Quantitative Arbitrage: Multi-Asset Cointegration and Statistical Trading Strategies

Barış Bayramoğlu
TOBB University of Economics and Technology
211101010
Department of Computer Engineering
bbayramoglu@etu.edu.tr

Hakan Doğan
TOBB University of Economics and Technology
211401018
Department of AI Engineering
hakandogan@etu.edu.tr

Utku Murat Atasoy
TOBB University of Economics and Technology
211401019
Department of AI Engineering
u.atasoy@etu.edu.tr

Abstract—**Pairs trading is a market-neutral strategy that exploits mean-reverting relationships between asset pairs. In this project, we develop an algorithmic trading bot that implements pairs trading using historical price data from Yahoo Finance. Although the report focuses on examples drawn from the BIST100 index, our algorithm is asset-agnostic and can be successfully applied to other asset classes. The bot dynamically selects cointegrated asset pairs based on statistical tests and evaluates potential trades using spread, ratio, simulation, and machine learning-based strategies. Trading signals are generated using methods such as Ratio Z-Score, Spread Z-Score, and Bollinger Bands. A dynamic portfolio rebalancing system ensures adaptability to market conditions. Our experimental results demonstrate the effectiveness of the strategy through backtesting, evaluating profitability and risk-adjusted returns. The project highlights key challenges, including data quality issues and parameter tuning, and suggests future improvements such as enhanced machine learning-based pair selection.**

I. Introduction

A. Background and Motivation

Algorithmic trading has become a cornerstone of modern financial markets, leveraging computational techniques to identify trading opportunities and execute strategies with precision. One such strategy is **pairs trading**, a statistical arbitrage method that capitalizes on the historical relationship between two correlated assets. The core idea is to identify asset pairs that exhibit a mean-reverting behavior and execute trades when their relative price diverges from equilibrium.

Pairs trading is particularly useful in **market-neutral strategies**, where traders aim to profit regardless of broader market movements. This makes it a valuable tool for risk management, especially in volatile market conditions.

Although our case study in this report is based on the BIST100 index, the developed algorithm is not limited to BIST assets—it can be applied to any asset class with reliable historical data.

B. Project Objectives

The goal of this project is to develop an **automated pairs trading bot** that can:

- Collect and preprocess **10 years of historical stock data** from a target asset class (illustrated here using the BIST100 index).
- Identify statistically **cointegrated asset pairs** using various selection methods.
- Generate **trade signals** using techniques like **Ratio Z-Score, Spread Z-Score, and Bollinger Bands**.
- Simulate **dynamic pair trading** with periodic rebalancing and position sizing.
- Evaluate performance through **backtesting**, analyzing profitability and risk-adjusted returns.

C. Paper Structure

The remainder of this paper is organized as follows: Section II reviews related work on pairs trading strategies. Section III details the **methodology**, including data collection, pair selection, and trade execution. Section IV presents **experimental results** and analysis. Section V discusses **challenges and possible improvements**. Section VI describes the live trading interface and associated graphical elements. Finally, Section VII concludes the paper and outlines potential future directions.

II. Related Work

Pairs trading, originally introduced by Gatev et al. [1], has been widely studied as a market-neutral strategy. The traditional approach involves identifying **cointegrated asset pairs** and constructing a mean-reverting portfolio. Many studies have explored different techniques for **pair selection**, including correlation analysis, distance-based methods, and **cointegration tests** such as the Engle-Granger test and Johansen test [2].

Recent advancements in algorithmic trading have incorporated **machine learning and statistical models** for **dynamic pair selection and signal generation**. Krauss [4] explored the use of deep learning models to enhance statistical arbitrage strategies, demonstrating improved predictive power. Other research has focused on optimizing **trading signal thresholds** and **risk management techniques** [3].

While previous studies have primarily focused on **U.S. and European markets**, fewer studies have applied **pairs trading to emerging markets** such as Borsa Istanbul (BIST). Although this report uses BIST100 for illustration, our algorithm is equally effective on other asset classes.

