GARCH Model

Armand Tossou

9/7/2021

Introduction

Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) models are good for times series data that are very volatile. This sample project showcases how various GARCH models can be fitted and how predictions can be made. Apple stock prices are used for the application.

Figure 1: apple stock returns

Apple daily prices

We'll use the `getSymbols()` function from R's `quantmod` package to retrieve stock data for Apple.

```
#install.packages("quantmod")
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
```

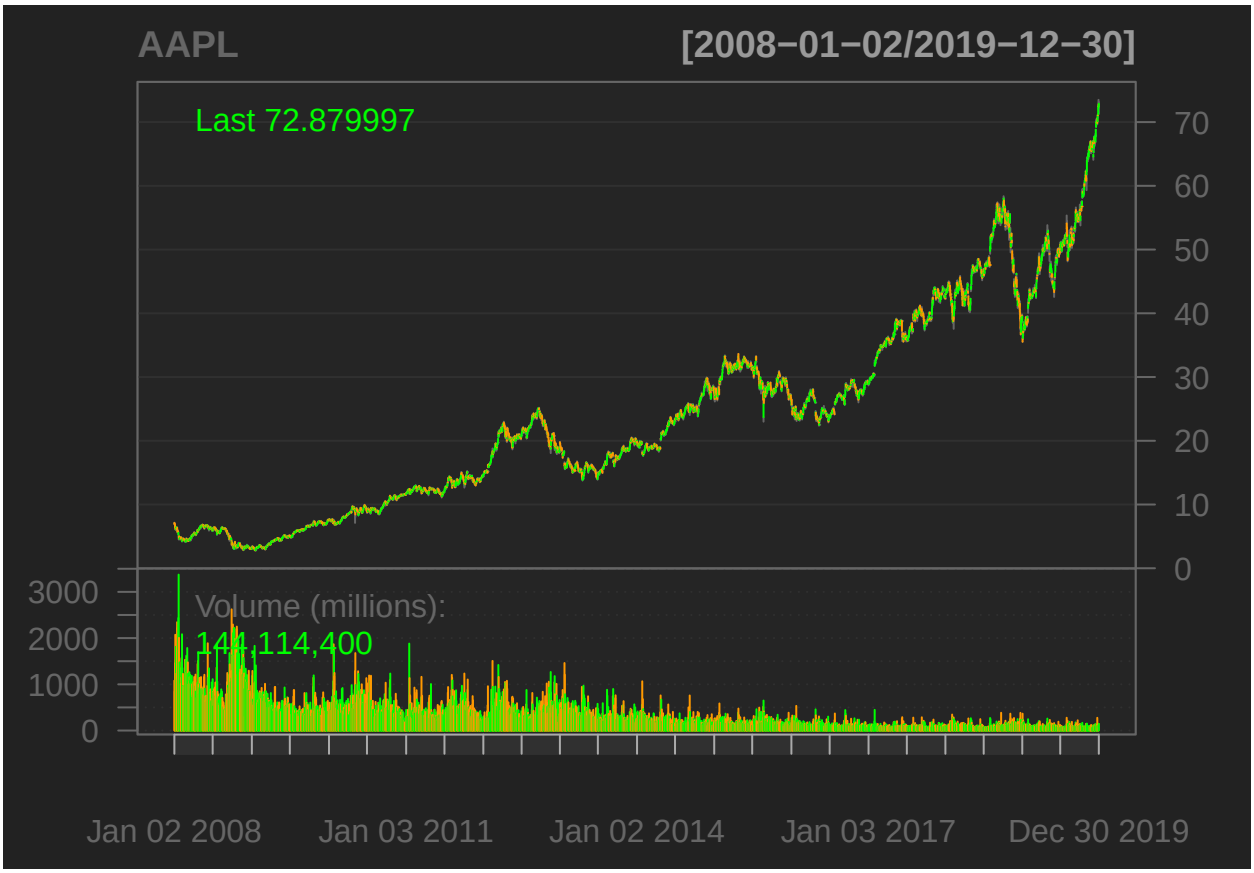
```
# retrive Apple stock data
getSymbols("AAPL",from = "2008-01-01",to = "2019-12-31")
```

```
## '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] "AAPL"
```

Let's plot the series.

```
chartSeries(AAPL)
```



Daily returns

We'll use the `CalculateReturns()` function from R's `PerformanceAnalytics` package.

```
# install and call library
install.packages("PerformanceAnalytics")
```

```
## Installing package into 'C:/Users/adtos/Documents/R/win-library/4.1'
## (as 'lib' is unspecified)
```

```
## Error in contrib.url(repos, "source"): trying to use CRAN without setting a mirror
```

```
library(PerformanceAnalytics)
```

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

```r
# get daily returns series
return <- CalculateReturns(AAPL$AAPL.Close)

# preview the data
head(return)
```

```
##               AAPL.Close
## 2008-01-02            NA
## 2008-01-03  0.0004620201
## 2008-01-04 -0.0763351531
## 2008-01-07 -0.0133851044
## 2008-01-08 -0.0359717390
## 2008-01-09  0.0475913376
```

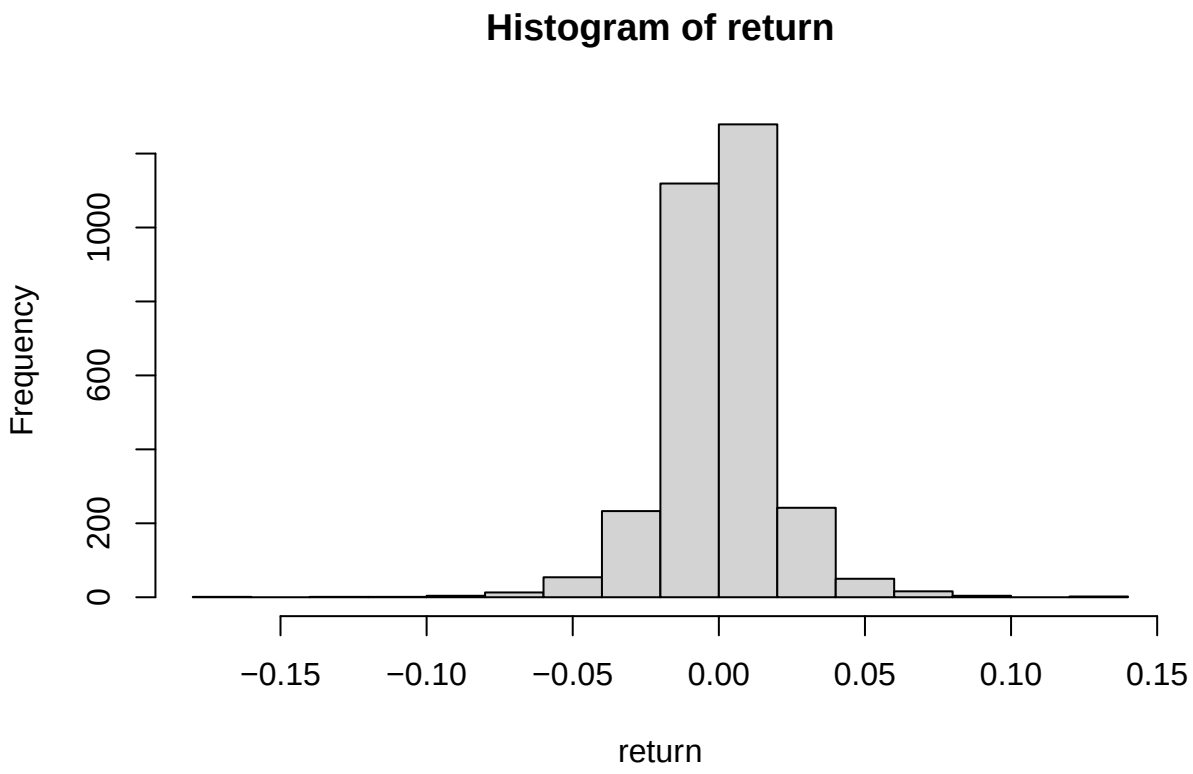
Let's drop the first observation from the dataset:

```r
return <- return[-1]

head(return)
```

```
##               AAPL.Close
## 2008-01-03  0.0004620201
## 2008-01-04 -0.0763351531
## 2008-01-07 -0.0133851044
## 2008-01-08 -0.0359717390
## 2008-01-09  0.0475913376
## 2008-01-10 -0.0076923521
```

