# A Time Series Approach to Pairs Trading on the Indian Stock Market

Prepared by: Aryan Gupta, Billa Cherish, Vibha Gupta

Implementation available on Github

## 1 Introduction

Predicting stock prices is quite a difficult task, but also very rewarding, and hence hedge funds and trading firms do spend a lot of their resources in this area of research. In this project, we would be trying to develop a **profitable** time series based strategy in predicting stock prices.

The efficient-market hypothesis (EMH)[3] is a hypothesis in financial economics that states that asset prices reflect all available information, essentially meaning that the price movements are nothing but a **random walk**. A direct implication is that it is impossible to "beat the market" consistently on a risk-adjusted basis since market prices should only react to new information. Our own analysis does show that applying classical time series models such as an auto-regressive integrated moving-average (ARIMA) model to the individual stock prices does not help forecast the prices, as they would only fit an ARIMA(0,1,0), confirming the hypothesis that the prices do follow a random walk. This finding demonstrates that a trading strategy based on a univariate Time Series forecasting approach is unlikely to be profitable, which is our main goal after all.

Hence, we pivoted towards a pairs trading strategy, which is a bivariate approach involving 2 stock price series which do show a high historical correlation. We then leverage this co-movement by developing a trading strategy that bets on the fact that, **any deviation in their co-movement should soon revert back** as their prices have historically moved together. Our following study shows that pairs trading is indeed an effective and a profitable strategy, while remaining market neutral (i.e. the portfolio returns aren't correlated to the market returns). The main catch is that, it is **synonymous to statistical arbitrage** (i.e. a High Frequency Trading Strategy), therefore 99% of these opportunities would vanish as soon as they appear. It is extremely sensitive to reflexivity and thereby a race to the bottom in risk-adjusted return, unless you find a pair that no one else has. This strategy was first explored by Gatev, Goetzmann, and Rouwenhorst (2006)[4].

However, simply finding 2 highly correlated stocks is not enough. According to Robert F. Engle, C.W.J Granger (1987)[5], even 2 independent random walks can appear highly correlated, leading to strategies that would break as soon as this mistaken relationship falls apart. Hence, we build on a more stable foundation called **cointegration**, as discussed in the above research paper.

**A brief preview on the approaches used:**

- We first begin by choosing a dataset, which contains stocks that are volatile enough to develop a profitable trading strategy and also from the same sector to ensure that pairs of them do show a strong co-movement, which is the foundation upon which we are building this project.
- We test whether the chosen pair do show a strong enough co-movement by doing a cointegration test.
- We then try to develop a trading strategy just based on their individual price series forecasting, as a benchmark to compare our pairs trading strategies with.
- Our first attempt on pairs trading would be a simple regression approach, in finding the spread between the 2 chosen stock prices.
- We build on this and fit a dynamic hedge ratio using kalman filters.
- We then move to a multivariate time series approach by fitting a vector error-correction model (VECM), which essentially combines all the above steps together in a more robust way.

**This paper is further organized as follows:**

- Section 2, Data description.
- Section 3, Problem formulation.
- Section 4, Methodology.
- Section 5, Results.
- Section 6, Final conclusions.
- Section 7, Future direction.
- Section 8, Shortcomings.

## 2 Data Description

The stocks are chosen such that the focus is on highly liquid, same-sector components from the National Stock Exchange (NSE). We selected the 13 stocks from **NIFTY AUTO**. The reasoning for this is that stocks within the same sector are subject to similar economic shocks and industry-specific factors, making them strong candidates for cointegrating relationships.

Our implementation utilizes historical daily adjusted closing prices taken from the `yfinance` library in Python. The data spans from June 1, 2017, to July 1, 2023, providing a dataset of over eight years or *1504 days*. We would be using a **80 : 20 split** for the training and testing of the data.

*Training Set:* June 1, 2017 to April 12, 2022 (*1203 days*)
*Testing Set:* April 13, 2022 to July 1, 2023 (*301 days*)

### Data Analysis

The raw price data is first cleaned and aligned. Fortunately, the `yfinance` library does not have any missing or null and creates a consistent daily time series for all stocks in our universe.

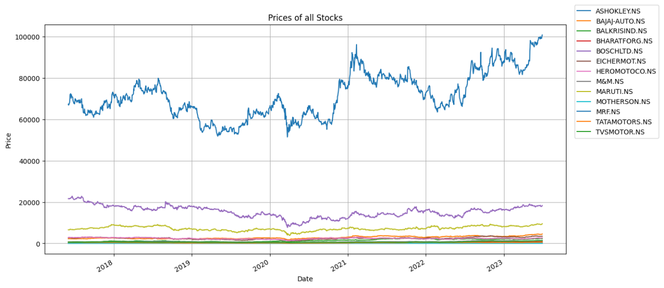

Figure 1: Raw Adjusted Closing Prices of Selected NSE Stocks.

### Logarithmic Transformation

Following standard practice in financial econo-metrics, we apply a natural logarithm transformation to the adjusted closing prices: $Y_t = \log(P_t)$. This transformation is critical for two reasons:

- It converts the exponential growth of stock prices into a linear trend, which is a core assumption for the linear regression models used later.

- It **stabilizes the variance** of the price series. Raw stock prices tend to become more volatile as their price level increases, whereas log prices generally have a more constant variance.

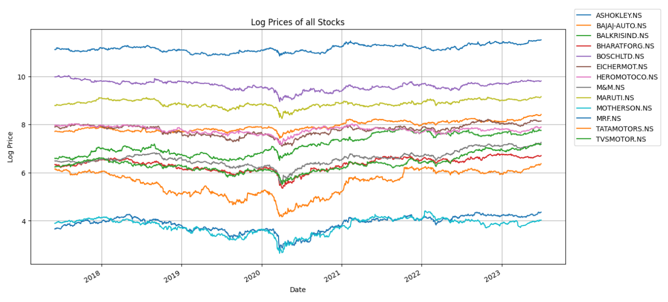

Figure 2: Log-Transformed Prices of Selected NSE Stocks.

### Stationarity Analysis

To verify the stationarity of our data, we perform the **Augmented Dickey-Fuller (ADF) test**[9] on all individual log-price series. It is a statistical test for a unit root that checks whether a time series is non-stationary (has a unit root) or stationary, with rejection of the null indicating stationarity. (We will talk about the methodology later in the report.) For all stocks tested, the ADF test did not reject the null hypothesis (p-value $> 0.05$), confirming that the log-price series $Y_t$ are non-stationary.

