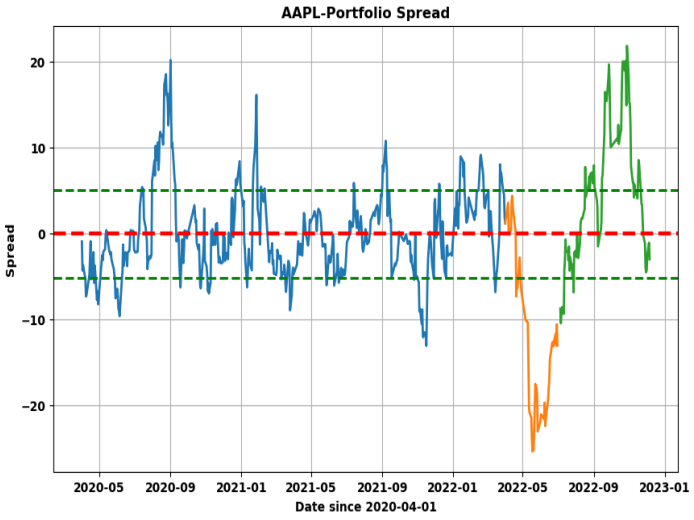

DYNAMIC TRADING STRATEGY DESIGN

ABSTRACT

Designing a Pairs Trading Algorithm, with optimizations from LASSO Regression, followed by retraining after performing feature subset selection/evaluation based on industry profile considerations.

Kartik Balodi

Contents

Foreword/Critique of Previous Model

Based on the previously submitted iteration of the pairs trade strategy, where I used four years of in-sample training period and 6 months out-of-sample for my analysis, I noticed some aspects that needed fine-tuning and wish to bring them to your attention for this analysis:

- In the previous model, in the data pre-processing phase, after removing rows with many missing entries (removing tradeable dates), and then removing columns with many missing entries (removing tradeable stocks if they have many missing entries, especially if these are consecutive/closely clustered), I proceeded to impute the missing dates' information for the remaining stocks. I believe the approximation for dates is largely acceptable for imputing data between in this fashion because after having cleaned my rows/columns for prices missing between consecutive date entries, the approximation for missing price at time t can be reasonably approximated as the date-imputed average of the price at time t-1 and price at time t+1. While an improvement on this could be to add a Gaussian noise term to the average to get a more realistic model of the price with random shocks accounted for, this is unlikely to matter much in our analysis because the time periods in question are simply far too small when taking isolated points of missing data (i.e., approximating P_t based on some kind of average between P_{t-1} and $P_{t+1)}$. While from a mathematical modelling standpoint this argument is reasonable/valid, one key assumption missed here, which has been accounted for in this analysis, is that dates with missing entries might just be the result of the stocks trading on a different foreign exchange. Thus, it would be meaningless to perform any mathematical modelling if it cannot translate into stocks being readily traded on the given date. Thus, here are some fixes to account for the issue of tradeable dates:
    - Acquire more nuanced data from sources of the respective exchanges/3rd parties/firm website and merge/update data of existing columns acquired from Yahoo Finance. While this is the more accurate approach than the next and will add more richness to the analysis, it comes at the cost of a lot more data cleaning and the codebase for merging is likely to be messier.

    - Remove any rows for dates with missing entries, i.e., trade every stock on a level playing field, as only rows with entries on them guarantee that the stock can be traded on their respective exchanges on that trading day. While this approach comes at the expense of the reduction of our tradeable sample space and thus possible opportunities to make more money/reduce losses due to more accurate representation of our models potentially, it is far less costly from a data acquisition and cleaning standpoint, which is my primary constraint right now, seeing that I am lacking the experience/time to acquire and merge more data. Thus, I decided to go with this latter approach. Also, in this study, I elected to use a two-year training period and three-month out of sample period, partially because in the previous analysis I thought a shorter out-of-sample window might be more accurate in predicting trends.
- Previously, I used adjusted closing price in my model estimations and price calculation, here I used closing price. Adjusted closing price can be used to create a model, but it needs to be tested and executed on the opening/spread/closing price, else it's just not accurate for generating real-time stock pricing. This was a mistake in made in the previously submitted version, in this version I have revised that by training and testing both on the closing prices.
- Previously, I lagged price by one day to observe when certain conditions were met to judge a time to enter/exit a trade position between a price time t and the lagged price at t-1. However, I calculated when enter the position at time t itself. This might not be possible realistically, so this time I elect to run the algorithm with a price at time t+1 – i.e., we observe a trend in closing price between t-1 and t and based on that elect to enter/exit/maintain position at time t+1. This is like the approach implemented in de Groot, Huij and Zhou (2012)'s reversal strategy investigation.
- Trading costs have been ignored in both studies, as have margin calls and their impact. A further investigation is needed for verifying both these assumptions to determine the viability of the strategy.