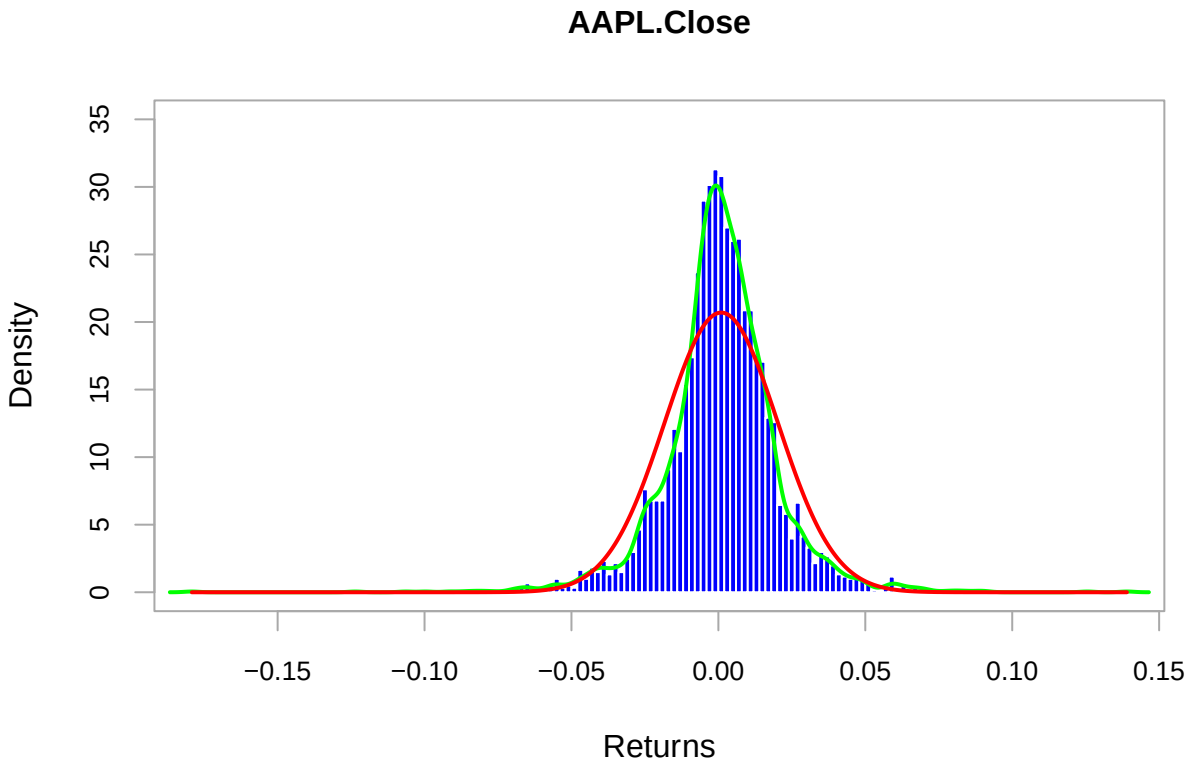
Create a histogram of daily returns:

```r
hist(return)
```



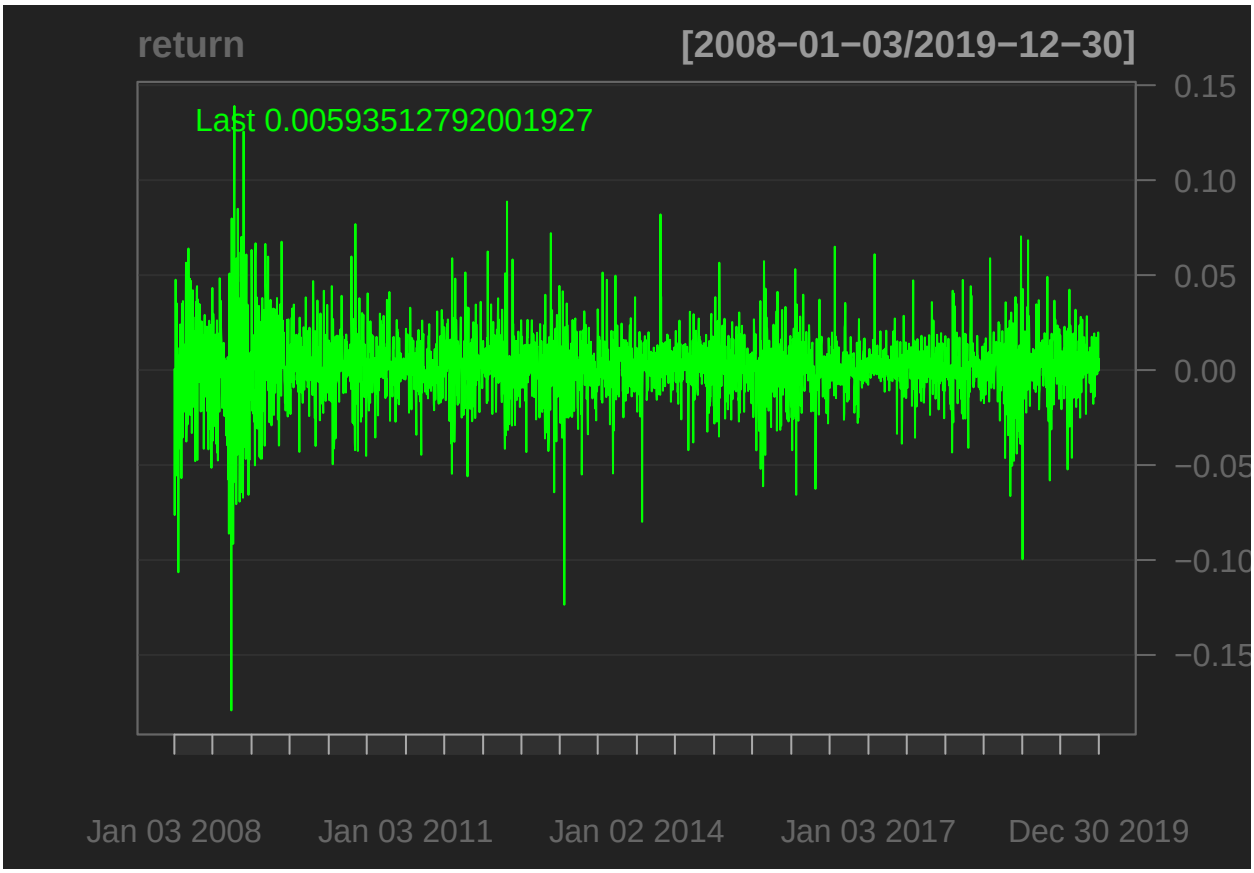
A more detailed histogram of daily returns:

```
chart.Histogram(return,
                methods = c('add.density', 'add.normal'),
                colorset = c('blue', 'green', 'red'))
```



