handout (complete version)

Zhengkai Fu

12/07/2020

Load Library

```
library(tidyverse)  # good old tidyverse, just in case
```

```
## Warning: package 'tidyverse' was built under R version 3.6.3
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
## Warning: package 'tibble' was built under R version 3.6.3
```

```
## Warning: package 'tidyr' was built under R version 3.6.3
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
## Warning: package 'forcats' was built under R version 3.6.3
```

```
library(quantmod)   # get information about stocks
```

```
## Warning: package 'quantmod' was built under R version 3.6.3
```

1. Get stocks with "quantmod"

```
# choose 2 stocks and 1 index
# example: apple, walmart, and SP500

# APPLE
apple <- getSymbols('AAPL', src = 'yahoo', from = '2010-01-01')
```

```
## '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.
```

```r
# APPLE
walmart <- getSymbols("WMT", src = 'yahoo', from = '2010-01-01')

# SP500
sp <- getSymbols('^GSPC', src = 'yahoo', from = '2010-01-01')
# index usually has '^' before its code name.
```

quantmod retrieves stock information based on their code name, and you can easily look up those code name with yahoo finance, which is also the source of the information we grabbed ("src = 'yahoo' ").

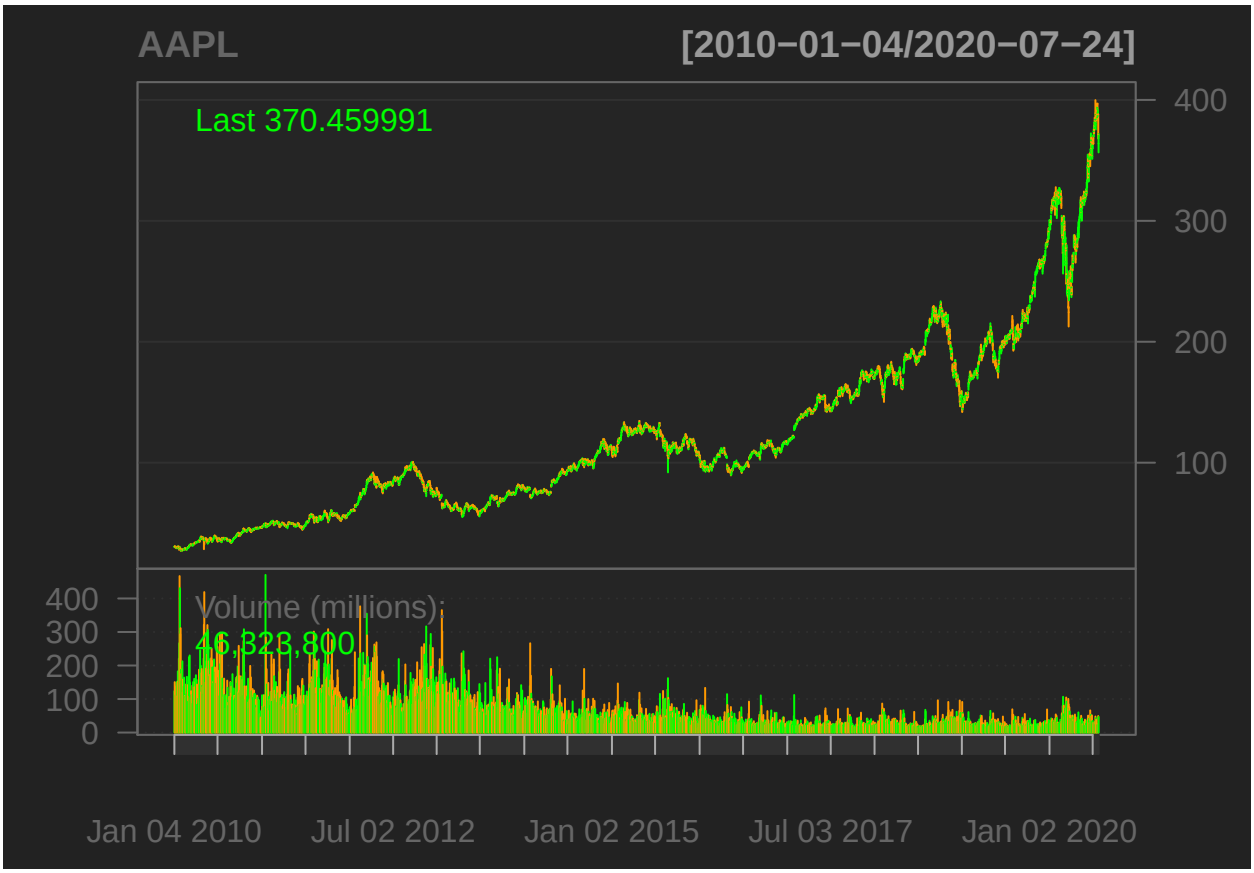
It was possible to set information source as google finance but it has been discontinued.