Introduction

Pairs trading involves forming a portfolio using historical data of two related securities that are known to move together and have some sort of long-run relationship, but their relative pricing is away from its equilibrium point at a specific period. The strategy hinges upon the assumption that future trends from that period will preserve the qualities of the long-run relationship, so any deviation from the equilibrium is short-lived and that forms a basis for generating alpha from the market. Thus, we can construct pairs trading portfolio of stocks, using the assumption that the spread, which defined as the difference in price between the paired assets, is mean-reverting, where an investor should take a short position to sell the overvalued asset and a long position to buy the undervalued asset in case of a deviation from the mean of the spread, and as soon as the spread converges back to its mean, the investor should unwind or separate both positions, resulting in a profit (Van der Have, 2017). The challenge of implementing a pairs-trading strategy lies in identifying stocks that are historically highly correlated and tend to move together to make a potential profitable pair. Typically, this is seen in the stocks of companies that are very similar, such as Coca-Cola Co and PepsiCo Inc., or Ford Motors company and General Motors company.

Data, Pre-Processing Steps & Interpretation

I obtained a list of the publicly traded companies around the world ranked by market capitalization from https://companiesmarketcap.com/, after which I downloaded their data from Yahoo Finance using the yfinance package. I downloaded daily closing prices of the top-100 ranked companies with data from the first trading day of 2019 onwards till Dec 6, 2022, and stored it into a DataFrame with the rows corresponding to dates and columns corresponding to each stock. There was a total of 1023 rows of entries, with an average of 34.5 missing data entries per stock, and a maximum of 71 missing entries. Some rows had missing entries for most of the 100 stocks, while the remaining rows had missing entries corresponding to dates missing on another nation's foreign exchange market open/close dates. To ensure a level playing field for comparison across each stock and avoid needing to impute data through interpolation, which might not even be an accurate assumption to make for the daily change in a stock's price from one day to another, I chose to remove all rows/dates with missing closing price for any of the stocks. With that, now we're left with 867 dates from the start of January 2019 to Dec 2022, which I proceed to store into a csv file titled "top100cap_stock_prices.csv".

From the data, the largest company by market capitalization was Apple (AAPL), which then became the first company for a pairs-trade strategy. To identify which stock is appropriate in forming a pairs-trade strategy with AAPL, I compared the correlations between AAPL and other companies in our first training/in-sample period (2019-01-01 to 2021-01-01). The convention according to Google search is that for a successful pairs-trade pair, the correlation between the two assets must be >0.8. After finding a list of companies that matched this criterion, the firm with the largest market capitalization satisfying this requirement turned out to be Microsoft (MSFT). Later, for an extra series of analyses I will use data extracted post-LASSO regression to determine a new potential pairs trade and truncated set of stocks to repeat LASSO regression on. For now, to determine an appropriate in-sample/training period for our analysis, we inspect the price of AAPL against MSFT for 2019-01-01 to 2021-01-01.

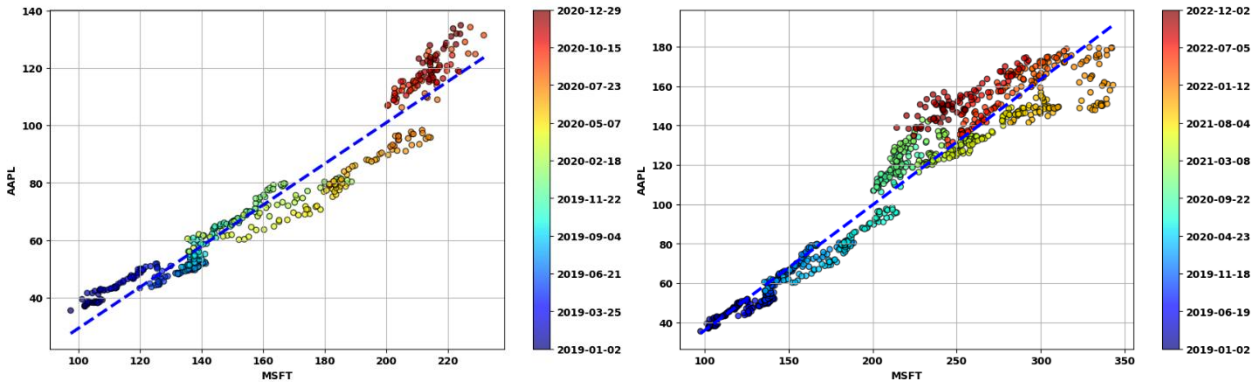

Figure 1 (a) Left: AAPL against MSFT from 2019-01-01 to 2021-01-01. (b) Right: AAPL against MSFT 2019-01-01 to 2022-12-06

