Dynamic Pairs Trading with Kalman Filters

A Statistical Arbitrage analysis

Mohammed El Azhar
GitHub · LinkedIn

June 2025

Abstract

Statistical arbitrage strategies seek to exploit temporary mispricings between assets with stable long–run relationships. Pairs trading is a classic example in which two highly correlated and cointegrated instruments are traded on their spread. This report analyses a pairs–trading project that implements both a static hedge ratio estimated via ordinary least squares and a dynamic hedge ratio estimated via a Kalman filter. We provide the mathematical background, derive the prediction and update equations for the Kalman filter, describe how these estimates feed into a trading strategy, and discuss the practical implementation in a Streamlit dashboard. Additionally we develop a streamlit based UI ready for deployment.

Contents

Chapter 1

Introduction

Statistical arbitrage targets relative mispricings rather than absolute price levels. One of the most widely studied strategies is *pairs trading*, which forms a market–neutral portfolio by going long one asset and short another in proportions designed to hedge the common component of their price movements. To construct such a portfolio one typically selects assets with a strong long–term relationship. A linear regression on historical prices yields a *hedge ratio*; when the hedged spread deviates significantly from its mean the strategy enters positions expecting reversion. Traditional implementations assume the hedge ratio is fixed over time, but in practice the relationship between assets is rarely stationary; this motivates dynamic estimation schemes. A state–space model with a Kalman filter can estimate the time–varying hedge ratio sequentially. Quantitative researchers have noted that the hedge ratio may vary through time and that a rolling regression is insufficient because it introduces an arbitrary lookback window. A state–space approach treats the true hedge ratio as an unobserved latent variable and uses the Kalman filter to update it with each new observation.

The project analysed in this report provides a complete pipeline for pairs trading. It collects price data via the `yfinance` API, computes correlation matrices to identify candidate pairs, filters pairs by cointegration using the augmented Dickey–Fuller (ADF) test, and then backtests static and dynamic strategies. A Streamlit interface allows the user to select tickers, training and testing windows, capital allocation, entry/exit thresholds and whether to optimise thresholds on the training window. We discuss each of these components in detail, introduce the relevant mathematics, and provide code excerpts where appropriate.

Chapter 2

Statistical Arbitrage and Pair Selection

2.1 Correlation and initial screening

Pairs trading begins by screening a universe of securities to identify those whose returns are highly correlated. Given price series P_t^a and P_t^b for assets a and b, one computes the Pearson correlation coefficient of their daily returns, $\rho(a, b) = \mathrm{corr}(\Delta \log P^a, \Delta \log P^b)$. In the implementation the user specifies a list of tickers and a training window $[t_0, t_1]$. Adjusted closing prices are fetched and converted to daily returns; the upper triangular part of the correlation matrix is flattened and sorted. Pairs with correlation above a minimum threshold (default 0.5) are retained. This step provides a computationally cheap first filter but correlation alone does not guarantee a stable long–run relationship.

2.2 Cointegration and the ADF test

Two price series may be highly correlated yet not mean reverting; for a pairs trade to be profitable the spread must be stationary. Cointegration theory states that if P^a and P^b are each integrated of order one but a linear combination $P^a - \beta P^b - \alpha$ is stationary, then they are cointegrated. The hedge ratio (β, α) can be estimated by ordinary least squares (OLS). After estimating β and α on the training window we form the spread $S_t = P_t^a - \beta P_t^b - \alpha$ and test S_t for a unit root. The augmented Dickey–Fuller (ADF) test examines the null hypothesis of a unit root against the alternative of stationarity. In pairs trading the ADF test is used to ensure that the series in each candidate pair are cointegrated; the spread must be stationary for the mean–reversion assumption to hold. The ADF test measures whether the pair is cointegrated, and only if the null hypothesis of a unit root is rejected can the pair be traded.

The project applies the ADF test to the residuals of each candidate pair and discards pairs whose p–values exceed a user–defined threshold (default 0.05). Only those pairs whose spreads are stationary proceed to the next stage. The OLS hedge ratios (β, α) computed on the training window serve as the static hedge ratio in the backtest and as initial values for the dynamic Kalman filter.

