Math 458 - Financial Mathematics for Actuaries II - Project # 1

Due: Midnight on Monday, October 2, 2023.

Objectives:

- Use R to acquire and graph real-life historical financial data on the stocks of your choice as well as compute each stock's expected return and volatility.
- Apply that information to the concept of a stock portfolio with its associated statistics to select the lowest expected risk stock allocation for a given expected rate of return.graph.

Directions:

1. Introduction: The analysis in this project will use R's data analysis package `tidyverse`, its financial quant package `tidyquant`, its publication quality package `ggpubr`, its matrix statistics package `matrixStats`, and its graphing packages `ggplot2` & `scales`. Load all of these libraries at the beginning of your R script for this project.

2. Phase One: The `quantmod` library includes the command `getSymbols` that will pull current stock price data from the internet and load it into your R environment as a time series (class `xts`). The command for loading in Apple, Google, and Lowe's from January 1, 2020 to June 30, 2020 is

   ```
   getSymbols(c("AAPL","GOOG","LOW"), from="2017-01-01",
   to="2020-09-19", warnings=FALSE, auto.assign = TRUE)
   ```

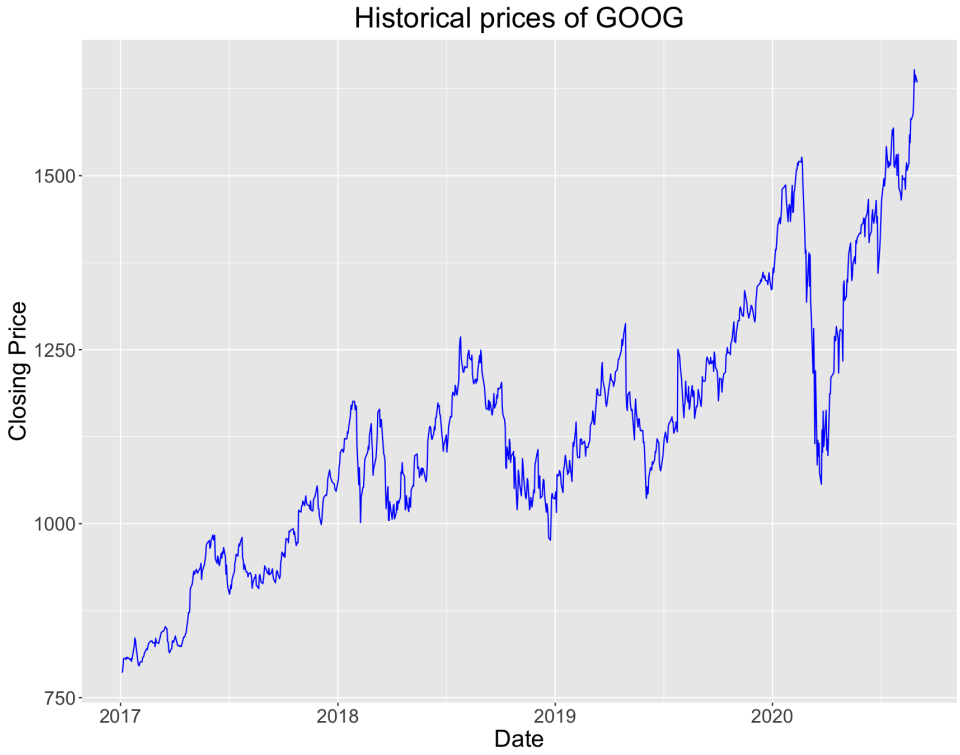
   The output is three separate time series with the names "AAPL", "GOOG", and "LOW", where these are the ticker names for the respective companies. In your code, choose three **different stocks** (i.e., not Apple, Google, or Lowe's) to load data from January 1, 2017 to TODAY. You will need to look up the ticker symbols on the web (try Yahoo Finance).