Figure 1a highlights the data fits a straight line well from 2019 to early-2020 but seems to fit slightly different gradients better in the second half of 2020. To account for the shift in gradients and the variability in stock pricing as a result, I choose a two-year in-sample period to forecast a three-month out-of-sample period, starting from 2019-01-01 to 2021-01-01 for training and 2021- 01-01 to 2021-04-01 for out-of-sample period. Figure 1b shows the same scatterplot, but for the whole timeframe, i.e., from 2019-01-01 to 2022-12-06. Notice that there is a slight difference in fit with respect to the overall regression line between 2021 and 2022, which further justifies the choice of a two-year training period, and a

relatively short three-month out-of-sample analysis, to keep our model updated for changes in spread/price behavior between both stocks. Analyzing the whole dataset, including the out-of-sample periods here together with the in-sample period does not constitute a look-ahead bias because we observe a clear trend prior to our first out-of-sample period, and are only using the additional data points afterwards to verify and judging from the trend, support our assumption made much earlier.

To capture the changing nature of the stock pair prices, I will retrain our model every three months to update the model for new information about the spread/behavior of the stock prices. Thus, for my pairs trading and LASSO regression optimization, every three months I will shift the training/out-of-sample windows forward by three months, so the second training period is from 2019-04-01 to 2021-04-01, and the second out-of-sample period is consequently from 2021-04-01 to 2021-07-01, and so on till the final out-of-sample period from 2022-10-01 to 2023-01-01 (or present data in our case since the most recent data acquired was till 2022-12-06). In so doing, we successfully incorporate the recent variability in data from the past two years' training period to predict an appropriate spread bound later for deciding when to enter a trading position in the market. I perform a more detailed analysis, demonstrating through correlation plots the breakdown of correlation between the in-sample and out-of-sample period of our analysis as well, and the details of the code for this can be found in the notebook "Data Preprocessing Top 100 Market Cap.ipynb".

Thus, so far, after interpreting our data, we have justified:

- forming a pairs trade strategy between AAPL and MSFT, &
- utilizing a rolling 24-month training period to predict for a three-month out-of-sample period.

Model Choice/Design

First, we run a basic pairs-trading strategy on AAPL/MSFT using ordinary least squares (OLS) regression on the prices of both stocks. Then, I attempt to optimize the performance of our strategy using regularized regression techniques - more specifically LASSO regression. In both cases, we use a 24-month in-sample training period to predict returns of a portfolio for a three-month out-of-sample horizon, after which we update our model by repeating this process by shifting the training/out-of-sample periods by three months.

OLS regression is standard linear regression that people are most familiar with. The OLS loss function aims to optimize the coefficients/betas for predicting a linear relationship between the independent and dependent variables by minimizing the mean squared error between the true and predicted values – in our case the true and predicted stock price. The OLS model (and the LASSO model too) assume noise is distributed randomly on a Gaussian distribution. By constructing OLS regression on the prices of AAPL against MSFT, we can determine the betas; B_0 corresponding to the intercept and B_1 corresponding to the coefficient that is multiplied MSFT to predict AAPL price, and thus come up with an estimate of what AAPL price should be for each given price of MSFT stock on a given day. The deviation of this prediction to the true value of the AAPL stock determines the spread of the stock price between AAPL and MSFT, which becomes our basis for creating trading signals. Based on our methodology, we create a new set of betas for each new non-overlapping three-month out-of-sample period based the trailing 24-months as the training period. All the observed t-stats for the beta coefficients estimated from training data (Figure 2) show a high significance level/low p-value, so we can reject the null hypotheses for each training periods' corresponding betas and note that the coefficients are all statistically significant

in our models trained. The high t-stats imply that the OLS model parameters are highly statistically significant, especially for B1, the coefficient for MSFT when predicting AAPL price. Details of my approach can be found in the Jupyter notebook “pairs_trading_v1_AAPL_MSFT.ipynb”.

Results: Ordinary least squares

Model:	OLS	Adj. R-squared:	0.791
Dependent Variable:	AAPL	AIC:	3227.6246
Date:	2022-12-08 12:04	BIC:	3235.7936
No. Observations:	439	Log-Likelihood:	-1611.8
Df Model:	1	F-statistic:	1656.
Df Residuals:	437	Prob (F-statistic):	9.31e-151
R-squared:	0.791	Scale:	90.903

	Coef.	Std.Err.	t	P>|t|	[0.025	0.975]
MSFT	0.4516	0.0111	40.6994	0.0000	0.4298	0.4735
const	20.1522	2.9384	6.8583	0.0000	14.3771	25.9274

Omnibus:	3.173	Durbin-Watson:	0.043
Prob(Omnibus):	0.205	Jarque-Bera (JB):	3.169
Skew:	-0.173	Prob(JB):	0.205
Kurtosis:	2.768	Condition No.:	1710

* The condition number is large (2e+03). This might indicate strong multicollinearity or other numerical problems.

