Time Series Final Project:Forecasting NIFTY Stock Market's Volume

David Lin

2023-06-09

Abstract:

In this report, I used time series analysis to predict NIFTY stock market's Volume, which measures the number of shares traded in a stock market in futures or options. At the beginning, I found out that yearly stock volume data is too noisy so that I decided to convert into monthly data set. After examining the variance and looking at the plot, it was not stationary, so I used Box-Cox transformation to reduce the variance and make the data stationary for future analysis of identification.

After that, I tried to remove linear trend by differencing at lag1 in order to get more stationary time series data. Then, I plotted ACF and PACF plots to select candidates, and I had identified three model selections, and select two of them to compare, according to AICc. After comparing those two models using different tests (like Q-Q plot, Box-Pierce test, Box-Ljung test, Shapiro-Wilk normality test, and so on), I chose the best fitted model to forecast the future Stock Volume. Finally, the forecast plot shows a reasonable prediction within the range by comparing the true values. Therefore, time series analysis can be a valuable means to predict the future Volume trend in the Stock Market.

Introduction

The top 50 businesses listed on the National Stock Exchange make up the NIFTY, an index of the Indian stock market. It gives information on general market trends and acts as a benchmark for the Indian equity market. To predict the Volume in the stock market is vital since even if a security's price is increasing, low volume can suggest that investors are not confident in it. On the other hand, huge volume accumulation of a specific security may be a sign that traders have trust in the investment over the long term. Therefore, the prediction can see the bias in the stock market which may influence the stock market price. The NIFTY dataset was my first choice since it offers useful information for undertaking a range of analyses, such as financial analysis, stock price forecasts, and operational efficiency evaluation. I select the data from 2010Q1 to 2017Q4. My study's main goal is to examine this dataset and produce predictions for future stock market Volume. The tools I will use in this report are Q-Q plot, Box-Pierce test, Box-Ljung test, Shapiro-Wilk normality test, Box-Cox transformation, ACF and PACF plots, and so on.

import necessary libraries

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

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

```
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(ggplot2)
library(dplyr)
```

```
##
## ########################## Warning from 'xts' package ##########################
## #                                                                              #
## # The dplyr lag() function breaks how base R's lag() function is supposed to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or       #
## # source() into this session won't work correctly.                            #
## #                                                                              #
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop           #
## # dplyr from breaking base R's lag() function.                                #
## #                                                                              #
## # Code in packages is not affected. It's protected by R's namespace mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.  #
## #                                                                              #
## ################################################################################
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:xts':
##
##     first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(TTR)
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(tseries)
```

Install data

```
df <- read.csv("RDatasets/NIFTY50_all.csv",header = TRUE)
```

```
##convert data types to numeric
df$Volume <- as.numeric(df$Volume)

# Convert 'datesold' column to proper date format
df$Date <- as.Date(df$Date)

# show the type of data after converting
class(df$Date)
```

```
## [1] "Date"
```

```
class(df$Volume)
```

```
## [1] "numeric"
```

```
#display summery
print("---- Summary of the dataset--------")
```

```
## [1] "---- Summary of the dataset--------"
```

```
summary(df)
```

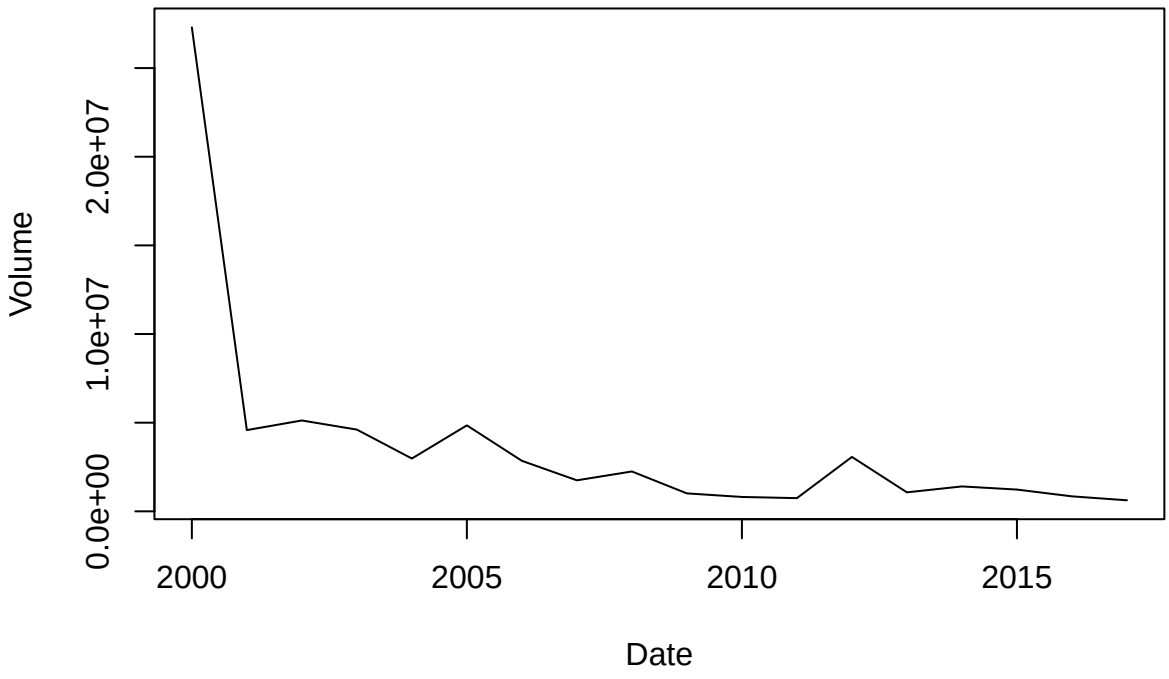
```
##       Date               Symbol             Series            Prev.Close
##  Min.   :2000-01-03   Length:235192      Length:235192      Min.   :    0.0
##  1st Qu.:2006-07-03   Class :character   Class :character   1st Qu.:  274.3
##  Median :2011-08-19   Mode  :character   Mode  :character   Median :  566.5
##  Mean   :2011-05-22                                        Mean   : 1266.2
##  3rd Qu.:2016-06-24                                        3rd Qu.: 1242.2
##  Max.   :2021-04-30                                        Max.   :32861.9
##
##       Open              High               Low               Last
##  Min.   :    8.5   Min.   :    9.75   Min.   :    8.5   Min.   :    9.1
##  1st Qu.:  275.0   1st Qu.:  279.50   1st Qu.:  269.6   1st Qu.:  274.4
##  Median :  567.0   Median :  576.90   Median :  556.5   Median :  567.0
##  Mean   : 1267.8   Mean   : 1286.58   Mean   : 1247.5   Mean   : 1266.4
##  3rd Qu.: 1243.3   3rd Qu.: 1263.00   3rd Qu.: 1221.7   3rd Qu.: 1242.9
##  Max.   :33399.9   Max.   :33480.00   Max.   :32468.1   Max.   :32849.0
##
##      Close               VWAP              Volume             Turnover
```

```
##  Min.   :    9.15   Min.   :    9.21   Min.   :        3   Min.   :1.047e+07
##  1st Qu.:  274.35   1st Qu.:  274.70   1st Qu.:   219010   1st Qu.:1.613e+13
##  Median :  566.70   Median :  566.94   Median :  1010938   Median :6.833e+13
##  Mean   : 1266.55   Mean   : 1267.13   Mean   :  3045903   Mean   :1.610e+14
##  3rd Qu.: 1242.40   3rd Qu.: 1242.66   3rd Qu.:  3019851   3rd Qu.:1.864e+14
##  Max.   :32861.95   Max.   :32975.24   Max.   :481058927   Max.   :3.564e+16
##
##      Trades        Deliverable.Volume   X.Deliverble
##  Min.   :     11   Min.   :        5   Min.   :0.024
##  1st Qu.:  21834   1st Qu.:   125383   1st Qu.:0.365
##  Median :  44068   Median :   501756   Median :0.511
##  Mean   :  61964   Mean   :  1315098   Mean   :0.503
##  3rd Qu.:  78936   3rd Qu.:  1452233   3rd Qu.:0.638
##  Max.   :1643015   Max.   :232530747   Max.   :1.000
##  NA's   :114848    NA's   :16077       NA's   :16077
```

Plot Raw Data

Lets convert the dataframe above into a time series object used a frequency of 1 since the data is noise.

```
#Converting data into a ts object with irregular frequency, extract data from Year 2000 to Year 2017.
ts_df <- ts(df$Volume,start=2000,end=2017, frequency = 1)
plot(ts_df, xlab = "Date", ylab = "Volume")
```



From above, the data plot is noise, since the interval of the data set is irregular.Thus,we can transform it to a monthly dataset.

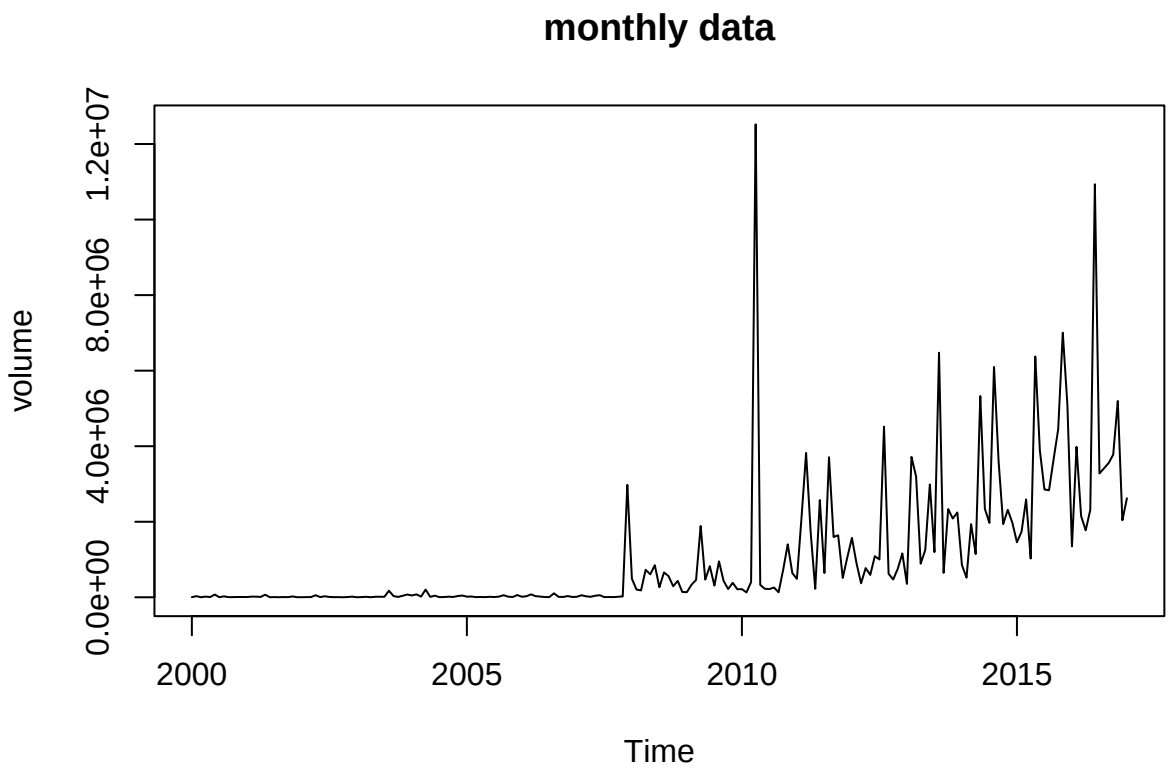
Monthly dataset

```
df_xts <- xts(df[, c("Date", "Volume")], order.by = df$Date)
df_monthly <- apply.monthly(df_xts, function(x) x[1, ])
df_monthly
```

```
##                   Date    Volume
## 2000-01-31 2000-01-03      3318
## 2000-02-29 2000-02-01     29884
## 2000-03-31 2000-03-01      1383
## 2000-04-28 2000-04-03     18297
## 2000-05-31 2000-05-02      4998
## 2000-06-30 2000-06-01     73556
## 2000-07-31 2000-07-03      1551
## 2000-08-31 2000-08-01     25371
## 2000-09-29 2000-09-04      3543
## 2000-10-31 2000-10-03      5398
##        ...
## 2020-07-31 2020-07-01   1202080
## 2020-08-31 2020-08-03   2408531
## 2020-09-30 2020-09-01   6726316
## 2020-10-30 2020-10-01   3004005
## 2020-11-27 2020-11-02   9364431
## 2020-12-31 2020-12-01   9170159
## 2021-01-29 2021-01-01   7815730
## 2021-02-26 2021-02-01  10259084
## 2021-03-31 2021-03-01  11600468
## 2021-04-30 2021-04-01  15017049
```