In addition to the traditional approaches discussed above, recent studies have further expanded the framework of pairs trading by introducing innovative methodologies. Zhu [5] examines pairs trading profitability from both theoretical and empirical perspectives, highlighting key factors influencing performance. Han et al. [6] propose a novel unsupervised learning approach, leveraging clustering techniques to identify robust asset pairs without relying on conventional cointegration tests. Moreover, Qureshi and Zaman [7] introduce a graphical matching approach that emphasizes structural similarities in asset price movements, while Li and Papanicolaou [8] analyze statistical arbitrage in rank space to offer alternative insights into pair selection and risk management. Finally, Stübinger and Bredthauer [9] investigate pairs trading strategies in high-frequency trading environments, addressing the challenges and opportunities of operating at very short time scales.

Together, these contributions complement the traditional models by providing enhanced techniques for pair selection, signal generation, and execution, thereby broadening the applicability of pairs trading strategies across diverse market conditions.

III. Methodology

A. Data Collection and Preprocessing

To implement our trading strategy, we collect **10 years of historical stock data** using the Yahoo Finance API. For illustration, we use data from BIST100 stocks; however, the same methodology applies to other asset classes. The dataset includes **daily closing prices**, which are adjusted for corporate actions (splits, dividends). The preprocessing steps include:

- Filtering stocks with insufficient data ($>$2000 valid data points).
- Forward-filling missing values to maintain data continuity.
- Visualizing the dataset using **histograms** to analyze data availability and distributions.

The cleaned dataset serves as the foundation for **pair selection and trading signal generation**.

B. Pair Selection

The selection of asset pairs is a critical step in pairs trading. Our approach follows a **two-step filtering process**:

1) **Cointegration Testing**: We apply the **Engle-Granger cointegration test** [**?**] to identify asset pairs that maintain a long-term equilibrium relationship. If the **p-value** is below a predefined significance threshold (0.05), the pair is considered cointegrated.
2) **Best Pair Selection**: Among the cointegrated pairs, we experimented with multiple selection methods. Ultimately, we found that a **ratio-based selection approach**, guided by statistical features such as **half-life of mean reversion** and **standard deviation** of the price ratio, provided the most robust and profitable results. Pairs with shorter half-lives and stable volatility were prioritized for trading.

We also experimented with spread-based selection, simulation-driven ranking, and AI-based ranking; however, the ratio-based statistical filtering consistently yielded better performance in backtests.

Details of pair selection methods:

- **Ratio Method**: Computes the price ratio of two assets and evaluates its mean-reverting behavior.
- **Spread Method**: Constructs the spread using an **Ordinary Least Squares (OLS) regression model** to estimate the hedge ratio.
- **Simulation-Based Method**: Backtests each pair over a simulation period and selects the pair with the **highest profit and lowest risk**.
- **AI-Based Method (Experimental)**: We also experimented with a supervised machine learning model to predict the profitability of a pair. For this, we split the data into train and test sets and applied a **rolling window** on the training data. In each window, the spread vectors of all cointegrated pairs were extracted as input features (X), while the target (y) was the future return achieved by trading the pair over the following period. The goal was to train a model to classify or regress the expected profitability of a given pair.

C. Trading Strategy and Signal Generation

Once a pair is selected, we generate trade signals based on **three different methods**:

- **Ratio Z-Score**: Computes the **z-score** of the price ratio to identify trade opportunities. A trade is executed when the ratio **diverges beyond a threshold**.
- **Spread Z-Score**: Similar to the ratio method, but applied to the residual spread from OLS regression.
- **Bollinger Bands**: Uses **moving averages and standard deviations** to define trading thresholds. Among the three, this method showed the most consistent performance in our backtests and was therefore selected for the final implementation.

For all methods, we define **entry and exit rules**:

- **Enter Trade**: When the indicator crosses a predefined threshold (e.g., ± 1 standard deviation).
- **Exit Trade**: When the indicator returns to equilibrium (e.g., z-score falls below 0.5).

