CW1

Code

29 September, 2022

Load libraries

```
library(quantmod)
library(tidyverse)
library(PerformanceAnalytics)
library(timeSeries)
library(tseries)
library(roll)
library(car)
library(MASS)
library(extraDistr)
library(rugarch)
library(rmgarch)
library(FKKS)
library(QRM)
library(dplyr)
library(rmarkdown)
```


Get stock data

Getting stock price data from Yahoo Finance using the function getSymbols

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances

```
rm(list=ls())
ENV.CW1 <- new.env() # Create environment where data are stored
Stocks <- c('SP500', 'JPM') # Stock names
tickers <- c('^GSPC', 'JPM') # Stock tickers
tickers_cleaned <- c('GSPC', 'JPM')
tickers_cleaned <- as.vector(sapply(tickers_cleaned,
                                    FUN = function(x) paste(x, '.Ad
        justed',
                                                            sep =
        '')))
# Function merge violates the original order of columns; tickers_cl
        eaned is used to restore the order

getSymbols(Symbols = tickers, src = 'yahoo',
           from = "1995-01-01", # Start date
           to = Sys.Date(), # For data up to today
           env = ENV.CW1)

# Create one XTS object (matrix + time) containing adjusted prices
        of all stocks
Adjusted_Stock_Prices <- do.call(merge, eapply(env = ENV.CW1, Ad))
# Ad extracts adjusted prices for every stock. Consequently, all ad
        justed prices are merged into one xts object
Adjusted_Stock_Prices <- Adjusted_Stock_Prices[, tickers_cleaned] #
        Restore the right order of columns
names(Adjusted_Stock_Prices) <- Stocks # Change names from tickers
        to real names
```


Convert into log returns

```
log_returns <- diff(log(Adjusted_Stock_Prices)) # Compute daily log
        returns
log_returns <- na.omit(log_returns) # Remove rows containing na's
```

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances

Alternative approach to compute returns using the function CalculateReturns

```
r = CalculateReturns(Adjusted_Stock_Prices, method = 'log')
```

Computing Summary Statistics

```
AvgRet = colMeans(log_returns)
StdDevRet = colSds(log_returns)
MaxRet = colMaxs(log_returns)
MinRet = colMins(log_returns)
SkewRet = colSkewness(log_returns)
KurtRet = colKurtosis(log_returns) # Excess Kurtosis
```


Alternative approach to compute summary statistics using the function apply function = mean, sd, max,..


```
AvgRet = apply(X = log_returns, MARGIN = 2, FUN = mean) # replace m
    ean by sd, max, min,....
```


Display daily summary statistics

```
DailyStats <- as.table(rbind(AvgRet, StdDevRet, MaxRet, MinRet, Ske
        wRet, KurtRet))
DailyStats
```

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances

```
##                    SP500           JPM
## AvgRet     0.0002995319  0.0004267344
## StdDevRet  0.0120819570  0.0236711629
## MaxRet     0.1095719677  0.2239169612
## MinRet    -0.1276521976 -0.2322773480
## SkewRet   -0.4241263954  0.2079889909
## KurtRet   10.2567826953 12.7039074861
```

Alternative approach to insert tables

```
knitr::kable(DailyStats, digits=4)
```


	SP500	JPM
AvgRet	0.0003	0.0004
StdDevRet	0.0121	0.0237
MaxRet	0.1096	0.2239
MinRet	-0.1277	-0.2323
SkewRet	-0.4241	0.2080
KurtRet	10.2568	12.7039

Converting averages and standard deviations to annual measures

```
AvgRetAnn = 252 * AvgRet
StdDevRetAnn = sqrt(252) * StdDevRet
knitr::kable(rbind(AvgRetAnn, StdDevRetAnn), digits=4)
```

