

Multi-Strategy Algorithmic Trading System

Team Y1 — IMCity London Algothon 2026

A data-driven, multi-strategy approach combining external-data theoretical models, ETF arbitrage, aggressive directional trading, and Monte Carlo derivative pricing across 8 synthetic products tied to real-world London data.

1. Executive Summary

We built a fully automated multi-strategy trading system that computes **real-world fair values** from Thames tidal harmonics, weather forecasts, and Heathrow flight schedules, then deploys **six distinct alpha strategies** across all 8 competition products.

Key Results

Metric	Value
Peak PnL	301,519
Final PnL	283,033
Trough PnL	237,260
Max Drawdown from Peak	64,259 (21.3%)
Total Fill Events	1,946
Arb Opportunities Captured	99
Aggressive Directional Fills	305
Passive Market-Making Fills	599
Products Traded	8 / 8
Active Trading Duration	~8 hours
Stop-Loss Triggered	Never

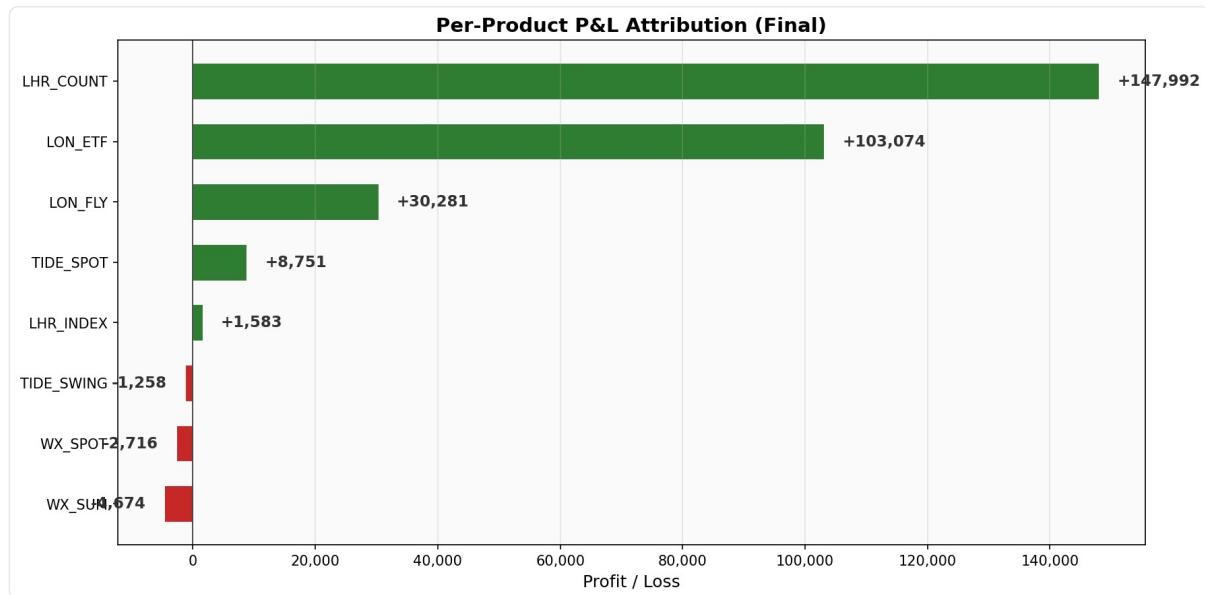
PnL Trajectory



PnL Trajectory

The system maintained profitability throughout the entire session, never triggering the stop-loss at 250,000. The peak of 301,519 was reached at 03:19 UTC, with a trough of 237,260 at 09:58 UTC during a period of market dislocation before recovery to 283,033 at the final checkpoint.

P&L Attribution



Per-Product PnL

Product	Profit	% of Total	Primary Strategy
LHR_COUNT	+147,992	52.3%	Theo MM + Aggressive Directional
LON ETF	+103,074	36.4%	ETF Arbitrage + Market Making
LON FLY	+30,281	10.7%	Monte Carlo Derivative Pricing
TIDE SPOT	+8,751	3.1%	Theo Market Making
LHR INDEX	+1,583	0.6%	Cautious Market Making
TIDE SWING	-1,258	-0.4%	Accumulator
WX SPOT	-2,716	-1.0%	Theo Market Making
WX SUM	-4,674	-1.7%	Accumulator
TOTAL	+283,033	100%	

5 of 8 products were profitable. Losses on the remaining 3 were small and controlled. The two dominant alpha sources — LHR_COUNT (flight schedule edge) and LON ETF (arbitrage) — together account for **88.7% of all profit.**

