Advanced Risk

Management

Assignment1

Name	Andrew ID	Email Address
Ningxin Zhang	coraz	coraz@andrew.cmu.edu
Shuming Xu	shumingx	shumingx@andrew.cmu.edu
Ping Wang	irenew	irenew@andrew.cmu.edu

1. Identifying Two Pairs of Cointegrated Instruments

Firstly, we need to find two pairs of market instruments with cointegration effect to implement the pair trading strategy, a pair of ETFs and a pair of stocks.

1.1 Choosing Cointegrated ETFs

To conduct pair trading, we first identify groups of ETFs representing different industries to ensure fundamental similarity. Within each group, we then perform cointegration tests using historical data from 01-01-2012 to 01-01-2022 to detect potential trading pairs.

Based on the FTS ETF lists provided in Appendices 3 and 4, we categorized and selected ETFs into five groups: “Technology & Communication,” “Consumer & Healthcare,” “Finance & Real Estate,” “Energy, Materials & Industrials,” and “Global & Macro Assets.” The selected ETFs in each category are summarized in the table 1.1.1.

For each group of ETFs, we tried every possible pair within group and plot the heat map (1.1.2) to identify possible pair. From the heatmaps, we can observe five pairs that are highlighted in red, indicating extremely low p-values and, therefore, a stronger likelihood of stable long-term relationships. Details are displayed in the table 1.1.3.

For the two cointegrated pairs within the Global & Macro Assets group — EWA & EWC and EWA & EWT — their cointegration effect primarily stems from strong export-import relationships. Australia maintains significant trade ties with both Canada and Taiwan, particularly in natural resources and intermediate goods. However, due to the impact of President Trump's import tariff policies and rising geopolitical tensions, international trade relationships have become increasingly unpredictable. As a result, cross-border transactions may weaken significantly, potentially diminishing the cointegration effect observed between these ETF pairs.

This observation is supported by the historical spread plot of the EWA & EWC pair. As shown in the graph 1.1.4, a significant deviation from the mean can be observed during the period from 2021 to 2022. This timeframe aligns with the COVID-19 pandemic, which was characterized by global trade disruptions and severe supply chain shocks. Therefore, we can conclude that shifts in macroeconomic conditions and interruptions in international trade can substantially weaken the cointegration relationship among pairs in the Global & Macro Assets group.

For the two pairs within the Consumer & Healthcare group, the main issues lie in the fact that they belong to different industries and exhibit a high level of randomness in the residual plots (Graph 1.1.5).

Therefore, we will choose XLF & IYR as our top pick among all the ETF pairs, with a hedge ratio of 0.45. The cointegration relationship between these two ETFs may come from their shared sensitivity to macroeconomic factors such as interest rates and economic cycles. Both sectors tend to react similarly to changes in consumer spending, inflation, and government policies, creating a long-term equilibrium relationship between the two. From the spread plot of XLF − 0.45 × IYR (Graph 1.1.6), we observe a clear mean-reverting pattern, particularly when compared to the 180-day moving average. This suggests that if we use the moving average to define trading thresholds, this pair would be a strong candidate for a pairs trading strategy.

1.2 Choosing Cointegrated Stocks

As for choosing cointegrated stock pairs, the steps are similar as the above-mentioned process for choosing ETFs. Firstly, we categorized and shortlisted the stocks into 6 groups for further test and analysis. Respectively, “financials”, “big tech”, “healthcare”, “energy stocks”, “consumer”, and “industrials”.

Then, we generated a heatmap for each group to visually assess the cointegration strength between pairs. Then we sorted the pairs according to the p-value, and choose the top five cointegrated pairs across all groups to facilitate further analysis and decision (Table 1.2.3).

For the first two pairs in the Big Tech group, the extremely small p-values indicate a strong cointegration relationship. However, upon examining the spread plot (1.2.4), it becomes clear that this strong cointegration is primarily driven by data prior to 2019. After 2019, the spread becomes significantly more volatile and appears to deviate from the mean, weakening the reliability of the relationship for pair trading. Therefore, to ensure that the trading strategy will work in the current financial market, we will not choose these two pairs. **Instead, we will choose the pair of “KO & PEP” as our market instrument to perform pair trading, with a hedge ratio of 0.27.**

From an economic perspective, the cointegration effect between PepsiCo (PEP) and Coca-Cola (KO) can be attributed to their similar business models, global market exposure, and sensitivity to common macroeconomic factors such as consumer spending, commodity prices. As both companies operate in the non-alcoholic beverage industry and target overlapping consumer bases, their revenues and stock performances tend to move together over time. Also, a closer look at the spread plot of PEP & KO (Graph 1.2.5) reveals a clear mean-reverting pattern. Although there is a notable deviation around 2020—likely caused by the unforeseen impact of COVID-19—this appears to be an isolated event. Overall, the long-term stability of the spread supports the conclusion that the cointegration relationship between the two stocks remains strong, making them suitable candidates for pair trading.

2. Defining Pairs Trading Strategy and Back-testing

2.1 Strategy Implementation

This is a **mean-reversion-based pairs trading strategy** between two pairs of cointegrated financial instruments. The strategy exploits the idea that the price spread between two historically cointegrated pairs will revert to a long-term mean over time.

2.1.1 Market Instrument Rules

Instruments Traded:

- Long/short positions in Coca-Cola (KO, asset1) and PepsiCo (PEP, asset2) equities.
- Long/short positions in SSgA Active Trust - Financial Select Sector SPDR (XLF, asset1) and BlackRock Institutional Trust Company N.A. - iShares U.S. Real Estate ETF (IYR, asset2).

Mechanism:

- Entering a long spread means buying asset1 and shorting corresponding units of asset2.
- Entering a short spread means selling asset1 and longing corresponding units of asset2.

2.1.2 Time Interval Rules: Price Monitoring Frequency set as Daily (1-day interval)

2.1.3 Data Rules for Strategy:

- **Spread Calculation (using data from historical period 2012-2021)**:

$$Spread_t = KO_t - 0.27 \times PEP_t$$

$$Spread_t = XLF_t - 0.45 \times IYR_t$$

- **Long-Term Mean**: Mean of the spread from the **10-year historical period (2012–2021)**.
- **Rolling Volatility**: 20-day rolling standard deviation for Z-score computation.
- **Trend Filter**: 5-day and 20-day moving averages of the spread used to detect trend direction.
- **Z-score**:

$$Z_t = \frac{Spread_t - LongTermMean}{RollingStd20(t)}$$

2.1.4 Entry Rules:

Long Spread Position (Buy asset1, Sell asset2): When Z-score < **−2** (spread significantly below mean).

Short Spread Position (Sell asset1, Buy asset2):When Z-score > **+2** (spread significantly above mean).

Note: If the strategy has been **paused after a forced exit**, no new entries are made until the spread reverts near the mean (|Z| < 1).

2.1.5 Exit Rules:

1. **Normal Exit**:

Long position: Exit when Z-score ≥ 0.25

Short position: Exit when Z-score ≤ -0.25

2. **Timeout Exit (Position Held > 150 Days)**:
   - If **no observable trend** (MA(5) ≈ MA(20)), exit using relaxed Z-score (±1).
   - If **trend strongly diverges** against position, **force exit (stop-loss)** and pause the strategy until mean reversion.

