Time Series Analysis final Project - Competitive

MATH 1318 Time Series Analysis Final Project

Team Members: 1.Rahul k. gupta (s3635232), 2.Terrie christensen (s3664899), 3.Napapach Dechawatthanaphokin (s3613572)

14 May 2018

Contents

1 Introduction

Bitcoin is a type of cryptocurrency, i.e. it is a digital currency which uses encryption techniques to generate units of the currency and verify the transfer of funds. Bitcoin is a decentralised currency, which operates independently of a central bank. An estimated 2.9 to 5.8 million unique users have a *cryptocurrency wallet*, of which most use bitcoin. The price of bitcoin has gone through various cycles of appreciation and depreciation, known as bubbles and bursts, with price fluctuations up to a magnitude of a few thousand USD in the space of a day, so that the currency has become renown for its volatility. The bitcoin historical price data gathered from the CoinMarketCap. This time series will be modelled using regression, ARIMA and GARCH methods. The report details;

- Description of the time series
- Model specification
- Model fitting and selection
- Diagnostic checking
- Predict the value of bitcoin for the following 10 days

2 Initial Diagnosis

```
# Import Libraries
library(TSA)
library(fUnitRoots)
library(forecast)
library(CombMSC)
library(lmtest)
library(fGarch)
library(rugarch)
library(zoo)
library(ggplot2)
require(readr)
require(FitAR)

Bitcoin <- read.csv("../data/Bitcoin_Historical_Price.csv", header=TRUE)
Bitcoin$Date = as.Date(Bitcoin$Date,'%Y-%m-%d')
Bitcoin.zoo <- zoo(Bitcoin$Close, Bitcoin$Date)
Bitcoin.raw = Bitcoin.zoo
```

Data is converted to time series object using zoo library. Figure 1 shows the daily closing price of bitcoin from the 27th Apr 2013 to the 3rd Mar 2018, given in USD.

```
autoplot.zoo(Bitcoin.zoo) +
  ylab('Closing Price (USD)') +
  xlab('Time (days)') +
  ggtitle("Time Series Plot for Daily Bitcoin Prices")
```

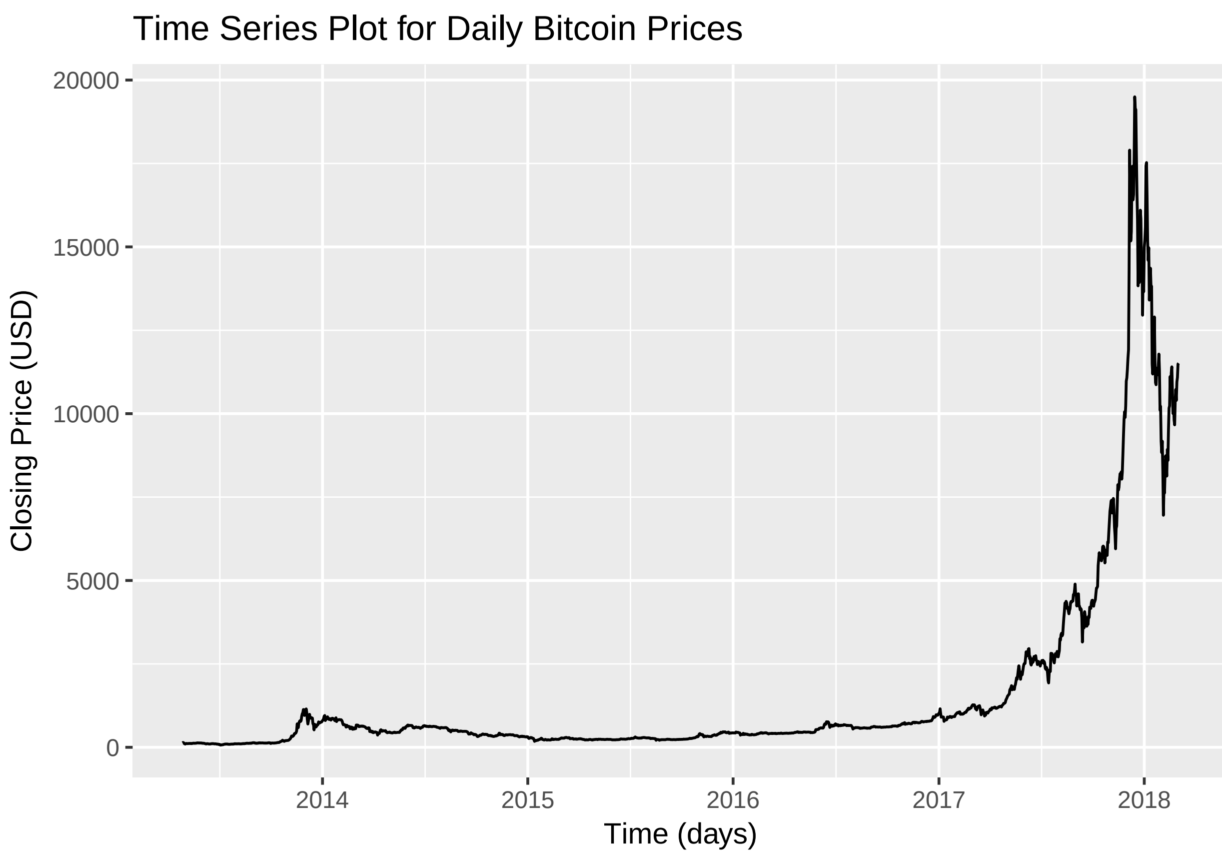

Figure 1: Time Series of Daily Bitcoin Prices

Figure 1 shows the time series for the daily closing price of bitcoin from the 27/04/2013 to the 03/03/2018, given in USD.

Main characteristics of the time series;

- Change in trend, at ~ 2017
- Change in variance, large spikes in price at the end of the time series
- Auto regressive behavior

A flat trend is observed from the start of the time series to early 2017.

```
Bitcoin.2017 = Bitcoin[Bitcoin$Date > as.Date("2017-04-01"),]
Bitcoin.2017.zoo = zoo(Bitcoin.2017$Close, Bitcoin.2017$Date)
autoplot(Bitcoin.2017.zoo) +
  geom_point(size=.5) +
  ylab('Closing Price (USD)') +
  xlab('Time (days)') +
  ggtitle("Time Series Plot for Daily Bitcoin Prices (2017-2018)")
```

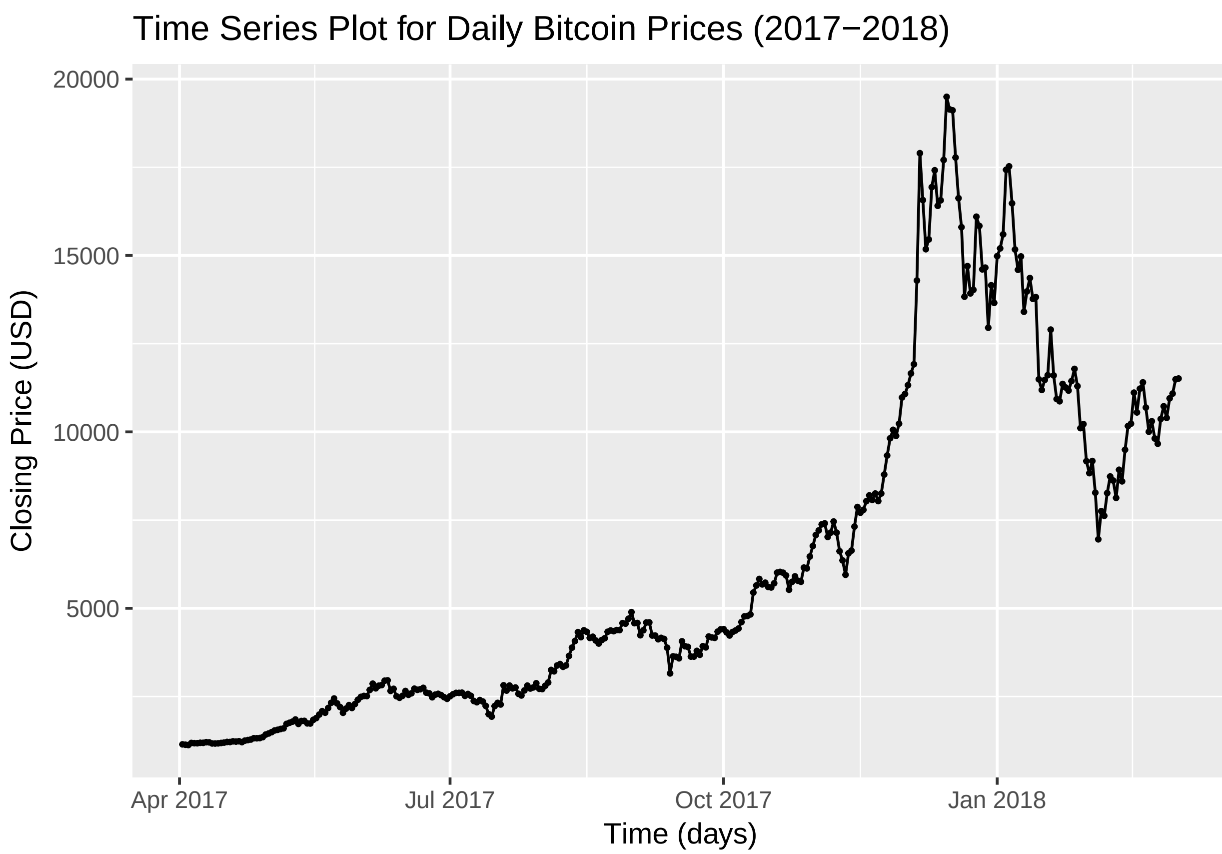

Figure 2: Subset Time Series of Daily Bitcoin Prices

Figure 2 shows The 'flat' part of the time series is removed, to better model the later part of the time series which shows a change in bitcoin price. The plot shows the daily closing price of bitcoin from the 01/04/2017 to the 03/03/2018, given in USD.

Main characteristics of the time series;

- Change in trend
- Change in variance, large spikes in price at the end of the time series
- Auto regressive behavior

2.1 Scatter Plot and correlation

```
ggplot(Bitcoin.2017,aes(zlag(Close), Close)) + geom_point() +
  ylab('Current Closing Price (USD)') +
  xlab('Previous Day Closing Price (USD)') +
  ggtitle("Scatter Plot of Bitcoin Daily Closing Prices")
```

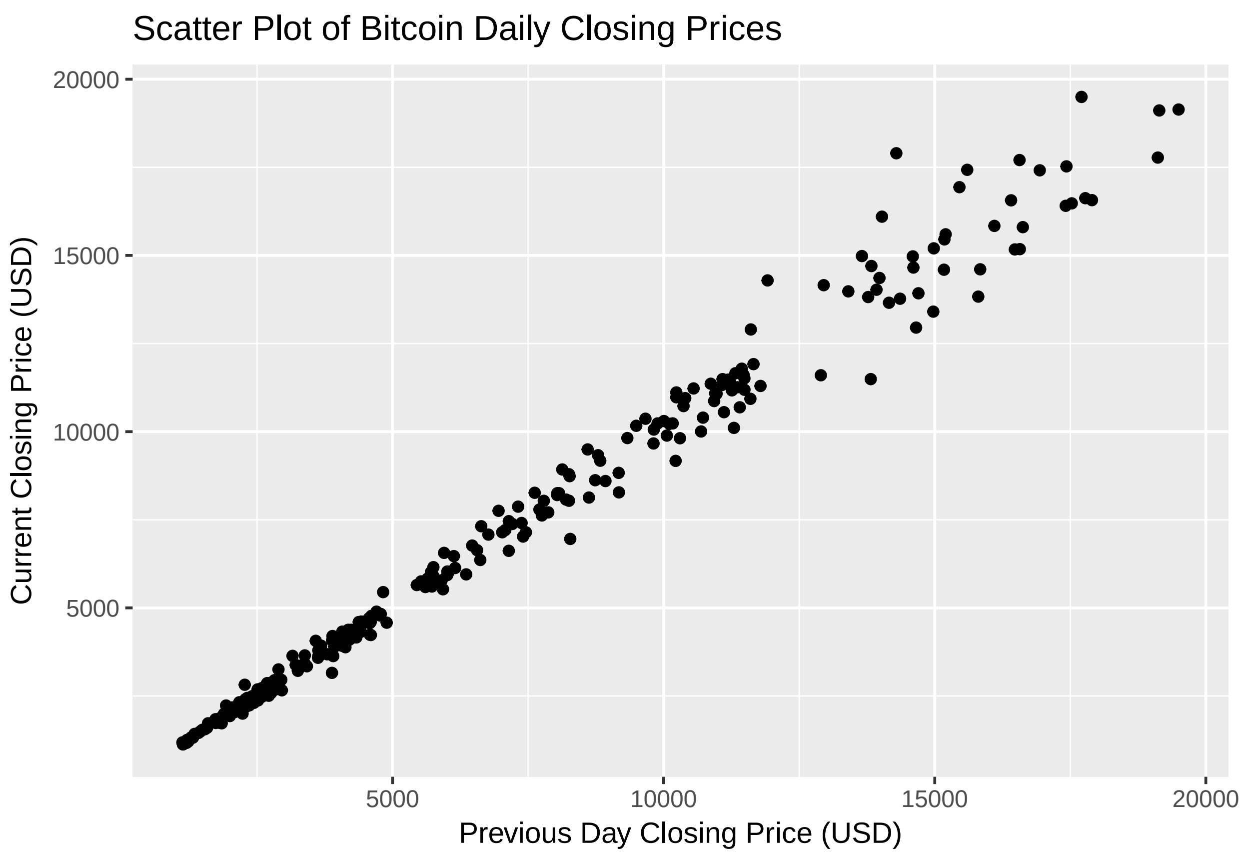

Figure 3: Scatter Plot of Bitcoin Daily Closing Prices

Figure 3 shows the scatter plot of the relationship between consecutive points, to determine the presence of auto regressive behaviour which a correlation index of 0.9935 was calculated.