plot monthly Volume data

```
ts_df_vol <- ts(df_monthly$Volume,start=2000,end=2017, frequency = 12)
plot(ts_df_vol, main = "monthly data", ylab = "volume")
```



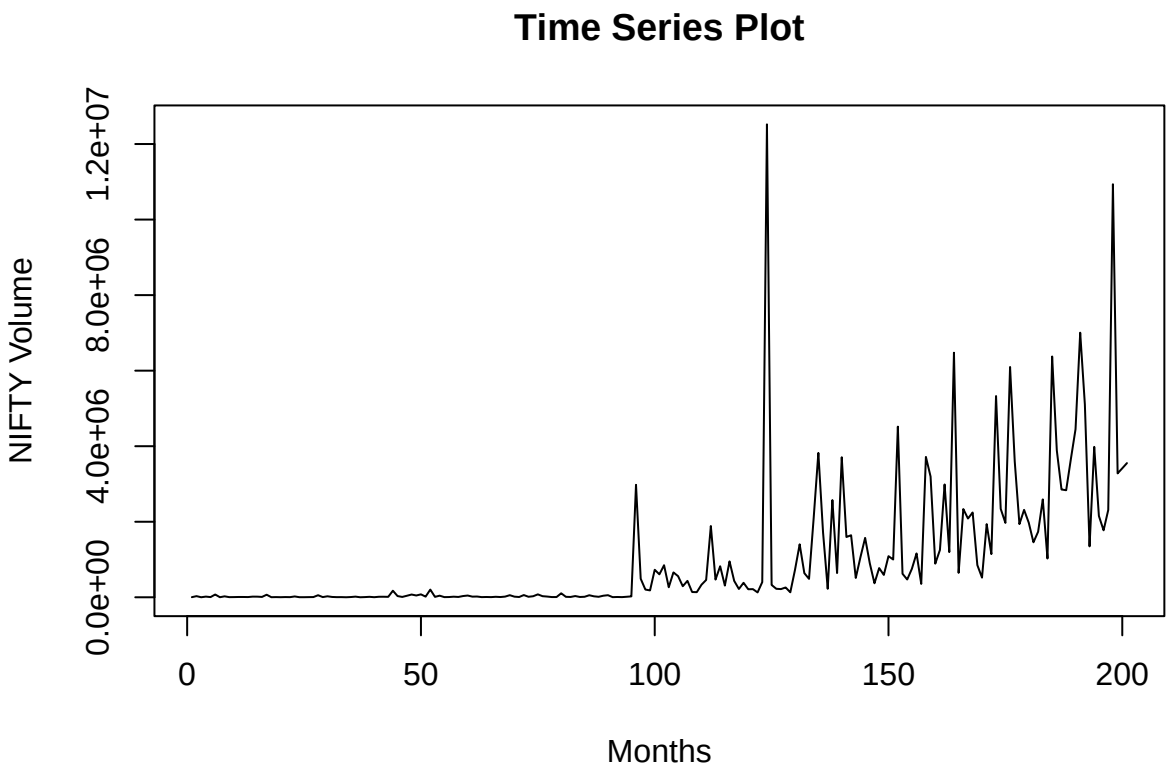

```
ts_df_vol <- as.numeric(ts_df_vol)
```

We can see that the data in two plots is sometimes in a sharply ascend and decline, which indicates that the variances of two data will change over time and may have strong seasonality.

Data division

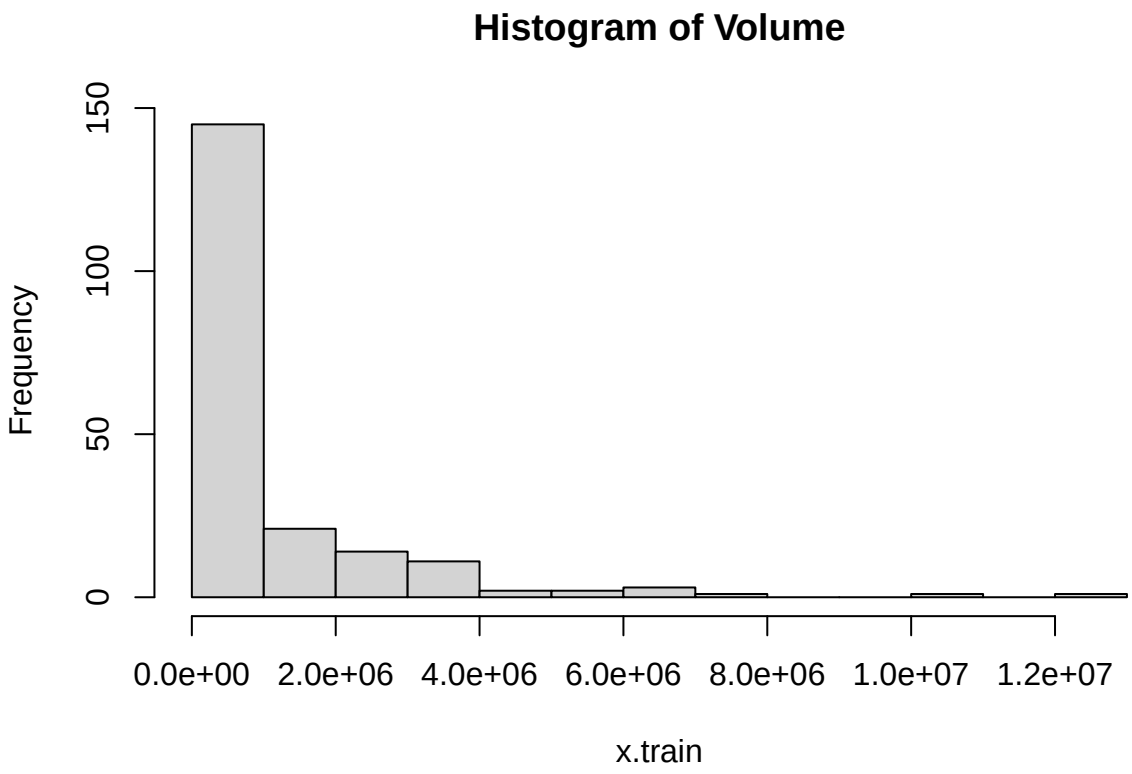
Divide transformed data into train and test data

```
n <- length(ts_df_vol)
x.train <- ts_df_vol[1:(n - 4)] #train data
x.train <- as.numeric(x.train)
x.test <- ts_df_vol[(n - 3):n] #test data
x.test <- as.numeric(x.test)
plot.ts(x.train, type = "l", main = "Time Series Plot", xlab = "Months", ylab = "NIFTY Volume")
```



checking if our model is stationary

```
hist(x.train, main="Histogram of Volume")
```



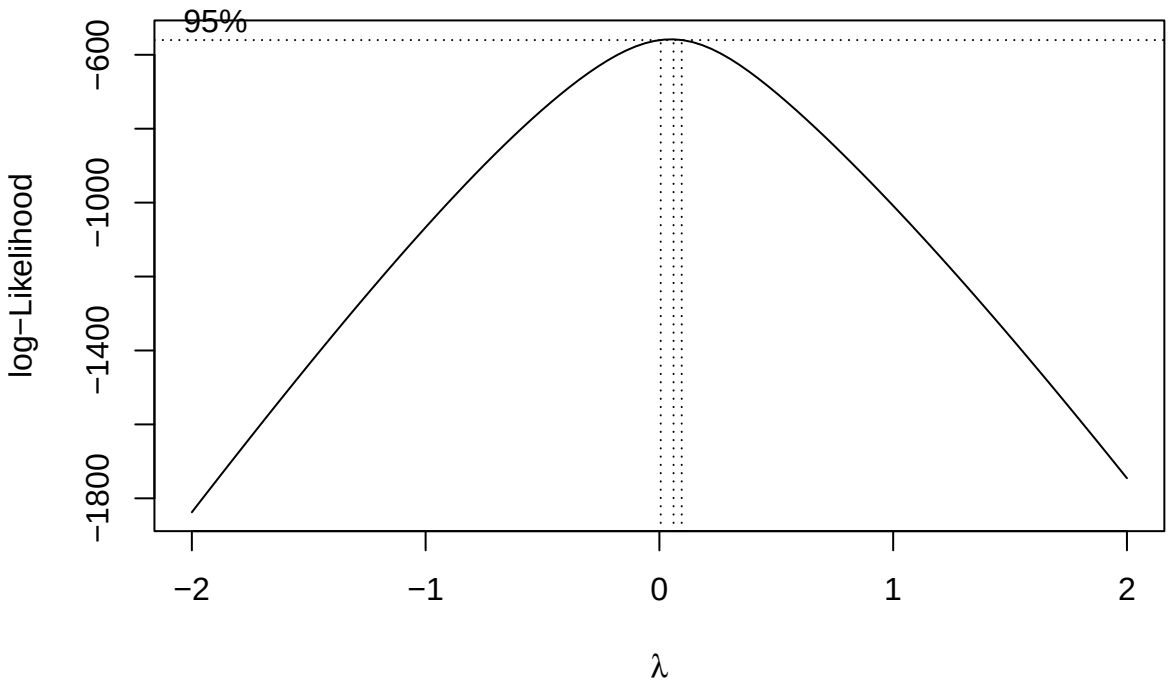
The histogram is right-skewed so it is not a normal distribution. Therefore I have to do transformation to stabilize the variance and differencing at lag1 to remove linear trend.