Plot the daily returns series:

```
chartSeries(return)
```



Annualized volatility

```
chart.RollingPerformance(R = return["2008::2019"],
                         width = 252,
                         FUN = "sd.annualized",
                         scale = 252,
                         main = "Apple's yearly rolling volatility")
```

sGARCH model with contant mean

We'll use the `ugarchspec()` function from R's 'rugarch" package.This method is used for creating a univariate GARCH specification object prior to fitting.

```
# install and load package
install.packages("rugarch")
```

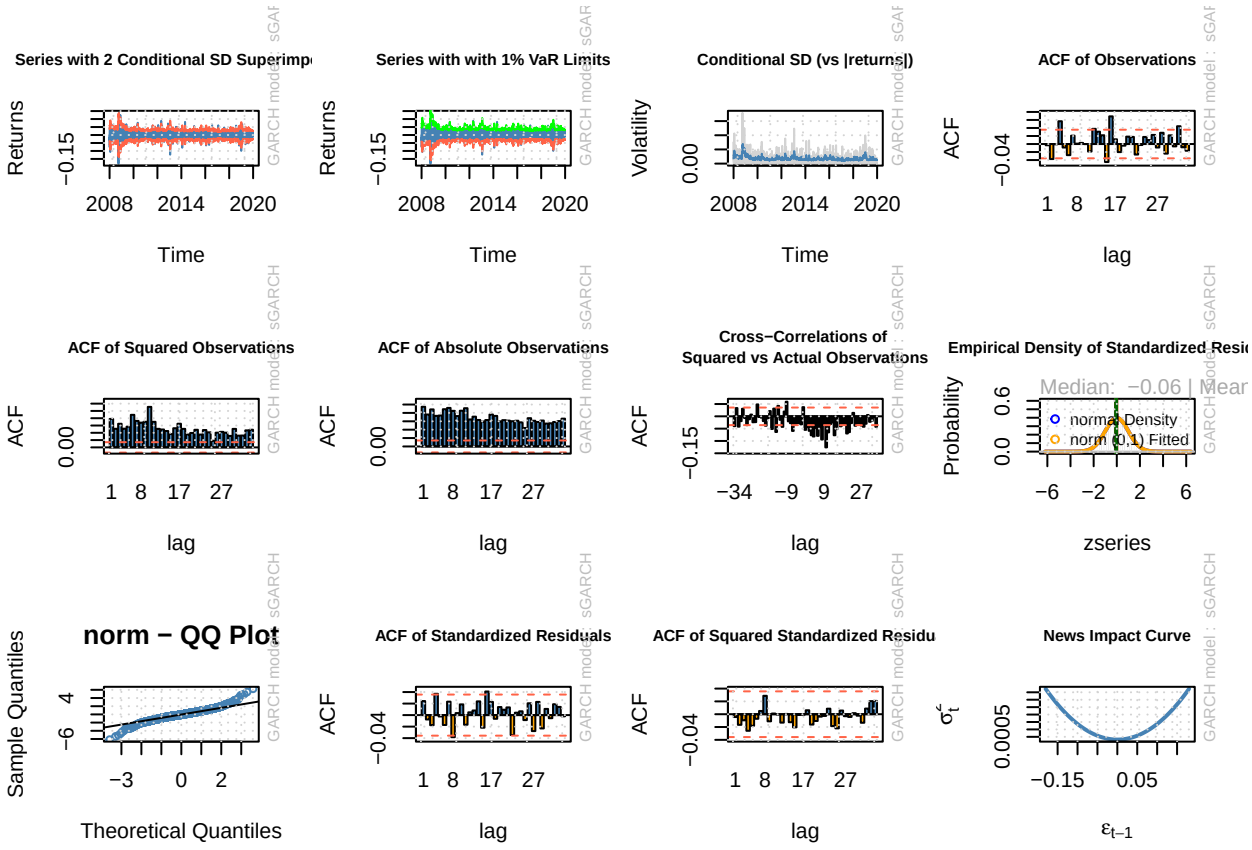
```
## Installing package into 'C:/Users/adtos/Documents/R/win-library/4.1'
## (as 'lib' is unspecified)

## Error in contrib.url(repos, "source"): trying to use CRAN without setting a mirror
```

```
library(rugarch)
```

```
## Loading required package: parallel

##
## Attaching package: 'rugarch'

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

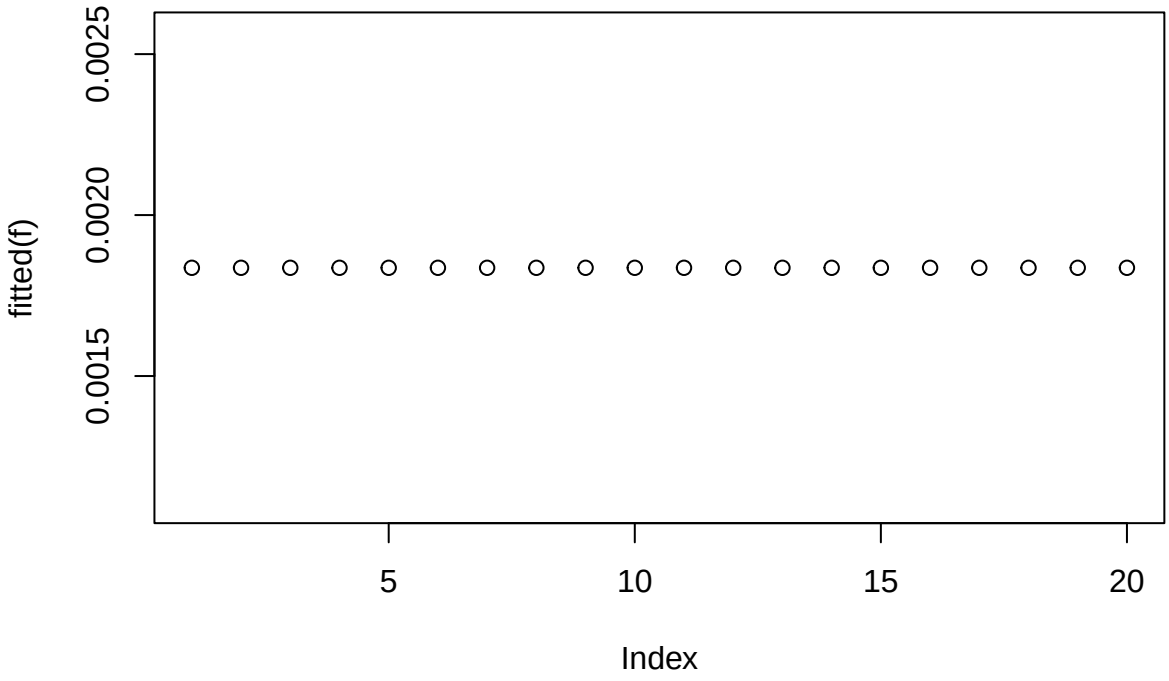
```
# create the plot
s <- ugarchspec(mean.model = list(armaOrder = c(0,0)),
                variance.model = list(model = "sGARCH"),
                distribution.model = 'norm')