This is indicative of a strong positive correlation between neighboring Bitcoin values, and in turn the presence of auto correlation.

```
y = as.vector(Bitcoin.2017.zoo)
x = zlag(Bitcoin.2017.zoo)
index = 2:length(x)
cor(y[index],x[index])
```

```
## [1] 0.9935557
```

2.2 Linear Model

```
model.ln = lm(Bitcoin.2017.zoo~time(Bitcoin.2017.zoo)) # label the linear trend model as model.ln
summary(model.ln)
```

```
##
## Call:
```

```
## lm(formula = Bitcoin.2017.zoo ~ time(Bitcoin.2017.zoo))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4954.5 -1579.6  -668.9   881.2  9660.6
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -7.021e+05  2.461e+04  -28.53   <2e-16 ***
## time(Bitcoin.2017.zoo)  4.065e+01  1.412e+00   28.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2511 on 334 degrees of freedom
## Multiple R-squared:  0.7127, Adjusted R-squared:  0.7119
## F-statistic: 828.6 on 1 and 334 DF,  p-value: < 2.2e-16
```

```r
ggplot(Bitcoin.2017,aes(Date,Close))+
  geom_line()  +
  ylab('Closing Price (USD)') +
  xlab('') +
  ggtitle('Linear Fitted Model - Bitcoin Prices') +
  geom_line(aes(y=fitted(model.ln)),color='#fc5e13')
```

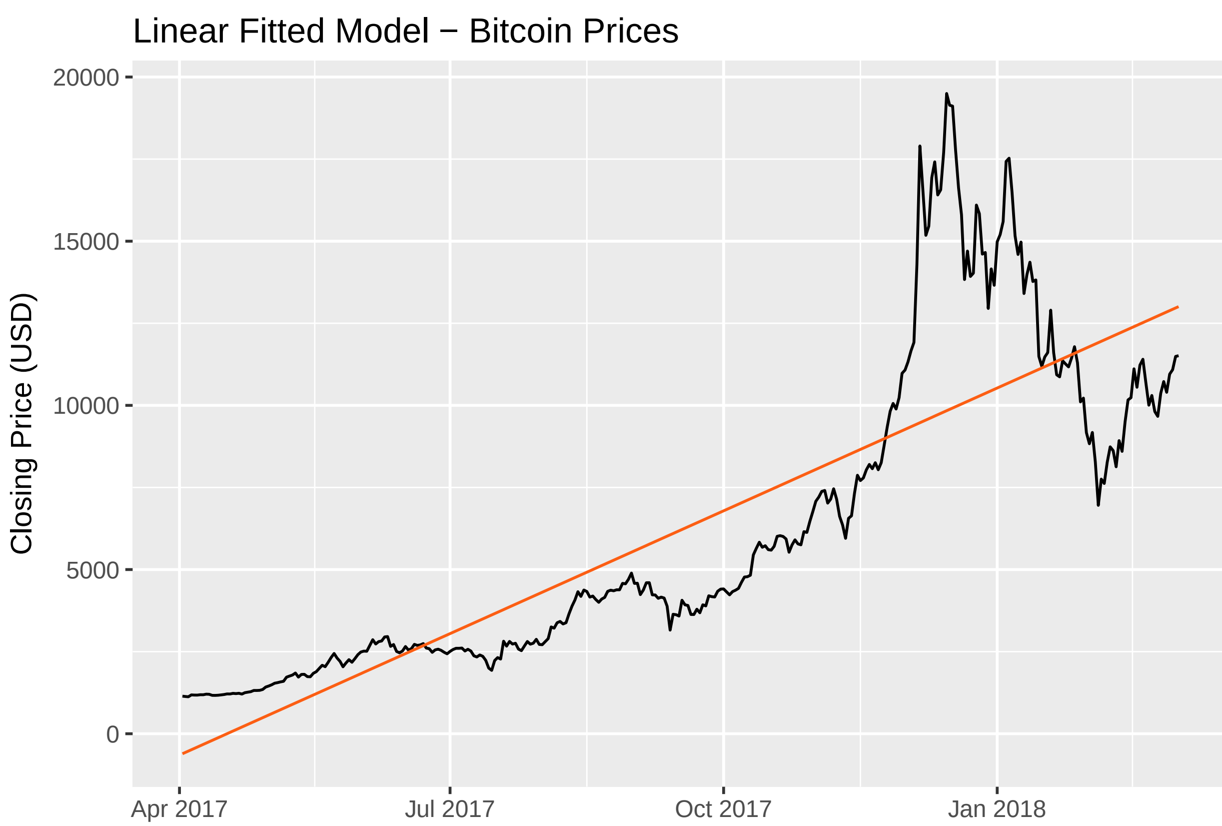

Figure 4: Linear Fitted Model - Bitcoin Prices

Figure 4 shows the plots give the regression models for the linear are statistically significant, with the same p-value of 2.2e-16 and R-squared values; 0.7119.

2.3 Residual Analysis - Linear Model

Below are the findings of residuals from linear model

```
residual_analysis_qq <- function(myresiduals, title = 'QQ Plot of Residuals') {
data=as.data.frame(qqnorm( myresiduals , plot=F))
ggplot(data,aes(x,y)) +
  geom_point() +
  geom_smooth(method="lm", se=FALSE, color='#e36209', size=.4)+
  xlab('Theoretical') +
  ylab('Sample') +
  ggtitle(title)
}

checkresiduals(model.ln)
```

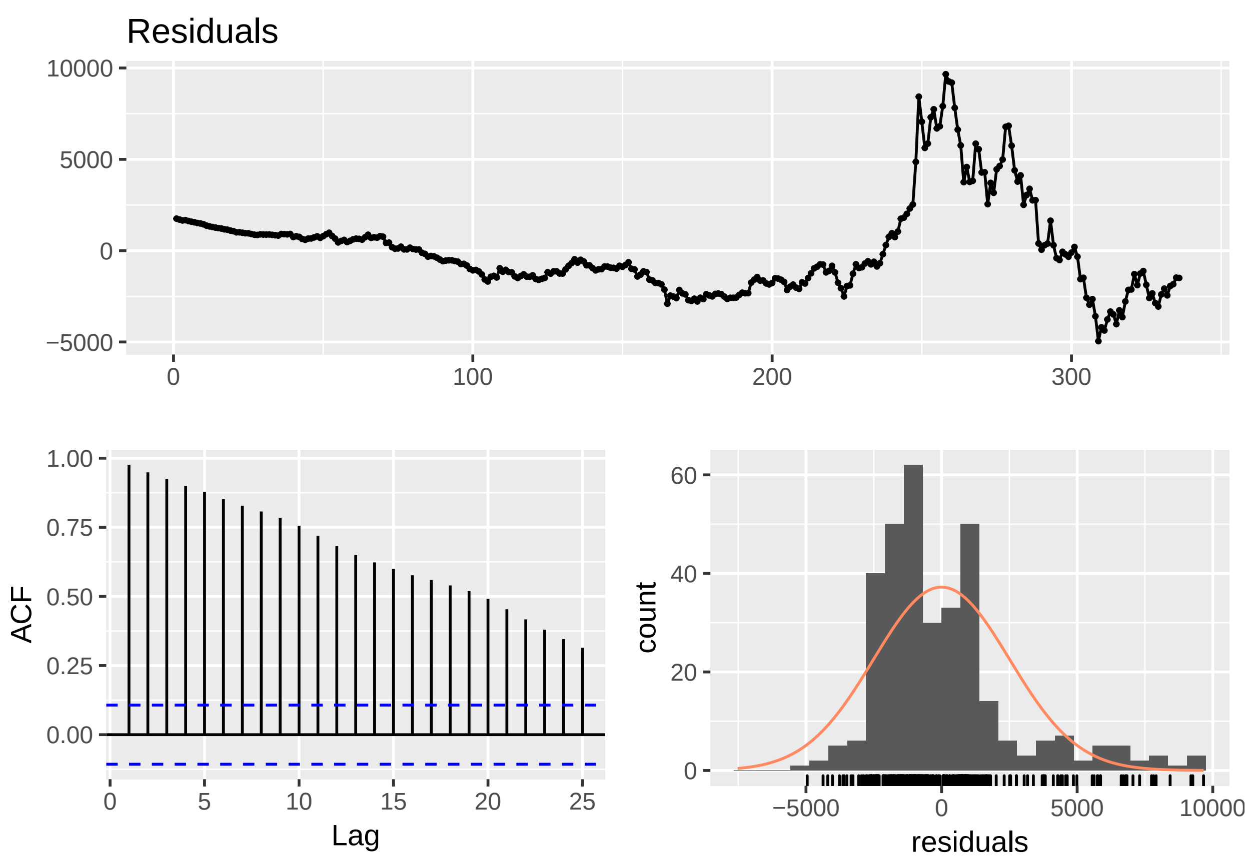

Figure 5: Residual Analysis Linear fitted Model

```
##
```

```
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 321.71, df = 10, p-value < 2.2e-16
```

```r
residual_analysis_qq(residuals(model.ln))
```

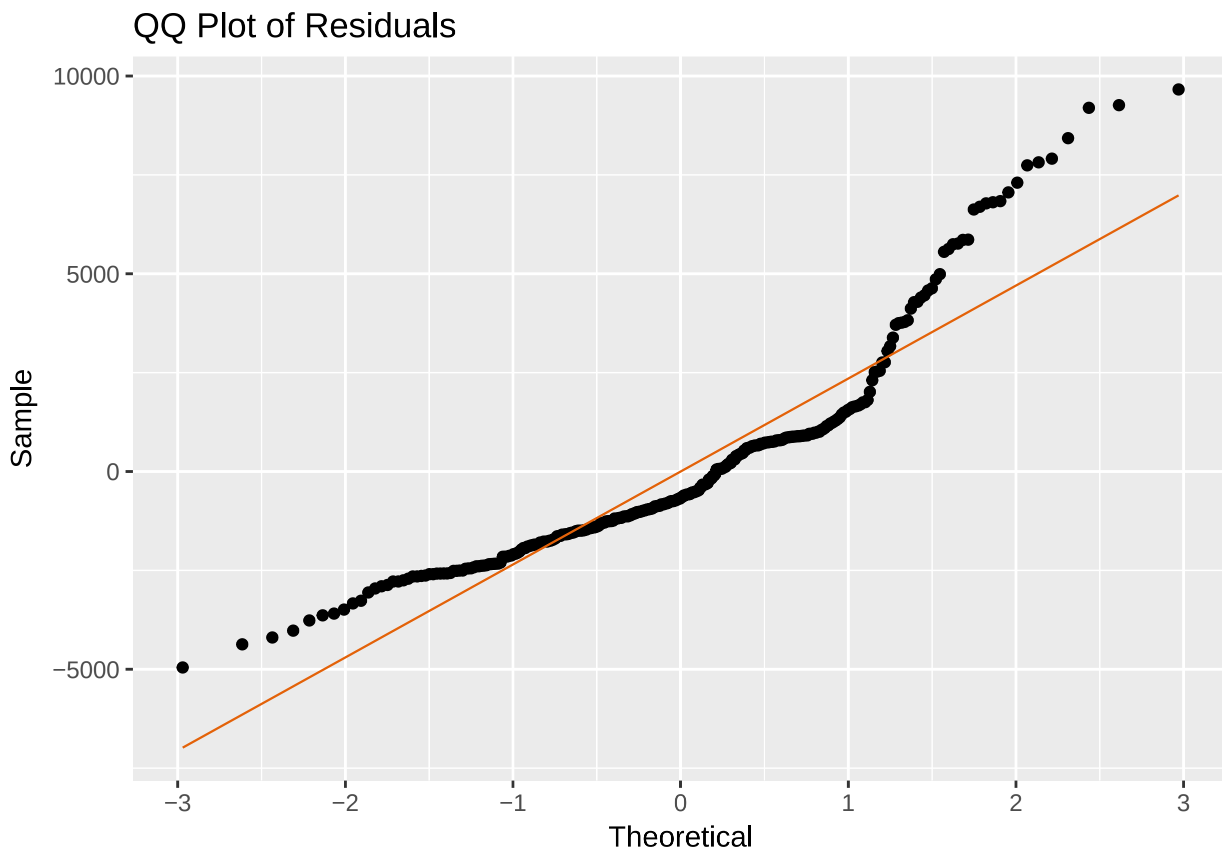

Figure 6: Residual Analysis Linear fitted Model

Figure 5 and 6 shows the panel above gives a (time series) plot, ACF, and histogram of the standardized residuals for the linear model of the Bitcoin time seriesshows and the normal QQ plot of residuals.

The standardized residuals show large deviations from the mean 0 in the (time series) plot, and an asymmetric distribution in the histogram, suggesting large errors. The QQ-plot also shows points diverging away from the straight line at either tail, indicating the residuals are not normally distributed. The ACF shows a slow decaying pattern with many significant lags, suggestive of auto correlation.