Conversely, when the ADF test was applied to the first difference of the log-prices (the log-returns, $W_t = \Delta Y_t$), the ADF test rejected the null hypothesis (p-value $\approx 0.0$), confirming the log-returns are stationary. This clearly shows that the log-price series $Y_t$ **are of the order I(1)**.

### Correlation Analysis

As shown in the correlation heatmap (Figure 3), many pairs in our dataset show very high correlation (e.g., >0.75). Note that this high correlation is calculated on the non-stationary log-prices. **The high value is likely because both stocks are trending together over time**, not necessarily because they share a stable economic bond.

This is precisely why we must use **cointegration**. Trading on a deceptive correlation is extremely risky, as the series have no statistical reason to revert. The Engle-Granger test[4] essentially confirms whether there is a true, mean-reverting relationship (cointegration).

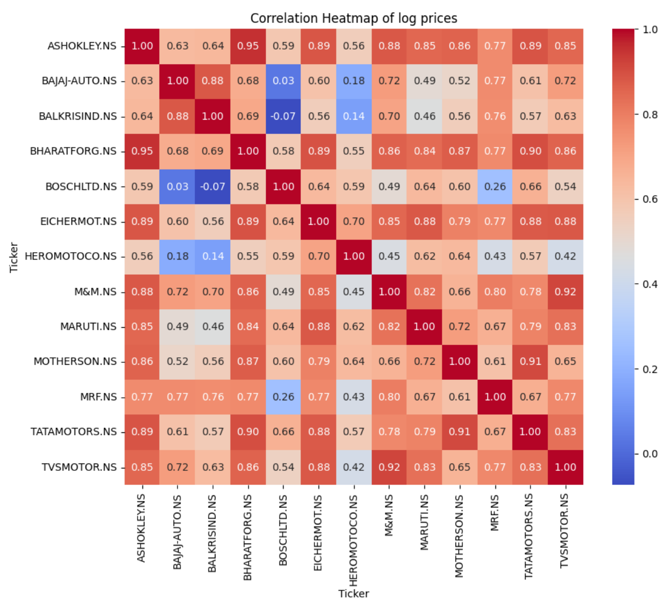

Figure 3: Correlation Heatmap of Log-Prices

## 3 Problem Formulation

Our main goal is to determine whether pairs trading can be a profitable trading strategy. We will be comparing 4 different strategies:

- Univariate forecasting of individual stock prices.
- Exploiting the co-movement using a static hedge ratio.
- Using Kalman filters to get a dynamic hedge ratio.
- Vector Error-Correction model (VECM).

**Research Questions we would like to address:**

1. Which NIFTY AUTO stock pairs, exhibit a strong, statistically significant long-term cointegration?
2. Does a dynamic hedge ratio (estimated via Kalman Filter) yield superior performance (e.g., higher Sharpe Ratio) compared to a static hedge ratio?
3. Does defining the spread using the cointegrating vector from a VECM lead to different performance compared to the OLS or Kalman Filter approaches when using the same Z-score signals?
4. How do these pairs trading strategies compare against a univariate ARIMA benchmark (forecasting each stock independently)?

### Choice of Technology: Python

Python is selected for its robust, integrated, and industry-standard ecosystem, allowing an automated and reproducible workflow from data acquisition to backtesting. Its advantages are clear:

- **Ecosystem:** Access to powerful, specialized libraries is essential. We will utilize `pandas` and `numpy` for data manipulation, `statsmodels` for econometric modeling (ADF, OLS, ARIMA, VECM), `pykalman` for dynamic state-space estimation, and `matplotlib`/`seaborn` for visualization.
- **Reproducibility & Automation:** Python scripts allow us to easily run walk-forward backtests, perform parameter sweeps, and ensure our results are verifiable and not the product of one-off data mining.
- **Industry Relevance:** Python is the dominant language in quantitative finance, making the implementation skills directly transferable.

## 4 Proposed Methodology

### Phase 1: Identifying Potential Pairs

In the first phase, we focus on finding pairs of stocks whose prices move together in the long run—a property called cointegration. We will specifically analyze a set of highly liquid stocks from the same economic sector (e.g., Banking or Automobile), obtained using the yfinance library in Python. Since these stocks belong to the same sector and are highly liquid, they are expected to exhibit strong co-movement, allowing us to proceed directly to formal cointegration tests to identify statistically significant relationships. The motivation is that even if two stocks drift apart over short periods, a confirmed cointegrating relationship can make the spread between them mean-reverting, which is useful for pairs trading.

### Engle-Granger Cointegration Test

**Step 1: Modelling Cointegration using Least Squares**

For two stocks, we work with their log price series: $Y_t^{(x)}$ for stock $x$, and $Y_t^{(y)}$ for stock $y$. This logarithmic transformation helps stabilize variance and makes returns additive, which is useful in statistical models. To check for a long-run equilibrium relationship, we establish a model:

$$Y_t^{(y)} = \alpha + \eta Y_t^{(x)} + \varepsilon_t$$

where:

- $\alpha$ is the intercept, capturing any consistent difference in log prices,
- $\eta$ is the hedge ratio, representing how much of one stock offsets changes in the other,
- $\varepsilon_t$ is the residual/spread, showing the instantaneous deviation from the equilibrium.

The coefficients $\hat{\alpha}$ and $\hat{\eta}$ are estimated by minimizing the sum of the squared residuals ($\sum \varepsilon_t^2$).

The key output from this regression is the residual series:

$$\hat{\varepsilon}_t = Y_t^{(y)} - (\hat{\alpha} + \hat{\eta} Y_t^{(x)})$$

This spread tells us at each time point how far the prices are from their long-run relationship.

**Step 2: Testing Residuals for Stationarity**
To be useful for pairs trading, the spread series $\hat{\varepsilon}_t$, should be stationary, meaning it reverts to its mean over time. We check this using the Augmented Dickey-Fuller (ADF) test, which is a statistical test designed to identify stationarity in time series.

