Financial Data Analytics
Final Project Assignment

Marklina Yakopa

November 19th 2024

```
#"Answers-to-Questions-1-to-7"
```

1. Exploratory data analysis I: 15 points

```
### Problem 1
da=read.table("d-sbux3dx-0715.txt",header=T)
head(da)

##   PERMNO     date      SBUX    vwretd    ewretd    sprtrn
## 1  77702 20070103 -0.004799 -0.001347 -0.000159 -0.001199
## 2  77702 20070104  0.001135  0.000547  0.000591  0.001228
## 3  77702 20070105 -0.004251 -0.007288 -0.009809 -0.006085
## 4  77702 20070108 -0.003700  0.002567  0.001731  0.002220
## 5  77702 20070109 -0.004284 -0.000001  0.000262 -0.000517
## 6  77702 20070110 -0.003155  0.002096  0.001338  0.001940

rtn=da[,3:6]
attach(rtn)
require(fBasics)

## Loading required package: fBasics

# (a)
basicStats(SBUX)

##                    SBUX
## nobs        2266.000000
## NAs            0.000000
## Minimum       -0.109742
## Maximum        0.183798
## 1. Quartile   -0.009258
## 3. Quartile    0.010326
## Mean           0.000801
## Median         0.000523
## Sum            1.815383
## SE Mean        0.000447
## LCL Mean      -0.000076
## UCL Mean       0.001678
## Variance       0.000453
## Stdev          0.021289
## Skewness       0.555707
## Kurtosis       6.575662

basicStats(vwretd)
```

```
##                   vwretd
## nobs         2266.000000
## NAs             0.000000
## Minimum        -0.089763
## Maximum         0.114898
## 1. Quartile    -0.004844
## 3. Quartile     0.006177
## Mean            0.000318
## Median          0.000857
## Sum             0.720300
## SE Mean         0.000286
## LCL Mean       -0.000244
## UCL Mean        0.000879
## Variance        0.000186
## Stdev           0.013630
## Skewness       -0.175139
## Kurtosis        8.807872
```

```
basicStats(ewretd)
```

```
##                   ewretd
## nobs         2266.000000
## NAs             0.000000
## Minimum        -0.078240
## Maximum         0.107422
## 1. Quartile    -0.004676
## 3. Quartile     0.005991
## Mean            0.000415
## Median          0.001174
## Sum             0.940663
## SE Mean         0.000263
## LCL Mean       -0.000101
## UCL Mean        0.000931
## Variance        0.000157
## Stdev           0.012521
## Skewness       -0.209028
## Kurtosis        7.765289
```

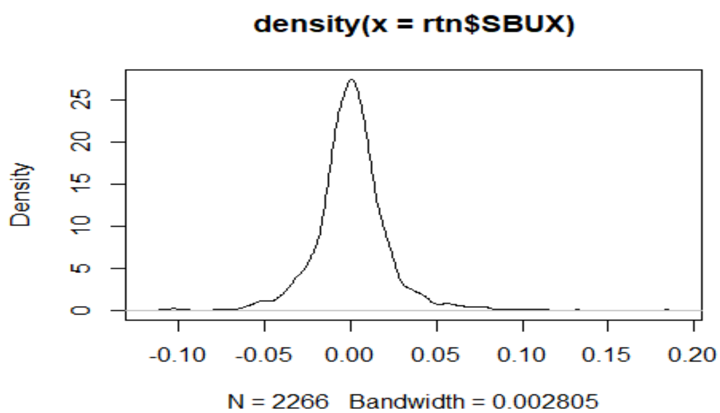
```
basicStats(sprtrn)
```

```
##                   sprtrn
## nobs         2266.000000
## NAs             0.000000
## Minimum        -0.090350
## Maximum         0.115800
## 1. Quartile    -0.004806
## 3. Quartile     0.006093
## Mean            0.000254
## Median          0.000688
## Sum             0.575829
## SE Mean         0.000286
```

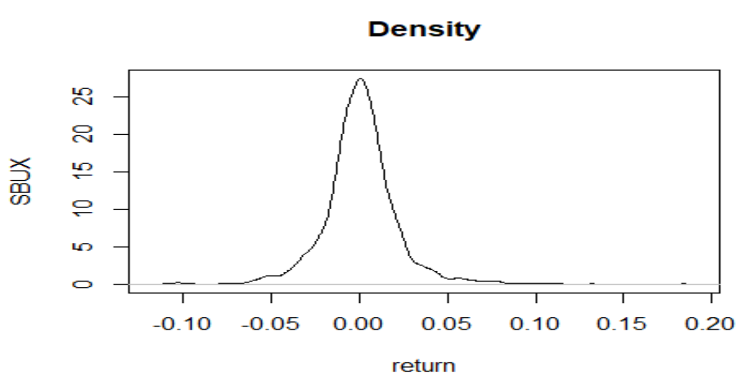
```
## LCL Mean       -0.000307
## UCL Mean        0.000815
## Variance        0.000185
## Stdev           0.013619
## Skewness       -0.078865
## Kurtosis        9.730943
```

Simple return series show comparable average returns, volatility, symmetry, and tail characteristics, with similar minimum and maximum values.

```
# (b)
densitySBUX=density(rtn$SBUX)
plot(densitySBUX)
```




```
# This is the same density plot but with labels
plot(densitySBUX,xlab='return',ylab='SBUX',main='Density')
```




```
normalTest(SBUX)

##
## Title:
##  Shapiro - Wilk Normality Test
```

```
##
## Test Results:
##   STATISTIC:
##     W: 0.9251
##   P VALUE:
##     < 2.2e-16
```

Simple returns of Starbucks stock appear to follow a normal distribution, as indicated by the normality test.

```
# (c)
lrtn=log(rtn+1) ### log returns
basicStats(lrtn$SBUX)
```

```
##              X..lrtn.SBUX
## nobs          2266.000000
## NAs              0.000000
## Minimum         -0.116244
## Maximum          0.168728
## 1. Quartile     -0.009301
## 3. Quartile      0.010273
## Mean             0.000576
## Median           0.000523
## Sum              1.305183
## SE Mean          0.000445
## LCL Mean        -0.000297
## UCL Mean         0.001449
## Variance         0.000449
## Stdev            0.021185
## Skewness         0.302033
## Kurtosis         5.910134
```

```
basicStats(lrtn$vwretd)
```

```
##              X..lrtn.vwretd
## nobs            2266.000000
## NAs                0.000000
## Minimum           -0.094050
## Maximum            0.108763
## 1. Quartile       -0.004856
## 3. Quartile        0.006158
## Mean               0.000225
## Median             0.000857
## Sum                0.509376
## SE Mean            0.000287
## LCL Mean          -0.000338
## UCL Mean           0.000787
## Variance           0.000186
## Stdev              0.013655
## Skewness          -0.392742
## Kurtosis           8.693544
```

```r
basicStats(lrtn$ewretd)
```

```
##               X..lrtn.ewretd
## nobs             2266.000000
## NAs                 0.000000
## Minimum            -0.081470
## Maximum             0.102035
## 1. Quartile        -0.004687
## 3. Quartile         0.005973
## Mean                0.000337
## Median              0.001173
## Sum                 0.762614
## SE Mean             0.000263
## LCL Mean           -0.000180
## UCL Mean            0.000853
## Variance            0.000157
## Stdev               0.012541
## Skewness           -0.389396
## Kurtosis            7.661540
```

```r
basicStats(lrtn$sprtrn)
```

```
##               X..lrtn.sprtrn
## nobs             2266.000000
## NAs                 0.000000
## Minimum            -0.094695
## Maximum             0.109572
## 1. Quartile        -0.004818
## 3. Quartile         0.006074
## Mean                0.000161
## Median              0.000688
## Sum                 0.365424
## SE Mean             0.000286
## LCL Mean           -0.000401
## UCL Mean            0.000723
## Variance            0.000186
## Stdev               0.013637
## Skewness           -0.314434
## Kurtosis            9.496296
```

The log return series of Starbucks stock shows a stable average return, moderate volatility, positive skewness (indicating a right tail), and kurtosis close to normal, with similar minimum and maximum values across the sample, while the other series display negative skewness.

```r
# (d)
t.test(lrtn$SBUX)
```

```
##
##  One Sample t-test
##
## data:  lrtn$SBUX
```

```
## t = 1.2942, df = 2265, p-value = 0.1957
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.0002967458  0.0014487168
## sample estimates:
##    mean of x
## 0.0005759855
```

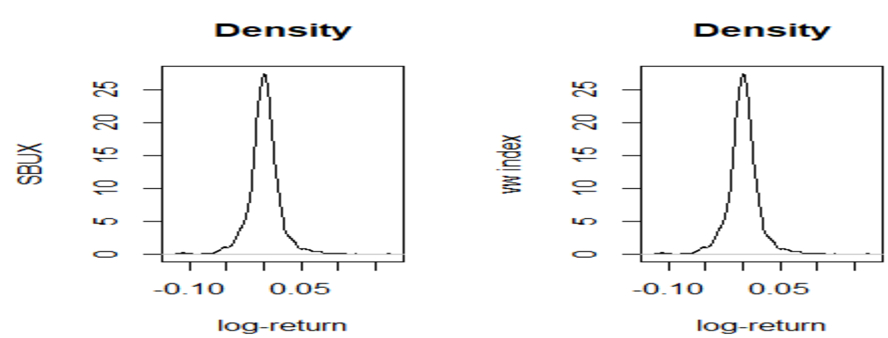
```
t.test(lrtn$sprtrn)
```

```
##
##  One Sample t-test
##
## data:  lrtn$sprtrn
## t = 0.56293, df = 2265, p-value = 0.5735
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.0004005110  0.0007230388
## sample estimates:
##    mean of x
## 0.0001612639
```

The mean log returns for both Starbucks stock and the S&P Composite Index are not significantly different from zero.

The mean log returns for both Starbucks stock and the S&P Composite Index are not significantly different from zero.

```
# (e)
densitylogSBUX=density(lrtn$SBUX)
densitylogvwretd=density(lrtn$vwretd)
par(mfcol=c(1,2)) ## Put two plots in a frame (left and right)
plot(densitylogSBUX,xlab='log-return',ylab='SBUX',main='Density')
plot(densitylogSBUX,xlab='log-return',ylab='vw index',main='Density')
```




```
par(mfcol=c(1,1))
```

The empirical density plots of daily log return for Starbucks stock and the value-weighted index show similar central peaks, with Starbucks displaying slight positive skewness compared to a more symmetrical distribution in the index.

