Mathematical Finance Magazine

Volume 6

February 2026

Massive thanks to this edition's writers:

Max Hamilton-Kerr, Ignacio Vigil Cardenal and Gonzalo García Suárez de Lezo, Etiene Inyang, Alberto Álvarez and Javier Fernández, Yiling Guo, Kubrat Hruzewicz, Nazmi Hanizam, Shrivar Singh, Ryan Tan Jun Xiong, Valerio Altissimo and Natan Boyd, Karthigan Sabanadesan, Ray Riyaz

Edited by: Abhishek Agarwal, Valerio Altissimo, Aggelos Goussiakis, Adam Lewandowski, Arsh Mir, Francesco Papini

Contents

Preface

Welcome to the Sixth Edition of the Warwick Mathematical Finance Magazine, brought to you by the Warwick Finance Societies! This is a student-run magazine, consisting of a collection of articles that lie at the intersection between mathematics and finance, all of which have been written by students at Warwick.

Our aim is to provide an insight into mathematics beyond that which students would learn in their degree - whether they are mathematics students or not. We have made this magazine accessible to a wide range of mathematical abilities, and hope to give a little bit of inspiration to people to use more maths in their work or read up on things that interest them.

This volume consists of several different articles, each covering a different area of mathematical finance. From financial models to theorems, we will look at their mathematical underpinnings and their real-world applications.

We thank the Warwick Finance Societies' board for all their help and support.

We hope you enjoy, and happy reading!

– Valerio Altissimo and Francesco Papini, Head Editors

Chapter 1

Continuous Signals, Binary Trades: The Mathematics Behind Long-Short Decision Systems

Max Hamilton-Kerr

Edited by: Aggelos Goussiakis

Abstract: This article establishes the mathematical framework through which quantitative researchers transform continuous data into discrete long-short trading decisions. By formalising signals as real-valued scores subject to rank normalisation and cross-sectional standardisation, we demonstrate how portfolio weights emerge mechanically from signal ordering under dollar-neutrality and beta-neutrality constraints. The study details how continuous variables are quantified and transformed into stable signals through normalisation, temporal smoothing, and regularisation to control overfitting. This framework extends to non-numerical inputs, showing how corporate language from earnings calls can be quantified via term-frequency weightings and processed through identical pipelines. Application to high-frequency microstructure signals illustrates the scale invariance of the decision architecture across different time horizons. A case study on Federal Reserve communications examines how markets respond to quantified signals in real time through surprise-based construction measuring deviations from historical expectations. Throughout, the emphasis is on demonstrating that the same mathematical logic governs long-short systems regardless of data modality.

1.1 Introduction

Financial markets produce endless data, but the final output of any strategy is discrete: you are long, short, or flat. The task is transforming high-dimensional information into these specific choices. Long-short systems handle this by ranking assets to exploit relative differences, which reduces exposure to general market swings.

The following questions are the motivation for this article: *how do quantitative researchers convert real-world data into long-short decisions? Is this general, or does it vary across different data modalities?* The process, in fact, follows a clear recipe: raw data becomes numerical features, features become a scalar signal, and the signal determines portfolio weights within

neutrality constraints (long–short neutral portfolios are considered optimal). This framework applies whether you are looking at equity factors or high-frequency data. We are investigating the mathematical logic of this mapping and how it stays rigorous when moving from numbers to more complex inputs like language.

To address this question, the article proceeds as follows. Section 1.2 introduces the mathematical structure of long–short neutral portfolios, providing intuition for why ranking and neutralisation are central to signal-based trading. Section 1.3 discusses how continuous variables are quantified and transformed into stable signals, emphasising the role of normalisation and regularisation. Section 1.4 applies this framework to language and speech data, using corporate communications as a concrete example of how non-traditional inputs can be mathematically operationalised. Section 1.5 extends the discussion to high-frequency trading, where the same decision architecture operates on vastly shorter time scales. Finally, Section 1.6 examines central bank communication—illustrated by Federal Reserve Chair Jerome Powell—as a case study in how markets respond to quantified informational signals in real time.

Throughout, the emphasis is not on proposing a single trading strategy, but on clarifying the mathematical logic that connects raw information to long–short decisions. By making this structure explicit, the article aims to highlight both the power and the limitations of quantitative models in translating complex economic signals into systematic market actions.

1.2 Long–Short Decision Systems

At the core of many quantitative trading strategies lies the long–short decision system. Rather than attempting to forecast the absolute direction of the market, these systems focus on identifying relative mispricings between assets. The intuition is quite straightforward: if one asset is expected to outperform another, a portfolio can be constructed that profits from this difference while remaining largely insulated from broad market movements.

This relative-value perspective is central to modern quantitative finance. By simultaneously taking long positions in assets with favourable signals and short positions in assets with unfavourable signals, long–short systems aim to isolate informational content while controlling for systematic risk. As emphasised by Grinold and Kahn [6], the objective is not prediction accuracy per se, but the consistent ranking of assets according to expected performance.

1.2.1 Signals as Continuous Scores

The starting point of any long–short system is a quantitative signal. For each asset i in a universe of N assets, researchers construct a real-valued score

$$f_i \in \mathbb{R},$$

which summarises the information believed to be predictive of future returns. Crucially, f_i is time-varying, as the predictive potency and magnitude of a signal fluctuate in response to a myriad of market factors. It is also important to note that this signal is continuous rather than categorical, meaning it does not label assets as simply “good” or “bad”, but instead assigns each a relative strength.

This formulation reflects a key principle in quantitative investing. Information arrives in noisy and graded forms, and it is generally more robust to rank assets by relative signal strength than to rely on hard classification thresholds. As noted by Green et al. [5], many distinct characteristics contain incremental information, and their usefulness often emerges only through cross-sectional comparison.

1.2.2 Ranking and Portfolio Construction

Once signals $\{f_i\}_{i=1}^N$ are computed, assets are ranked cross-sectionally. A simple and widely used transformation is the rank normalisation

$$r_i = \frac{\text{rank}(f_i)}{N},$$

which maps signals onto the unit interval. Ranking serves two purposes. First, it reduces sensitivity to outliers and scale differences across signals. Second, it enforces the relative nature of the strategy: only ordering matters, not absolute magnitude.

Portfolio weights are then constructed from these ranks. A common choice is

$$w_i = r_i - \bar{r},$$

where $\bar{r} = \frac{1}{N}\sum_{i=1}^N r_i$. By repeating this process over successive periods, the system generates a time series of weights that maintain these properties at each rebalancing interval. This ensures that

$$\sum_{i=1}^N w_i = 0,$$

producing a dollar-neutral long–short portfolio. Note that neutrality is maintained over every period in the sequence. As a result, assets with above-average ranks receive positive weights (long positions), while those with below-average ranks receive negative weights (short positions).

This construction highlights a defining feature of long–short decision systems: positions emerge mechanically from signal ordering rather than discretionary judgement.

1.2.3 Neutrality and Risk Control

In practice, dollar neutrality alone is insufficient. Signals may be unintentionally correlated with known risk factors such as market beta, sector exposures, or size effects. To address this, long–short systems impose additional neutrality constraints.

Let β_i denote the market beta of asset i. A beta-neutral portfolio satisfies

$$\sum_{i=1}^N w_i \beta_i = 0.$$

More generally, exposures to multiple factors can be removed via regression-based neutralisation, a standard technique in systematic portfolio construction [1]. This step ensures that portfolio performance reflects the intended signal rather than unintended factor bets.

From a decision-making perspective, neutrality transforms the signal into a purer measure of relative opportunity. It aligns the mathematical structure of the portfolio with the economic intuition of exploiting informational differences while controlling for systematic risk, a principle closely tied to risk-adjusted evaluation metrics such as the Sharpe ratio [9].

1.2.4 Link to the Research Question

This section establishes the mathematical backbone of long–short decision systems. Continuous signals are constructed from data, transformed through ranking, and converted into neutral portfolios via explicit constraints. Crucially, this architecture is agnostic to the origin of the signal. Whether the input arises from prices, volatility, language, or microstructure data, the same decision logic applies.

This observation directly motivates the central question of the article: how can diverse and increasingly unconventional data sources be quantified so that they fit into this long–short framework? The following sections examine how continuous variables are constructed and stabilised, before applying this structure to language, high-frequency trading, and central bank communication.

1.3 Quantifying Continuous Variables

Long–short decision systems rely on scalar signals that can be ranked, neutralised, and translated into portfolio positions. However, most real-world data do not naturally appear in this form. Prices, volumes, volatility measures, and macroeconomic indicators are continuous, noisy, and often non-comparable across assets or time. The role of quantitative modelling at this stage is therefore to transform raw continuous variables into stable numerical signals that are suitable for systematic decision-making.

This section describes the mathematical principles behind this transformation. Rather than surveying specific data sources, the focus is on the operations required to make any continuous variable compatible with the long–short framework introduced in Section 1.2.

1.3.1 Continuous Variables as Noisy Measurements

Let $X_i(t) \in \mathbb{R}$ denote a raw continuous variable associated with asset i at time t, such as realised volatility, trading volume, or a fundamental ratio. In practice, such variables are contaminated by measurement error, market microstructure noise, and regime-dependent effects. As a result, the raw value $X_i(t)$ is rarely a reliable input for portfolio construction.

From a decision-system perspective, the objective is not to estimate $X_i(t)$ precisely, but to extract the component of $X_i(t)$ that is informative about relative future returns. This distinction is critical. As emphasised by Grinold and Kahn [6], the usefulness of a signal lies in its ability to consistently rank assets rather than in its absolute accuracy.

1.3.2 Normalisation and Cross-Sectional Comparability

To enable ranking, signals must be comparable across assets. A common first step is standardisation:

$$\tilde{X}_i(t) = \frac{X_i(t) - \mu_X(t)}{\sigma_X(t)},$$

where $\mu_X(t)$ and $\sigma_X(t)$ denote the cross-sectional mean and standard deviation at time t. This transformation removes scale effects and ensures that signals derived from different assets contribute symmetrically to the ranking process.

In many applications, ranking itself provides an additional layer of robustness. As discussed in Section 1.2, rank-based signals reduce sensitivity to outliers and nonlinear distortions. In long–short systems, ranking is a structural necessity rather than a convenience: positions are determined by relative order rather than absolute levels.

1.3.3 Temporal Smoothing and Stability

Raw signals often exhibit excessive short-term variability. If left unaddressed, this instability leads to high turnover and poor out-of-sample performance. To mitigate this, quantitative researchers apply temporal smoothing, for example through exponentially weighted averages:

$$\bar{X}_i(t) = \sum_{k=0}^{\infty} \lambda^k X_i(t-k), \quad 0 < \lambda < 1.$$

Smoothing serves a dual purpose. First, it reduces noise by aggregating information over time. Second, it aligns the signal with the finite holding periods typically employed in long–short portfolios. This reflects an implicit modelling assumption: economically meaningful information persists long enough to be exploited systematically.

1.3.4 Regularisation and Overfitting Control

A central risk in signal construction is overfitting. When researchers have access to many continuous variables, it is easy to construct signals that perform well in-sample but fail out-of-sample. This issue is particularly acute in cross-sectional strategies, where small improvements in ranking can appear statistically significant but lack economic substance.

To address this, modern quantitative research emphasises regularisation and parsimony. Rather than fitting complex models to maximise in-sample performance, signals are constrained to be stable and interpretable. As argued by Harvey et al. [7], disciplined testing and conservative model design are essential to avoid false discoveries in empirical asset pricing.

From a long–short decision-system perspective, regularisation ensures that portfolio weights are driven by persistent structure rather than transient noise. This aligns with the broader asset pricing principle that expected returns should be linked to economically meaningful risk exposures [3], even when the precise structural model is only partially understood.

1.3.5 How Does This Result in Long/Short Decisions?

This section has established how continuous variables are transformed into stable, comparable signals suitable for long–short portfolio construction. Normalisation, ranking, smoothing, and regularisation are not optional refinements, but rather necessary steps that allow heterogeneous data to be integrated into a unified decision framework.

Having defined this mathematical machinery, the article now turns to specific data modalities. The following sections apply these principles to atypical inputs, beginning with language and corporate communication, to demonstrate how diverse forms of information can be systematically embedded within long–short decision systems.

1.4 Language and Corporate Communication as Quant Signals

The long–short decision framework described in Sections 1.2 and 1.3 is agnostic to the origin of the input signal. In this section, we demonstrate how corporate language can be treated as a continuous variable and embedded within the same mathematical structure used for traditional financial data.

Corporate communications, such as earnings calls and shareholder letters, convey information not only through explicit content but also through tone, emphasis, and uncertainty. A growing literature shows that these linguistic features are systematically related to subsequent asset returns. Using textual analysis of firm disclosures, Loughran and McDonald [8] demonstrate that finance-specific language measures contain incremental predictive power beyond standard numerical variables. Similarly, Tetlock [10] shows that media tone affects market prices and trading behaviour.

1.4.1 Quantifying Language as a Continuous Variable

From a quantitative perspective, language is transformed into a numerical representation through the construction of continuous-valued features. Let $d_i(t)$ denote a corporate document associated with firm i at time t. A simple sentiment-based signal can be expressed as

$$X_i(t) = \sum_k w_k \cdot \text{freq}_k(d_i(t)),$$

where freq_k denotes the frequency of term k in the document and w_k is a predefined weight reflecting its financial relevance.

Crucially, $X_i(t)$ is not used directly. As discussed in Section 1.3, the signal is normalised, smoothed, and ranked cross-sectionally to produce a scalar score suitable for long–short portfolio

construction. Firms with relatively optimistic or confident language receive positive weights, while those with comparatively pessimistic or uncertain language receive negative weights.

1.4.2 Decision-System Interpretation

Within the long–short framework, language-based signals are interpreted in exactly the same manner as volatility or valuation signals. The portfolio does not express an absolute view on whether corporate communication is "good" or "bad" in isolation; rather, it exploits relative differences in linguistic tone across firms at a given point in time.

This perspective is essential for robustness. By relying on ranking and neutrality, the decision system mitigates the risk that language measures capture broad market sentiment rather than firm-specific information. The result is a relative-value signal that can be combined with other factors without altering the underlying portfolio architecture.

1.4.3 How Does This Relate to Data Used by Quantitative Researchers?

This section illustrates that non-traditional data sources, such as language, can be systematically incorporated into long–short decision systems using the same mathematical transformations applied to conventional financial variables. The next sections extend this analysis to higher-frequency settings and macroeconomic communication, further emphasising the generality of the framework.

1.5 High-Frequency Trading and Microstructure Signals

While the previous section focused on low-frequency corporate communication, the same long–short decision architecture applies at much shorter time scales. In high-frequency trading (HFT), signals are derived not from documents or fundamentals, but from the dynamics of the limit-order book and transaction flow.

At these horizons, the primary inputs are continuous microstructure variables such as order arrival rates, queue imbalances, and short-term volatility. For example, a commonly studied quantity is order-flow imbalance:

$$\mathrm{OFI}(t) = \frac{V^{\mathrm{buy}}(t) - V^{\mathrm{sell}}(t)}{V^{\mathrm{buy}}(t) + V^{\mathrm{sell}}(t)},$$

where V^{buy} and V^{sell} denote executed buy and sell volumes over a short interval. Empirical and theoretical work shows that such imbalances are informative about short-term price movements [4].

Despite the difference in data frequency, the decision logic remains unchanged. Microstructure signals are normalised, smoothed over short horizons, and ranked across assets or trading venues. Positions are then taken in a market-neutral fashion, often holding long and short exposures for only fractions of a second. As in lower-frequency strategies, neutrality constraints ensure that profits arise from relative informational advantages rather than directional market exposure.

From a mathematical perspective, high-frequency trading illustrates the scale invariance of long–short decision systems. Whether the signal is derived from linguistic tone or order-book dynamics, the same mapping—from continuous variables to ranked signals to neutral portfolios—governs the trading decision. Differences arise in execution and latency, not in the underlying structure of the model [2].

1.6 Case Study: Jerome Powell

Central bank communication provides a natural setting in which to observe long–short decision systems operating in real time. Unlike corporate disclosures, Federal Reserve statements affect

the entire market simultaneously, influencing equities, rates, currencies, and volatility. Yet despite this breadth, market reactions are often highly structured and predictable in form, if not in magnitude.

In particular, the well-known market sensitivity to speeches by Federal Reserve Chair Jerome Powell - so much so that Powell has become infamous for tanking assets with a simple greeting - illustrates how linguistic and contextual signals are rapidly quantified and translated into trading decisions. Market participants frequently react not only to the explicit content of policy statements, but also to subtle deviations in tone, emphasis, and perceived confidence. The informal observation that markets move sharply when Powell says "good morning" reflects an underlying anticipation of informational content rather than randomness.

1.6.1 Event-Based Signal Construction

From a quantitative perspective, central bank communication is treated as a discrete event generating a burst of continuous signals. Let $d(t)$ denote a policy statement or speech delivered at time t. As in Section 1.4, textual features can be extracted to form a numerical representation $X(t)$, capturing sentiment, uncertainty, or deviation from historical language.

Crucially, the signal used in trading is not the absolute level of $X(t)$, but its relative surprise:

$$\Delta X(t) = X(t) - \mathbb{E}[X \mid \text{historical communications}].$$

Large deviations indicate that the communication differs materially from prior expectations. This surprise-based formulation aligns naturally with long–short decision systems, as it ranks events by informational intensity rather than attempting to interpret policy direction in isolation.

1.6.2 Market Implementation

Once quantified, central bank signals are mapped into market-neutral positions across asset classes. For example, an unexpectedly hawkish communication may generate short positions in equity indices and long positions in volatility or interest rate futures, while a dovish surprise may produce the opposite configuration. Importantly, these positions are often constructed to be neutral with respect to broad market trends, focusing instead on relative pricing adjustments across maturities or instruments.

This implementation mirrors the architecture described in earlier sections. Continuous inputs are transformed into scalar signals, ranked by magnitude, and converted into positions subject to explicit risk constraints. The speed at which markets respond to Powell's communication reflects not discretionary interpretation, but the rapid activation of pre-specified decision rules embedded in quantitative trading systems.

1.6.3 Synthesis and Interpretation

The Powell case study demonstrates the unifying theme of this article: diverse sources of information—language, tone, and contextual deviation—are processed through the same mathematical decision framework that governs long–short trading more broadly. The apparent drama of central bank speeches masks a mechanical reality in which markets respond to quantified signals rather than narrative interpretation.

In this sense, the reaction to central bank communication is not an anomaly but a clear illustration of how continuous real-world information is systematically translated into binary trading actions. Powell's words matter not because of their rhetorical force, but because they perturb established numerical expectations within long–short decision systems.

1.7 Conclusion

This article explores how researchers turn messy, varied data into the specific "buy, sell, or hold" choices that define long–short trading. While prices, corporate speech, and market microstructure look different on the surface, the mathematics used to process them is remarkably consistent.

The core of this system is a reliable pipeline: raw data is turned into numerical signals, cleaned through normalisation, and then ranked. These rankings allow a trader to take long and short positions that cancel out general market noise, focusing instead on relative value. This structure makes it possible to integrate diverse information while keeping the strategy robust against outliers and noise.

By applying this logic to high-frequency trading and Federal Reserve communications, we see that modern markets often respond to quantified deviations from expectations rather than "narrative" meaning. Even a speech by Jerome Powell becomes a numerical data point that a system compares against established benchmarks.

Ultimately, quantitative finance is less about predicting where the whole market goes and more about building rigorous transformations from information to action. Understanding this process is key to seeing how systematic trading really works, and what are its risks in a world surrounded by data.

Bibliography

[1] Andrew Ang. *Asset Management: A Systematic Approach to Factor Investing.* Oxford University Press, 2014.

[2] Álvaro Cartea, Sebastian Jaimungal, and José Penalva. *Algorithmic and High-Frequency Trading.* Cambridge University Press, 2015.

[3] John H. Cochrane. *Asset Pricing.* Princeton University Press, 2011.

[4] Rama Cont, Sasha Stoikov, and Rishi Talreja. A stochastic model for order book dynamics. *Operations Research*, 58:549–563, 2010.

[5] Jeremiah Green, John R. M. Hand, and X. Frank Zhang. The characteristics that provide independent information about average u.s. monthly stock returns. *Review of Financial Studies*, 30:4389–4436, 2017.

[6] Richard C. Grinold and Ronald N. Kahn. *Active Portfolio Management.* McGraw-Hill, 2000.

[7] Campbell R. Harvey, Yan Liu, and Heqing Zhu. ... and the cross-section of expected returns. *Review of Financial Studies*, 29:5–68, 2016.

[8] Tim Loughran and Bill McDonald. When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *Journal of Finance*, 66:35–65, 2011.

[9] William F. Sharpe. The sharpe ratio. *Journal of Portfolio Management*, 21:49–58, 1994.

[10] Paul C. Tetlock. Giving content to investor sentiment. *Journal of Finance*, 62:1139–1168, 2007.

Chapter 2

From Pairs to Portfolios: A Guide to Cointegration and Statistical Arbitrage

Ignacio Vigil Cardenal and Gonzalo García Suárez de Lezo

Edited by: Abhishek Agarwal and Valerio Altissimo

Abstract: This article provides a comprehensive bridge between econometric theory and the practical implementation of statistical arbitrage in quantitative finance. By distinguishing between superficial correlation and fundamental cointegration, we establish a rigorous foundation for market-neutral trading strategies. The study details the evolution from bivariate pairs trading using the Engle-Granger two-step method and Augmented Dickey-Fuller (ADF) tests to multivariate basket strategies employing the Johansen Test and Vector Error Correction Models (VECM).Beyond theoretical modelling, the article outlines the mechanics of signal generation via Z-score normalization and trade timing through the Ornstein-Uhlenbeck process, including the calculation of a trade's half-life for capital efficiency. Practical applications are demonstrated on the WorldQuant Brain platform.

2.1 Introduction

In the current landscape of quantitative finance, the distinction between correlation and cointegration is frequently misunderstood, leading to fragile trading strategies that fail when markets turn volatile. While many aspiring quants look for assets that simply "move together," this superficial relationship often breaks down exactly when reliability is needed most.

This article aims to bridge the gap between econometric theory and practical algorithmic trading. We will move beyond the basic technical analysis often found in retail trading blogs to offer a rigorous, academic approach to market neutrality. Our journey will take us from the theoretical foundations of simple pairs trading to the complex implementation of basket strategies using Vector Error Correction Models (VECMs), culminating in a practical application on the WorldQuant Brain platform.

2.1.1 The Drunk and the Dog: An Intuitive Framework

Before diving into the algebra of stationarity, it is crucial to dispel the myth that assets moving together are necessarily linked. To visualise the difference between a spurious correlation and a true economic relationship, we can utilise the classic "Drunk and the Dog" analogy.

Imagine a drunk man walking out of a bar. He stumbles randomly down the street. Now, imagine a stray dog wandering down the same street. Occasionally, they might both turn left or right at the same time. If you plotted their paths, you might calculate a high correlation coefficient—they are moving in the same direction by chance. However, this relationship is useless for prediction; if the drunk turns left, there is no guarantee the dog will follow. This represents correlation.

Now, imagine the drunk has his dog on a leash. The drunk walks in a random, stochastic path, and the dog wanders excitedly in various directions to sniff trees or fire hydrants. The distance between them expands and contracts, but they can never drift entirely apart because the leash—the fundamental economic tether—keeps them bound together.

In financial terms, the drunk and the dog are cointegrated. While their individual price paths (random walks) are unpredictable, the distance between them (the spread) is not. If the dog wanders too far away (the spread widens), the leash's tension eventually pulls him back toward the drunk (mean reversion).

This concept forms the bedrock of Statistical Arbitrage. We are not betting on where the drunk is going; we are betting that, eventually, the leash will pull the dog back.

2.2 Theoretical Foundation: Engle-Granger Method

With the intuition of the "Drunk and the Dog" established, we must translate this analogy into a rigorous statistical framework. The "leash" that binds our two assets together is mathematically defined by the concept of stationarity.

To identify this relationship, we turn to the Engle-Granger two-step method. While simple compared to modern multivariate methods, it remains the standard entry point for understanding the mechanics of pairs trading.

2.2.1 Regression

The first step is to identify the linear relationship between the two asset price series, P_t^A and P_t^B (representing the drunk and the dog). We regress one asset against the other to isolate the residuals [[1]].

We start with the linear model:

$$P_t^A = \alpha + \beta P_t^B + \epsilon_t$$

Here, β represents our Hedge Ratio—the amount of asset B required to hedge a unit of asset A. To derive the optimal β that minimises the variance of the residuals (the spread), we employ Ordinary Least Squares (OLS). Mathematically, we aim to find the estimator $\hat{\beta}$ that minimises the Sum of Squared Residuals (SSR):

$$SSR = \sum_{t=1}^{T} \left(P_t^A - \hat{\alpha} - \hat{\beta} P_t^B \right)^2$$

By taking the partial derivative with respect to β and setting it to zero, we arrive at the closed-form solution for the hedge ratio:

$$\hat{\beta} = \frac{\sum_{t=1}^{T}(x_t - \bar{x})(y_t - \bar{y})}{\sum_{t=1}^{T}(x_t - \bar{x})^2} = \frac{\text{Cov}(P_t^B, P_t^A)}{\text{Var}(P_t^B)}$$

Once we have derived the optimal hedge ratio ($\hat{\beta}$), calculating the intercept is straightforward. It is the difference between the mean of the dependent variable and the mean of the independent variable weighted by our beta:

$$\hat{\alpha} = \overline{P_t^A} - \hat{\beta}\overline{P_t^B}$$

Where: $\overline{P^A}$ is the sample mean of the prices of Asset A. $\overline{P^B}$ is the sample mean of the prices of Asset B.

This calculation ensures that the resulting spread (ϵ_t) is as "tight" as possible. Rearranging the original equation gives us the tradable portfolio:

$$\epsilon_t = P_t^A - \hat{\beta}P_t^B - \hat{\alpha}$$

2.2.2 Testing for Stationary

Generating a spread is easy; proving it is mean-reverting is difficult. For a pairs trade to be viable, the residual series ϵ_t must be stationary, meaning the spread fluctuates around a constant mean rather than drifting indefinitely.

To confirm this, we apply the Augmented Dickey-Fuller (ADF) test. While many traders simply look for a low p-value, understanding the underlying mechanism is crucial for avoiding false positives.

The ADF test evaluates whether the spread follows a "Unit Root" process (a random walk). This means:

1. If residuals follow the unit root process, it is not stationary, and shocks have permanent effects.

2. If residuals don't follow a unit root process, they are stationary.

Hence, if stationary, the residuals have constant mean and variance, confirming that cointegration exists between the assets. We estimate the following regression on the residuals ϵ_t:ϵ_t:

$$\Delta\epsilon_t = \alpha + \gamma\epsilon_{t-1} + \sum_{i=1}^{p} \delta_i \Delta\epsilon_{t-i} + u_t$$

There are two critical components here:

1. The Lagged Level ($\gamma\epsilon_{t-1}$): This is the heart of the test. We test the Null Hypothesis $H_0 : \gamma = 0$.

   - If $\gamma = 0$, the current change $\Delta\epsilon_t$ does not depend on the previous level. Shocks to the spread are permanent (the dog runs away).
   - If $\gamma < 0$, the spread corrects itself. The farther ϵ_{t-1} drifts from zero, the stronger the pull of $\Delta\epsilon_t$ in the opposite direction (the leash pulls back)

2. The Augmentation Terms ($\sum \delta_i \Delta\epsilon_{t-i}$): These lagged difference terms are added to account for serial correlation in the error term, ensuring the validity of the test statistics.

.

Crucially, under the null hypothesis of a unit root, the t-statistic for γ ($\hat{t}_\gamma$) does not follow the standard Student's t-distribution. Because the regressor ϵ_{t-1} is itself non-stationary under the null, the limiting distribution of the test statistic is the non-standard Dickey-Fuller distribution, which is skewed to the left. Consequently, standard critical values (e.g., -1.96 for 5% significance) are invalid and would lead to frequent false rejections (Type I errors). Instead, we must compare our calculated t-statistic against the specific critical values simulated and tabulated by MacKinnon (1996)1. If the t-statistic is more negative than these critical values (typically around -3.43 for a 1% significance level), we reject the null hypothesis, mathematically confirming that the relationship is not spurious and a true arbitrage opportunity exists [[3]].

2.2.3 Application of Engle-Granger Using Python

Having established the theoretical framework of the Engle-Granger two-step method, we now turn to a practical implementation using Python. The following script utilises the yfinance library for data acquisition and statsmodels to perform the cointegration tests. We specifically employ the coint function, which automates the regression and subsequent stationarity test on the residuals. This allows us to empirically validate the 'drunk and dog' hypothesis between our chosen pair, Goldman Sachs (GS) and Morgan Stanley (MS), to confirm if a statistically significant tether exists [[4]].

```python
# Necessary libraries
import yfinance as yf
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.tsa.stattools import coint


# Helper function to download and clean data
def get_prices(tickers, start_date='2021-01-01', end_date='2025-12-01'):
    data = yf.download(tickers, start=start_date, end=end_date)['Close']
    data = data.dropna() # Remove missing values
    return data

# ========================================
Engle-Granger Test (2 Stocks)
# ========================================

# Let's test two Banks: Goldman Sachs (GS) and Morgan Stanley (MS)
pair_tickers = ['GS', 'MS']
df_pair = get_prices(pair_tickers)

# The 'coint' function runs the Engle-Granger two-step process
# It returns the t-statistic, p-value, and critical values
score, pvalue, _ = coint(df_pair['GS'], df_pair['MS'])

print(f"Testing Pair: {pair_tickers}")
print(f"t-statistic: {score:.4f}")
print(f"p-value:     {pvalue:.4f}")
print(f"critical values:")
print(f"   1%: {crit_value[0]:.4f}")
print(f"  5%: {crit_value[1]:.4f}")
print(f" 10%: {crit_value[2]:.4f}")

if pvalue < 0.05:
    print(">> RESULT: Cointegration Detected! (The Leash is holding)")
else:
    print(">> RESULT: No Cointegration. (Relationship broken/spurious)")
```

Listing 2.1: Real-World Cointegration Verification

Results:

The empirical results indicate statistically significant cointegration between Goldman Sachs (GS) and Morgan Stanley (MS). The Augmented Dickey–Fuller test applied to the estimated spread yields a test statistic of –3.5678 and a p-value of 0.0268, allowing the null hypothesis of a unit root to be rejected at the 5% significance level. This suggests that the spread between the two assets is stationary and exhibits mean-reverting behaviour. From a trading perspective, this implies the existence of a stable long-run equilibrium linking the prices of GS and MS, consistent with their shared exposure to the investment banking sector. Deviations from this equilibrium are therefore expected to be temporary rather than persistent, providing a potential basis for relative-value strategies. However, while the result supports the presence of a tradable

```
Testing Pair: ['GS', 'MS']
t-statistic: -3.5678
p-value:     0.0268
critical values:
   1%: -3.9054
  5%: -3.3411
 10%: -3.0479
```

Figure 2.1: Values Found For GS and MS

relationship over the sample period, it also highlights the inherent concentration risk of pair-based approaches, reinforcing the motivation for extending the analysis to multi-asset baskets using more general cointegration frameworks.

2.3 Johansen Test for Basket Trading

While the Engle-Granger method is an excellent pedagogical tool for understanding the mechanics of mean reversion, it suffers from a significant limitation in practice: it is strictly bivariate. It can only assess the relationship between two variables at a time.

In modern high-frequency markets, betting on a single pair exposes the trader to substantial idiosyncratic risk. If "Asset A" suffers a catastrophic, company-specific event (e.g., an accounting scandal), the cointegration breaks, and the spread widens indefinitely. To mitigate this, institutional quants scale up to "Basket Trading," creating portfolios in which one asset is hedged against a weighted basket of peers.

To handle this complexity, we must graduate from simple regression to the Johansen Test.

2.3.1 From Regression to VECM

When dealing with a portfolio of three or more assets (e.g., trading Exxon Mobil against a basket of Chevron, BP, and Shell), we cannot simply regress one against the others without losing information about the dynamic interactions between them. The Johansen Test solves this by moving beyond simple regression into a Vector Error Correction Model (VECM) [[2]]. Instead of a single equation, we model the changes in the entire vector of prices Y_t simultaneously:

$$\Delta Y_t = \Pi Y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta Y_{t-i} + \epsilon_t$$

where:

- ΔY_t = change in the vector of prices Y_t
- Π = coefficient matrix of cointegrating relationships
- Γ = coefficient matrix of the lags of differenced variables of Y
- ϵ_t = error term

The critical component here is the matrix Π (Pi). In our previous "Drunk and the Dog" analogy, we only had one relationship to track. In a basket of N stocks, there might be multiple independent "leashes" binding them together. The matrix Π contains the information about these long-run relationships. The matrices Γ_i model the transient dynamics.

2.3.2 Eigenvalues and The Trace Statistic

The goal of the Johansen Test is to determine the Rank (r) of the matrix Π. The rank tells us exactly how many independent cointegrating relationships there are in our basket of stocks. If Rank (r) = 0: No cointegration exists. The assets are wandering randomly with no tether. If Rank (r) > 0: There are r stationary linear combinations.

To calculate this rank, we utilise Eigenvalues (λ). We treat the determination of the number of cointegrating vectors as a hypothesis testing problem using the Trace Statistic:

$$\lambda_{trace}(r) = -T \sum_{i=r+1}^{n} \ln(1 - \hat{\lambda}_i)$$

Intuitively, if the estimated eigenvalues ($\hat{\lambda}_i$) are significantly distinct from zero, it implies a strong cointegrating relationship. The Trace Statistic aggregates these values to test the null hypothesis that there are at most r cointegrating vectors.

2.3.3 Why This Matters: Robust Hedging

Understanding the rank allows us to construct a more resilient hedge. If we identify a basket of energy stocks with a rank of 1, we can find a single linear combination (a portfolio) that is stationary.

This allows us to trade the idiosyncratic noise of a single stock against the sector's systematic trend. Rather than betting that "Coke will outperform Pepsi" (a pair), we can bet that "Coke will revert to the mean of the entire Beverage Sector" (a basket). This diversification significantly reduces the risk of a single competitor's movement blowing up the trade.

2.3.4 Johansen Test Using Python:

While the Engle-Granger method suffices for pairs, analysing a portfolio requires a multivariate approach. We now implement the Johansen Test using the coint-johansen function from statsmodels to determine the rank of the cointegration matrix for a basket of assets. The code below fetches historical data for six major US banks and calculates the Trace Statistic. By comparing this statistic against critical values, we test the null hypothesis regarding the number of cointegrating relationships (r) present in the system, moving beyond simple pairs to a robust basket strategy [4]

```
# Necessary libraries
import yfinance as yf
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.tsa.stattools import coint_johansen


# Helper function to download and clean data
def get_prices(tickers, start_date='2021-01-01', end_date='2025-12-01'):
    data = yf.download(tickers, start=start_date, end=end_date)['Close']
    data = data.dropna() # Remove missing values
    return data

# ========================================
# Johansen Test (6 Stocks) (Could be as many as wanted)
# ========================================

# Let's test the six Banks that have the most impact in the USA:
basket_tickers = ['JPM', 'MS', 'GS','BAC','WFC','C']
df_basket = get_prices(basket_tickers)
```

```python

# Run Johansen Test
# det_order=0 implies we assume there is a constant term in the model
johansen_res = coint_johansen(df_basket, det_order=0, k_ar_diff=1)

# There are N (number of stocks) possible relationships (r=0, r=1, r=2...)
for r in range(len(basket_tickers)):
    trace_stat = johansen_res.lr1[r]       # The test statistic
    crit_val = johansen_res.cvt[r, 1]      # The 5% critical value (index 1)

    # Logic: If Trace Stat > Critical Value, we REJECT the null hypothesis
    if trace_stat > crit_val:
        res = f"Reject H0 (At least {r+1} relationships exist)"
    else:
        res = f"Cannot Reject H0 (Likely only {r} relationships)"

    print(f"r <= {r}    | {trace_stat:.4f}    | {crit_val:.4f}             | {
    res}")

# Eigenvalues help determine the strength of the relationship
print("\nEigenvalues of the matrix (Strength of Cointegration):")
print(johansen_res.eig)

print("\nINTERPRETATION:")
print("If we reject r<=0, it means there is at least 1 cointegrating
    relationship.")
print("If we reject r<=1, it means there are at least 2, and so on.")
```

Listing 2.2: Real-World Cointegration Verification Using Johansen Test

```
r <= 0    | 89.1784    | 95.7542             | Cannot Reject H0 (Likely only 0 relationships)
r <= 1    | 53.3018    | 69.8189             | Cannot Reject H0 (Likely only 1 relationships)
r <= 2    | 28.0519    | 47.8545             | Cannot Reject H0 (Likely only 2 relationships)
r <= 3    | 15.5642    | 29.7961             | Cannot Reject H0 (Likely only 3 relationships)
r <= 4    | 7.3227     | 15.4943             | Cannot Reject H0 (Likely only 4 relationships)
r <= 5    | 0.0275     | 3.8415              | Cannot Reject H0 (Likely only 5 relationships)

Eigenvalues of the matrix (Strength of Cointegration):
[2.38856961e-02 1.68708286e-02 8.37958620e-03 5.53817105e-03
 4.90384199e-03 1.85277308e-05]

INTERPRETATION:
If we reject r<=0, it means there is at least 1 cointegrating relationship.
If we reject r<=1, it means there are at least 2, and so on.
```

Figure 2.2: Results for Johansen Test

The Johansen Test applied to a basket of six major US banks (JPM, MS, GS, BAC, WFC, C) failed to identify a statistically significant cointegrating relationship. Specifically, in the test for rank $r \leq 0$ (the null hypothesis that zero cointegrating vectors exist), the calculated Trace Statistic of 89.18 fell below the 95% Critical Value of 95.75. Consequently, we are unable to reject the null hypothesis. This statistical result is further corroborated by the near-zero eigenvalues, indicating that any potential mean-reverting signal is negligible. Economically, this implies that while these firms share the same sector, their individual price paths during this specific timeframe were driven closer by idiosyncratic risks rather than a shared fundamental tether. Therefore, a stationarity-based arbitrage strategy cannot be reliably constructed for this specific six-stock basket without modification.

