CTA Strategy Development for Futures Calendar Spread Arbitrage

I : Understand Future's "Calendar Spread Arbitrage"

1.1 What is a Futures Contract?

A **futures contract** is a standardized, legally binding agreement to buy or sell a specific quantity and quality of an underlying asset at a predetermined price, on a specified date in the future. These contracts are traded on regulated futures exchanges (e.g., CME Group in the US, Shanghai Futures Exchange in China).

Key components of a futures contract include:

- **Underlying Asset:** The commodity (e.g., Crude Oil, Corn, Gold) or financial instrument (e.g., S&P 500 index, US Treasury Bonds) that the contract is based on.
- **Contract Size:** The specific quantity of the underlying asset covered by one contract (e.g., 1,000 barrels of crude oil, 5,000 bushels of corn).
- **Expiration Date (or Delivery Month):** The month in which the contract expires and delivery of the asset is expected, or the contract is cash-settled. Contracts for the same asset are distinguished by their expiration month (e.g., December 2025 Crude Oil vs. March 2026 Crude Oil).
- **Price:** The price agreed upon by the buyer (who goes "long") and the seller (who goes "short").

Unlike stocks, futures contracts have a limited lifespan. This finite life is the fundamental reason calendar spreads are possible.

1.2 Definition of Futures Calendar Spread Arbitrage

A **calendar spread** (also known as a term-structure spread or an intra-commodity spread) involves the simultaneous purchase of one futures contract and the sale of another futures contract on the *same underlying asset* but with *different expiration dates*.

- **Buying a Spread:** You buy the near-month contract and sell the far-month contract. You profit if the spread *widens* (i.e., the near-month price increases relative to the far-month price).
- **Selling a Spread:** You sell the near-month contract and buy the far-month contract. You profit if the spread *narrows* (i.e., the near-month price decreases relative to the far-month price).

The "arbitrage" in the name is somewhat of a misnomer in modern markets. True risk-free arbitrage is rare. Instead, this is more accurately described as a **statistical arbitrage** or

relative value strategy. It doesn't bet on the outright direction of the asset's price, but rather on the price relationship *between* the two contracts.

1.3 How This Strategy Makes Money: The Market Rationale

The strategy's profitability hinges on the behavior of the *spread* (the price difference between the two contracts). This behavior is driven by the concepts of **Contango** and **Backwardation**.

Let the spread be defined as:

$$Spread = P_{far-month} - P_{near-month}$$

- **Contango:** This is the market condition where the price of a far-month futures contract is *higher* than the price of a near-month contract (Spread>0). This is considered a "normal" market for non-perishable commodities. The price difference reflects the **cost of carry**, which includes:
    - Storage costs (e.g., warehousing for oil or grain)
    - Insurance costs
    - Financing costs (interest foregone on the money used to hold the commodity)
- A strategy in a contango market might involve selling the spread (sell the expensive far-month, buy the cheaper near-month) if the trader believes the cost of carry is exaggerated and the spread will narrow.
- **Backwardation:** This is the market condition where the price of a far-month futures contract is *lower* than the price of a near-month contract (Spread<0). This often occurs due to near-term supply shortages, geopolitical events, or high immediate demand. In this scenario, the market is willing to pay a premium for immediate access to the asset. This premium is known as the **convenience yield**.
  A strategy in a backward market might involve buying the spread (buy the strong near-month, sell the weaker far-month) if the trader believes the near-term supply crunch will worsen, causing the spread to widen further.

The core of the strategy is mean reversion. The spread between two contract months on the same asset tends to fluctuate around a historical mean. A calendar spread strategy identifies when the spread has deviated significantly from this mean and places a trade betting that it will revert back. For example, if the normal spread between December and March Corn is 10 cents, and it suddenly widens to 25 cents due to a temporary market event, a trader might sell the spread, betting it will return to the 10-cent level.

Because the two legs of the spread are on the same underlying asset, they are highly correlated. If the price of crude oil skyrockets, both the near-month and far-month contracts will rise, largely offsetting each other. The profit or loss comes from the small, more predictable changes in the spread itself. This makes the strategy inherently less volatile than an outright directional bet on the asset.