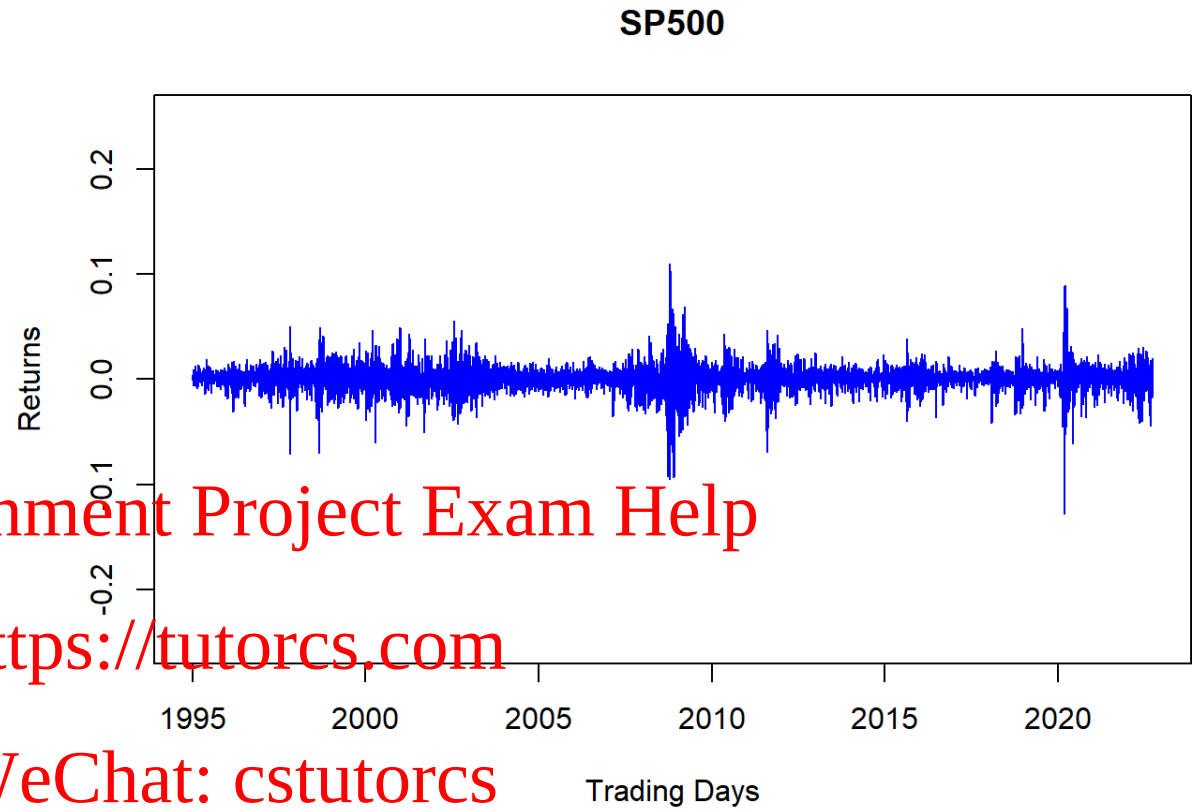
	SP500	JPM
AvgRetAnn	0.0755	0.1075
StdDevRetAnn	0.1918	0.3758