2.4 From Math to Markets

Theory is useless without execution. Having mathematically confirmed a cointegrating relationship using the ADF or Johansen tests, we must now transform these statistical outputs into actionable trading signals.

2.4.1 Signal Generation: The Z-Score

The raw value of the spread (ϵ_t) is difficult to trade directly because its magnitude varies based on the price of the underlying assets. A spread of \$0.50 might be huge for a penny stock pair but negligible for a pair of tech giants. To standardise our signals, we normalise the spread into a Z-score. This tells us how many standard deviations the current spread is from its historical mean. [[5]]

$$Z_t = \frac{\epsilon_t - \mu_\epsilon}{\sigma_\epsilon}$$

Where:

- ϵ_t is the current spread value.
- μ_ϵ is the rolling mean of the spread.
- σ_ϵ is the rolling standard deviation.

This normalisation allows us to define systematic entry and exit rules:

- Entry: When $|Z_t| > 2$ (or another threshold like 1.5 or 3), the spread has stretched significantly. If $Z_t > 2$, the spread is "too high" (Asset A is overvalued relative to B), so we Short A / Long B. If $Z_t < -2$, we do the reverse.
- Exit: When Z_t returns to 0, the relationship has reverted to the mean, and we close the position to capture the profit.

2.4.2 Timing the Trade: The Ornstein-Uhlenbeck Process

A critical question in statistical arbitrage is, "How long will I be stuck in this trade?" If a mean reversion takes three years, the capital lockup makes the trade unviable. To estimate the expected holding period, we model the spread as an Ornstein-Uhlenbeck (OU) process, a stochastic process that describes mean-reverting behaviour:

$$d\epsilon_t = \theta(\mu - \epsilon_t)dt + \sigma dW_t$$

Where:

- θ (theta) represents the speed of mean reversion.
- μ represents the long-run mean of the error term
- σ represents the long-run volatility of the error term
- dW_t represents a change in a Wiener process (Brownian motion)

By fitting this differential equation to our residual series, we can calculate the Half-Life of the trade—the time expected for the spread to revert halfway back to its mean.

$$\text{Half Life} = \frac{\ln(2)}{\theta}$$

If the calculated Half-Life is extremely long (e.g., months), the "leash" is too loose, and the pair should likely be discarded in favour of faster-moving opportunities.

2.4.3 Risk Management: Structural Breaks

Finally, we must address the primary killer of stat-arb strategies: Structural Breaks. Cointegration is not a permanent law of physics; it is an economic observation that can change. A merger, a regulatory change, or a shift in a company's business model can permanently snap the "leash". If the spread widens beyond, say, 4 or 5 standard deviations ($Z > 5$), it is often statistically more likely that the relationship has broken rather than extended. In these cases, a hard stop-loss is essential to prevent betting on a mean reversion that will never come.

2.5 Application Using WorldQuant's Brain platform

Finally, we bridge the gap between academic theory and industry practice. While Python is an excellent environment for researching specific pairs, it is computationally heavy to scale across thousands of assets. Institutional platforms like WorldQuant Brain are designed for high-throughput testing, allowing us to generalise our cointegration logic across the entire market universe.
In this section, we will translate the mathematical formulas from Sections 1 and 2 into WorldQuant's expression-based syntax to build a mean-reversion "Alpha".

2.5.1 Translating the Math: Constructing the Signal

The trading strategy is evaluated using WorldQuant's BRAIN platform, which backtests alphas by mapping them into a systematic long–short portfolio over a predefined universe. In this framework, the strategy's signal determines cross-sectional position weights, with assets exhibiting stronger signals placed in the long book and weaker signals in the short book. The resulting portfolio is approximately dollar-neutral, so gains and losses reflect relative-value performance rather than overall market movements. This backtesting setup is well aligned with cointegration-based strategies, as it isolates the mean-reversion component of the spread while minimising exposure to broad market risk.
The alpha for this strategy is

$$-\,\text{ts\_zscore}(\text{ts\_sum}(\text{returns} - \text{group\_mean}(\text{returns}, 1, \text{sector}), 60), 250).$$

The strategy first removes sector-wide effects by subtracting the sector mean return from each asset's return, thereby isolating relative price deviations within economically related groups. These sector-adjusted returns are then aggregated over a 60-day window to capture persistent departures from a local equilibrium. The resulting series is normalised using a 250-day time-series z-score, expressing deviations relative to their historical variability. The negative sign induces a contrarian positioning, allocating long positions to assets that have underperformed their sector and short positions to those that have outperformed. Interpreted through the lens of statistical arbitrage, the signal exploits mean reversion in relative prices while remaining largely insulated from broader market and sector trends.

2.5.2 Results

The backtest results indicate that the strategy captures a persistent mean-reversion effect when implemented within a systematic long–short portfolio. The cumulative PnL exhibits a stable upward trend, consistent with repeatedly trading relative price deviations rather than directional market movements. Mathematically, this behaviour aligns with the interpretation of the signal as exploiting approximately stationary, sector-adjusted spreads.

The observed turnover of roughly 12% reflects the medium-horizon structure imposed by the 60-day aggregation and 250-day normalisation windows, indicating that the strategy responds to sustained mispricings rather than short-term noise. The moderate Sharpe ratio should be interpreted in the context of the dollar-neutral long–short framework, where returns are generated

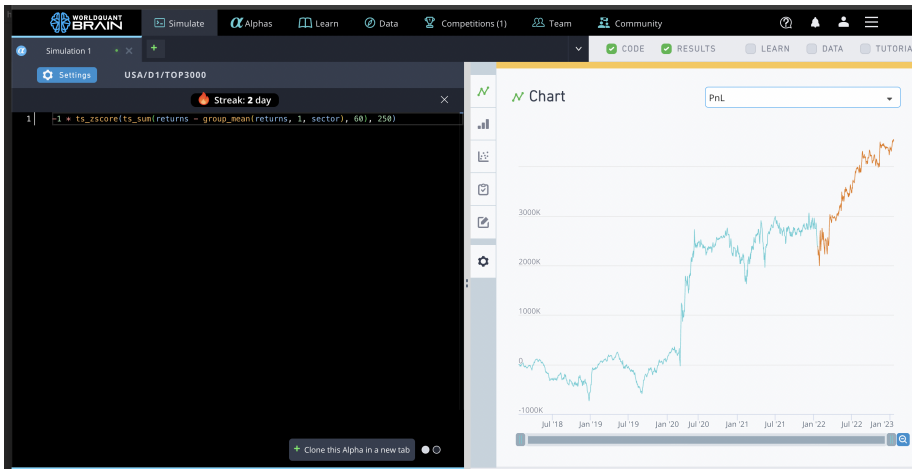


Figure 2.3: Brain platform results

solely from cross-sectional dispersion. By construction, this limits absolute risk-adjusted performance while enhancing robustness across market regimes. Overall, the results are consistent with a risk-controlled statistical arbitrage strategy designed to extract mean reversion in relative price relationships.

It is important to note that due to computational constraints on high-frequency platforms, this Alpha implementation relies on a simplified sector-neutral mean reversion approach rather than a full VECM estimation. While strictly distinct from cointegration, this method acts as a computationally efficient proxy by assuming the sector index acts as the 'integrating' tether for the individual stock.

2.6 Conclusion

This exploration has traversed the critical distinction between correlation and cointegration, moving from the intuitive "Drunk and the Dog" analogy to the rigorous mechanics of the Engle-Granger and Johansen tests. We established that while bivariate pairs trading offers a pedagogical foundation, modern statistical arbitrage requires the robustness of multi-asset baskets to mitigate idiosyncratic risk. By translating abstract econometric concepts such as stationarity and the Ornstein-Uhlenbeck process into actionable Z-score signals on platforms like WorldQuant Brain, we bridged the gap between theoretical math and practical execution. Ultimately, the durability of a quantitative strategy lies not in predicting random walks but in identifying and capitalising on the fundamental economic tethers that ensure mean reversion in a chaotic market.

Bibliography

[1] Gianna Boero. Cointegration – lecture notes. University of Warwick, n.d.. URL https://warwick.ac.uk/fac/soc/economics/staff/gboero/personal/hand2_cointeg.pdf. Accessed: 2025.

[2] Gianna Boero. Vector error correction models (vecm) – lecture notes. University of Warwick, n.d.. URL https://warwick.ac.uk/fac/soc/economics/staff/gboero/personal/hand3_vecm.pdf. Accessed: 2025.

[3] James G MacKinnon. Numerical distribution functions for unit root and cointegration tests. *Journal of Applied Econometrics*, 11(6):601–618, 1996. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291099-1255%28199611%2911%3A6%3C601%3A%3AAID-JAE417%3E3.0.CO%3B2-T.

[4] Cemal Ozturk. Unveiling cointegration: Johansen test explained with python examples, 2023. URL https://medium.com/@cemalozturk/unveiling-cointegration-johansen-test-explained-with-python-examples-db8385219f1f. Online Article.

[5] QuantStart Team. Basics of statistical mean reversion testing. QuantStart, n.d. URL https://www.quantstart.com/articles/Basics-of-Statistical-Mean-Reversion-Testing/. Accessed: 2025.

Chapter 3

Physics-Informed Neural Networks as Financial PDE solvers

Etiene Inyang

Edited by: Valerio Altissimo

Abstract: This article studies Physics-Informed Neural Networks (PINNs) as constrained function approximation methods for solving partial differential equations. Asset pricing under no-arbitrage leads to parabolic PDEs such as the Black–Scholes equation, which are traditionally solved using grid-based numerical methods or Monte Carlo simulations. However, these classical approaches suffer from scalability issues as the dimension of the state space increases or computation issues due to slow convergence of paths. We present PINNs as mesh-free residual minimisation schemes that approximate pricing functions directly while enforcing the financial PDE as a constraint. Focussing on the Black–Scholes framework, we develop the mathematical formulation of PINNs, analyse their approximation properties, and discuss their advantages and limitations relative to Finite Differences and Monte Carlo methods. Extensions to inverse problems, such as volatility calibration, are also discussed.

3.1 Introduction

Modern quantitative finance is defined by a persistent tension between model complexity and execution speed. As financial instruments become increasingly sophisticated, the mathematical models required to price them and hedge their risks grow in dimensionality: this complexity can push traditional numerical solvers to their limits.

The subject of derivative pricing in particular is fundamentally a problem of valuation under uncertainty subject to the principle of no-arbitrage. The Black–Scholes equation is the most prominent model for option pricing and serves as a foundational model in both theory and practice. The parabolic PDE is as follows:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

where,

V is the option value

S is the value of the underlying asset

σ is the volatility

t is the current time and T is the expiration date, so that $0 \leq t \leq T$

r is the risk-free interest rate

While closed-form solutions are easy to obtain for simple European options, the case of more complex instruments such as American options and Exotic derivatives requires the usage of numerical approximations which may not entail analytic solutions. Two notable examples of such numerical approximation methods are Finite Differences and Monte Carlo simulations.

The Finite Differences method operates by discretising the (S, t) domain into a mesh and approximating derivatives through local grid points. For the Black–Scholes equation, explicit, implicit and Crank–Nicolson schemes are commonly used. Although this works for low-dimensional problems, the number of grid points grows exponentially with the dimension of the state space (i.e., the number of underlying assets). This computational bottleneck is known as the "Curse of Dimensionality," [1]. On the other hand, we have the Monte Carlo method, which estimates prices by simulating thousands of potential market paths. Although their convergence rate is dimension-independent, they can be computationally intensive [9] (particularly when dealing with Greeks) and struggle with the early-exercise characteristic inherent in American options.

Physics-Informed Neural Networks (PINNs) provide a mesh-free alternative by treating the option price not as a set of points on a grid but as a continuous function learnt by a deep neural network. In this framework, "physics" denotes the governing mathematical constraints of finance: the fundamental laws of no-arbitrage and market equilibrium in particular. It should, however, be noted that although the 'Physics-Informed' nomenclature originates from applications in fluid dynamics and solid mechanics (conservation of mass/energy in particular), this specific adaptation to financial mathematics is often categorised under the emerging sub-genre of Finance-Informed Neural Networks. By embedding financial PDEs directly into the network's training objective, we ensure that the learnt solution is economically consistent and transform the pricing problem from a grid-based discretisation task into an optimisation problem. Such an approach contrasts the "black-box" empiricism of standard neural networks with PINNs which utilise deep neural networks as universal function approximators, capable of representing the complex, non-linear mappings inherent in (financial) PDEs [**?**].

3.2 Neural Network Foundations

PINNs are a type of Feedforward Neural Network (FNN). Loosely inspired by the biological neurones of the human brain, FNNs are best viewed as mathematical engines that seek statistical generalisation [4]. These "quintessential" deep learning models operate as universal function approximators: they are structured as a chain of functions where information flows from an input $\mathbf{x}$ (such as spatial coordinates and/or time) through various hidden layers to an output $\mathbf{y}$ (such as temperature and/or velocity) by learning a set of optimal parameters.

Such a chain may be composed of three functions $f^{(1)}$, $f^{(2)}$ and $f^{(3)}$ where $f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x})))$. With each function $f^{(i)}$ indicating a 'layer', the number of these layers defines the 'depth' of the model, hence the name "deep learning". Although the final 'output layer' is constrained by training data or, in the case of PINNs, physical boundary conditions, the intermediate 'hidden layers' are free to develop the internal representations required to minimise the error of the model.

Each function $f^{(i)}$ represents a linear transformation in a particular hidden 'units' part of a larger hidden layer: for a given input vector $\mathbf{x}$ the pre-activation value $\mathbf{z}$ is given by the weighted sum of its inputs, plus a bias term:

$$z = \sum_{i=1}^{n} w_i x_i + b$$

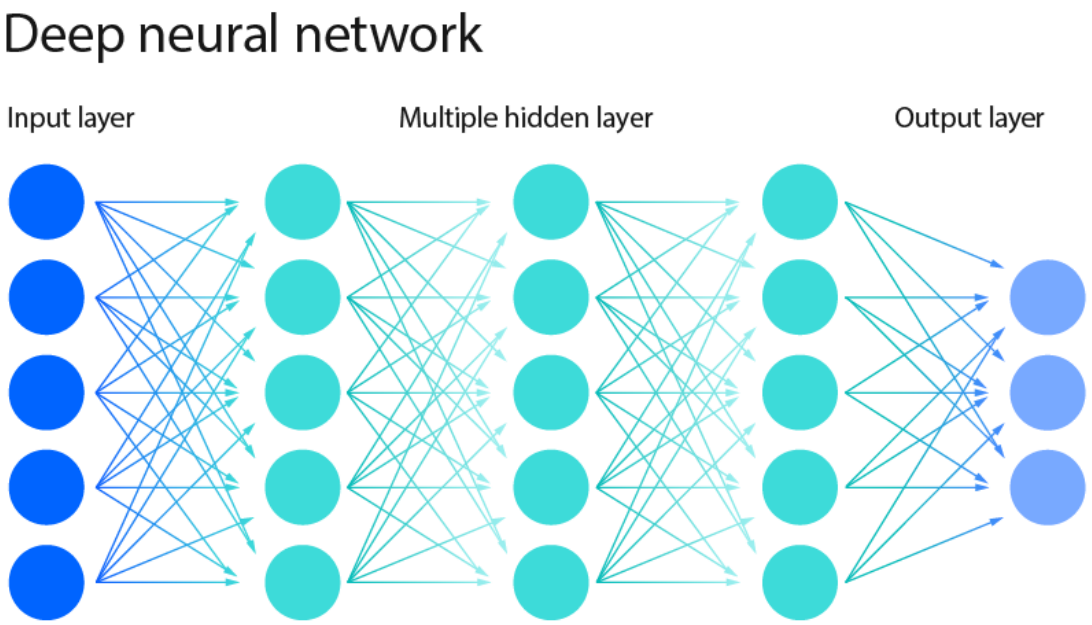


Figure 3.1: A Deep Learning Neural Network taken from here

where,

x_i is the input feature

w_i is the weight

b is the bias

Then to introduce the non-linearity necessary for the model to approximate complex surfaces, we apply an activation function, $\sigma(z)$, determined by our choice of hidden unit e.g., the Rectified Linear Unit (ReLU), the hyperbolic tangent (tanh) or sigmoid. The final output is then:

$$a = \sigma(z)$$

It can be intuitive to think of hidden units as musicians in a large orchestra with unique, simple instruments and a limited range of notes (their activation function). A hidden layer is similar to a section (e.g. strings, woodwinds), and the training process is the conductor (the optimisation algorithm) learning the score (the PDE solution). Initially, the musicians play randomly, but the conductor's goal is not to simply match a few recorded melodies (the data points) but to ensure that the entire piece obeys the composer's rules of harmony and rhythm (the PDE). By continually adjusting each musician's volume (the weight) and timing (the bias), the conductor blends their simple individual contributions into a complex, harmonious whole that satisfies all the constraints.

Neural networks learn by iteratively adjusting their weights and biases to minimise the difference between their predictions and the reality. This process begins with a forward pass, where the input data travels through the layers. Once a prediction is made in the output layer, a loss function calculates the error by comparing that prediction to the actual target. To correct this error, the network performs 'backpropagation', using the chain rule of calculus to work backward from the output and determine how much each individual weight contributed to the mistake. Finally, an optimisation algorithm like gradient descent slightly updates those weights

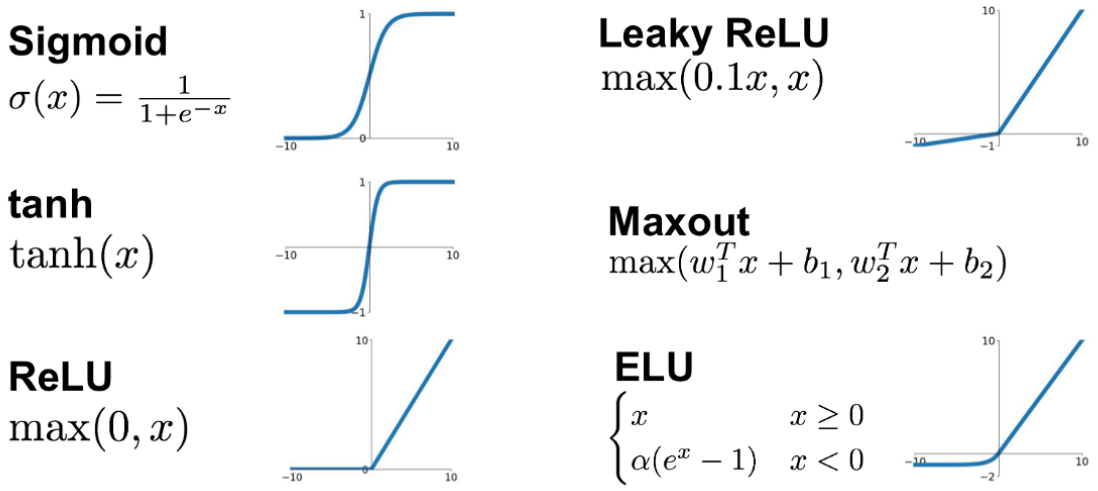


Figure 3.2: Different activation functions taken from here

in the direction that reduces the error, repeating this entire cycle until the model achieves high accuracy.

The choice of the activation function σ is critical in the context of the Black–Scholes equation, since the PDE has second-order derivatives. Thus, the activation function must be twice differentiable and non-vanishing. Although the ReLU is usually considered the go-to option for FNNs, the rectified linear function, $g = \max\{0, z\}$, is not differentiable at $z = 0$. In practice, gradient descent still works because we rarely hit exactly 0.000000... on a digital computer, and software usually returns the derivative from one side (either 0 or 1), which is "good enough" for the optimisation to converge [4]. In general, however, functions like tanh are preferred. There is still substantial research into hidden units, and the process of determining which one is most suited for an application is largely that of trial and error.

Before even applying a PINN to the Black–Scholes PDE we must ask, "can a neural network actually represent the complex, nonlinear surface of an option pricing model?". In fact, one of the strengths of FNNs in general are their ability to approximate non-linear functions as well as linear ones: this result is the subject of the 'Universal Approximation Theorem' (UAT). Hornik et al 1989 [7] and Cybenko 1989 [3], showed that feedforward networks with only 1 "squashing" activation function and at least one hidden layer are capable of arbitrarily accurate approximations to any real-valued continuous function over a compact set, provided that the network is given enough hidden units (in this context, a "squashing" activation function is one which maps inputs from an infinite range (e.g. $(-\infty, \infty)$ to a finite interval (e.g. $[0, 1]$).

For our application we wish to approximate the solution to the Black–Scholes PDE $V(S, t)$ and the UAT guarantees the existence of a neural network that can represent this function which is a surface. It should also be noted that the derivatives of the network can also be approximated arbitrarily well [6]. This is of importance, as Greeks, such as Delta ($\Delta = \frac{\partial V}{\partial S}$) and Gamma ($\Gamma = \frac{\partial^2 V}{\partial S^2}$) are part of what defines the theory behind the PDE.

Although the UAT makes a claim on existence, it does not specify the size of the layer required to guarantee an arbitrary level of accuracy. Crucially, Ian Goodfellow et al. [4] writes that, "In many circumstances, using deeper models can reduce the number of units required to represent the desired function and can reduce the amount of generalisation error", which justifies the use of deeper architectures in contrast to the single-layer approach suggested by the seminal papers.

Furthermore, it is important to note that the UAT only makes a claim on existence: it does not guarantee that the neural network will be able to learn the function it is trying to approximate correctly. This means that there exists a specific configuration of weights and biases that can

mimic the target function, but the optimisation algorithm (gradient descent) may never reach it – this could be due to overfitting, for example. It is like having all the necessary Lego bricks for a structure, but no exact manual to help.

PINNs also require a way to calculate the derivatives of that function to enforce the PDE – this is the role of Automatic Differentiation.

3.3 Automatic Differentiation

There are three major ways to compute the derivative of a function f': Numerical Differentiation, Symbolic Differentiation and Automatic Differentiation (the first two being very familiar already). Numerical Differentiation relies on the definition of the derivative, i.e. $f'(x) = \lim_{\delta \to 0} \frac{f(x+\delta)-f(x)}{\delta}$, whilst Symbolic Differentiation involves using mathematical expressions to produce an exact formula for the derivative, e.g. $\frac{d}{dx}(3x^2) = 6x$.

However, PINNs leverage Automatic Differentiation (AD) to compute the derivatives of the neural network's output with respect to its input coordinates (time and asset price). Unlike numerical differentiation (which suffers from truncation and rounding errors) or symbolic differentiation (which can lead to large and inefficient derivative expressions), AD decomposes complex functions into elementary operations to compute exact derivatives at machine precision

AD evaluates derivatives by applying the chain rule to the elementary operations of the 'computational graph': this is used to represent the mathematical expressions present in the neural network. Each node denotes a variable, and the edges show the flow of data.

Reverse-Mode AD (Backpropagation) is the mode for training deep learning models. It is most efficient when a function has many inputs (n) but few outputs (m), as it computes the gradient ∇f in a single backward pass. In PINNs, it is primarily used to update the weights θ relative to the loss function.

However, as an aside, it should be noted that the training process requires the derivatives of the loss with respect to the network weights. This suggests the use of what is known as 'Forward-Mode AD', which is efficient when the number of outputs is large relative to inputs [**?**]. For the Black–Scholes equation, where we differentiate the scalar output V with respect to only two inputs (S, t), Forward Mode is theoretically superior since it avoids the memory overhead of storing intermediate states required for a backward pass by carrying "dual numbers" through the network simultaneously with the primal calculation.

The Black–Scholes PDE also requires the second-order derivative $\frac{\partial^2 V}{\partial S^2}$ (Gamma). In standard frameworks like PyTorch, this is typically handled via 'Reverse-over-Reverse' differentiation. It involves a forward pass through the neural network, in which we compute V and build Graph 1. A backward pass (backpropagation) in which we compute $\Delta = \frac{\partial V}{\partial S}$ by traversing Graph 1 and building Graph 2). And then a second backward pass, where we finally compute $\Gamma = \frac{\partial^2 V}{\partial S^2}$ while traversing Graph 2. This nested backpropagation is memory-intensive because the computational graph grows linearly with the depth of the network and the order of differentiation. Consequently, the graph for the second derivative is effectively twice the size of the original network graph.

To mitigate the "double backpropagation" bottleneck, emerging research suggests Taylor-mode AD as a potential alternative: this method propagates truncated Taylor series coefficients in a single pass, allowing for the simultaneous evaluation of higher-order derivatives [2].

A distinct advantage of the PINN framework over Finite Differences or Monte Carlo methods is the extraction of the "Greeks." In traditional FD methods, Greeks are approximated via "bumping" (re-running the solver with perturbed inputs), which introduces a "step size dilemma" – balancing truncation error against subtractive cancellation. In Monte Carlo methods, pathwise derivatives can be used where the option payoff function is differentiated with respect to the parameters along each simulated price path – but this requires the payoff function to be differentiable (i.e., smooth). In a PINN, once the network $V(S, t)$ is trained, the Greeks are not

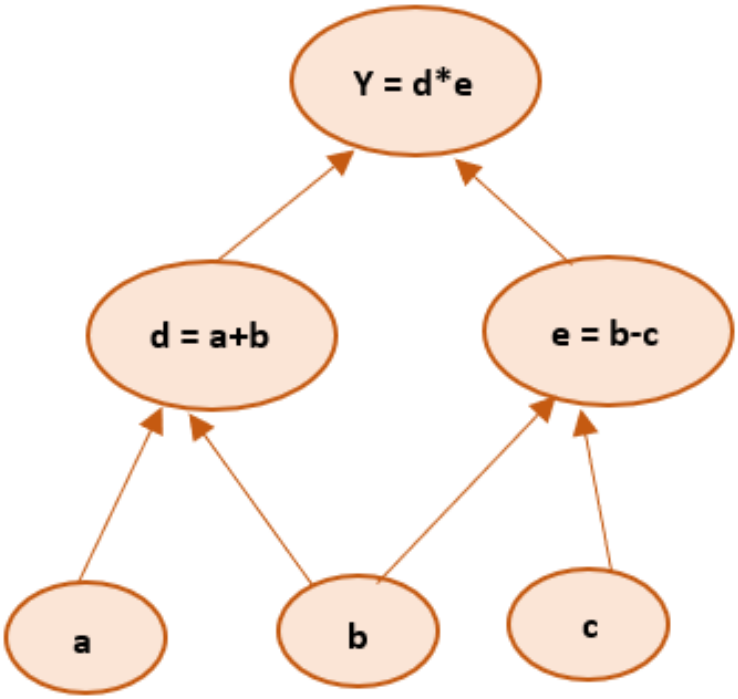


Figure 3.3: A basic computational graph taken from here

approximations, but exact analytical properties of the network. Because the PINN is trained to satisfy the PDE globally, these Greeks are often smoother and more consistent than those obtained from local approximations, particularly near payoff kinks or boundaries [10] [11] .

3.4 The PINN framework

Another core aspect of PINNs is the construction of a composite loss function that penalises deviations from both the observed data and the governing PDE [**?**]. The PINN residual for the Black–Scholes PDE is simply:

$$\mathcal{R} = \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

It is a measure of how much the neural network's current prediction fails to satisfy the Black–Scholes equation – if the network perfectly solves the PDE, the residual at that point is zero. PINNs utilise a set of 'collocation points': these are unstructured (meaning that they do not need to be evenly spaced) coordinates (S_i, t_i) where the PDE residual is evaluated. PINNs function as global collocation methods where the residual is enforced to be small in aggregate across these points. Because these points do not require fixed connectivity or a predefined mesh, instead relying on sampled points, PINNs can effectively combat the "Curse of Dimensionality" as they scale more efficiently to high-dimensional problems. The PDE loss is then calculated by summing the squared residuals at N_f points sampled from the domain.

The distribution of collocation points is critical to ensure the network learns the pricing surface accurately. Rather than simple random sampling, advanced techniques are used to ensure the domain is "well-covered" such as Latin Hypercube Sampling.

The number of collocation points (N_f) effectively defines the 'resolution' of the physical constraint. If N_f is too low, the network may fail to capture the curvature (Gamma) of the

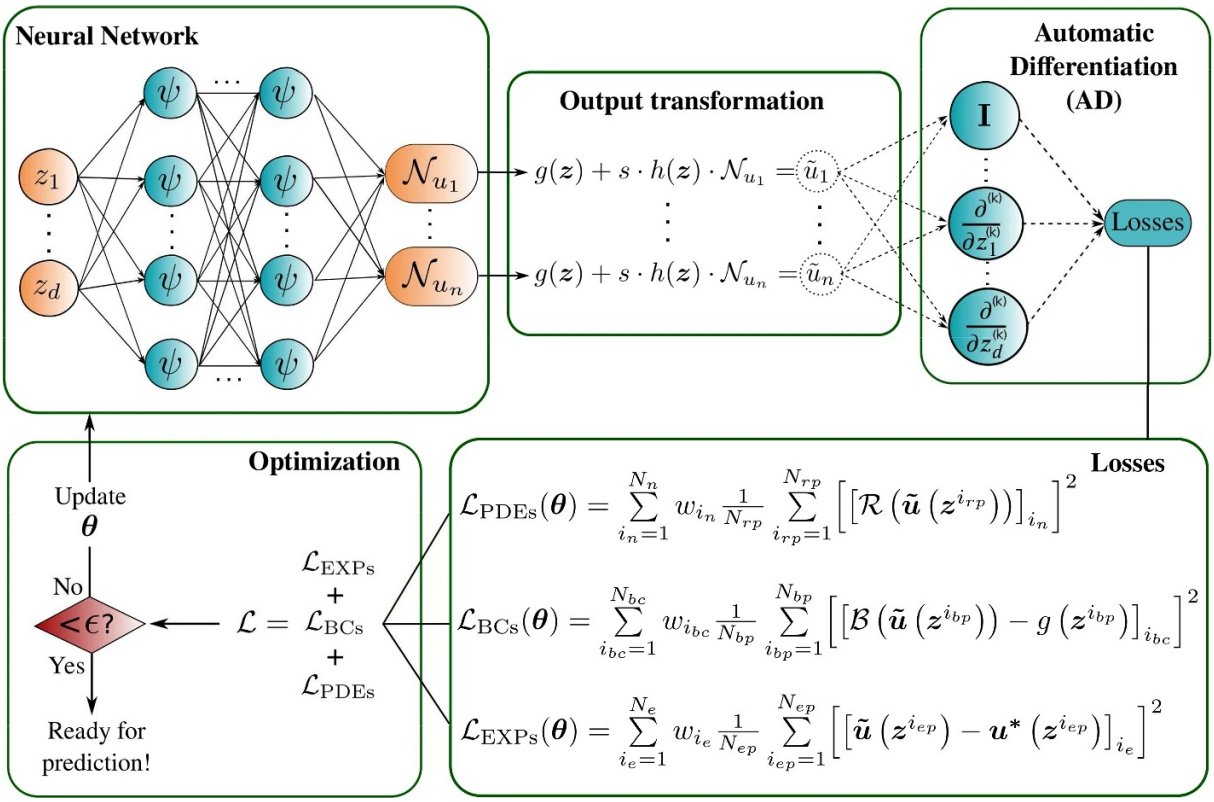


Figure 3.4: General representation of a physics-informed neural network for a boundary value problem taken from here. The diagram illustrates the four primary stages of the PINN framework. Namely: the forward pass, output transformation, automatic differentiation and loss optimisation.

option price correctly; if it is too high, the computational cost of Automatic Differentiation (AD) increases significantly. The loss is minimised by adjusting the network parameters θ until the PDE residual at these specific points effectively vanishes.

The PINN is then trained by minimising the mean squared error loss:

$$MSE = MSE_{PDE} + MSE_{Asymptotic} + MSE_{Payoff} \tag{3.1}$$

$MSE_{residual}$ ensures that the network's output satisfies the PDE at a set of randomly sampled collocation points within the domain. $MSE_{Asymptotic}$ and MSE_{Payoff} ensure that the respective boundary and terminal conditions are met, such as the option's behaviour as the asset price approaches zero and its payoff at expiration. The total loss enforces the PDE, boundary, and terminal conditions simultaneously. By minimising this loss, the network is regularised to find a solution that is both consistent with market data and theoretically sound.

This is illustrated in the Optimisation and Losses sections of Figure 4. MSE corresponds to $\mathcal{L}$, the total loss. $\mathcal{L}_{\text{PDEs}}$ corresponds to MSE_{PDE}. It calculates the squared residual $\mathcal{R}$ of the Black-Scholes equation at N_{rp} collocation points. $\mathcal{L}_{\text{BCs}}$ corresponds to $MSE_{Asymptotic}$. $\mathcal{L}_{\text{EXPs}}$ corresponds to MSE_{Payoff}. This data-driven term minimises the mean squared error between the network's prediction $\tilde{u}$ and the actual observed market prices u^* at N_{ep} data points. The other symbols present in the figure are as follows:

θ are the weights and biases (learnable parameters) of the network.

z are the input features S, asset price, and t, time till maturity.

z^{*p} are the collocation points sampled for each of the three losses.

w_i are adaptive weights used during training.

From a numerical analysis viewpoint, PINNs can be interpreted as global collocation methods: the residual is evaluated at collocation points in the domain and enforced to be small in an aggregate norm [**?**].

3.5 Inverse Problems and Data-Driven Discovery

A persistent challenge in quantitative finance is that, while market prices for options are easily observable, the underlying parameters, such as volatility σ, are latent variables. An Implied Volatility Surface (IVS) is a useful three-dimensional plot that shows how the market's expectation of future volatility varies across different strike prices and expiration dates for a single underlying asset. Calibration of IVS is a classically ill-posed inverse problem: multiple volatility configurations can often produce the same market price, and noisy or sparse data can lead to unstable results in traditional models. PINNs are uniquely suited for this task, transforming the calibration process from a search for a single value into a data-driven identification of the underlying governing dynamics.

In the inverse setup, volatility is treated as a learnable parameter or a functional output within the neural network's architecture. To ensure that this parameter accurately reflects market reality without violating financial laws, the PINN must balance competing objectives. Recent research by Hoshisashi et al. [8] introduced the "Whack-a-mole Learning" (WamL) framework to solve the "gradient pathology" where different loss terms (data vs. physics) conflict during training. WamL automatically adjusts the weights of the loss terms. If the model fits the market data but begins to violate the Black–Scholes PDE, the algorithm "whacks" the physics residual by increasing its penalty. Beyond the PDE itself, the framework enforces no-arbitrage conditions represented as differential inequalities. This ensures that the discovered IVS is not only mathematically accurate, but financially viable.

Although the canonical Black–Scholes model assumes constant volatility, PINNs are increasingly used to infer more complex, time-varying functions. For example, research by Hainaut and Casas [5] demonstrates that PINNs can replace computationally intensive standard pricing methods in Heston-type models. By treating the volatility process as an additional dimension, PINNs can value European options across a vast range of parameters without requiring retraining, offering a significant efficiency gain for real-time parameter estimation.

3.6 Limitations

While standard PINNs excel at smooth approximations, they suffer from spectral bias, where the network prioritises low-frequency components of the solution. In the context of Black–Scholes, this manifests itself as a failure to capture the "sharp" non-differentiable payoff at $t = T$ (e.g., the kink in a European Call). Furthermore, PINNs often struggle with "temporal failure," where errors from the initial or boundary conditions propagate throughout the domain because the network tries to solve all time points simultaneously. Emerging architectures such as xLSTM-PINNs [12] treat the PDE as a time-marching problem, using memory-gated units to maintain the sharp features of the payoff across the time horizon.

The loss function of a PINN is a multi-objective optimisation problem consisting of multiple elements whose terms often have conflicting gradients, leading to "stiff" optimisation landscapes where the network converges to a trivial solution (e.g., the zero function). Gradient-enhanced PINNs (gPINNs) and Adaptive Weighting methods mitigate this by embedding the gradient of the residual into the loss function or dynamically rescaling the loss terms during training to prevent one term from dominating the others [13].

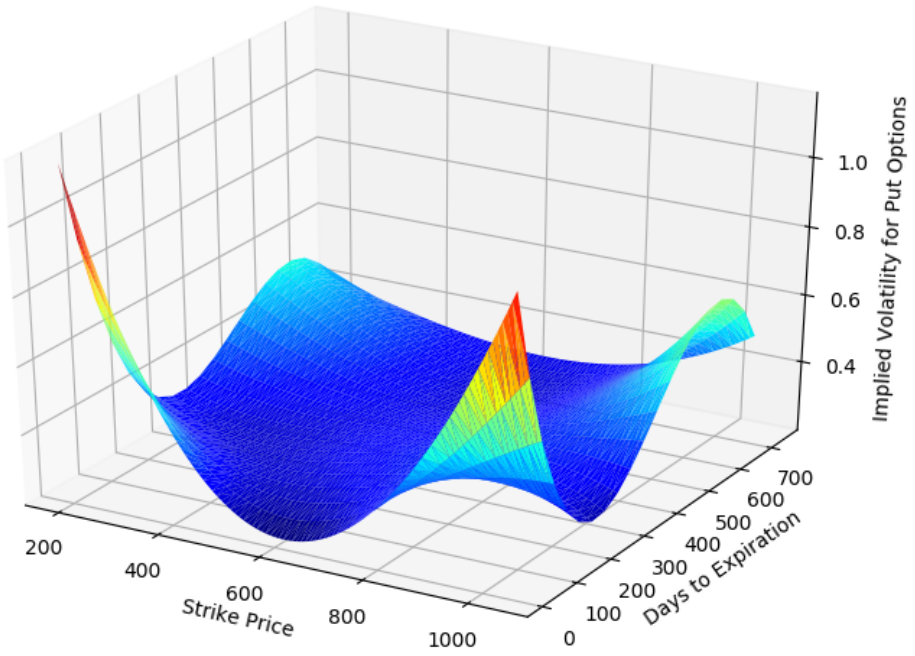


Figure 3.5: Example implied volatility surface taken from here.

