Bitcoin Closing Prices Forecasting

MATH 1318 Time Series Analysis - Final Project

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

27 May 2018

Contents

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 (Wikipedia 2018). 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.

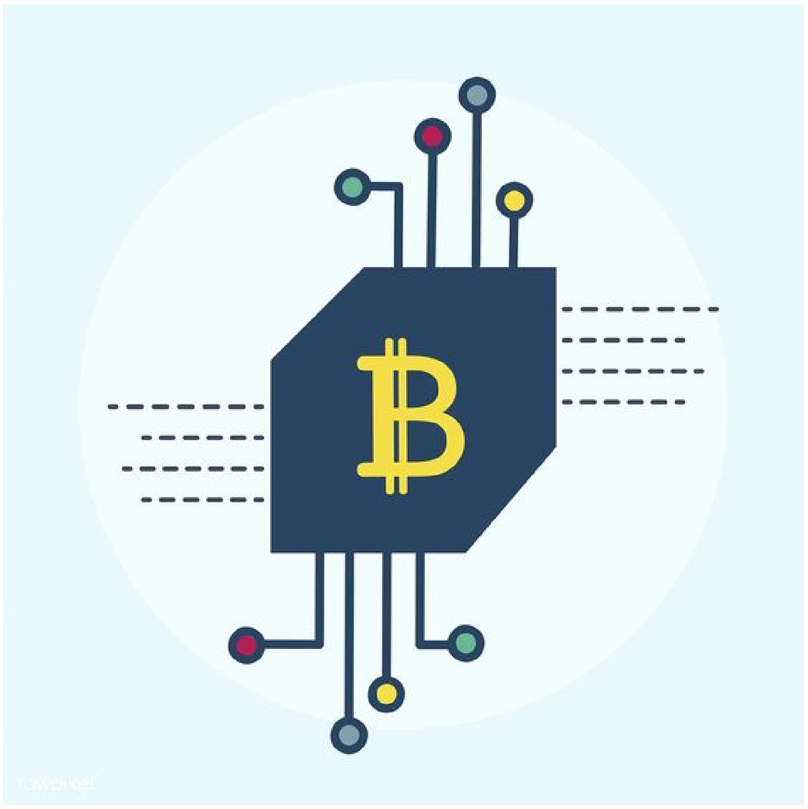


Figure 1: Bitcoin Illustration

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

Initial Diagnosis

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

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 2 shows the daily closing price of bitcoin from the 27th Apr 2013 to the 3rd Mar 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. Next we'll filter series to remove the initial flat trend.

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

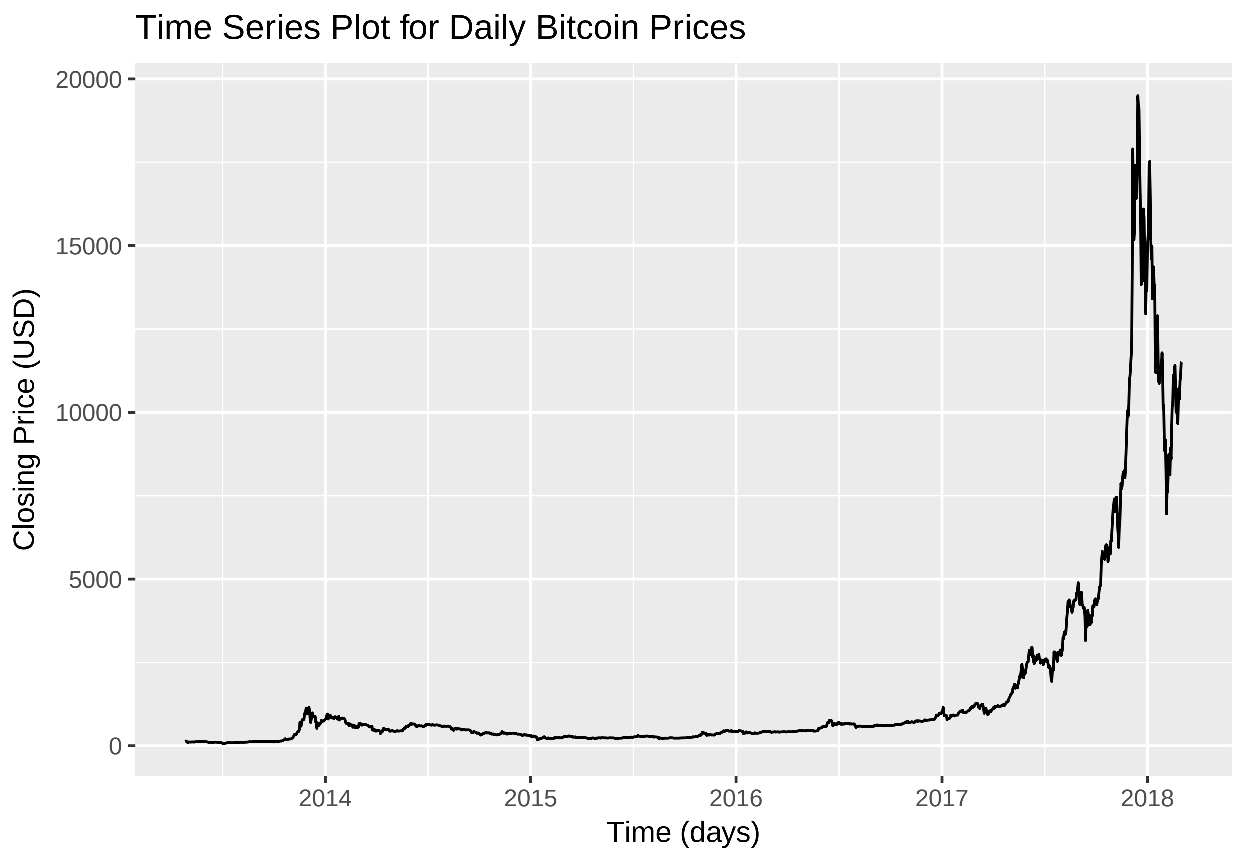


Figure 2: Time Series of Daily Bitcoin Prices

Subset Series

Figure 3 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-Apr-2017 to the 03-Mar-2018, given in USD. Features are similar to original series

```
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)")
```

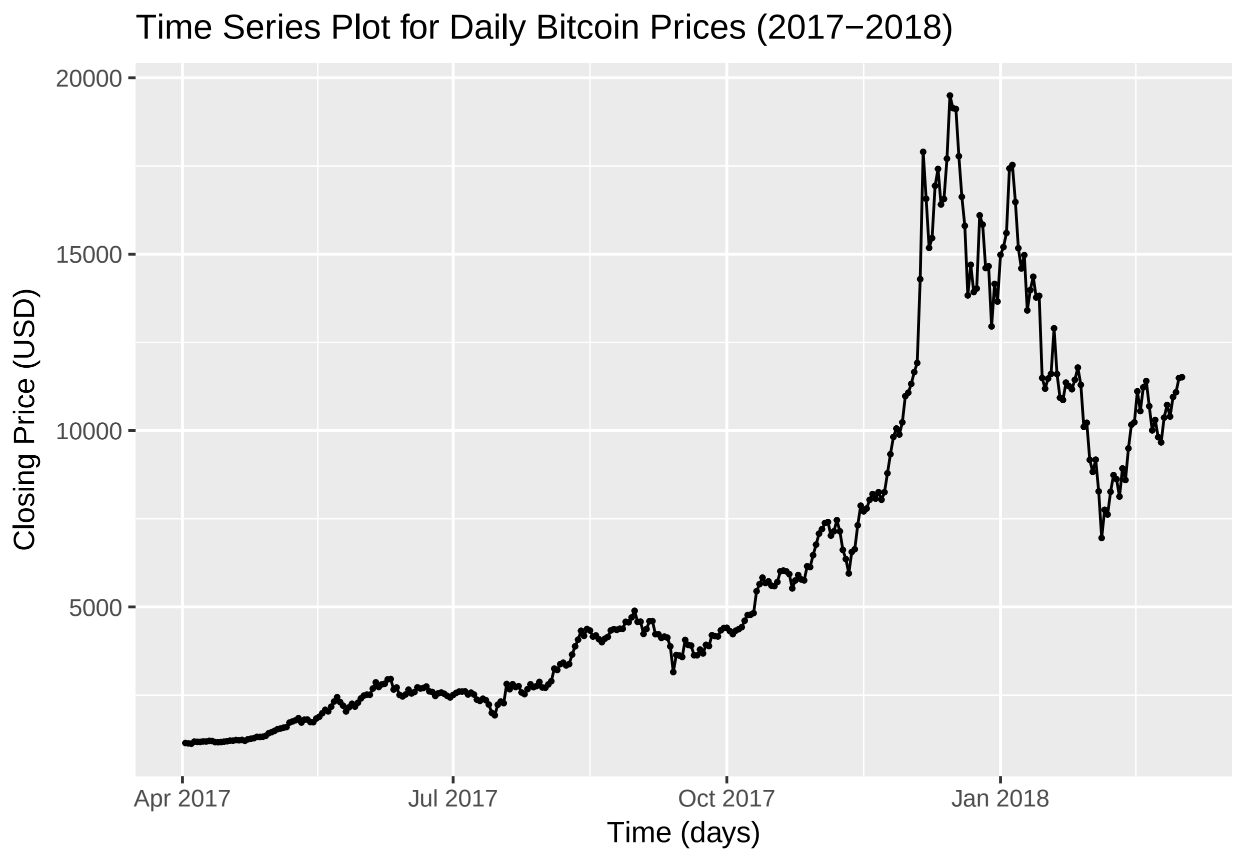


Figure 3: Subset Time Series of Daily Bitcoin Prices

Scatter Plot and correlation

Figure 4 shows the scatter plot of the relationship between consecutive points, to determine the presence of auto regressive behaviour. Correlation value 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.

```r
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")
```

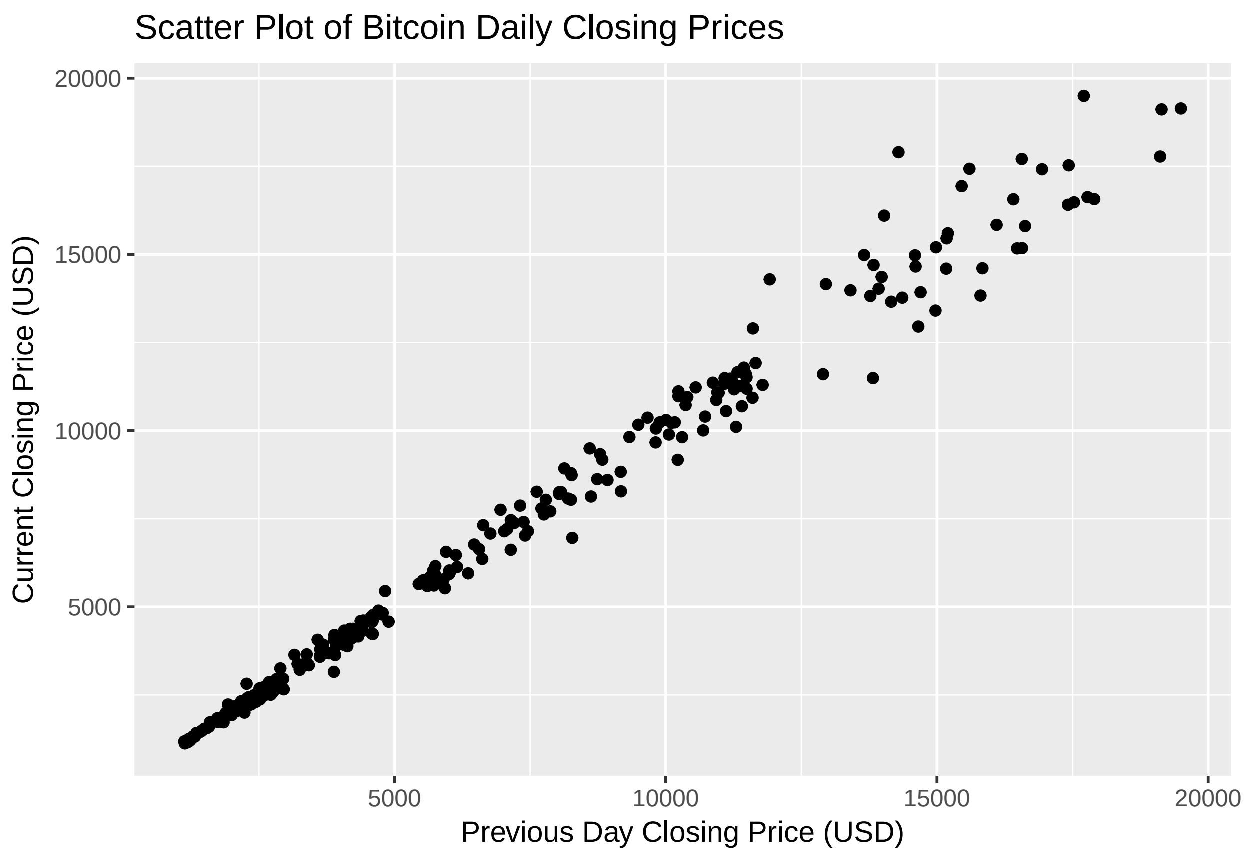


Figure 4: Scatter Plot of Bitcoin Daily Closing Prices

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

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

Regression Models

Here we'll try fitting linear and quadratic model in bitcoin closing prices. Data has no seasonality so harmonic model is not applicable.