The ADF test model is:

$$\Delta\hat{\varepsilon}_t = \gamma\hat{\varepsilon}_{t-1} + \sum_{j=1}^{p} \psi_j \Delta\hat{\varepsilon}_{t-j} + u_t$$

where:
- $\Delta\hat{\varepsilon}_t$ is the change in spread from one time period to the next,
- $\gamma$ describes the tendency of the spread to revert back to the mean,
- $\psi_j$ represents the dependence on previous changes in the spread,
- $u_t$ is random noise.

If the test rejects the null hypothesis that $\gamma = 0$ (i.e., the spread contains a unit root), then the spread is stationary. This confirms a cointegrated pair suitable for mean-reversion strategies.

## Phase 2: Univariate Benchmark Modelling the Price Series as an ARIMA

In this phase, we focus on modeling each individual stock time series independently, without considering their interrelationships. The objective is to establish a baseline performance using classical time series methods, specifically an ARIMA($p$,$d$,$q$) model. These approaches are standard for forecasting temporal data which may have trend, autocorrelation, and possibly seasonality. We first apply a log transformation to the price series, $Y_t = \log(P_t)$. The ARIMA model then represents the stationary series $W_t = (1-B)^d Y_t$ (after differencing $d$ times):

$$\phi(B)W_t = \theta(B)\varepsilon_t$$

Here, $\phi(B)$ and $\theta(B)$ are polynomials in the lag operator $B$ (where $BY_t = Y_{t-1}$), representing the AR and MA components respectively, and $\varepsilon_t$ is a white noise error term.

**Step 1: Automated Model Selection (auto_arima)**
We will use an automated algorithm like `auto_arima` (from the `pmdarima` library) to systematically determine the optimal model orders $(p, d, q)$. This function automates the following statistical procedures:

- **Determining $d$:** It runs sequential unit root tests (e.g., ADF, testing $H_0 : \gamma = 0$ in $\Delta W_t = c + \gamma W_{t-1} + \dots$) to find the minimum differencing $d$ required to achieve stationarity.
- **Finding $p, q$:** It conducts a stepwise search over a range of $p$ and $q$ values. For each combination, it estimates the parameters using Maximum Likelihood Estimation (MLE), which finds the parameters $(\hat{\phi}, \hat{\theta})$ that maximize the log-likelihood function, $\ln(L(\hat{\phi}, \hat{\theta}|W_t))$.
- **Model Selection:** It compares the fitted models using an information criterion, typically the Akaike Information Criterion (AIC):

$$AIC = -2\ln(\hat{L}) + 2k$$

where $k$ is the total number of parameters $(p + q + 1)$ and $\hat{L}$ is the maximized log-likelihood. The model with the lowest AIC score is selected as the best.

**Step 2: Forecasting and Trading Strategy**
Using the best-fit ARIMA model from Step 1, we generate one-step-ahead forecasts ($\hat{Y}_{t+1}$) in a rolling walk-forward fashion. These log-price forecasts are converted back to price units: $\hat{P}_{t+1} = \exp(\hat{Y}_{t+1})$.

- **Entry Signal (Long):** If $\hat{P}_{t+1} > P_t \times (1 + k)$ (forecast predicts significant rise) and no position is held, enter a long position (Buy).
- **Entry Signal (Short):** If $\hat{P}_{t+1} < P_t \times (1 - k)$ (forecast predicts significant fall) and no position is held, enter a short position (Sell Short).
- **Exit Signal (from Long):** If a long position is held AND $\hat{P}_{t+1} < P_t \times (1 - k)$ (forecast signal reverses to 'Sell'), exit the long position (Sell).
- **Exit Signal (from Short):** If a short position is held AND $\hat{P}_{t+1} > P_t \times (1 + k)$ (forecast signal reverses to 'Buy'), exit the short position (Buy to Cover).

Here, $P_t$ is the current price and $k$ is a small threshold to cover transaction costs and filter noise.

**Purpose of This Phase:** This process establishes a baseline: what kind of forecast accuracy we can achieve without considering the relationships between different stocks. It serves as a reference to compare with pairs trading methods, and helps validate our implementation and modelling choices as we proceed to more complex approaches.

## Phase 3: Static Hedge Ratio Estimation

This phase acts as a baseline approach to pairs trading by assuming a constant relationship between two stocks, rather than allowing it to change dynamically. The key is to estimate a fixed hedge ratio using simple linear regression and generate trading signals from the spread.

**Step 1: Calculating the Hedge Ratio**
We model the log price series of two stocks as a linear relationship:

$$Y_t^{(y)} = \hat{\alpha} + \hat{\eta} Y_t^{(x)} + \hat{\varepsilon}_t$$

which is essentially same as what we have done previously for cointegration test.

The spread, which we use for trading, is simply the deviation from the predicted value:

$$\text{Spread}_t = Y_t^{(y)} - (\hat{\alpha} + \hat{\eta} Y_t^{(x)})$$

**Step 2: Z-Score Normalization**
To make the spread comparable across time and assets, we standardize it into a z-score, which measures how far the current spread is from its rolling average, in units of standard deviation:

$$Z_t = \frac{\text{Spread}_t - \mu_T(\text{Spread})}{\sigma_T(\text{Spread})}$$

where $\mu_T$ and $\sigma_T$ are the rolling mean and standard deviation calculated over a look-back window of length $T$.

This z-score allows us to set simple entry and exit rules that are robust against scale changes and market shifts.

**Step 3: Trading Signal Logic**
Trading rules are based on thresholds. For example:
- **Enter Long:** If $Z_t < -k$ (spread is unusually low): Buy Stock Y and Short Sell Stock X (scaled by $\eta$). Bet on the spread increasing.
- **Enter Short:** If $Z_t > +k$ (spread is unusually high): Short Sell Stock Y and Buy Stock X (scaled by $\eta$). Bet on the

spread decreasing.
- **Exit:** If a position is held AND $|Z_t| < k_{exit}$ (spread has reverted to mean): Exit the position as mentioned before.

**Purpose and Limitations:** This static approach gives us the simplest possible implementation of a mean-reversion strategy. While effective in stable environments, it may miss opportunities or fail in changing markets where relationships between stocks evolve over time.

## Phase 4: Dynamic Hedge Ratio Estimation Kalman Filters

This phase introduces the Kalman Filter to estimate the hedge ratio dynamically, allowing it to change over time as market conditions evolve. Unlike a static value from ordinary regression, the Kalman Filter continuously updates the hedge ratio as new data comes in. The motivation is that the relationship between two stocks (the hedge ratio) may not remain constant due to shifts in economic regimes, company fundamentals, or other factors.