3.7 Conlcusion

In conclusion, Physics-Informed Neural Networks offer a promising alternative for solving financial PDEs such as the Black–Scholes equation. They provide a compelling solution to the scalability issues inherent in traditional grid-based and simulation-based pricing methods. Even though challenges remain in optimisation and convergence, the mesh-free nature of PINNs provides a useful tool for the next generation of financial modelling, particularly in high-dimensional and inverse problem domains.

Bibliography

[1] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[2] J. Bettencourt, M. J. Johnson, and D. Duvenaud. Taylor-mode automatic differentiation for higher-order derivatives. In *NeurIPS 2019 Workshop on Program Transformations for Machine Learning*, 2019.

[3] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.

[4] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[5] D. Hainaut and A. Casas. Option pricing in the Heston model with physics inspired neural networks. *Annals of Finance*, 20(3):353–376, 2024.

[6] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[7] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[8] K. Hoshisashi, C. E. Phelan, and P. Barucca. Whack-a-mole learning: physics-informed deep calibration for implied volatility surface. In *2024 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics*, pages 1–8. Institute of Electrical and Electronics Engineers, 2024.

[9] J. C. Hull. *Options, Futures, and Other Derivatives, Global Edition*. Pearson Higher Ed, 2021.

[10] C. Nwankwo, N. Umeorah, T. Ware, and W. Dai. Deep learning and American options via free boundary framework. *Computational Economics*, 64(2):979–1022, 2024.

[11] S. Park, M. Kyoung-Sook, and K. Hongjoong. Asian option pricing using the physics-informed neural networks method. *Economic Computation and Economic Cybernetics Studies and Research*, 59:5–20, 2025.

[12] Z. Tao, D. Zhao, F. Liu, K. Xu, and X. Hu. xLSTM-PINN: memory-gated spectral remodeling for physics-informed learning. arXiv preprint arXiv:2511.12512, 2025.

[13] J. Yu, L. Lu, C. Wohlever, and G. E. Karniadakis. Gradient-enhanced physics-informed neural networks for solving forward and inverse PDE problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022.

Chapter 4

Interest Rate Parity as a Pricing Identity, Not a Forecast

Alberto Álvarez and Javier Fernández

Edited by: Arsh Mir

Abstract: This article examines the predictive interpretation of interest rate parity in foreign exchange markets. While parity conditions arise as no-arbitrage pricing identities, they are frequently misused as forecasting tools for future exchange rate movements. Using a deliberately restrictive empirical framework, the analysis evaluates the ability of parity-implied forward rates to predict realised spot exchange rates and revisits the forward premium puzzle from a structural perspective.

4.1 Introduction: Arbitrage, Parity, and the Promise of Predictability

Arbitrage plays a fundamental role in international finance (Duffie, 2001; Cochrane, 2005). At its core, arbitrage refers to the existence of a trading strategy that delivers a risk-free profit with zero net investment. The absence of such opportunities is not merely a simplifying assumption, but a structural requirement for equilibrium in competitive financial markets. Asset pricing theory, yield curve construction, and derivative valuation all rely on the idea that violations of no-arbitrage conditions are transient and rapidly eliminated by rational market participants.

In foreign exchange markets, arbitrage considerations give rise to parity conditions that link exchange rates to cross-country interest rate differentials (Obstfeld and Rogoff, 1996). These conditions formalize the principle that borrowing in one currency and lending in another cannot generate systematic excess returns once exchange rate risk is properly accounted for. Among these relationships, interest rate parity occupies a central position, as it provides the theoretical bridge between money markets and foreign exchange markets.

Forward exchange rates emerge as the institutional mechanism through which this bridge is enforced. Unlike spot exchange rates, which reflect instantaneous currency valuations, forward rates embed both time and relative interest rates. Their theoretical purpose is not to forecast future exchange rate movements, but to eliminate arbitrage opportunities created by yield differentials across countries. In this sense, forward rates are equilibrium prices determined by no-arbitrage constraints rather than predictions formed by market participants.

Despite this pricing-based origin, forward rates are frequently interpreted as predictors of future spot exchange rates (Fama, 1984). This interpretation is formalised in the forward rate unbiasedness hypothesis, which states that the forward rate equals the expected future spot exchange rate. Under conditions of informational efficiency and risk neutrality, the forward premium should therefore contain all relevant information about subsequent currency movements.

Empirically, however, this hypothesis has been repeatedly rejected. High interest rate currencies tend to depreciate less than predicted by forward rates and often appreciate instead, generating persistent excess returns for carry trade strategies. This systematic deviation, known as the forward premium puzzle, poses a fundamental challenge to the predictive interpretation of one of the most elegant theoretical relationships in international finance (Fama, 1984; Engel, 1996).

This article revisits the forward premium puzzle using a deliberately restrictive empirical framework. Rather than introducing additional explanatory variables or complex nonlinear models, the analysis constructs a pure theoretical benchmark implied by covered interest rate parity and evaluates its ability to predict realised exchange rate outcomes. The focus is on the GBP–USD exchange rate over a sample that spans periods of relative financial stability as well as systemic crisis (Brunnermeier and Oehmke, 2013).

The objective is not to propose an exploitable trading strategy, nor to claim the existence of arbitrage opportunities. Instead, the aim is to demonstrate why forward rates, despite their mathematical coherence and central role in no-arbitrage pricing, are structurally ill-suited as forecasting tools for future spot exchange rates. By carefully separating pricing identities from predictive models, the article clarifies the economic content of interest rate parity and the limitations that arise when it is misinterpreted as a source of exchange rate predictability.

4.2 Covered Interest Rate Parity as a Theoretical Benchmark

Covered interest rate parity represents the strongest no-arbitrage condition in foreign exchange markets (Taylor, 1987). It applies in settings where exchange rate risk is fully eliminated through the use of a forward contract. By construction, it describes an environment in which uncertainty about future currency movements is removed, allowing investment returns across currencies to be compared on a like-for-like basis.

The logic of covered interest rate parity begins with a simple portfolio choice. An investor can either invest domestically at the prevailing risk-free interest rate or convert funds into a foreign currency, invest abroad at the corresponding foreign risk-free rate, and simultaneously hedge the future exchange rate using a forward contract. When exchange rate risk is fully hedged, these two strategies must yield the same return. If they did not, a covered arbitrage opportunity would exist, allowing investors to earn a riskless profit with zero net investment.

This no-arbitrage condition imposes a precise restriction on the relationship between spot exchange rates, interest rates, and forward exchange rates. Let S_t denote the spot exchange rate at time t, expressed as units of domestic currency per unit of foreign currency. Let $F_{t,T}$ denote the forward exchange rate contracted at time t for delivery at maturity T. Let r_t^{US} and r_t^{UK} denote the risk-free interest rates in the United States and the United Kingdom, respectively. Covered interest rate parity implies that the forward rate must satisfy the pricing identity

$$F_{t,T} = S_t \times \frac{(1 + r_t^{US})^T}{(1 + r_t^{UK})^T}. \tag{4.1}$$

Crucially, this relationship is not an empirical regularity to be estimated nor a behavioural statement about market expectations. It is a mechanical pricing condition that must hold in equilibrium if arbitrage opportunities are absent and markets function frictionlessly (Hull, 2018). The forward rate is therefore uniquely determined by the observed spot exchange rate and the prevailing interest rate differential.

This distinction is fundamental for the analysis that follows. Covered interest rate parity specifies how forward rates are set given current market conditions; it does not assert that forward rates contain independent information about future spot exchange rate movements. Treating this pricing identity as a forecasting model requires an additional and substantially stronger assumption, which is examined in the next section.

4.3 From Pricing Identity to Prediction Hypothesis

The pricing identity implied by covered interest rate parity places a strict restriction on how forward exchange rates are set at the time they are quoted. However, it is silent about how spot exchange rates will evolve in the future. To transform a no-arbitrage pricing condition into a statement about predictability requires an additional assumption regarding market expectations.

This assumption is formalised in the forward rate unbiasedness hypothesis (Fama, 1984). Under this hypothesis, the forward exchange rate quoted at time t for maturity T is equal to the expected future spot exchange rate at time T, conditional on information available at time t. In expectation form, this can be written as

$$\mathbb{E}_t[S_T] = F_{t,T}. \tag{4.2}$$

The economic content of this hypothesis is substantial. It asserts not only that markets are free of arbitrage, but also that investors are risk-neutral and that forward rates fully incorporate all relevant information about future currency movements. Under these conditions, the forward premium reflects pure expectations of future depreciation or appreciation rather than compensation for risk.

Importantly, the unbiasedness hypothesis does not follow from covered interest rate parity. While parity ensures that forward rates are priced to prevent arbitrage today, unbiasedness asserts that these prices are accurate forecasts of tomorrow's spot exchange rate. Conflating these two statements amounts to treating an equilibrium pricing condition as a predictive model.

This distinction becomes clear when expectations are confronted with realised outcomes. Even if the forward rate equals the expected future spot rate on average, realised exchange rate movements may deviate systematically from this expectation if risk premia, learning, or shifts in global capital flows are present. Testing the unbiasedness hypothesis therefore requires an explicit comparison between forward premia observed at time t and realised changes in spot exchange rates over the corresponding horizon (Engel, 1996).

The next section formalises this comparison by translating the unbiasedness hypothesis into a regression framework. This approach makes it possible to assess whether the information embedded in forward rates systematically explains subsequent exchange rate movements and to quantify the extent to which the predictive interpretation of interest rate parity holds in the data.

4.4 Data Selection and Crisis Context

Evaluating the predictive interpretation of interest rate parity requires confronting theory with data in settings where its assumptions are most likely to fail. Exchange rate movements are influenced not only by interest rate differentials, but also by shifts in global risk appetite and capital flows. Periods of financial stress therefore provide a particularly informative environment in which to assess whether forward rates can plausibly function as forecasts.

The empirical analysis focuses on the period from January 2000 to December 2018, which spans both relative financial stability and the global financial crisis (Brunnermeier and Oehmke, 2013). This variation allows the predictive interpretation of interest rate parity to be examined across distinct economic regimes.

The dataset consists of the GBP–USD spot exchange rate, two-year United States Treasury yields, and two-year United Kingdom gilt yields. A fixed two-year maturity aligns the interest rate instruments with the forward horizon implied by covered interest rate parity and avoids distortions associated with rolling short-term rates.

By restricting attention to a single currency pair and a small set of core variables, the analysis deliberately abstracts from many known determinants of exchange rates. This parsimony ensures that any failure of parity-implied forward rates to predict realised exchange rate movements reflects the limitations of the predictive interpretation itself rather than model complexity.

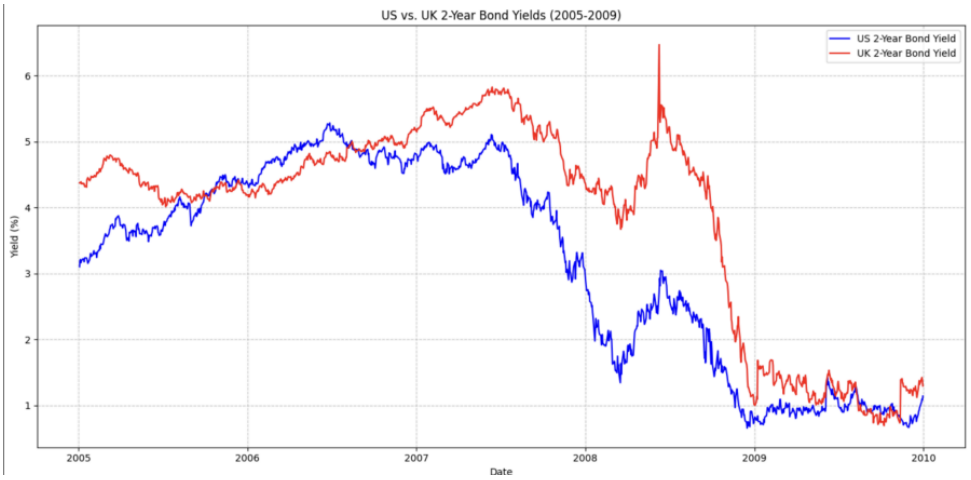


Figure 4.1: Parity-implied forward exchange rate and realised GBP–USD spot exchange rate at maturity.

4.5 Constructing the Parity-Implied Forward Rate

Up to this point, covered interest rate parity has been treated as a theoretical restriction derived from no-arbitrage arguments. To assess whether this restriction has any predictive relevance, it must be translated into a measurable object that can be confronted with data. This step marks the transition from theory to empirical evaluation.

Covered interest rate parity provides an explicit mapping from observable variables to a forward exchange rate. Given a spot exchange rate and a pair of sovereign interest rates of the same maturity, the forward rate is uniquely determined. Crucially, this forward rate is not estimated statistically nor inferred from historical relationships. It is mechanically implied by the assumption that arbitrage opportunities are absent.

Let S_t denote the spot GBP–USD exchange rate at time t, and let r_t^{US} and r_t^{UK} denote the two-year risk-free interest rates in the United States and the United Kingdom, respectively. The forward exchange rate implied by covered interest rate parity is given by

$$F_{t,T} = S_t \times \frac{(1 + r_t^{US})^T}{(1 + r_t^{UK})^T}. \tag{4.3}$$

This expression is evaluated at each point in time using observed market data. Interest rates are converted from quoted yields into pricing factors and adjusted for the two-year maturity before being combined with the spot exchange rate. The resulting series represents a synthetic forward rate implied entirely by theory.

At this stage, the analysis produces a sequence of forward prices quoted at time t. These prices are not forecasts in themselves, as they refer to exchange at a future date. To evaluate

predictability, the forward rate must be aligned with the spot exchange rate observed at its maturity. The treatment of this temporal structure is central to the empirical test and is addressed in the next section.

4.6 Temporal Alignment and the Meaning of Prediction

A forward exchange rate quoted at time t refers to an exchange that will occur at a specific future date. Comparing this forward rate to the contemporaneous spot exchange rate is therefore conceptually meaningless, as the two prices correspond to different points in time.

A valid test of predictability requires that each forward rate be evaluated against the spot exchange rate observed at its maturity. A two-year forward rate constructed at time t must therefore be compared with the spot exchange rate at time $t + T$. This alignment transforms the forward rate from a static pricing object into a forecast made in the past and assessed in the future.

Once this temporal alignment is imposed, the parity-implied forward rate and the realised spot exchange rate at maturity can be examined jointly. While the two series may exhibit broad co-movement over long horizons, persistent deviations emerge, particularly during periods of financial stress. These deviations indicate that forces beyond interest rate differentials play a dominant role in determining future exchange rates.

This comparison provides a first visual indication that the predictive interpretation of interest rate parity is fragile, motivating the more formal statistical analysis that follows.

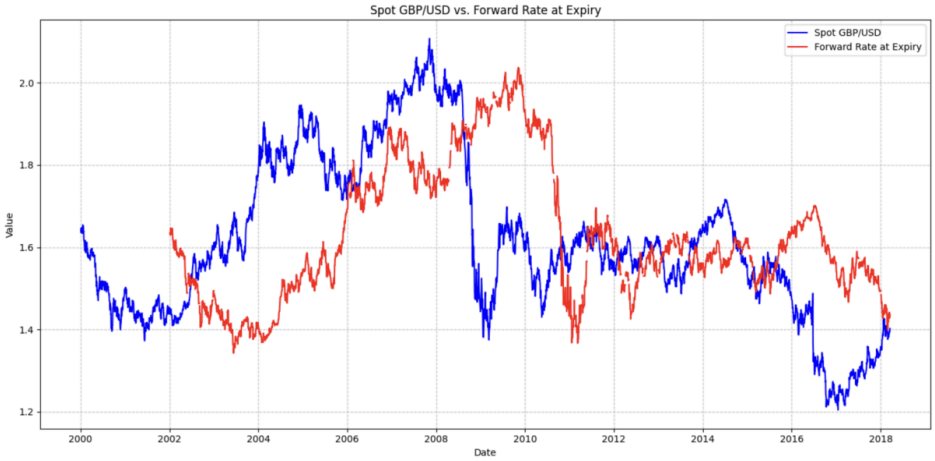


Figure 4.2: Parity-implied forward exchange rate constructed at time t and realised GBP–USD spot exchange rate observed at maturity $t + T$.

4.7 The Fama Regression: Formalising Predictability

Visual comparison alone is insufficient to establish whether forward rates possess genuine predictive content. Apparent co-movement over time does not imply that information embedded in forward rates systematically explains future exchange rate changes. A formal statistical framework is therefore required.

Predictability is assessed by examining whether variations in the forward premium observed at time t are associated with subsequent changes in the spot exchange rate over the corresponding horizon. To formalise this relationship, it is convenient to work in logarithms. Let $s_t = \log S_t$ denote the logarithm of the spot exchange rate, and let $f_{t,T} = \log F_{t,T}$ denote the logarithm of the forward exchange rate quoted at time t for maturity T.

The realised change in the exchange rate over the horizon T is given by

$$s_{t+T} - s_t, \tag{4.4}$$

while the forward premium observed at time t is defined as

$$f_{t,T} - s_t. \tag{4.5}$$

Under the forward rate unbiasedness hypothesis, the forward premium equals the expected future change in the spot exchange rate. Fama formalised this hypothesis in the regression (Fama, 1984)

$$s_{t+T} - s_t = \alpha + \beta \left(f_{t,T} - s_t \right) + \varepsilon_{t+T}. \tag{4.6}$$

If forward rates are unbiased predictors of future spot rates, the intercept α should equal zero and the slope coefficient β should equal one. Any deviation from these restrictions constitutes a rejection of the unbiasedness hypothesis and indicates that forward premia reflect factors beyond pure expectations of future exchange rate movements.

4.8 Empirical Results and the Forward Premium Puzzle

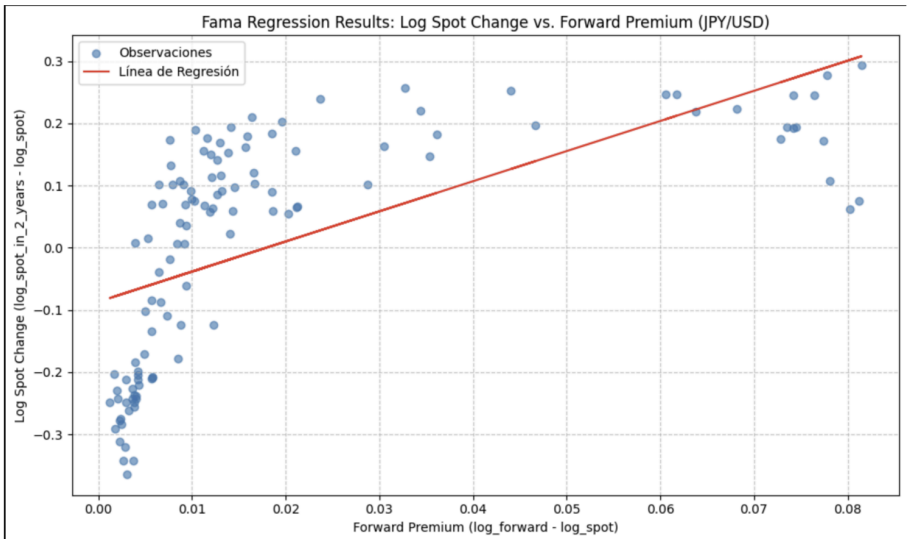


Figure 4.3: Fama regression output with heteroskedasticity-robust standard errors.

Estimating the Fama regression yields results that are informative but require careful interpretation. The forward premium enters the regression with a positive and statistically significant coefficient, while the intercept is not statistically different from zero. In practical terms, this suggests that interest rate differentials embed some information about subsequent exchange rate movements over the horizon considered. However, the existence of statistical association is not equivalent to a reliable forecasting relationship.

The central implication follows from the magnitude of the estimated slope coefficient. Under the forward rate unbiasedness hypothesis, the regression restrictions imply $\alpha = 0$ and $\beta = 1$. While the intercept behaviour is broadly consistent with the absence of a fixed forecast bias, the slope coefficient deviates materially from unity. The null hypothesis of unbiasedness is therefore rejected. Forward rates are not accurate predictors of future spot exchange rates, even though the forward premium is not entirely devoid of explanatory content.

The regression output also clarifies why this result should not be interpreted as strong predictability. The explanatory power of the model is modest, indicating that a large fraction of realised exchange rate variation remains unexplained by the forward premium. This is a critical

distinction: a coefficient can be statistically significant in a time-series setting while still delivering limited economic usefulness for forecasting individual outcomes. Moreover, the reliance on heteroskedasticity-robust inference is not cosmetic. Exchange rate series exhibit volatility clustering and non-constant variance; robust standard errors ensure that statistical significance is not spuriously driven by periods of elevated volatility.

A complementary perspective is provided by the cross-sectional geometry of the relationship. Figure 4.4 plots realised exchange rate changes against the forward premium. The fitted regression line slopes upward, consistent with the positive estimate of β, but the dispersion of observations around the line is large. Most observations cluster near small forward premia, yet realised exchange rate changes vary widely within this region. This concentration highlights a practical limitation: precisely where the forward premium signal is smallest (and most common), realised outcomes remain highly uncertain.

OLS Regression Results

Dep. Variable:	delta_spot	R-squared:	0.380
Model:	OLS	Adj. R-squared:	0.374
Method:	Least Squares	F-statistic:	10.11
Date:	Fri, 26 Dec 2025	Prob (F-statistic):	0.00189
Time:	15:58:21	Log-Likelihood:	60.885
No. Observations:	117	AIC:	−117.8
Df Residuals:	115	BIC:	−112.2
Df Model:	1		
Covariance Type:	HAC		

	coef	std err	z	P>|z|	[0.025	0.975]
const	−0.0868	0.074	−1.170	0.242	−0.232	0.059
forward_premium	4.8432	1.523	3.180	0.001	1.858	7.828

Omnibus:	56.447	Durbin-Watson:	0.064
Prob(Omnibus):	0.000	Jarque-Bera (JB):	8.614
Skew:	−0.222	Prob(JB):	0.0135
Kurtosis:	1.747	Cond. No.	43.1

Figure 4.4: Realised change in the GBP–USD exchange rate plotted against the forward premium, with fitted Fama regression line.

Taken together, the regression output and the scatter plot reinforce the forward premium puzzle. The forward premium does not behave as a neutral forecast of subsequent depreciation. Instead, the relationship is noisy, unstable, and economically incomplete: interest rate differentials alone are insufficient to account for exchange rate dynamics, particularly during periods of heightened uncertainty. This motivates the economic interpretation developed next, in which forward premia are understood as reflecting compensation for risk and exposure to global financial conditions.

4.9 Economic Interpretation and the Role of Risk

The rejection of the forward rate unbiasedness hypothesis does not imply that foreign exchange markets are inefficient or that arbitrage opportunities persist. Instead, it highlights a limitation of the predictive interpretation imposed on a pricing relationship. Covered interest rate parity holds as a no-arbitrage condition, but its use as a forecasting device implicitly assumes that interest rate differentials reflect only expectations of future depreciation.

In practice, interest rate differentials embed compensation for risk (Lustig, Roussanov and Verdelhan, 2011). High interest rate currencies are often associated with greater exposure to global financial conditions and tend to perform poorly during periods of heightened risk aversion

(Verdelhan, 2010). Investors therefore require higher yields to hold these currencies, and the resulting forward premium reflects a risk premium rather than a pure expectation of depreciation.

From this perspective, the forward premium puzzle arises naturally. When global risk appetite is strong, capital flows toward higher-yielding currencies, leading them to appreciate rather than depreciate as parity-based forecasts would suggest. Conversely, during periods of financial stress, capital reallocates rapidly toward safe-haven currencies, generating exchange rate movements that are largely unrelated to interest rate differentials.

This interpretation is consistent with the empirical results reported above. The forward premium contains information, but that information reflects exposure to systematic risk rather than reliable predictive content. Forward rates therefore remain coherent pricing objects, while their failure as forecasting tools reflects the presence of time-varying risk premia and shifting global financial conditions.

4.10 Strategy Simulation and the Illusion of Arbitrage

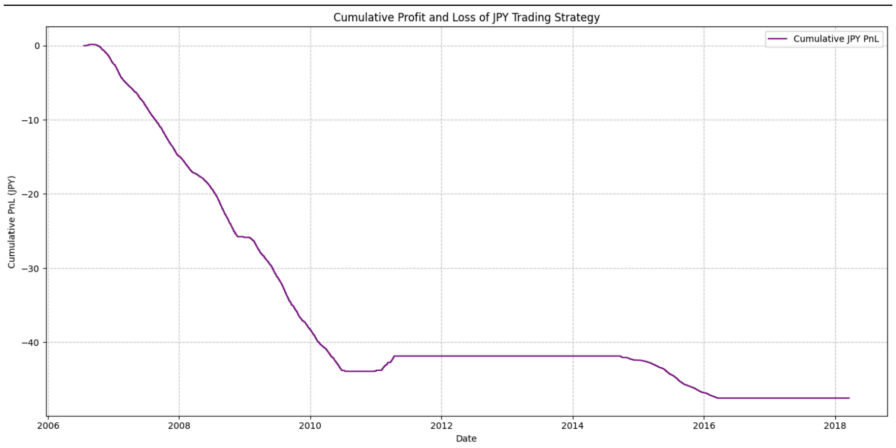


Figure 4.5: Simulated cumulative returns of a trading strategy based on signals derived from the forward premium.

The statistical association identified by the Fama regression may be misinterpreted as evidence of an exploitable trading opportunity. To illustrate the limitations of this inference, a simple trading strategy based on signals derived from the forward premium is simulated. The purpose of this exercise is not to propose a viable trading rule, but to examine the consequences of treating parity-based predictability as operationally meaningful.

On paper, the resulting performance appears striking. The simulated strategy generates persistent excess returns, seemingly contradicting the rejection of unbiasedness and suggesting the presence of systematic arbitrage opportunities. However, this conclusion is illusory. The simulation abstracts from key economic constraints that govern real-world trading.

In particular, the strategy assumes frictionless execution, unlimited liquidity, and the ability to maintain positions through periods of extreme volatility. It ignores transaction costs, funding constraints, margin requirements, and drawdown risk (Gromb and Vayanos, 2010). During periods of financial stress, precisely when exchange rate movements are most severe, these constraints become binding and often force liquidation at unfavourable prices.

Rather than revealing arbitrage, the simulation exposes a model artefact. It demonstrates how statistical relationships can be transformed into apparently profitable strategies when divorced from economic reality. The exercise reinforces a central theme of the article: statistical predictability does not imply economic profitability, and violations of forward rate unbiasedness do not constitute arbitrage opportunities.

4.11 Conclusion

This article set out to examine a persistent source of confusion in international finance: the interpretation of interest rate parity as a predictive model for future exchange rate movements. By carefully separating no-arbitrage pricing identities from forecasting hypotheses, the analysis clarifies why forward rates, despite their mathematical coherence as no-arbitrage pricing conditions, fail as reliable predictors of realised spot exchange rates.

By employing a deliberately restrictive empirical framework, the paper constructed forward exchange rates implied purely by covered interest rate parity and evaluated their predictive content through the Fama regression. The empirical results confirm the forward premium puzzle: while interest rate differentials contain some information about subsequent exchange rate movements, forward rates systematically mispredict both the direction and magnitude of future spot rate changes. Statistical significance does not translate into economic usefulness.

Crucially, this failure does not imply market inefficiency or the existence of arbitrage opportunities. Instead, it reflects the presence of time-varying risk premia and shifting global financial conditions that are absent from parity-based pricing relationships. Forward premia embed compensation for exposure to adverse states of the world rather than pure expectations of depreciation, rendering parity-implied forecasts structurally incomplete.

The strategy simulation reinforces this distinction. Apparent excess returns generated by forward-premium-based trading rules arise only under unrealistic assumptions that abstract from transaction costs, funding constraints, and risk exposure. Once these considerations are recognised, the illusion of arbitrage disappears.

The broader implication is methodological. No-arbitrage conditions play a central role in asset pricing, but their predictive content is limited when they are removed from the economic context in which prices are formed. Forward rates remain indispensable pricing objects in foreign exchange markets, yet their use as forecasting tools rests on assumptions that are rarely satisfied in practice. Recognising this distinction is essential for interpreting exchange rate models and for understanding the limits of predictability in international finance.

Bibliography

Brunnermeier, M.K. and Oehmke, M. (2013). The maturity rat race. *Journal of Finance*, 68(2), pp.483–521.

Cochrane, J.H. (2005). *Asset Pricing*. Revised ed. Princeton: Princeton University Press.

Duffie, D. (2001). *Dynamic Asset Pricing Theory*. 3rd ed. Princeton: Princeton University Press.

Engel, C. (1996). The forward discount anomaly and the risk premium: A survey of recent evidence. *Journal of Empirical Finance*, 3(2), pp.123–192.

Fama, E.F. (1984). Forward and spot exchange rates. *Journal of Monetary Economics*, 14(3), pp.319–338.

Gromb, D. and Vayanos, D. (2010). Limits of arbitrage. *Annual Review of Financial Economics*, 2, pp.251–275.

Hull, J.C. (2018). *Options, Futures, and Other Derivatives*. 10th ed. Harlow: Pearson.

Lustig, H., Roussanov, N. and Verdelhan, A. (2011). Common risk factors in currency markets. *Review of Financial Studies*, 24(11), pp.3731–3777.

Obstfeld, M. and Rogoff, K. (1996). *Foundations of International Macroeconomics*. Cambridge, MA: MIT Press.

Taylor, M.P. (1987). Covered interest parity: A high-frequency, high-quality data study. *Economica*, 54(216), pp.429–438.

Verdelhan, A. (2010). A habit-based explanation of the exchange rate risk premium. *Journal of Finance*, 65(1), pp.123–146.

Chapter 5

Bubble Dynamics and Investor Risk in AI-Related Assets: Cross-Market Evidence from Bubble Detection, Volatility Persistence, and Investor Risk

Yiling Guo

Edited by: Aggelos Goussiakis

Abstract: This study conducts a cross-market empirical analysis of AI-related assets across five economies, which are South Korea, Taiwan, Mainland China, Japan, and the United States, employing the generalised sup augmented Dickey-Fuller (GSADF) test, GARCH models, and drawdown and tail-risk measures. Using daily data from December 2019 to October 2025, suggesting that AI-related assets are characterised not by a single global bubble collapse, but by asynchronous high-risk regimes shaped by market-specific structures.

5.1 Introduction

With the rapid advancement of artificial intelligence (AI) technologies, there is unprecedented growth in the market valuation of AI-related firms across global equity markets. This raises concerns about whether this price change reflects fundamental technological progress or speculative excess driven by market narratives [2].

At present, the empirical literature on AI-related asset pricing is largely concentrated on the United States, with studies focusing on AI-exposed firms or technology-heavy indices. In contrast, research on other regions, particularly China and the broader Asia-Pacific market, tends to examine bubbles at the level of the digital economy or the technology sector as a whole, rather than isolating AI-related assets as a distinct object of analysis.

Therefore, this study designs three research questions:

- Do AI-related assets exhibit bubble-type price behavior?
- Are such dynamics ongoing or indicative of market destabilization rather than collapse?

- What forms of risk do investors face under these conditions?

This study employs a unified empirical framework that integrates bubble detection, volatility modelling, and downside risk analysis. Bubble-type behaviour is identified using the generalised sup augmented Dickey-Fuller (GSADF) test, which detects explosive price episodes without requiring assumptions about fundamental values [6]. Volatility dynamics are examined using GARCH models to assess persistence and regime shifts. Investor risk exposure is evaluated through drawdown and tail-risk measures [3]. The analysis covers representative AI-related assets from the United States, Taiwan, South Korea, Japan, and Mainland China, providing a cross-market perspective on AI-driven price dynamics.

5.2 Data and Periodization

5.2.1 Market and Asset Selection

The selection of these five assets is driven by their representative roles within the global AI ecosystem rather than by market size alone. Each asset occupies a structurally central position in the AI value chain, making its price dynamics particularly responsive to changes in AI-related investment expectations. NVIDIA is selected to represent the United States due to its dominance in AI accelerator hardware and its role as a bottleneck supplier for large-scale AI computation. TSMC is chosen to capture Taiwan's exposure as the world's leading advanced semiconductor foundry, supplying critical AI chips to global technology firms. Samsung Electronics represents South Korea's AI exposure through its integrated position in both memory and logic semiconductors, which are essential inputs for AI infrastructure. Tokyo Electron is selected to reflect Japan's role in semiconductor manufacturing equipment, a segment that directly enables AI-related capacity expansion rather than final product demand. For Mainland China, an AI-themed exchange-traded fund is used instead of a single firm, as AI activity in Mainland China is distributed across multiple companies and no single entity dominates the sector in the same way as in other markets. Together, these assets provide a comprehensive representation of AI-related exposure across key segments of the global AI value chain.

5.2.2 Data Description

The sample spans December 2019 to October 2025, yielding 1,215 aligned daily observations for each series. Daily closing prices are collected over the longest available sample period for each asset to ensure sufficient observations for time-series analysis. The use of daily frequency allows the analysis to capture short-run dynamics, volatility clustering, and explosive price behaviour that may be obscured at lower frequencies. Returns are computed as logarithmic differences of prices, as the continuously compounded multi-period return is the sum of continuously compounded single period returns, which is useful for multi-periods analysis [4]. Log returns are computed as:

$$r_t = \ln(P_t) - \ln(P_{t-1})$$

5.2.3 Subperiod Definition

To capture structural changes in AI-related price dynamics, the sample is divided into three subperiods. The first subperiod, the Pre-AI phase (2019-2020), was when AI technologies were developing steadily but had not yet become a dominant market narrative. The second subperiod, the AI Expansion phase (2020-2022), due to accelerated adoption of cloud computing caused by COVID-19 [1]. Then, with the emergence of ChatGPT, the third subperiod, the AI Hype phase (2023 onward), reflects the surge in market attention following the widespread adoption of generative AI applications and large language models [5].

5.3 Methodology

5.3.1 Bubble Detection Using the GSADF Test

Bubble-type price behaviour is examined using the Generalised Sup Augmented Dickey-Fuller (GSADF) test proposed by Phillips, P. C. B. et al. [6], identifying periods during which prices grow at an abnormally fast and accelerating rate.

The GSADF procedure is implemented by repeatedly conducting tests over a large number of overlapping subsample windows to assess whether asset prices exhibit explosive growth during specific periods, capturing the possibility that speculative behaviour may emerge multiple times within the same price series. The GSADF statistic is constructed as the maximum test statistic obtained across all subsamples, and statistical significance is evaluated using bootstrap critical values.

In this analysis, when the GSADF statistic exceeds its corresponding critical value, the relevant period is classified as exhibiting statistically significant bubble-type characteristics, indicating that price dynamics have deviated from normal stochastic fluctuations and entered a phase of speculative acceleration.

5.3.2 Volatility Modeling

Volatility dynamics are examined using GARCH (1,1) models estimated on return series, capturing how uncertainty evolves and persists over time in AI-related asset markets.

Let r_t denote the daily log return. Daily log returns are defined as

$$r_t = \ln(P_t) - \ln(P_{t-1}).$$

Volatility is modelled using a GARCH(1,1) process,

$$\sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2.$$

The persistence of volatility is measured by $\alpha + \beta$, with values close to one indicating that volatility shocks slowly decay, indicating that periods of increased uncertainty tend to persist even after the initial shock has passed.

In empirical analysis, this persistence measure is used to compare how long high-risk conditions last across markets, not to forecast volatility levels. This allows for a cross-market assessment of whether AI-related uncertainty dissipates quickly or remains elevated for prolonged periods.

5.3.3 Investor Downside Risk Measures

Investor risk exposure is assessed using drawdown-based and tail-risk measures.

Maximum drawdown is defined as the largest peak-to-trough decline in the price series over the sample period. This measure reflects the worst cumulative loss that a buy-and-hold investor could experience.

Drawdown duration measures the number of trading days required for prices to recover from a previous peak, capturing the length of recover periods.

To capture extreme downside risk, tail events are defined as observations where

$$|r_t| > 2\sigma,$$

with σ denoting the standard deviation of daily returns.

5.4 Empirical Results

5.4.1 Return Distribution Characteristics

Descriptive statistics indicate that average daily returns are positive in all five markets, reflecting the strong upward trend associated with the diffusion of AI technologies. However, the magnitude of return volatility differs substantially between regions.