2. Various way to plot

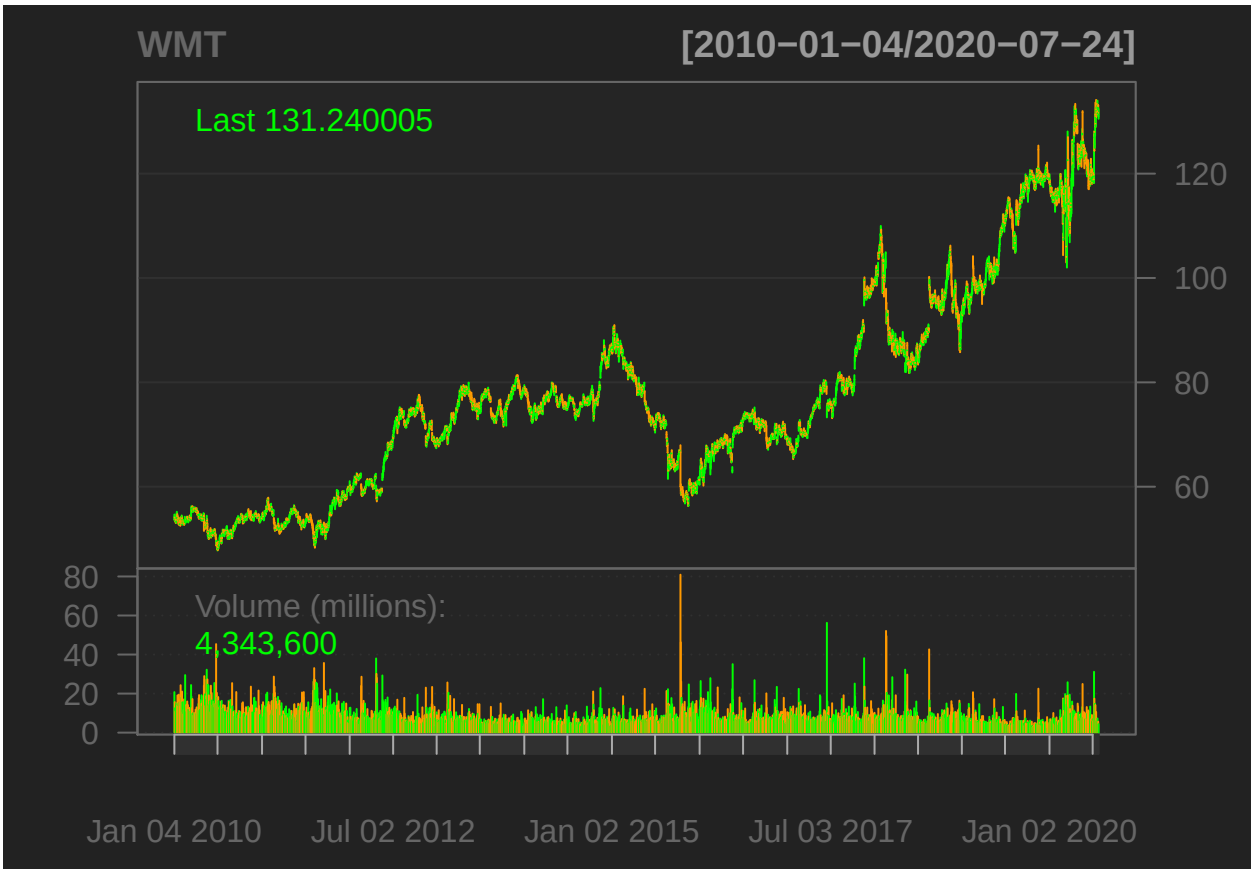
An important detail is that quantmod does not store the stock data on the variable you assigned. All the data would create a new variable with the variable name being the stock code used above, and the variable name you used to trigger the function is merely a character variable.

