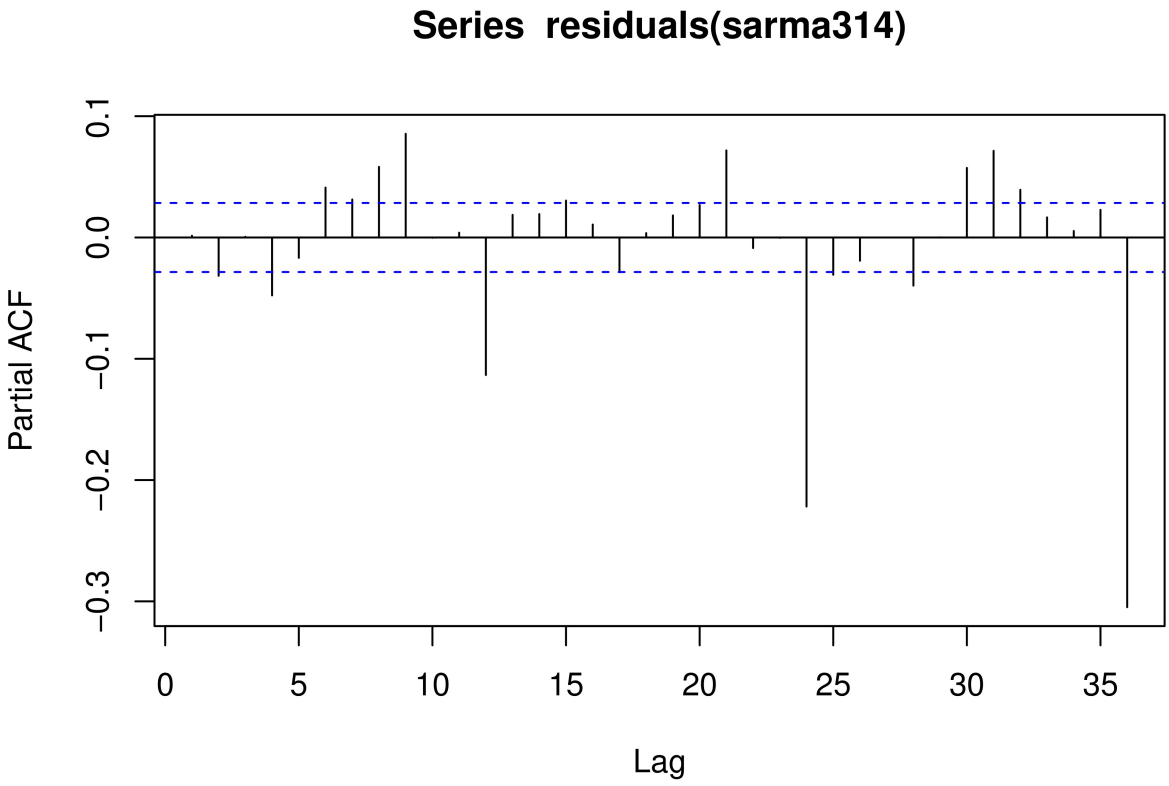
```
##########################################

## will select which has minimum BIC

bic_values <- c(BIC(sarma113), BIC(sarma112), BIC(sarma314))

# Find the minimum BIC value and its index
min_bic <- min(bic_values)
min_bic_index <- which.min(bic_values)

min_bic_index
```

```
## [1] 3
```

```
## so SARMA314 is our go to model

#########################################################
## Residuals analysis

acf(residuals(sarma314))
```




```
pacf(residuals(sarma314))
```


Series residuals(sarma314)
Partial ACF
0.1
0.0
−0.1
−0.2
−0.3
0
5
10
15
20
25
30
35
Lag