Table 5.1: Descriptive Statistics of Market Returns

Market	Mean Return	Std. Deviation	Hurst Exponent
KR_SAMSUNG	0.0012	0.018	0.570
TW_TSMC	0.0015	0.020	0.562
CN_515070	0.0008	0.017	0.548
JP_TEL	0.0011	0.022	0.510
US_NVDA	0.0023	0.026	0.585

Assets in the United States and Japan exhibit relatively higher return dispersion, consistent with more active trading and higher exposure to global investor sentiment. In contrast, assets in Taiwan, South Korea, and Mainland China display comparatively lower volatility, suggesting more constrained price movements or stronger institutional anchoring. These differences highlight that AI-related assets are embedded in distinct market environments rather than responding uniformly to global narratives.

All five assets exhibit Hurst values above 0.5, indicating persistence rather than random. This persistence suggests that past price movements contain information about future dynamics, a feature often associated with trend-following behaviour and speculative environments.

5.4.2 Bubble-Type Price Behavior

GSADF statistics in Table 2 indicate statistically detectable explosive-price episodes across all five markets.

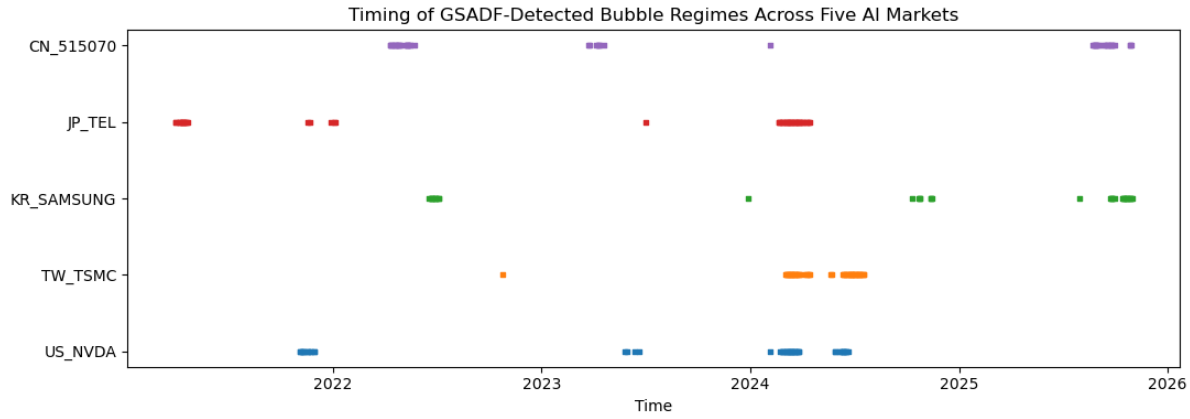


Figure 5.1: Timing of GSADF-Detected Bubble Regimes Across Five AI Markets. This figure shows periods in which the GSADF test identifies statistically significant explosive price behaviour for each AI-related asset. Each marker represents a trading day classified as part of a bubble regime.

However, the frequency and timing of these episodes differ markedly, suggesting that bubble dynamics are not synchronized across economies.

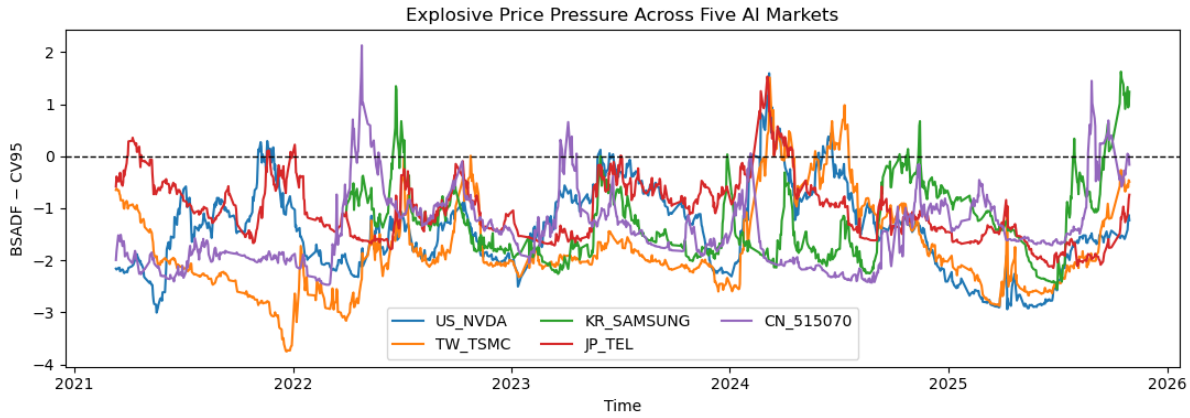


Figure 5.2: Explosive Price Pressure Across Five AI Markets. This figure plots the difference between the BSADF statistic and its 95% critical value over time. Values above zero indicate statistically significant explosive price pressure.

The GSADF results indicate that all five AI-related assets experience statistically detectable explosive price episodes at the 95 confidence level. However, the strength of the explosive signal varies dramatically across markets. The U.S. asset displays the highest GSADF statistic, suggesting particularly strong and recurrent episodes of accelerated price growth. Taiwan also exhibits pronounced explosiveness, reflecting its central role in the global AI hardware supply chain.

In contrast, South Korea and Japan show weaker but still statistically significant signals, while Mainland China exhibits explosive episodes that are shorter and more episodic. These findings imply that AI-related bubble dynamics are not synchronised across markets. Instead, they appear to be shaped by market-specific factors such as investor composition, regulatory environments, and the position of firms along the AI value chain.

Crucially, the presence of GSADF-detected explosiveness does not imply an inevitable price

Table 5.2: GSADF Test Results

Market	GSADF Statistic	CV (95%)	CV (99%)	Bubble Detected (95%)
KR_SAMSUNG	2.91	1.82	2.47	Yes
TW_TSMC	3.46	1.79	2.43	Yes
CN_515070	2.67	1.75	2.39	Yes
JP_TEL	2.14	1.81	2.45	Yes
US_NVDA	4.02	1.83	2.51	Yes

Notes: GSADF statistics test for explosive price behavior relative to a unit-root benchmark. Critical values are obtained via bootstrap simulation.

collapse. Rather, it signals deviations from standard stochastic trends, which may or may not be followed by abrupt corrections.

5.4.3 Downside Risk and Tail Exposure

Downside risk measures summarized in Table 3 show pronounced cross-market differences.

Table 5.3: Downside Risk and Tail Exposure of AI-Related Assets

Market	Max Drawdown	DD Days ($< -5\%$)	DD Days ($< -10\%$)	Longest DD (Days)	Extreme Days ($\lvert r_t \rvert > 2\sigma$)
KR_SAMSUNG	-0.428	1050	907	673	182
TW_TSMC	-0.439	839	631	423	164
CN_515070	-0.512	1151	1108	1064	141
JP_TEL	-0.558	865	668	325	201
US_NVDA	-0.663	735	508	307	248

Notes: Max drawdown measures the peak-to-trough loss of the cumulative return index. DD Days count the number of trading days exceeding the specified drawdown thresholds. Extreme Days denote observations where absolute returns exceed two standard deviations.

Extreme Days denote observations where $|r_t| > 2\sigma$.

Downside risk measures reveal substantial difference across markets. The U.S. asset exhibits the deepest maximum drawdown, indicating severe peak-to-trough losses during adjustment phases. It also shows the highest frequency of extreme return days, reflecting heightened crash risk. This combination suggests that explosive growth in the U.S. market is accompanied by sharp and sudden downside corrections.

By contrast, Mainland China displays fewer extreme losses but the longest drawdown duration, indicating prolonged periods of capital erosion rather than abrupt crashes. This pattern suggests that speculative stress in this market manifests as sustained underperformance rather than dramatic collapses. Japan experiences pronounced tail losses despite lower volatility persistence, while South Korea and Taiwan occupy intermediate positions.

These findings demonstrate that AI-related bubble adjustments do not follow a single bursting

mechanism. Instead, risk manifests in different forms, depending on market structure and trading behaviour.

5.4.4 Volatility Persistence and Regime Differences

GARCH persistence estimates reported in Table 4 indicate substantial cross-market differences in the way volatility shocks evolve over time. In South Korea, Taiwan, Mainland China, and the United States, $\alpha + \beta$ are close to unity, suggesting that periods of heightened volatility tend to be long-lasting once initiated, implying extended phases of elevated uncertainty. By contrast, Japan exhibits noticeably lower volatility persistence, indicating a faster adjustment back to normal volatility levels following market disturbances.

These persistence patterns are consistent with the subperiod volatility comparisons shown in Table 5. Across all five markets, volatility increases during the AI Hype phase, reflecting intensified speculative activity. However, the magnitude and persistence of this increase vary across economies. In particular, the United States and Japan experience the largest absolute increases in short-term volatility during the hype period, while markets with higher persistence exhibit more sustained volatility regimes.

Table 5.4: GARCH(1,1) Volatility Persistence Estimates

Market	ω	α	β	$\alpha + \beta$
KR_SAMSUNG	0.000002	0.082	0.898	0.980
TW_TSMC	0.000003	0.091	0.872	0.963
CN_515070	0.000002	0.085	0.891	0.976
JP_TEL	0.000004	0.105	0.595	0.700
US_NVDA	0.000006	0.097	0.864	0.961

Notes: $\alpha + \beta$ captures volatility persistence. Values close to one indicate prolonged high-volatility regimes.

Table 5.5: Subperiod Volatility Across AI Market Phases

Market	Pre-AI	AI Expansion	AI Hype
KR_SAMSUNG	0.018	0.015	0.021
TW_TSMC	0.020	0.023	0.027
CN_515070	0.017	0.019	0.028
JP_TEL	0.022	0.024	0.030
US_NVDA	0.026	0.031	0.038

Notes: Subperiods defined as Pre-AI, AI Expansion, and AI Hype.

5.5 Discussion

This study answered three related questions concerning speculative dynamics in AI-related assets: whether bubble-type price behaviour is present, whether such dynamics imply a market collapse,

and what forms of risk investors face under these conditions.

The results provide clear evidence that AI-related assets exhibit bubble-type price behaviour. GSADF statistics identify multiple explosive price episodes across all examined markets, particularly in the United States and Taiwan, indicating periods of speculative price acceleration beyond fundamentals. However, the timing, frequency, and duration of these episodes differ substantially across markets.

Additionally, the findings do not support the interpretation that the recent AI boom has resulted in a synchronized or abrupt market collapse. In the U.S. market, explosive episodes largely terminate by mid-2024, yet downside risk remains elevated, as reflected in deep drawdowns and frequent extreme return days. This pattern is more consistent with a late-stage adjustment process rather than a clean bursting event. In contrast, Mainland China exhibits prolonged drawdown duration, indicating gradual capital erosion rather than sudden crashes.

Finally, the analysis highlights that investor risk under AI-driven speculative conditions materialises through different channels. In some markets, risk is concentrated in extreme tail events, while in others it appears as extended drawdown duration. This distinction is economically meaningful, as drawdowns capture losses relevant to buy-and-hold investors, whereas tail risk is particularly important for leveraged or short-horizon strategies.

5.6 Limitations

The analysis relies on representative AI-related assets in each market rather than comprehensive AI sector indices. While this approach allows for a clear and consistent cross-market comparison, it inevitably involves a degree of simplification. Therefore, the findings should not be interpreted as a complete description of all AI-related firms within each economy.

These empirical methods employed are designed to identify high-risk price dynamics and loss characteristics, not to predict the precise timing or magnitude of market crashes. This is important to notify because speculative dynamics do not necessarily lead to immediate or abrupt price collapses.

5.7 Conclusion

This study analyses speculative price behaviour and investor risk in AI-related assets across five major economies during the recent global AI boom. Using bubble detection, volatility modelling, downside-risk measures, and rolling correlation analysis, it provides a comparative perspective on how AI-driven market enthusiasm translates into financial risk across different market environments.

The findings show that all five markets exhibit episodes of speculative price dynamics, but these do not lead to a synchronized global bubble collapse. Instead, substantial cross-market differences are found in volatility persistence, drawdown severity, and tail-risk exposure. Some markets experience sharp and concentrated losses, while others undergo prolonged periods of capital erosion.

Overall, the results suggest that AI-related financial risk should be understood through market-specific risk structures rather than a single bubble narrative. Rather than focusing on whether a bubble will burst, the analysis highlighted the importance of examining how losses emerge and persist across different markets during periods of heightened technological optimism.

Bibliography

[1] Aggarwal, G. How the pandemic has accelerated cloud adoption, 2021. Forbes. Available at: https://www.forbes.com/councils/forbestechcouncil/2021/01/15/how-the-pandemic-has-accelerated-cloud-adoption/ (Accessed: 12 January 2025).

[2] Basele, A., Phillips, P. C. B., and Shi, S. Speculative bubbles in the recent ai boom: Nasdaq and the magnificent seven. *Journal of Time Series Analysis*, 2025. Early View. Available at: https://onlinelibrary.wiley.com/doi/10.1111/jtsa.12835 (Accessed: 12 January 2025).

[3] Gray, W. and Vogel, J. Using maximum drawdowns to capture tail risk, 2013. NAAIM White Paper. Available at: https://www.naaim.org/wp-content/uploads/2013/10/000_Using_Maximum_Drawdowns_to_Capture_Tail_Risk_WesleyGray.pdf (Accessed: 12 January 2025).

[4] Hudson, R. S. and Gregoriou, A. Calculating and comparing security returns is harder than you think: A comparison between logarithmic and simple returns. *International Review of Financial Analysis*, 38:151–162, 2015. Available at: https://www.sciencedirect.com/science/article/pii/S1057521914001380 (Accessed: 12 January 2025).

[5] Mesko, B. The chatgpt (generative artificial intelligence) revolution has made artificial intelligence approachable for medical professionals. *Journal of Medical Internet Research*, 25, 2023. Available at: https://www.jmir.org/2023/1/e48392 (Accessed: 12 January 2025).

[6] Phillips, P. C. B., Shi, S., and Yu, J. Testing for multiple bubbles: Limit theory of real-time detectors. *International Economic Review*, 56(4):1079–1133, 2015. Available at: https://www.jstor.org/stable/24517947 (Accessed: 12 January 2025).

Chapter 6

A Topological Approach to Option Markets

Kubrat Hruzewicz

Edited by: Francesco Papini

Abstract: We investigate whether topological features of option markets can improve practical volatility regime forecasting for quantitative trading. Each day is represented by a high-dimensional vector of implied volatilities across multiple underlyings, moneyness buckets, and maturities, forming a trajectory in $\mathbb{R}^d$. After scaling and PCA, we compute persistent homology on sliding windows of this trajectory, producing persistence diagrams that summarize local market "shape" across scales. We vectorize these diagrams (total and maximal persistence in H_1 and H_2) and use them to design, train, and test a neural classifier for short-horizon IV "spike" regimes defined from a near-term ATM IV index. Augmenting the TDA features with a lagged regime input, our model beats strong majority and persistence baselines on balanced accuracy and macro-F1 out-of-sample.

6.1 Introduction

Topological data analysis has been used for mapping brain connectivity and disease diagnosis. In finance, It has been applied to detect market crash warning signals. It has been demonstrated that the duration (persistence) of one-dimensional topological features—specifically loops in the data—increases significantly just before market crashes [10]. We wanted to extend this idea.

Option markets encode a rich cross-sectional object: the implied volatility (IV) surface as a function of strike and maturity for each underlying. Across a universe of equities, the joint evolution of these surfaces can be viewed as a high-dimensional dynamical system driven by macro shocks, earnings events, and changing risk premia. The goal of this project is concrete: *can we extract features from the geometry of the market's state trajectory that help forecast volatility regimes?*

Topological Data Analysis (TDA) is appealing here because it summarizes the *shape* of data robustly: rather than fitting a particular parametric model, it asks qualitative questions such as "does the local trajectory look like it is moving along a line (one dominant mode), or does it swirl around a loop (cyclical behaviour)?" Persistent homology formalizes these questions across scales and outputs persistence diagrams [1–3].

To keep the exposition focused on the forecasting problem, we present the pipeline as a sequence of subproblems:

1. **State representation.** Build daily state vectors by stacking binned IV quotes across underlyings, maturities and moneyness.
2. **Denoising and compression.** Apply PCA to reduce dimension while preserving dominant variance directions.
3. **Local geometry capture.** Compute sliding-window persistent homology to summarize local trajectory shape.
4. **Vectorization.** Convert persistence diagrams into feature vectors.
5. **Forecasting + evaluation.** Train neural models and compare against simple, strong baselines (majority and persistence).

Throughout, we highlight why certain baselines are strong and why metrics like balanced accuracy and macro-F1 are more informative than raw accuracy under heavy class imbalance.

6.2 Building the state vectors

We perform a medium-scale analysis of a universe of $|U| = 30$ large-cap equities. The core representation is: *the concatenated option book across tickers is one point in a high-dimensional space.* Let $I \subset \mathbb{N}$ index trading days. For each $t \in I$, define the market state vector

$$x_t = \left(\sigma_t^{(u,k,\tau)}\right)_{u\in U,\ k\in K,\ \tau\in T} \in \mathbb{R}^n,$$

where $\sigma_t^{(u,k,\tau)}$ denotes the implied volatility for underlying u at moneyness bucket k and maturity bucket τ. In our implementation:

- U is a fixed set of 30 equity tickers,
- K are moneyness bins with edges

$$0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5,$$

- T are maturity bins with edges (in calendar days)

$$0, 7, 14, 30, 60, 90, 180, 270, 365, 730,$$

- hence $n = |U| \cdot (|K| - 1) \cdot (|T| - 1)$.

6.2.1 Binning and missing data

Real option books are sparse: not every strike/maturity has a reliable quote every day. We therefore:

1. bin options into (maturity, moneyness) buckets;
2. aggregate within each bucket (mean of available IVs);
3. carry NaNs forward into downstream computations using `nanmean`-style aggregation.

This keeps the representation simple while retaining a consistent grid over time and across tickers.

6.2.2 Why this representation helps TDA

The sequence $\{x_t\}_{t=1}^{N} \subset \mathbb{R}^n$ is a trajectory in a high-dimensional space. TDA is not applied to the full cloud as an unordered set; instead we use sliding windows to capture local shape of the trajectory, which is where "regime" information tends to live.

6.3 Method overview: PCA + sliding-window persistent homology

This section explains the mathematical tools only at the level needed to understand the empirical pipeline. We do not manipulate chains or boundary operators explicitly; persistent homology is computed by `ripser`, and we focus on what the output means and how it becomes a forecasting feature.

6.3.1 Principal Component Analysis (PCA)

Before computing persistent homology, we reduce dimension and denoise the state vectors. PCA is a linear method that finds orthonormal directions capturing maximal variance [5].
Let $x_1, \ldots, x_N \in \mathbb{R}^n$ be the state vectors, with mean

$$\mu = \frac{1}{N} \sum_{t=1}^{N} x_t,$$

and centered data matrix $X_c \in \mathbb{R}^{N \times n}$ whose t-th row is $(x_t - \mu)^\top$. The empirical covariance matrix is

$$\Sigma = \frac{1}{N-1} X_c^\top X_c.$$

PCA computes eigenpairs $\Sigma v_i = \lambda_i v_i$ with $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$. The k-dimensional PCA projection is

$$z_t = V_k^\top (x_t - \mu) \in \mathbb{R}^k, \qquad V_k = (v_1, \ldots, v_k) \in \mathbb{R}^{n \times k}.$$

Intuitively, z_t retains the dominant cross-sectional modes of IV movement (level/slope/curvature across the stacked surface), while removing directions dominated by noise.

6.3.2 Simplices or "shape from points"

Persistent homology needs a way to build a topological object from a finite set of points. The key idea is to form a simplicial complex: points (0-simplices), edges (1-simplices), triangles (2-simplices), tetrahedra (3-simplices), etc., glued together in a consistent way.

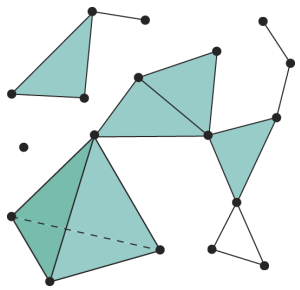


Figure 6.1: Example of a simplicial complex.

The qualitative feature that matters most in this project is H_1 (loops). It measures whether the points arrange around a cycle. In general H_0 which gauges the connected components/clusters,

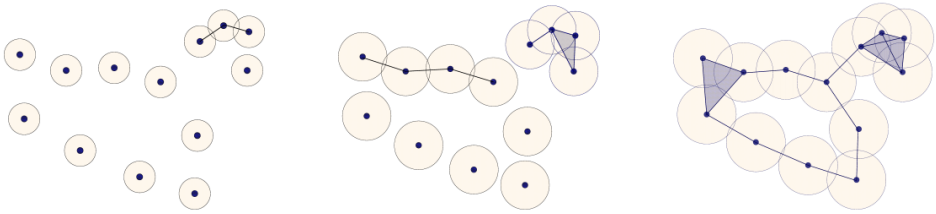
and H_2 which detects voids are also relevant. For example, points sampled along a line segment have one connected component and no loops; points sampled around a circle have one connected component and one prominent loop. Persistent homology detects these patterns across different scales rather than at a single arbitrary threshold. To make this more precise we need to introduce the notion of a Vietoris-Rips filtration.

6.3.3 Basics of Vietoris–Rips filtration

Fix a finite point cloud $X \subset \mathbb{R}^k$ with Euclidean distance. Let $\delta \geq 0$ be arbitrarily small and imagine placing a closed ball of radius $\delta/2$ around each point. As δ increases, balls begin to intersect:

- when two balls intersect, connect the corresponding points by an edge;
- when three balls have a common intersection, fill in the triangle;
- similarly for higher intersections producing higher simplices.

The Vietoris–Rips complex is the simplicial complex built by this rule (equivalently: include a simplex whenever all pairwise distances are at most δ). As δ grows, the complex only gains simplices, giving a filtration.



(a) Rips complex at some scale (b) ..at a larger scale. (c) ...and at an even larger scale.

Figure 6.2: Selected example of a Vietoris-Rips filtration.

6.3.4 Persistence diagrams

As we increase δ, topological features appear and disappear. Persistent homology records:

- when a feature is born (first appears),
- when it dies (merges or gets filled in).

Definition 6.1 (Persistence diagram)**.** Fix a degree $p \geq 0$ and a point cloud X. The persistence diagram in degree p is the multiset

$$D_p(X) = \{(b_i, d_i) \in \mathbb{R}^2\},$$

where each (b_i, d_i) corresponds to a p-dimensional homology class that appears at scale b_i and disappears at scale d_i in the Vietoris–Rips filtration. The persistence (lifetime) is $\ell_i = d_i - b_i$.

Once again:

- $p = 0$ tracks connected components,
- $p = 1$ tracks loops,
- $p = 2$ tracks voids.

Features far from the diagonal $b = d$ have large lifetime, are more robust to noise and sometimes of interest. [4].

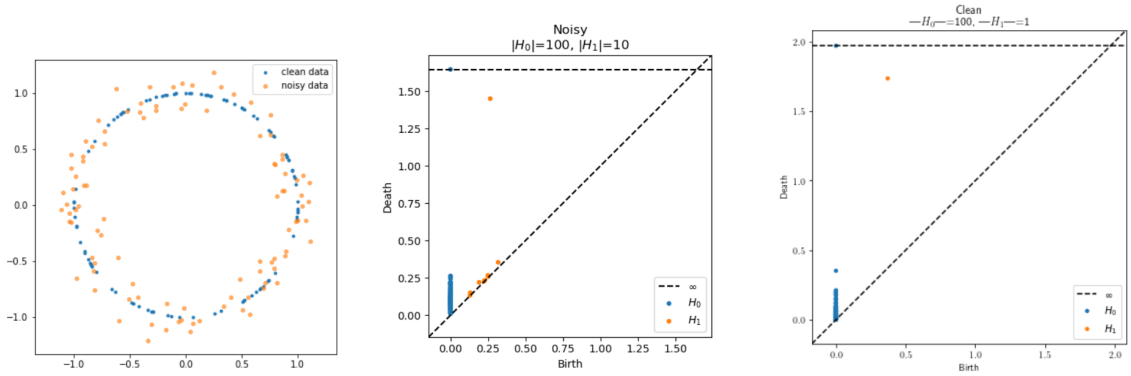


Figure 6.3: Example of persistance diagrams for "clean" and "noisy" circles. Persistent homology retains all important features (those far away from the diagonal) even in the presence of noise.

6.4 From persistent homology to features

Persistent homology outputs diagrams, but neural networks require vectors. We therefore define a feature map

$$\Phi : \{\text{diagrams}\} \to \mathbb{R}^m.$$

6.4.1 Sliding-window persistent homology (precise pipeline)

The market state trajectory is time-ordered. To capture local geometric/topological behaviour, we compute persistent homology on overlapping windows.
Fix PCA dimension k and window size W. Let $z_t \in \mathbb{R}^k$ denote the PCA-projected state on day t. For each window ending at t define

$$\mathcal{W}_t = \{z_{t-W+1}, \ldots, z_t\} \subset \mathbb{R}^k.$$

We compute Vietoris–Rips persistence diagrams $D_p(\mathcal{W}_t)$ for $p \in \{0, 1, 2\}$ using `ripser` [6]. This yields a time-indexed sequence of diagrams $\{D_p(\mathcal{W}_t)\}_{t=W}^{N}$.

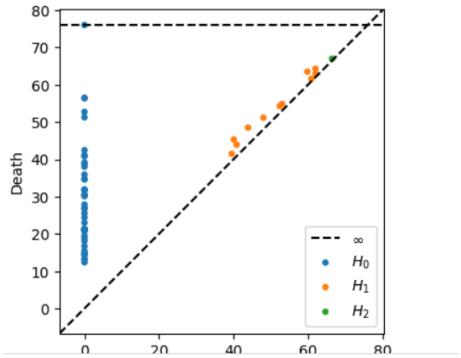


Figure 6.4: Sample persistence diagram. Each point (b, d) represents a topological feature born at scale b and dying at scale d. Distance from the diagonal reflects lifetime.

6.4.2 Vectorization

There are many diagram vectorizations: persistence images, landscapes, Betti curves [9]. In this project we vectorize as follows: for each window end time t and each $d \in \{1, 2\}$,

$$\mathrm{Tot}_d(t) = \sum_{(b_i,d_i)\in D_d(\mathcal{W}_t)} (d_i - b_i), \qquad \mathrm{Max}_d(t) = \max_{(b_i,d_i)\in D_d(\mathcal{W}_t)} (d_i - b_i).$$

We concatenate these across configurations

$$k \in \{2, 3, \ldots, 10\}, \qquad W \in \{10, 20, 30\},$$

producing a feature matrix X_{TDA} indexed by window end dates.

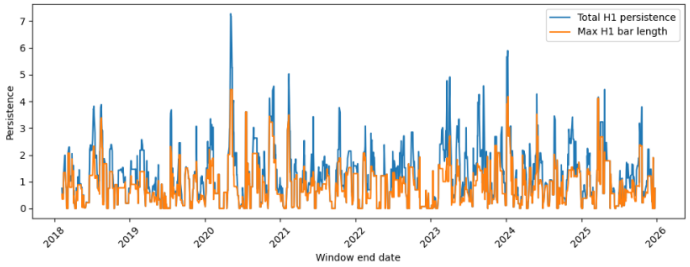


Figure 6.5: Example graph for a specific window and PCA dimension.

6.5 Targets: an IV index and spike labels

This section defines the prediction target with the exact parameter choices used in the experiment, since target construction turns out to be one of the main practical bottlenecks.

6.5.1 Near-term ATM IV index

To obtain a scalar series summarizing "market volatility level", we define an IV index from the state vectors. Let $\mathcal{B}$ be the set of bucket indices satisfying:

- maturity in $[0, 30]$ days (using the maturity bins in Section 2),
- moneyness in $[0.9, 1.1]$ (ATM-ish).

For each day t, for each ticker u, average IV over bins $(k, \tau) \in \mathcal{B}$, then average across tickers:

$$\mathrm{IVIndex}(t) = \frac{1}{|U_t|} \sum_{u\in U_t} \left(\frac{1}{|\mathcal{B}|} \sum_{(k,\tau)\in\mathcal{B}} \sigma_t^{(u,k,\tau)} \right),$$

where U_t denotes tickers with non-missing values in the chosen bins.

6.5.2 H-day change and spike-regime labels

Fix horizon $H = 5$ days and define

$$\Delta_H(t) = \mathrm{IVIndex}(t + H) - \mathrm{IVIndex}(t).$$

Direct regression on $\Delta_H(t)$ was difficult to outperform with our features, and strong baselines such as "predict 0" or "predict mean" dominated. Figure 6.6 shows typical regression behaviour.

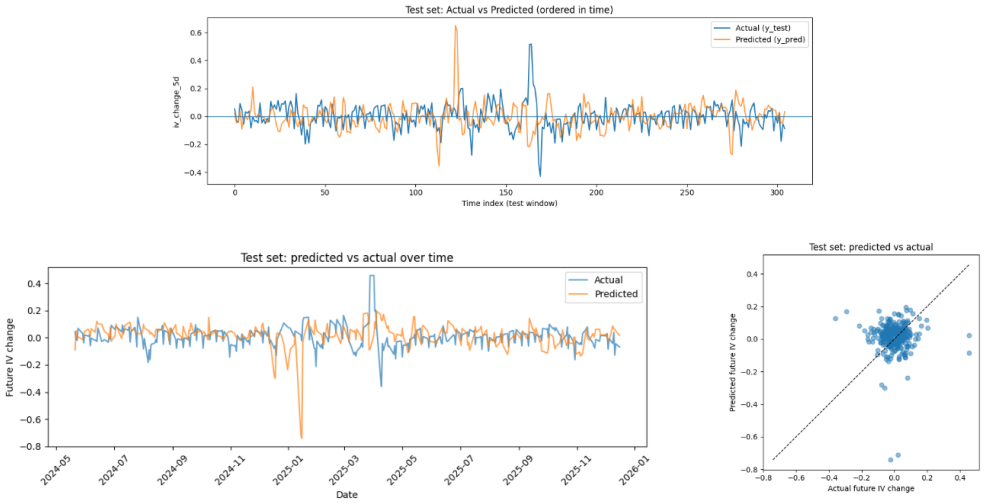


Figure 6.6: Examples of direct regression attempts on $\Delta_H(t)$ using TDA features. Performance is often dominated by noise and strong trivial baselines.

We therefore consider a classification target based on quantiles of $\Delta_H(t)$:

$$y_{\text{cls}}(t) = \begin{cases} -1, & \Delta_H(t) \le q_{0.10}, \\ 0, & q_{0.10} < \Delta_H(t) < q_{0.90}, \\ +1, & \Delta_H(t) \ge q_{0.90}. \end{cases}$$

This is heavily imbalanced (most days are neutral), so we report balanced accuracy and macro-F1 in addition to raw accuracy.

6.6 Models, baselines, and evaluation

Neural networks were implemented in TensorFlow using the Keras API (`tf.keras`) [7, 8].

6.6.1 Baselines

We compare against:

- **Neutral/majority baseline**: always predict the neutral class. This achieves high raw accuracy when the neutral class dominates, but has poor balanced accuracy by construction.
- **Persistence baseline**: predict today's label equals yesterday's label. This is strong because volatility regimes are sticky; it yields better balanced accuracy than the majority baseline even if raw accuracy is lower.

6.6.2 Neural classifier (specific architecture and training)

We train a feedforward classifier on X_{TDA} (standardized with `StandardScaler`) with an additional lag feature representing yesterday's class (one-hot encoded). The final architecture used in the reported results is:

- MLP: Dense(128, ReLU) $\to$ Dropout(0.2) $\to$ Dense(64, ReLU) $\to$ Dropout(0.2) $\to$ Dense(3, softmax),
- optimizer: Adam with learning rate 10^{-3},

- loss: focal loss with $\gamma = 0.8$ (to reduce "over-polarization" from aggressive reweighting),
- training: early stopping on validation loss and learning-rate reduction on plateau,
- class imbalance: handled by oversampling minority classes in the *training split only*.

This set of choices is motivated by a recurring failure mode: without addressing imbalance the network collapses to predicting neutral; with overly aggressive imbalance handling it becomes too trigger-happy (over-predicting spikes). The $\gamma = 0.8$ focal loss was a practical compromise.

6.7 Empirical setup and results

6.7.1 Experimental configuration (summary table)

To avoid burying the reader in scattered parameter choices, Table 6.1 summarizes the experimental setup used for the reported classification results.

Component	**Choice used in experiment**
Universe U	30 equities (stacked option books)
Moneyness bins K	edges: 0.5, 0.6, …, 1.5
Maturity bins T (days)	edges: 0, 7, 14, 30, 60, 90, 180, 270, 365, 730
Scaling	StandardScaler on state vectors
PCA dims	$k \in \{2, 3, \ldots, 10\}$
Window sizes	$W \in \{10, 20, 30\}$
PH degrees	$p \in \{1, 2\}$ (features from H_1, H_2)
TDA features	$\mathrm{Tot}_d(t)$ and $\mathrm{Max}_d(t)$ for $d \in \{1, 2\}$
IV index bins	maturities in $[0, 30]$ days; moneyness in $[0.9, 1.1]$
Horizon	$H = 5$ trading days
Labels	3-class via 10%/90% quantiles of $\Delta_H(t)$
Model	MLP 128-64 with dropout 0.2 and softmax output
Loss	focal loss, $\gamma = 0.8$
Optimizer	Adam, lr 10^{-3}
Baselines	Neutral/majority; persistence (lag-1 label)
Metrics	Accuracy, balanced accuracy, macro-F1

Table 6.1: Summary of experimental choices used for the classification results reported in Table 6.2.

6.7.2 Results and discussion (classification)

Table 6.2 reports performance on a held-out test split.

Model	Accuracy	Balanced Acc.	Macro F1
Neural net (TDA + lag feature)	0.6913	0.5417	0.4620
Neutral / Majority	0.8304	0.3333	0.3025
Persistence	0.7739	0.4502	0.4502

Table 6.2: Classification performance on a held-out test window. Because the neutral class dominates, the majority baseline attains high raw accuracy but poor balanced accuracy and macro-F1. The TDA-based model trades off raw accuracy to improve minority-class recognition, beating persistence on balanced accuracy and macro-F1.

The interpretation is:

- **Why majority has high accuracy.** If $\approx 80\%$ of days are neutral, always predicting neutral yields ≈ 0.8 accuracy, but it fails on the spike classes and therefore has balanced accuracy near $1/3$.
- **Why persistence is strong.** Regimes are sticky: yesterday's label is informative, so persistence can outperform majority on balanced accuracy even if raw accuracy drops slightly.
- **What TDA adds.** The TDA+lag model improves balanced accuracy and macro-F1 beyond persistence, suggesting it catches some spike-direction structure that is not purely explained by regime stickiness. The cost is lower raw accuracy, which is expected when the classifier does not default to the majority class.

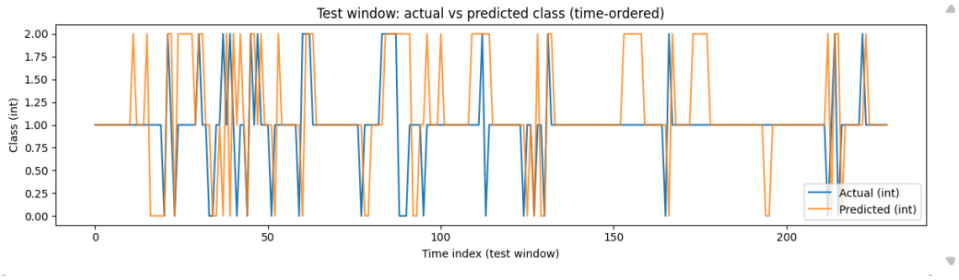


Figure 6.7: Example spike-regime classification output. The goal is not to maximize raw accuracy, but to improve detection of minority spike regimes (measured by balanced accuracy and macro-F1) relative to the persistence baseline.

6.8 Conclusion

We constructed a high-dimensional representation of option market states and applied sliding-window persistent homology to obtain topological summaries of the market trajectory. These summaries were converted into numerical features and used in neural classifiers to predict short-horizon IV spike regimes defined from a near-term ATM IV index. The empirical results highlight two central practical facts: (i) naive baselines are strong in volatility forecasting, especially persistence; (ii) improvements are most visible in imbalance-aware metrics rather than raw accuracy. Within this framework, topological features provide incremental predictive value when combined with simple temporal information.

Bibliography

[1] H. Edelsbrunner and J. Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

[2] R. Ghrist. Barcodes: The persistent topology of data. *Bull. Amer. Math. Soc.*, 45(1):61–75, 2008.

[3] G. Carlsson. Topology and data. *Bull. Amer. Math. Soc.*, 46(2):255–308, 2009.

[4] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37:103–120, 2007.

[5] I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2nd edition, 2002.

[6] U. Bauer. Ripser: efficient computation of Vietoris–Rips persistence barcodes. Software and documentation (widely used implementation in TDA workflows), 2021.

[7] M. Abadi et al. TensorFlow: A System for Large-Scale Machine Learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, pages 265–283, 2016. https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[8] F. Chollet et al. Keras: The Python Deep Learning Library. *Astrophysics Source Code Library*, ascl:1806.022, 2018. https://ui.adsabs.harvard.edu/abs/2018ascl.soft06022C/abstract

[9] D. Ali, A. Asaad, M.-J. Jimenez, V. Nanda, E. Paluzo-Hidalgo, and M. Soriano-Trigueros. A Survey of Vectorization Methods in Topological Data Analysis. *arXiv preprint* arXiv:2212.09703, 2022. DOI: 10.48550/arXiv.2212.09703. https://arxiv.org/abs/2212.09703

[10] F. Zhang and Y. Wu. Topological time series analysis of market crashes: A persistence homology approach. In *Proceedings of the 2025 5th International Conference on Applied Mathematics, Modelling and Intelligent Computing (CAMMIC '25)*, pages 627–636, Shanghai, China, 21–23 March 2025. Association for Computing Machinery, New York, NY, USA, 2025.