2. Factor models: 30 points

```
# Problem 2
FFfactors <- read.csv("C:/Users/Administrator/Desktop/Assignment for 
Marklina/FamaFrenchFactors (1).csv",header=T) # Load csv data with names.
head(FFfactors) # See the first 6 rows

##        X   Mkt   SMB   HML
## 1 196307 -0.39 -0.46 -0.82
## 2 196308  5.07 -0.85  1.63
## 3 196309 -1.57 -0.50  0.19
## 4 196310  2.53 -1.30 -0.11
## 5 196311 -0.85 -0.83  1.66
## 6 196312  1.83 -1.87 -0.11

attach(FFfactors) #Now you can refer directly to the names of the variables.
#Note that you might receive a warning: "The following objects are masked 
from FFfactors ..."
# This just means that you are overwriting variables with these names that 
you have used before.
# For our purposes you can simply ignore this message.
names(FFfactors)

## [1] "X"   "Mkt" "SMB" "HML"

rf <- read.csv("C:/Users/Administrator/Desktop/Assignment for 
Marklina/RiskFreeRate.csv",header=T) # Load csv data with names.
head(rf) # See the first 6 rows

##        X   RF
## 1 196307 0.27
## 2 196308 0.25
## 3 196309 0.27
## 4 196310 0.29
## 5 196311 0.27
## 6 196312 0.29

sv <- read.csv("C:/Users/Administrator/Desktop/Assignment for 
Marklina/SizeValuePortfolios.csv",header=T) # Load csv data with names.
head(sv) # See the first 6 rows

##        X SMALL.LoBM ME1.BM2 ME1.BM3 ME1.BM4 SMALL.HiBM ME2.BM1 ME2.BM2 
ME2.BM3
## 1 196307       0.85    0.24    0.56   -0.02      -1.22   -1.87    0.29   -
0.84
## 2 196308       3.80    2.15    1.32    2.29       4.73    5.40    4.65    
4.36
## 3 196309      -2.70    0.26   -1.09   -1.59      -0.38   -3.92   -1.57   -
0.64
## 4 196310       1.36   -0.63    1.24    0.05       2.37    1.19    4.30    
2.34
## 5 196311      -3.11   -4.32   -1.60   -1.00      -1.11   -4.18   -1.76   -
```

```
0.71
## 6 196312      -2.94   -0.55   -0.90   -1.75      -0.97   -0.68   -0.91
1.15
##   ME2.BM4 ME2.BM5 ME3.BM1 ME3.BM2 ME3.BM3 ME3.BM4 ME3.BM5 ME4.BM1 ME4.BM2
## 1   -1.90   -1.19   -1.84   -1.87   -0.81   -2.21   -1.80   -0.93   -1.63
## 2    4.33    8.23    5.36    4.62    5.63    4.72    5.15    5.59    4.81
## 3   -1.13   -2.91   -4.65   -1.52   -0.69   -0.10   -1.82   -2.67   -1.95
## 4    2.26    3.93    2.26    0.50    2.70    2.16    1.07   -0.25    0.93
## 5   -0.09   -0.11   -2.92   -1.58   -0.91   -0.90   -1.21   -0.91   -0.86
## 6    0.97    0.12    0.60    1.17    1.52    1.52    0.47   -0.17    1.27
##   ME4.BM3 ME4.BM4 ME4.BM5 BIG.LoBM ME5.BM2 ME5.BM3 ME5.BM4 BIG.HiBM
## 1   -2.07   -1.67   -1.86     0.14    0.46    1.23   -0.45    -1.11
## 2    6.12    7.56    5.35     5.77    4.22    4.77    8.16     6.25
## 3   -2.00   -3.58   -1.99    -1.37   -0.77   -0.98   -0.12    -3.82
## 4    2.30    5.34    0.61     5.33    1.73   -0.26    2.36     0.48
## 5   -0.51    1.15    3.54    -1.26    0.98   -1.55   -2.05     1.37
## 6    1.63    3.14    6.15     2.62    2.30    2.28    2.38     2.37
```

```
names(sv)
```

```
##  [1] "X"          "SMALL.LoBM" "ME1.BM2"    "ME1.BM3"    "ME1.BM4"
##  [6] "SMALL.HiBM" "ME2.BM1"    "ME2.BM2"    "ME2.BM3"    "ME2.BM4"
## [11] "ME2.BM5"    "ME3.BM1"    "ME3.BM2"    "ME3.BM3"    "ME3.BM4"
## [16] "ME3.BM5"    "ME4.BM1"    "ME4.BM2"    "ME4.BM3"    "ME4.BM4"
## [21] "ME4.BM5"    "BIG.LoBM"   "ME5.BM2"    "ME5.BM3"    "ME5.BM4"
## [26] "BIG.HiBM"
```

```
Industry=read.csv("C:/Users/Administrator/Desktop/Assignment for
Marklina/IndustryPortfolios.csv",header=T) # Load csv data with names.
head(Industry) # See the first 6 rows
```

```
##        X  Food  Beer Smoke Games Books Hshld Clths  Hlth Chems Txtls Cnstr
## 1 196307  0.05 -2.19 -2.54 -3.48 -0.04 -0.15 -0.42  0.56 -1.00  3.49 -0.78
## 2 196308  4.82  2.14  7.15  6.08  4.66  6.22  4.59  9.56  4.61  4.30  6.04
## 3 196309 -1.43  1.23 -4.23 -4.05  2.67 -1.37 -3.97 -4.06 -0.78 -1.14 -1.25
## 4 196310  2.41 -1.28  5.78 10.34 -1.15  4.25  2.70  3.38  3.48  5.72 -0.52
## 5 196311 -0.44 -0.72 -5.63 -1.96 -0.18 -0.18 -1.44 -1.65  2.22  1.99 -1.05
## 6 196312  2.99  1.21  5.98 -2.13 -0.21  4.26 -0.50  1.54  4.51  3.30  0.81
##   Steel FabPr ElcEq Autos Carry Mines  Coal   Oil  Util Telcm Servs BusEq
Paper
## 1 -1.59 -3.08 -2.43 -0.15 -3.67 -2.90  0.32  2.34  0.81 -0.23 -3.48 -0.55
-1.38
## 2  8.31  4.62  5.17  7.20  3.27  4.02  7.48  3.85  4.22  4.29  2.68  5.23
7.70
## 3  0.29 -2.55 -1.68  0.31 -4.60 -0.96 -4.48 -3.62 -2.49  2.36 -2.65  0.29
1.45
## 4  1.58  4.19 -0.49 10.14  2.28  3.16  9.45 -0.52 -0.67  3.40  1.51  8.66
1.65
## 5 -1.83 -0.79 -1.72 -5.42  5.54  1.56  0.67 -1.19 -1.02  4.16 -3.78 -0.27
-0.93
## 6  1.51  1.86 -2.55 -1.20 -4.22  6.05  5.42  4.65  2.42 -0.14  0.93  3.26
```

```
0.31
##   Trans Whlsl Rtail Meals   Fin Other
## 1 -2.54  0.04 -1.06 -4.21 -0.61 -2.63
## 2  7.37  4.22  6.62  1.07  4.28  5.63
## 3 -3.85 -1.65  1.15 -2.51 -3.25 -3.23
## 4  1.77  1.41  0.41  1.11  0.68  2.88
## 5  3.78 -4.03 -1.07 -3.61 -2.29  1.44
## 6  4.17 -0.66  0.35  1.75  1.61 -2.54
```

```
names(Industry)
```

```
##  [1] "X"     "Food"  "Beer"  "Smoke" "Games" "Books" "Hshld" "Clths"
"Hlth"
## [10] "Chems" "Txtls" "Cnstr" "Steel" "FabPr" "ElcEq" "Autos" "Carry"
"Mines"
## [19] "Coal"  "Oil"   "Util"  "Telcm" "Servs" "BusEq" "Paper" "Trans"
"Whlsl"
## [28] "Rtail" "Meals" "Fin"   "Other"
```

```
# (a)
Industry=Industry[,2:31]
attach(Industry) #Now you can refer directly to the names of the variables.
#Note that you might receive a warning: "The following objects are masked
from..."
# This just means that you are overwriting variables with these names that
you have used before.
# For our purposes you can simply ignore this message.
industrypca=prcomp(Industry)
summary(industrypca)
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6
PC7
## Standard deviation     26.6625 9.41932 7.08614 5.96143 5.87202 4.92238
4.53696
## Proportion of Variance  0.6195 0.07732 0.04376 0.03097 0.03005 0.02112
0.01794
## Cumulative Proportion   0.6195 0.69684 0.74060 0.77157 0.80162 0.82273
0.84067
##                            PC8     PC9    PC10    PC11    PC12    PC13
PC14
## Standard deviation     4.13409 4.00556 3.67210 3.55002 3.42637 3.27602
3.13419
## Proportion of Variance 0.01489 0.01398 0.01175 0.01098 0.01023 0.00935
0.00856
## Cumulative Proportion  0.85557 0.86955 0.88130 0.89228 0.90251 0.91187
0.92043
##                           PC15    PC16    PC17    PC18    PC19    PC20
PC21
## Standard deviation     3.00135 2.94679 2.86401 2.80948 2.61719 2.53213
2.45083
```

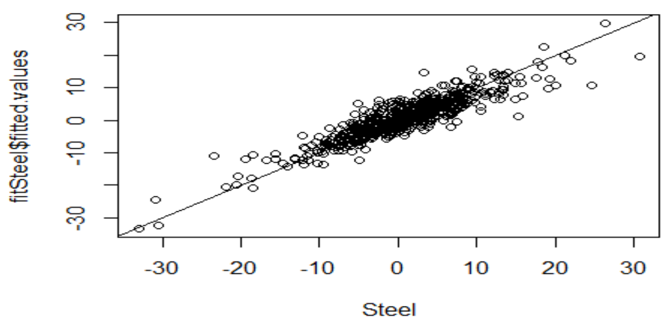
```
## Proportion of Variance 0.00785 0.00757 0.00715 0.00688 0.00597 0.00559
0.00523
## Cumulative Proportion  0.92828 0.93584 0.94299 0.94987 0.95584 0.96143
0.96666
##                           PC22    PC23    PC24   PC25   PC26    PC27
PC28
## Standard deviation     2.39389 2.27062 2.21289 2.1145 2.0600 2.01431
1.91264
## Proportion of Variance 0.00499 0.00449 0.00427 0.0039 0.0037 0.00354
0.00319
## Cumulative Proportion  0.97166 0.97615 0.98042 0.9843 0.9880 0.99155
0.99474
##                           PC29   PC30
## Standard deviation     1.78116 1.6934
## Proportion of Variance 0.00276 0.0025
## Cumulative Proportion  0.99750 1.0000

plot(industrypca)
```

While PC30 explains 100% of the variance; for 85% use PC8 factor, for 90% use PC12 and for 95% use PC19 factor where elbow of curve might occur.

```
# (b)
fitSteel=lm(Steel~industrypca$x[,1:3])
summary(fitSteel)

##
## Call:
## lm(formula = Steel ~ industrypca$x[, 1:3])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.5130  -1.8227   0.0946   1.8095  14.1687
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              0.767904   0.126789   6.057 2.45e-09 ***
## industrypca$x[, 1:3]PC1  0.227396   0.004759  47.780  < 2e-16 ***
## industrypca$x[, 1:3]PC2 -0.178089   0.013472 -13.220  < 2e-16 ***
```

```
## industrypca$x[, 1:3]PC3  0.322247   0.017907  17.995  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.121 on 602 degrees of freedom
## Multiple R-squared:  0.8221, Adjusted R-squared:  0.8212
## F-statistic: 927.2 on 3 and 602 DF,  p-value: < 2.2e-16
```

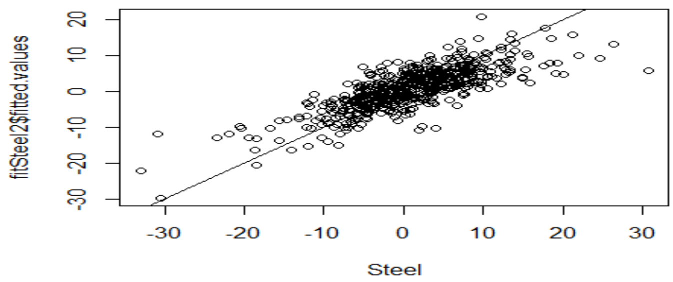
```
plot(Steel,fitSteel$fitted.values)
abline(c(0,1))
```



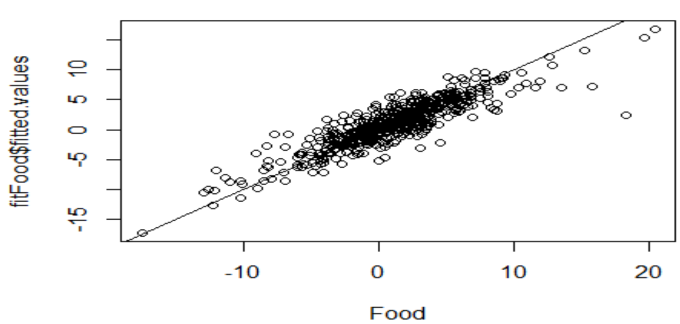

```
fitSteel2=lm(Steel~Mkt)
summary(fitSteel2)
```

```
##
## Call:
## lm(formula = Steel ~ Mkt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.1426  -2.8816  -0.1664   2.6262  24.9623
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.12522    0.18755   0.668    0.505
## Mkt          1.28925    0.04158  31.007   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.589 on 604 degrees of freedom
## Multiple R-squared:  0.6142, Adjusted R-squared:  0.6135
## F-statistic: 961.5 on 1 and 604 DF,  p-value: < 2.2e-16
```

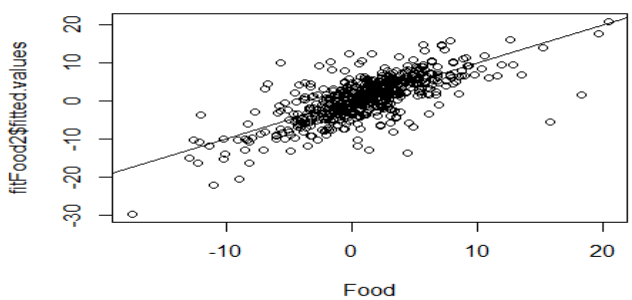
```
plot(Steel,fitSteel2$fitted.values)
abline(c(0,1))
```




```r
fitFood=lm(Food~industrypca$x[,1:3])
summary(fitFood)

##
## Call:
## lm(formula = Food ~ industrypca$x[, 1:3])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9077 -1.1683 -0.0056  0.9667 15.9115
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.074884   0.082316   13.06   <2e-16 ***
## industrypca$x[, 1:3]PC1  0.120394   0.003090   38.96   <2e-16 ***
## industrypca$x[, 1:3]PC2  0.093515   0.008746   10.69   <2e-16 ***
## industrypca$x[, 1:3]PC3 -0.279068   0.011626  -24.00   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.026 on 602 degrees of freedom
## Multiple R-squared:  0.7858, Adjusted R-squared:  0.7847
## F-statistic: 736.2 on 3 and 602 DF,  p-value: < 2.2e-16

plot(Food,fitFood$fitted.values)
abline(c(0,1))
```



```
fitFood2=lm(Steel~Mkt)
summary(fitFood2)

##
## Call:
## lm(formula = Steel ~ Mkt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.1426  -2.8816  -0.1664   2.6262  24.9623
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.12522    0.18755   0.668    0.505
## Mkt          1.28925    0.04158  31.007   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.589 on 604 degrees of freedom
## Multiple R-squared:  0.6142, Adjusted R-squared:  0.6135
## F-statistic: 961.5 on 1 and 604 DF,  p-value: < 2.2e-16

plot(Food,fitFood2$fitted.values)
abline(c(0,1))
```



From regression results using the first three principal components factors explains more of the variation in the steel and food industries' returns than just the market factor, with the PCA model providing a better fit to the actual data.

```
#(c)
svrf=sv[,2:26]-rf$RF
```

The second and fifteenth size-value sorted portfolios are constructed by combining rankings based on market capitalization and book-to-market ratios. The provided code calculates excess returns for these portfolios by subtracting the risk-free rate.

```r
# (d)
fit1=lm(svrf[[2]]~Mkt)
summary(fit1)
```

```
##
## Call:
## lm(formula = svrf[[2]] ~ Mkt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -26.844  -2.390  -0.242   2.012  34.923
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1971     0.1709   1.153    0.249
## Mkt           1.2283     0.0379  32.413   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.182 on 604 degrees of freedom
## Multiple R-squared:  0.635,  Adjusted R-squared:  0.6344
## F-statistic:  1051 on 1 and 604 DF,  p-value: < 2.2e-16
```

```r
fit3=lm(svrf[[15]]~Mkt)
summary(fit3)
```

```
##
## Call:
## lm(formula = svrf[[15]] ~ Mkt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.285  -1.732  -0.176   1.567  14.015
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.55721    0.12486   4.463 9.65e-06 ***
## Mkt          1.02837    0.02768  37.152  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.055 on 604 degrees of freedom
## Multiple R-squared:  0.6956, Adjusted R-squared:  0.6951
## F-statistic:  1380 on 1 and 604 DF,  p-value: < 2.2e-16
```

Interpretation of Results:

Portfolio 2:

The intercept (0.1971) is not statistically significant (p-value: 0.249), indicating that the average excess return of this portfolio is not significantly different from zero when the market excess return is zero.