II: Understand Backtesting

2.1 What is Backtesting?

Backtesting is the process of using historical data to simulate a trading strategy's performance as if it had been employed in the past. The primary goal is to assess the viability of a strategy before risking real capital. A robust backtest provides statistical evidence of how a strategy might perform in the future, including estimates of:

- **Profitability:** Total return, annualized return.
- **Risk:** Volatility (standard deviation of returns), Sharpe Ratio (risk-adjusted return).
- **Drawdown:** The maximum peak-to-trough decline in portfolio value, which measures downside risk.

A backtest is an essential tool for any quantitative trader to filter out bad ideas, refine good ones, and understand a strategy's risk profile.

2.2 Vectorized vs. Event-Driven Backtesting

There are two primary methods for conducting a backtest in a computational environment: vectorized and event-driven.

A. Vectorized Backtesting

A vectorized backtester performs calculations on entire arrays (or "vectors") of data at once, typically using libraries like **pandas** and **numpy**. It avoids sequential ***for*** loops that iterate through each time step.

- **How it works:**
    - Prepare all data into arrays (e.g., open, high, low, close prices for the entire period).
    - Generate trading signals for the entire period in a single vectorized operation (e.g., *signals = prices > moving_average*).
    - Calculate returns based on these signals, again using array-wise operations.
- **Pros:**
    - **Extremely Fast:** Leverages optimized C or Fortran code underlying *numpy/pandas* for massive speed gains.
    - **Simple Code:** Can be written in just a few lines of code for simple strategies.
- **Cons:**
    - **Look-ahead Bias:** It is very easy to accidentally use information from the future in a calculation. For example, calculating a signal based on the day's closing price and then assuming you could trade at that same day's opening price.
    - **Inflexibility:** Poorly suited for strategies with complex logic, such as path-dependent stops, portfolio-level risk management (e.g., sizing positions based on current equity), or handling multiple interacting assets.

- **Unrealistic:** Does not accurately model transaction costs, slippage, or the step-by-step process of order management.

B. Event-Driven Backtesting

An event-driven backtester simulates the flow of time by processing data one time-step at a time, in a sequential loop. It mimics how a real trading system operates.

- **How it works:** The system is built around an **event loop** that processes different types of events sequentially:
  - A **Market Event** occurs (e.g., a new price bar for 10:30 AM arrives).
  - The ***Strategy*** object receives the ***Market Event*** and decides whether to generate a **Signal Event** (e.g., "GO LONG SPREAD").
  - The ***Portfolio*** object receives the ***Signal Event***, determines the order size, and creates an **Order Event** (e.g., "BUY 10 CLZ25, SELL 10 CLH26").
  - The ***Execution Handler*** receives the ***Order Event***, simulates the trade execution (including costs/slippage), and generates a **Fill Event** (confirming the trade was executed at a certain price).
  - The ***Portfolio*** receives the ***Fill Event*** and updates its positions and cash.
  - The loop repeats for the next time-step.
- **Pros:**
  - **Realistic:** Closely simulates a live trading environment, allowing for accurate modeling of transaction costs, slippage, and latency.
  - **Flexible:** Can handle virtually any level of strategy complexity, including portfolio-level risk rules and interactions between many assets.
  - **Avoids Look-ahead Bias:** By processing data sequentially, it is structurally impossible to use future information.
- **Cons:**
  - **Slower:** The ***for*** loop structure and object-oriented overhead make it significantly slower than a vectorized approach.
  - **More Complex to Build:** Requires a more sophisticated software architecture with multiple interacting components.

2.3 Recommendation for Calendar Spread Arbitrage Strategies

For backtesting futures calendar spread arbitrage, I will select to use an **Event-Driven Backtesting** framework.

Justification:

1. **Multiple Assets:** A calendar spread, by definition, involves two separate instruments (the near-month and far-month contracts). An event-driven system is naturally designed to handle data streams from multiple assets simultaneously.
2. **Portfolio-Level Logic:** The decision to enter or exit a spread trade depends on the *spread's* value, not the individual leg's price. Furthermore, risk management (e.g.,

margin calculations, overall portfolio exposure) requires a holistic view that an event-driven ***Portfolio*** object provides.