Transformations

```
##Steps to make the ts stationary
model <- lm(x.train ~ time(x.train))
fitted_values <- predict(model)
stationary_ts <- x.train - fitted_values

## lets stabilize the variance by using log
log.ts_monthly <- log(x.train)
sqrt.ts_monthly <- sqrt(x.train)

##lets fit lenear regression model
t = 1:length(x.train)
fit = lm(x.train ~ t)
## applying Box-Cox Transformation
bc_transform = boxcox(x.train ~ t,plotit = TRUE)
```



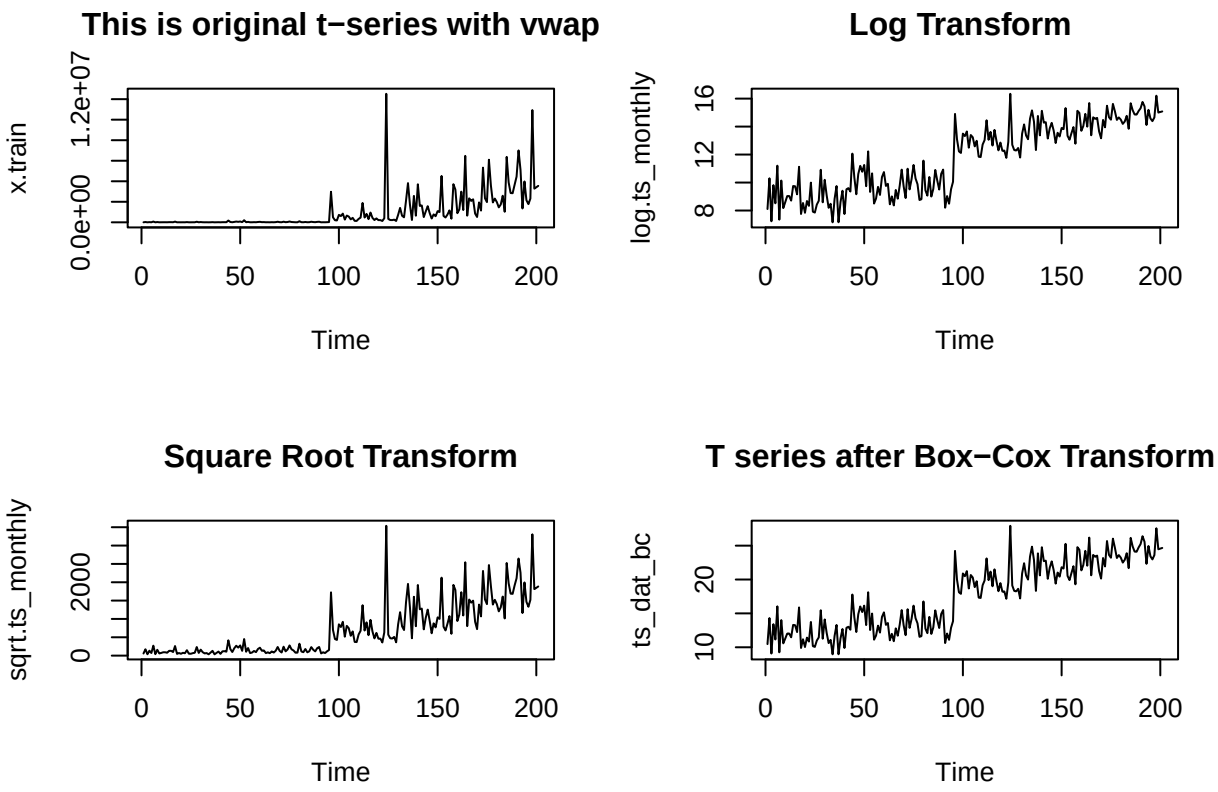

```
##performing Power Transformation, Based on the Box-Cox transformation
lambda = bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
```

```
## [1] 0.06060606
```

```
ts_dat_bc = (1/lambda)*(x.train^lambda-1)
```

Now we plot the original data vs transformed data:

```
op= par(mfrow=c(2,2))
plot.ts(x.train, main = "This is original t-series with vwap")
plot.ts(log.ts_monthly, main = "Log Transform ")
plot.ts(sqrt.ts_monthly, main = "Square Root Transform")
plot.ts(ts_dat_bc, main = "T series after Box-Cox Transform ")
```



Then, we can compare the variance to see the aftermath of transformation:

```
# show the variance
var(x.train)
```

```
## [1] 3.176555e+12
```

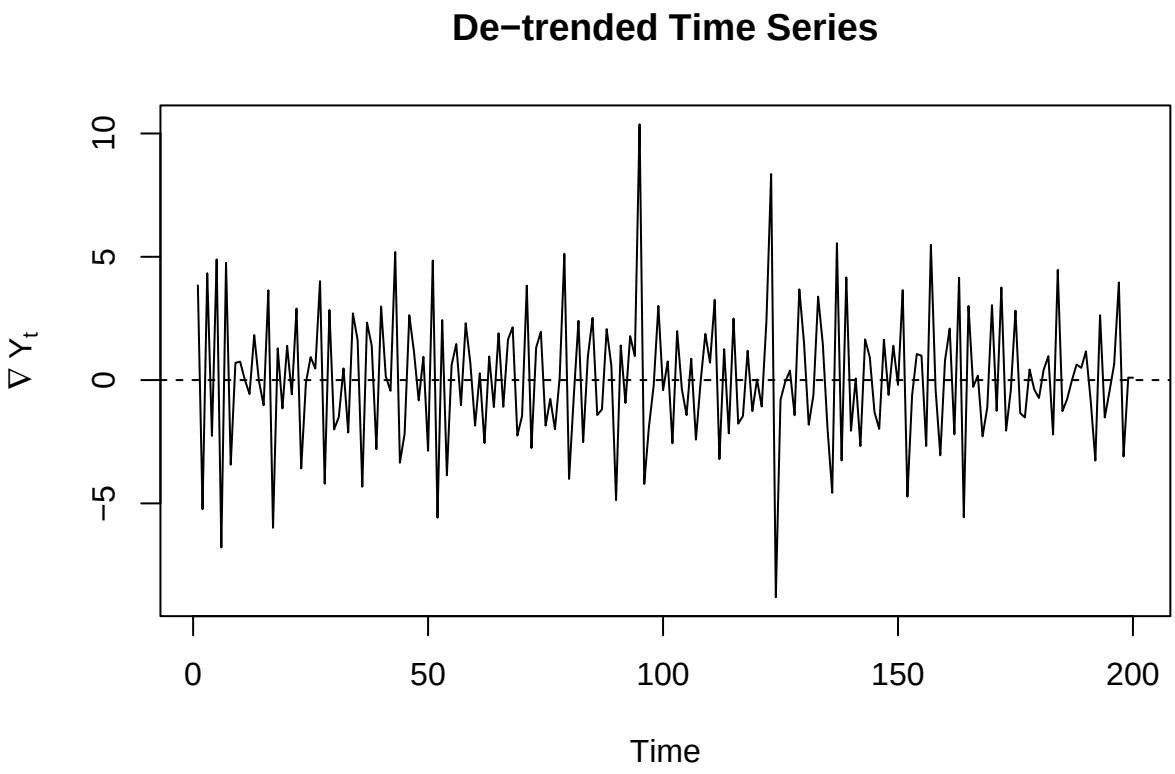
```
var(ts_dat_bc)
```

```
## [1] 26.02639
```

Decomposing the results

Then, we use differencing at lag1 to remove linear trend

```
# Difference at lag  = 1 to remove trend component
differenced_ts1 <- diff(ts_dat_bc,lag=1)
ts.plot(differenced_ts1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
```




```
var(differenced_ts1)
```

```
## [1] 7.294928
```

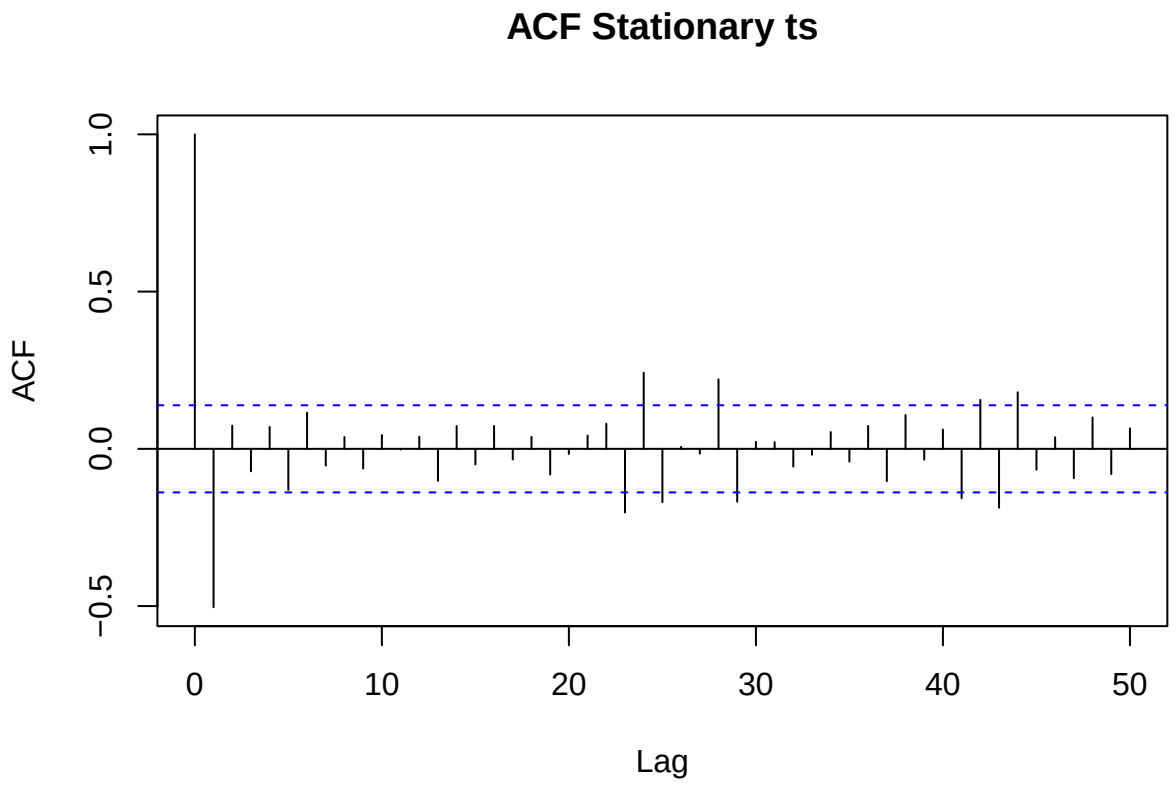
```
adf.test(differenced_ts1, alternative = "stationary")
```

```
## Warning in adf.test(differenced_ts1, alternative = "stationary"): p-value
## smaller than printed p-value
```