Quantmod

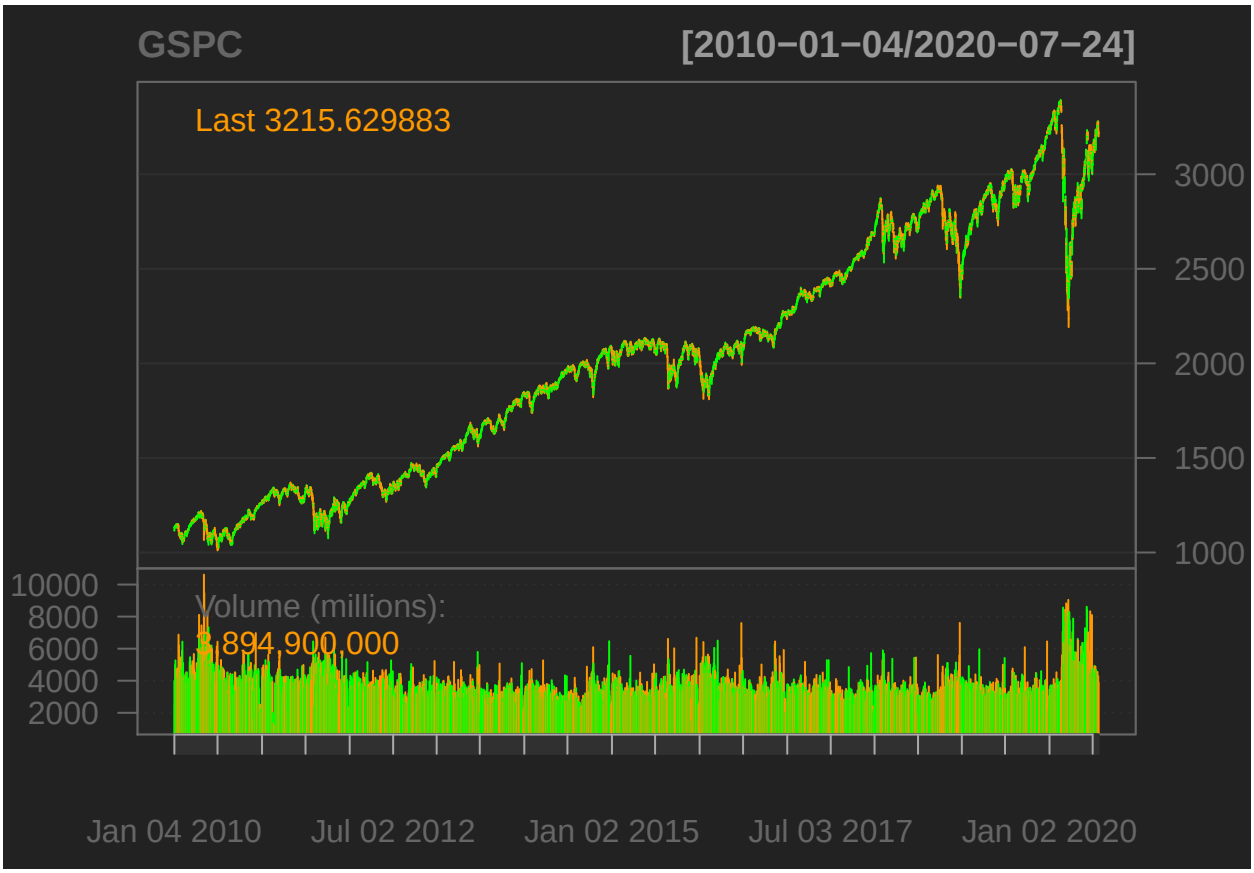
```r
barChart(AAPL)
```



```
barChart(WMT)
```




```
barChart(GSPC)
```



The coolest looking option. It shows up with green and down with red, and the upper portion is the price of the stock while the lower portion shows the trading volume for each day.

The biggest disadvantage is that the chart is not a ggplot or base R plot, and I haven't found any ways to customize this.