3. **Realistic Costs:** Profit margins on spread strategies can be thin. Therefore, accurately modeling transaction costs (commissions) and slippage for *both legs* of the trade is critical for a realistic performance assessment. This is trivial in an event-driven system but cumbersome and inaccurate in a vectorized one.
4. **Contract Rollover Logic:** Futures contracts expire. A continuous backtest over several years requires logic to "roll" the spread from an expiring pair of contracts to the next pair. This is a complex, path-dependent event that is perfectly suited for an event-driven engine but nearly impossible to implement correctly in a vectorized backtester.

While an event-driven engine is more complex to build initially, its realism and flexibility are non-negotiable for developing a robust and trustworthy calendar spread strategy. For our project, we will build a simplified but functional event-driven backtester in Python.

III: Collect Futures Data

Acquiring high-quality historical data is arguably the most critical step in quantitative research. For the future, this process has unique challenges that we must address before writing any code.

3.1 The Challenge of Futures Data: The Need for Continuous Contracts

Unlike stocks, which have a perpetual life, individual futures contracts expire. For example, the December 2025 Crude Oil contract (CLZ25) will only trade for a specific period (typically one to two years) before it ceases to exist.

This presents a problem for backtesting a strategy over a long historical period (e.g., 10-15 years). We cannot simply use a single contract. To solve this, data providers create a **continuous futures contract**. This is a synthetic, long-term price series constructed by stitching together individual monthly contracts.

However, simply pasting the price series of one contract after another creates large, artificial price gaps at each "roll" date, rendering the data useless for calculating returns. To solve this, price adjustments are made. The most common method, and the one most suitable for backtesting, is **back-adjustment**:

- **Back-Adjustment:** When it's time to roll from an expiring contract (e.g., December) to the next contract (e.g., March), a price difference (or "gap") between the two is calculated. The entire historical price series of all previous contracts is then adjusted up or down by this difference. This process is repeated at every roll. The result is a smooth price series that accurately reflects historical percentage returns, which is essential for strategy evaluation.

Important Note for Calendar Spreads: While a continuous front-month contract is useful for analyzing the general trend of a commodity, our specific strategy requires the actual, unadjusted historical prices of *two different contracts* (e.g., the December contract and the March contract) trading simultaneously. Therefore, our data acquisition process will involve downloading the historical data for specific contract pairs for each year (e.g., CLZ23 vs CLH24, then CLZ24 vs CLH25, etc.) and managing the "roll" of the entire spread position in our backtesting code.

3.2 Where to Get Futures Data Using Python

We need data sources that provide historical daily (or intraday) price and volume data for individual futures contracts. Below are recommended sources and the Python libraries used to access them.

A. Data Sources for US/Global Markets

1. **Nasdaq Data Link (formerly Quandl):** This is an excellent starting point. It offers a vast repository of financial and economic data. For our purposes, their "Continuous Futures" dataset is very valuable. Which can get both back-adjusted continuous data and the raw data for individual contracts.
    - **Python Library:** ***nasdaq-data-link*** (the official client).
    - **Cost:** Offers a generous free tier, with premium datasets available for a fee.
    - **Recommendation:** This will be our primary choice for the US market example due to its accessibility and data quality.
2. **Interactive Brokers (IBKR):** For those with a brokerage account, the IBKR Trader Workstation (TWS) API is a gold-standard source for high-quality historical and real-time data.
    - **Python Library:** ***ib_insync*** is a popular, user-friendly library for interacting with the TWS API.
    - **Cost:** Requires a funded brokerage account (data access is generally free for account holders, though limitations apply).
    - **Recommendation:** The best choice for serious retail quants moving towards live trading.

B. Data Sources for Chinese Markets

The data ecosystem in China is distinct. Accessing data requires using local providers.

1. **Tushare:** This is the most popular and accessible data provider for Chinese financial markets. It provides comprehensive data on stocks, funds, bonds, and, importantly for us, futures.
    - **Python Library:** ***tushare*** (the official client).
    - **Cost:** It uses a points-based system. A free account provides basic access, while a "Pro" account (requiring a modest annual fee) grants access to more detailed data, including historical daily and minute-bar data for futures.