m <- ugarchfit(data = return, spec = s)

plot(m, which = 'all')
```

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

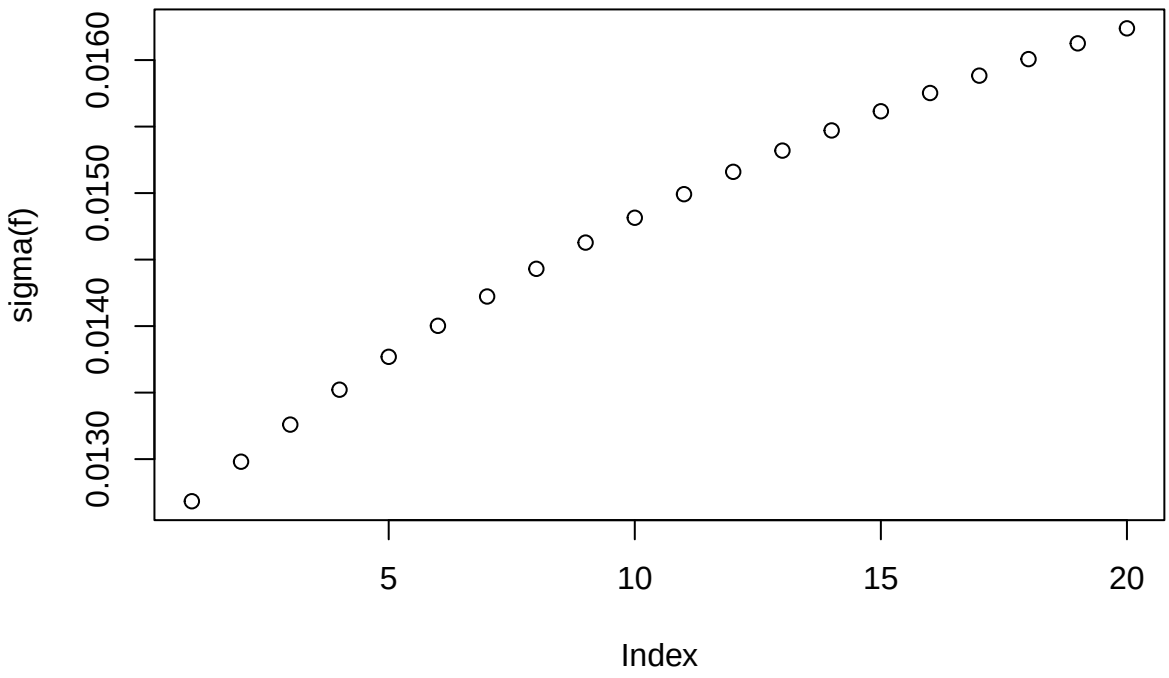


Fitted series

```
f <- ugarchforecast(fitORspec = m, n.ahead = 20)