2.2 Backtesting Result

We use 2012-2021 period for modelling to get the fixed hedge ratio and fixed mean, and use 2022-2025 for backtesting. Details of trading during the backtesting period are attached in the appendix (Appendix 2.2.2 to 2.2.6), with description of last 5 trade in Table 2.2.2 and Table 2.2.5.

The backtesting results for the stock and ETF pair trading are as follows:

Metric	Stock Pairs	ETF Pairs
Final Wealth	104,738.87	102,966.95
Annual Return (%)	1.40%	1.71%
Sharpe Ratio (daily)	8.6775	4.2095
Max Drawdown (%)	1.65%	2.17%
Win Ratio	71.43% (5/7)	80.00% (4/5)
Profitability (Gain/Loss)	3.1639	3.2531
Total Trades	7	5

2.2.1 Strength Across Both Instruments

- **High Annual Returns**: Both results achieved high Sharpe ratios, indicating strong return consistency relative to volatility.
- **Strong Profitability**: Gain/loss ratios over **3x** show both strategies had a favorable trade-off between winning and losing trades.
- **High Win Rates**: 71% and 80% win ratios suggest robust signal quality.

2.2.2 Weaknesses

- **Low Trade Frequency**: Both results show low trade counts over long horizons, which may hinder scalability or make capital deployment inefficient.
- **Modest Absolute Return (Stock Pairs)**: The current trading unit might be too conservative.

3. Live Trading Screenshot

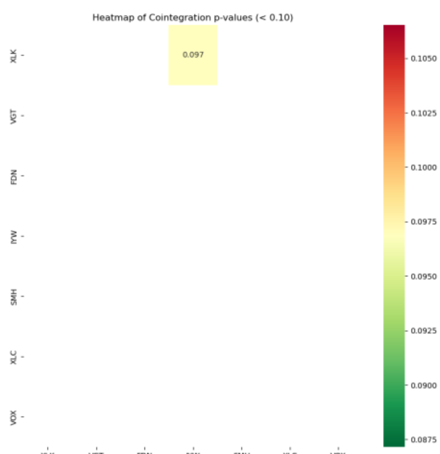
Security	Position	Last Value	Position Value	Overall Gain Loss
US Dollar	991,792	991,791.80	991,791.80	0.00
The Coca-Cola Company (KO)	100	7,191.50	7,191.50	-1.50
PepsiCo, Inc. (PEP)	-27	-3,563.73	1,781.46	-0.27

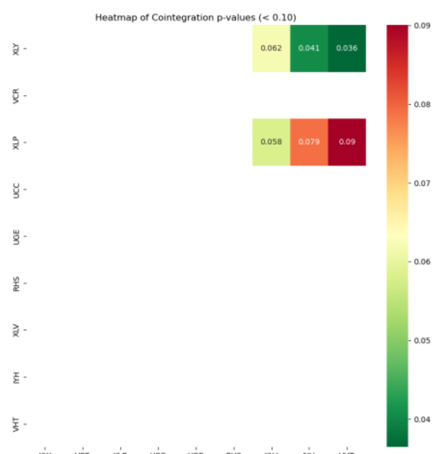
Security	Position	Last Value	Position Value	Overall Gain Loss
US Dollar	993,676	993,675.70	993,675.70	0.00
iShares U.S. Real Estate ETF (IYR)	-45	-4,210.20	2,105.78	0.45
Financial Select Sector SPDR Fund (XLF)	100	4,978.50	4,978.50	3.50

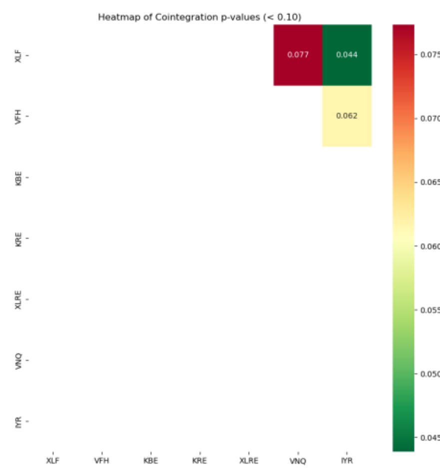
Appendix:

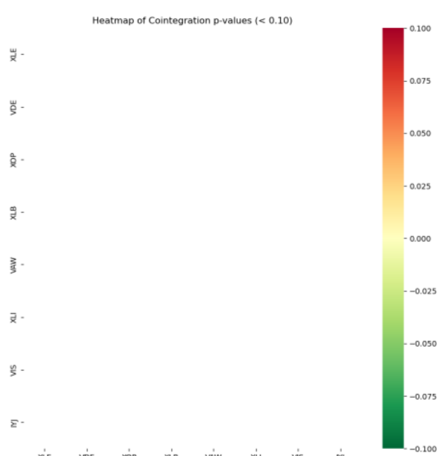
Group	ETF List
Technology & Communication	'XLK', 'VGT', 'FDN', 'IYW', 'SMH', 'XLC', 'VOX'
Consumer & Healthcare	'XLY', 'VCR', 'RCD', 'XLP', 'UCC', 'RHS', 'XLV', 'IYH', 'VHT'
Finance & Real Estate	'XLF', 'VFH', 'KBE', 'KRE', 'XLRE', 'VNQ', 'IYR'
Energy, Materials & Industrials	'XLE', 'VDE', 'XOP', 'XLB', 'VAW', 'XLI', 'VIS', 'IYJ'
Global & Macro Assets	'UUP', 'FXE', 'FXY', 'FXB', 'GLD', 'SLV', 'DBC', 'DBA', 'USO', 'EWA', 'EWC', 'EWG', 'EWJ', 'EWQ', 'EWU', 'EWH', 'EWT', 'EWZ'