Figure 2 OLS summary statistics for training period from 2020-07-01 to 2022-07-01 for AAPL against MSFT.

LASSO regression stands for “Least Absolute Shrinkage and Selection Operator” regression. It’s a type of penalized regression that performs feature selection using the L1-norm penalty term, and eliminates the least important variables, (hopefully) minimizing prediction error. In the optimized attempt, I run LASSO regression on the price of AAPL against the price of the remaining 99 large-cap stocks, utilizing the sparsity induced by the L1-regularizer in the LASSO penalty term to generate a curtailed list of stocks for each period. Then, we use the coefficients for betas generated by the LASSO regression model for the in-sample period as weights to long/short the selected stocks to construct a portfolio for each out-of-sample period. Then, we then run a pairs-trading strategy like before, but this time for AAPL stock against the beta-weighted portfolio just formed from the LASSO regression model. For the LASSO regression model, we need to determine the regularization parameter that minimizes the loss function – this is done by ten-fold cross-validation. K-fold cross-validation means splitting the training sample into k folds (equally-sized sections), and then training the model on k-1 folds while testing on the kth omitted fold. This step is repeated k times as each of the k folds is omitted once, and then the regularization parameter which gives the lowest average validation error across the k folds is selected as the regularization parameter for the LASSO regression model. Also, it is important to note here that cross-validating data is okay here because we are not regressing AAPL prices as a function of time, but rather as a function of the other stock’s prices at a series of fixed dates. Details of my approach can be found in the Jupyter notebook “pairs_trading_AAPL_LASSO_v1.ipynb”.

Since this is an arbitrage-based strategy, so technically no cash input is needed to start the strategy until a margin call occurs/loss is booked, the performance of this strategy is evaluated based on how much profit is generated on a $100 trade position for each position taken. We also attempt to model an appropriate time to book a loss for our strategy based on the deviation of the spread from our in-sample/training data. In this study, we simplify our model by ignoring margin calls made between trades. I will also consider the number of trades made, if there is a disproportionate difference in the number of trades made between models. Ideally, fewer trades for the same amount of/more money made is better as it means taking fewer active positions to get the same arbitrage returns, thus minimizing trading costs upon scaling this model towards more pairs/market positions. For the LASSO model, we will also consider

the number and type of stocks pick by the model and attempt to draw some meaningful analysis/conclusions from them.

Next, based on insights from the analysis of features selected by LASSO regression, I will run another OLS pairs trade strategy on AAPL against another stock (Samsung). As I'll discuss later in further detail, for the eight out-of-sample periods between Jan 2021-Dec 2022, Samsung was always a predictor variable for the return of AAPL. Details of my approach can be found in the Jupyter notebook "pairs_trading_v2_postLASSO_feature_selection_AAPL_Samsung.ipynb". Finally, based on the union of the set of predictors generated across each LASSO model and after analyzing this curtailed set of stocks for their industry/sector-relevance to AAPL and removing stocks that are in irrelevant industries to AAPL's performance, I will retrain the model using LASSO regression once more to see if the curtailed set of stocks will generate better performance. This form of feature selection using LASSO can be seen as a proxy to incorporating quantitative approaches with some prior fundamental/industry/economic knowledge, and future extensions of this model could also consider analyzing how this set of stocks evolves with time in greater detail so that future LASSO regression models can be optimized further. Details of my approach can be found in the Jupyter notebook

"pairs_trading_AAPL_LASSO_v2_postLASSO_feature_selection.ipynb".

Finally, in this analysis, LASSO was preferred to alternative dimension reduction techniques like PCA/PCR, because of its interpretability – LASSO regression shrinks and removes uncorrelated variables by setting their betas to zero. In our context, this means retaining only the subset of stocks that behave closely with AAPL, which is what we want because it in theory should create a strong mean-reverting spread if the correlation is true. On the other hand, PCA/PCR picks the vector that maximizes variance, then picks the next vector orthogonal to the first that maximizes variance from the residuals, and so on. In my opinion, this is not very informative, because we will likely end up having many small coefficient values attached to each of the 99 other stocks to predict AAPL stock prices after aggregating the coefficients for every principal component. I feel that this approach is unhelpful to finding meaningful relationship between AAPL and other highly correlated stocks, relative to LASSO.

Trade Signal

The trade signals to enter/exit a position can be summarized by the diagram found at the cover of this report. Figure 3a shows the spread for AAPL against the LASSO-selected portfolio for the training period from 2020-04-01 to 2022-04-1 and an out-of-sample period from 2022-04-01 to 2022-07-01. Here, the training period is reflected by the solid blue line, the out-of-sample period with the solid green line and the model training from the training period is then extended as far out as possible beyond the out-of-sample period (solid green line). The solid green out-of-sample line is only used to close existing positions, but not enter any new ones. To avoid excessive leverage, I decided to only enter a new position after having closed out an old one for the sake of this analysis. This means that if a position is entered in a prior three-month out-of-sample period and has not been exited, we will not invest in the new out-of-sample period (even if there is an opportunity to invest) until we have exited the previous position.

