MATH1318 Semester 1, 2018

Time Series - Final Project(Competitve)

s3650497- Mohammad,s3689517-Malgorzata Sikora,s3638787 - Ravi Pandey

Importing Data

```
Bitcoin <- read.csv("C:/Users/Wel/Downloads/R/Bitcoin_Historical_Price.csv")
head(Bitcoin)
```

```
##    Close
## 1 134.21
## 2 144.54
## 3 139.00
## 4 116.99
## 5 105.21
## 6  97.75
```

Converting to Time series Object

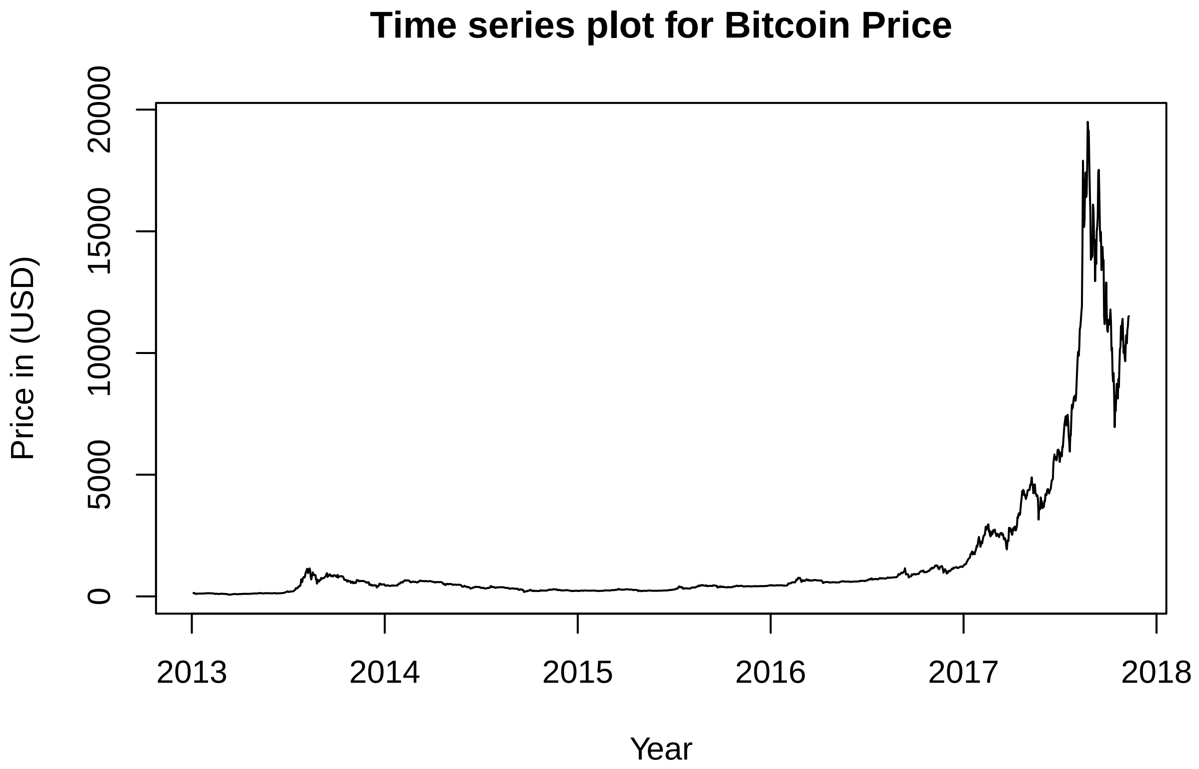
```
Bitcoin.ts<- ts(Bitcoin,start=c(2013,4,27),frequency=365.25)
class(Bitcoin.ts)
```

```
## [1] "ts"
```

```
head(Bitcoin.ts)
```

```
## Time Series:
## Start = 2013.00821355236
## End = 2013.0219028063
## Frequency = 365.25
##       Close
## [1,] 134.21
## [2,] 144.54
## [3,] 139.00
## [4,] 116.99
## [5,] 105.21
## [6,]  97.75
```

```
plot(Bitcoin.ts,ylab='Price in (USD)',xlab='Year', main = "Time series plot for Bitcoin Price")
```



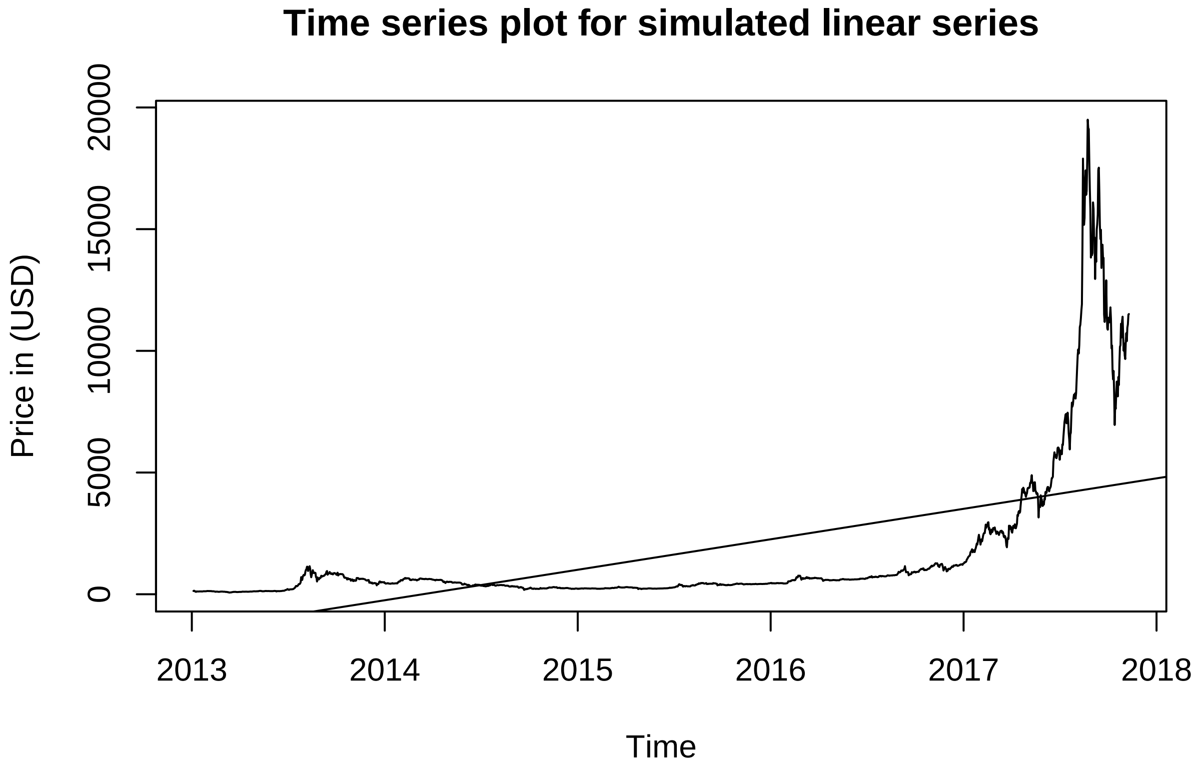
Random Walk(Linear Regression Model)

```
model1 = lm(Bitcoin.ts~time(Bitcoin.ts))
summary(model1)
```

```
##
## Call:
## lm(formula = Bitcoin.ts ~ time(Bitcoin.ts))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2462.4 -1658.8  -558.4  1049.6 15182.3
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -2.522e+06  8.502e+04  -29.66   <2e-16 ***
## time(Bitcoin.ts)  1.252e+03  4.218e+01   29.68   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2487 on 1770 degrees of freedom
## Multiple R-squared:  0.3322, Adjusted R-squared:  0.3319
## F-statistic: 880.7 on 1 and 1770 DF,  p-value: < 2.2e-16
```

Added the fitted least squares line from model1

```
plot(Bitcoin.ts, ylab='Price in (USD)', main = "Time series plot for simulated linear series")
abline(model1)
```

