STATS 461 FINAL PROJECT

Volatility of AMZN

Abstract

An investigation of asset returns of Amazon (AMZN) from 12/15/2017 – 12/14/2020. This report focuses on the volatility of the returns

SHANE WILLIAM MCINTYRE

Introduction:

Everyone is aware of how influential Amazon has been over the past 20+ years. The company opened public stock back in May of 1997 with an IPO price of $18.00 (Amazon.com). Today, the price for 1 share of stock is above $3,000, about a 400% increase over the last 5 years (Yahoo! Finance). What started out as a simple online retailer has evolved into a monster company, controlling the competition when it comes to online shopping. Over the years Amazon has added features like movies and tv shows, groceries, online books on kindle, and even a cloud service known as AWS. The evolution of Amazon with the ability to adapt to the modern world has made them one of the most powerful and influential companies in the world, ranking 9th on the Fortune Global 500 ranking, accumulating a revenue of over $280,000 million in the last fiscal year (Ventura).

Throughout the last year today's generation has experienced the most difficult situation, a global pandemic. The last time we had a pandemic of this size was the Spanish flu back in 1918. As covid-19 spread across the globe we witnessed millions get sick, and as the virus spread, economies declined on a world-wide basis. From how much the pandemic has affected the world, I was curious to see how companies of the statute like Amazon performed. The change of stock price over the last year is some of the most aggressive in history. I will investigate these changes by looking at Amazon's volatility.

Amazon Stock Price over time:

The adjusted stock market price of one share of Amazon's (AMZN) stock was $1179.14 on December 18, 2017. Over the next 3 years one share would grow to a price of $3156.97. That is an increase of 267.74%.

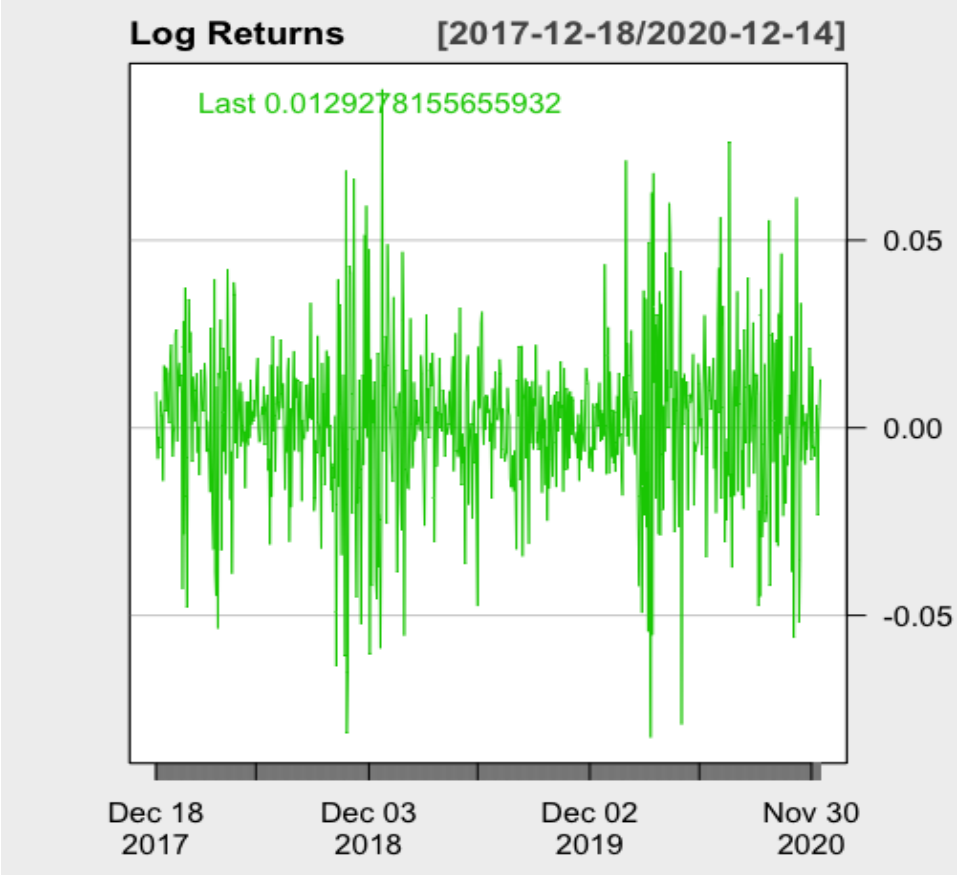
The above graph displays the stock prices from 12/15/2017 through 12/14/2020. From this graph we can see the stock price of AMZN appears to change rapidly and in large amounts. Each tick mark on the x-axis is a month, and we see that around February-March of 2020, when universities and cities started to go into full lockdown, the stock price fell. Soon after the crash, stock prices started to trend upward at a very fast rate. Up until the crash in earlier 2020, the stock price fluctuated between $1,500-$2,000, and today the price is well above $3,000.

Log Returns over time:

Next, after studying the general shape of the stock price, I was interested in the day-to-day simple return. Simple net return is calculated as follows:

$$1 + R_t = \frac{P_t}{P_{t-1}}$$

I converted this to log returns, which produced the following graph:



Here we see the log returns between 2017 and the end of 2020 are relatively diverse, with spikes around December 2018/ January 2019. This happens because of the first large price drop in stock that appears in the Stock Price graph near the end of 2018. However, I am more so interested in the behavior when the US went into lockdown in early 2020, and the price growth since. From the Log Returns graph we see the intensity of change increase in the beginning parts of 2020. This is due to the large price drop off when the economy crashed from covid-19 hitting the US, and the returns kept an intense log return throughout 2020 from the massive growth in the stock price of AMZN. By looking at the stock price over time, in addition to the log returns, we can already see that the stock price of Amazon appears to be very volatile. I will go on to implement a GARCH model to visualize and investigate the volatility.

Basic Statistics of Stock Price and Log Returns:

Before implementing the GARCH model I investigated some basic features of the behavior of the Amazon stock.

Basic Statistics of Amazon stock price over studied time period:

Mean	$2,008.27
Minimum	$1,168.36
Maximum	$3,531.45
Variance	314,157.1
Standard Deviation	560.4972
Skewness	1.208254
Excess Kurtosis	0.3213765

Typically, a variance of 314,157.1 is a terrifying number. However, the variance being so large for the stock price makes sense. The difference between the minimum and maximum stock prices is $2,363.09. With the mean stock price being $2,008.27, and the prices being so widely spread, we expect the variance to be high because the time series of Amazon stocks appears to be very volatile.

Basic Statistics of Log Returns:

Mean	0.001308
Minimum	-0.082535
Maximum	0.090254
Variance	0.000436
Standard Deviation	0.020890
Skewness	0.020890
Excess Kurtosis	2.292832

The statistics for the log returns are much smaller, which is to be expected considering the returns are the log of the difference of the daily prices. We can use the mean, skewness, and kurtosis of the log returns to test for normality assumptions of the log returns.

Mean Test Statistic:

$$\mu_{test} = \frac{\mu}{\sqrt{\frac{\sigma^2}{n}}} = 1.718$$

Skewness Test Statistic:

$$S_{test} = \frac{S}{\sqrt{\frac{6}{n}}} = -1.376$$

Kurtosis Test Statistic:

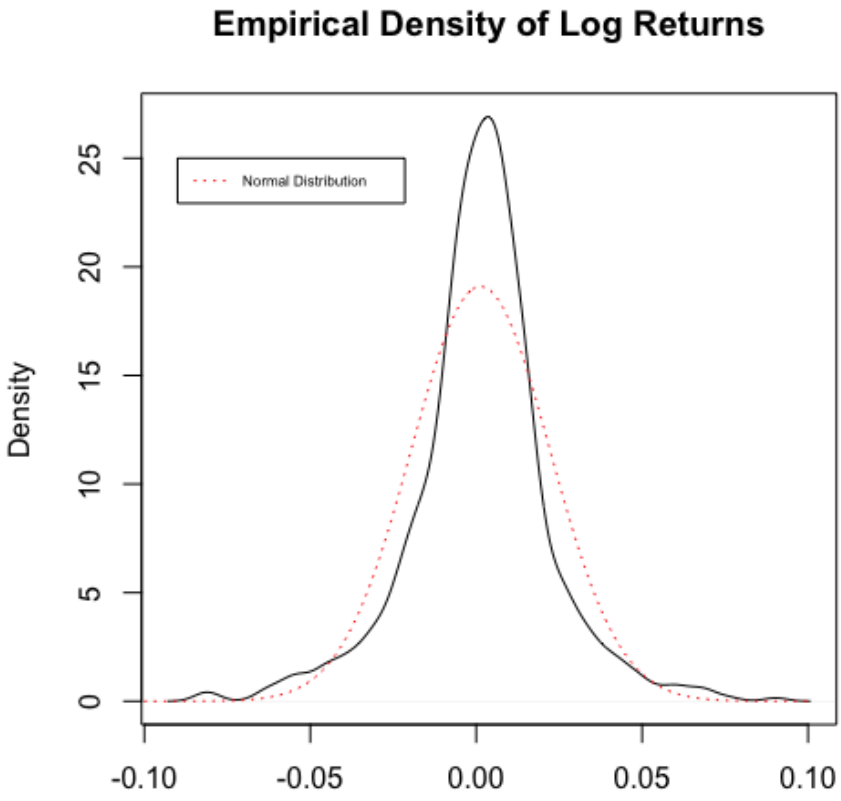
$$K_{test} = \frac{K}{\sqrt{\frac{24}{n}}} = 12.843$$

Jarque – Bera Normaility Test:

I used the JB test to see if the distribution of the log returns is normally distributed. My results were as follows:

JB Test	JB $= S_{test}^2 + K_{test}^2$
X – squared Test Statistic	168.8757
P – value	< 2.2e-16

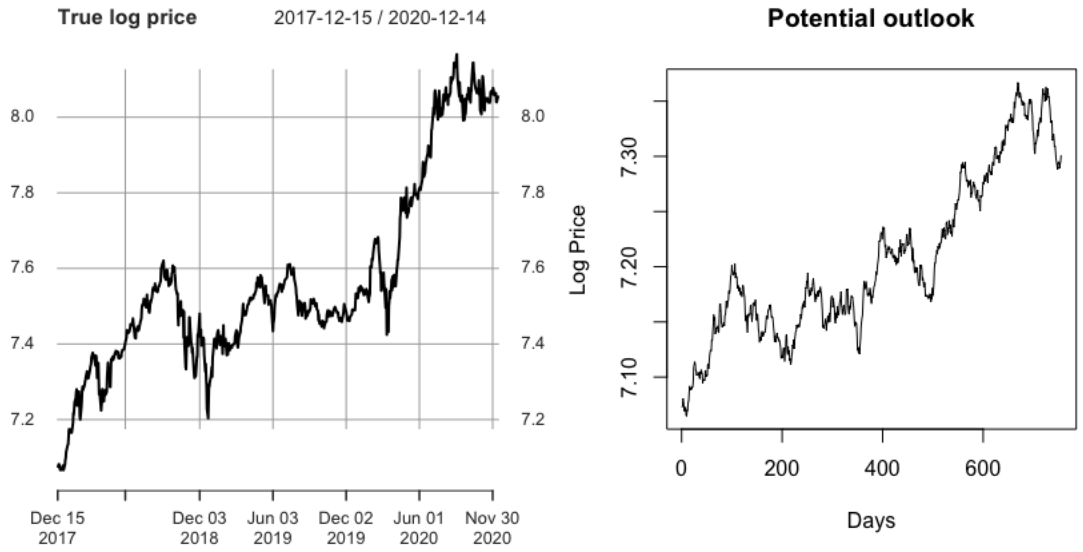
The p – value from the test is very small. Much smaller than the 0.05 value needed to accept the null hypothesis. Hence, the null hypothesis of the distribution of log returns being normally distributed is rejected. It turns out the distribution appears to be normally distributed, but when looking at the empirical density of the log returns vs the empirical density of a normal distributed function with the same mean and standard deviation as the log returns, we see this is not the case.



While the general shape of the distribution of log returns appears to be normally distributed, the peak of its density is why the normality test does not hold, which is due to such a high variance for the stock prices.

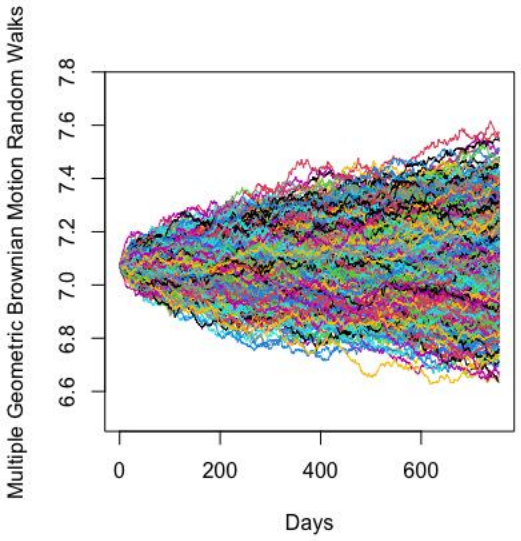
Geometric Brownian Motion:

Before creating my GARCH model I wanted to see how a random walk with geometric Brownian motion would predict the stock price of Amazon over the same period. We know that the distribution of log returns is not normally distributed, and the weight of the Brownian motion is normally distributed according to the change of time. If the true stock prices of Amazon were truly a normal distribution, a collection of random walks could give a potential outlook for how the price would change over time. If stock market prices were normally distributed, and the market grew randomly over time, a potential outlook of the stock price change over time could look as follows:



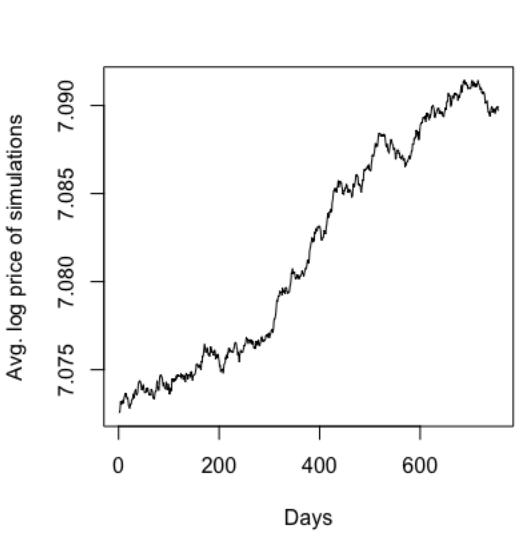
The above graph compares the true log prices of Amazon stock over time (left) versus a random walk (right) with the starting price of the stock being the same, but the price change over time is random and normally distributed. While the shape of a random walk has the potential to be similar to the shape of the true price change, the true log price has values over 8.0 while the random walk doesn't get above 7.4. This visualizes how the change of Amazon stock over time is not random, and not normally distributed.