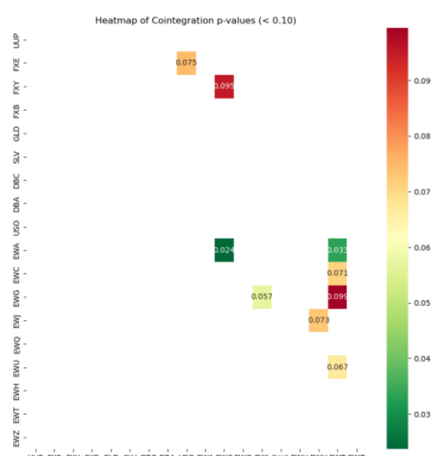
1.1.1 ETF lists for each group









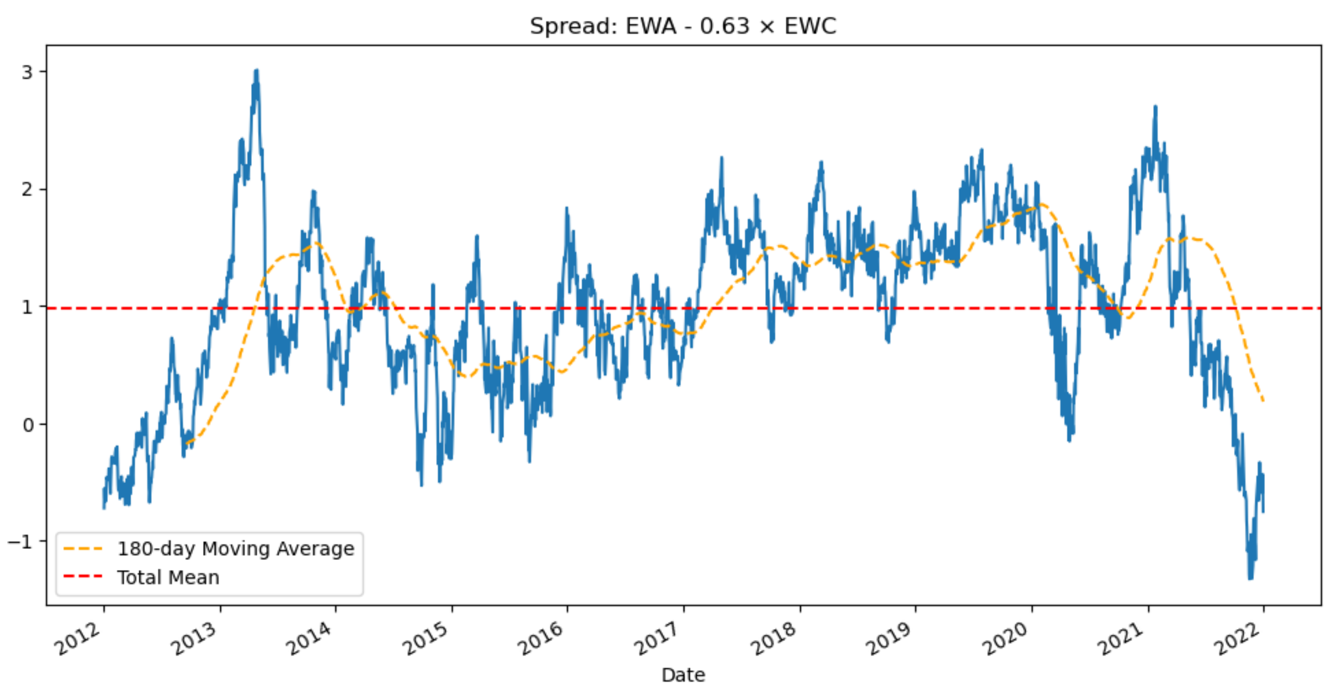


1.1.2 Heat maps for the five ETF groups.

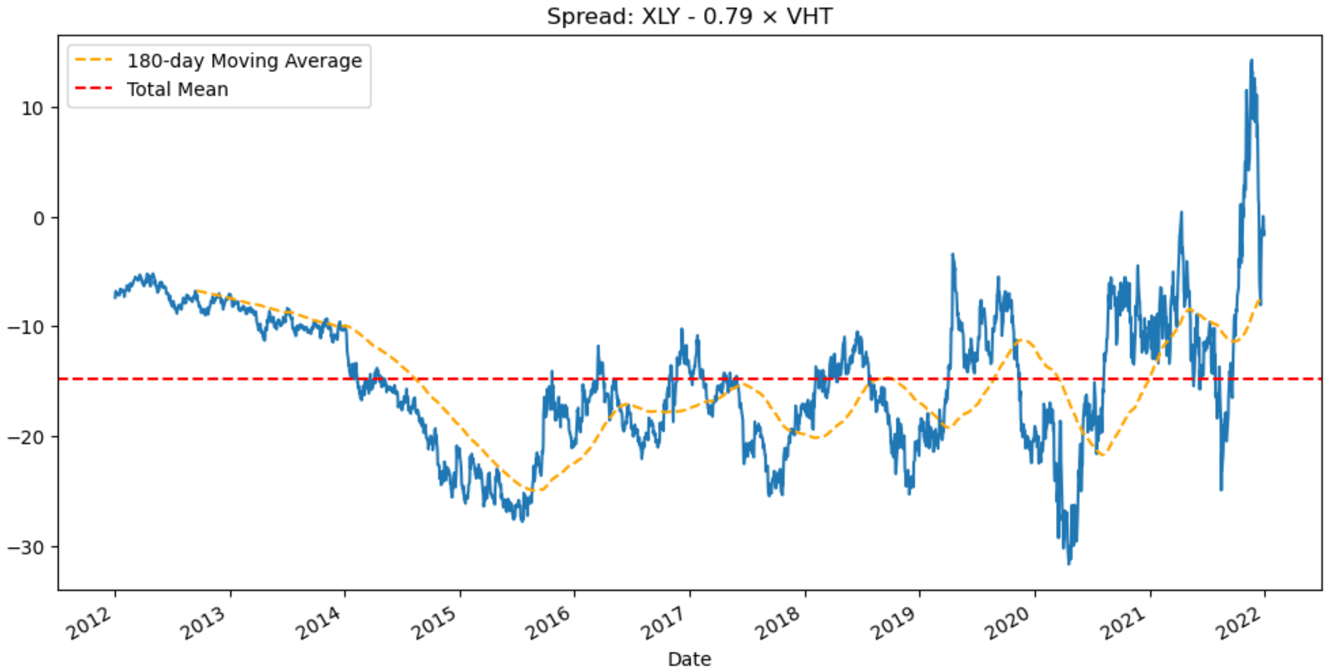
Layout follows the group order in the table, read horizontally (left to right) and vertically (top to bottom).

Pair	Group	p-value	Hedge Ratio
EWA & EWC	Global & Macro Assets	0.0237	0.6318
EWA & EWT	Global & Macro Assets	0.0334	0.2733
XLY & VHT	Consumer & Healthcare	0.0364	0.7864
XLY & IYH	Consumer & Healthcare	0.0406	3.5458
XLF & IYR	Finance & Real Estate	0.0439	0.4467

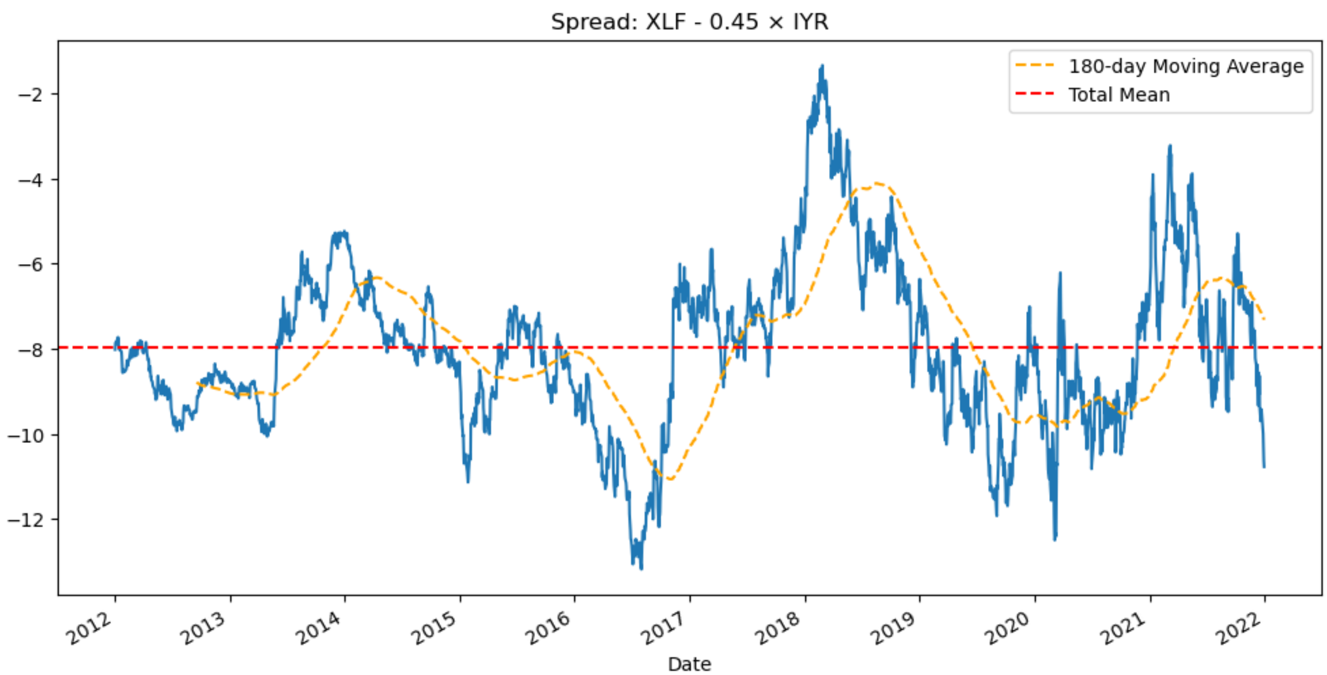
1.1.3 Top 5 Cointegrated Pairs Across All Groups