    - **Recommendation:** This will be our primary choice for the Chinese market example. It's reliable, well-documented, and the standard for many Chinese quants.
2. **AkShare:** A widely-used open-source Python library that sources data by scraping major Chinese financial portals (like East Money and Sina Finance). It is incredibly comprehensive.
    - **Python Library:** ***akshare*** itself.
    - **Cost:** Free.
    - **Recommendation:** An excellent free alternative to Tushare. Its main drawback is that it can be less stable if the source websites change their format.

IV: Futures Trading Rules, Details, and Strategy Development

In this step, we bridge the gap between theory and practice. Understanding the precise mechanics of how futures spreads are traded and how profits are calculated.

4.1 Core Trading Mechanics and Rules

A. Margin: The Key to Capital Efficiency

In futures trading, "margin" is not a down payment but a good-faith deposit required to open and maintain a position. It ensures you can cover potential daily losses. There are two types:

- **Initial Margin:** The amount required in your account to *open* a new futures position.
- **Maintenance Margin:** A lower amount. If your account equity drops below this level due to losses, you will receive a "margin call" and must deposit more funds to bring your account back up to the initial margin level.

Crucial Concept: Spread Margin Credits This is one of the biggest advantages of trading calendar spreads. Exchanges like the CME Group or the Shanghai Futures Exchange (SHFE) recognize that a calendar spread is a hedged position with significantly lower risk than two outright, speculative positions.

Therefore, they provide **margin credits**. The initial margin required to hold a calendar spread is drastically lower than the sum of the margins for the two individual legs.

- **Example (Hypothetical):**
    - Initial Margin for 1 outright WTI Crude Oil (CL) contract: $8,000.
    - Initial Margin for 2 outright CL contracts (1 long, 1 short): *$8,000 + $8,000 = $16,000.*
    - Initial Margin for 1 CL Calendar Spread: **Often as low as $500 - $1,000**.

This massive reduction in required capital makes spread trading highly efficient, allowing a trader to control the same number of contracts with a fraction of the capital, which can significantly enhance returns on capital.

B. How Futures Trading Profits are Calculated

Profit and Loss (P&L) for a calendar spread is calculated based on the *change in the spread's value* between the entry and exit points.

The formula is:

$$P\&L = (\text{Spread Price at Exit} - \text{Spread Price at Entry}) \times \text{Multiplier} \times \text{Number of Spreads}$$

Where the Spread Price is the difference between the prices of the two contracts. Let's use a specific, real-world example.

- **Asset:** WTI Crude Oil (CL) on the CME.
- **Contract Multiplier:** Each $1.00 move in the price of a CL contract corresponds to a $1,000 change in value. The smallest price fluctuation (tick size) is $0.01, which equals $10.
- **Spread:** December 2025 (CLZ25) vs. March 2026 (CLH26).
- **Scenario:** You believe the current contango is too wide and will narrow. You decide to **sell the spread**.

Trade Execution:

1. **Entry (e.g., on August 2, 2025):**
    - You sell the far-month contract: **Sell 1 CLH26** at a price of $75.50.
    - You buy the near-month contract: **Buy 1 CLZ25** at a price of $74.00.
    - **Your Entry Spread Price: *$74.00 - $75.50 = -$1.50*.** (Note: We consistently define the spread as Near - Far. You can also define it as Far - Near, as long as you are consistent).
2. **Exit (e.g., one month later):**
    - Your thesis was correct; the spread narrowed.
    - You close the position by doing the opposite trades:
    - **Sell 1 CLZ25** at its new price of $76.20.
    - **Buy 1 CLH26** at its new price of $77.20.
    - **Your Exit Spread Price: *$76.20 - $77.20 = -$1.00*.**

P&L Calculation:

- **P&L on the Spread: *(Exit Spread - Entry Spread) = (-$1.00) - (-$1.50) = +$0.50*.**
- **Total P&L: *+$0.50 \times*** $1,000 Multiplier = +$500 (before commissions).

Notice that the absolute price of oil went up, but that was irrelevant to our profit. Our profit came exclusively from the spread narrowing by $0.50.