Simulation of 1000 Random Walks with Geometric Brownian Motion:



The above graph is 1000 simulated random walks with the same starting price as the true starting price of the Amazon stock. For the true stock, the max log price is 8.17, while out of 1000 random simulation with normal distribution the max log price of any simulation is only 7.57. This is a good visualization of why the price of Amazon stock does not grow truly randomly.

A few more plots for visualizing why I believe Amazon stock does not grow randomly to a normal distribution:

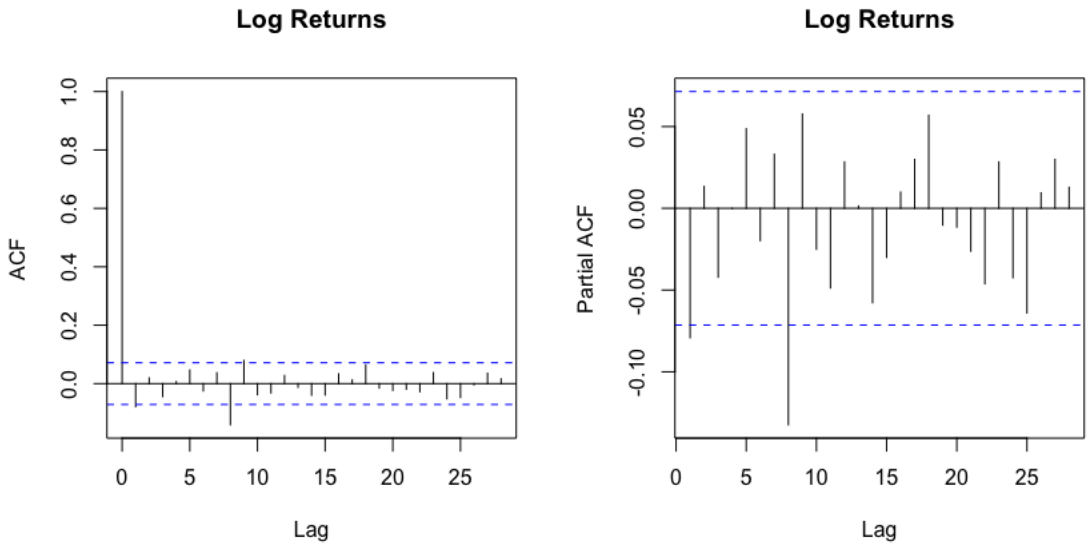


The graph on the left is the average log price of all 1000 simulation. We see that the price values are too low compared to the true values. The graph on the right shows the average difference in price between the true price and the average of estimated prices. This is a visualization of why the variance is so large.

General GARCH Model:

Before constructing my GARCH model I first look at a simple ARMA(0,0) model to confirm my suspicion that a GARCH model is needed to account for the volatility of the returns. I conduct a Box – Pierce test which returns a X – squared value of 175.28, and a p – value < 2.2e-16. The goodness of fit is rejected, telling us that the squared residuals have significant serial correlation. From this I will continue building a GARCH model to account for these correlations.

To begin constructing my model I first look at the autocorrelation and partial autocorrelation functions of the log returns.



From these graphs there does not appear to be a large amount of significantly important correlation, but there are some that could influence the trajectory of the model. From the measures of the autocorrelation and partial autocorrelation as the lag increases, there is no definitive pattern of growth or decay of the correlation.

Since the change of stock price and log returns appears to have a lot of change from our earlier graphs, I will start with a GARCH(1,1) model to get a feel for how a simple GARCH model fits the data. The general model for GARCH is as follows:

Log Return Series:

$$r_t = \mu_t - a_t, \quad a_t = \sigma_t \epsilon_t$$

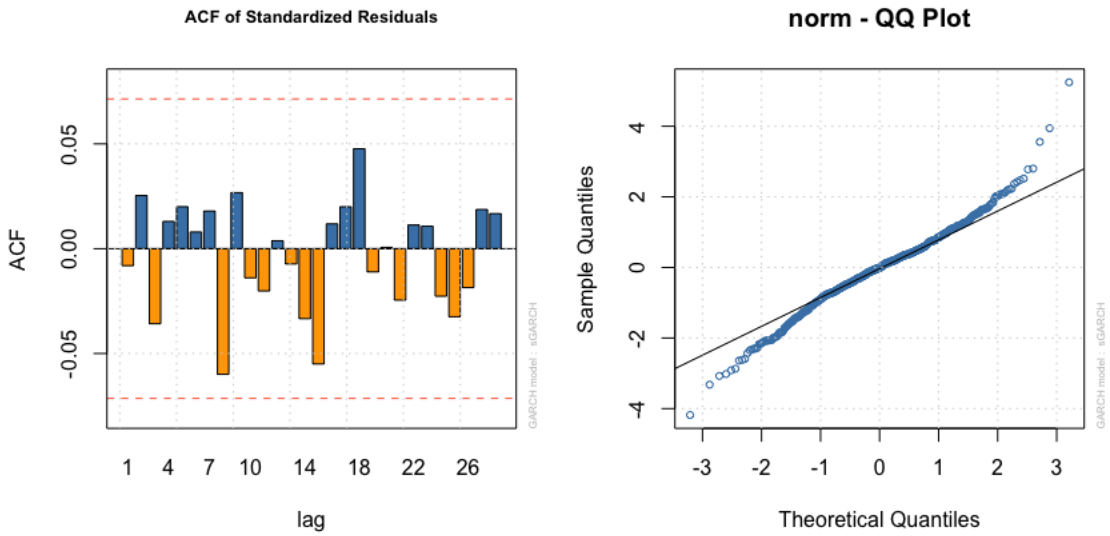
Sigma squared:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{m} \alpha_i a_{t-i}^2 + \sum_{j=1}^{s} \beta_j \sigma_{t-j}^2$$

For a GARCH(1,1) model the coefficients are

Mu	Omega	Alpha1	Beta1
0.00196	0.0000219	0.1995	0.7608

The fitted GARCH(1,1) model returned the following graphs:

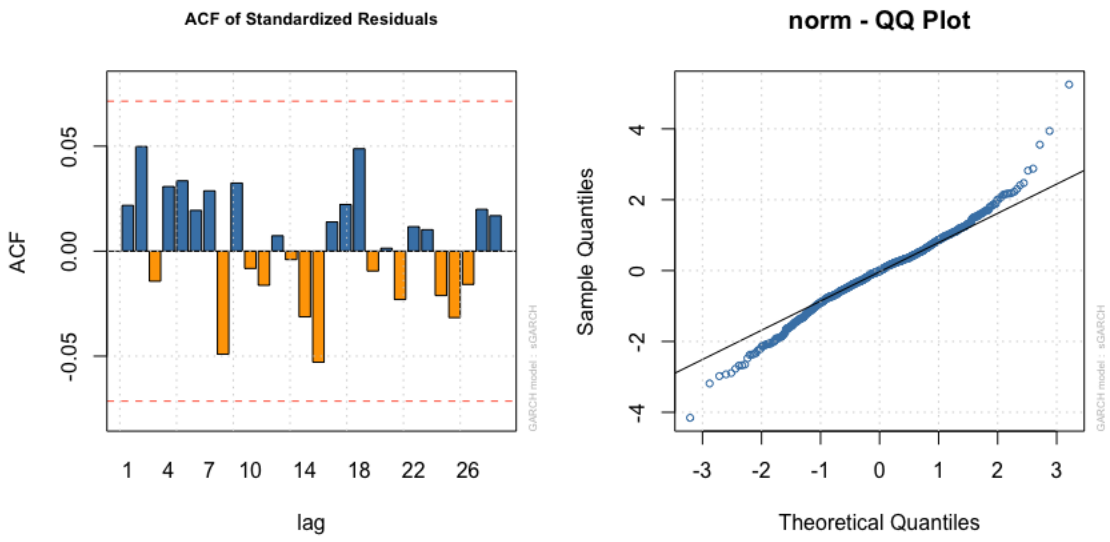


The ACF of the residuals in the GARCH model fall below the 0.05 significant level. However, the QQ Plot is not fit well at the tails. This leads me to tune my model.

ARMA(1,1) – GARCH(1,1) Model:

To tune my model I changed two thing. I added the ARMA(1,1) values and changed the distribution of the model to the student t distribution. To get my values for the ARMA portion I compared the AIC and BIC of an ARMA model with both AR and MA accounted for. The AIC values that minimized the number of parameters while fitting the data best was 4,5. Comparably, the BIC values that minimized the parameters while fitting the data well was 1,1. Since the AIC asked for 9 parameters, and the BIC asked for 2 I continued with the BIC selection. I decided to change the distribution of my model since we know the data doesn't fit a normal distribution well, but the empirical density looked similar to a normal distribution. This is what led me to try the student t distribution, which is a transformation of the normal distribution.

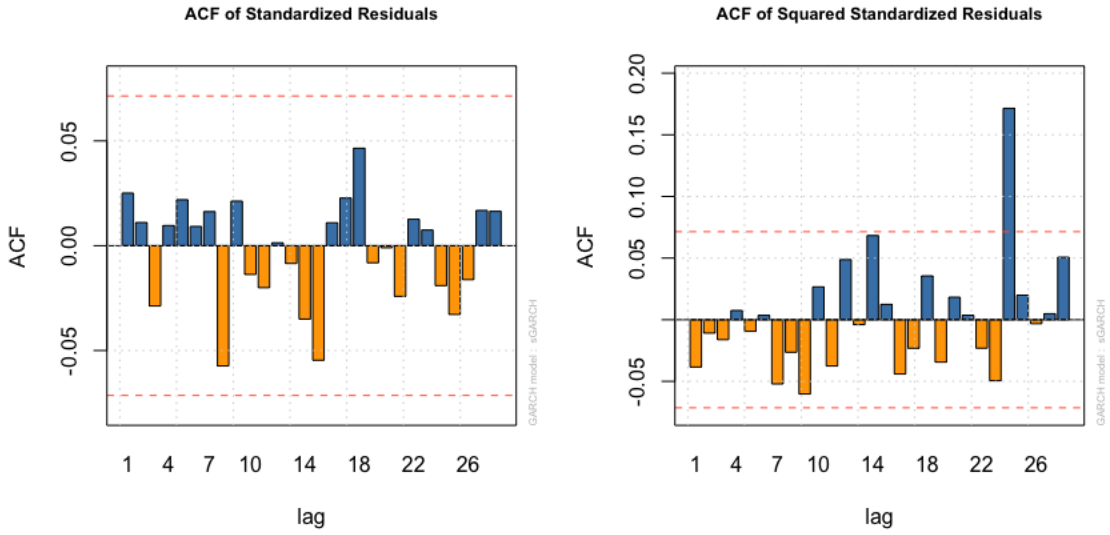
Before trying out the student t distribution I tried an ARMA(1,1) – GARCH(1,1) with normal distribution. It returned the following graphs:



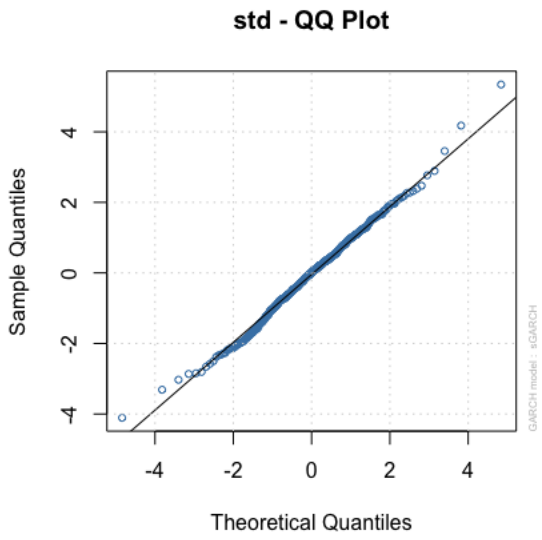
Again, the QQ Plot is not fit well, confirming I should test out the student t distribution.

Final Model:

My final model was an ARMA(1,1) – GARCH(1,1) with student t distribution. It produced the following graphs:



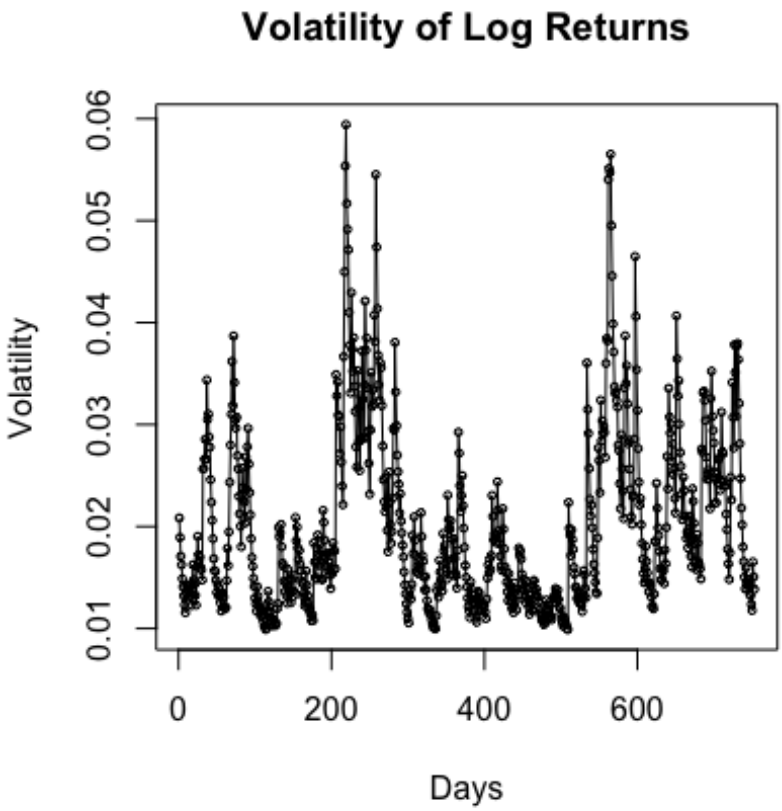
Again, the ACF of the residuals falls below the significant level of 0.05. The squared residuals fall below the 0.05 level as well except for one point. However, when comparing the squared residuals of this model to the others there was no clear outcome that performed better for the squared residuals. Finally, the QQ plot looked as follows:



The tails fit the line much better, and the QQ Plot fits much better overall, confirming that a student t distribution is better than a normal distribution for this model.

Volatility and Forecast:

Finally, we can look at a graph that shows the volatility of the returns.



From the above graph we see that the log returns of Amazon stock over the time period of December 15, 2017 through December 14, 2020 are very volatile. There isn't a time period where the change in log returns is "calm". It appears to always be bouncing around, which we can see by the many peaks in the graph.