1.1.4 Spread Plot for EWA & EWC



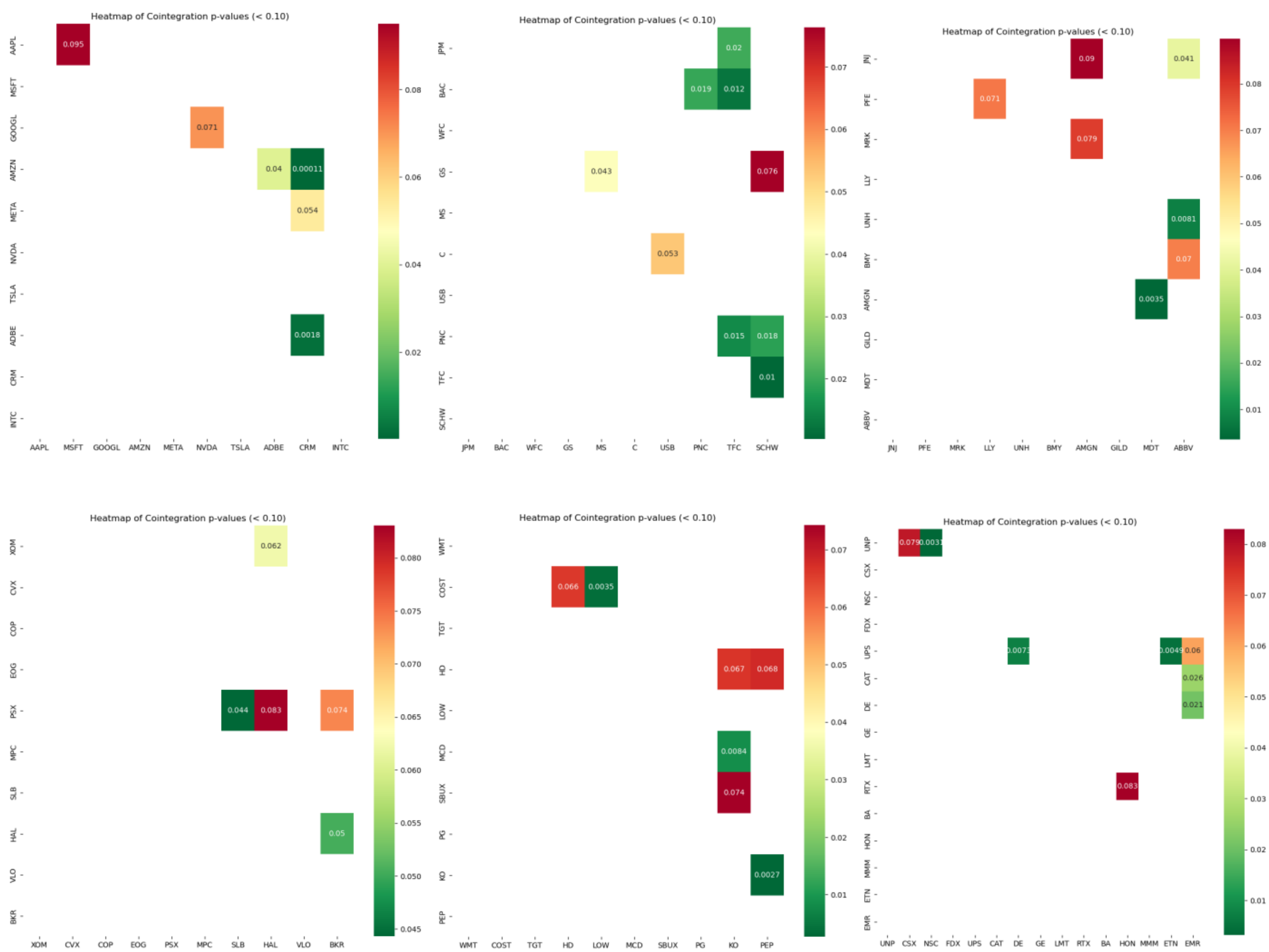
1.1.5 Spread Plot for XLY & VHT



1.1.6 Spread Plot for XLF & IYR

Group	Stock List
Financials	'JPM', 'BAC', 'WFC', 'GS', 'MS', 'C', 'USB', 'PNC', 'TFC', 'SCHW'
Big Tech	'AAPL', 'MSFT', 'GOOGL', 'AMZN', 'META', 'NVDA', 'TSLA', 'ADBE', 'CRM', 'INTC'
Healthcare	'JNJ', 'PFE', 'MRK', 'LLY', 'UNH', 'BMY', 'AMGN', 'GILD', 'MDT', 'ABBV'
Energy	'XOM', 'CVX', 'COP', 'EOG', 'PSX', 'MPC', 'SLB', 'HAL', 'VLO', 'BKR'
Consumer	'WMT', 'COST', 'TGT', 'HD', 'LOW', 'MCD', 'SBUX', 'PG', 'KO', 'PEP'
Industrials	‘UNP’, ‘CSX’, ‘NSC’, ‘FDX’, ‘UPS’, ‘CAT’, ‘DE’, ‘GE’, ‘LMT’, ‘RTX’, ‘BA’, ‘HON’, ‘MMM’, ‘ETN’, ‘EMR’