Chapter 3

Mathematical Background

3.1 Ordinary least squares and static hedge ratio

To estimate the static hedge ratio, we regress the dependent asset P^a on the independent asset P^b using ordinary least squares. Let $y_t = P_t^a$ and $x_t = P_t^b$. The OLS model posits

$$y_t = \beta x_t + \alpha + \varepsilon_t, \tag{3.1}$$

where β is the slope (hedge ratio), α is the intercept and ε_t are i.i.d. residuals. Stacking observations in a design matrix $X = [x\ 1]$ and vector y yields the closed–form solution $(\hat{\beta}, \hat{\alpha}) = (X^\top X)^{-1} X^\top y$. When a static strategy is used, these parameters remain fixed throughout the test period. A static spread $S_t = y_t - \hat{\beta} x_t - \hat{\alpha}$ is computed each day and z–score normalisation on a rolling window produces trading signals.

3.2 Augmented Dickey–Fuller test

The ADF test models the first difference of a time series as a function of its lagged level and lagged differences:

$$\Delta S_t = c + \phi S_{t-1} + \sum_{i=1}^{p} \gamma_i \Delta S_{t-i} + \epsilon_t. \tag{3.2}$$

Under the null hypothesis $\phi = 0$ (unit root) the series is non–stationary; rejection of the null implies stationarity. The test statistic is compared to critical values that depend on the lag order and presence of drift or trend. In the context of pairs trading the ADF test is applied to the residual spread S_t to verify cointegration. If the spread fails to reject the unit–root hypothesis the pair is excluded from trading. For a detailed treatment of the ADF test and cointegration, including finite–sample properties and critical values, see references 4 and 5.

3.3 State–space models and Kalman filter

The Kalman filter provides a recursive solution to the linear state estimation problem for systems with Gaussian noise. A state–space model comprises two equations: State (transition): $\mathrm{x}_k = \Phi_{k-1} x_{k-1} + w_{k-1}, \quad w_{k-1} \sim \mathcal{N}(0, Q_{k-1}),$
$Measurement: \quad y_k = H_k x_k + v_k, \quad v_k \sim \mathcal{N}(0, R_k)$. Here $x_k \in R^n$ is the hidden state vector, $y_k \in R^m$ are observations, Φ_{k-1} is the state transition matrix, Q_{k-1} is the process noise covariance, H_k is the measurement matrix and R_k is the observation noise covariance. The Kalman filter

operates in two phases: *prediction* and *update*. During the prediction step the prior state and covariance are propagated: $\mathrm{x}_k^- = \Phi_{k-1}\, x_{k-1}^+, P_k^- = \Phi_{k-1} P_{k-1}^+ \Phi_{k-1}^\top + Q_{k-1}$, where the superscripts $-$ and $+$ denote prior (predicted) and posterior (updated) estimates, respectively. The process noise term Q allows the state to vary through time; if $Q = 0$ the model reduces to a constant parameter. Derivations of these prediction equations can be found in Kalman's original paper [1] and in modern expositions such as Särkkä [2].

When a new observation y_k arrives, the filter computes the innovation (measurement residual) $\tilde{y}_k = y_k - H_k x_k^-$ and its covariance $S_k = H_k P_k^- H_k^\top + R_k$. The optimal Kalman gain is then

$$K_k = P_k^- H_k^\top (H_k P_k^- H_k^\top + R_k)^{-1}, \tag{3.3}$$

which weights the prediction against the measurement. The posterior state and covariance are updated via $\mathrm{x}_k^+ = x_k^- + K_k(y_k - H_k x_k^-)$,
$P_k^+ = (I - K_k H_k) P_k^-$. These equations ensure that the updated estimate minimises the mean–square error of the state estimate. Once updated, the filter proceeds to the next time step. For derivations of the Kalman gain and update equations, see references 1 and 2.

For a comprehensive introduction to Kalman filtering, including derivations of the prediction and update equations, see references 1 and 2.

3.4 Kalman filter for dynamic hedge ratio

In the pairs–trading project the hidden state consists of the time–varying hedge ratio and intercept, $x_k = [\beta_k, \alpha_k]^\top$. We assume a random walk transition model

$$x_k = x_{k-1} + w_{k-1}, \quad w_{k-1} \sim \mathcal{N}(0, Q), \tag{3.4}$$

where $Q = qI_2$ is a diagonal covariance with small parameter q controlling the smoothness of the state. The observation equation relates the dependent and independent prices through

$$y_k = H_k x_k + v_k, \quad H_k = [x_k \quad 1], \quad v_k \sim \mathcal{N}(0, R). \tag{3.5}$$

Here y_k is the price of asset a at time k, x_k is the price of asset b, and R is the measurement noise variance. At each time step the Kalman filter updates estimates of β_k and α_k based on the latest pair of prices. The project uses a small process noise parameter ($\delta = 10^{-5}$) and initial covariance $P = I$; measurement noise R can be set to unity because it acts as a scaling factor. For further discussion of why the hedge ratio is time varying and how to estimate it sequentially using a Kalman filter, refer to reference **??**.