### Step 1: State-space Representation

We define a vector representing the hidden "state" of our system:

$$\beta_t = \begin{bmatrix} \alpha_t \\ \eta_t \end{bmatrix}$$

where $\alpha_t$ is the intercept and $\eta_t$ is the time-varying hedge ratio at time $t$. The observed log price of stock $y$ at time $t$ is modeled as:

$$Y_t^{(y)} = H_t \beta_t + v_t$$

Here, $H_t = [1 \quad Y_t^{(x)}]$ is a vector incorporating a constant (1) and the current log price of stock $x$. $v_t$ is random measurement noise. The state itself evolves over time as a simple random walk:

$$\beta_t = \beta_{t-1} + w_t$$

$w_t$ is process noise, representing small, unpredictable changes in the hedge ratio.

### Step 2: Kalman Filter Recursive Steps

The Kalman Filter combines prediction (based on previous state estimates) and correction (using new evidence from observations).

- Prediction step: Estimate the next state and its uncertainty.

$$\hat{\beta}_{t|t-1} = \hat{\beta}_{t-1|t-1}$$

$$P_{t|t-1} = P_{t-1|t-1} + Q$$

  where $Q$ is the process noise covariance, and $P$ is the error covariance matrix.
- Update step: Adjust the predicted state in light of the new observation.

$$e_t = Y_t^{(y)} - H_t \hat{\beta}_{t|t-1}$$

This is the innovation, or measurement residual (difference between what we expect and what we observe).

$$S_t = H_t P_{t|t-1} H_t^T + R$$

$S_t$ is the total uncertainty, combining predicted uncertainty and measurement noise $R$.

$$K_t = P_{t|t-1} H_t^T S_t^{-1}$$

$K_t$ is the Kalman Gain, determining how much trust to put in the new observation versus the prediction.

$$\hat{\beta}_{t|t} = \hat{\beta}_{t|t-1} + K_t e_t$$
$$P_{t|t} = (I - K_t H_t) P_{t|t-1}$$

This recursive process continues, updating our best estimate of the hedge ratio and its uncertainty with each new time point.

### Step 3: Dynamic Spread and Z-score Signal Generation

Once the dynamic hedge ratio $\hat{\eta}_t$ is estimated, we re-calculate the spread:

$$\hat{\varepsilon}_t = Y_t^{(y)} - \hat{\alpha}_t - \hat{\eta}_t Y_t^{(x)}$$

To filter out normal fluctuations and focus on statistically significant deviations, we standardize the spread using a z-score:

$$Z_t = \frac{\hat{\varepsilon}_t - \mu(\hat{\varepsilon})}{\sigma(\hat{\varepsilon})}$$

Here, $\mu(\hat{\varepsilon})$ and $\sigma(\hat{\varepsilon})$ are the rolling mean and standard deviation of the spread.

Signals are generated when the z-score crosses pre-defined thresholds. For example, we enter a trade when the z-score is unusually high or low, expecting it to revert toward zero, as done previously in static hedge ratio case.

**Intuitive Summary:** The Kalman Filter offers a mathematically principled way to adapt the hedge ratio as the joint behavior of the stocks changes over time.

## Phase 5: Multivariate Modeling Vector Error Correction Model – VECM

This phase extends pairs trading to a multivariate setup, where we can simultaneously model and exploit relationships between more than two stocks. The Vector Error Correction Model (VECM) generalizes cointegration and allows us to capture both short-term fluctuations and long-term equilibrium in a group of non-stationary series.

### Step 1: Mathematical Structure and Intuition

For $N$ stocks, we arrange their log-prices into a vector:

$$\mathbf{y}_t = [y_t^{(1)}, y_t^{(2)}, ..., y_t^{(N)}]^T$$

Each $y_t^{(i)}$ is typically non-stationary (e.g., their mean changes over time), but a particular linear combination of them may be stationary (mean-reverting). VECM focuses on these combinations to find robust trading signals.

First, we take differences (log-returns):

$$\mathbf{r}_t = \Delta \mathbf{y}_t = \mathbf{y}_t - \mathbf{y}_{t-1}$$

VECM is specified as:

$$\mathbf{r}_t = \Pi \mathbf{y}_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \mathbf{r}_{t-i} + \mathbf{w}_t$$

Here:
- $\Pi$ (N*N matrix) captures the long-term equilibrium relationships,
- $\Gamma_i$ (N*N matrix) reflects short-term interactions,
- $\mathbf{w}_t$ (N*1 vector) is random noise,
- $p$ is the lag order (how many previous periods we look back on).

### Step 2: Understanding Cointegration and Error Correction

The matrix $\Pi$ can be decomposed if there are $r$ cointegrating relationships ($r < N$):

$$\Pi = \alpha \beta^T$$

where:
- $\beta$ (N*r matrix) gives the cointegrating vectors (linear combinations that are stationary),
- $\alpha$ (N*r matrix) tells us how quickly the prices adjust toward equilibrium.

For trading, the key signal is the Error Correction Term (ECT):

$$ECT_{t-1} = \beta^T \mathbf{y}_{t-1}$$

When $ECT_{t-1}$ is far from zero, it signals the system is away from equilibrium and is likely to revert, creating a trading opportunity.

**Step 3: Estimation and Trading Signal Generation**
We typically estimate the parameters (how many cointegrating relationships there are, and what they look like) using procedures like Johansen's test. We will use *statsmodels.tsa.vector_ar.vecm* in Python to implement the VECM. The practical steps are mentioned below:-