Graphical analysis of returns

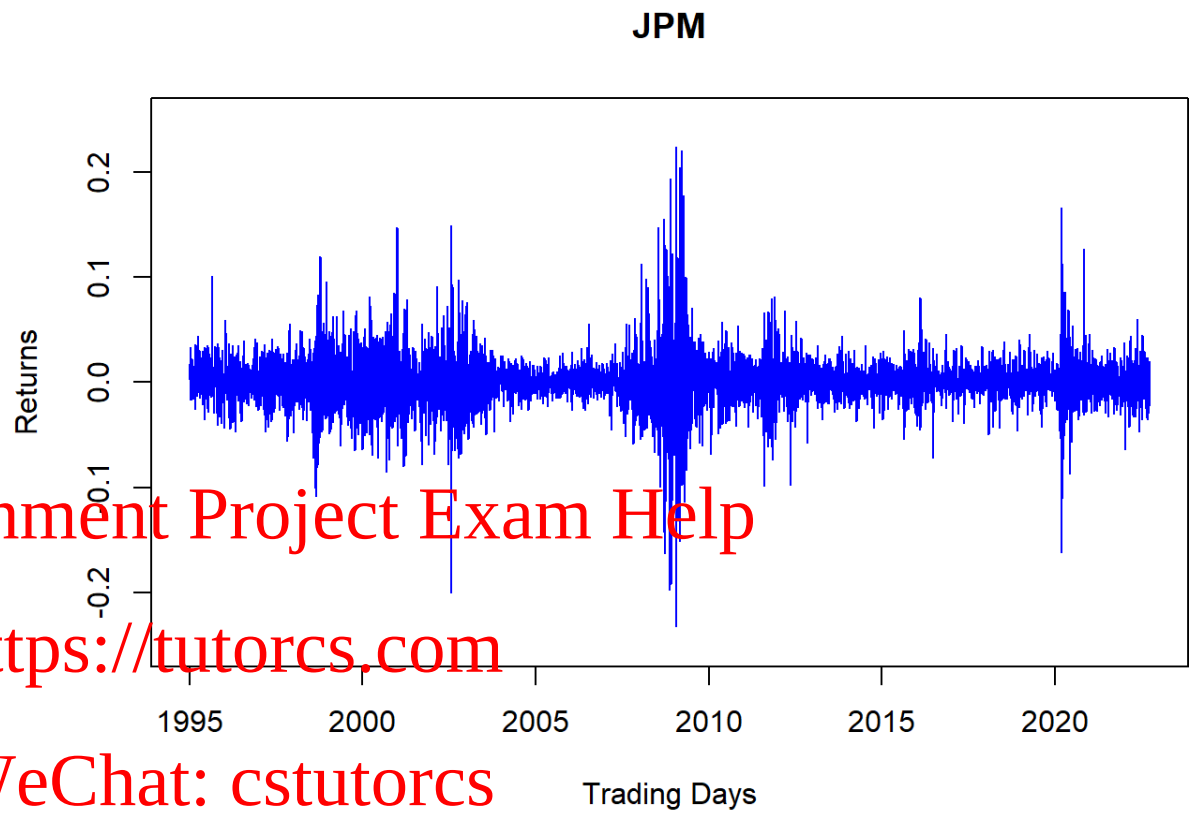
Plot daily returns for SP500 Index and JPM stock. We will use a FOR loop, which repeats a same piece of code as many times as we indicate.

```
for (i in 1:2) {
  plot(x = index(log_returns), y = log_returns[,i], type = 'l',
       main = Stocks[i], xlab = 'Trading Days', ylab = 'Returns', y
       lim = c(-0.25, 0.25), col = 'blue')
}
```

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances



- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances



Simulate daily returns with same sample mean and standard deviation as JPM.
Compare observed returns with simulated returns.