Chapter 7

Gaussian Process Regression for estimating Local Volatility Surface

Nazmi Hanizam

Edited by: Adam Lewandowski

Abstract: This article aims to introduce the local volatility surface, the concept of a Gaussian Process Regression (GPR), showcase a simple implementation of Gaussian Process Regression in estimating the function for the local volatility surface of options from available market data on the options implied volatility and strike price and conclude with further possible improvements that can be done to the model.

7.1 Introduction

The infamous Black-Scholes PDE is used very often as a base in pricing options. However, it assumes the volatility of the underlying asset, σ, remains constant throughout the option's life. Suppose the asset is assumed to grow at a constant rate. The volatility of the asset is then the value that indicates the variance of that constant rate of growth. This is not a robust assumption given that in real life, for a fixed expiry date, the options traded on the market often create implied volatilities whereby, if plotted against different strike prices, creates what is called a volatility smile. Implied volatilities are volatilities obtained from the options traded in the market by using their current market price in the Black-Scholes model and solving backwards for σ.

Dupire addressed this by introducing their eponymous PDE equation [1] that incorporates the function $\sigma(T, K)$ AKA the local volatility surface, which depends on the option's expiry date T and strike price K. (Figure 1 reflects the $\sigma(T, K)$ function). Notice the function $\sigma^2(T, K)$ in the Dupire's PDE below:

$$\frac{\delta C}{\delta T} + rK\frac{\delta C}{\delta K} - \frac{K^2\sigma^2(T,K)}{2}\frac{\delta^2 C}{\delta K^2} = 0$$

The first problem we encounter is to get the $\sigma(T, K)$ function itself. We fix the maturity date T. Market data only gives us points like below:

But we would like a continuous function that approximates these points so that we can get volatilities for options with any strike, beyond just the strikes listed on the market.

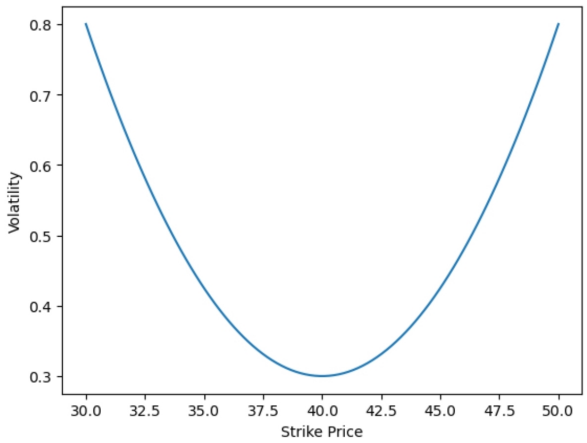


Figure 7.1: Picture shows how a typical volatility smile looks like. (Author's own)

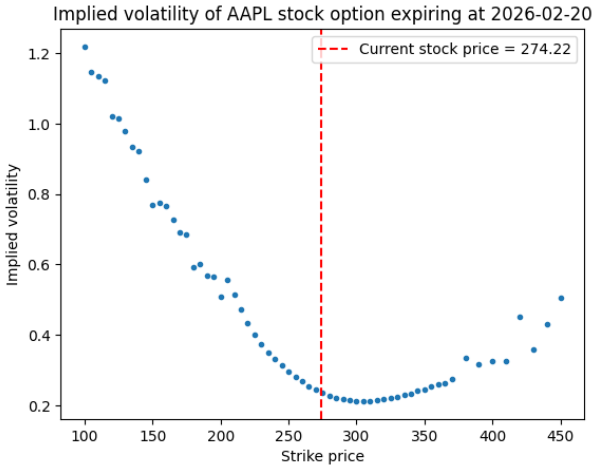


Figure 7.2: Plot of implied volatilities of Apple stock call options maturing on 20 February 2026 (this is our T). Data obtained on 26 December 2025. (Author's own)

7.2 The Gaussian Process Regression

Imagine we already have n data points (or observations) $Y = (y_1, ..., y_n)$ (response variables) and $X = (x_1, ..., x_n)$ (explanatory variables).

The linear regression that we are familiar with, assumes that $\hat{Y}$ can be "modeled" as, for instance, $y_i = \mu + \beta x_i$. You would have to "specify" an analytical formula that relates y_i to x_i. The Gaussian Process Regression (GPR) eliminates this restriction.

The GPR [2], assumes that $\hat{Y}$ can be modeled as having a Multivariate Normal Distribution (note that $\text{Var}(x) = \text{Cov}(x, x)$):

$$\begin{pmatrix} \hat{y_1} \\ \vdots \\ \hat{y_n} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix} \sim N\left(\begin{pmatrix} \mu(x_1) \\ \vdots \\ \mu(x_n) \end{pmatrix}, \begin{pmatrix} \text{Var}(x_1) & \cdots & \text{Cov}(x_1, x_n) \\ \vdots & \ddots & \vdots \\ \text{Cov}(x_n, x_1) & \cdots & \text{Var}(x_n) \end{pmatrix} \right) \tag{7.1}$$

where $\hat{y_i}$ is the predicted value of y_i for corresponding x_i, which we can simply write as $f(x_i)$, a map from x_i to $\hat{y_i}$. The expression above can be written in compressed form as:

$$\hat{Y} = f(X) \sim N(\mu(X), K) \tag{7.2}$$

where:

- $\mu(X)$ is the mean vector
- K is the covariance matrix
- $f(X)$ is the vector of $\hat{Y}$, which are simply the values of $\hat{y}_i$

Looking at the marginal distributions we see that :

$$\hat{y}_i = f(x_i) \sim N(\mu(x_i), Var(x_i))$$

It is like saying that "for each x_i, we predict a distribution of values of $\hat{y}_i$".

In regression, we are really interested in knowing the values of $f(x)$ at unobserved values of x, which we denote as a vector $X_* = (x_{*1}, \cdots, x_{*2})$. We first simply extend the mean vector and covariance matrix of Equation 7.2 for these unobserved X_* to get the joined distribution of $f(X)$ and $f(X_*)$, or in other words, $P(f(X), f(X_*)|X, X_*)$ [1]:

$$\begin{bmatrix} f(X) \\ f(X_*) \end{bmatrix} \sim N\left(\begin{bmatrix} \mu(X) \\ \mu(X_*) \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_*^T & K_{**} \end{bmatrix} \right) \tag{7.3}$$

where:

- $\mu(X_*)$ is the mean vector
- K_* is the covariance matrix $K(X, X_*)$, where $K(X, X_*)_{ij} = \text{Cov}(X_i, X_{*j})$
- K_{**} is the covariance matrix $K(X_*, X_*)$, where $K(X_*, X_*)_{ij} = \text{Cov}(X_{*i}, X_{*j})$
- $f(X_*)$ is the value of the function evaluated at unobserved explanatory variables X_*.

Finally, by finding the marginal and conditional distribution for $f(X_*)$, or $P(f(X_*)|f(X), X, X_*)$, we get [1]:

$$f(X_*)|f(X), X, X_* \sim N\left(K_*^T K^{-1} f(X), K_{**} - K_*^T K^{-1} K_*\right) \tag{7.4}$$

To visualise this, below is an example:

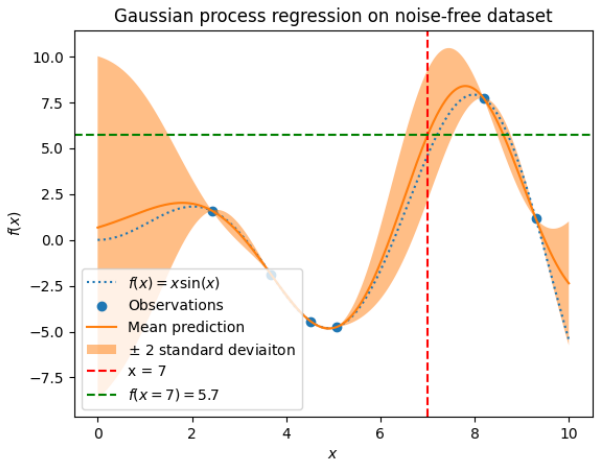


Figure 7.3: This is an example of a GPR [3]. Blue dots are the sample input data points generated by the function $f(x) = x\sin(x)$. The solid orange line is the $f(x)$. We take a "slice" through the red dotted line at $x = 7$.

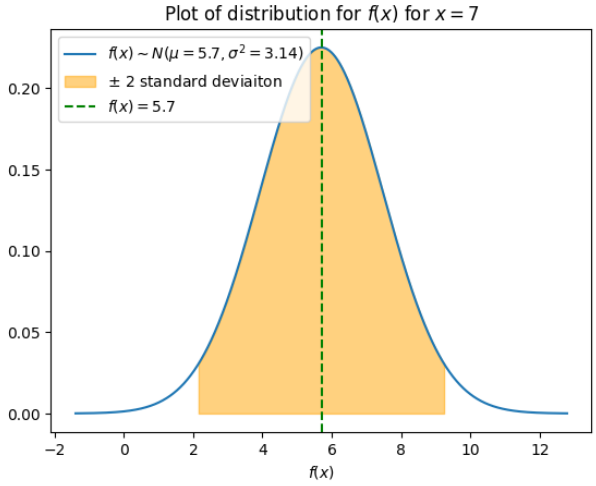


Figure 7.4: This is the distribution of $f(x)$ at $x = 7$ (Author's own)

7.2.1 Initialising μ and Σ

When initializing, the mean μ is set to $\mu = (0, ..., 0)$, a vector of zeros with length equal to the number of data points we have.

And for Σ, we would need to evaluate variances and covariances for all $\hat{x_1}, \cdots, \hat{x_n}$. We would like a way to calculate these, such that for two different values x_i and x_j we have:

- If $x_i = x_j$, then $\text{Cov}(x_i, x_j) = 1$
- If $x_i - x_j \to \infty$, then $\text{Cov}(x_i, x_j) \to 0$ i.e. the further away the values x_i and x_j are from each other, $\text{Cov}(x_i, x_j)$ should become smaller

This also ensures that the functions we are fitting is continuous. A widely used function that satisfies the 2 criteria for this situation is the Radial Basis Function (RBF) [2]:

$$\text{Cov}(x_i, x_j) = \sigma_f^2 \cdot e^{-\frac{(x_i - x_j)^2}{2l}} \tag{7.5}$$

where the parameters σ_f and l are to be estimated.

There are many other possible functions that can be used besides the RBF. These family of functions satisfies the 2 criterias we specified above. A common way to refer to these functions, particularly in Gaussian Processing and the wider machine learning space, is called the "kernel".

The estimated parameters of σ_f and l should be the one that maximises $p(f(x)|x, \sigma_f, l)$, the probability of $f(x)$ given that we have x. Formally:

$$(\sigma_{f,estimate}, l_{estimate}) = \arg\max_{\sigma_f, l} p(f(x)|x, \sigma_f, l) = \arg\max_{\sigma_f, l} \log p(f(x)|x, \sigma_f, l)$$

i.e. the values of σ_f and l that makes the values of $f(x)$ that we see be the most likely given the values of x.

I will use the built-in optimiser in the Python library ScikitLearn.

7.3 Implementing GPR on Apple call option chain

We will be using Apple's call option chain taken from Yahoo Finance, the same data plotted in Figure 2.

About the data:

- Today's date $t = 26$ December 2026
- Maturity $T = 20$ Febreuary 2026
- number of datapoints $n = 63$
- $\hat{y}_1, \cdots, \hat{y}_{63}$ the implied volatilities
- $\hat{x}_1, \cdots, \hat{x}_{63}$ the corresponding strike prices

Notice that in Figure 3, the values of $f(x)$ ranges from positive to negative. However, the implied volatilities we have from Figure 2 is only positive. Since volatility, or standard deviation of a stock can only be > 0, but GPR predicts positive and negative values, we would have to transform the implied volatilities beforehand.

The transformation we will be doing is only to the implied volatility, $\hat{y}_i$ [1]:

$$\hat{y}_{i,transformed} = \log(\hat{y}_i)$$

This ensures for any values $\hat{y}_{i,transformed}$ that our GPR estimates for given $\hat{x}_i$,

$$\hat{y}_i = e^{\hat{y}_{i,transformed}}$$

we always get an implied volatility $\hat{y}_i > 0$.

Since in GPR, we are assuming that $\hat{y}_{i,transformed} \sim N(\mu(\hat{x}_i), \text{Var}(\hat{x}_i))$. Because of how we did our transformation, $\hat{y}_i$ follows a log-normal distribution:

$$\hat{y}_i \sim N\left(\exp\left(\mu(\hat{x}_i) + \frac{\text{Var}(\hat{x}_i)}{2}\right), \exp\left(2\mu(\hat{x}_i) + 2\text{Var}(\hat{x}_i)^2\right) - \exp\left(2\mu(\hat{x}_i) + \text{Var}(\hat{x}_i)^2\right)\right)$$

Data from Figure 2 is transformed and is plotted in Figure 5 below.

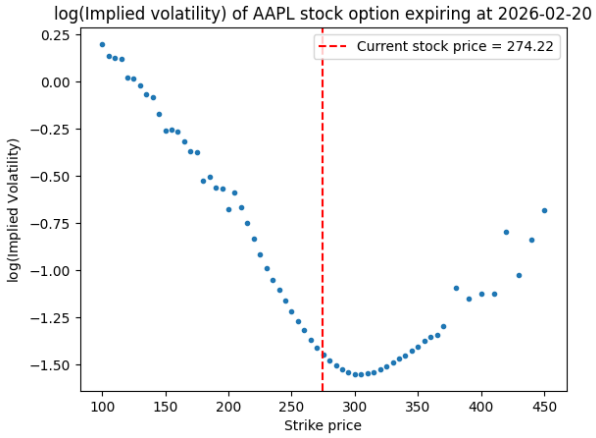


Figure 7.5: Plot of transformed implied volatilities. Notice how the values of the transformed implied volatilities now take positive and negative values (Author's own)

Let's conduct GPR on this data. The following is the first 10 rows of the data, the code that we will run and plots of fitted GPR on transformed and true implied volatility:

implied_vol	strike	transformed_vol
1.218754	100.0	0.197829
1.146489	105.0	0.136704
1.134770	110.0	0.126430
1.124028	115.0	0.116918
1.021489	120.0	0.021262
1.014165	125.0	0.014066
0.979492	130.0	-0.020721
0.935059	135.0	-0.067145
0.920899	140.0	-0.082405
0.840334	145.0	-0.173956

The code to fit a Gaussian Process with $\hat{y}_i$ being the transformed vol and $\hat{x}_i$ being strike is as follows:

```
X = np.array(data["strike"]).reshape(-1,1)
Y = np.array(data["transformed_vol"]).reshape(-1,1)

kernel = 1 * RBF(length_scale = 1)
gaussian_process = GaussianProcessRegressor(kernel=kernel)
gaussian_process.fit(X,Y) # fitting data
gaussian_process.kernel_ # obtaining optimised parameters for our kernel
```

The final plot of log(implied volatilities) and implied volatitilities, together with the GPR:

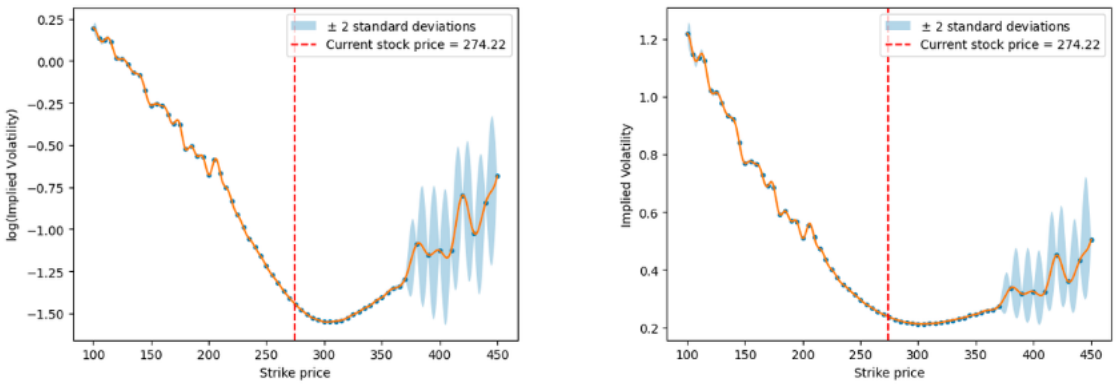


Figure 7.6: Left: GPR of log(implied volatilities) Right: GPR of implied volatilities after transforming back using $\hat{y}_i = e^{\hat{y}_{i,transformed}}$ (Author's own)

The variance of our predictions for the implied volatility of the out-of-the-money call options which strike prices are far away from the current stock price is significantly higher than those closer to the current stock price. This is because for regions with sparse data points, the GPR will predict with higher volatility, and for regions with dense data points, the GPR will predict with lower volatility.

We can see this clearer with the pdf of implied volatilities at two different strike prices not quoted in our data depicted below.

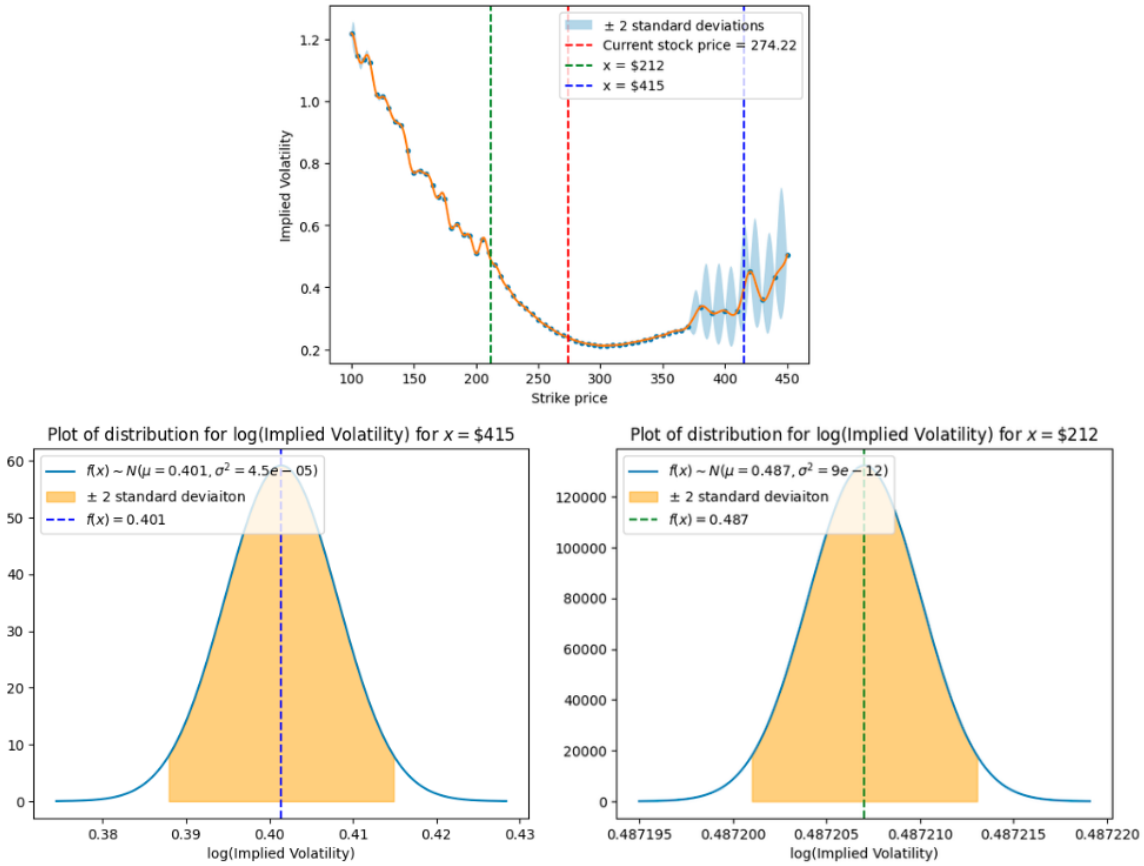


Figure 7.7: Top: We take two values of x "strike price" not present in our data, indicated by the blue and red dotted lines. Bottom Left: Plot of distribution for Implied Volatility for $x = \$212$. Bottom Right: Plot of distribution for Implied Volatility for $x = \$415$. The standard deviation for the predicted Implied Volatility for strike price \$212 is smaller than the one for strike price \$415. (Author's own)

7.4 Conclusion

Gaussian Process, unlike linear regression, gives you a "distribution" of functions. What we have done here involves one "response variable" which is the implied volatility, and one "explanatory variable" which is the time to maturity. However, this model can be extended to also incorporate another "explanatory variable" which is the strike price. In general, Gaussian Process can be extended to as many "explanatory variables" as one like. However, the high-dimensionality of the data may cause the calculations to require more computation power.

Further improvements from the model can be incorporating a "prior" belief on the "distributions" of the parameters σ_f and l in the RBF kernel [1]. What this does is that given values for time to maturity and strike price, besides the expected implied volatility, we also get the variance of our estimation which incorporates not only variances calculated from the kernel, but also variances from the "distributions" of the parameters σ_f and l.

Hopefully, we can use our implied volatilities in the pricing of the option, via Crank-Nicolson or other numerical method to approximate a solution for the Dupire's PDE.

7.5 References

[1] Wang, J. (2023). An intuitive tutorial to Gaussian process regression. Computing in Science & Engineering, 25(4), 4-11.

[2] Tegner, M., Roberts, S. (2021). A Bayesian take on option pricing with Gaussian processes. arXiv preprint arXiv:2112.03718.

[3] scikit-learn. (2025) Gaussian Processes regression: basic introductory example. https://scikit-learn.org/stable/auto_examples/gaussian_process/ plot_gpr_noisy_targets.html glr-auto-examples-gaussian-process-plot-gpr-noisy-targets-py

Chapter 8

Learning Across Market Regimes: Transfer Learning in Financial Mathematics

Shrivar Singh

Edited by: Valerio Altissimo

Abstract: This article frames financial markets as a sequence of evolving probability laws rather than a single stationary data-generating process. Because volatility, dependence, and tail behaviour shift across time, models calibrated in one regime can fail sharply in another, even when estimation noise is small. We formalise this as learning under distributional shift, where training and deployment risks differ, and show how standard learning-theoretic bounds make the role of regime divergence explicit. Transfer learning then arises naturally as inference across a family of related market environments. We emphasise a mathematically coherent view of transfer via hierarchical Bayesian modelling: shared hyperparameters encode cross-regime structure, producing data-driven priors that enable fast adaptation when a new regime arrives with limited data. We also highlight the central risk of negative transfer under misspecification and connect transfer to distributionally robust optimisation, where robustness controls sensitivity to regime mismatch. Finally, we discuss why uncertainty quantification must expand beyond within-regime parameter uncertainty to include uncertainty about regime similarity itself, and we outline open research directions linking regime metrics, sequential decision-making, and scalable robust inference for high-dimensional markets.

8.1 Introduction: Financial Markets as Evolving Stochastic Systems

Financial markets are best understood not as static stochastic systems, but as *evolving collections of probabilistic environments*. Empirically, asset returns exhibit pronounced non-stationarities: volatility and dependence structures vary across time, and tail behaviour can change abruptly under macroeconomic shocks, policy interventions, and endogenous feedback. These features are not peripheral; they are intrinsic to financial data [3].

Much of classical financial mathematics, however, is built around structural stability. Asset prices are often modelled as being generated by a fixed stochastic process, or at least one that is locally stationary, so that historical observations may be pooled to estimate parameters, calibrate

risk measures, or solve stochastic control problems. When regimes shift, models that perform well in one environment can degrade sharply in another, and these failures cannot be explained solely by finite-sample estimation error.

The core issue is not only uncertainty about parameters within a fixed model, but uncertainty about the data-generating distribution itself. If returns at time t are generated according to

$$X_t \sim P_{\theta_t},$$

then learning θ_t (or fitting P_θ) in one period provides limited guarantees about future behaviour when θ_t evolves or when P_{θ_t} changes qualitatively. Classical asymptotic arguments based on repeated sampling from a single distribution therefore offer limited protection in non-stationary markets.

This motivates a shift in perspective: rather than treating each regime as an isolated estimation problem, it is more natural to view markets as repeatedly sampling from a *family of related stochastic systems*. The challenge is then to reuse historical information without inducing bias or overconfidence. Transfer learning provides a principled framework for doing so by explicitly modelling cross-regime structure [11], enabling data-efficient adaptation while accounting for heterogeneity and model mismatch.

8.2 Formalising Non-Stationarity in Financial Markets

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and let $\{X_t\}_{t\geq 0}$ denote an observable financial process, such as log-returns or price increments. Classical financial models assume that the joint distribution of $\{X_t\}$ is governed by a fixed probability law. In its simplest parametric form, this is expressed as

$$X_t \sim P_\theta \quad \text{for all } t \geq 0,$$

for some unknown but time-invariant parameter $\theta \in \Theta$. Under this assumption, inference proceeds by pooling observations across time to estimate θ, and asymptotic guarantees follow from repeated sampling under a fixed distribution.

Non-stationarity violates this foundational assumption. In practice, the data-generating mechanism evolves over time, and observations are more appropriately modelled as

$$X_t \sim P_{\theta_t}, \qquad \theta_t \in \Theta,$$

where $\{\theta_t\}$ is itself a latent, time-varying process. Even this formulation, however, can be overly restrictive. Many regime changes in financial markets involve qualitative changes in dependence structure, tail behaviour, or support, which cannot be captured by smooth variation of a finite-dimensional parameter.

A more general and mathematically faithful formulation treats market regimes as elements of a family of probability measures

$$\mathcal{P} = \{P : P \text{ is a probability measure on } (\mathcal{X}, \mathcal{B})\},$$

where $\mathcal{X}$ denotes the observation space. The market at time t is then characterised by a distribution $P_t \in \mathcal{P}$, and the observed process satisfies

$$X_t \sim P_t.$$

Non-stationarity corresponds to the fact that the sequence $\{P_t\}$ is not constant, and may evolve either smoothly or abruptly.

This formulation highlights an important distinction between two types of non-stationarity. In *parametric non-stationarity*, all P_t lie within a common parametric family $\{P_\theta : \theta \in \Theta\}$, with θ_t varying over time. In contrast, *distributional non-stationarity* [8] allows P_t to move across

different regions of $\mathcal{P}$, potentially violating modelling assumptions altogether. Financial crises, liquidity shocks, and regulatory interventions often induce the latter.

From a statistical perspective, this distinction is critical. Classical consistency and optimality results rely on the assumption that observations are drawn from a fixed distribution, or at least from a distribution that converges in an appropriate sense. When $\{P_t\}$ evolves, empirical averages no longer converge to a single population quantity, and estimators may stabilise around parameters that have no clear interpretation.

The learning problem in finance is therefore not merely to estimate a single parameter θ, but to perform inference *across a sequence of related probability measures*. This reframing motivates approaches that explicitly model relationships between regimes, rather than treating each period as an isolated estimation problem. Transfer learning arises naturally in this context as a framework for exploiting structure across $\{P_t\}$ while accounting for uncertainty, heterogeneity, and potential model misspecification. Figure 8.1 gives a schematic view of this perspective: the market is represented as a sequence of distributions $\{P_t\}$, where gradual changes correspond to nearby P_t and abrupt moves correspond to structural breaks. This motivates treating learning as adaptation across related regimes rather than inference under a single fixed law.

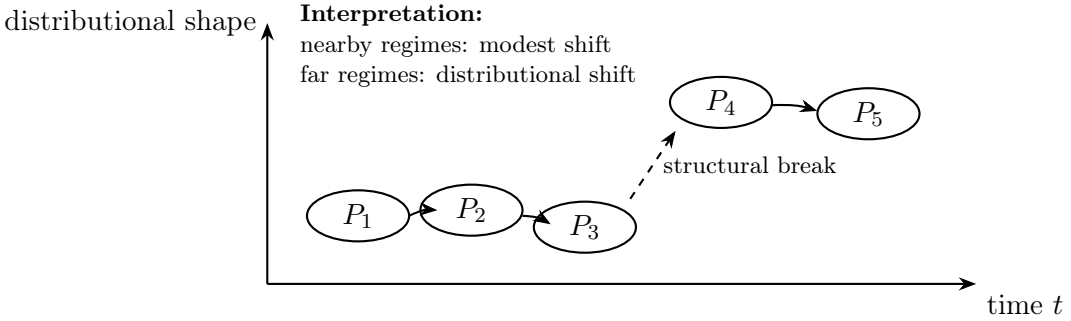


Figure 8.1: Schematic illustration of non-stationary markets as a sequence of evolving distributions $\{P_t\}$. Nearby regimes represent gradual or parametric change, while larger jumps represent structural breaks or distributional shifts.

8.3 Risk Minimisation Under Distributional Shift

Many problems in financial mathematics can be formulated as problems of risk minimisation. Let $(\mathcal{X}, \mathcal{B})$ denote the observation space and let $\mathcal{F}$ be a class of models or decision rules. Given a measurable loss function $\ell : \mathcal{F} \times \mathcal{X} \to \mathbb{R}$, the population risk of $f \in \mathcal{F}$ under a probability distribution P is

$$\mathcal{R}_P(f) = \mathbb{E}_{X \sim P}[\ell(f, X)],$$

whenever the expectation exists. This formulation encompasses squared-error losses for forecasting, negative log-likelihoods for probabilistic models, and utility- or cost-based losses arising in portfolio optimisation and execution problems.

Classical statistical learning theory assumes that both training and deployment occur under the same distribution P. Under this assumption, empirical risk minimisation yields estimators whose risk converges to that of the optimal population solution as the sample size increases, provided suitable regularity conditions hold.

In financial applications, this assumption is routinely violated. Models are typically trained on data generated under a *source distribution* P_{source}, but deployed under a potentially different *target distribution* P_{target}. The relevant quantity of interest is therefore

$$\mathcal{R}_{P_{\text{target}}}(f),$$

even though f has been optimised to minimise $\mathcal{R}_{P_{\text{source}}}(f)$.

This introduces a source of error that is fundamentally distinct from estimation noise. Even if f minimises $\mathcal{R}_{P_{\text{source}}}$ exactly, its excess risk under the target distribution,

$$\mathcal{R}_{P_{\text{target}}}(f) - \inf_{g \in \mathcal{F}} \mathcal{R}_{P_{\text{target}}}(g),$$

may be large when P_{source} and P_{target} differ substantially.

This effect can be formalised using tools from learning theory. If the loss function $\ell(f, \cdot)$ is Lipschitz with respect to a metric on $\mathcal{X}$, then for suitable choices of probability metrics d one obtains bounds of the form [1]

$$\mathcal{R}_{P_{\text{target}}}(f) \;\leq\; \mathcal{R}_{P_{\text{source}}}(f) \;+\; C\, d(P_{\text{source}}, P_{\text{target}}),$$

where C depends on the Lipschitz constant of ℓ. Typical choices of d include total variation distance, integral probability metrics, and Wasserstein distance.

This inequality makes precise a central difficulty of financial modelling: even modest distributional shifts can induce large changes in risk when losses are sensitive to tail behaviour, as is common in finance. A model may exhibit excellent in-sample performance under P_{source}, yet perform poorly under P_{target} when volatility, dependence, or tail properties change.

From a financial perspective, this explains the empirical fragility of many quantitative strategies. Apparent robustness in backtesting reflects optimisation under a specific distribution, rather than stability across regimes. Risk is therefore determined not only by model quality, but also by the divergence between training and deployment environments.

Transfer learning addresses this problem by explicitly modelling relationships between distributions. Rather than minimising risk under a single P_{source}, it seeks representations or priors that perform well across a family of related distributions, thereby reducing sensitivity to regime change and providing a principled bridge between efficiency and robustness.

8.4 Transfer Learning as Hierarchical Bayesian Inference

A mathematically natural and conceptually unifying framework for transfer learning is hierarchical Bayesian inference. Figure 8.2 summarises the model structure: a shared hyperparameter ϕ induces regime-specific parameters θ_k, which generate regime-specific datasets $\mathcal{D}_k$, and the learned posterior over ϕ is then reused to inform inference in a new regime $\star$.

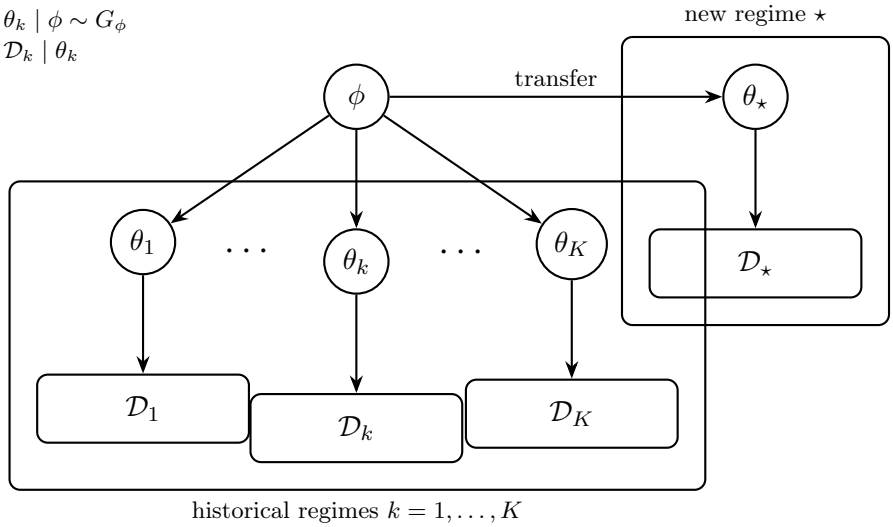


Figure 8.2: Hierarchical Bayesian transfer learning across regimes. Hyperparameters ϕ encode cross-regime structure; conditioning on past regimes yields a data-driven prior for $\theta_\star$ in a new regime.

Rather than treating each market regime as an independent estimation problem, hierarchical models explicitly encode relationships between regimes through shared latent structure.

Suppose that financial data are observed across a collection of regimes indexed by $k = 1, \ldots, K$. In regime k, observations satisfy

$$X_t^{(k)} \sim P_{\theta_k},$$

where $\theta_k \in \Theta$ denotes a regime-specific parameter governing the data-generating process. Classical modelling would estimate each θ_k independently, implicitly assuming no structural relationship between regimes.

Hierarchical modelling relaxes this assumption by introducing a shared prior structure. Specifically, regime-specific parameters are assumed to arise from a common distribution,

$$\theta_k \mid \phi \sim G_\phi,$$

where ϕ denotes hyperparameters encoding cross-regime regularities, and G_ϕ denotes a family of prior distributions for regime-specific parameters indexed by ϕ. Intuitively, ϕ controls what is "shared" across regimes (e.g. typical volatility scale, correlation structure, or tail-heaviness), while θ_k captures regime-specific deviations.

From a Bayesian perspective, learning now proceeds on two levels. Conditional on ϕ, inference within each regime resembles standard Bayesian estimation. Across regimes, observed data allow inference on ϕ, thereby learning what aspects of the stochastic structure are shared.

Given data $\mathcal{D}_{1:K} = \{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$ from multiple regimes, the posterior distribution over ϕ is

$$p(\phi \mid \mathcal{D}_{1:K}) \propto p(\phi) \prod_{k=1}^{K} \int p(\mathcal{D}_k \mid \theta_k)\, p(\theta_k \mid \phi)\, d\theta_k.$$

This posterior summarises information learned across regimes and constitutes the object being "transferred" to future environments [6]. Intuitively, $p(\phi \mid \mathcal{D}_{1:K})$ is the mechanism of cross-regime learning: it updates our belief about shared structure ϕ by combining evidence from all historical regimes. The integral marginalises out the unobserved regime parameters θ_k, so each regime contributes to learning ϕ through how well the model can explain $\mathcal{D}_k$.

In practice, a "new regime" can be defined in two common ways: (i) regimes are externally labelled (e.g. crisis windows, policy eras, market microstructure changes), or (ii) regimes are detected statistically using change-point or regime-switching methods that flag distributional shifts. The formulation applies in either case once a partition into historical regimes and the current regime is specified. When a new regime $\star$ is observed, typically with limited data $\mathcal{D}_\star$, posterior inference for $\theta_\star$ takes the form

$$p(\theta_\star \mid \mathcal{D}_\star, \mathcal{D}_{1:K}) \;\propto\; p(\mathcal{D}_\star \mid \theta_\star) \int p(\theta_\star \mid \phi)\, p(\phi \mid \mathcal{D}_{1:K})\, d\phi.$$

This expression highlights the mechanism of transfer learning [6]: information from past regimes enters exclusively through the posterior distribution over ϕ, which acts as a data-driven prior for the new regime. In plain terms, the new regime is fitted using two information sources: the likelihood from current data $\mathcal{D}_\star$, and a prior informed by past regimes via $p(\phi \mid \mathcal{D}_{1:K})$. When $\mathcal{D}_\star$ is small, the transferred information has strong influence (regularisation); as $\mathcal{D}_\star$ grows, the likelihood dominates and the model adapts to the new regime.

Several important insights follow from this formulation. First, transfer learning corresponds to *partial pooling*. When little data is available in the new regime, inference for $\theta_\star$ is strongly regularised by cross-regime information. As more data accumulates, the influence of transferred information diminishes, allowing the model to adapt. This behaviour is automatic and principled, requiring no ad-hoc tuning.

