Intraday Mean-Reversion Trading on Futures Spreads

A Minute-Frequency, Execution-Aware Quantitative Study

TKSS Ashok
Department of EE , IIT Bombay

Abstract

This report presents the design, implementation, and evaluation of an intraday mean-reversion trading strategy applied to relative futures spreads using minute-frequency data. The study emphasizes realistic execution modeling, robust data handling, and systematic hyperparameter selection. Multiple strategy variants are evaluated under different slippage assumptions, demonstrating that profitability in intraday statistical arbitrage critically depends on conservative entry thresholds and careful modeling of transaction costs. Extensions toward more advanced statistical and machine learning frameworks are also discussed.

1 Assumptions and Market Setting

The strategy is developed under the following assumptions:

- Trades are executed on exchange-traded equity futures with sufficient intraday liquidity.
- Minute-level OHLC and volume data are available for both front- and next-month contracts.
- Transaction costs and bid–ask spreads are observable and stable within short intraday horizons.
- Positions are intraday-only; no overnight risk is carried.
- Execution occurs at or within the bid–ask spread, subject to modeled slippage.

These assumptions are standard in intraday statistical arbitrage research and allow isolation of signal quality from overnight risk premia.

2 Mathematical Formulation

2.1 Spread Definition

For each underlying asset, let $P_t^{(1)}$ and $P_t^{(2)}$ denote the front-month and next-month futures prices at time t. The relative futures spread is defined as:

$$S_t = \frac{P_t^{(2)} - P_t^{(1)}}{P_t^{(1)}}. \tag{1}$$

This formulation normalizes the spread by the price level, enabling comparison across assets with different price scales.

2.2 Normalization and Z-score

To capture mean-reverting deviations, the spread is standardized using rolling statistics:

$$Z_t = \frac{S_t - \mu_t}{\sigma_t}, \tag{2}$$

where (μ_t, σ_t) are rolling estimates of the mean and standard deviation computed over a long lookback window (approximately 60 trading days). A warm-up period is enforced to ensure statistical stability.

3 Trading Strategy Logic

3.1 Entry and Exit Rules

The strategy is symmetric and threshold-based:

- **Entry:** Enter a spread position when $|Z_t| \geq E$.
- **Take-Profit:** Exit when $|Z_t| \leq E - TP$.
- **Stop-Loss:** Exit when $|Z_t| \geq E + SL$.
- **Forced Exit:** All positions are closed at end-of-day or contract expiry.

Only incremental single-lot position changes are allowed per minute, reflecting realistic intraday execution constraints.

3.2 Position Sizing and Constraints

Position sizes are capped using:

- Maximum delta exposure limits
- Liquidity-based throttling using traded volume (TTQ)
- Time-of-day filters to avoid illiquid market phases

These constraints significantly reduce backtest over-optimism.

4 Execution Cost and Slippage Modeling

Transaction costs are decomposed into brokerage, exchange fees, and slippage. Slippage is modeled explicitly for both legs of the spread.

4.1 Slippage Models

Two slippage regimes are analyzed:

$$\text{Half-spread:} \quad \frac{|s|}{2} \times \text{shares per lot} \times |\Delta| \tag{3}$$

$$\text{Quarter-spread:} \quad \frac{|s|}{4} \times \text{shares per lot} \times |\Delta| \tag{4}$$

This sensitivity analysis illustrates the strong dependence of intraday profitability on execution assumptions.

5 Mean-Reversion Backtesting Engine

1. Key Parameters and Knobs

- **Rolling window:** $ROLL\_WINDOW\_DAYS = 60$, $MIN\_PERIODS = 120$ (lookback for SMA/STD)
- **Winsorization:** $TRIM\_Q\_LOW/HIGH$, $WINS\_Q\_LOW/HIGH$ for outlier clipping
- **Warmup:** $NO\_TRADE\_WARMUP\_DAYS = 30$ to avoid unreliable early trades
- **Trading thresholds:** Encapsulated in `RelZParams(entry, tp_off, stop_off)`

