PAIR TRADING STRATEGY: ALGORITHMIC STATISTICAL ARBITRAGE
DUKE COMPUTER SCIENCE 216: EVERYTHING DATA

Anisa Tapia, Apoorv Jha, Eleane Ye, Claire Hutchinson, Samhitha Sunkara
https://microgel.github.io/CS-216-Project/

I. Introduction and Research Questions

Research Question: How to generate profits without exposure to market risks? Using Data Science and Algorithmic Techniques

Our project is based on the concept of pairs trading, a market neutral trading strategy enabling traders to profit from virtually any market conditions: uptrend, downtrend, or sideways movement. This method is especially intriguing in today's market—the extreme volatility of which we saw recently with the GameStop short squeeze. Trading and generating profits in an increasingly saturated market is difficult, so we want to use data science and algorithmic techniques to see if we can generate profits without exposure to market risks. Our goal is thus to algorithmically implement a pairs trading strategy and test if it is profitable.

Pairs trading uses pairs of securities that have some sort of underlying economic link. An example could be two companies that manufacture the same type of product (i.e., same industry), Like Microsoft and Apple. Because of this underlying link, we would expect the spread (ratio or difference in prices) between these two to remain constant with time. However, on occasion there might be a divergence in the spread between these two pairs caused by temporary supply/demand changes, large buy/sell orders for one security, reaction for important news about one of the companies, etc. When there is a temporary divergence between the two securities, i.e. one stock moves up while the other moves down, the pairs trade would be to short the outperforming stock and to long the underperforming one, betting that the "spread" between the two would eventually converge. The idea is that in the long run, the price ratio between two stocks fluctuates less than stocks themselves. So when the price of a particular stock deviates too much from the calculated mean, there is an opportunity for profits as the price will eventually go back to the mean ratio.

I. Summary of Results

We implement a statistical arbitrage trading strategy using a Microsoft Corporation (NASDAQ: MSFT) and Micron Technology, Inc. (NASDAQ: MU) pair chosen on the basis of their historic correlation and cointegration (p value ≤ 0.0005) in the training period (2016 - 2018). We use parameters (short window = 3 days, long window = 38 days, number of standard deviations = 1.0) optimized on our training data to generate trading signals for optimal entry and exit points in order to maximize profits (measured by CAGR). We then use these parameters to test our strategy on out-of-sample data (2018 - 2020).

Our strategy returns an approximate return of 40% Compound Annual Growth Rate (CAGR) during the testing period. Compared to the overall market's performance for the same period, our strategy is highly profitable as the market performance (measured by S&P 500 index) merely returns a 10% CAGR. Since the strategy is beta neutral, we have no (or negligible) effects from directional movements in the market. Moreover, our strategy can be scaled for any testing period and provides an end-to-end algorithmic solution for generating profits in the markets without exposure to market risks.

