MGMTMSA 403: Optimization

Assignment 2: Portfolio Optimization

Due on Sunday, November 29th, at 11:59pm on CCLE.

Background

The file `Prices.csv` contains *monthly* prices for 390 stocks, collected over a 5-year period. Each column of the file corresponds to one equity. The first row of the file contains the *ticker* of each equity, which is a 3-4 letter that identifies the company. For example, the ticker for Microsoft is 'MSFT'.

This assignment will require you to formulate and solve three different portfolio optimization models. In order to formulate the models, you will first have to compute the monthly *returns* for each equity, as well as the covariance matrix. Let $t = 1, \ldots, 60$ index the months over the 5-year period, and let $Price_t$ be the price of an equity in period t. The average monthly return of the equity (in %) in period t can then be calculated as

$$Return_t = \frac{Price_{t+1} - Price_t}{Price_t} \times 100\%.$$

Note that the covariance of a matrix M can be calculated using the function `numpy.cov(M)` from the NumPy package.

HINT: You may use the Assignment 2 Preprocessing file posted on CCLE to compute the average returns and covariance information.

Models

The description of each model is given below.

Model 1. Start by focusing on a four-asset portfolio: Suppose you can only invest in Microsoft (MSFT), Goldman Sachs (GS), Proctor & Gamble (PG), and U.S. Treasury Bonds (SCHP). Construct a minimum-variance portfolio with an expected monthly return of at least 0.5%.

Model 2. Now suppose you can invest in all 390 stocks. Construct a minimum-variance portfolio with an expected monthly return of at least 0.5%.

Model 3. In practice, there are transaction fees associated with buying stocks. One way of keeping transaction fees low while still attaining desirable performance is to limit the total number of stocks that are purchased (i.e. limit the number of stocks that have a strictly positive weight). Construct a minimum-variance portfolio that selects *at most 4* of the 390 stocks, and has an expected monthly return of at least 0.5%. (Note: By introducing binary variables into a quadratic program, we obtain a **quadratic integer program**. Fortunately, this particular quadratic integer program can be solved by Gurobi.)

Questions

1. Formulate and solve each of the three models in Python, and then answer the following questions. For each part, also include your Gurobi output.

   a) For **Model 1**, write down the optimal risk (i.e. the optimal objective function value), solver time, and the weight on each of the four stocks.

   b) For **Model 2**, write down the optimal risk and solver time.

   c) For **Model 3**, report the optimal risk, solver time, and the ticker and weight on each of the four stocks selected by the model.

2. Use your solution to Question 1 above to answer the following questions:

   a) Is the optimal risk in **Model 1** higher or lower than **Model 2**? Explain why in 1-2 sentences.

   b) Is the optimal risk in **Model 2** higher or lower than **Model 3**? Explain why in 1-2 sentences.

3. In some cases, we may want to get an approximate solution quickly by terminating the branch-and-bound algorithm *before* it finds an optimal solution. There are two ways to terminate Gurobi early: (a) by setting a maximum time limit, and (b) by setting a maximum acceptable optimality gap (the *tolerance*). Use **Model 3** to answer the following two questions. For each part, also include your Gurobi output.

   a) Set Gurobi to terminate after 30 seconds by including `XYZ.Params.TimeLimit = 30.0` in your code for **Model 3**, where 'XYZ' is the name of your model. How does the objective function value at termination compare the optimal value obtained in question 1c)?

   b) Set Gurobi to terminate after reaching a gap of 10% by including `XYZ.Params.MIPGap = 0.1` in your code for **Model 3**, where 'XYZ' is the name of your model. (Note: The default gap in Gurobi is 0.0001.) How does the solver time compare with the solution time obtained in question 1c)?