Stock Portfolio Optimization and Automation Project

Daniel Carpenter*

April 1, 2020

Abstract

This paper will outline the methods and uses of a program that optimizes a stock portfolio. The portfolio consists of any stock that the user desires to include in the portfolio. The user can adjust the date range. If the tool is used for active portfolio management, the program suggests that the date range ends on the current date. The methods are based primarily on modern portfolio theory. This theory minimizes the risk of a portfolio, given the user's desired return. The program uses a linear optimizer, which adjusts the weights of each stock to minimize the risk while setting the expected return to the user's input. The program will output the stock weights, risk, and return characteristics, and some visualizations that aid the user in decision-making.

*University of Oklahoma. E-mail address: danielcarpenter@ou.edu

1 Introduction

All individuals want to retire from their careers after some duration of time, but not all individuals can retire at the age that they desire. One may decide that their marginal propensity to consume satisfies their immediate needs and aspirations; however, focusing on delayed gratification will allow someone to foster realities like earlier retirement. As an individual who will soon enter the workplace, I know that I will also be eager to leave the workplace to enjoy the finer elements of my lifetime. While at The University of Oklahoma, my pursuit of a bachelor's in business administration in Economics and Finance has motivated me to think critically about investing portions of my income. There are many forms of investments that someone may discover, but one of the more common are investments in the stock market. Many argue that the volatility of the stock market creates too much risk for individual investors, which their claims have validity within certain scopes (Zuckerman and Gregory (2020)). Corporate investors pay high premiums to minimize the time to trade on the stock market, which puts the individual investor at a disadvantage (Patterson and Osipovich (2020)). Although these circumstances hold, alternative investments may not be as liquid as stocks; if the investment is more liquid, such as a United States Treasury Bill, the "risk-less" asset will not realize significant returns (Canady (2020)). However, individual investors can minimize their risk when using certain financial methods. Within this paper, we will discuss how Modern Portfolio Theory allows an individual to minimize their risk of any portfolio, which can generate a return that satisfies the individual.

2 Literature Review

Within this paper, combining the two concepts of Modern Portfolio Theory and the Capital Market Line underlines the financial principals that the program uses to optimize a stock portfolio. The reader may know the simple principle to "never put all your eggs into one basket;" Modern

Portfolio Theory implements this idea using statistical methods and linear optimization to create a portfolio that maximizes utility and minimizes risk for an investor (Benninga (2008)). This method maximizes utility because it allows the investor to initially create their portfolio, given their preference in certain stocks (Benninga (2008)). Harry Markowitz announced this method is the 1950's, who would receive a Nobel Prize in Economics in 1990 for the advancement of portfolio choice (Benninga (2008)). In addition to this idea, the Capital Market Line allows the user to allocate their optimal, or efficient, stock portfolio with "riskless" assets, such as the United States Treasury Bills (Benninga (2008)). The reader may find this discovery obvious, but the positive relationship between interest rate yields and the standard deviation of the yield commonly referred to as the risk of the asset, allows the Capital Allocation Line to easily reduce the risk, if the investor holds more weight in less risky assets (Benninga (2008)).

Overall, these two ideas allowed investors to have more power over their portfolios; at the time of this discovery, individuals relied on their brokers to choose their stocks. This concept announced the allowance for "risk tolerance," which the investor could best "fit their risk/reward profile" (Beattie (2020)). However, these ideas are not commonly used by individual investors anymore; after extensive analysis, it is clear that "indexing," which is a form of passive investing, outperforms "active" portfolio management (Malkiel (2017)). According to a study that the Wall Street Journal conducted, over 90 percent of actively managed portfolios underperform the SP 500, which individuals can find indexes that combine the firms included in this value-weighted index fund (Malkiel (2017)). These index funds involve high-frequency trading tactics, which computer algorithms trade in fractions of a second; these machines are unbeatable by active portfolio managers, but individual investors may choose to index these funds to take part in the computerized gains (Patterson and Osipovich (2020)). Among these tactics, the introduction of artificial intelligence is at the forefront of portfolio management (Aenlle (2018)). These implementations may allow for more powerful index funds, but full implementation of these ideas is not widely used yet (Aenlle (2018)).