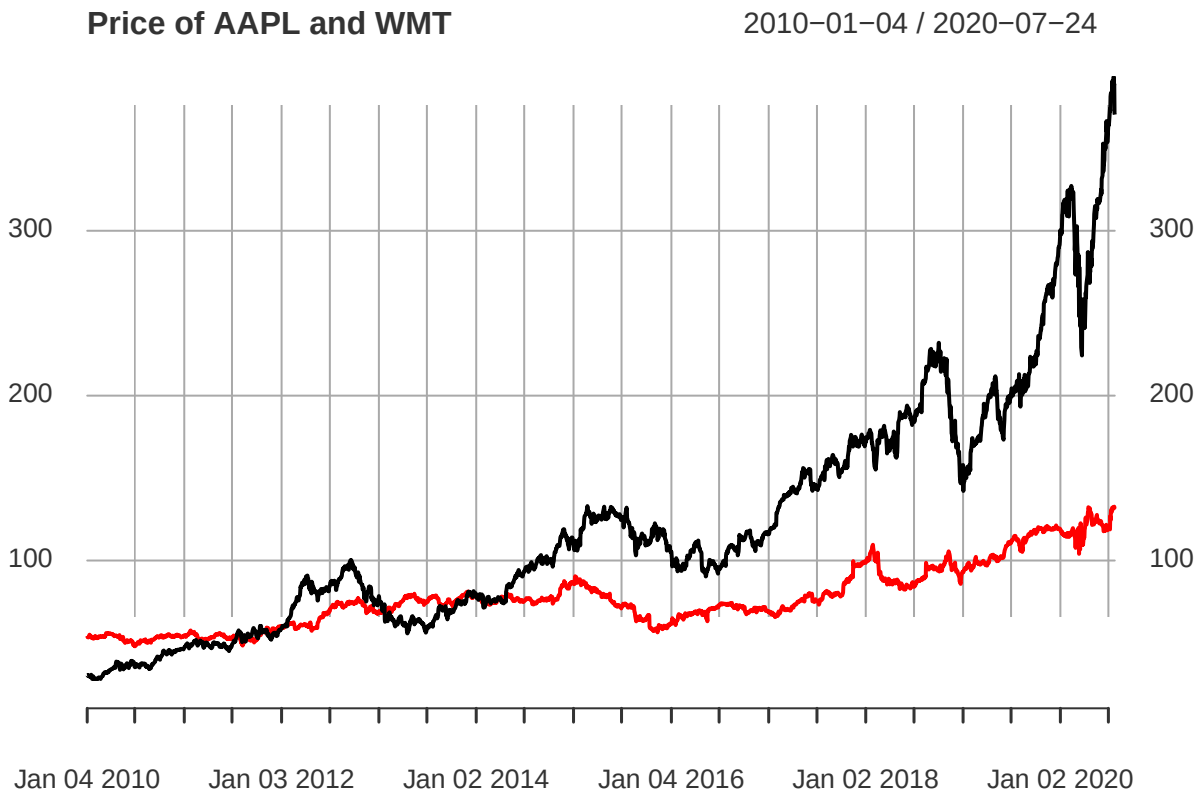
XTS

```
# extract close price for each stock
apple_close <- AAPL$AAPL.Close
walmart_close <- WMT$WMT.Close
sp_close <- GSPC$GSPC.Close

# plot with xts
plot.xts(apple_close, grid.col = 'white', yaxis.right = FALSE,
         main = "Price of Apple stock", )
```

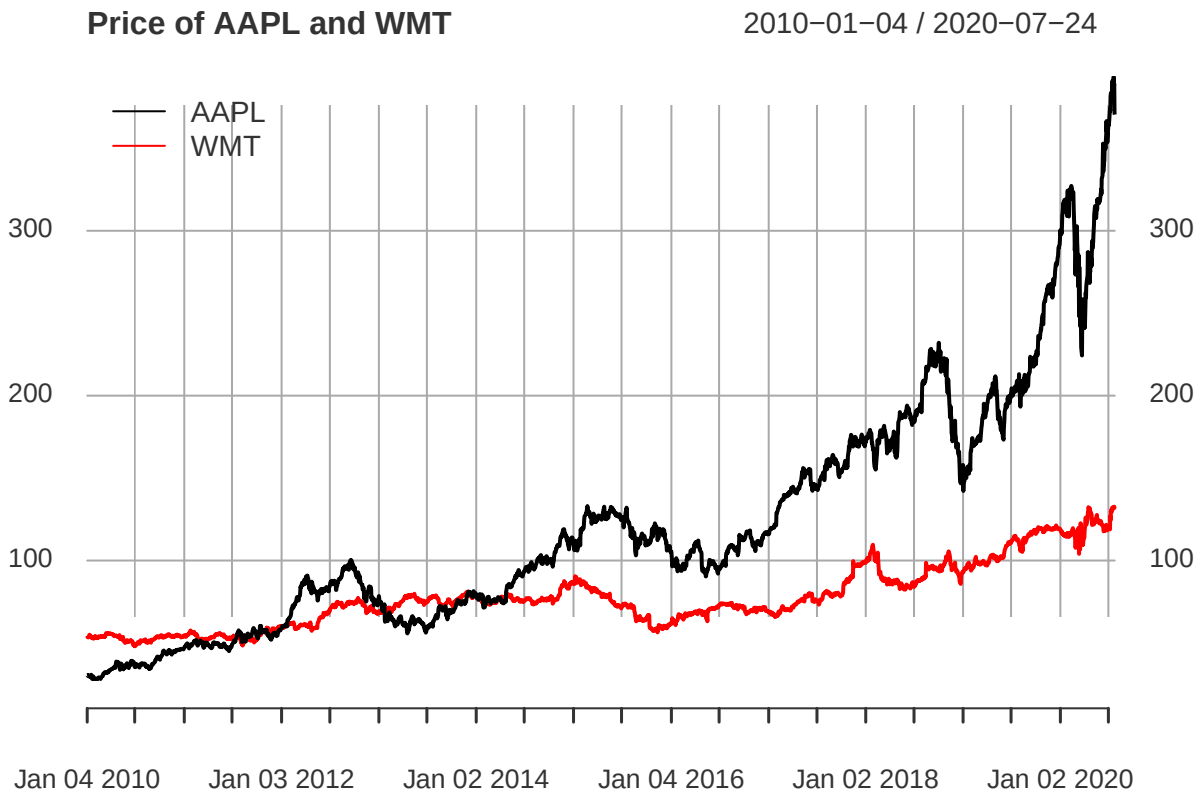

```
plot.xts(walmart_close, grid.col = 'white', yaxis.right = FALSE,
         main = "Price of Walmart stock", )
```


```
plot.xts(sp_close, grid.col = 'white', yaxis.right = FALSE,
         main = "Value of S&P 500 Index" )
```


```
# plot all in 1
plot.xts(cbind(apple_close, walmart_close),
         main = "Price of AAPL and WMT")
```




```
addLegend(legend.loc = "topleft", legend.names = c("AAPL", "WMT"), lty = 1)
```



Xts is a very powerful class of data designed for time series that involves real dates.