1. **Input:** price series for $N$ assets, lookback window $W$, lag order candidate set for $p$, thresholds $(z_{\text{enter}}, z_{\text{exit}})$.
2. Compute log-prices $(y_t)$ over the lookback window.
3. Choose lag $(p)$ via information criteria (AIC/BIC) fitted on a VAR$(p)$ in *levels.*
4. Run Johansen test on $y_{t-1}$ levels to obtain cointegration rank $r$ and estimate $\beta$. Johansen test uses eigenvectors and eigenvalues to estimate $\beta$. Also obtain $\alpha$ and preliminary $\Gamma_i$ if available.
5. For each time $t$ (or on each re-estimation step):
   (a) compute $\text{ECT} * t = \beta^\top y_t \qquad (r \times 1) vector$
   (b) compute standardized scores $z_t = (\text{ECT}_t - \mu_{\text{ECT}})/\sigma_{\text{ECT}})$, where $\mu, \sigma$ are estimated with a long or rolling window
   (c) if any $|z_{t,i}| > z_{\text{enter}}$:
      - determine trade sign from the sign of $z_{t,i}$ (positive $\rightarrow$ short the $\beta$-defined basket; negative $\rightarrow$ long the basket)
      - compute hedged positions using the corresponding column(s) of $\beta$ (weights $w \propto -\beta$ for that ECT), normalize (e.g., sum abs weights = 1) and scale by risk budget (target volatility)
   (d) if all $|z_{t,i}| < z_{\text{exit}}$ then close corresponding positions

**Summary and Advantages:** VECM is useful because it doesn't just consider isolated pairs — it can include many stocks, finding robust combinations that revert together. It handles both fast short-term moves and slow drift back to economic equilibrium, making it especially powerful for multi-dimensional statistical arbitrage.

## Performance Evaluation and Backtesting

After designing trading strategies using cointegration, ARIMA, static hedge ratio, Kalman filter, and VECM, it is necessary to evaluate their performance. This is done through backtesting and by calculating several standard financial metrics.

**Step 1: Backtesting Approach**
Backtesting means simulating how the strategy would have performed in the past, using historical data. We operate in a rolling or walk-forward fashion: at each step, the model is trained on recent data, then used to predict and generate signals on the next unseen period.

**Step 2: Key Financial Metrics**
Several measures help us objectively evaluate and compare strategies:

- **Net Profit and Loss (Net PnL):** The total profit earned after deducting all transaction costs and fees. NetPL is the real amount that matters to a trader or investor.
$$\text{Net PnL} = \text{Gross PnL} - \sum_{i=1}^{N_{\text{trades}}} \text{TC}_{\text{total},i}$$
- **Compound Annual Growth Rate (CAGR):** This tells us the yearly growth rate of the portfolio, assuming profits are reinvested. The formula is:
$$CAGR = \left(\frac{\text{End Value}}{\text{Start Value}}\right)^{1/\text{Years}} - 1$$
- **Sharpe Ratio (SR):** This is a measure of risk-adjusted return. It shows how much extra return is received per unit risk taken. The formula is:
$$SR = \frac{ER_p - R_f}{\sigma_p}\sqrt{T}$$
where $ER_p$ is expected portfolio return, $R_f$ is risk-free return, $\sigma_p$ is standard deviation of returns, and $T$ is the number of periods in a year.
- **Maximum Drawdown (MDD):** This quantifies the largest drop from a peak to the trough in the portfolio value:
$$MDD = \max_{0 \le t < T} \frac{PV_{peak} - PV_t}{PV_{peak}}$$
where $PV_{peak}$ is the highest value reached and $PV_t$ is value at time $t$.
- **Transaction Costs:** Real strategies must account for costs per trade (broker fees, taxes like STT, slippage). We model total transaction costs using a combined approach:
$$TC_{\text{total}} = (C_{fixed\_pct} + \alpha) \times \text{Transaction Value}$$
where $C_{fixed\_pct}$ represents fixed percentage costs (brokerage, statutory fees $\approx$ 0.05%) applied to turnover (Transaction Value), and $\alpha$ represents estimated variable costs (slippage/impact) proportional to the Transaction Value.

**Step 3: How Metrics Guide Development**
These metrics allow us to answer key questions:

- Did the mean-reversion model actually translate into profits?
- Was the profit stable and risk-adjusted, or volatile and risky?
- How much of the returns were eaten up by trading costs?
- Did the strategy survive market downturns (MDD)?

By comparing results for all models on these metrics, we objectively determine which approach offers the best combination of returns, risk management, and robustness.

## 5 Results

This section presents the findings of our proposed methodology.

### Cointegration Test Results

We first applied the Engle-Granger (OLS+ADF) cointegration test to all unique intra-sector pairs in our dataset. Our criteria for cointegration requires an ADF p-value on the residuals to be less than 0.05.
The analysis successfully identified **37 statistically significant cointegrated pairs** that meet this criterion. The top 5 pairs with the lowest p-values are given below:

- Pair: `ASHOKLEY.NS - BHARATFORG.NS`, p-value: 0.0000
- Pair: `MOTHERSON.NS - TVSMOTOR.NS`, p-value: 0.0000
- Pair: `TATAMOTORS.NS - TVSMOTOR.NS`, p-value: 0.0001
- Pair: `MOTHERSON.NS - TATAMOTORS.NS`, p-value: 0.0003
- Pair: `BHARATFORG.NS - MOTHERSON.NS`, p-value: 0.0003

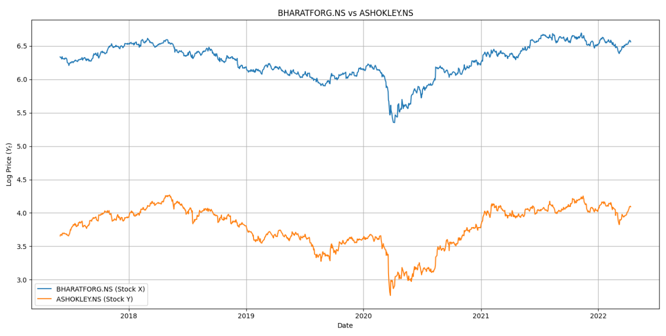

Figure 4: Log-Price Series: BHARATFORG.NS (Blue) vs. ASHOKLEY.NS (Orange), 2017-2023. Their strong co-movement is visually evident.

### Univariate Benchmarking (ARIMA)

As a benchmark, we applied the `auto_arima` function to the log-price series of individual stocks. The consistent result, as shown for ASHOKLEY.NS in Figures 5 and 6, was that the ACF and PACF plots show no significant autocorrelation at any non-zero lags. This means that the log-return series (after differencing once, d=1) beahves as white noise, and hence the best model fit for the log price series is ARIMA(0,1,0), a pure random walk.

This finding confirms the Efficient Market Hypothesis; past returns do not predict future returns. Therefore, we cannot forecast stock prices using this traditional time series model, and any trading strategy based on it would fail when accounted for transactional costs. Hence, we move onto multivariate approaches.