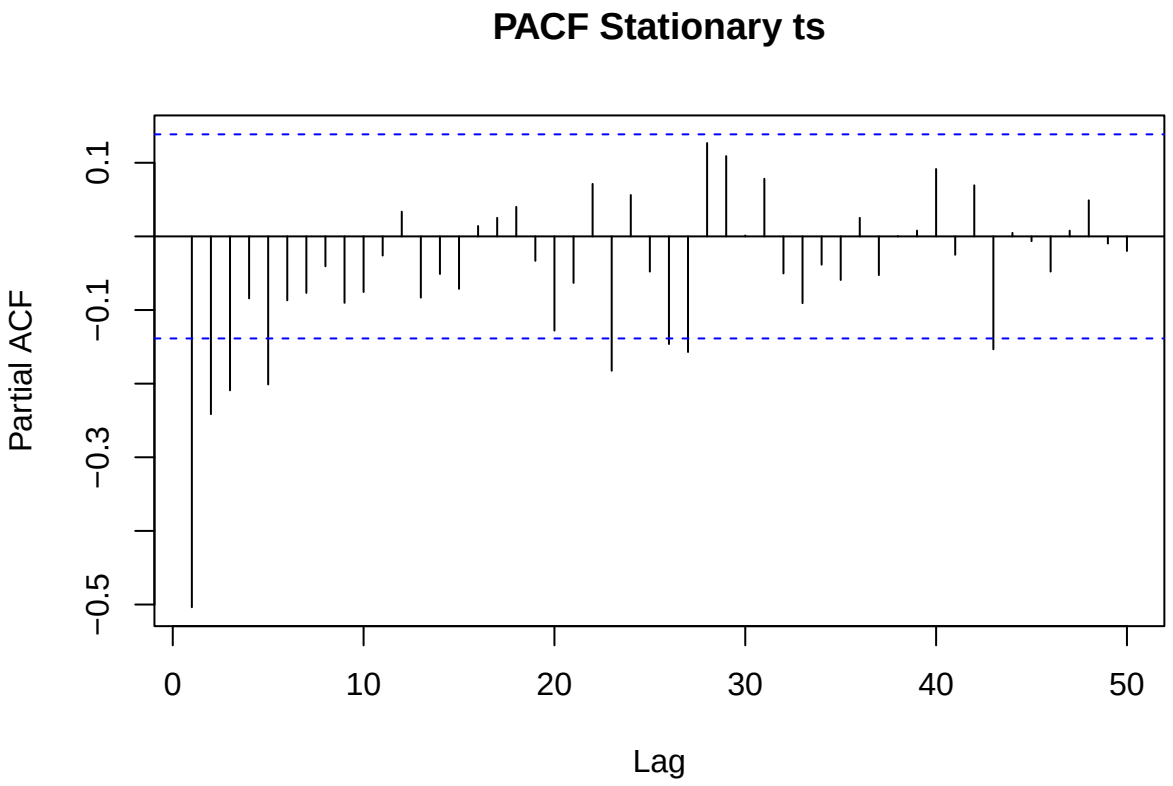
```
##
##  Augmented Dickey-Fuller Test
##
## data:  differenced_ts1
## Dickey-Fuller = -8.7292, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

Model Identification

```
acf(differenced_ts1, main="ACF Stationary ts", lag.max = 50)
```




```
pacf(differenced_ts1,main="PACF Stationary ts", lag.max = 50)
```



Based on the PACF plot, there is strong peak at h=0 and smaller peaks at h=2,3,5, so may consider p=3,4,5. Based on ACF plot, it shows strong peak at h=0 or 1, so may consider q=1. We choose d = 1,since we difference at lag1. Therefore, we can see that the model may be ARIMA(5,1,1), ARIMA(3,1,1), and ARIMA(4,1,1) as candidates.

```
#Building ARIMA models
fit1 <- arima(ts_dat_bc, order = c(5,1,1), method = "ML") #model1
fit2 <- arima(ts_dat_bc, order = c(3,1,1), method = "ML") #model2
fit3 <- arima(ts_dat_bc, order = c(4,1,1), method = "ML") #model3

#Checking for AICc
library(qpcR)
```

```
## Loading required package: minpack.lm

## Loading required package: rgl

## Loading required package: robustbase

## Loading required package: Matrix
```

```
AICc(fit1)
```

```
## [1] 887.3378
```

```
AICc(fit2)
```

```
## [1] 885.2076
```

```
AICc(fit3)
```

```
## [1] 887.2334
```

By building ARIMA model and checking AICc of the models, we select ARIMA(3,1,1) and ARIMA(4,1,1) since they have smaller AICc.

Estimate Model Parameter

First, we do estimation for model2

```
fit = arima(ts_dat_bc, order = c(3,1,1), method="ML")
fit
```

```
##
## Call:
## arima(x = ts_dat_bc, order = c(3, 1, 1), method = "ML")
##
## Coefficients:
##          ar1     ar2      ar3      ma1
##       0.0576  0.0978  -0.0234  -0.8016
## s.e.  0.1092  0.0961   0.0852   0.0845
##
## sigma^2 estimated as 4.632:  log likelihood = -437.5,  aic = 885
```

Then, we do estimation for model3

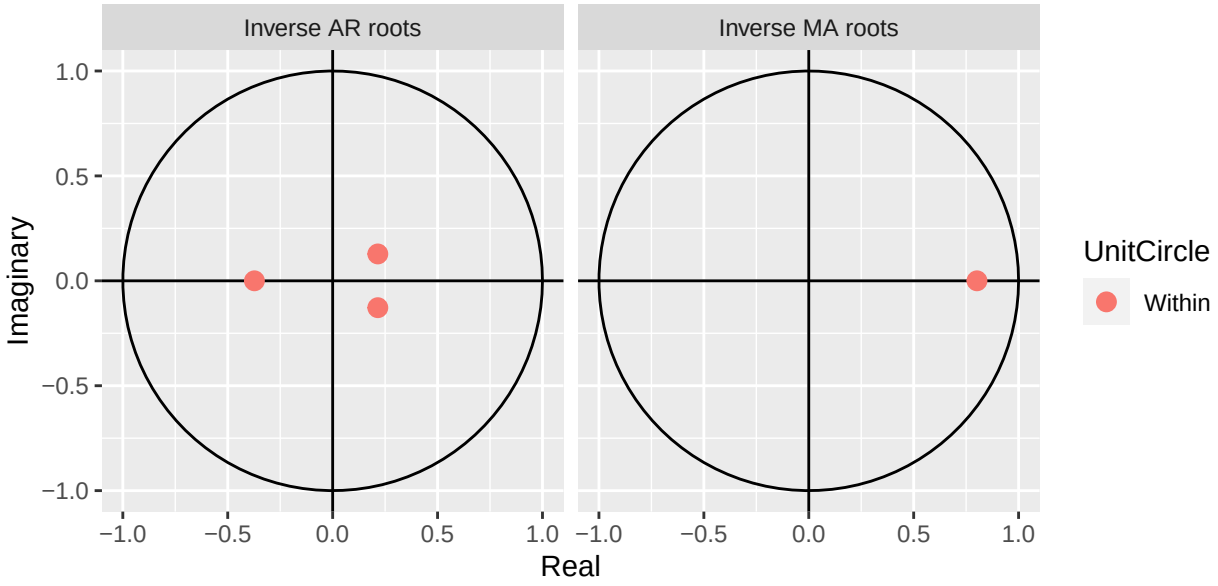
```
fit2 = arima(ts_dat_bc, order = c(4,1,1), method="ML")
fit2
```

```
##
## Call:
## arima(x = ts_dat_bc, order = c(4, 1, 1), method = "ML")
##
## Coefficients:
##          ar1     ar2      ar3     ar4      ma1
##       0.0724  0.1069  -0.0171  0.0236  -0.8161
## s.e.  0.1174  0.0990   0.0870  0.0847   0.0932
##
## sigma^2 estimated as 4.63:  log likelihood = -437.46,  aic = 886.93
```

Model Diagnostics of model2

Unit Root Test

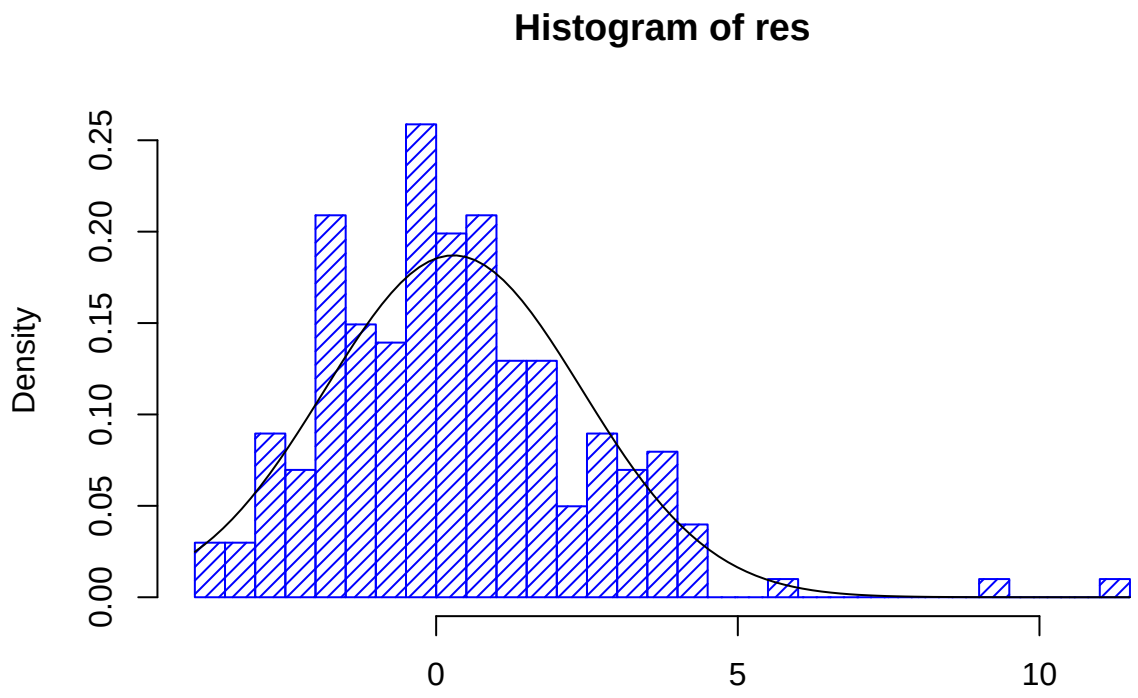
```
#check invertibility of MA part of model model4:
autoplot(fit, type = c("both", "ar", "ma"))
```



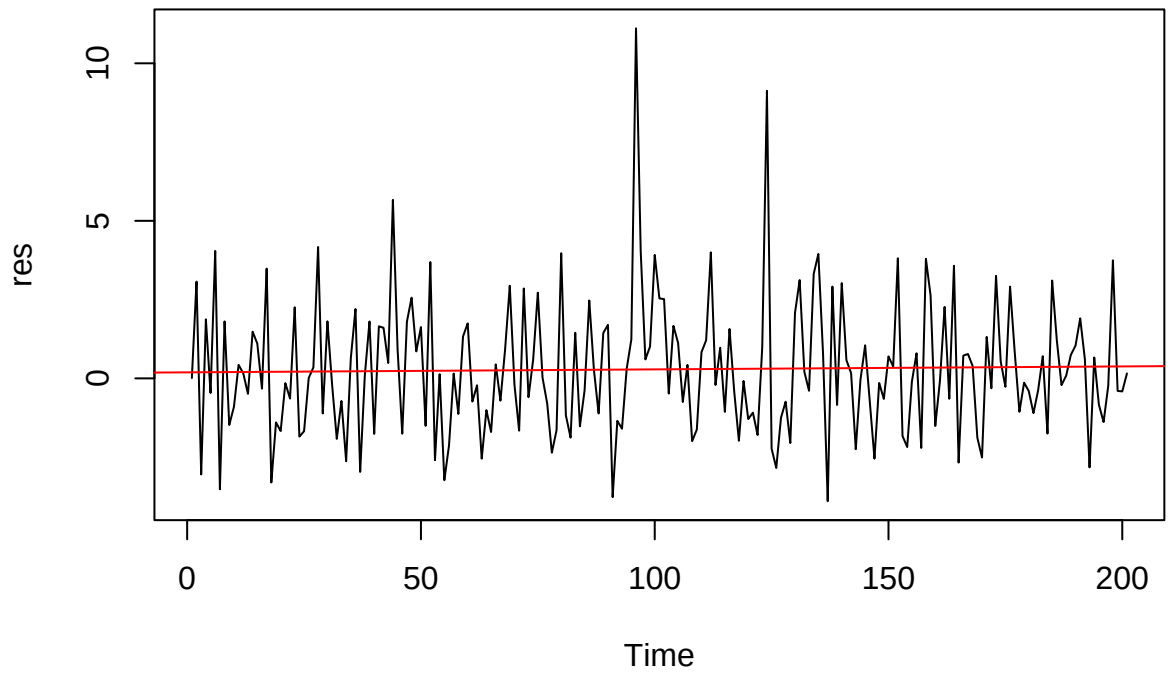