The Shapiro-Wilk test of normality (not shown) returned a p-value of 1.204e-15, so the NULL hypothesis is rejected, again suggesting that the residuals are not derived from a normally distributed population. The linear model does not pass the diagnostic checks, thus the linear model does not capture all the information in the time series and is not suitable for forecasting.

```r
shapiro.test(as.vector(residuals(model.ln)))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  as.vector(residuals(model.ln))
## W = 0.87841, p-value = 1.204e-15
```

2.4 Quadratic Model

```r
t = as.vector(time(Bitcoin.2017.zoo))
t2 = t^2
model.qa = lm(Bitcoin.2017.zoo~ t + t2) # label the quadratic trend model as model.qa
summary(model.qa)
```

```
##
## Call:
## lm(formula = Bitcoin.2017.zoo ~ t + t2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5490.1 -1286.7  -408.4   497.0  9733.1
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.504e+07  4.874e+06   3.085  0.00221 **
## t           -1.766e+03  5.594e+02  -3.156  0.00174 **
## t2           5.183e-02  1.605e-02   3.229  0.00137 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2476 on 333 degrees of freedom
## Multiple R-squared:  0.7214, Adjusted R-squared:  0.7198
## F-statistic: 431.2 on 2 and 333 DF,  p-value: < 2.2e-16
```

```r
ggplot(Bitcoin.2017,aes(Date,Close))+
  geom_line()  +
  ylab('Closing Price (USD)') +
  xlab('') +
  ggtitle('Quadratic fitted Model Curve - Bitcoin Daily Prices') +
  geom_line(aes(y=fitted(model.qa)),color='#fc5e13')
```

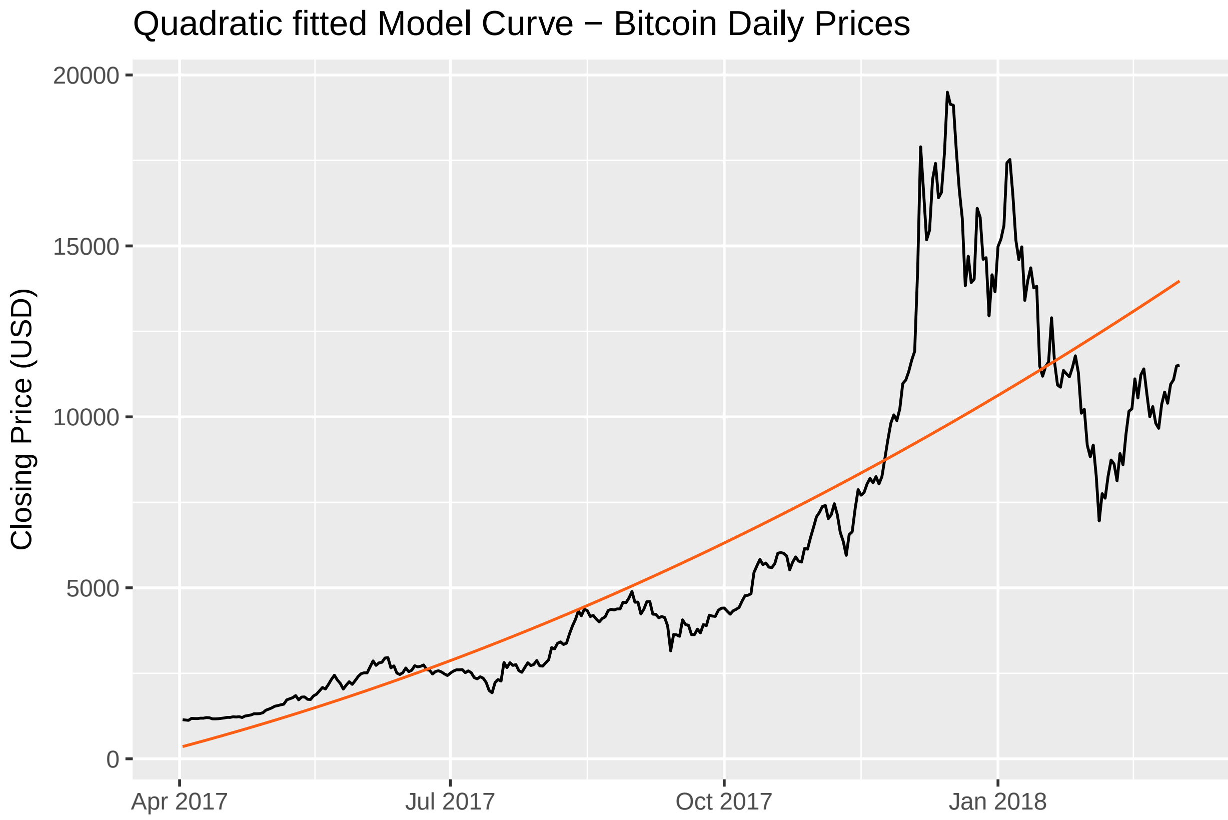

Figure **??** shows the plots give the regression models for the quadratic are statistically significant, with the same p-value of 2.2e-16 and R-squared values; 0.7198.

2.5 Residual Analysis - Quadratic Model

Below are the findings of residuals from linear model

```
checkresiduals(model.qa)
```

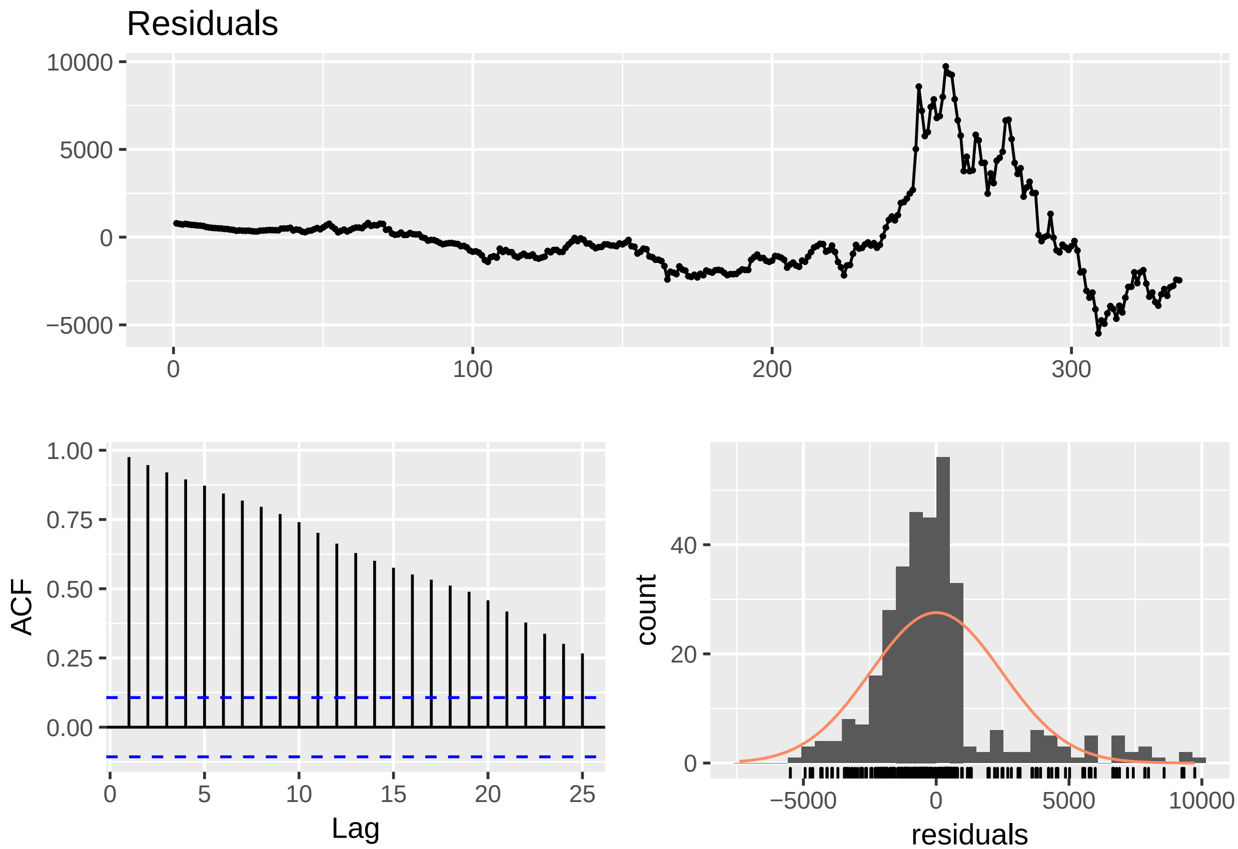

Figure 7: Residual Analysis Quadratic fitted Model

```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 321.7, df = 10, p-value < 2.2e-16
```

```r
residual_analysis_qq(residuals(model.qa))
```

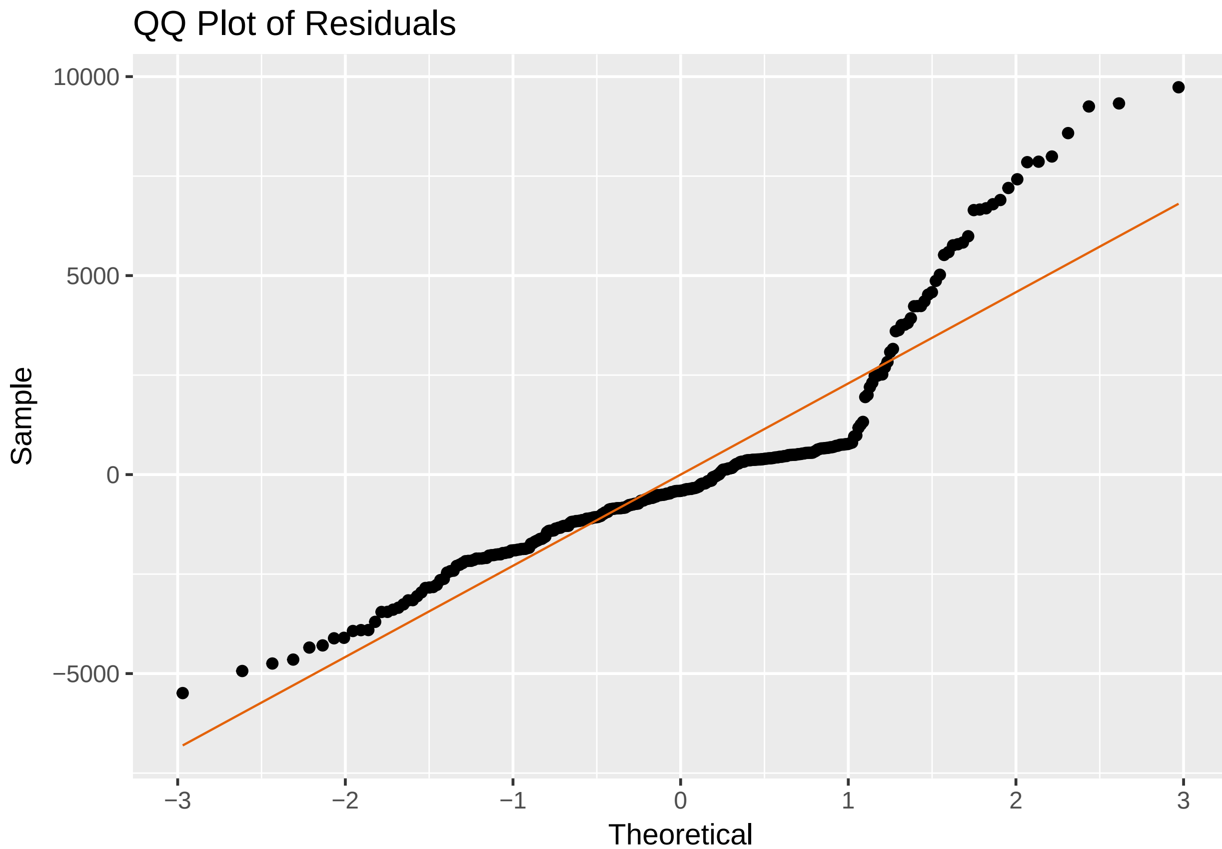

Figure 7 and **??** shows the panel above gives a (time series) plot, ACF, and histogram of the standardized residuals for the quadratic Model of the Bitcoin time series shows and the normal QQ plot of residuals.

The standardized residuals show large deviations from the mean 0 in the (time series) plot, and an asymmetric distribution in the histogram, suggesting large errors. The QQ-plot also shows points diverging away from the straight line at either tail, indicating the residuals are not normally distributed. The ACF shows a slow decaying pattern with many significant lags, suggestive of auto correlation.

The Shapiro-Wilk test of normality (not shown) returned a p-value of 2.2e-16, so the NULL hypothesis is rejected, again suggesting that the residuals are not derived from a normally distributed population. The linear model does not pass the diagnostic checks, thus the linear model does not capture all the information in the time series and is not suitable for forecasting.

```r
shapiro.test(as.vector(residuals(model.qa)))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  as.vector(residuals(model.qa))
## W = 0.86085, p-value < 2.2e-16
```

3 Models for Nonstationary Time Series

Auto regressive behaviour and non staionay Staionay is the first thing we need to check.

```
ggtsdisplay(Bitcoin.2017.zoo,
            main = 'ACF and PACF of Bitcoin Prices',
            ylab='Closing Price (USD)')
```

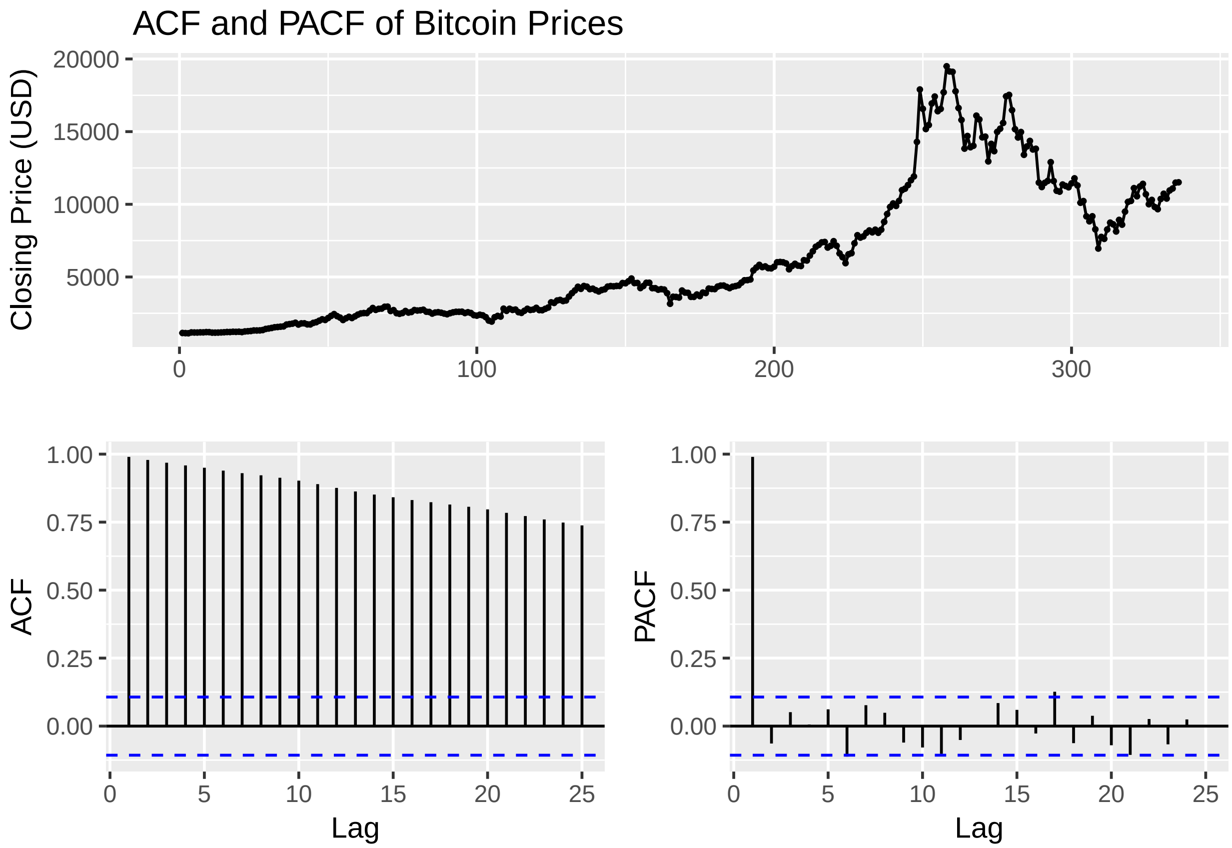

Figure **??** shows the uppermost plot in the panel shows points in the time series, with a trend and change in variance.

