Stochastic Control and Optimization
Project 3 – Non-Linear Programming – Finance

Deliverables

One RMarkdown (.Rmd) file and one HTML file, submitted to Canvas. Your report should go into some detail about how you solved the problem, include some graphs that explain your results, and include relevant code chunks in the final output. 66% of your grade will be based on whether you get the problem right or not, the remaining 34% will be based on the quality of the presentation of your analysis. We will re-run your code with a new data set. If you don't get the right answer or your Rmd file doesn't run, we will go through your code and give partial credit accordingly. The easier it is to read your code the easier it is for us to understand what you're doing, so use a lot of comments in your code!

Problem Overview

One of the biggest problems with Markowitz portfolio optimization is that it relies on the assumption that the covariance matrix and mean return vector (of ***future*** asset returns) are known. This assumption is often overlooked by estimating the covariance matrix and mean return vector using historical/***past*** data. But this leaves many questions unanswered. Is the future the same as the past? If so, how much data should you use? What frequency of data should you use? Is data from 10 years ago relevant to tomorrow's market? Unfortunately, there are no clear-cut answers to these questions. On top of this, small statistical errors in the covariance matrix or mean return vector can lead to 'optimal' portfolios with very poor performance on new data. These problems are common not only in portfolio optimization, but all optimization problems where you use some data sample to fit a model and then optimize that model for decision making.

One proposed solution to these problems is called Distributionally Robust Optimization (DRO). The premise of DRO is that if you know that fitting a model can lead to error then you should not fit the model to just the data that you observe, but also all data sets that are 'close' to your data set; where 'close' is defined in some abstract way. Specifically, you should look for an optimal decision that performs as well as possible on all of the 'close' data sets. This way, when new data comes in, as long as it is 'close' to the original data set then your decisions will still have good performance. The exact definition of 'close' is beyond the scope of this project.

In the example of portfolio optimization, we want to find a stock weight vector that has low variance not just for the covariance matrix and mean return vector that we observe in the historical data but also is as low as possible for all covariance matrices and mean return vectors that we could possibly observe from other data sets that are 'close' to the actual data. Therefore, if next year's data is similar to this year's data then we will still have pretty low variance. The variance won't be as low as if we knew what the real covariance matrix and mean return vector for next year were going to be, but we're never going to know those anyway, so let's hedge our bets!

In this project you will test DRO portfolios using the same NASDAQ data as you used in project 2.

DRO Portfolio Optimization

There are many ways to formulate distributionally robust portfolio optimization. One of the more popular formulations is based on using the 'Wasserstein distance' to define the closeness of two datasets. We say the 'distance' between the observed data set, D_0, and some other data set, D, is $d(D_0, D)$, where $d(D_0, D_0) = 0$. Then the goal of DRO portfolio optimization is to find a portfolio that makes the variance as small as possible for all data sets that satisfy $d(D_0, D) \leq \delta$, where δ is some number that quantifies how robust we want to be. Larger values of δ will be more robust, meaning data

sets further from the historical data will still have decent variance, but this comes at the cost of not performing as well on new data if that data is actually pretty 'close' to the historical data. This all sounds very abstract, but it will become clearer soon.

This was a very difficult problem to solve for a very long time, but it turns out that in the past 5-10 years there have been some major breakthroughs in this area and the problem is actually kind of easy to solve now. Professor Rui Gao in the IROM department is one of the people that has made some of these breakthroughs. After some complicated math it can be shown that solving the DRO problem is equivalent to solving the following NLP:

$$\min_{w} \sqrt{w^T \Sigma_0 w} + \sqrt{\delta \sum_{i=1}^{n} w_i^2}$$

such that

$$\bar{r}_0^T w - \sqrt{\delta \sum_{i=1}^{n} w_i^2} \geq R$$

$$1^T w = 1$$

$$w \geq 0,$$

where Σ_0 is the covariance matrix calculated from the observed historical data set, $\bar{r}_0$ is the mean return vector calculated from the observed historical data, R is our portfolio's mean return threshold, and n is the number of stocks in the data set (100 in our NASDAQ example). Another lucky turn of fate is that this is a convex optimization problem, so we can just plug it right into any NLP solver (this is NOT a QP) and know that the returned locally optimal portfolio is also globally optimal.

To solve this problem, you need to take just a few steps

- Get a historical data set of asset returns
- Calculate the covariance matrix, Σ_0, and mean return vector, $\bar{r}_0$
- Decide what minimum return threshold, R, you want to meet
- Decide how robust you want your solution to be, δ
- Plug this all into an NLP solver to get your optimal portfolio

The most important part is to understand how to pick δ. Exploring the choice of δ will be a large component of this project.

Picking δ

If we were only concerned about the performance of the model on historical data then we would want to set $\delta = 0$, but in the financial world nobody cares how your portfolio would have done looking backwards, we want to know how well it will do looking forwards!

