Hernan Barajas, Danny Botek, Beatriz Ribeiro, Johnathan DeMarco and Nathan Gandawa
High Frequency Trading Technologies (CSE-40438)
Professor Matthew Belcher

High Frequency Trading Technologies - Group 7 Project Report

1. Background information and our objectives

Statistical arbitrage first gained traction in the 1980s thanks to Nunzio Tartaglia's automated trading group at Morgan Stanley. Their work marked a turning point in financial markets, introducing more systematic and data-driven approaches to trading. The original idea was rooted in pairs trading, where traders would go long on one asset they believed was undervalued and short another they thought was overpriced, usually within the same sector or industry. The goal was to profit when the price relationship between the two reverted to its historical average—an approach that doesn't rely on the overall direction of the market, but instead on relative mispricings.

Since then, statistical arbitrage has evolved into a broad category of strategies that use quantitative models to find and exploit temporary pricing inefficiencies. These strategies have grown increasingly complex, ranging from classic equity pairs trading to more advanced forms like factor-based arbitrage, which takes advantage of return patterns tied to characteristics like value, momentum, and size. Many of these strategies are market-neutral, meaning they balance long and short positions to minimize exposure to general market movements.

In recent years, high-frequency trading has pushed statistical arbitrage into even faster territory, with algorithms now executing trades in microseconds to take advantage of tiny price discrepancies. At the same time, machine learning has become an increasingly popular tool, allowing traders to detect subtle and non-obvious patterns in large datasets that traditional models might miss. Mean reversion strategies are also widely used in this space, based on the idea that asset prices tend to drift back toward historical averages over time. Altogether, statistical arbitrage today reflects a mix of traditional financial thinking and cutting-edge technology, making it a key area of interest in both academic research and professional trading.

Our group planned to develop a high-frequency statistical arbitrage strategy based on pairs trading to exploit temporary pricing inefficiencies among historically correlated stocks from similar sectors (e.g., Home Depot/Lowes, Apple/Microsoft). We wanted to identify moments where the relationship between prices breaks (i.e. they don't move in the same direction in the expected magnitude) and go long one and short the other expecting the spread to shrink.

2. Our initial solution (pre-presentation)

In our initial approach, we actually decided to analyze foreign exchange data instead of stocks, since one of our team members had some previous experience in pairs trading with foreign exchange data. Initially, we fetched data from an API managed by Oanda, a forex and cryptocurrency trading platform.

The API provided clean OHLCV candle data at the hourly level, and we focused on a selection of major and cross currency pairs, including EUR/USD, GBP/USD, USD/JPY, EUR/GBP, and GBP/JPY.

Once we had the data, we proceeded to check for strong currency pairs, which was done in two steps. First, we focused on identifying currency pairs with historically strong correlations. This step ensured that the pairs we examined were likely to exhibit predictable relationships. Second, we used cointegration testing to model these relationships. Specifically, we looked for pairs whose price series had a stationary linear combination, a key signal that the spread between them would tend to revert to a historical mean. This was achieved using linear regression to estimate the hedge ratio, and the Augmented Dickey-Fuller (ADF) test to confirm stationarity. cointegrated pairs that we would want to further analyze.

Once cointegrated pairs were identified, we generated trading signals by calculating the spread between the two prices (adjusted by the hedge ratio), and then computing the Z-score of this spread. When the Z-score deviated significantly from zero—typically beyond ±2 standard deviations—we interpreted this as a trading opportunity, expecting the spread to mean-revert. A Z-score above 2 prompted a short on the first currency and a long on the second, while a Z-score below -2 triggered the reverse. We exited positions when the Z-score returned to ±0.5, or if it moved beyond ±3, which we treated as a stop-loss threshold.

Our backtesting procedure simulated trades across cointegrated pairs by applying our Z-score logic and tracking entry/exit points, profits, and losses. We computed metrics such as Sharpe ratio, win rate, annualized return, and volatility to evaluate the strategy. However, we noticed our solution was far from being profitable. This made sense considering that we were not training using a statistical or machine learning model, which meant our thresholds for entry and exit signals were hard-coded. Moreover, we noticed that, given the lack of cointegration between currencies when compared to stocks, we decided to not only review our approach, but also change the data we were feeding into our code.

