# 1 Problem Setup: Two Asynchronous Price Streams

We observe *two* mid-price series

$$\{\, s_{t_i} \}_{i=1}^{N_S}, \qquad \{\, p_{u_j} \}_{j=1}^{N_P},$$

where the time–stamps $t_i$ (spot) and $u_j$ (perpetual) are irregular and generated by their respective matching engines (micro- to millisecond resolution). Importantly, the two calendars rarely coincide:

$$\Pr\{t_i = u_j\} \;\approx\; 0.$$

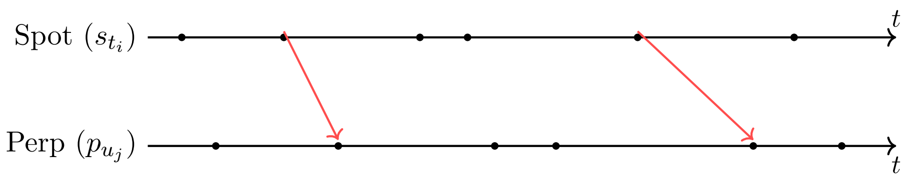


Figure above illustrates the core difficulty: $t_4 = 4.7$ ms in the spot feed has no counterpart in the perpetual feed, while the perpetual tick at $u_3 = 5.1$ ms arrives between two spot stamps.

Our objective is to exploit the few-millisecond causal link between the streams while avoiding the artefacts introduced by naïve time alignment.

### Observed micro-structure facts

- Exchange clocks show **asynchronous arrival**: the same crypto pair prints on spot and perpetual feeds at different micro-/millisecond stamps.
- When a $\geq 0.07\%$ mid-price jump appears in one feed, the **follower market typically echoes it within** 3–5 **ms**.
- Quotes arrive in **bursts**: several book-updates can share the *exact* time-stamp (exchange writes sequence ID, not true nanos).
- Quotes also show **idle gaps**: stretches of 50–100 ms with no update at all (overnight hours, maintenance blips, or bandwidth throttling).

v

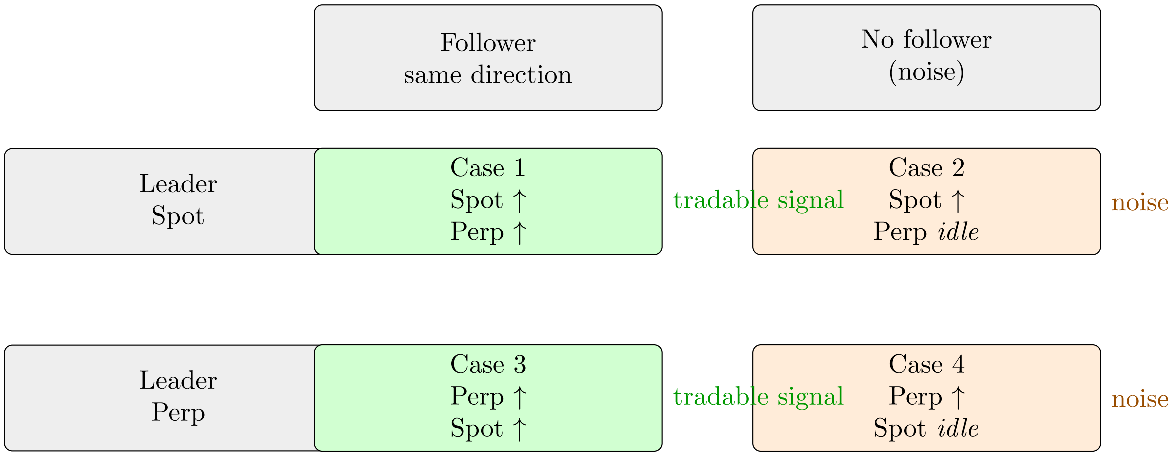


Figure 1: Only four possible outcomes under the model. Cases 2 and 4 are labelled *noise* because the follower market remains idle.

## How we decide "who leads whom" and what is noise

1. **Native-tick pairing (no filling)**
   - Use the exchange time-stamps exactly as received.
   - A jump in Market A is marked "leader" if Market B prints a $\geq 0.07\%$ jump of the *same sign* within the next 5 ms.
   - If no such follower arrives, the event is labelled *noise*.