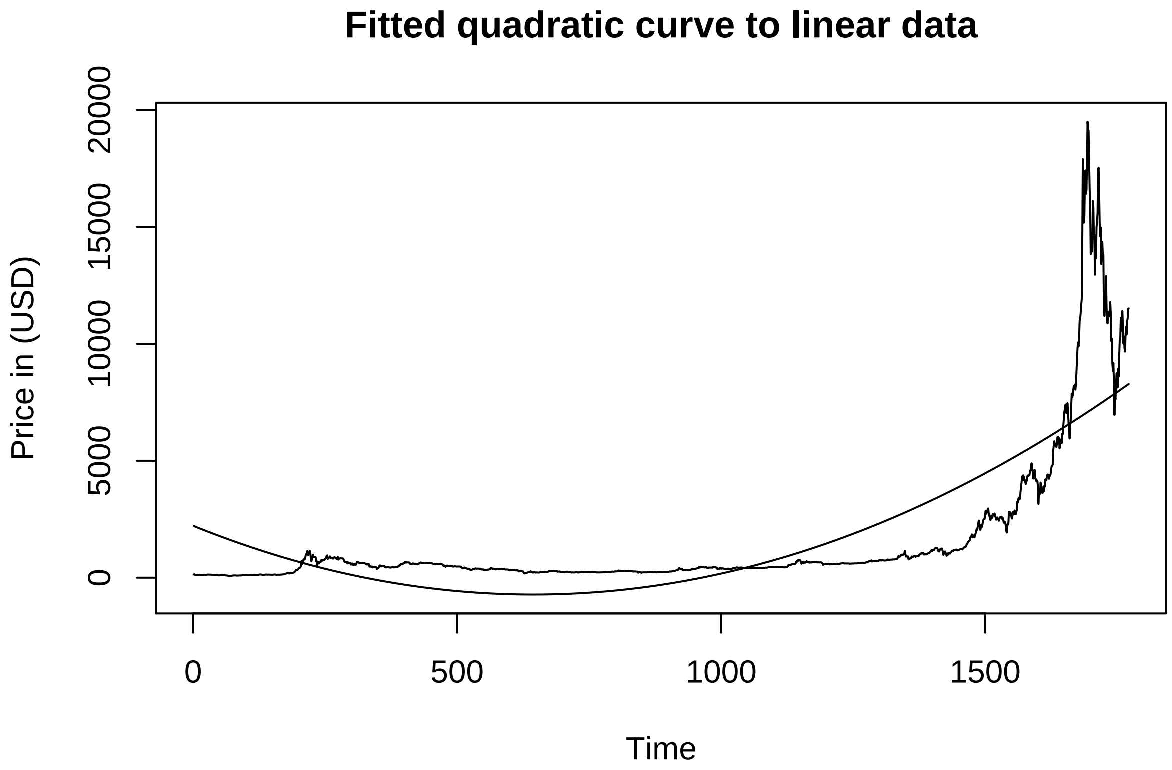


Quadratic Model

```
t = time(Bitcoin.ts)
t2 = t^2
model1.1 = lm(Bitcoin.ts~t+t2) # label the model as model1
summary(model1.1)
```

```
##
## Call:
## lm(formula = Bitcoin.ts ~ t + t2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3041.0 -1317.1   248.7   893.0 12418.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.834e+09  1.020e+08   37.59   <2e-16 ***
## t           -3.806e+06  1.012e+05  -37.60   <2e-16 ***
## t2           9.444e+02  2.511e+01   37.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1854 on 1769 degrees of freedom
## Multiple R-squared:  0.629,  Adjusted R-squared:  0.6285
## F-statistic:  1499 on 2 and 1769 DF,  p-value: < 2.2e-16
```

```
plot(ts(fitted(model1.1)), ylim = c(min(c(fitted(model1.1),
as.vector(Bitcoin.ts))), max(c(fitted(model1.1),as.vector(Bitcoin.ts)))),ylab='Price in (USD)' ,
main = "Fitted quadratic curve to linear data")
lines(as.vector(Bitcoin.ts))
```



Cyclical or Seasonal Trends

```
Bitcoin.ts1=(ts(Bitcoin.ts, start = 2013,end = 2018, frequency = 12))
month.=season(Bitcoin.ts1)
model2=lm(Bitcoin.ts1~month.-1)
summary(model2)
```

```
##
## Call:
## lm(formula = Bitcoin.ts1 ~ month. - 1)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -21.972  -8.364  -1.266   7.398  23.058 
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## month.January    116.492      4.835   24.09   <2e-16 ***
## month.February   121.482      5.296   22.94   <2e-16 ***
## month.March      122.110      5.296   23.06   <2e-16 ***
## month.April      118.470      5.296   22.37   <2e-16 ***
## month.May        113.988      5.296   21.52   <2e-16 ***
## month.June       112.034      5.296   21.15   <2e-16 ***
## month.July       113.952      5.296   21.52   <2e-16 ***
## month.August     117.176      5.296   22.12   <2e-16 ***
## month.September  116.220      5.296   21.94   <2e-16 ***
## month.October    114.676      5.296   21.65   <2e-16 ***
## month.November   114.564      5.296   21.63   <2e-16 ***
## month.December   112.364      5.296   21.22   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 11.84 on 49 degrees of freedom
## Multiple R-squared:  0.9917, Adjusted R-squared:  0.9897 
## F-statistic: 489.2 on 12 and 49 DF,  p-value: < 2.2e-16
```

```r
model3=lm(Bitcoin.ts1~month.)
summary(model3)
```

```
## 
## Call:
## lm(formula = Bitcoin.ts1 ~ month.)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -21.972  -8.364  -1.266   7.398  23.058 
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)     116.4917     4.8348  24.094   <2e-16 ***
## month.February    4.9903     7.1712   0.696    0.490    
## month.March       5.6183     7.1712   0.783    0.437    
## month.April       1.9783     7.1712   0.276    0.784    
## month.May        -2.5037     7.1712  -0.349    0.728    
## month.June       -4.4577     7.1712  -0.622    0.537    
## month.July       -2.5397     7.1712  -0.354    0.725    
## month.August      0.6843     7.1712   0.095    0.924    
## month.September  -0.2717     7.1712  -0.038    0.970    
## month.October    -1.8157     7.1712  -0.253    0.801    
## month.November   -1.9277     7.1712  -0.269    0.789    
## month.December   -4.1277     7.1712  -0.576    0.568    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 11.84 on 49 degrees of freedom
## Multiple R-squared:  0.07783,    Adjusted R-squared:  -0.1292 
## F-statistic: 0.3759 on 11 and 49 DF,  p-value: 0.9594
```

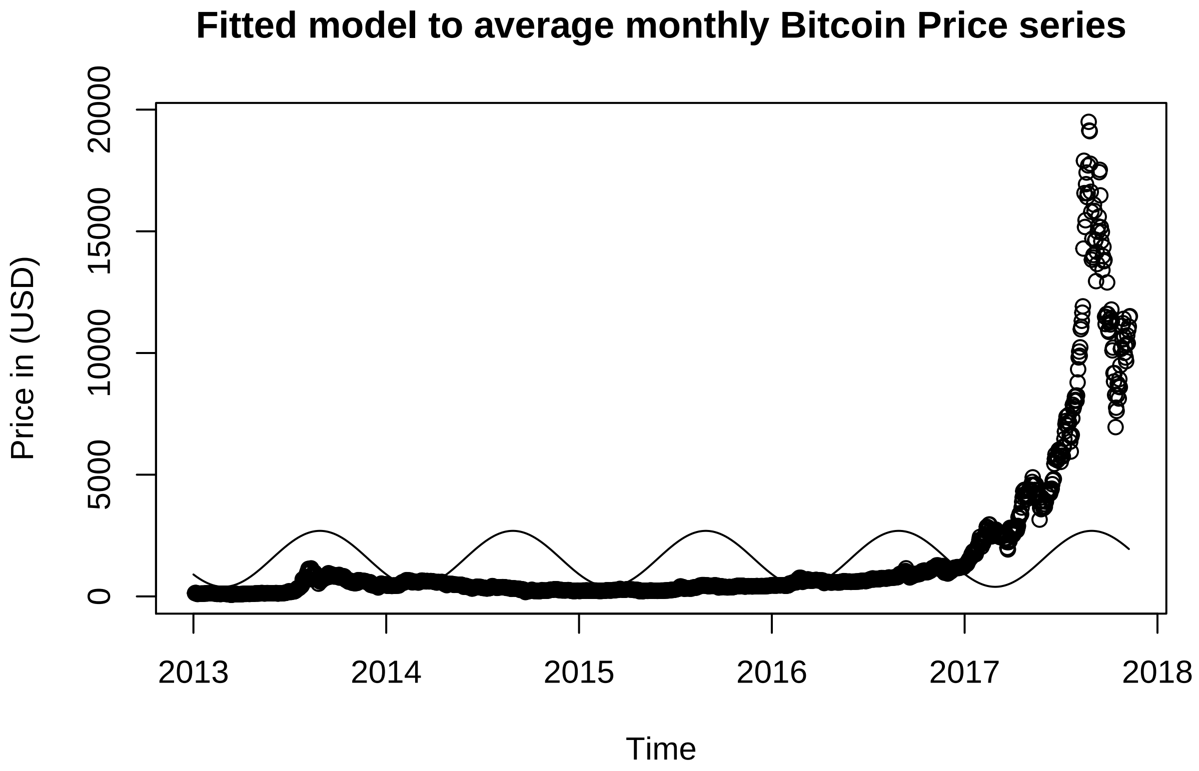
Cosine Models(Harmonic Models)

```
har.=harmonic(Bitcoin.ts,1) # calculate cos(2*pi*t) and sin(2*pi*t)
model4=lm(Bitcoin.ts~har.)
summary(model4)
```

```
##
## Call:
## lm(formula = Bitcoin.ts ~ har.)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2439.0 -1665.5  -718.9    44.3 16814.0
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1543.40      69.68  22.149  < 2e-16 ***
## har.cos(2*pi*t)  -591.69      99.42  -5.951  3.2e-09 ***
## har.sin(2*pi*t)  -985.67      97.64 -10.095  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2931 on 1769 degrees of freedom
## Multiple R-squared:  0.07325,    Adjusted R-squared:  0.07221
## F-statistic: 69.92 on 2 and 1769 DF,  p-value: < 2.2e-16
```