Chapter 4

Trading Strategy

4.0.1 Spread construction and z–score

Whether using a static or dynamic hedge ratio, the spread is defined as

$$S_k = \frac{y_k - \beta_k x_k - \alpha_k}{1 + \beta_k} \quad (normalised), \tag{4.1}$$

where division by $(1 + \beta_k)$ normalises the spread by the dollar value of the portfolio. The spread series is standardised using a rolling window of length w to compute a z–score:

$$z_k = \frac{S_k - \mu_k}{\sigma_k}, \tag{4.2}$$

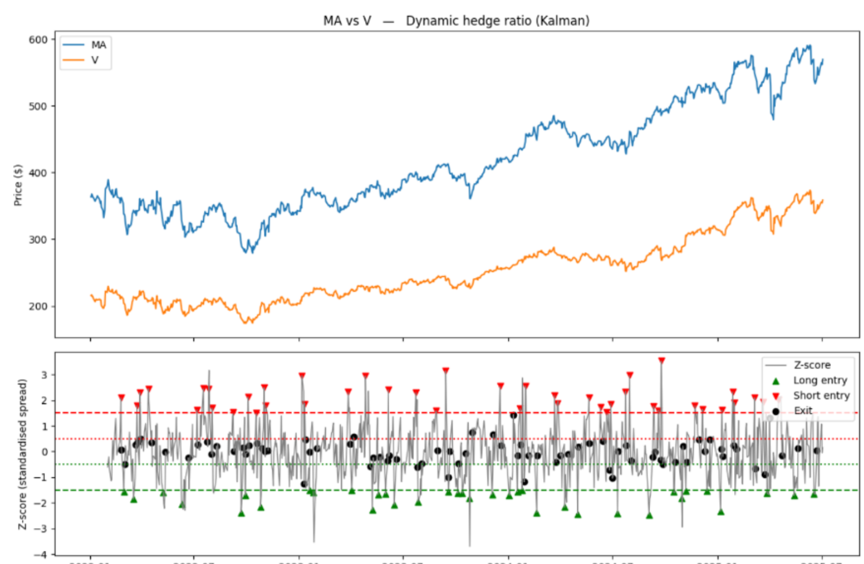

Figure 4.1: Visualisation of the trading strategy applied to MA vs V using a dynamic hedge ratio (Kalman filter). The top panel shows the price evolution of the pair. The bottom panel displays the standardised Z-score of the spread, highlighting long (green triangles) and short (red triangles) entry signals when the Z-score breaches the $\pm z_{\text{entry}}$ thresholds, as well as trade exits (black dots) upon mean reversion.

where μ_k and σ_k are the rolling mean and standard deviation of S_k over the previous w observations. The strategy enters a short position when $z_k > z_{\text{entry}}$ and a long position when $z_k < -z_{\text{entry}}$. Positions are closed when z_k reverts within a threshold $\pm z_{\text{exit}}$ or when a stop–loss threshold is breached. Adaptive volatility scaling optionally multiplies the entry and exit thresholds by the ratio of local to global standard deviation.

4.1 Backtesting and capital allocation

Backtests divide the data into a training window $[t_0, t_1]$ and a test window $[t_1 + 1, t_2]$. Static hedge ratios $(\hat{\beta}, \hat{\alpha})$ are estimated on the training window; dynamic hedge ratios are initialised using these values and updated throughout the test window. For each pair the strategy computes daily profit and loss (P&L)

$$\text{P\&L}_k = p_{k-1} \Delta S_k, \tag{4.3}$$

where $p_{k-1} \in \{-1, 0, 1\}$ is the position on day $k-1$ and $\Delta S_k = S_k - S_{k-1}$ is the change in spread. A cash account tracks the capital deployed and released by trades; the fraction of the total budget allocated per pair is a user parameter. Stop–losses and maximum holding periods impose risk controls.