3. Phase Two: Use the `fortify` command to convert each of the three stock time series into a data frame with the date column named "Date" so that ggplot2 can easily use the data. For example, to convert the "GOOG" time series above into an analogous "GOOGdf" data frame, use the commands:

   - `GOOGdf<-fortify(GOOG)` - this converts the time series in `xts` format to a data frame called `GOOGdf`. You should end up with three data frames, each with name consisting of the ticker in capital letters followed by "df".

- Use the `colnames` command to make the first column of each of your data frames have the name "Date". Check your work using the `head` command to verify that your data frames have a first column with name "Date".

4. Phase Three: Use the ggplot command and your data frames from Phase Two above to graph the historical closing stock prices for each of your three stocks. For example, a graph for Google would look like

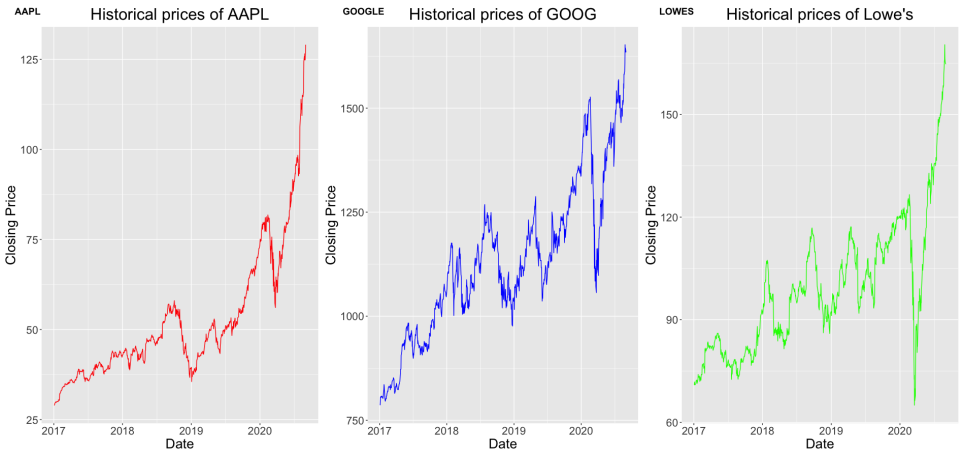


Each of your plots should be named TICKERplot. For example, my AAPL plot is called "`AAPLplot`". The appropriate `ggplot2` command should be similar to

```
AAPLplot<-ggplot(AAPLdf,
aes(x=Date,y=AAPL.Close))+geom_line(color="red")+theme(legend.position
= "none")+ ···
```

Now use the `ggpubr` command `ggarrange` to combine all three of your graphs (one for each stock) into a single image called `ThreeStockPlot`. For example, the following graph is the result of the commands (where `"ThreeStockPlot"` is the command to display the plot in the viewing pane.

```
ThreeStockPlot<-ggarrange(AAPLplot,GOOGplot,LOWplot,
  labels=c("AAPL","GOOGLE","LOWES"),ncol=3,nrow=1)
ThreeStockPlot
```



5. Phase Four: This phase of the problem will require you to compute daily stock returns, find their mean and standard deviation, and then plot each stock as a single point on a graph with expected return the horizontal axis and historical volatility the vertical axis. Follow the following steps:

   (a) For each stock, create a vector called "TICKER_Returns" (mine is called "GOOG_Returns", consisting of the numbers

   $$r_t = \ln\left(\frac{S_{t+1}}{S_t}\right),$$

   where S_{t+1} is the stock price "tomorrow' and S_t is the stock price "today". You should have one less number in your returns vector than the number days in your data frame. Note that the natural logarithm ln() in R is spelled `log()`.

   (b) Use your previous answer and the `cbind` command to create a matrix called "returns" with three columns, each of which consists of the stock returns r_t for each of your three stocks.

   (c) Apply the command `colMeans` to the matrix "returns" to find the average return of each stock:

   $$\hat{r} = \frac{1}{N}\sum_{t=1}^{N} r_t.$$

   Here N is the number of rows in the returns matrix. Use the `round` function to round each element of the mean vector to 10 decimal places. Then multiply this rounded mean vector by 252 (the rough number of trading days in a year) and assign the result to the vector "expectedreturns". This should be a vector with three numbers, one for the annual expected return of each stock.

