Short Term Trading Models – Mean Reversion Trading Strategies and the Black Swan Events

Article *in* SSRN Electronic Journal · January 2020

DOI: 10.2139/ssrn.3538891

Ritabrata Bhattacharyya
WorldQuant University

Design and Development of Mean Reversion Strategies on QuantConnect Platform

Vu Duc Long (vuduclong0309@gmail.com)
Ritabrata Bhattacharyya (ritabrata.bhattacharyya@wqu.edu)

Abstract

This practical research investigates the effectiveness of Mean Reversion strategies within the realm of Intraday trading, focusing specifically on the New York Stock Exchange. In contrast to many existing studies that often rely on theoretical assumptions and overlook real-life settings such as transaction fees, this research employs a hands-on approach utilizing the modern platform such as QuantConnect to build real life strategy that is deployable to broker like InteractiveBroker. The study conducts extensive backtesting of fundamental strategies like MACD, RSI, and Bollinger Bands, comparing their performance against popular metrics such as Sharpe ratio. Moreover, the research delves into the impact of risk mitigation techniques, such as stop-loss, on trading performance. By incorporating transaction fees and leveraging real-world platforms, the research aims to solidify the construction of a Mean-Reversion strategy that not only outperforms the market over time but also provides a practical, iterative, and intuitive methodology for translating the theoretical strategy and automate into real-life scenarios.

Keywords: *Mean-Reversion, MACD, RSI, Bollinger Bands, Stop Loss, Kelly Criterion, Strategy Building, Back-testing, Transaction Fee, Options Leverages, QuantConnect, time constraint, resource constraint.*

1. Introduction

Intraday trading is a form of trading where positions are held for a brief period, spanning from seconds to hours. Despite its notable potential to earn significant gain, several research indicated that only a minority of traders achieve significant profitability in day trading. A study analyzing 66,465 households from 1991 to 1996 by Barber and Odean (2000) [1] revealed that individuals with high trading activity often underperform the market. Another study on Taiwan's market day traders by Barber et al. (2004) [2] pointed out while heavy day traders may earn gross profits, these profits dwindle by a large amount of transaction cost, resulting in more than eight out of ten-day traders to experience losses. Although such findings might advocate for buy-and-hold strategies, the research also noted the exceptional returns by a small fraction of traders, with their long stock outperforming their short positions by 62 basis points per day.

In the context of intraday trading, technical analysis stands out as a well-established and widely used approach, relying on historical price and trading volume data to evaluate and predict future prices. This project aims to go beyond existing research by delving into the methodologies of outperforming traders through an extensive literature review. The objective is to construct a comprehensive framework outlining effective intraday trading strategies, culminating in the development of a practical strategy grounded in this framework. To address the often-overlooked real-life nuances of trading, such as transaction costs, this research leverages the

QuantConnect platform and interactive brokers for a more pragmatic exploration. Within the scope of this project, the focus is narrowed down to mean reversion techniques within the realm of technical analysis, aiming to bridge the gap between theoretical assumptions and real-world applicability.

2. Theoretical Framework.

2.1 Core Strategy: Mean reversion

The mean reversion strategy, grounded in the notion that prices tend to revert to an average value over time, holds a prominent position within the realm of technical analysis. Despite ongoing debates questioning its efficacy, this strategy has given rise to numerous derivative approaches.

While certain studies, such as the work of Lo and MacKinlay (1988) [3], have presented contentious findings, highlighting challenges like market shocks and significant events causing deviations from normal price ranges, mean reversion strategies continue to enjoy widespread use.

To initiate our research, we have delved into select studies on this topic, such as examination of the Bollinger Band by John Bollinger (2002) [4]. Then, our focus was on MACD and RSI, indicators that were explored in the research conducted by Terence et al. (2008) [5] on the London Stock Exchange FT30 Index. Their findings suggested that these strategies consistently outperformed buy-and-hold strategies in most cases. We chose this study as our foundation due to several factors: its relative recency compared to other peers while remain purely technical analysis, favorable results, and its focus on a significant market.

We are also aware that at the current state of technical analysis research, as demonstrated by the study of Ghosh et al. (2021) [6], Faraz et al. (2020) [7], Tomar et al. (2020) [8] and Miao (2020) [9], machine learning technique can potentially boost the performance of technical analysis indicator significantly, such as deep learning, LSTM and Random Forest. However, due to the complexity of incorporating the full framework into the platform, we would not mention it in this article. Nonetheless, those technique can assist in helping identify the parameters range for our strategy.

2.1.1 Relative Strength Index (RSI)

The Relative Strength Index (RSI) serves as a momentum oscillator extensively utilized in technical analysis, providing insights into the velocity and extent of price movements within financial markets. Developed by J. Welles Wilder, RSI is instrumental in identifying potential overbought and oversold conditions, offering insights into potential reversals or corrections [10].

2.1.1.1 Basic Calculation - RSI
The calculation of RSI involves comparing average gains and average losses over a specified period, commonly set at 14 periods. The formula provides a numerical value, which is then plotted on a scale ranging from 0 to 100.

1. Select a period, denoted as N, and compute the Average Gain and Average Loss. These are determined by summing the gains and losses over N periods, respectively, and dividing each sum by N.
2. Compute the Relative Strength (RS) as the ratio of Average Gain to Average Loss.
3. Utilize the RS to calculate the Relative Strength Index (RSI) using the formula: RSI = 100 - (100 / (1 + RS)).

2.1.1.2 Interpretation of RSI Values

- RSI above 70 suggests an overbought condition, indicating a potential reversal or correction.
- RSI below 30 suggests an oversold condition, indicating a potential upward bounce or correction.

2.1.1.3 Divergence and Convergence - RSI

Examining the divergence and convergence between price and RSI enhances predictive analysis:

- Bullish divergence materializes when the price establishes lower lows while the RSI registers higher lows, indicating potential upward movement.
- Conversely, bearish divergence unfolds when the price achieves higher highs while the RSI records lower highs, signaling potential downward movement.

2.1.1.4 Volatility and RSI

Consideration of volatility is essential in RSI analysis:

- High volatility may lead to more false signals, making extreme RSI levels less reliable.
- Low volatility enhances the reliability of RSI signals, with extreme levels having stronger predictive value.

2.1.1.5 Limitations and Considerations - RSI

Despite its utility, RSI has limitations that necessitate caution:

- False signals may arise in strong trending markets.
- RSI should be used in conjunction with other technical analysis tools for a comprehensive understanding.

2.1.2 Moving Average Convergence Divergence (MACD)

The Moving Average Convergence Divergence (MACD) is another versatile and widely used technical indicator that fall in trend-following momentum indicator category. Created by Gerald Appel, MACD is employed to identify the strength, direction, momentum, and duration of a trend in a financial market [11].

2.1.2.1 Basic Calculation - MACD

MACD is derived from the differences between two exponential moving averages (EMAs) – typically a 12-period EMA and a 26-period EMA, more specifically, it's the subtraction of the 26-period EMA from the 12-period EMA. Moreover, a Signal line is also included, which is a 9-period EMA of the MACD line.

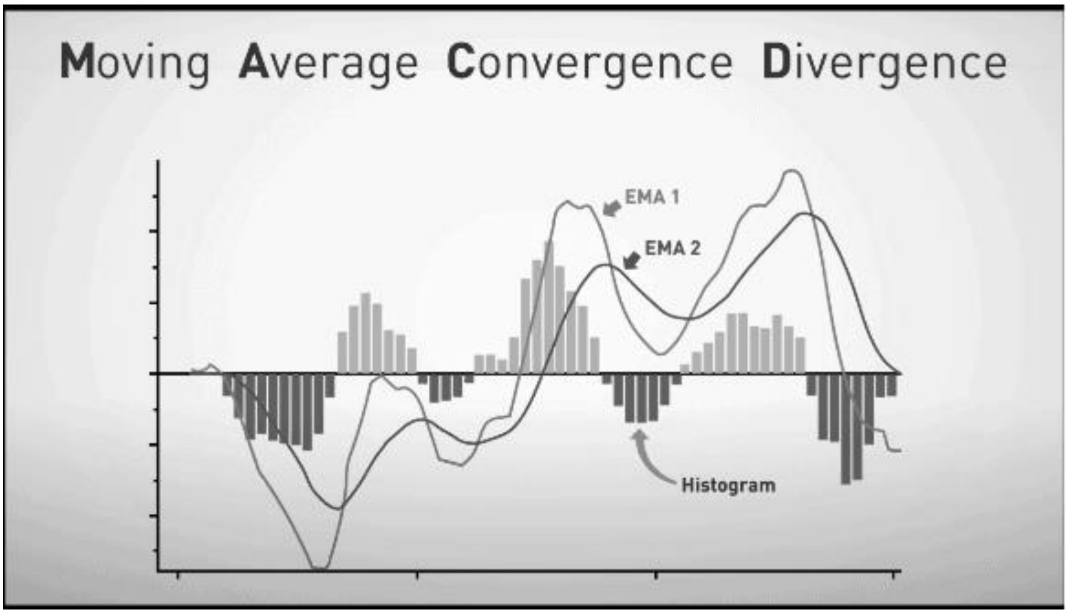


Figure 1: MACD Demonstration. Source: https://www.tradingsim.com/blog/macd

2.1.2.2 Interpretation of MACD Components

- **MACD Line (12-26):** It represents the difference between the short-term and long-term EMAs, providing a signal of the strength and direction of the current trend.
- **Signal Line (9):** A 9-period EMA applied to the MACD line, acting as a trigger for buy and sell signals.
- **MACD Histogram:** The difference between the MACD line and the Signal line, providing insights into the momentum of price movements.