Portfolio 15:

*The intercept (0.55721) is statistically significant (p-value: 9.65e-06 ***), suggesting that, on average, Portfolio 15 has a positive excess return even when the market excess return is zero.*

```
# (e)
fit1=lm(svrf[[2]]~Mkt)
summary(fit1)
```

```
##
## Call:
## lm(formula = svrf[[2]] ~ Mkt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -26.844  -2.390  -0.242   2.012  34.923
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1971     0.1709   1.153    0.249
## Mkt           1.2283     0.0379  32.413   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.182 on 604 degrees of freedom
## Multiple R-squared:  0.635,  Adjusted R-squared:  0.6344
## F-statistic:  1051 on 1 and 604 DF,  p-value: < 2.2e-16
```

```
fit3=lm(svrf[[15]]~Mkt)
summary(fit3)
```

```
##
## Call:
## lm(formula = svrf[[15]] ~ Mkt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.285  -1.732  -0.176   1.567  14.015
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.55721    0.12486   4.463 9.65e-06 ***
## Mkt          1.02837    0.02768  37.152  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.055 on 604 degrees of freedom
## Multiple R-squared:  0.6956, Adjusted R-squared:  0.6951
## F-statistic:  1380 on 1 and 604 DF,  p-value: < 2.2e-16
```

Interpretation and Conclusion:

*For **2nd Portfolio**: We can accept the null hypothesis that the intercept is zero (p-value = 0.249), suggesting no significant evidence at 0.05 alpha.*

For 15th Portfolio: *We reject the null hypothesis that the intercept is zero (p-value = 9.65e-06), indicating significant evidence at 0.05 alpha.*

```
# (f)
fit2=lm(svrf[[2]]~Mkt+SMB+HML)
summary(fit2)
```

```
##
## Call:
## lm(formula = svrf[[2]] ~ Mkt + SMB + HML)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9039 -1.0194 -0.0210  0.8858  9.3661
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.008427   0.068121   0.124    0.902
## Mkt          0.952416   0.016131  59.044  < 2e-16 ***
## SMB          1.292597   0.022582  57.241  < 2e-16 ***
## HML         -0.124334   0.024276  -5.122 4.08e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.639 on 602 degrees of freedom
## Multiple R-squared:  0.9441, Adjusted R-squared:  0.9438
## F-statistic:  3390 on 3 and 602 DF,  p-value: < 2.2e-16
```

```
fit4=lm(svrf[[15]]~Mkt+SMB+HML)
summary(fit4)
```

```
##
## Call:
## lm(formula = svrf[[15]] ~ Mkt + SMB + HML)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -7.6577 -1.1512 -0.1066  0.9625  9.1959
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.11962    0.07434   1.609    0.108
## Mkt          1.05867    0.01760  60.141   <2e-16 ***
## SMB          0.54495    0.02464  22.113   <2e-16 ***
## HML          0.70857    0.02649  26.747   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.789 on 602 degrees of freedom
## Multiple R-squared:  0.896,  Adjusted R-squared:  0.8955
## F-statistic:  1729 on 3 and 602 DF,  p-value: < 2.2e-16
```

Interpretation:

2nd Portfolio: *We can accept the null hypothesis that the intercept is zero (p-value = 0.902), suggesting no significant evidence at 0.05 alpha.*

15th Portfolio: *We can accept the null hypothesis that the intercept is zero (p-value = 0.108), indicating no significant evidence at 0.05 alpha.*

```
# (g)
library(lmtest)
```

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

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

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

```
library(sandwich)
coeftest(fit1, vcov = vcov(fit1))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.197141   0.170935  1.1533   0.2492
## Mkt         1.228318   0.037896 32.4130   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(fit1, vcov = NeweyWest(fit1))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.197141   0.178033  1.1073   0.2686
## Mkt         1.228318   0.042979 28.5795   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coeftest(fit2, vcov = vcov(fit2))
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  0.0084275  0.0681207  0.1237    0.9016
## Mkt          0.9524163  0.0161306 59.0441 < 2.2e-16 ***
## SMB          1.2925966  0.0225817 57.2409 < 2.2e-16 ***
## HML         -0.1243336  0.0242759 -5.1217 4.082e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coeftest(fit2, vcov = NeweyWest(fit2))
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0084275  0.0745373  0.1131  0.91002
## Mkt          0.9524163  0.0244968 38.8793  < 2e-16 ***
## SMB          1.2925966  0.0651197 19.8496  < 2e-16 ***
## HML         -0.1243336  0.0599509 -2.0739  0.03851 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coeftest(fit3, vcov = vcov(fit3))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  0.55721    0.12486  4.4628 9.653e-06 ***
## Mkt          1.02837    0.02768 37.1517 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coeftest(fit3, vcov = NeweyWest(fit3))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
```

```
## (Intercept) 0.557210   0.155085  3.5929 0.0003536 ***
## Mkt         1.028372   0.048553 21.1802 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(fit4, vcov = vcov(fit4))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.119616   0.074340  1.6091   0.1081
## Mkt         1.058674   0.017603 60.1411   <2e-16 ***
## SMB         0.544946   0.024643 22.1134   <2e-16 ***
## HML         0.708573   0.026492 26.7466   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(fit4, vcov = NeweyWest(fit4))
```

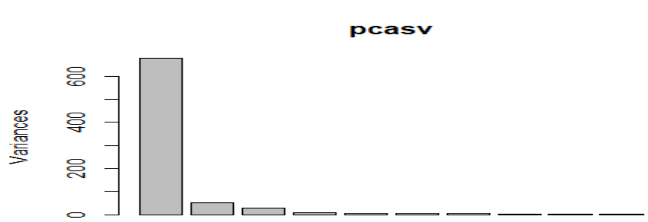
```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.119616   0.085994  1.3910   0.1647
## Mkt         1.058674   0.030630 34.5634   <2e-16 ***
## SMB         0.544946   0.059167  9.2104   <2e-16 ***
## HML         0.708573   0.063483 11.1617   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Comparison and Conclusion:

2nd Portfolio (CAPM) and 2nd Portfolio (Fama-French)*: The intercept remains insignificant under both standard and Newey-West standard errors given their p-values.*

15th Portfolio (CAPM) and 15th Portfolio (Fama-French): *The intercept becomes even less significant with Newey-West standard errors.*

```
# (h)
pcasv=prcomp(svrf)
plot(pcasv)
```

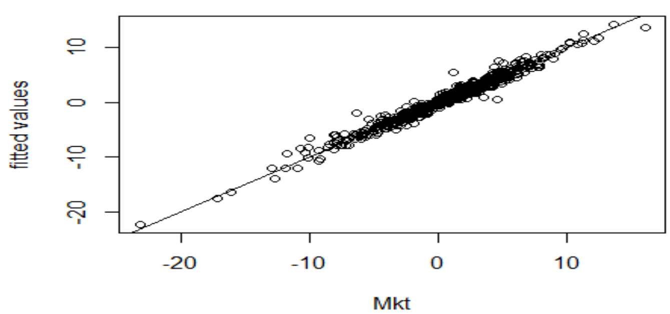


*By examining the elbow plot, there are three statistical factors needed to capture the variance from the **initial steep slope** where it indicates or explains a large portion of the variance as seen from the graph above.*

```
# (i)
fit5=lm(Mkt~pcasv$x[,(1:3)])
summary(fit5)
```

```
##
## Call:
## lm(formula = Mkt ~ pcasv$x[, (1:3)])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3581 -0.3673 -0.0185  0.4113  4.1285
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.498498   0.031558   15.80   <2e-16 ***
## pcasv$x[, (1:3)]PC1 -0.160839   0.001213 -132.56   <2e-16 ***
## pcasv$x[, (1:3)]PC2 -0.096706   0.004372  -22.12   <2e-16 ***
## pcasv$x[, (1:3)]PC3 -0.235693   0.006050  -38.96   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7769 on 602 degrees of freedom
## Multiple R-squared:  0.9702, Adjusted R-squared:   0.97
## F-statistic:  6526 on 3 and 602 DF,  p-value: < 2.2e-16
```

```
plot(Mkt,fit5$fitted.values,ylab="fitted values")
abline(c(0,1))
```

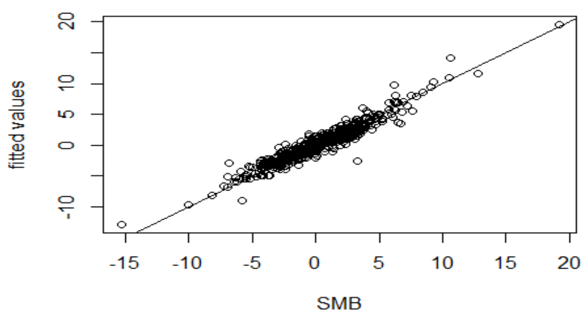



```
fit6=lm(SMB~pcasv$x[,(1:3)])
summary(fit6)
```

```
##
## Call:
## lm(formula = SMB ~ pcasv$x[, (1:3)])
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8730 -0.4823  0.0128  0.4786  5.8184
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.288416   0.036432   7.917 1.18e-14 ***
## pcasv$x[, (1:3)]PC1 -0.069551   0.001401 -49.653  < 2e-16 ***
## pcasv$x[, (1:3)]PC2  0.248744   0.005047  49.286  < 2e-16 ***
## pcasv$x[, (1:3)]PC3  0.282664   0.006984  40.473  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8968 on 602 degrees of freedom
## Multiple R-squared:  0.9156, Adjusted R-squared:  0.9152
## F-statistic:  2178 on 3 and 602 DF,  p-value: < 2.2e-16
```

```
plot(SMB,fit6$fitted.values,ylab="fitted values")
abline(c(0,1))
```

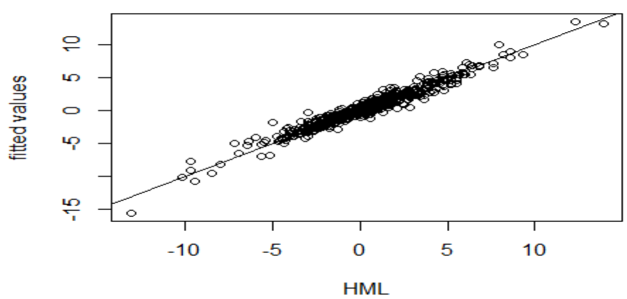



```
fit7=lm(HML~pcasv$x[,(1:3)])
summary(fit7)
```

```
##
## Call:
## lm(formula = HML ~ pcasv$x[, (1:3)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.13357 -0.38904  0.00342  0.39520  2.49947
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.374439   0.027573   13.58   <2e-16 ***
## pcasv$x[, (1:3)]PC1  0.022227   0.001060   20.96   <2e-16 ***
## pcasv$x[, (1:3)]PC2 -0.237740   0.003820  -62.24   <2e-16 ***
## pcasv$x[, (1:3)]PC3  0.408564   0.005286   77.29   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6788 on 602 degrees of freedom
## Multiple R-squared:  0.9447, Adjusted R-squared:  0.9444
## F-statistic:  3429 on 3 and 602 DF,  p-value: < 2.2e-16
```

```r
plot(HML,fit7$fitted.values,ylab="fitted values")
abline(c(0,1))
```



Comment on the Statement:

"The Fama-French factors are simply the first three principal components based on the size-value sorted portfolios."

Interpretation:

- *Fama-French factors are defined based on economic theory and portfolio construction rules.*

- *PCA is a statistical method, whereas the Fama-French factors have economic interpretations related to market risk, size, and value.*

- *Therefore, while there may be overlap in variance explained, the Fama-French factors and the principal components are conceptually different.*

Bonus Questions: Note these questions are optional.

3. Exploratory data analysis II: 15 points

```r
## Problem 3
da <- read.table("d-exuseu-0516.txt",header=T)
head(da)

##   year mon day   euro
## 1 2005   1   3 1.3476
## 2 2005   1   4 1.3295
## 3 2005   1   5 1.3292
## 4 2005   1   6 1.3187
```

```
## 5 2005   1   7 1.3062
## 6 2005   1  10 1.3109
```

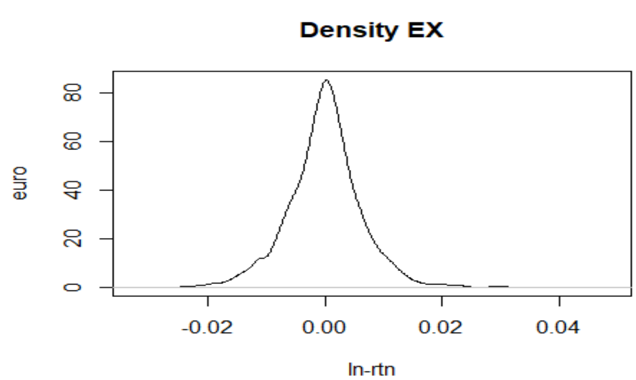
```
# (a)
rtn=diff(log(da$euro)) ## Compute log return
```

All positives log returns indicates that the Euro appreciated relative to the US Dollar from one day to the next.

