Pairs Trading Strategies & Optimal Execution: Profit Evaluation

Benjamin Granat

ISE537 – 12/11/2024

Introduction

This project delves into pairs trading, a strategy that capitalizes on the mean-reverting behavior of stock pairs. By leveraging linear and polynomial regression techniques, we model the spread between two correlated stocks and identify profitable trading opportunities. An additional aspect of this project was exploring the effectiveness of polynomial regression in pairs trading. While linear regression models the relationship between stock pairs, polynomial regression captures more complex, non-linear dependencies. By incorporating polynomial regression, we extended the analysis of spread behavior, providing a more nuanced understanding of how stock prices interact in pairs trading. We also focused on the concept of optimal execution, which involves determining the duration for which mean reversion strategies should be maintained before liquidating the positions and exiting the strategy. This requires an understanding of when the spread returns to its mean and when it may no longer revert as expected. The broader concept here is understanding the time decay of the strategy and when reversion to the mean weakens, allowing for more efficient trading decisions.

Dataset Format & Pre-Processing

The dataset used in this project consists of historical stock prices and company-specific information for the top 100 publicly traded companies based on their market capitalization. The data includes daily adjusted closing prices for each stock, spanning the period from September 1, 2020, until a year before today's date. This decision was made to avoid the significant market disruptions caused by the COVID-19 pandemic. The pandemic-induced volatility and uncertainty could have distorted historical patterns and relationships between stocks. Excluding this period allows us to focus on more typical market behavior, reducing the risk of overfitting our model to anomalous data. My out of sample dataset spans the last year, starting from today's date.

The source of this data is **Yahoo Finance**, accessed through the yfinance Python library. The data is organized into two primary components: static company-specific information and time series stock price data. Static data includes ticker symbols, company names, market capitalization, PE

ratios, sectors, and industries, while the time series data captures daily adjusted closing prices. Since there is no direct method in yfinance to filter stocks by market capitalization, I manually curated a list of top 100 tickers for data retrieval. I also downloaded a csv_file containing the closing price against all dates (YYYY-MM-DD), aggregated by tickers.

During pre-processing, market capitalization values were converted into numeric types, and invalid or missing values were handled. The GTOFF ticker was removed due to a lack of meaningful data (e.g., prices were either missing or unrealistically low, such as $0.0109). To maintain dataset integrity, I replaced GTOFF with Toyota Motor Corporation (TM), the 101st largest company by market cap, ensuring a complete and reliable dataset for analysis. Additionally, adjusted closing prices were log-transformed to standardize scale and improve correlation analysis. To remove duplicate tickers representing the same company (e.g., GOOG and GOOGL), I mapped tickers to company names and retained the ticker that exhibited higher correlation with another stock, as this was more relevant for our pairs trading strategy. Also, dimension reduction techniques were not needed due to the manageable size of the dataset and the focus on high-capitalization stocks, which are more stable, less volatile, and have comprehensive historical data available.

Overview of Model(s)

Ordinary Least Squares (OLS)

Ordinary Least Squares (OLS) regression was chosen as the initial model for this project due to its simplicity, interpretability, and efficiency. In the context of pairs trading, OLS provides a direct way to assess the linear dependency between stock pairs, which is crucial for identifying reliable spreads. By estimating the relationship between one stock's price movements (independent variable) and another's (dependent variable), OLS regression allows us to capture and quantify these linear relationships. A key benefit of OLS is its ability to provide clear insights into the strength and direction of the relationship through its coefficients, enabling a straightforward interpretation of results. However, OLS has notable limitations. It assumes a strictly linear relationship between variables, which may not fully encapsulate the complexities of financial data, potentially overlooking non-linear dependencies. Additionally, OLS is sensitive to outliers and heteroscedasticity, which can distort model estimates if not accounted for properly.