Linear Model

```r
model.ln = lm(Bitcoin.2017.zoo~time(Bitcoin.2017.zoo))
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
```

Figure 5 shows the plots give the regression models for the linear are statistically significant, with the same *p-value* < 2.2e-16 and *R-squared values* 0.7119.

```
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')
```

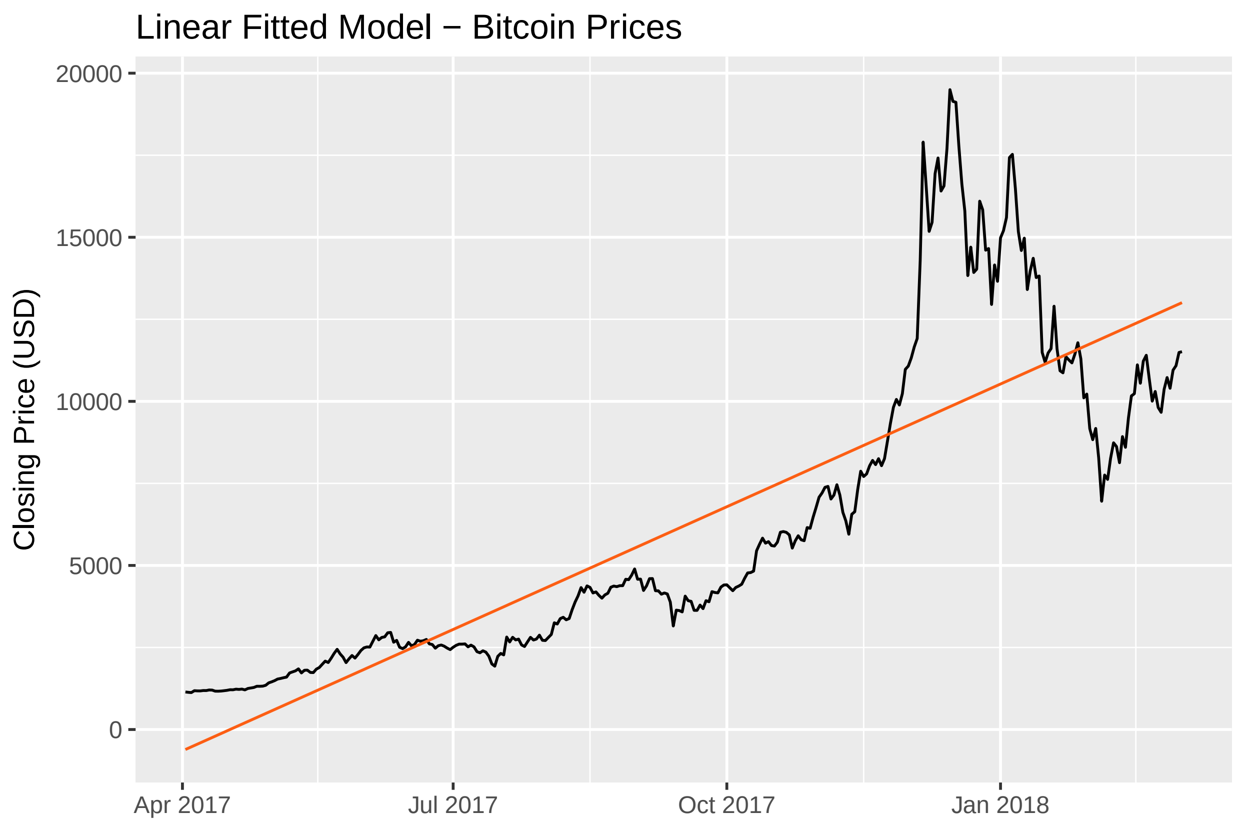


Figure 5: Bitcoin - Linear Fitted Model

Residual Analysis - Linear Model

Below are the findings of residuals from linear model:

- Figure 6 shows the time series plot, ACF, histogram of the standardized residuals for the linear model of the Bitcoin time series
- 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 in figure 7 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 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
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)
```

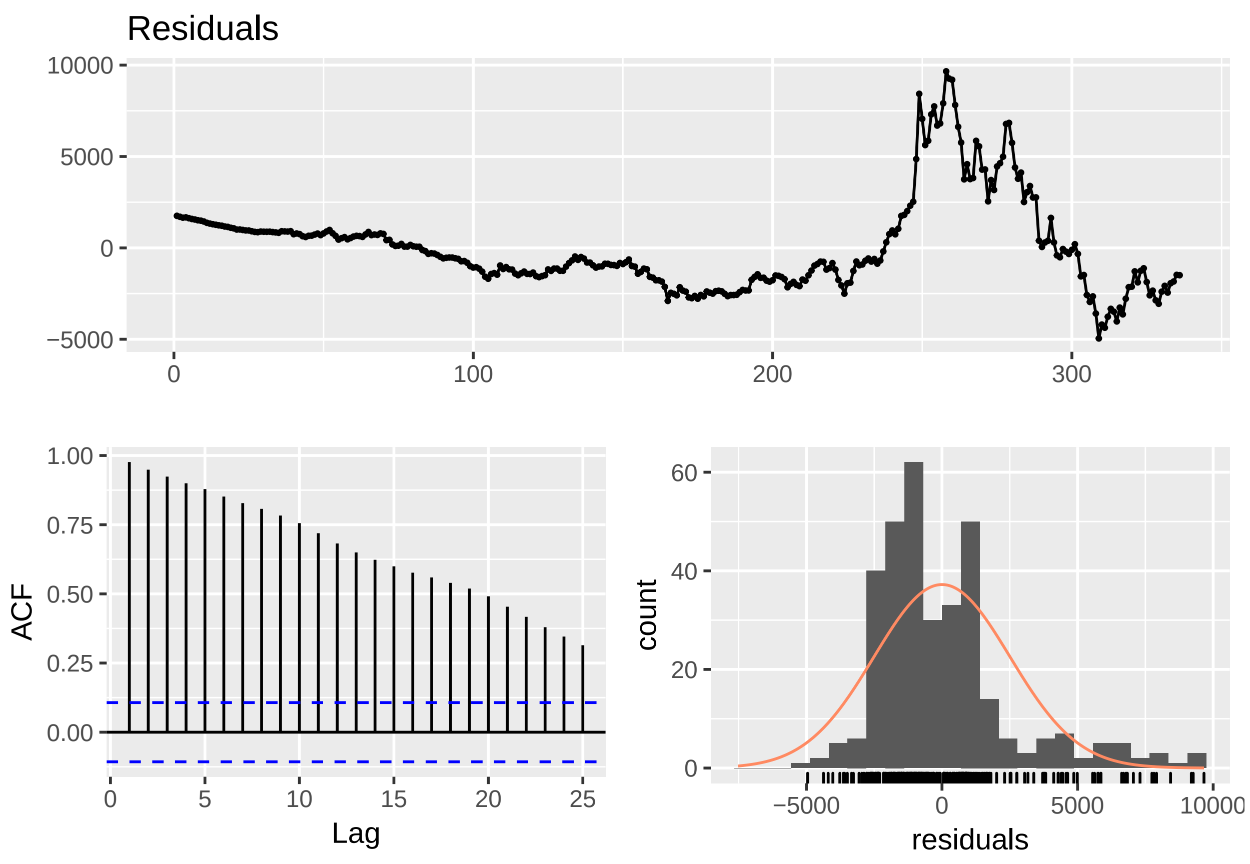


Figure 6: Residual Analysis Linear fitted Model

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

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

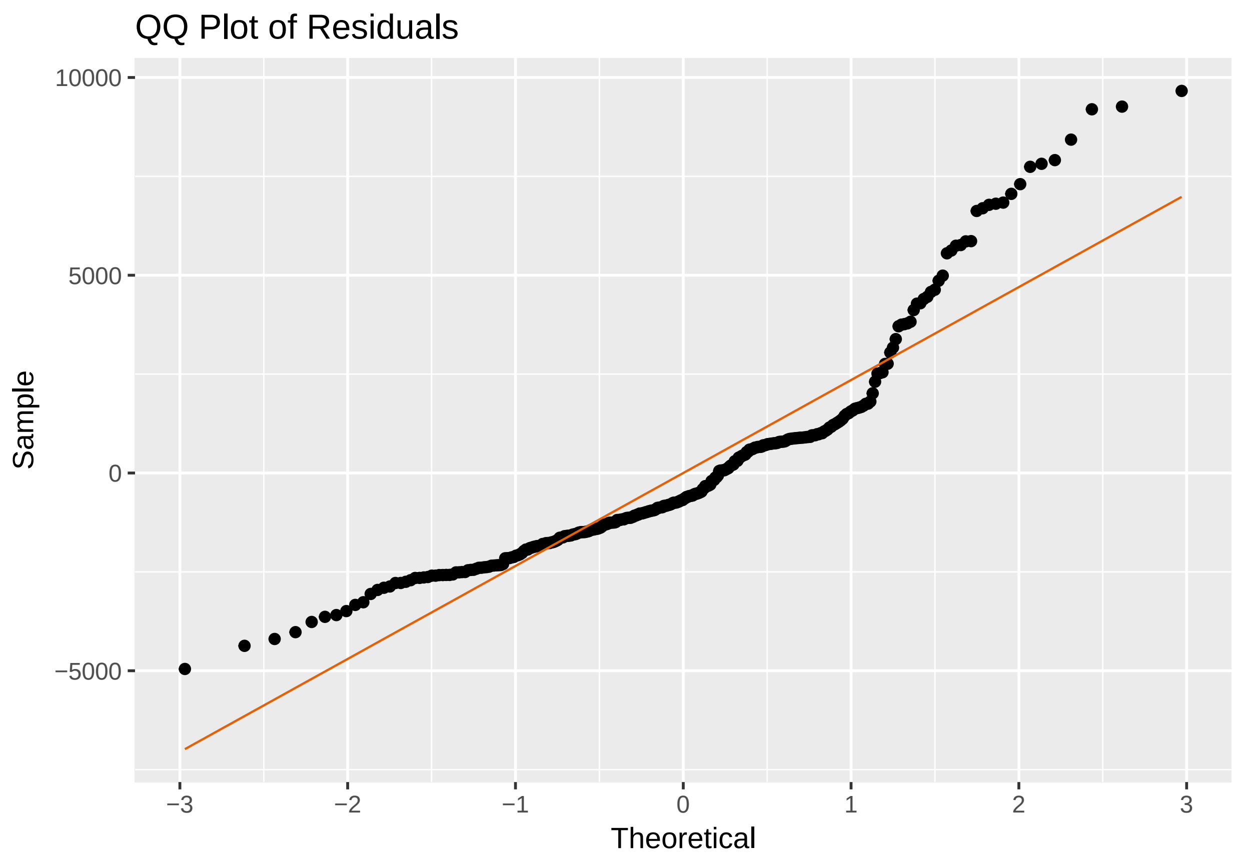


Figure 7: QQ plot Linear Model

```
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
```

Quadratic Model

```
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
```

Figure 8 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.