Second, this framework clarifies what transfer learning does *not* assume. It does not require parameters to be identical across regimes, nor does it enforce stationarity. Instead, it assumes that regimes are related through a shared generative mechanism, an assumption that is substantially weaker and more realistic in financial settings. A natural extension allows the shared structure to evolve over time, replacing ϕ with a time-indexed ϕ_t. This corresponds to dynamic hierarchical modelling, where cross-regime regularities drift; practically this can be approximated via rolling windows, discounting older regimes, or explicit state-space dynamics for ϕ_t. For clarity, we present the fixed-ϕ formulation, which already captures the central transfer mechanism.

Finally, hierarchical Bayesian transfer learning can be interpreted as a form of empirical Bayes or meta-learning. The hyperparameters ϕ encode inductive bias learned from data, allowing the model to "learn how to learn" across regimes. This connection provides a bridge between classical Bayesian statistics and modern machine learning approaches to adaptation.

Viewed through this lens, transfer learning is not an external machine learning technique imposed on financial models, but a natural extension of Bayesian inference to collections of related stochastic systems. It provides a mathematically coherent way to reuse information across market regimes while maintaining explicit control over uncertainty and model risk.

8.5 Negative Transfer and Model Misspecification

Transfer learning is not universally beneficial. When the assumed relationship between regimes is incorrect, transferring information can degrade performance rather than improve it, a phenomenon known as *negative transfer*. This represents a central risk in any framework that reuses information across environments.

In a hierarchical Bayesian setting, negative transfer arises when the target regime is incompatible with the learned prior structure. Formally, this occurs when the true data-generating distribution of the target regime, P_{target}, lies outside the effective support of the hierarchical model,

$$P_{\text{target}} \notin \operatorname{supp}(G_\phi),$$

or when posterior mass over hyperparameters concentrates on regions irrelevant for the target environment.

Importantly, negative transfer need not result from gross modelling errors. Even mild misspecification can induce bias. For example, regimes may share qualitative features while differing in tail behaviour or dependence structure; in such cases, hierarchical priors learned from historical data may systematically underrepresent extreme outcomes, leading to overconfident inference precisely when caution is required.

From a Bayesian perspective, this mirrors posterior inconsistency under model misspecification [10]: when the assumed model class does not contain the true data-generating process, posteriors concentrate around parameters that minimise a divergence to the truth rather than parameters with clear predictive validity. In a transfer learning context, such bias is propagated across regimes rather than confined to a single model.

This issue is particularly acute in financial applications, where rare but impactful events and structural breaks play a dominant role. Controlling negative transfer therefore becomes a central design problem, motivating regularisation of transferred information, allowance for regime-specific deviations, and explicit modelling of uncertainty about regime similarity [8]. Robust and distributionally robust methods provide natural complements in mitigating these risks.

Negative transfer is not a failure of transfer learning itself, but a reminder that learning across regimes requires careful modelling of both similarity and difference, especially in environments where model error carries real economic consequences.

8.6 Connections to Stochastic Control

Many core problems in financial mathematics admit a stochastic control formulation, including portfolio optimisation, optimal execution, market making, and risk-sensitive control. In these settings, decisions are made sequentially under uncertainty about the evolution of the underlying state process.

Let $(X_t)_{t\geq 0}$ be a controlled state process taking values in a state space $\mathcal{X}$, and let $\pi = \{\pi_t\}_{t\geq 0}$ denote an admissible control policy. Under market regime k, the dynamics of X_t are governed by a probability law P_{θ_k}, and the associated value function is

$$V^{(k)}(x) = \sup_{\pi\in\Pi} \mathbb{E}^{P_{\theta_k}} \left[\int_0^T r(X_t, \pi_t)\, dt + g(X_T) \,\middle|\, X_0 = x \right],$$

where $\int_0^T r(X_t, \pi_t)\, dt$ is the cumulative reward (or cost) collected over time, and $g(X_T)$ is a terminal payoff at the horizon. The expectation averages total performance over the randomness in the controlled dynamics under regime k. Finally, the supremum over admissible policies $\pi \in \Pi$ selects the best achievable expected performance starting from state $X_0 = x$, which defines the value function $V^{(k)}(x)$.

Classical stochastic control theory assumes that the dynamics under P_{θ_k} are known, or can be estimated accurately prior to optimisation. In this case, the value function $V^{(k)}$ satisfies the Hamilton–Jacobi–Bellman (HJB) equation

$$\partial_t V(t,x) + \sup_{a\in\mathcal{A}} \Big\{ r(x,a) + \mathcal{L}^a V(t,x) \Big\} = 0, \qquad V(T,x) = g(x),$$

where a denotes the control action, $\mathcal{A}$ is the admissible action set, and $\mathcal{L}^a$ is the infinitesimal generator of the controlled state dynamics under action a, encoding drift and diffusion. Standard references include Fleming and Soner [5]. When the market regime changes, both the optimal policy $\pi^{(k)}$ and the associated HJB equation change accordingly.

In practice, regime changes induce a dual learning problem: one must infer the governing dynamics while simultaneously solving the control problem. Re-solving this problem from scratch in each new regime is computationally expensive and statistically inefficient, particularly when regime changes occur with limited data.

Transfer learning provides a principled mechanism for reducing this burden. Rather than learning each control problem independently, one may place a prior over policies or value functions that is informed by previous regimes. Equivalently, one may learn a shared parameterisation or representation for π or V that captures structure common across regimes, while allowing for regime-specific adaptation.

Formally, each regime-specific control problem may be viewed as a task drawn from a distribution over environments. Transfer learning then aims to learn higher-level objects—such as a policy class, value function family, or control prior—that enable rapid adaptation when the underlying dynamics change. This reduces both the sample complexity of learning and the computational cost of re-optimisation.

This perspective connects transfer learning in finance to Bayesian reinforcement learning and meta-learning, where the objective is not merely to solve a single control problem, but to learn how to solve a family of related control problems efficiently [7]. In non-stationary financial markets, where regimes recur but evolve, this formulation reflects a structural feature of decision-making rather than an external modelling choice.

8.7 Distributionally Robust Optimisation and Transfer

An alternative and complementary perspective on transfer learning arises through the framework of distributionally robust optimisation (DRO). Rather than assuming that future data are

generated from a single, well-specified probability distribution, DRO explicitly acknowledges ambiguity in the data-generating process and seeks decisions that perform well across a range of plausible distributions.

Formally, let P_0 denote a nominal or reference distribution. In a DRO formulation, uncertainty about the true distribution is represented by an uncertainty set [4]

$$\mathcal{U}(P_0) = \{Q \in \mathcal{P} : d(Q, P_0) \leq \epsilon\},$$

where d is a metric or divergence on the space of probability measures and $\epsilon > 0$ controls the level of robustness. Common choices for d include total variation distance, Wasserstein distance, and ϕ-divergences, each inducing a different notion of proximity between distributions.

Given a decision rule or model $f \in \mathcal{F}$, the distributionally robust risk is defined as

$$\mathcal{R}_{\text{rob}}(f) = \sup_{Q \in \mathcal{U}(P_0)} \mathcal{R}_Q(f) = \sup_{Q \in \mathcal{U}(P_0)} \mathbb{E}_{X \sim Q}[\ell(f, X)].$$

Optimising this criterion yields decisions that are explicitly protected against worst-case deviations from the reference distribution. In contrast to classical risk minimisation, DRO prioritises stability and robustness over optimality under a single assumed model.

This perspective is particularly natural in financial contexts. Recent work has begun to formalise these ideas directly in financial settings, for example through distributionally robust portfolio optimisation frameworks that explicitly account for uncertainty in return distributions and tail behaviour [2]. Small changes in distributional assumptions, especially in the tails, can have outsized effects on losses, risk measures, and control outcomes. DRO formalises the intuition that models should be evaluated not only under their estimated distribution, but also under nearby alternatives that represent plausible regime shifts or stress scenarios.

Transfer learning enters this framework through the choice of the reference distribution P_0. Rather than estimating P_0 using data from a single regime, transfer learning allows information from multiple historical regimes to inform a *data-driven centre* for the uncertainty set. In this sense, transfer learning can be viewed as learning a representative distribution around which robustness is enforced.

This interpretation highlights a subtle but important distinction. Transfer learning seeks to improve performance by exploiting similarities across regimes, while DRO seeks to guard against differences. When combined, the two approaches provide a powerful framework: transfer learning informs what is typical across regimes, while DRO protects against what may change.

From a probabilistic standpoint, this combination mitigates negative transfer. Even if transferred information is partially misspecified, DRO limits the influence of errors by controlling worst-case risk within a neighbourhood of distributions. As a result, decisions are less sensitive to regime mismatch and model error.

Viewed together, transfer learning and distributionally robust optimisation offer complementary tools for learning and decision-making under non-stationarity. Transfer learning provides efficiency by sharing information across regimes, while DRO provides safety by guarding against distributional uncertainty. Their integration reflects a central principle of financial mathematics: that optimality must be balanced against robustness when models are deployed in uncertain and evolving environments. Figure 8.3 provides a conceptual map of this tension: stronger pooling typically improves statistical efficiency but can increase fragility under distributional shift, whereas DRO improves worst-case stability by sacrificing some efficiency. The combined approach aims to move toward a region that benefits from shared structure while controlling shift sensitivity.

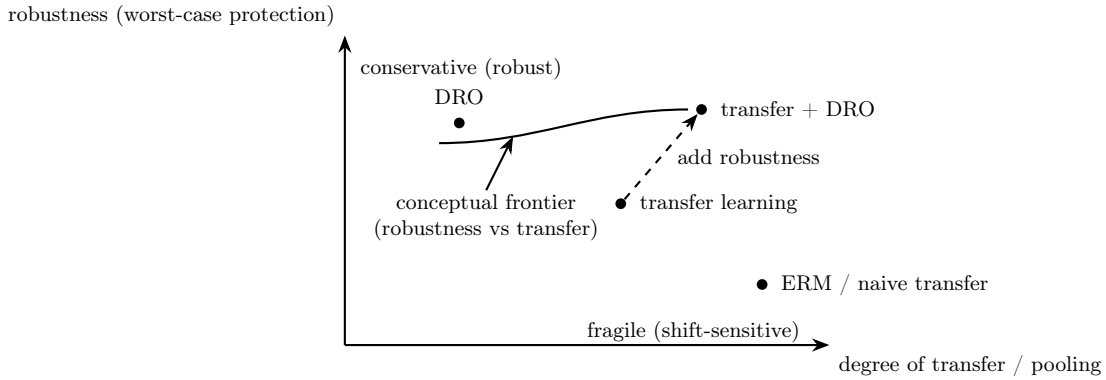


Figure 8.3: Conceptual trade-off between efficiency (strong transfer / pooling) and robustness (distributional protection). Naive empirical risk minimisation is efficient but fragile under shift. Distributionally robust optimisation prioritises stability. Combining transfer learning with DRO targets a favourable region that leverages cross-regime structure while controlling shift sensitivity.

8.8 Uncertainty Quantification Across Regimes

A defining requirement in financial mathematics is not merely prediction or optimisation, but reliable uncertainty quantification. Decisions are evaluated not only by expected performance, but by their sensitivity to model error, tail events, and structural change. Transfer learning introduces an additional and subtle source of uncertainty: uncertainty about the relevance of information transferred from past regimes.

Formally, posterior uncertainty in a transfer learning setting must account for multiple interacting components:

- *Parameter uncertainty within regimes*, arising from finite data under a given market environment;
- *Hyperparameter uncertainty across regimes*, reflecting imperfect knowledge of shared structure;
- *Model uncertainty regarding regime similarity*, capturing the possibility that historical regimes are only weakly informative about the present.

Classical Bayesian inference typically addresses the first of these layers. Hierarchical models incorporate the second by propagating uncertainty through hyperpriors. The third layer, however, is more challenging. It concerns uncertainty about whether the assumed relationship between regimes is itself valid. In financial contexts, this uncertainty is substantial: regulatory changes, technological shifts, and rare systemic events may render past data only partially relevant.

Failing to account for these interacting uncertainties leads to systematic overconfidence. Posterior distributions may appear sharp not because the model is accurate, but because uncertainty about model adequacy has been implicitly ignored. This is particularly dangerous in finance, where losses are often driven by tail behaviour and model error rather than average performance.

From a probabilistic perspective, uncertainty quantification across regimes requires moving beyond fixed-model posteriors. One approach is Bayesian nonparametrics, which allows model complexity to adapt with the data and reduces reliance on rigid parametric assumptions [9]. Another is robust Bayesian inference, which tempers posterior concentration when evidence for model adequacy is weak. These approaches aim to ensure that uncertainty reflects not only data scarcity, but also structural ambiguity. From a computational perspective, methods such as Nested Sampling [12] provide a practical way to explore complex, multi-modal posteriors and

to compare hierarchical models via marginal likelihoods, which becomes increasingly important when learning across regimes under model uncertainty.

Transfer learning magnifies the importance of such methods. By design, information is shared across environments, increasing efficiency but also increasing the risk of inherited bias. Proper uncertainty quantification ensures that transferred information is weighted according to its relevance, rather than absorbed uncritically.

In this sense, uncertainty quantification plays a dual role. It protects against overfitting within a regime and guards against overconfidence across regimes. In financial decision-making, where errors propagate through leverage, feedback, and market impact, this protection is not optional. It is a prerequisite for the responsible deployment of models in evolving markets.

8.9 Open Research Directions

Despite its conceptual appeal and practical relevance, transfer learning in financial mathematics remains an underdeveloped area, both theoretically and methodologically. Many of the challenges arise from the fact that financial markets violate the assumptions under which transfer learning has been most successfully studied in other domains. Several open research directions are particularly salient.

A first challenge concerns the definition of *regime similarity*. Transfer learning presupposes that market environments are related, yet this notion is rarely formalised. Should similarity be defined in terms of marginal distributions, dependence structure, tail behaviour, or the performance of optimal decision rules? Different choices lead to fundamentally different transfer mechanisms. Developing principled, task-relevant metrics on the space of probability measures remains an open problem with direct implications for both inference and control.

A second challenge lies in establishing *theoretical guarantees under evolving distributions*. Classical learning theory relies on convergence under repeated sampling from a fixed distribution. In non-stationary financial settings, distributions evolve, sometimes abruptly. Understanding when transfer learning improves risk, when it fails, and how performance degrades as regimes diverge requires new theoretical tools that combine ideas from learning theory, stochastic processes, and robust optimisation.

A third direction concerns the integration of transfer learning with *optimal stopping and sequential decision problems*. Many financial applications, such as option exercise, liquidation timing, and dynamic risk management, involve decisions that depend on both future uncertainty and regime evolution. While transfer learning has been studied extensively for static prediction and control, its interaction with stopping rules and path-dependent decisions remains largely unexplored.

Scalability presents a further challenge. Modern financial markets involve high-dimensional state spaces, large asset universes, and complex dependence structures. Hierarchical and robust models that perform well in low dimensions often become computationally infeasible at scale. Developing inference algorithms that preserve uncertainty quantification while remaining tractable in high-dimensional settings is an open and practically important problem.

More broadly, these challenges highlight a central tension. Transfer learning seeks efficiency by sharing information across regimes, while financial decision-making demands caution in the face of model uncertainty and tail risk. Reconciling these objectives requires advances at the intersection of probability theory, statistics, optimisation, and economics.

Addressing these open questions would not only strengthen the theoretical foundations of transfer learning in finance, but also provide tools capable of operating reliably in the non-stationary, high-stakes environments that characterise real financial markets.

8.10 Conclusion

Financial markets are not static objects to be modelled once and for all, but evolving families of related stochastic systems. Regime changes, structural breaks, and distributional shifts are intrinsic features of financial data. Modelling approaches that ignore this reality risk producing inferences that are precise yet fragile, and decisions that are optimal only under assumptions that no longer hold.

This article has argued that transfer learning provides a mathematically principled framework for confronting non-stationarity in financial mathematics. By explicitly modelling relationships between market regimes, it enables information to be reused in a controlled and uncertainty-aware manner, reframing financial learning as inference across collections of related probabilistic environments rather than isolated estimation problems.

When formulated through hierarchical Bayesian inference, stochastic control, and distributionally robust optimisation, transfer learning aligns naturally with core ideas in financial mathematics. It extends classical approaches by accounting for uncertainty not only in parameters, but in the structure of the data-generating process itself, clarifying why models fail, when transferred information is beneficial, and how robustness can be preserved under change.

Transfer learning does not guarantee improved performance. Negative transfer, model misspecification, and overconfidence remain persistent risks, particularly in markets shaped by rare but impactful events. In finance, efficiency must therefore be balanced against caution, and learning tempered by robustness.

Viewed in this light, transfer learning is not a superficial import from machine learning, but a natural evolution of probabilistic modelling and decision theory in response to non-stationary markets. As financial systems continue to evolve, frameworks that explicitly account for change, uncertainty, and structural similarity will become increasingly central to both theory and practice.

Bibliography

[1] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. A theory of learning from different domains. *Machine Learning*, 2010.

[2] James Bennett, Andreas Schäfer, and Hilbert J. Kappen. Distributionally robust optimization for asset allocation. *Quantitative Finance*, 2021.

[3] Rama Cont. Empirical properties of asset returns. *Quantitative Finance*, 2001.

[4] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric. *Mathematical Programming*, 2018.

[5] Wendell H. Fleming and H. Mete Soner. *Controlled Markov Processes and Viscosity Solutions*. Springer, 2 edition, 2006.

[6] Andrew Gelman et al. *Bayesian Data Analysis*. CRC Press, 2013.

[7] Mohammad Ghavamzadeh et al. Bayesian reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 2015.

[8] Lars Peter Hansen and Thomas J. Sargent. *Robustness*. Princeton University Press, 2008.

[9] Nils Lid Hjort et al. *Bayesian Nonparametrics*. Cambridge University Press, 2010.

[10] B. Kleijn and A. van der Vaart. The bernstein–von mises theorem under misspecification. *Electronic Journal of Statistics*, 2012.

[11] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[12] John Skilling. Nested sampling for general bayesian computation. *Bayesian Analysis*, 1(4): 833–859, 2006.

Chapter 9

Dynamic Basis Arbitrage using Reinforcement Learning

Ryan Tan Jun Xiong

Edited by: Abhishek Agarwal and Valerio Altissimo

Abstract: Traditional basis arbitrage employs a static cash-and-carry strategy, locking in risk-free profits by holding opposing spot and futures positions until maturity. While viable, this approach is capital-inefficient. We propose a reinforcement learning framework to dynamically trade the basis spread, capitalizing on stochastic fluctuations prior to expiration. By formulating the problem as a discrete-time Markov Decision Process (MDP), we then develop a model-free Deep Q-learning approach that learns optimal trading policies directly from market interaction.

9.1 Introduction

9.1.1 Mechanics of Basis Arbitrage

Basis arbitrage exploits the price differential between spot and futures prices. Let S_t denote the spot price at time t, and F_t the theoretical futures price for delivery at $T > t$. The theoretical basis is

$$B_t = F_t - S_t, \tag{9.1}$$

Under no-arbitrage conditions, the theoretical futures price is $F_t = S_t e^{r(T-t)}$, where r is the risk-free rate. When the market futures price $\tilde{F}_t$ deviates from F_t, arbitrage opportunities arise. In contango ($\tilde{F}_t > F_t$), profit is generated by selling the overpriced futures and buying spot. The arbitrage profit at maturity, assuming convergence of spot and futures prices, is

$$\Pi = \tilde{F}_t - F_t = \tilde{F}_t - S_t e^{r(T-t)} \tag{9.2}$$

We focus exclusively on contango markets in this paper, as backwardation scenarios are less common and involve different mechanics.

9.1.2 Capital Inefficiency of Traditional Basis Arbitrage

Traditional basis arbitrage requires holding positions until maturity regardless of market movements, resulting in capital inefficiency. The basis fluctuates prior to maturity due to changes in

interest rates, volatility, and investor sentiment. This motivates the central question: *Can we dynamically capture basis profits without holding to maturity?*

9.1.3 Optimal Control to Reinforcement Learning

In Angoshtari and Leung (2018), the basis follows a Brownian bridge with a time-dependent drift that forces the spread to narrow as expiration approaches. Closed-form trading rules arise from solving the Hamilton-Jacobi-Bellman (HJB) equation under this parametric assumption.

However, the SDE approach introduces significant model misspecification risk. The closed-form solutions require strict assumptions on market dynamics, such as mean reversion and constant volatility. When market dynamics deviate from these assumptions, the optimal control strategy is suboptimal. Moreover, the HJB equation is only tractable for low-dimensional problems with simple dynamics.

Instead, we propose a model-free reinforcement learning [4] approach that learns optimal trading policies directly from data without requiring explicit models of price dynamics. Recent work demonstrates RL's effectiveness for trade execution [3] and minimizing slippage relative to traditional TWAP algorithms [2]. By formulating the problem as a discrete-time Markov Decision Process (MDP) and invoking the Bellman optimality relation, we can then apply RL algorithms to approximate the optimal policy for basis arbitrage.

9.2 Background

9.2.1 Markov Decision Process (MDP) Formulation

We formulate the dynamic basis arbitrage problem as a discrete-time Markov Decision Process (MDP). The key components of the MDP are the state space $\mathcal{S}$, action space $\mathcal{A}$, transition dynamics $\mathcal{P}$, reward function $\mathcal{R}$, and discount factor γ. The state space $\mathcal{S}$ aggregates the relevant market information (spot, futures, time-to-maturity, carry costs) and the agent's inventory at time step t. The action space $\mathcal{A}$ comprises the possible trading actions. For basis arbitrage we use $\mathcal{A} = \{\text{Wait}, \text{Enter}, \text{Exit}\}$, where `Wait` holds the current position, `Enter` initiates the long-spot/short-futures trade, and `Exit` closes it. The transition kernel $\mathcal{P}(s'|s,a) = \mathbb{P}(s_{t+1} = s' \mid s_t = s, a_t = a)$ captures how the market reacts to actions; in a model-free setting it is unknown and learned implicitly. The stepwise reward function $\mathcal{R}(s,a)$ encodes the immediate profit and loss from taking action a in state s. The discount factor $\gamma \in [0,1]$ controls the weight placed on future rewards relative to immediate ones.

9.2.2 Objective Function and Bellman Optimality Equation

Given the context of basis arbitrage, where the trading horizon is finite (up to contract maturity T), we define the objective of the RL agent as maximizing the expected cumulative discounted reward over the trading period. The agent's goal is to select a sequence of actions that maximizes the expected cumulative discounted return,

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{T-t} \gamma^k R_{t+k+1} \mid s_t = s\right] \tag{9.3}$$

where R_{t+k+1} is the reward received at time step $t+k+1$ after taking action a_{t+k} in state s_{t+k}, following policy π. The optimal value function $V^*(s) = \max_{\pi} V^{\pi}(s)$ represents the maximum expected return achievable from state s. The optimal *action-value function* $Q^*(s,a)$ gives the expected return of taking action a in state s and following the optimal policy thereafter:

$$Q^*(s,a) = \mathbb{E}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a') \mid s_t = s, a_t = a\right]. \tag{9.4}$$

which is the *Bellman optimality equation*. This recursive relationship states that the optimal value of taking action a in state s equals the immediate reward plus the discounted value of the best action in the next state.

9.2.3 Optimal Policy Determination

Once the optimal action-value function $Q^*(s,a)$ is known (or sufficiently approximated), the optimal trading policy $\pi^*(s)$ is derived by acting greedily with respect to the Q-values:

$$\pi^*(s) = \arg\max_{a\in\mathcal{A}} Q^*(s,a). \tag{9.5}$$

Because Q^* itself satisfies the Bellman optimality equation above, the greedy policy in this expression chooses, at each state, the action that maximizes expected cumulative reward. In the context of basis arbitrage, where $\mathcal{P}$ is unknown and $\mathcal{S}$ is continuous and high-dimensional, directly solving the Bellman equation is intractable. We therefore resort to Deep Q-Networks (DQN) to approximate $Q^*(s,a)$ numerically.

9.3 Reinforcement Learning for Dynamic Basis Arbitrage

9.3.1 Deep Q-Learning

In Q-learning, the optimal action-value function satisfies the Bellman optimality condition [1]:

$$Q^*(s,a) = \mathbb{E}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a') \mid s_t = s, a_t = a\right]. \tag{9.6}$$

It can be approximated iteratively via the classical matrix update

$$Q^{l+1}(s,a) = Q^l(s,a) + \alpha_t \left(R(s,a) + \gamma \max_{a'\in\mathcal{A}} Q^l(s',a') - Q^l(s,a)\right), \tag{9.7}$$

where $Q^l(s,a)$ denotes the estimate at iteration l, s' is the realized next state after taking a in s, and α_t is the learning rate at time t.

Deep Q-learning replaces the matrix with a neural network $Q(s,a \mid \theta)$ [2]. The network parameters θ are trained to minimize the temporal-difference error between current predictions and bootstrap targets derived from the Bellman equation. A standard squared-error loss at iteration l is

$$L(\theta\,;\,\theta^l) = \left(R(s,a) + \gamma \max_{a'\in\mathcal{A}} Q(s',a' \mid \theta^l) - Q(s,a \mid \theta)\right)^2, \tag{9.8}$$

where θ^l denotes the parameters of the target network (or the previous iterate). Gradient descent on L yields the parameter update

$$\theta^{l+1} = \theta^l - \eta\, \nabla_\theta L(\theta\,;\,\theta^l)\big|_{\theta=\theta^l}. \tag{9.9}$$

9.3.2 Algorithm

We implement the DQN training loop using the Bellman targets from the previous section and the squared-error loss in (9.8). The target-network parameters ϑ are a delayed copy of the online network weights θ; keeping ϑ fixed for several updates stabilizes the bootstrap targets. The flag d_t denotes whether the episode has terminated at time t. An ε-greedy exploration strategy is employed, where with probability ε a random action is selected to encourage exploration, and with probability $1-\varepsilon$ the action maximizing the Q-value is chosen. The exploration rate ε decays over time to shift from exploration to exploitation. The replay buffer $\mathcal{D}$ holds transitions

$(s_t, a_t, r_t, s_{t+1}, d_t)$ up to capacity $\mathcal{N}$; each episode runs for N steps (per-episode horizon), and every C steps the target network is refreshed ($\vartheta \leftarrow \theta$). **Input:** replay buffer $\mathcal{D}$ (size $\mathcal{N}$); online network $Q(\cdot \mid \theta)$; target network $Q(\cdot \mid \vartheta) \leftarrow Q(\cdot \mid \theta)$; stopping condition (e.g., B episodes reached, validation return plateau, or ε below $\varepsilon_{\min}$).

1. While the stopping condition is not met, for each episode $b = 1, \ldots, B$ and each time step $t = 0, \ldots, N-1$:
   (a) With probability ε pick a random action a_t; otherwise $a_t = \arg\max_{a \in \mathcal{A}} Q(s_t, a \mid \theta)$.
   (b) Execute a_t, observe reward r_t and next state s_{t+1}; store $(s_t, a_t, r_t, s_{t+1}, d_t)$ in $\mathcal{D}$.
   (c) Sample a minibatch $\{(s_j, a_j, r_j, s'_j)\}$ from $\mathcal{D}$ and set Bellman targets $y_j = r_j + \gamma \max_{a'} Q(s'_j, a' \mid \vartheta)$ (cf. Bellman relation above).
   (d) Update θ with one gradient step on $\sum_j (y_j - Q(s_j, a_j \mid \theta))^2$ (loss in (9.8)).
   (e) Every C steps, refresh the target network by setting $\vartheta \leftarrow \theta$.
2. Decay exploration: $\varepsilon \leftarrow \tau\varepsilon$; optionally cap at $\varepsilon_{\min}$ and check stopping condition.

9.4 Conclusion

This paper builds a framework for dynamic basis arbitrage using a Deep Q-learning algorithm to approximate the optimal policy. Future work will involve empirical backtesting to evaluate the performance against the traditional basis arbitrage strategy.

Bibliography

[1] Hasselt, H. van, Guez, A., & Silver, D. (2016). *Deep Reinforcement Learning with Double Q-learning*. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1). https://arxiv.org/abs/1509.06461

[2] Liu, X., Xiang, Z., Zheng, Y., Zhou, Z., & Liu, Z. (2018). *Double Deep Q-Learning for Optimal Execution*. arXiv preprint arXiv:1812.06600. https://arxiv.org/abs/1812.06600

[3] Nevmyvaka, Y., Feng, Y., & Kearns, M. (2006). *Reinforcement learning for optimized trade execution*. In Proceedings of the 23rd International Conference on Machine Learning (pp. 673-680).

[4] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press.

Chapter 10

Modelling Stochastic Volatility and Pricing Options with the Heston Model

Valerio Altissimo and Natan Boyd

Edited by: Francesco Papini

Abstract: The Heston model models the volatility of an asset as a stochastic process itself, accepting that it has randomness of its own. We study the Heston model and discuss methods of pricing options using this model, combined with Monte Carlo methodologies and Fast-Fourier Transform methods. We conclude by calibrating the parameters given a time series and discussing the sensitivity of the model option price with respect to the underlying parameters.

10.1 Introduction

In the modelling of options and derivatives, the standard deviation, or volatility, of the underlying asset is a fundamental parameter. This parameter is in fact so important that a fully fledged theory has developed to model it; the development of this theory came about in an attempt to minimise errors from the initial models of options. In the Black-Scholes model [2], the volatility is assumed to be constant, which is, of course, one of the key drawbacks of this model.

To address this critical flaw, time varying volatility models were introduced, like ARCH [9] and GARCH [3]. These models are effective in modelling the volatility clustering observed in asset returns; yet they fail to explain the volatility smile observed across different strikes and maturities. To rectify these issues, systems modelling volatility as a continuous-time stochastic process became necessary. An example of these stochastic volatility models is the Heston model [10]. The combination of this model's analytic solution and the computational capacity of the Cooley-Tukey algorithm [7] allows the pricing of options in this way to be incredibly accurate and swift.

We aim to give a brief account of this model which, though not completely mathematically rigorous, should still showcase the powerful mathematical machinery which is in use. The rest of the article is arranged as follows; Section 2 outlines the necessary background in stochastic calculus, Section 3 introduces the Heston model, Sections 4 discusses how pricing would be done using a Monte Carlo simulation, Section 5 discusses pricing using Fourier theory, Section 6 explores parameter calibration in Python, Section 7 briefly discusses sensitivities and the Heston greeks and Section 8 concludes.

All complete Jupyter Notebooks used for this article can be found on GitHub via the following link `https://github.com/NMB001/heston-project-mathfinmag`.

10.2 Preliminaries: Review of Stochastic Calculus

When considering continuous deterministic time-evolving systems, we tend to model change within these systems using differential equations. Unfortunately for us, randomness tends to play a rather key role in the evolution of financial markets, so to incorporate this inherent uncertainty, it is necessary to replace deterministic differential equations with so-called stochastic differential equations (SDEs). In doing so, we use Brownian motion to model this randomness.

The Heston model is formulated in terms of SDEs, and stochastic calculus (the topic of this section) provides the tools needed to define and analyse these SDEs rigorously.

We begin by recalling the definition of a Brownian motion.

Definition 10.1. [12, p. 7] Let $(\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{t\in T})$ be a filtered probability space. A standard Brownian motion $W = \{W_t\}_{t\geq 0}$ is a continuous-time stochastic process satisfying the following conditions:

1. The process almost surely starts at 0: $\mathbb{P}(W_0 = 0) = 1$,
2. Independent increments: For $0 \leq s < t$, the increment $W_t - W_s$ is independent of $\mathcal{F}_t$,
3. Gaussian increments: $W_t - W_s \sim N(0, t-s)$,
4. Almost surely continuous sample paths.

Suppose that now we naively attempt to define the integral of Brownian motion W_t using Riemann sums, as in classical calculus. Since any Brownian sample path is almost surely nowhere differentiable and has unbounded variation, the Riemann-Stieltjes integral $\int_0^t f(s)\, dW_s$ is almost surely infinite, and hence this approach fails.

Instead, we consider the quadratic variation of a function f over $[0,t]$, defined as

$$[f]_t := \lim_{\|\Pi\|\to 0} \sum_{i=0}^{n-1} \left(f(t_{i+1}) - f(t_i)\right)^2,$$

where $\Pi = \{0 = t_0 < t_1 < \cdots < t_n = t\}$ is a partition of $[0,t]$, and $\|\Pi\|$ denotes the mesh size. For any smooth function f, we have $[f]_t = 0$, but crucially, for Brownian motion W_t, we have

$$[W]_t = \lim_{\|\Pi\|\to 0} \sum_{i=0}^{n-1} \left(W_{t_{i+1}} - W_{t_i}\right)^2 = t,$$

almost surely. We are now able to use this finite quadratic variation of Brownian sample paths to define a new integral, namely the Itô integral [11, p. 130].

We proceed as in Shreve [15, pp. 126-128] similarly to the standard approach to Lebesgue's theory of integration. There, one first defines the integral on a small, straightforward class of functions (namely simple functions) before extending to the general case by approximation.

Definition 10.2 (Simple adapted process)**.** [13, p. 26] A *simple adapted process* is a stochastic process of the form

$$H_t(\omega) = \sum_{i=0}^{n-1} H_i(\omega)\, \mathbf{1}_{(t_i, t_{i+1})}(t),$$

where $0 = t_0 < t_1 < \cdots < t_n = T$ is a partition of $[0,T]$, and each random variable H_i is $\mathcal{F}_{t_i}$-measurable.

For such a process, we may then define the Itô integral with respect to Brownian motion.

Definition 10.3 (Itô integral for simple processes)**.** Let H_t be a simple adapted process as above. Then the Itô integral of H_t with respect to Brownian motion is defined by

$$\int_0^T H_t\, W_t := \sum_{i=0}^{n-1} H_i\, (W_{t_{i+1}} - W_{t_i}).$$

We have now defined the Itô integral for simple adapted processes. Suppose H is an adapted process that is not simple but which can be approximated in L^2 by a sequence of simple processes $H^{(n)}$. Then, the sequence of integrals

$$\int_0^T H_t^{(n)}\, W_t$$

is Cauchy in L^2, and hence converges to a well-defined limit in L^2.

Definition 10.4 (Itô integral)**.** [13, p. 29] For any H an adapted process, the Itô integral is defined as the L^2-limit of integrals of simple adapted processes approximating H. That is,

$$\int_0^T H_t\, W_t := \lim_{n\to\infty} \int_0^T H_t^{(n)}\, W_t,$$

where $H^{(n)}$ is a sequence of simple adapted processes converging to H in L^2.

Definition 10.5 (Square-integrable adapted processes)**.** An adapted process H_t on $[0, T]$ is said to be square-integrable if

$$\mathbb{E}\left[\int_0^T H_t^2\, t\right] < \infty.$$

We will denote the collection of all such processes by $L^2_{\text{ad}}([0, T])$.

Theorem 10.1 (Itô Isometry)**.** [15, p. 134] For all $H \in L^2_{ad}([0, T])$,

$$\mathbb{E}\left[\left(\int_0^T H_t W_t\right)^2\right] = \mathbb{E}\left[\int_0^T H_t^2 t\right].$$

Due to the Itô isometry, the definition of the Itô integral is unambiguous. That is, the limit exists and does not depend on the approximating sequence. Thus, we have successfully extended the definition of the Itô integral from simple processes to the much broader class of square-integrable adapted processes.

We now have the background to understand one of the most key results of stochastic calculus, and subsequently in the theory of pricing options. In classical calculus, one of the most fundamental tools is the chain rule, which allows us to compute the differential of a function of a differentiable variable. In stochastic calculus, the analogous result is the so-called *Itô lemma* which, unlike the classical chain rule, contains an addition term arising from quadratic variation.

Theorem 10.2 (Itô's Lemma)**.** [13, pp. 44-45] Let X_t be an Itô process of the form

$$X_t = \mu_t\, t + \sigma_t\, W_t,$$

where μ_t, σ_t are adapted processes. If $f \in C^2(\mathbb{R})$, then

$$f(X_t) = f'(X_t)\, X_t + \tfrac{1}{2} f''(X_t)\, [X]_t.$$

In particular,

$$f(X_t) = f'(X_t)\mu_t\, t + f'(X_t)\sigma_t\, W_t + \tfrac{1}{2} f''(X_t)\sigma_t^2\, t.$$

Ito's lemma is arguably the central tool for manipulating stochastic processes and, crucially, incorporates the effect of the variance of the stochastic term. It allows us to derive differential equations for functions of stochastic processes, which in turn, provides a natural gateway to SDEs.

Definition 10.6. [15, p. 143] A *stochastic differential equation* (SDE) is an equation of the form

$$X_t = \mu(X_t, t)\, t + \sigma(X_t, t)\, W_t,$$

where μ is the drift coefficient and σ the diffusion coefficient.

Remark 10.1. Implicitly, the general SDE $X_t = \mu(X_t, t)\, t + \sigma(X_t, t)\, W_t$ represents the integral equation

$$X_t = X_0 + \int_0^t \mu(s, X_s)\, s + \int_0^t \sigma(s, X_s)\, W_s$$

but in this article (and in most standard literature) the former notation is used for conciseness.

It turns out that these SDEs have been remarkably useful in the theory and development of option pricing models as they provide a rigorous framework for capturing the random nature of asset price dynamics. By modelling prices as continuous-time stochastic processes, SDEs formally account for drift, volatility and randomness. To illustrate this, we consider two important examples, namely geometric Brownian motion in the Black-Scholes framework and the Ornstein-Uhlenbeck process.

Example 10.1 (Geometric Brownian Motion)**.** The SDE

$$S_t = \mu S_t\, t + \sigma S_t\, W_t$$

has the explicit solution

$$S_t = S_0 \exp\left(\left(\mu - \tfrac{1}{2}\sigma^2\right) t + \sigma W_t\right)$$

and is the standard model for asset prices in the Black-Scholes framework.