The ACF plot shows a slowly decaying pattern with many significant lags. There is no indication of a wave/sine pattern, so that a seasonal component is not determined.

PACF plot show a large 1st significant lag, with 4 smaller significant/near significant lags. The time series needs to be stabilized and de-trended prior to specifying a set of possible ARIMA models.

stategy to make stationay is transfromation.

```
Bitcoin.transform = BoxCox.ar(Bitcoin.2017.zoo, method = 'yule-walker')
```

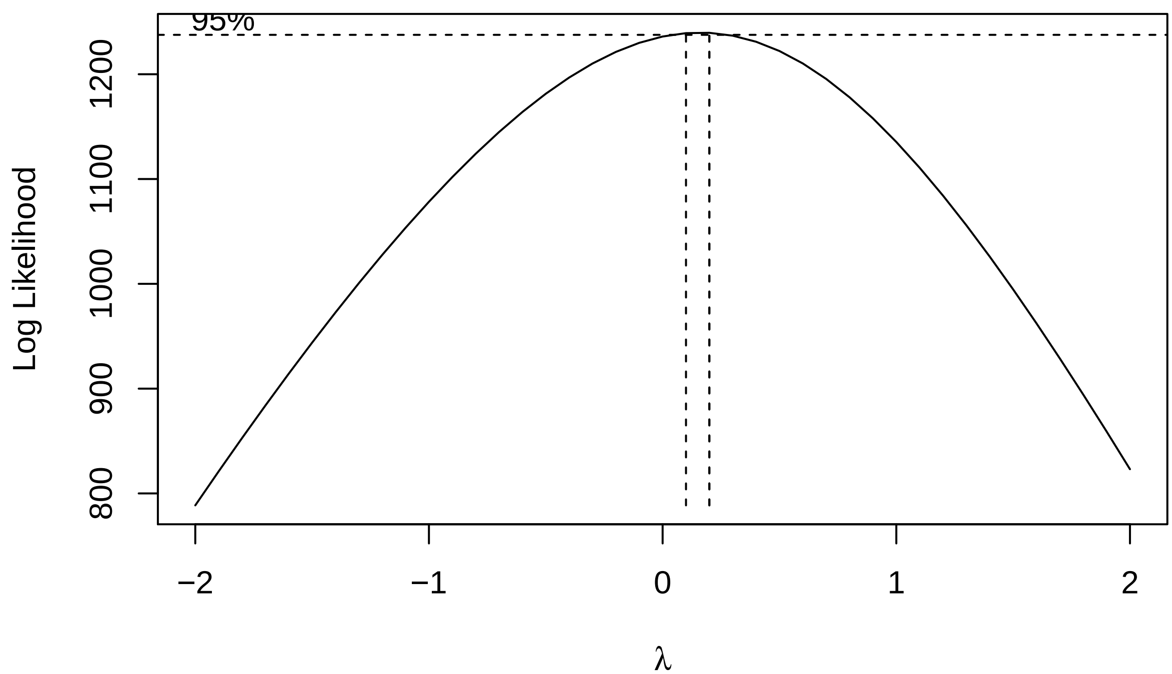

Figure 8: BoxCox Transformation

```
lambda = sum(Bitcoin.transform$ci)/length(Bitcoin.transform$ci)
Bitcoin.boxcox = (Bitcoin.2017.zoo^lambda - 1) / lambda
Bitcoin.diff = base::diff(Bitcoin.boxcox, differences = 1)
autoplot(Bitcoin.diff) +
  ylab('Closing Price (USD)') +
  ggtitle('Boxcox Transformed & First Differenced Series')
```

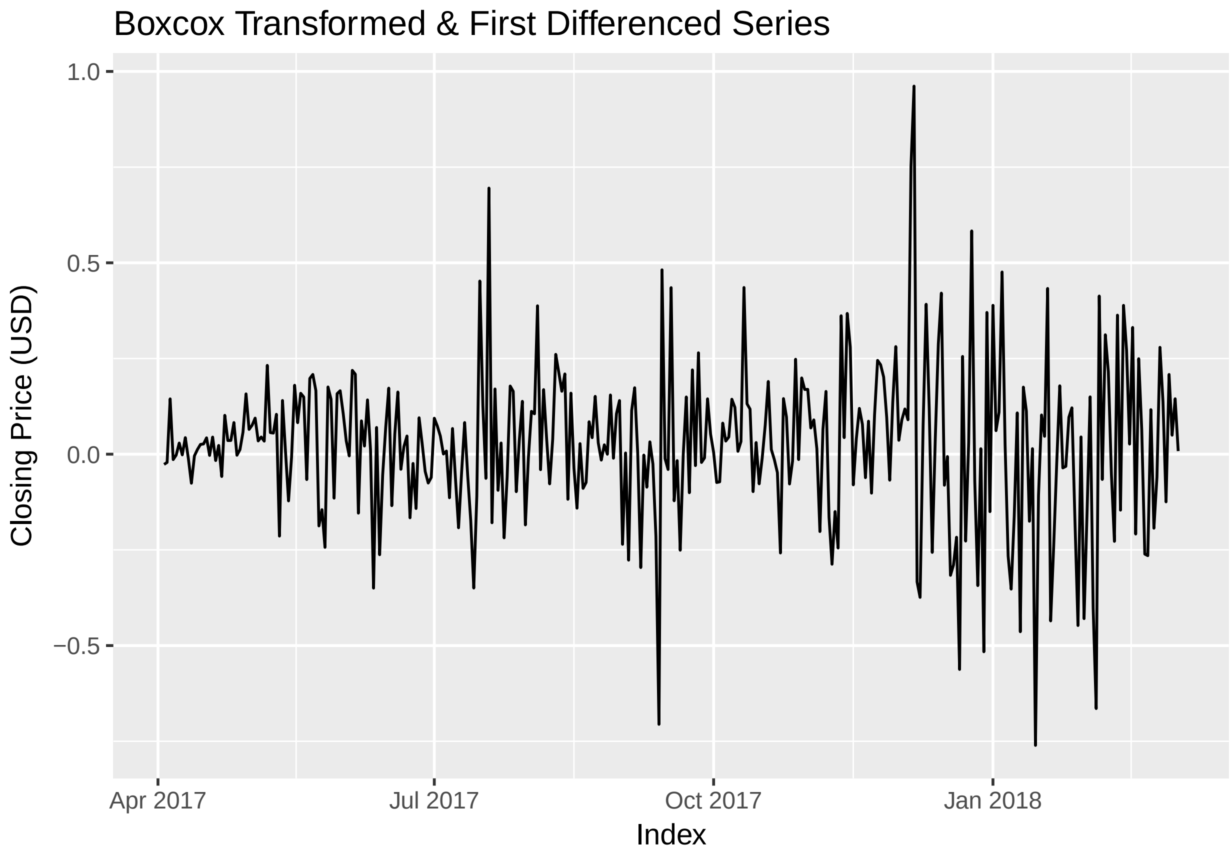

Figure 9: Boxcox Transformed & First Differenced Series

```r
ggtsdisplay(Bitcoin.diff, lag.max = 96,ci.type='ma',
            main = 'Boxcox Transformed & First Differenced ACF and PACF plots',
            ylab='')
```

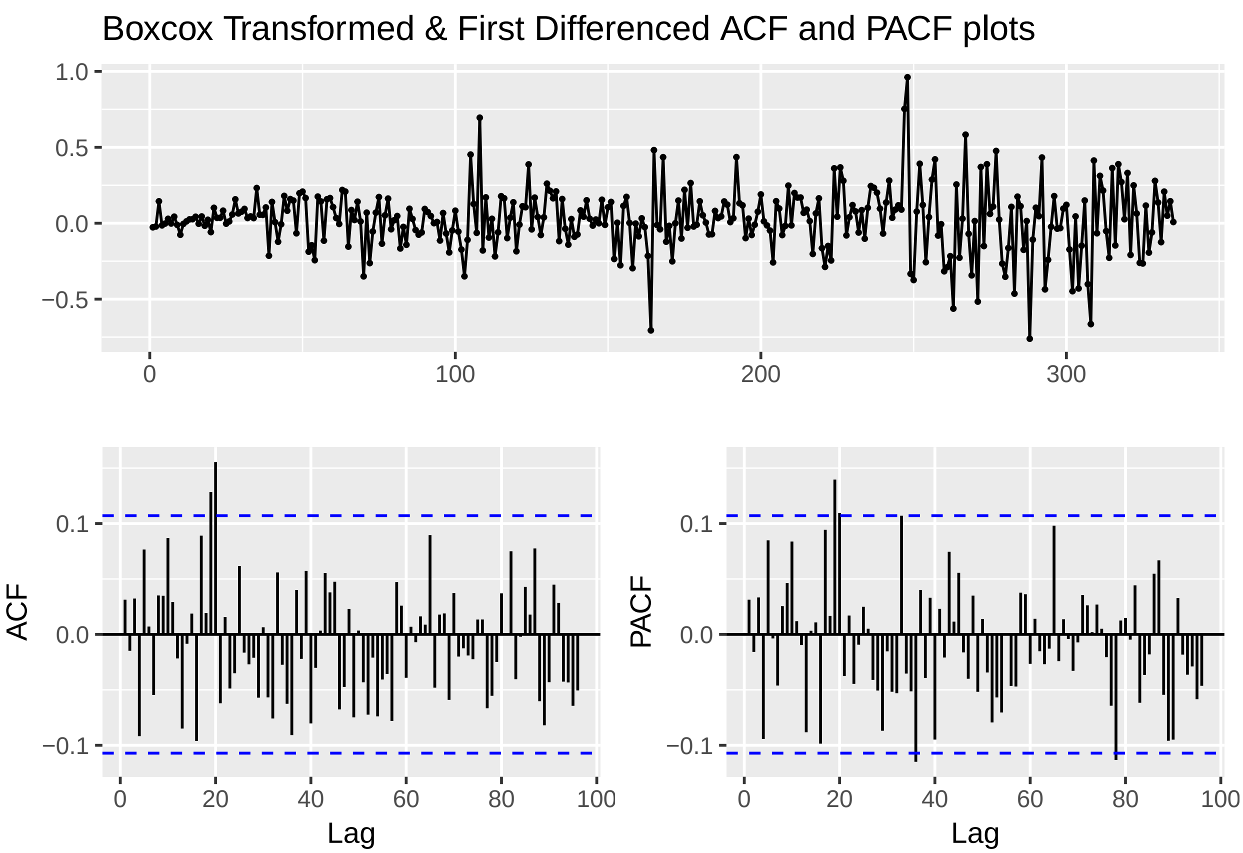

Figure 10: Boxcox Transformed & First Differenced ACF and PACF plots

Figure 10 shows calculation of the log-likelihood for each lambda value, using the yule-walker method gave a lambda value 0.1-0.2.

A BoxCox transformation was applied to stabilize the time series.

1 differencing was applied to de-trend the time series, i.e. d=1.

The uppermost plot in the panel shows the result of transformation and differencing.

Dickey-Fuller Unit-Root test gave a p-value of 0.01, thus the NULL hypothesis is rejected, and in turn the time series is determined to be stationary.

The ACF plot shows 2 significant lags, i.e. q=2.

PACF plot shows 5 significant lags, however the 5th is ~lag 80, so not included, i.e. p=4, but the adjacent value p=3 may also be considered. {ARIMA(4,1,2), ARMIA(3,1,2)} are included in the set of possible models.

```
adf.test(Bitcoin.diff)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Bitcoin.diff
## Dickey-Fuller = -6.968, Lag order = 6, p-value = 0.01
```

```
## alternative hypothesis: stationary
```

```r
eacf(Bitcoin.diff)
```

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

```r
# ARIMA(0,1,0),ARIMA(1,1,1),ARIMA(2,1,2),ARIMA(4,1,3)
```

The eacf plot shown above left, identified possible smaller ARIMA models with a p=1, 2 and q=1, 2.

```r
res1 = armasubsets(y=Bitcoin.diff,nar=14,nma=14,y.name='test',ar.method='mle')
plot(res1)
```

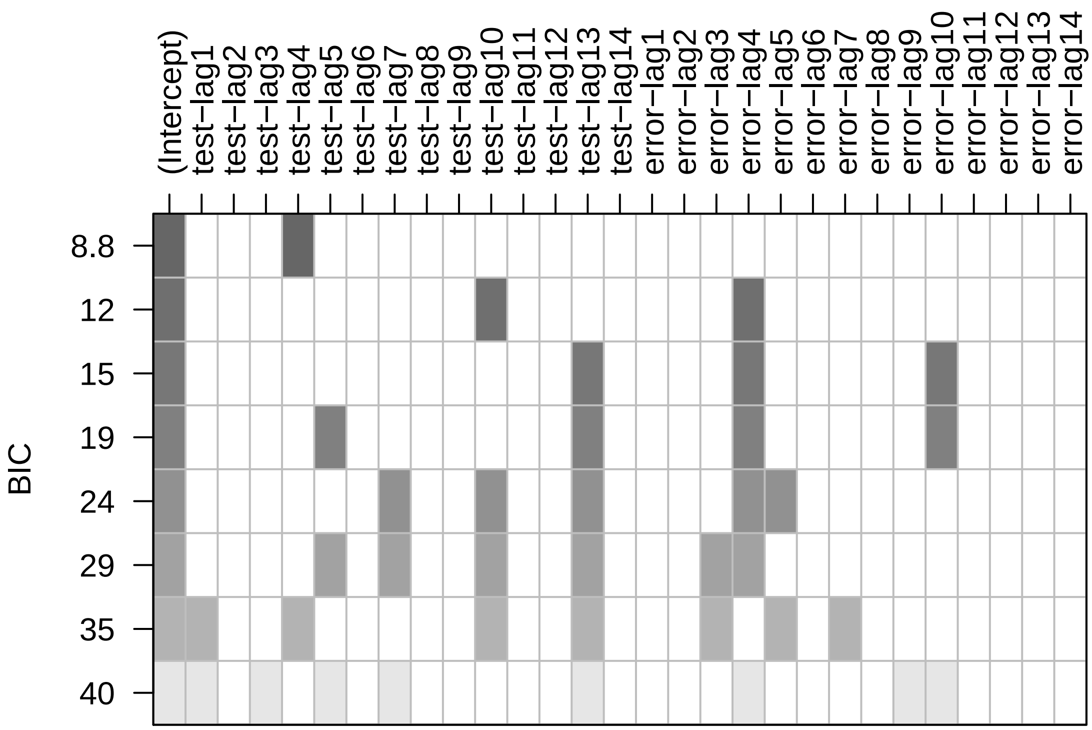

Figure 11: BIC Table

```r
#ARIMA(4,1,4),ARIMA(5,1,4)
```

Figure 11 shows the BIC plot given above right, identified possible larger ARIMA models with a p=4, 5 and q=4. A p=10 was disregarded with smaller values identified.