   (d) Repeat the previous step on "returns" except use the command `colSds` to compute

   $$\hat{S} = \sqrt{\frac{\sum_{t=1}^{N}(\hat{r} - r_t)^2}{N-1}},$$

   the historical volatility of each stock. This will create a vector of three positive numbers representing the estimated volatility/trading day, rounded to 10 decimal

places. This vector you will need to multiply by $\sqrt{252}$ (instead of 252 as in the case of means), resulting in a vector called "volatilityreturns", a vector of three positive numbers.

(e) Create a data frame containing this information via

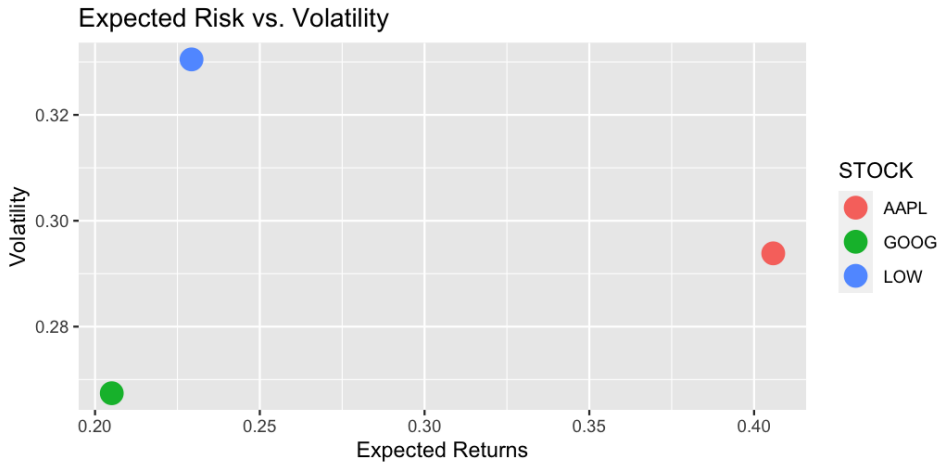
```
plotdata<-as.data.frame(cbind(expectedreturns,
        volatilityreturns),as.numeric)
```

and then add a column with the tickers for each stock in your project as the first column of the data frame, and names (left to right) for the columns of "STOCK", "ER", and "SD". This should be a data frame with three rows and three columns, with the last two columns respectively containing the expected returns and volatility estimate for each stock.

(f) Use the `ggplot` command

```
ggplot(plotdata, aes(x = ER, y = SD,color=STOCK))
```

with appropriate `geom_point`, `ggtitle`, and `xlab/ylab` attributes to create a plot similar to



6. Phase Five: This phase of the problem consists of graphing the expected return and risk associated with portfolios representing 1000 different allocations between two assets. This graph will allow you to find the portfolio with the highest expected return for a given risk (volatility) tolerance. Alternatively, you can find the portfolio composition that minimizes risk and subsequently identify the corresponding expected return.

(a) Let $R_1 = \{r_{1,j}\}_{j=1}^N$ and $R_2 = \{r_{2,j}\}_{j=1}^N$ be the N returns for the first two stocks in your problem - as contained in the first two columns of the `returns` matrix, each with mean $\hat{r}_k$ for $k = 1, 2$. Assign to the variable `cov_12` the *sample covariance*

$$\operatorname{cov}(R_1, R_2) := \frac{1}{N-1} \sum_{j=1}^{N} (r_{1,j} - \hat{r}_1)(r_{2,j} - \hat{r}_2)$$

using the R covariance function `cov` as follows

$$\texttt{cov\_12<-cov}(R_1, R_2),$$

where $R_{1,2}$ are the first an second columns of your `returns` matrix. Repeat this process for the stock pairs R_2, R_3 and R_1, R_3, storing your answers in variables `cov_23` and `cov_13`.

Remark: The sample covariance between two sample vectors is a measure of magnitude and direction of any relationship between two random samples. If the the variables are independent then the covariance is necessarily zero.