4.2 Performance metrics

The strategy evaluates each pair and the aggregate portfolio using several metrics: total P&L, volatility (standard deviation of daily P&L), maximum drawdown, and the Sharpe ratio. The Sharpe ratio measures the risk–adjusted return as the mean excess return divided by its standard deviation, annualised by $\sqrt{252}$ for daily data. A higher Sharpe ratio indicates better risk–adjusted performance. In practice the risk–free rate is small relative to intraday variation and may be omitted; the implementation therefore computes. A more complete treatment of the Sharpe ratio and its annualisation can be found in reference 6.

$$\text{Sharpe} = \frac{\overline{r}}{\sigma_r} \sqrt{252}, \tag{4.4}$$

where r are daily returns.

4.3 Threshold optimisation

The project optionally optimises entry and exit thresholds on the training window. Two methods are provided. The *grid search* method evaluates the P&L or Sharpe ratio over a grid of candidate entry thresholds and selects the one maximising the chosen objective; the exit threshold is set as a fraction of the entry threshold. The *quantile method* sets the entry threshold to a chosen quantile of the absolute z–score distribution and the exit threshold to a lower quantile. These optimised thresholds may be applied per pair during the test period.

4.4 Static versus dynamic hedge ratios

Static strategy: With a fixed hedge ratio $(\hat{\beta}, \hat{\alpha})$ estimated on the training window, the spread is computed on the test window and trading signals are generated using the same thresholds. Although simple, this approach assumes the relationship between assets remains constant. In reality economic conditions and idiosyncratic shocks cause the hedge ratio to drift; static strategies may suffer when the true hedge ratio changes.

Dynamic strategy: The Kalman filter updates (β_k, α_k) with each new observation, allowing the spread to adapt to evolving relationships. As noted by quant researchers, the hedge ratio is likely to be time varying and dynamic estimation may improve profitability. The project delays the use of the updated hedge ratio by one time step to avoid look–ahead bias: the posterior estimate at time k is used as the prior for $k+1$ when computing the spread.

4.5 Comparative analysis

Figures 4.2 and 4.3 illustrate the cumulative P&L obtained from backtesting the static and dynamic hedge ratios, respectively. The static strategy (Figure 4.2) experiences a persistent decline and ends the sample with a negative cumulative profit, whereas the dynamic strategy (Figure 4.3) adapts to changing market conditions and delivers a steadily increasing P&L. This contrast underscores the benefit of updating the hedge ratio through a Kalman filter rather than assuming it is constant.

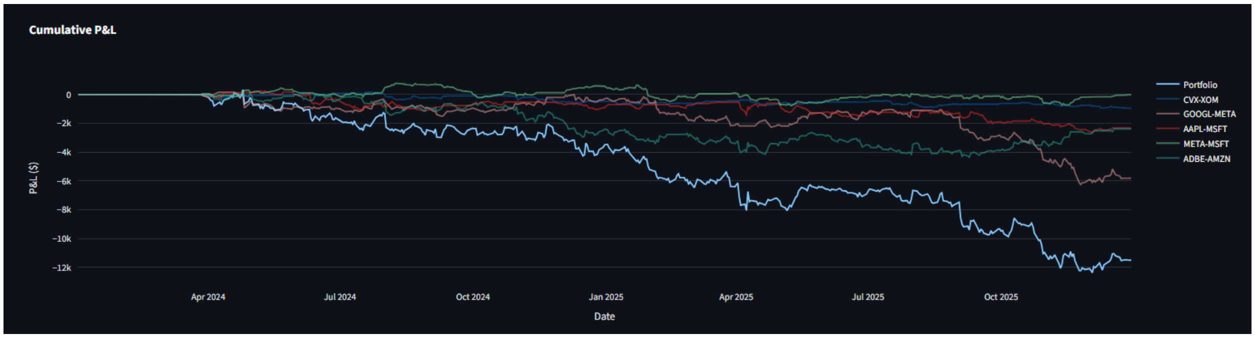

Figure 4.2: Cumulative P&L for static strategy.