The fit of the ARMA(1,1) – GARCH(1,1) model is:

ARMA(1,1)

Log Return Series:

$$r_t - \phi_1 r_{t-1} = \phi_0 + a_t - \phi_1 a_{t-1}, \quad a_t = \sigma_t \epsilon_t$$

GARCH(1,1)

Log Return Series:

$$r_t = \mu_t - a_t, \quad a_t = \sigma_t \epsilon_t$$

Sigma squared:

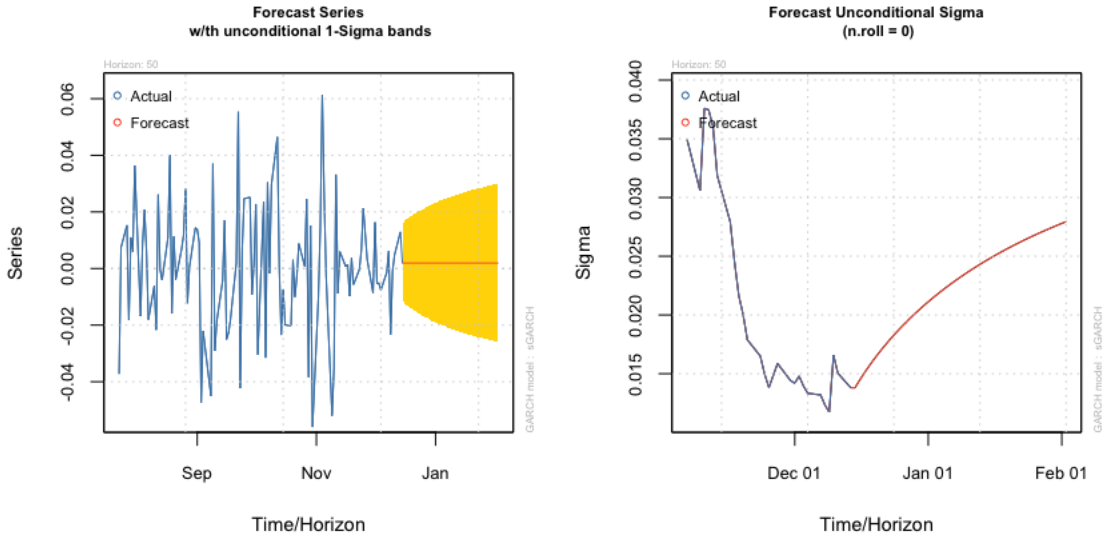
$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{m} \alpha_i a_{t-i}^2 + \sum_{j=1}^{s} \beta_j \sigma_{t-j}^2$$

For a ARMA(1,1) – GARCH(1,1) model the coefficients are

Mu	Omega	Alpha1	Beta1	AR(1)	MA(1)	Shape
0.0019	0.0000196	0.238	0.747	-0.550	0.5172	5.352

Forecast:

I chose a larger step size of a forecast outlook of 50 steps ahead. With the ARMA(1,1) – GARCH(1,1) model and known volatility, the forecast outlook looked as such.



From the forecast we see there is a wide range of volatility, with the potential for a growth in future returns. Given the time of year, I would assume Amazon is experiencing high traffic, which means increased revenue. While the returns are extremely volatile, the potential for a big return is there if you want to take the risk.

Citations

"Amazon.com, Inc. (AMZN) Stock Price, News, Quote & History." *Yahoo! Finance*, Yahoo!, 14 Dec. 2020, finance.yahoo.com/quote/AMZN?p=AMZN.

Author: Luca Ventura. "World's Largest Companies 2020." *Global Finance Magazine*, www.gfmag.com/global-data/economic-data/largest-companies.

"FAQs." *Amazon.com, Inc. - FAQs*, ir.aboutamazon.com/faqs/default.aspx.

Stats 461 Final Report Code

Shane McIntyre

12/16/2020

```
library(quantmod)
```

```
## Loading required package: xts
```

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

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

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

```
## Loading required package: TTR
```

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

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
library(fBasics)
```

```
## Loading required package: timeDate
```

```
## Loading required package: timeSeries
```

```
##
## Attaching package: 'timeSeries'
```

```
## The following object is masked from 'package:zoo':
##
##     time<-
```

```
##
## Attaching package: 'fBasics'
```

```
## The following object is masked from 'package:TTR':
##
##     volatility
```

```
library(rugarch)
```

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

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

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

```
getSymbols("AMZN",from="2017-12-15",to="2020-12-15")
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
## [1] "AMZN"
```

```
head(AMZN)
```

```
##            AMZN.Open AMZN.High AMZN.Low AMZN.Close AMZN.Volume AMZN.Adjust
ed
## 2017-12-15   1179.03   1182.75  1169.33    1179.14     4778600       1179.
14
## 2017-12-18   1187.37   1194.78  1180.91    1190.58     2947600       1190.
58
## 2017-12-19   1189.15   1192.97  1179.14    1187.38     2587800       1187.
38
## 2017-12-20   1190.50   1191.00  1176.00    1177.62     2371200       1177.
62
## 2017-12-21   1175.90   1179.17  1167.64    1174.76     2123100       1174.
76
## 2017-12-22   1172.08   1174.62  1167.83    1168.36     1585100       1168.
36
```

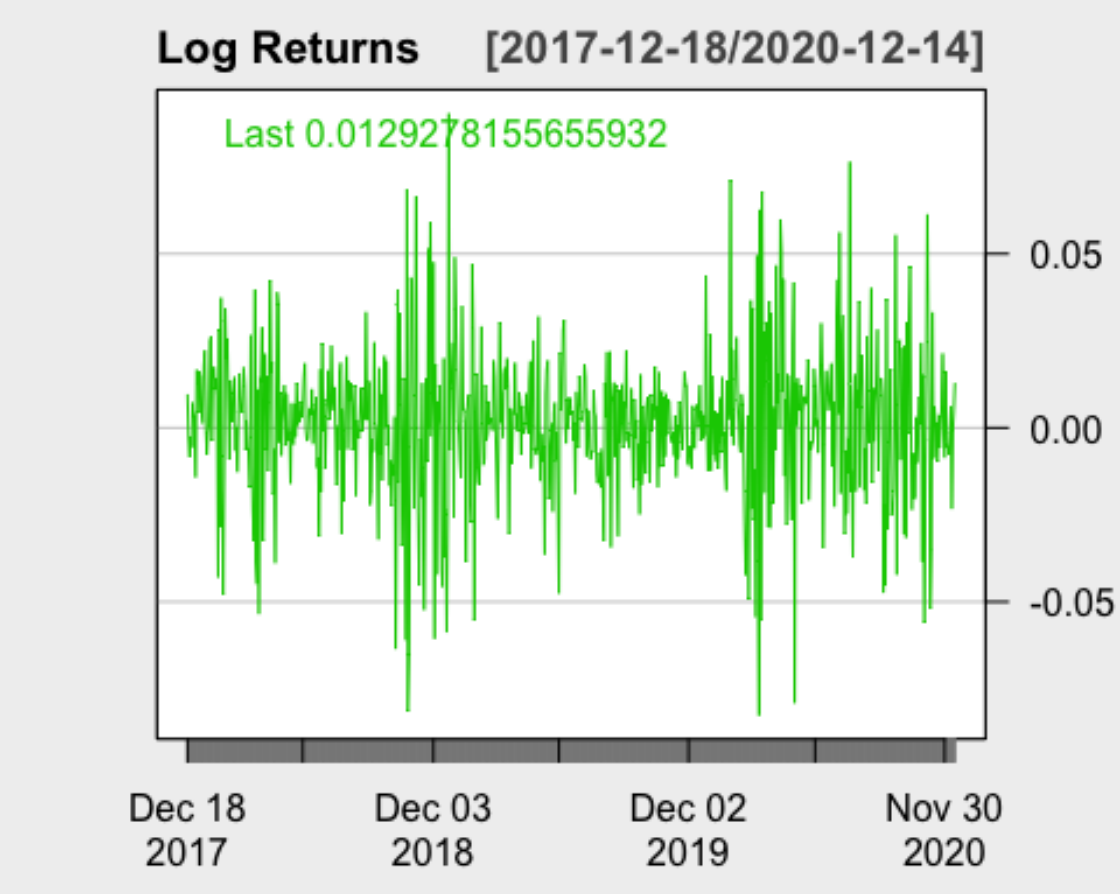
```
tail(AMZN)
```

```
##            AMZN.Open AMZN.High AMZN.Low AMZN.Close AMZN.Volume AMZN.Adjust
ed
## 2020-12-07   3156.48   3180.76  3141.69    3158.00     2751300       3158.
00
## 2020-12-08   3158.90   3184.13  3120.02    3177.29     3286300       3177.
29
## 2020-12-09   3167.89   3174.43  3088.00    3104.20     4100800       3104.
20
## 2020-12-10   3088.99   3142.10  3076.00    3101.49     3030200       3101.
49
## 2020-12-11   3096.66   3118.67  3072.82    3116.42     3064700       3116.
42
## 2020-12-14   3143.00   3190.47  3126.00    3156.97     4155800       3156.
97
```

```
amzn.lrtn = diff(log(AMZN$AMZN.Adjusted))[-1,]
head(amzn.lrtn)

##             AMZN.Adjusted
## 2017-12-18   0.009655174
## 2017-12-19  -0.002691343
## 2017-12-20  -0.008253755
## 2017-12-21  -0.002431568
## 2017-12-22  -0.005462837
## 2017-12-26   0.007163864

chartSeries(AMZN$AMZN.Adjusted, theme = 'white', name = 'Stock Price over tim
e')
```


```
chartSeries(log(AMZN$AMZN.Adjusted), theme = 'white')
```


```
chartSeries(amzn.lrtn, theme = 'white', name = 'Log Returns')
```




```
basicStats(AMZN$AMZN.Adjusted)
```

```
##             AMZN.Adjusted
## nobs         7.540000e+02
## NAs          0.000000e+00
## Minimum      1.168360e+03
## Maximum      3.531450e+03
## 1. Quartile  1.666248e+03
## 3. Quartile  2.020095e+03
## Mean         2.008269e+03
## Median       1.821995e+03
## Sum          1.514235e+06
## SE Mean      2.041210e+01
## LCL Mean     1.968197e+03
## UCL Mean     2.048340e+03
## Variance     3.141571e+05
## Stdev        5.604972e+02
## Skewness     1.208254e+00
## Kurtosis     3.213770e-01
```

```
kurtosis(AMZN$AMZN.Adjusted)
```

```
## [1] 0.3213765
## attr(,"method")
## [1] "excess"
```

```
basicStats(amzn.lrtn)
```

```
##             AMZN.Adjusted
## nobs           753.000000
## NAs              0.000000
## Minimum         -0.082535
## Maximum          0.090254
## 1. Quartile     -0.008000
## 3. Quartile      0.011696
## Mean             0.001308
## Median           0.001749
## Sum              0.984827
## SE Mean          0.000761
## LCL Mean        -0.000187
## UCL Mean         0.002802
## Variance         0.000436
## Stdev            0.020890
## Skewness        -0.122862
## Kurtosis         2.292832
```

```
max(AMZN$AMZN.Adjusted) - min(AMZN$AMZN.Adjusted)
```

```
## [1] 2363.09
```

```
# Mean test statistic
m = mean(amzn.lrtn)/(sqrt(var(amzn.lrtn)/length(amzn.lrtn)))
m
```

```
##                AMZN.Adjusted
## AMZN.Adjusted      1.718036
```

```
#Skewness test statistic
s = skewness(amzn.lrtn)/sqrt(6/length(amzn.lrtn))
s
```

```
## [1] -1.376386
## attr(,"method")
## [1] "moment"
```

```
#Excess Kurtosis test statistic
k = kurtosis(amzn.lrtn)/sqrt(24/length(amzn.lrtn))
k
```

```
## [1] 12.84293
## attr(,"method")
## [1] "excess"
```

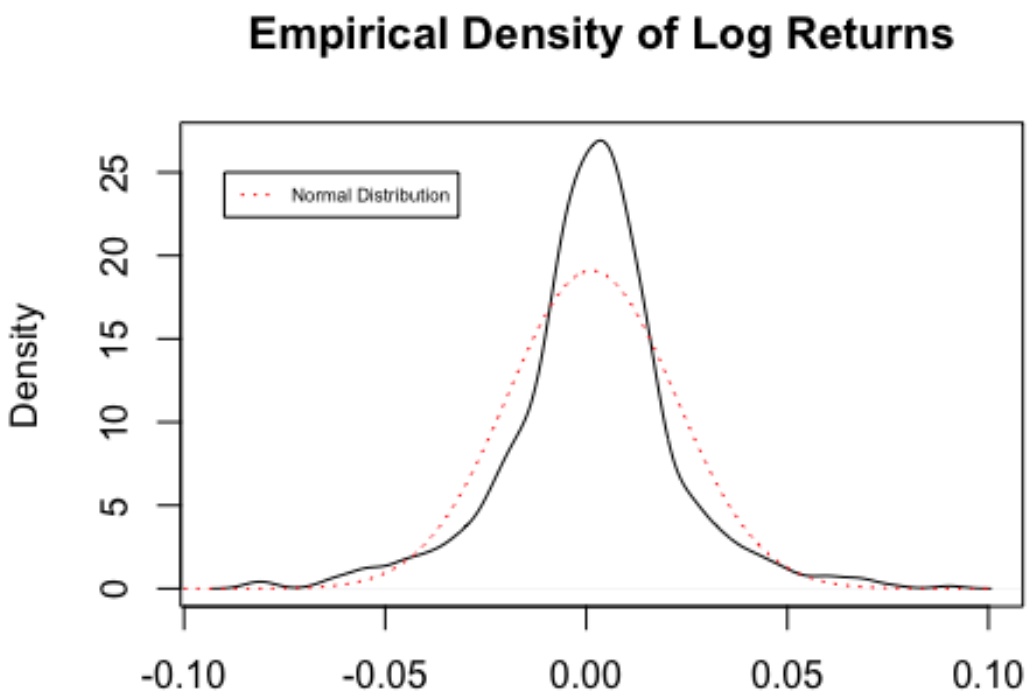
```r
#JB test statistic from calculated values
jb = s^2 + k^2
jb[1]
```