The user of this program must understand that actively managing a portfolio may not prove

as successful as indexing, but it is important to understand that investors can establish preference that set their risk tolerance in the portfolio.

3 Data

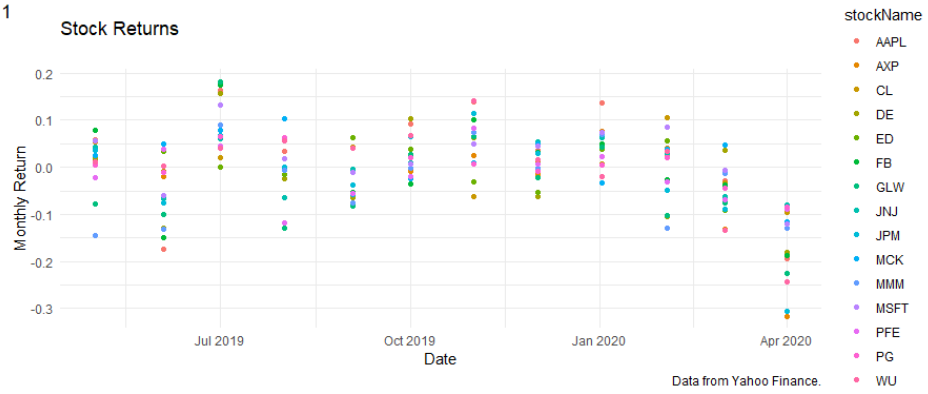
3.1 Using Yahoo Finance's API

The program uses one source of data, which houses stock-related information to the user. To allow for successful implementation of this program and continuous optimization of their stock portfolio, the use of the "BatchGetSymbols" package in R allows the user to have real-time data for any given stock. The package uses the API feature that Yahoo Finance allows to their constituents; Yahoo Finance's allowance for API integration to their data provides seamless integration when applying the Modern Portfolio Theory. Not only can this package retrieve the most current data, but the user can also set the beginning period to their choice, which may prove valuable when wanting to consider only more recent periods. Within this program, the data includes month-end stock prices, which "BatchGetSymbols" will automatically calculate the monthly returns of the stock. See below graph for example of stock returns that the data automatically generates.

3.1.1 Figure 1: Example of Historical Returns using "BatchGetSymbols" package in R

Additionally, the user can select as many stocks as they would like to include in their portfolio. Conveniently, the package allows the operator to import the SP 500, which is the optimization program's default setting. This capability offers a vast span of the market, in which the optimization program can analyze and eliminate stocks that do not fit the user's risk profile. An example of the data for stocks "JNJ," "MSFT," and "PG" from February through April 2020 can be seen below.

Figure 1



	date	stockName	stockPrice	stockReturn
1	2020-02-03	JNJ	149.22	0.03
2	2020-03-02	JNJ	140.02	-0.06
3	2020-04-01	JNJ	128.81	-0.08
4	2020-02-03	MSFT	173.90	0.07
5	2020-03-02	MSFT	172.79	-0.01
6	2020-04-01	MSFT	152.11	-0.12
7	2020-02-03	PG	125.11	0.01
8	2020-03-02	PG	119.56	-0.04
9	2020-04-01	PG	109.33	-0.09

3.1.2 Table 1: Example of ”BatchGetSymbols” API Export (Cleaned)

3.2 Altering the Dataset

Once the optimization program successfully imports the dataset, it will manipulate the data to create the optimal format for the optimization portion of the program. The program will adjust aesthetic features, such as renaming the variables to a more familiar format. Additionally, the program will drop rows of the dataset that are null, which could be from the firm not having been publicly traded during the dates that the user selected, or it could be because of the dataset having null values in the stock return column. Further manipulation of the dataset occurs, which the “Empirical Methods” section covers in detail, due to the manipulation relying on statistical manipulation.

4 Empirical Methods

The optimization program uses financial and statistical methods to manipulate the dataset, which sets up the data for linear optimization.