```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')
```

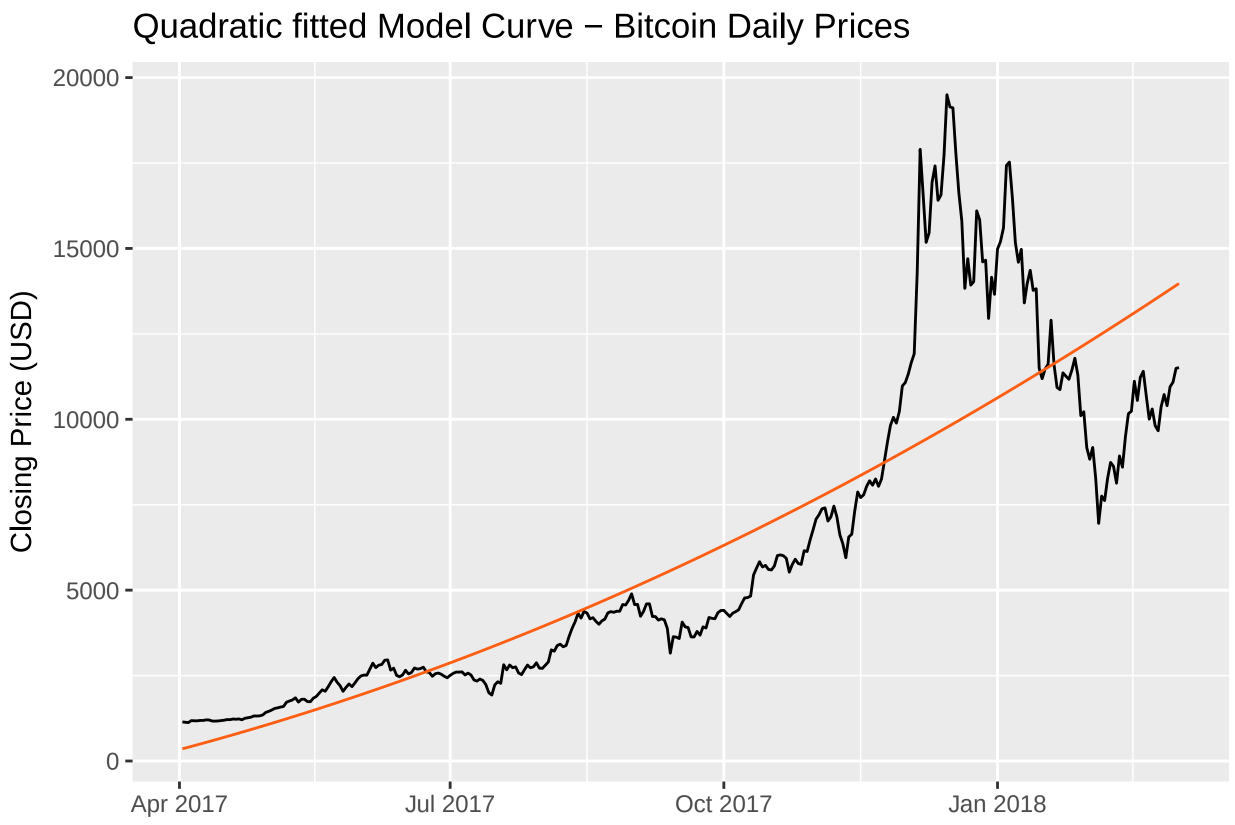


Figure 8: Quadratic fitted Model Curve

Residual Analysis - Quadratic Model

Below are the findings of residuals from Quadratic model:

- Figure 9 shows the time series plot, ACF, and histogram of the standardized residuals for the quadratic Model
- 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 in figure **??** 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 quadratic model does not pass the diagnostic checks, thus the quadratic model does not capture all the information in the time series and is not suitable for forecasting.

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

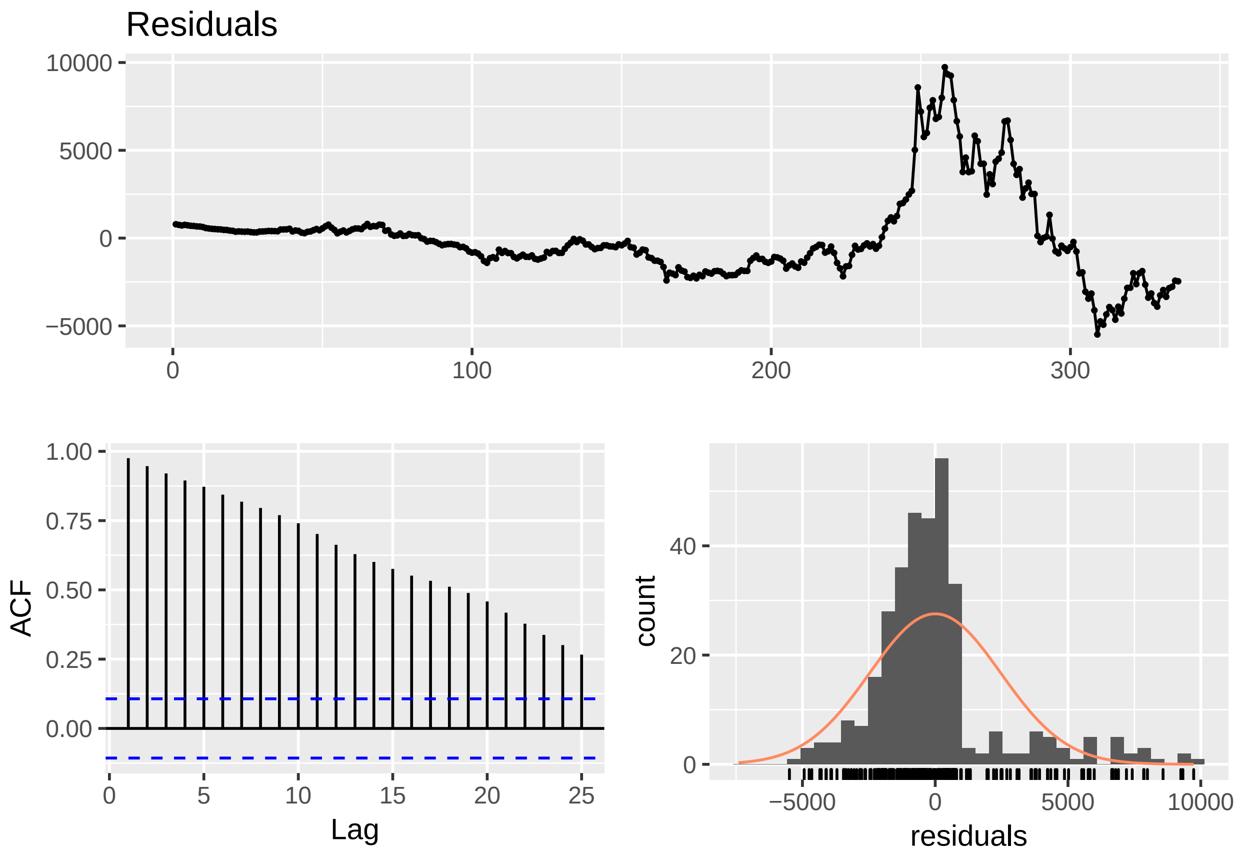
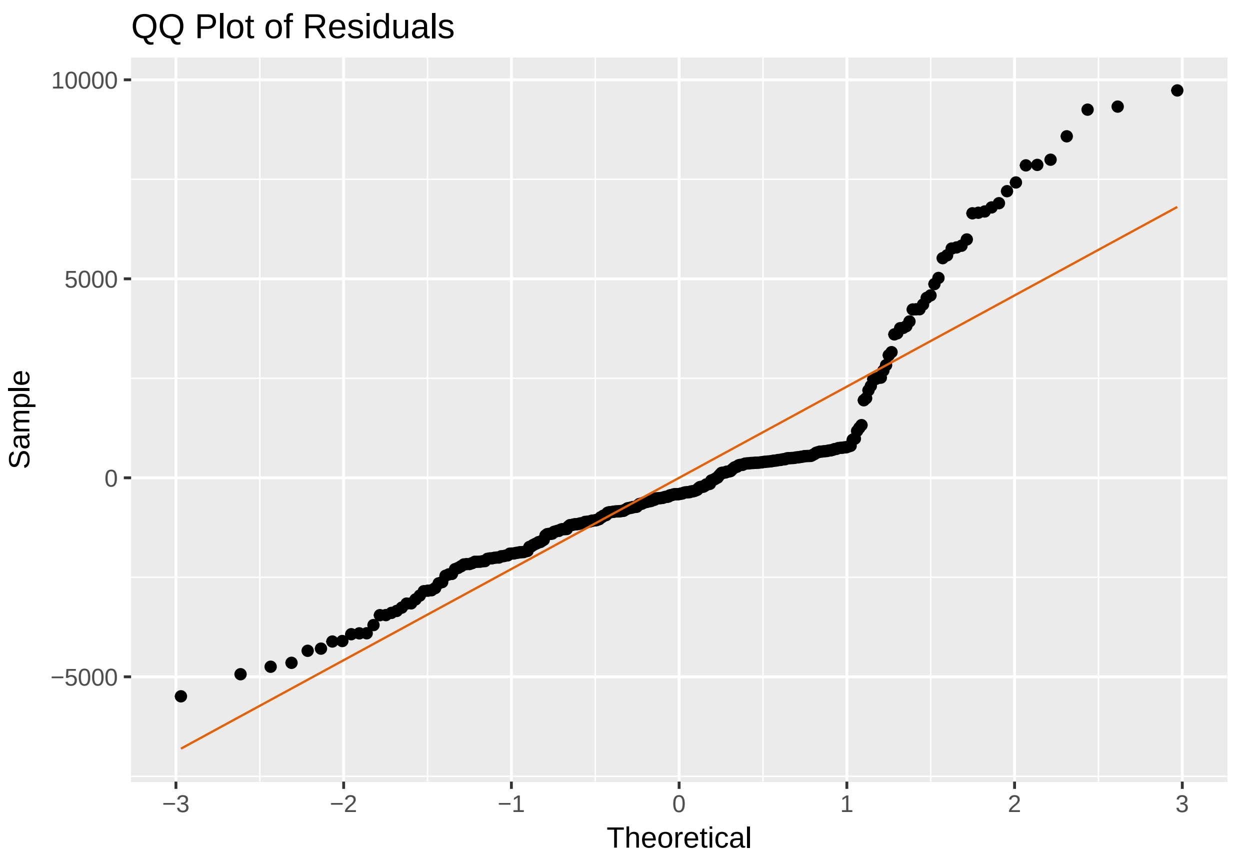


Figure 9: Residual Analysis Quadratic fitted Model

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

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