2.1.2.3 Crossings and Signals - MACD

MACD generates signals through its crossovers:

- Bullish Cross: A buying opportunity is signaled when the MACD line crosses above the Signal line.
- Bearish Cross: A selling opportunity is signaled when the MACD line crosses below the Signal line.

2.1.2.4 MACD Histogram and Trend Strength

The MACD histogram provides insights into the strength of a trend:

- Increasing histogram bars indicate strengthening momentum.
- Decreasing histogram bars suggest weakening momentum.

2.1.2.5 Limitations and Considerations - MACD

While MACD is a powerful tool, considerations must be made for its limitations:

- Delayed signals may occur during consolidating markets.

2.2 Strategy Building Frameworks

In the preceding section, we outlined the foundational framework of our trading strategy. However, to develop a comprehensive and robust strategy, attention must be focused on several critical components, including entry/exit points, optimal position sizes, and the implementation of effective risk management protocols. In our pursuit of a well-rounded approach, this study extensively researched prevalent methodologies within the trading domain using search engines. The aim of this paper is to contribute to the existing body of knowledge by exploring and synthesizing insights from various sources to inform the development of a nuanced and effective trading strategy.

2.2.0 General Framework

In this section, we adopted the framework introduced by Robert Pardo in his book "The Evaluation and Optimization of Trading Strategies." [12] Pardo outlines an 8-step process to design a strategy:

""""

1. Formulate the Trading Strategies
2. Translate the Rules into a Definitive Form
3. Preliminary Testing
4. Optimize the Trading Strategies
5. Walk-Forward Analysis
6. Trade the System
7. Evaluate Real-Time Performance
8. Improving The System

""""

Using Pardo's instruction as a basis, and leveraging the powerful backtesting system of QuantConnect, we can seamlessly integrate steps 4-8 into an iteration of backtesting. In this process, we introduce new optimization ideas and backtest them with the platform. Additionally, due to the use of a bot to execute our strategy, an extra step is required before step 3, which involves finding a technical framework.

As a result, a significant portion of our article will be structured in the sequence of the revised steps:

1. Formulate the Trading Strategies (Section 1)

2. Translate the Rules into a Definitive Form (Section 2.1)
3. Choosing Tech Stack (Sections 3 & 4)
4. Preliminary Testing (Section 5.0)
5. Optimize the Trading Strategies with Iteration (Section 5.1 onwards)

Furthermore, Pardo emphasizes that the three principal components of a strategy are Entry and Exit, Risk Management, and Position Sizing. These will be the main aspects considered when generating new ideas for each optimization iteration.

2.2.1 Sharpe Ratio

The Sharpe ratio evaluates the relationship between an investment's return and its associated risk [13]. Introduced in 1966 by economist William F. Sharpe, the ratio originated from his contributions to the Capital Asset Pricing Model (CAPM). The Sharpe ratio is referred to as the reward-to-variability ratio. The numerator signifies the temporal variance between realized or anticipated returns and a benchmark, such as the risk-free rate of return or the performance of a specific investment category. Meanwhile, the denominator represents the standard deviation of returns during the same timeframe, serving as a gauge for volatility and risk. Sharpe, recognized for his work on CAPM, was awarded the Nobel Prize in economics in 1990. Since its inception, the Sharpe Ratio has established itself as a highly dependable indicator, employed as a benchmark or integral component for assessing strategies/alpha. It notably serves as a crucial metric for appraising alpha performance.

Formula: The Sharpe Ratio is calculated using the following formula:

$$\text{Sharpe Ratio} = \frac{\text{Rp} - \text{Rf}}{\sigma}$$

Where:

- Rp is the average return of the portfolio or strategy,
- Rf is the risk-free rate of return,
- σ is the standard deviation of the portfolio or strategy's excess return.

Given its simple yet powerful significance, we have chosen the Sharpe Ratio as the central focus in the ongoing refinement of our strategy. Our goal is to maximize this metric, seeking to achieve an optimal balance between risk and return. This strategy ensures that our approach not only generates positive returns but does so with a strong emphasis on effective risk management.

2.2.2 Position Size: Kelly’s Criterion

The Kelly criterion, conceived by John L. Kelly Jr. during his tenure at AT&T's Bell Laboratories, is a mathematical formula designed to govern the sustained growth of capital. Its application is prevalent among gamblers and investors seeking effective risk and money management strategies, aiding in the calculation of the appropriate percentage of their bankroll or capital to allocate in each bet or trade for optimal long-term growth.

Originally published in 1956 [14], the Kelly criterion found rapid adoption among gamblers who successfully employed it in the context of horse racing. Its integration into investment practices occurred later. In recent times, the strategy has experienced a revival, fueled by assertions that renowned investors like Warren Buffett and Bill Gross employ variations of the Kelly criterion.

This formula is embraced by investors aiming to expand their capital, assuming the reinvestment of profits and their exposure to future trade risks. The overarching objective of the formula is to ascertain the optimal amount to allocate to any given trade.

The Kelly Criterion formula is expressed as:

$$f * = \frac{bp - q}{b}$$

Where:

- f∗ is the fraction of the current capital to be invested,
- b is the net odds received on the wager (b to 1),
- p is the probability of success,
- q is the probability of failure (which is equal to 1 - p).

The formula helps determine the optimal percentage of capital to allocate in a bet or investment to maximize the long-term growth of capital while considering the probabilities of success and failure.

We will use this formula in some iteration of our strategy building process in the later part of this paper.

2.2.3 Entry/Exit: Dynamic Entry and Exit Strategy Integration

The cornerstone of a successful trading strategy lies in effective entry and exit points. In our exploration of various strategies, which predominantly encompass static stop loss/take profit or utilize indicators like the average true range, we have discerned the superior efficacy of trailing stop loss over traditional stop loss methods. Recognizing the absence of a dominant idea, we propose a unique approach: considering the choice between static stop loss, take profit, and trailing stop loss as a hyperparameter within a learning model. To address this, we employ grid search techniques to optimize this hyperparameter dynamically. This innovative integration aims to leverage the advantages of trailing stop loss while enhancing adaptability and responsiveness, thus contributing to the robustness of our overall trading strategy.

Integrating these auxiliary frameworks into our mean reversion strategy enhances its resilience and comprehensiveness, considering critical elements like risk management, position sizing, and trade execution. This all-encompassing strategy seeks to maximize both risk-adjusted returns and the sustained growth of capital over the long term.

The thorough theoretical framework established here serves as a sturdy basis for grasping and implementing both RSI and MACD within the realm of technical analysis. Subsequent sections will further explore empirical research and case studies to substantiate the effectiveness of RSI and MACD across diverse market conditions.

3. Technical Framework

QuantConnect is one of the leading platform for quant / trader to build trading strategies and deploy them to real life broker such as InteractiveBroker.

The technical framework primarily relies on Python, utilizing key components such as the QuantConnect library for both the research and strategy construction phases.

QuantConnect divides the strategy into separate projects, with each project typically having a main file, 'main.py,' for building strategies, and 'research.ipynb' for further exploratory data analysis (EDA) if needed.

To create a strategy for backtesting/live trading in QuantConnect, it's essential to create a strategy class that extends the QCAlgorithm class. The QCAlgorithm class plays a pivotal role in defining and implementing trading strategies within the QuantConnect platform. It provides a structured framework for coding and executing quantitative algorithms, streamlining the development and deployment of algorithmic trading strategies.

This class encapsulates key components such as Securities, Portfolio, Transactions, Schedule, Notify, and Universe, offering essential functionalities for data analysis and trading strategy implementation. The 'Initialize' method is employed for setting up requested data, cash, and time periods, while event handlers like 'OnData,' 'OnEndOfDay,' and 'OnEndOfAlgorithm' facilitate the algorithm's decision-making process during different stages of execution. Additionally, the class includes indicator helpers like SMA, enabling users to calculate Simple Moving Averages for specific symbols and periods.

```
class QCAlgorithm:
    Securities   # Array of Security objects.
    Portfolio    # Array of SecurityHolding objects
    Transactions # Transactions helper
    Schedule     # Scheduling helper
    Notify       # Email, SMS helper
    Universe     # Universe helper

    # Set up Requested Data, Cash, Time Period.
    def Initialize(self) -> None:

    # Other Event Handlers
    def OnData(self, slice: Slice) -> None:
    def OnEndOfDay(self, symbol: Symbol) -> None:
    def OnEndOfAlgorithm(self) -> None:

    # Indicator Helpers
    def SMA(self, symbol: Symbol, period: int) -> SimpleMovingAverage:
```

Figure 2. QC Algorithm Structure. Source: https://www.quantconnect.com/docs/v2/writing-algorithms/key-concepts/algorithm-engine

QuantBook is a wrapper class of QCAlgorithm, with extra functionality to support EDA process. This class is handy for more complex back testing in Jupyter notebook environments.