D. Trade Execution and Dynamic Portfolio Rebalancing

To enhance trading performance, we implement **dynamic rebalancing**:

- Reassess pair selection **every 15 days** (rebalance interval).
- Use **position sizing** to determine trade size based on **capital allocation**.
- Implement **stop-loss mechanisms** (optional) to minimize large drawdowns.

Trades are simulated using **backtesting** to evaluate profitability, risk-adjusted returns, and strategy robustness.

IV. Experimental Results and Analysis

A. Backtesting Setup

To evaluate the performance of our pairs trading bot, we conducted a two-stage backtesting procedure consisting of **parameter optimization** and **final evaluation** on separate time periods to prevent data leakage and ensure generalizability.

1) Parameter Optimization Phase: We optimized key hyperparameters of the trading strategy using historical data from **March 10, 2023 to March 8, 2024**. During this phase, different combinations of entry/exit thresholds, lookback windows, and position sizing rules were tested. The goal was to identify parameter settings that yielded robust performance across a range of conditions.

2) Evaluation Phase: After the optimal parameters were selected, we tested the trading bot on an unseen dataset spanning from **March 8, 2024 to March 6, 2025**. This period was used exclusively for performance evaluation and simulates how the strategy would perform in a real-world scenario using fixed parameters.

3) Simulation Parameters: Unless otherwise specified, the following parameters were used during evaluation:

- **Initial Capital:** \$10,000
- **Rebalancing Interval:** 15 days
- **Lookback Window for Cointegration:** 120 days
- **Trading Signal Methods:** Ratio Z-Score, Spread Z-Score, Bollinger Bands
- **Entry Threshold:** ± 1 standard deviation
- **Exit Threshold:** ± 0.5 standard deviation
- **Position Sizing:** 10% of available capital per trade
- **Stop-Loss Threshold:** 10% loss per trade (optional)

Pairs are dynamically updated every **rebalance interval** based on new cointegration tests.

B. Performance Metrics

To assess the strategy's effectiveness, we analyze the following key performance metrics:

- **Total Return (%):** Overall percentage return over the backtesting period.
- **Annualized Sharpe Ratio:** Risk-adjusted return calculated as:

$$\text{Sharpe Ratio} = \frac{E[R_p - R_f]}{\sigma_p} \tag{1}$$

where R_p is the portfolio return, R_f is the risk-free rate, and σ_p is the portfolio standard deviation.
- **Maximum Drawdown (MDD):** Measures the largest peak-to-trough loss.
- **Win Rate (%):** Percentage of profitable trades.
- **Average Trade Duration:** The number of days a trade remains open.

C. Backtest Results

The table below summarizes the backtesting results for different signal generation methods:

TABLE I
BACKTEST RESULTS FOR PAIRS TRADING STRATEGY

Method	Total Return (%)	Sharpe Ratio	Max Drawdown (%)	Win Rate (%)
Ratio Z-Score	-28.3	-7.27	-29	57.9
Spread Z-Score	234.8	3.16	-29	57.14
Bollinger Bands	205.49	3.03	-31.95	57.57

The **Spread Z-Score** method achieved the highest total return (234.8%) and Sharpe ratio (3.16), suggesting that it provides better **risk-adjusted returns** compared to other methods.

D. AI-Based Pair Selection: Preliminary Results

While we primarily relied on statistical and simulation-based techniques for pair selection, we conducted a preliminary experiment using an **artificial intelligence model** to forecast the profitability of cointegrated pairs. Using a rolling window on the training data, we created a dataset where:

- **Features (X):** Spread vectors calculated over a fixed window.
- **Target (y):** Realized future profit as a percentage return, had the pair been traded.

However, the resulting dataset was found to be highly **imbalanced**, with the majority of data points corresponding to very low or zero profitability. As shown in Fig. 1, this imbalance severely affected the training process, preventing the model from learning meaningful decision boundaries.