4.1 Calculating Excess Returns

First, the program calculates excess returns for each stock.

$$Returns_{Excess} = Return_{Stock_i} - RiskFreeRate \tag{1}$$

Calculating the excess return of a stock allows the user to understand what the stock returns beyond a riskless asset. A common proxy for a "riskless" asset is the U.S. T-Bill, which is used in this analysis (Benninga (2008)). The use of this method is to understand the tradeoff between a riskless asset and another asset that bears greater risk. For example, if a stock returns less during a period on average than the U.S. T-Bill, which has little to no risk, then the investor should not take on more risk if there is no excess return. Remember that the relationship between risk and return is positive, in which the previous example would be negative (Benninga (2008)). Note that all calculations are per period, in which excess return should be denoted as the monthly-excess return within this analysis.

4.2 Calculating the Variance-Covariance Matrix

The following method includes calculating the "Variance-Covariance Matrix," which Benninga's textbook, Financial Modelling, refers to as "central to all portfolio optimization" (Benninga (2008)). This matrix helps the program's optimizer find the optimal weights that the investor should allocate to each stock. Broadly, this calculation discovers which stocks hold lower risk when implementing them into one portfolio (Benninga (2008)).

The goal of this method is to understand how volatile an individual stock is, but it also will unveil increased risk of multiple stocks if they tend to "move" together (Benninga (2008)). Referring to the example of "putting your eggs into one basket," this matrix shows which eggs to choose, which are coming from varying baskets. Continuing this example concerning the Variance-Covariance Matrix, if multiple baskets have a higher risk to be harmed at the same time as each other, then the Variance-Covariance Matrix will identify to the user which baskets should be removed. The calculation is as follows.

$$Variance - Covariance - Matrix = \frac{X^{T}_{ExcessReturns} * X_{ExcessReturns}}{NumberOfStocks_{X_{ExcessReturns}}} \tag{2}$$

Using the stocks from Table 1 while increasing the date range to 10-years, the reader can view the Variance-Covariance Matrix for that portfolio below.

4.2.1 Table 2: Example of Variance-Covariance Matrix for Three-Stock Portfolio

	JNJ	MSFT	PG
JNJ	0.04	0.03	0.02
MSFT	0.03	0.08	0.02
PG	0.02	0.02	0.04

Within this matrix, the diagonal entries resemble the individual variances per stock; the off-diagonal entries show the covariance between two varying stocks (Benninga (2008)). If there is a higher covariance between certain stocks, then they have historically fluctuated in similar directions (Benninga (2008)). Notice that in this example, the diagonal entries are larger than the off-diagonal entries, which theoretically will always occur (Benninga (2008)).

4.3 Optimizing the Weights of Each Stock in the Portfolio

After the program calculates the Variance-Covariance Matrix, the program will then use linear optimization techniques to minimize the risk of the portfolio, in which the equation is outlined in

the following section, given the desired return an investor expects. Using the R package called "Rglpk," the program will set the risk of the portfolio as its objective, in which it will minimize. The program will also set constraints on the optimization function, which will ensure that the combined weights of each stock add up to the portfolio's 100 percent. For example, it is impossible to have 150 percent of our investment in some set of stocks; the allocation must not be greater than the investment. Another constraint is that the return must be greater than or equal to the user's desired return. This assumption will allow for the package to minimize the risk of the portfolio, while still meeting a given return. Once the constraints have been satisfied, the package will give the program the optimal weights for each stock. Note that the user inputs the dollar amount of their investment, which in the case below is 10,000 dollars. See the following table for an example of the optimized output using the stocks in the previous examples.

4.3.1 Table 3: Optimized Portfolio Allocation

	Stock Name	Stock Weight	Dollar Investment
1	JPM	1.00	10000.00

In the above, case the optimal choice for a 10 percent expected return is to allocation your entire 10,000 dollar investment into JNJ's stock.