Regarding the transaction cost, we will follow InteractiveBroker commision scheme for IBKR Pro account, which is the default account available for all international individuals. The details can be found at https://www.interactivebrokers.com/en/pricing/commissions-home.php. In this research context, a good estimate for most transactions would be 1.00 USD per stock / contract, which is the minimum commission fee of an order.

InteractiveBrokers — Why IBKR | Pricing | Trading | Services | Education | Log In | Open Account

UNITED STATES

Monthly Volume (Contracts)[1]	Tiered
≤ 10,000	
Premium < USD 0.05	USD 0.25/contract
Premium ≥ USD 0.05 and <0.10	USD 0.50/contract
Premium ≥ USD 0.10	USD 0.65/contract
10,001-50,000	
Premium < USD 0.05	USD 0.25/contract
Premium ≥ USD 0.05	USD 0.50/contract
50,001-100,000	
All Premiums	USD 0.25/contract
≥ 100,001	
All Premiums	USD 0.15/contract
Minimum Per Order	USD 1.00

Figure 3. InteractiveBroker's pricing scheme for options / contract

Nevertheless, we set the brokerage model to InteractiveBroker by adding it into the initialization phase, and the fee will be calculated automatically during the backtest. self.SetBrokerageModel(BrokerageName.InteractiveBrokersBrokerage)

This comprehensive framework ensures a robust approach to researching potential stocks and constructing effective trading strategies in real life setting.

4. Methodology

We utilized Algoseek US Equity and Options Data, encompassing a universe of 27,500 US equities since January 1998. Our selection process involves identifying a subset of stocks that consistently demonstrate superior performance when subjected to mean reversion strategies. The dataset is accessible on the QuantConnect Cloud platform for QuantConnect users at https://www.quantconnect.com/datasets/algoseek-us-equities.

Initially, we implement and test basic strategies outlined in theoretical frameworks on prominent equities like the S&P 500 index ETF SPY, serving as a natural starting point. This choice is motivated by several reasons:

- S&P 500 stocks and indices usually represent a highly liquid market, minimizing slippage costs.
- Large-cap companies and markets, like the S&P 500, are less susceptible to manipulation.
- The S&P 500 is one of the most well-known equities, with data that is often well-documented and recorded.

Following the initial phase, we embark on an iterative process, systematically analyzing strategies with a keen emphasis on identifying areas for improvement. We then modify the strategies accordingly and repeat this cycle, aiming for continuous refinement and optimization of the mean reversion strategies employed. This iterative approach ensures a dynamic and adaptive strategy development process, leading to enhanced performance over time.

5. Results

All strategies will be executed within a one-minute timeframe, covering a backtest period from January 1, 2023, to January 1, 2024. The NYSE stock exchange operates with 252 trading days per year, featuring 6.5 trading hours each day (from 9:30 AM to 4:00 PM). This equates to a total of 98,280 data points (60 minutes per hour * 6.5 hours per day * 252 trading days). We find this time range subjectively reasonable — it is sufficiently small for us to execute a significant number of test iterations without consuming excessive time, yet large enough to yield convergent results.

Furthermore, we will share our backtest online through the QuantConnect link. To test the code, a reader can easily sign up for a free account and clone the algorithm to reproduce the results.

Alternatively, our code will also be included in the appendix of this paper. This additional measure ensures accessibility and transparency, allowing readers to review and reproduce our methodology with ease.

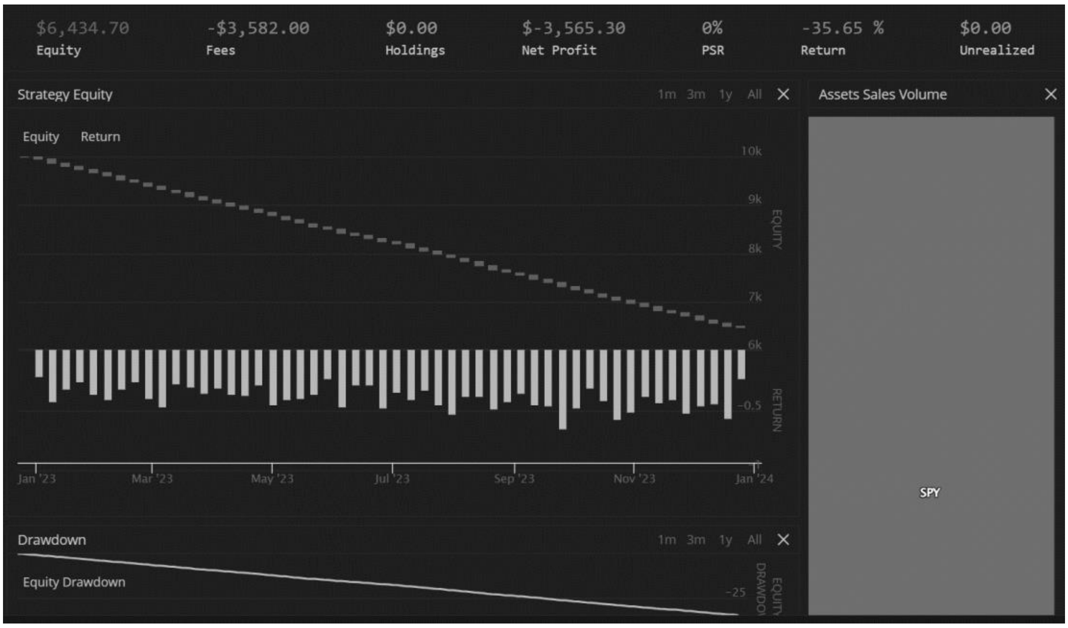
5.1 First Iteration: Default Mean Reversion strategy.

Our initial implementation involved a default RSI strategy applied to the S&P500 ETF index (ticker: SPY) with the following settings:

- Starting Amount: $10,000
- Overbought/Oversold limit: 70/30
- Market buy/sell 1 SPY stock when RSI crosses oversold/overbought limits
- Never hold more than 1/-1 position
- No stop loss/target profit set

QuantConnect backtest: https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_49734c18686a6e53ca5dd700e22b4758.html

We opted for this configuration as it represents the most basic form of the mean reversion strategy, and similar source codes are widely available on the platform and in the BootCamp section of the QuantConnect tutorial. This choice allows for a clear baseline comparison and facilitates the understanding of subsequent iterations.



PSR	0%	Sharpe Ratio	-22.075
Total Trades	3582	Average Win	0.01%
Average Loss	-0.03%	Compounding Annual Return	-35.835%
Drawdown	35.700%	Expectancy	-0.965
Net Profit	-35.653%	Sortino Ratio	-42.854
Loss Rate	98%	Win Rate	2%

Figure 4.1 & 4.2. First iteration result

The results were unexpectedly poor, even after experimenting with MACD and Bollinger Band. Beyond the disappointing Sharpe ratio, a noteworthy issue emerged in the form of a mere 2% win rate, significantly deviating from the anticipated ~50%. A closer examination of the trade history unveiled that most trades concluded with negative outcomes when factoring in a $1 transaction fee. In essence, 98% of the trades indicated that the SPY moved less than $1. Considering that in 2023, the ETF price never fell below 350, this implies that the underlying security's price movement was less than 1/350, approximately 0.29%, during most of the trades. This realization highlights a critical aspect of the strategy's performance and necessitates a reassessment of its underlying assumptions and parameters.

Date Time	Symbol	Type	Price	Quantity	Status
+ 2023-01-03 12:48:00	SPY	Buy Market	Fill: $372.45 USD	1	Filled
+ 2023-01-03 13:27:00	SPY	Sell Market	Fill: $373.87 USD	-1	Filled
+ 2023-01-03 13:28:00	SPY	Sell Market	Fill: $373.71 USD	-1	Filled
+ 2023-01-03 14:13:00	SPY	Buy Market	Fill: $373.45 USD	1	Filled

Figure 4.3. Log of first few transaction

For instance, the trade executed from 13:28 to 14:13, yielding a gross profit of 26 cents, transformed into a loss after accounting for commission fees.

This iteration highlighted a crucial observation: unless a security is sufficiently large or volatile, transaction costs can become a detrimental factor. Intraday trading often involves a substantial number of trades, as evident in this example with 3,582 transactions. Without the fee, the strategy would have yielded a minor profit (+$16.7). However, even without the transaction cost, the overall result remains mediocre.