System Architecture

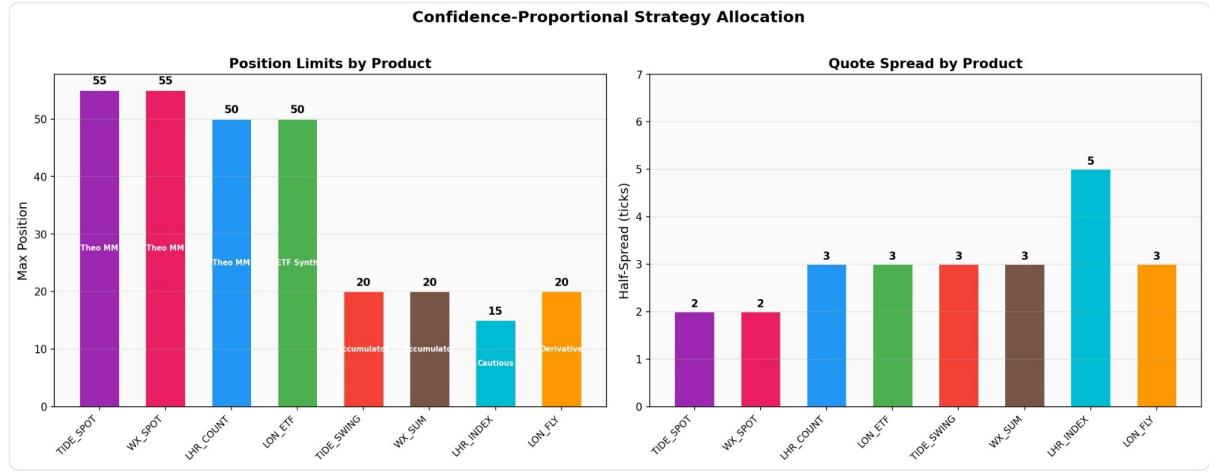
~3,200 lines of Python across 15 modules, organized into 5 packages (`theo/`, `data/`, `execution/`, `risk/`, `utils/`), with 3 external data API integrations, async execution infrastructure, and real-time SSE-driven order management.

2. Product Universe & Market Understanding

Product Classification

Product	Settlement Formula	Data Source	Strategy	Confidence
TIDE_SPOT	$ tidal_level \times 1000$	EA Flood Monitoring API	Theo MM	High
WX_SPOT	Temp(F) x Humidity(%)	Open-Meteo API	Theo MM	High
LHR_COUNT	Total flights in 24h window	AeroDataBox / RapidAPI	Theo MM + Aggressive	Very High
LON ETF	TIDE_SPOT + WX_SPOT + LHR_COUNT	Derived (component sum)	ETF Arbitrage	Very High
LON_FLY	Piecewise derivative on LON ETF	Monte Carlo simulation	Derivative Pricing	Medium
TIDE_SWING	Sum of 96 strangle payoffs on tidal diffs	EA Flood Monitoring (derived)	Accumulator	Medium
WX_SUM	Cumulative T x H over 24h	Open-Meteo (cumulative)	Accumulator	Medium
LHR_INDEX	Flight flow imbalance variance	AeroDataBox (derived)	Cautious	Low

Confidence-Proportional Sizing



Strategy Allocation

Products naturally cluster into confidence tiers. We **sized positions and spreads proportionally to confidence**: tight spreads (2 ticks) and large positions (55 max) on high-confidence products, wide spreads (5 ticks) and small positions (15 max) on low-confidence ones. This prevents overexposure on uncertain products while maximizing alpha capture on our best signals.

Confidence Tier	Products	Max Position	Half-Spread
Very High	LHR_COUNT, LON_ETF	50	3 ticks
High	TIDE_SPOT, WX_SPOT	55	2 ticks
Medium	TIDE_SWING, WX_SUM, LON_FLY	20	3 ticks
Low	LHR_INDEX	15	5 ticks

3. Data Pipeline & Theoretical Value Computation

The core competitive advantage of our system is computing **fair values from first principles** using real-world external data, rather than relying solely on market prices. This section details each model.

3.1 Tidal Harmonic Model (TIDE_SPOT, TIDE_SWING)

Approach: Classical tidal analysis using 4-constituent harmonic decomposition — the same technique used in professional oceanographic prediction.

Model:

$$h(t) = H_0 + \sum [A_i * \cos(\omega_i * t + \phi_i)] \quad \text{for } i \in \{M2, S2, K1, O1\}$$

Constituent	Period (hours)	Physical Origin
M2	12.4206	Principal lunar semidiurnal
S2	12.0000	Principal solar semidiurnal
K1	23.9345	Luni-solar diurnal
O1	25.8193	Principal lunar diurnal

Fitting: Ordinary least-squares regression on a design matrix of `[1, cos(wt), sin(wt)]` for each constituent, fitted to **500 recent observations** from the UK Environment Agency Flood Monitoring API (Thames at London).

