# Project 05 — Strategy Backtesting

Momentum vs Mean Reversion (with transaction costs)

Quant Finance Portfolio (Karim)

December 19, 2025

**Abstract**

This report supports **Project 05**, a compact but realistic backtesting framework for two canonical signals: **time-series momentum** and **z-score mean reversion**. It focuses on implementation quality (no look-ahead), transaction cost modeling via turnover, and recruiter-relevant outputs (equity curve, drawdowns, Sharpe, cost drag, sensitivity). The goal is not to "prove" a universal edge, but to demonstrate a clean research workflow and risk-aware evaluation.

## Contents

# 1 What you build in Project 05

In this project you implement and compare two classic trading signals on a single liquid asset (and the design generalizes to portfolios):

- **Signal engineering**
  - **Momentum:** sign of past return over a lookback window (trend-following / TSMOM).
  - **Mean reversion:** z-score of price relative to a moving average (contrarian).
- **Position generation:** long-only or long/short with explicit state and no-lookahead alignment.
- **Backtest engine:** log-returns → strategy returns, with transaction costs modeled from turnover.
- **Evaluation:** annualized return/volatility, Sharpe ratio, max drawdown, turnover, and total cost impact.
- **Sensitivity analysis:** interactive controls (lookback / z-thresholds / cost bps) to test robustness.

**Recruiter takeaway.** The main value of this project is the *research discipline*: clear definitions, clean data alignment, cost-aware performance, and reproducible metrics.

# 2 Prerequisites (math and modeling you must know)

## 2.1 Returns, log-returns, and compounding

Given prices $(P_t)$, the log-return is

$$r_t = \log\left(\frac{P_t}{P_{t-1}}\right).$$

If strategy returns are $(R_t)$ in decimal form, the equity curve is

$$\text{Equity}_t = \prod_{s\le t}(1+R_s).$$

Log-returns are convenient because they add over time; equity uses standard compounding.

## 2.2 Backtesting as a time-indexed cashflow

A backtest is a deterministic transformation of historical information into positions and then into P&L:

$$\text{prices} \Rightarrow \text{signal}_t \Rightarrow \text{position}_t \Rightarrow \text{P\&L}_t.$$

The critical constraint is **no look-ahead**: position decisions at time $t$ may only depend on data up to time $t$.

## 2.3 Risk-adjusted performance (Sharpe) and drawdown

A common summary metric is the annualized Sharpe ratio:

$$\text{Sharpe} = \frac{\mathbb{E}[R_t]\cdot K}{\text{Std}(R_t)\sqrt{K}},$$

where $K$ is periods per year (e.g. 252 for daily data).

Max drawdown is computed from the equity curve:

$$\text{MDD} = \min_t\left(\frac{\text{Equity}_t}{\max_{s\le t}\text{Equity}_s} - 1\right).$$

### 2.4 Transaction costs via turnover

A simple and transparent cost model is proportional to traded notional. If positions are $(\Delta_t)$, turnover is approximately $|\Delta_t - \Delta_{t-1}|$. With a cost of $c$ (in decimal),

$$R_t^{\text{net}} = R_t^{\text{gross}} - c \cdot |\Delta_t - \Delta_{t-1}|.$$

This is the **minimum** you need to prevent unrealistic high-frequency switching from looking profitable.

## 3 Data, conventions, and reproducibility

### 3.1 Asset and data source

The notebook downloads adjusted prices (e.g. via `yfinance`) and builds a clean price series. A robust fallback (synthetic data) is included to keep the notebook runnable even without network access.

### 3.2 Conventions

- Returns are computed from prices using log-returns.
- Positions are applied with a **one-step lag** (position at $t$ uses information up to $t-1$).
- Transaction costs are expressed in **basis points (bps)** and applied from turnover.

### 3.3 Minimal folder structure (recommended)

- `notebooks/Project05_backtest.ipynb`
- `reports/Project05.pdf` (this file)
- `assets/` for exported interactive plots (Plotly HTML)

## 4 Signal 1: Time-series momentum (trend-following)

### 4.1 Definition

Momentum uses the sign of the past return over a lookback window $L$:

$$m_t = \frac{P_t}{P_{t-L}} - 1.$$

A simple trading rule is

$$\Delta_t = \operatorname{sign}(m_t),$$

with optional **long-only** restriction $\Delta_t \leftarrow \max(\Delta_t, 0)$.

### 4.2 Interpretation

Time-series momentum is a stylized trend-following rule: it tends to stay invested when trends persist, and it flips in regimes where medium-term reversals dominate.

## 5 Signal 2: Mean reversion via z-score

### 5.1 Definition

Compute a moving average and rolling volatility over a window $W$:

$$\mathrm{MA}_t = \frac{1}{W}\sum_{k=0}^{W-1} P_{t-k}, \qquad s_t = \sqrt{\frac{1}{W-1}\sum_{k=0}^{W-1}(P_{t-k} - \mathrm{MA}_t)^2}.$$

The z-score is

$$z_t = \frac{P_t - \mathrm{MA}_t}{s_t}.$$

A common state-machine rule is:

- enter short if $z_t > z_{\text{entry}}$, enter long if $z_t < -z_{\text{entry}}$,
- exit to flat when $|z_t| < z_{\text{exit}}$,

with an optional long-only restriction.

### 5.2 Interpretation

Mean reversion assumes deviations from a local reference level tend to correct. The entry/exit band prevents constant flipping around the mean.

## 6 Backtest engine (no look-ahead, with costs)

### 6.1 Alignment and no look-ahead

The backtest applies a lagged position:

$$R_t^{\text{gross}} = \Delta_{t-1}\, r_t,$$

where $r_t$ is the asset log-return. This ensures the signal computed at time $t-1$ is the one traded over $(t-1, t]$.