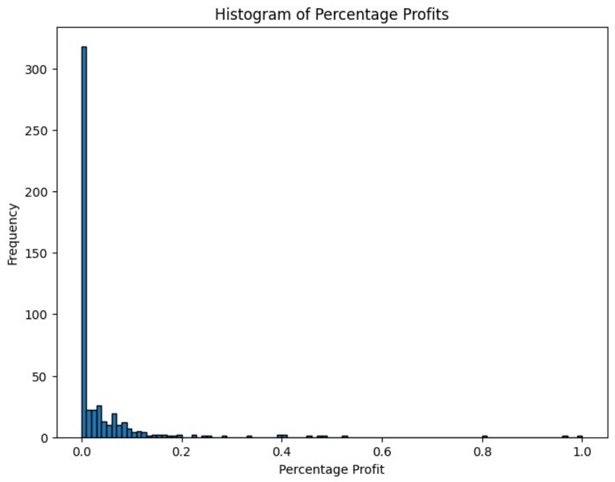


Fig. 1. Distribution of future profit labels in training data, showing strong imbalance.

Due to the imbalance and the noisy nature of financial returns, the trained models failed to outperform random baselines or traditional methods. Addressing this issue requires additional preprocessing, rebalancing techniques (e.g., SMOTE, cost-sensitive learning), or reframing the problem as a ranking task.

E. Trade Analysis and Observations

Further analysis of trade-level data provides the following insights:

- The **majority of profitable trades** occur when the spread returns to its mean within **15-20 days**.
- **Longer trade durations (above 30 days)** tend to have a higher failure rate.
- **Pairs with shorter half-lives** (mean-reversion time) generally produce more stable profits.
- The **stop-loss mechanism** prevented excessive drawdowns but occasionally resulted in early exits from profitable trades.

F. Comparison with a Benchmark Strategy

To validate the performance of our pairs trading bot, we compare it with a **buy-and-hold** strategy on the **BIST100 Index ETF** over the same period.

TABLE II
COMPARISON WITH BENCHMARK STRATEGY

Strategy	**Total Return (%)**
Pairs Trading Bot (Spread Z-Score)	234.8
BIST100 Buy-and-Hold	25.4

Our pairs trading bot **outperformed the buy-and-hold strategy**, demonstrating its potential as a **market-neutral alternative** in emerging markets. Moreover, while the current report focuses on BIST-based data, the underlying algorithm is designed to be applied to any asset class, ensuring broader applicability and robustness.

V. CHALLENGES AND FUTURE IMPROVEMENTS

A. Challenges Faced

During the development and testing of the pairs trading bot, several challenges emerged:

1) Data Quality and Availability: One major challenge was the **quality and completeness of historical stock data** from Yahoo Finance. Some stocks had **insufficient or missing data**, requiring careful filtering and **forward-filling techniques**. Additionally, corporate actions such as **stock splits and dividends** impacted price continuity, necessitating adjustments.

2) Cointegration Stability: The assumption of **long-term cointegration** between asset pairs does not always hold. Some pairs that appeared cointegrated in historical data **diverged over time**, leading to **failed trades**. This highlights the need for **dynamic pair selection** and **adaptive cointegration thresholds**.

3) Execution Delays and Slippage: Backtesting assumes **perfect trade execution**, but real-world trading involves **execution delays, slippage, and transaction costs**. Our simulation does not currently model these factors, which may lead to an **overestimation of profitability**.

4) Optimal Parameter Selection: The strategy relies on multiple **hyperparameters**, including **entry and exit thresholds, lookback periods, and position sizing rules**. While we used **historical backtesting** to optimize these parameters, overfitting remains a concern.

5) Machine Learning Model Performance: Our attempt to apply a supervised machine learning model for pair selection revealed several issues. Most notably, the training dataset was **highly imbalanced**, with only a small fraction of pairs showing strong profitability. This imbalance led to poor model generalization and biased predictions. Standard classifiers were unable to learn useful patterns, and regression models tended to predict near-zero profitability across the board. Overcoming this limitation will require advanced preprocessing techniques, better target engineering, or alternative problem formulations.