2. Data Preparation

- `load_one_file()` reads CSVs, normalizes numeric columns, builds timestamps.
- `parse_future_name()` extracts underlying, expiry year, and month.
- `prepare_spread_frame()`:
  - Selects front (FUT1) and next (FUT2) contracts per underlying
  - Merges cash market data
  - Computes normalized spread: spread $= \frac{2(mid\_FUT2 - mid\_FUT1)}{mid\_FUT1 + mid\_FUT2}$
  - Keeps cumulative traded quantities (TTQ) for liquidity throttling

3. Rolling Statistics and Winsorization

- `add_global_stats()` computes rolling SMA/STD on spreads.
- `_rolling_winsor_stats()` computes winsorized mean/std for spreads and traded volume.
- `build_rolling_winsor_stats_per_underlying()` runs this per underlying in parallel.

4. Position Management

- `decide_delta_relative()` determines lot change:
  - Flat $\rightarrow$ entry if Z crosses $\pm$ entry threshold
  - In position $\rightarrow$ exit on take-profit or stop-loss
  - Applies small-threshold checks using transaction costs
- `simulate_mean_reversion()` executes the backtest:
  - Iterates over timestamps per underlying
  - Updates realized/unrealized PnL, gross/net, slippage, and trading costs
  - Applies throttling, max lots, liquidity checks
  - Handles forced close on FUT1 expiry and spread limit exits

5. Daily Aggregation

- `daily_breakdown()` computes per-day metrics:
  - $daily\_realized\_cf$, $daily\_pnl\_total$, end-of-day equity
  - Lots traded and
  - Carries forward previous day's unrealized PnL

6. Universe Construction

- `build_universe()`:
  - Reads all CSVs in a folder (parallelized)
  - Prepares spreads per timestamp per underlying
  - Adds rolling winsorized statistics and Z-scores

7. Simulation Flow

1. Build universe: `build_universe()` → clean data → add stats
2. Per-minute backtest: `simulate_mean_reversion()` → update PnL, lots, and trade logs
3. Daily aggregation: `daily_breakdown()`
4. Grid search (optional): `run_relz_grid()` for $(entry, TP, SL)$ combinations

8. PnL Components

- **Gross PnL:** own_amount + current value of open lots
- **Net PnL:** Gross PnL minus costs (trading cost + slippage)
- **Slippage:** spr_slpg_pnl1 and spr_slpg_pnl2 capture execution impact
- **Trading Costs:** proportional to lot notional, applied per transaction

9. Improvements / Notes

- Trades are liquidity-aware via TTQ limits.
- Vectorization could improve performance over row-wise loops.
- Dynamic lot sizing or multi-lot scaling could enhance realism.
- Modular architecture allows easy addition of new filters or risk rules.

6 Hyperparameter Selection

This study evaluates a limited set of manually chosen parameter configurations for the entry (E), take-profit (TP), and stop-loss (SL) thresholds. No exhaustive grid search is performed. Each configuration is tested independently, and parameter values are encoded in output filenames to ensure traceability. Performance is compared at both the per-underlying and portfolio levels.

7 PnL Decomposition and Performance Attribution

Intraday strategies are highly sensitive to execution frictions. To properly evaluate economic viability, portfolio performance is decomposed into its fundamental components: signal-driven returns, transaction costs, and slippage.

7.1 PnL Decomposition Framework

For each strategy, total profitability can be expressed as:

$$\text{Net PnL} = \underbrace{\text{Gross PnL}}_{\text{Signal Quality}} - \underbrace{\text{Transaction Costs}}_{\text{Fees and Brokerage}} - \underbrace{\text{Slippage}}_{\text{Execution Impact}} . \tag{5}$$

Gross PnL reflects the strength of the mean-reversion signal prior to execution effects, while transaction costs and slippage represent structural frictions inherent to intraday trading.

7.2 Gross PnL: Signal Strength Analysis

Strategy	Gross PnL ()	Contracts Traded
Strategy 1	39,326	1,422
Strategy 2	73,662	338
Strategy 3	819,044	1,554

Aggressive entry thresholds in Strategy 1 lead to frequent trades dominated by noise, resulting in weak gross profitability. Strategy 2 improves signal quality by filtering smaller deviations but remains limited in magnitude. Strategy 3 exhibits significantly stronger gross returns, indicating that wider entry thresholds capture larger and more persistent mean-reverting dislocations.