3. Reviewed solution

In the second phase of the project, we took Professor Belcher's and the class' feedback and made changes to the system. We moved away from currency data and started working with high-frequency stock data that included actual bid and ask quotes, which we obtained from Wharton Research Data Services. This allowed us to model trades more realistically and avoid lookahead bias by making sure trades were executed on the next available price, not the one used to generate the signal. We also introduced lead-lag analysis—checking whether one stock tends to move slightly before another—and adjusted the strategy to only trade the lagging asset when a relationship was found.

Another change was the addition of a training/optimization strategy. We read a paper by Barthelemy et al. (2024) that laid out a method for parameter tuning in pair trading, and we used that to guide our use of Optuna in Python for hyperparameter search. Instead of manually picking thresholds, we optimized entry and exit points based on past data, all within a point-in-time setup to avoid overfitting. We also added a random trading baseline so we could compare our performance against something that has no intelligence behind it, just to make sure our system was actually doing something meaningful.

By the end of the phase, our codebase had been refectored—modularized across data loading, pair selection, strategy logic, and optimization—and our strategy was more flexible which we believe would reflect more real-world trading constraints.

4. **Code structure**

Currently, our code for the second iteration of the project has the following components:

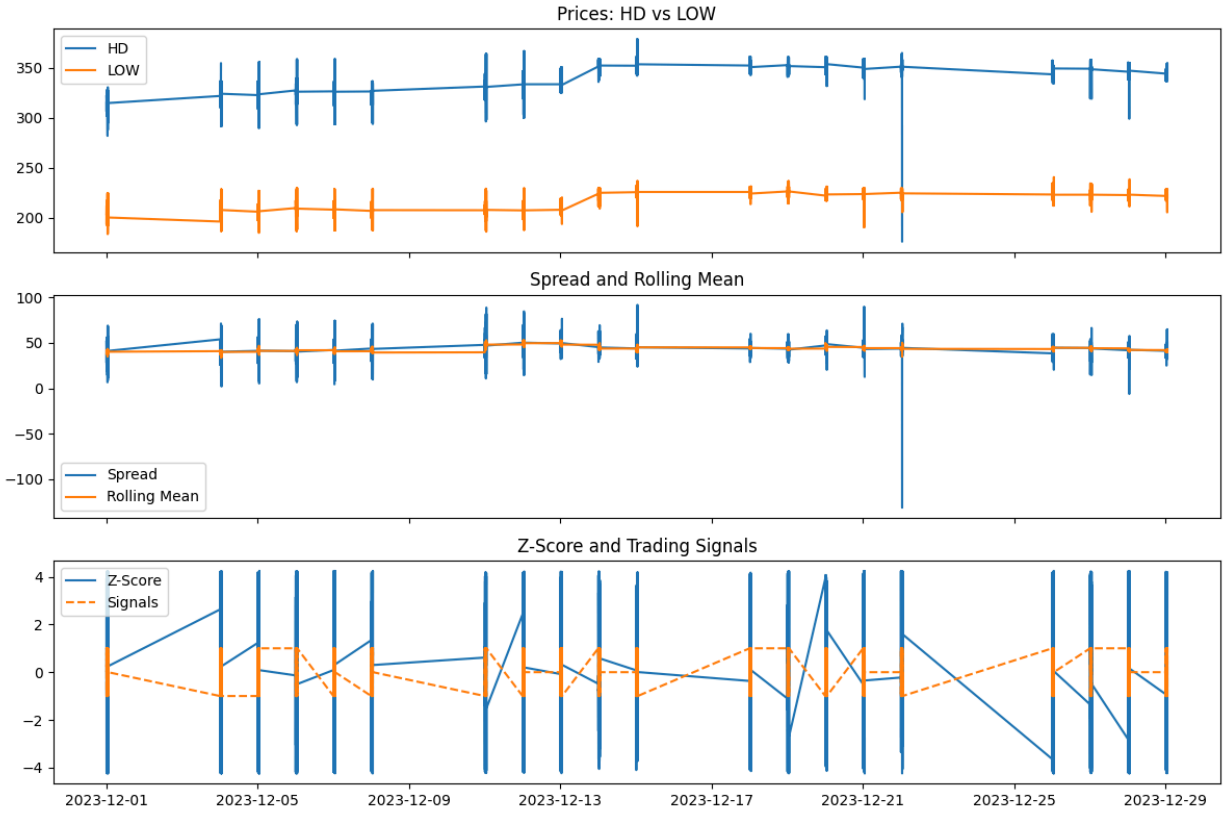
- **data_loader.py**: High-frequency CSV → Parquet pipeline with Dask.
- **pair_selector.py**: Correlation & cointegration.
- **stat_arb_strat.py:** Hedge ratio, spread, & z-score.
- **optuna_optimizer.py:** Hyperparameter search with trade-frequency constraint and random baseline.
- **main.py**: main file that executes the project steps in order.