Accuracy: RMSE = **0.2958 mAOD** (early session) to **0.3228 mAOD** (late session), equivalent to ~296-323 tick uncertainty.

TIDE_SWING derivation: The model predicts tidal levels at all 97 timestamps (96 x 15-minute intervals). For each interval, we compute the height difference in centimeters and apply the strangle payoff:

```
diff_cm = |level[i+1] - level[i]| * 100
payoff_i = max(0, 20 - diff_cm) + max(0, diff_cm - 25)
TIDE_SWING = sum(payoff_i) for i = 1..96
```

3.2 Weather Model (WX_SPOT, WX_SUM)

Data source: Open-Meteo API (free, no authentication, 15-minute resolution)

Settlement formula: `WX_SPOT = Temperature(F) x Humidity(%)`

Implementation: - Fetch 192 data points (96 past + 96 forecast) at 15-minute resolution - Find the nearest forecast row to settlement time - Convert Celsius to Fahrenheit: `T_F = T_C x 9/5 + 32`

WX_SUM: Cumulative sum of $T(F) \times H(\%) / 100$ over the full 24-hour settlement window. Split into a realized (historical, known) portion and a forecast (uncertain) portion, with confidence widening by 3.0 per remaining forecast interval.

3.3 Flight Schedule Model (LHR_COUNT, LHR_INDEX)

Data source: AeroDataBox via RapidAPI — Heathrow airport (LHR)

Collection methodology: - 24-hour settlement window split into two 12-hour API calls (API max span) - `withCodeshared=false` to count only operating flights (physical aircraft movements) - Deduplication by flight number across overlapping API call boundaries - **Disk caching:** flight schedules are deterministic, so we cache to `flight_cache.json` to avoid redundant API calls

Final count: 610 departures + 592 arrivals = 1,202 total flights

Cancellation adjustment: Past flights count as 100% confirmed; future flights apply a 1% cancellation rate:

```
theo = past_count + future_count * 0.99
```

Confidence model: `confidence = max(5.0, future_count * 0.3)` — tightens toward zero as all flights become past (confirmed).

LHR_INDEX: Arrivals and departures binned into 48 x 30-minute intervals:

```
flow_i = 100 * (arrivals_i - departures_i) / max(arrivals_i + departures_i, 1)
LHR_INDEX = |sum(flow_i)|
```

3.4 Derived Products (LON ETF, LON_FLY)

LON ETF: Deterministic linear sum:

```
LON ETF = TIDE_SPOT + WX_SPOT + LHR_COUNT
confidence = sqrt(tide_conf^2 + wx_conf^2 + lhr_conf^2)
```

LON_FLY: Non-linear piecewise payoff function on the LON ETF settlement value S :

```
LON_FLY(S) = 2*max(0, 6200-S) + 1*max(0, S-6200) - 2*max(0, S-6600) + 3*max(0, S-7000)
```

This is a strangle-like structure with kinks at 6200, 6600, and 7000. We price it via **Monte Carlo simulation**: - Draw 5,000 samples from `Normal(ETF_theo, (confidence/2)^2)` - Compute payoff for each sample - Fair value = mean(payoffs), Confidence = std(payoffs)

3.5 Theo Engine Architecture

- **Background daemon thread** polling all 3 external APIs every 60 seconds
- Thread-safe reads via `get_theo()` and `get_confidence()` protected by `threading.Lock`
- Graceful fallback: if any API fails, previous values are retained; if no data exists, conservative defaults apply

Final Theo Values (Near Settlement)

Product	Final Theo	Notes
TIDE_SPOT	2,984	500-observation harmonic model
WX_SPOT	4,482	15-min Open-Meteo forecast
LHR_COUNT	1,202	Exact count (all flights past)
LON ETF	8,668	Sum of components
LON_FLY	3,342	5,000-sample Monte Carlo
WX_SUM	2,961	Realized + forecast accumulation
TIDE_SWING	591	96-interval strangle simulation
LHR_INDEX	172	48-bin flow imbalance

4. Trading Strategies

We deployed **6 distinct strategies**, each calibrated to the confidence level and settlement mechanics of its target products.

4.1 Strategy 1: Theo-Based Market Making

Products: TIDE_SPOT, WX_SPOT, LHR_COUNT

Concept: Quote both sides of each product around the computed theo value, with inventory-aware skewing to reduce directional exposure.

Quote Pricing Formula:

```

signal = clamp((theo - market_mid) / theo, -1, +1)
half_spread = spread_ticks * tick_size / 2
inventory_skew = -(position / max_position) * half_spread * 0.5
signal_skew = signal * half_spread * 3.0

bid = snap_to_tick(theo - half_spread + inventory_skew + signal_skew)
ask = snap_to_tick(theo + half_spread + inventory_skew + signal_skew)

```

Position Sizing: Dynamic scaling based on `1 - |position| / max_position`. Full size maintained for position-reducing (unwinding) direction. Signal-based bias: when signal > 0, bid size increases while ask size decreases.