C. Contract Rollover

Since futures contracts expire, any strategy intended to run for longer than a few months must incorporate **rollover logic**. As the near-month contract of our spread approaches its expiration period (specifically, the First Notice Day), its liquidity decreases and its price can become volatile and disconnected from fundamentals.

To avoid this, a trader must "roll" the position. This means:

1. Closing the current spread position (e.g., closing the Dec/Mar spread).
2. Simultaneously opening a new spread position in the next contract cycle (e.g., opening a Mar/Jun spread).

This must be accounted for in our backtester. We will need rules to trigger a roll, for example, "30 days before the near-month contract expires, roll the position."

4.2 How to Develop a Competitive and Profitable Strategy

A successful strategy requires a robust, data-driven "alpha" signal to time entries and exits. Here are the most common approaches.

A. Signal Generation: Mean Reversion with Z-Scores

This is the quintessential statistical arbitrage approach for spreads. It assumes that a spread's value will revert to its historical average over time.

The **Z-Score** is a statistical measure that tells us how many standard deviations a data point is from the mean. It's perfect for identifying statistically significant deviations.

Methodology:

1. **Calculate the Spread Series:** Create a time series of the historical spread value (e.g., ***Price_Near - Price_Far***).
2. **Calculate Rolling Mean & Standard Deviation:** At each point in time, calculate the mean and standard deviation of the spread over a lookback window (e.g., the last 60 days).
3. **Calculate the Z-Score:**

$$Z_{\text{score}} = \frac{(\text{Current Spread} - \text{Rolling Mean of Spread})}{\text{Rolling Std. Dev. of Spread}}$$

4. **Generate Signals:**

   - Entry Signal: When the Z-Score exceeds a certain threshold (e.g., > 2.0 or < -2.0), it signals that the spread is unusually wide or narrow. This triggers a trade betting on a reversion to the mean.
     - If Z-Score > 2.0, the spread is unusually high. Sell the spread.
     - If Z-Score < -2.0, the spread is unusually low. Buy the spread.

- Exit Signal: The position is closed when the Z-Score returns to a level closer to the mean (e.g., crosses back to 0).

B. Signal Generation: Seasonality

Many commodity markets have strong, predictable seasonal supply and demand patterns that influence their term structure.

- **Example: Chinese Rebar (螺纹钢, RB) Futures**
    - Rebar is steel used in construction. Construction activity in China slows dramatically during the cold winter months and ramps up in the spring.
    - This creates a seasonal pattern in the spread between the January (winter) and May (spring) contracts.
    - **Strategy Idea:** A trader might systematically sell the January/May spread (Short Jan / Long May) in the late autumn, hypothesizing that the January contract's price will weaken relative to the May contract as winter construction halts. They would then close the spread in the new year as the market begins to price in the spring construction boom.

C. A Specific, Real Strategy Example

Let's define a complete strategy that we can aim to backtest in the next step.

- **Asset:** Soybean Meal (豆粕, M) on the Dalian Commodity Exchange (DCE).
- **Spread:** May Contract vs. September Contract (e.g., M2605 vs M2609). This spread is influenced by the US planting season (for May) vs. the harvest season (for September).
- **Strategy Logic:** Mean Reversion.
- **Lookback Window:** 90 days for rolling mean and standard deviation.
- **Entry Rule:**
    - If Z-Score < -1.75, **Buy the Spread** (Buy May / Sell Sept).
- **Exit Rule:**
    - Exit the long spread position when the Z-Score crosses back above -0.25 (partial reversion to the mean).
- **Risk Management:**
    - **Stop Loss:** Exit the position immediately if the Z-Score moves further against the position to -2.75.
    - **Position Sizing:** Allocate a fixed amount of capital (e.g., $10,000) per spread traded.

This detailed, rule-based framework is exactly what we need to translate into Python code for our backtester.

V: Implement of Python code

The details of the code can be seen in the ipynb file. Here is the results:

This report provides an analysis of the Z-score-based calendar spread strategy backtested on two distinct futures markets: Soybean Meal and WTI Crude Oil. The results indicate that the strategy's viability is highly dependent on the underlying asset's characteristics and, most critically, is severely impacted by transaction costs in its current configuration.

