OPER 527 Quadratic Programming Assignment

Christopher Flippen

November 10, 2022

In this assignment, we use quadratic programming in R to determine an optimal allocation of money into a set of stocks on the New York Stock Exchange. In order to diversify our portfolio, we select 25 securities, 5 in technology, 5 in retail, 5 in finance, and 10 other miscellaneous securities. We are not allowed to put more than 20% of our portfolio into any of the first three categories.

Our technology securities are: Qualcomm (QCOM), Zoom (ZM), Nvidia (NVDA), Intel (INTC), and IBM (IBM).

Our retail securities are: Amazon (AMZN), Costco (COST), Kroger (KR), Dick's Sporting Goods (DKS), and CVS (CVS).

Our finance securities are: Citigroup (C), PayPal (PYPL), Goldman Sachs (GS), Visa (V), and JP Morgan (JPM).

Our miscellaneous securities are: Pfizer (PFE), Hershey Chocolate (HSY), John Deere (DE), Tootsie Roll Industries (TR), Kellogg's (K), New York Times (NYT), PepsiCo (PEP), Procter and Gamble (PG), Stanley Black and Decker (SWK), Tyson Foods (TSN).

When we represent these stocks in R, they are indices of a vector x where each x_i represents the proportion of our portfolio we invest in stock i. The x_i's are: x_1 is Qualcomm, x_2 is Zoom, x_3 is Intel, x_4 is Nvidia, x_5 is IBM, x_6 is Amazon, x_7 is Costco, x_8 is Kroger, x_9 is Dick's Sporting Goods, x_10 is CVS, x_{11} is Citigroup, x_{12} is PayPal, x_{13} is Goldman Sachs, x_{14} is Visa, x_{15} is JP Morgan, x_{16} is Pfizer, x_{17} is Hershey Chocolate, x_{18} is John Deere, x_{19} is Tootsie Roll Industries, x_{20} is Kellogg's, x_{21} is the New York Times, x_{22} is PepsiCo, x_{23} is Proctor and Gamble, x_{24} is Stanley Black and Decker, and x_{25} is Tyson Foods.

In R, we used the `getSymbols` command to get the daily closing values of each stock for each day since 2007. If the stock is newer than 2007, then the starting date is when the stock first became available. From these data, we computed the daily returns and the average returns. The daily returns were used with `covReturns` to compute the covariance matrix. The covariance matrix is used to determine the risk of the investment. The list of average returns is used to determine the anticipated return of the investment.

Our goal with finding the optimal allocation of our portfolio becomes a quadratic programming problem:

$$\max\left(x'Dx + d'x\right) \text{ subject to } Ax \geq b.$$

In this formulation, D is the covariance matrix, d is unused so it is the zero vector, and A and b are explained on the next page.

In order to determine the highest possible rate of return, we tried several values of r^*. If the value was too large, then the R code will say that we have inconsistent constraints. For our selected set of stocks and constraints, **the highest rate of return we have is 0.00099. (0.099%)**. **The risk associated with this set of investments is 0.0001542235**. The rate of return and risk that we obtained show that this investment is very safe but also provides a very small rate of return. After running our code, with this goal rate of return, we obtained:

$$x = \begin{pmatrix} 6.016780 * 10^{-17} \\ 3.341492 * 10^{-2} \\ 1.665851 * 10^{-1} \\ 1.901170 * 10^{-16} \\ -1.647046 * 10^{-17} \\ 2.000000 * 10^{-1} \\ 2.192605 * 10^{-17} \\ 4.139869 * 10^{-17} \\ 7.384219 * 10^{-17} \\ 6.687690 * 10^{-17} \\ -2.247953 * 10^{-16} \\ 1.253000 * 10^{-16} \\ -1.124901 * 10^{-16} \\ 2.000000 * 10^{-1} \\ 1.420380 * 10^{-16} \\ 5.532888 * 10^{-17} \\ 0 \\ 4.000000 * 10^{-1} \\ 1.750173 * 10^{-16} \\ 9.380634 * 10^{-17} \\ -1.182256 * 10^{-16} \\ -3.453546 * 10^{-16} \\ -2.870061 * 10^{-16} \\ 2.660193 * 10^{-16} \\ 0 \end{pmatrix}$$

This initial set of values has some extremely small negative values. These negatives are a result of the finite machine precision associated with the numerical method used to perform the quadratic programming. These values are so small that even if we were investing millions of dollars, they would still impact the investment by less than one penny. However, we still want to have positive allocations of our money, so we want to remove the negative values. We do this by finding the most negative value, $-3.453546 * 10^{-16}$ and add its absolute value to each x_i. Now each x_i is non-negative, but we need the sum to still be equal to 1, so we divide the resulting vector by the norm to "rescale" the values of x_i back to where they sum to 1. Since the number we added was so small, the updated x_i's look nearly identical aside from no longer containing any negative values.

The new x vector is

$$x = \begin{pmatrix} 4.055224 * 10^{-16} \\ 3.341492 * 10^{-2} \\ 1.665851 * 10^{-1} \\ 5.354716 * 10^{-16} \\ 3.288841 * 10^{-16} \\ 2.000000 * 10^{-1} \\ 3.672806 * 10^{-16} \\ 3.867533 * 10^{-16} \\ 4.191968 * 10^{-16} \\ 4.122315 * 10^{-16} \\ 1.205593 * 10^{-16} \\ 4.706546 * 10^{-16} \\ 2.328645 * 10^{-16} \\ 2.000000 * 10^{-1} \\ 4.873926 * 10^{-16} \\ 4.006835 * 10^{-16} \\ 3.453546 * 10^{-16} \\ 4.000000 * 10^{-1} \\ 5.203719 * 10^{-16} \\ 4.391609 * 10^{-16} \\ 2.271290 * 10^{-16} \\ 0 \\ 5.834849 * 10^{-17} \\ 6.113738 * 10^{-16} \\ 3.453546 * 10^{-16} \end{pmatrix}$$

This vector tells us that we should invest about 3.3415% of our portfolio into stock 2 (Zoom), about 16.6585% of our portfolio into stock 3 (Intel), about 20% of our portfolio into stock 6 (Amazon), about 20% of our portfolio into stock 14 (Visa), and about 40% of our portfolio into stock 18 (John Deere). The proportions for the remaining stocks are so close to 0 that it would not be worth investing in them aside from possibly a few cents.