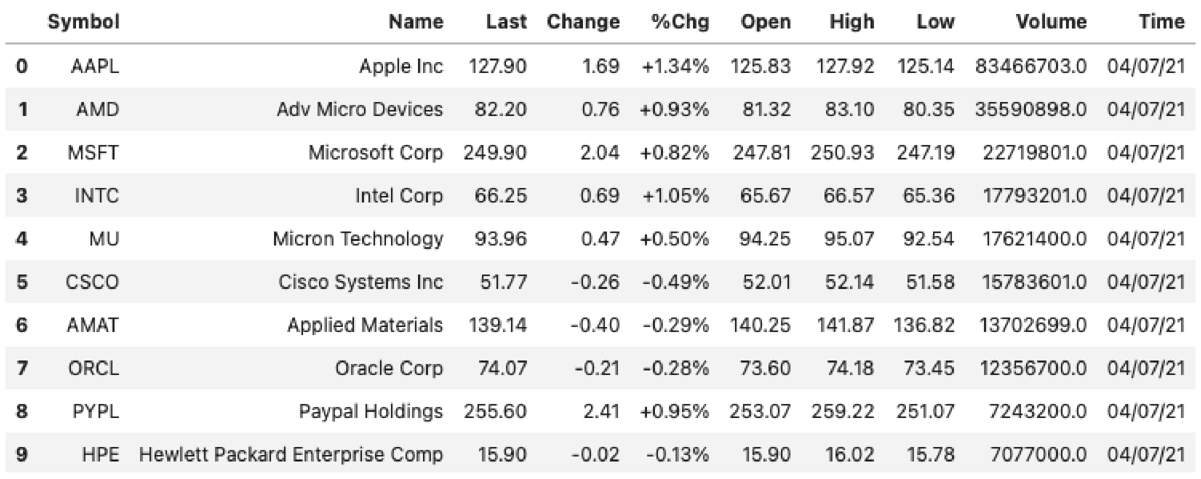
II. Data Sources

We have collected an initial list of all tech industry stocks (US) from Investopedia. For the 10 stocks we chose from the list, we have fetched price data from 2016 to 2020 using Yahoo Finance and Pandas Datareader. We further split the stock price data into two equal parts for in-sample (2016 - 2018) and out-of-sample (2018 - 2020).

The data sources are appropriate since Yahoo Finance and Investopedia have both deep-rooted trust established within the quantitative finance industry. Furthermore, we verified the quality of data ourselves.

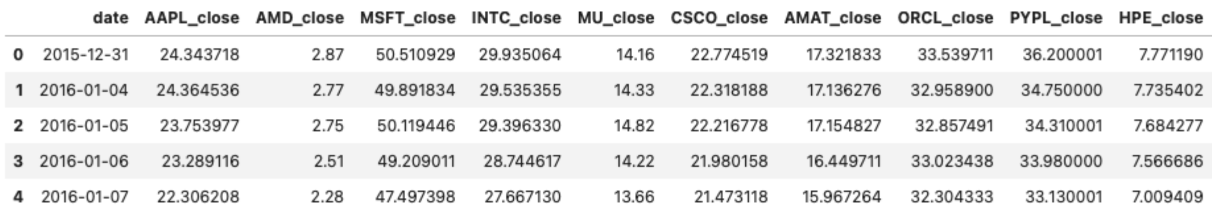
A unique problem with historic stock price data is that stock prices deviate due to stock splits, dividends, etc. In order to mitigate these challenges, we have conducted our in sample testing on close prices of stocks which have been adjusted for stock splits, dividends, mergers, etc.

Table 1: Investopedia – Top 10 U.S. Technology Industry Stocks (ranked by daily trading volume)



	Symbol	Name	Last	Change	%Chg	Open	High	Low	Volume	Time
0	AAPL	Apple Inc	127.90	1.69	+1.34%	125.83	127.92	125.14	83466703.0	04/07/21
1	AMD	Adv Micro Devices	82.20	0.76	+0.93%	81.32	83.10	80.35	35590898.0	04/07/21
2	MSFT	Microsoft Corp	249.90	2.04	+0.82%	247.81	250.93	247.19	22719801.0	04/07/21
3	INTC	Intel Corp	66.25	0.69	+1.05%	65.67	66.57	65.36	17793201.0	04/07/21
4	MU	Micron Technology	93.96	0.47	+0.50%	94.25	95.07	92.54	17621400.0	04/07/21
5	CSCO	Cisco Systems Inc	51.77	-0.26	-0.49%	52.01	52.14	51.58	15783601.0	04/07/21
6	AMAT	Applied Materials	139.14	-0.40	-0.29%	140.25	141.87	136.82	13702699.0	04/07/21
7	ORCL	Oracle Corp	74.07	-0.21	-0.28%	73.60	74.18	73.45	12356700.0	04/07/21
8	PYPL	Paypal Holdings	255.60	2.41	+0.95%	253.07	259.22	251.07	7243200.0	04/07/21
9	HPE	Hewlett Packard Enterprise Comp	15.90	-0.02	-0.13%	15.90	16.02	15.78	7077000.0	04/07/21

Table 2: Yahoo Finance – Stock Price Data fetched for stocks chosen from Table 1 (first 5 rows of training data displayed)



	date	AAPL_close	AMD_close	MSFT_close	INTC_close	MU_close	CSCO_close	AMAT_close	ORCL_close	PYPL_close	HPE_close
0	2015-12-31	24.343718	2.87	50.510929	29.935064	14.16	22.774519	17.321833	33.539711	36.200001	7.771190
1	2016-01-04	24.364536	2.77	49.891834	29.535355	14.33	22.318188	17.136276	32.958900	34.750000	7.735402
2	2016-01-05	23.753977	2.75	50.119446	29.396330	14.82	22.216778	17.154827	32.857491	34.310001	7.684277
3	2016-01-06	23.289116	2.51	49.209011	28.744617	14.22	21.980158	16.449711	33.023438	33.980000	7.566686
4	2016-01-07	22.306208	2.28	47.497398	27.667130	13.66	21.473118	15.967264	32.304333	33.130001	7.009409