plot(fitted(f))
```

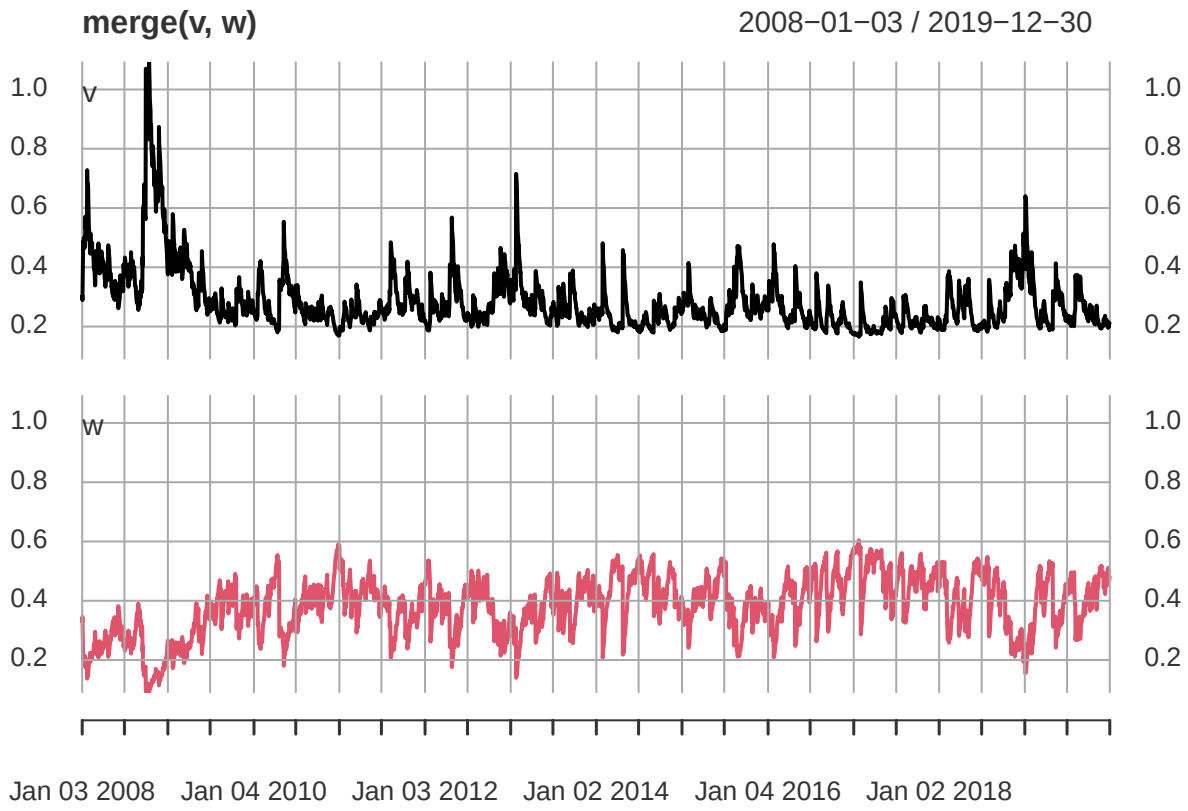



```
plot(sigma(f))
```



Application example - portfolio allocation

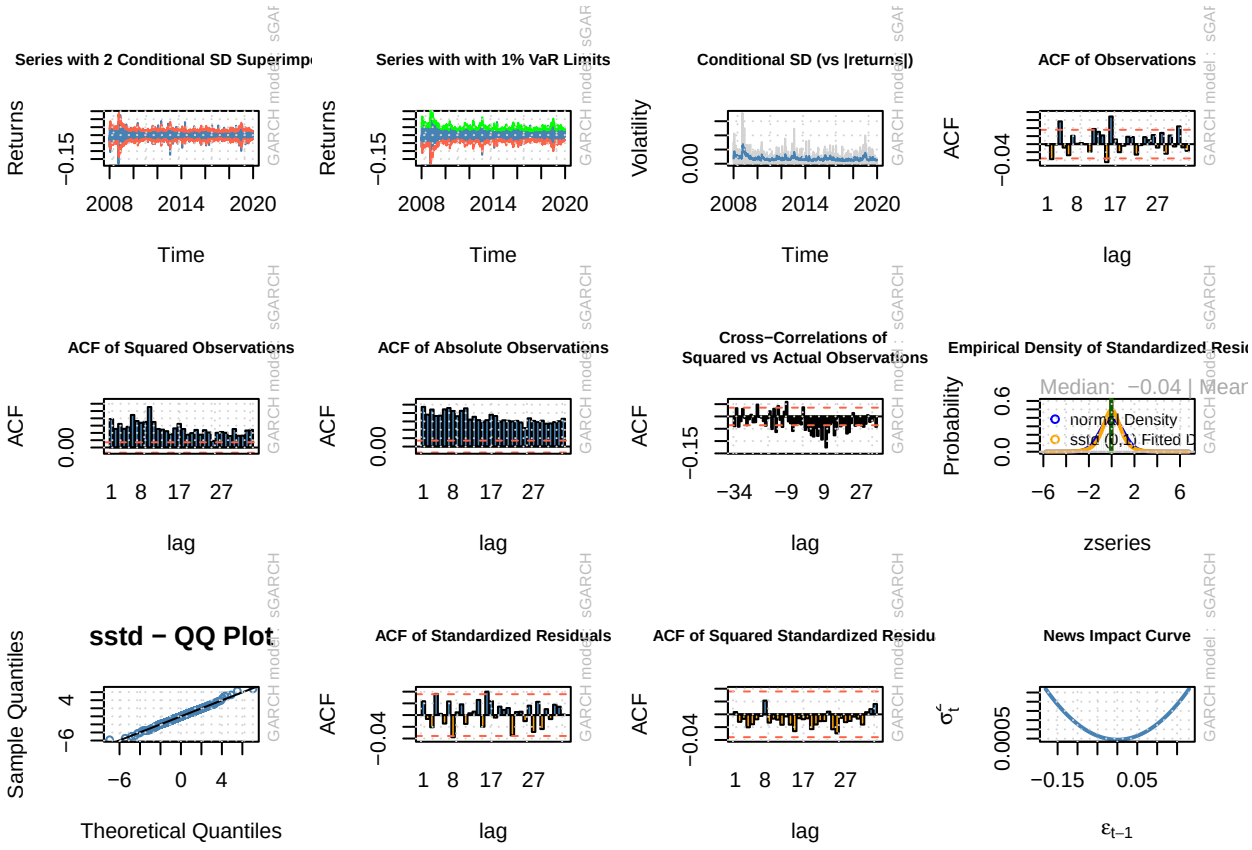
```
v <- sqrt(252) * sigma(m)

w <- 0.1/v

plot(merge(v, w), multi.panel = T)
```



GARCH with sstd

```
s <- ugarchspec(mean.model = list(armaOrder = c(0,0)),
                variance.model = list(model = "sGARCH"),
                distribution.model = 'sstd')

m <- ugarchfit(data = return, spec = s)

plot(m, which = 'all')
```

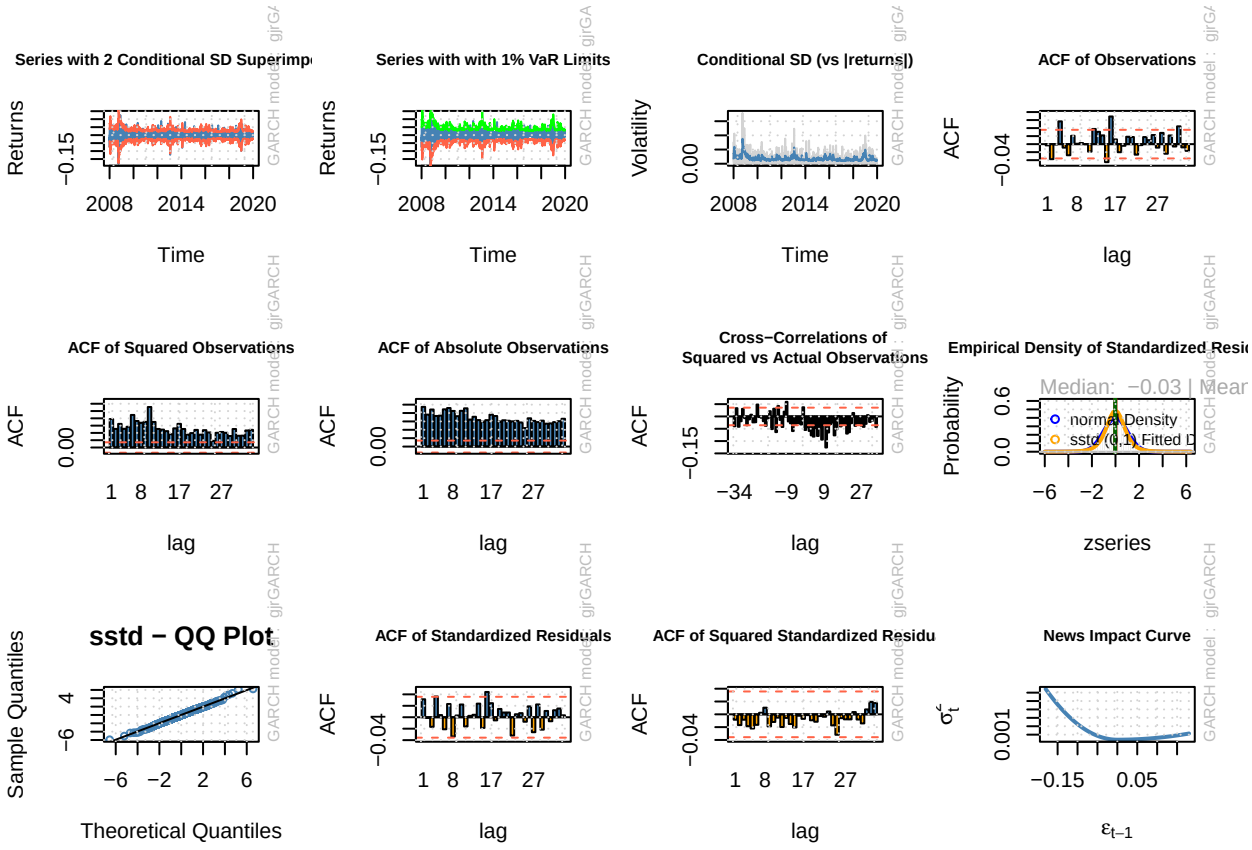
```
##
## please wait...calculating quantiles...
```



GJR-GARCH