Evaluation of OLS Model Performance

The performance of the OLS models was evaluated using adjusted R^2 (R^2_adj), F-statistics, p-values, and test mean squared error (MSE). R^2_adj accounts for the number of predictors,

preventing overfitting by penalizing the addition of irrelevant variables, while test MSE evaluates the model's predictive accuracy on unseen data. The benchmarks of $R^2\_adj \geq 0.9$ and test MSE $\leq$ 0.1 were established to ensure the models captured reliable relationships while being robust to overfitting. These thresholds are especially relevant in quantitative finance, where models need to perform reliably in changing market conditions.

Coefficients were assessed for statistical significance using p-values, ensuring that only meaningful relationships were included. Although we didn't have a pre-established F-statistic benchmark, we were surprised by how large the F-statistics of our models were, implying a very strong overall significance of the regression model and indicating that the variations explained by our models were highly unlikely to be due to random chance. Several stock pairs met the defined benchmarks and exceeded expectations, demonstrating the strength of their linear relationships and validating the appropriateness of OLS for these pairs.

Metrics such as the condition number, often used to detect multicollinearity, were not considered due to the single-variable nature of the OLS regression, which inherently eliminates multicollinearity concerns. Additionally, metrics like the Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC), commonly used to compare models with varying numbers of predictors, were not applicable here.

Polynomial Regression (Additional Material)

To address the limitations of OLS in capturing non-linear dependencies, polynomial regression was introduced. This model extends OLS by including higher-degree terms, enabling the identification of complex relationships between stock pairs. In the context of pairs trading, polynomial regression provides a deeper understanding of how stock price movements may interact in non-linear ways, offering potential opportunities beyond linear spreads. For example, suppose we are analyzing the relationship between two technology stocks, Stock A and Stock B. In a linear model, we might only consider the direct proportionate relationship between their price movements. However, the relationship might be more complex due to factors like shared market segments, competitive dynamics, or macroeconomic variables. Polynomial regression can capture these intricacies by modeling the non-linear dependencies—such as quadratic or cubic terms that reflect acceleration or deceleration in their correlation.

The primary benefit of polynomial regression is its flexibility in modeling non-linear patterns, but this comes at the cost of increased complexity and susceptibility to overfitting. High-degree polynomials can lead to models that fit the training data too closely, compromising their

predictive performance on unseen data. This overfitting occurs because higher-degree polynomials introduce many additional parameters, which allow the model to capture not just the underlying trend or pattern but also the noise within the data. Consequently, the model becomes less effective when applied to new, unseen data, as the intricate patterns it learned from the training data are likely to be irrelevant or absent in new data. Moreover, high-degree polynomial functions can exhibit extreme oscillations between data points, creating unrealistic and unintuitive predictions.

Evaluation of Polynomial Regression Performance

Polynomial regression was evaluated using the same metrics as OLS—adjusted R^2 (R^2_adj) and test mean squared error (MSE)—along with additional benchmarks for improvement over OLS. Specifically, a polynomial model was considered better if it increased R^2_adj by at least 2.5% and decreased test MSE by at least 2.5% compared to the corresponding OLS model. These benchmarks ensured that only meaningful improvements were considered, balancing the trade-off between model complexity and performance gains. Additionally, we evaluated the p-values of all coefficients to ensure that each term, including the higher-degree terms, was statistically significant. This step was crucial to confirm that the inclusion of these terms was justified and contributed meaningfully to the model's performance. Parameter selection for polynomial regression involved testing degrees 2 and 3, chosen to strike a balance between capturing non-linear dynamics and avoiding overfitting.

Other potential evaluation metrics, such as Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), could have been useful in assessing the trade-offs of including higher-degree terms but were not explicitly applied here. Cross-validation was not used because our focus was on optimizing performance for specific stock pairs rather than generalizing across a broader dataset. In the context of pairs trading, the goal was to develop and refine models that work particularly well for the selected stock pairs. This specific focus means that the models are tailored to the unique relationships and dynamics between these pairs, making cross-validation less critical. Additionally, since our dataset already accounted for potential overfitting by including out-of-sample tests, we ensured that the models were robust and performed well on new data.