III. Results and Methods

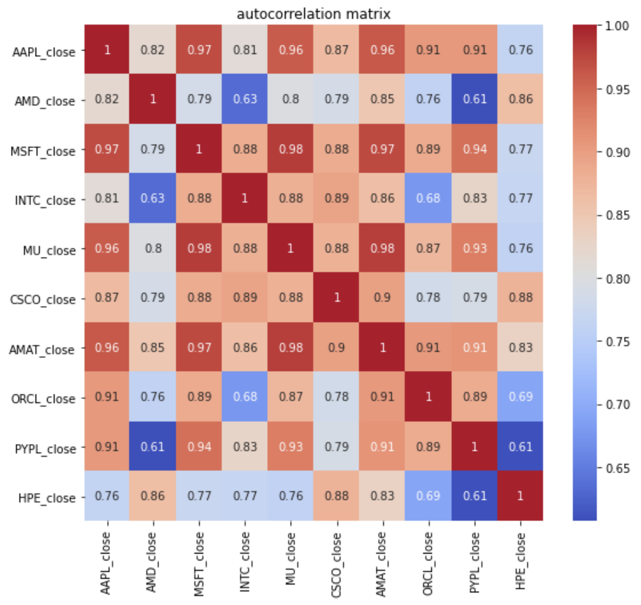
We first researched into pairs trading as well as the market conditions recently that made this topic more timely. From there since we had honed in on the topic we began actually wrangling and implementing our predicted strategy. Using Yahoo Finance and Investopedia we first identified the top 10 US tech stocks by largest daily trading volume. Then, we fetched stock price data for a period of 4 years from 2016 to 2020 (identified in the section above) for each of those stocks. We verified the quality of data (such as checking for null values, etc.) and then moved on to the next step of our project.

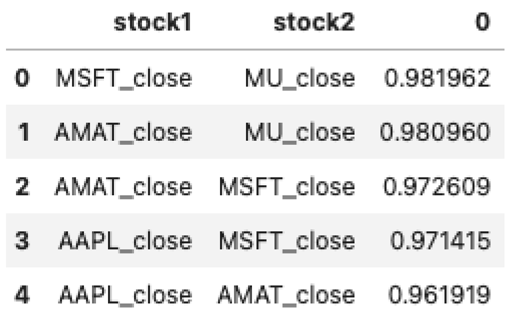
After having collected the data on the pandas datareader, we wanted to ensure that

We combined the data collected into a single data frame which we used to conduct our exploratory data analysis. Since we are essentially betting on the mean reversion of pairs, it is important that we examine the correlation and cointegration for each of the pairs. Correlation is defined by a coefficient that suggests the degree of correlation between two variables. We calculate it using the following formula:

$$Corr(X, Y) = \frac{Cov(X.Y)}{\sigma_x \sigma_y}$$

Now that we defined correlation we needed to understand its implications for our research. A high p value would suggest that there is a strong relationship between two stocks and that given an increase in stock B, the chance of stock A increasing is high. This underlying assumption is key for the market neutral strategy.
Using the pearson method, we computed the correlation and visualized it in the form of a heatmap. Next, we ranked the top 5 pairs based on their correlation score.





	stock1	stock2	0
0	MSFT_close	MU_close	0.981962
1	AMAT_close	MU_close	0.980960
2	AMAT_close	MSFT_close	0.972609
3	AAPL_close	MSFT_close	0.971415
4	AAPL_close	AMAT_close	0.961919