```
library(forecast)
res <- residuals(fit)
hist(res,density=20, breaks = 30, col="blue", xlab="", prob=TRUE)
m <- mean(res)
m
```

```
## [1] 0.2861266
```

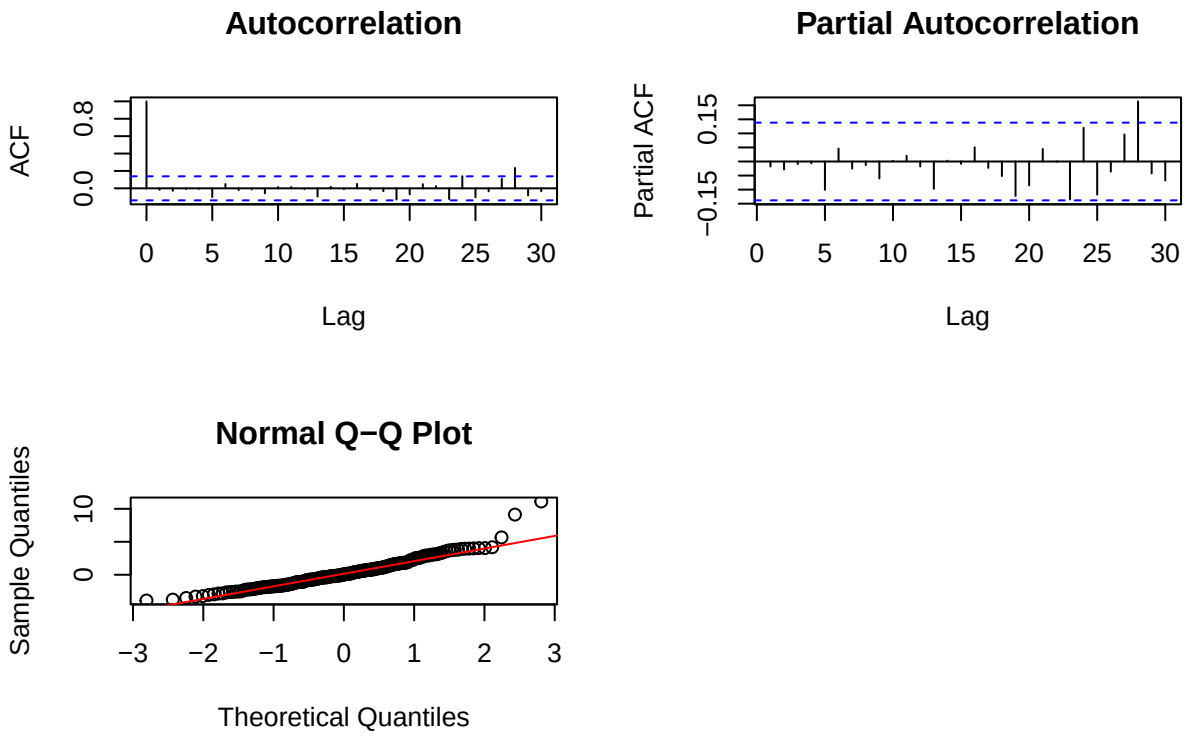
```
std <- sqrt(var(res))
curve(dnorm(x,m,std), add=TRUE)
```




```
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res)))
abline(fitt, col="red")
```




```
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit), lag.max=30, main = "Autocorrelation")
# pacf
pacf(residuals(fit), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit))
qqline(residuals(fit),col ="red")
# Add overall title
title("Model4 Fitted Residuals Diagnostics", outer=TRUE)
```




```
# Box test
Box.test(res, lag = 12, type = c("Box-Pierce"), fitdf = 4)
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 3.6363, df = 8, p-value = 0.8884
```

```
Box.test(res, lag = 12, type = c("Ljung-Box"), fitdf = 4)
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 3.7839, df = 8, p-value = 0.8761
```

```
Box.test(res^2, lag = 12, type = c("Ljung-Box"), fitdf = 4)
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 2.8629, df = 8, p-value = 0.9427
```

```r
# Test for normality of residuals
shapiro.test(residuals(fit))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit)
## W = 0.94188, p-value = 3.161e-07
```

```r
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

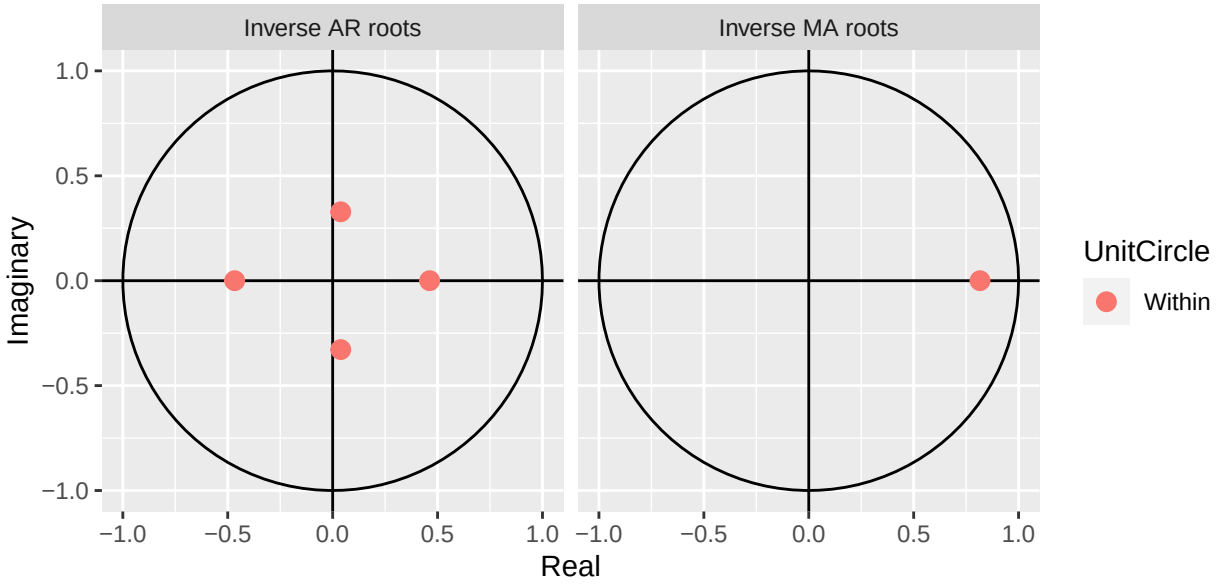
```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  4.549
```

Residuals is white noise.

Model Diagnostics of model3

Unit Root Test

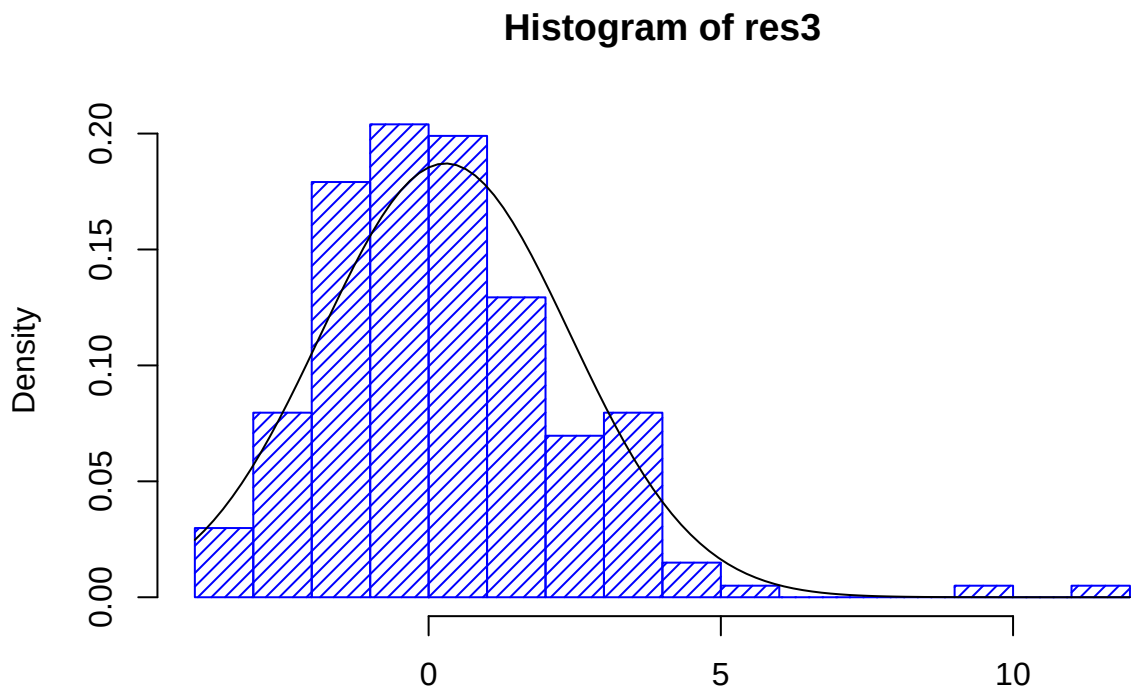
```r
#check invertibility of MA part of model model1:
autoplot(fit3, type = c("both", "ar", "ma"))
```