The standard way of answering this question is with a 'back-test.' The idea of a back-test is to pretend that you are in the past, for example July 1, 2019, and use old data to fit your model, for example data from Jan 1 – June 30, 2019, and then examine the performance of that model on the pretend future, for example July 1-31, 2019. And then you repeat this on a new pretend present date, for example August 1, 2019, then September 1, 2019, …. You then aggregate all your out-of-sample analyses to quantify how well your portfolio construction strategy would have worked if you were doing it for real.

You are going to run a back test of the DRO portfolio optimization problem for several values of δ and see which value of δ has the best performance over the time frame of the back-test. You will compare this with the traditional Markowitz mean-variance portfolio optimization. You will also compare to the performance of the index.

Specifics

1) You will use the same data set as in project 2. This time you need to paste the two sets of data together to get one big data set. Be sure to do this in your Rmd file, so that the input files stay the same… That is, don't create a new csv file that has the data pasted together; load both data sets and then use rbind to combine them in R. There are 504 days of stock prices in this data, leading to 503 days of returns.
2) This step explains how to do the back-test for given values of R, δ. The next steps will explain which values of R, δ to use. For a given value of R, δ, use days 1 – 200 to fit the covariance matrix and mean return vector. Find the DRO optimal portfolio by solving the above NLP. Calculate the variance of that portfolio on days 201 – 220 (out-of-sample). Now use days 21-220 to fit the covariance matrix and mean return vectors, and calculate the DRO optimal portfolio. Calculate the variance of that portfolio on days 221-241, … In the last time interval you will calculate the out-of-sample variance using 23 days of data instead of 20, but that's ok. This will lead to 15 calculations of out-of-sample variances. Average those variances. We want to find the value of δ that makes this average as small as possible. While you're doing this, also calculate the out-of-sample mean return for each of the 12 time periods and average those 12 values. You will use this later.
3) You will need to repeat step 2 for $\delta = 10^{-8}, 10^{-7.8}, 10^{-7.6}, \dots, 10^{-3}$. This is 26 values of δ. You will first do this for the for the NLP that does not use the mean return constraint; just don't include that constraint in problem formulation. For this you're just looking for the minimum variance portfolio, without caring about the mean return. Find the value of δ, from above, that makes the average out-of-sample variance as small as possible. Be sure to include a graph of δ vs this out-of-sample average variance.
    a. It is important to note that you are looking for 1 value of δ that does as well as possible for all time periods. You are not getting the best δ for each time period individually and then averaging the variance using 12 different values of δ.
4) Take the DRO robust portfolio that corresponds to the best value of δ from step 3, without a mean return constraint, and call these $\widehat{w}$ and $\hat{\delta}$. Set R_0 to be $\bar{r}_0^T \widehat{w} - \sqrt{\hat{\delta} \sum_{i=1}^{n} \widehat{w}_i^2}$. Go back and

redo the problem with R equal to 1.05 * R_0. Also set R equal to 1.1, 1.15, 1.2, 1.25 times R_0, to make a DRO out-of-sample efficient frontier. For each value of R, find a new best value of δ . Remember efficient frontiers use standard deviation, not variance, on the x-axis, so just take the square root of your average variance. Be sure that the y-axis on your efficient frontier is the actually calculated out-of-sample average mean return, which won't be exactly equal to R.

   a. For some values of R, if δ is too large the problem becomes infeasible. If you find that for a specific value of R, δ you cannot solve the problem, then you need not try to solve it for larger values of δ for that R.

5) Now using those values of R, compare the portfolio that uses the best value of δ to the Markowitz portfolio. That is, make an out-of-sample efficient frontier for the Markowitz portfolio and compare it to the efficient frontier from step (4). Also, incorporate a point on your efficient frontier for the performance of the actual index. Calculate where it goes in the same way you calculated the average mean returns and average variance of returns in the appropriate time periods.
6) Pretend you are an analyst at an actively managed mutual fund. Your boss has heard of DRO but doesn't know all the details. Your team has been asked to write a report about the effectiveness of Wasserstein DRO portfolio optimization in comparison to Markowitz portfolio optimization. Write this project as if this is what you're going to deliver to your boss. Your boss is pretty technical and understands optimization, so don't be afraid to include quantitative material. Your boss is also busy, so be sure to include some visualizations to get the important points across. For the purpose of your report, you can assume that your boss is interested in the data posted with the project.

Notes

- This is a little computationally time consuming; for a single value of R it took my laptop (a MacBook Air) about 5 minutes to try all the values of δ. I would suggest you do most of your work creating the code in a regular .R file to make sure everything goes right, and then when you're done and ready to do analysis paste this code into an Rmd file.
- The problem formulation used in this project comes from a paper called "Distributionally Robust Mean-Variance Portfolio Selection with Wasserstein Distances" by Jose Blanchet, Lin Chen and Xun Yu Zhou.