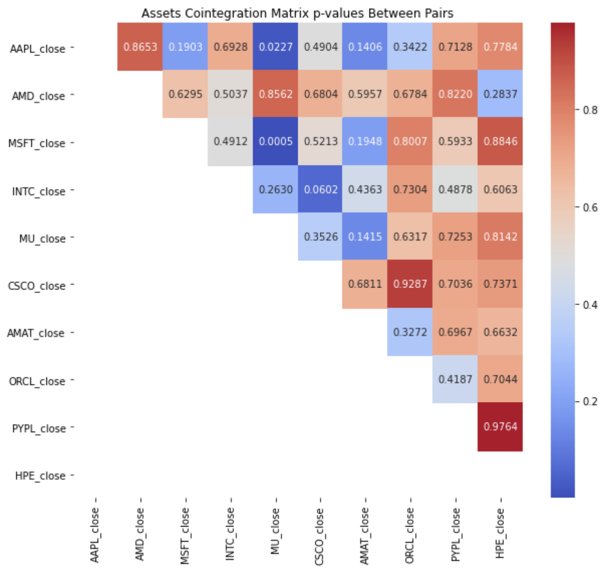
While correlation describes the tendency of the assets to move in similar directions, cointegration describes the distance between the two assets in price over time. Cointegration is also criteria for a pairs trade and is oftentimes the more reliable strategy for successful pairs trading. We calculate it using the following:

If x_t and y_t are non-stationary and Order of Integration d=1, then a linear combination of them must be stationary for some value of β and u_t. In other words:

$$y_t - \beta x_t = u_t$$

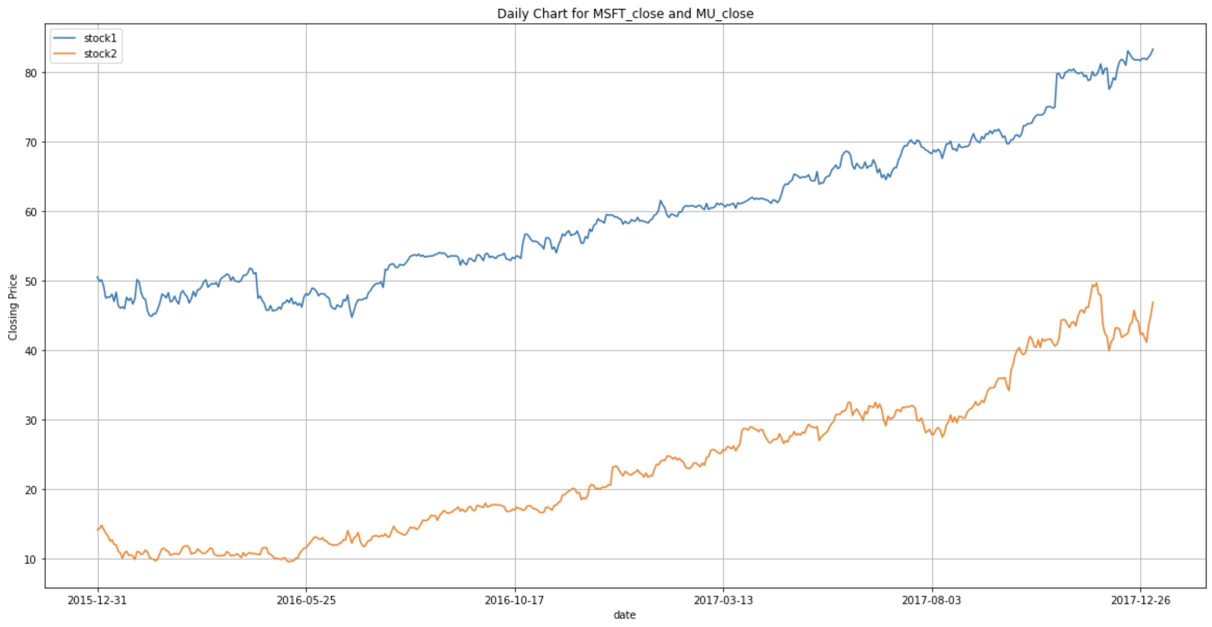
where u_t is stationary.

We conducted a test for cointegration of the pairs and visualized the pairwise p-values.



On the basis of statistical significance of cointegration (p-value ≤ 0.0005) and high correlation, we chose Microsoft Corporation (NASDAQ: MSFT) and Micron Technology, Inc. (NASDAQ: MU) pair for further developing and testing our strategy.