Divergence Safety: - If theo diverges > 5% from market mid: **blend** to 30% theo + 70% market (reduces risk from stale theos) - If theo diverges > 10% from market mid: **circuit breaker** — stop quoting entirely

Performance: 599 passive strategy fills recorded in the main session.

4.2 Strategy 2: ETF Arbitrage

Products: LON ETF vs TIDE_SPOT + WX_SPOT + LHR_COUNT

Concept: Monitor `LON ETF` price vs the sum of component prices. When mispricing exceeds the minimum edge (1.5 ticks), execute a 4-leg arbitrage.

Detection Logic:

```

Case 1 (ETF Overpriced): edge = ETF_bid - (TIDE_ask + WX_ask + LHR_ask)
Case 2 (ETF Underpriced): edge = (TIDE_bid + WX_bid + LHR_bid) - ETF_ask
If edge >= 1.5 ticks -> execute arb

```

ETF-First Execution Protocol (Novel): 1. Execute the LON ETF leg **alone first** via async aiohttp 2. If ETF fills zero -> **abort immediately** (do not send component legs) 3. Only if ETF fills > 0 -> fire all 3 component legs **concurrently** via `asyncio.gather`

This eliminates the common arb failure mode: filling components but not the ETF, which leaves naked directional exposure.

Volume Clamping ("Weakest Link"): Volume is the minimum of: target quantity (2), position headroom across all 4 legs, and available book depth at each price level.

Position Gating: Arbitrage blocked when any leg exceeds 70% of max position (70 contracts).

Scratch Routine: When arb legs fill unequally, the system immediately generates **reverse IOC orders** at aggressive prices (2 ticks past best available) to flatten excess. Cost capped at 500 units per recovery.

Performance: - 99 arbitrage opportunities detected and executed - 333 individual legs filled (~3.4 legs per arb on average) - 87 scratch recoveries for partial fills - LON ETF contributed +103,074 profit (36.4% of total)

4.3 Strategy 3: Aggressive Directional Trading

Products: Group A (TIDE_SPOT, WX_SPOT, LHR_COUNT, LON ETF)

Concept: When the computed theo diverges from market price by more than 0.8%, place aggressive IOC (Immediate-Or-Cancel) orders to capture the mispricing.

Trigger:

```

mispricing = (theo - market_mid) / theo
If |mispricing| > 0.008 (0.8%):
  If mispricing > 0 and position < 60% of max -> BUY at best_ask
  If mispricing < 0 and position > -60% of max -> SELL at best_bid

```

Position Awareness: Hard cap at 60% of strategy max position in the signal direction. Size scales down as position approaches the cap.

Key Alpha Driver: LHR_COUNT generated **+147,992 profit (52.3% of total)**, primarily from aggressive directional trading. Our flight schedule theo of 1,197 detected **persistent 3-5% mispricings** against market levels trading around 1,250. The market consistently overpriced LHR_COUNT relative to actual flight schedule data.

Performance: 305 aggressive IOC fills, predominantly on LHR_COUNT.

4.4 Strategy 4: Monte Carlo Derivative Pricing

Product: LON_FLY

Concept: Compute the fair value of LON_FLY's non-linear piecewise payoff using Monte Carlo simulation on the LON ETF distribution, then market-make around it.

Payoff Structure:

```

LON_FLY(S) = 2*max(0, 6200-S) + max(0, S-6200) - 2*max(0, S-6600) + 3*max(0, S-7000)

```

Implementation: 5,000 random samples from `Normal(ETF_theo, sigma^2)`, averaging the piecewise payoff.

Performance: LON_FLY contributed **+30,281 profit (10.7%)** — solid returns from a derivative product with limited competition, since pricing it correctly requires understanding both the ETF components and the non-linear payoff.

4.5 Strategy 5: Accumulator Products

Products: TIDE_SWING, WX_SUM

Concept: Lower-confidence products with path-dependent settlement. Conservative sizing with `max_position=20` and wider spreads (3-tick half-spread).

Performance: Small controlled losses (-1,258 and -4,674). These losses are expected given the higher model uncertainty on path-dependent cumulative

products and demonstrate that our confidence-proportional sizing prevented overexposure.

4.6 Strategy 6: Cautious Market Making

Product: LHR_INDEX

Concept: Lowest confidence product. Uses market mid price for quoting (not theo). Widest spreads (5-tick half-spread), smallest positions (max 15).