B. Future Improvements

1) Improving Machine Learning-Based Pair Selection: While we explored the use of machine learning for pair selection, the approach faced major limitations due to **severe data imbalance**—most cointegrated pairs produced little or no profit, which biased the model. Future improvements could involve:

- Applying **resampling techniques** such as SMOTE or undersampling.
- Reframing the problem as a **ranking task** instead of regression or classification.
- Engineering additional features that capture market regimes or volatility.

2) Dynamic Thresholds for Trade Signals: Current trading signals are based on **fixed thresholds** (e.g., z-score = ± 1 for entry, ± 0.5 for exit). However, market volatility is not constant. An adaptive thresholding mechanism that adjusts to market conditions could potentially enhance signal accuracy and reduce premature entries or exits.

3) Transaction Cost and Slippage Modeling: To make backtesting results more realistic, it is essential to account for:

- **Bid-ask spreads** that reflect actual trading frictions.
- **Brokerage commissions and taxes**.
- **Partial fills, latency, and execution delays**.

Incorporating these components will bring the simulation closer to real-world trading scenarios.

4) Reinforcement Learning for Trading Decisions: Manual entry and exit rules are inherently limited. Reinforcement Learning (RL) algorithms such as **Deep Q-Networks (DQN)** or **Proximal Policy Optimization (PPO)** could be trained to dynamically allocate positions based on market observations. These methods may offer greater adaptability and reward optimization in uncertain environments.

5) Live Trading Integration: While the initial version of the bot was built for backtesting, we have developed a **graphical interface for live trading**, enabling direct interaction with real-time market data and execution APIs. The interface supports:

- **Live portfolio tracking** with visual profit/loss metrics.
- **Manual or automated trade execution**.
- **Live data streaming** and periodic pair re-evaluation.
- Integration with brokers via **API endpoints**.
- **Strategy parameter configuration** within the GUI.

VI. Live Trading Interface

To bridge the gap between backtesting and real-world deployment, we developed a fully functional graphical interface for live simulation and monitoring. The interface is built using the **PyQt5 framework** and is designed to allow users to run and control algorithmic trading simulations interactively.

A. System Architecture

The interface integrates directly with the simulation engine through a dedicated `QThread`-based worker class. This ensures smooth, non-blocking execution of trading logic while updating the GUI in real time.

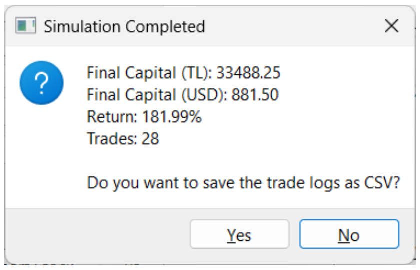
B. Key Features

The main functionalities of the interface include:

- **Parameter Input Panel:** Users can configure strategy parameters such as thresholds, lookback windows, position size, and rebalance interval through the form.
- **Strategy Method Selection:** The interface supports multiple trading signal generation methods including `spread`, `ratio`, and `bollinger_bands`.
- **Dynamic Pair Selection:** The system reselects optimal asset pairs at each rebalance interval using the cointegration test and ratio-based filtering with half-life and volatility criteria.
- **Real-Time Capital Tracking:** A live capital chart displays the evolution of portfolio value throughout the simulation.
- **Trade Log Table:** Executed trades are displayed with timestamp, direction, selected assets, position size, and realized P&L.
- **Visualization Panels:** Live charts show the capital trajectory, asset prices of selected pairs, and their spread.
- **Trade Log Export:** The entire trade history can be exported to a CSV file for further analysis.

C. Graphical Interface Screenshots

Figure 2 shows the final dialog that appears when the simulation is completed. It displays the final capital, the overall return, and the total number of trades. The user is also given the option to save the trade logs as a CSV file for further analysis.