The plot generated is very close to base R with many shared syntax and options, which provides some degree of customizability, and it plots exactly what we need. In fact, if xts package is loaded, using the function plot() would get the exact same result as demonstrated above.

3. Rate of Return

Normal way

$$r = \frac{y_2 - y_1}{y_1}$$

Better way

$$r = log(y_2) - log(y_1) = log(y_2/y_1)$$

They are not the same in value, but the difference is usually small enough for assets in real world.

Advantages of logarithmic return:

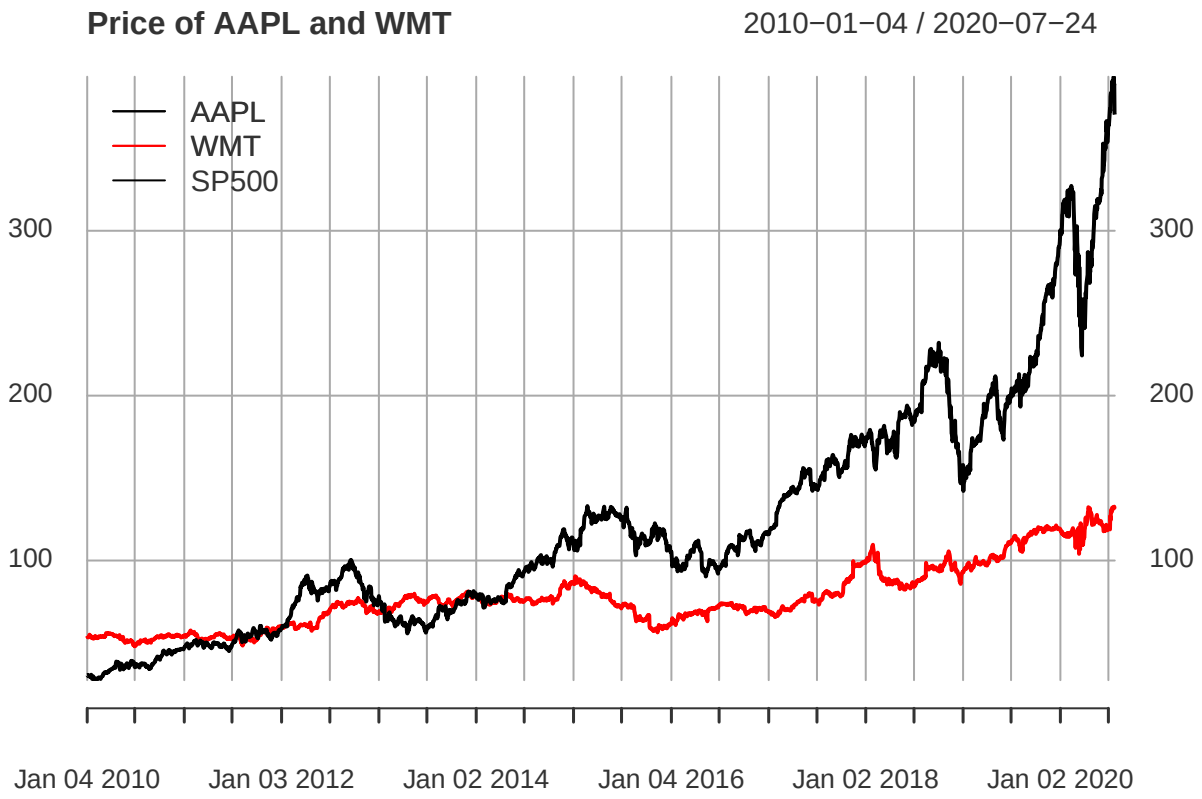
- Logarithmic returns are symmetric, while ordinary returns are not: positive and negative percent ordinary returns of equal magnitude do not cancel each other out and result in a net change, but logarithmic returns of equal magnitude but opposite signs will cancel each other out. This means that an investment of $100 that yields an arithmetic return of 50% followed by an arithmetic return of -50%

will result in $75, while an investment of $100 that yields a logarithmic return of 50% followed by a logarithmic return of -50% will come back to $100.