We also repeated the same back-testing with MACD and Bollinger Band as well with similar result.

5.2 Second Iteration: Leverage with Derivative Security.

In response to the challenges highlighted in prior sections, where transaction costs significantly impacted performance and the underlying asset exhibited limited price movement, leveraging emerged as a key strategy to address these issues.

There are several popular methods for leverage, such as margin accounts or loans. However, given the minimal price movement (0.2%), derivatives quickly emerged as a promising candidate, given their almost unique ability to extremely leverage. By amplifying returns to this factor, we hope to counteract the impact of transaction fees effectively.

Initially, we explored the 5 DTE SPY option chain, which yielded a modestly positive Sharpe Ratio. Seeking further amplification, we pursued an alternative solution—utilizing the futures market, specifically the E-Mini S&P 500 Future Continuous Contract. This derivative closely tracks the S&P 500, with a contract unit of $50 per index point.

Since E-Mini is a relatively specialized instrument, we realized that slippage might be a risk factor. Thus, to both simplify the process and mititgate this risk, we prioritize contracts with the highest open interest to ensure liquidity, recognizing that futures are not as liquid as ETFs. However, the backtesting result quickly yielded a negative Sharpe ratio.

Further investigation uncovered instances where our purchased contracts simply expired, resulting in a 100% loss of over $3,000. Suspecting that this might be linked to futures contracts with zero days until expiration, which tend to have higher overall open interest, we shifted our focus to contracts with more than 5 days until expiration to alleviate the impact of time decay.

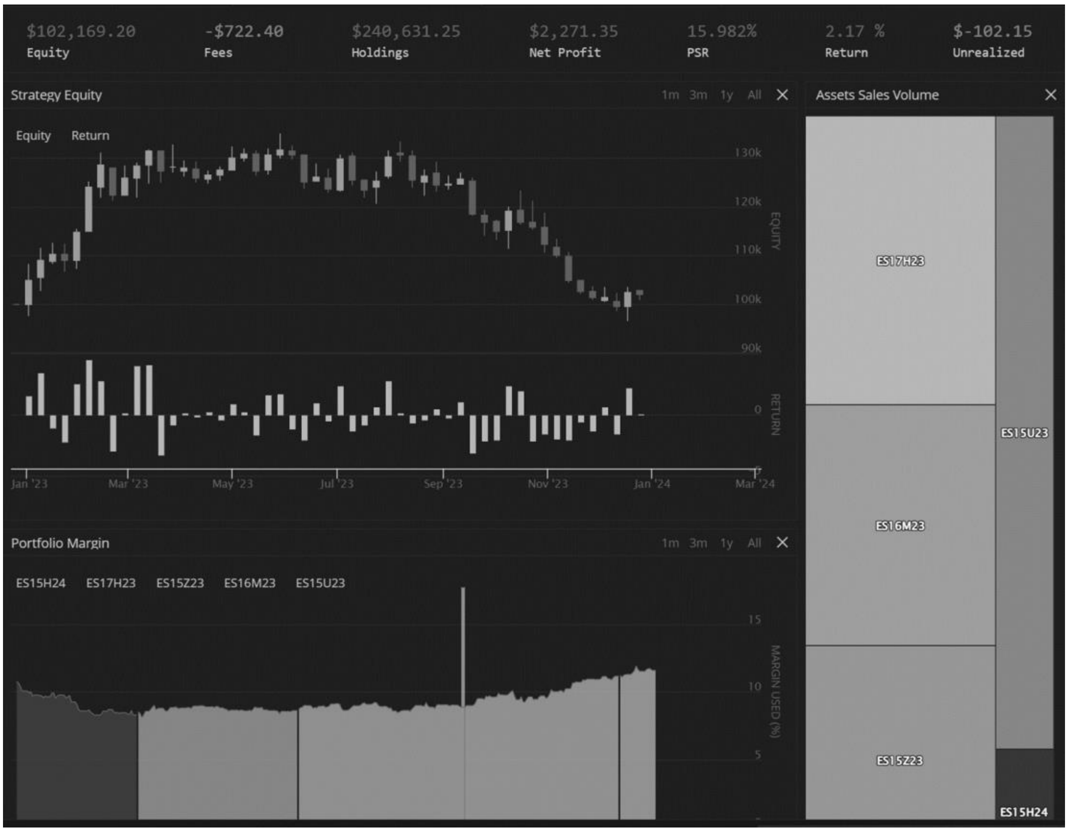
This experience prompted us to prioritize basic risk management measures. Consequently, we implemented a hard rule of a 1% stop loss/take profit to mitigate the risk of losing the entire contract value if it expires, aligning with the prudent trading principle that a loss should not exceed 1%.

Finally, recognizing the substantial value of each futures contract, amounting to a few thousand dollars, we increased the starting capital from $10,000 to $100,000. This adjustment ensures the sustainability of the strategy in the long run.

On of our most decent backtest during this iteration involved used default RSI strategy applied to the E-mini S&P 500 with the following settings:

- Starting Amount: $100,000
- Overbought/Oversold limit: RSI cross 70 / 30
- Market buy/sell 1 E-mini contract between 5-120 day to expire with highest open interest instead of 1 SPY position, when MACD indicator reach oversold/overbought condition.
- For each market buy / sell, attached take profit / stop loss order at 1.01 times and 0.99 times the market price.

The backtest is available online at https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_44f576e1c2e3f11ef3bd30d7a887d201.html



PSR	15.982%	Sharpe Ratio	-0.084
Total Trades	337	Average Win	1.77%
Average Loss	-1.84%	Compounding Annual Return	2.169%
Drawdown	27.900%	Expectancy	0.016
Net Profit	2.169%	Sortino Ratio	-0.123
Loss Rate	48%	Win Rate	52%

Figure 5.1 & 5.2. Second iteration result – RSI strategy - After switching to E-mini Futures

After implementing all the aforementioned changes, we achieved a substantial improvement from iteration 1. The strategy is now slightly profitable, aligning with the iteration 1 result without factoring in transaction costs.

A notable observation is the significant reduction in the number of trades, decreasing from over 3000 to around 300. This reduction is a consequence of switching our underlying securities from SPY to E-mini S&P 500, making it less likely to enter overbought/oversold ranges. This strategic shift reflects an adjustment to the chosen securities, potentially mitigating challenges associated with frequent overbuying or overselling conditions. The positive impact on profitability and trade frequency highlights the effectiveness of this modification in enhancing overall strategy performance.

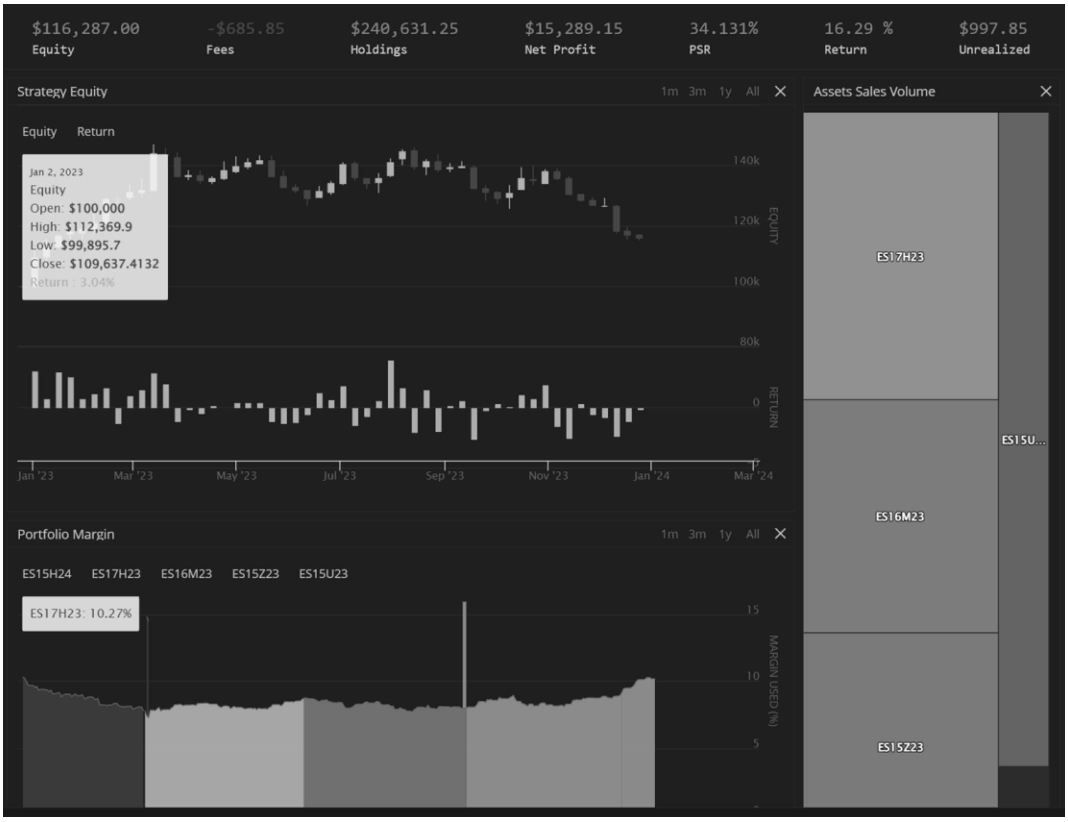
Nonetheless, the Sharpe Ratio is still in the negative side. Thus, we further tested default MACD strategy with following config.