1.2.1 Stock lists for each group

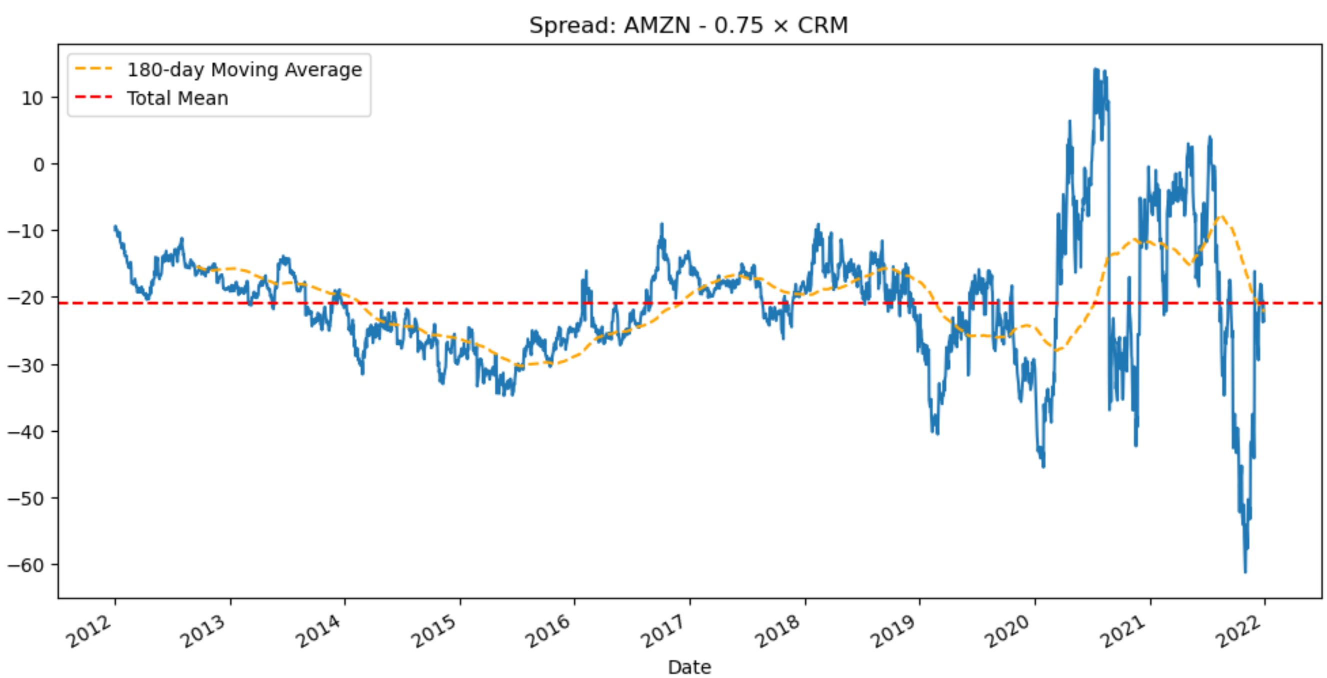


1.2.2 Heat maps for the six stock groups.

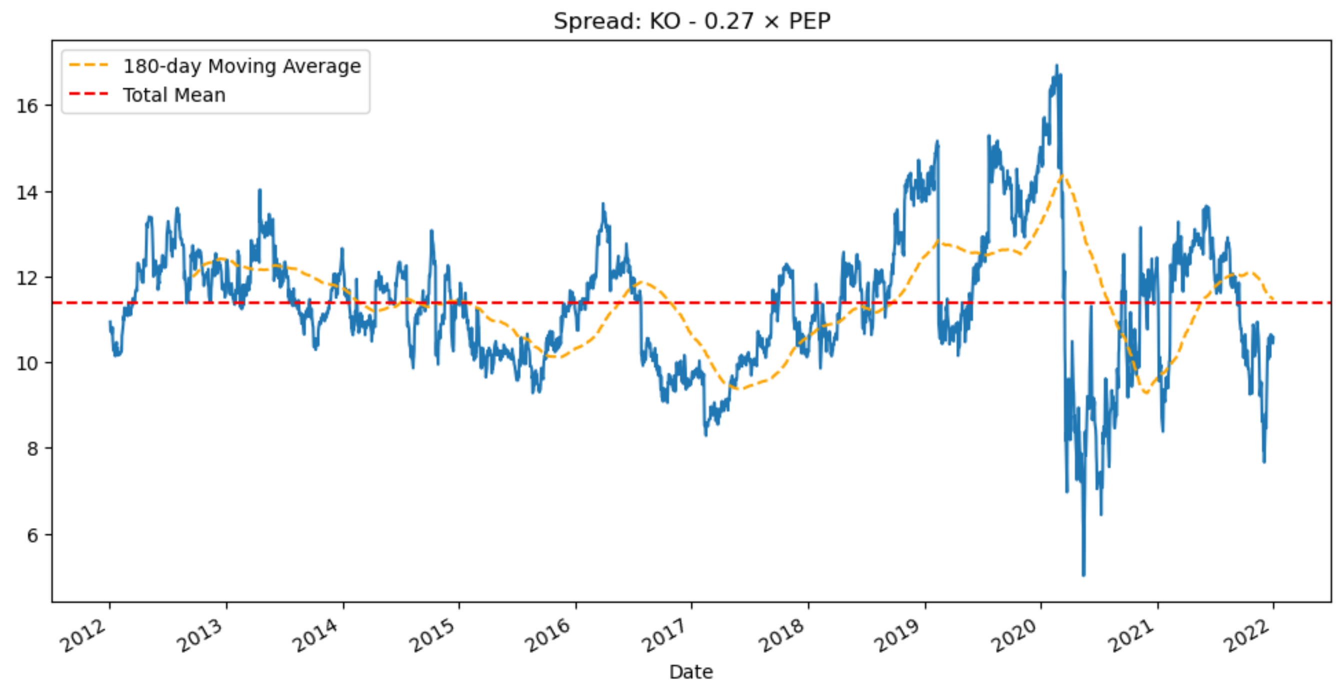
Layout follows the group order in the table, read horizontally (left to right) and vertically (top to bottom).

Pair	Group	p-value	Hedge Ratio
AMZN & CRM	Tech Mega Caps	0.0001	0.7542
ADBE & CRM	Tech Mega Caps	0.0018	2.4604
KO & PEP	Consumer Sector	0.0027	0.2731
UNP & NSC	Industrials & Transportation	0.0031	0.8031
AMGN & MDT	Healthcare	0.0035	1.9913