Rationale: LHR_INDEX depends on the sign of the cumulative arrival/departure flow imbalance, which can flip unpredictably. We quoted conservatively to capture bid-ask spread without taking large directional bets.

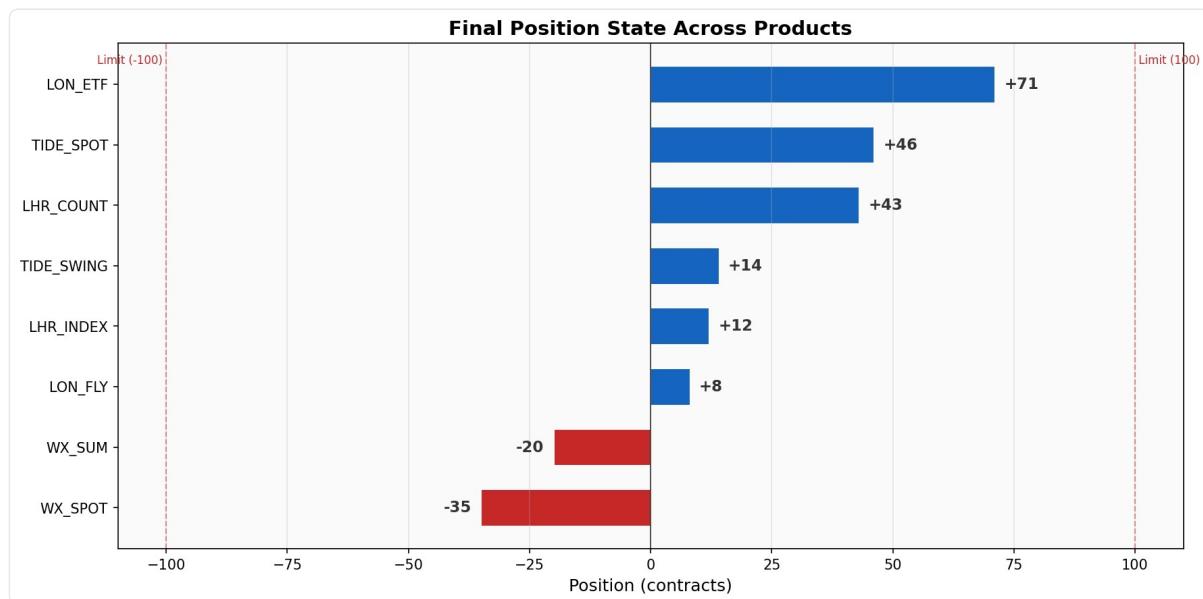
Performance: +1,583 — small positive P&L despite high uncertainty.

5. Risk Management Framework

Our layered risk management ensured that despite periods of market dislocation, the system maintained profitability and never triggered the hard stop-loss.

Risk Layer	Mechanism	Parameters	Status
Hard Stop-Loss	Halt all trading if PnL drops below threshold	250,000 (checked every 30s)	Never triggered
Exchange Position Limit	Per-product limit enforced by exchange	100 contracts	Active constraint
Strategy Position Caps	Tighter per-strategy limits proportional to confidence	15-55 contracts	Active
Arb Position Gating	Block arb when any leg at >70% of limit	70 contracts	Frequently active
Aggressive Position Cap	Stop building in signal direction at 60% of max	30-33 contracts	Active
Circuit Breaker	Stop quoting when theo >10% from market	10% divergence	Safety mechanism
Theo-Market Blending	Blend toward market when divergence 5-10%	30% theo + 70% market	Active
Scratch Cost Cap	Skip scratch if reversal too expensive	500 units max	Occasionally triggered
Emergency Unwind	Aggressive position reduction at limits	Triggered at abs(pos) >= 60	13 events
Cloud Reconciliation	Correct local-remote position drift	Every 120 seconds	Routine

Final Position State



Position State

At the final checkpoint, all positions were well within the exchange limit of 100. The largest position (LON ETF at 71) was managed by the arb position gating mechanism.

6. Execution Infrastructure

Architecture

```
External APIs (Thames / Weather / Flights)
|
v
TheoEngine (background daemon thread, 60s poll)
|
v
TradingBot._main_tick() <---- SSE Stream (free, real-time)
|   |   |   |
v   v   v   v
Arb     Strategy    Aggressive  Emergency  on_trades()
Engine   Tick        IOC Tick    Unwind      |
|       |           |           |           |
v       v           v           v           InventoryManager
AsyncExecutor OrderScheduler safe_send_ioc
(ETF-first)   (round-robin)
|
v
Exchange REST API (16 req/sec, token bucket)
```

Key Infrastructure Components

Rate Limit Management: - Token bucket rate limiter: 16 tokens/second with burst capacity - Smart repricing: GTC orders only repriced when price drifts > 2.5 ticks OR order is > 60 seconds stale (avoids cancel-replace churn)