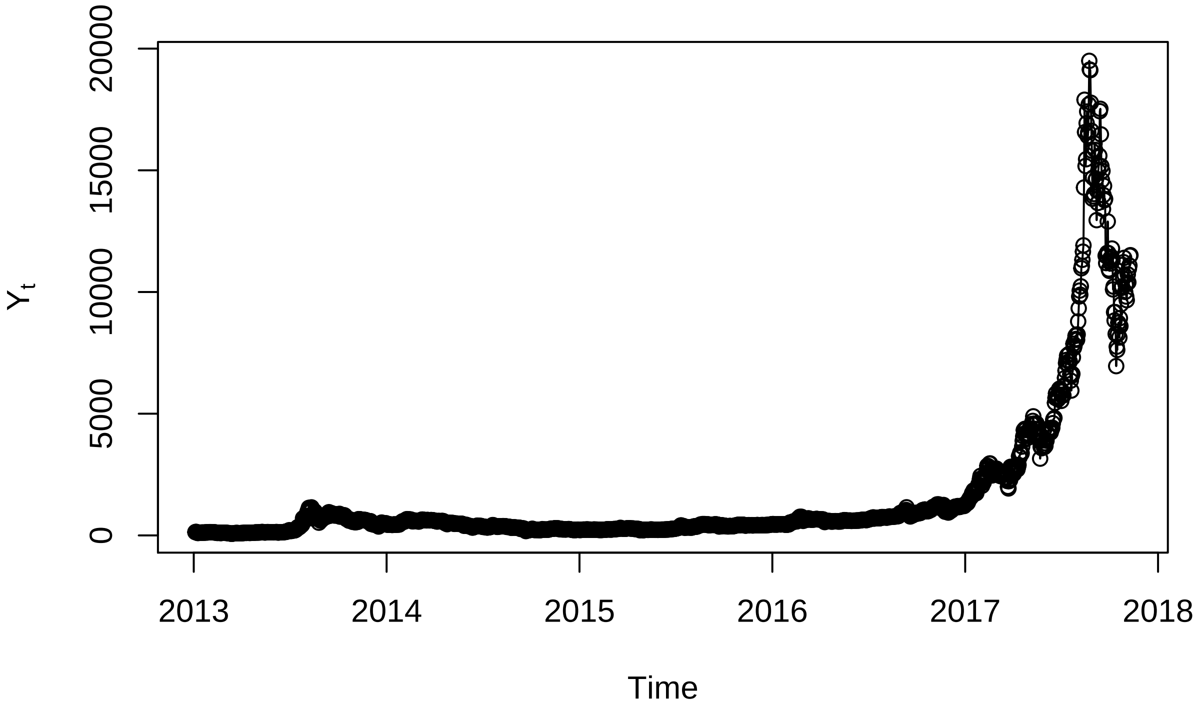
```
##Plotting Harmonic models

plot(ts(fitted(model4),freq=365,start=c(2013,1)),ylab='Price in (USD)',type='l',
ylim=range(c(fitted(model4),Bitcoin.ts)),main="Fitted model to average monthly Bitcoin Price series")
points(Bitcoin.ts)
```



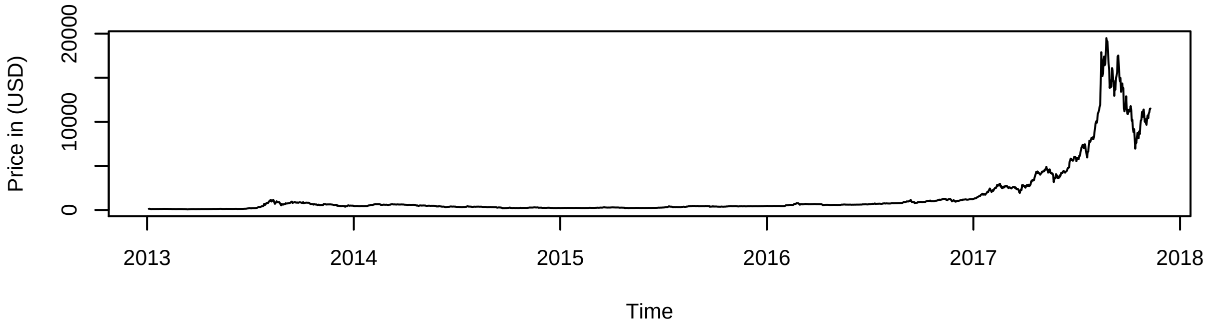
Stationarity Through Differencing(White noise is generated from the standard normal distribution)

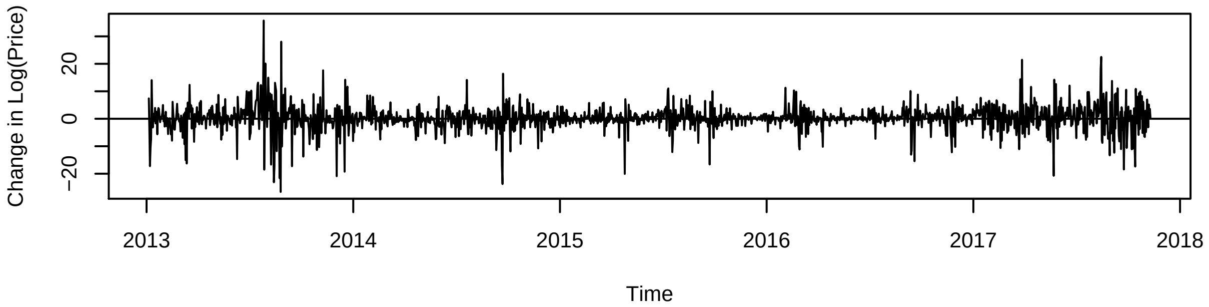
```
plot(Bitcoin.ts,ylab=expression(Y[t]),type='o')
```