```
## [1] 166.8352
```

```r
#R test for normality
normalTest(as.vector(amzn.lrtn), method = 'jb')
```

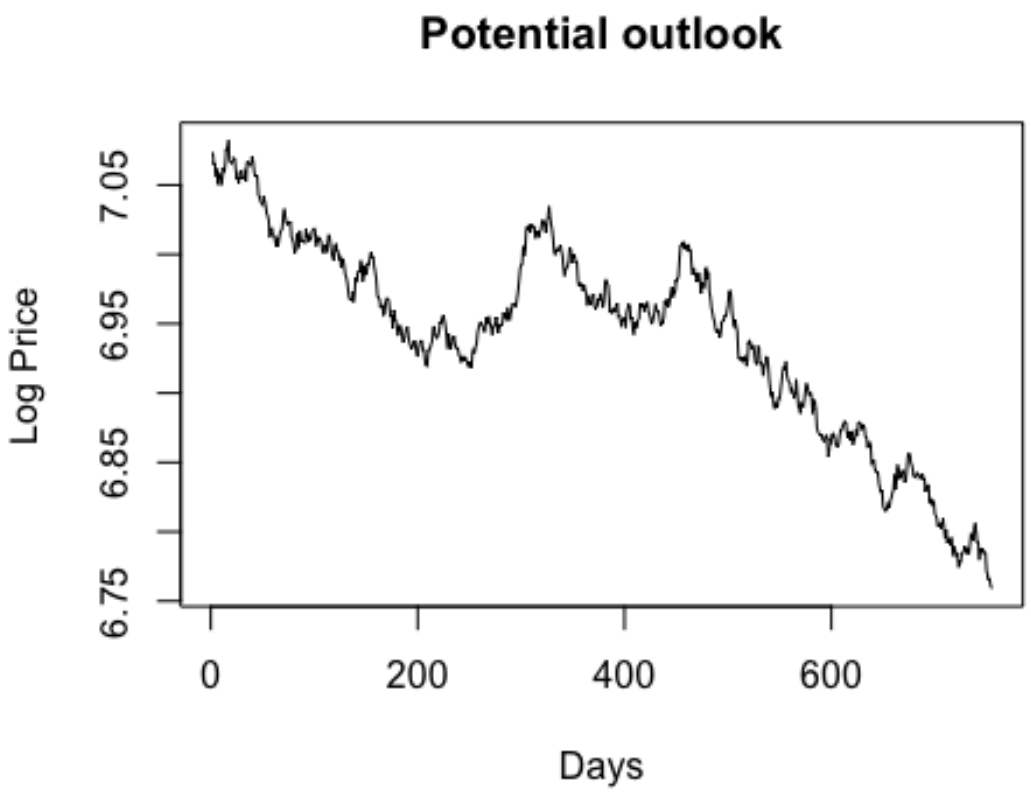
```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 168.8757
##   P VALUE:
##     Asymptotic p Value: < 2.2e-16
##
## Description:
##  Wed Dec 16 22:25:57 2020 by user:
```

```r
#Reject null hypothesis of normality assumption

#Density plots
plot(density(amzn.lrtn), main = 'Empirical Density of Log Returns', xlab = ''
)
x = seq(-.1,.1,.001)
y = dnorm(x,mean(amzn.lrtn),sd(amzn.lrtn))
lines(x,y,lty = 3,col = 'red')
legend(-.09, 25, legend = c('Normal Distribution'), col = 'red', lty = 3, cex
= 0.5)
```

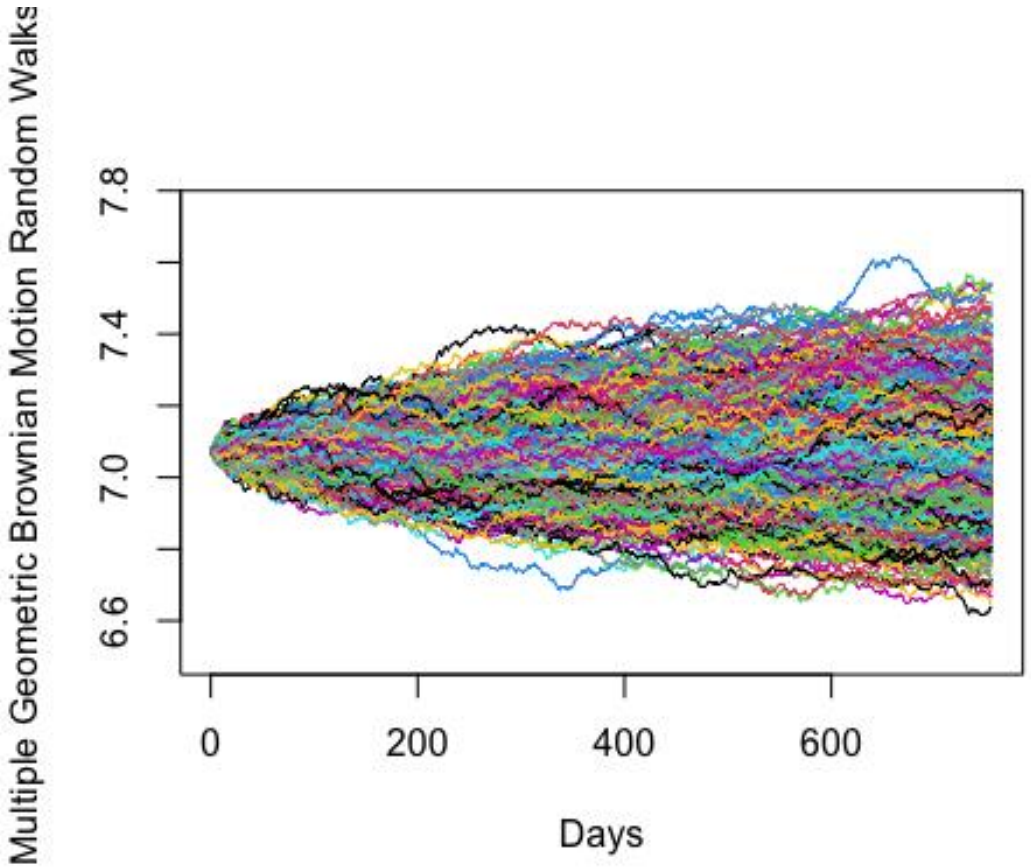



```
#Geometric Brownian Motion
log_price = log(AMZN$AMZN.Adjusted)
m = mean(amzn.lrtn)
s = sd(amzn.lrtn)
s0 = as.numeric((log_price$AMZN.Adjusted[1]))
n = length(AMZN$AMZN.Adjusted)
dt = 1/n

#One simulated path
r = m*dt + s*rnorm(n, mean = 0, sd = sqrt(dt))
S = c(s0, rep(0,n))
for (i in 1:n) {
  S[i+1] = S[i]*r[i] + S[i]
}
head(S)

## [1] 7.072541 7.072461 7.064983 7.064450 7.055771 7.061475

tail(S)

## [1] 6.771486 6.769175 6.765736 6.765382 6.761139 6.759016

plot(log_price, main = 'True log price')
```

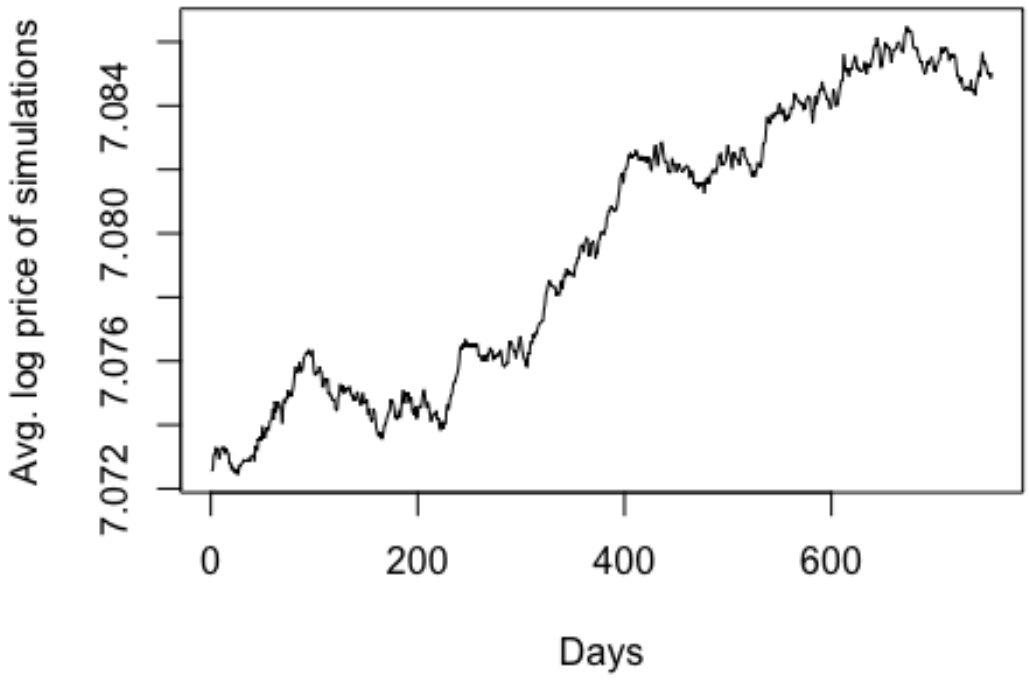

```
plot(S, type = 'l', main = 'Potential outlook', ylab = 'Log Price', xlab = 'D
ays')
```




```r
#Multiple paths
N = 1000
r = m*dt + s*rnorm(n*N, mean = 0, sd = sqrt(dt))
r = matrix(r,n,N)

s = matrix(rep(0,n*N),n,N)
s = rbind(rep(s0,N), s)
for (j in 1:N) {
  for (i in 1:n) {
    s[i+1,j] = s[i,j]*r[i,j] + s[i,j]
  }
}

plot(s[,1], ylab = 'Multiple Geometric Brownian Motion Random Walks', xlab =
'Days', type = 'l', ylim = c(6.5,7.75))
for (i in 2:N) {
  lines(s[,i], col = i)
}
```



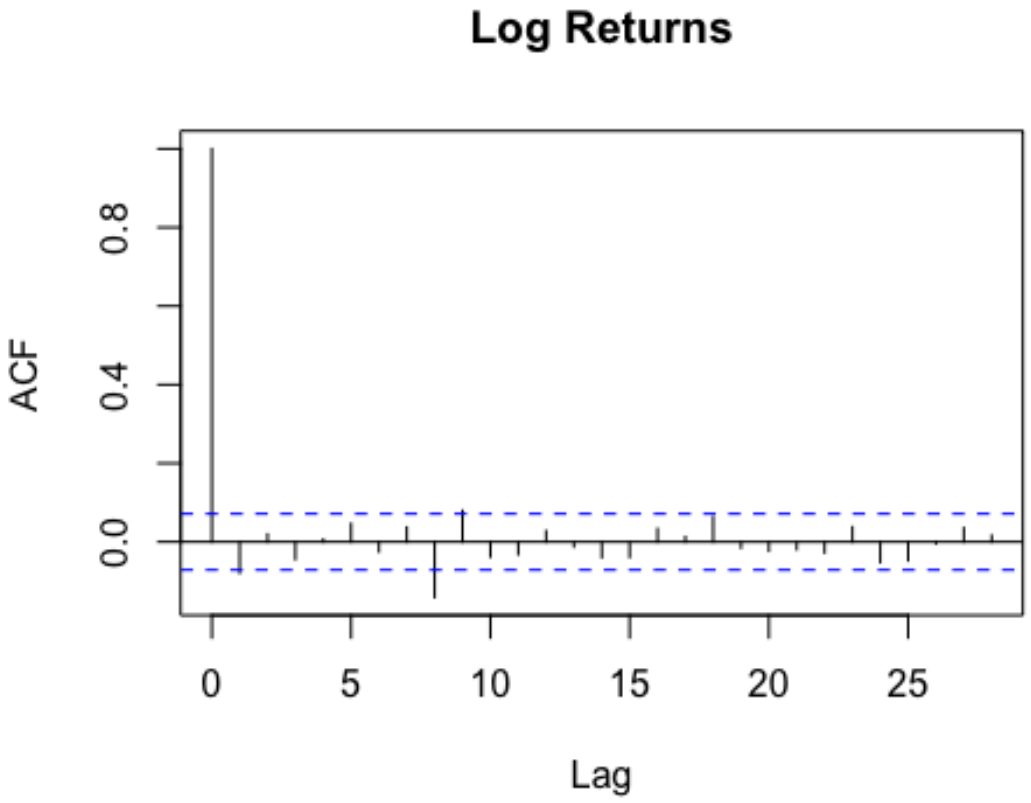

```
max(log_price)

## [1] 8.169464

max(s[n+1,])

## [1] 7.541177

avg_s = rowMeans(s)
plot(avg_s, type = 'l', xlab = 'Days', ylab = 'Avg. log price of simulations'
)
```



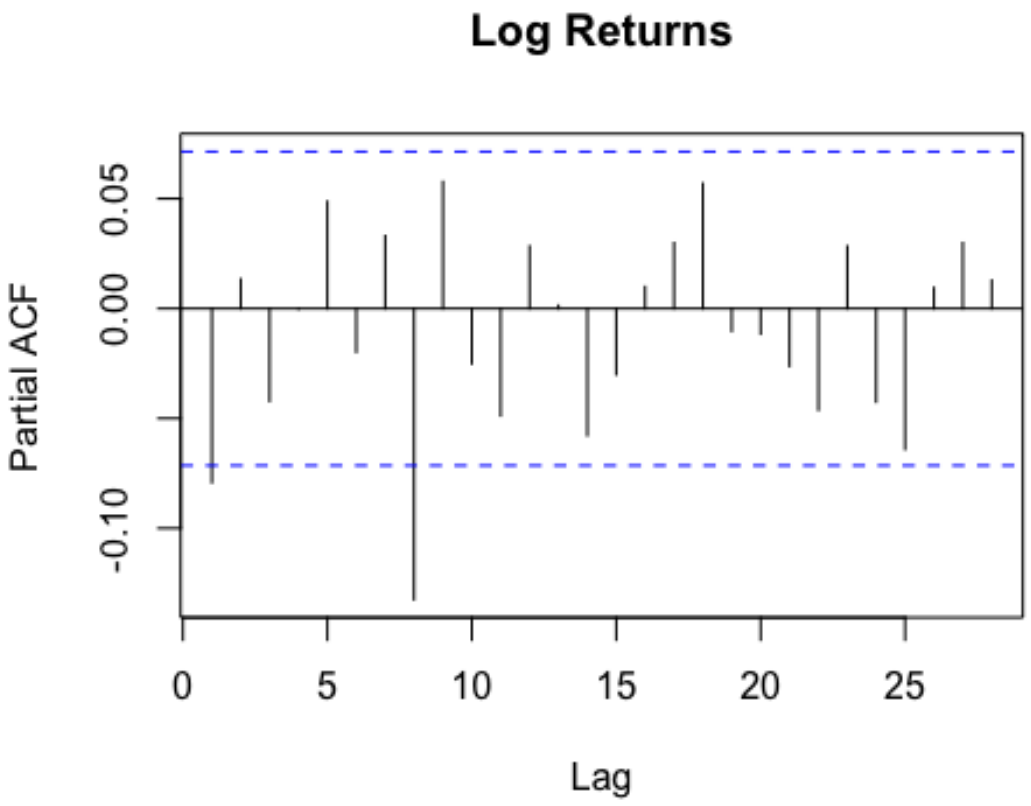

```
dif = log_price - avg_s[-1]
plot(dif, type = 'l', main = 'Difference in price')
```


```
# GARCH Model
m1 = arima(amzn.lrtn, order = c(0,0,0))
m1

##
## Call:
## arima(x = amzn.lrtn, order = c(0, 0, 0))
##
## Coefficients:
##       intercept
##          0.0013
## s.e.     0.0008
##
## sigma^2 estimated as 0.0004358:  log likelihood = 1845.02,  aic = -3686.05

resid = as.numeric(residuals(m1))
Box.test(resid^2, lag = 5, fitdf = 0)