Thus {ARIMA(1,1,1), ARMIA(1,1,2), ARIMA(2,1,1), ARIMA(2,1,2), ARIMA(4,1,4), ARIMA(5,1,4)} are added to the set of possible models.

```
#The final set of possible models is
# ARIMA(0,1,0),ARIMA(1,1,1),ARIMA(2,1,2),ARIMA(4,1,3)
# ARIMA(4,1,4),ARIMA(5,1,4)

# ARIMA(1,1,1)
model_111_css = arima(Bitcoin.boxcox, order=c(1,1,1),method='CSS')
coeftest(model_111_css)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 0.021830         NA      NA       NA
## ma1 0.022332         NA      NA       NA
```

```
model_111_ml = arima(Bitcoin.boxcox, order=c(1,1,1),method='ML')
coeftest(model_111_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 0.020106         NA      NA       NA
## ma1 0.024617         NA      NA       NA
```

```
# ARIMA(1,1,2)
model_112_css = arima(Bitcoin.boxcox,order=c(1,1,2),method='CSS')
coeftest(model_112_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.698636   0.258350 -2.7042 0.006846 **
## ma1  0.749424   0.260011  2.8823 0.003948 **
## ma2 -0.012098   0.059763 -0.2024 0.839579
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_112_ml = arima(Bitcoin.boxcox,order=c(1,1,2),method='ML')
coeftest(model_112_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.713575   0.268293 -2.6597 0.007821 **
## ma1  0.764360   0.269563  2.8356 0.004575 **
## ma2 -0.010088   0.061085 -0.1651 0.868831
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# ARIMA(2,1,1)
model_211_css = arima(Bitcoin.boxcox,order=c(2,1,1),method='CSS')
coeftest(model_211_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.729981   0.236531 -3.0862 0.0020274 **
## ar2 -0.010729   0.062640 -0.1713 0.8640056
## ma1  0.779953   0.230417  3.3850 0.0007119 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_211_ml = arima(Bitcoin.boxcox,order=c(2,1,1),method='ML')
coeftest(model_211_ml)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error z value Pr(>|z|)
## ar1  0.0091928         NA      NA       NA
## ar2 -0.0035792  0.0541984  -0.066   0.9473
## ma1  0.0335949         NA      NA       NA
```

```r
# ARIMA(2,1,2)
model_212_css = arima(Bitcoin.boxcox,order=c(2,1,2),method='CSS')
coeftest(model_212_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value Pr(>|z|)
## ar1 -0.028545   0.079418  -0.3594   0.7193
## ar2  0.906965   0.075874  11.9535   <2e-16 ***
## ma1  0.085822   0.084577   1.0147   0.3102
## ma2 -0.913644   0.083827 -10.8991   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_212_ml = arima(Bitcoin.boxcox,order=c(2,1,2),method='ML')
coeftest(model_212_ml)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value Pr(>|z|)
## ar1  0.0051321  0.0690326   0.0743   0.9407
## ar2  0.9315260  0.0654587  14.2307   <2e-16 ***
## ma1  0.0502674  0.0857493   0.5862   0.5577
## ma2 -0.9495800  0.0856749 -11.0835   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# ARIMA(3,1,2)
model_312_css = arima(Bitcoin.boxcox,order=c(3,1,2),method='CSS')
```

```r
coeftest(model_312_css)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 -0.96048    0.32229 -2.9802 0.002881 **
## ar2 -0.39084    0.21351 -1.8305 0.067170 .
## ar3  0.10504    0.06089  1.7250 0.084524 .
## ma1  1.01485    0.32371  3.1351 0.001718 **
## ma2  0.45021    0.21280  2.1157 0.034370 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_312_ml = arima(Bitcoin.boxcox,order=c(3,1,2),method='ML')
coeftest(model_312_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1 -1.612866   0.084119 -19.1737 < 2.2e-16 ***
## ar2 -0.854203   0.108453  -7.8762 3.374e-15 ***
## ar3  0.048060   0.058338   0.8238      0.41
## ma1  1.679184   0.064867  25.8866 < 2.2e-16 ***
## ma2  0.925916   0.065388  14.1603 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# ARIMA(4,1,2)
model_412_css = arima(Bitcoin.boxcox,order=c(4,1,2),method='CSS')
coeftest(model_412_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.737242   0.415004 -1.7765  0.07566 .
## ar2 -0.499844   0.577987 -0.8648  0.38715
## ar3  0.056956   0.071481  0.7968  0.42556
## ar4 -0.080260   0.070306 -1.1416  0.25363
## ma1  0.793473   0.414413  1.9147  0.05553 .
## ma2  0.540068   0.603325  0.8952  0.37070
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_412_ml = arima(Bitcoin.boxcox,order=c(4,1,2),method='ML')
coeftest(model_412_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value Pr(>|z|)
## ar1 -1.109588   0.055297 -20.0661   <2e-16 ***
## ar2 -0.902688   0.082158 -10.9872   <2e-16 ***
## ar3  0.059431   0.082539   0.7200   0.4715
```

```
## ar4 -0.029706   0.055097  -0.5392   0.5898
## ma1  1.185968   0.015665  75.7078   <2e-16 ***
## ma2  0.997414   0.023943  41.6570   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# ARIMA(4,1,4)
model_414_css = arima(Bitcoin.boxcox,order=c(4,1,4),method='CSS')
coeftest(model_414_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1 -0.523664         NA       NA        NA
## ar2  0.162762   0.041831   3.8910 9.985e-05 ***
## ar3  0.917653   0.074325  12.3465 < 2.2e-16 ***
## ar4  0.415562         NA       NA        NA
## ma1  0.552550         NA       NA        NA
## ma2 -0.181660   0.040086  -4.5317 5.850e-06 ***
## ma3 -0.928353   0.071005 -13.0744 < 2.2e-16 ***
## ma4 -0.532390         NA       NA        NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_414_ml = arima(Bitcoin.boxcox,order=c(4,1,4),method='ML')
coeftest(model_414_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1 -0.857298   0.256941  -3.3366 0.0008482 ***
## ar2  0.075458   0.057469   1.3130 0.1891720
## ar3  1.120648   0.060162  18.6273 < 2.2e-16 ***
## ar4  0.637578   0.254442   2.5058 0.0122178 *
## ma1  0.922555   0.230314   4.0056 6.185e-05 ***
## ma2 -0.031454   0.060127  -0.5231 0.6008852
## ma3 -1.116176   0.056809 -19.6479 < 2.2e-16 ***
## ma4 -0.717939   0.228129  -3.1471 0.0016492 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# ARIMA(5,1,4)
model_514_css = arima(Bitcoin.boxcox,order=c(5,1,4),method='CSS')
coeftest(model_514_css)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value  Pr(>|z|)
## ar1  0.0994981  0.0424432   2.3443 0.0190647 *
## ar2  0.1381344  0.0384323   3.5942 0.0003254 ***
## ar3  0.5069664  0.0030855 164.3061 < 2.2e-16 ***
## ar4  0.1650015  0.0261166   6.3179 2.652e-10 ***
## ar5  0.0902766  0.0577983   1.5619 0.1183062
```

```
## ma1 -0.0794650         NA       NA        NA
## ma2 -0.1856691  0.0430209  -4.3158 1.590e-05 ***
## ma3 -0.5132720  0.0420131 -12.2169 < 2.2e-16 ***
## ma4 -0.3071109         NA       NA        NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
model_514_ml = arima(Bitcoin.boxcox,order=c(5,1,4),method='ML')
coeftest(model_514_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1  0.216615   0.582131  0.3721   0.7098
## ar2  0.125234   0.255391  0.4904   0.6239
## ar3  0.472675   0.520008  0.9090   0.3634
## ar4 -0.054061   0.425997 -0.1269   0.8990
## ar5  0.109269   0.078409  1.3936   0.1634
## ma1 -0.166104   0.585688 -0.2836   0.7767
## ma2 -0.144673   0.251820 -0.5745   0.5656
## ma3 -0.438177   0.523688 -0.8367   0.4028
## ma4 -0.056627   0.413964 -0.1368   0.8912
```

```r
source('sort.score.r')
sort.score(stats::AIC(model_111_ml,model_112_ml,model_211_ml,model_212_ml,model_312_ml,model_412_ml,mode
```

```
##              df        AIC
## model_412_ml  7 -103.15254
## model_312_ml  6 -102.82566
## model_212_ml  5 -102.63500
## model_414_ml  9 -101.76633
## model_111_ml  3 -100.97940
## model_112_ml  4 -100.65054
## model_211_ml  4  -98.98447
## model_514_ml 10  -95.11157
```

```r
sort.score(stats::BIC(model_111_ml,model_112_ml,model_211_ml,model_212_ml,model_312_ml,model_412_ml,mode
```

```
##              df       BIC
## model_111_ml  3 -89.53701
## model_112_ml  4 -85.39402
## model_211_ml  4 -83.72795
## model_212_ml  5 -83.56435
## model_312_ml  6 -79.94088
## model_412_ml  7 -76.45362
## model_414_ml  9 -67.43915
## model_514_ml 10 -56.97026
```

```r
fit <- Arima(Bitcoin.2017.zoo, order=c(3,1,2), lambda = lambda)
summary(fit)
```

```
## Series: Bitcoin.2017.zoo
## ARIMA(3,1,2)
## Box Cox transformation: lambda= 0.15
##
```

```
## Coefficients:
##           ar1      ar2     ar3     ma1     ma2
##       -1.6127  -0.8540  0.0479  1.6791  0.9257
## s.e.   0.0843   0.1085  0.0583  0.0651  0.0654
##
## sigma^2 estimated as 0.04215:  log likelihood=57.41
## AIC=-102.83   AICc=-102.57   BIC=-79.94
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 28.56452 517.2228 289.9566 0.5060551 3.975302 0.9891755
##                    ACF1
## Training set 0.06609031
```

3.1 Residual Analysis - ARIMA Model

Below are the findings of residuals from linear model

```r
checkresiduals(fit)
```

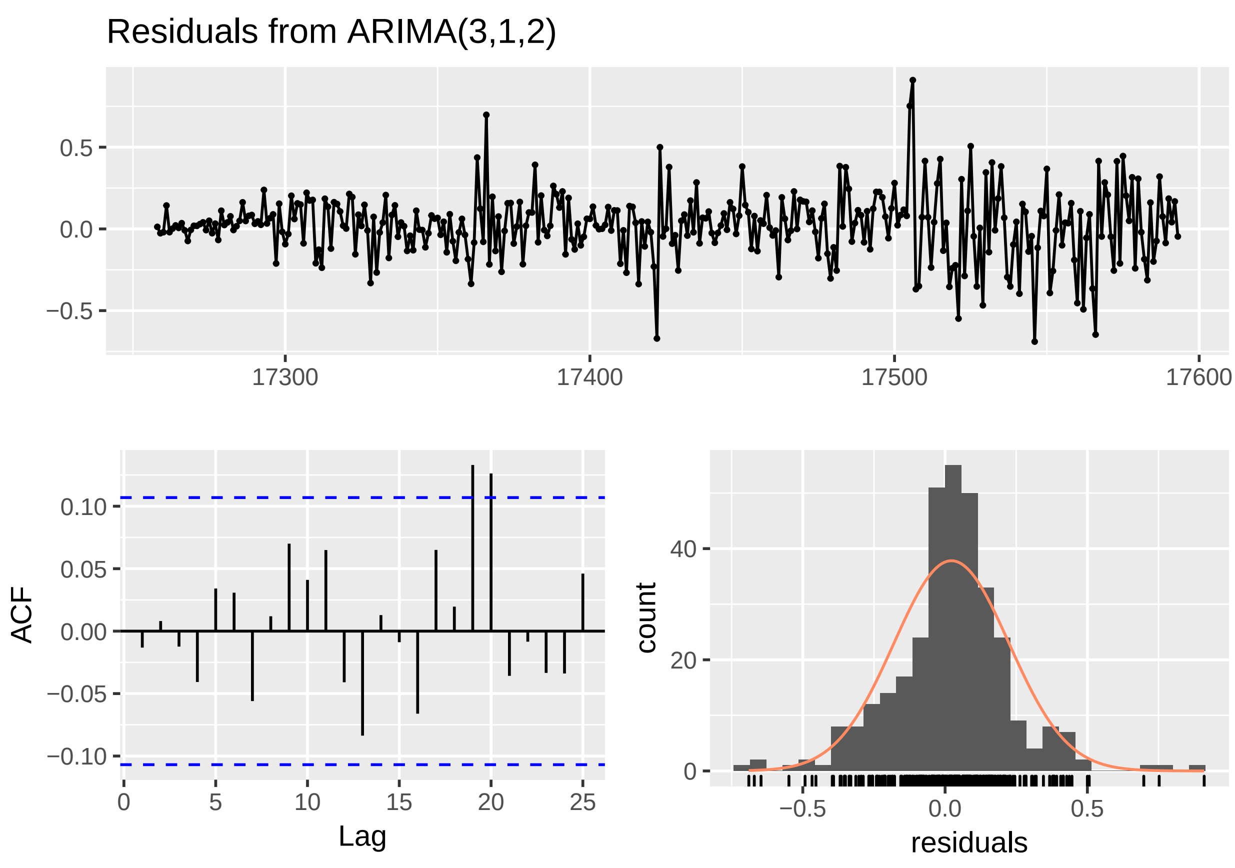

Figure 12: Residual Analysis Quadratic fitted Model

```
##
```

```
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,1,2)
## Q* = 4.8476, df = 5, p-value = 0.4348
##
## Model df: 5.   Total lags used: 10
```

```
residual_analysis_qq(residuals(fit))
```