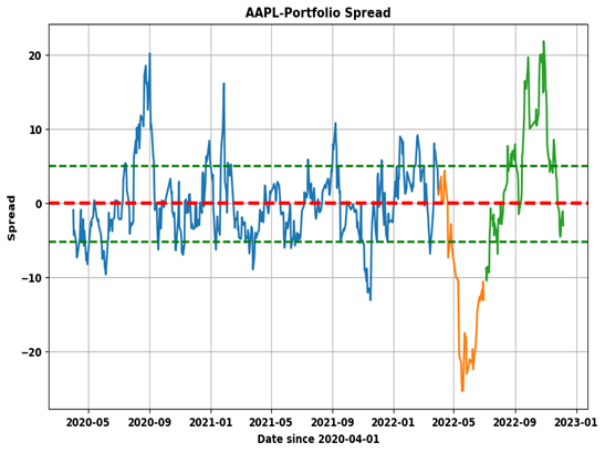

Figure 3 (a) Spread for AAPL against LASSO-selected portfolio for the training period from 2020-04-01 to 2022-04-1

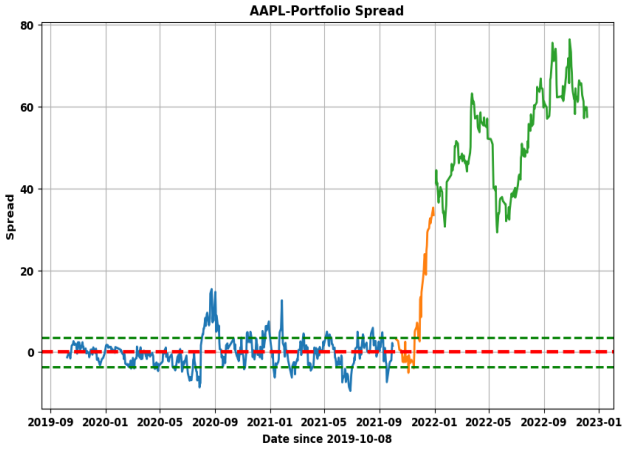

Figure 3 (b) Spread for AAPL against LASSO-selected portfolio for the training period from 2019-10-01 to 2021-10-01

To decide when to enter/exit a position, I make use of the standard deviation of the spread from the 24-month training period, reflected by the green dotted lines on the diagram. I enter a position when the spread in the out-of-sample period exceeds one standard deviation, and revert the position when it returns to zero, assuming everything goes well. However, in this model, we need to also consider an appropriate time to book a loss, as there's no guarantee that our model trained is commensurate with the ground truth – it is entirely possible that the out-of-sample spread can deviate way past our expectations of the spread's standard deviation, as evidenced above in Figure 3b. Again, to estimate this, we look towards the standard deviation calculated from our training data, and the maximum absolute value of the spread observed to determine a ceiling for when to book a loss in the out-of-sample period. For this study, I set an upper limit for the normalized spread by taking highest value observed across training batches for (1 + the maximum absolute spread/standard deviation of spread for the spread of each training period). Mathematically, this can be expressed as follows:

$$Exit\ condition\ \epsilon_t\ for\ training\ sample\ t =$$

$$\frac{\left|spread_{OoS,t}\right|}{stdev(spread_{training,t})} > \max_{i\in\{1,\dots,t\}}\left(1+\left\lceil\frac{\max(\left|spread_{training,i}\right|)}{stdev(spread_{training,i})}\right\rceil\right)$$

This approach includes feedback of historical spread by taking the max over all training samples. This way over time, the model is robust towards periods of low and high volatility, assuming mean-reverting behavior of the spread continues.

Finally, another consideration I made is that after the three-month out-of-sample period, there is no guarantee that the model will continue to accurately represent the reality. In fact, Figure 4 suggests that the range of normalized spread when extending the trained model for the original out-of-sample period (solid orange line in Figure 3) to the most recent date (solid green line in Figure 3) is only somewhat constrained/accurate for around nine months, before it seems to blow out of proportion. Thus, it is important to exit a position at the earliest opportunity presented if it lies outside the out-of-sample period. For this I relaxed the exit criterion where if position is not closed at the end of three-month out-

of-sample period (solid orange lines in above diagrams), maintain position with the old model (solid green lines in above diagrams) until absolute value of normalized spread is within 0.25 with the same old model.[1]