1.2.3 Top 5 Cointegrated Pairs Across All Groups



1.2.4 Spread Plot for AMZN & CRM



1.2.5 Spread Plot for KO & PEP

```
def mean_reverting_spread_strategy(asset1: str,
                                   asset2: str,
                                   spread: pd.Series, long_term_mean: float,
                                   std_window: int = 20,
                                   entry_threshold: float = 2,
                                   exit_threshold: float = -0.25,
                                   relaxed_exit: float = 1,
                                   ma_short_window: int = 5,
                                   ma_long_window: int = 20,
                                   min_start_index: int = 20,
                                   tolerance: float = 1):
    """
    """
    Mean-reversion pairs trading strategy using a fixed mean and rolling volatility.

    Strategy Logic:
    - Enter long if z-score < -entry_threshold
    - Enter short if z-score > +entry_threshold
    - Exit if z-score crosses back within +/- exit_threshold
    - If holding a position over 180 days:
        - If spread is trending away (against position), force exit (stop-loss) and pause new trades
        - If trend is flat (no strong signal), allow exit with a relaxed threshold
    - After a forced exit, wait for spread to return near long-term mean before resuming trading

    Parameters:
    - asset1 / asset2: asset names
    - spread: Series of spread values (e.g., KO - 0.27 * PEP)
    - long_term_mean: Precomputed fixed mean of the spread
    - std_window: Rolling window size for volatility (standard deviation)
    - entry_threshold: Z-score to enter a trade
    - exit_threshold: Z-score to exit a trade
    - relaxed_exit: relaxed exit threshold, a Z-score to exit a trade
    - ma_short_window: Rolling window for short-term mean (for trend detection)
    - ma_long_window: Rolling window for long-term mean (for trend detection)
    - min_start_index: Index to start backtesting after warm-up period
    - tolerance: Absolute z-score distance to resume trading after pause
```

```python
    Returns:
    - trades: DataFrame of trade history
    """

    # Calculate z-score and moving averages
    rolling_std = spread.rolling(window=std_window).std()
    z_score = (spread - long_term_mean) / rolling_std
    spread_short_mean = spread.rolling(window=ma_short_window).mean()
    spread_long_mean = spread.rolling(window=ma_long_window).mean()

    # Plotting thresholds
    up_threshold = long_term_mean + entry_threshold * rolling_std
    down_threshold = long_term_mean - entry_threshold * rolling_std

    position = 0              # 0 = no position, 1 = long spread, -1 = short spread
    trade_log = []            # Store trade history
    signals = []              # Store visual signals
    paused = False            # If True, skip trade signals until spread mean-reverts

    for i in range(min_start_index, len(spread)):
        date = spread.index[i]
        if rolling_std[i] == 0 or np.isnan(rolling_std[i]):
            continue

        # Resume trading only if spread has reverted back near long-term mean
        if paused and abs(z_score[i]) < tolerance:
            paused = False

        # Entry logic (only if not paused)
        if position == 0 and not paused:
            if z_score[i] > entry_threshold:
                # Short the spread (KO expensive, PEP cheap)
                position = -1
                entry_price = spread[i]
                entry_date = date
                direction = f"Short {asset1} / Long {asset2}"
                signals.append((date, spread[i], 'sell'))
            elif z_score[i] < -entry_threshold:
                # Long the spread (KO cheap, PEP expensive)
                position = 1
                entry_price = spread[i]
                entry_date = date
                direction = f"Long {asset1} / Short {asset2}"
                signals.append((date, spread[i], 'buy'))

        elif position == 1:
            holding_days = (date - entry_date).days
            trend = spread_short_mean[i] - spread_long_mean[i]

            # Normal exit if z-score reverts
            if z_score[i] >= -exit_threshold:
                pnl = spread[i] - entry_price
                trade_log.append({
                    "Entry Date": entry_date,
                    "Exit Date": date,
                    "Direction": direction,
                    "Entry Spread": entry_price,
                    "Exit Spread": spread[i],
                    "PnL": pnl
                })
                position = 0

            # Stop-loss or relaxed exit after 180 days of holding
            elif holding_days > 150:
                if trend > 0:  # Trend is continuing in wrong direction → force stop
                    pnl = spread[i] - entry_price
                    trade_log.append({
                        "Entry Date": entry_date,
                        "Exit Date": date,
                        "Direction": direction,
                        "Entry Spread": entry_price,
                        "Exit Spread": spread[i],
```