Time series plot of the differences of logarithms of that series

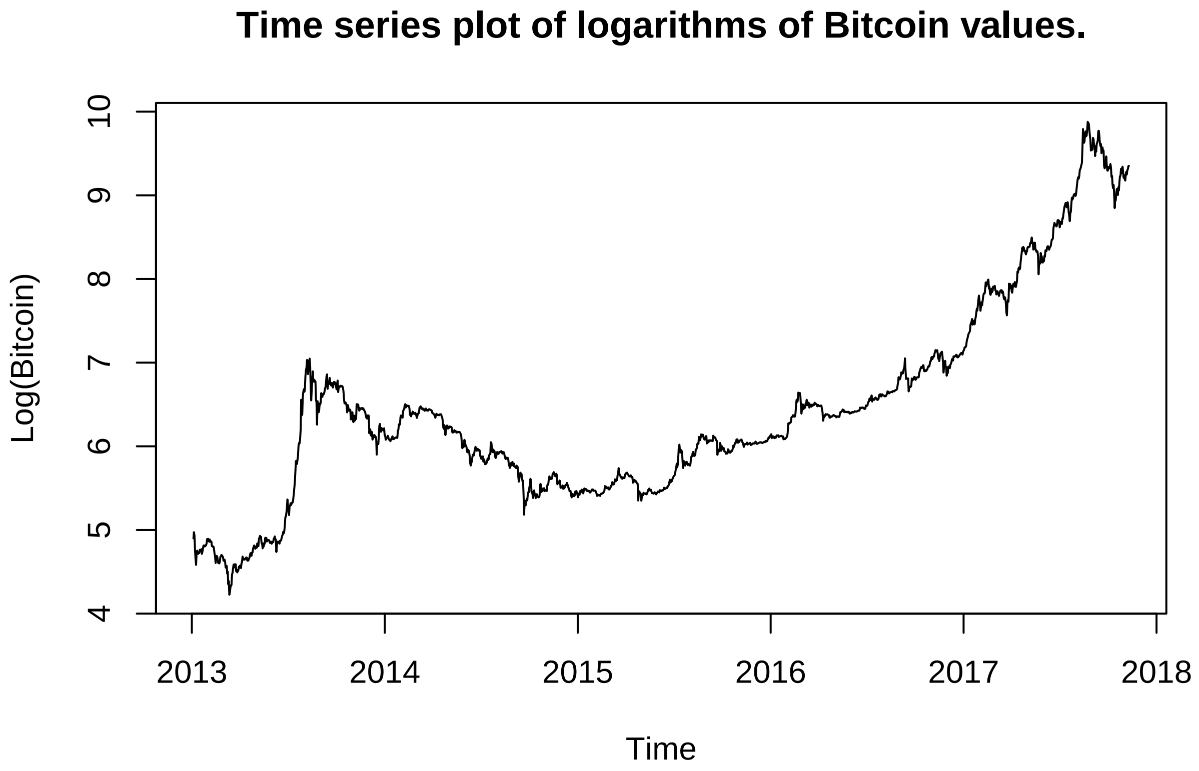
```
par(mfrow=c(2,1))
par(cex=0.7)
plot(Bitcoin.ts, ylab='Price in (USD)',type='l')
first.diff = diff(log(Bitcoin.ts))*100
plot(first.diff,ylab='Change in Log(Price)',type='l')
abline(h=0)
```





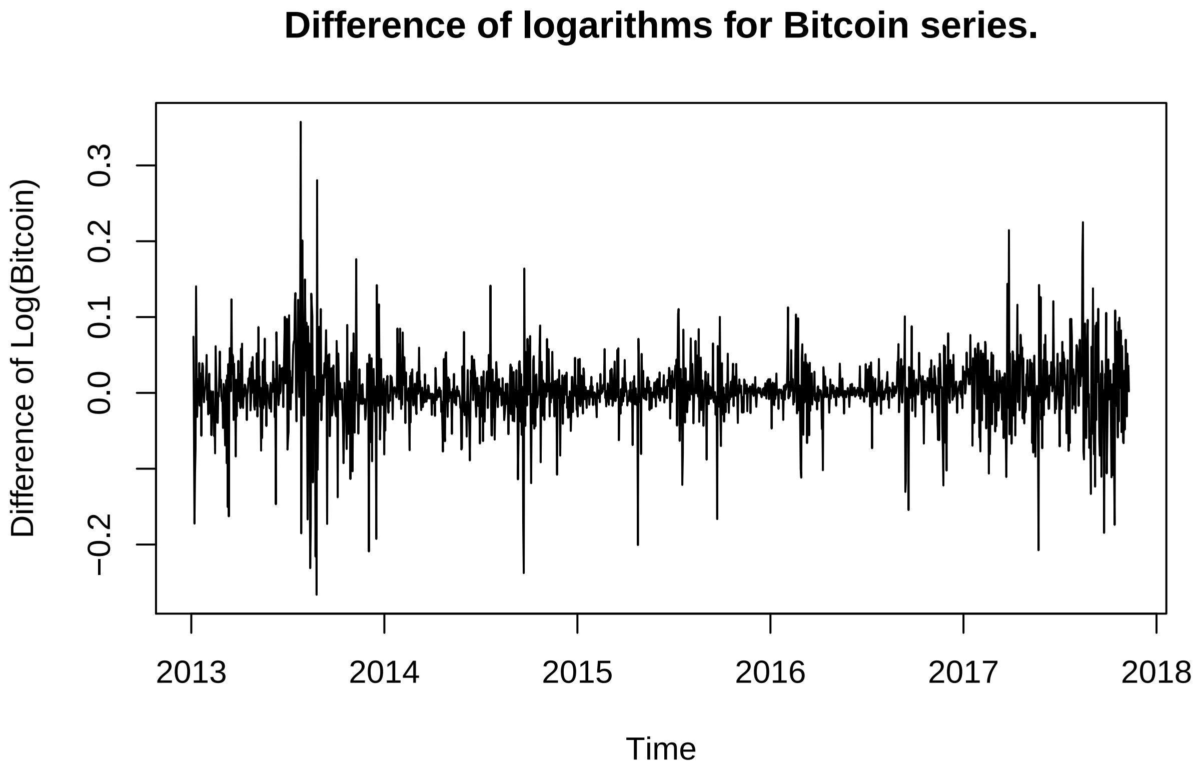
Natural log transformation

```
plot(log(Bitcoin.ts),ylab='Log(Bitcoin)',
     main = "Time series plot of logarithms of Bitcoin values.")
```



Differences of the logarithms of the Bitcoin values

```
plot(diff(log(Bitcoin.ts)), ylab='Difference of Log(Bitcoin)',
     main = "Difference of logarithms for Bitcoin series.")
```



#ADF

```
adf.test(first.diff)
```

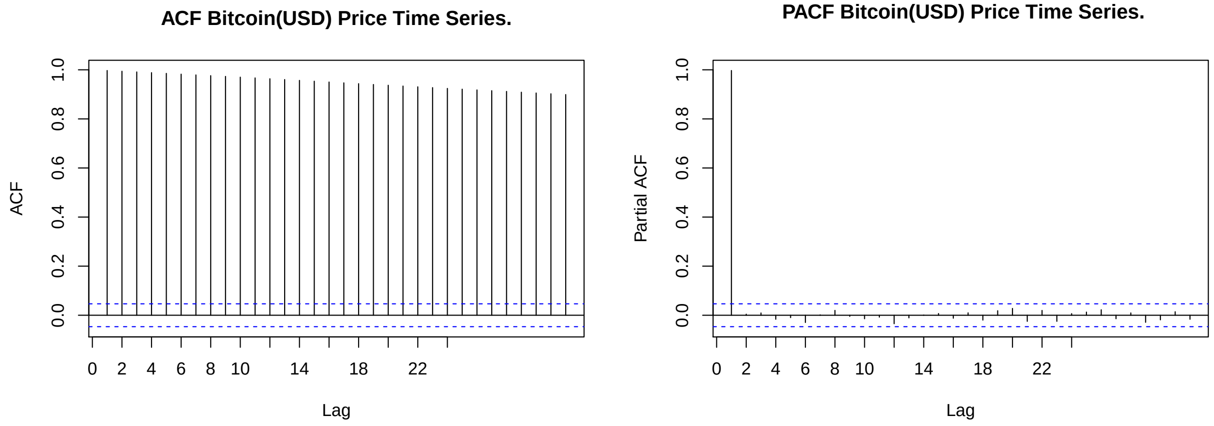
```
## Warning in adf.test(first.diff): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  first.diff
## Dickey-Fuller = -10.167, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
```

ARMA

Significant Lags can be seen in ACF with first significant lag in PACF. From Eacf, we got ARMA(1,1), ARMA (1,2)

```
#Arma (1,1) (1,2)
par(mfrow=c(1,2))
acf(as.vector(log(Bitcoin.ts)),xaxp=c(0,24,12), main="ACF Bitcoin(USD) Price Time Series.")
pacf(as.vector(log(Bitcoin.ts)),xaxp=c(0,24,12), main=" PACF Bitcoin(USD) Price Time Series.")
```