2. **Forward-fill pairing (1 ms grid, LOCF)**
   - Resample both feeds to an equally-spaced 1 ms clock.
   - Missing slots are filled with the last seen price (Last Observation Carried Forward).
   - A jump is now a $\geq 0.07\%$ change over the previous three grid-points, then the same 5 ms leader/follower rule is applied.
   - Forward-fill can create *synthetic* jumps or hide micro-bursts, so counts differ from the native-tick method.

| Metric | Value | Explanation |
|---|---|---|
| total_spot_jumps | 7411 | Number of price jumps in the spot market exceeding $\pm 0.07\%$ within any 3 ms interval. |
| total_perp_jumps | 12841 | Number of price jumps in the perpetual market exceeding $\pm 0.07\%$ within any 3 ms interval. |
| paired_events | 4126 | Count of jump events in either market that found a matching same-direction jump in the other market within 5 ms (each pair counted once). |
| noise_events | 12000 | Total number of jumps (spot + perp) that did not have a matching reaction in the other market within 5 ms. |
| noise_spot_jumps | 3004 | Number of spot jumps that remained unpaired—i.e. noise in the spot market. |
| noise_perp_jumps | 8996 | Number of perpetual jumps that remained unpaired—i.e. noise in the perpetual market. |
| spot_leads_perp | 2684 | Count of spot jumps that led matching perp jumps within 5 ms (spot drives perp). |
| perp_leads_spot | 1161 | Count of perp jumps that led matching spot jumps within 5 ms (perp drives spot). |

Table 1: Lead–lag summary ($\pm 0.07\%$, 5 ms window) with metric definitions.

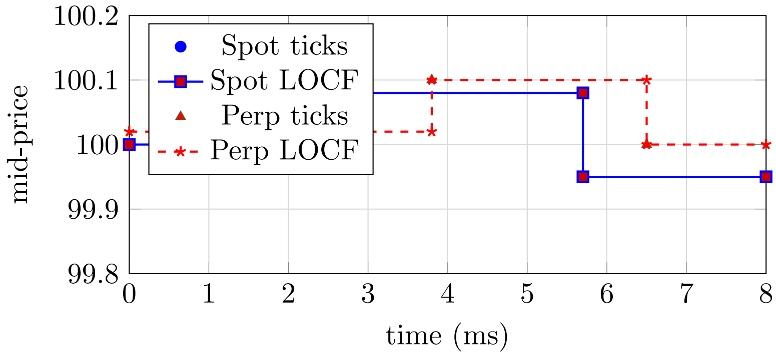


Figure 2: Three-point feeds for spot (blue) and perp (red), with LOCF drawn as step lines holding each price until the next tick.

## Cross-Correlation

Cross-correlation measures how well two time-series align when one is shifted in time.

- **Definition**: Compute the correlation between

  $$s_t \quad \text{and} \quad p_{t+\tau}$$

  over various lags $\tau$.

- **Purpose**:
  - Detect lead–lag relationships (i.e. determine who leads whom).
  - Quantify similarity at different time offsets.
  - Find the time shift that maximizes predictability.

## Cross-Correlation Analysis (Forward-Filled 1 ms Grid)

We know that the spot market leads the futures (perpetual) market. Using a forward-filled, uniform 1 ms mid-price grid and computing log-returns, the Pearson cross-correlation at various lags $\tau$ (spot leads futures by $\tau$ ms) is:
Interpretation:

- The highest correlation occurs at $\tau = 2\,\text{ms}$, with $r \approx 0.063$.
- Correlation declines for larger lags, indicating diminishing predictive power beyond a few milliseconds.

| Lag, $\tau$ (ms) | Cross-corr. |
|---|---|
| 1 | 0.061641 |
| 2 | 0.062578 |
| 3 | 0.049808 |
| 4 | 0.042298 |
| 5 | 0.026826 |

Table 2: Cross-correlation $r(\tau)$ between spot and future returns on a forward-filled 1 ms grid. Positive $r$ indicates spot returns at time $t$ predict future returns at time $t + \tau$.

- All values are positive, confirming that spot returns lead futures returns on average.

## Brief Argumentative Summary

Our jump-pairing analysis found that over 59% of spot and futures jumps remain unpaired (noise), and the peak cross-correlation of log-returns is only $r \approx 0.063$ at a 2ms lag. Together, these facts imply:

- **Dominant Noise:** Most sudden price moves do not carry over between markets within our 5ms window.
- **Weak Signal:** Even at the optimal lag, spot returns explain less than 0.4% of futures variance.