Example 10.2 (Ornstein-Uhlenbeck Process)**.** The SDE

$$X_t = \kappa(\theta - X_t)\, t + \sigma\, W_t$$

has solution

$$X_t = \theta + (X_0 - \theta)e^{-\kappa t} + \sigma \int_0^t e^{-\kappa(t-s)}\, W_s.$$

This describes a mean-reverting process and provides the prototype for the volatility process in the Heston model. The Heston model modifies the Ornstein-Uhlenbeck structure by making the diffusion term proportional to $\sqrt{v_t}$, the square-root of the instantaneous variance instead of constant. This ensures positivity of variance and leads to a Cox-Ingersoll-Ross (CIR) process.

We finish this section by introducing a very important tool in the analysis of SDEs, deriving from the world of quantum physics and Feynman's path integrals, which are incredibly related to the problems mathematical finance faces. This tool can allow us to solve an SDE by considering a related PDE, which is often simpler to solve.

Theorem 10.3 (Feynman-Kac)**.** [11, p. 268] Suppose $\mathbf{x}_t$ follows an n-dimensional stochastic process

$$\mathbf{x}_t = \boldsymbol{\mu}(\mathbf{x}_t, t)t + \boldsymbol{\sigma}(\mathbf{x}_t, t)\mathbf{W}_t,$$

where $\mathbf{x}_t, \boldsymbol{\mu}(\mathbf{x}_t, t) \in \mathbb{R}^n$, $\mathbf{W}_t$ is a vector of m independent Brownian motions and $\boldsymbol{\sigma}(\mathbf{x}_t, t)$ is a volatility matrix of dimension $n \times m$. Suppose further the generator of the process is defined as the operator

$$\mathcal{A} = \sum_{i=1}^{n} \mu_i \partial_i + \frac{1}{2} \sum_{i,j=1}^{n} (\boldsymbol{\sigma}\boldsymbol{\sigma}^T)_{ij} \partial^2_{i,j}.$$

Then for $r, f : \mathbb{R}^n \times [0, T] \to \mathbb{R}$, the PDE given by

$$\partial_t f + \mathcal{A} f - rf = 0,$$

and boundary $f(\mathbf{x}_T, T)$, has the solution

$$f(\mathbf{x}_t, t) = \mathbb{E}\left[\exp\left(-\int_t^T r(x_u, u)du\right) f(\mathbf{x}_T, T) \mid \mathcal{F}_t\right].$$

10.3 Deriving the Heston Model

The Heston model assumes that the stock price follows a Black-Scholes–like model, but the volatility itself follows a stochastic process. Thus, the model is characterised by the following pair of stochastic differential equations, one accounting for the randomness of the underlying stock price and one for that of the volatility. We define the SDEs, derive the relevant PDE, obtain the characteristic functions and show that the Black-Scholes model is a subcase of the Heston model.

For mathematical completeness, we are currently working in the filtered probability space $(\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{t\in T})$; where Ω is a sample space containing all possible continuous price trajectories, $\mathcal{F}$ is the event space, or σ-algebra, generated by all possible market events, $\mathbb{P}$ is the historical probability measure and $\{\mathcal{F}_t\}_{t\in T}$ is the relevant filtration and takes into account how the market events evolve through time.

Definition 10.7 (The Heston model)**.** [10] The stock price at time t, $S(t)$, and the volatility at time t, $v(t)$, follow the SDEs

$$S_t = \mu S_t t + \sqrt{v_t} S_t W_t, \tag{10.1}$$
$$v_t = \kappa(\theta - v_t)t + \sigma\sqrt{v_t} B_t, \tag{10.2}$$

and $S(0) = S_0$, $v(0) = v_0$. Where $\mu \in \mathbb{R}$, $S_0, v_0, \kappa, \theta, \sigma \in \mathbb{R}_{>0}$ and W_t, B_t are two Brownian motions with a correlation of $\rho \in [-1, 1]$.

In the model, μ, is the drift of the stock; κ the mean reversion speed for the variance; θ the mean reversion level of the variance and σ the volatility of the volatility. Essentially, the equations state that the change in volatility at time t is given by how far $v(t)$ is from θ, multiplied by κ, added to a random increment with mean 0 and variance $\sigma^2 v_t$. Further, the change in the stock price at time t is given by μS_t added to a random increment with mean 0 and variance $v(t)S_t^2$. To justify calling this a Black-Scholes–like model, note the similarities between (1) and the Black-Scholes model.

Definition 10.8 (The Black-Scholes model)**.** [2] The stock price follows the SDE

$$S_t = \mu S_t t + \sigma S_t W_t, \tag{10.3}$$

where $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}_{>0}$ and W_t is a Brownian motion.

The similarities with the Black-Scholes model, though striking in this instance, do not end there. Much like the Black-Scholes model, the Heston model has a closed form solution to the system of SDEs, which is an incredibly useful and rare property. Now, we write the Heston model as a PDE, much like the famous Black-Scholes equation, though more complicated. We split the argument given by [14, pp. 5–10] and the application of theorem 10.3 into multiple subsections to make it easier to follow. It is important to note that the option we are pricing is assumed to not pay dividends.

10.3.1 Risk-Neutral Measure

We want to consider this problem in the risk-neutral probability measure, largely to ease up the notation and also to be able to apply Feynman-Kac correctly. Assume the existence of a risk-free rate, $r \in \mathbb{R}$, let $\mathbb{Q}$ the risk-neutral probability measure on the same sample space, event space and filtration as before. This probability measure takes into account how much more (or less) than a guaranteed baseline something is worth. We adjust (1) and (2) to this new probability space, hence the risk-neutral SDEs are

$$S_t = rS_t t + \sqrt{v_t} S_t \tilde{W}_t, \tag{10.4}$$

$$v_t = [\kappa(\theta - v_t) - \lambda v_t]t + \sigma\sqrt{v_t}\tilde{B}_t. \tag{10.5}$$

Where $\tilde{W}_t = W_t + \frac{\mu - r}{\sqrt{v_t}}t$ and $\tilde{B}_t = B_t + \frac{\lambda\sqrt{v_t}}{\sigma}t$ are the risk-neutral Brownian motions. Note λv_t is the volatility risk premium. We go on to price the option in this space.

10.3.2 Log Transform of the Price

In order to exploit the fact S_t is a coefficient to both the deterministic and stochastic term, we consider the log-price of the system and let $x_t = \ln S_t$. This simplifies (4), which, by theorem 10.2, becomes

$$x_t = (r - \frac{v_t}{2})t + \sqrt{v_t} W_t. \tag{10.6}$$

To apply the theorem 10.3, we need the Brownian motion processes to be independent, so consider $W_t^{(1)}$ and $W_t^{(2)}$ two independent Brownian motion such that

$$\tilde{W}_t = W_t^{(1)} \text{ and } \tilde{B}_t = \rho W_t^{(1)} + \sqrt{1 - \rho^2} W_t^{(2)}.$$

So, the pair of SDEs (4) and (5) can now be considered as the system

$$\begin{aligned} x_t &= (r - \frac{v_t}{2})t + \sqrt{v_t} W_t^{(1)}, \\ v_t &= [\kappa(\theta - v_t) - \lambda v_t]t + \sigma\sqrt{v_t}\rho W_t^{(1)} + \sigma\sqrt{v_t(1 - \rho^2)} W_t^{(2)}. \end{aligned}$$

Thus, for $\mathbf{x}_t = (x_t, v_t)$, $\mathbf{W}_t = (W_t^{(1)}, W_t^{(2)})$, $\boldsymbol{\mu}(\mathbf{x}_t, t) = \left(r - \frac{v_t}{2}, \kappa(\theta - v_t) - \lambda v_t\right)$ and $\boldsymbol{\sigma}(\mathbf{x}_t, t) = \begin{pmatrix} \sqrt{v_t} & 0 \\ \sigma\sqrt{v_t}\rho & \sigma\sqrt{v_t(1 - \rho^2)} \end{pmatrix}$ it is the case that

$$\mathbf{x}_t = \boldsymbol{\mu} t + \boldsymbol{\sigma} \mathbf{W}_t.$$

10.3.3 The PDE

All that is now necessary is to apply Feynman-Kac and derive the PDE for the log-price process. Thus, by theorem 10.3, the PDE for the Heston model under these variables is

$$\partial_t U + \mathcal{A}_x U = rU,$$

where U is the option price, $\mathcal{A}_x$ is the generator of the process and is, in this case,

$$\mathcal{A}_x = (r - \frac{v_t}{2})\partial_x + [\kappa(\theta - v_t) - \lambda v_t]\partial_v + \frac{v_t}{2}\partial_x^2 + \sigma v_t \rho \partial_{xv}^2 + \frac{\sigma^2 v_t}{2}\partial_v^2.$$

So, substituting back for S, gives that the process satisfies $\partial_t U + \mathcal{A}U = rU$ where

$$\mathcal{A} = rS_t\partial_S + [\kappa(\theta - v_t) - \lambda v_t]\partial_v + \frac{v_t S_t^2}{2}\partial_S^2 + \sigma v_t \rho S_t \partial_{Sv}^2 + \frac{\sigma^2 v_t}{2}\partial_v^2.$$

10.3.4 The Characteristic Functions

We thus have a PDE with an analytic solution. The fact that an analytic solution exists is not obvious and finding such a solution is, though not hard, tedious. We refer the reader to [14, pp. 12-16] for a detailed treatise on this and we state the characteristic functions, f_j, and their parameters, for completeness. The characteristic functions are, for $j = 1, 2$

$$f_j(\phi; x, v) = \exp\left[C_j(\tau, \phi) + D_j(\tau, \phi)v_t + i\phi x\right], \tag{10.7}$$

$$C_j(\tau, \phi) = ri\phi\tau + \frac{\kappa\theta}{\sigma^2}\left[(b_j - \rho\sigma i\phi - 2\ln\left(\frac{1 - g_j e^{d_j\tau}}{1 - g_j}\right)\right], \tag{10.8}$$

$$D_j(\tau, \phi) = \frac{b_j - \rho\sigma i\phi + d_j}{\sigma^2}\left(\frac{1 - e^{d_j\tau}}{1 - g_j e^{d_j\tau}}\right). \tag{10.9}$$

Where $b_1 = \kappa + \lambda - \rho\sigma, b_2 = \kappa + \lambda, u_1 = 1/2, u_2 = -1/2$,
$d_j = \sqrt{(\rho\sigma i\phi - b_j)^2 - \sigma^2(2u_j i\phi - \phi^2)}$ and $g_j = \frac{b_j - \rho\sigma i\phi + d_j}{b_j - \rho\sigma i\phi - d_j}$. Note $\tau = T - t$ is the remaining time to maturity and ϕ is the integration variable.
From the characteristic functions, we can compute the in-the-money probabilities, P_j, which are given by

$$P_j = \frac{1}{2} + \frac{1}{\pi}\int_0^\infty \text{Re}\left[\frac{e^{-i\phi} f_j(\phi; x_t, v_t)}{i\phi}\right] d\phi.$$

We thus have that the European call and put option prices are, respectively,

$$C(K) = S_t P_1 - K e^{-r\tau} P_2,\ P(K) = C(K) + K e^{-r\tau} - S_t. \tag{10.10}$$

10.3.5 The Black-Scholes case

To end the section, we showcase how the Black-Scholes model is a subcase of the Heston model, as $\sigma \to 0$. Taking this limit, equation (2) becomes $v_t = \kappa(\theta - v_t)t$, producing a deterministic model of the variance. Further, setting $\theta = v_0$, the initial variance, will give that the model has a constant volatility of $v_t = v_0$ for all t. Substituting these parameters into the generator for the Heston model gives

$$\mathcal{A}_{BS} = \mu S_t \partial_S + \frac{v_0 S_t^2}{2}\partial_S^2,$$

which is the generator for the Black-Scholes model. Hence, we have successfully shown that the Heston model reduces to the Black-Scholes model.

10.4 SDE Discretisation and Monte Carlo Pricing

In this section, we move from the analytical development of the Heston model to its numerical implementation, considering a straightforward Monte Carlo approach. We will describe how the SDEs for the spot and volatility processes are discretised, and will build a Monte Carlo pricing model using these discretisation schemes.

Consider the general SDE of the form

$$X_t = \mu(X_t, t)\, t + \sigma(X_t, t)\, W_t,$$

where W_t is a standard Brownian motion. Analytical solutions are rarely available, so various numerical discretisation schemes have been developed to approximate paths. The simplest approach is the Euler-Maruyama scheme, which for time step Δt takes the form

$$X_{t+\Delta t} = X_t + \mu(X_t, t)\, \Delta t + \sigma(X_t, t)\, \Delta W_t,$$

where $\Delta W_t \sim \mathcal{N}(0, \Delta t)$. Although this scheme provides a first-order weak approximation, it may generate negative values for processes that are strictly positive in continuous time, such as the asset price or volatility process in the Heston model. To preserve positivity, one may apply the log-Euler discretisation for the asset price. Applying Itô's lemma to $\ln S_t$ gives

$$(\ln S_t) = \left(\mu - \tfrac{1}{2} v_t\right) t + \sqrt{v_t}\, W_t.$$

Discretising this equation with Euler-Maruyama yields

$$\ln S_{t+\Delta t} = \ln S_t + \left(\mu - \tfrac{1}{2} v_t\right) \Delta t + \sqrt{v_t}\, \Delta W_t,$$

so

$$S_{t+\Delta t} = S_t \exp\left[\left(\mu - \tfrac{1}{2} v_t\right) \Delta t + \sqrt{v_t}\, \Delta W_t\right].$$

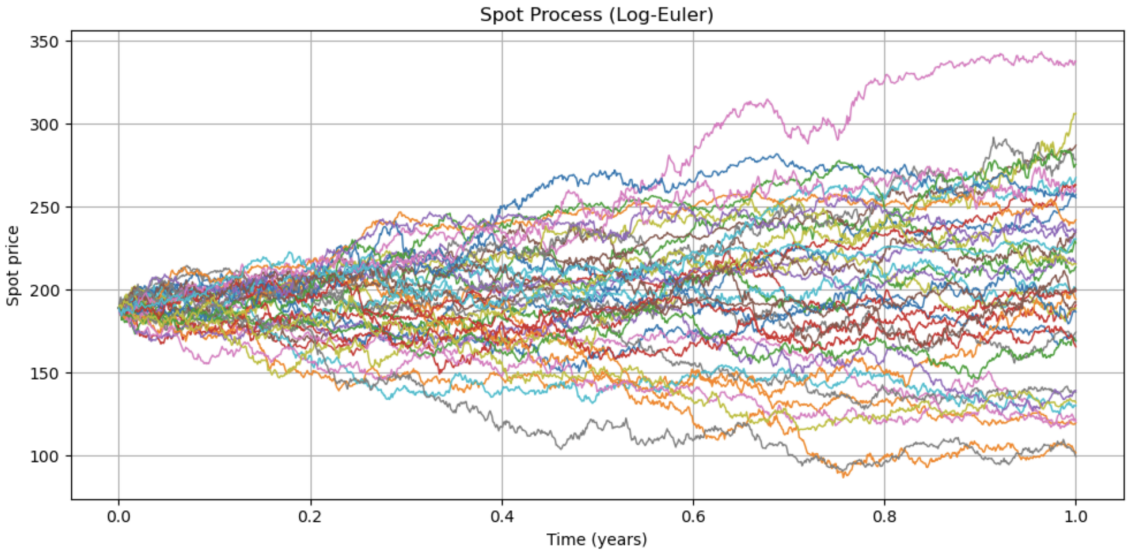


Figure 10.1: Sample Trajectories of Spot Process using Log-Euler

For the volatility process in the Heston model,

$$v_t = \kappa(\theta - v_t)\, t + \sigma \sqrt{v_t}\, B_t,$$

we must again ensure that numerical approximations remain non-negative. A simple Euler-Maruyama discretisation,

$$v_{t+\Delta t}^{(\mathrm{EM})} = v_t + \kappa(\theta - v_t)\, \Delta t + \sigma \sqrt{v_t}\, \Delta B_t,$$

will produce negative values for $v_{t+\Delta t}^{(\mathrm{EM})}$ when Δt is not sufficiently small or when v_t is close to zero so this will not be sufficient. To address this, the *full truncation Euler scheme* modifies the drift

and diffusion terms by replacing v_t with its positive part, $\max(v_t, 0)$, while leaving the update itself untruncated. The discretised volatility process is therefore defined as

$$v_{t+\Delta t} = v_t + \kappa\big(\theta - \max(v_t, 0)\big)\Delta t + \sigma\sqrt{\max(v_t, 0)}\,\Delta B_t,$$

where $\Delta B_t \sim \mathcal{N}(0, \Delta t)$.

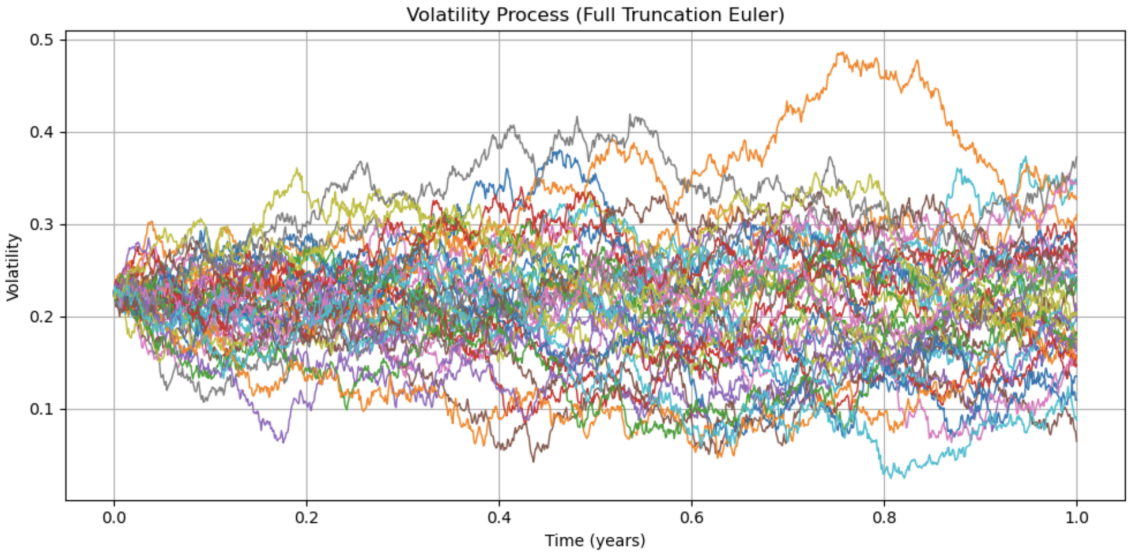


Figure 10.2: Sample Trajectories of Volatility Process using Full Truncation Euler

We now have the mathematical background to implement a Monte-Carlo option pricing model in Python. For each path, we use log-Euler discretisation for the asset price SDE and the full truncation Euler scheme for the volatility SDE. We then compute the mean payoff of each path and discount back to the present value.

Using this method to price a Nvidia call (strike of US$190 and spot of US$188.85 at the time of writing) the Monte Carlo simulation estimates the option price at US$20.696 per share, with a standard error of US$0.195 using 10,000 paths. For comparison, QuantLib's engine returns a price of US$20.603 per share.

10.5 Pricing using the Fourier Transform

10.5.1 Fourier Theory

Fourier Transform to price Options

We use Fourier transform methods to price a call option. To do this, we first give a brief rundown of the relevant aspects of Fourier analysis and the Cooley - Tukey algorithm. We begin by defining the Fourier transform (FT) according to the following conventions.

Definition 10.9 (Fourier Transform)**.** [5]For $f \in L^1(\mathbb{R}, \mathbb{R})$ the Fourier transform of f is given by

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)e^{-i\xi x}dx. \tag{10.11}$$

And the inverse Fourier transform as

$$\check{f}(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(\xi)e^{ix\xi}d\xi. \tag{10.12}$$

For a nice enough function, knowing the function and knowing the amplitude of the sine and cosine curves which uniquely add to this function at every frequency, is the same thing. Loosely, the FT is a linear operator which switches between these two characterisations of the function, the inverse Fourier transform (IFT) switches back, as the name suggests. Note, in this text, like much of the mathematical finance literature, we assume the FT and IFT are exact inverses of each other, as in $\check{\hat{f}} = \hat{\check{f}} = f$. This is incredibly useful, though technically incorrect, but is not what interests us about this operation, we are interested in exploiting the computational power of the Cooley - Tukey algorithm. We follow the derivation given by [5] and refer to that work for further details.

We are interested in pricing a European call option with strike price K, maturity T and time to maturity τ. Let $k = \log K$ and $C_T(k)$ the value of this option. Consider the risk-neutral density function of the log of the asset price at time T, s_T, as $q_T(s)$. The characteristic function of a density function is its inverse FT, so ϕ_T is

$$\phi_T(\xi) = \mathbb{E}\left[e^{i\xi s}\right] = \check{q_T}(\xi). \tag{10.13}$$

The value of a call option is the discounted present value of the expectation of $S_T - K$, if $S_T > K$, and so is given by the relation

$$C_T(k) = e^{-r\tau}\mathbb{E}_{S_T}\left[\chi_{S_T>K}(S_T)(S_T - K)\right] = \int_k^\infty e^{-r\tau}(e^s - e^k)q_T(s)ds, \tag{10.14}$$

where $\chi_{S_T>K}$ is 1 if $S_T > K$ and 0 otherwise. However, since $C_T(k)$ tends to S_0, the spot price, as $k \to -\infty$, this function is not square integrable. Since we need the function to be square integrable, we have to consider a slightly different function which instead tends to 0 as $k \to -\infty$ fast enough so that this new function is square integrable. An exponential might seem overkill but ends up being correct, the relevant damping is given by

$$c_T(k) = e^{\alpha k}C_T(k)$$

for $\alpha > 0$ such that $\mathbb{E}\left[S_T^{\alpha+1}\right] < \infty$, a necessary condition on α which will ensure the modified call price is integrable. Introducing this α, instead of considering $\alpha = 1$, might seem arbitrary, but it proves to be the correct way of analysing the problem. We now study this damped call price and consider its inverse Fourier transform, it is the case that

$$C_T(k) = e^{-\alpha k}\hat{c_T}(k) = \frac{e^{-\alpha k}}{2\pi}\int_{-\infty}^\infty \check{c_T}(\xi)e^{-i\xi k}d\xi, \tag{10.15}$$

we will find an analytic representation for $\check{c_T}(\xi)$ and so will be able to apply the FFT to calculate the call price. The derivation is as follows

$$\begin{aligned}
\check{c_T}(\xi) &= \int_{-\infty}^\infty e^{i\xi k}\int_k^\infty e^{\alpha k - r\tau}(e^s - e^k)q_T(s)dsdk, \\
&= e^{-r\tau}\int_{-\infty}^\infty q_T(s)\int_{-\infty}^s e^{i\xi k + \alpha k}(e^s - e^k)dkds, \\
&= e^{-r\tau}\int_{-\infty}^\infty q_T(s)\left[\frac{e^{(\alpha+1+i\xi)s}}{\alpha + i\xi} - \frac{e^{(\alpha+1+i\xi)s}}{\alpha + 1 + i\xi}\right]ds, \\
&= \frac{e^{-r\tau}\phi_T(\xi - (\alpha+1)i)}{\alpha^2 + \alpha - \xi^2 + i(2\alpha+1)\xi}.
\end{aligned}$$

Note, this function is finite provided $\phi_T(-(\alpha+1)i)$ is finite, which is exactly the condition we placed on α. This representation is even valued in its real part, and so, since C_T is real, the imaginary part of the right side of (15) must be 0. We can thus simplify (15) to give

$$C_T(k) = \frac{e^{-\alpha k}}{\pi}\mathrm{Re}\left[\int_0^\infty e^{-ik\xi}\check{c_T}(\xi)d\xi\right]. \tag{10.16}$$

For the Heston model, ϕ_T is known analytically and is given by the f_i in (7), so giving an analytic formula for $\check{c_T}$. Importantly, this procedure does not work for options with very short maturities, for technical reasons for which the Fourier inversion procedure, and so equation (15), is highly unstable and so does not always hold. Thus, these need to be priced separately and we refer to [5] for detailed analysis of the problem.

The Discrete Fourier Transform

We now give a brief rundown of how the required integral in (16) can be numerically approximated, to do so the discrete Fourier transform (DFT) must be defined.

Definition 10.10. [5] For $\mathbf{x} \in \mathbb{C}^N$, its discrete Fourier transform is, for $k = 1, \dots, N$,

$$\hat{\mathbf{x}}_k = \sum_{j=1}^{N} e^{-i\frac{2\pi}{N}(j-1)(k-1)}\mathbf{x}_j. \tag{10.17}$$

And its inverse discrete Fourier transform is, for $k = 1, \dots, N$,

$$\check{\mathbf{x}}_k = \frac{1}{N}\sum_{j=1}^{N} e^{i\frac{2\pi}{N}(j-1)(k-1)}\mathbf{x}_j. \tag{10.18}$$

The DFT can be used to numerically approximate the Fourier transform, in much the same way a finite sum can be used to approximate an integral. A simple study of (16) results that the integrand is $O(1/\xi^2)$, so a truncation can be made in the domain of integration resulting in an arbitrarily small error, provided the truncaton is large enough. This justifies using a finite sum as an approximation. We consider, for a uniform spacing $\eta > 0$ to be chosen later, the Riemann sum approximation

$$C_T(k) \approx \frac{e^{-\alpha k}}{\pi}\mathrm{Re}\left[\sum_{j=1}^{N} e^{-i\eta(j-1)k}\check{c_T}(\eta(j-1))\eta\right]. \tag{10.19}$$

To use the DFT, we require the frequency space (in this case k) to also be discretised, so we discretise the log strike levels around 0 with spacing $\lambda > 0$, resulting in a uniform mesh of $[-N\lambda/2, N\lambda/2]$. We place a constraint on η, λ and N, which we can do since they have not yet been chosen, as

$$\lambda\eta = \frac{2\pi}{N}.$$

Letting $k_u = -\frac{N\lambda}{2} + \lambda(u-1)$,

$$C_T(k_u) \approx \frac{e^{-\alpha k_u}}{\pi}\mathrm{Re}\left[\sum_{j=1}^{N} e^{-i\eta(j-1)k_u}\check{c_T}(\eta(j-1))\eta\right] \tag{10.20}$$

$$= \frac{e^{-\alpha k_u}}{\pi}\mathrm{Re}\left[\sum_{j=1}^{N} e^{-i\frac{2\pi}{N}(j-1)(u-1)}e^{ib\eta(j-1)}\check{c_T}(\eta(j-1))\eta\right]. \tag{10.21}$$

This is a direct application of the DFT, which we will learn to compute efficiently soon. It is clear that this method involves a lot of extra constants – α, η, λ and N – the choosing of which is left to the discretion of the reader. The choice of these constants is largely dependent on the model which is being used to price these options, however, to be able to choose a larger η, and thus a smaller λ, techniques such as Simpson's rule can be employed.

The Cooley-Tukey Algorithm

Computing discrete Fourier transforms is the point of fast Fourier transform algorithms, of which, the Cooley - Tukey algorithm [7] is the most famous example. For N a power of 2, the Cooley - Tukey algorithm splits the input vector into two vectors of length $N/2$, calculates their DFT and recombines the result using the periodicity of the complex exponential to give the required result. A rough derivation is as follows.

Let $\mathbf{x} \in \mathbb{C}^N$, for N a power of 2, and let $\mathbf{E} = (x_2, x_4, \ldots, x_N)$ and $\mathbf{O} = (x_1, \ldots, x_{N-1})$. Thus, for $k = 1, \ldots, N/2$

$$\hat{\mathbf{x}}_k = \sum_{j=1}^{N/2} e^{-i\frac{2\pi}{N}(2j-1)(k-1)}\mathbf{x}_{2j} + \sum_{j=1}^{N/2} e^{-i\frac{2\pi}{N}(2j)(k-1)}\mathbf{x}_{2j+1} = e^{-\frac{2\pi i}{N}k}\hat{\mathbf{E}}_k + \hat{\mathbf{O}}_k.$$

This is useful, but the real utility of this method derives from the fact that $\hat{\mathbf{x}}_{k+N/2}$ can also be obtained from $\hat{\mathbf{E}}_k$ and $\hat{\mathbf{O}}_k$, as follows

$$\begin{aligned}
\hat{\mathbf{x}}_{k+N/2} &= \sum_{j=1}^{N/2} e^{-i\frac{2\pi}{N}(2j-1)(k+N/2-1)}\mathbf{x}_{2j} + \sum_{j=1}^{N/2} e^{-i\frac{2\pi}{N}(2j)(k+N/2-1)}\mathbf{x}_{2j+1}, \\
&= e^{-\frac{2\pi i}{N}k+\pi i}\sum_{j=1}^{N/2} e^{-2\pi ji}e^{-i\frac{2\pi}{N}(2j)(k-1)}\mathbf{x}_{2j} + \sum_{j=1}^{N/2} e^{-2\pi mi}e^{-i\frac{2\pi}{N}(2j)(k-1)}\mathbf{x}_{2j+1}, \\
&= -e^{-\frac{2\pi i}{N}k}\hat{\mathbf{E}}_k + \hat{\mathbf{O}}_k.
\end{aligned}$$

This process can clearly be iterated, provided N is a power of 2. This reduces the complexity of calculating the DFT from $O(N^2)$ to $O(N \log N)$, a considerable reduction when N is very large.

10.5.2 Pricing Model

We now discuss a semi-analytical implementation of the Carr-Madan Fourier pricing method for European call options under the Heston model. We say 'semi-analytical' since we use the closed form of the characteristic function $\phi_T(\xi)$ of the log-asset price $s_T = \log S_T$ under the risk-neutral measure, but evaluate the integral of this characteristic function numerically. Under the Heston model, this characteristic function is given by

$$\phi_T(u) = \exp\left(C(u,T) + D(u,T)v_0 + iu\log S_0\right),$$

where v_0 is the initial variance, S_0 the initial asset price, and $C(u,T)$, $D(u,T)$ are given in closed form [10].

The FFT procedure [5] allows us to compute call prices across a uniform grid of log-strike values k_u. The numerical procedure follows equation 10.17, discretising the integral and applying the FFT as described in the previous section. We use the following parameter values.

```
s0 = 188.85; K = 190; r = 0.05; tau = 1.0; kappa = 2.0; theta = 0.05;
sigma = 0.3; rho = -0.5; v0 = 0.05
```

Using this model, we return a price of US$20.625 for this call. In comparison, our Monte Carlo pricing model returns US$20.696, while QuantLib's semi-analytical Heston pricing engine gives US$20.603.

Crucially, this FFT based approach uses the closed-form availability of the characteristic function and thus is significantly more computationally efficient than a Monte Carlo simulation. In fact, this method is so much more efficient that we can plot a smooth surface of the call price against strike and maturity as in Figure 10.3.

On the other hand, the Monte-Carlo approach allows for far more flexibility for more complex payoffs or exotics.

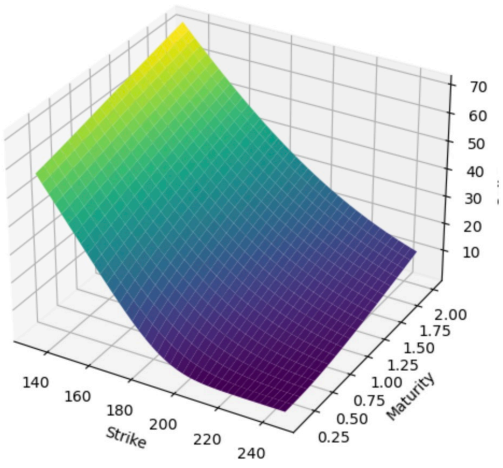


Figure 10.3: Call Price against Strike and Maturity

10.6 Parameter Calibration in Python

We calibrate parameters for European call options for Nvidia with a maturity of the 18th June 2026, τ, and the available strike prices for that date, $K = \{K_1, \ldots, K_n\}$. The spot price is US$183.69 per share. Note that when this is done in practice many more maturities are used, so our results fit incredibly well also because of this fact – some overfitting is in play. We use QuantLib's semi-analytical pricing engine as Monte Carlo simulation with too few paths means that the function would be unstable in minimisation, and would take too long with the necessary paths. We use the simple loss function

$$\text{Loss}(\tau, K, \Theta) = \frac{1}{n} \sum_{k \in K} |C_{\tau,k} - C^{\Theta}_{\tau,k}|,$$

where $\Theta = (v_0, \kappa, \theta, \sigma, \rho)$, $C_{\tau,\kappa}$ is the market price and $C^{\Theta}_{\tau,\kappa}$ is the Heston model price. We use Scipy's `optimize.minimize` to find a local minimum of the loss function. The code is in the Github. This gives an estimate of $\hat{\Theta} = (0.14, 2.16, 0.073, 1.10, -0.16)$. Using this estimate, we created a Heston model for the price with these parameters and the output prices for the relevant strikes is plotted in 10.4, showing remarkable similarity to the market prices.

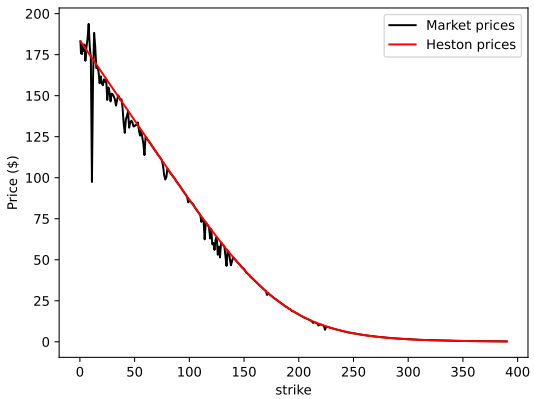


Figure 10.4: Price in USD against Strike Price

To highlight the improvements when compared to the Black-Scholes model, we compute the

Black-Scholes implied volatility of these options and the implied volatility of the modelled options using the Heston model, plotted in 10.5. This somewhat shows a volatility smile and exhibits remarkable similarities for in-the-money options. We compare with the Black-Scholes model, which has constant implied volatility, and highlight the difference particularly for out-the-money options. The fact that the Heston model implied volatility is so different at low strike prices is a clear drawback of the model. The function used to compute the implied volatility is also described in the Github.

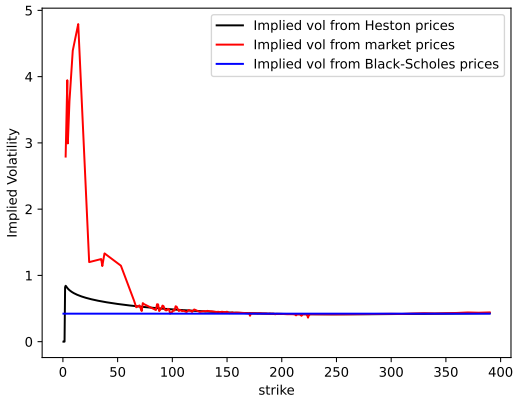


Figure 10.5: Implied Volatility against Strike Price

The similarity of the implied volatility curves is so important that the distance between implied volatilities can be used as a loss function to estimate parameters [14, p. 148], though we do not utilise this here.

10.7 The Heston Greeks

The Greeks of a model investigate the sensitivity of the pricing model to change in a specific parameter; mathematically, they are the partial derivatives of the call (or put) price with respect to the relevant parameter. The most relevant first order Greeks measure sensitivity with respect to the spot price, delta; the risk free rate, rho; the time to maturity, theta; and the implied volatility, vega. The only one we will study here is delta. As the Heston model gives a closed form for the call price, a closed form for most Greeks can easily be computed. In the case for delta, this is, using the earlier notation,

$$\Delta_C = \partial_S C = e^{-q\tau} P_1,$$

where q is the dividend rate of the option. For the model calibrated using the Nvidia options, we plot the delta against the strike price and maturity in figure 10.6.

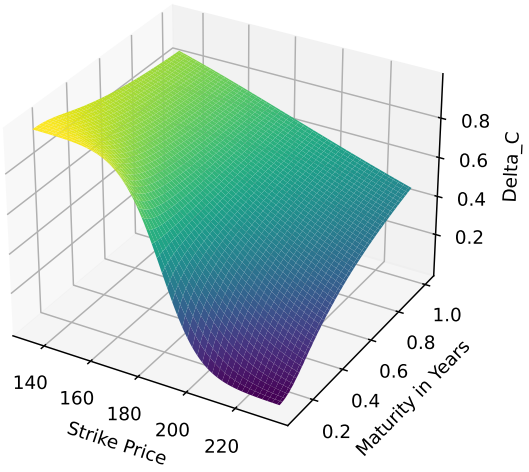


Figure 10.6: Delta against Maturity and Strike price

Considering that the spot price at this time was US$184, figure 10.6 tells us the sensitivity of the call price to a change in spot price is highest when the option is far in-the-money and has a short expiry. Meanwhile, it is lowest when the option is far out-the-money and has a short expiry. Interstingly, at maturity of 1 year, the delta behaves almost linearly with respect to the strike price.

10.8 Conclusion

The Heston model is not perfect, it has some serious drawbacks in the large amount of high-quality data necessary to train it and how slow the numerical integration, necessary to use the analytic solution, can be. Further, the obvious next theoretical question arising from the Heston model is: what if σ is itself stochastic?