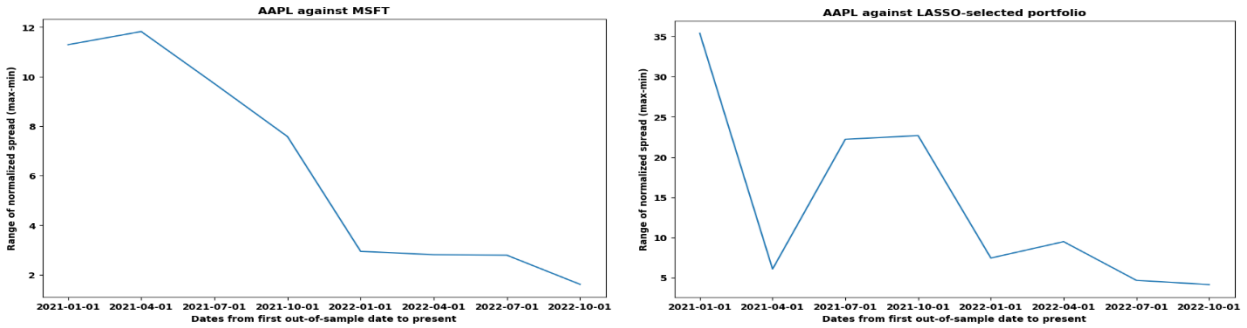

Figure 4 Both diagrams show the range of normalized spread when extending the model from the original training of the out-of-sample period to now. (a) Left: AAPL against MSFT (b) Right: AAPL against LASSO-selected portfolio

With all the justifications and considerations taken into account, my trading strategy can be summarized as follows:

- When normalized spread >+1 and <ϵ_t, short $100 worth of AAPL long $100 worth of MSFT, revert position when normalized spread first becomes a negative number (i.e., reaches 0).
- When the normalized spread <-1 and >-ϵ_t, long $100 worth of AAPL short $100 worth of MSFT, revert position when normalized spread first becomes a positive number (i.e., reaches 0).
- If position is not closed at the end of three-month out-of-sample period (solid orange lines in Figure 2), maintain position with the same model (solid green lines in Figure 2) until absolute value of normalized spread is within 0.25 with the same old model.
- Do not take a new position in an updated model until past position has been exited/reverted.

Results

Like mentioned above, here I will compare the performance of each strategy based on overall profits and number of trades made. For the LASSO model I will also discuss the features selected at the end.

The original OLS model shows five positions taken in the out-of-sample periods, with four of them closed and one still open at the point of analysis. The details of the type of action can be found in Figure 5 below:

[1] Another thing I noticed here is that if the previous out-of-sample model spread is well-above the standard deviation from training, the subsequently trained model corrects for this by shifting that spread value down through adjusting the new betas, and vice versa if the previous out-of-sample model spread is well-below the standard deviation from training. Relaxing the exit condition to 0.25 implicitly accounts for this model update, which adds further credibility to its inclusion. I leave this point as a footnote because it seems to be a very subjective analysis, and more observations are needed before I can make any conclusive statements about this claim.

Date	MSFT	AAPL	action type	MSFT ownership	AAPL ownership	arbitrage returns
2021-01-05	212.250000	126.599998	1.0	0.471143	-0.789889	NaN
2021-02-18	240.970001	129.869995	0.0	0.000000	-0.000000	10.948278
2021-04-07	253.250000	130.360001	-1.0	-0.394867	0.767106	0.000000
2021-08-20	304.649994	149.710007	0.0	0.000000	-0.000000	-5.452633
2021-08-20	304.649994	149.710007	-1.0	-0.328246	0.667958	0.000000
2021-10-28	331.619995	149.800003	0.0	0.000000	-0.000000	-8.792669
2021-10-28	331.619995	149.800003	-1.0	-0.301550	0.667557	0.000000
2022-01-05	313.880005	172.000000	0.0	0.000000	-0.000000	20.169251
2022-01-05	313.880005	172.000000	1.0	0.318593	-0.581395	0.000000

Figure 5 Leftmost two columns represent prices of MSFT/AAPL the day after the date reflected on row index as we trade on the day after the signal is found, action type corresponds to whether to take up a L/S position (1,-1) or revert position (0), MSFT/AAPL ownership columns reflect the amount of stock to Long(positive number)/Short(negative number), and arbitrage returns represents the dollar amount of money earned/lost after reverting a position.

The net profits made here was $16.87 on four trades (with one more position still open) between the start of 2021 up to 2022-12-06. Note that the completed trades were all made in 2021, and the position open at the start of 2022 still hasn't closed. This is consistent with the trends observed so far this year between MSFT and AAPL, where the spread out-of-sample has consistently exceeded the one standard deviation mark predicted by the training samples throughout the year (Figure 6). This could reflect either a changing trend in the relationship between MSFT and AAPL prices or simply be a long-term spread reversion trend due to the ongoing uncertainty in markets because of increasing interest rates and economic conditions. Depending on how this position unravels, it could reveal a further scope for improving the existing trade signal/model – a further iteration of this signal should incorporate some knowledge and use of positions held but not exited for a longer out-of-sample period towards current/future investment positions, instead of the current approach to not invest until one position has closed.