```
s <- ugarchspec(mean.model = list(armaOrder = c(0,0)),
                variance.model = list(model = "gjrGARCH"),
                distribution.model = 'sstd')

m <- ugarchfit(data = return, spec = s)

plot(m, which = 'all')
```

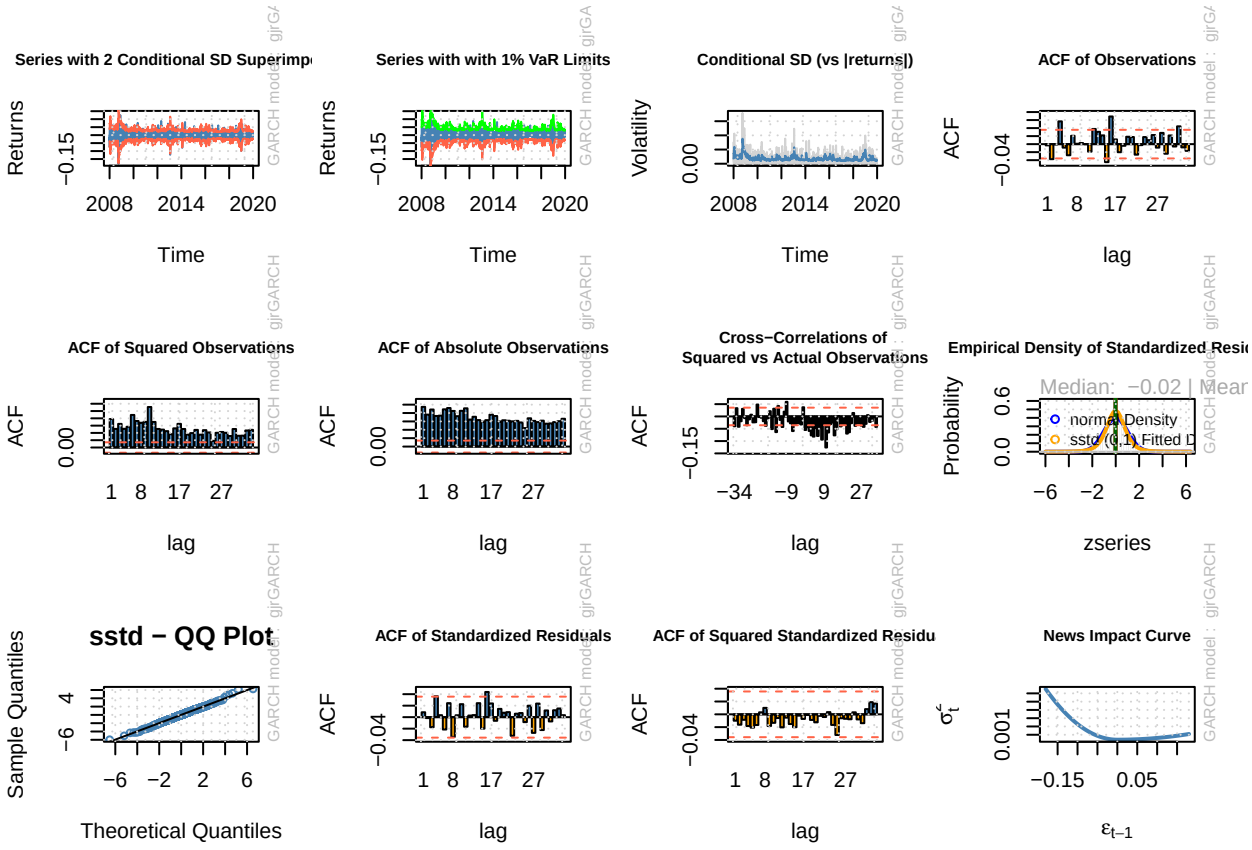
```
##
## please wait...calculating quantiles...
```



AR(1) GJR-GARCH

```
s <- ugarchspec(mean.model = list(armaOrder = c(1,0)),
                variance.model = list(model = "gjrGARCH"),
                distribution.model = 'sstd')

m <- ugarchfit(data = return, spec = s)

plot(m, which = 'all')
```

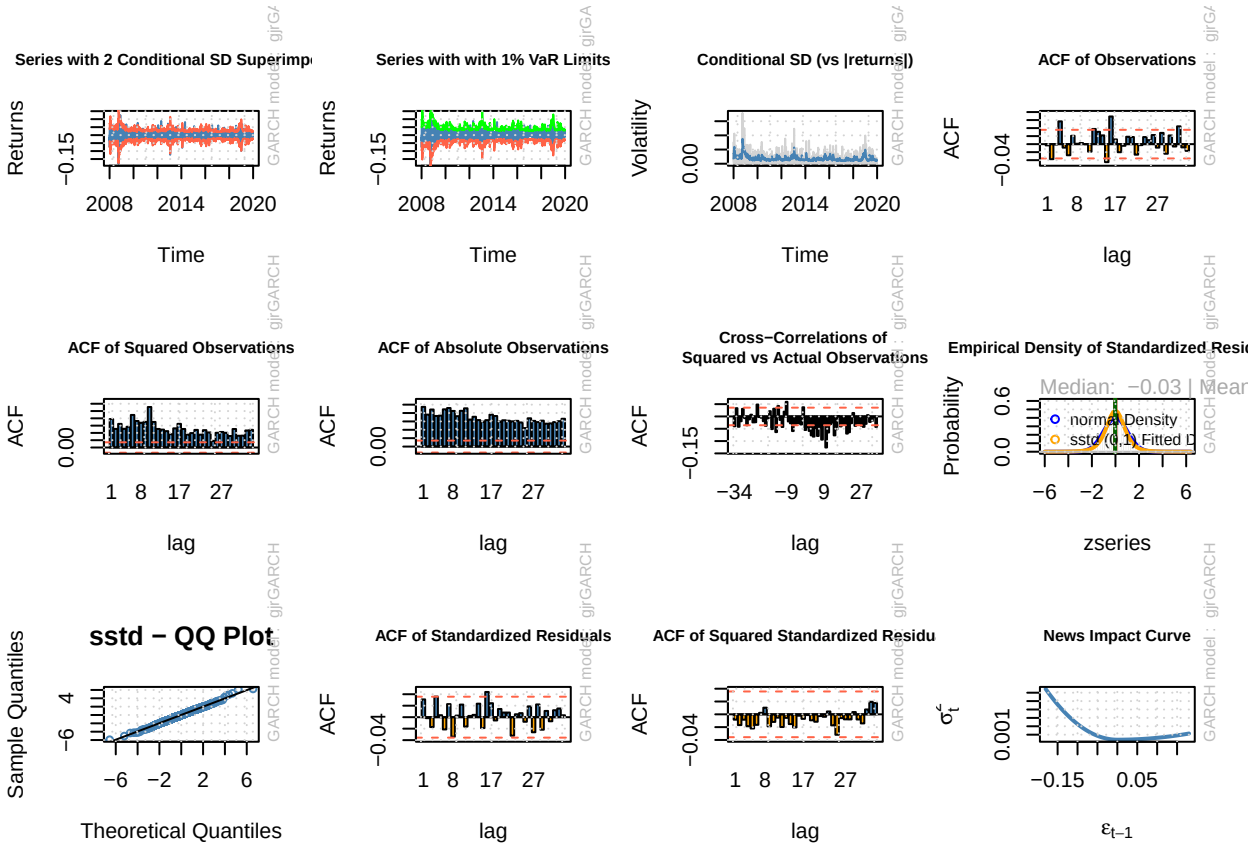
```
##
## please wait...calculating quantiles...
```



GJR-GARCH in mean