```
# (b)
basicStats(rtn)
```

```
##                      rtn
## nobs         2816.000000
## NAs             0.000000
## Minimum        -0.030031
## Maximum         0.046208
## 1. Quartile    -0.003446
## 3. Quartile     0.003304
## Mean           -0.000063
## Median          0.000000
## Sum            -0.176816
## SE Mean         0.000120
## LCL Mean       -0.000298
## UCL Mean        0.000172
## Variance        0.000040
## Stdev           0.006363
## Skewness        0.206116
## Kurtosis        2.890412
```

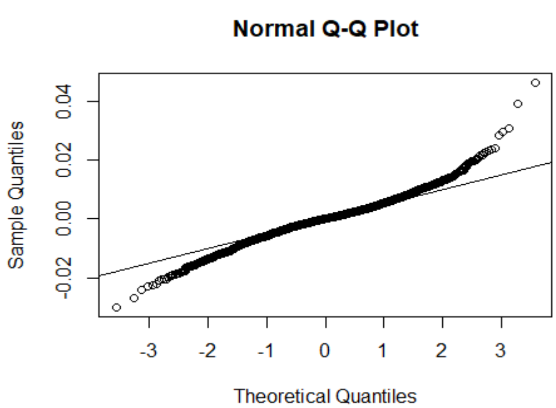
```
# (c)
densityEX=density(rtn)
plot(densityEX,xlab="ln-rtn",ylab='euro',main='Density EX')
```



```r
# (d)
t.test(rtn)

##
##  One Sample t-test
##
## data:  rtn
## t = -0.52367, df = 2815, p-value = 0.6005
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.0002978961  0.0001723167
## sample estimates:
##     mean of x
## -6.278971e-05
```

*Since the p-value (0.6005) is much greater than the typical significance level of 0.05, we **fail to reject the null hypothesis**. This means that there is insufficient evidence to conclude that the mean of the daily log returns is different from zero.*

```r
# (e)
normalTest(rtn)

##
## Title:
##  Shapiro - Wilk Normality Test
##
## Test Results:
##   STATISTIC:
##     W: 0.9732
##   P VALUE:
##     < 2.2e-16

qqnorm(rtn) # Quantile-quantile plot
qqline(rtn) # Include a line in the quantile-quantile plot.
```

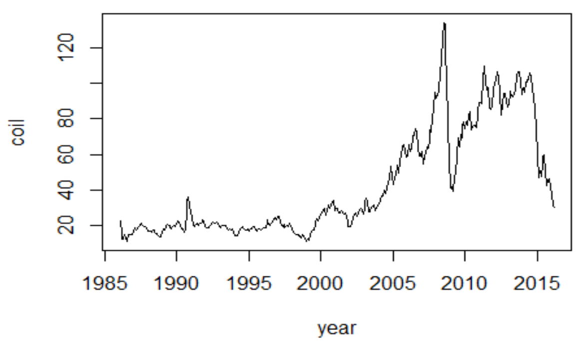



```r
# If empirical values are normally distributed
# then the quantiles should all be on the line.
```

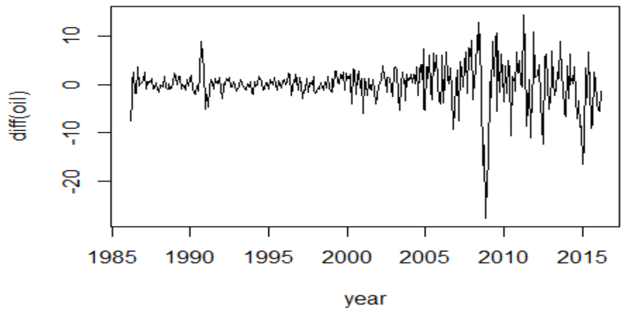
***p-value is extremely small (< 2.2e-16)**, which means we **reject the null hypothesis** and conclude that the log returns are **not normally distributed**. Also, from the QQ plot generated the data points do not closely follow the reference line provides additional evidence that the log returns are **not normally distributed**.*

4. Time-series model: 20 points

```
### Problem 4
da=read.table("m-COILWTICO.txt",header=T)
head(da)

##         DATE VALUE
## 1 1986-01-01 22.93
## 2 1986-02-01 15.46
## 3 1986-03-01 12.61
## 4 1986-04-01 12.84
## 5 1986-05-01 15.38
## 6 1986-06-01 13.43

# (a)
oil=da$VALUE
doil=diff(oil)
dim(da)

## [1] 362   2

# (b)
tdx <- c(1:362)/12+1986
plot(tdx,oil,xlab='year',ylab='coil',type='l')
```




```
plot(tdx[-1],doil,xlab='year',ylab='diff(oil)',type='l')
```



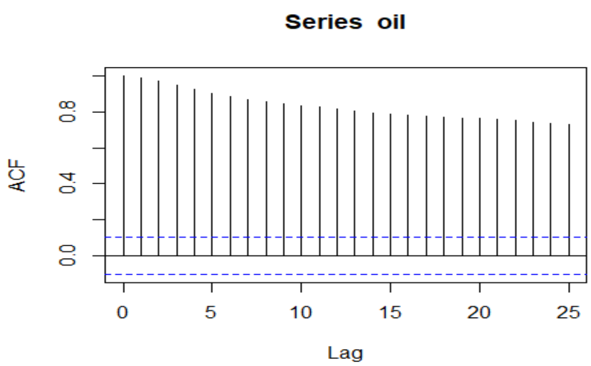
Original series (oil) *is* ***non-stationary*** *because of upwards and downwards trend while* ***Differenced series (diff(oil)) is weakly stationary due to no obvious trend or seasonality.***

```
# (c)
# If you have not installed the package fUnitRoots please use the following
command:
#install.packages("fUnitRoots")
require(fUnitRoots)

## Loading required package: fUnitRoots

adfTest(oil,lags=11,type="c")

##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 11
##   STATISTIC:
##     Dickey-Fuller: -1.6257
##   P VALUE:
##     0.4521
##
## Description:
##  Tue Nov 19 11:37:58 2024 by user: Administrator

adfTest(doil,lags=11,type="c")

## Warning in adfTest(doil, lags = 11, type = "c"): p-value smaller than
printed
## p-value

##
## Title:
##  Augmented Dickey-Fuller Test
```

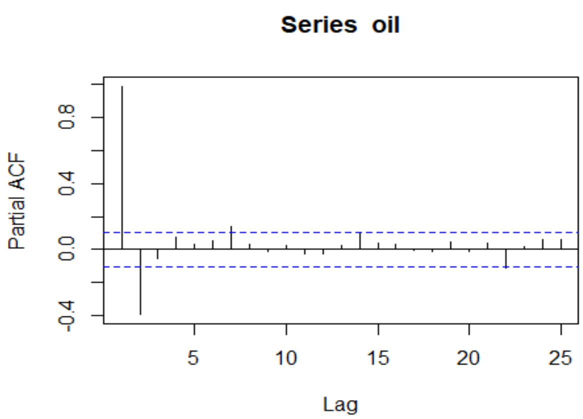
```
##
## Test Results:
##   PARAMETER:
##     Lag Order: 11
##   STATISTIC:
##     Dickey-Fuller: -5.7315
##   P VALUE:
##     0.01
##
## Description:
##  Tue Nov 19 11:37:58 2024 by user: Administrator
```

*Since **p-value** of **0.4521** which is greater than the common significance level of **0.05**, we **fail** to **reject the null hypothesis**. Thus, indicates oil prices is likely non-stationary.*

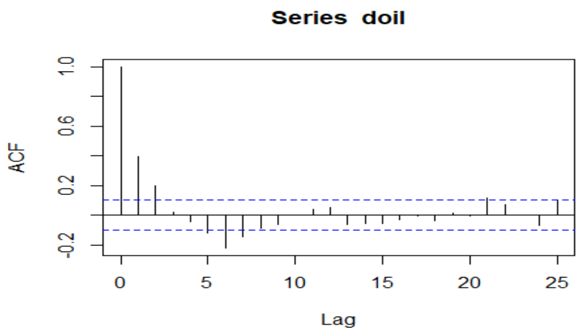
```r
# (d)
acf(oil)
```



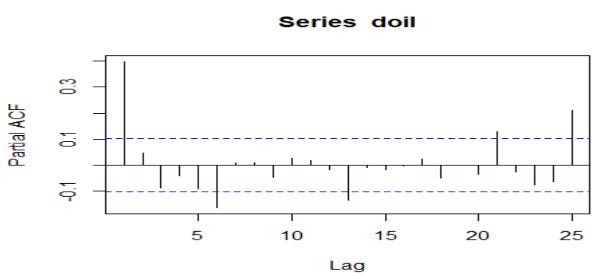

```r
pacf(oil)
```




```r
acf(doil)
```




```r
pacf(doil)
```



ACF of the original series shows slow decay, it indicates non-stationarity. PACF plot also show significant lags, which can help identify the order of differencing needed.

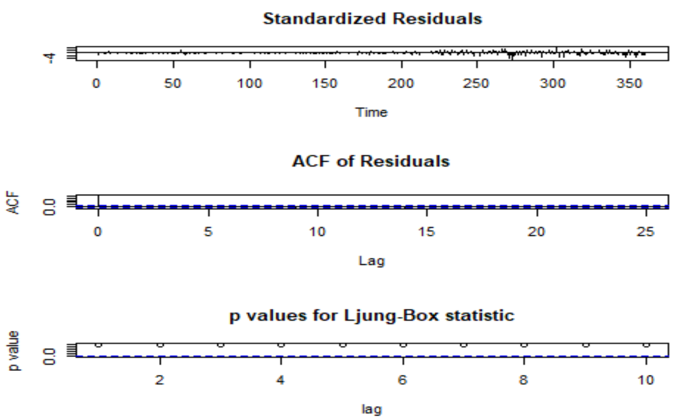
```r
# (e)
Box.test(doil,lag=12,type="Ljung")

##
##  Box-Ljung test
##
## data:  doil
## X-squared = 110.11, df = 12, p-value < 2.2e-16
```

Since p-value is less than 0.05 common significance value, we reject null hypothesis.

```r
# (f)
library("forecast")

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

ARoil=arima(doil,c(6,0,0))
ARoil

##
## Call:
## arima(x = doil, order = c(6, 0, 0))
##
```

```
## Coefficients:
##          ar1     ar2      ar3     ar4      ar5      ar6  intercept
##        0.362  0.0779  -0.0761  0.0059  -0.0291  -0.1626     0.0260
## s.e.   0.052  0.0554   0.0556  0.0555   0.0553   0.0520     0.2483
##
## sigma^2 estimated as 14.95:  log likelihood = -1000.7,  aic = 2017.4
```

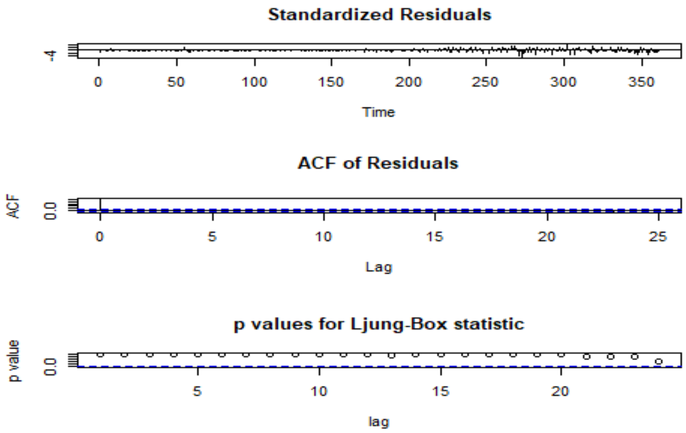
```
tsdiag(ARoil)
```




```
# You can also use the auto.arima command to fit an AR model
# ARoil2=auto.arima(doil,max.p = 20, max.q = 0, d = 0)
# ARoil2
# tsdiag(ARoil2)
# (g)
#You can use the auto.arima command to fit an ARMA model
# ARIMAoil=auto.arima(doil)
# ARIMAoil
# tsdiag(ARIMAoil)
ARIMAoil2=arima(doil,order=c(1,0,6))
ARIMAoil2
```

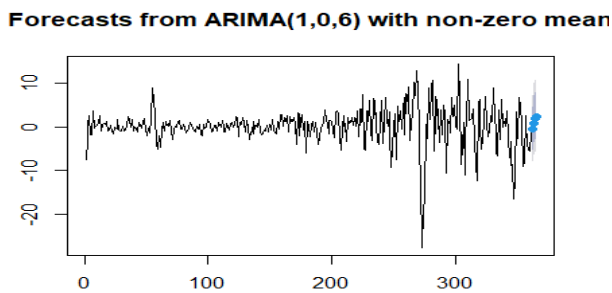
```
##
## Call:
## arima(x = doil, order = c(1, 0, 6))
##
## Coefficients:
##          ar1      ma1      ma2      ma3      ma4      ma5      ma6
intercept
##       0.6892  -0.3330  -0.0384  -0.1289  -0.0149  -0.0312  -0.1717
0.0599
## s.e.  0.1240   0.1274   0.0672   0.0585   0.0543   0.0547   0.0551
0.1897
##
## sigma^2 estimated as 14.91:  log likelihood = -1000.14,  aic = 2018.28
```

```
tsdiag(ARIMAoil2,gof=24)
```

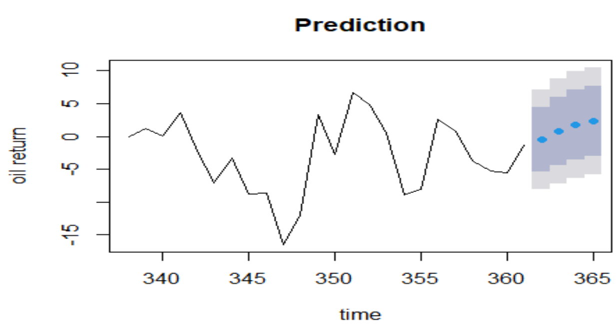



```
# (h)
Oilpredict=predict(ARIMAoil2,4)
Oilpredict

## $pred
## Time Series:
## Start = 362
## End = 365
## Frequency = 1
## [1] -0.4453396  0.8000120  1.8028413  2.3216605
##
## $se
## Time Series:
## Start = 362
## End = 365
## Frequency = 1
## [1] 3.860809 4.098314 4.175542 4.175879

lcl=Oilpredict$pred-1.96*Oilpredict$se
ucl=Oilpredict$pred+1.96*Oilpredict$se
cf=cbind(lcl,ucl)
cf