- Starting Amount: $100,000

- Overbought/Oversold limit: When MACD line crosses above / below the Signal line.
- Market buy/sell 1 E-mini contract between 5-120 day to expire with highest open interest instead of 1 SPY position, when MACD indicator reach oversold/overbought condition
- For each market buy / sell, attached take profit / stop loss order at 1.01 times and 0.99 times the market price.

This strategy yielded a Sharpe Ratio of 0.421. Despite it's well under 1 since it was the better strategy out of the two, we decided to choose MACD as the core strategy going forward.

https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_cd368f954695b5b8ee6abb70a8c69f6c.html



PSR	34.131%	Sharpe Ratio	0.421
Total Trades	321	Average Win	1.67%
Average Loss	-1.67%	Compounding Annual Return	16.287%
Drawdown	21.400%	Expectancy	0.062
Net Profit	16.287%	Sortino Ratio	0.654
Loss Rate	47%	Win Rate	53%

Figure 5.3 & 5.4. Second iteration result – MACD strategy - After switching to E-mini Futures

5.3 Iteration 3: Fine-Tuning Win Rate by Modifying Original Indicators.

In this iteration, we focused on refining our strategy's performance by making adjustments to the original indicators and their parameters. The goal was to optimize the win rate and overall profitability.

5.3.1 Sub Iteration 1: Grid Searching Indicator Parameters.

Treating model parameters as hyperparameters in machine learning models, we conducted a grid search technique to explore various combinations of indicator parameters. Starting with the common parameters indicated in previous iterations, we systematically adjusted one parameter at a time. For instance, the common setting for the Moving Average Convergence Divergence (MACD) is (12, 26, 9), we explored variations like MACD(10-14, 26, 9), MACD(12, 24-28, 9), and MACD(12, 26, 7-11).

Despite meticulous adjustments, we observed random fluctuations in the Sharpe ratio during this sub-iteration. The challenge lay in finding a parameter combination that consistently improved the strategy's performance.

5.3.2 Sub Iteration 2: Adjusting Overbuy/Oversold Threshold.

Another avenue we explored to enhance the strategy's win rate was by modifying the overbuy/oversold conditions.

The idea is simple: instead of waiting for MACD line to cross the Signal line to be the entry for our position, we can instead measure the distance between those two lines by subtract the value of MACD line and Signal Line. When the distance crosses the threshold, we called “tolerance” then we enter the position.

We performed a grid search for the optimal distance and simply chose the best configuration according to Sharpe Ratio.

Table 1. MACD stategy performance with different tolerance level

Tolerance	Trade Win-rate	Sharpe Ratio	Number of Trade	Return
0	53%	0.421	321	16.29%
0.05	54%	0.741	321	24.64%
0.1	55%	1.03	323	31.37%
0.15	52%	-0.009	317	4.60%
0.2	52%	0.007	321	5%

At the end of this iteration, we got a decent strategy with Sharpe Ratio of 1.03, with 55% win rate, 12% drawdown with the following config:

- Starting Amount: $100,000
- Overbought/Oversold limit:
    - Overbought: When MACD line crosses above Signal line + 0.1 line.

- Oversold: When MACD line crosses below Signal line - 0.1 line.
- Market buy/sell 1 E-mini contract between 5-120 day to expire with highest open interest instead of 1 SPY position, when MACD indicator reach oversold/overbought condition.
- For each market buy / sell, attached take profit / stop loss order at 1.01 times and 0.99 times the market price.

https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_e594f9c925d01a2d5e44b714cdb994b7.html

PSR	60.183%	Sharpe Ratio	1.03
Total Trades	323	Average Win	1.66%
Average Loss	-1.64%	Compounding Annual Return	31.370%
Drawdown	14.400%	Expectancy	0.112
Net Profit	31.370%	Sortino Ratio	1.613
Loss Rate	45%	Win Rate	55%
Profit-Loss Ratio	1.01	Alpha	0.185

Figure 6.1 & 6.2. Third iteration result – MACD strategy with E-mini Futures with adjusted overbuy / oversold threshold

In summary, Iteration 3 involved a thorough exploration of indicator parameters and overbuy/oversold thresholds to fine-tune the strategy's win rate. The challenges encountered underscored the complexity of finding a consistent and optimal set of parameters, emphasizing the need for further refinement and experimentation in subsequent iterations.

Nevertheless, we still have one more principal aspect that can be improved the portfolio margin.

5.4 Fourth Iteration: Managing Capital Size with Kelly Criterion

In this iteration, our target is to define best starting capital amount to allocate to this strategy. To this problem, Kelly Criterion has been one of the most popular strategies for allocation.

From the result of the third iteration, we have profit – loss ratio = 1.01, win rate = 55%.

$$f* = \frac{bp - q}{b} = \frac{1.01 * 0.55 - 0.45}{1.01} \sim 0.11$$

This means for each trade we should risk about 11% of our capital.

Even though we have set stop loss and take profit for each of the orders. Technically in the worst-case scenario our option would have expire which is about $3500 – $4800 per contract. Thus, in this iteration we set our starting amount to $50000 (~ 4800 / 0.11) and rerun.

PSR	67.677%	Sharpe Ratio	1.466
Total Trades	323	Average Win	2.72%
Average Loss	-2.60%	Compounding Annual Return	62.740%
Drawdown	21.600%	Expectancy	0.131
Net Profit	62.740%	Sortino Ratio	2.396
Loss Rate	45%	Win Rate	55%

Figure 7.1 & 7.2. Fourth iteration result – MACD strategy with E-mini Futures with adjusted overbuy / oversold threshold and adjusted amount of capital

After this iteration, the Sharpe ratio sharply improved to 1.466. However, due to the starting margin is smaller, the drawdown increased subsequently.

5.5 Fifth iteration: Trailing stop loss & entry point adjustment.

In previous iterations, we adhered to the rule of thumb setting the stop loss/take profit at 1% of the position. Intuitively, we attempted to fine-tune this parameter through another grid search. However, coincidentally, the strategy performed best with a 1% take profit and stop loss based on the configuration.

As we thoroughly examined and optimized various aspects in the process of building the strategy, we have concluded our first cycle in constructing a mean reversion alpha. This comprehensive exploration and refinement process has provided valuable insights, allowing us to better understand the intricacies of the strategy and refine it for improved performance.

Our final config for the first attempt is as follows:

- Starting Amount: $50,000
- Overbought/Oversold limit:
    - Overbought: When MACD line crosses above Signal line + 0.1 line.
    - Oversold: When MACD line crosses below Signal line - 0.1 line.
- Market buy/sell 1 E-mini contract between 5-120 day to expire with highest open interest instead of 1 SPY position, when MACD indicator reach oversold/overbought condition.
- For each market buy / sell, attached take profit / stop loss order at 1.01 times and 0.99 times the market price.

https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_79f41a4cf37551bf7ca0c600838b9c1f.html

5.6 Second cycle: Readjust default MACD strategy different stop loss and take profit level.

We didn’t yield meaningful result in the final iteration of the first attempt, however, we still wanted to test if the default MACD strategy without fine-tuning parameters will be benefit from take profit / stop loss instead.

PSR	70.681%	Sharpe Ratio	1.758
Total Trades	305	Average Win	3.89%
Average Loss	-2.11%	Compounding Annual Return	92.653%
Drawdown	31.000%	Expectancy	0.237
Net Profit	92.653%	Sortino Ratio	2.196
Loss Rate	57%	Win Rate	43%
Profit-Loss Ratio	1.85	Alpha	0.706

Figure 8.1 & 8.2. Second cycle result with different optimization approach

https://www.quantconnect.com/terminal/processCache?request=embedded_backtest_41dc41bb40890ae330726d38917f2773.html

Thus, we have initiated another cycle to build a new instance of the MACD strategy. In this attempt, we repeated all the iterations from the first attempt, excluding iteration 3 where we kept the default MACD strategy. During one of these iterations, we encountered a backtest result with a Sharpe Ratio of 1.758, using the following configuration:

- Starting Amount: $50,000
- Overbought/Oversold limit: When MACD line crosses above / below the Signal line.
- Market buy/sell 1 E-mini contract between 5-120 day to expire with highest open interest instead of 1 SPY position, when MACD indicator reach oversold/overbought condition.
- For each market buy / sell, attached take profit / stop loss as below:
    - Trailing stop loss order 1% amount of the contract price
    - Hard take profit at 3% the contract price

There is an interesting observation when we compare the performance. Despite the lower win rate, the Sharpe ratio reached an impressive 1.75. This can be attributed to the profit-loss reward ratio being much better at 1.88, with the maximum profit around three times the maximum loss.