##
##  Box-Pierce test
##
## data:  resid^2
## X-squared = 175.28, df = 5, p-value < 2.2e-16
```

```r
acf(amzn.lrtn, main = 'Log Returns')
```




```r
pacf(amzn.lrtn, main = 'Log Returns')
```




```
spec1 = ugarchspec(
  mean.model = list(armaOrder = c(0,0)),
  variance.model = list(model = 'sGARCH', garchOrder = c(1,1)),
  distribution.model = 'norm')
m = ugarchfit(amzn.lrtn, spec = spec1)
m

##
## *---------------------------------*
## *          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      0.001960    0.000569   3.4444 0.000572
## omega   0.000022    0.000006   3.5873 0.000334
```

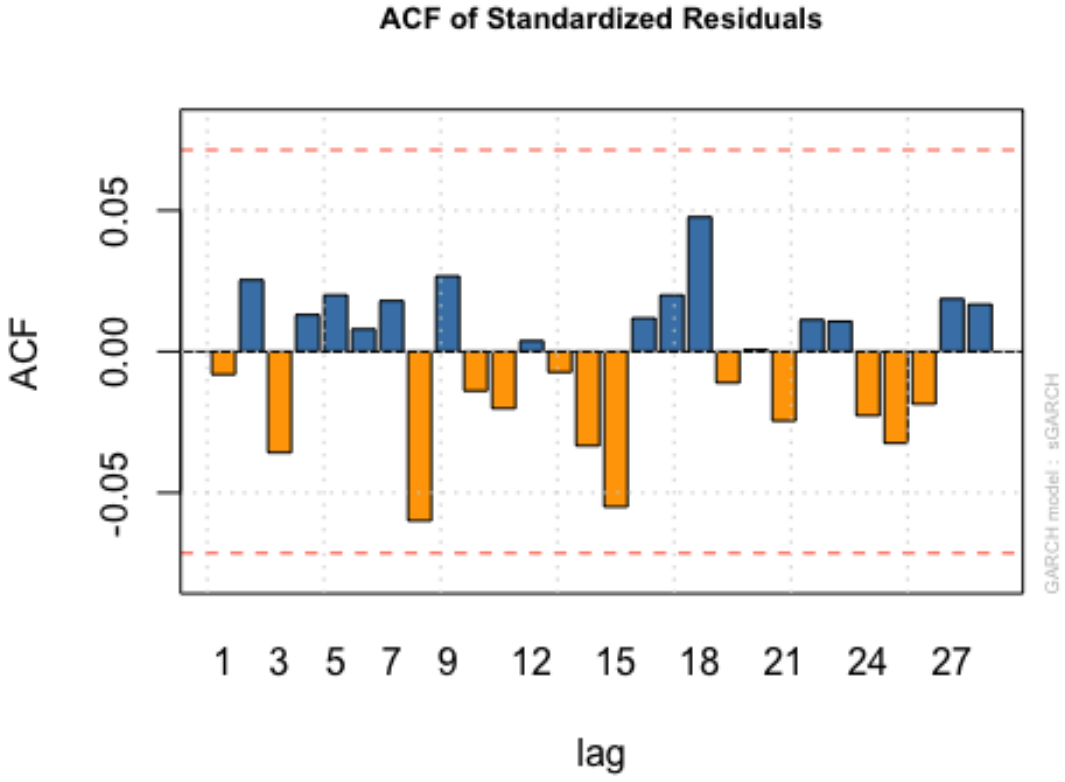
```
## alpha1  0.199523    0.038005   5.2499 0.000000
## beta1   0.760768    0.038460  19.7806 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.001960    0.000577   3.3968 0.000682
## omega   0.000022    0.000008   2.7304 0.006327
## alpha1  0.199523    0.040153   4.9691 0.000001
## beta1   0.760768    0.043874  17.3397 0.000000
##
## LogLikelihood : 1940.638
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.1438
## Bayes        -5.1192
## Shibata      -5.1438
## Hannan-Quinn -5.1343
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.04965  0.8237
## Lag[2*(p+q)+(p+q)-1][2]   0.29322  0.7995
## Lag[4*(p+q)+(p+q)-1][5]   1.13478  0.8285
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.6779  0.4103
## Lag[2*(p+q)+(p+q)-1][5]    0.7685  0.9094
## Lag[4*(p+q)+(p+q)-1][9]    1.8100  0.9264
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01876 0.500 2.000  0.8911
## ARCH Lag[5]   0.11774 1.440 1.667  0.9834
## ARCH Lag[7]   0.92317 2.315 1.543  0.9257
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.5382
## Individual Statistics:
## mu     0.14995
## omega  0.16846
```

```
## alpha1 0.09778
## beta1  0.16174
##
## 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          0.02739 0.9782
## Negative Sign Bias 0.29514 0.7680
## Positive Sign Bias 0.61633 0.5379
## Joint Effect       0.50491 0.9178
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     39.27    0.0040742
## 2    30     59.95    0.0006269
## 3    40     66.07    0.0043496
## 4    50     83.85    0.0014209
##
##
## Elapsed time : 0.1157761
```

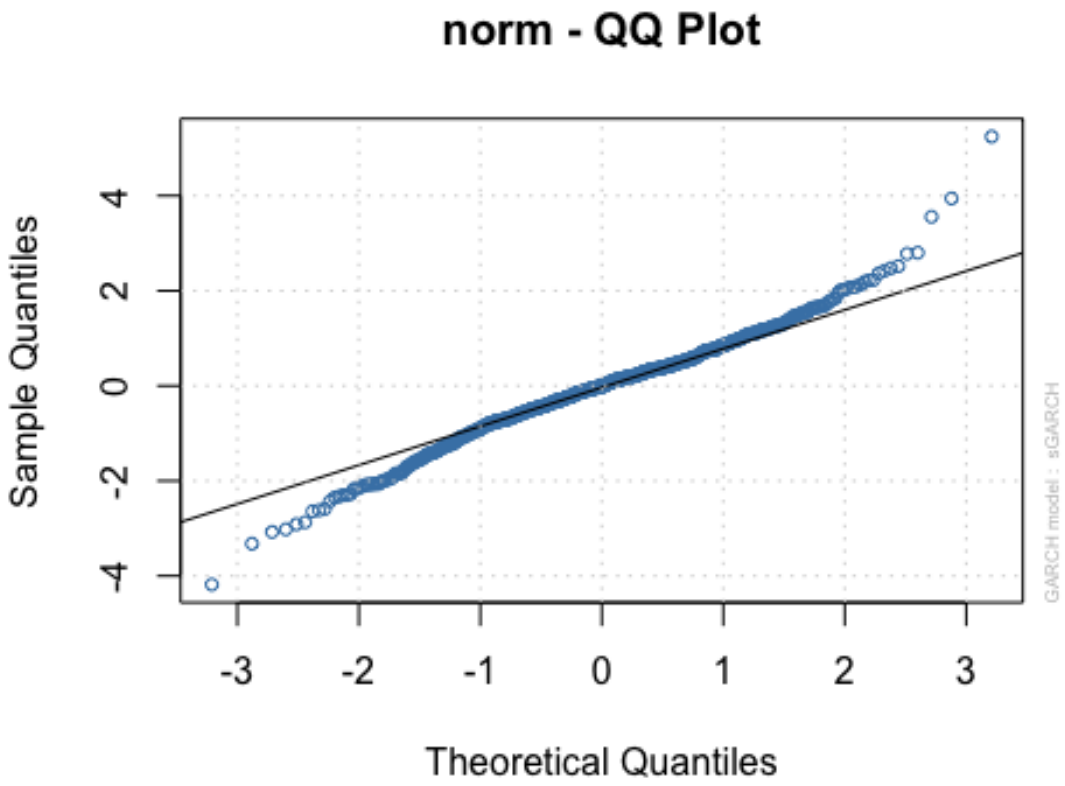
```
coef(m)
```

```
##           mu        omega       alpha1        beta1
## 1.960477e-03 2.186144e-05 1.995234e-01 7.607675e-01
```

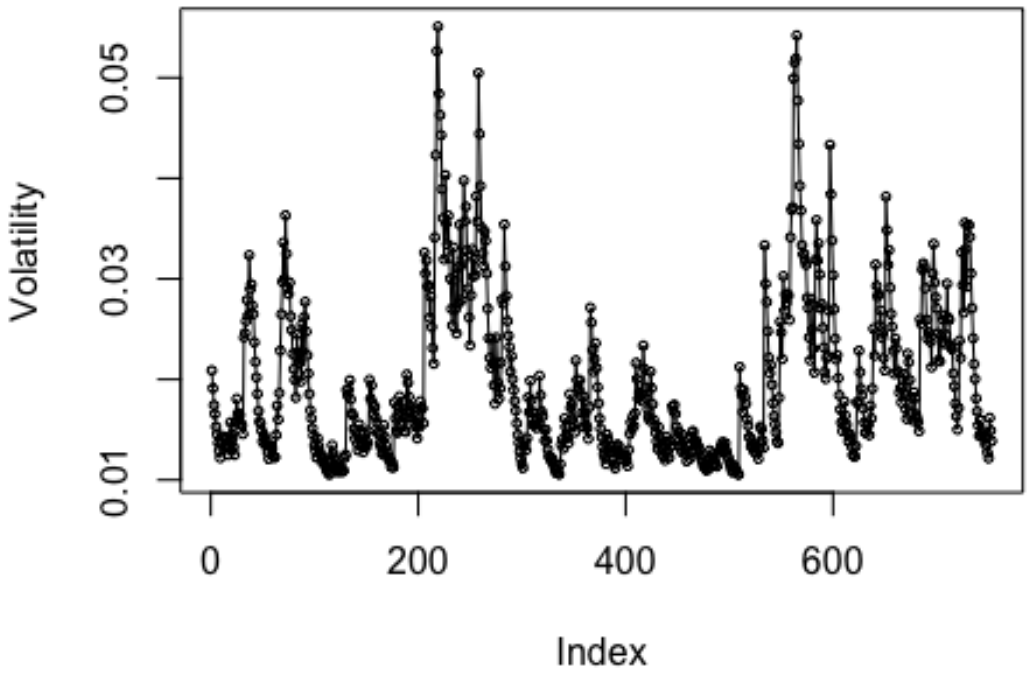
```
plot(m, which = 10)
```




```
plot(m, which = 9, main = '')
```

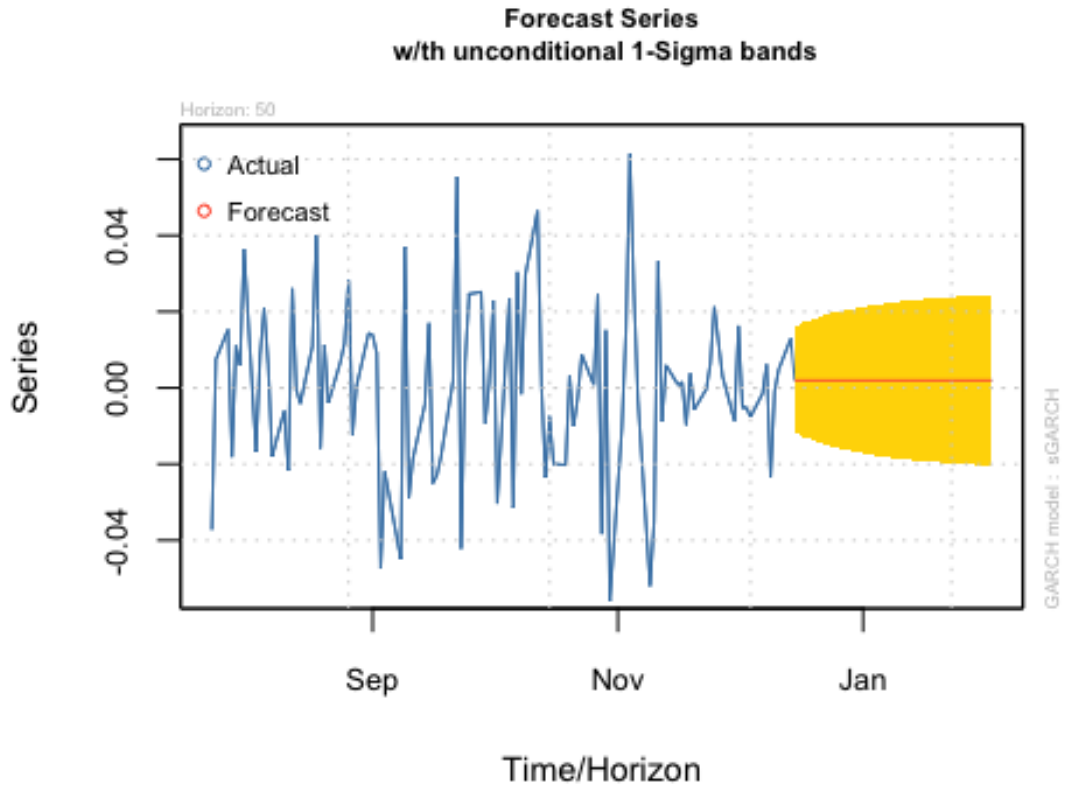



```
plot(as.numeric(sigma(m)), main = '', type = 'o', ylab = 'Volatility', cex =
0.5)
```

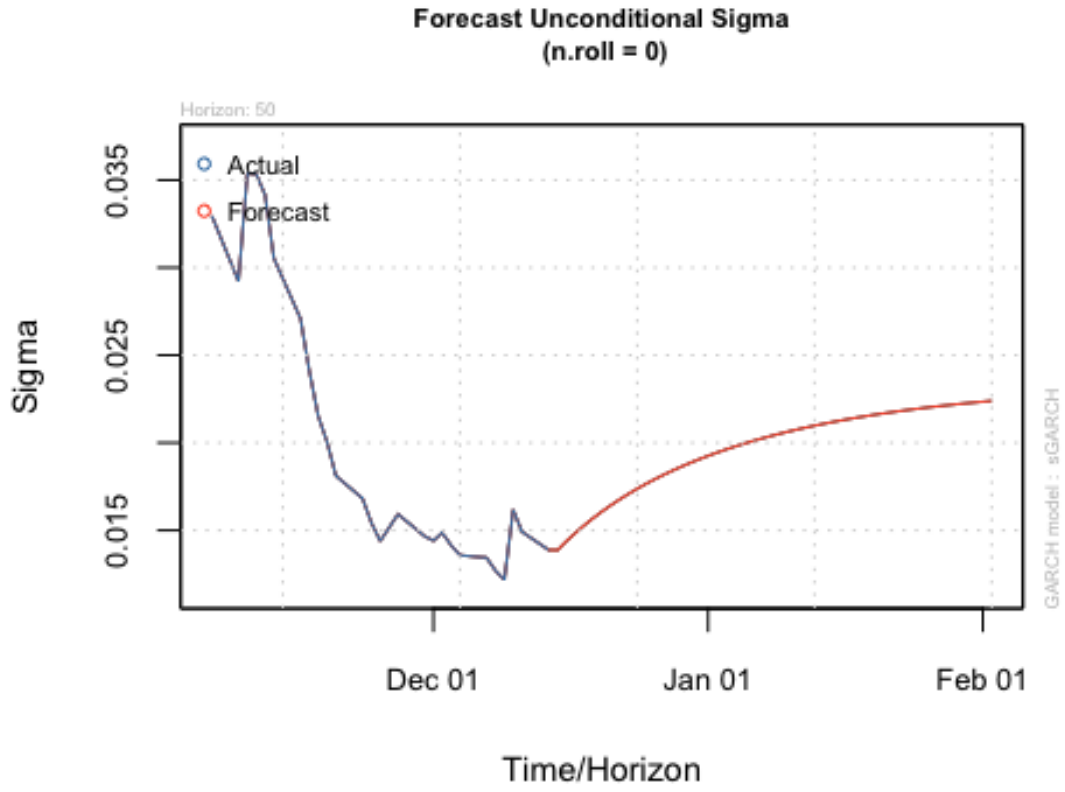