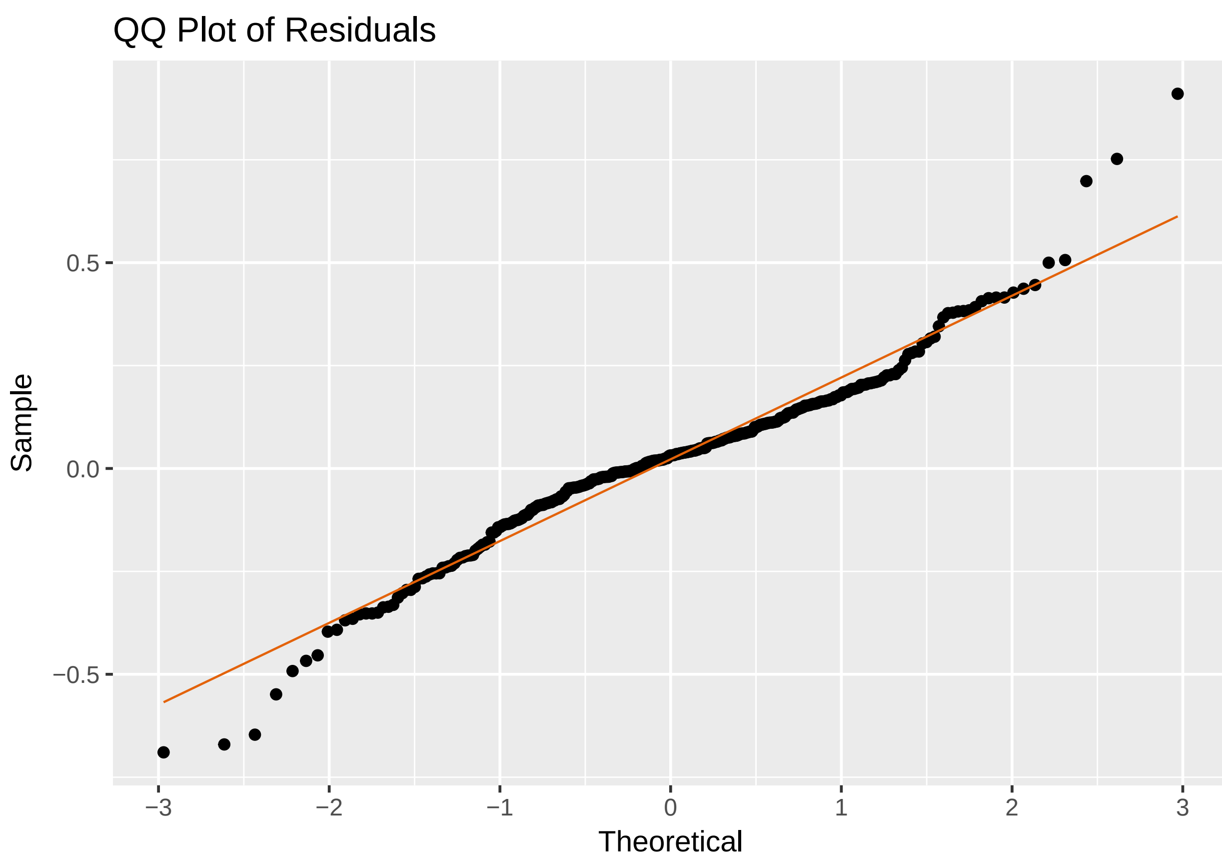

Figure 13: Residual Analysis Linear fitted Model

```
shapiro.test(as.vector(residuals(fit)))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  as.vector(residuals(fit))
## W = 0.96352, p-value = 1.919e-07
```

```
x = residuals(fit)
k=0
LBQPlot(x, lag.max = length(x)-1 , StartLag = k + 1, k = 0, SquaredQ = FALSE)
```

```r
grid()
```

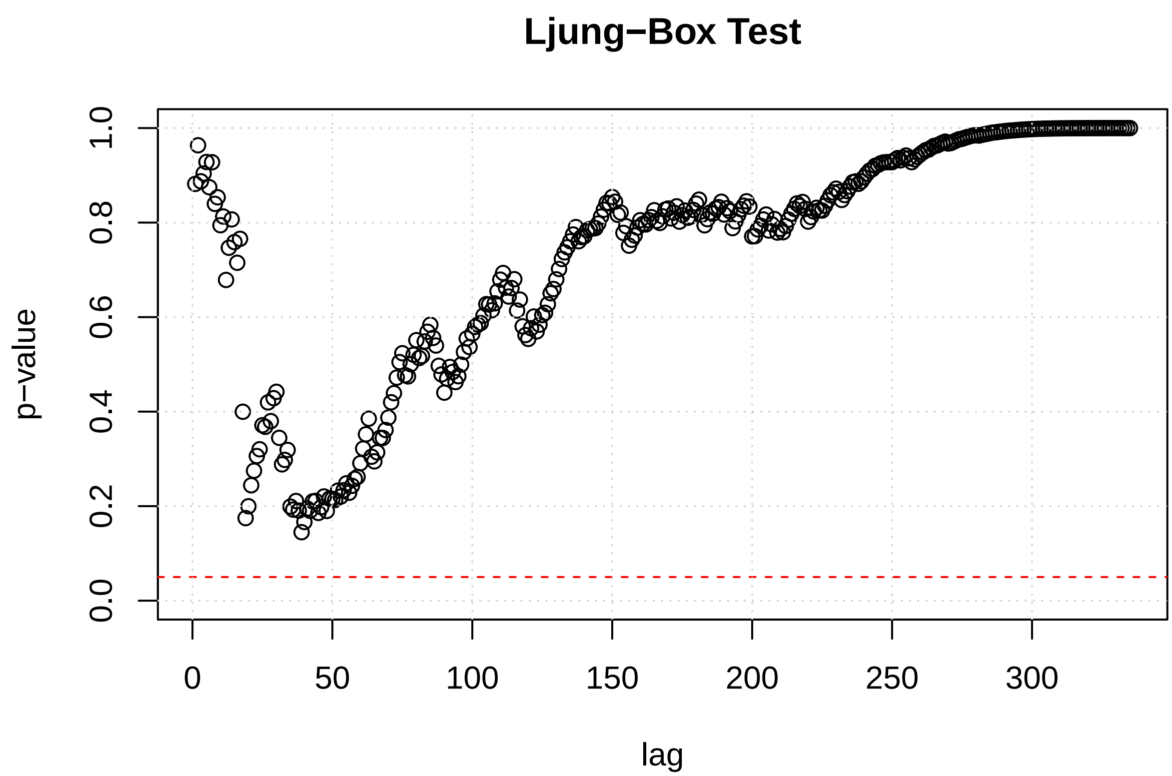

Figure 14: Ljung-Box Test

3.2 Forecast

```r
autoplot(forecast(fit,h=10))
```

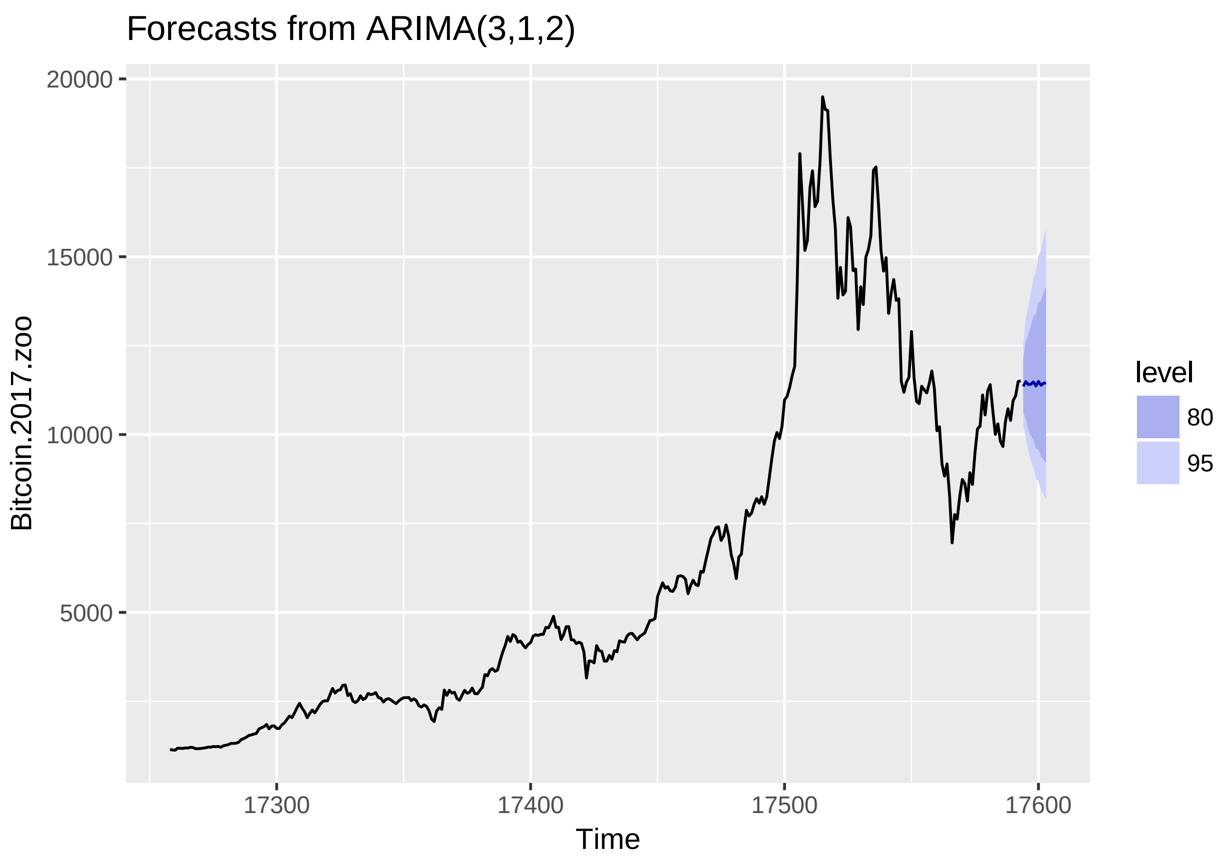

Figure 15: Forecast from ARIMA[3,1,2]

```
Bitcoin.forecast <- read_csv("../data/Bitcoin_Prices_Forecasts.csv")
Bitcoin.forecast$Date = as.Date(Bitcoin.forecast$Date,'%d/%m/%y')
```

3.3 MASE Error

```
source('MASE.r')
MASE(Bitcoin.forecast$`Closing price`,
     as.vector(tail(fitted(forecast(fit,h=10)),10)))
```

```
## $MASE
##       MASE
## 1 3.232567
```

```
McLeod.Li.test(y=Bitcoin.2017.zoo,main="McLeod-Li Test Statistics for Bitcoin")
```

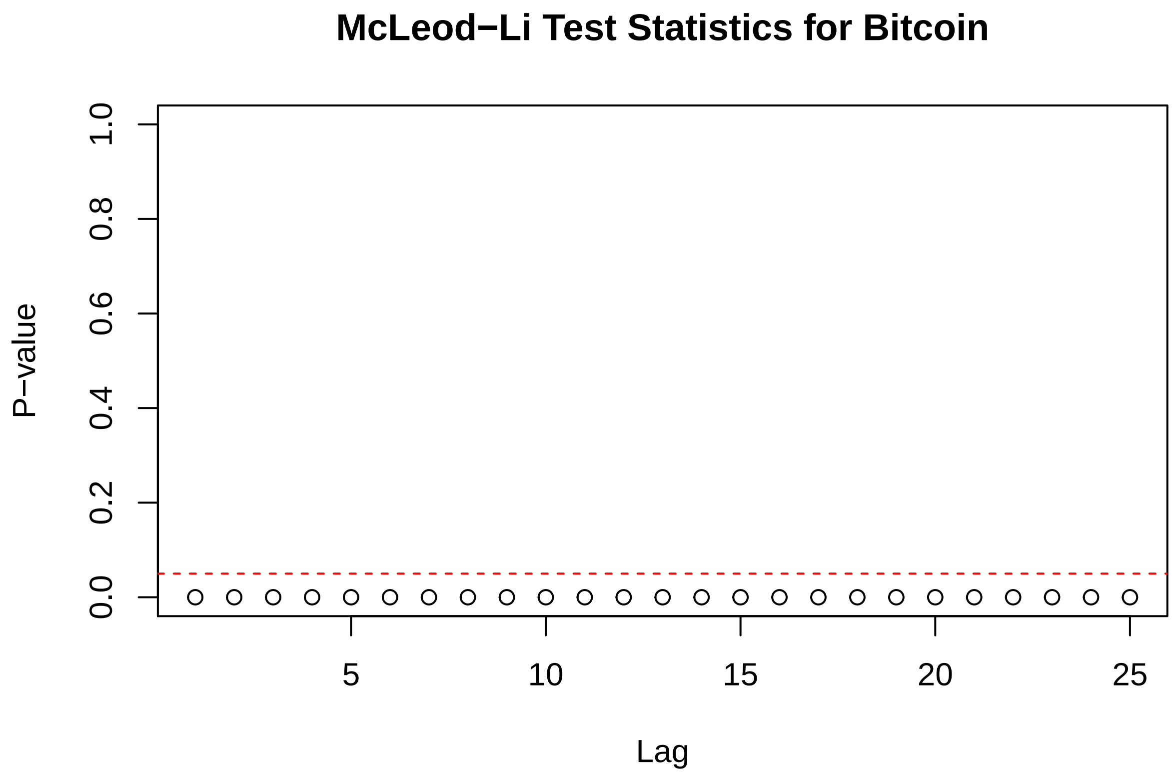

Figure 16: McLeod Li Test Statistics for Bitcoin

```
residual_analysis_qq(Bitcoin.2017.zoo, 'QQ Plot')
```