At this point, the process becomes more and more non-linear as more observation emerge. For example, we began our third attempt with the observation that we can have a better strategy as long as the profit – loss ratio is good.

With this idea, we can continue repeating the process to improve based on further observation and idea. Combining MACD and RSI to tighten entry conditions which tightens entry conditions to improve win rate, for instance, could have been another approach we choose for iteration 3 of first attempt.

As we believe that we have gathered sufficient observations and outlined a systematic process for building real-life trading strategies, which was our original objective, we have concluded the process at this point. Any additional ad-hoc observations from our exploration will be briefly discussed in the Discussion section. This marks the completion of our initial cycle of strategy development, providing a foundation for future refinements and iterations based on the insights gained.

6. Discussion

6.1 Insight from previous result

Throughout the previous sections, we successfully developed several ready-to-deploy trading strategies in real life using existing online resources and encountered unique challenges not commonly addressed in theoretical research articles. In this section, we aim to consolidate the distinct problems faced throughout the project and elaborate on how we overcame them:

- Transactions cost can be a huge problem, can be handled with proper leveraging. Whereas this aspect is often neglected in research environments.
- Despite being the same strategy (e.g MACD), a large amounts of variations with significantly different performance can still be generated due to different combination of parameter. Especially the fact that some ideas for fine tuning might only work for specific configurations. We came across several paper that offer solution that work in their dataset but become awful when we adapted to our dataset. In the end's it's the fine-tuning process that make the magic work, even though researching can offer an excellent head start at the beginning.
- The strategy building process is not linear (as what we experienced with our trailing stop loss approach doesn't work for some iteration). The process is much more akin to finding a maximum value of a function with multiple local maxima. Thus, several distinct attempt in optimizing the strategies should be encouraged.

6.2 Limitation

6.2.1 Solely choosing Sharpe Ratio to optimize

Firstly, regarding the selection of performance metrics, we chose to primarily focus on the Sharpe Ratio despite acknowledging the importance of other key indicators such as maximum drawdown, profit margin, and turnover rate. Singularly, the Sharpe Ratio is favored over metrics like Sortino and Treynor ratios. Given these complexities, we deemed the Sharpe Ratio to be reasonable within the scope of this study, providing a comprehensive, objective, and manageable assessment of strategy performance. Introducing multiple benchmarks can pose a trade-off problem; however, the derivation of composite metrics from multiple indicators can potentially offer a better assessment. Yet, this approach requires careful consideration and time for validation.

6.2.2 On back-testing strategy

We are aware that in this back testing we are backtesting the strategy on the whole dataset which can cause an overfitting problem. Even though this can be improved by introducing Outsample Data Testing method, we find that the strategy we tested can have very high volatility in term of Sharpe Ratio for small amount of time, which reduce the meaning of out sample testing result, while time and resource constraints hindered us to test on a larger dataset.

Nonetheless, due to the nature of our indicator, where the indicator only take value from a set number of past data point (14 for RSI, 26 for MACD), by simulating historical data we already unconsciously did a Walk Forward Analysis backtesting.

To partially avoid overfitting problem, we also limited the number of variations that we apply to default mean reversion strategies.

6.2.3 Using techniques from other families.

Due to the scope of the project, our focus was exclusively on mean reversion techniques in the indicator section. While mean reversion is a robust approach, various aspects of the strategies could potentially benefit from the inclusion of other indicators. For instance, indicators like Average True Range can be employed to identify optimal stop loss and take profit levels. However, for the sake of project elegance and within the constraints of our timeframe, we chose to adhere to basic techniques in this iteration. The incorporation of additional indicators remains a potential avenue for future refinement and expansion of the strategies.

6.3 Unsuccessful iteration

We underwent numerous iterations in an effort to enhance the strategy. While we successfully discovered several ideas that contributed to improvements, it's important to acknowledge that there were also iterations where we fell short of reaching our targets. Nevertheless, these experiences provided valuable insights and spawned additional ideas that are worth mentioning and could potentially serve as the foundation for future work. In this section, we will briefly highlight some of these noteworthy experiences and ideas.

6.3.1 Using other underlying security.

Another potential avenue for reducing risk is diversifying the portfolio. While SPY inherently offers partial diversification, individual stocks might exhibit better mean-reversion characteristics. One approach to identify suitable candidates is by conducting a unit root test, such as the Augmented Dickey Fuller Test on log prices. This process would involve a similar strategy-building approach to what we demonstrated with SPY in this paper. For example, E-mini contracts could be replaced with long-term-to-expire options, although liquidity trade-offs must be considered.

We have included a version of the ADF test for all S&P500 constituents in the appendix section. The results highlight 20 potential single stocks with the best statistics from running the code. This exploration into individual stock mean-reversion characteristics could present an interesting avenue for further research and strategy development.

	Symbol	Test Statistics
367	PFE	-2.538307
199	FMC	-1.930975
73	BMY	-1.781146
148	DG	-1.707736
120	CAG	-1.670008
234	HRL	-1.632312
318	MRNA	-1.383215
23	AMCR	-1.367715
214	GIS	-1.362823
177	EL	-1.354828
181	ES	-1.332995
150	D	-1.270803
82	CPB	-1.259273
7	AES	-1.221950
416	SJM	-1.196813
336	NEE	-1.172727
227	HSY	-1.140818
168	ENPH	-1.136022
476	WBA	-1.124278
270	KVUE	-1.097410

Figure 9: ADF test result for different S&P500 stocks

However, several challenges arise in pursuing this approach:

- **Data Volume and Testing Time:** The sheer amount of data would make testing time-consuming if applying MACD/RSI separately to each stock.
- **Correlation between Stocks:** The intercorrelation between stocks is another challenge to consider, as it could impact the effectiveness of mean-reversion strategies across a diverse portfolio.
- **Liquidity and Transaction Costs:** Stock options are generally more illiquid and limited compared to E-mini 500 S&P, and transaction costs tend to be higher. This adds an additional layer of complexity and cost considerations.

Despite these challenges, this methodology provides a systematic way to identify potential candidates for applying mean-reversion strategies, particularly those capable of managing multiple stocks in a portfolio, such as the "On-Line Moving Average Reversion" strategy. Further exploration and refinement of this approach could yield valuable insights into enhancing mean-reversion strategies across diverse assets.

6.3.2 Identify "Golden Window" for trading.

During inspecting our trading strategy, we also broke down the bot performance into smaller timeframe and discovered that there are specific periods where our strategy will consistently perform surprisingly across our iterations. Nonetheless, since there are many potential lead, some of which is not in technical analysis realm, it is formidable to estimate the amount of research on this idea. We have been finding the cause for this event, which could offer a high reward for future work.

	1 Month
01/31/2023	-1.4569
02/28/2023	4.2744
03/31/2023	29.3181
04/30/2023	4.6835
05/31/2023	-1.9879
06/30/2023	-1.3152
07/31/2023	-2.5382
08/31/2023	37.0552
09/30/2023	2.6420
10/31/2023	1.0272
11/30/2023	15.3713
12/31/2023	-1.4036
01/01/2024	0

Figure 10. Trailing 1 month Sharpe Ratio of strategy in section 5.6

7. Conclusion

This applied research endeavors to bridge the gap between theoretical strategies and their practical implementation in the realm of intraday trading on the New York Stock Exchange. By leveraging resources from existing journals and advancements in quantitative platforms such as QuantConnect, we translated ideas from theoretical research and books into a framework applicable in real-life environments. This effort resulted in ready-to-deploy strategies capable of executing live trades on brokers.

Furthermore, this study provides unique insights into the challenges encountered during this process. The iterative approach taken in this research aims to deliver a practical and deployable Mean-Reversion strategy that not only surpasses market performance but also addresses real-world challenges and intricacies.

This research directly addresses the lack of theoretical trading research that is applicable to real-life environments by translating theoretical methodology into a working trade strategy through a newly discovered framework. The detailed discussion and analysis of each iteration contribute to the development of a robust and adaptive intraday trading strategy. However, we acknowledge the inherent complexity and ever-changing nature of the market, necessitating frequent updates and adaptations to ensure the strategy remains relevant.

References

[1] Barber, B. M., & Odean, T. (2000). "Trading is hazardous to your wealth: The common stock investment performance of individual investors." *The Journal of Finance*, 55(2), 773-806.

[2] Barber, B. M., Lee, Y.-T., Liu, Y.-J., & Odean, T. (2004). "Do Individual Day Traders Make Money? Evidence from Taiwan." Available at: https://faculty.haas.berkeley.edu/odean/papers/Day%20Traders/Day%20Trade%20040330.pdf

[3] Lo, A. W., MacKinlay, A. C. (1988). "Stock Market Prices Do Not Follow Random Walks: Evidence from a Simple Specification Test." *The Review of Financial Studies*, 1(1), 41–66. https://doi.org/10.1093/rfs/1.1.41

[4] Bollinger, J. (2002). *Bollinger on Bollinger Bands*. New York: McGraw-Hill.