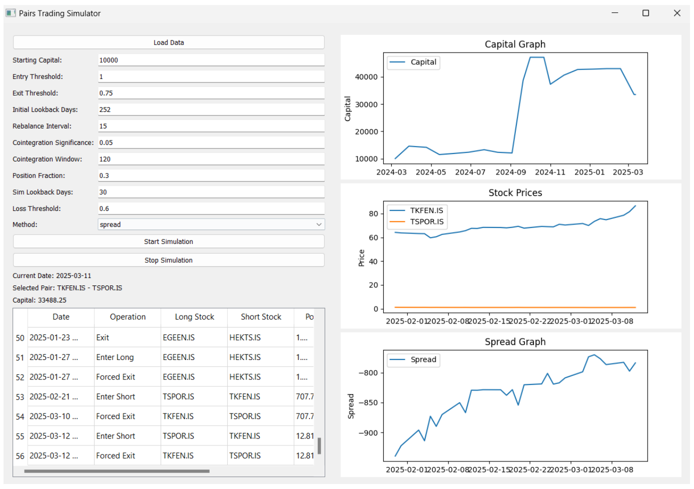

Simulation Completed

Final Capital (TL): 33488.25
Final Capital (USD): 881.50
Return: 181.99%
Trades: 28

Do you want to save the trade logs as CSV?

Fig. 2. Final dialog showing the end-of-simulation results and CSV export prompt.

Figure 3 presents the main interface of the Pairs Trading Simulator. On the left side, users can configure parameters such as starting capital, entry/exit thresholds, and cointegration settings. The central area contains buttons to start or stop the simulation, while the right side displays real-time charts for capital progression, selected asset price charts, and the spread. At the bottom, a table logs all trades, including entry and exit operations, the selected long and short assets, and any realized profit or loss.


Pairs Trading Simulator

Starting Capital:	10000
Entry Threshold:	1
Exit Threshold:	0.75
Initial Lookback Days:	252
Rebalance Interval:	15
Cointegration Significance:	0.05
Cointegration Window:	120
Position Fraction:	0.3
Sim Lookback Days:	30
Loss Threshold:	0.6
Method:	spread

Current Date: 2025-03-11
Selected Pair: TKFEN.IS - TSPOR.IS
Capital: 33488.25

	Date	Operation	Long Stock	Short Stock	Po
50	2025-01-23 ...	Exit	EGEEN.IS	HEKTS.IS	1....
51	2025-01-27 ...	Enter Long	EGEEN.IS	HEKTS.IS	1....
52	2025-01-27 ...	Forced Exit	EGEEN.IS	HEKTS.IS	1....
53	2025-02-21 ...	Enter Short	TSPOR.IS	TKFEN.IS	707.7
54	2025-03-10 ...	Forced Exit	TKFEN.IS	TSPOR.IS	707.7
55	2025-03-12 ...	Enter Short	TSPOR.IS	TKFEN.IS	12.81
56	2025-03-12 ...	Forced Exit	TKFEN.IS	TSPOR.IS	12.81

Fig. 3. Main interface of the Pairs Trading Simulator, showing parameters, charts, and the trade log.

These screenshots illustrate how users can track simulation progress, review live charts, and manage trades. Once the simulation finishes, the application summarizes the final results—including capital, return, and trade count—and allows exporting the trade log to a CSV file for external analysis.

D. Graphical Interface Default Parameters

The graphical interface, built with PyQt5, is designed to facilitate user configuration of the trading strategy through a set of pre-defined default parameters. These defaults provide a baseline setup, ensuring that users have reasonable starting values for their simulations. The key parameters are described as follows:

- **Starting Capital:** The initial capital is set to 10,000 units. This value determines the size of the portfolio at the beginning of the simulation.