1. **Comparative Performance Summary**

The table below provides a side-by-side comparison of the key performance metrics for both assets.

Metric	Soybean Meal (AKShare)	WTI Crude Oil (Investing.com)	Analysis
Total Return	+2.24%	-0.19%	Profitable on Soy Meal, but not on WTI.
Sharpe Ratio	4.45	-0.12	Excellent risk-adjusted return for Soy Meal; poor for WTI.
Initial Capital	$500,000	$100,000	Different starting capitals noted.
Final Portfolio Value	$511,221	$99,806	Modest absolute gain for Soy Meal; slight loss for WTI.
Total Trades	693	1209	Extremely high for both, indicating over-trading.
Total Commission Fees	$501,980	$712,700	Critically high. Costs exceed net profits/losses by a massive margin.

Return Before Costs	~102.6%	~712.5%	The raw strategy has a strong profitable signal (alpha).

Note on Return Before Costs: This crucial metric is estimated by adding the Total Commission Fees back to the final P&L. For Soybean Meal, the gross profit was $11,221 (net) + $501,980 (costs) ≈ $513,201. For WTI, it was -$194 (net) + $712,700 (costs) ≈ $712,506.

2. **Asset-Specific Analysis**

Soybean Meal: A Potentially Viable Strategy Hindered by Costs

The backtest on Soybean Meal futures successfully achieved profitability..

- **Performance**: The strategy generated a positive return with an exceptionally high **Sharpe Ratio of 4.45**, indicating that the returns were very consistent and low-risk.
- **Equity Curve**: The top chart shows a smooth, consistently rising equity curve. This is the ideal appearance for a mean-reversion strategy, suggesting the spread's behavior is stable and predictable.
- **The Problem**: The strategy's underlying signal is clearly profitable (generating over 100% in gross returns). However, the **$501,980 in commission fees** consumed nearly all of that profit, leaving a net return of just 2.24%. The issue is **over-trading**, as evidenced by the 693 trades.

WTI Crude Oil: The Wrong Strategy for the Market

The same strategy applied to WTI Crude Oil futures failed to perform.

- **Performance**: The strategy resulted in a net loss and a negative Sharpe Ratio.
- **Equity Curve**: The equity curve is highly volatile and trends sideways, with a significant dip in 2022. This indicates the strategy is not well-suited to the WTI market's dynamics.
- **The Problem**: The WTI spread chart (middle) reveals why. Unlike the stable Soybean Meal spread, the WTI spread experiences major **regime shifts**. Notice the large, sustained spike in 2022. A simple mean-reversion strategy is crushed during such periods because it repeatedly bets against a strong, fundamentally-driven trend. While the raw signal still generated a massive gross profit, the high number of losing trades combined with astronomical commission costs (**$712,700 on a $100k account**) led to a net loss.