(b) Create a vector called `weight1` which contains 1001 numbers, starting at 0.0 and increasing by 0.001 until the value 1 is reached. Then form the vector `weight2`, where each element in `weight2` is one minus the corresponding element in `weight1`. Adding `weight1` to `weight2` should result in a vector containing 1001 copies of the number 1.

(c) Consider a portfolio $\pi = \{w_1, w_2\}$ with $w_1\%$ allocated to stock 1 and $w_2\%$ allocated to stock 2. Here $w_i \geq 0$ for each i and $w_1 + w_2 = 1$. Then the expected return of this portfolio would be

$$\mu_\pi = w_1\hat{r}_1 + w_2\hat{r}_2$$

and the corresponding portfolio volatility σ_π is

$$\sigma_\pi = \sqrt{w_1^2 \cdot \hat{\sigma}_1^2 + 2 \cdot w_1 \cdot w_2 \cdot \text{cov}\,(R_1, R_2) + w_2^2 \cdot \hat{\sigma}_2^2}.$$

Create two vectors, one of which contains μ_π for each of the 1001 weight allocations and the other contains σ_π for the 1001 weight allocations.

(d) Use your answers to the previous two parts and the `cbind` command to create a matrix with four columns corresponding (left-to-right) containing w_1, w_2, μ_π, and σ_π. Then apply the `as.data.frame()` command to your matrix to create a data frame with the same four columns. Add column names to the data frame of "`weight1`", "`weight2`", "`ER`", and "`VOL`", and then name the data frame `portfolio12`.

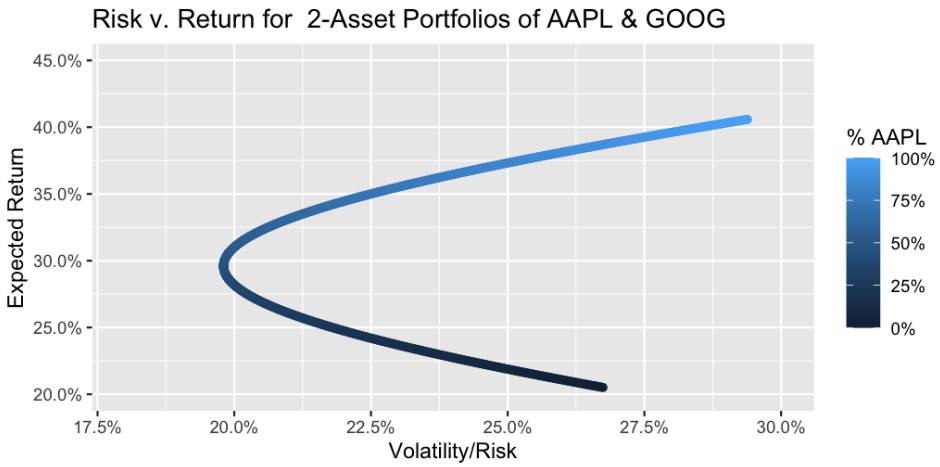
(e) Use `ggplot` with the aesthetic

```
aes(x=VOL,y=ER,color=weight1)
```

and the attributes

```
scale_y_continuous(label = percent, limits = c(y1, y2)) +
scale_x_continuous(label = percent, limits = c(x1, x2)) +
scale_color_continuous(name="% TICKER 1",label=percent)
```

where x_1, x_2, y_1, y_2 are chosen so that your graph is nicely centered, and TICKER 1 represents the stock ticker corresponding to R_1. Your graph should look similar to the below graph I created for Apple and Google:



Use this graph and your data to find the portfolio allocation w_1 and w_2 that maximizes return given a risk volatility of $(x_1 + x_2)/2$.

(f) Repeat steps (a)-(e) for the stock pairs (R_2, R_3) and (R_1, R_3), making variable name changes to account for the changed indices.

When finished, your code should able to run on my computer and generate all of the above automatically. Each group's output should be similar, but not identical, since you are all choosing different triples of stocks to analyze.