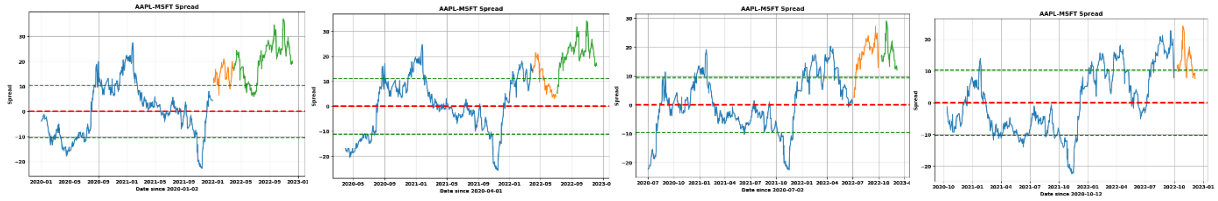

Figure 6 Out-of-sample date ranges (solid orange lines) left to right: (a) 2022-01-01 to 2022-04-01, (b) 2022-04-01 to 2022-07-01, (c) 2022-07-01 to 2022-10-01, (d) 2022-10-01 to 2023-01-01.

The LASSO regression model has a similar table outputted like in Figure 5, but this time the LASSO regression model takes eleven positions in total, with ten closed positions and one still-open position. The profits of the LASSO regression model were far poorer than the OLS model, with a net profit of $1.41 made between 2021-01-01 and 2022-12-06. An explanation for this could be that the feature space/set of stocks chosen for the LASSO regression are not appropriate in modeling the price change of AAPL. Our dataset took the top 100 companies by market cap, which includes firms that have little relevant industry

relationship with Apple. Thus, if they happen to be spuriously well-correlated to AAPL stock prices during our 24-month training period, the LASSO regressor will pick it erroneously and fit to noise/meaningless data, which in turn affects the out-of-sample performance.

On average, the LASSO model chose 6.125 stocks per out-of-sample period and chose from a subset of fourteen stocks across these periods out of the 99 presented to the model initially. A deeper analysis of the stocks chosen confirm that a few of the selected fourteen stocks were almost definitely irrelevant towards providing meaningful predictive value for AAPL stock prices, ultimately leading to poorer out-of-sample performance. For example, the model chose the Chinese liquor company "600519.SS" or Kweichow Moutai as a relevant predictor in six of the eight out-of-sample periods, even though we can with a high degree confidence claim that Kweichow Moutai has little predictive ability towards AAPL stock pricing. We also noticed that Samsung was present in every model, implying that Samsung might be a fantastic company to run a pairs trade algorithm against AAPL. This shouldn't surprise us as they both are huge in the tablet/laptop/phone markets and are the biggest direct competitors to each other globally. Thus, we notice that while LASSO regression wasn't particularly helpful in providing predictive value in terms of increasing profits, it allowed us to identify the most relevant features to run a simple pairs trade algorithm, highlighting it's potential for feature selection.

```
In [21]: Counter(features_selected_flat_list)
Out[21]: Counter({'005930.KS': 8,
                  '600519.SS': 6,
                  'BABA': 2,
                  'RMS.PA': 6,
                  'ADBE': 6,
                  'CDI.PA': 3,
                  '3690.HK': 3,
                  'TMO': 4,
                  'MA': 1,
                  'ASML': 1,
                  'AVGO': 3,
                  'COST': 3,
                  'NFLX': 2,
                  '300750.SZ': 1})
```

```
In [22]: set(features_selected_flat_list)
Out[22]: {'005930.KS',
          '300750.SZ',
          '3690.HK',
          '600519.SS',
          'ADBE',
          'ASML',
          'AVGO',
          'BABA',
          'CDI.PA',
          'COST',
          'MA',
          'NFLX',
          'RMS.PA',
          'TMO'}
```

Figure 7 List of stocks selected over eight out-of-sample periods between 2021-01-01 to 2022-12-06 using LASSO regression to generate portfolios, and their frequency (mode) of selection.

Feature Selection (an extended analysis)

Based on the subset of fourteen features identified by the LASSO model, I first removed irrelevant stocks like Kweichow Moutai, Costco and Christian Dior and kept the remaining stocks as they belonged to tech/adjacent support industries (like semiconductor/hardware research industries), and then retrained my LASSO model with just the prices of AAPL and these eleven remaining stocks. Because we observed that Samsung was a predictor out-of-sample performance for every period for the original LASSO model, it suggested that Samsung might be more suited to create a pairs trade with AAPL than MSFT. Thus, I reran the OLS model as well between AAPL and Samsung.