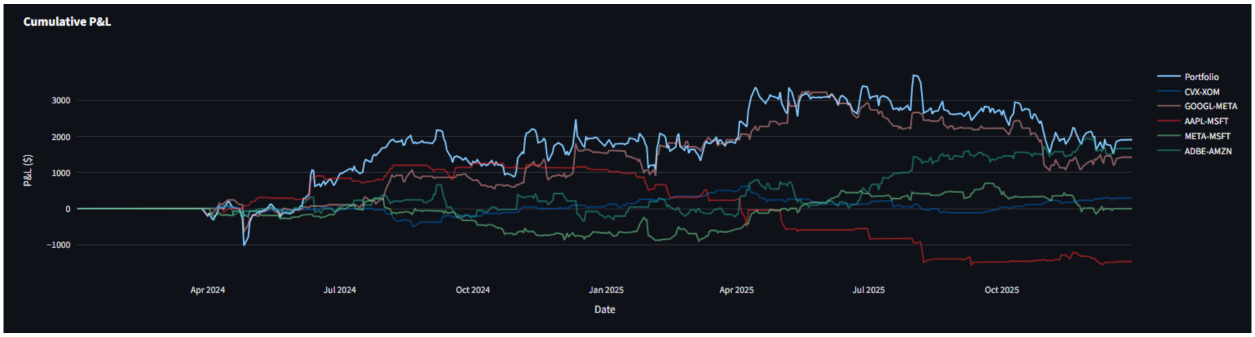

Figure 4.3: Cumulative P&L for dynamic strategy.

Figure 4.4: Comparison of cumulative P&L for static and dynamic strategies.

Chapter 5

Streamlit Interface Development

5.1 User input and data loading

The Streamlit application provides an intuitive dashboard for running the backtests. Users input a comma–separated list of tickers, select training and test windows via calendar widgets, specify total budget and risk per pair, and choose entry/exit thresholds via sliders. Upon clicking the *Run backtest* button the app fetches price data for all tickers across both windows.

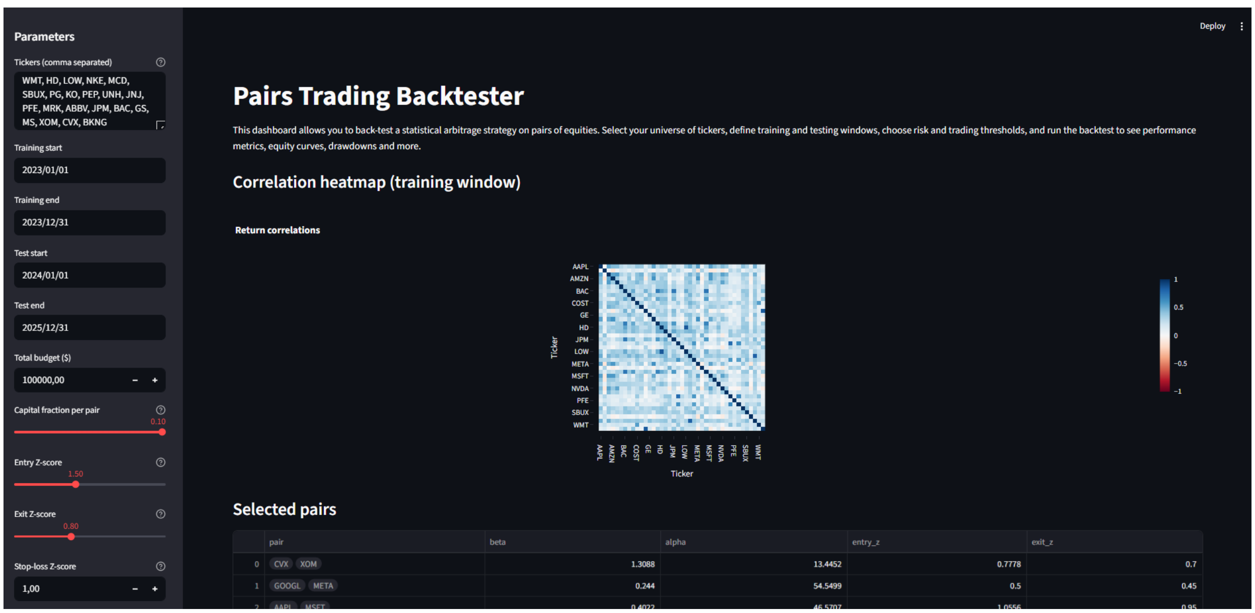

Figure 5.1: The Streamlit interface

5.2 Correlation heatmap and pair selection

Within the training window the dashboard displays a correlation heatmap of returns, using Plotly or seaborn depending on installed packages. Highly correlated pairs are then listed in a table alongside their OLS hedge ratios. Stationary pairs that pass the ADF test are retained. Users may enable threshold optimisation, selecting either grid or quantile methods and, for grid search, the objective to maximise (P&L or Sharpe).