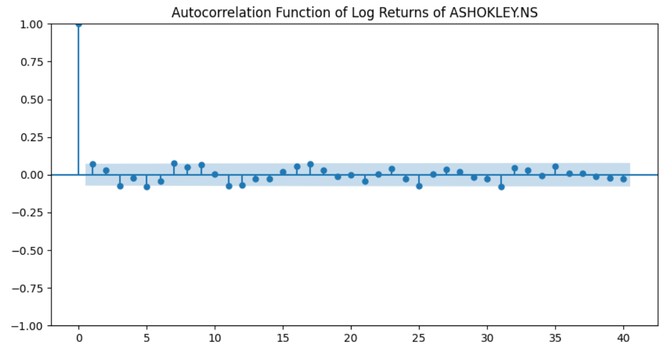

Figure 5: ACF Plot of ASHOKLEY.NS Log-Returns

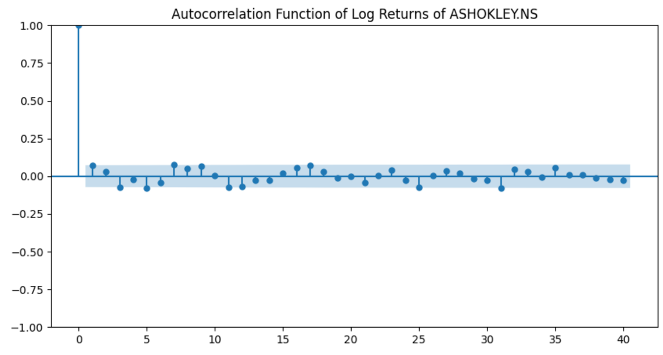

Figure 6: PACF Plot of ASHOKLEY.NS Log-Returns

## Static Hedge Ratio Results

For this strategy, we selected the cointegrated pair BHARATFORG.NS and ASHOKLEY.NS. The static hedge ratio ($\eta$) for the entire period was confirmed at 1.1256. The backtest was run with the following parameters:

- Pair: ASHOKLEY.NS (Y) / BHARATFORG.NS (X)
- Static Hedge Ratio ($\eta$): 1.1256
- Entry Z-score ($k$): 1.5
- Exit Z-score ($k_{exit}$): 0.25
- OLS Lookback: 1203 days
- Z-Score Window: 60 days

Figure 7 shows the calculated OLS spread over time with the Z-score entry/exit thresholds. Figure 8 illustrates the specific entry and exit points generated by the strategy. The resulting performance is shown in the equity curve (Figure **??**) and summarized in Table 2.

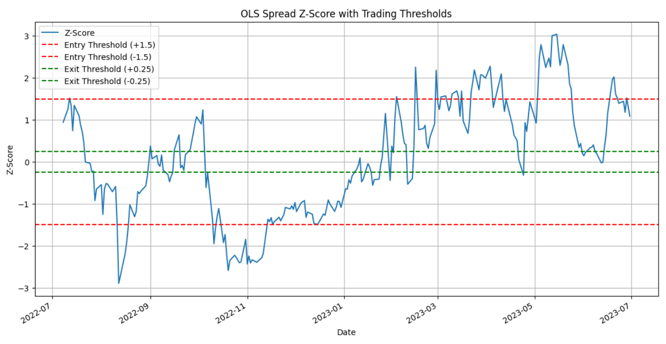

Figure 7: Static OLS spread with Z-score entry/exit thresholds

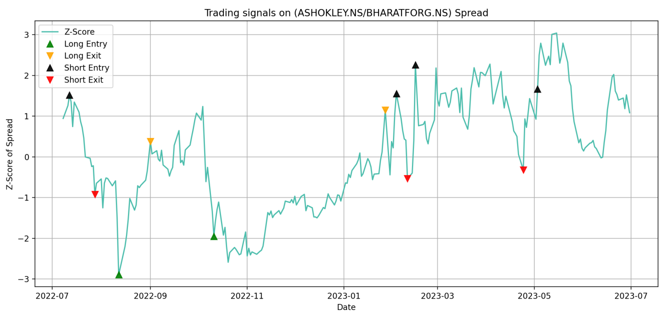

Figure 8: Static OLS spread with Trading Signals

Table 1: Final Performance Matrix of Static OLS Strategy

| **Performance Metric** | **Result** |
|---|---|
| Initial Capital (₹) | 100,000.00 |
| Final Capital (₹) | 108515.31 |
| Net Profit and Loss (%) | 13.67% |
| CAGR (%) | 6.97% |
| **Annualized Sharpe Ratio** | **0.30** |
| Maximum Drawdown (MDD) | -12.23% |
| Total Transaction Costs (₹) | 1288.03 |
| Total Trades Executed | 11 |

**Initial Conclusions**

**Risk-Adjusted Efficiency:** The Annualized Sharpe Ratio (**0.30**) is positive but low, confirming that while the statistical edge exists, it is not highly efficient at generating returns for the risk taken.

## Dynamic Hedge Ratio using Kalman filter

This strategy uses a dynamic, time-varying hedge ratio ($\eta_t$) estimated by the Kalman Filter. Figure 10 shows the trade signals generated.

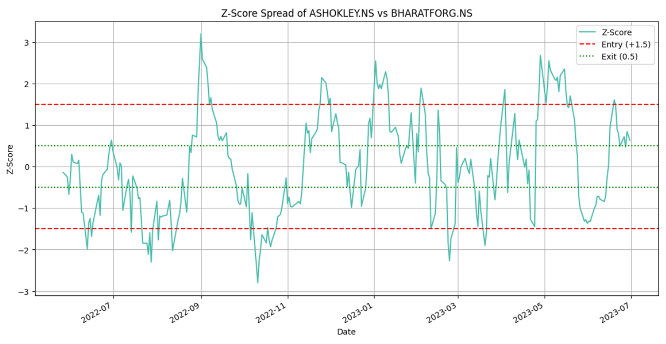

Figure 9: Dynamic Spread (Enter ±1.5, Exit 0.25)