```
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
```

ARIMA Models

ARIMA Initial Diagnosis

We'll analyze ACF and PACF to identify AR and MA orders. Below are the results:

- Figure 10 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. *Trend suggests non stationarity.*
- 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.

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

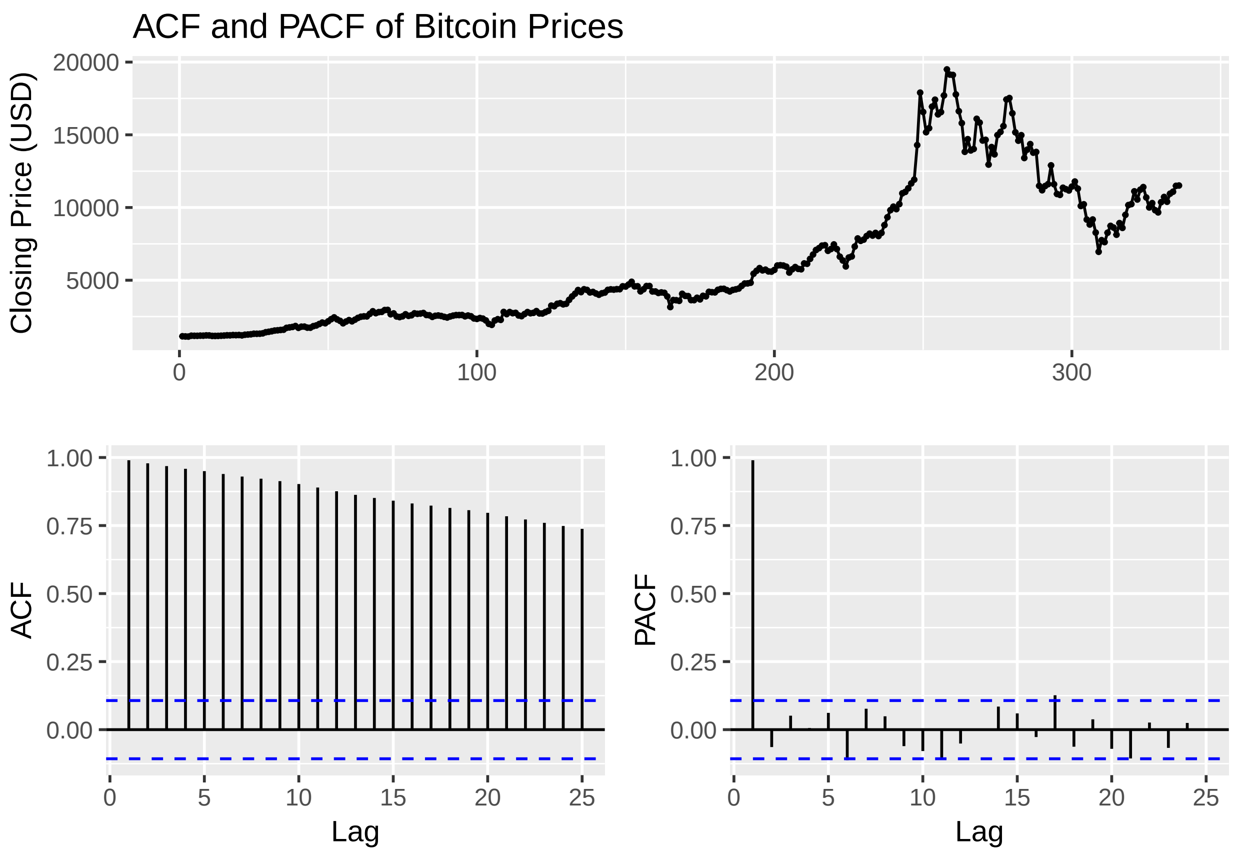


Figure 10: ACF and PACF of Bitcoin Prices

Transformation

Figure 11 shows calculation of the log-likelihood for each lambda value, using the yule-walker method gave a lambda value 0.1-0.2. log transformation is not the best transformation according to boxcox plot.

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

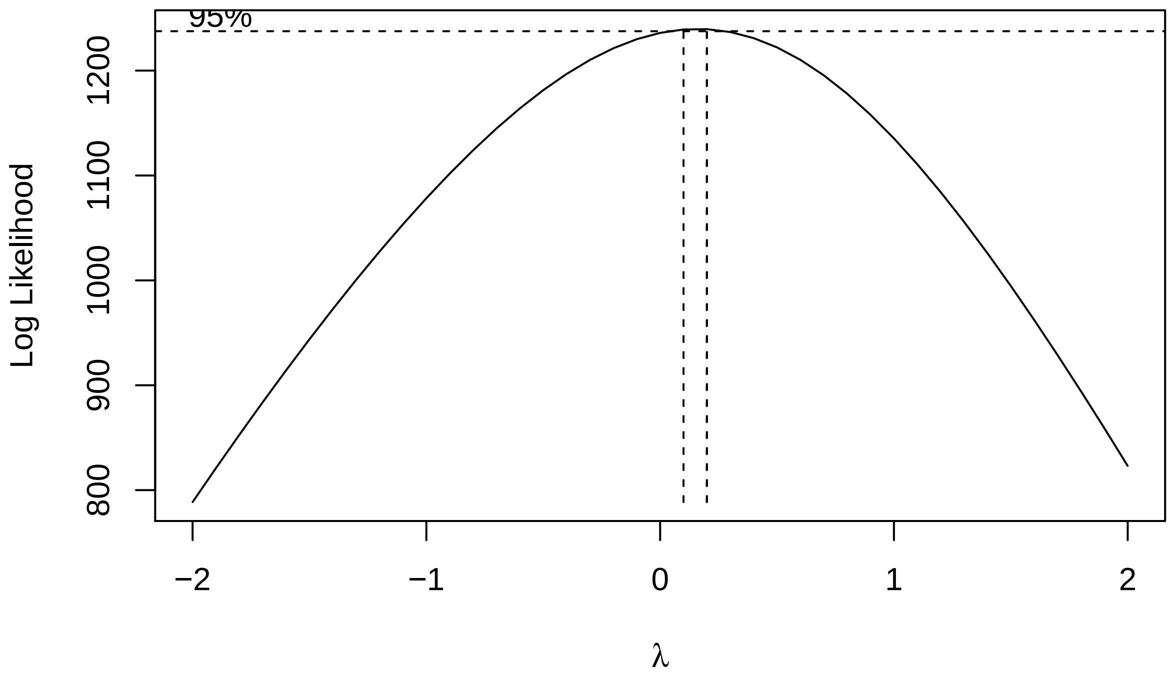


Figure 11: BoxCox Transformation

BoxCox transformation was applied to stabilize the time series. First differencing was applied to de-trend the time series. Figure 12 transformation and differencing.

```r
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')
```

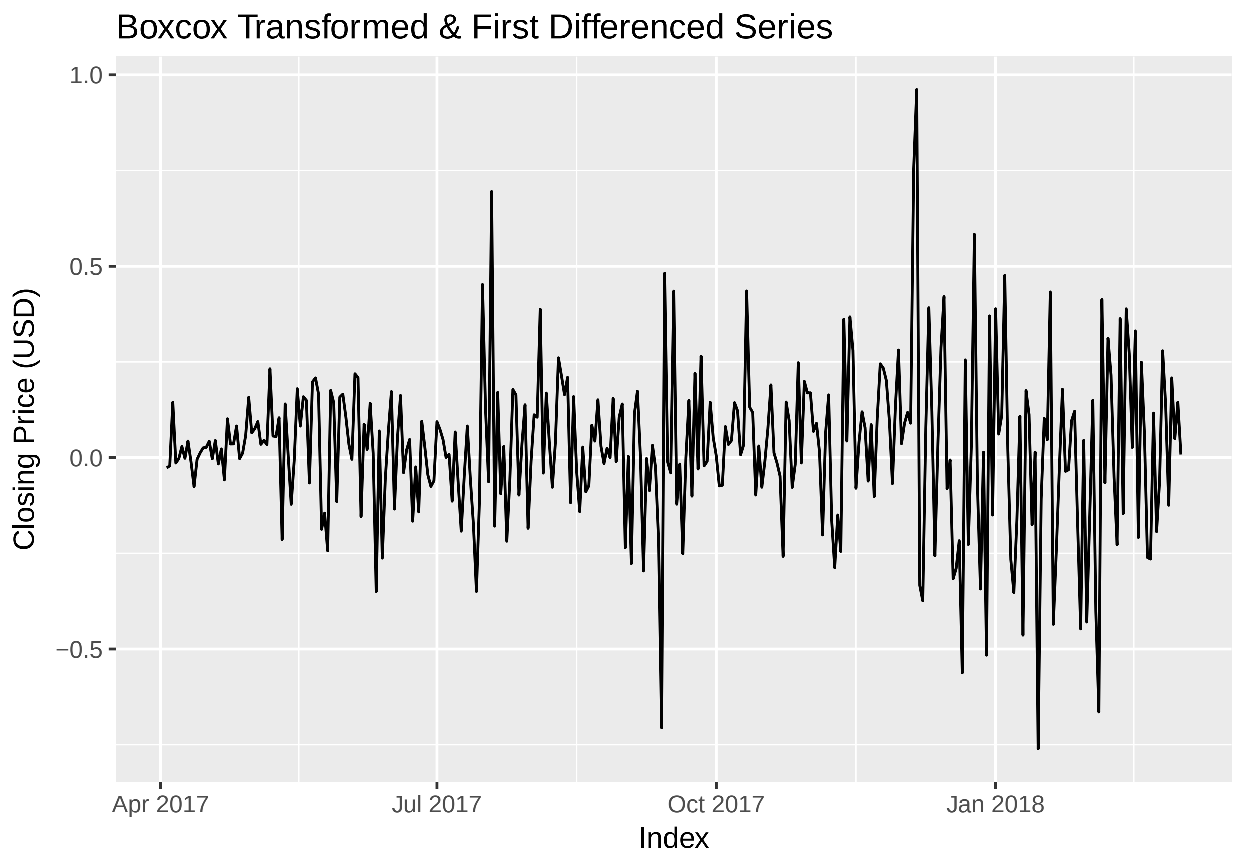


Figure 12: Boxcox Transformed & First Differenced Series

Diagnosis - Transformed Series

Figure 13 shows the results of transformed and differenced series.

- 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.
- 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.

ARIMA(4,1,2), ARMIA(3,1,2)} are included in the set of possible models

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

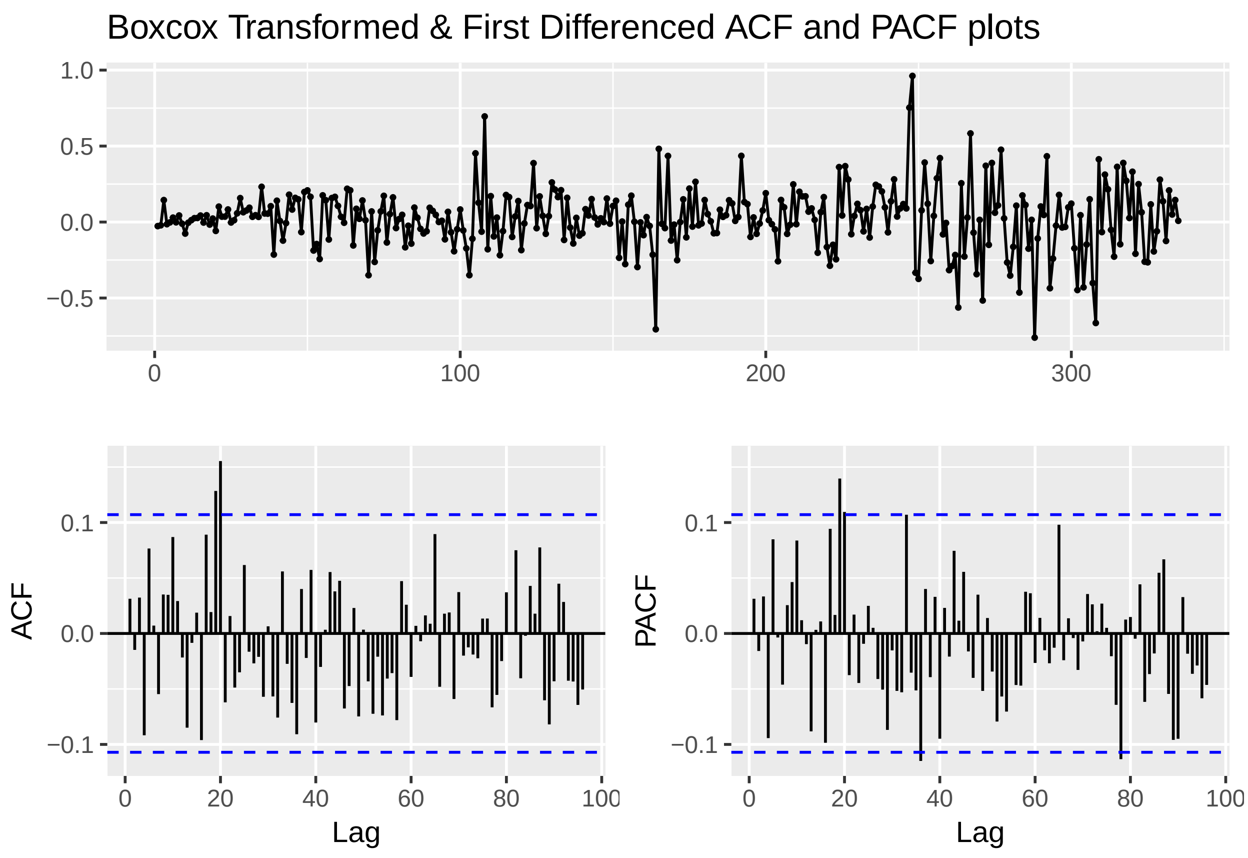


Figure 13: Boxcox Transformed & First Differenced ACF and PACF plots

```r
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
```

Extended Autocorrelation Function

The eacf matrix shown below, identified possible smaller ARIMA models with a p=1, 2 and q=1, 2.

```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
```

ARMA Subsets

Figure 14 shows the BIC plot, 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.

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

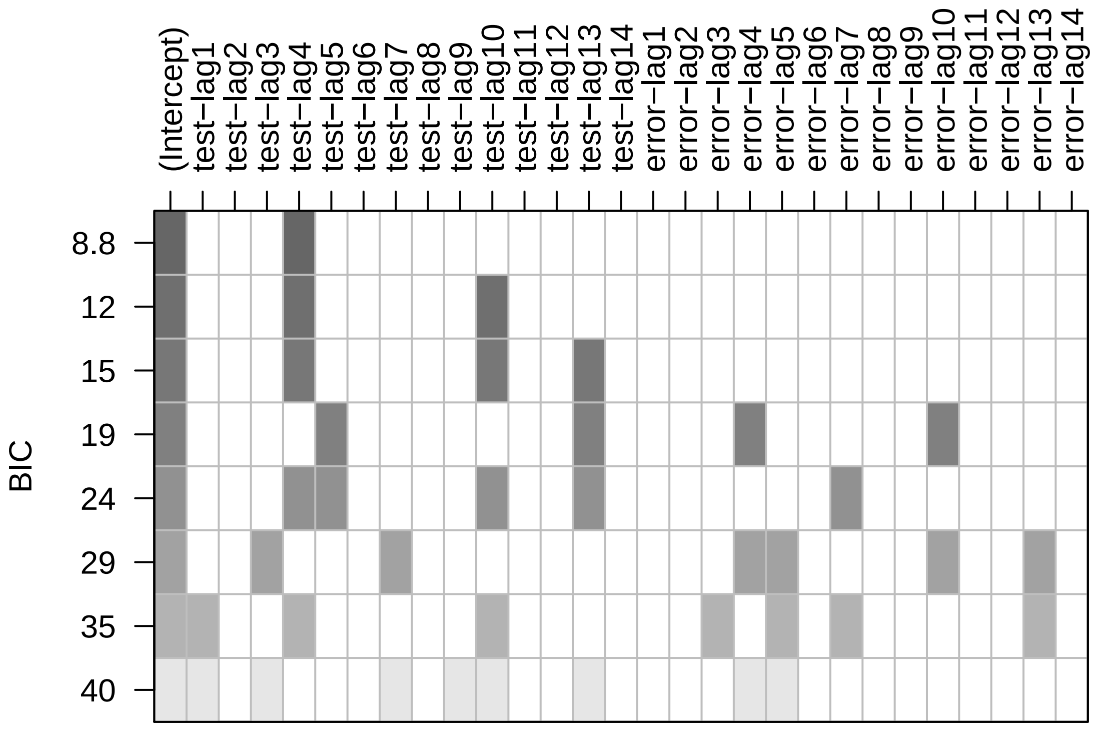


Figure 14: BIC Table

Model Fitting & Significance Tests

From the decided set of ARIMA models, we'll perform coefficient significance tests using maximum likelihood and conditional sum-of-squares methods.