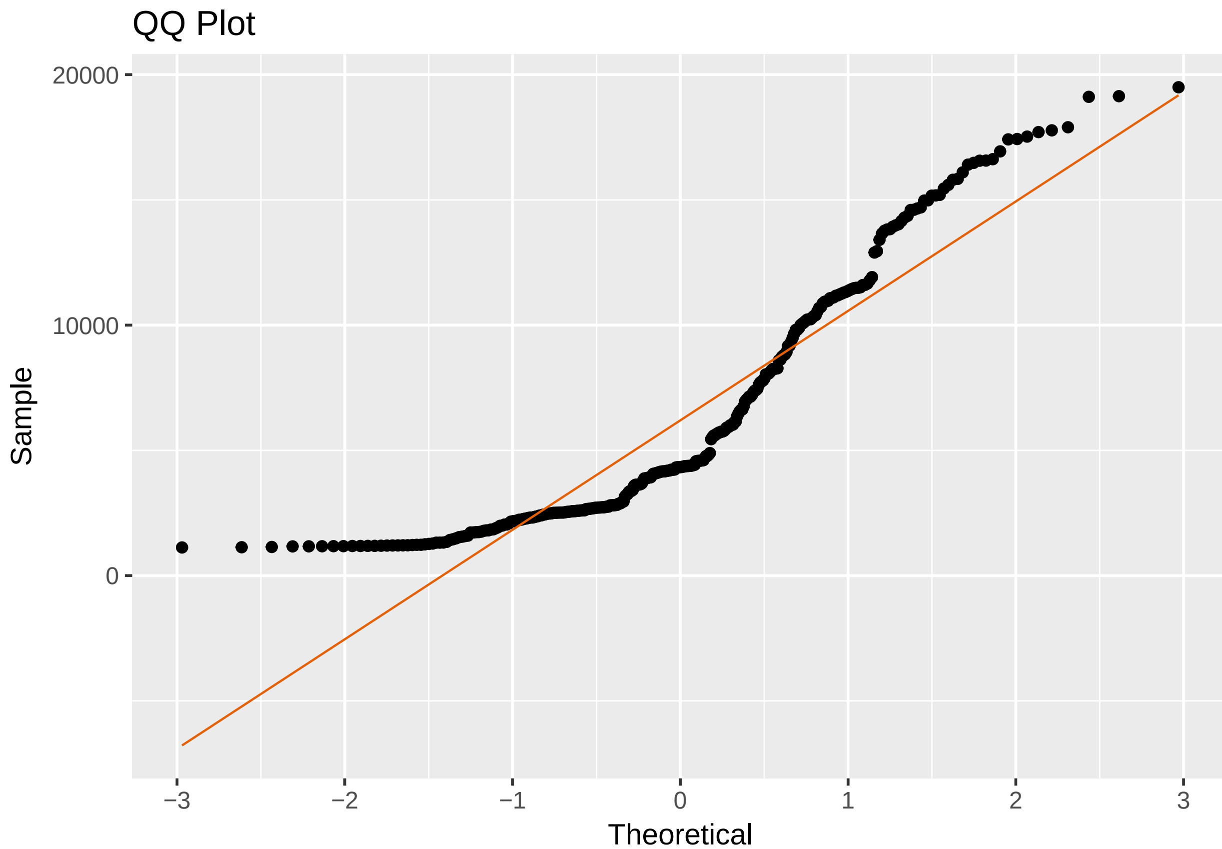

Figure 17: McLeod Li Test Statistics for Bitcoin

4 Heteroscedasticity Models

```
McLeod.Li.test(y=Bitcoin.2017.zoo,main="McLeod-Li Test Statistics for Daily Google Returns")
```

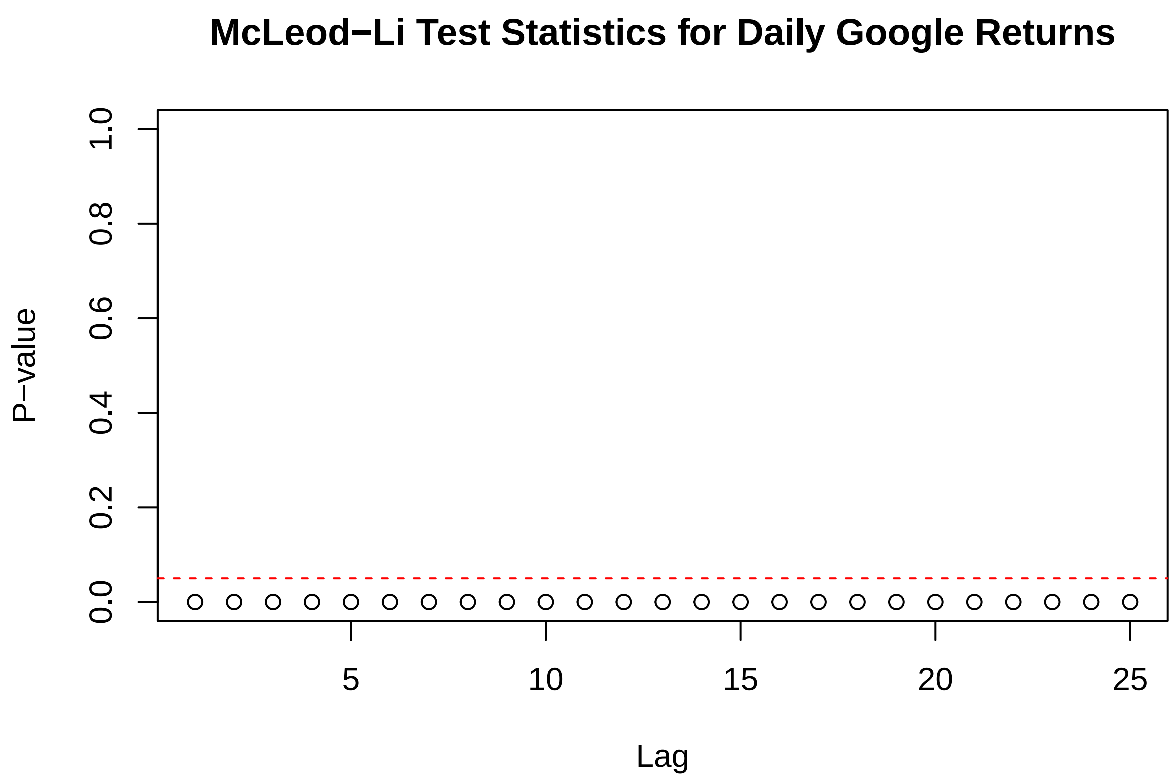

Figure 18: McLeod Li Test Statistics for Daily Google Returns

Figure 18 shows McLeod-Li test is significnat at 5% level of significance for all lags. This gives a strong idea about existence of volatiliy clustering.

```
#So we'll use absolute value and square transformations to figure out this ARCH effect.
abs.bitcoin = abs(Bitcoin.2017.zoo)
sq.bitcoin = Bitcoin.2017.zoo^2

par(mfrow=c(1,2))
acf(abs.bitcoin, ci.type="ma",main="The sample ACF plot for absolute return series")
pacf(abs.bitcoin, main="The sample PACF plot for absolute return series")
```

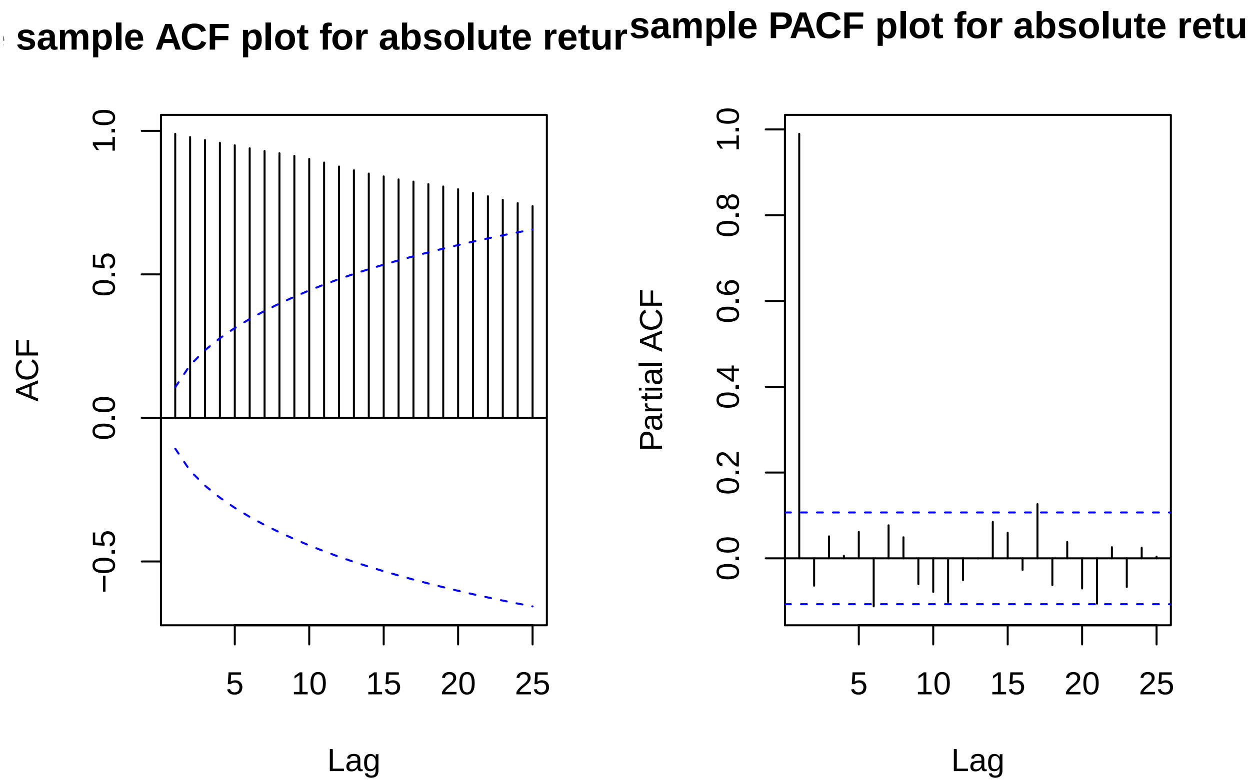

Figure 19: The sample ACF PACFplot for absolute return series

Figure 19 shows an absolute value transformation, we observe many significant lags in both ACF and PACF. EACF do not suggest an ARMA(0,0)

EACF, we can identify ARMA(1,0), ARMA(1,1), and ARMA(2,1) models for absolute value series. Proposed GARCH models are GARCH(0,1), GARCH(1,1), GARCH(1,2).

```
eacf(abs.bitcoin)
```

```
## 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  x  x
## 1 o o o o x o o o o x o  o  x  o
## 2 x o o o o x o o o x o  o  o  o
## 3 o x o o o o o o o x o  o  o  o
## 4 o x x o o o o o o x o  o  o  o
## 5 x x x o o o o o o x o  o  o  o
## 6 x x x o o o o o o x o  o  o  o
## 7 x x x o o o o o o o o  o  o  o
```

5 After the absolute value transformation, we boserve many signficicant lags in

6 both ACF and PACF. Also, EACF do not suggest an ARMA(0,0) model.

7 From the EACF, we can identify ARMA(1,0), ARMA(1,1), and ARMA(2,1) models for absolute

8 value series.

9 These models correspond to parameter settings of [max(1,1),1], [max(1,2),1] and [max(2,2),2].

10 So the corresponding tentative GARCH models are GARCH(0,1), GARCH(1,1), GARCH(1,2).

```r
par(mfrow=c(1,2))
acf(sq.bitcoin, ci.type="ma",main="The sample ACF plot for squared return series")
pacf(sq.bitcoin, main="The sample PACF plot for squared return series")
```

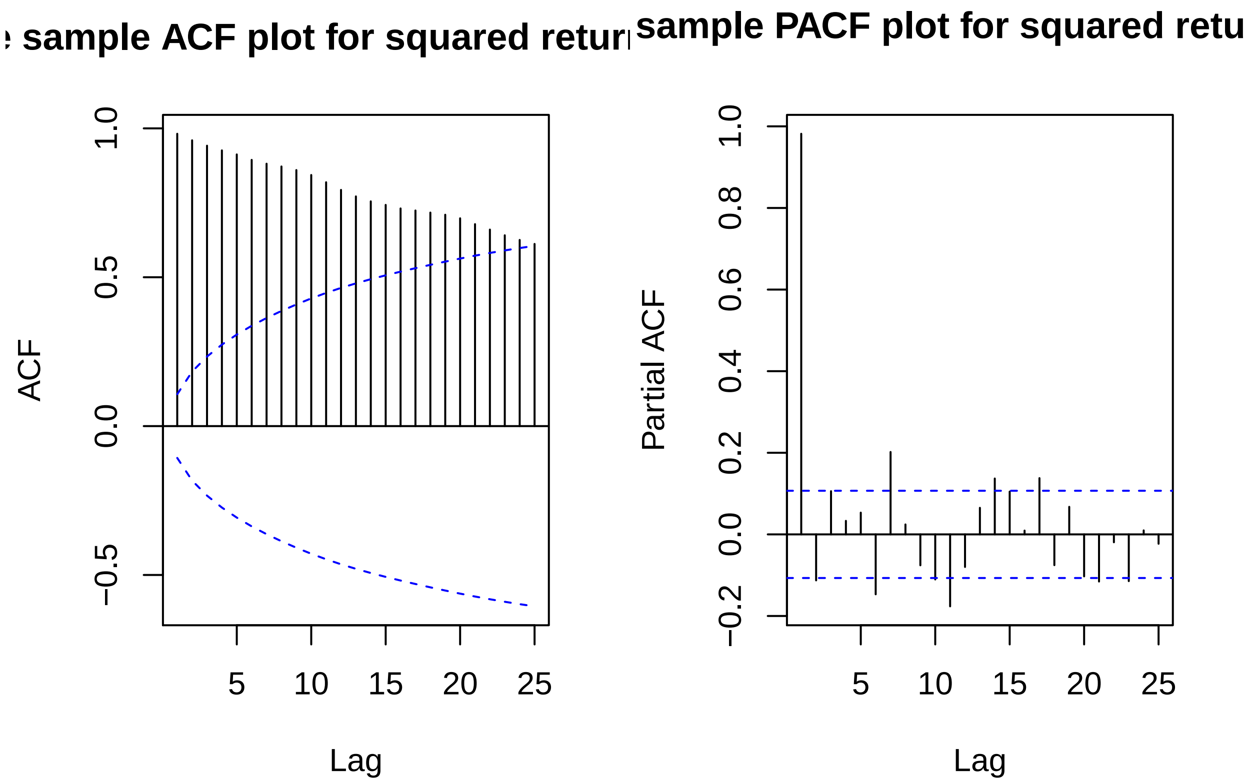

Figure 20: The sample ACF and PACF plot for squared return series

```r
eacf(sq.bitcoin)
```

```
## 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  x  x
## 1 x o o o x x o o x x o  x  x  x
## 2 x x o o o x x o o x o  o  o  x
## 3 x x o o o x o o o x o  o  o  o
## 4 x o x o o x o x o x o  o  o  o
## 5 x x o x o o o o o x o  o  o  o
## 6 x x o x o o o x o x o  o  o  o
## 7 o x x x x x x o o o o  o  o  o
```

11 After the square transformation, we boserve many signficicant lags in both ACF and PACF. Also, EACF do not suggest an ARMA(0,0) model.