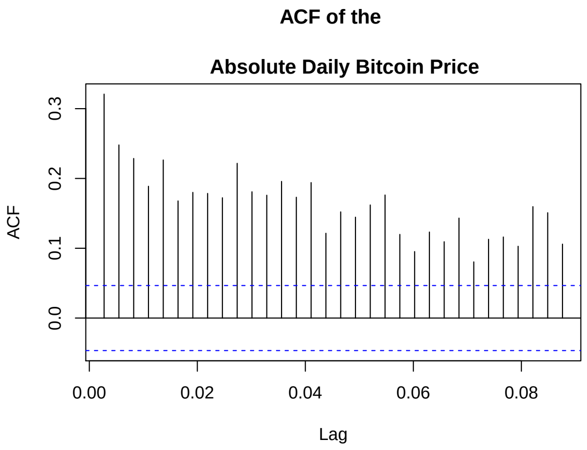
```r
eacf(first.diff)
```

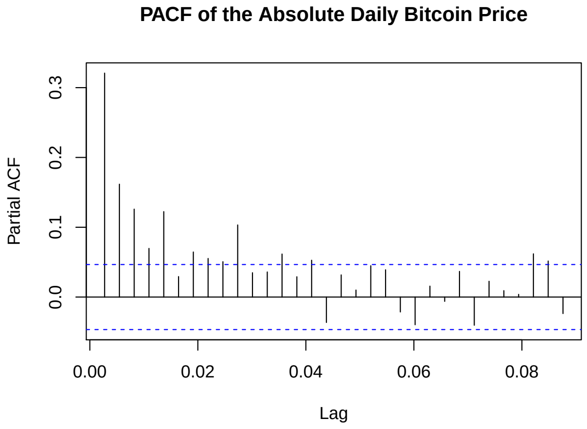
```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 o o o o x x o o o x x  o  o  o
## 1 x o o o o x o o o o x  o  o  o
## 2 x x o o o x o o o o x  o  o  o
## 3 x x x o o x o o o o x  o  o  o
## 4 x x o x o x o o o o x  o  o  o
## 5 x x x x x o o o o o x  o  o  o
## 6 x x o x x x o o o o o  o  o  o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

ACF and PACF with absolute values and returns

There are significant lags in both ACF and PACF with deteoriating trend . From Eacf we get ARMA(2,2)& ARMA(2,3)

```r
#ARMA(2,2)(2,3)
par(mfrow=c(1,2))
acf(abs(first.diff),main="ACF of the \n
    Absolute Daily Bitcoin Price")
pacf(abs(first.diff),main="PACF of the Absolute Daily Bitcoin Price")
```





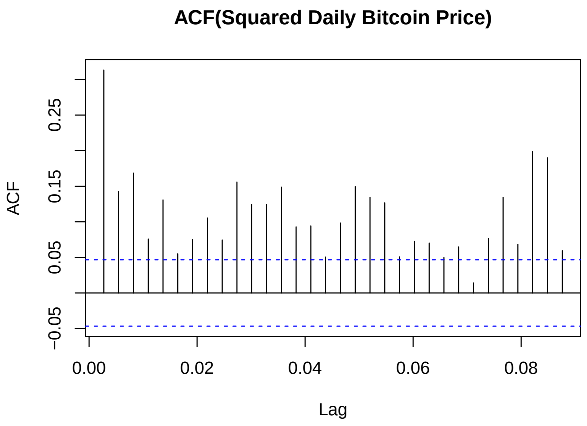
```
eacf(abs(first.diff))
```

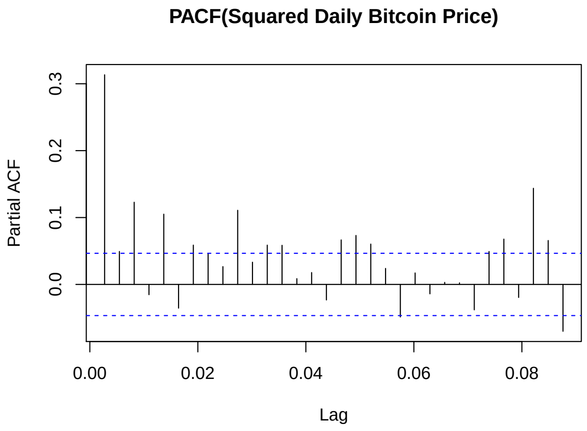
```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x x x x x x x  x  x  x 
## 1 x o o x x x o o o x o  o  o  o 
## 2 x x o o o o o o o x o  o  o  o 
## 3 x x x o o o o o o x o  o  o  o 
## 4 x x x x o o o o o x o  o  o  o 
## 5 x x x x x o o o o o o  o  o  o 
## 6 x x x x x o o o o o o  o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

ACF and PACF for the squared absolute values

It was squared and we found out that there were no obvious models due to volitality clustering which is more visible for the squared values there is fuziness due to high variablilty

```
par(mfrow=c(1,2))
acf(first.diff^2,main="ACF(Squared Daily Bitcoin Price)")
pacf(first.diff^2,main="PACF(Squared Daily Bitcoin Price)")
```



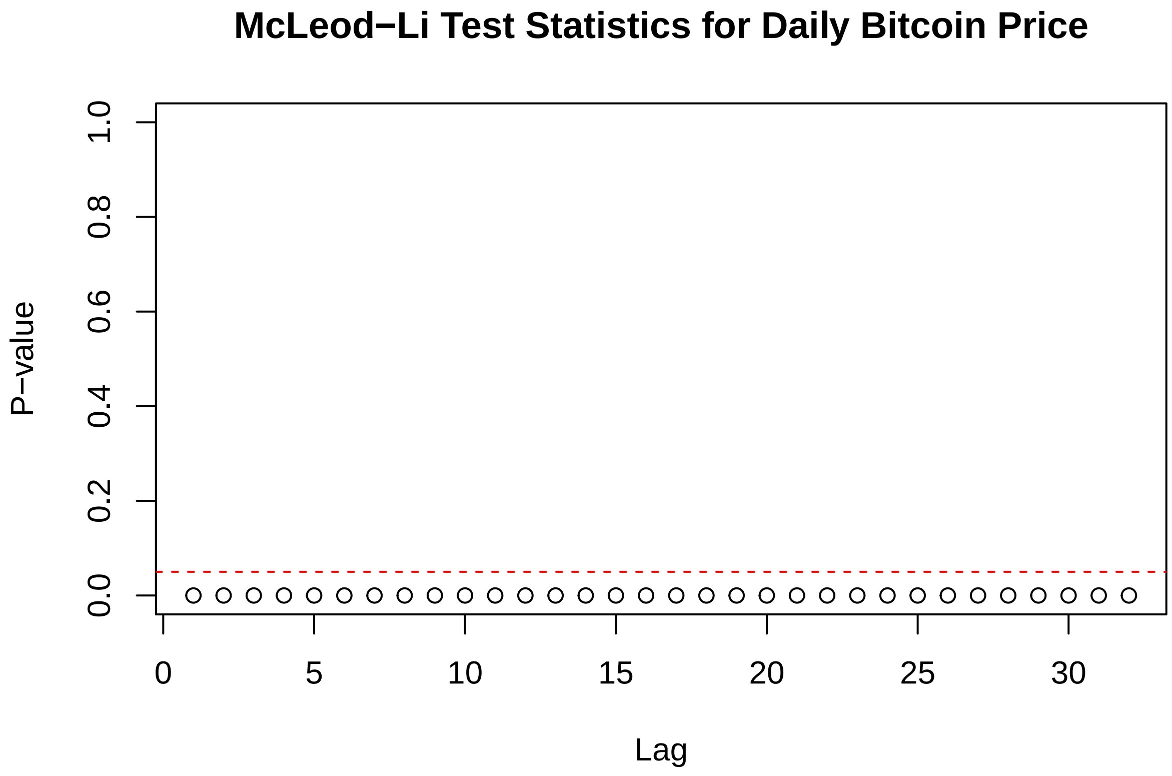


```r
eacf(first.diff^2)
```

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

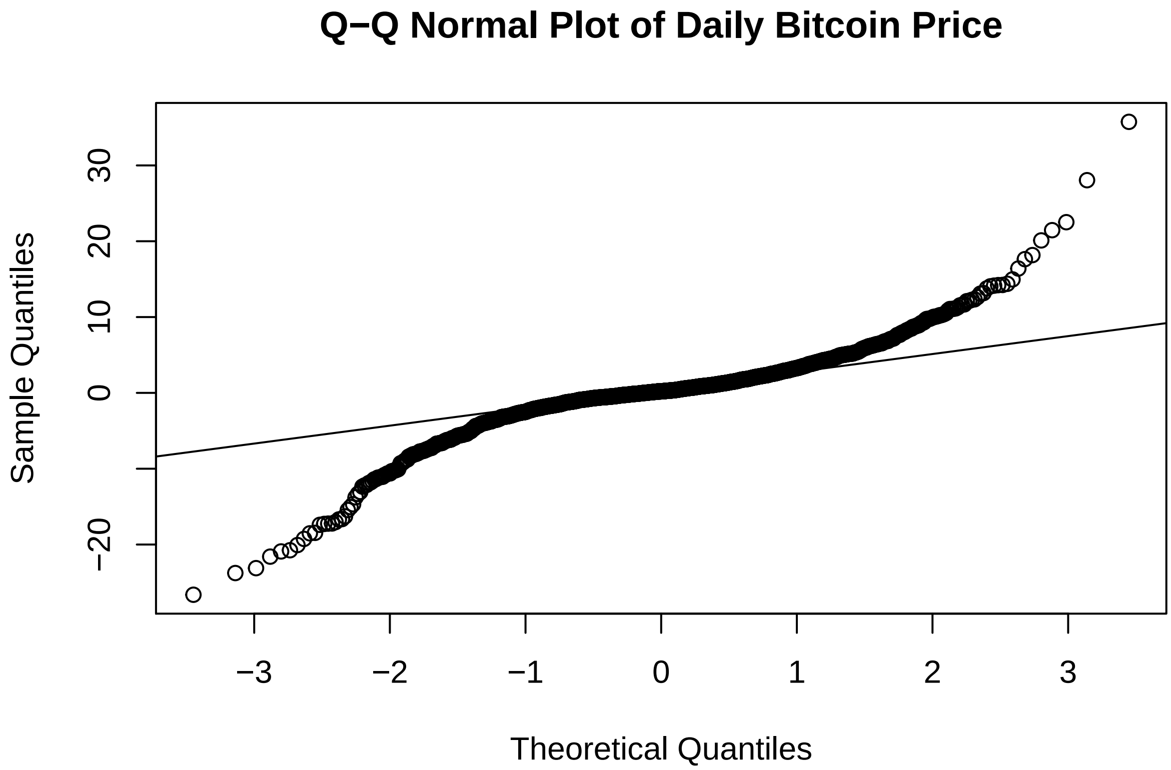
McLeod Li Test

```r
McLeod.Li.test(y=first.diff,main="McLeod-Li Test Statistics for Daily Bitcoin Price")
```