```r
# 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
```

```r
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
```

```r
# 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
```

```r
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 -0.713306   0.267548 -2.6661 0.007674 **
## ma1  0.764098   0.268822  2.8424 0.004478 **
## ma2 -0.010152   0.061030 -0.1663 0.867889
## ---
## 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
```

```
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  0.0335948         NA      NA       NA
```

```
# 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 -0.028530   0.079410  -0.3593   0.7194
## ar2  0.906936   0.075863  11.9549   <2e-16 ***
## ma1  0.085848   0.084591   1.0149   0.3102
## ma2 -0.913597   0.083838 -10.8972   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
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 -0.0021222  0.0653257  -0.0325   0.9741
## ar2  0.9248274  0.0619597  14.9263   <2e-16 ***
## ma1  0.0590080  0.0774408   0.7620   0.4461
## ma2 -0.9409861  0.0773613 -12.1635   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# ARIMA(3,1,2)
model_312_css = arima(Bitcoin.boxcox,order=c(3,1,2),method='CSS')
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 0.084523 .
## 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
```

```
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 -1.612748   0.083945 -19.2120 < 2.2e-16 ***
## ar2 -0.854073   0.108337  -7.8835 3.183e-15 ***
## ar3  0.048182   0.058346   0.8258    0.4089
## ma1  1.679110   0.064674  25.9626 < 2.2e-16 ***
## ma2  0.925998   0.065107  14.2228 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# 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
```

```
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 -1.121373   0.055563 -20.1819   <2e-16 ***
## ar2 -0.911364   0.082083 -11.1030   <2e-16 ***
## ar3  0.053735   0.081959   0.6556   0.5121
## ar4 -0.022210   0.055153  -0.4027   0.6872
## ma1  1.175469   0.021048  55.8464   <2e-16 ***
## ma2  0.988325   0.027979  35.3241   <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 -0.551983         NA       NA         NA
## ar2  0.172921   0.038259   4.5198 6.190e-06 ***
## ar3  0.915619   0.060298  15.1850 < 2.2e-16 ***
## ar4  0.435558         NA       NA         NA
## ma1  0.580501         NA       NA         NA
## ma2 -0.196556   0.043484  -4.5202 6.178e-06 ***
## ma3 -0.920758   0.052864 -17.4176 < 2.2e-16 ***
## ma4 -0.551651         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 -0.858500   0.281071 -3.0544 0.0022552 **
## ar2  0.075681   0.128007  0.5912 0.5543661
## ar3  1.122140   0.098181 11.4293 < 2.2e-16 ***
## ar4  0.639474   0.253747  2.5201 0.0117314 *
## ma1  0.923109   0.264539  3.4895 0.0004839 ***
## ma2 -0.032671   0.177524 -0.1840 0.8539858
## ma3 -1.117752   0.155372 -7.1940  6.29e-13 ***
## ma4 -0.719323   0.226069 -3.1819 0.0014633 **
## ---
## 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  9.9074e-02  1.9514e-04  507.703 < 2.2e-16 ***
## ar2  1.3895e-01  3.0111e-04  461.475 < 2.2e-16 ***
## ar3  5.0499e-01  7.2128e-05 7001.257 < 2.2e-16 ***
## ar4  1.6450e-01  1.5559e-04 1057.251 < 2.2e-16 ***
## ar5  9.0947e-02  3.6146e-04  251.613 < 2.2e-16 ***
## ma1 -8.1368e-02  5.7083e-03  -14.254 < 2.2e-16 ***
## ma2 -1.9260e-01  1.6938e-02  -11.371 < 2.2e-16 ***
## ma3 -5.1953e-01  1.7609e-02  -29.504 < 2.2e-16 ***
## ma4 -3.1371e-01  6.3856e-03  -49.127 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
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  0.216613   0.581949  0.3722   0.7097
## ar2  0.125243   0.255393  0.4904   0.6239
## ar3  0.472679   0.519846  0.9093   0.3632
## ar4 -0.054074   0.426025 -0.1269   0.8990
## ar5  0.109269   0.078398  1.3938   0.1634
## ma1 -0.166104   0.585508 -0.2837   0.7766
## ma2 -0.144679   0.251819 -0.5745   0.5656
## ma3 -0.438182   0.523533 -0.8370   0.4026
## ma4 -0.056614   0.413995 -0.1368   0.8912
```

Table 1: AIC sorted models

	df	AIC
model_312_ml	6	-102.82561
model_212_ml	5	-102.65580
model_412_ml	7	-102.28591
model_414_ml	9	-101.78404
model_111_ml	3	-100.97940
model_112_ml	4	-100.65053
model_211_ml	4	-98.98447
model_514_ml	10	-95.11157

Table 2: BIC sorted models

	df	BIC
model_111_ml	3	-89.53701
model_112_ml	4	-85.39400
model_211_ml	4	-83.72795
model_212_ml	5	-83.58515
model_312_ml	6	-79.94082
model_412_ml	7	-75.58699
model_414_ml	9	-67.45687
model_514_ml	10	-56.97026

AIC and BIC Values

From the AIC table 1, BIC table 2 and coefficient significance tests, we selected ARIMA (3,1,2) model. Next we'll perform diagnostic check of selected model

```r
source('sort.score.r')
aic.scores <- sort.score(stats::AIC(model_111_ml,model_112_ml,model_211_ml,model_212_ml,
                                    model_312_ml,model_412_ml,model_414_ml,model_514_ml),
                         score = "aic")
kable(x=aic.scores, caption="\\label{tab:aic}AIC sorted models") %>%
  kable_styling(latex_options = "striped")
```

```r
bic.scores = sort.score(stats::BIC(model_111_ml,model_112_ml,model_211_ml,model_212_ml,
                                    model_312_ml,model_412_ml,model_414_ml,model_514_ml),
                         score = "bic" )
kable(bic.scores, caption="\\label{tab:bic}BIC sorted models") %>%
  kable_styling(latex_options = "striped")
```

Residual Analysis - ARIMA Model

Below are the findings of residuals from ARIMA(3,1,2) model:

- From the time series plot Figure 15 of standard residuals, we can see that residuals are randomly distributed with few spiles, around zero mean level.

- Histogram of studentised residuals within [-2,2] range are not distributed so well. Still it can be considered.
- ACF plot has 2 significant lags at 19 and 20.
- qq plot in figure 16 shows deviation between sample and theoretical values.
- Shapiro-Wilk normality test fails to reject null hypothesis. Studentised residuals are coming from normally distributed population
- Ljung box test (Figure 17) of auto correlation for all lags fails to reject null hypothesis. Data point correlations result from randomness in the sampling process i.e. data are independently distributed

From the overall results, we can make forecasting using ARIMA(3,1,2) model.

```
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
##       -1.6132  -0.8545  0.0479  1.6795  0.9262
## s.e.   0.0837   0.1081  0.0584  0.0643  0.0646
##
## 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 28.56341 517.2301 289.9583 0.5060474 3.975244 0.9891814
##                    ACF1
## Training set 0.06611109
```

```
checkresiduals(fit)
```

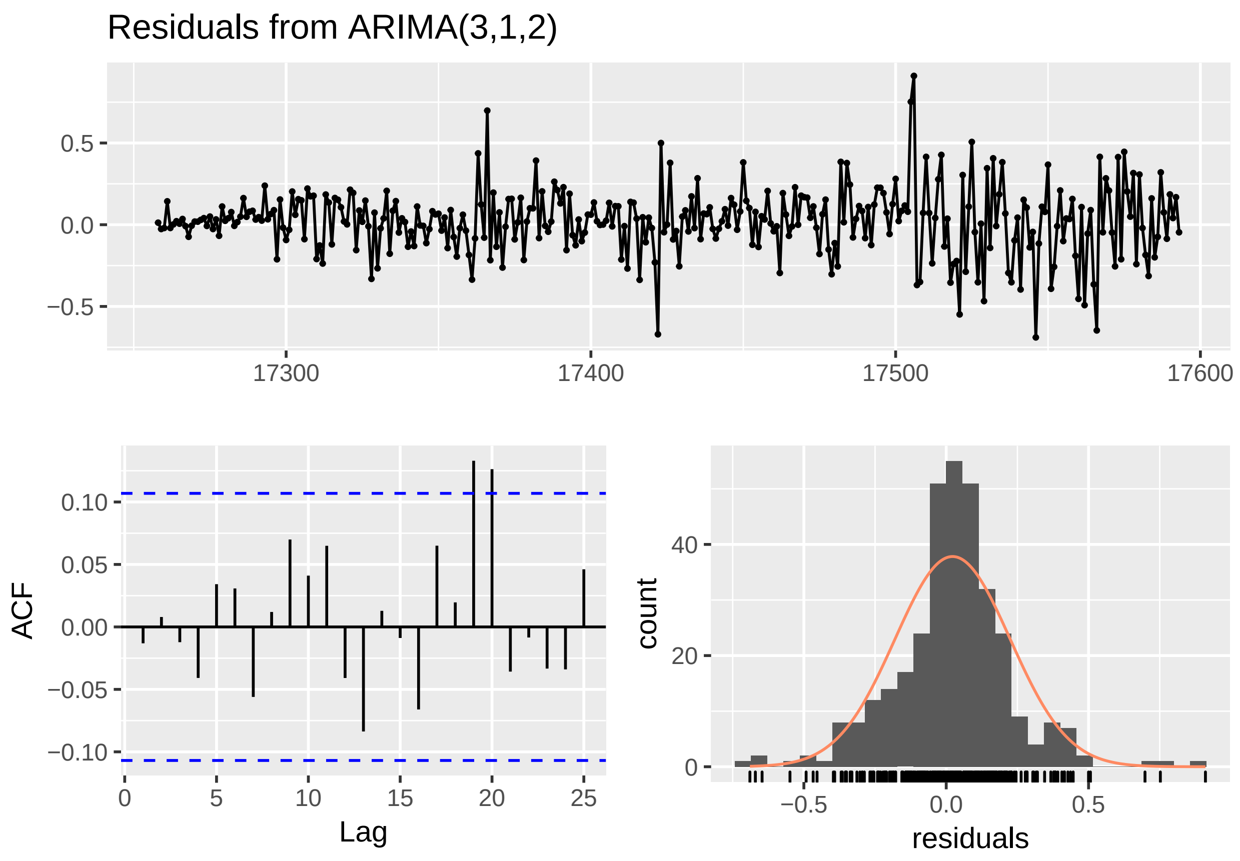


Figure 15: Residual Analysis Quadratic fitted Model

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,1,2)
## Q* = 4.8513, df = 5, p-value = 0.4343
##
## Model df: 5.   Total lags used: 10
```