Implementation

Correlation Analysis

In my analysis, I implemented the compute_correlations function to identify highly correlated pairs of stocks from my dataset, which included historical stock prices and company-

specific information. The goal was to ensure that only unique and meaningful pairs were selected, which is critical for effective pairs trading. To achieve this, I standardized the prices to normalize the data, making it easier to compare different stocks on the same scale. Additionally, I excluded pairs of stocks belonging to the same company, such as GOOG and GOOGL from Alphabet Inc. This step was crucial to prevent redundancy and ensure that my analysis focused on unique stock pairs. I iterated over pairs of stocks, computed their correlations, and excluded redundant pairs by comparing their correlation values and deciding which stock to keep. This process allowed me to identify several highly correlated pairs which provided a solid foundation for regression model implementation. Now, I was going to take all the highly correlated pairs and apply regression to examine the relationships between them and evaluate their candidacy for a pairs trading strategy. The identification of these pairs ensured that I could reliably proceed with further analysis and model implementation.

OLS Regression Model Implementation

The run_ols_regression function was responsible for performing Ordinary Least Squares (OLS) regression on each stock pair and evaluating its performance. I split the data into training and test sets to ensure that the model's performance could be validated on unseen data, thus preventing overfitting. I trained the OLS model and then calculated several key metrics, including the R^2_adj, p-values, F-statistics, and mean squared errors (MSE) for both training and test sets. The scale_prices function was designed to standardize stock prices for consistency. Standardizing the data was crucial to ensure that the prices of different stocks could be accurately compared on the same scale. The model's performance was filtered based on quality thresholds: an adjusted R^2 of at least 0.9 and a test MSE of no more than 0.1. This filtering ensured that only the most reliable models were considered for further analysis. The results were promising, with all F-statistics significantly above any reasonable threshold, indicating that the variations explained by these models were highly unlikely to be due to random chance. Additionally, all pairs had high adjusted R^2 values and low MSE's, reflecting strong linear relationships. The test MSE for these models was close to 0.1, indicating good predictive performance but also highlighting room for improvement.

It makes sense that our models are performing well with these pairs because they consist of companies with strong industry ties or similar market dynamics. For instance, CVX and XOM are both major players in the oil and energy sector, naturally leading to correlated stock movements due to shared economic influences. Similarly, the pairing of LLY and NVO can be explained by

their involvement in the healthcare and pharmaceuticals industry, which often results in synchronized price movements due to sector-specific trends and government policy. Given these results however, I aimed to refine the models further to achieve an adjusted R^2 above 0.95 and a test MSE below 0.05. While the OLS models showed promising results, with test MSEs closer to 0.1, there was room to enhance the predictive accuracy and reduce the mean squared error further.

Polynomial Regression Model Implementation

To enhance my pairs trading strategy, I introduced polynomial regression to address the limitations of OLS in capturing non-linear dependencies. I created the run_polynomial_regression function to perform polynomial regression on each stock pair and evaluate its performance.

The run_polynomial_regression function aimed to identify and evaluate the best polynomial model for each stock pair by testing different degrees of polynomial features. I split the data into training and test sets, maintaining a 70-30 split to ensure the robustness of the test set. For each degree, I transformed the independent variable (X) into polynomial features and then fit a linear regression model. I calculated key metrics, including mean squared errors (MSE) for both training and test sets and R^2_adj to assess model performance. I also used the statsmodels library to fit the model and obtain p-values for the coefficients, ensuring that only statistically significant models were considered. The best model for each pair was identified based on the highest adjusted R^2 and the lowest test MSE. I only considered polynomial models that significantly outperformed their OLS counterpart (a polynomial model was considered better if it increased OLS R^2_adj by at least 2.5% and decreased OLS test MSE by at least 2.5%). I replaced the stock pair's OLS model with the corresponding polynomial model that performed better.