- Logarithmic return is also called the continuously compounded return. This means that the frequency of compounding does not matter, making returns of different assets easier to compare.
- Logarithmic returns are time-additive, meaning that if $R_{\log,1}$ and $R_{\log,2}$ are logarithmic returns in successive periods, then the overall logarithmic return $R_{\log}$ is the sum of the individual logarithmic returns, i.e. $R_{\log} = R_{\log,1} + R_{\log,2}$.
- The use of logarithmic returns prevents investment prices in models from becoming negative.

Implementation

```
# difference in log value
apple_return <- na.omit(diff(log(apple_close)))
walmart_return <- na.omit(diff(log(walmart_close)))
sp_return <- na.omit(diff(log(sp_close)))

# use na.omit() because first day has no previous day to calculate its return, producing NA


# combine 3 assets' return into 1 variable
all_return <- cbind(apple_return, walmart_return, sp_return)

#plot.xts(all_return)
addLegend(legend.loc = "topleft", legend.names = c("AAPL", "WMT", 'SP500'), lty = 1)
```




```
# a table of average return for easy demonstration
return_table <- data.frame(sapply(all_return, mean))
rownames(return_table) <- c(apple, walmart, "SP500")
colnames(return_table) <- "Average return"

kableExtra::kable(return_table, format = "markdown")
```

	Average return
AAPL	0.0009389
WMT	0.0003326
SP500	0.0003926

Looks like they are mostly fluctuatiing around 0%.

4. Dynamic Variance

There are many ways to measure variance in time series, some giving equal weight to every past observation and others give more weight to recent observations. Many require modelling.

To keep it simple, the 2 types of variance we use here are: ## Equal weight

```r
sd_table <- cbind(return_table, sapply(all_return, sd))
colnames(sd_table)[2] <- "SD"

kableExtra::kable(sd_table, format = "markdown")
```

	Average return	SD
AAPL	0.0009389	0.0174378
WMT	0.0003326	0.0119144
SP500	0.0003926	0.0110757

Every point in time

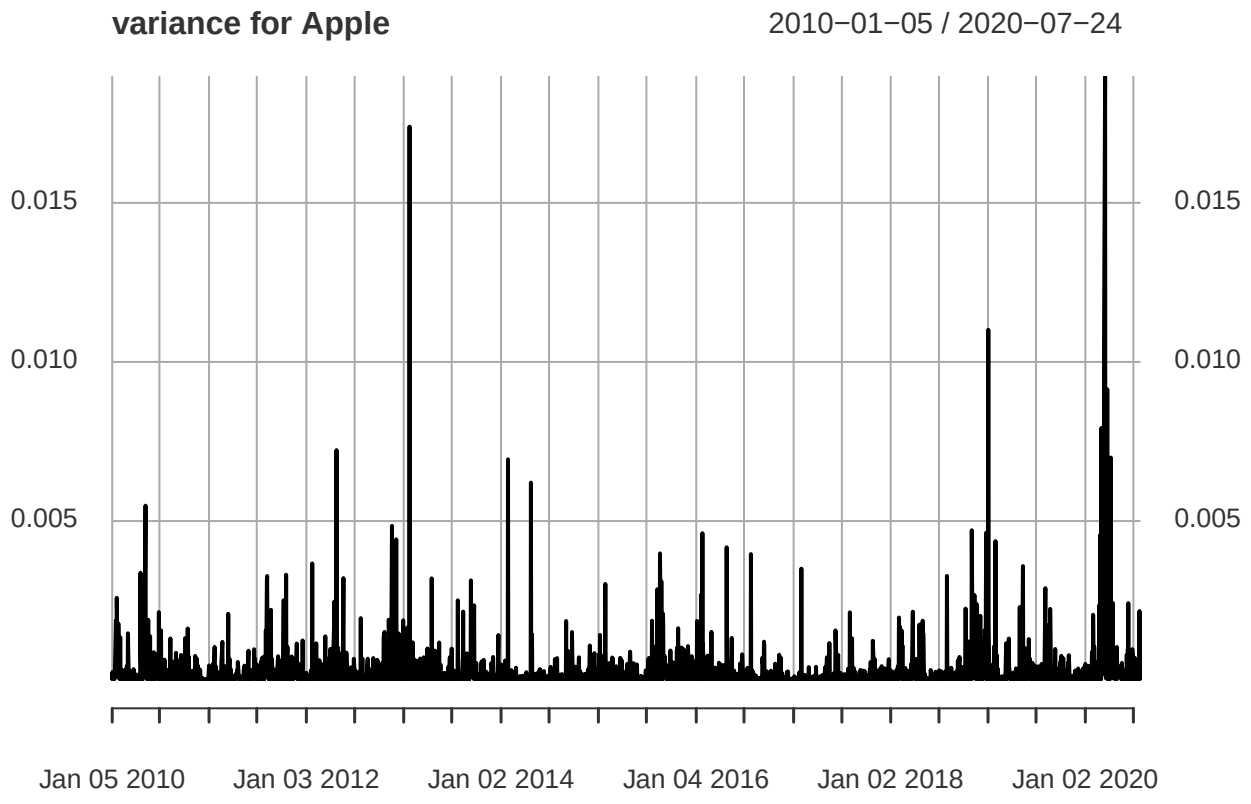
Since each observation in time series is a realization of a distribution unique to that point in time, we can calculate standard deviation or variance in this way:

$$Var(Y_t) = E(Y_t^2) - E(Y_t)^2 = E(Y_t^2)$$

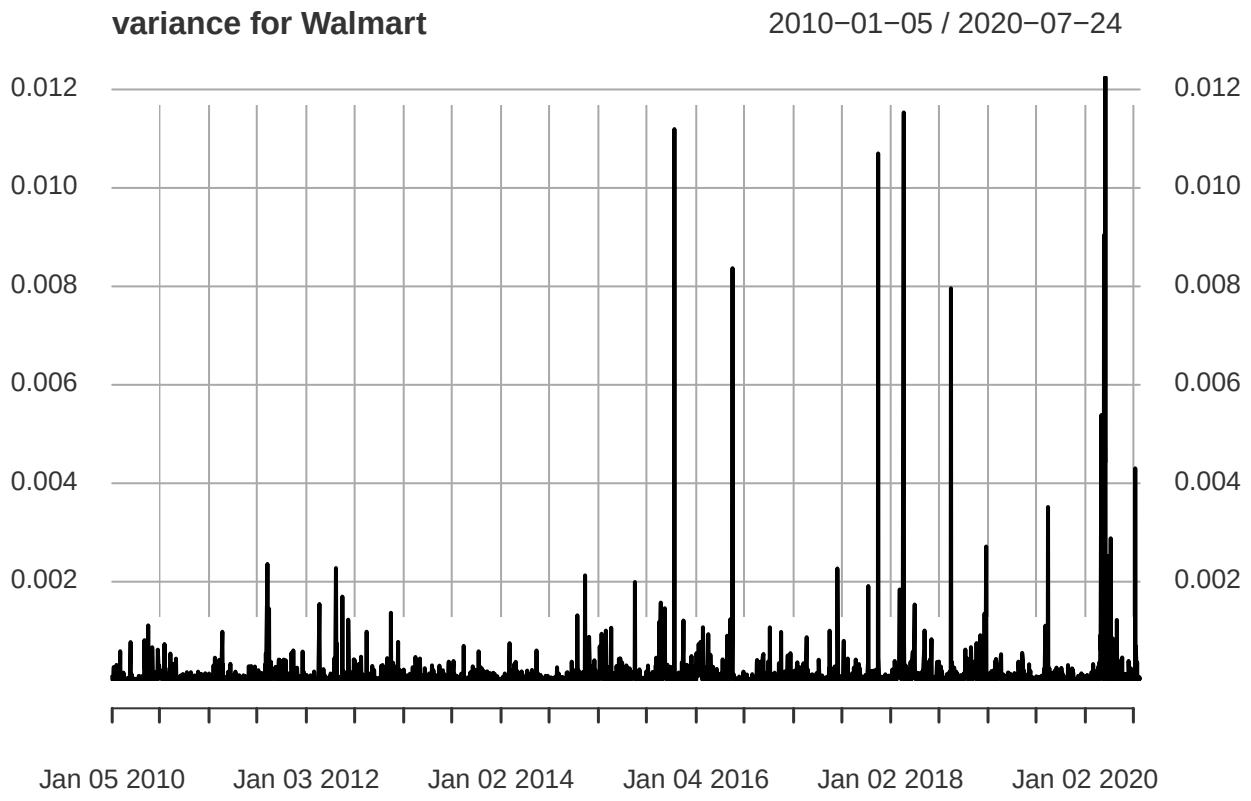
This is a measurment of how far each observation deviates from their expected result at that point in time. For financial assets, the expected mean of return is usually considered zero. (For more information, please look up "Stationarity").