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

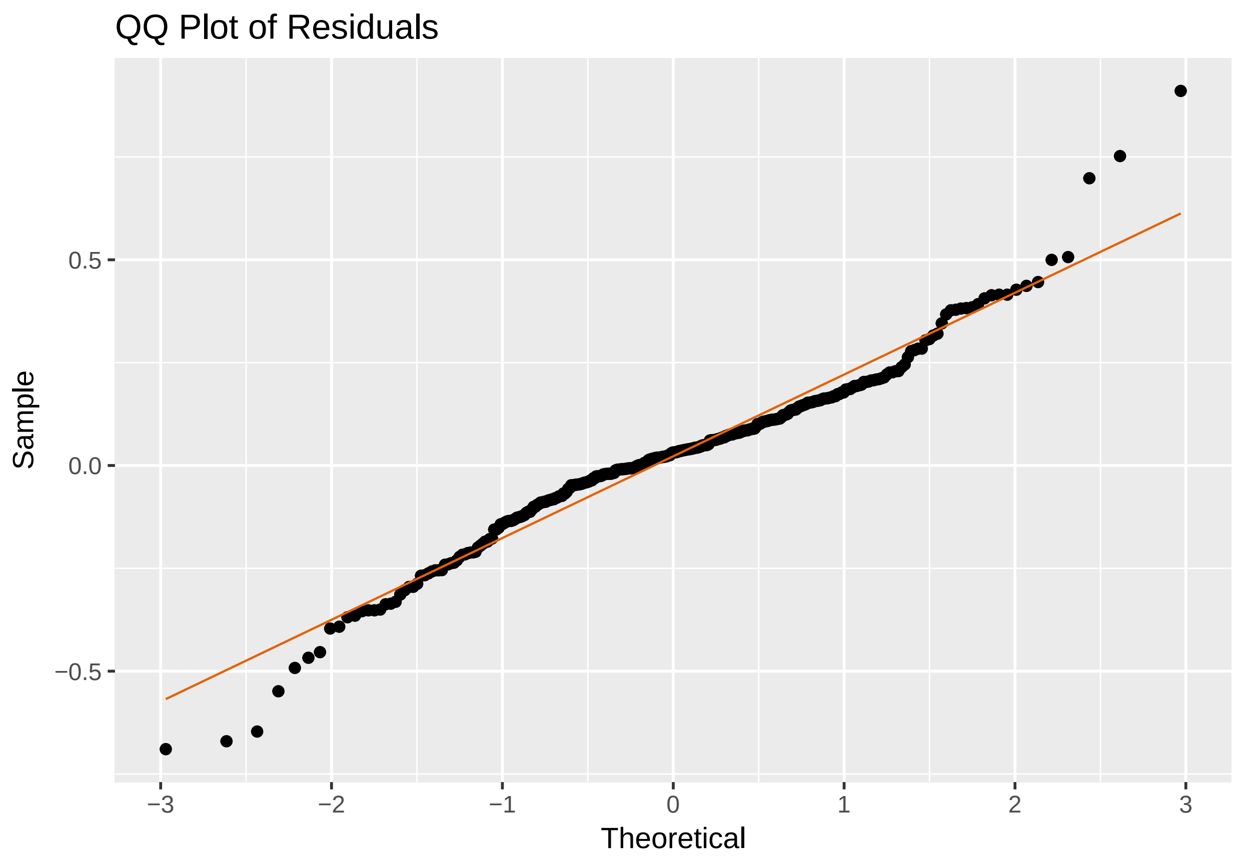


Figure 16: Residual Analysis Linear fitted Model

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

```
##
##  Shapiro-Wilk normality test
##
## data:  as.vector(residuals(fit))
## W = 0.96352, p-value = 1.918e-07
```

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

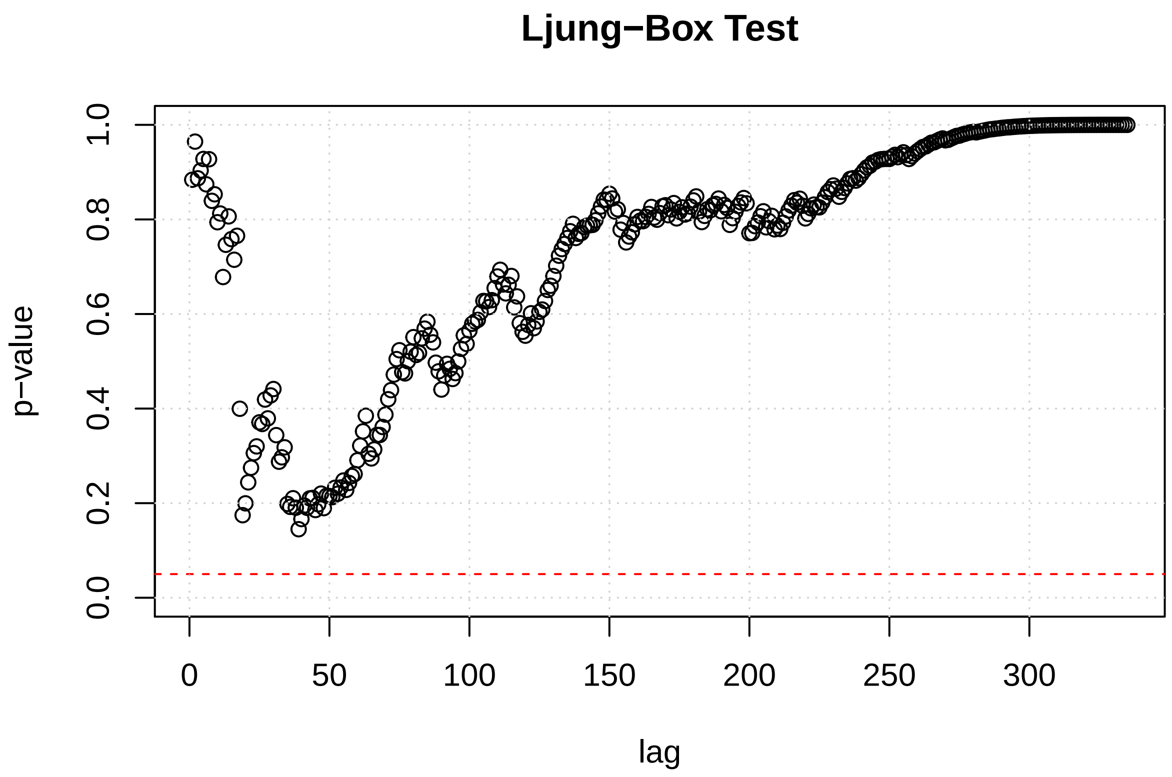


Figure 17: Ljung-Box Test

ARIMA Forecasting

Figure 18 shows the original time series for the historical price of bitcoin. A prediction of the following 10 days is given with the ARIMA(3,1,2) model.

The forecast shows a steady bitcoin price, with a possible increase or decrease within the range of the confidence interval (80 and 95%) bounds.

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

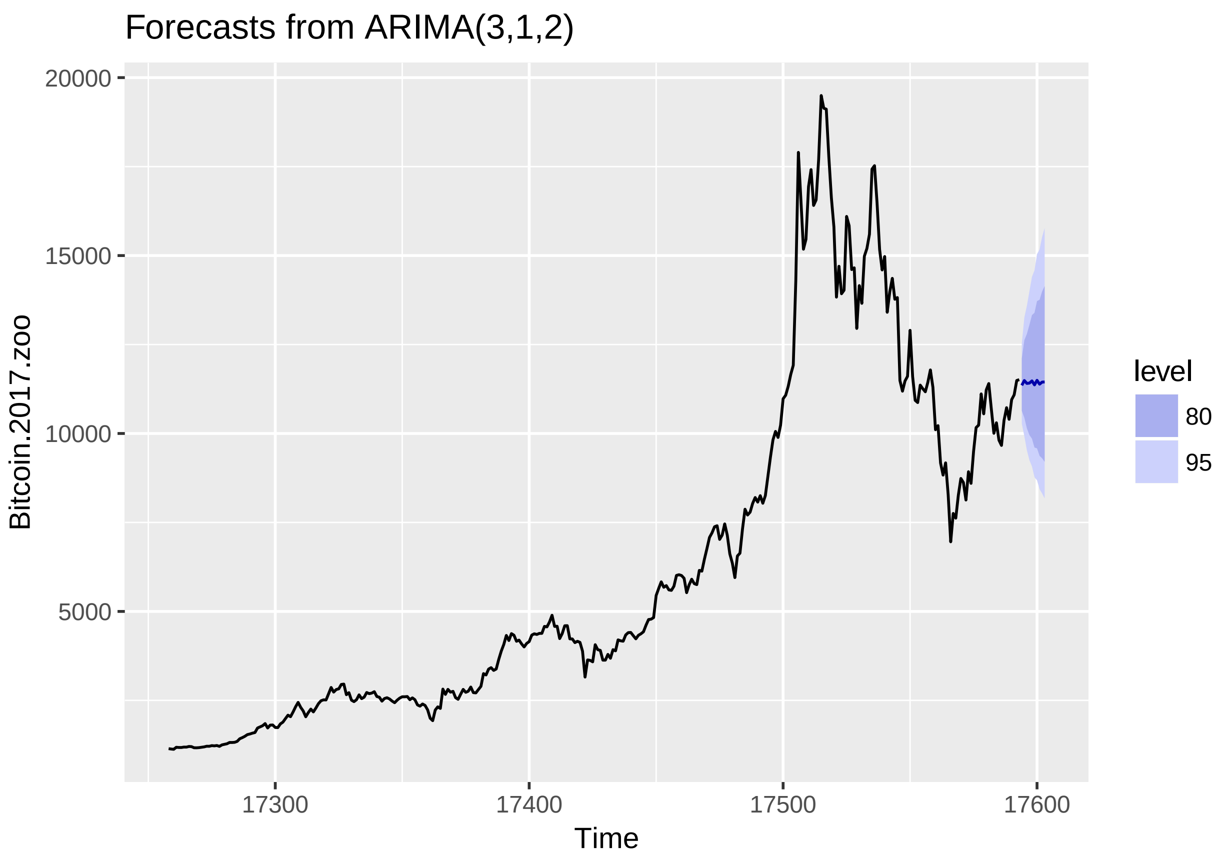


Figure 18: Forecast from ARIMA(3,1,2)

MASE Error

The Model has mean absolute squared error of 0.9891814, while the forecast has error of 3.232775. Error is 3 times the original model. Here we notice that ARIMA model is not suitable for bitcoin prices forecasting. It has not able to capture variation in series. We will try fitting GARCH model for changing variance.

```
source('MASE.r')

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

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