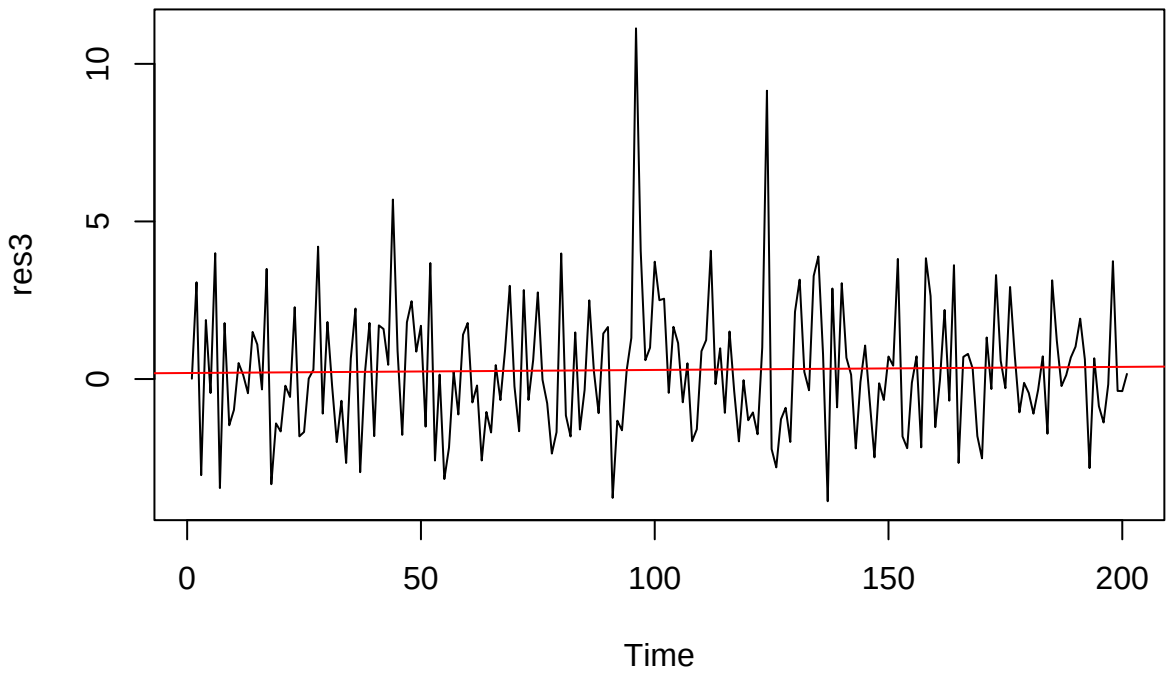

```
library(forecast)
res3 <- residuals(fit3)
hist(res3,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res3)
m
```

```
## [1] 0.2893858
```

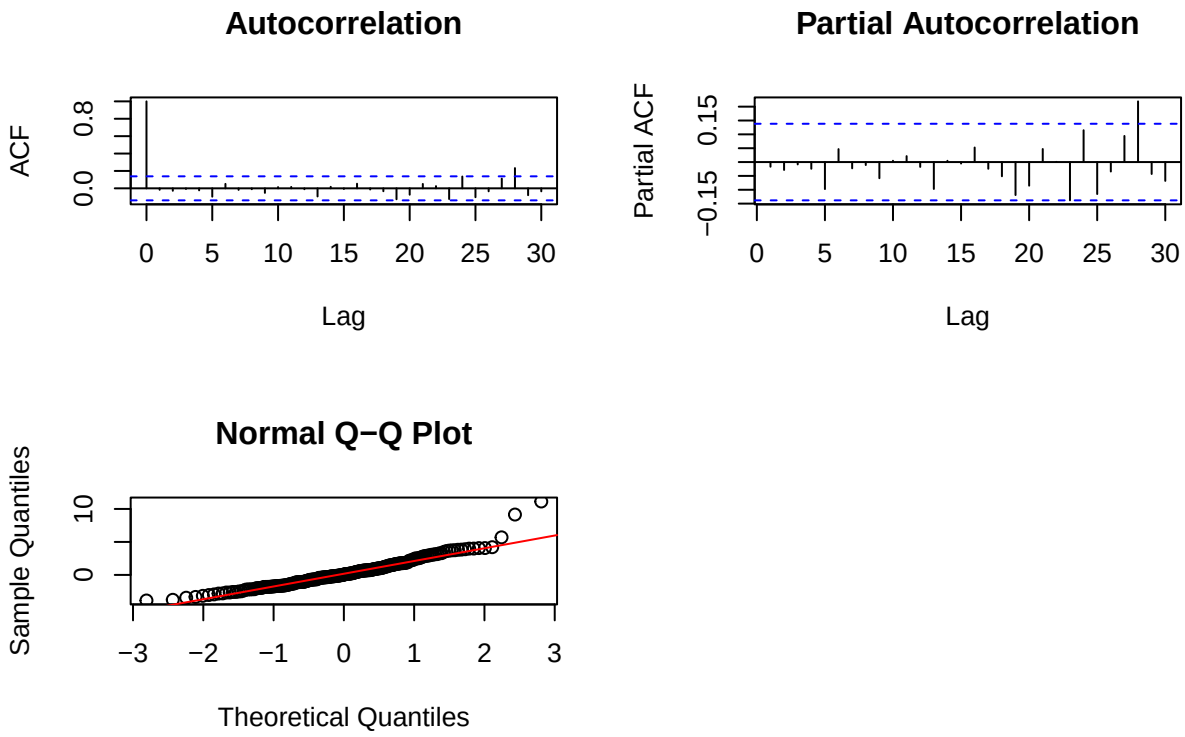
```
std <- sqrt(var(res3))
curve(dnorm(x,m,std), add=TRUE)
```




```
plot.ts(res3)
fitt <- lm(res3 ~ as.numeric(1:length(res3)))
abline(fitt, col="red")
```




```
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit3), lag.max=30, main = "Autocorrelation")
# pacf
pacf(residuals(fit3), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit3))
qqline(residuals(fit3),col ="red")
# Add overall title
title("Model 3 Fitted Residuals Diagnostics", outer=TRUE)
```




```
# Box test
Box.test(res3, lag = 12, type = c("Box-Pierce"), fitdf = 5)
```

```
##
##  Box-Pierce test
##
## data:  res3
## X-squared = 3.468, df = 7, p-value = 0.8386
```

```
Box.test(res3, lag = 12, type = c("Ljung-Box"), fitdf = 5)
```

```
##
##  Box-Ljung test
##
## data:  res3
## X-squared = 3.6068, df = 7, p-value = 0.8238
```

```
Box.test(res3^2, lag = 12, type = c("Ljung-Box"), fitdf = 5)
```

```
##
##  Box-Ljung test
##
## data:  res3^2
## X-squared = 2.8797, df = 7, p-value = 0.8959
```

```r
# Test for normality of residuals
shapiro.test(residuals(fit3))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit3)
## W = 0.94128, p-value = 2.798e-07
```

```r
ar(res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  4.546
```

The residuals is white noise.

Diagonostic conclusion: Both models pass all tests but Shapiro-Wilk normality test. Based on principle of parsimony, we choose the simplest model with less parameters. Therefore, choose model2 as the Final Model. Also, conclude from residuals, my model is satisfactory.

Forecast the model

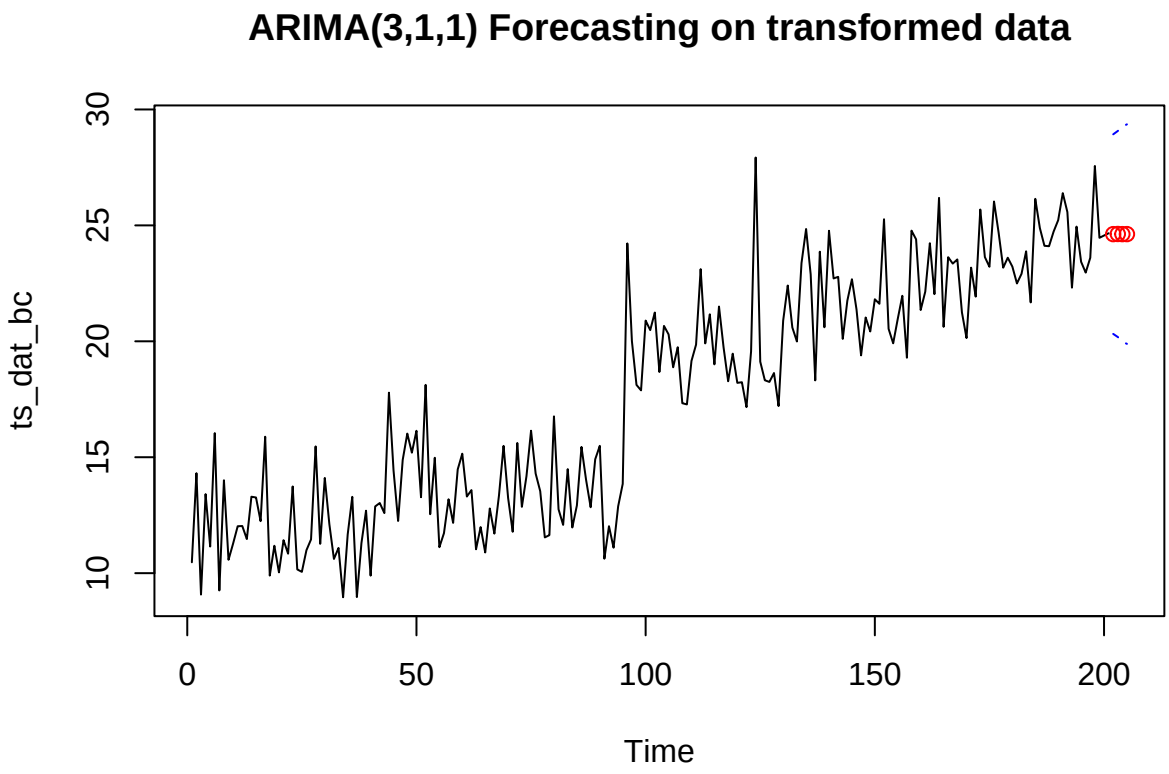
To produce graph with 4 forecasts on transformed data

```r
library(forecast)
library(astsa)
```

```
##
## Attaching package: 'astsa'
```