#Shapiro Test

```r
qqnorm(first.diff,main="Q-Q Normal Plot of Daily Bitcoin Price")
qqline(first.diff)
```



```
shapiro.test(first.diff)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  first.diff
## W = 0.87737, p-value < 2.2e-16
```

Estimation of Parameters

```
model1 = garch(first.diff,order=c(2,2),trace = FALSE)
summary(model1) # All the coefficients are significant at 5% level of significance.
```

```
##
## Call:
## garch(x = first.diff, order = c(2, 2), trace = FALSE)
##
## Model:
## GARCH(2,2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.24961 -0.26953  0.04809  0.44019  5.76487
##
## Coefficient(s):
```

```
##     Estimate  Std. Error  t value Pr(>|t|)    
## a0 1.619e+01   1.322e+00   12.249  < 2e-16 ***
## a1 1.538e-01   1.525e-02   10.084  < 2e-16 ***
## a2 1.125e-01   2.701e-02    4.165 3.11e-05 ***
## b1 2.788e-13   1.150e-01    0.000    1.000    
## b2 7.301e-03   5.967e-02    0.122    0.903    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Diagnostic Tests:
##  Jarque Bera Test
## 
## data:  Residuals
## X-squared = 2757.7, df = 2, p-value < 2.2e-16
## 
## 
##  Box-Ljung test
## 
## data:  Squared.Residuals
## X-squared = 6.9254, df = 1, p-value = 0.008498
```

```
model1.1 = garchFit(formula = ~garch(2,2), data =first.diff )
```

```
## 
## Series Initialization:
##  ARMA Model:                arma
##  Formula Mean:              ~ arma(0, 0)
##  GARCH Model:               garch
##  Formula Variance:          ~ garch(2, 2)
##  ARMA Order:                0 0
##  Max ARMA Order:            0
##  GARCH Order:               2 2
##  Max GARCH Order:           2
##  Maximum Order:             2
##  Conditional Dist:          norm
##  h.start:                   3
##  llh.start:                 1
##  Length of Series:          1771
##  Recursion Init:            mci
##  Series Scale:              4.499265
## 
## Parameter Initialization:
##  Initial Parameters:          $params
##  Limits of Transformations:   $U, $V
##  Which Parameters are Fixed?  $includes
##  Parameter Matrix:
##                      U           V     params includes
##     mu     -0.55869489   0.5586949 0.05586949     TRUE
##     omega   0.00000100 100.0000000 0.10000000     TRUE
##     alpha1  0.00000001   1.0000000 0.05000000     TRUE
##     alpha2  0.00000001   1.0000000 0.05000000     TRUE
##     gamma1 -0.99999999   1.0000000 0.10000000    FALSE
##     gamma2 -0.99999999   1.0000000 0.10000000    FALSE
##     beta1   0.00000001   1.0000000 0.40000000     TRUE
##     beta2   0.00000001   1.0000000 0.40000000     TRUE
##     delta   0.00000000   2.0000000 2.00000000    FALSE
##     skew    0.10000000  10.0000000 1.00000000    FALSE
##     shape   1.00000000  10.0000000 4.00000000    FALSE
##  Index List of Parameters to be Optimized:
##     mu  omega alpha1 alpha2  beta1  beta2
##      1      2      3      4      7      8
##  Persistence:                  0.9
##
##
## --- START OF TRACE ---
## Selected Algorithm: nlminb
##
## R coded nlminb Solver:
##
##   0:     2276.2288: 0.0558695 0.100000 0.0500000 0.0500000 0.400000 0.400000
##   1:     2247.9835: 0.0558672 0.0773139 0.0552636 0.0495434 0.387486 0.387320
##   2:     2229.9289: 0.0558634 0.0730970 0.0784165 0.0648966 0.393423 0.393248
##   3:     2218.0767: 0.0558605 0.0533761 0.0800700 0.0627795 0.384556 0.384243
##   4:     2205.0845: 0.0558535 0.0502299 0.0981058 0.0738019 0.391533 0.391348
##   5:     2202.4562: 0.0558501 0.0377075 0.100118 0.0728041 0.387744 0.387514
##   6:     2196.6705: 0.0558411 0.0367573 0.109779 0.0765010 0.394045 0.394093
##   7:     2196.0359: 0.0558096 0.0159333 0.124401 0.0725377 0.400714 0.401552
##   8:     2193.7579: 0.0557909 0.0255357 0.133124 0.0688645 0.402487 0.403974
##   9:     2190.5286: 0.0557640 0.0247611 0.141623 0.0609174 0.396879 0.399154
##  10:     2188.3260: 0.0556220 0.0226784 0.164804 0.0241805 0.398597 0.407603
##  11:     2188.2937: 0.0556195 0.0231320 0.165737 0.0245337 0.399285 0.408409
##  12:     2188.1837: 0.0555974 0.0218661 0.165908 0.0242648 0.398880 0.408956
##  13:     2188.1089: 0.0555267 0.0214863 0.167096 0.0243548 0.398238 0.411347
##  14:     2188.0100: 0.0553698 0.0210424 0.169106 0.0247048 0.393661 0.413343
##  15:     2187.9338: 0.0552215 0.0220237 0.171160 0.0252734 0.389174 0.415466
##  16:     2187.3061: 0.0507429 0.0165422 0.166222 0.0128693 0.283248 0.542611
##  17:     2187.0140: 0.0487850 0.0162417 0.171257 0.0107668 0.228997 0.588527
##  18:     2186.7091: 0.0467236 0.0189561 0.178438 0.0112072 0.177608 0.635940
##  19:     2186.2440: 0.0462731 0.0164278 0.183094 0.0149871 0.202007 0.610285
##  20:     2186.1120: 0.0462693 0.0173464 0.182534 0.0139261 0.199742 0.607798
##  21:     2185.9488: 0.0462196 0.0188431 0.182685 0.0141824 0.202405 0.605955
##  22:     2185.8951: 0.0460175 0.0185454 0.182871 0.0140651 0.201961 0.605520
##  23:     2185.8338: 0.0456243 0.0194903 0.183996 0.0143936 0.200350 0.605738
##  24:     2185.7518: 0.0448482 0.0190714 0.185207 0.0143695 0.196300 0.608332
##  25:     2185.5431: 0.0413050 0.0201658 0.186304 0.0130331 0.211125 0.592379
##  26:     2185.4015: 0.0384602 0.0200734 0.192670 0.0142110 0.189568 0.609189
##  27:     2185.3880: 0.0384596 0.0200201 0.192798 0.0143981 0.189140 0.608590
##  28:     2185.3731: 0.0384943 0.0203729 0.192990 0.0146102 0.189093 0.608605
##  29:     2185.3590: 0.0385694 0.0202934 0.193226 0.0148692 0.188580 0.608172
##  30:     2185.3404: 0.0387366 0.0206605 0.193632 0.0152358 0.188179 0.608140
##  31:     2185.3147: 0.0390718 0.0206960 0.194319 0.0157919 0.187089 0.607666
##  32:     2185.2064: 0.0407879 0.0221875 0.198697 0.0189990 0.182030 0.605577
##  33:     2185.1177: 0.0369560 0.0229093 0.198816 0.0229007 0.181939 0.601478
##  34:     2185.1004: 0.0367706 0.0227510 0.202997 0.0186907 0.192939 0.591298
##  35:     2185.0929: 0.0375845 0.0231102 0.203887 0.0237792 0.190509 0.589493
##  36:     2185.0864: 0.0367108 0.0235883 0.204958 0.0232956 0.186372 0.592255
##  37:     2185.0858: 0.0368695 0.0234028 0.204561 0.0224509 0.187816 0.591901
##  38:     2185.0858: 0.0368549 0.0234244 0.204606 0.0225859 0.187632 0.591936
##  39:     2185.0858: 0.0368548 0.0234241 0.204605 0.0225851 0.187631 0.591938
```