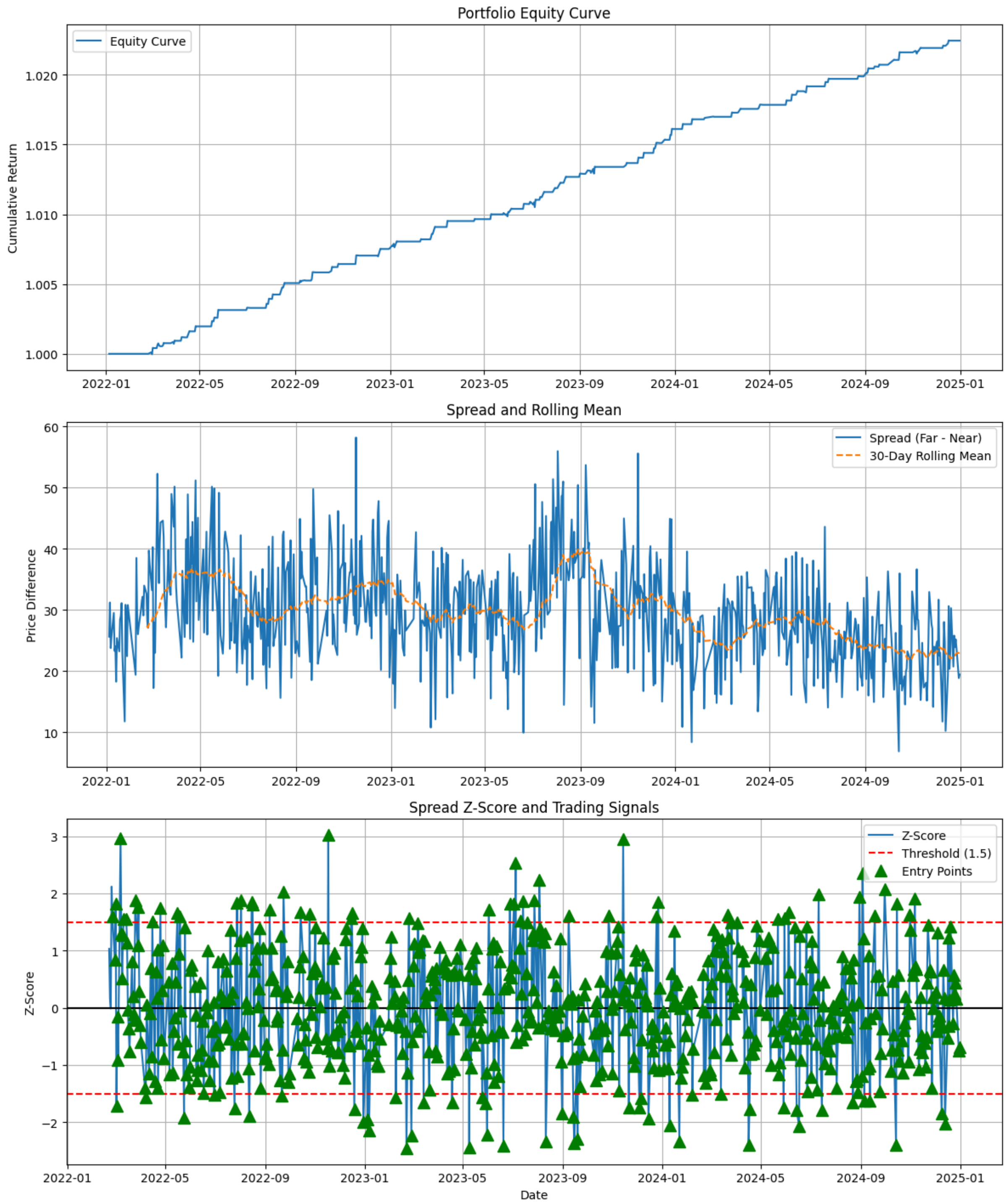
3. **Key Observations & Critical Issues**
    a. **Transaction Costs Are the Primary Obstacle**