```
forecast = ugarchforecast(m, n.ahead = 50)
forecast

##
## *----------------------------------*
## *       GARCH Model Forecast       *
## *----------------------------------*
## Model: sGARCH
## Horizon: 50
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=2020-12-14]:
##       Series   Sigma
## T+1  0.00196 0.01387
## T+2  0.00196 0.01438
## T+3  0.00196 0.01484
## T+4  0.00196 0.01528
## T+5  0.00196 0.01569
## T+6  0.00196 0.01607
## T+7  0.00196 0.01642
## T+8  0.00196 0.01676
## T+9  0.00196 0.01708
```

```
## T+10 0.00196 0.01737
## T+11 0.00196 0.01766
## T+12 0.00196 0.01792
## T+13 0.00196 0.01818
## T+14 0.00196 0.01841
## T+15 0.00196 0.01864
## T+16 0.00196 0.01886
## T+17 0.00196 0.01906
## T+18 0.00196 0.01925
## T+19 0.00196 0.01944
## T+20 0.00196 0.01961
## T+21 0.00196 0.01978
## T+22 0.00196 0.01994
## T+23 0.00196 0.02009
## T+24 0.00196 0.02024
## T+25 0.00196 0.02037
## T+26 0.00196 0.02051
## T+27 0.00196 0.02063
## T+28 0.00196 0.02075
## T+29 0.00196 0.02087
## T+30 0.00196 0.02098
## T+31 0.00196 0.02108
## T+32 0.00196 0.02118
## T+33 0.00196 0.02127
## T+34 0.00196 0.02137
## T+35 0.00196 0.02145
## T+36 0.00196 0.02154
## T+37 0.00196 0.02162
## T+38 0.00196 0.02169
## T+39 0.00196 0.02177
## T+40 0.00196 0.02184
## T+41 0.00196 0.02190
## T+42 0.00196 0.02197
## T+43 0.00196 0.02203
## T+44 0.00196 0.02209
## T+45 0.00196 0.02214
## T+46 0.00196 0.02220
## T+47 0.00196 0.02225
## T+48 0.00196 0.02230
## T+49 0.00196 0.02235
## T+50 0.00196 0.02239

plot(forecast, which=1)
```




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




```
#ARIMA model
p = 5
q = 5
aic = matrix(rep(0, p*q),p,q)
bic = matrix(rep(0, p*q),p,q)
for (i in 1:p) {
  for (j in 1:q) {
    m = arima(amzn.lrtn, order = c(i,0,j))
    aic[i,j] = AIC(m)
    bic[i,j] = BIC(m)
  }
}
```

```
## Warning in arima(amzn.lrtn, order = c(i, 0, j)): possible convergence prob
lem:
## optim gave code = 1

## Warning in arima(amzn.lrtn, order = c(i, 0, j)): possible convergence prob
lem:
## optim gave code = 1
```

```
aic
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -3687.186 -3685.205 -3684.357 -3682.710 -3686.211
## [2,] -3685.201 -3684.165 -3687.542 -3689.855 -3689.634
## [3,] -3684.241 -3692.278 -3691.114 -3689.119 -3682.859
## [4,] -3682.250 -3690.762 -3688.871 -3697.381 -3698.844
## [5,] -3688.081 -3689.315 -3691.724 -3698.805 -3695.234
```

```
bic
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] -3668.689 -3662.085 -3656.613 -3650.342 -3649.218
## [2,] -3662.081 -3656.421 -3655.174 -3652.862 -3648.018
## [3,] -3656.497 -3659.910 -3654.121 -3647.503 -3636.619
## [4,] -3649.881 -3653.769 -3647.254 -3651.140 -3647.980
## [5,] -3651.088 -3647.698 -3645.484 -3647.940 -3639.745
```

```
which(aic == min(aic), arr.ind = T) #4,0,5
```

```
##      row col
## [1,]   4   5
```

```
which(bic == min(bic), arr.ind = T) #1,0,1
```

```
##      row col
## [1,]   1   1
```

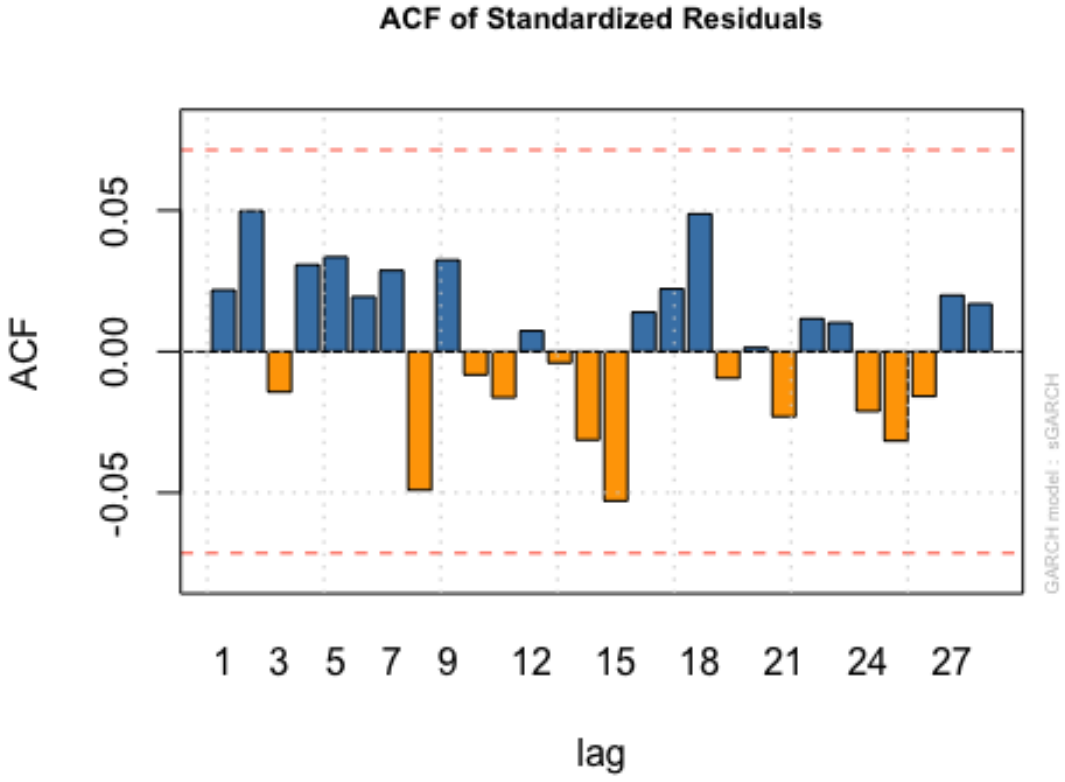
```
# Garch with ARMA(1,1)
spec1 = ugarchspec(
  mean.model = list(armaOrder = c(1,1)),
  variance.model = list(model = 'sGARCH', garchOrder = c(1,1)),
  distribution.model = 'norm')
m = ugarchfit(amzn.lrtn, spec = spec1)
m
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.002003    0.000473   4.2394 0.000022
## ar1     0.821861    0.129522   6.3454 0.000000
## ma1    -0.855367    0.115726  -7.3913 0.000000
## omega   0.000022    0.000006   3.6185 0.000296
```

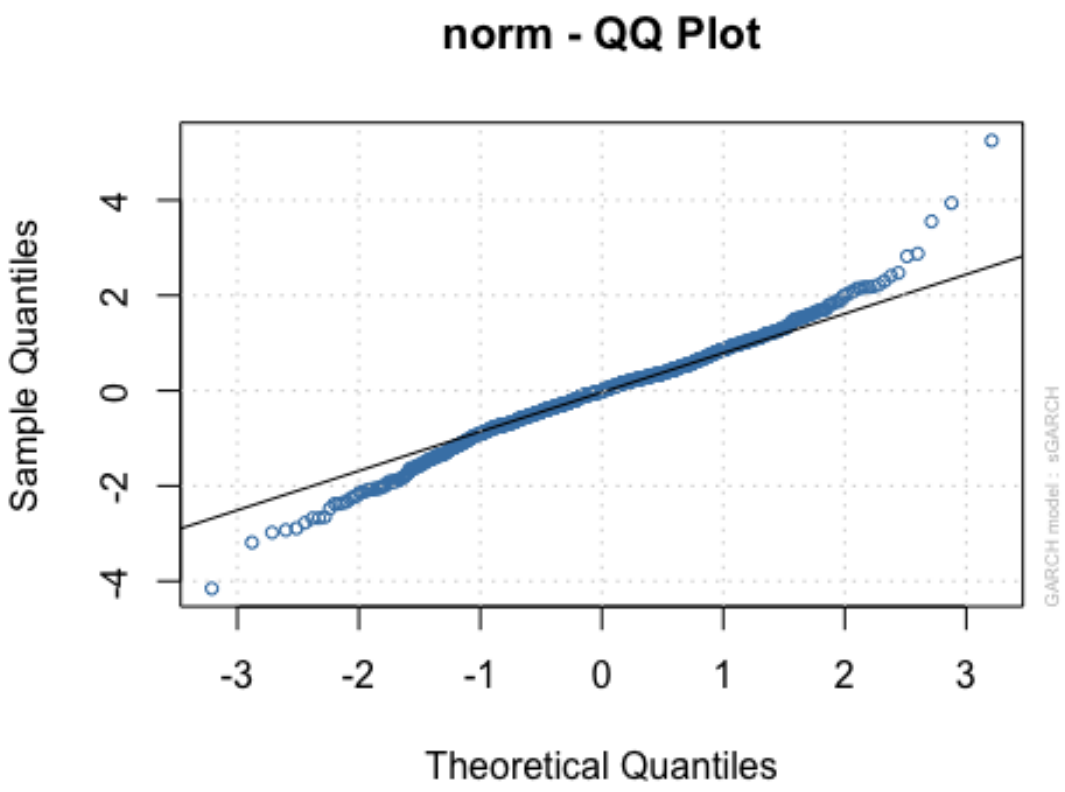
```
## alpha1  0.203992    0.039060   5.2225 0.000000
## beta1   0.756357    0.039105  19.3417 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.002003    0.000521   3.8418 0.000122
## ar1     0.821861    0.093651   8.7758 0.000000
## ma1    -0.855367    0.081918 -10.4418 0.000000
## omega   0.000022    0.000008   2.7960 0.005174
## alpha1  0.203992    0.041729   4.8885 0.000001
## beta1   0.756357    0.044836  16.8693 0.000000
##
## LogLikelihood : 1941.85
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.1417
## Bayes        -5.1049
## Shibata      -5.1418
## Hannan-Quinn -5.1275
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.3593  0.5489
## Lag[2*(p+q)+(p+q)-1][5]    2.4148  0.8217
## Lag[4*(p+q)+(p+q)-1][9]    3.9394  0.7051
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.6410  0.4233
## Lag[2*(p+q)+(p+q)-1][5]    0.7125  0.9205
## Lag[4*(p+q)+(p+q)-1][9]    1.7872  0.9288
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01056 0.500 2.000  0.9182
## ARCH Lag[5]   0.14046 1.440 1.667  0.9788
## ARCH Lag[7]   0.95517 2.315 1.543  0.9207
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.9325
## Individual Statistics:
```