```
s <- ugarchspec(mean.model = list(armaOrder = c(0,0),
                                  archm =T,
                                  archpow = 2),
                variance.model = list(model = "gjrGARCH"),
                distribution.model = 'sstd')

m <- ugarchfit(data = return, spec = s)

plot(m, which = 'all')
```

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

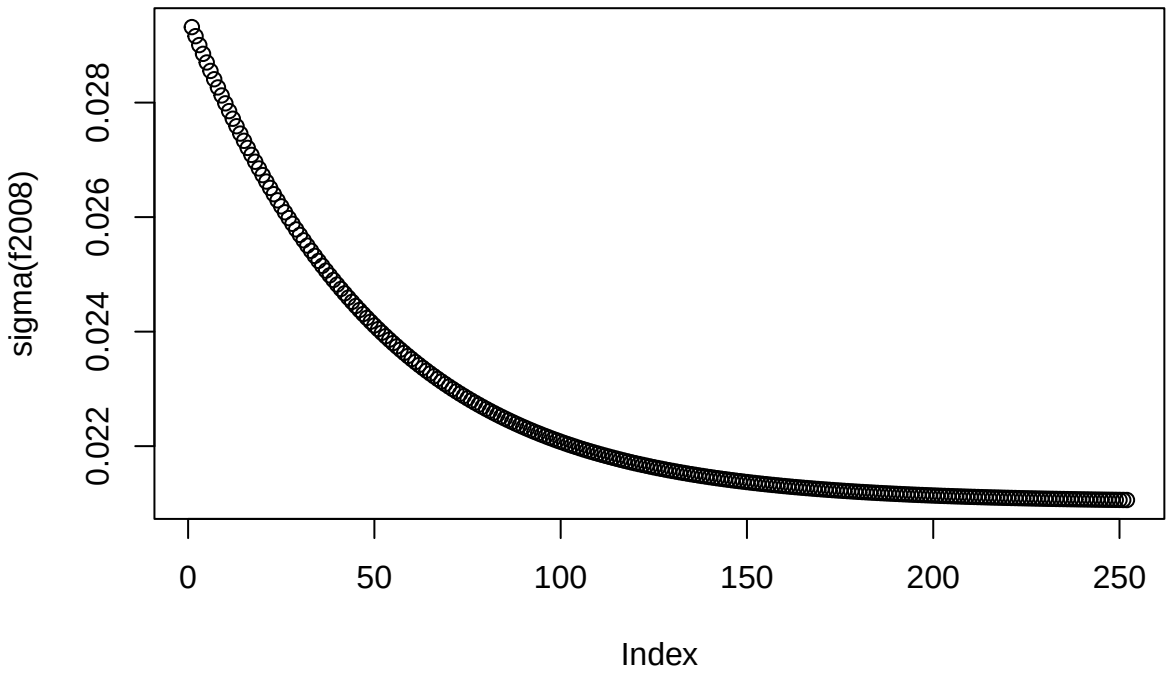


Simulation

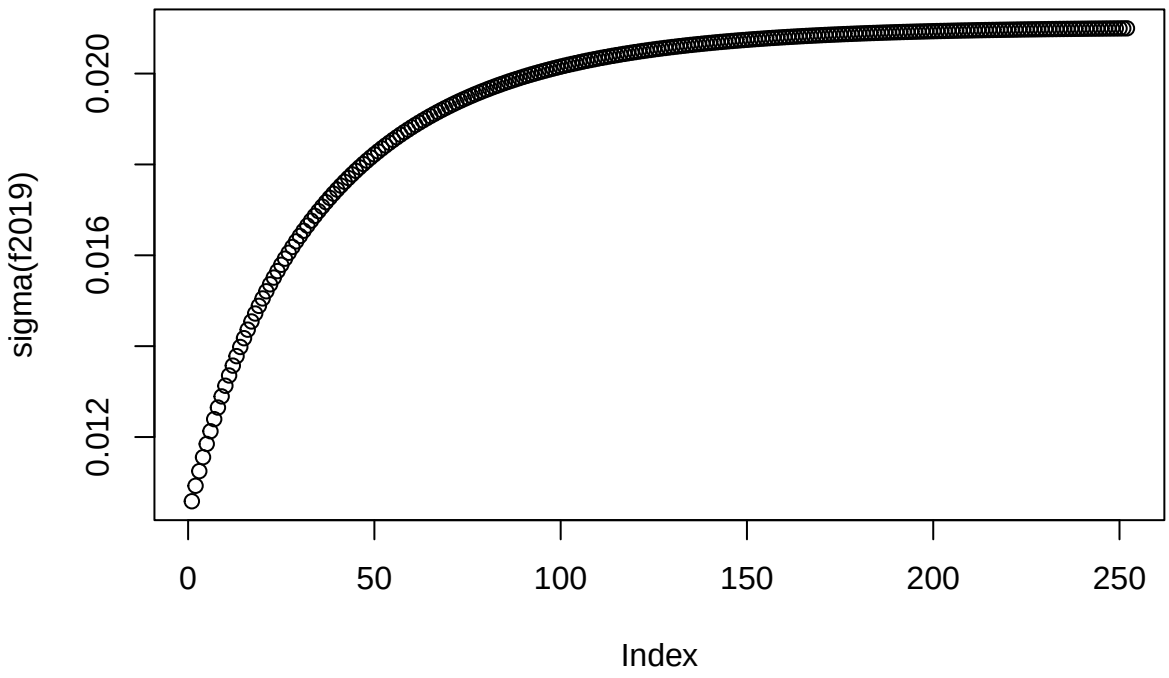
```
s <- ugarchspec(mean.model = list(armaOrder = c(0,0)),
                variance.model = list(model = "gjrGARCH"),
                distribution.model = 'sstd')

m <- ugarchfit(data = return, spec = s)

sfinal <- s

setfixed(sfinal) <- as.list(coef(m))

f2008 <- ugarchforecast(data = return["/2008-12"],
                        fitORspec = sfinal,
                        n.ahead = 252)

f2019 <- ugarchforecast(data = return["/2019-12"],
                        fitORspec = sfinal,
                        n.ahead = 252)
```

Plots

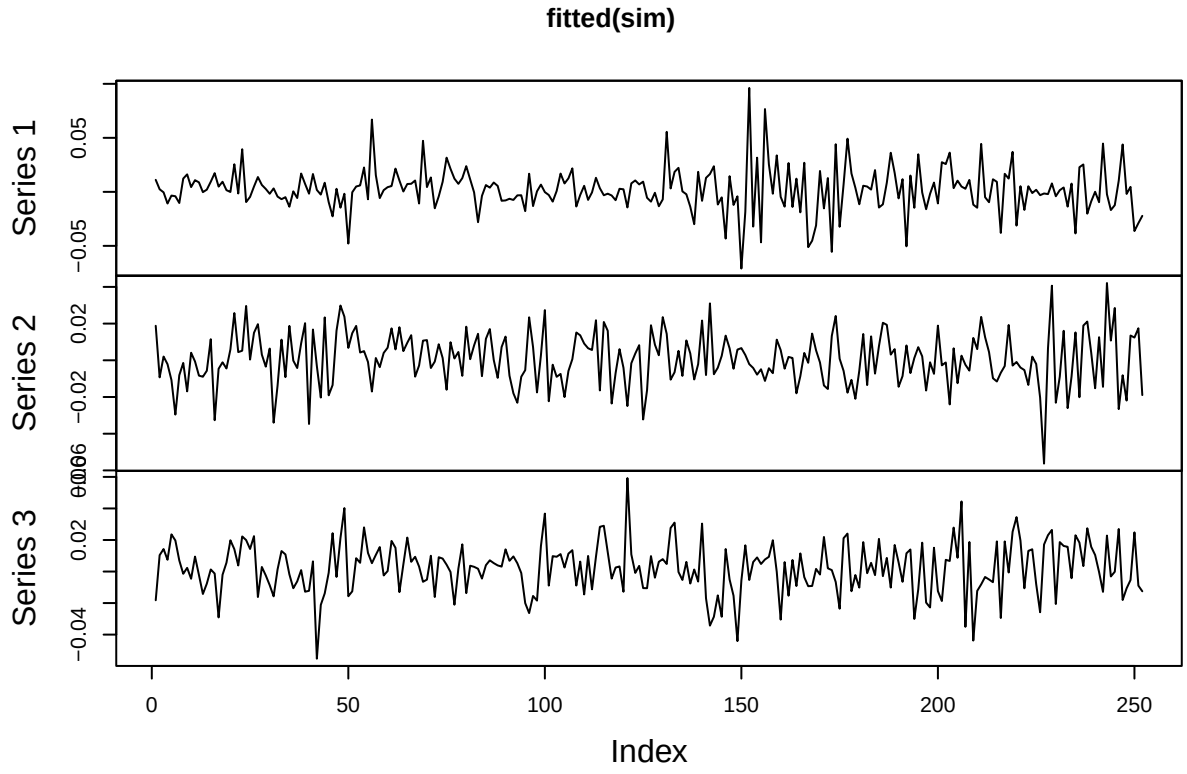
```
par(mfrow = c(1,1))