4.4 Calculating the Risk of the Optimized Portfolio

Now that the program has optimized the weight of each stock in the portfolio, the program will calculate print the expected risk of the portfolio. It should be noted that the program has already calculated the risk so that it could minimize the equation; however, the program will validate the risk by printing the output to the user. If the portfolio included only a single stock, then the risk would simply be the standard deviation of the stock's excess return. However, since a portfolio will typically include more than one stock, the program will use matrix algebra to compute this metric. Let the weights of each stock be a 1 x m vector, which includes the weight of each stock that the program optimized. The program multiplies this equation by the square root of 12, which provides

the annualized risk for the portfolio. Given this vector and the Variance-Covariance Matrix, see the calculation of the risk below.

$$Risk_{Portfolio} = \sqrt{12} * \sqrt{(StockWeights * Variance - Covariance - Matrix) * StockWeights^T} \quad (3)$$

4.5 Calculating the Expected Return of the Optimized Portfolio

To calculate the expected return of the portfolio, the program will use matrix multiplication to handle this problem. Calculating the return of the portfolio is a much simpler process than the risk of the portfolio. Conceptually, the program multiplies the stock weights by their respective expected returns, then adds these values to find the expected return of the portfolio. See the below equation for the expected return of a portfolio using matrix notation.

$$Return_{Portfolio} = 12 * StockWeights * \overline{Return_{Stocks}}^T \quad (4)$$

4.6 Calculating the Sharpe Ratio of the Optimized Portfolio

The final calculation that the program performs is the Sharpe Ratio. The Sharpe Ratio is a measure of risk-adjusted returns. Explicitly, the measure gives the investor a value that shows the excess return per unit risk that they take by investing in this portfolio. The investor can compare Sharpe Ratio's against another portfolio, which is a useful tactic when making investment decisions. The calculation of the Sharpe Ratio is below.

$$Sharpe_{Portfolio} = \frac{Return_{Portfolio} - RiskFreeRate}{Risk_{Portfolio}} \quad (5)$$

Although investors commonly use the Sharpe Ratio, note that the main use is to compare this value to other portfolios that yield similar excess returns. For example, a portfolio that yields a very high yield and has relatively high risk may have a Sharpe Ratio that is higher than a portfolio with very low return and a small level of risk. Analysis of this method will vary between investors, based on their risk tolerance.

4.7 Summary Output of Risk, Return, and Sharpe Ratio

Below shows the final output of the risk, return, and Sharpe Ratio of the optimized portfolio, using the example stocks from above.

4.7.1 Table 4: Summary Output of Risk, Return, and Sharpe Ratio of the Optimal Portfolio

	Value
Risk	15.97
Expected Return	0.13
Sharpe Ratio	0.01

5 Research Findings

Through the establishment of this tool, it is clear that passive index funds lead in superiority, compared to Modern Portfolio Theory. Although minimizing the risk of any portfolio does have validity in its fundamental nature, the competition of trading is fierce (Zuckerman and Gregory (2020)). Additionally, the active management of a portfolio may be costly in labor, which investment banks will charge minuscule rates on the return of a portfolio (Zuckerman and Gregory (2020)).

6 Conclusion

The use of this program is to assist any investor in creating a stock portfolio and comparing these portfolios to other portfolios. As mentioned at the beginning of this paper, most individuals aspire to meet their retirement with plentiful amounts of wealth so that they may finish their lives with splendor and ease. Although indexing has crowded out Modern Portfolio Theory in many ways, this tool allows any investor to understand potential areas of investments, which require low levels of capital on the front end. Additionally, the high levels of liquidity that these assets offer give flexibility to the investor, which is a typical aspiration when pooling your funds into investments.

References

Aenlle, Conrad De. 2018. "A.I. Has Arrived in Investing. Humans Are Still Dominating." .

Beattie, Andrew. 2020. "Understanding The History Of The Modern Portfolio." .

Benninga, Simon. 2008. "Financial Modeling." 4:197–272.