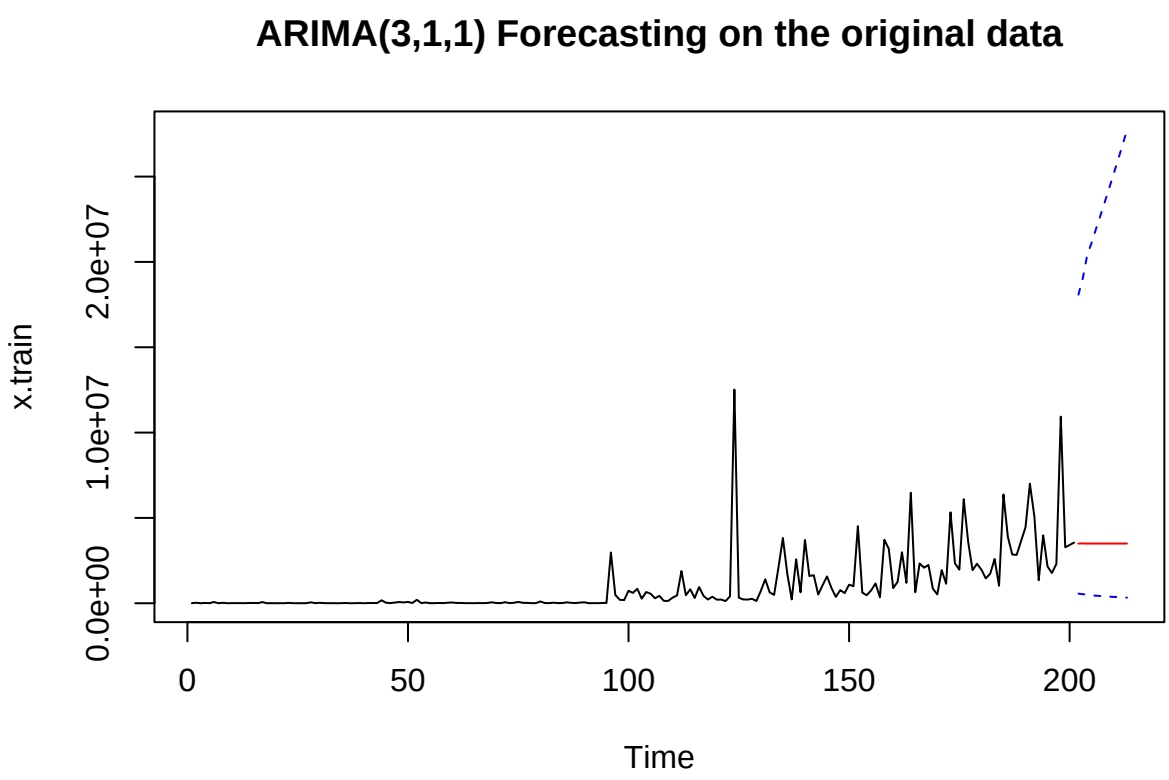
```
## The following object is masked from 'package:forecast':
##
##     gas
```

```r
fitA <- arima(ts_dat_bc, order=c(3,1,1), method="ML")
# To produce graph with 8 forecasts on transformed data:
pred.tr <- predict(fitA, n.ahead = 4)
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(ts_dat_bc, xlim=c(1,length(ts_dat_bc)+4), ylim = c(min(ts_dat_bc),max(U.tr)), main = "ARIMA(3,1
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(ts_dat_bc)+1):(length(ts_dat_bc)+4), pred.tr$pred, col="red")
```

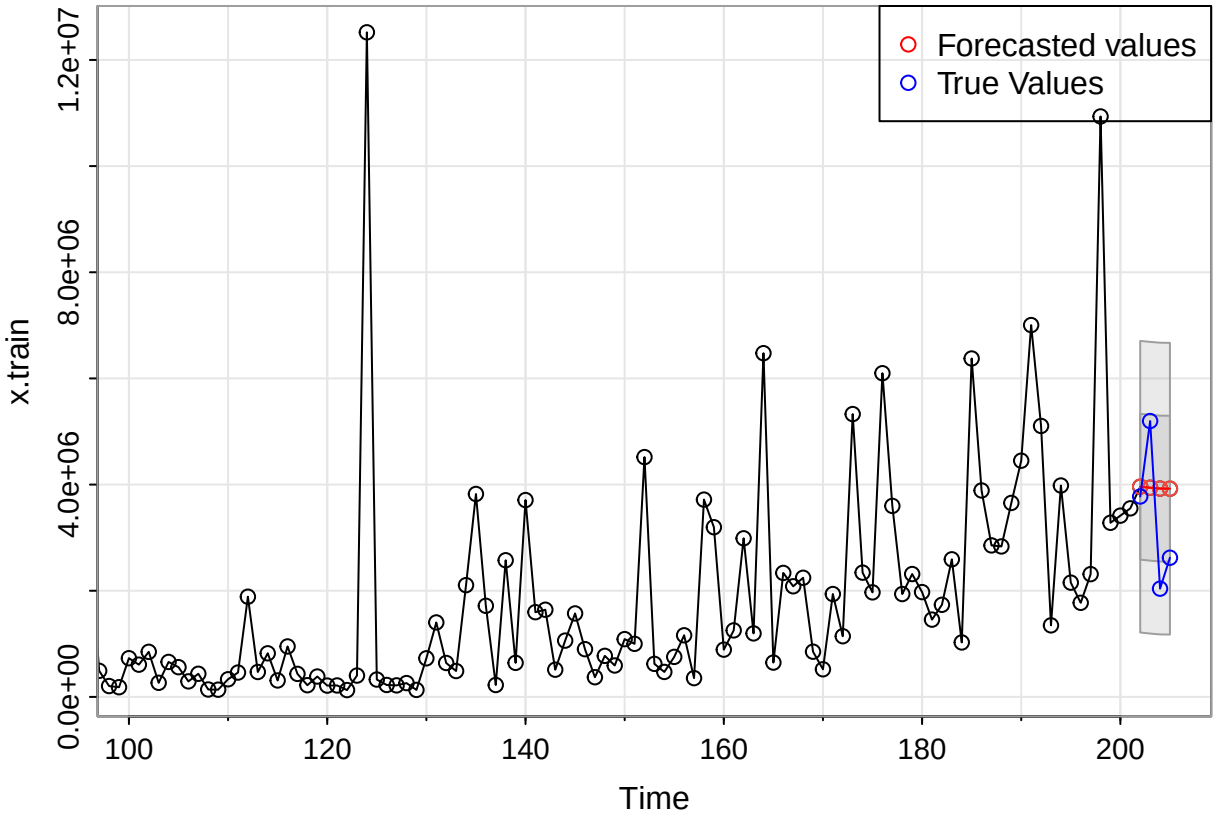


Forecasting on the original data

```
pred <- predict(fitA, n.ahead = 12)
origin <- (pred$pred*lambda+1)^(1/lambda)
U.tr<-pred$pred + 2*pred$se
L.tr<- pred$pred - 2*pred$se
U<-(U.tr*lambda+1)^(1/lambda)
L<- (L.tr*lambda+1)^(1/lambda)
ts.plot(x.train, xlim=c(1,length(x.train)+12), ylim = c(min(ts_df_vol),max(U)), main = "ARIMA(3,1,1) Fo
lines(origin, col = 'red')
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
```




```
par(mfrow=c(1, 1))
pred.try <- sarima.for(x.train, n.ahead=4, plot.all=F, p=3, d=1, q=1, D=0,P=0,Q=0)
lines((n - 3):n, pred.try$pred, col="red")
lines((n - 3):n, x.test, col="blue")
points((n - 3):n, x.test, col="blue")
legend("topright", pch=1, col=c("red", "blue"),  legend=c("Forecasted values", "True Values"))
```



Conclusion

In conclusion, the model we used to predict the stock market's Volume has passed all tests except Shapiro-Wilk normality test. Also, the forecasting plot shows that the predicted values aligned with the tests values within the confidence interval. Therefore, within a confidence interval, the forecasting model successfully predicts the trend of stock market's volume in 4 months. However, our forecasting model still needs more accuracy as we can see the strainght-line prediction on plots. In addition, the fitted model in algebraic form is:

$$X_t = 0.0576X_{t-1} + 0.0978X_{t-2} - 0.0234X_{t-3} - 0.8016Z_{t-1} + W_n$$

Finally, I am writing this acknowledgement to acknowledge the receipt of support and guaidance in my final project. I would like to give my gratitude to our brilliant Professor Feldman and TA, AUNG T. Their feedback are instructive, and I appreciate their promptness in dealing with data selection and model identifications. It will allow me to progress efficiently and effectively.

Reference

https://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html https://www.kaggle.com/code/paytonfisher/s-p-500-analysis-using-r https://gauchospace.ucsb.edu/courses/pluginfile.php/11850413/mod_resource/content/1/Lecture%2015-AirPass%20slides.pdf labs from https://gauchospace.ucsb.edu/courses/course/view.php?id=53366

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(forecast)
library(xts)
library(ggplot2)
library(dplyr)
library(TTR)
library(MASS)
library(tseries)
df <- read.csv("RDatasets/NIFTY50_all.csv",header = TRUE)
##convert data types to numeric
df$Volume <- as.numeric(df$Volume)

# Convert 'datesold' column to proper date format
df$Date <- as.Date(df$Date)

# show the type of data after converting
class(df$Date)
class(df$Volume)

#display summery
print("---- Summary of the dataset--------")
summary(df)
#Converting data into a ts object with irregular frequency, extract data from Year 2000 to Year 2017.
ts_df <- ts(df$Volume,start=2000,end=2017, frequency = 1)
plot(ts_df, xlab = "Date", ylab = "Volume")
df_xts <- xts(df[, c("Date", "Volume")], order.by = df$Date)
df_monthly <- apply.monthly(df_xts, function(x) x[1, ])
df_monthly
ts_df_vol <- ts(df_monthly$Volume,start=2000,end=2017, frequency = 12)
plot(ts_df_vol, main = "monthly data", ylab = "volume")
ts_df_vol <- as.numeric(ts_df_vol)
n <- length(ts_df_vol)
x.train <- ts_df_vol[1:(n - 4)] #train data
x.train <- as.numeric(x.train)
x.test <- ts_df_vol[(n - 3):n] #test data
x.test <- as.numeric(x.test)
plot.ts(x.train, type = "l", main = "Time Series Plot", xlab = "Months", ylab = "NIFTY Volume")
hist(x.train, main="Histogram of Volume")
##Steps to make the ts stationary
model <- lm(x.train ~ time(x.train))
fitted_values <- predict(model)
stationary_ts <- x.train - fitted_values

## lets stabilize the variance by using log
log.ts_monthly <- log(x.train)
sqrt.ts_monthly <- sqrt(x.train)

##lets fit lenear regression model
t = 1:length(x.train)
fit = lm(x.train ~ t)
```

```
## applying Box-Cox Transformation
bc_transform = boxcox(x.train ~ t,plotit = TRUE)
##performing Power Transformation, Based on the Box-Cox transformation
lambda = bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
ts_dat_bc = (1/lambda)*(x.train^lambda-1)
op= par(mfrow=c(2,2))
plot.ts(x.train, main = "This is original t-series with vwap")
plot.ts(log.ts_monthly, main = "Log Transform ")
plot.ts(sqrt.ts_monthly, main = "Square Root Transform")
plot.ts(ts_dat_bc, main = "T series after Box-Cox Transform ")
# show the variance
var(x.train)
var(ts_dat_bc)
# Difference at lag  = 1 to remove trend component
differenced_ts1 <- diff(ts_dat_bc,lag=1)
ts.plot(differenced_ts1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
var(differenced_ts1)
adf.test(differenced_ts1, alternative = "stationary")
acf(differenced_ts1, main="ACF Stationary ts", lag.max = 50)
pacf(differenced_ts1,main="PACF Stationary ts", lag.max = 50)
#Building ARIMA models
fit1 <- arima(ts_dat_bc, order = c(5,1,1), method = "ML") #model1
fit2 <- arima(ts_dat_bc, order = c(3,1,1), method = "ML") #model2
fit3 <- arima(ts_dat_bc, order = c(4,1,1), method = "ML") #model3