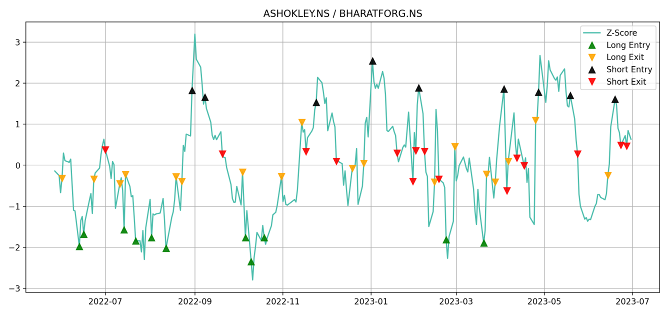

Figure 10: Dynamic Spread with Trading Signals (Enter ±1.5, Exit 0.25)

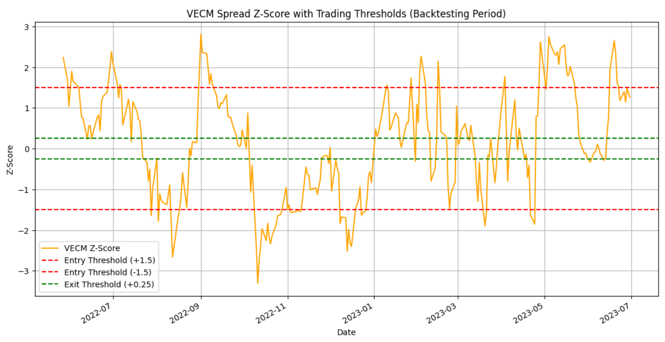

Figure 11: VECM Spread Z-Score with Trading Thresholds

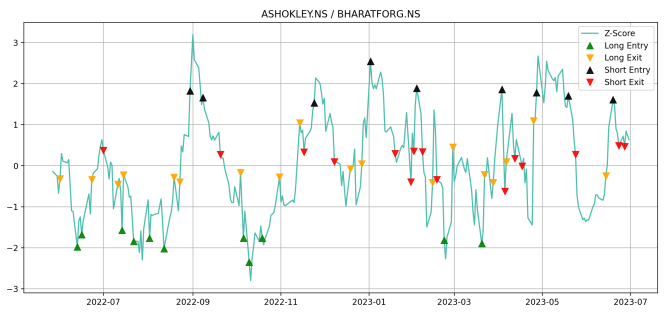

Figure 12: VECM Spread with Trading Signals

Table 2: Final Performance Matrix

| Performance Metric | Result |
|---|---|
| Initial Capital (₹) | 100,000.00 |
| Final Capital (₹) | 119731.55 |
| Net Profit and Loss (%) | 19.73% |
| CAGR (%) | 12.28% |
| **Annualized Sharpe Ratio** | **1.31** |
| Maximum Drawdown (MDD) | -7.37% |
| Total Transaction Costs (₹) | 399.60 |
| Total Trades Executed | 53 |

The implementation successfully utilized the Kalman Filter to generate a dynamic hedge ratio ($\eta_t$), moving beyond the limiting assumption of a static relationship. The KF's adaptive estimation proved effective in stabilizing the spread, which is essential when analyzing volatile market periods.

The Kalman filter, using tighter entry/exit boundaries (1.5 / 0.25), confirmed the strategy's robustness. The comprehensive performance matrix shows a Net P&L of ₹19731.56 on a ₹100k initial capital over the 301-day testing period. Most importantly, the **Annualized Sharpe Ratio of 1.31** and **Compound Annual Growth Rate of 12.28** is a dramatic improvement over the OLS strategy, demonstrating the clear value of the dynamic hedge ratio in delivering superior risk-adjusted returns.

## Strategy 3: VECM + Z-Score

This strategy utilizes a **Vector Error-Correction Model (VECM)** to assess the cointegrating relationship and generate trading signals. The model naturally provides a dynamic hedge ratio from the cointegrating vector. The backtest was conducted with these parameters:

- Pair: `ASHOKLEY.NS` (Y) / `BHARATFORG.NS` (X)
- Entry Z-score ($k$): 1.5
- Exit Z-score ($k_{exit}$): 0.0
- VECM Model Lag: 1
- Cointegration Rank: 1
- Z-Score Window: 30 days

Figure 11 presents the calculated VECM spread and its Z-score over time, alongside the established entry and exit thresholds. Figure 12 displays the specific entry and exit points generated by the strategy on the spread. Table 3 summarizes the resulting performance.

Table 3: Final Performance Matrix of VECM Strategy

| Performance Metric | Result |
|---|---|
| Initial Capital (₹) | 100,000.00 |
| Final Capital (₹) | 103,997.72 |
| Net Profit and Loss (%) | 4.99% |
| CAGR (%) | 3.65% |
| **Annualized Sharpe Ratio** | **0.02** |
| Maximum Drawdown (MDD) | -6.88% |
| Total Transaction Costs (₹) | 2,016.95 |
| Total Trades Executed | 25 |

**Strategic Conclusions: VECM Strategy**

- **Risk-Adjusted Efficiency:** The **Annualized Sharpe Ratio of 0.02** is low, indicating that while the VECM approach is statistically solid for system modeling, it did not effectively convert the cointegration into superior risk-adjusted returns during the backtesting period.
- **Comparative Performance:** The VECM strategy performed worse than both the Static OLS and significantly, the Kalman Filter strategy, which had an Annualized Sharpe Ratio of 1.31. This implies that for this pair, the error-correction term's impact on trading signals was less practically beneficial than the dynamically estimated hedge ratio of the Kalman Filter.

## 6 Conclusion

This project conducted a rigorous comparison of three distinct pairs trading strategies: the **Static Hedge Ratio (OLS)**, the **Kalman Filter (KF)**, and the **Vector Error-Correction Model (VECM)**. The analysis was performed on a cointegrated pair (`ASHOKLEY.NS` and `BHARATFORG.NS`), with the objective of comparing their effectiveness in generating profitable signals and managing risk within a unified backtesting framework.

Our analysis revealed a clear hierarchy of performance:

- **Static Hedge Ratio (OLS):** As anticipated, this baseline strategy, which relies on a fixed hedge ratio, demonstrated moderate performance. While it achieved a positive **CAGR of 6.97%** and a **Sharpe Ratio of 0.30**, its inability to adapt to evolving market dynamics led to a higher **Maximum Drawdown (MDD) of -12.23%**. This underscores the critical limitation of assuming a constant relationship between assets over time.