A notable improvement was observed with the polynomial model for the pair **SHEL and XOM**. By introducing a third-degree polynomial, the model captured more complex relationships between these stocks, resulting in an adjusted R^2 of 0.9403 and a test MSE of 0.0611. This indicates that the polynomial model was able to capture non-linear dependencies between SHEL and XOM, providing a deeper understanding of their correlated movements. The relationship between SHEL and XOM might be non-linear due to complex factors like fluctuating oil prices, regulatory changes, and economic cycles, which don't affect stock prices in a straightforward manner. For instance, sharp changes in oil prices or new regulations can create disproportionate impacts on these companies, leading to non-linear correlations in their stock movements. By using polynomial regression, I could capture these intricate dynamics more accurately.

Spread Calculation & Mean-Reversion Strategy Evaluation

To analyze the relationships between stock pairs, I implemented functions to calculate spreads using both OLS and polynomial regression models. The calculate_spread_using_ols function computes the spread based on the OLS model's intercept and beta coefficients, while the calculate_spread_using_poly function uses polynomial regression to account for non-linear relationships. I calculated and plotted these spreads for each stock pair, including mean and standard deviation lines, to visualize their behavior over time.

The mean_reverting_strategy function implements a pairs trading strategy by calculating the mean and standard deviation of the spread between two stocks. It generates buy signals when the spread is abnormally low, meaning it falls below one standard deviation below the mean, and sell signals when the spread is abnormally high, meaning it exceeds one standard deviation above the mean. When we execute a buy signal, we are buying stock2 and selling stock1. Conversely, when executing a sell signal, we are selling stock2 and buying stock1. An exit signal is triggered when the spread reverts to within one standard deviation of the mean, meaning we sell stock2 and buy stock1 to close the position. The function tracks positions, executes trades based on these signals, and records the actions and profits. At the end of the iteration, any open positions are liquidated.

In my analysis, I used the buy-and-hold strategy as a benchmark model to evaluate the effectiveness of the mean-reverting pairs trading strategy. The buy-and-hold approach involves purchasing stocks and holding them over a period, with the profit calculated as the difference between the initial and final prices of the stocks. For this strategy, I bought and held one share of each stock. This strategy is straightforward and widely used by investors, making it a useful comparison point. By contrasting the profits from the buy-and-hold strategy with those from the pairs trading strategy, I could assess whether the more involved mean-reverting approach provided a significant advantage in terms of profitability.

The calculation functions assess the profitability of the pairs trading strategy compared to a buy-and-hold benchmark. The calculate_profit_from_trades function computes net profit from the strategy's buy and sell actions, while the calculate_profit_buy_and_hold function calculate profit from simply holding both stocks. The calculate_profit_for_pair function integrates these methods, generating buy/sell signals and comparing strategy profits to the buy-and-hold approach for a given stock pair.

Conclusion & Sources of Error

Profitability Analysis & Optimal Execution (Additional Material)

The last cell aims to evaluate the profitability of the pairs trading strategy compared to the buy-and-hold approach by defining specific periods and computing profits for those periods. The strategy is analyzed over different time periods—15 days, 1 month, 45 days, 2 months, etc.—to understand how the trading strategy's profitability decays over time. In our implementation, cumulative profit differences represent the aggregate difference between the profits earned from the pairs trading strategy and the buy-and-hold strategy over specified periods for all pairs. By evaluating this metric, we can estimate an approximate date range limit for holding positions, thereby enhancing the effectiveness of the mean reversion strategy.