```
## mu     0.22551
## ar1    0.08641
## ma1    0.08110
## omega  0.17543
## alpha1 0.09632
## beta1  0.16435
##
## 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           0.8649 0.3874
## Negative Sign Bias  0.6805 0.4964
## Positive Sign Bias  0.1327 0.8944
## Joint Effect        1.1932 0.7546
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     52.60    0.0000537
## 2    30     58.43    0.0009633
## 3    40     73.29    0.0007267
## 4    50     64.07    0.0728635
##
##
## Elapsed time : 0.133441
```

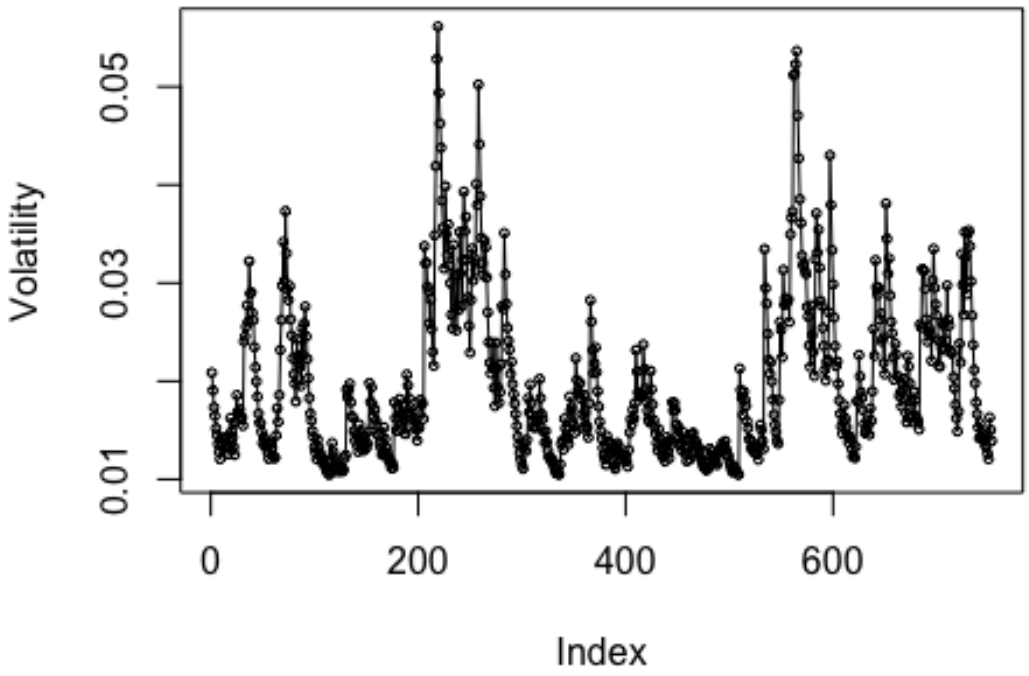
```
plot(m, which = 10)
```




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

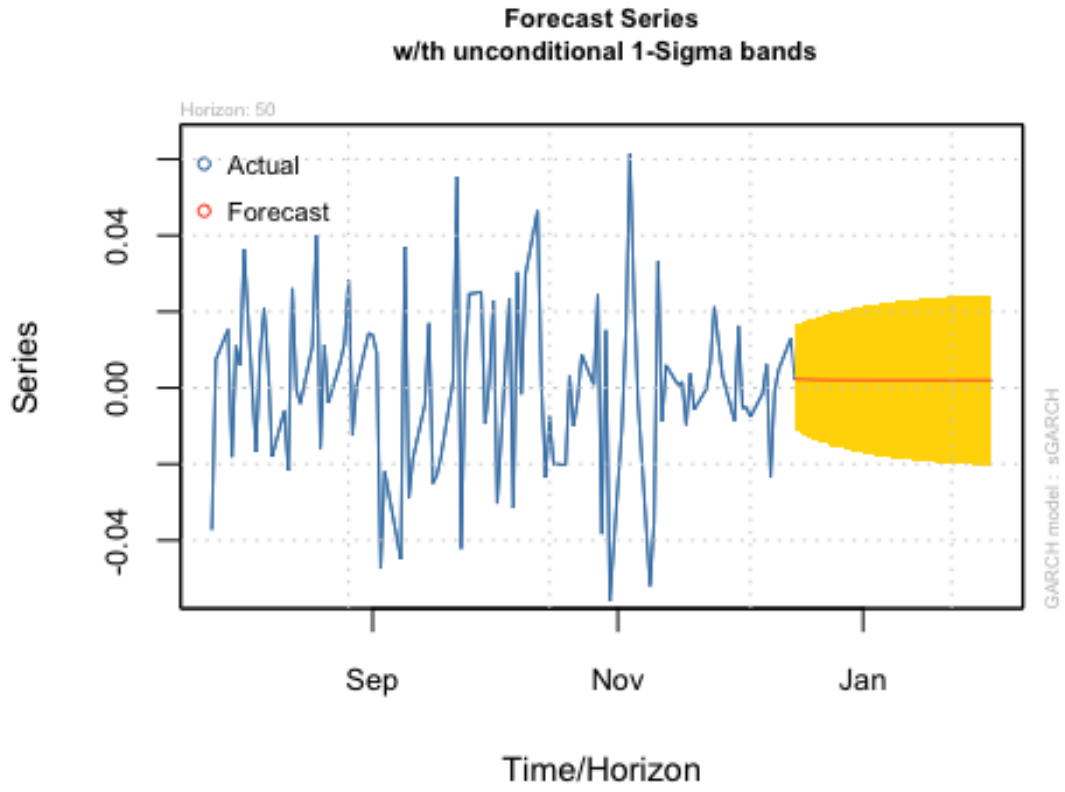



```
plot(as.numeric(sigma(m)), main = '', type = 'o', ylab = 'Volatility', cex =
0.5)
```

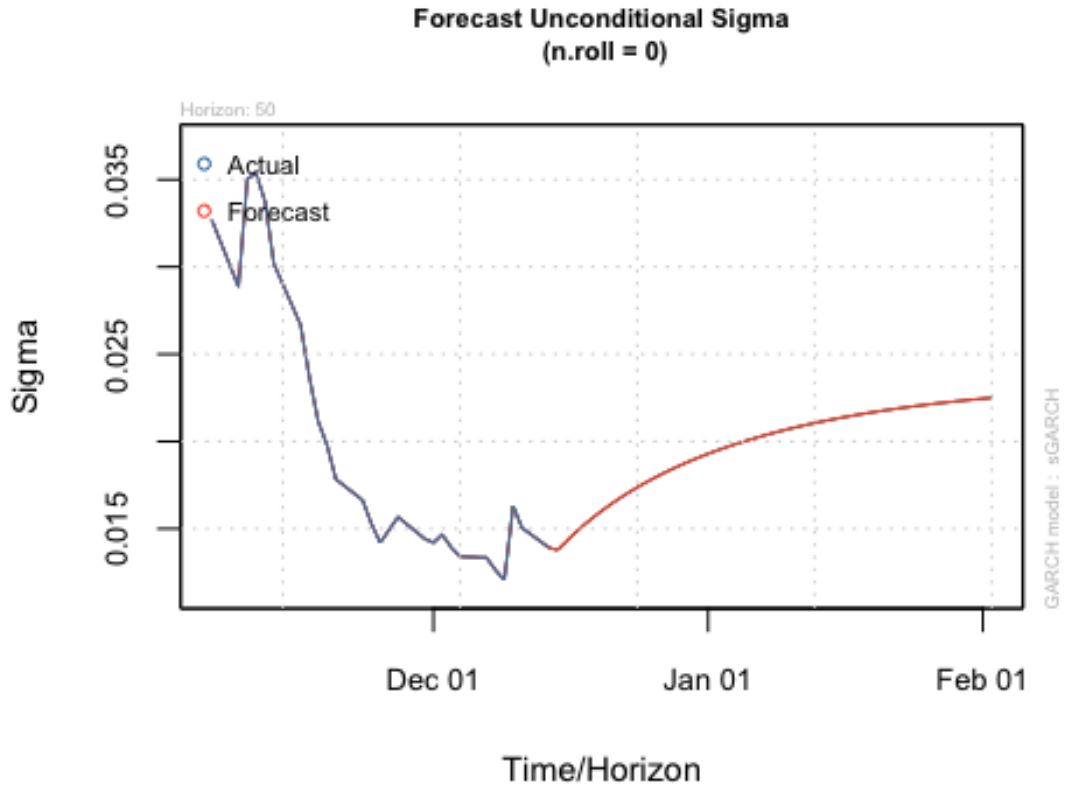



```
forecast = ugarchforecast(m, n.ahead = 50)
forecast

##
## *----------------------------------*
## *       GARCH Model Forecast       *
## *----------------------------------*
## Model: sGARCH
## Horizon: 50
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=2020-12-14]:
##        Series   Sigma
## T+1  0.002339 0.01377
## T+2  0.002279 0.01429
## T+3  0.002230 0.01477
## T+4  0.002190 0.01522
## T+5  0.002156 0.01564
## T+6  0.002129 0.01603
## T+7  0.002107 0.01639
## T+8  0.002088 0.01674
## T+9  0.002073 0.01706
```

```
## T+10 0.002061 0.01737
## T+11 0.002050 0.01766
## T+12 0.002042 0.01793
## T+13 0.002035 0.01819
## T+14 0.002030 0.01843
## T+15 0.002025 0.01866
## T+16 0.002021 0.01888
## T+17 0.002018 0.01909
## T+18 0.002015 0.01929
## T+19 0.002013 0.01948
## T+20 0.002011 0.01966
## T+21 0.002010 0.01983
## T+22 0.002009 0.01999
## T+23 0.002008 0.02015
## T+24 0.002007 0.02030
## T+25 0.002006 0.02044
## T+26 0.002006 0.02057
## T+27 0.002005 0.02070
## T+28 0.002005 0.02082
## T+29 0.002005 0.02094
## T+30 0.002004 0.02105
## T+31 0.002004 0.02116
## T+32 0.002004 0.02126
## T+33 0.002004 0.02136
## T+34 0.002004 0.02145
## T+35 0.002004 0.02154
## T+36 0.002004 0.02162
## T+37 0.002004 0.02171
## T+38 0.002004 0.02178
## T+39 0.002004 0.02186
## T+40 0.002003 0.02193
## T+41 0.002003 0.02200
## T+42 0.002003 0.02206
## T+43 0.002003 0.02213
## T+44 0.002003 0.02219
## T+45 0.002003 0.02224
## T+46 0.002003 0.02230
## T+47 0.002003 0.02235
## T+48 0.002003 0.02240
## T+49 0.002003 0.02245
## T+50 0.002003 0.02250

plot(forecast, which=1)
```



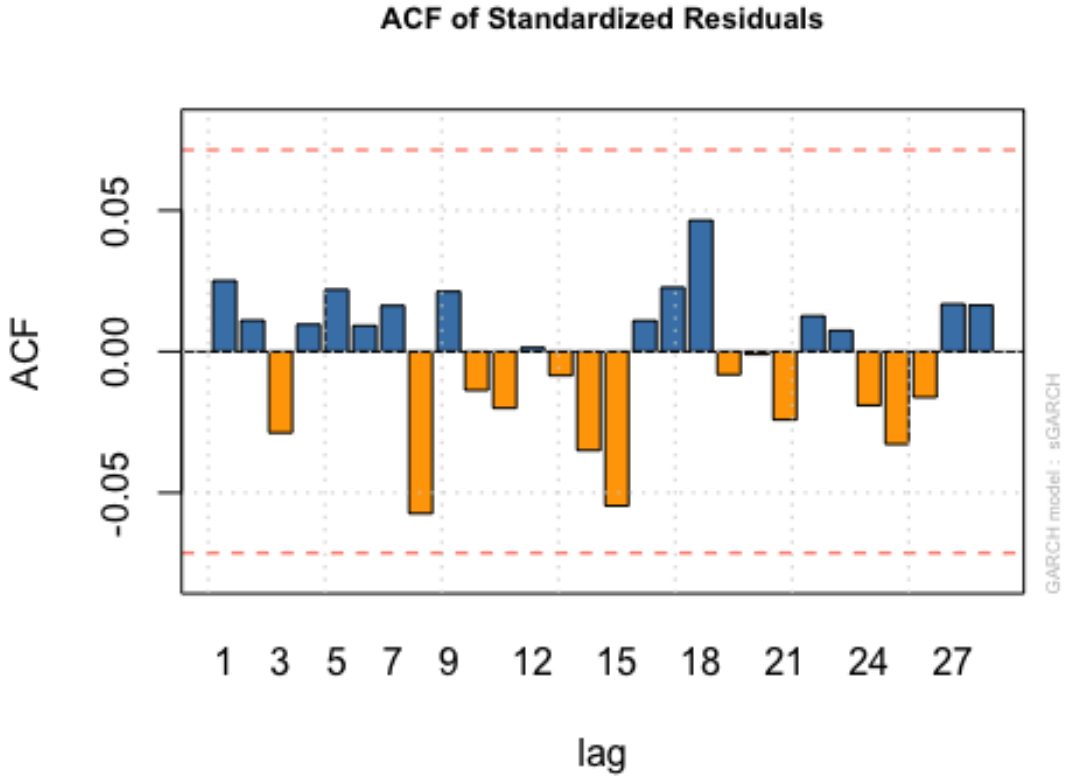

```
plot(forecast, which=3)
```



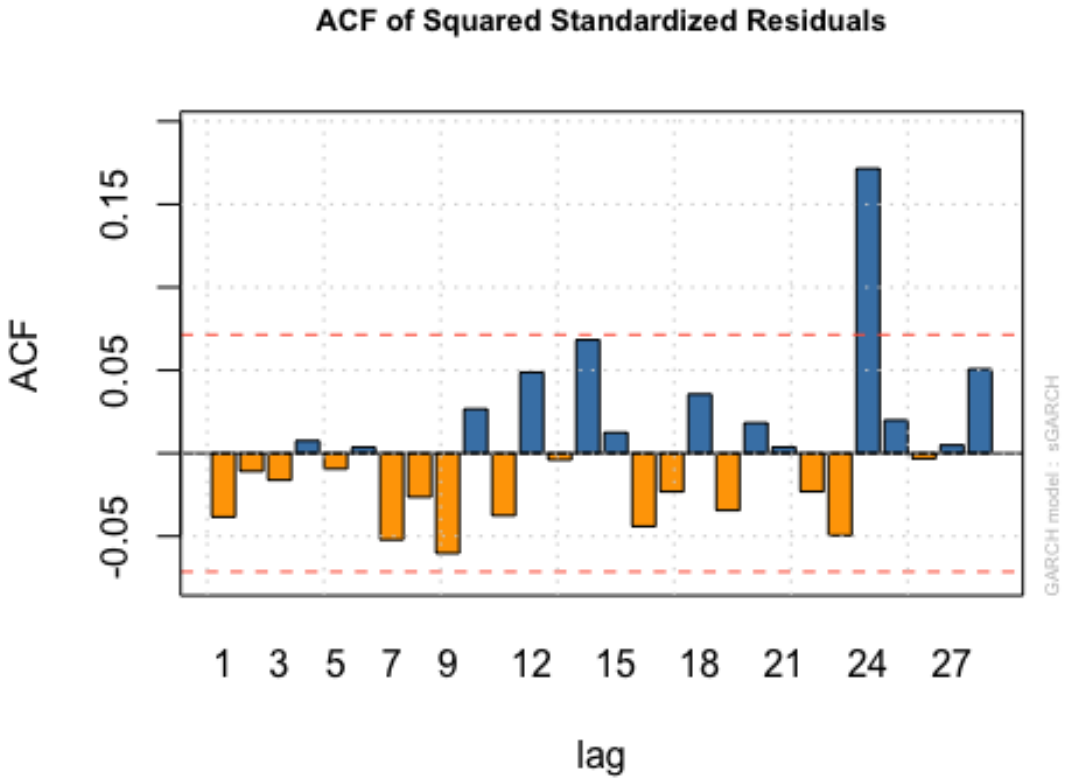

```
#Final Model
spec1 = ugarchspec(
  mean.model = list(armaOrder = c(1,1)),
  variance.model = list(model = 'sGARCH', garchOrder = c(1,1)),
  distribution.model = 'std')
m = ugarchfit(amzn.lrtn, spec = spec1)
m

##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.001933    0.000514   3.7579 0.000171
```