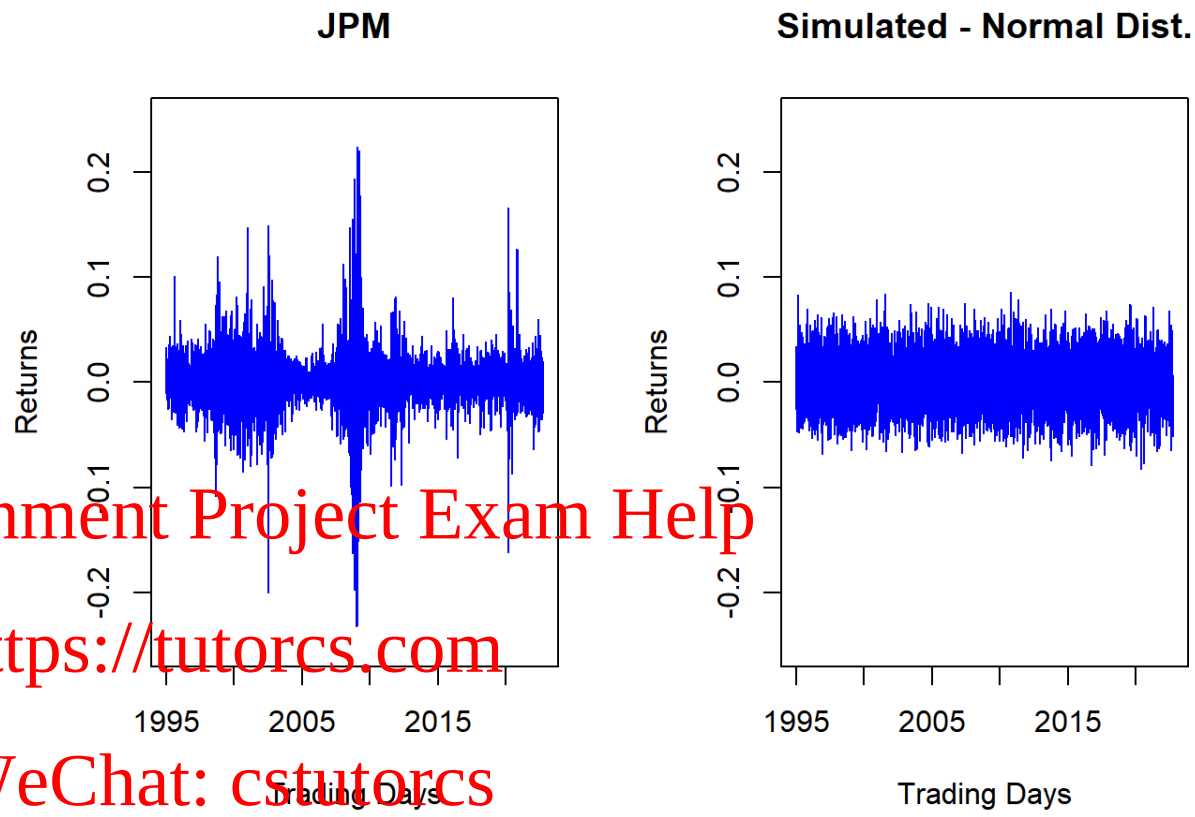
- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances

```
simret <- rnorm(n = length(index(log_returns)), mean = AvgRet['JP
        M'],
                sd = StdDevRet['JPM'])

par(mfrow=c(1,2)) # prepares space for two graphs in one window

plot(x = index(log_returns), y = log_returns[, 'JPM'], type = 'l',
     main = 'JPM', xlab = 'Trading Days', ylab = 'Returns', ylim =
        c(-0.25, 0.25), col = 'blue')
plot(x = index(log_returns), y = simret, type = 'l', main = 'Simula
        ted - Normal Dist.',
     xlab = 'Trading Days', ylab = 'Returns', ylim = c(-0.25, 0.2
        5), col = 'blue')
```

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances



After using par(mfrow = c()), make sure you reset the grid by running par(mfrow = c(1,1)) or dev.off().

Compare index performances

```
x1<-t(t(Adjusted_Stock_Prices)/drop(coredata(Adjusted_Stock_Prices
        [1,])))
xt<-xts(x=x1,order.by = index(Adjusted_Stock_Prices))
plot(x=index(xt),y=x1[,2], type = 'l', main = 'Index S&P500 and JPM
        organ',
    ylab = 'Index', xlab = 'Trading Days', col='red')
lines(x=index(xt),y=x1[,1], col = 'blue')

# Adding the legend
legend('topleft', legend = names(xt), col = c('red', 'blue'), lty=
    1)
```

- Load libraries
- Get stock data
- Convert into log returns
- Display daily summary statistics
- Graphical analysis of returns
- Compare index performances