```
## $MASE
##       MASE
## 1 3.232775
```

Heteroscedasticity Models

Figure 19 shows McLeod-Li test is significnat at 5% level of significance for all lags. This gives a strong idea about existence of volatiliy clustering. Figure 20 of QQ plot with fat tails is in accordance with volatiliy clustering

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

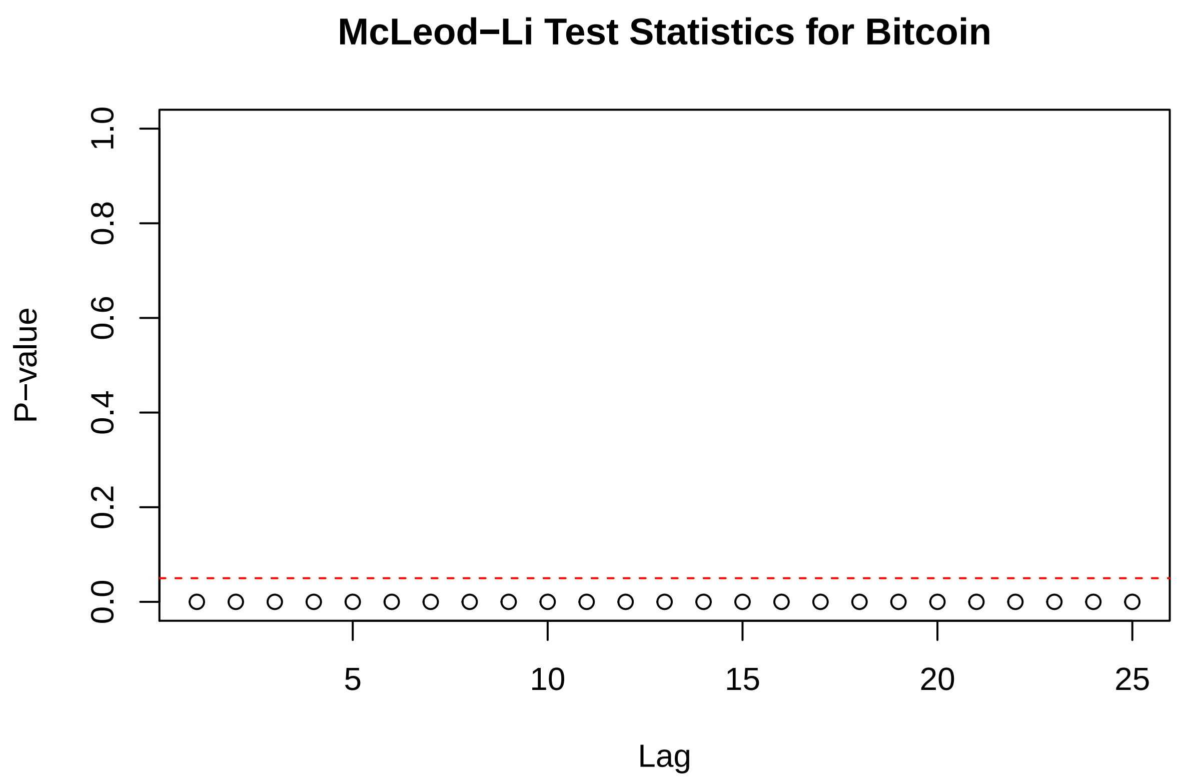


Figure 19: McLeod Li Test Statistics for Bitcoin

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

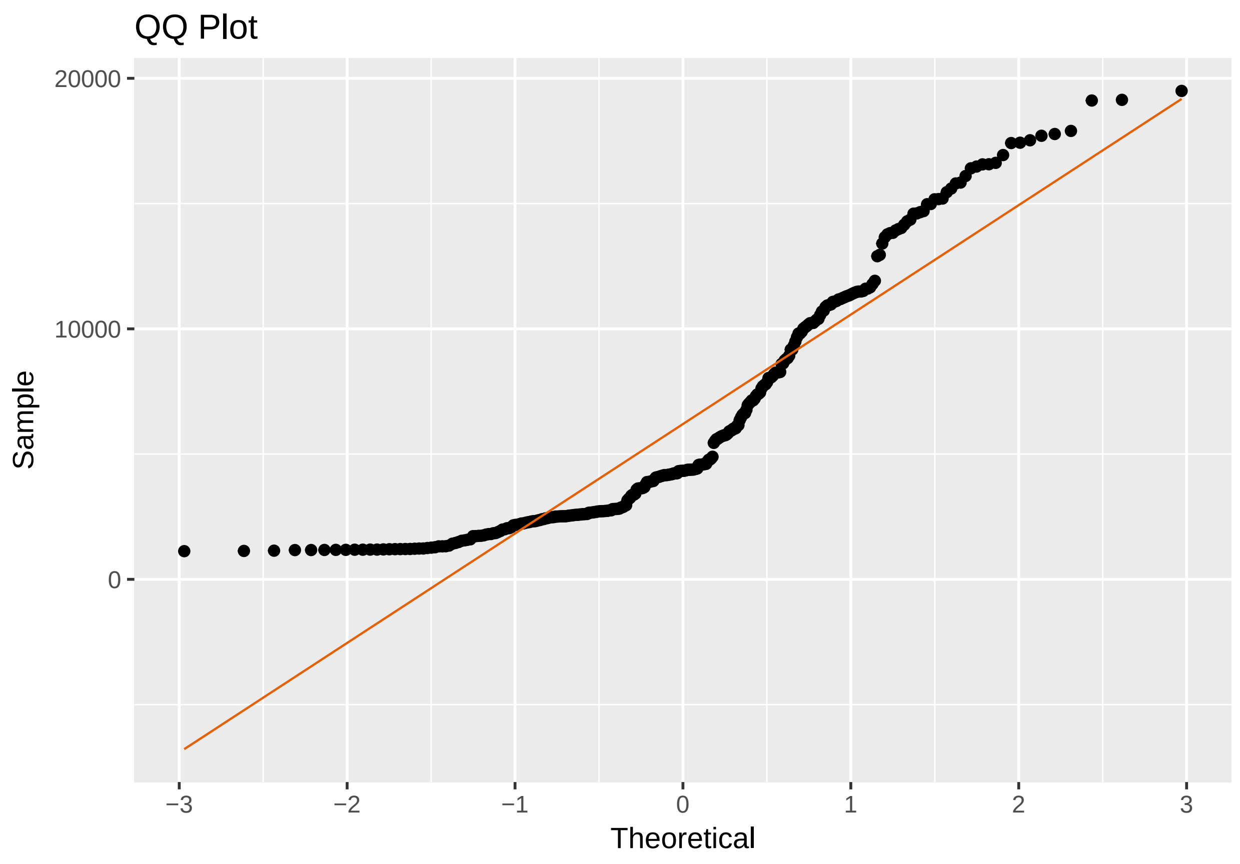


Figure 20: QQ plot for Bitcoin

We will see ACF and PACF of Absolute and squared series to see ARCH effect.

GARCH tests

Figure 21 shows an absolute ACF and FACF plots. We observe many significant lags in both ACF and PACF. EACF do not suggest an ARMA(0,0)

We can identify ARMA(1,0) model for absolute value series. Proposed GARCH models are GARCH(1,0). *Here we are estimating with GARCH(p,q) orders, where p is for AR and q for MA.*

```
abs.bitcoin = abs(Bitcoin.2017.zoo)
sq.bitcoin = Bitcoin.2017.zoo^2

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

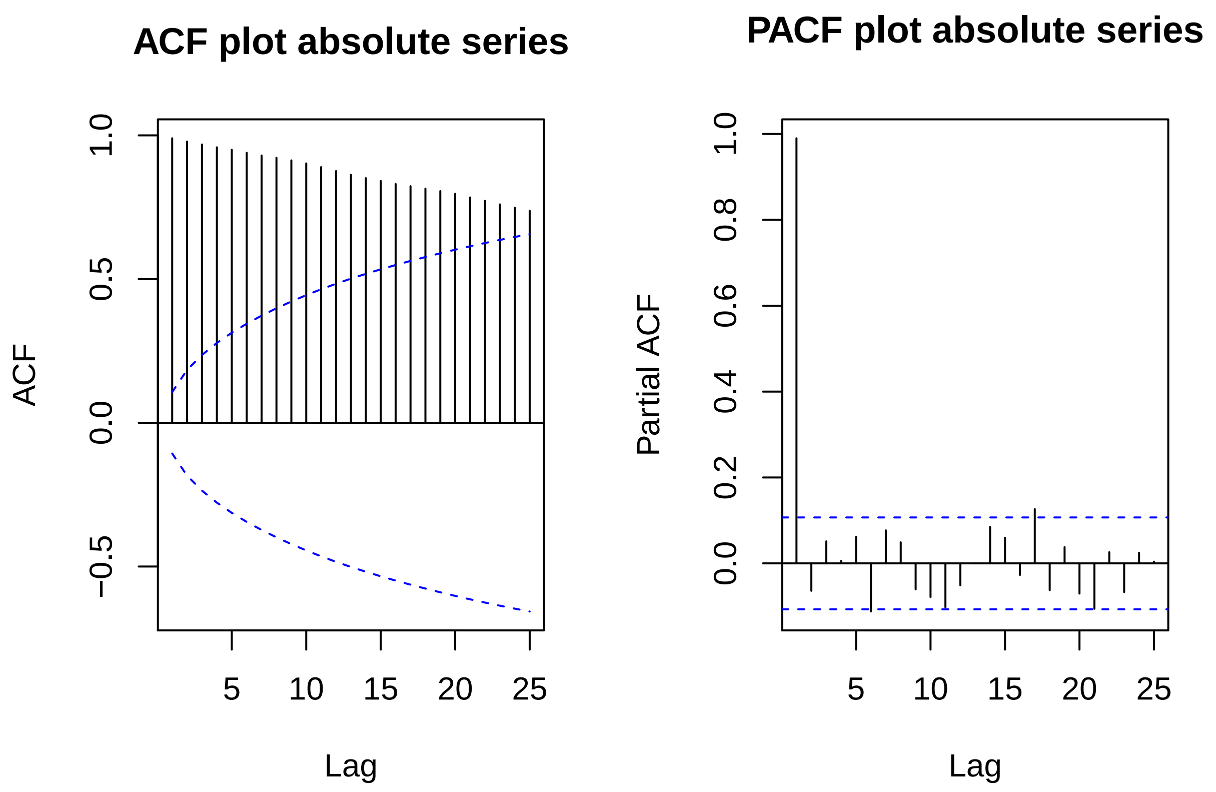


Figure 21: ACF & PACF plot for absolute bitcoin series

```r
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
```

Figure 22 shows an squared ACF and FACF plots. We observe many significant lags in both plots. EACF Matric do not suggest an ARMA(0,0)

We can identify ARMA(1,1) model for absolute value series. Proposed GARCH models are GARCH(1,1).

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

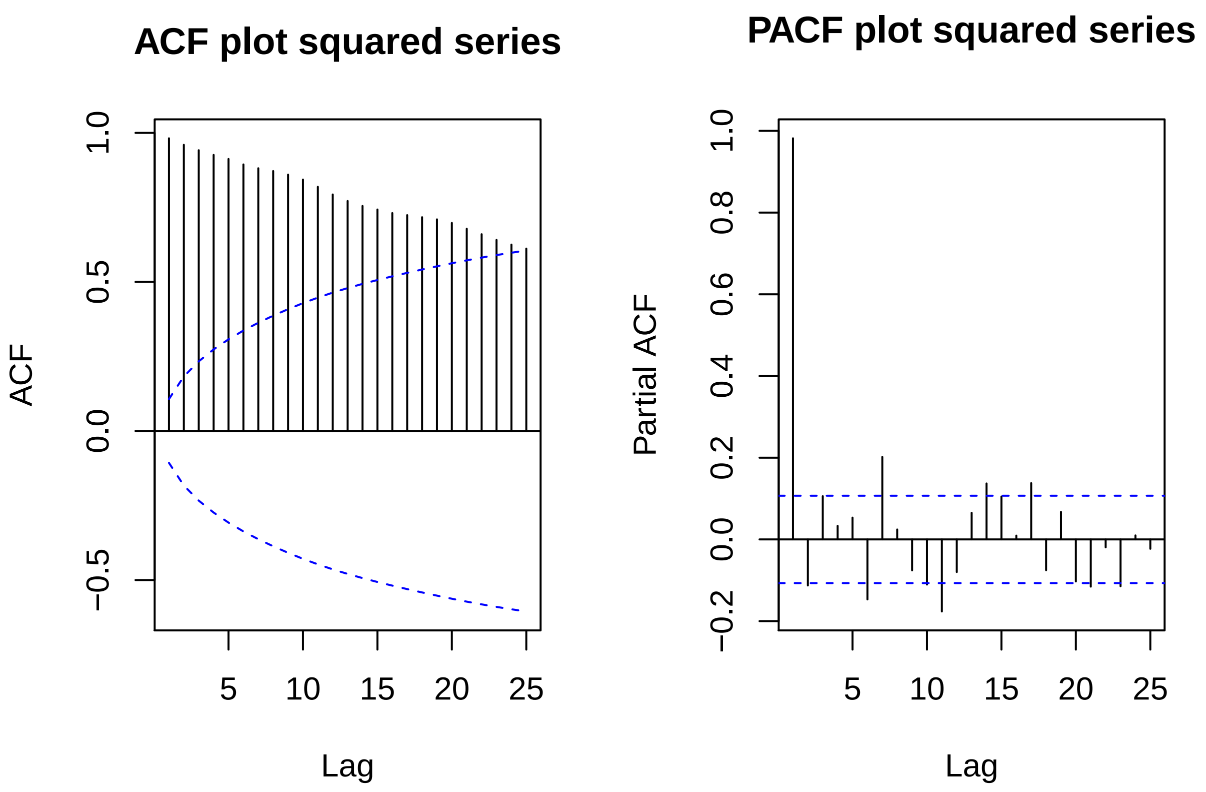


Figure 22: ACF & PACF plot for Squared bitcoin series

```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
```

GARCH Models

From the predicted tentative models, we'll fit GARCH and check the parameters significance.

```r
m.10 = garchFit(formula = ~garch(1,0), data =Bitcoin.diff, trace = FALSE, cond.dist = "QMLE" )
summary(m.10)
```