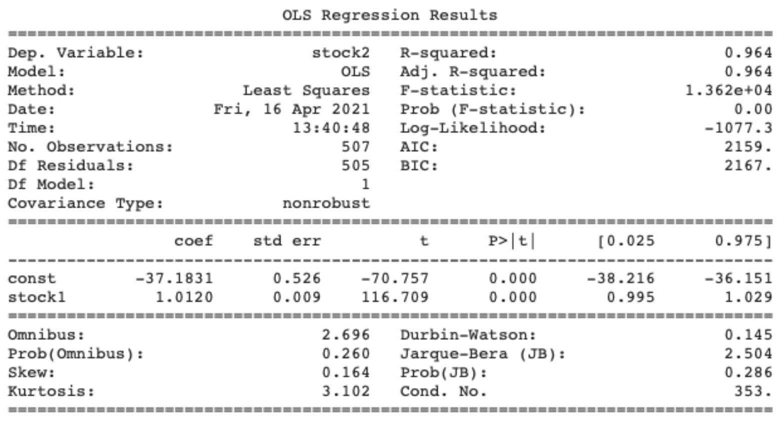
Since it would be inefficient to work with a large data frame in memory, we created a new data frame with just the stock price data for the chosen pair. We visualized the price data to verify if the correlation was indeed true or not.



Next we calculated the spread for the chosen pair. For any pair of stocks, we define spread as

Spread = log(a) – beta * log(b)