### 6.2 Turnover and cost model

Turnover is

$$\text{Turnover}_t = |\Delta_t - \Delta_{t-1}|.$$

With costs of `cost_bps` basis points per unit turnover,

$$R_t^{\text{net}} = R_t^{\text{gross}} - \frac{\texttt{cost\_bps}}{10^4}\,\text{Turnover}_t.$$

The equity curve is computed from net returns.

## 7 Performance metrics reported

For each strategy, the notebook reports:

- **Total return** and **annualized return**,
- **annualized volatility**,
- **Sharpe ratio** (risk-adjusted return),
- **maximum drawdown** (worst peak-to-trough decline),
- **average turnover** (trading intensity),
- **total transaction costs** (cost drag over the backtest).

**Reading the results.** A high Sharpe with extreme turnover often disappears after costs; a strategy that remains reasonable under costs and parameter changes is more credible.

## 8 Robustness checks and common pitfalls

### 8.1 Sensitivity analysis (what recruiters look for)

A professional backtest is not one "best" parameter set; it is evidence the behavior is stable:

- Sweep lookback $L$ (momentum) and $(W, z_{\text{entry}}, z_{\text{exit}})$ (mean reversion).
- Increase costs (bps) and verify whether performance degrades smoothly or collapses.
- Check the trade-off: higher turnover $\Rightarrow$ higher cost drag.

### 8.2 Pitfalls explicitly controlled or discussed

- **Look-ahead bias:** avoided via position lagging.
- **Overfitting/data snooping:** mitigated by reporting sensitivity (not a single tuned point).
- **Regime dependence:** trend vs mean-reversion can dominate in different periods.
- **Missing frictions:** spreads, slippage, and market impact are not fully modeled; costs here are a transparent proxy.

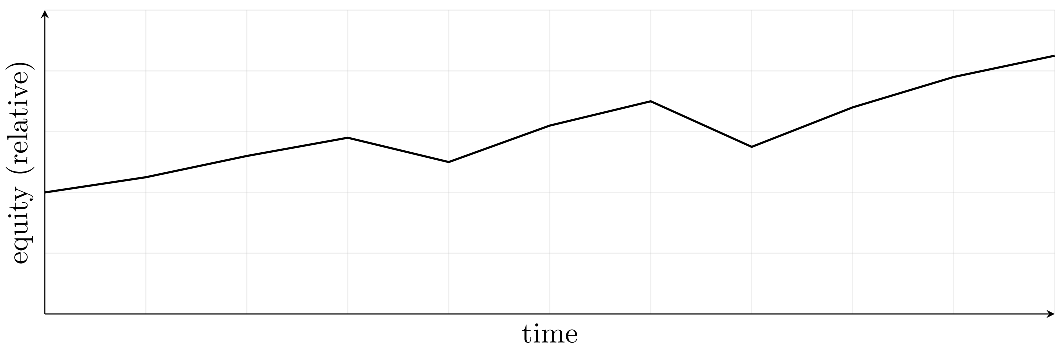
## 9 Sanity checks you should always do

- Verify that strategy returns change when you shift the signal by one day (to detect accidental look-ahead).
- Set costs to zero and confirm net returns match gross returns exactly.
- Force $\Delta_t \equiv 0$ and verify P&L $\equiv 0$.
- Raise costs significantly and confirm performance decreases (often sharply for high-turnover rules).
- Compare long-only vs long/short behavior and confirm exposures match the design.

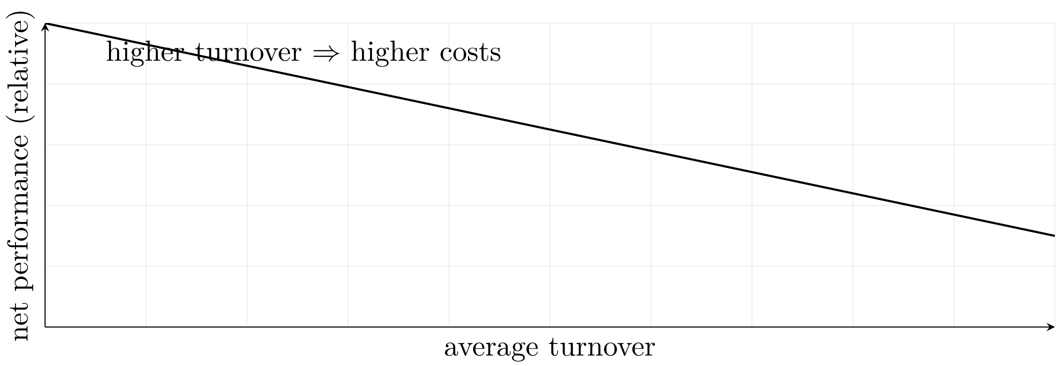
## 10 Overleaf plots (generated directly here)

These plots are conceptual (fast to compile) and help a reader understand the evaluation logic.

### 10.1 Equity curve and drawdown (illustration)




### 10.2 Cost drag vs turnover (illustration)



## 11 Interview pitch

I built a clean backtesting framework and implemented two canonical signals: time-series momentum and z-score mean reversion. I enforced no-lookahead by lagging positions, modeled transaction costs from turnover, and reported professional metrics like Sharpe, max drawdown, and cost drag. I also added parameter and cost sensitivity controls to check robustness instead of relying on one tuned configuration.

I treat this as a research-grade baseline: costs are modeled transparently, but real execution would also add spreads, slippage, and impact. The framework is designed to plug those in.