While clearly there is look-ahead bias in this approach, I am making a claim that this information is proxy for an experienced businessperson using some fundamental/industry knowledge in selecting an appropriate set of features to regress against the price prediction of AAPL. Furthermore, it's reasonable for me to claim that with more out-of-sample period analysis over multiple iterations of the first LASSO model, I can acquire a deeper insight to build more appropriate features subspaces over time /more

educated prior knowledge on which stocks to pick to optimize with each iteration. Also, I can make a claim for saying that this doesn't necessarily have to come from look-ahead bias, as I can truncate the feature space/set of stocks to limit myself to those within the tech/adjacent supporting industries. Thus, I can argue that this selection of stocks could represent a more nuanced quanta-mental approach to pairs trading.

The OLS pairs trade model for AAPL and Samsung yielded a profit of $13.07 and saw three trades made during this period. Notice the huge returns per trade for AAPL/Samsung compared to AAPL/MSFT and AAPL/Lasso for both LASSO models. This suggests that there could be more scope for optimizing the OLS pairs trade signal strategy.

Date	005930.KS	AAPL	action type	Samsung ownership	AAPL ownership	arbitrage returns
2021-01-05	82200.0	126.599998	-1.0	-0.001217	0.789889	NaN
2021-08-13	74200.0	150.190002	0.0	0.000000	-0.000000	28.365855
2021-08-13	74200.0	150.190002	1.0	0.001348	-0.665823	0.000000
2022-08-10	59900.0	168.490005	0.0	0.000000	-0.000000	-31.456805
2022-08-10	59900.0	168.490005	1.0	0.001669	-0.593507	0.000000
2022-11-10	62900.0	149.699997	0.0	0.000000	-0.000000	16.160349

```
df_execution_days['arbitrage returns'].sum()
```

```
13.06939862454945
```

Figure 8 AAPL/Samsung pairs trade strategy performance summary between 2021-01-01 to 2022-12-06

The updated LASSO pairs trade model with eleven features shows an overall profit of $19.42 from 2021-01-01 to 2022-12-06, with a total of ten positions taken and nine positions closed (so one open position remaining still). This is an improvement on the AAPL/MSFT returns of $16.87 earlier, suggesting that a judicious pre-selection of stock prior to fitting a LASSO model will lead to better prediction of stock prices. Interestingly, all four models seemed to have losses for positions opened in late 2021 to early 2022, which accounted for majority of the losses made with this strategy, suggesting excessive volatility in the markets during this period, which in turn triggered the signal to exit a position when the normalized spread exceeded our threshold based on the normalized spread behavior of the training data.

	arbitrage returns
count	9.000000
mean	2.157626
std	8.249795
min	-8.799803
25%	-5.879772
50%	3.969965
75%	8.121501
max	14.562162

```
#returns if negative values/major losses are excluded
df_execution_days[df_execution_days['arbitrage returns']>0][['arbitrage returns']].sum()
```

```
arbitrage returns    42.663928
dtype: float64
```

```
#true return
df_execution_days['arbitrage returns'].sum()
```

```
19.41863427546788
```

Figure 9 Statistics for profits made during using the updated feature space of LASSO regression.

Conclusion

To summarize, I fixed AAPL as the first choice of stock in a pairs trading strategy. Then I found the largest firm by market cap that had a correlation >0.8 with AAPL for our first training period, and then based on information from the data presented proceeded to create a signal for OLS regression-based pairs trading between them. One extension of this analysis could include a regular check for whether AAPL correlates with MSFT in the training period and once the correlation falls below a certain level, to stop trading between them. After performing OLS pairs trading, we proceeded to take the full set of 99 remaining stocks and ran a LASSO regression model to select subsets that best predicted the price of AAPL and created a pairs trading strategy based on the weights/betas of the stocks and that of AAPL. The performance here was significantly poorer than simple pairs trading, which could have been attributed to having selected irrelevant stocks in the training model. We also noticed that Samsung was chosen as a predictor for AAPL stock prices for every model created, which suggests the potential of Samsung being a better pair for AAPL.

Thus, I performed another OLS pairs trade model on AAPL/Samsung, and I performed LASSO regression on the subsets of stocks selected across the training periods of the previous LASSO model after omitting irrelevant stocks from that analysis. This led to an improvement in the performance of the LASSO regression model significantly, ultimately delivering more profits than in the original pairs trade strategy between AAPL/MSFT. This highlights the feature selection ability of the LASSO model, and the results suggest that pair some industry/fundamental knowledge with the statistical technique has the potential to yield higher returns consistently. To truly be able to judge the effectiveness of this algorithm, trading cost modelling and margin calls need to be factored in the analysis too.

References

Van der Have, (2017), Pairs Trading Using Machine Learning: An Empirical Study

Prof. Renyuan Xu, for guidance in the displaying the plots in Figures 1 and 3.