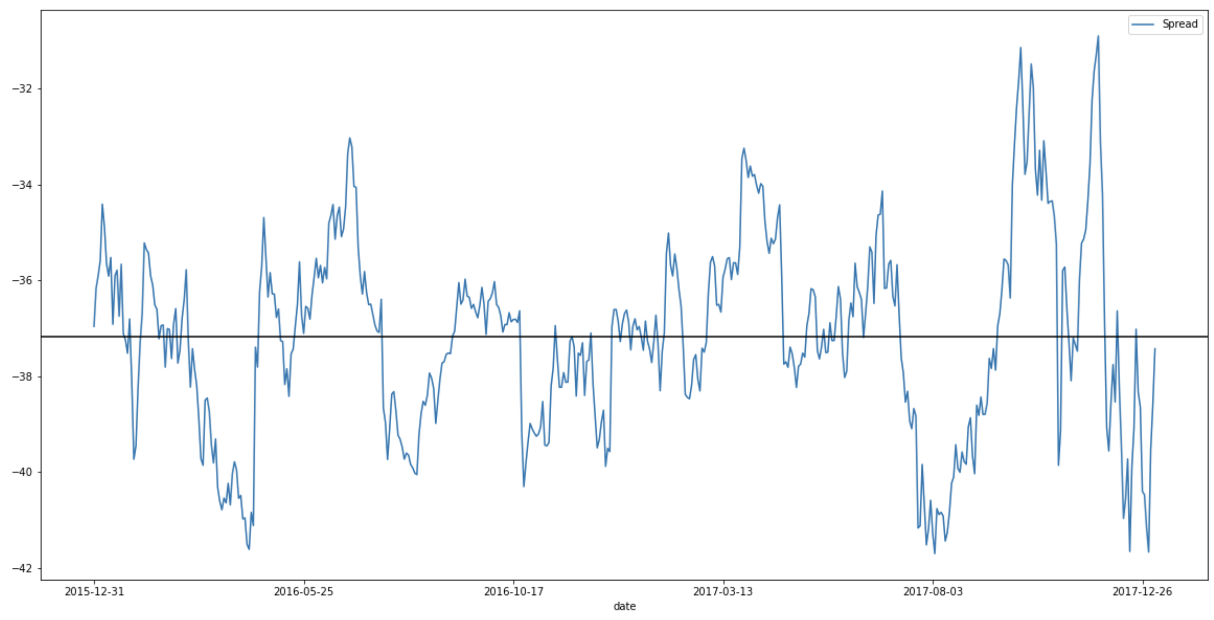
where a and b are the stock prices of stock A and B. These further tests on the spread and mean help us see if prices deviate from an expected stationary value of 0. In order to calculate beta, which is the hedge ratio, we run a linear regression model on the stock prices. Doing so allows us to see the residuals which tell us how much the actual value of 'spread' deviates from 0 for the calculated 'beta'. These residuals are studied so that we understand whether or not they form a trend. If they do not form a trend, that indicates that the spread moves around 0 randomly and is stationary.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 stock2   R-squared:                       0.964
Model:                            OLS   Adj. R-squared:                  0.964
Method:                 Least Squares   F-statistic:                 1.362e+04
Date:                Fri, 16 Apr 2021   Prob (F-statistic):               0.00
Time:                        13:40:48   Log-Likelihood:                -1077.3
No. Observations:                 507   AIC:                             2159.
Df Residuals:                     505   BIC:                             2167.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        -37.1831      0.526    -70.757      0.000     -38.216     -36.151
stock1         1.0120      0.009    116.709      0.000       0.995       1.029
==============================================================================
Omnibus:                        2.696   Durbin-Watson:                   0.145
Prob(Omnibus):                  0.260   Jarque-Bera (JB):                2.504
Skew:                           0.164   Prob(JB):                        0.286
Kurtosis:                       3.102   Cond. No.                         353.
==============================================================================
```

Then, in order to calculate the hedge ratio, we conduct the Augmented Dickey Fuller (ADF) test. ADF is a hypothesis test which gives a p-value as the result. If this value is less than 0.05 or 0.01, we can say with 95% or 99% confidence that the signal is stationary and we can choose this pair.

We can then find the hedge ratio, which is the slope of the linear regression, and use it to calculate and visualize the spread. We can see how our spread moves around a mean:



Next, we compute the price ratio of the pair and visualize it. Using the price ratio, its mean, and its standard deviation, we get the z score statistics, which we further use to define our entry and exit points.

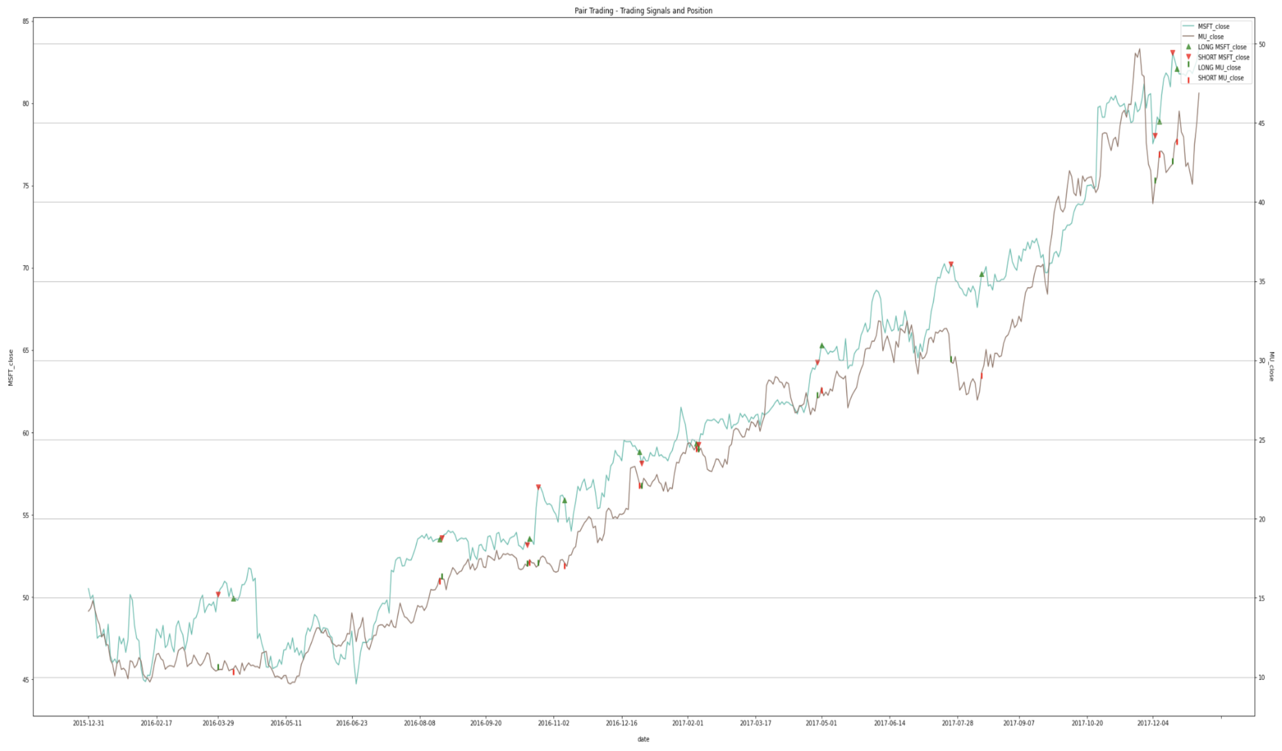