## Time Series:
## Start = 362
## End = 365
## Frequency = 1
##           lcl       ucl
## 362 -8.012524  7.121845
## 363 -7.232684  8.832708
## 364 -6.381221  9.986903
## 365 -5.863062 10.506383

Oilforecast=forecast(ARIMAoil2,4)
Oilforecast
```

```
##     Point Forecast       Lo 80     Hi 80      Lo 95      Hi 95
## 362     -0.4453396 -5.393165 4.502486 -8.012385  7.121706
## 363      0.8000120 -4.452189 6.052213 -7.232536  8.832560
## 364      1.8028413 -3.548331 7.154013 -6.381070  9.986753
## 365      2.3216605 -3.029943 7.673264 -5.862911 10.506232

plot(Oilforecast)
```




```
# The next plot includes only the last 24 observations and labels
plot(Oilforecast,include=24,xlab="time",ylab="oil return",main="Prediction")
```



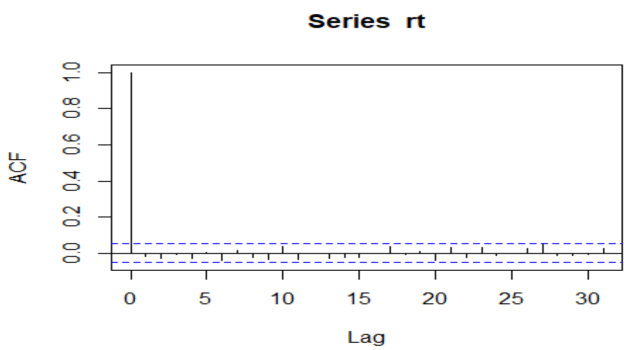
5. GARCH: 20 points

```
#Problem 5
require(forecast)
da=read.table("d-amzn3dx0914.txt",header=T)
head(da)

##   PERMNO     date      amzn    vwretd    ewretd    sprtrn
## 1  84788 20090102  0.060062  0.030501  0.038274  0.031608
## 2  84788 20090105 -0.005519 -0.000579  0.016764 -0.004668
## 3  84788 20090106  0.061043  0.011298  0.033647  0.007817
## 4  84788 20090107 -0.020223 -0.030489 -0.022271 -0.030010
## 5  84788 20090108  0.017082  0.006284  0.011896  0.003397
## 6  84788 20090109 -0.028866 -0.022409 -0.018748 -0.021303
```

```r
rt=log(da$amzn+1)*100
# (a)
t.test(rt)
```

```
##
##  One Sample t-test
##
## data:  rt
## t = 2.0296, df = 1509, p-value = 0.04257
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.003999691 0.234462533
## sample estimates:
## mean of x
## 0.1192311
```

```r
# (b)
acf(rt)
```




```r
Box.test(rt,lag=10,type='Ljung')
```

```
##
##  Box-Ljung test
##
## data:  rt
## X-squared = 10.974, df = 10, p-value = 0.3595
```

```r
# (c)
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following objects are masked from 'package:fBasics':
##
##     qgh, qnig
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
garch.norm = ugarchspec(mean.model=list(armaOrder=c(0,0)),
variance.model=list(garchOrder=c(1,1)))
amazonGarch = ugarchfit(data=rt, spec=garch.norm)
show(amazonGarch)
```

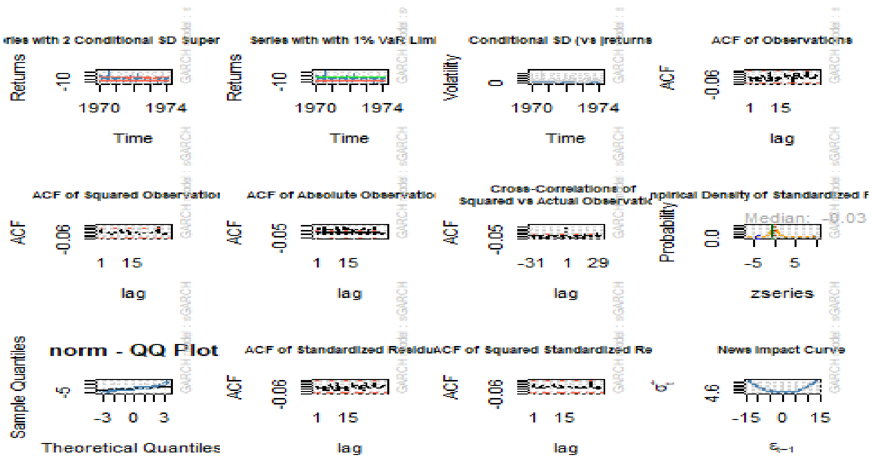
```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.109363    0.056675    1.9297 0.053650
## omega   0.023032    0.005897    3.9059 0.000094
## alpha1  0.007252    0.001621    4.4739 0.000008
## beta1   0.987686    0.000758 1302.8981 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.109363    0.052929    2.0662 0.038808
## omega   0.023032    0.014680    1.5689 0.116678
## alpha1  0.007252    0.004172    1.7382 0.082168
## beta1   0.987686    0.000882 1119.9290 0.000000
##
## LogLikelihood : -3365.187
##
## Information Criteria
## ------------------------------------
##
## Akaike       4.4625
## Bayes        4.4766
## Shibata      4.4625
## Hannan-Quinn 4.4677
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      0.913  0.3393
## Lag[2*(p+q)+(p+q)-1][2]     1.184  0.4424
```

```
## Lag[4*(p+q)+(p+q)-1][5]     2.094  0.5964
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.240  0.2654
## Lag[2*(p+q)+(p+q)-1][5]     1.386  0.7679
## Lag[4*(p+q)+(p+q)-1][9]     1.610  0.9461
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.05111 0.500 2.000  0.8211
## ARCH Lag[5]   0.25510 1.440 1.667  0.9520
## ARCH Lag[7]   0.40838 2.315 1.543  0.9859
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.6148
## Individual Statistics:
## mu     0.23836
## omega  0.10117
## alpha1 0.12173
## beta1  0.09862
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.07 1.24 1.6
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.7049 0.4810
## Negative Sign Bias  0.1673 0.8671
## Positive Sign Bias  0.9319 0.3515
## Joint Effect        1.1292 0.7700
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     101.3    3.098e-13
## 2    30     114.3    4.486e-12
## 3    40     124.0    8.834e-11
## 4    50     138.8    1.605e-10
##
```

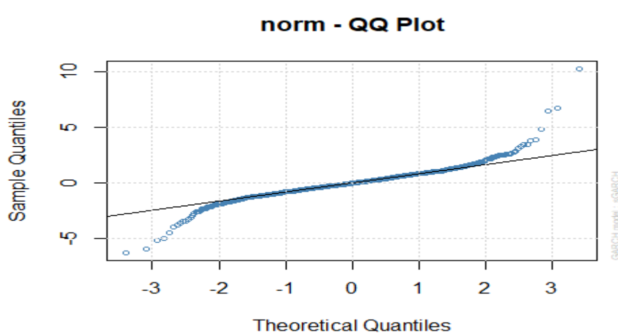
```
##
## Elapsed time : 0.1822562
```

```r
plot(amazonGarch, which="all")
```

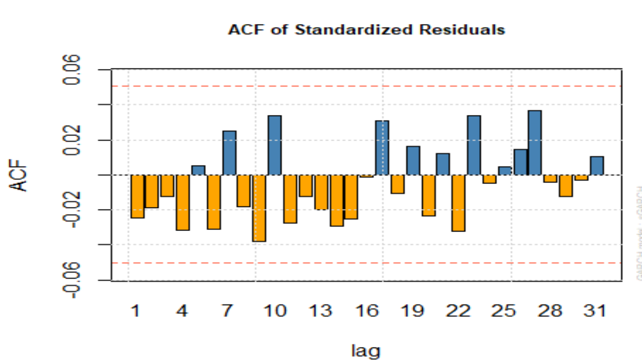
```
##
## please wait...calculating quantiles...
```



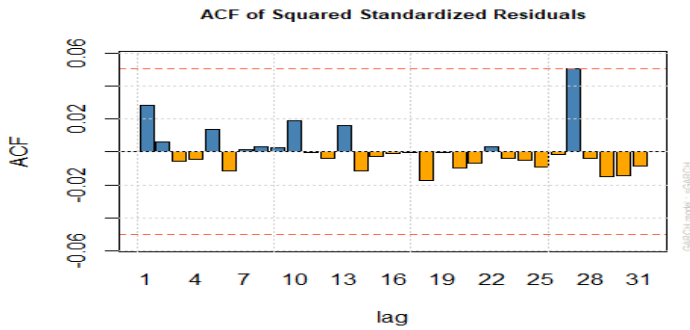

```r
plot(amazonGarch, which=9)
```




```r
plot(amazonGarch, which=10)
```




```r
plot(amazonGarch, which=11)
```




```r
# (d)
arma.garch.t = ugarchspec(mean.model=list(armaOrder=c(0,0)),
variance.model=list(garchOrder=c(1,1)),
distribution.model = "std")
amazonGarch.t = ugarchfit(data=rt, spec=arma.garch.t)
show(amazonGarch.t)
```

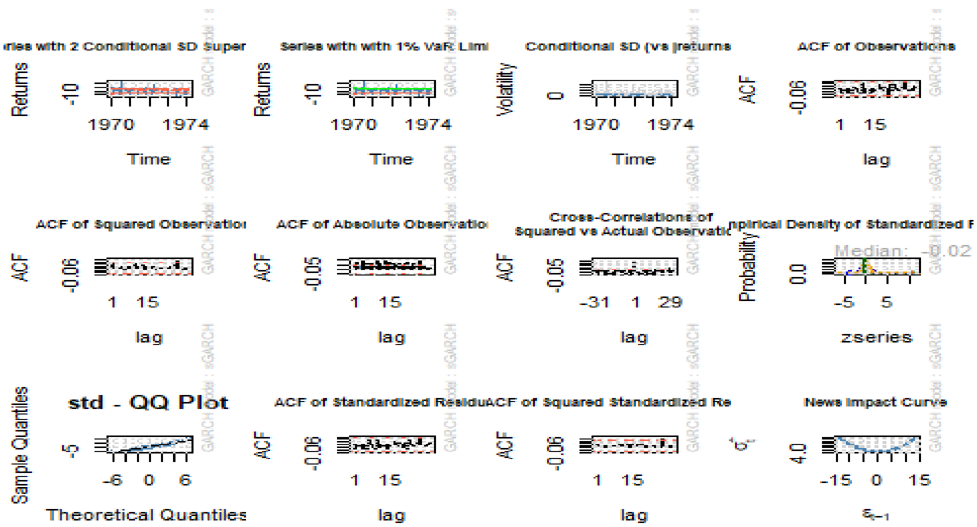
```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model   : sGARCH(1,1)
## Mean Model    : ARFIMA(0,0,0)
## Distribution  : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.087060    0.046113   1.8880 0.059032
## omega   0.060535    0.017407   3.4776 0.000506
## alpha1  0.017064    0.003726   4.5796 0.000005
## beta1   0.968300    0.004272 226.6839 0.000000
## shape   4.311221    0.428313  10.0656 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.087060    0.041887   2.0785 0.037665
## omega   0.060535    0.015855   3.8181 0.000134
## alpha1  0.017064    0.003701   4.6111 0.000004
## beta1   0.968300    0.001813 534.1900 0.000000
## shape   4.311221    0.452458   9.5284 0.000000
##
## LogLikelihood : -3195.838
##
## Information Criteria
## ------------------------------------
```

```
##
## Akaike       4.2395
## Bayes        4.2571
## Shibata      4.2395
## Hannan-Quinn 4.2461
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.9535  0.3288
## Lag[2*(p+q)+(p+q)-1][2]    1.1516  0.4516
## Lag[4*(p+q)+(p+q)-1][5]    2.0691  0.6022
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.2485  0.6182
## Lag[2*(p+q)+(p+q)-1][5]    0.4458  0.9657
## Lag[4*(p+q)+(p+q)-1][9]    0.6631  0.9962
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.2343 0.500 2.000  0.6284
## ARCH Lag[5]    0.3352 1.440 1.667  0.9308
## ARCH Lag[7]    0.5063 2.315 1.543  0.9778
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.4469
## Individual Statistics:
## mu     0.01125
## omega  0.51229
## alpha1 0.92420
## beta1  0.62430
## shape  0.73278
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.6780 0.4979
## Negative Sign Bias  0.3314 0.7404
## Positive Sign Bias  0.4491 0.6534
```

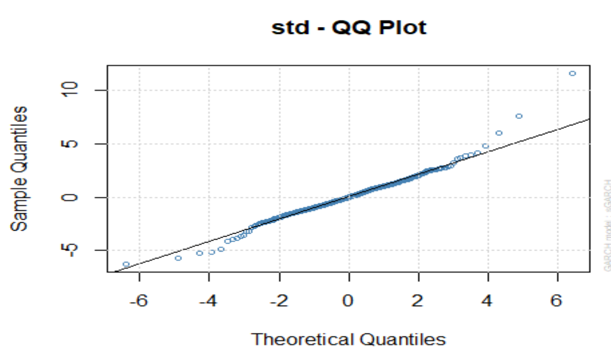
```
## Joint Effect         0.4823 0.9228
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     19.38       0.4329
## 2    30     22.82       0.7847
## 3    40     31.03       0.8146
## 4    50     36.29       0.9109
##
##
## Elapsed time : 0.2266591
```

