Kalman Filter Applications in Quantitative Finance (WIDS)

Midterm Progress Report

Shaurya Gupta

January 6, 2026

Abstract

This report summarizes the progress made in designing and implementing Kalman Filters for algorithmic trading. The project focuses on deriving the mathematical foundations of optimal estimation and applying them to financial time-series analysis. Key achievements include the development of a reusable Python implementation, the application of state-space models for trend detection, and the estimation of dynamic hedge ratios for pairs trading.

1 Objective

The primary objective of this project is to move beyond standard technical indicators (such as Moving Averages) and utilize Control Theory to separate "signal" (true asset value or trend) from "noise" (market microstructure volatility). The project aims to build a robust "Trading Toolbox" capable of adaptive parameter estimation in real-time.

2 Theoretical Framework

2.1 The Estimation Problem

Financial data is inherently noisy. We model the true state of an asset as a hidden variable that cannot be observed directly but must be inferred from noisy measurements. The Kalman Filter addresses this by recursively estimating the state using a "Predict-Update" loop:

- **Predict (Time Update):** Projects the current state estimate forward using a physical model (process equation).
- **Update (Measurement Update):** Corrects the prediction based on new incoming data, weighted by the *Kalman Gain* (K).

2.2 Matrix Formulation

The system is defined by the following state-space matrices, which have been successfully implemented and tested:

- **State Vector (x):** Represents the hidden variables (e.g., Price and Velocity/Trend).
- **Covariance Matrix (P):** Represents the uncertainty of the estimate.
- **Transition Matrix (F):** Defines how the state evolves over time (the "physics" of the price).
- **Noise Matrices (Q & R):** Q models process noise (system variance), while R models measurement noise (sensor/market variance).

3 Implementation & Experimentation

3.1 Python Architecture

A robust, object-oriented implementation was developed using the `numpy` library. The architecture separates the mathematical core (the `KalmanFilter` class) from the financial logic, allowing for reusable application across different strategies.

3.2 Application I: Trend Following (Kinematic Model)

We modeled stock price movement as a kinematic system where Price is equivalent to "Position" and Trend is equivalent to "Velocity."

- **Methodology:** The filter was configured to infer the hidden "Velocity" state from raw price data (e.g., Apple Inc.).
- **Result:** The filter successfully generated buy/sell signals based on the polarity of the estimated velocity. This approach demonstrated lower lag compared to traditional Moving Averages, as it reacts to changes in the instantaneous slope rather than a historical average.

3.3 Application II: Pairs Trading (Dynamic Regression)

We extended the filter to solve a regression problem for Pairs Trading (e.g., Asset A vs. Asset B).

- **Methodology:** The regression coefficients (α and β) were treated as state variables. The observation matrix (H) was made dynamic, updating at every time step with the price of the independent asset.

- **Result:** The system successfully tracked a "Regime Change" in a synthetic dataset. Unlike static Ordinary Least Squares (OLS) regression, the Kalman Filter adapted the Hedge Ratio (β) in real-time, proving its utility for mean-reversion strategies in changing market conditions.

4 Key Findings

1. **Lag vs. Sensitivity Trade-off:** The performance of the filter is highly sensitive to the ratio of Q to R. A high Q/R ratio results in a reactive but jittery filter, while a low ratio results in a smooth but lagging filter.

2. **Hidden State Extraction:** The primary value driver is the extraction of unobservable states (Trend Velocity, Dynamic Beta) rather than simple price smoothing.

3. **Numerical Stability:** Implementation of the covariance update equation required careful handling to maintain symmetry and positive definiteness.