Instead of simply using an overall z score, we give our algorithmic strategy an edge by using a rolling z score. For initial testing, we use arbitrary parameters and get the rolling z score. Having a rolling z score means that our strategy is dynamic and can be deployed for different market conditions. It also gives us the opportunity to tune those parameters which will further make our strategy more profitable.

Using the arbitrary parameters, we develop the backtesting module using pandas and numpy functions. The use of pandas and numpy library enables a much faster computation (since it is implemented in C and much much faster than Python), better and targeted functions for backtesting, and perhaps most importantly, eliminates the usage of any ‘for loops’.

We generate the trading signals as follows:

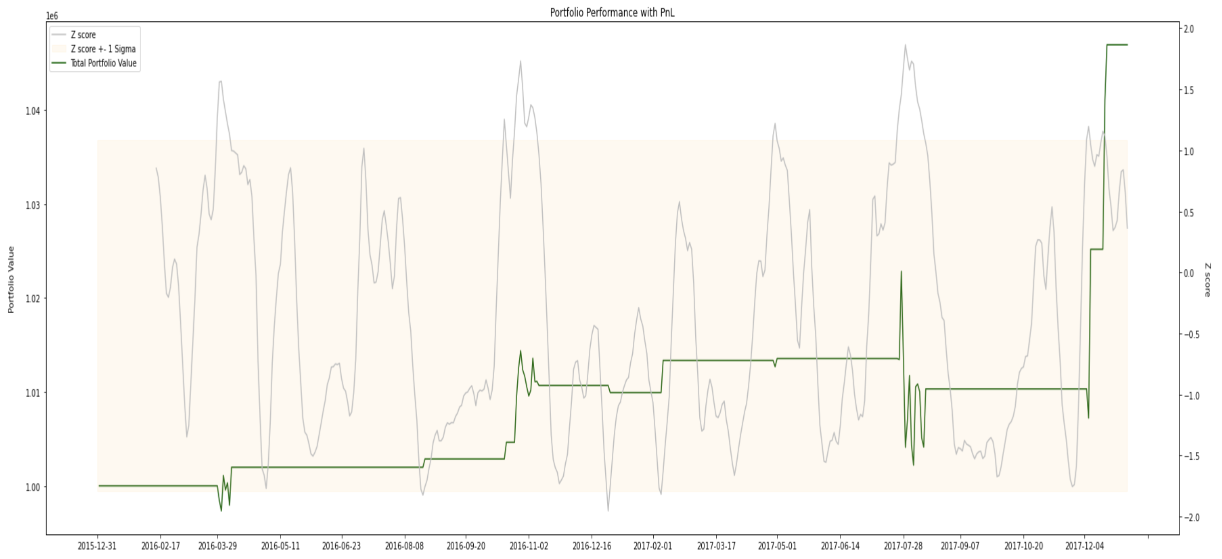
- If z score > upper threshold (for z score), we short the outperforming stock
- If z score < lower threshold (for z score), we long the underperforming stock

Upon backtesting using the arbitrary parameters, the trades take place as follows: (see webpage for better quality figure)



Next, we have to determine the profitability of our strategy, for which we use Compound Annual Growth Rate (CAGR). CAGR is the rate of return that would be required for an investment to grow from its beginning balance to its ending balance, assuming the profits would be reinvested. CAGR is one of the most accurate ways to calculate and determine returns for anything that rises and falls in value.

For the arbitrary parameters on training data, here is how our portfolio value changes:



We get the CAGR to be 44.339%.

We then implement parameter tuning for the parameters affecting entry and exit points for our strategy:

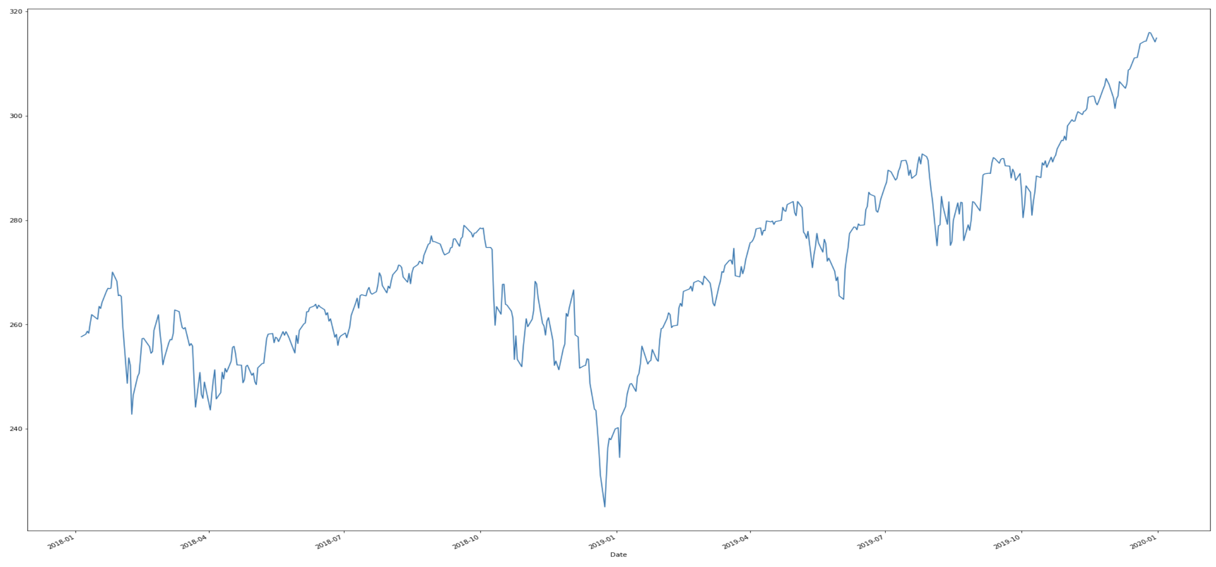
- Short window (for rolling z score)
- Long window (for rolling z score)
- Number of Standard Deviations (for upper and lower thresholds of z score)

Upon parameter tuning, we get the optimal parameters for maximizing CAGR for the training period as follows:

- Short window: 3
- Long window: 38
- Number of standard deviations: 1.0
- CAGR: 54.893814854097144 %

Finally, we use the parameters we get from optimizing on training data to test our strategy on test data. The CAGR for pair trading strategy on test data is 40.68978555534064 % for parameters tuned on training dataset above.

We finally compare our strategy's performance with that of the market performance for the test period. Using the SPY price data, which is the ETF tracking the S&P 500 index, we set it as a benchmark and evaluate the CAGR for holding SPY for the test period. The SPY price moves as follows (in the test period):