```r
plot(amazonGarch.t, which="all")
```

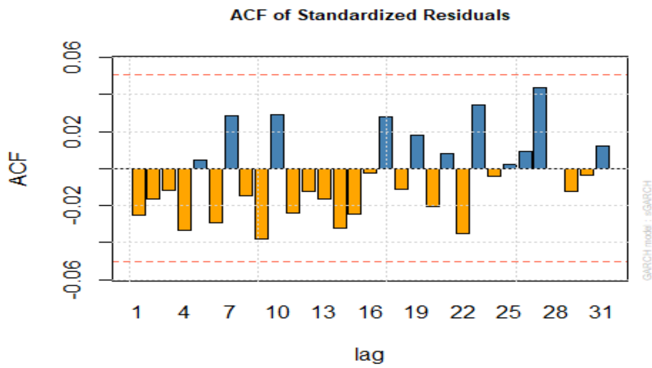
```
##
## please wait...calculating quantiles...
```



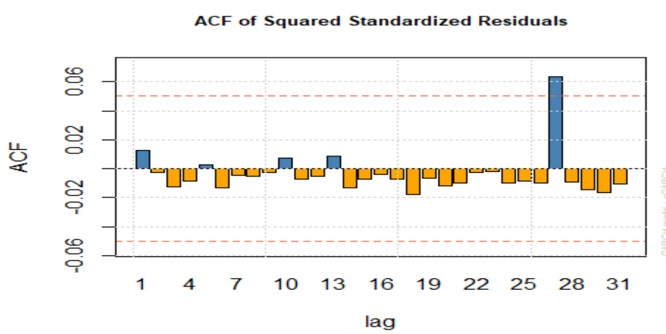
```r
plot(amazonGarch.t, which=9)
```



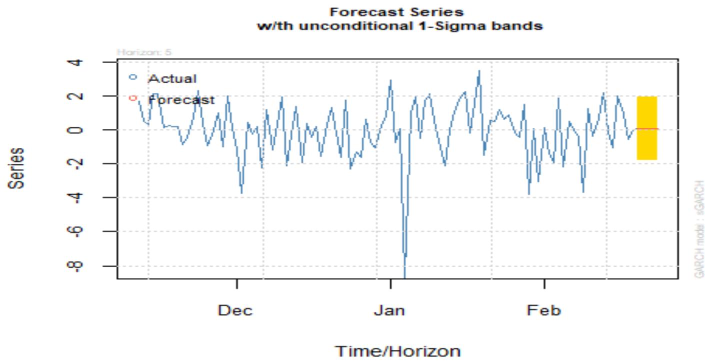

```r
plot(amazonGarch.t, which=10)
```



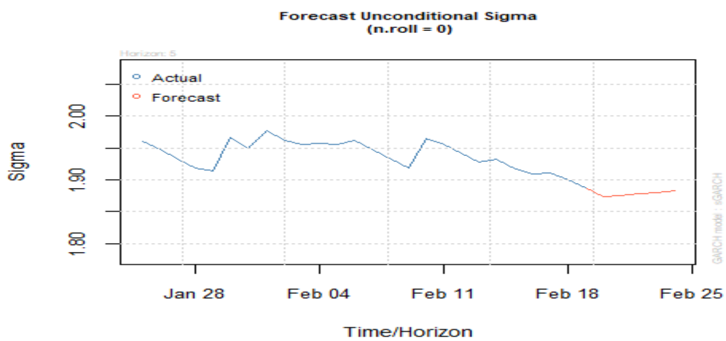

```r
plot(amazonGarch.t, which=11)
```




```r
# (f)
amazonforecast=ugarchforecast(amazonGarch.t, data = rt, n.ahead = 5)
show(amazonforecast)

##
## *------------------------------------*
## *       GARCH Model Forecast         *
## *------------------------------------*
## Model: sGARCH
## Horizon: 5
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=1974-02-19]:
##      Series Sigma
## T+1 0.08706 1.873
## T+2 0.08706 1.876
## T+3 0.08706 1.878
## T+4 0.08706 1.881
## T+5 0.08706 1.883

plot(amazonforecast,which=1)
```

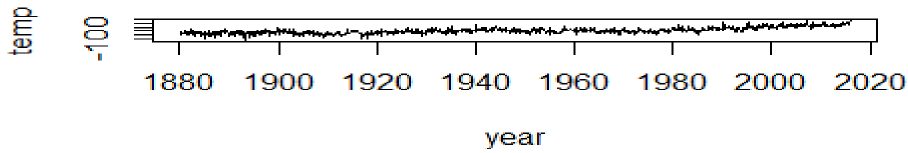


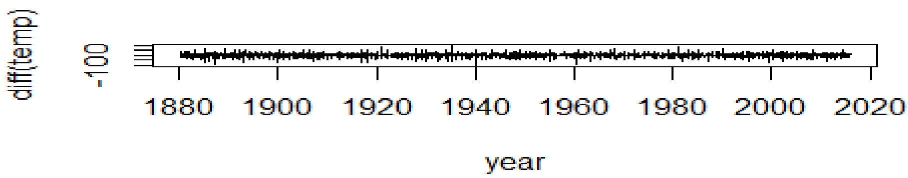

```
plot(amazonforecast,which=3)
```



6.Time-series model II: 30 bonus points

```
#### Problem 6
require(forecast)
da <- read.table("m-globaltemp.txt",header=T)
dd <- da[,2:13]
xt <- c(t(dd))
zt <- diff(xt)
length(xt)

## [1] 1632

tdx <- c(1:1632)/12+1880
# (a)
par(mfcol=c(2,1))
plot(tdx,xt,xlab='year',ylab='temp',type='l')
plot(tdx[-1],zt,xlab='year',ylab='diff(temp)',type='l')
```






```r
par(mfcol=c(1,1))
# (b)
require(fUnitRoots)
adfTest(xt,lags=11,type="c")
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 11
##   STATISTIC:
##     Dickey-Fuller: -1.7197
##   P VALUE:
##     0.4179
##
## Description:
##  Tue Nov 19 11:38:08 2024 by user: Administrator
```

```r
adfTest(zt,lags=11,type="c")
```

```
## Warning in adfTest(zt, lags = 11, type = "c"): p-value smaller than
printed
## p-value
```

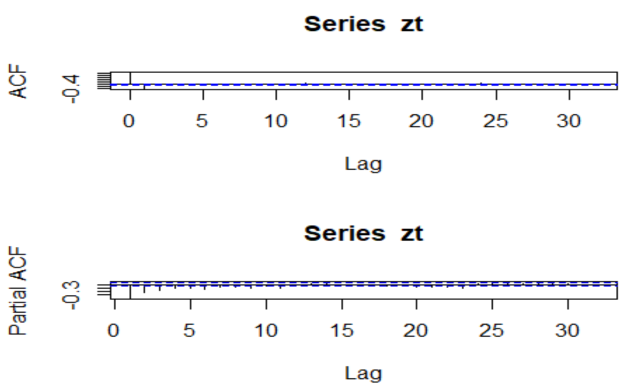
```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 11
##   STATISTIC:
```

```
##     Dickey-Fuller: -17.6881
##   P VALUE:
##     0.01
##
## Description:
##  Tue Nov 19 11:38:08 2024 by user: Administrator
```

```r
# (c)
t.test(zt)
```

```
##
##  One Sample t-test
##
## data:  zt
## t = 0.22712, df = 1630, p-value = 0.8204
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.9270074  1.1698033
## sample estimates:
## mean of x
## 0.1213979
```

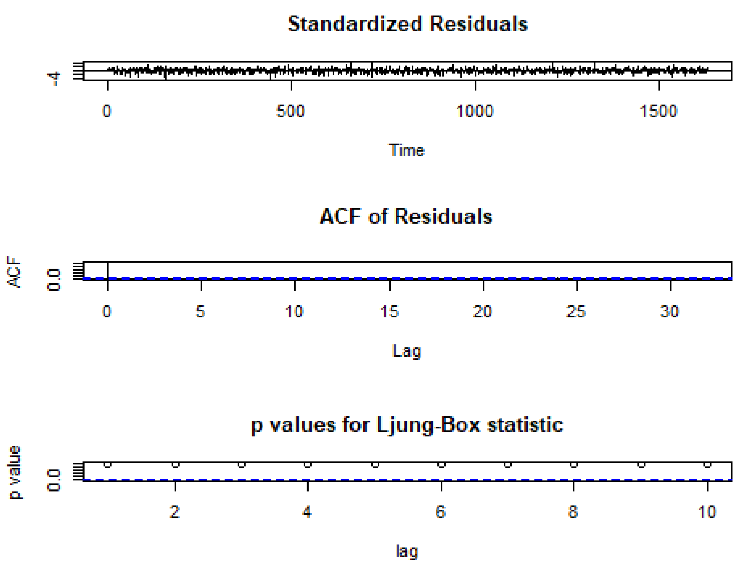
```r
# (d)
par(mfcol=c(2,1))
acf(zt)
pacf(zt)
```




```r
par(mfcol=c(1,1))
# (e)
Box.test(zt,lag=12,type='Ljung')
```

```
##
##  Box-Ljung test
##
## data:  zt
## X-squared = 253.49, df = 12, p-value < 2.2e-16
```

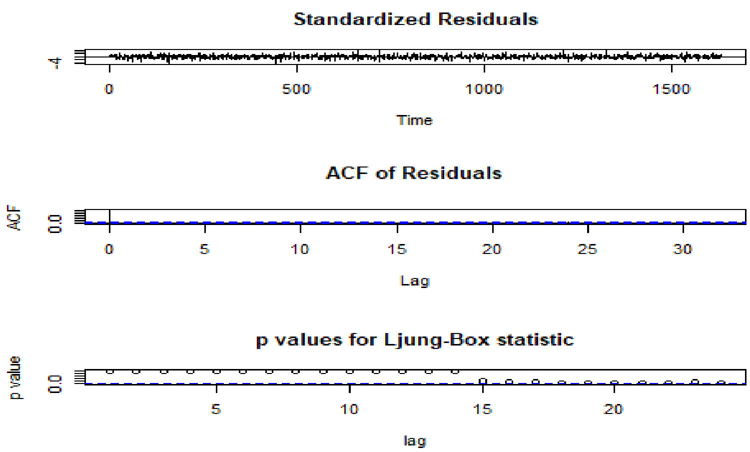
```
# (f)
ARtemp=auto.arima(zt,max.p = 20, max.q = 0, d = 0)
ARtemp

## Series: zt
## ARIMA(11,0,0) with zero mean
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7
ar8
##      -0.5723  -0.4264  -0.3559  -0.3062  -0.2923  -0.2732  -0.2137  -
0.2041
## s.e.   0.0247   0.0283   0.0299   0.0308   0.0313   0.0314   0.0313
0.0308
##          ar9     ar10     ar11
##      -0.1894  -0.1132  -0.0929
## s.e.   0.0299   0.0283   0.0247
##
## sigma^2 = 346.1:  log likelihood = -7077.32
## AIC=14178.64   AICc=14178.83   BIC=14243.4

tsdiag(ARtemp)
```




```
ARtemp2=arima(zt,order=c(11,0,0),include.mean=F)
ARtemp2

##
## Call:
## arima(x = zt, order = c(11, 0, 0), include.mean = F)
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ar5      ar6      ar7
ar8
```

```
##        -0.5723  -0.4264  -0.3559  -0.3062  -0.2923  -0.2732  -0.2137  -
0.2041
## s.e.   0.0247   0.0283   0.0299   0.0308   0.0313   0.0314   0.0313
0.0308
##            ar9     ar10     ar11
##        -0.1894  -0.1132  -0.0929
## s.e.   0.0299   0.0283   0.0247
##
## sigma^2 estimated as 343.8:  log likelihood = -7077.32,  aic = 14178.64
```

```
tsdiag(ARtemp2,gof=24)
```




```
# (g)
ARtemp3=arima(xt,order=c(11,1,0))
ARtemp3

##
## Call:
## arima(x = xt, order = c(11, 1, 0))
##
## Coefficients:
##            ar1      ar2      ar3      ar4      ar5      ar6      ar7
ar8
##        -0.5723  -0.4264  -0.3559  -0.3062  -0.2923  -0.2732  -0.2137  -
0.2041
## s.e.   0.0247   0.0283   0.0299   0.0308   0.0313   0.0314   0.0313
0.0308
##            ar9     ar10     ar11
##        -0.1894  -0.1132  -0.0929
## s.e.   0.0299   0.0283   0.0247
##
## sigma^2 estimated as 343.8:  log likelihood = -7077.32,  aic = 14178.64

PredictTemp <- predict(ARtemp3,12)
PredictTemp
```

```
## $pred
## Time Series:
## Start = 1633
## End = 1644
## Frequency = 1
##  [1] 129.1232 123.5386 121.3828 118.4179 115.6651 115.6414 115.1640
115.9398
##  [9] 118.1312 120.5299 121.8315 122.8075
##
## $se
## Time Series:
## Start = 1633
## End = 1644
## Frequency = 1
##  [1] 18.54211 20.16716 21.06854 21.67045 22.14065 22.45831 22.71378
23.04834
##  [9] 23.31226 23.55386 23.96549 24.35400
```

```
# (h)
lcl <- PredictTemp$pred-1.96*PredictTemp$se
ucl <- PredictTemp$pred+1.96*PredictTemp$se
cf <- cbind(lcl,ucl)
cf[1:2,]
```

```
##             lcl      ucl
## [1,] 92.78064 165.4657
## [2,] 84.01097 163.0662
```