Implementation

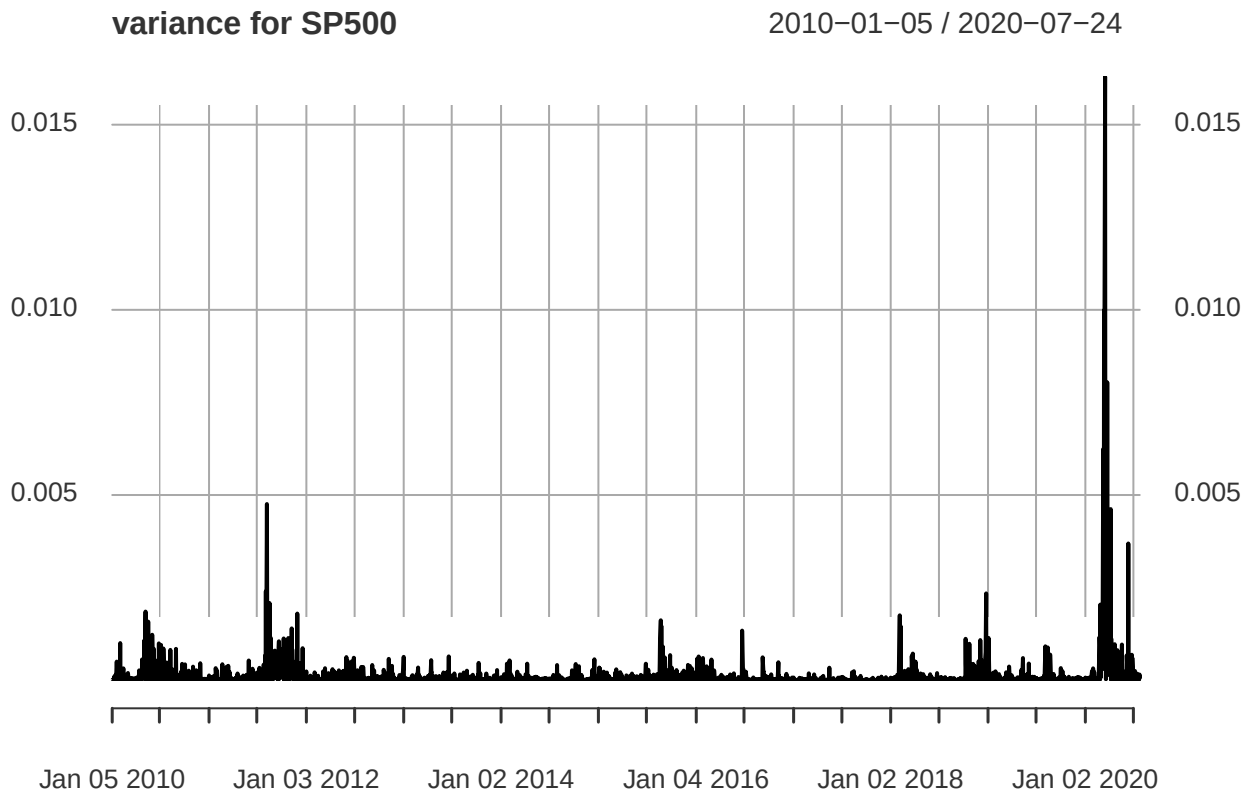
```r
# obtain variance for each point in time
apple_var <- apple_return^2
walmart_var <- walmart_return^2
sp_var <- sp_return^2

# plot them
plot.xts(apple_var, main = "variance for Apple")
```




```
plot.xts(walmart_var, main = "variance for Walmart")
```




```
plot.xts(sp_var, main = "variance for SP500")
```



5. Apply the portfolio optimization formula

$$L = w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + \lambda(\overline{\mu}_p - w_1\mu_1 - w_2\mu_2)$$

$$w_1 = \frac{\mu_1/\sigma_1^2}{\mu_1^2/\sigma_1^2 + \mu_2^2/\sigma_2^2}\overline{\mu_p}$$

$$w_2 = \frac{\mu_2/\sigma_2^2}{\mu_1^2/\sigma_1^2 + \mu_2^2/\sigma_2^2}\overline{\mu_p}$$

```
goal <- 0.0007

denominator <- sd_table$`Average return`[1]^2/sd_table$SD[1]^2 + sd_table$`Average return`[2]^2/sd_table

w1 <- (sd_table$`Average return`[1]/sd_table$SD[1]^2) / denominator * goal
w2 <- (sd_table$`Average return`[2]/sd_table$SD[2]^2) / denominator * goal

# the sum becomes 1 or 100%
w1_reweight <- w1 / (w1+w2)
w2_reweight <- w2 / (w1+w2)

portfolio <- apple_close * w1 + walmart_close * w2
portfolio_return <- na.omit(diff(log(portfolio)))

portfolio_table <- rbind(sd_table, c(mean(portfolio_return), sd(portfolio_return)))
```

```r
rownames(portfolio_table)[4] <- "portfolio"

kableExtra::kable(portfolio_table)
```

	Average return	SD
AAPL	0.0009389	0.0174378
WMT	0.0003326	0.0119144
SP500	0.0003926	0.0110757
portfolio	0.0007076	0.0131107

The return is not exactly what we expected because those two assets are not uncorrelated in real life. Their correlation is roughly 0.299.

Also we have constructed a portfolio that has much higher return than SP500 while having slightly higher risk. But whether this portfolio can perform better in long term in the future is unknown.