5.3 Backtesting controls and results display

The sidebar allows the user to choose between a *Dynamic (Kalman)* or *Static (OLS)* hedge ratio. Additional risk controls include a stop–loss z–score, maximum holding days and a checkbox for volatility–adaptive thresholds. Once the backtest finishes the app shows the head of the P&L and cash series, a table of basic performance metrics for each pair, and the final cash balance. Extended analytics—such as equity curves, rolling Sharpe ratios, drawdown charts and monthly P&L heatmaps—are generated using the functions in `analytics_utils.py`.

5.4 Trade log and download

When signals are available, the application constructs a trade log capturing entry and exit dates, positions sizes and profits. Users can download this log as a CSV file for further analysis. The trade log also records the capital committed to each trade relative to the total budget, enabling calculation of metrics such as hit rate, average profit per trade and average holding period.

Chapter 6

Limitations and Discussion

While the project provides a solid sandbox for pairs trading research, several limitations must be acknowledged:

- **Transaction costs and market frictions.** The backtests ignore bid–ask spreads, commissions and taxes. In real trading these costs erode profitability and may alter optimal thresholds.
- **Borrowing and funding constraints.** Shorting an asset incurs borrow fees and may be restricted. The code assumes unrestricted shorting and ignores financing costs.
- **Slippage and execution delays.** Trades are assumed to occur at the closing price. Execution at more realistic intraday prices with slippage can materially impact performance.
- **Parameter sensitivity.** The Kalman filter requires tuning of the process noise Q and measurement noise R. Choosing these values incorrectly may lead to overfitting or overly noisy estimates. The implementation uses a simple constant Q and $R = 1$; more sophisticated adaptive schemes could be explored.
- **Nonlinear dynamics.** The state–space model assumes linear relationships and Gaussian noise. Real financial data exhibit heavy tails and structural breaks; an extended or unscented Kalman filter or particle filter could better handle such features.
- **Universe selection bias.** The strategy relies on the user's input tickers. Without a systematic selection process there is a risk of data snooping. A more comprehensive study should consider an unbiased universe, multiple formation and test windows, and out–of–sample validation.
- **Stationarity assumption.** Even after passing the ADF test, the cointegration relationship may break down during the test period. Monitoring stationarity dynamically could avoid trading pairs that diverge.
- **Look–ahead bias.** The dynamic strategy properly lags the Kalman estimates by one time step; however, the static strategy uses the hedge ratio estimated on the entire training window. If the training window selection is not independent of the test window there could be implicit look–ahead bias.

Future work could incorporate transaction cost models, dynamic allocation across multiple pairs, regime detection to switch on and off trading, and more advanced state–space models that allow for stochastic volatility or exogenous factors.

Chapter 7

Conclusion

This report has dissected a pairs–trading project that combines statistical screening, cointegration testing, dynamic state estimation and intuitive visualisation. We derived the Kalman filter equations and explained how they produce a time–varying hedge ratio that adapts to changing market conditions. Trading signals were built on a z–score of the spread, with risk controls and optional threshold optimisation. A Streamlit dashboard wraps the entire workflow into an accessible application. While the dynamic strategy addresses the time variation of hedge ratios and thus may improve over static approaches, careful attention to transaction costs, parameter tuning and validation is essential. The placeholders for empirical results can be filled once the code is run on historical data, enabling quantitative researchers to evaluate performance rigorously.

Bibliography

[1] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1), 35–45.

[2] Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press.

[3] Brown University Computer Science. (1996). *Kalman Filters*. Available at https://cs.brown.edu/stc/education/course95-96/Kalman-Filters/kalman.html.

[4] Engle, R. F., & Granger, C. W. J. (1987). Co-integration and error correction: representation, estimation, and testing. *Econometrica*, 55(2), 251–276.

[5] Said, S. E., & Dickey, D. A. (1984). Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3), 599–607.

[6] Sharpe, W. F. (1966). Mutual fund performance. *Journal of Business*, 39(1), Part II, 119–138.