```
cf[1:12,]
```

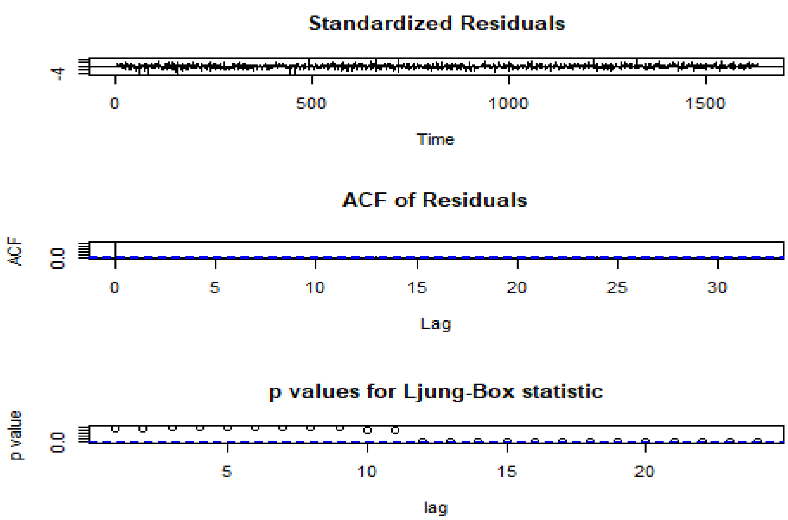
```
##              lcl      ucl
##  [1,] 92.78064 165.4657
##  [2,] 84.01097 163.0662
##  [3,] 80.08851 162.6772
##  [4,] 75.94382 160.8920
##  [5,] 72.26940 159.0607
##  [6,] 71.62310 159.6597
##  [7,] 70.64502 159.6830
##  [8,] 70.76509 161.1146
##  [9,] 72.43918 163.8232
## [10,] 74.36439 166.6955
## [11,] 74.85917 168.8039
## [12,] 75.07368 170.5414
```

```
# (i)
require(forecast)
ARIMAtemp=auto.arima(xt)
ARIMAtemp
```

```
## Series: xt
## ARIMA(1,1,2) with drift
```

```
##
## Coefficients:
##          ar1      ma1     ma2   drift
##       0.7189  -1.3106  0.3329  0.0795
## s.e.  0.0472   0.0593  0.0540  0.0369
##
## sigma^2 = 342.5:  log likelihood = -7072.41
## AIC=14154.82   AICc=14154.86   BIC=14181.8
```

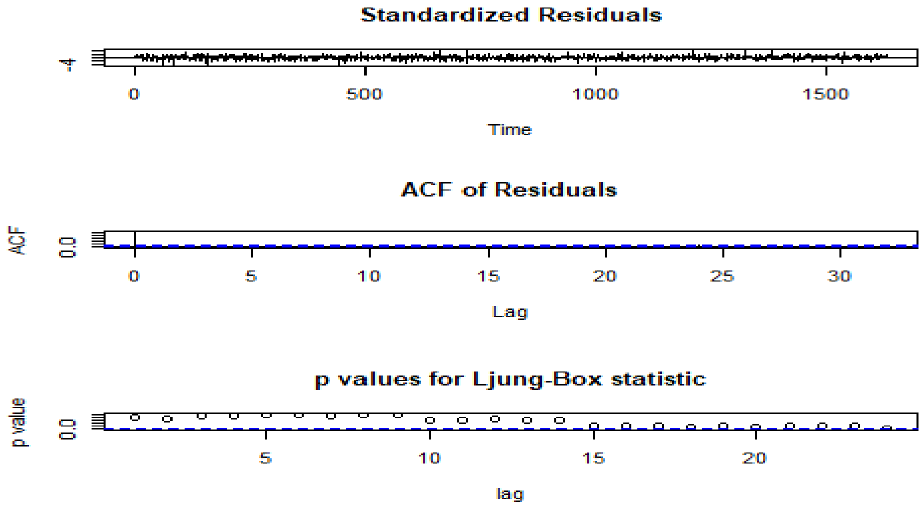
```
tsdiag(ARIMAtemp,gof=24)
```



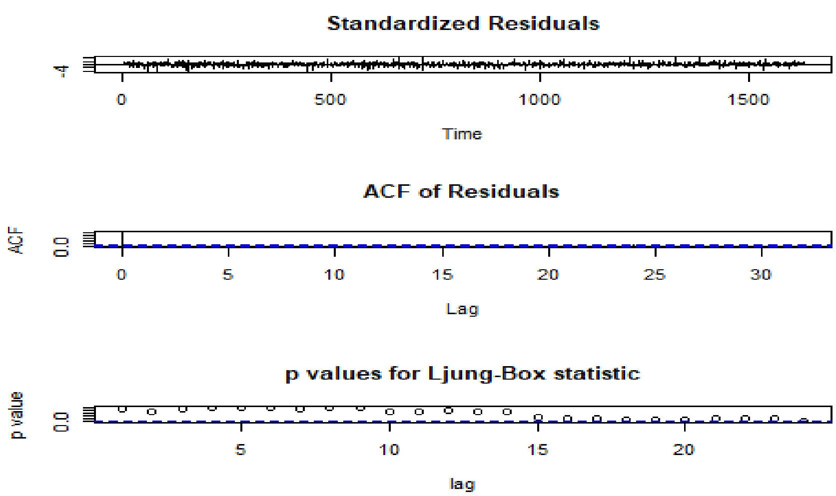

```
ARIMAtemp2=arima(zt,order=c(1,0,2),seasonal=list(order=c(1,0,0),period=12))
ARIMAtemp2
```

```
##
## Call:
## arima(x = zt, order = c(1, 0, 2), seasonal = list(order = c(1, 0, 0),
period = 12))
##
## Coefficients:
##          ar1      ma1     ma2    sar1  intercept
##       0.7423  -1.3429  0.3600  0.0937     0.0793
## s.e.  0.0413   0.0546  0.0511  0.0252     0.0342
##
## sigma^2 estimated as 338.8:  log likelihood = -7065.56,  aic = 14143.12
```

```
tsdiag(ARIMAtemp2,gof=24)
```




```
# (j)
ARIMAtemp3=arima(xt,order=c(1,1,2),seasonal=list(order=c(1,0,0),period=12))
ARIMAtemp3

##
## Call:
## arima(x = xt, order = c(1, 1, 2), seasonal = list(order = c(1, 0, 0),
period = 12))
##
## Coefficients:
##          ar1      ma1     ma2    sar1
##       0.7320  -1.3288  0.3503  0.0921
## s.e.  0.0441   0.0570  0.0526  0.0253
##
## sigma^2 estimated as 339.7:  log likelihood = -7067.82,  aic = 14145.65

tsdiag(ARIMAtemp3,gof=24)
```



```r
PredictTemp=predict(ARIMAtemp3,12)
PredictTemp
```

```
## $pred
## Time Series:
## Start = 1633
## End = 1644
## Frequency = 1
##  [1] 122.06472 115.74604 111.78999 106.32858 103.62970 102.24940  99.42832
##  [8]  99.45501 100.02232 100.41474 101.22510 101.77075
##
## $se
## Time Series:
## Start = 1633
## End = 1644
## Frequency = 1
##  [1] 18.43157 19.87371 20.71308 21.23270 21.57245 21.80668 21.97682
22.10682
##  [9] 22.21097 22.29805 22.37360 22.44122
```

```r
lcl <- PredictTemp$pred-1.96*PredictTemp$se
ucl <- PredictTemp$pred+1.96*PredictTemp$se
cf <- cbind(lcl,ucl)
cf[1:2,]
```

```
##           lcl      ucl
## [1,] 85.93885 158.1906
## [2,] 76.79356 154.6985
```

```r
cf[1:12,]
```

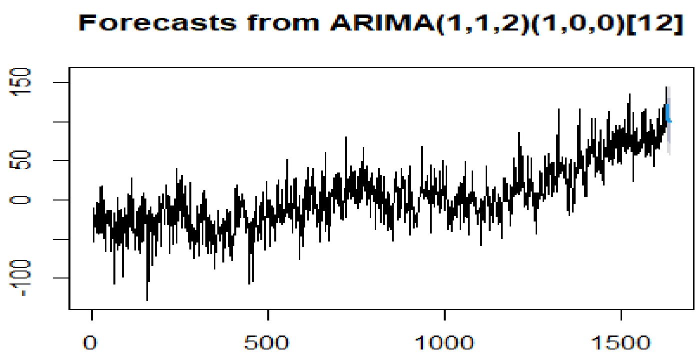
```
##            lcl      ucl
##  [1,] 85.93885 158.1906
##  [2,] 76.79356 154.6985
##  [3,] 71.19235 152.3876
##  [4,] 64.71247 147.9447
##  [5,] 61.34770 145.9117
##  [6,] 59.50831 144.9905
##  [7,] 56.35376 142.5029
##  [8,] 56.12564 142.7844
##  [9,] 56.48882 143.5558
## [10,] 56.71057 144.1189
## [11,] 57.37285 145.0774
## [12,] 57.78596 145.7555
```

```r
Tempforecast=forecast(ARIMAtemp3,12)
Tempforecast
```

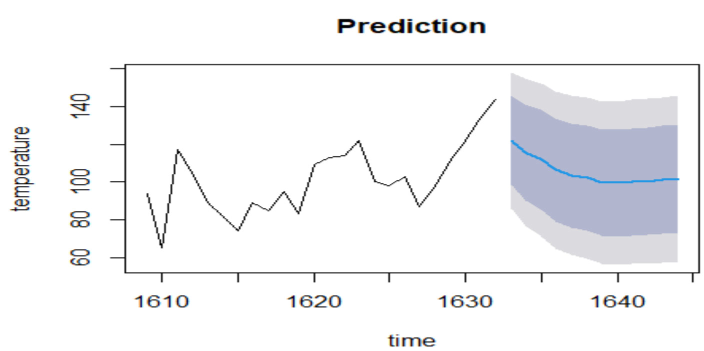
```
##      Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 1633      122.06472 98.44372 145.6857 85.93951 158.1899
## 1634      115.74604 90.27685 141.2152 76.79427 154.6978
```

```
## 1635       111.78999 85.24511 138.3349 71.19309 152.3869
## 1636       106.32858 79.11777 133.5394 64.71324 147.9439
## 1637       103.62970 75.98350 131.2759 61.34848 145.9109
## 1638       102.24940 74.30302 130.1958 59.50910 144.9897
## 1639        99.42832 71.26390 127.5928 56.35455 142.5021
## 1640        99.45501 71.12398 127.7860 56.12643 142.7836
## 1641       100.02232 71.55782 128.4868 56.48962 143.5550
## 1642       100.41474 71.83864 128.9908 56.71137 144.1181
## 1643       101.22510 72.55218 129.8980 57.37365 145.0766
## 1644       101.77075 73.01117 130.5303 57.78677 145.7547
```

```
plot(Tempforecast)
```



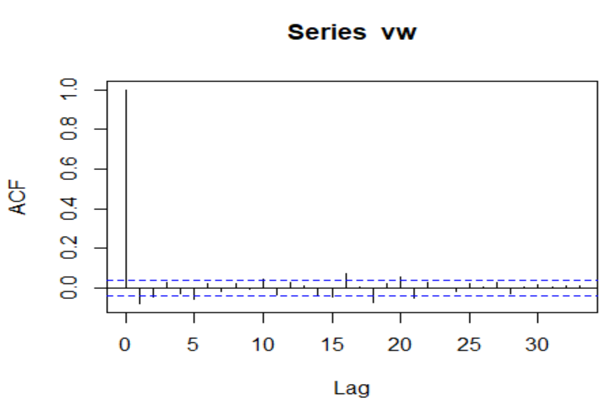

```
plot(Tempforecast,include=24,xlab="time",ylab="temperature",main="Prediction"
)
```



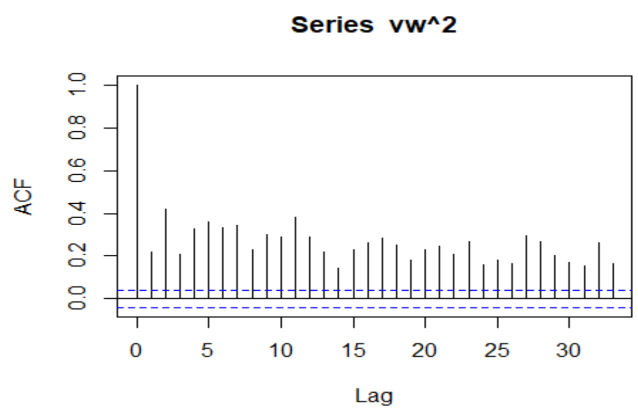
7. GARCH II: 30 bonus points

```
#Problem 7
da=read.table("d-sbux3dx-0715.txt",header=T)
head(da)

##    PERMNO     date      SBUX    vwretd    ewretd    sprtrn
## 1   77702 20070103 -0.004799 -0.001347 -0.000159 -0.001199
```

```
## 2  77702 20070104  0.001135  0.000547  0.000591  0.001228
## 3  77702 20070105 -0.004251 -0.007288 -0.009809 -0.006085
## 4  77702 20070108 -0.003700  0.002567  0.001731  0.002220
## 5  77702 20070109 -0.004284 -0.000001  0.000262 -0.000517
## 6  77702 20070110 -0.003155  0.002096  0.001338  0.001940

rtn=da[,3:6]
attach(rtn)

## The following objects are masked from rtn (pos = 13):
##
##     ewretd, SBUX, sprtrn, vwretd

vw=log(da$vwretd+1)
# (a)
t.test(vw)

##
##  One Sample t-test
##
## data:  vw
## t = 0.78365, df = 2265, p-value = 0.4333
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.0003377266  0.0007873080
## sample estimates:
##    mean of x
## 0.0002247907

Box.test(vw,lag=12,type='Ljung')

