MASTER of SCIENCE PROGRAM
Mathematics in Finance

NEW YORK UNIVERSITY
Courant Institute of Mathematical Sciences

Algorithmic Trading and Quantitative Strategies – First Homework

Petter Kolm and Lee Maclin
Due date: March 8, 2018

General guidelines in regards to homework:

- Please post any questions about homework on the NYU Classes course discussion board so that all students can benefit.
- You may collaborate in groups of up to 3 people when solving the homework.
- *Important: Each student is expected to write up their own solutions in their own words. If you collaborate with others, you must list the names of the other participants on your homework submission to get full credit for your homework.*
- Submitting your homework: Submit all files to NYU Classes. No submission via email, please! Homework needs to be submitted on time for full credit.

Overview

In the first part of this homework you will prepare the TAQ Data used in this class. Each student in the class should make sure they have their own copy of the cleaned data, either on their computer or on an external hard drive.

Tips and suggestions:

- Make sure to use the supplied file “sp500.xlsx” (on NYU Classes) to identify the tickers and other useful information of the companies in the S&P 500 index.
- Use Python for all data “wrangling” and computation; all codes and unit tests are part of the homework deliverables.[1]
- Work with a small piece of data (or “fake data”) when you are developing and testing your codes. Once your codes work, then set up to run the analysis for the whole dataset in question. Some computations may require longer runtimes – so make sure to plan ahead!

[1] Codes need to be delivered with fully functional unit tests.

Part A. Preparing the TAQ Data

Follow these steps to prepare the TAQ Data:

1. Download the TAQ Data from the data disks in the library. You will obtain the trades and quotes on all stock on the NYSE on business days throughout the period June 20, 2007 through September 20, 2007. *Note that while you are welcome to work with all the stocks, for the homework in this course it is sufficient to work with the constituents of the S&P 500 index.*
2. Create a Python method TAQAdjust.py to adjust prices, quotes and number of shares for corporate actions such as stock splits/etc. by using the "Cumulative Factor to Adjust Prices" and "Cumulative Factor to Adjust Shares/Vol" (see, "sp500.xlsx"). Note that if these factors did not change for a particular stock during the period, then no adjustment is necessary.[2]
    a. Print side-by-side graphs of Google and Microsoft before and after adjustments.
    b. What conclusions do you draw from your results?
3. Create a Python method TAQCleaner.py that cleans the adjusted TAQ Data using the procedure described below. Your methods should give you the option to store the cleaned data to files.
    a. Print side-by-side graphs of Google and Microsoft before and after cleaning.
    b. What conclusions do you draw from your results?

Part B. Compute statistics

1. Create a Python method to compute X-minute returns of trades and mid-quotes, where X is arbitrary.
2. Create a Python class called TAQStats.py that calculates basic statistics of the TAQ Data. Basic statistics to calculate for each stock are:
    a. Sample length in days.
    b. Total number of trades, total number of quotes, fraction of trades to quotes.
    c. Based on trade and mid-quote returns: mean return (annualized), median return (annualized), standard deviation of return (annualized), median absolute deviation (annualized), skew, kurtosis, 10 largest returns, 10 smallest returns.
3. Use Google and Microsoft to analyze the impact of the cleaning from part A above. Suggested procedure:

[2] For more information about standard adjustment principles, see https://blog.quandl.com/guide-to-stock-price-calculation.

a. After adjusting for corporate actions but before cleaning, calculate basic statistics for the full sample of each one of the stocks. Produce graphs of the data.
b. Clean the data using the filter using the initial settings suggested below.
c. Recalculate the basic statistics, produce graphs of the data, and compare these with what you got before the cleaning.
d. Use a few different stocks to experiment with different parameter inputs to the filter to see their impact in the difference of the basic statistics (before and after cleaning). What are the "best" cleaning parameters?
e. Repeat this procedure for $X = 10$ seconds, 30 seconds, 1 minute, 5 minutes, 10 minutes, 30 minutes.

Part C. Analysis of autocorrelation
Later in this course we will be constructing covariance matrices to serve as one of the components of our algorithmic trading framework. However, before we can do that, we must determine the frequency at which we should compute returns. For this purpose, we need to better understand the distributional characteristics of high frequency data.

For example, we can compute returns for periods of every ten seconds. This would mean using prices at 9:30:00 AM, 9:30:10 AM, 9:30:20 AM, and so on. Or we could compute returns for periods of every five minutes.

The problem of using trade print prices – the prices of executed trades – to compute returns is that these prices exhibit a *bid-ask bounce*. A trade at the bid is likely to be followed by a trade at the offer and so on. To accurately compute covariances, we will need to sample prices at frequencies that are not affected by the bid-ask bounce.

We can arbitrarily choose a long time frame that would satisfy this objective. However, we also want to maximize the amount of data we use. The approach of using too long a time frame is wasteful, as it will leave us with a smaller number of data points. The idea is to find the shortest time frame over which we see no "undesired statistical behavior."

1. Create Python code that performs this analysis and find the optimal bucket size. (Hint: Using the Ljung-Box test[3], and examine the lag k serial cor-

[3] For this test, see for example https://en.wikipedia.org/wiki/Ljung%E2%80%93Box_test

relations for different return frequencies to determine when the serial correlation is negligible.)

2. Extra credit question: Test trade and mid-quote returns for stationary and the bucket size you determined in #1.

Part D. Mean-variance optimization in Python

1. Install CVXOPT in Python, see http://cvxopt.org/index.html
2. Run the example http://cvxopt.org/examples/book/portfolio.html.
    a. Make sure you understand how to use the optimizer and analyze its outputs. To what tolerance did the optimization converge?
    b. Write down a mathematical description of the problem being solved.
3. Assume you are in a CAPM world. Compute the market portfolio.

Cleaning procedure to use in Part A

We will use a similar approach to that of Brownlees and Gallo (2006) for cleaning the TAQ Data. While not perfect, this is a simple and pretty effective measure for removing outliers.

Let $p_0,\dots,p_I$ be the tick data series of price (or quotes) and $v_0,\dots,v_I$ the number of shares. We will decide whether to retain or remove a price volume pair (p_i, v_i) based on the application of a Bollinger band with a 2 standard deviation threshold and a k-day window. If the condition

$$\left|p_i - \overline{p}_i(k)\right| < 2s_i(k) + \gamma$$

then the price volume pair is retained, otherwise it is removed. $\overline{p}_i(k)$ and $s_i(k)$ refer to k-day rolling window price and standard deviations. The parameter γ is a multiple of the minimum price variation allowed for the stock analyzed, and is used to avoid cases where $s_i(k)$ produced by a series of k equal prices. The choice of the parameters k and γ is crucial to cleaning the raw tick data and will require some adjustment by you. A good starting point is $k = 5$ and $\gamma = 0.0005\overline{p}$. Calibrate the parameters on a few stocks.