- **Kalman Filter (KF):** This strategy emerged as the **most robust and profitable**. By dynamically estimating the hedge ratio ($\eta_t$), the Kalman Filter adapted effectively to changes in the cointegrating relationship. It delivered a superior **CAGR of 12.28%** and an impressive **Annualized Sharpe Ratio of 1.31**. Furthermore, it managed risk more efficiently, exhibiting the lowest **MDD of -7.37%** among the three strategies.

- **Vector Error-Correction Model (VECM):** Despite offering the most complete theoretical framework, the VECM strategy exhibited the poorest practical performance. It resulted in a modest **CAGR of 3.65%** and a near-zero **Annualized Sharpe Ratio of 0.02**. This underperformance, suggests that while a statistical long-term relationship might exist, its practical exploitability for profitable trading signals in this specific context was minimal. This might be due to a poorer hyperparameter tuning from our side.

**Final Conclusion** The empirical evidence from this study demonstrates a definitive takeaway: strategies that incorporate adaptive, time-varying mechanisms are definitively superior to static models for pairs trading. While the VECM provides a robust theoretical framework, the **Kalman Filter strategy proved to be the most effective and adaptable method** for exploiting the mean-reverting behavior of the selected cointegrated pair. Its ability to dynamically adjust the hedge ratio was very critical.

## 7 Future Direction

This analysis provides a strong foundation for comparing pairs trading strategies, but several key areas for expansion and improvement remain.

### 1. Dynamic and Continuous Position Sizing

A significant limitation of our current backtest is the static position sizing (allocating 50% of capital per trade, regardless of signal strength). A more sophisticated approach would be to scale the position size ($N$) based on the magnitude of the deviation. The conviction in a mean-reversion trade is much higher when the Z-score is $\pm 3.0$ than when it is $\pm 2.0$. This can be implemented in two primary ways:

- **Discrete/Tiered Sizing:** Implement a "scaling in" strategy. For example:
  - If $Z_t > 2.0$, enter a $1N$ short position.
  - If $Z_t$ continues to rise $> 2.5$, add another $N$ to the short position.
  - If $Z_t$ rises $> 3.0$, add a final $N$ to the position.

  This "pyramiding" approach concentrates capital on what are perceived to be the most extreme (and thus most probable) reversions.
- **Continuous Sizing:** A more elegant method is to make the position size a continuous function of the Z-score. We could define our target allocation $N_{\text{target}}$ using a smooth function, like the hyperbolic tangent:

$$N_{\text{target}} = N_{\max} \times \tanh(Z_t / k_{\text{sensitivity}})$$

  This would automatically scale the position: very small for weak signals (low $Z_t$), and smoothly increasing to a maximum size ($N_{\max}$) for extreme signals.

### 2. Predictive Signal Generation

Our strategies relied on a reactive Z-score (trading after a deviation has occurred). A powerful extension would be to use predictive signals. One could fit an ARIMA model to the stationary spread series itself and use its one-step-ahead forecast ($\hat{S}_{t+1}$) to anticipate the direction of the mean reversion. A stronger signal might be: "Trade short if $Z_t > 1.5$ and the ARIMA model forecasts $\hat{S}_{t+1} < S_t$."

### 3. High Frequency Approach

Since pairs trading is synonymous to a statistical arbitrage, one would get a much better performance if done on a higher frequency, i.e., on on hourly basis, instead of a daily basis. It is infact a very competitive strategy.

### 4. Expanded Universe and Rigorous Testing

Our study was focused on a specific set of sectors within the NSE. A more comprehensive study should expand this universe to include other sectors (e.g., Pharma, FMCG, Energy) as well as cross-asset pairs (e.g., Nifty vs. Bank Nifty futures, or commodities). Furthermore, all identified strategies (OLS, Kalman, VECM) should be rigorously backtested on the same set of identified pairs and across identical out-of-sample periods to enable a direct, "apples-to-apples" performance comparison.

### 7.1 5. Rolling Backtesting

Rolling backtesting involves training followed by testing followed by training again and so on, which is proven to give a better performance as the stock price's cointegration might change over time due to various market factors.

## 8 Shortcomings

The primary shortcoming of our project is the implementation of VECM. We predicted VECM as a more robust alternative, but the results show that it formed the poorest among all the pairs trading strategies. We believe that the main reason behind this might be due to a poor hyperparameter tuning (i.e., OLS_LookBack Window, Portion Sizing, Entry/Exit Thresholds)

Finally, while our methodology is robust, key parameters (e.g., OLS/Z-Score lookback windows, Z-score entry/exit thresholds $k$ and $k_{exit}$, and Kalman noise parameters $Q$ and $R$) were set manually based on common heuristics. The portfolio values were indeed very dependent on the values of portion sizing, z-score window and entry/exit thresholds. A fully rigorous study would involve optimizing these parameters through cross-validation or walk-forward analysis, which could lead to further performance improvements.

## References

[1] DA341 Course Materials. (2025). *Applied Time Series Analysis.* Course lectures on ARIMA modeling.

[2] Palomar, D. P. (2020). *Pairs Trading.* Lecture Slides, MAFS5310, HKUST. Retrieved from: `https://palomar.home.ece.ust.hk/MAFS5310_lectures/slides_pairs_trading.pdf`

[3] Wikipedia. (2024). *Efficient-market hypothesis.* Retrieved from: `https://en.wikipedia.org/wiki/Efficient-market_hypothesis`

[4] Gatev, E., Goetzmann, W. N., & Rouwenhorst, K. G. (2006). Pairs trading: Performance of a relative-value arbitrage rule. *The Review of Financial Studies, 19*(3), 797-827.

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

[6] Wikipedia. (2024). *Augmented Dickey–Fuller test.* Retrieved from: `https://en.wikipedia.org/wiki/Augmented_Dickey%E2%80%93Fuller_test`

[7] Wikipedia. (2024). *Error correction model.* Retrieved from: `https://en.wikipedia.org/wiki/Error_correction_model`

[8] Justin Eloriaga, YouTube. (2021). *Introduction to the Vector Error Correction Model.* Retrieved from: `https://youtu.be/e6CgA1TuZiA?si=H1mj-xSJ1Mf-jAJP`

[9] Statsmodels. (2025). *statsmodels.tsa.stattools.adfuller - Augmented Dickey-Fuller unit root test.* Retrieved from: `https://www.statsmodels.org/dev/generated/statsmodels.tsa.stattools.adfuller.html`