12 From the EACF, we can identify ARMA(1,1), ARMA(1,2), and ARMA(2,2) models for squared series.

13 These models correspond to parameter settings of [max(1,1),1], [max(1,2),1], [max(1,2),2], and [max(2,2),2]. So the corresponding

14 tentative GARCH models are GARCH(1,1), GARCH(2,1), GARCH(2,2).

```
m.11 = garch(Bitcoin.2017.zoo,order=c(1,1),trace = FALSE)
summary(m.11) # All the coefficients are significant at 5% level of significance.
```

```
##
## Call:
## garch(x = Bitcoin.2017.zoo, order = c(1, 1), trace = FALSE)
##
## Model:
## GARCH(1,1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## 0.2458 0.4968 0.7083 0.8982 1.2064
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.969e+07          NA       NA       NA
## a1 9.815e-01          NA       NA       NA
## b1 4.590e-08          NA       NA       NA
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 18.774, df = 2, p-value = 8.381e-05
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 302.55, df = 1, p-value < 2.2e-16
```

```
m.11_2 = garchFit(formula = ~garch(1,1), data =Bitcoin.2017.zoo )
```

```
##
```

```
## Series Initialization:
##  ARMA Model:                arma
##  Formula Mean:              ~ arma(0, 0)
##  GARCH Model:               garch
##  Formula Variance:          ~ garch(1, 1)
##  ARMA Order:                0 0
##  Max ARMA Order:            0
##  GARCH Order:               1 1
##  Max GARCH Order:           1
##  Maximum Order:             1
##  Conditional Dist:          norm
##  h.start:                   2
##  llh.start:                 1
##  Length of Series:          336
##  Recursion Init:            mci
##  Series Scale:              4677.035
##
## Parameter Initialization:
##  Initial Parameters:          $params
##  Limits of Transformations:   $U, $V
##  Which Parameters are Fixed?  $includes
##  Parameter Matrix:
##                      U         V   params includes
##     mu     -13.25374276  13.25374 1.325374     TRUE
##     omega    0.00000100 100.00000 0.100000     TRUE
##     alpha1   0.00000001   1.00000 0.100000     TRUE
##     gamma1  -0.99999999   1.00000 0.100000    FALSE
##     beta1    0.00000001   1.00000 0.800000     TRUE
##     delta    0.00000000   2.00000 2.000000    FALSE
##     skew     0.10000000  10.00000 1.000000    FALSE
##     shape    1.00000000  10.00000 4.000000    FALSE
##  Index List of Parameters to be Optimized:
##     mu  omega alpha1  beta1
##      1      2      3      5
##  Persistence:                  0.9
##
##
## --- START OF TRACE ---
## Selected Algorithm: nlminb
##
## R coded nlminb Solver:
##
##   0:     414.12432:  1.32537 0.100000 0.100000 0.800000
##   1:     395.93293:  1.31047 0.0659485 0.101483 0.781411
##   2:     233.09559: 0.811091 1.00000e-06 0.386126 0.513406
##   3:     228.28802: 0.811068 0.00278674 0.386136 0.513414
##   4:     228.09216: 0.810228 0.00267505 0.387047 0.512580
##   5:     226.17230: 0.815568 0.00128314 0.384634 0.515234
##   6:     226.12595: 0.815611 0.000874644 0.384694 0.515272
##   7:     226.07325: 0.815975 0.00108646 0.384569 0.515464
##   8:     226.05615: 0.816396 0.00101887 0.384432 0.515684
##   9:     226.02974: 0.817252 0.00112365 0.384227 0.516156
##  10:     225.99259: 0.818917 0.00102218 0.383500 0.516962
##  11:     225.95230: 0.822303 0.00115447 0.382071 0.518538
```

```
##  12:     225.84830: 0.830744 0.000983166 0.380929 0.520082
##  13:     225.65941: 0.827567 0.00117393 0.385684 0.516095
##  14:     225.24016: 0.800259 0.000796139 0.424438 0.483201
##  15:     224.76096: 0.808031 0.00226237 0.467075 0.449230
##  16:     222.71183: 0.816061 0.000740948 0.509530 0.415067
##  17:     222.15583: 0.823177 0.00191793 0.551904 0.380675
##  18:     220.80218: 0.837996 0.00167042 0.639493 0.310151
##  19:     218.20747: 0.869164 0.00215705 0.724541 0.239552
##  20:     216.01827: 0.881098 0.00181985 0.754890 0.213999
##  21:     214.76116: 0.881180 0.000740494 0.754893 0.213994
##  22:     214.73188: 0.881593 0.00115532 0.755620 0.213385
##  23:     214.52256: 0.881788 0.000945344 0.755984 0.213079
##  24:     214.47259: 0.882076 0.000800222 0.756347 0.212776
##  25:     214.39931: 0.882293 0.000903503 0.756734 0.212451
##  26:     211.73937: 0.895468 0.000502912 0.789999 0.184468
##  27:     211.62683: 0.895493 0.000730736 0.790000 0.184468
##  28:     211.57356: 0.895550 0.000665235 0.790158 0.184322
##  29:     211.55884: 0.895622 0.000624245 0.790318 0.184175
##  30:     210.18553: 0.912225 0.000431817 0.848511 0.130412
##  31:     208.35499: 0.903062 0.000765140 0.983141 1.00000e-08
##  32:     208.35475: 0.909066 0.000371878  1.00000 1.00000e-08
##  33:     208.08405: 0.904282 0.000685178  1.00000 1.00000e-08
##  34:     207.99150: 0.905788 0.000610466  1.00000 1.00000e-08
##  35:     207.97768: 0.906003 0.000528316  1.00000 1.00000e-08
##  36:     207.97047: 0.906213 0.000558753  1.00000 1.00000e-08
##  37:     207.97029: 0.906127 0.000556062  1.00000 1.00000e-08
##  38:     207.97029: 0.906147 0.000555322  1.00000 1.00000e-08
##  39:     207.97029: 0.906143 0.000555482  1.00000 1.00000e-08
##  40:     207.97029: 0.906143 0.000555482  1.00000 1.00000e-08
##
## Final Estimate of the Negative LLH:
##  LLH:  3047.311    norm LLH:  9.069379
##           mu        omega       alpha1        beta1
## 4.238061e+03 1.215098e+04 1.000000e+00 1.000000e-08
##
## R-optimhess Difference Approximated Hessian Matrix:
##                   mu         omega         alpha1         beta1
## mu     -0.0010267428 -2.306595e-04 -9.495482e-03 -1.263897e-01
## omega  -0.0002306595 -3.139387e-08 -1.167578e-02  5.037773e-03
## alpha1 -0.0094954818 -1.167578e-02 -1.651509e+02 -1.757162e+02
## beta1  -0.1263897442  5.037773e-03 -1.757162e+02 -4.305337e+02
## attr(,"time")
## Time difference of 0.009865046 secs
##
## --- END OF TRACE ---
##
##
## Time to Estimate Parameters:
##  Time difference of 0.05921984 secs
```

```r
summary(m.11_2)
```

```
##
## Title:
##  GARCH Modelling
```

```
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = Bitcoin.2017.zoo)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0x000000001f14fae8>
##  [data = Bitcoin.2017.zoo]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1       beta1
## 4.2381e+03  1.2151e+04  1.0000e+00  1.0000e-08
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     4.238e+03   5.899e+00   718.42   <2e-16 ***
## omega  1.215e+04          NA       NA       NA
## alpha1 1.000e+00   9.908e-02    10.09   <2e-16 ***
## beta1  1.000e-08   6.125e-02     0.00        1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -3047.311    normalized:  -9.069379
##
## Description:
##  Fri May 25 23:33:42 2018 by user: Napatr
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  36.47852  1.19892e-08
##  Shapiro-Wilk Test  R    W      0.8008819 0
##  Ljung-Box Test     R    Q(10)  2450.448  0
##  Ljung-Box Test     R    Q(15)  3329.88   0
##  Ljung-Box Test     R    Q(20)  4010.308  0
##  Ljung-Box Test     R^2  Q(10)  68.1558   1.005253e-10
##  Ljung-Box Test     R^2  Q(15)  74.25125  7.728455e-10
##  Ljung-Box Test     R^2  Q(20)  83.69672  9.175625e-10
##  LM Arch Test       R    TR^2   60.79107  1.618925e-08
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 18.16257 18.20801 18.16229 18.18068
```

```r
m.12 = garch(Bitcoin.2017.zoo,order=c(1,2),trace = FALSE)
summary(m.12)# All the coefficients but aplha_2 are significant at 5% level of significance.
```

```
##
```

```
## Call:
## garch(x = Bitcoin.2017.zoo, order = c(1, 2), trace = FALSE)
##
## Model:
## GARCH(1,2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## 0.2490 0.4838 0.6508 0.7765 1.1059
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.859e+07          NA       NA       NA
## a1 7.087e-01          NA       NA       NA
## a2 6.945e-01          NA       NA       NA
## b1 1.771e-07          NA       NA       NA
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 15.797, df = 2, p-value = 0.0003713
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 299.33, df = 1, p-value < 2.2e-16
```

```
m.12_2 = garchFit(formula = ~garch(2,1), data =Bitcoin.2017.zoo, trace = FALSE )
summary(m.12_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 1), data = Bitcoin.2017.zoo, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(2, 1)
## <environment: 0x0000000023d6de28>
##  [data = Bitcoin.2017.zoo]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2       beta1
## 2.5750e+03  5.3452e+03  1.0000e+00  3.1818e-02  1.2038e-03
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
```

```
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     2.575e+03   1.379e+01  186.761   <2e-16 ***
## omega  5.345e+03          NA       NA       NA
## alpha1 1.000e+00   1.202e-01    8.317   <2e-16 ***
## alpha2 3.182e-02   1.209e-01    0.263    0.792
## beta1  1.204e-03   7.531e-02    0.016    0.987
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -2989.655    normalized:  -8.897782
##
## Description:
##  Fri May 25 23:33:42 2018 by user: Napatr
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  58.98889  1.550982e-13
##  Shapiro-Wilk Test  R    W      0.7675237 0
##  Ljung-Box Test     R    Q(10)  1945.51   0
##  Ljung-Box Test     R    Q(15)  2684.468  0
##  Ljung-Box Test     R    Q(20)  3201.47   0
##  Ljung-Box Test     R^2  Q(10)  4.552476  0.9190026
##  Ljung-Box Test     R^2  Q(15)  35.54104  0.002058026
##  Ljung-Box Test     R^2  Q(20)  48.24354  0.0003931735
##  LM Arch Test       R    TR^2   11.52186  0.4848018
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 17.82533 17.88213 17.82489 17.84797
```

```
m.22 = garch(Bitcoin.2017.zoo,order=c(2,2),trace = FALSE)
summary(m.22) # Higher order parameters are insignificant
```

```
##
## Call:
## garch(x = Bitcoin.2017.zoo, order = c(2, 2), trace = FALSE)
##
## Model:
## GARCH(2,2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## 0.2564 0.4932 0.6589 0.7803 1.1086
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.750e+07          NA       NA       NA
## a1 7.072e-01          NA       NA       NA
## a2 6.931e-01          NA       NA       NA
## b1 1.661e-03          NA       NA       NA
## b2 1.168e-07          NA       NA       NA
##
## Diagnostic Tests:
```

```
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 15.913, df = 2, p-value = 0.0003504
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 298.35, df = 1, p-value < 2.2e-16
```

```
m.22_2 = garchFit(formula = ~garch(2,2), data =Bitcoin.2017.zoo, trace = FALSE, cond.dist = "QMLE" )
summary(m.22_2)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 2), data = Bitcoin.2017.zoo, cond.dist = "QMLE",
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(2, 2)
## <environment: 0x0000000025e981a0>
##  [data = Bitcoin.2017.zoo]
##
## Conditional Distribution:
##  QMLE
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2       beta1       beta2
## 3.7323e+03  3.3504e+04  9.0427e-01  9.7561e-02  1.0000e-08  1.0000e-08
##
## Std. Errors:
##  robust
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     3.732e+03   1.991e+02   18.744   <2e-16 ***
## omega  3.350e+04   2.275e+03   14.728   <2e-16 ***
## alpha1 9.043e-01   3.782e-01    2.391   0.0168 *
## alpha2 9.756e-02   3.116e-01    0.313   0.7542
## beta1  1.000e-08   2.205e-01    0.000   1.0000
## beta2  1.000e-08   3.022e-01    0.000   1.0000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -3059.749    normalized:  -9.106397
##
## Description:
##  Fri May 25 23:33:42 2018 by user: Napatr
##
##
```

```
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  47.53842  4.755152e-11
##  Shapiro-Wilk Test  R    W      0.7674797 0
##  Ljung-Box Test     R    Q(10)  2714.587  0
##  Ljung-Box Test     R    Q(15)  3840.044  0
##  Ljung-Box Test     R    Q(20)  4839.897  0
##  Ljung-Box Test     R^2  Q(10)  11.28675  0.3356185
##  Ljung-Box Test     R^2  Q(15)  39.93851  0.0004632923
##  Ljung-Box Test     R^2  Q(20)  43.65357  0.00167418
##  LM Arch Test       R    TR^2   12.60656  0.3982738
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 18.24851 18.31667 18.24789 18.27568
```

Conclusions:

Regression models are not suitable for daily bitcoin closing prices. ARIMA model is good, but the mean absolute scaled error of 3.2, (three times higher than original) is not practical for prediction. ARIMA models are not suitable as they are unable to capture high volatility in series. Combination of ARIMA (mean) and GARCH (variance) prediction model can be better for daily bitcoin closing prices.