##
##  Box-Ljung test
##
## data:  vw
## X-squared = 41.512, df = 12, p-value = 4.023e-05

acf(vw)
```



```
acf(vw^2)
```



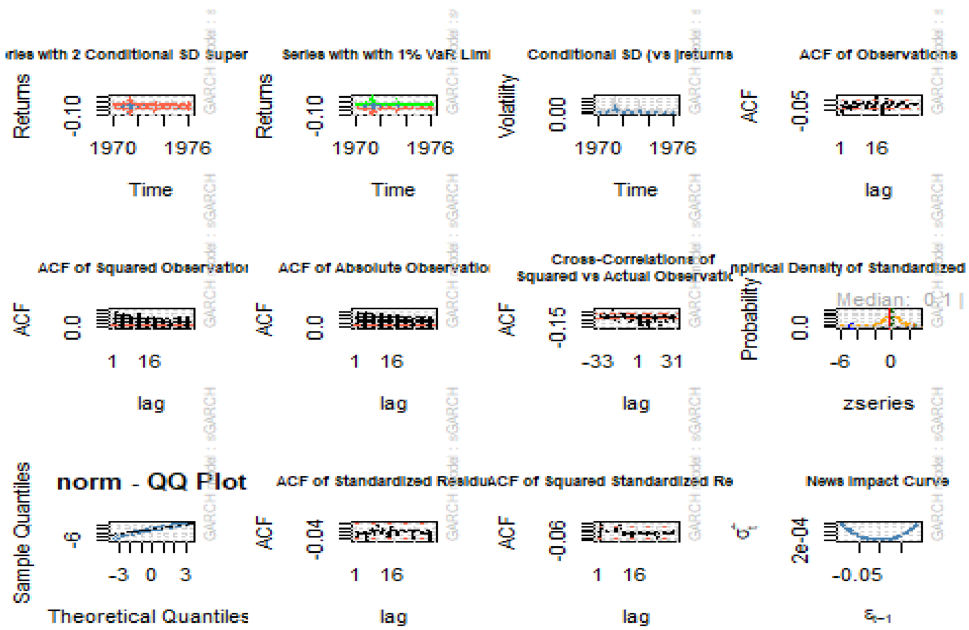

```
garch.norm = ugarchspec(mean.model=list(armaOrder=c(0,0),include.mean =
FALSE),
variance.model=list(garchOrder=c(1,1)))
vwGarch = ugarchfit(data=vw, spec=garch.norm)
show(vwGarch)

##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## omega   0.000003    0.000001   1.9402 0.052357
## alpha1  0.106140    0.013764   7.7113 0.000000
## beta1   0.875661    0.015626  56.0379 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## omega   0.000003    0.000007  0.38952 0.696889
## alpha1  0.106140    0.039123  2.71298 0.006668
## beta1   0.875661    0.057114 15.33175 0.000000
##
## LogLikelihood : 7118.418
##
## Information Criteria
## ------------------------------------
##
```

```
## Akaike       -6.2802
## Bayes        -6.2726
## Shibata      -6.2802
## Hannan-Quinn -6.2774
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      3.482 0.06204
## Lag[2*(p+q)+(p+q)-1][2]     3.483 0.10387
## Lag[4*(p+q)+(p+q)-1][5]     4.826 0.16751
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic  p-value
## Lag[1]                      6.885 0.008693
## Lag[2*(p+q)+(p+q)-1][5]    13.513 0.001110
## Lag[4*(p+q)+(p+q)-1][9]    15.073 0.003435
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]  0.003359 0.500 2.000  0.9538
## ARCH Lag[5]  0.474006 1.440 1.667  0.8913
## ARCH Lag[7]  0.589874 2.315 1.543  0.9695
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  12.9665
## Individual Statistics:
## omega  0.791
## alpha1 1.023
## beta1  1.233
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          0.846 1.01 1.35
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            3.051 2.304e-03 ***
## Negative Sign Bias   1.529 1.264e-01
## Positive Sign Bias   2.063 3.919e-02  **
## Joint Effect        25.186 1.411e-05 ***
##
##
```

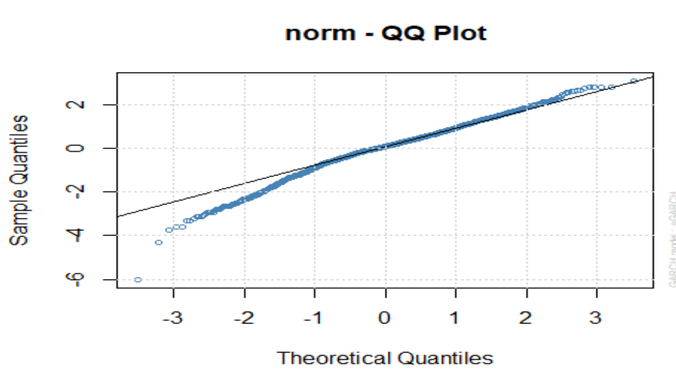
```
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     125.2    1.151e-17
## 2    30     134.8    1.392e-15
## 3    40     160.9    9.583e-17
## 4    50     181.2    4.756e-17
##
##
## Elapsed time : 0.178808
```

```
plot(vwGarch, which="all")
```

```
##
## please wait...calculating quantiles...
```




```
plot(vwGarch, which=9)
```

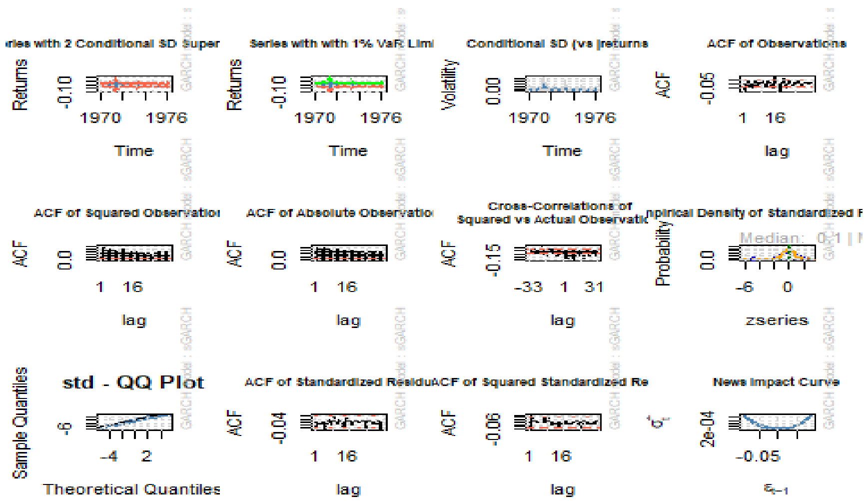


```r
arma.garch.t = ugarchspec(mean.model=list(armaOrder=c(0,0),include.mean =
FALSE),
variance.model=list(garchOrder=c(1,1)),
distribution.model = "std")
vwGarch.t = ugarchfit(data=vw, spec=arma.garch.t)
show(vwGarch.t)
```

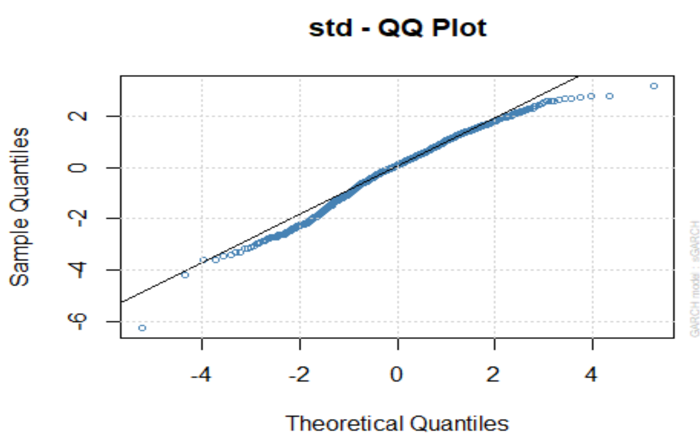
```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## omega   0.000002    0.000002   1.1069 0.268325
## alpha1  0.114004    0.026049   4.3765 0.000012
## beta1   0.876326    0.025452  34.4308 0.000000
## shape   6.980602    1.244478   5.6093 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## omega   0.000002    0.000008  0.25476 0.798908
## alpha1  0.114004    0.096632  1.17978 0.238089
## beta1   0.876326    0.098073  8.93549 0.000000
## shape   6.980602    2.846402  2.45243 0.014189
##
## LogLikelihood : 7151.623
##
## Information Criteria
## ------------------------------------
##
## Akaike       -6.3086
## Bayes        -6.2985
## Shibata      -6.3086
## Hannan-Quinn -6.3049
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      3.390  0.0656
## Lag[2*(p+q)+(p+q)-1][2]     3.393  0.1099
## Lag[4*(p+q)+(p+q)-1][5]     4.779  0.1716
## d.o.f=0
```

```
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic  p-value
## Lag[1]                      6.988 0.008208
## Lag[2*(p+q)+(p+q)-1][5]    11.808 0.003128
## Lag[4*(p+q)+(p+q)-1][9]    12.862 0.011494
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01185 0.500 2.000  0.9133
## ARCH Lag[5]   0.19013 1.440 1.667  0.9678
## ARCH Lag[7]   0.20868 2.315 1.543  0.9967
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  38.3296
## Individual Statistics:
## omega  5.3989
## alpha1 0.6390
## beta1  0.7948
## shape  0.1110
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.07 1.24 1.6
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            3.081 2.087e-03 ***
## Negative Sign Bias   1.845 6.519e-02   *
## Positive Sign Bias   2.230 2.583e-02  **
## Joint Effect        26.475 7.586e-06 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     83.78    4.114e-10
## 2    30    107.12    6.871e-11
## 3    40    120.23    3.280e-10
## 4    50    132.68    1.218e-09
##
##
## Elapsed time : 0.318897
```

```r
plot(vwGarch.t, which="all")

##
## please wait...calculating quantiles...
```



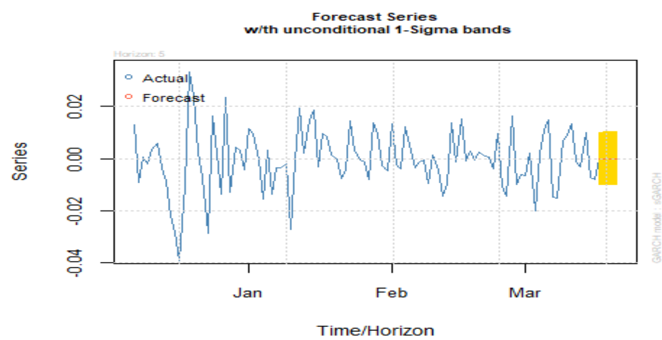

```r
plot(vwGarch.t, which=9)
```



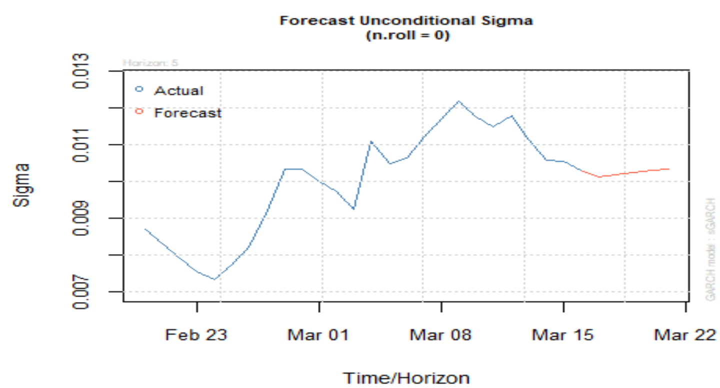

```r
# (b)
vwforecast=ugarchforecast(vwGarch.t, data = vw, n.ahead = 5)
show(vwforecast)

##
## *------------------------------------*
## *       GARCH Model Forecast         *
## *------------------------------------*
## Model: sGARCH
## Horizon: 5
## Roll Steps: 0
## Out of Sample: 0
##
```

```
## 0-roll forecast [T0=1976-03-16]:
##      Series    Sigma
## T+1       0 0.01014
## T+2       0 0.01020
## T+3       0 0.01025
## T+4       0 0.01031
## T+5       0 0.01036
```

```r
plot(vwforecast,which=1)
```




```r
plot(vwforecast,which=3)
```




```r
# (c)
arma.aparch.t = ugarchspec(mean.model=list(armaOrder=c(0,0),include.mean =
FALSE),
variance.model=list(model="apARCH",
garchOrder=c(1,1)),
distribution.model = "std")
vwGarch.a = ugarchfit(data=vw, spec=arma.aparch.t)
show(vwGarch.a)
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
```

```
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : apARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## omega   0.000487    0.000329    1.4784 0.139313
## alpha1  0.100537    0.021684    4.6365 0.000004
## beta1   0.904242    0.024547   36.8374 0.000000
## gamma1  1.000000    0.000980 1020.6524 0.000000
## delta   0.900756    0.123228    7.3097 0.000000
## shape   7.872547    1.297146    6.0691 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## omega   0.000487         NaN      NaN      NaN
## alpha1  0.100537         NaN      NaN      NaN
## beta1   0.904242         NaN      NaN      NaN
## gamma1  1.000000         NaN      NaN      NaN
## delta   0.900756         NaN      NaN      NaN
## shape   7.872547         NaN      NaN      NaN
##
## LogLikelihood : 7215.757
##
## Information Criteria
## ------------------------------------
##
## Akaike       -6.3634
## Bayes        -6.3483
## Shibata      -6.3634
## Hannan-Quinn -6.3579
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      3.802 0.05118
## Lag[2*(p+q)+(p+q)-1][2]     3.862 0.08214
## Lag[4*(p+q)+(p+q)-1][5]     4.653 0.18309
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic   p-value
## Lag[1]                      12.56 0.0003951
## Lag[2*(p+q)+(p+q)-1][5]     12.88 0.0016288
## Lag[4*(p+q)+(p+q)-1][9]     13.90 0.0065675
```

```
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.1770 0.500 2.000  0.6740
## ARCH Lag[5]    0.3049 1.440 1.667  0.9390
## ARCH Lag[7]    1.2287 2.315 1.543  0.8738
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  NaN
## Individual Statistics:
## omega  1.737
## alpha1 1.514
## beta1  2.060
## gamma1   NaN
## delta  1.805
## shape  0.136
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            1.363 0.1730265
## Negative Sign Bias   2.973 0.0029797 ***
## Positive Sign Bias   2.460 0.0139612  **
## Joint Effect        16.909 0.0007378 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     85.37    2.171e-10
## 2    30    104.29    1.990e-10
## 3    40    110.24    1.012e-08
## 4    50    134.53    6.632e-10
##
##
## Elapsed time : 2.761971
```