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

```
## Call:
##  garchFit(formula = ~garch(1, 0), data = Bitcoin.diff, cond.dist = "QMLE",
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 0)
## <environment: 0x7fa0026a2850>
##  [data = Bitcoin.diff]
##
## Conditional Distribution:
##  QMLE
##
## Coefficient(s):
##       mu     omega    alpha1
## 0.028347  0.034290  0.168884
##
## Std. Errors:
##  robust
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.028347    0.010885    2.604  0.00921 **
## omega   0.034290    0.004811    7.127 1.02e-12 ***
## alpha1  0.168884    0.079702    2.119  0.03410 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  64.48749    normalized:  0.1925
##
## Description:
##  Sun May 27 22:38:01 2018 by user:
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  77.42719  0
##  Shapiro-Wilk Test  R    W      0.9651639 3.502547e-07
##  Ljung-Box Test     R    Q(10)  10.64782  0.3856055
##  Ljung-Box Test     R    Q(15)  15.20767  0.4365648
##  Ljung-Box Test     R    Q(20)  35.80253  0.01622367
##  Ljung-Box Test     R^2  Q(10)  31.42701  0.0004986272
##  Ljung-Box Test     R^2  Q(15)  39.29263  0.0005793926
##  Ljung-Box Test     R^2  Q(20)  68.45095  3.253958e-07
##  LM Arch Test       R    TR^2   21.76202  0.0402752
##
## Information Criterion Statistics:
##        AIC        BIC        SIC       HQIC
## -0.3670895 -0.3329331 -0.3672480 -0.3534724
```

```
m.11 = garchFit(formula = ~garch(1,1), data =Bitcoin.diff, trace = FALSE, cond.dist = "QMLE" )
summary(m.11)
```

```
##
```

```
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = Bitcoin.diff, cond.dist = "QMLE",
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0x7fa0031157a8>
##  [data = Bitcoin.diff]
##
## Conditional Distribution:
##  QMLE
##
## Coefficient(s):
##        mu      omega     alpha1      beta1
## 0.0268080  0.0014228  0.1716853  0.8138073
##
## Std. Errors:
##  robust
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.026808    0.008741    3.067  0.00216 **
## omega   0.001423    0.001008    1.411  0.15812
## alpha1  0.171685    0.068729    2.498  0.01249 *
## beta1   0.813807    0.063644   12.787  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  89.07748    normalized:  0.2659029
##
## Description:
##  Sun May 27 22:38:01 2018 by user:
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  134.2896  0
##  Shapiro-Wilk Test  R    W      0.9633369 1.868987e-07
##  Ljung-Box Test     R    Q(10)  11.30396  0.3343319
##  Ljung-Box Test     R    Q(15)  13.02518  0.600353
##  Ljung-Box Test     R    Q(20)  27.27717  0.1276445
##  Ljung-Box Test     R^2  Q(10)  6.660232  0.7570844
##  Ljung-Box Test     R^2  Q(15)  7.45071   0.943916
##  Ljung-Box Test     R^2  Q(20)  11.97325  0.9169935
##  LM Arch Test       R    TR^2   6.237726  0.9036271
##
## Information Criterion Statistics:
##        AIC        BIC        SIC       HQIC
## -0.5079253 -0.4623834 -0.5082059 -0.4897691
```

GARCH(1,0) model has all significant parameters. Next we'll perform residual analysis on selected model.

Residual Analysis - GARCH

Results of residual from figure 23:

- From the time series plot of standard residuals, we can see that residuals are randomly distributed with high variation in end
- Histogram of studentised residuals within [-4,4] range are not distributed well but also positive skewed
- ACF plot has 2 significant lags at 19 and 20. PACF has significant lag at 19
- qq plot shows deviation between sample and theoretical values.
- Ljung box test of auto correlation for lags up to 19 fails to reject null hypothesis. Data point correlations result from randomness in the sampling process i.e. data are independently distributed. After lag 20, there is significant correlation for few lags

From the overall results, we will go with GARCH(1,0) model.

```r
source('residual.analysis.R')
fit = garch(Bitcoin.diff,order=c(1,0),trace = FALSE)
residual.analysis(fit,class="GARCH",start=2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.95829, p-value = 3.746e-08
```

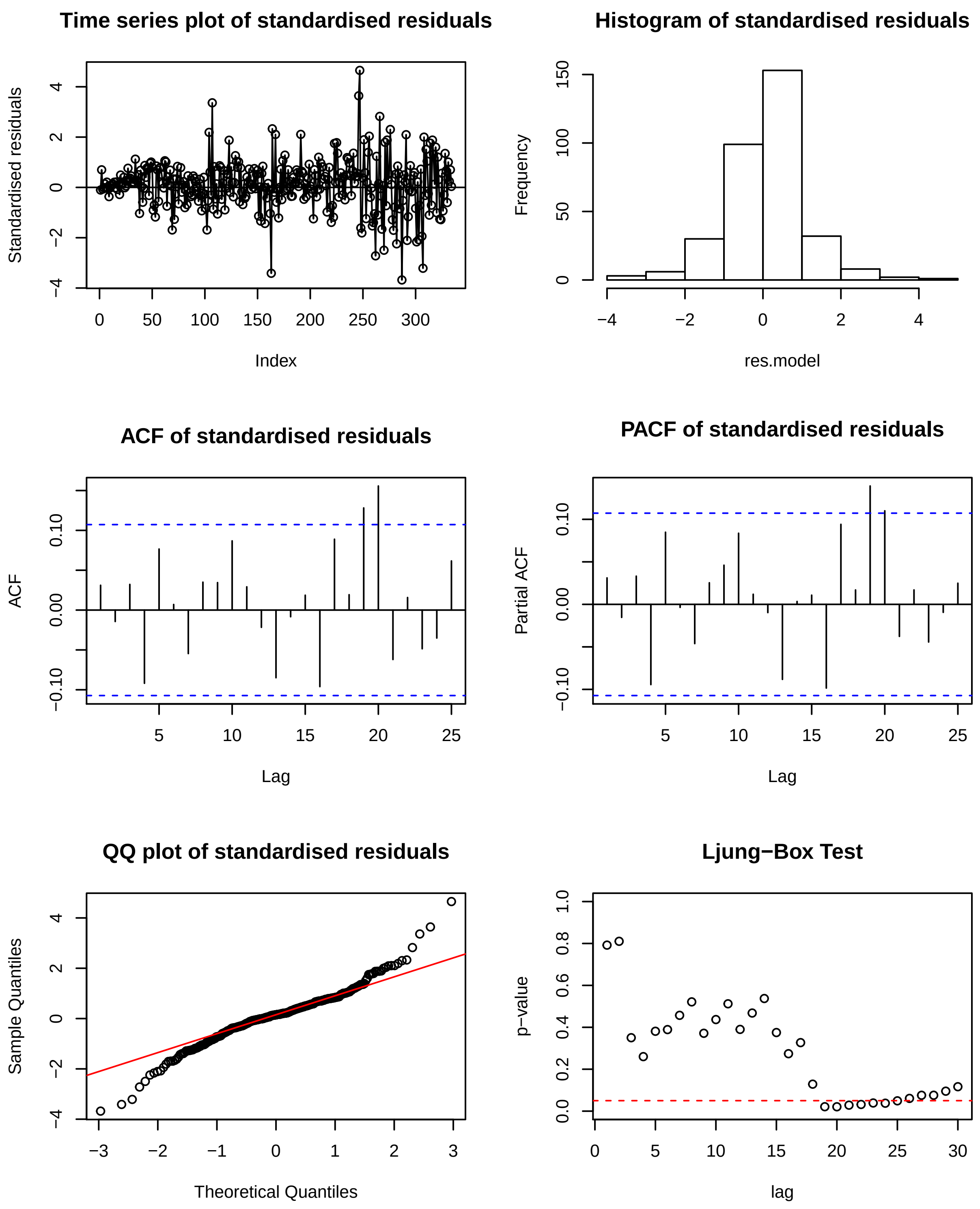


Figure 23: Residual Analysis GARCH(1,0)

Forecasting with mean and variance

Here we'll forecast bitcoin values with ARMA and standard GARCH model for next 10 values. Figure 24 shows the next 10 forecast.

```
model<-ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)),
                  mean.model = list(armaOrder = c(3, 2), include.mean = FALSE),
                  distribution.model = "norm")
model.fit<-ugarchfit(spec=model,data=Bitcoin.2017.zoo)

forecast.model.fit = ugarchforecast(model.fit, n.ahead = 10)
plot(forecast.model.fit, which=1)
```

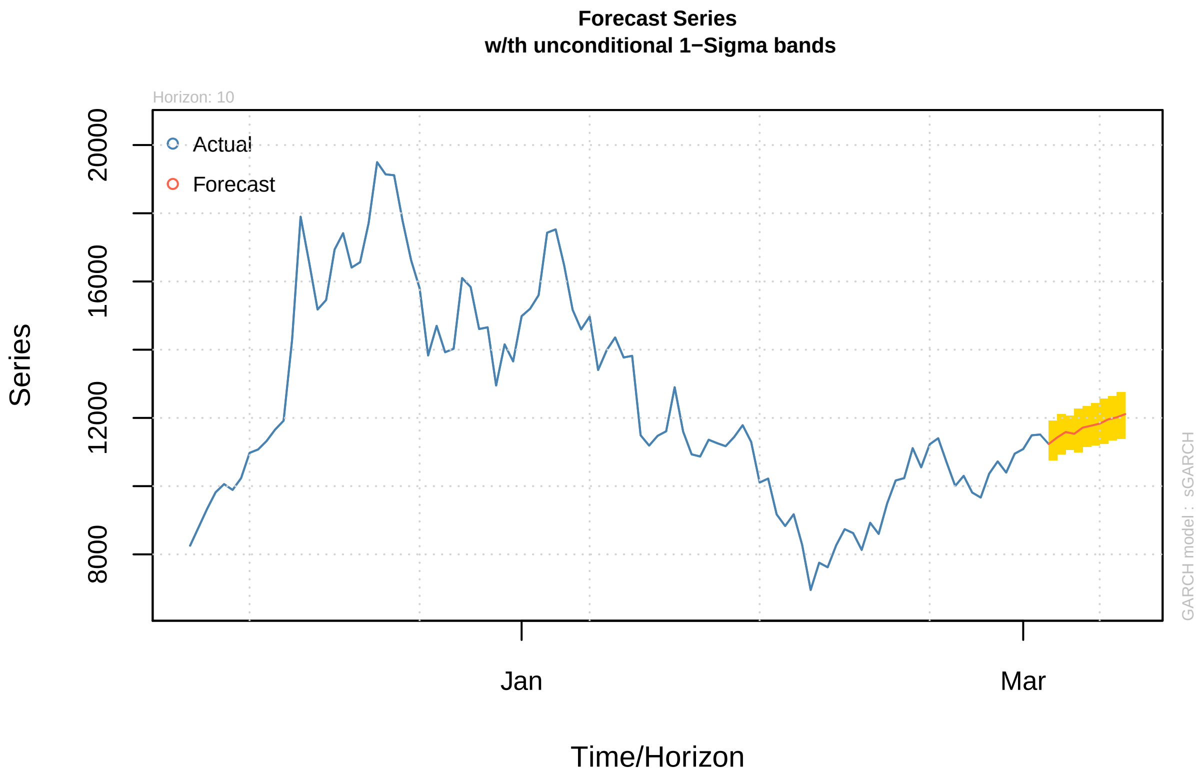


Figure 24: Forecast with ARMA and GARCH

Forecasting with GARCH Bootstrap

Here we'll predict 10 values with bootstrap method. Below are the summary statistics for forecast.

```
bootp = ugarchboot(model.fit, method = "Partial", n.ahead = 10, n.bootpred = 500)
show(bootp)
```

```
##
## *-----------------------------------*
## *     GARCH Bootstrap Forecast      *
## *-----------------------------------*
## Model : sGARCH
## n.ahead : 10
## Bootstrap method:  partial
## Date (T[0]): 2018-03-03
##
## Series (summary):
```

```
##            min  q.25  mean  q.75     max forecast[analytic]
## t+1     8787.4 10905 11218 11530   13047              11235
## t+2     4418.4 10996 11355 11809   16358              11428
## t+3   -19595.3 11107 11386 12073   20475              11587
## t+4   -25985.5 10983 11452 12140   71204              11534
## t+5   -30139.3 11112 11916 12404  191018              11716
## t+6   -29584.4 11114 12433 12539  369792              11772
## t+7   -31243.1 11013 12854 12587  561584              11834
## t+8   -32789.6 10961 11755 12714   53775              11958
## t+9   -32866.8 10954 12240 12840  249453              12017
## t+10 -38531.9 11024 14766 12998 1419705              12111
## ....................
##
## Sigma (summary):
##          min  q0.25    mean  q0.75       max forecast[analytic]
## t+1  473.94 473.94  473.94 473.94    473.94             473.94
## t+2  147.95 202.03  473.84 615.46   2450.68             496.27
## t+3  147.94 176.58  532.58 558.40   6373.27             517.61
## t+4  147.94 169.95  547.14 476.68  26208.20             538.09
## t+5  147.94 166.37  678.38 391.46 100185.74             557.80
## t+6  147.94 165.11  589.66 418.49  70875.48             576.81
## t+7  147.94 165.35  857.78 419.91 181770.05             595.20
## t+8  147.94 172.12  745.02 402.79 128353.76             613.02
## t+9  147.94 160.86 1536.86 405.17 537135.76             630.32
## t+10 147.94 162.75 1407.32 409.29 456932.75             647.15
## ....................
```

Limitations

- The series is highly volatile, regression and ARIMA models are not suitable for prediction
- Other time series models can be applied like exponential state space models, distributed lag models and neural network (`nnetar`)

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 ARMA (mean) and GARCH (variance) prediction model are better for daily bitcoin closing prices.

References

Wikipedia. 2018. "Bitcoin." https://en.wikipedia.org/wiki/Bitcoin.