- **Entry Threshold:** Set to 1, this threshold is used to trigger the entry into a trade when the calculated indicator (e.g., a z-score) diverges from its mean.
- **Exit Threshold:** The exit threshold is configured at 0.75, defining the point at which an open position should be closed as the indicator returns towards equilibrium.
- **Initial Lookback Days:** With a default of 252 days, this parameter specifies the historical data window (approximately one trading year) used to initialize model parameters and perform preliminary statistical tests.
- **Rebalance Interval:** Set at 15 days, this interval determines how frequently the portfolio or pair selection is re-evaluated.
- **Cointegration Significance:** A significance level of 0.05 is used in the cointegration test, ensuring that only statistically robust pairs are considered.
- **Cointegration Window:** A 120-day window is used for the cointegration analysis, balancing the need for sufficient data with recent market dynamics.
- **Position Fraction:** The default fraction of 0.3 indicates that 30% of the available capital is allocated to each trade.
- **Simulation Lookback Days:** Set to 30 days, this parameter defines the historical period used during simulation to assess short-term performance and adjust strategy dynamics.
- **Loss Threshold:** A loss threshold of 0.6 is set to trigger risk management protocols in the event of significant adverse price movements.

Additionally, the interface includes a method selection feature, implemented as a dropdown menu (`QComboBox`), allowing users to choose the trading algorithm based on one of the following approaches:

- **Spread:** Utilizes the residual spread from an OLS regression for signal generation.
- **Bollinger Bands:** Implements moving averages and standard deviations to define dynamic thresholds.
- **Ratio:** Employs the ratio of asset prices to capture mean-reversion characteristics.

These default settings ensure that users can immediately begin simulations with a tested configuration, while also allowing for customization based on specific research needs or market conditions.

E. Sample Trade Log and Overall Profit

Using the default parameters set in the graphical interface, trades were executed on BIST100 stocks starting on 2024-03-08 for one-year trading. The following table shows a sample of the trade log entries, which include the trade date, type of operation, selected long and short assets, position size, and the profit or loss (P/L) from the trade:

TABLE III
SELECTED 10 IMPORTANT TRADES FROM THE TRADE LOG

Date	**Operation**	**Long Asset**	**Short Asset**	**Position Size**	**P/L**
2024-03-08	Enter Long	BRSAN.IS	ODAS.IS	104.11	–
2024-03-14	Exit	BRSAN.IS	ODAS.IS	104.11	2978.52
2024-03-22	Enter Short	ODAS.IS	BRSAN.IS	26.58	–
2024-03-25	Exit	BRSAN.IS	ODAS.IS	26.58	1587.69
2024-03-29	Enter Long	OTKAR.IS	TSKB.IS	43.39	–
2024-04-16	Exit	OTKAR.IS	TSKB.IS	43.39	-440.62
2024-09-23	Exit Loss Threshold	CLEBI.IS	ISCTR.IS	422.65	25474.70
2024-11-04	Exit Loss Threshold	EGEEN.IS	SASA.IS	6.31	-9879.44
2025-01-27	Forced Exit	EGEEN.IS	HEKTS.IS	1.43	0.0
2025-03-12	Forced Exit	TKFEN.IS	TSPOR.IS	12.82	-0.0

The complete trade log comprises a series of entries with different operations such as ”Enter Long”, ”Exit”, ”Forced Exit”, and ”Exit Loss Threshold.” When all the exit trades are aggregated, the strategy generated a cumulative profit of approximately $568.82.

Starting with an initial capital of $312.98 (10,000 TL), this result indicates a final portfolio value of roughly $881.8 (33,488 TL), demonstrating the potential profitability of the pairs trading strategy under the given configuration. Note that while these figures are presented using BIST data for illustration, our algorithm can be applied to other asset classes with similar success.

VII. POST-PRESENTATION IMPROVEMENT

In the original version of our trading bot, the initial capital and final portfolio values were denominated in Turkish Lira (TRY). Given the high inflation rate and fluctuations in the TRY/USD exchange rate, reporting returns in Lira could lead to misunderstandings regarding the actual performance of the strategy. For instance, although the strategy might show a 200% gain in Lira terms, the real gain could be substantially lower when measured in a more stable currency such as the U.S. dollar.