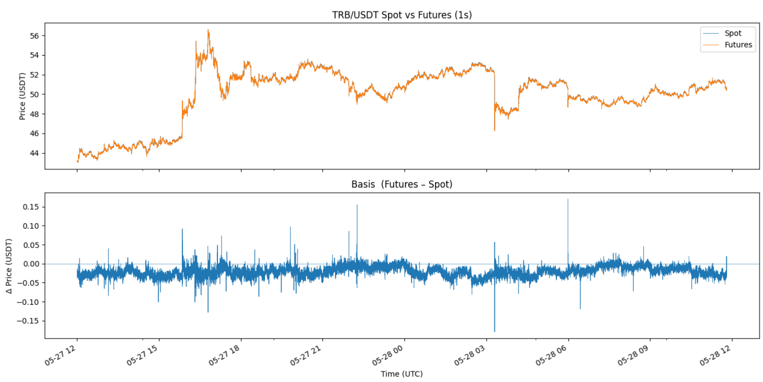


Figure 3: My plot caption.

## Compact Supervised Classification Pipeline

- **Trade-side inference:** Merge each trade to the most recent spot quote (using `pd.merge_asof`), then

$$\text{side} = \begin{cases} +1, & \text{if } P_{\text{trade}} \geq \text{ask}, \\ -1, & \text{if } P_{\text{trade}} \leq \text{bid}, \\ 0, & \text{otherwise.} \end{cases}$$

- **Feature construction (at each spot-jump trigger):**

$$\big[\,3 \text{ ms spot return},\ 20 \text{ ms volatility},\ \text{spread},\ \text{net signed volume},\ \text{trade count}\big].$$

- **Label:** perp mid-price move of ±0.07% within 5 ms *lead* (1) vs. noise (0).
- **Sampling:** 1 000 leads + 1 000 noise for training; remainder for testing (or 70/30 split).
- **Model:**
  - XGBoost (with balanced sampling or `class_weight`)

## Comparison of Two XGBoost Pipelines

### 1. Threshold Calibration (PR AUC Method)

**Average precision (PR AUC)**: 0.1536 **Chosen threshold for ≈50% recall**: 0.6300 (precision = 0.1337)

| Class | Precision | Recall | $F_1$ | Support |
|---|---|---|---|---|
| No-lead (0) | 0.9463 | 0.7310 | 0.8248 | 8431 |
| Lead (1) | 0.1337 | 0.5000 | 0.2110 | 700 |
| Accuracy = 0.7133 | | | | 9131 |

Table 3: XGBoost with threshold calibrated to PR AUC on full hold-out set.

### 2. 70/30 Train/Test Split (Balanced Sample)

| **Class** | **Precision** | **Recall** | $\mathbf{F_1}$ | **Support** |
|---|---|---|---|---|
| No-lead (0) | 0.6575 | 0.6076 | 0.6316 | 316 |
| Lead (1) | 0.5974 | 0.6479 | 0.6216 | 284 |
| Accuracy = 0.6267 | | | | 600 |

Table 4: XGBoost on balanced 70%/30% train/test split.

### 3. Metric Definitions

| **Metric** | **Formula** | **Interpretation** |
|---|---|---|
| Precision | $\dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$ | Fraction of predicted-positive events that are true leads. High precision means few false alarms. |
| Recall | $\dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$ | Fraction of actual leads that are correctly identified. High recall means few missed signals. |
| $F_1$-score | $2 \times \dfrac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$ | Harmonic mean of precision and recall. Balances false alarms vs. missed detections. |
| Support | — | Number of true instances in each class (i.e. actual count of leads vs. no-leads in the test set). |
| Accuracy | $\dfrac{\mathrm{TP}+\mathrm{TN}}{\mathrm{Total}}$ | Overall correct classification rate. Can be misleading if classes are imbalanced. |

Table 5: Key evaluation metrics for the lead–lag classification problem.

## Conclusion

- Price changes are predominantly noise; a spot-market move rarely induces a perp-market response.
- Genuine lead–lag events are so infrequent that our supervised learning faces a severe class imbalance.
- The limited feature set (returns, spread, volume, count, filtered velocities) lacks the richness needed to separate real spill-overs from noise.
- Unfortunately, even though a supervised model(XGBoost) can recognize some patterns in short term, it is risky to rely solely on it for noise detection. Alternative approaches—such as Kalman filters, autoregressive models (AR/ARIMA), and LSTMs—should also be explored.