[5] Chong, T. T.-L., Ng, W.-K. (2008). "Technical analysis and the London stock exchange: testing the MACD and RSI rules using the FT30." *Applied Economics Letters*, 15(14), 1111-1114.

[6] Ghosh, P., Neufeld, A., & Sahoo, J. K. (2020). "Forecasting directional movements of stock prices for intraday trading using LSTM and random forests." *arXiv*. https://doi.org/10.48550/arXiv.2004.10178

[7] Faraz, M., Khaloozadeh, H., & Abbasi, M. (2020). "Stock Market Prediction-by-Prediction Based on Autoencoder Long Short-Term Memory Networks." *2020 28th Iranian Conference on Electrical Engineering (ICEE)*, 1-5. doi: 10.1109/ICEE50131.2020.9261055

[8] Taroon, G., Tomar, A., Manjunath, C., Balamurugan, M., Ghosh, B., & Krishna, A. V. N. (2020). "Employing Deep Learning In Intraday Stock Trading." *2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, 209-214. doi: 10.1109/ICRCICN50933.2020.9296174

[9] Miao, Y. (2020). "A Deep Learning Approach for Stock Market Prediction." *Computer Science Department, Stanford University*.

[10] Wilder Jr., J. W. (1978). *New Concepts in Technical Trading Systems*. Trend Research. https://archive.org/details/newconceptsintec00wild/page/5/mode/2up

[11] Appel, G. (2005). *Technical Analysis Power Tools for Active Investors*. Financial Times Prentice Hall. p. 166. ISBN 0-13-147902-4.

[12] Pardo, R. (2011). *The Evaluation and Optimization of Trading Strategies*. John Wiley & Sons.

[13] Stanford University. "The Sharpe Ratio." https://web.stanford.edu/~wfsharpe/art/sr/SR.htm

[14] Kelly, J. L. (1956). "A New Interpretation of Information Rate." *Bell System Technical Journal*, 35(4), 917–926. doi:10.1002/j.1538-7305.1956.tb03809.x

Disclaimer

This paper was created as part of a WorldQuant University degree program towards an MSc in Financial Engineering. This paper is reproduced with the consent and permission of WorldQuant University.

Appendix

Source Code: Default RSI strategy on SPY (section 5.1)

```
# Import necessary modules and classes
from AlgorithmImports import *
from QuantConnect.Algorithm.Framework import *
from QuantConnect.Algorithm.Framework.Selection import *
from QuantConnect.Algorithm.Framework.Portfolio import *
from QuantConnect.Indicators import *

# Define a class for the RSI-based trading strategy
class RSIStrategy(QCAlgorithm):
    def Initialize(self):
        # Set the start date for backtesting
        self.SetStartDate(2023, 1, 1)
        # Set the end date for backtesting
        self.SetEndDate(2024, 1, 1)
        # Set initial cash balance
        self.SetCash(10000)

        # Add SPY data
        security = self.AddEquity("SPY")

        # Initialize RSI, Bollinger Bands, and MACD indicators
        self.rsi = self.RSI("SPY", 14, MovingAverageType.Simple,
Resolution.Minute)
        self.bollinger = self.BB("SPY", 20, 2, MovingAverageType.Simple,
Resolution.Minute)
        self.macd = self.MACD("SPY", 12, 26, 9, MovingAverageType.Exponential,
Resolution.Minute)

        # Set warm-up period to ensure the indicators have enough data
        self.SetWarmUp(26)

        # Set brokerage model to Interactive Brokers
        self.SetBrokerageModel(BrokerageName.InteractiveBrokersBrokerage)

        # Set benchmark to SPY
        self.SetBenchmark("SPY")

    def OnData(self, data):
        # Check if indicators are ready
        if not self.rsi.IsReady or not self.bollinger.IsReady or not self.macd.IsReady:
```

```
            return

        # Get the current SPY holding
        spy_holdings = self.Portfolio["SPY"]

        # Set tolerance level for MACD signal delta
        tolerance = 0.000
        signalDeltaPercent = self.macd.Current.Value -
self.macd.Signal.Current.Value

        # Trading logic based on RSI, Bollinger Bands, and MACD
        if (
            # RSI condition
            self.rsi.Current.Value > 70
            # Bollinger Bands condition
            #and self.bollinger.UpperBand.Current.Value < data["SPY"].Close
            # MACD condition
            # signalDeltaPercent > tolerance
            and spy_holdings.Quantity >= 0
        ):
            # If conditions met, sell SPY
            self.MarketOrder("SPY", -1)
            # Uncomment the following lines for additional order types
            #self.takeProfit = self.LimitOrder("SPY", -1, data["SPY"].Close*1.01)
            #self.stopLoss = self.StopMarketOrder("SPY", -1,
data["SPY"].Close*0.99)
        elif (
            # RSI condition
            self.rsi.Current.Value < 30
            # Bollinger Bands condition
            #and self.bollinger.LowerBand.Current.Value > data["SPY"].Close
            # MACD condition
            #signalDeltaPercent < -tolerance
            and spy_holdings.Quantity <= 0
        ):
            # If conditions met, buy SPY
            self.MarketOrder("SPY", 1)
            # Uncomment the following lines for additional order types
            #self.takeProfit = self.LimitOrder("SPY", 1, data["SPY"].Close*0.99)
            #self.stopLoss = self.StopMarketOrder("SPY", 1, data["SPY"].Close*1.01)
```

Source Code: RSI strategy on E-Mini S&P500 (section 5.2)

```
from QuantConnect.Indicators import MovingAverageConvergenceDivergence

# Import necessary modules and classes
from AlgorithmImports import *

# Define a class for the RSI-based trading strategy
```

```
class RSI(QCAlgorithm):

    def Initialize(self):
        # Set brokerage model to Interactive Brokers
        self.SetBrokerageModel(BrokerageName.InteractiveBrokersBrokerage)
        # Set the start date for backtesting
        self.SetStartDate(2023, 1, 1)
        # Set the end date for backtesting
        self.SetEndDate(2024, 1, 1)
        # Set initial cash balance
        self.SetCash(100000)
        # Add E-mini S&P 500 futures data
        self.spy = self.AddFuture(Futures.Indices.SP500EMini)
        # Set the filter for the futures contract
        self.spy.SetFilter(5, 120)


        # Initialize variables
        self.oi_contract = None
        self.macd = None
        self.takeProfit = None
        self.stopLoss = None

    def OnData(self, slice):
        # Iterate through available future chains in the data slice
        for chain in slice.FutureChains:
            contracts = [contract for contract in chain.Value]
            if len(contracts) == 0:
                self.oi_contract = None
                self.macd = None
                break

            # Select the contract with the highest open interest
            contract = sorted(contracts, key=lambda k : k.OpenInterest,
reverse=True)[0]

            # If the selected contract is the same as the previous one, skip processing
            if self.oi_contract is not None and contract.Symbol ==
self.oi_contract.Symbol:
                break

            # Update the selected contract and calculate RSI
            self.oi_contract = contract
            self.rsi = self.RSI(contract.Symbol, 14, MovingAverageType.Simple,
Resolution.Minute)

        # Check if RSI is available and ready
        if self.rsi is None or not self.rsi.IsReady:
            return
```

```
        # Get information about the selected contract
        symbol = self.oi_contract.Symbol
        security = self.Securities[symbol]
        price = security.Price

        # Only new positions for which the algorithm is not invested
        if security.Invested:
            # Look to exit the position
            return

        # Trading logic based on RSI conditions
        if self.rsi.Current.Value < 30 and self.Portfolio[symbol].Quantity <= 0:
            # Go long
            self.MarketOrder(symbol, 1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, -1, price*1.01)
            self.stopLoss = self.StopMarketOrder(symbol, -1, price*0.99)
        if self.rsi.Current.Value > 70 and self.Portfolio[symbol].Quantity >= 0:
            # Go short
            self.MarketOrder(symbol, -1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, 1, price*0.99)
            self.stopLoss = self.StopMarketOrder(symbol, 1, price*1.01)

    def OnOrderEvent(self, orderEvent):
        # Cancel the corresponding order if the other one is filled
        if orderEvent.Status != OrderStatus.Filled:
            return
        self.Cancel(orderEvent.OrderId)

    def Cancel(self, id):
        '''Cancel one order if the other was filled'''
        if self.takeProfit is not None and id == self.takeProfit.OrderId:
            self.stopLoss.Cancel()
        elif self.stopLoss is not None and id == self.stopLoss.OrderId:
            self.takeProfit.Cancel()
        else:
            return
        self.takeProfit = None
        self.stopLoss = None
```