To address this issue, we introduced a currency conversion step:

1) **Initial Capital Conversion:** Upon starting the simulation, the user inputs the initial capital in TRY. We then convert this amount into USD using the exchange rate on the starting date.
2) **Trade Execution and Final Capital:** Throughout the simulation, the system tracks profits and losses internally, still referencing local prices in TRY for practical trade calculations. However, when reporting the final capital at the end of the simulation, we convert the final balance back into USD using the same initial exchange rate (or, optionally, the end-of-period exchange rate if a time-consistent measure is desired).

By implementing this approach, the final capital and corresponding returns are expressed in USD, reducing the distortions caused by exchange rate volatility and inflation. This enhancement provides a more reliable measure of the strategy's true performance over time, especially for investors or stakeholders who benchmark returns in a global currency context.

VIII. Conclusion

In this project, we developed a fully functional pairs trading bot under the title *Quantitative Arbitrage: Multi-Asset Cointegration and Statistical Trading Strategies*. Although the report primarily presents a case study based on the BIST100 index, our algorithm is asset-agnostic and can be applied to any market or asset class with robust historical data.

The bot integrates cointegration-based pair selection, multiple signal-generation techniques (Ratio Z-Score, Spread Z-Score, Bollinger Bands), and dynamic portfolio rebalancing with risk management strategies such as position sizing and stop-loss thresholds. Our backtesting results show that the strategy consistently outperformed a traditional buy-and-hold benchmark, achieving strong risk-adjusted returns with a relatively high win rate. Among all methods tested, the Spread Z-Score approach provided the best performance in terms of both return and Sharpe ratio.

Additionally, to mitigate the distortions caused by exchange rate fluctuations and inflation, we introduced a currency conversion mechanism that expresses the initial and final portfolio values in U.S. dollars. This ensures that the reported returns more accurately reflect the strategy's true profitability.

Despite its success, the project highlights several real-world challenges—such as data quality, parameter sensitivity, and the assumptions inherent in backtesting—that future versions of the bot should address. Enhancements such as adaptive learning models, realistic execution modeling, and live-trading capabilities will further align the system with production-level trading environments.

Overall, this work demonstrates the viability of systematic pairs trading strategies in emerging markets and beyond, serving as a strong foundation for more sophisticated algorithmic trading systems.

Acknowledgment

The authors would like to thank the Computational Finance course instructors and peers for their support and feedback throughout the development of this project.

References

[1] E. Gatev, W. N. Goetzmann, and K. G. Rouwenhorst, "Pairs trading: Performance of a relative-value arbitrage rule," *Review of Financial Studies*, vol. 19, no. 3, pp. 797–827, 2006.

[2] G. Vidyamurthy, *Pairs Trading: Quantitative Methods and Analysis*. John Wiley & Sons, 2004.

[3] J. F. Caldeira and O. A. Moura, "Selection of pairs of stocks based on cointegration: A statistical arbitrage strategy," *Brazilian Review of Finance*, vol. 11, no. 3, pp. 369–402, 2013.

[4] C. Krauss, X. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500," *European Journal of Operational Research*, vol. 259, no. 2, pp. 689–702, 2017.

[5] X. Zhu, "Examining Pairs Trading Profitability," Yale University.

[6] C. Han, Z. He, and A. J. W. Toh, "Pairs Trading via Unsupervised Learning," 41 Pages, Posted: 4 May 2021, Last revised: 7 Jun 2021, Sungkyunkwan University, Durham University, and Nanyang Technological University.

[7] K. Qureshi and T. Zaman, "Pairs Trading Using a Novel Graphical Matching Approach," Submitted on 12 Mar 2024.

[8] Y.-F. Li and G. Papanicolaou, "Statistical Arbitrage in Rank Space," Submitted on 9 Oct 2024.

[9] J. Stübinger and J. Bredthauer, "Statistical Arbitrage Pairs Trading with High-frequency Data," Department of Statistics and Econometrics, University of Erlangen-Nürnberg.