7.3 Transaction Costs: Turnover Effects

Strategy	Transaction Costs ()	Cost per Contract ()
Strategy 1	327,255	230
Strategy 2	74,880	221
Strategy 3	334,720	215

Transaction costs scale approximately linearly with turnover. While Strategy 2 achieves substantial cost reduction via lower trading frequency, Strategy 3 remains cost-efficient on a per-contract basis due to better trade selection and higher per-trade profitability.

7.4 Slippage: Execution Sensitivity

Strategy	Total Slippage ()
Strategy 1	1,239,670
Strategy 2	291,854
Strategy 3	1,169,531

Slippage is the dominant negative PnL component across all strategies. Strategy 1 is particularly vulnerable due to frequent position changes. Strategy 3 incurs high absolute slippage but successfully amortizes execution costs over larger price reversions, preserving profitability.

7.5 Net PnL and Economic Viability

Strategy	Net PnL ()	Net PnL per Contract ()
Strategy 1	-287,929	-202
Strategy 2	-1,219	-4
Strategy 3	484,324	312

Only Strategy 3 generates economically meaningful returns, demonstrating that intraday mean-reversion strategies must generate sufficient gross PnL per trade to overcome execution frictions.

7.6 Cross-Strategy Insights

Strategy 1 fails due to excessive trading and slippage dominance. Strategy 2 demonstrates robustness and capital preservation but lacks sufficient edge magnitude. Strategy 3 achieves an optimal balance between signal strength, trading frequency, and execution realism, resulting in superior net and risk-adjusted performance.

7.7 Opportunities for Improvement

Potential enhancements include:

- Slippage-aware execution using volume participation limits and time-of-day filters
- Volatility-scaled entry thresholds to adapt to changing market conditions
- Regime-aware signal modulation using Hidden Markov Models
- Trade netting and execution batching to reduce turnover

8 Empirical Results

8.1 Strategy Variants

Three representative configurations are evaluated:

- Strategy 1: $E = 1.5$, $TP = 0.5$, $SL = 1.5$ (half-spread slippage)
- Strategy 2: $E = 2.5$, $TP = 0.5$, $SL = 2.5$ (quarter-spread slippage)
- Strategy 3: $E = 2.75$, $TP = 1.0$, $SL = 2.25$ (quarter-spread slippage)

8.2 Portfolio-Level Comparison

Strategy	Net PnL	Contracts	Max Drawdown	Profit Factor
Strategy 1	-287,929	1,422	377,808	0.12
Strategy 2	-1,219	338	197,244	0.98
Strategy 3	484,324	1,554	245,186	2.45

8.3 Discussion of Results

Aggressive entry thresholds lead to frequent trading and high sensitivity to slippage, resulting in large drawdowns (Strategy 1). Conservative thresholds dramatically reduce trading frequency and costs (Strategy 2), but profitability remains marginal. Strategy 3 achieves a balance between signal strength and execution realism, yielding superior net and risk-adjusted returns.

9 Rationale Behind the Final Strategy

The final configuration is chosen for its robustness rather than peak gross returns. Wider entry thresholds reduce noise-driven trades, while asymmetric take-profit and stop-loss levels improve payoff convexity. The result is a strategy that scales better across assets and market conditions.

10 Limitations and Future Work

10.1 Improved Data Imputation

The current pipeline relies on basic forward-filling and defensive skipping of defective data. Future improvements include:

- State-space models (Kalman filtering) for latent price reconstruction
- Variational Autoencoders (VAEs) for high-dimensional intraday data imputation

10.2 Advanced Trading Models

Beyond static threshold-based rules, future research directions include:

- Hidden Markov Models for regime-aware mean reversion
- Reinforcement learning for adaptive execution
- Volatility-scaled and risk-targeted position sizing

11 Conclusion

This project demonstrates that intraday mean-reversion strategies can be viable when developed with execution realism, robust data handling, and systematic parameter evaluation. The results emphasize that modeling assumptions are as critical as signal design, and they motivate the use of more advanced statistical and machine learning techniques in future iterations.