Source Code: MACD strategy on E-Mini S&P500 with tolerance (section 5.2 & 5.3)

```
from QuantConnect.Indicators import MovingAverageConvergenceDivergence

# Import necessary modules and classes
from AlgorithmImports import *

# Define a class for the MACD-based trading strategy
```

```
class MACD(QCAlgorithm):

    def Initialize(self):
        # Set brokerage model to Interactive Brokers
        self.SetBrokerageModel(BrokerageName.InteractiveBrokersBrokerage)
        # Set the start date for backtesting
        self.SetStartDate(2023, 1, 1)
        # Set the end date for backtesting
        self.SetEndDate(2024, 1, 1)
        # Set initial cash balance
        self.SetCash(100000)
        # Add E-mini S&P 500 futures data
        self.spy = self.AddFuture(Futures.Indices.SP500EMini)
        # Set the filter for the futures contract
        self.spy.SetFilter(5, 120)

        # Initialize variables
        self.oi_contract = None
        self.macd = None
        self.takeProfit = None
        self.stopLoss = None

    def OnData(self, slice):
        # Iterate through available future chains in the data slice
        for chain in slice.FutureChains:
            contracts = [contract for contract in chain.Value]
            if len(contracts) == 0:
                self.oi_contract = None
                self.macd = None
                break

            # Select the contract with the highest open interest
            contract = sorted(contracts, key=lambda k : k.OpenInterest,
reverse=True)[0]

            # If the selected contract is the same as the previous one, skip processing
            if self.oi_contract is not None and contract.Symbol ==
self.oi_contract.Symbol:
                break

            # Update the selected contract and calculate MACD
            self.oi_contract = contract
            self.macd = self.MACD(contract.Symbol, 12, 26, 9,
MovingAverageType.Exponential, Resolution.Minute)

        # Check if MACD is available and ready
        if self.macd is None or not self.macd.IsReady:
            return

        # Get information about the selected contract
```

```
        symbol = self.oi_contract.Symbol
        security = self.Securities[symbol]
        price = security.Price

        # Only new positions for which the algorithm is not invested
        if security.Invested:
            # Look to exit the position
            return

        # Define tolerance and calculate the MACD signal delta percentage
        tolerance = 0.1
        signalDeltaPercent = self.macd.Current.Value -
self.macd.Signal.Current.Value

        # Trading logic based on MACD signal delta
        if signalDeltaPercent < -tolerance and self.Portfolio[symbol].Quantity <= 0:
            # Go long
            self.MarketOrder(symbol, 1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, -1, price*1.01)
            self.stopLoss = self.StopMarketOrder(symbol, -1, price*0.99)
        if signalDeltaPercent > tolerance and self.Portfolio[symbol].Quantity >= 0:
            # Go short
            self.MarketOrder(symbol, -1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, 1, price*0.99)
            self.stopLoss = self.StopMarketOrder(symbol, 1, price*1.01)

    def OnOrderEvent(self, orderEvent):
        # Cancel the corresponding order if the other one is filled
        if orderEvent.Status != OrderStatus.Filled:
            return
        self.Cancel(orderEvent.OrderId)

    def Cancel(self, id):
        '''Cancel one order if the other was filled'''
        if self.takeProfit is not None and id == self.takeProfit.OrderId:
            self.stopLoss.Cancel()
        elif self.stopLoss is not None and id == self.stopLoss.OrderId:
            self.takeProfit.Cancel()
        else:
            return
        self.takeProfit = None
        self.stopLoss = None
```

Source Code: MACD with Trailing Stop Loss

```
# Import necessary modules and classes
from QuantConnect.Indicators import MovingAverageConvergenceDivergence
```

```
from AlgorithmImports import *

# Define a class for the MACD-based trading strategy
class MACD(QCAlgorithm):

    def Initialize(self):
        # Set brokerage model to Interactive Brokers
        self.SetBrokerageModel(BrokerageName.InteractiveBrokersBrokerage)
        # Set the start date for backtesting
        self.SetStartDate(2023, 1, 1)
        # Set the end date for backtesting
        self.SetEndDate(2024, 1, 1)
        # Set initial cash balance
        self.SetCash(50000)
        # Add E-mini S&P 500 futures data
        self.spy = self.AddFuture(Futures.Indices.SP500EMini)
        # Set the filter for the futures contract
        self.spy.SetFilter(5, 120)

        # Initialize variables
        self.oi_contract = None
        self.macd = None
        self.takeProfit = None
        self.stopLoss = None

    def OnData(self, slice):
        # Iterate through available future chains in the data slice
        for chain in slice.FutureChains:
            contracts = [contract for contract in chain.Value]
            if len(contracts) == 0:
                self.oi_contract = None
                self.macd = None
                break

            # Select the contract with the highest open interest
            contract = sorted(contracts, key=lambda k : k.OpenInterest,
reverse=True)[0]

            # If the selected contract is the same as the previous one, skip processing
            if self.oi_contract is not None and contract.Symbol ==
self.oi_contract.Symbol:
                break

            # Update the selected contract and calculate MACD
            self.oi_contract = contract
            self.macd = self.MACD(contract.Symbol, 12, 26, 9,
MovingAverageType.Exponential, Resolution.Minute)

        # Check if MACD is available and ready
        if self.macd is None or not self.macd.IsReady:
```

```
            return

        # Get information about the selected contract
        symbol = self.oi_contract.Symbol
        security = self.Securities[symbol]
        price = security.Price

        # Only new positions for which the algorithm is not invested
        if security.Invested:
            # Look to exit the position
            return

        # Define tolerance and calculate the MACD signal delta percentage
        tolerance = 0.0
        signalDeltaPercent = self.macd.Current.Value -
self.macd.Signal.Current.Value

        # Trading logic based on MACD signal delta
        if signalDeltaPercent < -tolerance and self.Portfolio[symbol].Quantity <= 0:
            # Go long
            self.MarketOrder(symbol, 1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, -1, price*1.03)
            # Uncomment the following line for using a fixed stop loss order
            #self.stopLoss = self.StopMarketOrder(symbol, -1, price*0.99)
            # Uncomment the following line for using a trailing stop loss order
            self.stopLoss = self.TrailingStopOrder(symbol, -1, 0.01, True)
        if signalDeltaPercent > tolerance and self.Portfolio[symbol].Quantity >= 0:
            # Go short
            self.MarketOrder(symbol, -1)
            # Set take profit and stop loss orders
            self.takeProfit = self.LimitOrder(symbol, 1, price*0.97)
            # Uncomment the following line for using a fixed stop loss order
            #self.stopLoss = self.StopMarketOrder(symbol, 1, price*1.01)
            # Uncomment the following line for using a trailing stop loss order
            self.stopLoss = self.TrailingStopOrder(symbol, 1, 0.01, True)

    def OnOrderEvent(self, orderEvent):
        # Cancel the corresponding order if the other one is filled
        if orderEvent.Status != OrderStatus.Filled:
            return
        self.Cancel(orderEvent.OrderId)

    def Cancel(self, id):
        '''Cancel one order if the other was filled'''
        if self.takeProfit is not None and id == self.takeProfit.OrderId:
            self.stopLoss.Cancel()
        elif self.stopLoss is not None and id == self.stopLoss.OrderId:
            self.takeProfit.Cancel()
        else:
```

```
            return
        self.takeProfit = None
        self.stopLoss = None
```

Source Code: Augment Dicky Fuller Test for S&P500 stock:

```
# Import necessary libraries
import yfinance as yf
import pandas as pd
import numpy as np
from statsmodels.tsa.stattools import adfuller

# Function to conduct ADF test for stationarity
def test_stationarity(price_series):
    result = adfuller(price_series, regression='n', autolag='BIC')
    return result[0] if result else None

def get_historical_prices(symbol, start_date, end_date):
    stock_data = yf.download(symbol, start=start_date, end=end_date)
    return stock_data['Close']

# Get SPY constituents
spy_tickers =
pd.read_html('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')[0]['
Symbol'].tolist()

# Choose a subset of tickers (e.g., first 10) for demonstration purposes
selected_tickers = spy_tickers + ["SPY"]

# Define the date range for historical prices
start_date = '2023-01-01'
end_date = '2024-01-01'

# Create a DataFrame to store test statistics
data = {'Symbol': [], 'Test Statistics': []}

# Fetch historical prices and conduct ADF test for each stock
for symbol in selected_tickers:
    try:
        prices = get_historical_prices(symbol, start_date, end_date)
        test_statistic = test_stationarity(np.log(prices))

        data['Symbol'].append(symbol)
        data['Test Statistics'].append(test_statistic)
    except Exception as e:
        print(f"Error fetching data for {symbol}: {e}")

# Create a table from the DataFrame
```

```
test_statistics_table = pd.DataFrame(data)

# Display the table
test_statistics_table = test_statistics_table.sort_values(by='Test Statistics',
ascending=True)

# Display the sorted table
print(test_statistics_table.head(20))
```