```
##
## Final Estimate of the Negative LLH:
##  LLH:  4848.518    norm LLH:  2.737729
##        mu     omega    alpha1    alpha2     beta1     beta2
## 0.1658196 0.4741829 0.2046053 0.0225851 0.1876306 0.5919379
##
## R-optimhess Difference Approximated Hessian Matrix:
##                  mu        omega       alpha1       alpha2        beta1
## mu     -198.2453653    -3.392661     30.04548     59.84732     11.09326
## omega    -3.3926605  -288.583367  -1273.75572  -1384.59887  -2419.61436
## alpha1   30.0454832 -1273.755722 -10320.37133 -10738.82758 -15318.68352
## alpha2   59.8473165 -1384.598874 -10738.82758 -13459.23250 -17398.10908
## beta1    11.0932638 -2419.614362 -15318.68352 -17398.10908 -26558.75189
## beta2    -0.2774075 -2465.219489 -15517.94037 -17222.51636 -26865.79322
##                beta2
## mu     -2.774075e-01
## omega  -2.465219e+03
## alpha1 -1.551794e+04
## alpha2 -1.722252e+04
## beta1  -2.686579e+04
## beta2  -2.749136e+04
## attr(,"time")
## Time difference of 0.2740159 secs
##
## --- END OF TRACE ---
##
##
## Time to Estimate Parameters:
##  Time difference of 1.063061 secs
```

```r
summary(model1.1)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(2, 2), data = first.diff)
##
## Mean and Variance Equation:
##  data ~ garch(2, 2)
## <environment: 0x0000000017362398>
##  [data = first.diff]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##       mu     omega    alpha1    alpha2     beta1     beta2
## 0.165820  0.474183  0.204605  0.022585  0.187631  0.591938
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
```

```
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       0.16582     0.07128    2.326  0.02000 *
## omega    0.47418     0.14796    3.205  0.00135 **
## alpha1   0.20461     0.02995    6.832 8.38e-12 ***
## alpha2   0.02259     0.03160    0.715  0.47477
## beta1    0.18763     0.08186    2.292  0.02189 *
## beta2    0.59194     0.06674    8.869  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  -4848.518    normalized:  -2.737729
##
## Description:
##  Sun Jun 03 16:57:12 2018 by user: Wel
##
##
## Standardised Residuals Tests:
##                                 Statistic p-Value
##  Jarque-Bera Test   R    Chi^2  5385.597  0
##  Shapiro-Wilk Test  R    W      0.9063864 0
##  Ljung-Box Test     R    Q(10)  37.96593  3.847341e-05
##  Ljung-Box Test     R    Q(15)  44.46622  9.292405e-05
##  Ljung-Box Test     R    Q(20)  54.66252  4.611051e-05
##  Ljung-Box Test     R^2  Q(10)  7.565348  0.6712095
##  Ljung-Box Test     R^2  Q(15)  10.16227  0.8094103
##  Ljung-Box Test     R^2  Q(20)  12.73401  0.8885027
##  LM Arch Test       R    TR^2   8.543814  0.7413215
##
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC
## 5.482233 5.500797 5.482210 5.489091
```

```
model2 = garch(first.diff,order=c(2,3),trace = FALSE)
summary(model2 )# All the coefficients but aplha_2 are significant at 5% level of significance.
```

```
##
## Call:
## garch(x = first.diff, order = c(2, 3), trace = FALSE)
##
## Model:
## GARCH(2,3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.33771 -0.27766  0.04911  0.45160  5.94652
##
## Coefficient(s):
##     Estimate  Std. Error  t value Pr(>|t|)
## a0 1.518e+01   2.209e+00    6.873   6.3e-12 ***
## a1 1.459e-01   1.428e-02   10.213   < 2e-16 ***
## a2 9.227e-02   3.074e-02    3.002   0.00269 **
## a3 7.683e-02   2.490e-02    3.085   0.00203 **
## b1 2.133e-13   1.668e-01    0.000   1.00000
## b2 1.179e-02   6.737e-02    0.175   0.86104
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Diagnostic Tests:
##  Jarque Bera Test
##
## data:  Residuals
## X-squared = 2928.7, df = 2, p-value < 2.2e-16
##
##
##  Box-Ljung test
##
## data:  Squared.Residuals
## X-squared = 8.2223, df = 1, p-value = 0.004138
```

```
model2.1 = garchFit(formula = ~garch(3,2), data =first.diff, trace = FALSE )
summary(model2.1)
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(3, 2), data = first.diff, trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(3, 2)
## <environment: 0x00000000118785c0>
##  [data = first.diff]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##         mu       omega      alpha1      alpha2      alpha3       beta1
## 0.16218100  0.45510254  0.20771297  0.01217754  0.00000001  0.20654062
##      beta2
## 0.57978910
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu     1.622e-01   7.140e-02    2.271   0.0231 *
## omega  4.551e-01   1.837e-01    2.478   0.0132 *
## alpha1 2.077e-01   3.337e-02    6.225 4.82e-10 ***
## alpha2 1.218e-02   3.208e-02    0.380   0.7042
## alpha3 1.000e-08   5.006e-02    0.000   1.0000
## beta1  2.065e-01   8.814e-02    2.343   0.0191 *
## beta2  5.798e-01   7.864e-02    7.373 1.67e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
```

```
##  -4848.6    normalized:  -2.737775 
## 
## Description:
##  Sun Jun 03 16:57:13 2018 by user: Wel 
## 
## 
## Standardised Residuals Tests:
##                                 Statistic p-Value     
##  Jarque-Bera Test   R    Chi^2  5336.825  0           
##  Shapiro-Wilk Test  R    W      0.9065387 0           
##  Ljung-Box Test     R    Q(10)  37.71507  4.253901e-05
##  Ljung-Box Test     R    Q(15)  44.20358  0.0001021775
##  Ljung-Box Test     R    Q(20)  54.35771  5.118887e-05
##  Ljung-Box Test     R^2  Q(10)  7.561231  0.6716092   
##  Ljung-Box Test     R^2  Q(15)  10.07275  0.8151391   
##  Ljung-Box Test     R^2  Q(20)  12.66779  0.8911786   
##  LM Arch Test       R    TR^2   8.559354  0.7400351   
## 
## Information Criterion Statistics:
##      AIC      BIC      SIC     HQIC 
## 5.483456 5.505113 5.483425 5.491457
```