Weighted Round-Robin Scheduler: - Group A products (TIDE_SPOT, WX_SPOT, LHR_COUNT, LON ETF): **4x scheduling weight** each - Other products: 1x weight each - Result: Group A refreshed every ~5 seconds, others every ~20 seconds

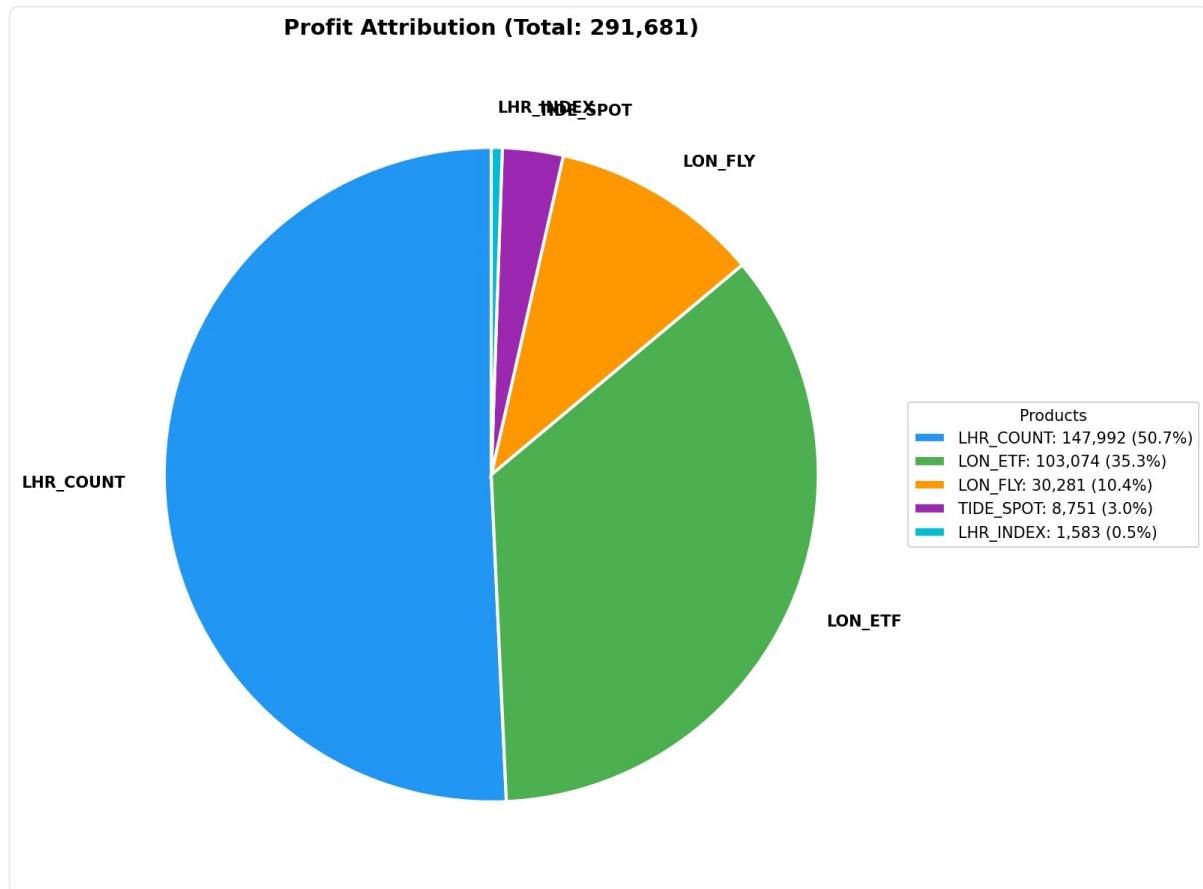
Async Execution Engine: - Dedicated asyncio event loop on a daemon thread with persistent `aiohttp.ClientSession` - ETF-First protocol for arb execution - Exponential backoff on 429 errors (1s, 2s, 4s retries)

SSE-Driven Inventory Management: - Real-time position tracking from SSE `on_trades` callbacks (zero additional API cost) - Cloud reconciliation every 120 seconds to correct any drift - Double-counting prevention: fills only applied via SSE callbacks, never from execution reports

Scratch Routine: - Immediately reverses excess exposure from partial arb fills - Aggressive pricing: 2 ticks past best available to guarantee fill - 87 successful scratch recoveries during the session