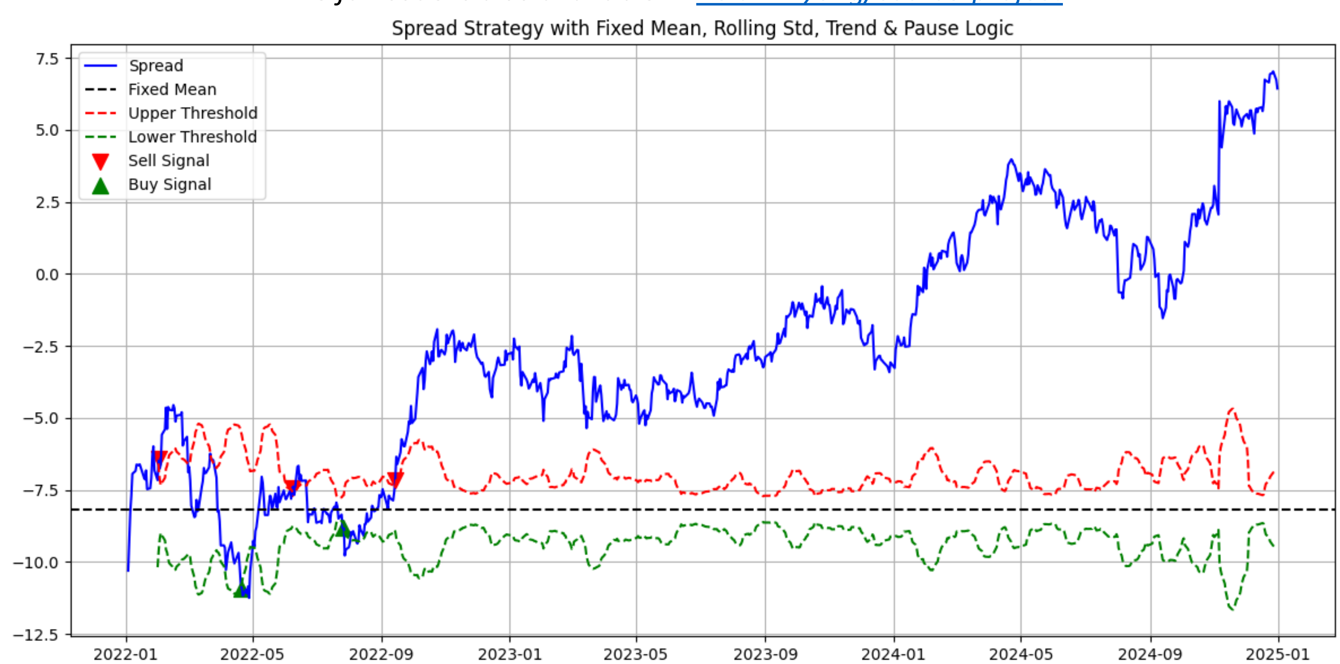
```python
                    "PnL": pnl
                })
                position = 0
                paused = True
            elif abs(trend) < 0.01:  # Flat trend, allow relaxed exit
                if z_score[i] >= -relaxed_exit:
                    pnl = spread[i] - entry_price
                    trade_log.append({
                        "Entry Date": entry_date,
                        "Exit Date": date,
                        "Direction": direction,
                        "Entry Spread": entry_price,
                        "Exit Spread": spread[i],
                        "PnL": pnl
                    })
                    position = 0

    elif position == -1:
        holding_days = (date - entry_date).days
        trend = spread_short_mean[i] - spread_long_mean[i]

        if z_score[i] <= exit_threshold:
            pnl = entry_price - spread[i]
            trade_log.append({
                "Entry Date": entry_date,
                "Exit Date": date,
                "Direction": direction,
                "Entry Spread": entry_price,
                "Exit Spread": spread[i],
                "PnL": pnl
            })
            position = 0

        elif holding_days > 150:
            if trend < 0:  # Wrong trend → stop-loss
                pnl = entry_price - spread[i]
                trade_log.append({
                    "Entry Date": entry_date,
                    "Exit Date": date,
                    "Direction": direction,
                    "Entry Spread": entry_price,
                    "Exit Spread": spread[i],
                    "PnL": pnl})
                position = 0
                paused = True
            elif abs(trend) < 0.01:  # Flat trend → allow relaxed exit
                if z_score[i] <= relaxed_exit:
                    pnl = entry_price - spread[i]
                    trade_log.append({
                        "Entry Date": entry_date,
                        "Exit Date": date,
                        "Direction": direction,
                        "Entry Spread": entry_price,
                        "Exit Spread": spread[i],
                        "PnL": pnl
                    })
                    position = 0

# Compute performance metrics
trades = pd.DataFrame(trade_log)

# Plot spread and signals
plt.figure(figsize=(12, 6))
plt.plot(spread, label="Spread", color='blue')
plt.axhline(long_term_mean, color="black", linestyle="--", label="Fixed Mean")
plt.plot(up_threshold, color="red", linestyle="--", label="Upper Threshold")
plt.plot(down_threshold, color="green", linestyle="--", label="Lower Threshold")

for date, val, sig in signals:
    if sig == 'buy':
        plt.scatter(date, val, color='green', marker='^', s=100,
                    label='Buy Signal' if 'Buy Signal' not in plt.gca().get_legend_handles_labels()[1] else "")
    elif sig == 'sell':
```

```python
            plt.scatter(date, val, color='red', marker='v', s=100,
                        label='Sell Signal' if 'Sell Signal' not in plt.gca().get_legend_handles_labels()[1] else "")

    plt.title("Spread Strategy with Fixed Mean, Rolling Std, Trend & Pause Logic")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()

    return trades

def run_backtest(
    trades: pd.DataFrame,
    data1: pd.Series,
    data2: pd.Series,
    hedge_ratio: float,
    initial_capital: float = 100_000,
    default_trade_unit: float = 10_000
):
    """
    Simulate backtest with actual trade execution based on data1 (e.g., KO) and data2 (e.g., PEP) prices.

    Parameters:
    - trades: DataFrame of trade history
    - data1: price series of asset 1 (e.g., KO)
    - data2: price series of asset 2 (e.g., PEP)
    - hedge_ratio: weight applied to asset 2 (e.g., 0.27)
    - initial_capital: starting capital in USD
    - default_trade_unit: max dollar exposure per leg

    Returns:
    - metrics: performance dictionary
    - wealth_series: pd.Series of portfolio value over time
    """

    wealth = initial_capital
    wealth_history = [wealth]
    wealth_dates = [data1.index[0]]

    for _, trade in trades.iterrows():
        entry_date = pd.to_datetime(trade["Entry Date"])
        exit_date = pd.to_datetime(trade["Exit Date"])
        direction = trade["Direction"]

        # Get prices at entry and exit
        entry_price1 = data1.loc[entry_date]
        entry_price2 = data2.loc[entry_date]
        exit_price1 = data1.loc[exit_date]
        exit_price2 = data2.loc[exit_date]

        # Capital-based trade sizing
        max_units1 = wealth // entry_price1
        max_units2 = wealth // entry_price2
        trade_unit = min(default_trade_unit, max_units1, max_units2)

        # PnL based on trade direction
        if "Long" in direction:
            pnl = ((exit_price1 - entry_price1) + (entry_price2 - exit_price2) * hedge_ratio) * trade_unit
        else:
            pnl = ((entry_price1 - exit_price1) + (exit_price2 - entry_price2) * hedge_ratio) * trade_unit

        wealth += pnl
        wealth_history.append(wealth)
        wealth_dates.append(exit_date)

    wealth_series = pd.Series(wealth_history, index=pd.to_datetime(wealth_dates)).sort_index()

    # Metrics
    final_wealth = wealth_series.iloc[-1]
    days = (wealth_series.index[-1] - wealth_series.index[0]).days
    annual_return = (final_wealth / initial_capital) ** (252 / days) - 1 if days > 0 else np.nan
    daily_returns = wealth_series.pct_change().dropna()
```

```python
    sharpe_ratio = daily_returns.mean() / daily_returns.std() * np.sqrt(252) if len(daily_returns) > 1 else np.nan
    peak = wealth_series.cummax()
    max_drawdown = ((peak - wealth_series) / peak).max()

    win_ratio = (trades["PnL"] > 0).mean() if not trades.empty else np.nan
    total_gain = trades.loc[trades["PnL"] > 0, "PnL"].sum()
    total_loss = -trades.loc[trades["PnL"] < 0, "PnL"].sum()
    profitability = total_gain / total_loss if total_loss > 0 else np.nan

    # Plot
    plt.figure(figsize=(12, 5))
    plt.plot(wealth_series, label='Cumulative Wealth', color='darkblue')
    plt.title(f"Cumulative Wealth | Final: ${final_wealth:,.2f}")
    plt.xlabel("Date")
    plt.ylabel("Wealth ($)")
    plt.grid(True)
    plt.legend()
    plt.tight_layout()
    plt.show()

    metrics = {
        "Final Wealth": final_wealth,
        "Annual Return (%)": annual_return,
        "Sharpe Ratio (daily)": sharpe_ratio,
        "Max Drawdown (%)": max_drawdown,
        "Win Ratio": win_ratio,
        "Profitability (Gain/Loss Ratio)": profitability,
        "Total Trades": len(trades)
    }

    print("\n📊 Performance Summary:")
    for k, v in metrics.items():
        if isinstance(v, float):
            print(f"{k}: {v:.2%}" if "%" in k else f"{k}: {v:,.4f}")
        else:
            print(f"{k}: {v}")

    return metrics, wealth_series
```

2.1 strategy code

The full code is also available in [irelamwyong/course-project]()