CAGR for holding SPY over test period = 10.617%

Our strategy is the clear winner here (~40% CAGR vs 10% CAGR). Moreover, our strategy is beta neutral, and hence, we have no (or negligible) exposure to market risks or any directional movements.

Overall, our research question has been addressed and we have made an end-to-end solution that can be used to backtest the pair trading strategy for any pairs from any industries, given that data is available.

IV. Limitations and Future Work

One limitation is that correlation and cointegration aren't the only methods we can use to pick the pairs of stocks. We can also use DBSCAN or K nearest neighbors to find our pairs of stocks. Using these may have given us different preliminary results of which pair we ended up choosing. If we use DBSCAN or K nearest neighbors in the future, we can easily switch out which two stocks we are using in our pairs trading algorithm in our code.

Another limitation we can address in future work is that we currently are not incorporating risk management techniques. These include maximum drawdown, value at risk, stop loss, and slippage. Maximum drawdown is the maximum observed loss from a peak to a trough of a portfolio, which is an indicator of downside risk over a specified time period. Value at risk estimates how much a set of investments might lose in a set period of time. Stop loss can set buy or sell orders when the price reaches a certain point, with the goal being to limit your loss. Slippage is the difference between where the computer noted we entered or exited the trade versus where the actual trade was entered or exited. We can also look into minimizing transaction costs, which include brokers commissions and spreads. We can also look into Markowitz Portfolio Theory, which can be used to construct a portfolio that maximizes expected returns for a given level of risk. This is done using mean-variance analysis.

There are also limitations to our data and testing. Right now, we are only looking at the close prices, but for example, we can use minute by minute data or second by second data. This would help us get into high frequency trading. Furthermore, we are currently only looking at a small subset of tech stocks. In the future, further research could address questions of how well our pair trading strategy applies to and works for stocks in other industries. For example, asking what pairs of stocks in the healthcare industry tend to converge, and how profits may differ. Applying our project to more data (eg. time increments, different stocks) and doing rigorous backtesting can help extend our results.

LINK TO PRESENTATION VIDEO (IN BOX)