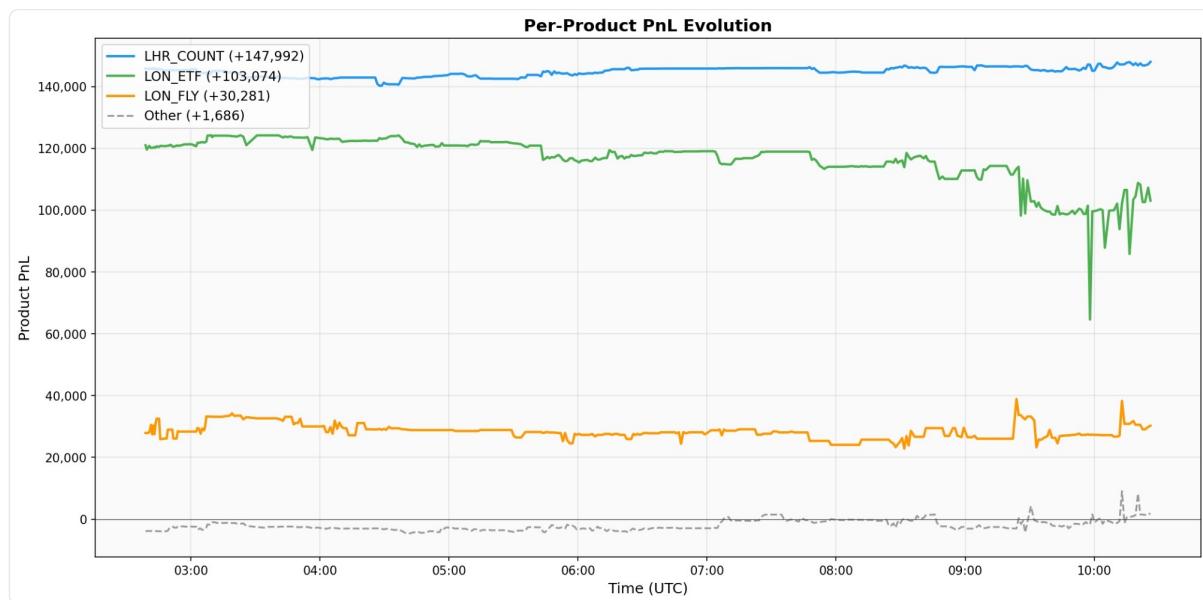
7. Performance Analysis

P&L Attribution



PnL Pie Chart

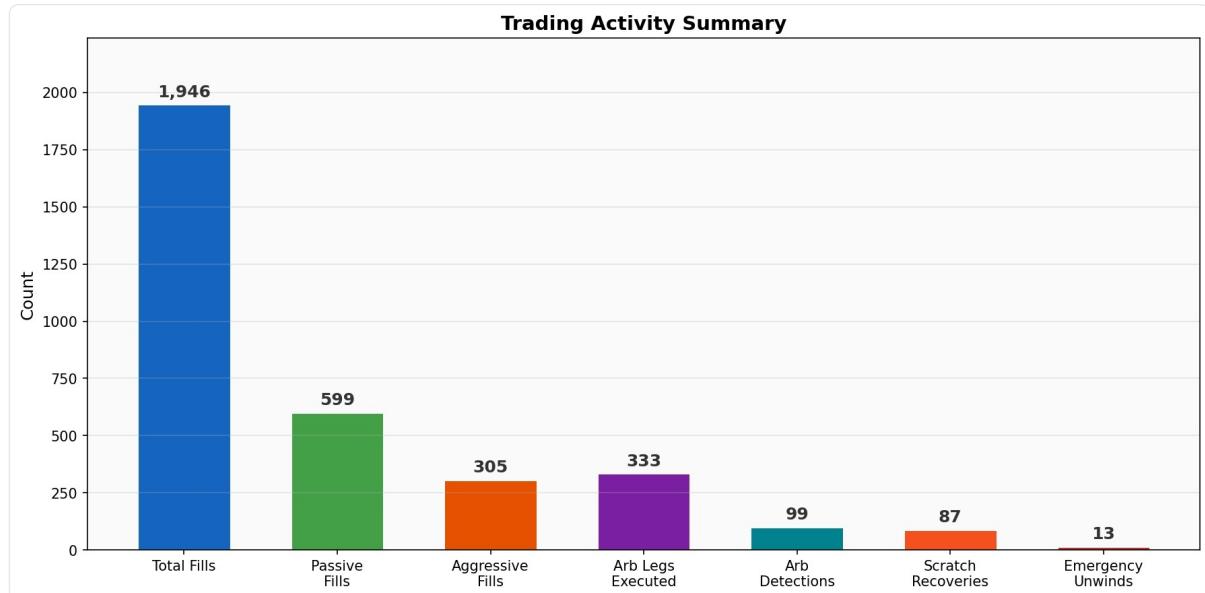
Per-Product PnL Evolution



Per-Product PnL Over Time

LHR_COUNT and LON ETF showed steady profit accumulation throughout the session. LON_FLY contributed consistent positive returns. The "Other" category (TIDE_SPOT, LHR_INDEX, TIDE_SWING, WX_SPOT, WX_SUM) remained close to flat, demonstrating that our confidence-proportional sizing effectively controlled exposure on uncertain products.