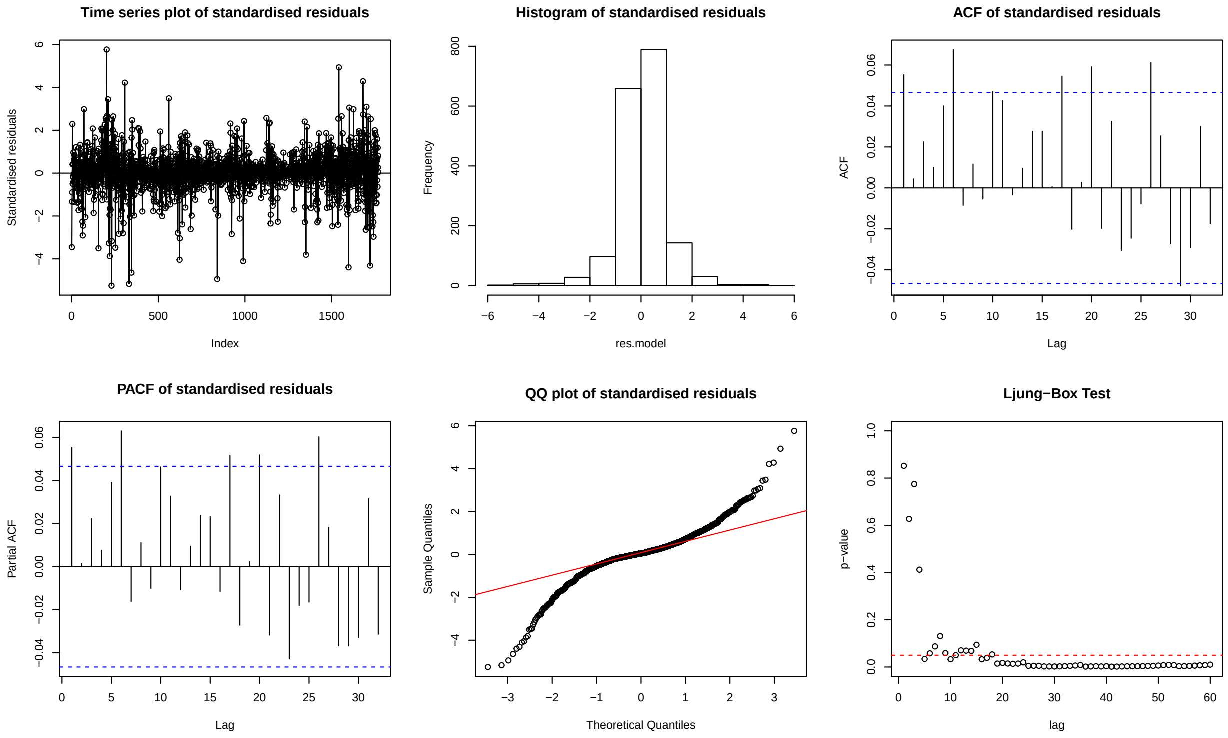
Residual Analysis

The residual analysis for both the models are almost same, by lookin at the time series of the residual analysis we can say that the series is stochastic and without a trend, the histogram is almost normally distributed and the PACF and ACF plots have significant lags and the qq plot for standardised residuals is normally distributed

```r
residual.analysis(model1,class="GARCH",start=3)
```

```
## 
##  Shapiro-Wilk normality test
## 
## data:  res.model
## W = 0.90436, p-value < 2.2e-16
```

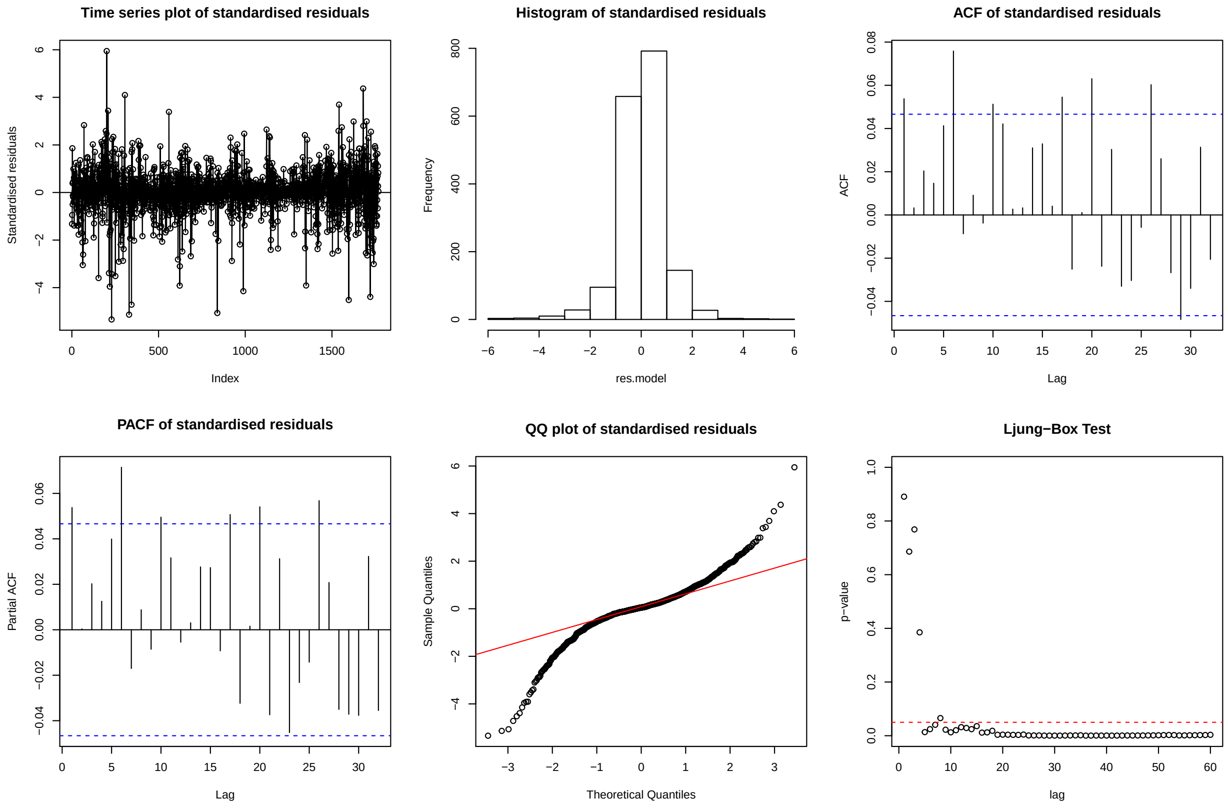
```
## Warning in (ra^2)/(n - (1:lag.max)): longer object length is not a multiple
## of shorter object length
```



```r
residual.analysis(model2,class="GARCH",start=4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.90502, p-value < 2.2e-16
```

```
## Warning in (ra^2)/(n - (1:lag.max)): longer object length is not a multiple
## of shorter object length
```



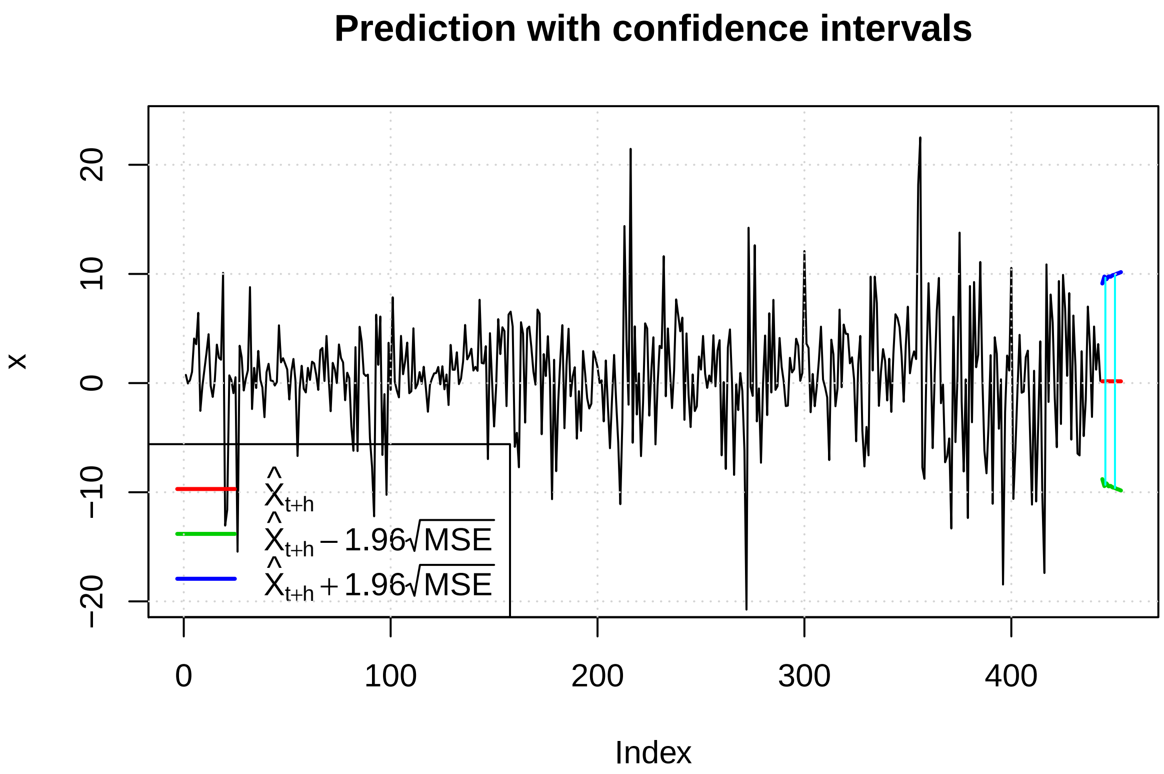
Shapiro Test

```
shapiro.test(first.diff)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  first.diff
## W = 0.87737, p-value < 2.2e-16
```

Forecast

```
forecast <- fGarch::predict(model2.1,n.ahead=10,trace=FALSE,plot=TRUE)
```


Prediction with confidence intervals
x
Index
$\hat{X}_{t+h}$
$\hat{X}_{t+h} - 1.96\sqrt{MSE}$
$\hat{X}_{t+h} + 1.96\sqrt{MSE}$