```
## ar1    -0.550034    0.324209  -1.6965 0.089783
## ma1     0.517218    0.332138   1.5572 0.119414
## omega   0.000020    0.000008   2.6055 0.009173
## alpha1  0.238133    0.054161   4.3967 0.000011
## beta1   0.747047    0.047401  15.7601 0.000000
## shape   5.352153    1.111306   4.8161 0.000001
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.001933    0.000520   3.7134 0.000204
## ar1    -0.550034    0.142036  -3.8725 0.000108
## ma1     0.517218    0.151752   3.4083 0.000654
## omega   0.000020    0.000008   2.5521 0.010707
## alpha1  0.238133    0.051457   4.6278 0.000004
## beta1   0.747047    0.052035  14.3566 0.000000
## shape   5.352153    1.045537   5.1190 0.000000
##
## LogLikelihood : 1962.409
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.1937
## Bayes        -5.1507
## Shibata      -5.1938
## Hannan-Quinn -5.1771
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.4764   0.490
## Lag[2*(p+q)+(p+q)-1][5]    1.0278   1.000
## Lag[4*(p+q)+(p+q)-1][9]    1.9886   0.985
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.114  0.2911
## Lag[2*(p+q)+(p+q)-1][5]     1.336  0.7802
## Lag[4*(p+q)+(p+q)-1][9]     2.535  0.8325
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.1972 0.500 2.000  0.6570
## ARCH Lag[5]    0.2712 1.440 1.667  0.9478
## ARCH Lag[7]    1.1630 2.315 1.543  0.8857
```

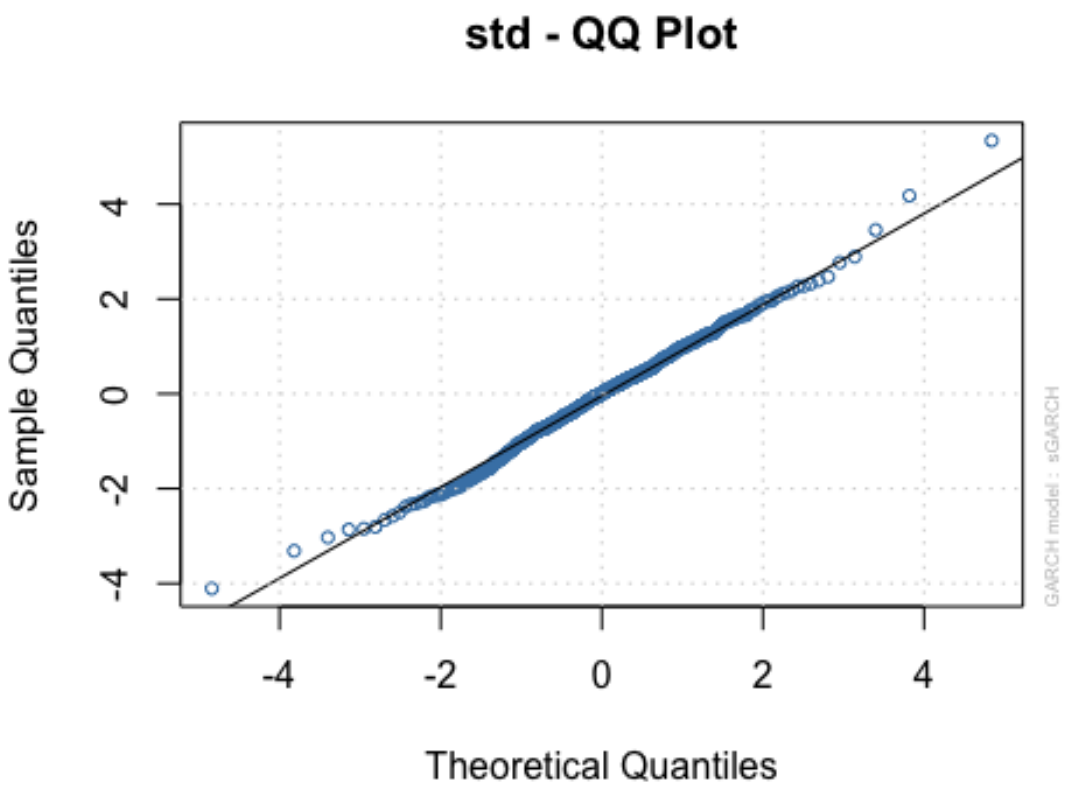
```
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.0874
## Individual Statistics:
## mu     0.43098
## ar1    0.22737
## ma1    0.23100
## omega  0.09871
## alpha1 0.07303
## beta1  0.10167
## shape  0.06351
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.69 1.9 2.35
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias          0.06413 0.9489
## Negative Sign Bias 0.59769 0.5502
## Positive Sign Bias 1.02732 0.3046
## Joint Effect       1.44886 0.6941
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     22.91       0.2413
## 2    30     25.92       0.6295
## 3    40     43.55       0.2840
## 4    50     54.24       0.2816
##
##
## Elapsed time : 0.1702201

plot(m, which = 10)
```



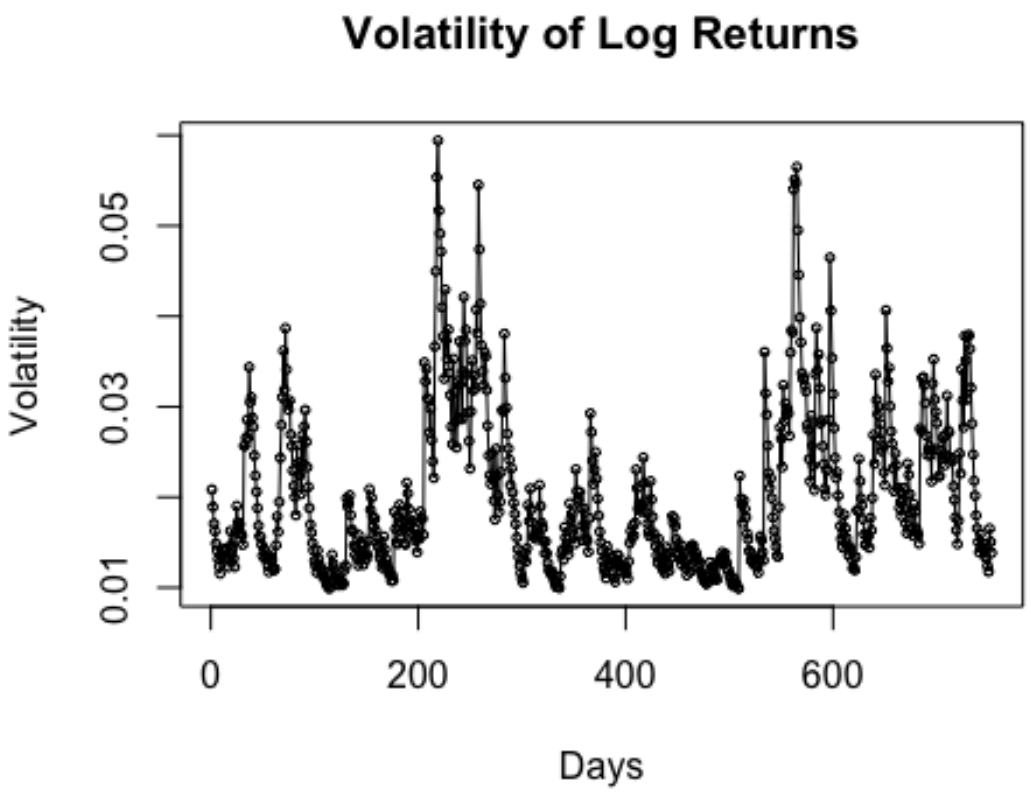

```
plot(m, which = 11)
```




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




```
plot(as.numeric(sigma(m)), main = 'Volatility of Log Returns', type = 'o', yl
ab = 'Volatility', cex = 0.5, xlab = 'Days')
```




```
coef(m)
```

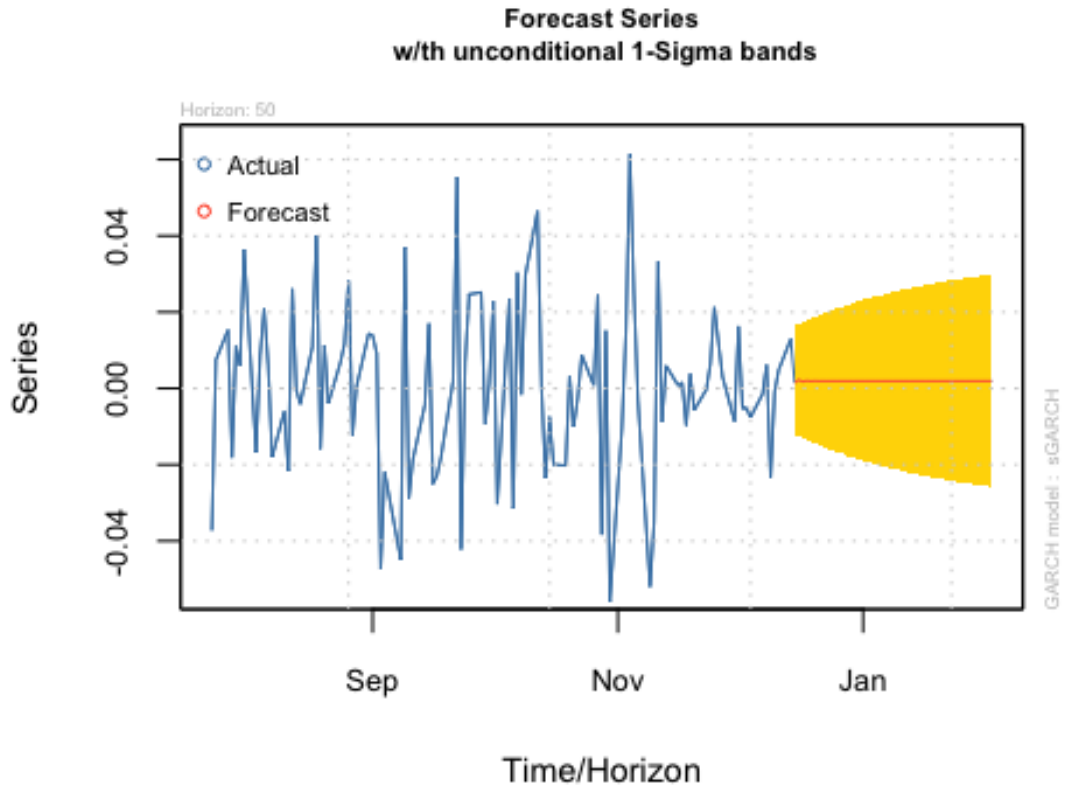
```
##           mu          ar1          ma1        omega       alpha1        be
ta1
##  0.001932756 -0.550034337  0.517217735  0.000019648  0.238132766  0.747046
589
##        shape
##  5.352153204
```

```
forecast = ugarchforecast(m, n.ahead = 50)
forecast
```

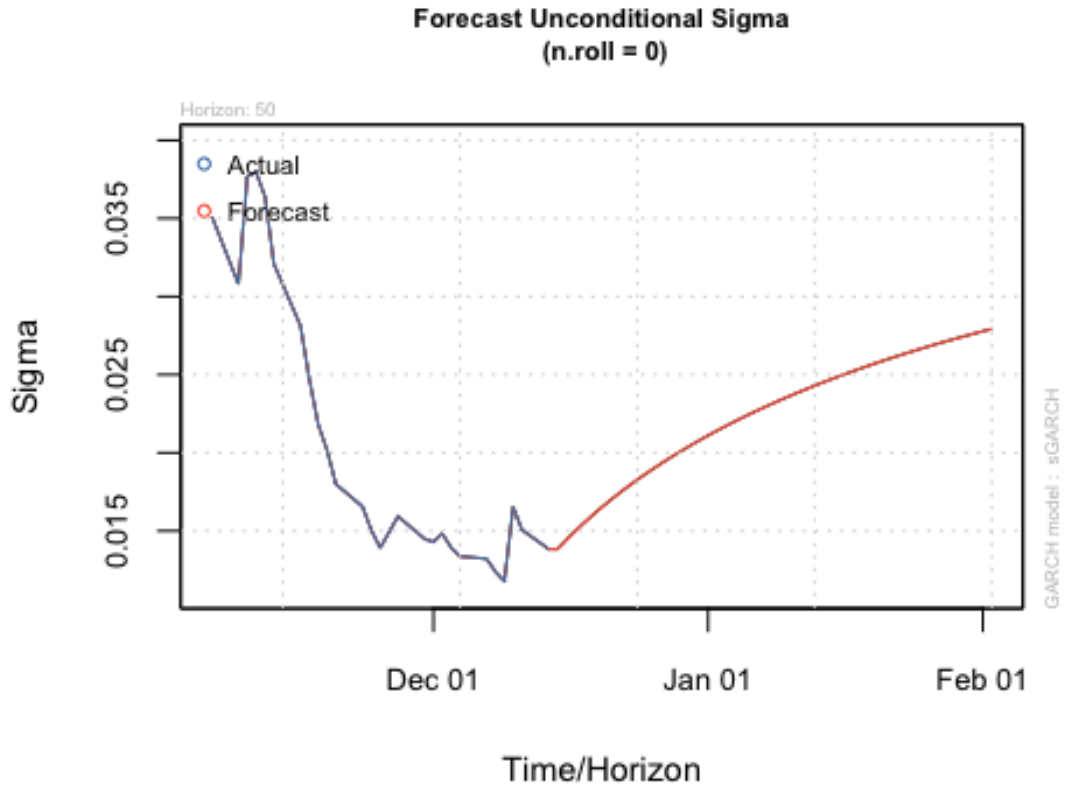
```
##
## *----------------------------------*
## *       GARCH Model Forecast       *
## *----------------------------------*
## Model: sGARCH
## Horizon: 50
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=2020-12-14]:
##        Series   Sigma
## T+1  0.001523 0.01383
```

```
## T+2  0.002158 0.01442
## T+3  0.001809 0.01498
## T+4  0.002001 0.01552
## T+5  0.001895 0.01603
## T+6  0.001953 0.01652
## T+7  0.001921 0.01698
## T+8  0.001939 0.01743
## T+9  0.001929 0.01786
## T+10 0.001935 0.01827
## T+11 0.001932 0.01867
## T+12 0.001933 0.01905
## T+13 0.001932 0.01942
## T+14 0.001933 0.01978
## T+15 0.001933 0.02013
## T+16 0.001933 0.02046
## T+17 0.001933 0.02079
## T+18 0.001933 0.02111
## T+19 0.001933 0.02141
## T+20 0.001933 0.02171
## T+21 0.001933 0.02200
## T+22 0.001933 0.02228
## T+23 0.001933 0.02256
## T+24 0.001933 0.02282
## T+25 0.001933 0.02308
## T+26 0.001933 0.02334
## T+27 0.001933 0.02358
## T+28 0.001933 0.02382
## T+29 0.001933 0.02406
## T+30 0.001933 0.02429
## T+31 0.001933 0.02451
## T+32 0.001933 0.02473
## T+33 0.001933 0.02494
## T+34 0.001933 0.02515
## T+35 0.001933 0.02535
## T+36 0.001933 0.02555
## T+37 0.001933 0.02575
## T+38 0.001933 0.02594
## T+39 0.001933 0.02612
## T+40 0.001933 0.02630
## T+41 0.001933 0.02648
## T+42 0.001933 0.02666
## T+43 0.001933 0.02683
## T+44 0.001933 0.02699
## T+45 0.001933 0.02716
## T+46 0.001933 0.02732
## T+47 0.001933 0.02747
## T+48 0.001933 0.02763
## T+49 0.001933 0.02778
## T+50 0.001933 0.02793
```

```
plot(forecast, which=1, main = "Forecast")
```




```
plot(forecast, which=3, main = "Forecast")
```


Forecast Unconditional Sigma
(n.roll = 0)
Horizon: 50
Actual
Forecast
Sigma
0.015
0.025
0.035
Dec 01
Jan 01
Feb 01
Time/Horizon
GARCH model : sGARCH