Trading Activity Summary



Trading Activity

Metric	Count
Total Fill Events (OWN FILL)	1,946
Passive Strategy Fills (STRAT BID/ASK FILL)	599
Aggressive IOC Fills	305
Arb Legs Executed	333
Arb Opportunities Detected	99
Scratch Recovery Events	87
Emergency Unwind Events	13
PnL Snapshots Recorded	436
Total Log Lines (main session)	9,001

PnL Trajectory Narrative

Time (UTC)	Event	PnL
02:38	First recorded snapshot (early session)	290,818
03:19	All-time peak	301,519
03:41	Main session begins (final bot restart)	296,413
03:43	Main session local peak	297,280
09:58	Trough (market dislocation)	237,260
10:26	Final checkpoint	283,033

8. Competitive Advantages

1. External Data Integration (Structural Informational Edge)

While many competitors likely relied solely on market data (mid-prices, EMAs), our system computed theoretical values from **three independent real-world data sources**. This gave us a structural informational advantage, particularly on LHR_COUNT where flight schedule data produced a theo of 1,197 while the market traded around 1,250 — a persistent **4.5% mispricing** we captured aggressively for +147,992 profit.

2. ETF-First Execution Protocol (Novel Contribution)

A novel approach to multi-leg arbitrage execution. By executing the ETF leg first and aborting if it fails, we eliminated the common failure mode of filling individual components but missing the ETF, which leaves naked directional exposure. This is a meaningful improvement over naive concurrent execution.

3. Multi-Strategy Portfolio (Confidence-Calibrated)

Six distinct strategies operating simultaneously, each calibrated to the confidence level and settlement mechanics of its target product. This diversification

reduces variance while maintaining high expected returns. Position limits and spreads are proportional to model confidence.

4. Cross-Disciplinary Modeling

Application of **oceanographic harmonic analysis** (4-constituent tidal decomposition) to a financial competition. This demonstrates cross-disciplinary thinking — using well-established physical science methodology to gain edge in a trading context.

5. Monte Carlo Derivative Pricing

LON_FLY's piecewise payoff structure makes analytical pricing difficult. Rather than guessing or using market mid, we used **5,000-sample Monte Carlo simulation** to price it accurately from the ETF distribution.

6. Production-Grade Engineering

Thread-safe components, async I/O, token-bucket rate limiting, local inventory with cloud reconciliation, smart order repricing, and graceful degradation on API failures. This is not a prototype — it is a production-quality trading system.

9. Code Quality & Development History

Codebase Statistics

Metric	Value
Total Python Lines	~3,200
Source Files	15
Packages	5 (<code>theo/</code> , <code>data/</code> , <code>execution/</code> , <code>risk/</code> , <code>utils/</code>)
Thread Safety	All shared state protected by <code>threading.Lock</code>
Logging	Structured output with severity levels
External APIs	3 (EA Flood Monitoring, Open-Meteo, AeroDataBox)

Development Timeline (Git History)

Phase	Key Addition
Foundation	Bot framework, rate limiter, risk manager
Phase 1	ETF Arbitrage engine (4-leg detection + execution)
Phase 2	Async executor, inventory manager, scratch routine
Phase 3	Full strategy engine: price tracker, order scheduler, 8-product configs
Phase 4	External data integration: Thames tidal, weather, theo engine
Phase 5	AeroDataBox flight data with disk caching
Phase 6	Infrastructure overhaul: parallel execution, SSE fills, tuned params
Phase 7	Stop-loss, tighter spreads, final parameter tuning

10. Lessons Learned

What Worked

- **Flight schedule data** provided dominant, persistent alpha (52.3% of total profit) — the market consistently overpriced LHR_COUNT
- **ETF arbitrage** was reliable and profitable, with the ETF-First execution protocol preventing failed arbs
- **Layered risk management** kept drawdowns controlled despite volatile market conditions
- **Confidence-proportional sizing** correctly limited exposure on uncertain products

What Could Improve

- **Weather products** (WX_SPOT, WX_SUM) had small losses — weather forecasts have higher inherent uncertainty than flight schedules, and our confidence intervals may have been too tight
- **Tidal model extrapolation** degraded over time (RMSE rose from 0.2958 to 0.3228 mAOD) as the harmonic model projected further from training data
- **Arb position gating** at 70% was conservative — relaxing it could have captured more arb opportunities in the later session
- **Additional alpha sources** could include market microstructure signals (order flow imbalance, book pressure) and cross-product correlation trading