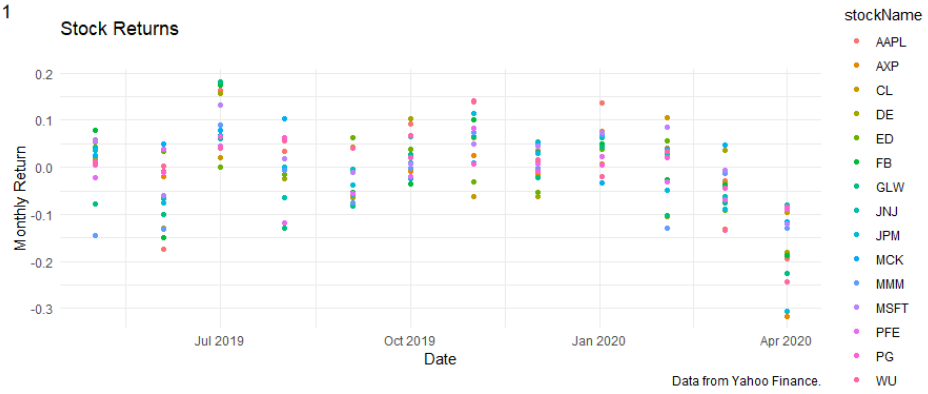
Canady, Dr. Tisa Silver. 2020. "How Interest Rate Cuts Affect Consumers." .

Malkiel, Burton G. 2017. "Index Funds Still Beat 'Active' Portfolio Management." .

Patterson, Scott and Alexander Osipovich. 2020. "High-Frequency Traders Feast on Volatile Market." .

Zuckerman, Gunjan Banerji and Gregory. 2020. "Why Are Markets So Volatile? It's Not Just the Coronavirus." .

Figure 1



7 Tables and Figures

7.1 Figure 1: Example of Historical Returns using "BatchGetSymbols" package in R

7.2 Table 1: Example of "BatchGetSymbols" API Export (Cleaned)

	date	stockName	stockPrice	stockReturn
1	2020-02-03	JNJ	149.22	0.03
2	2020-03-02	JNJ	140.02	-0.06
3	2020-04-01	JNJ	128.81	-0.08
4	2020-02-03	MSFT	173.90	0.07
5	2020-03-02	MSFT	172.79	-0.01
6	2020-04-01	MSFT	152.11	-0.12
7	2020-02-03	PG	125.11	0.01
8	2020-03-02	PG	119.56	-0.04
9	2020-04-01	PG	109.33	-0.09