plot(sigma(f2008))
```



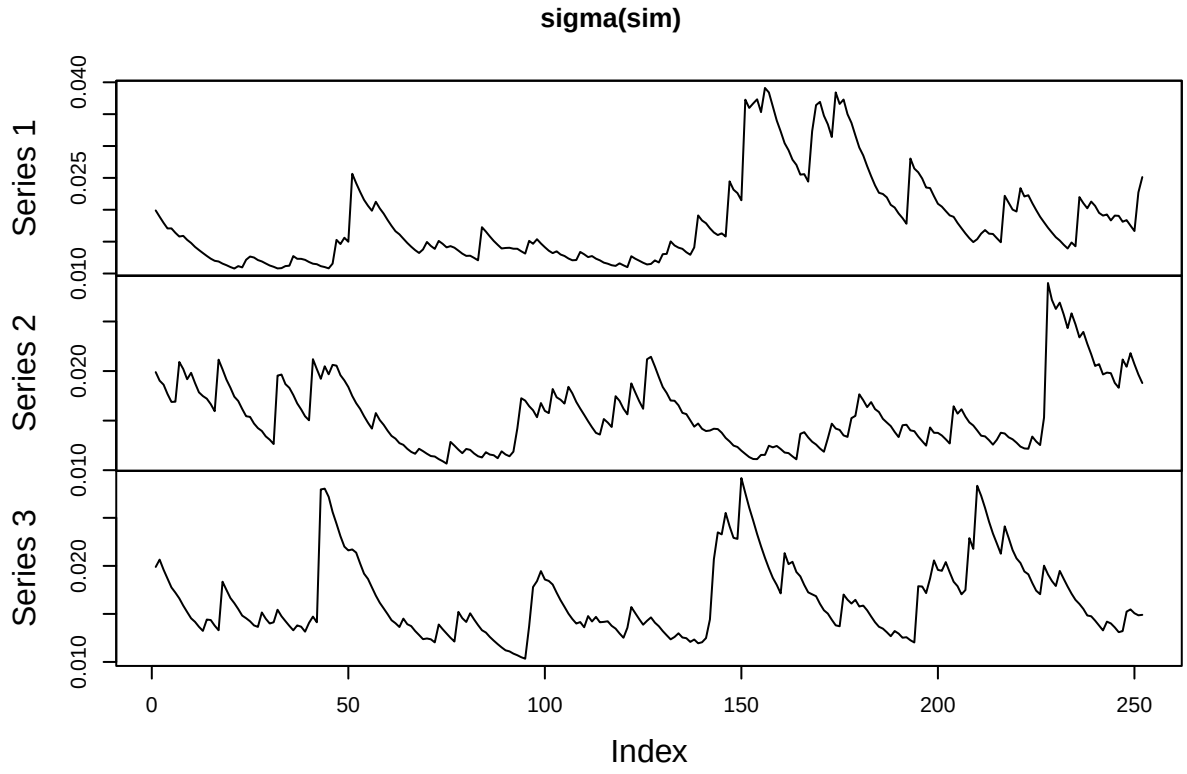

```
plot(sigma(f2019))
```



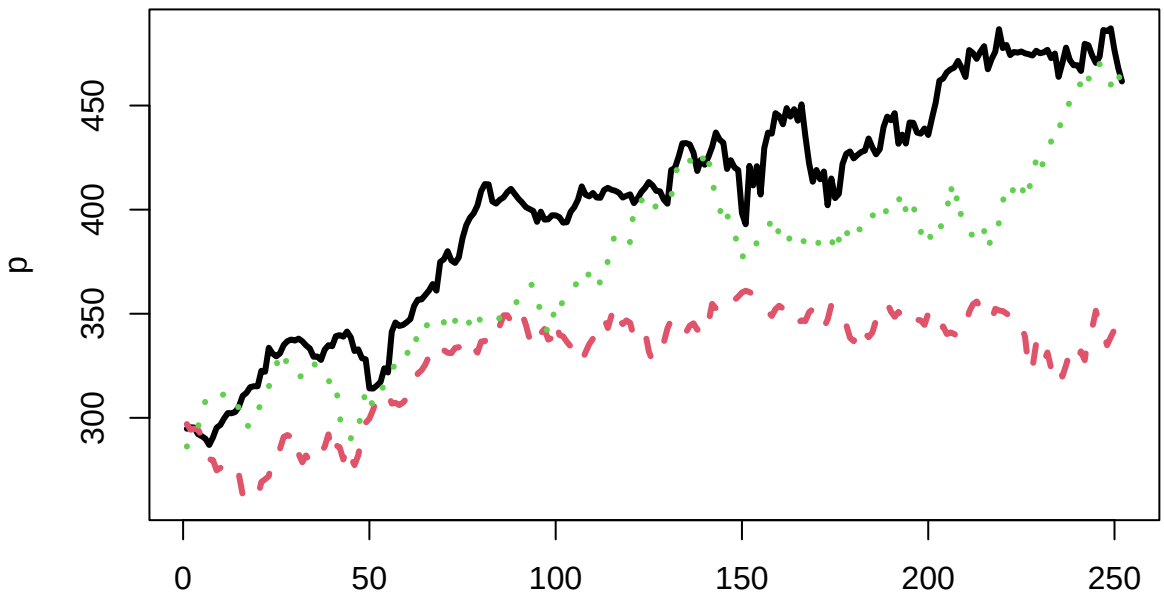

```
sim <- ugarchpath(spec = sfinal,
                  m.sim = 3,
                  n.sim = 1*252,
                  rseed = 123)

plot.zoo(fitted(sim))
```




```
plot.zoo(sigma(sim))
```




```
p <- 291.52*apply(fitted(sim), 2, 'cumsum') + 291.52

matplot(p, type = "l", lwd = 3)
```



The End!

Reference:

https://hounnou-machine-blog.netlify.app/post/project-5/