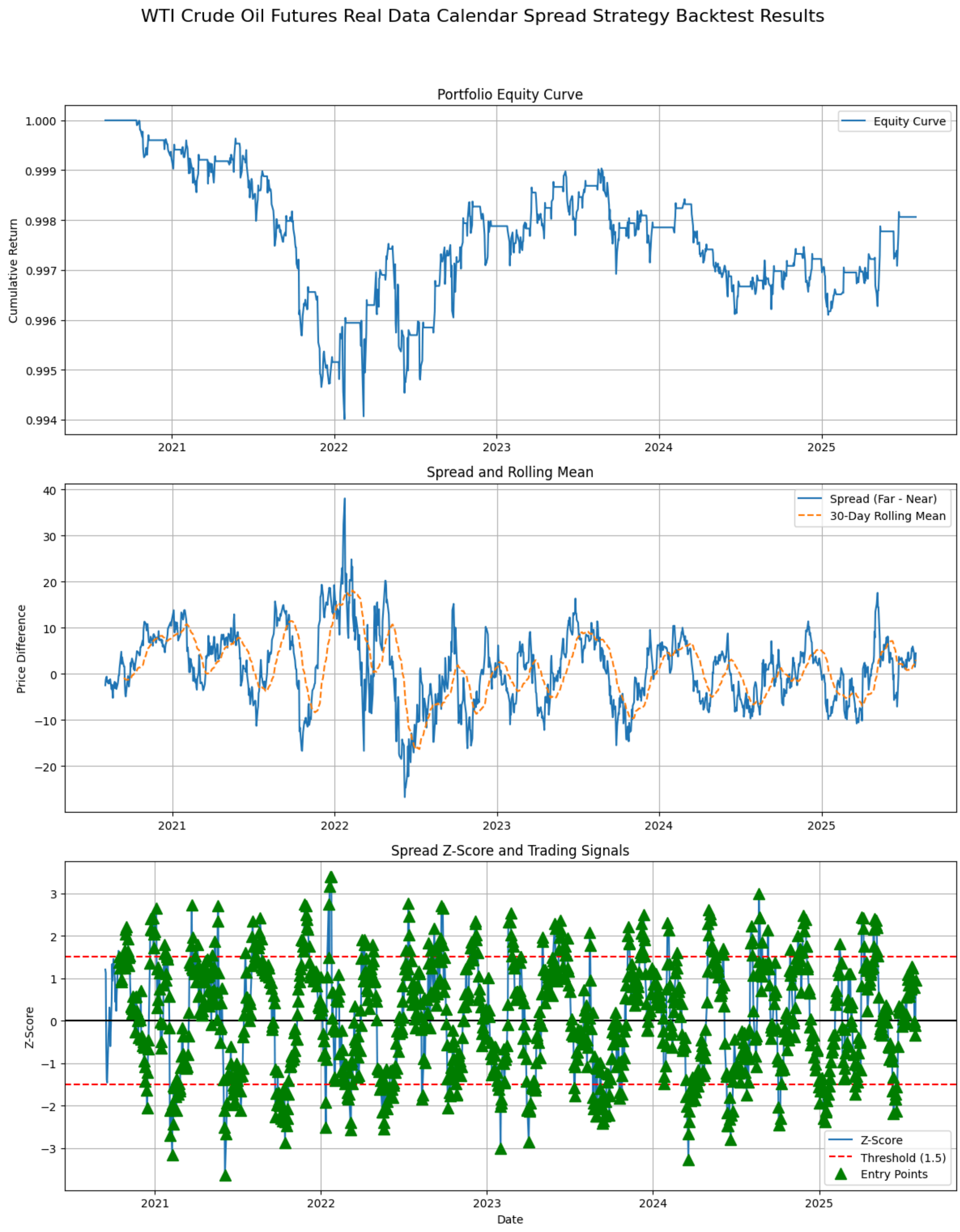
This is the most critical finding. The current strategy parameters, specifically the **Z-score threshold of ±1.5**, are too sensitive. The strategy is triggered by minor, noisy fluctuations around the mean, leading to an unsustainable number of trades. The commission costs are so high that they render both strategies unprofitable in practice.

    b. **Strategy-Market Fit is Crucial**

The results clearly demonstrate that a strategy that works in one market can completely fail in another. The Soybean Meal spread appears to be a classic, stable mean-reverting instrument. The WTI spread, however, is heavily influenced by global supply/demand shocks, geopolitical events, and inventory reports, causing its behavior to deviate from a simple mean for long periods.

Soybean Meal Futures Real Data Calendar Spread Strategy Backtest Results





WTI Crude Oil Futures Real Data Calendar Spread Strategy Backtest Results
Portfolio Equity Curve
Equity Curve
Cumulative Return
1.000
0.999
0.998
0.997
0.996
0.995
0.994
2021
2022
2023
2024
2025
Spread and Rolling Mean
Spread (Far - Near)
30-Day Rolling Mean
Price Difference
40
30
20
10
0
−10
−20
Spread Z-Score and Trading Signals
Z-Score
Threshold (1.5)
Entry Points
3
2
1
0
−1
−2
−3
Date

References

- https://rjofutures.rjobrien.com/rjo-university/a-guide-to-spread-trading-futures
- https://www.cmegroup.com/education/courses/understanding-futures-spreads/futures-spread-overview.html
- https://medium.com/%40marketing_474/calendar-spreads-in-futures-contracts-a-simple-arbitrage-trading-system-a3554411664a
- https://www.marketcalls.in/system-trading/comparision-of-event-driven-backtesting-vs-vectorized-backtesting.html
- https://onepagecode.substack.com/p/the-contrasting-worlds-of-vectorized-fa3