Some pairs performed significantly better in terms of profit for both in-sample and out-of-sample data. Several pairs demonstrated strong profit differences across both in-sample and out-of-sample data. In the short-term, the pairs LLY & NONOF and LLY & NVO, and NVO & PCCYF consistently showed positive profit differences in the 15D period, indicating strong performance under short-term market conditions. For the longer-term, the pairs SHEL & XOM and LLY & NVO exhibited positive differences over the 2M and 1Y period (respectively), suggesting it outperformed in extended timeframes. However, the pairs LLY & NONOF and LLY & NVO are the worst-performing pairs across in-sample and out-of-sample data in longer time horizons. Based on the scaled average log(profit_diff) for all holding periods, the optimal endpoint for pairs trading strategies appears to be 45 days, as it yields the highest mean log(profit_diff), highlighting the strategies' consistency in generating returns compared to longer holding periods. The cumulative profit differences reveal this trend: while the strategies deliver positive returns in shorter periods (e.g., 15D, 1M), profitability diminishes significantly as the strategy period extends. Despite the models' metrics being adequate and leveraging a breadth of data, strategies consistently struggle to maintain profitability over extended horizons. This decay can be attributed to the mean-reversion assumption's decreasing reliability over time.

Conclusion

Our hypothesis rests on the assumption of mean reversion, which posits that the spread between two stocks will return to its historical average over time. The mean reversion assumption generally holds in markets where stock prices exhibit periodic or cyclical movements, such as stocks with highly correlated sectors. However, this assumption weakens during periods of

structural market changes, such as shifts in macroeconomic conditions, industry disruptions, or significant market volatility. Additionally, the effectiveness of mean reversion may diminish over longer periods of time, as stock prices could drift away from their historical correlations or experience new long-term trends. The hypothesis, rooted in mean reversion, explains the model's efficacy at identifying short-term opportunities because the spread between two stocks often reverts to its historical average more consistently in the short term. This behavior is typical in mean-reverting pairs, where market inefficiencies or temporary mispricing are corrected over shorter intervals.

Sources of Error

One source of error in this study is the uncertainty around overfitting. While the model showed improved mean squared error (MSE) from training to testing, it may still have learned noise rather than generalizable patterns. Another concern is the assumption of mean reversion, which may not hold in non-stationary financial markets. Statistical tests, such as the Augmented Dickey-Fuller (ADF) test, could have validated this assumption. Beyond mean reversion, other models could have been explored to improve robustness, such as cointegration analysis or Kalman filter models, which dynamically adjust the pair relationship over time. These models are particularly useful for capturing evolving market dynamics and ensuring that the pair's relationship remains statistically significant. Additionally, external factors like interest rates, macroeconomic trends, or sector-specific shocks were not accounted for, potentially affecting correlations. The selected date ranges may also limit accuracy, as specific market conditions might not represent broader trends. Finally, managing multiple dictionaries and dataframes added complexity and potential for errors, highlighting the need for a streamlined data pipeline. These challenges emphasize limitations and suggest improvements for future work.

Works Cited

yfinance

Tiemann, Ran A., et al. *yfinance: Yahoo! Finance API*. GitHub, 2023, https://github.com/ranaroussi/yfinance.

ChatGPT

OpenAI. *ChatGPT: Language Model Assistance*. OpenAI, 2024, https://openai.com/chatgpt.

Stack Overflow

Stack Overflow. *Programming Community and Q&A Platform*. Stack Overflow, 2024, https://stackoverflow.com.

Lecture PDF

Skiena, Steven S. *Lecture 23: Introduction to Machine Learning*. Stony Brook University, 2023, https://www3.cs.stonybrook.edu/~skiena/691/lectures/lecture23.pdf.

Hudson & Thames. "Definitive Guide to Pairs Trading." *Hudson & Thames*, https://hudsonthames.org/definitive-guide-to-pairs-trading/.

Zhu, Rong. "Pairs Trading." *Yale Department of Economics*, May 2024, https://economics.yale.edu/sites/default/files/2024-05/Zhu_Pairs_Trading.pdf.

Gatev, Evan, William N. Goetzmann, and K. Geert Rouwenhorst. "Pairs Trading: Performance of a Relative-Value Arbitrage Rule." *National Bureau of Economic Research*, Mar. 1999, https://www.nber.org/system/files/working_papers/w7032/w7032.pdf.

Palomar, Daniel P. "Pairs Trading: Theory and Practice." *Hong Kong University of Science and Technology*, https://palomar.home.ece.ust.hk/MAFS5310_lectures/slides_pairs_trading.pdf.