	JNJ	MSFT	PG
JNJ	0.04	0.03	0.02
MSFT	0.03	0.08	0.02
PG	0.02	0.02	0.04

7.3 Table 2: Example of Variance-Covariance Matrix for Three-Stock Portfolio

7.4 Table 3: Optimized Portfolio Allocation

	Stock Name	Stock Weight	Dollar Investment
1	JPM	1.00	10000.00

7.5 Table 4: Summary Output of Risk, Return, and Sharpe Ratio of the Optimal Portfolio

	Value
Risk	15.97
Expected Return	0.13
Sharpe Ratio	0.01

8 Source Code

```
# LIBRARY ----------------------------------------------------------------------
  library(tidyverse)
  library(dplyr)
  library(knitr)             # Used to concatenate words
  library(lubridate)         # Used to clean up dates
  library(tidyr)             # Used to pivot data frame
  library(BatchGetSymbols)   # Used to Pull in Stock Data
  library(Rglpk)             # Used to optimize portfolio weights
  library(stargazer)
  library(ggthemes)
  library(xtable)


# INPUTS -----------------------------------------------------------------------
    startDate      <- Sys.Date() - 365 * 10
    endDate        <- Sys.Date()


    #df.SP500      <- GetSP500Stocks()
    #stockList     <- df.SP500$Tickers
    stockList      <- c('JNJ', 'PG', 'JPM')
    riskFreeRate   <- .0160
```

```
  desiredReturn        <- 0.15    # 10% = 0.10
  dollarsInvested      <- 10000


# DATA PULL ------------------------------------------------------------------
  # Make Variables Monthly
  desiredReturn <- desiredReturn / 12


  tBill <- c('^IRX')


  # Pull Stock Data
  df <- BatchGetSymbols(tickers = c(tBill, stockList),
                        first.date = startDate,
                        last.date = endDate,
                        freq.data = 'monthly',
                        cache.folder = file.path(tempdir(), 'BGS_Cache'))


  # Mutate Data (Drop Cols and Rename)
  df <- as.data.frame(df$df.tickers) %>%
    select(ref.date,
           ticker,
           price.adjusted,
           ret.adjusted.prices) %>%
```

```
    mutate(ref.date = as.Date(ref.date)) %>%
    rename(date = ref.date) %>%
    rename(stockPrice = price.adjusted) %>%
    rename(stockName = ticker) %>%
    rename(stockReturn = ret.adjusted.prices) %>%
    drop_na()

  exampleData <- df %>%
    mutate(date = as.character(date)) %>%
    filter(stockName != '^IRX')

  df <- df %>% select(-stockPrice)

# Create Avg. Return Table (by Stock)
  df.return <- as.data.frame(df %>%
    filter(stockName != '^IRX') %>%
    group_by(stockName) %>%
    summarise(avgMonthlyReturn = mean(stockReturn)))
    rownames(df.return) = df.return$stockName
  df.return <- t(data.matrix(df.return %>%
    select(-stockName)))
```

```
  df <- pivot_wider(df,
                    names_from = stockName,
                    values_from = stockReturn)

  # Create Matrix for Var-Cov Matrix Calculation
  df.matrix      <- data.matrix(df %>%
                                  select(-date) %>%
                                  drop_na())

# Calculate Excess Returns
  df.excess <-  df.matrix[,2:NCOL(df.matrix)] - df.matrix[,1] # Stock Pric

# Clean up Some Tables for Below Calculatons
  df <- df %>% select(-'^IRX')
  df.matrix      <- data.matrix(df %>%
                                  select(-date) %>%
                                  drop_na())

  stockList <- as.data.frame(stockList) #Needs to be other object on orig
  stockList <- t(df.matrix)

# MAIN ------------------------------------------------------------------
```

```
# Create Variance-Covariance Matrix
    VarCovMatrix  <- (t(df.excess) %*% df.excess) / NROW(stockList) # Cre

# Optimize Risk, Given desired Expected Return
    objectiveFun      <- rep(1,nrow(VarCovMatrix)) %*% sqrt(VarCovMatrix)
    constraintInputs  <- rbind(rep(1, nrow(VarCovMatrix)), as.vector(df.r
    direction         <- c("==", ">=")
    constraintValues  <- c(1, desiredReturn)

    optimalWeights    <- Rglpk_solve_LP(objectiveFun,
                                        constraintInputs,
                                        direction,
                                        constraintValues,
                                        max = FALSE)

    portfolioOptimal <- cbind(as.data.frame(optimalWeights$solution),
                              as.data.frame(optimalWeights$solution) * do
      colnames(portfolioOptimal)  <- c("Stock Weight", "Dollar Investment
      rownames(portfolioOptimal)  <- rownames(stockList)

# Calculate Risk of Porfolio
    risk              <- sqrt(12) * sqrt((t(portfolioOptimal$`Stock Weight`)
```

```
# Calculate Expected Return of Portfolio
    expectedReturn  <- 12 * (df.return %*% portfolioOptimal$`Stock Weight

# Calculate Sharpe Ratio for Portfolio
    sharpeRatio     <- (expectedReturn - riskFreeRate) / risk

# Summary Table
    summaryTable <- as.data.frame(rbind(risk,expectedReturn, sharpeRatio))
      colnames(summaryTable) <- c("Value")
      rownames(summaryTable) <- c("Risk", "Expected Return", "Sharpe Rat

    portfolioOptimal <- portfolioOptimal %>%
                        mutate("Stock Name" = rownames(portfolioOptimal))
                        select("Stock Name", "Stock Weight", "Dollar Inves
                        filter(portfolioOptimal[1] > 0)

    cat("Total Investment:",dollarsInvested, sep = " ")
    print(portfolioOptimal)
    print(summaryTable)

# VISUALIZATIONS ----------------------------------------------------------
```

```
# Example Data
  xtable(exampleData)

# Variance-Covariance Matrix Example
  xtable(VarCovMatrix)

# Optimized Portfolio Stats
  xtable(summaryTable)
  xtable(portfolioOptimal)

# Monthly Returns by Stock
  ggplot(data = exampleData, aes(
    x = date,
    y = stockReturn,
    color = stockName)) +
    geom_point() +
    theme_minimal() +
    labs(title = "Stock Returns",
         subtitle = "",
         caption = "Data from Yahoo Finance.",
         tag = "Figure 1",
         x = "Date",
```

```
y = "Monthly Return")
```