#Checking for AICc
library(qpcR)
AICc(fit1)
AICc(fit2)
AICc(fit3)
fit = arima(ts_dat_bc, order = c(3,1,1), method="ML")
fit
fit2 = arima(ts_dat_bc, order = c(4,1,1), method="ML")
fit2
#check invertibility of MA part of model model4:
autoplot(fit, type = c("both", "ar", "ma"))
library(forecast)
res <- residuals(fit)
hist(res,density=20, breaks = 30, col="blue", xlab="", prob=TRUE)
m <- mean(res)
m
std <- sqrt(var(res))
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res)))
abline(fitt, col="red")
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit), lag.max=30, main = "Autocorrelation")
# pacf
```

```
pacf(residuals(fit), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit))
qqline(residuals(fit),col ="red")
# Add overall title
title("Model4 Fitted Residuals Diagnostics", outer=TRUE)
# Box test
Box.test(res, lag = 12, type = c("Box-Pierce"), fitdf = 4)
Box.test(res, lag = 12, type = c("Ljung-Box"), fitdf = 4)
Box.test(res^2, lag = 12, type = c("Ljung-Box"), fitdf = 4)
# Test for normality of residuals
shapiro.test(residuals(fit))
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
#check invertibility of MA part of model model1:
autoplot(fit3, type = c("both", "ar", "ma"))
library(forecast)
res3 <- residuals(fit3)
hist(res3,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res3)
m
std <- sqrt(var(res3))
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res3)
fitt <- lm(res3 ~ as.numeric(1:length(res3)))
abline(fitt, col="red")
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit3), lag.max=30, main = "Autocorrelation")
# pacf
pacf(residuals(fit3), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit3))
qqline(residuals(fit3),col ="red")
# Add overall title
title("Model 3 Fitted Residuals Diagnostics", outer=TRUE)
# Box test
Box.test(res3, lag = 12, type = c("Box-Pierce"), fitdf = 5)
Box.test(res3, lag = 12, type = c("Ljung-Box"), fitdf = 5)
Box.test(res3^2, lag = 12, type = c("Ljung-Box"), fitdf = 5)
# Test for normality of residuals
shapiro.test(residuals(fit3))
ar(res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))
library(forecast)
library(astsa)
fitA <- arima(ts_dat_bc, order=c(3,1,1), method="ML")
# To produce graph with 8 forecasts on transformed data:
pred.tr <- predict(fitA, n.ahead = 4)
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(ts_dat_bc, xlim=c(1,length(ts_dat_bc)+4), ylim = c(min(ts_dat_bc),max(U.tr)), main = "ARIMA(3,1
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(ts_dat_bc)+1):(length(ts_dat_bc)+4), pred.tr$pred, col="red")
pred <- predict(fitA, n.ahead = 12)
```

```
origin <- (pred$pred*lambda+1)^(1/lambda)
U.tr<-pred$pred + 2*pred$se
L.tr<- pred$pred - 2*pred$se
U<-(U.tr*lambda+1)^(1/lambda)
L<- (L.tr*lambda+1)^(1/lambda)
ts.plot(x.train, xlim=c(1,length(x.train)+12), ylim = c(min(ts_df_vol),max(U)), main = "ARIMA(3,1,1) Fo
lines(origin, col = 'red')
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
par(mfrow=c(1, 1))
pred.try <- sarima.for(x.train, n.ahead=4, plot.all=F, p=3, d=1, q=1, D=0,P=0,Q=0)
lines((n - 3):n, pred.try$pred, col="red")
lines((n - 3):n, x.test, col="blue")
points((n - 3):n, x.test, col="blue")
legend("topright", pch=1, col=c("red", "blue"),  legend=c("Forecasted values", "True Values"))
library(ggplot2)
library(forecast)
library(xts)
library(ggplot2)
library(dplyr)
library(TTR)
library(MASS)
library(tseries)
df <- read.csv("RDatasets/NIFTY50_all.csv",header = TRUE)
##convert data types to numeric
df$Volume <- as.numeric(df$Volume)

# Convert 'datesold' column to proper date format
df$Date <- as.Date(df$Date)

# show the type of data after converting
class(df$Date)
class(df$Volume)

#display summery
print("---- Summary of the dataset--------")
summary(df)
#Converting data into a ts object with irregular frequency, extract data from Year 2000 to Year 2017.
ts_df <- ts(df$Volume,start=2000,end=2017, frequency = 1)
plot(ts_df, xlab = "Date", ylab = "Volume")
# Create an xts object with the data
df_xts <- xts(df[, c("Date", "Volume")], order.by = df$Date)
# Aggregate to monthly intervals using the first observation in each month
df_monthly <- apply.monthly(df_xts, function(x) x[1, ])
ts_df_vol <- ts(df_monthly$Volume,start=2000,end=2017, frequency = 12)
plot(ts_df_vol, main = "monthly data", ylab = "volume")
n <- length(ts_df_vol)
x.train <- ts_df_vol[1:(n - 4)] #train data
x.train <- as.numeric(x.train)
x.test <- ts_df_vol[(n - 3):n] #test data
x.test <- as.numeric(x.test)
plot.ts(x.train, type = "l", main = "Differenced Time Series Plot", xlab = "Months", ylab = "NIFTY Volu
hist(x.train, main="Histogram of Volume")
```

```r
acf(x.train, main = "ACF of x.train")
##Steps to make the ts stationary
model <- lm(x.train ~ time(x.train))
fitted_values <- predict(model)
stationary_ts <- x.train - fitted_values

## lets stabilize the variance by using log
log.ts_monthly <- log(x.train)
sqrt.ts_monthly <- sqrt(x.train)

##lets fit lenear regression model
t = 1:length(x.train)
fit = lm(x.train ~ t)
## applying Box-Cox Transformation
bc_transform = boxcox(x.train ~ t,plotit = TRUE)
##performing Power Transformation, Based on the Box-Cox transformation
lambda = bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
bc_transform$x[which(bc_transform$y == max(bc_transform$y))]
ts_dat_bc = (1/lambda)*(x.train^lambda-1)
op= par(mfrow=c(2,2))
plot.ts(x.train, main = "This is original t-series with vwap")
plot.ts(log.ts_monthly, main = "Log Transform ")
plot.ts(sqrt.ts_monthly, main = "Square Root Transform")
plot.ts(ts_dat_bc, main = "T series after Box-Cox Transform ")
# show the variance
var(x.train)
var(ts_dat_bc)
# Difference at lag  = 1 to remove trend component
differenced_ts1 <- diff(ts_dat_bc,lag=1)
ts.plot(differenced_ts1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
var(differenced_ts1)
adf.test(differenced_ts1, alternative = "stationary")
acf(differenced_ts1, main="ACF Stationary ts", lag.max = 50)
pacf(differenced_ts1,main="PACF Stationary ts", lag.max = 50)
#Building ARIMA models
fit1 <- arima(ts_dat_bc, order = c(5,1,1), method = "ML") #model1
fit2 <- arima(ts_dat_bc, order = c(3,1,1), method = "ML") #model2
fit3 <- arima(ts_dat_bc, order = c(4,1,1), method = "ML") #model3

#Checking for AICc
library(qpcR)
AICc(fit1)
AICc(fit2)
AICc(fit3)
fit = arima(ts_dat_bc, order = c(3,1,1), method="ML")
fit
fit2 = arima(ts_dat_bc, order = c(4,1,1), method="ML")
fit2
autoplot(fit, type = c("both", "ar", "ma"))
library(forecast)
res <- residuals(fit)
hist(res,density=20, breaks = 30, col="blue", xlab="", prob=TRUE)
```

```r
m <- mean(res)
m
std <- sqrt(var(res))
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res)))
abline(fitt, col="red")
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit), lag.max=30, main = "Autocorrelation")
# pacf
pacf(residuals(fit), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit))
qqline(residuals(fit),col ="red")
# Add overall title
title("Model4 Fitted Residuals Diagnostics", outer=TRUE)
Box.test(res, lag = 12, type = c("Box-Pierce"), fitdf = 4)
Box.test(res, lag = 12, type = c("Ljung-Box"), fitdf = 4)
Box.test(res^2, lag = 12, type = c("Ljung-Box"), fitdf = 4)
shapiro.test(residuals(fit))
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
autoplot(fit3, type = c("both", "ar", "ma"))
library(forecast)
res3 <- residuals(fit3)
hist(res3,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res3)
m
std <- sqrt(var(res3))
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res3)
fitt <- lm(res3 ~ as.numeric(1:length(res3)))
abline(fitt, col="red")
par(mfrow=c(1,2),oma=c(0,0,2,0)) # Plot diagnostics of residuals
op <- par(mfrow=c(2,2))
# acf
acf(residuals(fit3), lag.max=30, main = "Autocorrelation")
# pacf
pacf(residuals(fit3), lag.max=30, main = "Partial Autocorrelation")
qqnorm(residuals(fit3))
qqline(residuals(fit3),col ="red")
# Add overall title
title("Model 3 Fitted Residuals Diagnostics", outer=TRUE)
Box.test(res3, lag = 12, type = c("Box-Pierce"), fitdf = 5)
Box.test(res3, lag = 12, type = c("Ljung-Box"), fitdf = 5)
Box.test(res3^2, lag = 12, type = c("Ljung-Box"), fitdf = 5)
shapiro.test(residuals(fit3))
ar(res3, aic = TRUE, order.max = NULL, method = c("yule-walker"))
library(forecast)
library(astsa)
fitA <- arima(ts_dat_bc, order=c(3,1,1), method="ML")
# To produce graph with 8 forecasts on transformed data:
pred.tr <- predict(fitA, n.ahead = 4)
```

```
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(ts_dat_bc, xlim=c(1,length(ts_dat_bc)+4), ylim = c(min(ts_dat_bc),max(U.tr)), main = "ARIMA(3,1
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(ts_dat_bc)+1):(length(ts_dat_bc)+4), pred.tr$pred, col="red")
pred <- predict(fitA, n.ahead = 12)
origin <- (pred.tr$pred*lambda+1)^(1/lambda)
pred.orig <- (pred$pred*lambda+1)^(1/lambda)
U.tr<-pred$pred + 2*pred$se
L.tr<- pred$pred - 2*pred$se
U<-(U.tr*lambda+1)^(1/lambda)
L<- (L.tr*lambda+1)^(1/lambda)
ts.plot(x.train, xlim=c(1,length(x.train)+12), ylim = c(min(ts_df_vol),max(U)), main = "ARIMA(3,1,1) Fo
lines(pred.orig, col = 'red')
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
par(mfrow=c(1, 1))
pred.try <- sarima.for(x.train, n.ahead=4, plot.all=F, p=3, d=1, q=1, D=0,P=0,Q=0)
lines((n - 3):n, pred.try$pred, col="red")
lines((n - 3):n, x.test, col="blue")
points((n - 3):n, x.test, col="blue")
legend("topright", pch=1, col=c("red", "blue"),  legend=c("Forecasted values", "True Values"))
```