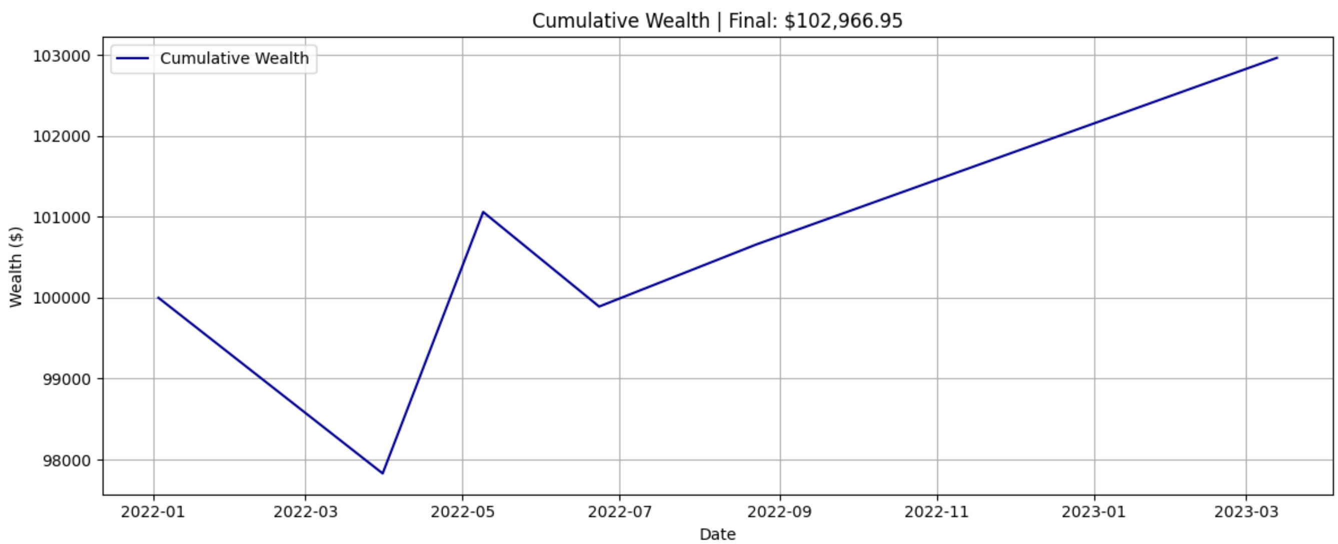


2.2.1 Plot of trading spread, threshold, and trading signal in ETF pair trading

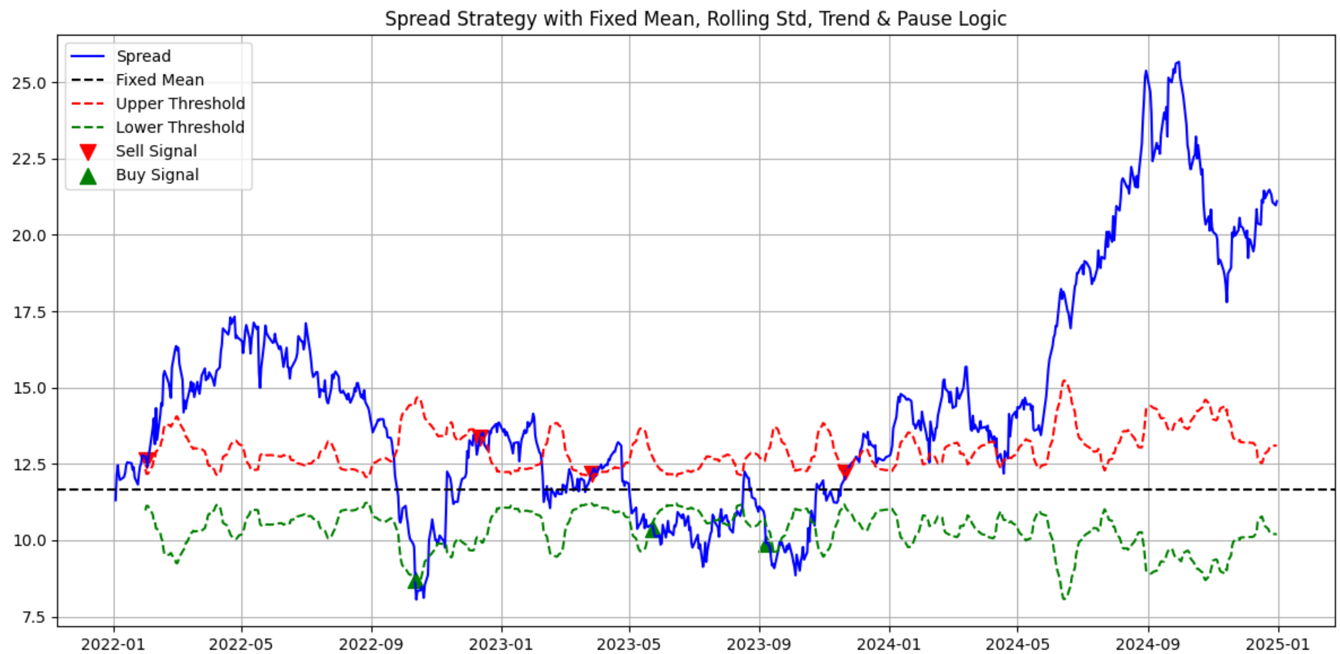
	Entry Date	Exit Date	Direction	Entry Spread	Exit Spread	PnL
0	2022-02-01	2022-03-31	Short XLF / Long IYR	-6.412049	-8.584177	2.172128
1	2022-04-21	2022-05-09	Long XLF / Short IYR	-10.924137	-7.527618	3.396519
2	2022-06-08	2022-06-23	Short XLF / Long IYR	-7.452415	-8.622137	1.169722
3	2022-07-26	2022-08-23	Long XLF / Short IYR	-8.818950	-8.051045	0.767905
4	2022-09-14	2023-03-13	Short XLF / Long IYR	-7.165044	-4.857636	-2.307409

2.2.2 last 5 trading records in ETF pair trading

Analysis: In the first four trades, the spread behaved in line with the strategy's expectations of mean-reverting, yielding positive returns. In the final trade, the spread moved adversely relative to the mean, triggering a forced exit and resulting in a realized loss.



2.2.3 cumulative wealth in ETF pair trading



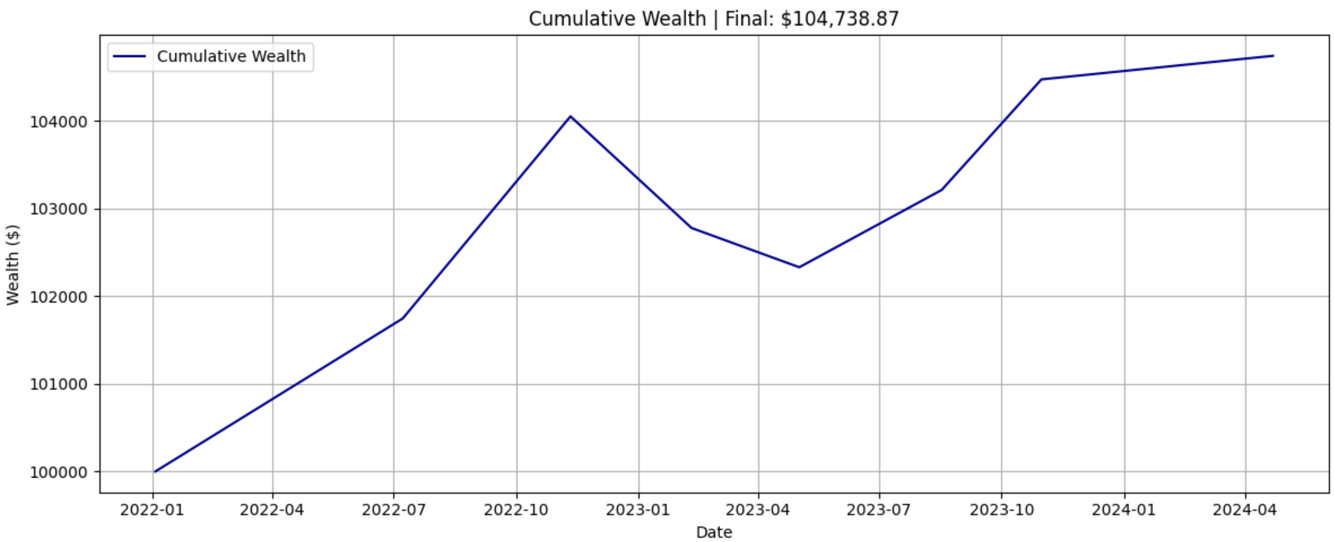
2.2.4 Plot of trading spread, threshold, and trading signal in stock pair trading

	Entry Date	Exit Date	Direction	Entry Spread	Exit Spread	PnL
2	2022-12-12	2023-02-10	Short KO / Long PEP	13.357421	11.254162	2.103259
3	2023-03-27	2023-05-02	Short KO / Long PEP	12.173902	11.438421	0.735482
4	2023-05-24	2023-08-17	Long KO / Short PEP	10.361161	11.859510	1.498349
5	2023-09-07	2023-10-31	Long KO / Short PEP	9.904561	11.961595	2.057034
6	2023-11-21	2024-04-22	Short KO / Long PEP	12.269983	12.681175	-0.411192

2.2.5 last 5 trading records in stock pair trading

Analysis:

For the first four trades, the spread evolved according to the strategy’s mean-reverting expectations, resulting in positive PnL. In the fifth trade, although the spread moved contrary to the closing threshold, it remained relatively close to the long-term mean. As a result, the exit condition was relaxed, leading to a controlled and limited loss.



2.2.6 last 5 trading records in stock pair trading