5. **Sample run and trading results**

Attached below is our terminal output and logging from a sample run, along with a graphic visualization of its trading results.

```
[I 2025-05-09 21:30:04,228] Trial 498 finished with value: 928515.4597170912 and parameters: {'entry_z': 0.53212
34347912867, 'exit_z': 0.36336888356970254, 'window': 5}. Best is trial 102 with value: 928515.1200605737.
2025-05-09 21:30:04,245 - INFO - Initialized StatArbStrategy with transaction cost: 0.0001
2025-05-09 21:30:04,254 - INFO - Hedge ratio: 0.9477
2025-05-09 21:30:04,268 - INFO - Generated 195853 signals
2025-05-09 21:30:04,273 - INFO - Backtest complete, net returns length: 195852
[I 2025-05-09 21:30:04,301] Trial 499 finished with value: 929116.8602904413 and parameters: {'entry_z': 0.53610
022038117, 'exit_z': 0.3496388196235623, 'window': 5}. Best is trial 102 with value: 928515.1200605737.
2025-05-09 21:30:04,301 - INFO - Best parameters: {'entry_z': 0.5311821039760103, 'exit_z': 0.23074464558195512,
 'window': 5}, value: 928515.1201
2025-05-09 21:30:04,302 - INFO - Optimized parameters: {'entry_z': 0.5311821039760103, 'exit_z': 0.2307446455819
5512, 'window': 5}
2025-05-09 21:30:04,302 - INFO - Backtesting on validation set...
2025-05-09 21:30:04,302 - INFO - Initialized StatArbStrategy with transaction cost: 0.0001
2025-05-09 21:30:04,314 - INFO - Hedge ratio: 0.8389
2025-05-09 21:30:04,327 - INFO - Generated 192502 signals
2025-05-09 21:30:04,332 - INFO - Backtest complete, net returns length: 192501
2025-05-09 21:30:04,332 - INFO - Validation simple PnL for HD/LOW: 701.2487
2025-05-09 21:30:04,332 - INFO - Backtesting on test set...
2025-05-09 21:30:04,332 - INFO - Initialized StatArbStrategy with transaction cost: 0.0001
2025-05-09 21:30:04,338 - INFO - Hedge ratio: 1.3673
2025-05-09 21:30:04,342 - INFO - Generated 57880 signals
2025-05-09 21:30:04,343 - INFO - Backtest complete, net returns length: 57879
2025-05-09 21:30:04,344 - INFO - Number of trades: 32144
2025-05-09 21:30:04,344 - INFO - Test simple PnL for HD/LOW: 190.8793
2025-05-09 21:30:05,994 - INFO - Saved visualization to trading_results.png
```

Hernan Barajas, Danny Botek, Beatriz Ribeiro, Johnathan DeMarco and Nathan Gandawa
High Frequency Trading Technologies (CSE-40438)
Professor Matthew Belcher



6. Future work and points of improvement

While we are happier with the second iteration of our project when compared to its first, there is still room for improvement. A couple of things we want to implement are

- Better and more thorough visualizations, in order for us to be able to visually identify any issues with our strategy in case we don't catch them in the code. For example, it cou
- A way to actually trade would also be nice–as of right now we only produce trading signals and simulate PnL based on that. Therefore, implementing a way to more realistically simulate trading (even with fake money) would be a great addition.
- We also need better metrics to understand our PnL. Even if we make profits, that might not mean we are trading well or at our best, so it'd be nice to be more thorough about it.

7. References

C. Barthelemy, R. Chen, and E. Lucyszyn, Parameters Optimization of Pair Trading Algorithm, arXiv preprint arXiv:2412.12555, Dec. 2024. [Online]. Available: https://arxiv.org/abs/2412.12555