Notes and comments for Part C and Part D.

Part C:

First of all, it would help to understand what the Ljung-Box statistic is, how it's calculated and how to interpret it. <https://en.wikipedia.org/wiki/Ljung%E2%80%93Box_test>

Note the H\_0 is telling you the data are i.i.d. and the H\_a is that there is some degree of serial correlation (up to lag k). Also note how Q scales by order N.   
i.e. if the number of observations goes up, you need the square of the autocorrelation to go down by about this rate as well. This is important for later.

What we first want to do is to find the window size at which the serial autocorrelation is minimized, and we wish to do so via the Ljung-Box statistic. However, we need to know what the right value of k, the number of lags, to use in the test.

Plot the square of the autocorrelation against the lag number (Autocorrelation\_graphs) for trades and see how the autocorrelation scales:  
- On small windows, behavior is expected. Large negative correlation for small lag due to bid-ask bounce which slowly decays.  
- Interestingly, on larger scales, we start to see some autocorrelation at medium to larger lags of magnitude around 0.05 to 0.1. Can potentially be explained by seasonality creeping in. e.g. returns at the start of every day correlated with each other.  
- Also note that autocorrelations of around 0.05 to 0.1 are actually very bad in the Ljung-Box (even though they look good) as we need the square of the autocorrelation to drop on the scale of !/N. Solution is to increase the window size to reduce N.  
- This tells us that we want to select lag number k to use in the LB test that’s not too large (or we capture seasonality + increased chance of failing the test) (we know empirically that data that’s too far lagged is in practice not useful in predicting future returns). We want to hit the range that’s on the scale of 600s on the charts, where the lag 1 and lag 2 autocorrelation are of similar scale to the other autocorrelations. Empirically (through lots of trials on my side) lag number k=5 is a good size to use and is not too big.

Run the LB test with k=5. Let’s use a 0.95 confidence interval to reject H\_0. Start with a small window size and gradually increase it. Stop when the p-value is less than 0.95. This is the time we do not reject H\_0 and the data are sufficiently i.i.d. In practice, most of the p-values are very close to 1 because of the reason I have explained with the N and the magnitude of autocorrelation.

(Under MSFT/GOOG\_usethis in the Ljung Box and Stationarity folder) We can see that for GOOG, the optimal window is 13 mins and for MSFT it is 25 mins. This will vary from stock to stock, but in general, an optimal window to use will be of this scale: (~15-20mins) as we need the initial large negative autocorrelation on the graphs to disappear.

Test for Stationarity:

We will use the Augmented Dicky-Fuller test:

<https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test>

This tests if there is a unit root that can be constructed from the lagged data. It produces a statistic that’s negative. The more negative, the more likely the data is non-stationary. The first ADF test is on trades and the second on quotes. Our ADF statistics are very negative, so it’s almost certain that the data is non-stationary at the observed optimal window sizes in both MSFT and GOOG.

Note that there are 3 different types of ADF tests that can be used:

<https://en.wikipedia.org/wiki/Dickey%E2%80%93Fuller_test>

The last one isn’t often used unless the data is suggestive of deterministic trends in the deltas. We need to decide between the first and the second. Typically the second is used if the y\_t themselves are trending in time and the first if they are stationary. The second is the default choice in most ADF tests (same here) and all 3 tests have different distributions for their test statistics.

To conclude: Data is highly correlated relative to the number of obeservations, N. This doesn’t really disappear even at larger window sizes. Data is not stationary: i.e. there is no mean reversion that can be constructed using the past prices of the stock alone. This is in line with our intuition.

Part D:

CVXOPT: The model fitted is in the pictures I’ve sent. Note that they’ve included a risk-free asset that’s uncorrelated with the stocks. I’ll restrict the universe to the 10 largest stocks on the S&P500. Cleaning and adjusting 500 stocks is computationally infeasible.

Next, I pick a window size of 15 mins (i.e. 15 min returns) determined from Part C. Grab the data, feed it into the mean return vector and the covariance matrix in the convex optimizer and it returns the optimal portfolio as a combination of the stocks. In this world, we CANNOT short stocks or have negative cash.

To calculate the mkt portfolio, calculate the return of the portfolio at any std\_dev and subtract the return at std\_dev=0 (the risk-free rate). Divide by the std\_dev to get the sharpe ratio and maximize the sharpe ratio.

We are maximizing returns minus risk\_tolerance\*standard deviation subject to the sum of weights being 0 and all weights positive, for any given risk\_tolerance.

I used a model of the following companies: AAPL, MSFT, AMZN, JPM, JNJ, GOOG,XOM,BAC,WFC, INTC.

All large S&P500 constituents. I used 15 minute returns.

Unfortunately, the mean returns (annualized) over the period were

[-0.21588142 -0.05338182 -0.17470272 -0.08859158 0.13647993 -0.17148711

0.31491572 -0.0931225 0.04362007 -0.02606961 0.0378 ]

Stocks are about as likely to do rise as to fall, as we would expect, since our observation window is small.

XOM did extremely well in the period, so the results are heavily skewed towards it

('Optimal portfolio returns:', 0.3149157221717626)

('Optimal portfolio std dev:', 0.12309788153118852)

Optimal portfolio weights:

[ 4.54e-10]

[ 7.22e-10]

[ 4.83e-10]

[ 5.83e-10]

[ 2.72e-10]

[ 4.81e-10]

[ 1.00e+00]

[ 6.31e-10]

[ 7.98e-10]

[ 7.01e-10]

[ 1.04e-09]

Weights are in order AAPL, MSFT, AMZN, JPM, JNJ, GOOG,XOM,BAC,WFC, INTC, cash.

Will try rerunning with a larger dataset, but cleaning will take a while. Gonna use this result if it doesn’t process in time.