These considerations lead to the double Heston model, which models σ as a stochastic process [14, pp. 357-382]. In addition to this, there have been developments in modelling higher moments as stochastic processes, including research into stochastic skew [6]. However, there are other models which try to allow for other market behaviours this model does not explain. These range from the rough Heston model, which seeks to exploit the fact that log-volatility behaves as a fractional Brownian motion [8]. Moreover, stochastic volatility jump diffusion models, like the Bates model [1], seek to take into account unexpected market shocks. The Heston model has also been on the frontier of deep learning initiatives in mathematical finance, this is because the model is well understood and acts as an ideal benchmark case for these more advanced techniques [4].

The Heston model is a stochastic volatility model which has enjoyed success in pricing vanilla options, like the Nvidia ones we priced, and was used especially in the early 2000s. We have outlined some of the theory for the Heston model and the powerful mathematics which it requires; derived the Heston PDE and corresponding characteristic functions which are used to analytically price relevant options and have shown how the Heston model simplifies down to the Black-Scholes model. The Heston model however, has some drawbacks in its computation speed and ability to deal with more advanced market behaviours, this has kept stochastic volatility models forefront of research in mathematical finance.

Bibliography

[1] David S Bates. Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options. *The Review of Financial Studies*, 9(1):69–107, 1996.

[2] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654, 1973.

[3] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[4] Damiano Brigo, Xiaoshan Huang, Andrea Pallavicini, and Haitz Sáez de Ocáriz Borde. Interpretability in deep learning for finance: A case study for the heston model. *Risk Sciences*, 2:100030, 2026. ISSN 2950-6298.

[5] Peter Carr and Dilip Madan. Option valuation using the fast fourier transform. *Journal of computational finance*, 2(4):61–73, 1999.

[6] Peter Carr and Liuren Wu. Stochastic skew in currency options. *Journal of Financial Economics*, 86(1):213–247, 2007.

[7] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[8] Omar El Euch and Mathieu Rosenbaum. The characteristic function of rough heston models. *Mathematical Finance*, 29(1):3–38, 2019.

[9] Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007, 1982.

[10] Steven L Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies*, 6(2):327–343, 1993.

[11] Ioannis Karatzas and Steven Shreve. *Brownian motion and stochastic calculus*, volume 113. springer, 2014.

[12] Peter Mörters and Yuval Peres. *Brownian motion*, volume 30. Cambridge University Press, 2010.

[13] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[14] Fabrice D Rouah. *The Heston model and its extensions in Matlab and C*. John Wiley & Sons, 2013.

[15] Steven E Shreve et al. *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer, 2004.

Chapter 11

How machine learning is being used in the assessment and augmentation of the CAPM Fama-French models

Karthigan Sabanadesan

Edited by: Arsh Mir

Abstract: The Capital Asset Pricing Model (CAPM) and the Fama-French 3 and 5 Factor models remain central tools in asset pricing, yet their empirical limitations and restrictive assumptions have motivated continued refinement. In parallel, machine learning techniques have demonstrated increasing relevance in financial modelling, owing to their capacity to capture complex, non-linear relationships. This article reviews how machine learning methods, including artificial neural networks, recurrent neural networks, long-short-term memory networks, and gated recurrent units, have been applied to evaluate and augment the CAPM, Fama-French 3-Factor, and 5-Factor frameworks. By surveying empirical studies across multiple markets, the article examines whether these techniques improve explanatory and predictive performance, and assesses attempts to combine traditional models with neural networks.

11.1 Introduction

Applying machine learning in financial research papers has been rapidly increasing, from a paltry 2 articles in 1995 to 64 in 2020.[15] This trend is expected to continue, with 99 articles in 2023, 50 percent more than the end total of 2020. Notable influential work in the early 2000s used neural networks to forecast the prices of derivatives (Hutchinson 1994, Yao 2000), which prompted later researchers to further examine their feasibility in contemporary financial markets.[17][34]
The Fama-French three-factor (FF3) and five-factor (FF5) models are cornerstones of modern finance and thus are good candidates for these types of improvements, given that they are still imperfect models that have received substantial criticism. FF3 was developed in the 1990s by Eugene Fama and Kenneth French, in their work to build upon the Capital Assets Pricing Model (CAPM), through the introduction of additional factors to capture the effects of the size of the firms and book-to-market ratio. FF3 considers the excess risk (captured by the excess market return), size of company (measured by the excess in returns from small-cap minus large-cap stocks), and value (reflected in the spread from high book-to-market ratio minus low book-to-market ratio

of stocks).[9] FF5 also accounts for profitability (measured by the difference between high- and low-operating profitability stocks) and investment (capturing the spread between stocks with high and low levels of investment). [10]
In this article, I will explain examples of machine learning methods being applied to evaluate the efficiency of FF3 and FF5, and the effectiveness of those attempts, by reviewing the literature published regarding the comparison between these models. I will also explore attempts to combine machine learning to FF3 and FF5, to further improve the accuracy of their predictions and add additional factors, given that machine learning models are excellent at identifying correlations between variables.

11.2 Capital Asset Pricing Model (CAPM)

CAPM was built upon the work of modern portfolio theory independently by several economists, namely William Sharpe[29], John Linter[20] and Jan Mossin[21]. It has had far reaching consequences in how portfolios were constructed for pension and insurance funds that serve millions of people, and popularised the concept of stock market 'efficiency' (the idea that the stock market is what offers a correct price for a share). It is a model for pricing an individual security or portfolio, and starts by making several assumptions of all investors and then builds upon these assumptions to create relationships between several key variables. [14]
Namely, that all Investors:[14]

1. Aim to maximize economic utilities (Asset quantities are given and fixed). This is to say that they will choose portfolios that are mean-variance efficient: That it provides the highest level of returns for a given level of risk.
2. Are rational and risk-averse: Which is to say that they prefer higher returns for a given level of risk and will make decisions to maximize their expected benefit
3. Are broadly diversified across a range of investments.
4. Are price takers, i.e., they cannot influence prices: Institutional investors often have the ability to influence the prices through their market power.
5. Can lend and borrow unlimited amounts under the risk free rate of interest.
6. Trade without transaction or taxation costs.
7. Deal with securities that are all highly divisible into small parcels, meaning all assets are perfectly divisible and liquid.
8. Have homogeneous expectations. Which is to say that they have the same beliefs regarding returns, variances and covariances of assets (how the returns of two different asset move in relations to each other), and will lead to them all identifying the same optimal portfolio of risky assets.
9. Have all information available all at the same time

The equation that relates the key variables take the form:

$$E(R_i) - R_f = \beta_i(E(R_m) - R_f) \tag{11.1}$$

$E(R_i)$ Expected return on asset i.

R_f Risk-free rate of return.

$E(R_m)$ Expected return of the market portfolio.

β_i Sensitivity of the asset's excess return to the market excess return

A central tenet of the CAPM is that systemic risk, captured by the β , is the only factor that affects the level of return on a share for a completely diversified investor. This risk is thus directly correlated to the amount that the share will move when the stock market moves as a whole.[14]

11.3 Weakness of CAPM

Academics had started to criticise the model in the 1970s, with one particular line of criticism from Richard Roll, directed at the heart of the model, was that the model is essentially untestable. This was due to the fact the market portfolio is unobservable[25], because it would need to include every possible available asset, including items such as jewellery and stamp collections. Additionally, any mean-variance efficient (offers a highest possible return for a given level of risk) portfolio satisfies the CAPM equation exactly. [25] Testing the CAPM is is equivalent to testing the mean-variance efficiency of the Portfolio, so it provides little unique insights to the investor, which couldn't be gain more easily.

Furthermore, of the many assumption listed above, whilst we acknowledge that all of these assumption are on some level unrealistic, some are much more outlandish than others. This could explain the significant amount of anomalies that do not fit the CAPM, as the cases were the assumptions don't hold would be the biggest sources of error. It goes without saying that the idea of all investor having all information available at all, are rational and risk averse and are broadly diversified, are not going to hold true. The main problem is that every investor is assumed to be rational and risk averse and broadly diversified. It may be the case a single investor could be described as such, but this assumption fails because it is applied to all participants. Additionally, even the idea that all investors are price taker doesn't always hold true, as institutional investor often have the ability to exert significant influence on the market.

Eugene Fama and Kenneth French focussed their criticism of the CAPM around two phenomenons they noticed throughout the data they presented, which covered the range 1963-1991. The first criticism, dubbed the size premium, was that small stocks as measured by market capitalization outperformed what CAPM would have expected.[8] There are several hypotheses that can explain this phenomenon, such as that smaller companies have reduced access to capital, less customers and lower liquidity, and are thus exposed to greater risk.

The second criticism they levied, dubbed the value premium, was the out-performance of high book/market versus low book/market companies. A high book/market ratio stock is one where their market value is lower than their book value (net asset value). This would usually suggest that the stock was undervalued by the market or the company is in financial turmoil. One reason for this performance are that these stocks are considered more risky, and thus increased returns are the compensation for this increased risk. [8] The alternative behavioural reason is that this is an anomaly due to market inefficiency and investor biases. Investors overreact to recent news, and thus underprice these High book/market stocks. The subsequent corrections of these mispricings leads to the higher returns. [36]

Nevertheless, the CAPM has been incredibly influential in modern finance, and many attempts have been made to find a more grounded and accurate model to accurately price securities and commodities. Fama and French were two such economists, who would collaborate to create their own model to address the criticisms they identified.

11.4 Fama-French Three Factor Model(FF3)

In 1992, Economists Eugene Fama and Kenneth French published "The Cross Section of Expected Stock Returns", which proposed that firm size and book-to-market value as better explanatory variables than beta.[11] Then in 1993, they proposed their three-factor model (given below), which introduced the SMB_t and HML factors, which accounts for the over performance of

small market capitalisation companies and high book to market ratio companies than what would have been prediction via CAPM.[8] It retains the Market Risk Premium $(R_m - R_f)$ from CAPM, but eschews all the other terms, and adds the previously mentioned factors:

$$r = R_f + \beta(R_m - R_f) + b_s \cdot SMB + b_v \cdot HML + \alpha \tag{11.2}$$

r	Expected return of the portfolio.
R_f	Risk-free rate of return.
R_m	Return on the market portfolio.
β	Market sensitivity coefficient, analogous to CAPM beta.
SMB	Small Minus Big factor, capturing the size premium.
HML	High Minus Low factor, capturing the value premium.
α	Abnormal return not explained by the included risk factors.

$$SMB_t = \frac{(S/L + S/H)}{2} - \frac{(B/L + B/H)}{2},$$
$$HML_t = \frac{(H/S + H/B)}{2} - \frac{(L/S + L/B)}{2}.$$

11.5 Shortcomings of FF3

Though the 3 factor model is an improvement over the CAPM, and has been able out-perform attempts to improve the model (as will be explored later in this article), there have still been critiques levied against FF3. Even though SMB and HML do appear to have some correlation to expected returns, they do not appear to have any substantive economic theory that explains why they should be risk factors. There are many factors in life that are correlated with each other, but that doesn't imply that one causes the other - or that another factor is what actually affects SMB and/or HML, and also more correlated to expected returns. As such, the model has been accused of "data mining," meaning the size (SMB) and value (HML) factors were chosen because they empirically fit historical data, rather than being derived from a robust underlying economic theory of risk, which could explain how these are related. Critics argue that by using empirically-derived factors (SMB and HML), the model is shifting away from a risk-based approach to a more ambiguous, "quality-specific" method. There is this no guarantee that these factors will continue to hold, an there has been some analysis that shows that expected results diverge from actual results.[2] Additional criticisms were taken onboard by Fama and French, namely that there were two additional factors that were correlated to expected returns, which were incorporated into their adjusted 5 Factor model. These two additional factors were that higher returns of stocks with high operating profitability and the higher than expected results of firms that adopt conservative investment strategies. [10]

11.6 Fama-French Five Factor Model(FF5)

In 2015, Fama and French updated their model, adding two additional factors to account for the anomalies from the original FF3. These included Robust-Minus-Weak (RMW), to capture the higher returns from companies with greater profitability and Conservative-Minus-Aggressive (CMA), to account for companies which invested conservatively tending to offer higher returns, despite their low asset growth. [10] Their justification was that these two additional factors produce a resulting model, which provides a more accurate framework, and better captures the average returns. However, Fama and French themselves have also shown that HML is redundant

in US markets, as it is completely explained by the other four factors.[12] And in the Eastern European, Latin American and Asian Markets have shown that the HML is the only one of the five factors that is not redundant when the FF5 is applied.[18] This indicates that whilst not every term is totally independent in all regions, there exists some use case for their inclusion in the model for broader application.

$$R_{it} - R_{Ft} = \alpha_i + \beta_1(R_{Mt} - R_{Ft}) + \beta_2 SMB_t + \beta_3 HML_t + \beta_4 RMW_t + \beta_5 CMA_t + \varepsilon_{it} \quad (11.3)$$

R_{it} Return on asset or portfolio i at time t.

R_{Ft} Risk-free return at time t.

R_{Mt} Value-weighted market portfolio return.

SMB_t Size factor (Small Minus Big).

HML_t Value factor (High Minus Low).

RMW_t Profitability factor (Robust Minus Weak).

CMA_t Investment factor (Conservative Minus Aggressive).

$\beta_{1,\ldots,5}$ Factor loadings.

α_i Abnormal return unexplained by the factors.

ε_{it} Zero-mean error term. [12]

11.7 Shortcomings of FF5

Despite the addition of extra components, FF5 doesn't always outperform FF3 in all situations. In fact, several studies show FF3 performs as well as, or is a better performing model in Asian markets.[22][18] HML is the only factor that is not redundant in Asia, Eastern Europe or Latin America, RMW is not significant in Asia, and CMA and SMB are redundant in Asian, Eastern Europe and Latin American markets.[18] The value factor is redundant for US equity returns, but has substantial explanatory ability in emerging markets, and CMA is redundant in Chinese markets.[24] Given that portfolios are optimised in specific regional markets, it means that we have a useless coefficient much of the time. We must also consider that if we were to assume that all markets are bound by similar rules, then the existence of factors not applying to all regions places some doubt in the logical basis of the model, as we cannot be sure that these are fundamental factors, which will always be true for any market. One has to consider that these factors may only have been usefully explanatory for a certain time or developmental period, given that the nature, organisation and mindset of firms has changed significantly in years since the 1980s. All of this paints the picture of a model that has some useful explanatory ability, albeit with the caveat that it is still a imperfect model that is not drawing its components from the fundamental forces which govern returns from the market. It makes sense for us to then pursue alternative methods to analyse the returns of a stock, if for no other reason than to compare whether we should go for a new approach.

11.8 Machine learning tools used in analysis

Before going into the descriptions of the neural networks used for analysis, it is import to define certain definitions and functions. Below, equation (4) illustrates a mathematical description of a simple node of a network, to aid explanation.

$$y = \sigma(w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b) \quad (11.4)$$

Inputs: $x = (x_1, x_2, \ldots, x_n)$ denotes the input vector to a node.

Weights: $w = (w_1, w_2, \ldots, w_n)$ determines the strength of each input signal. Analogous to the slope a in the linear relation $z = ax + b$, where the weighted sum forms the argument of the activation function.

Bias: b provides a constant offset, shifting the activation function output, analogous to the intercept in a linear model.

Activation function: $\sigma(\cdot)$ introduces non-linearity; a commonly used example is the sigmoid function, satisfying $\sigma(x) = 1 - \sigma(-x)$.

Hyperbolic tangent: $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$, an activation function bounded in $(-1, 1)$.

Hadamard product: The element-wise matrix product defined by $(A \circ B)_{i,j} = A_{i,j}B_{i,j}$.

Node (Neuron): A computational unit within a neural network layer. It computes a weighted sum of inputs, applies an activation function, and passes the result to subsequent layers [5].

Edges: Connections between nodes that transmit information through the network.

Concatenation: The operation of combining vectors without loss of elements, e.g. $[1, 2]$ and $[3, 4, 5]$ concatenate to $[1, 2, 3, 4, 5]$.

Input gate: Controls the flow of current input and previous hidden state into the cell state.

Cell state: Represents long-term memory in an LSTM, aggregating information across time steps.

Hidden state: Short-term memory updated at each time step via a non-linear transformation influenced by the cell state.

Output gate: Regulates how much cell-state information is passed to the next hidden state.

Reset gate: Controls how much of the previous hidden state is discarded at the current step.

11.8.1 Artificial Neural Network (ANN)

Neural Networks are a powerful class of computational models with a wide range of applications, ranging from data analysis in science and healthcare to voice and speech transcription/translation. Their Core design is based around a simplified model for how neurons are structured within the human brain, where nodes (neurons) are connect by 'edges' and are organised into many hidden layers along with an input layer and an output layer on either side of the hidden layers. The purpose of the nodes is to transform the data passed into it, by applying weight and biases, passing it through an activation function, and then passing on the output through the edges, further along the network. The nodes can take many different forms, ranging from simple linear classifiers, to complicated, multi-gated 'cells', which take much more complicated structure.

A mathematical description of the hidden layer of an artificial neural network:

$$Y_j = \sigma(W_{i,j} \times X_i - b_j)$$

Y_j Input information of the j^{th} input of neuron Y

$W_{i,j}$ The connection weight between the i^{th} input and the j^{th} neuron, which is an element of the matrix W

X_i The i^{th} input variable where X represents the input vector

b_j The j^{th} element of the bias/threshold variable

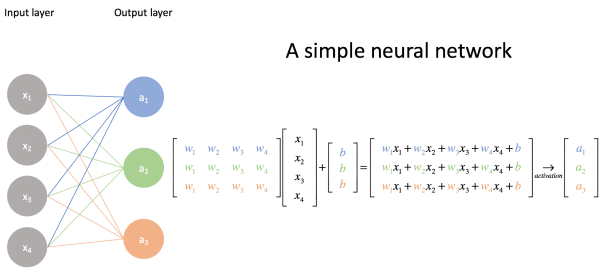


Figure 11.1: The matrix representation of the most simple and how results are calculated through matrices. Scaling up the layers by adding more layers involves repeating this process multiple times

In it's simplest form, an Artificial Neural network (ANN) is made up of nodes of linear classifiers, called perceptrons, that takes d inputs $(x_1, ..., x_d)$ and outputs 0 or 1. We need a function to represent these nodes, and a common candidate is the sigmoid function for its simplicity and resistance to large fluctuations in values. After computing the results for the nodes for a hidden layer, the results are passed on to next layer and the process is repeated until we receive an output through the output layer. Each of the nodes contains weights and biases, which can be seen as the vector (or row of a matrix), which we combine with the input vector to obtain the resulting scalar product, and that result is modified with a bias and passed through an activation function, to give an output which will make a single value of the input vector for successive nodes. The arrangement and values of these weights can be seen as way to store information which is then accessed through the input of a specific vector, with the information obtained through the output vector. Thus, the precise calibration of these weights is necessary in order for the information to be stored, and this is achieved through the process of 'supervised learning'. The weights are initialised at random, then through the process of backpropogation, the model is given training data labelled with appropriate answers. In an implementation of backpropogation, a 'forward pass' occurs, where data is input into the model to give and output. Then, through the use of a 'cost function', the output value is used to calculate the 'loss' or deviation from the expected answer. The output layer is then sent in the reverse, from the output layer to the input layer, and the chain rule of calculus is used to compute the derivative of the loss function, with respect to each weight. This determines each component's contribution to the deviation from the expected value, and through the use of an optimiser, such as Stochastic Gradient Descent or Adaptive Moment Estimation (sometimes abbreviated as Adam), the components are update to a new value that corrects for the error in expected value. This process is repeated for each element of the training data to obtain a model with the finalised weights and biases. The final weights and biases are then stored in arrays/lists, and it is through these combinations of weights values and positions, where this information is stored.
Whilst the above neural networks are quite effective for analysing relationships within data, there have been many improvements on the design of neural networks, mainly centred around improving the ability of the node itself to improve information retention.

11.8.2 Recurrent Neural Network (RNN

Recurrent Neural Networks (RNN) are a class of neural network for specialised for processing sequential data, because they are networks with loops in them, which act as an internal memory. Though this seems like a significant divergence from the class we described earlier, we can unroll the loop to see that it is a repeating chain of the same unit block, with input data being processed from one side to the next and giving an output to be passed on through the other. Because of this chain like nature, they have been found to excel in speech recognition, language modelling

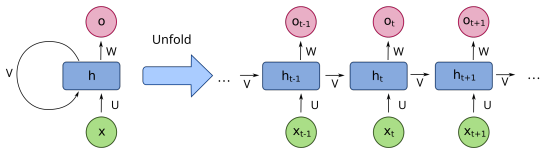


Figure 11.2: RNN compressed on the left, and unfolded on the right to better show the structure. x_t - input vector, h_t - hidden vector, y_t - output vector, θ - neural network parameter. [13]

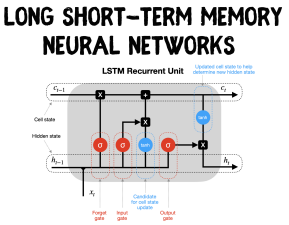


Figure 11.3: Overview of the Long Short Term Memory (LSTM) recurrent unit.[6]

and translation, where in order to give a correct answer, it is necessary to consider the context (which is the order, or the placement) of each word in the sentence or paragraph as a whole. This property is called 'long-term dependency', and it is a quality that we desire for the particular types of analysis we are conducting (which is to say capturing trends over a long time series in a financial market). Much like a traditional neural network, it requires training on data to store information and calibrate the weights and biases. The diagram (Figure 2.) shows the loop unravelled, to demonstrate the similarity between the conventional neural network.
Though they are extremely capable, RNNs suffer from the 'vanishing gradient' problem, as they squash a large input space into [0,1], and for the sigmoid function, a large change in input results in a competitively small change in output. Over time, the looping nature of the network nodes feeding into one another, causes the derivative to become small and in turn results in the gradient tending to 0. Since the gradient is the major factor in the updating, this prevents the weight from updating. Additionally, they can also suffer from 'exploding' gradients during backpropogation. This occurs due to the repeating multiplication of weights >1, which leads to exponential growth across the layers. These two classes of problems are subsets of what is referred to as the 'Vanishing Exploding Gradient' (VEG) problem. For both of these situations, increasing the amount of layers worsens the issue, as multiplying a group of numbers <1 results in a very small number that is $\ll 1$, and multiplying a group of larger numbers >1 will result in a larger number $\gg 1$. To manage these scenarios, the number of layers must be moderated, along with making sure the starting weights and biases are not too big or small. However, the more interesting solution is to use 'gates' that can restrict the gradient from increasing or decreasing too quickly, as we shall soon see. In theory, RNNs should be capable of developing long-term dependencies, but as they lack an explicit notion of memory, such as the gates in LSTM and GRU, they are unable to maintain strong long-term dependencies in practice. The main reason for this is aforementioned VEG problem, as the very large or small gradients prevent the weights from being updated, which limits the ability to take in new information. Whilst it is possible for RNNs to capture long-term dependencies, it requires significant modifications to their structure through the use of gates.

11.8.3 Long Short Term Memory (LSTM)

The LSTM neural network was introduced by Sepp Hochreiter and Jurgen Schmidhuber, with it's principles introduced in Hochreiter's 1991 German Diploma thesis. Through the use of gating mechanisms, the LSTM is able to avoid the issue of vanishing or exploding gradients that were exhibited by RNN. LSTM were designed to avoid the problems with capturing long-term dependencies, so remembering information for long periods of time is their default behaviour.
The LSTM has 3 Gates; the input gate, the forget gate and output gates, which controls information influx, memory retention and output determination respectively. Because of the gates structure and the separate cell state, an additive path exist for the gradient, instead of a multiplicative path. Addition is much less prone to exponential increases, and given that sigmoids output values in the range [0,1], this stops exponential increases/decreases in the gradient, thus preventing the VEG problem.
These networks are best used for capturing long term dependencies, due to the nature of their more sophisticated memory, and have many use cases in language modelling, speech recognition and anomaly detection. Below are equations that describe the main features of the LSTM.

Forget gate: $f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$, which determines how much of the previous cell state C_{t-1} is retained.

Input gate: $i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$, which controls how much new information is written to the cell state.

Cell input activation: $\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C)$, which represents the candidate values for updating the cell state.

Cell state: $C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t$, which combines retained past information with newly generated content.

Output gate: $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$, which regulates the information passed to the hidden state.

Hidden state: $h_t = o_t \circ \tanh(C_t)$,which represents the output of the LSTM cell at time t.
The first step is to decide what information we are retaining in our cell and what we are throwing away. Starting from the bottom left corner of Figure 3, the forget gate receives the input vector (x_t) and the previous hidden state (h_{t-1}), and the value computed from the concatenation of x_t and h_{t-1}, which is then combined with the weight matrix (W_f), to calculate the scalar product. The scalar product is modified with the bias and passed through the sigmoid function (σ), to give the final output of the forget gate. These values are passed on to C_{t-1}, which decides how much to keep of x_t and h_{t-1} depending on the value of the number passed on. A value of 0 mean that nothing should be remembered about this data and it should forgotten, whilst a 1 means that it must be completely remembered. Consequentially, if the value is in-between 0 and 1, then size of the value is proportion should be remembered or forgotten.
The cell state must then decide what of the new information will be stored in the cell, which has two steps. First, the input gate (i_t) gives an output in a similar manner to to forget gate (albeit with different weights and biases), and passes the result onto the candidate for the cell state ($\tilde{C}_t$). Then $\tilde{C}_t$ uses the *tanh* function to give it's output for the calculation of the new cell state to commence.
We must now update the previous cell state, (C_{t-1}), to the new cell state (C_t), using the results passed on from the previous steps. We preform the Hadamard product between C_{t-1}, with the output from the forget gate, f_t, which is equivalent to forgetting what the forget gate chose to forget from all of the inputs. Then we add $i_t \circ \tilde{C}_t$, which the Hadamard product on the candidate cell state and the input gate result, which is equivalent to keeping what the input gate decided to keep from the inputs provided.
The final step will be to calculate the output from the cell state, which would be passed on to other cell states. To decide what we will passed on, we use a sigmoid function in a similar manner

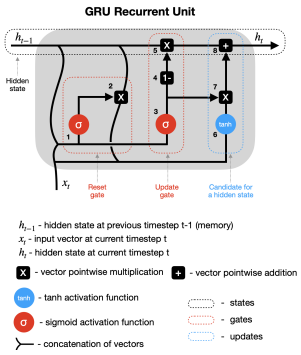


Figure 11.4: Overview of the Gated Recurrent Unit (GRU) [7]

to the previous states (again with different weights and biases), which takes concatenation of x_t and h_{t-1} as an argument, and gives and output (o_t). Then we calculate the value of C_t passed through *tanh*, which gives the value for the new hidden state (h_t). These to values are then multiplied element-wise via the Hadamard product, to give the output, h_t that is passed on to the next cell. This value plays the same role as h_{t-1} in the current cell, but for the next cell.[23] LSTM neural networks have shown significant performance improvements over the RNNs that they are based on, and have spawned further improved designs, though not without a cost.[35] There are multiple variations of LSTM available, but the all share the similar issue of requiring large amount of memory for computation. Researcher have examined alternative models such as Gated Recurrent Units (GRUs) that can maintain some of the benefit of the LSTM, whilst lowering the memory cost.

11.8.4 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network that handles sequential data by using gates (reset and update) to control information flow. It is similar to Long Short Term Memory (LSTM) but has fewer parameters, which results in considerably less memory intensive and being faster for training, at the expense of some accuracy. Developed in 2014 by Kyunghyun Cho et al.,[4] it diverges from the LSTM by reducing the number of gates to 2. These are the update gate and the reset gate, which decide what information will be allowed to output and can be trained to retain information from further back, which allows better predictions in chains of events.
Like the LSTM, it also addresses the vanishing gradient problem in the values used to update weights, but with a lower memory cost of computation. The usage of computer memory is different to the memory of the cell itself, as it pertains to the cost of running/training the model. Below are equations that describe the main components of the GRU, follow by a description of their interaction.

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z), \quad \text{(update gate)} \tag{11.5}$$
$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r), \quad \text{(reset gate)} \tag{11.6}$$
$$\tilde{h}_t = \tanh(W_h[r_t \circ h_{t-1}, x_t] + b_h)\,, \quad \text{(candidate state)} \tag{11.7}$$
$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t, \quad \text{(hidden state)} \tag{11.8}$$

$x_t \in \mathbb{R}^d$ Input vector.

$h_t \in \mathbb{R}^e$ Hidden state vector.

$\tilde{h}_t \in \mathbb{R}^e$ Candidate hidden state.

$z_t, r_t \in (0,1)^e$ Update and reset gate vectors.

W_z, W_r, W_h Weight matrices.

b_z, b_r, b_h Bias vectors.
Where $W_r, W_z, W_h \in \Re^{(e+d)\times d}, W_z \in \Re^{e\times d}, r_t \in \Re^e$ and $b_z, b_r, b_h \in \Re$ are the parameter matrices, vectors and bias values, which are what need to be learned during the training

Information enters the cell as the input vector (x_t), and the hidden state of the previous time-step (h_{t-1}), which are concatenated. This concatenation is then multiplied with weight matrices (W_z and W_r), and the result of these calculations is then used as the argument of the sigmoid to calculate the update gate vector (z_t), and the reset gate vector (r_t) respectively. The candidate hidden state vector ($\tilde{h}_t$) is then calculated in two stages; first through the Hadamard product of h_{t-1} and r_t, which is then concatenated with x_t. Then, the Hadamard product of the resulting vector and the weight matrix (W_h) (along with the modification of the bias (b_h)), is passed through the *tanh* function to give $\tilde{h}_t$. Similarly, the new hidden state is also calculated in two stages, first through the Hadamard product of $\tilde{h}_t$ and z_t. Then this summed with the result of the Hadamard product of $1 - z_t$ and the previous hidden state, h_{t-1}, to obtain the new hidden state, which is output from the GRU to take the role of the previous hidden state in the next iteration.
Compared to LSTMs, GRUs situationally perform better for low complexity and low data application, with the LSTMs performing better than GRUs for high complexity and capturing very long term dependencies. A GRU requires less data to train, as well as less time, so can perform better in situations where training data is limited or the are few insights to obtain with long term dependencies.[26] GRUs are also prone to over-fitting to the training data, which can hamstring their ability to function as expected on unseen data. The total number of parameters in the GRU RNN equals $3 \times (n^2 + nm + n) = 3n(n + m + 1)$ (where $n, m \in \aleph$), [26] whereas the total number of parameters in the LSTM RNN is equal to $4(n^2 + nm + n) = 4n(n + m + 1)$.[27] Therefore, the GRU requires a smaller number of total parameters that must be calculated, which can be beneficial when working with large datasets and larger models. It is worth noting that there is no definitive assertion as to in what circumstances these improved performances would arise,[26] which gives us a strong motivation to analyse the research that compares theses models directly to the previously discussed financial models to gauge their effectiveness against comparable methods.

11.8.5 Methods of Error Mitigation in Neural Networks

The training process of Neural Networks is crucial to the success of implementing the networks in a practical setting, as any errors in this stage will carry over during the inference process. Spurned by this challenge, researchers have developed many methods/techniques we can use to mitigate errors in the training process, since a good training setup is crucial to optimal results. There are two main classes of problems that cause errors during the training process; the brittle co-adaptations of nodes, and the numerical instability of the weights. During training, the backpropogation learning process creates brittle co-adaptations between the nodes of the network, which are optimised for the training data. These co-adaptations are not as effective when applied to the unseen data, which limits the effectiveness of the model during inference. The VEG problem is an example of the numerical instability of weights, which occurs when the gradients of the loss function become too large too quickly, in turn caused by the weights of the network increasing too rapidly. The rapidly increasing weights cause the model to produce NaN (Not a Number) or INF (infinity) value weights and overflow errors, preventing calculations from being possible or give sensible answers, and thus limiting the model's ability to learn during the training process.
Gradient Clipping is a common technique to mitigate the VEG problem, as it directly modifies the gradients to prevent them for becoming problematic values. The 'Clipping' starts with defining a

threshold, beyond which gradients are scaled down to ensure that their norm (magnitude) doesn't exceed the threshold. These clipped gradients are then used to update the model parameters, which limits the weights from becoming excessively high NaN, and ensures that the training process proceeds smoothly. Gradient clipping has been noted as for its ability to accelerate the training process and converges arbitrarily faster than gradient descent with fixed a step-size.[37]
The purpose of the drop out layer (sometimes referred to as dilution) is to guard against over-fitting, by randomly temporarily omitting the output of (for example) 20 percent of the cells, by setting their value to zero. These dropped out cells don't contribute to the forward pass, and don't participate in the back propagation process. This means that each time that an input is given to the neural network, it is sampling from the pool of LSTM cell a different combination of neurons, but all of these combinations share the same pool of possible cells. The goal of this technique is to reduce the complex co-adaptation of neurons, because the neurons can no longer rely on the other neurons to provide the correct answers. When the full set of neurons is used after training, the total output of the network is multiplied by 0.8 (from the previous example) to correct for additional cells by providing the geometric mean of the 128 combined cells. Another advantage of the dropout layer is that they are highly computationally efficient, as they randomly set a proportion of neurons to zero, thus they reduce the number of calculations required in the process. Because of the low (computational) cost of implementation, combined with the significant improvement in error reduction of 50 percent[31], dropout layers have seen wide-scale usage in modern machine learning training processes.

11.9 Attempts at evaluating Financial Models using machine Learning tools

We have discussed the motivation, development and criticism of FF3 and FF5; and the mechanics, theory and error mitigation of neural networks such as RNN, GRU and LSTM. In order for us to make direct comparisons, that produce produce a good comparison which applies broadly, we must consider a wide range of markets. In this section, we will discuss implementations in Asian, Eastern European, Latin American and US markets, so that we can draw good conclusions. Kuang-Hsun Shih's (2024) paper[30] detailed their teams' attempts to analyse CAPM, FF3 and FF5 to neural networks such as LSTM, GRU and ANN, as well as convolutional neural networks (CNN), with data on the time period 2010-2019. Their focus was on the key variables, such as market risk premium; book-to-market ratio; size factor; short-term and long-term reversal factors; profitability; investment; and momentum factors, which were used to compare the predictions given by the models and neural networks. The study also examined combining the CAPM, FF3 and FF5 with ANN, LSTM, GRU and CNN to improve their forecasting accuracy. In their analysis, they found that the Fama-French three-factor model, when paired with LSTM achieves the minimum error consistently, thus having the best forecasting accuracy. The factors that were the best predictor of returns were the market risk premium ($R_f - R_i$), the book-to-market ratio (HML) and the size premium (SMB). The remaining factors did not provide any improvements when they were included in the model. They stated that, though they had the option of 1,2 or 3 hidden layers, the LSTM with a single hidden layer and GRUs with double hidden layers were the most effective, and that the results of LSTM, GRU, and GRU+LSTM with three hidden layers are ineffective. Thus they only used double and single layers neural networks in their experiments, as those would be the most promising candidates.
The study primarily relied on the Mean Absolute Error (MAE) as the error metric, which provides a straightforward interpretation, though it might not capture all dimensions of the model's prediction performance. Additionally, when models were mixed and stacked (e.g., stacked LSTM, stacked GRU, CNN-LSTM, and CNN-GRU), the results did not seem to reduce errors. This is, however, contrary to findings in previous studies on this matter, which makes it difficult to draw conclusions with respect to mixing and stacking models.[3] [16]

RNN and LSTM has also been investigated in use for trading strategies. Most financial data is time-serially correlated, and as a result, Long Short-Term Memory (LSTM) and Recurrent Neural Network perform better than traditional trading algorithms, with gains of around 175 percent over the a period of 2 years.[28]
GRUs have been shown to have 25 percent less parameters than the LSTM per cell,[26][27] which allows them to be more resistant to noise and be more computationally efficient.[1] Additional analysis on stock price predictions shows that GRUs achieve a reduction in training speeds by 30-40 percent and lowered inference latency by 25 percent compared to LSTM under conventional conditions.[32] However, the LSTM still performs better than GRU in terms of less deviation error (0.24, vs 0.38 for GRU) for the complex critical events, which indicates better performance during the more unpredictable periods in the markets. The Mean Absolute Error (MAE) of the GRU was slightly higher than that of the LSTM, by 3 percent, which indicates the consistently superior, if small, performance of the LSTM in this regard.[32] This suggests that there are some very long term dependencies that are captured by the LSTM, but are not able to be captured by the GRU. Training GRUs requires fewer iterations to reach convergence, with and reduction in time of 19 percent.[32] The faster training times for the GRU, despite the larger deviation error, indicate that GRU have good use case in real time trading environments, where a good estimate that is achieved quickly is desired.
There has also been further attempts to combine the factor models with machine learning, with some attempting to add additional factors to FF3 and FF5. Yao et als.'s 2022 paper on implementing a six-factor model, based on LSTM and FF5, with the model being augmented with a short-term momentum (sMOM) factor, which is simulated by the LSTM. The sMOM factor refers to the tendency of recently well performing stock to remain well performing and the poorly performing stocks to remain poorly performing. Their work focussed around the Chinese stock markets, in Shanghai and Shenzhen stock exchanges, training from the period January 1st, 2008 to December 31st, 2019, and to avoid look-ahead bias, the portfolios were simulated using the data from January 1st 2020, to May 31st, 2021. Their model used a LSTM layer with 128 neuron (LSTM cells), with a dropout layer, as dropout layers can effectively reduce over-fitting.[19] They had 3 experiments, Experiment A examines the explanatory power of the model by classifying the current price movements; Experiment B also examines the explanatory power but considers the time effect; Experiment C examines the prediction power of the model on next-day returns. Their results from these experiments were then used to construct a portfolio, and that portfolio was was then predicted by the model, which would be using the out of sample data. In their analysis, the found that their six-factor best explains current price movements and best predicts future price movements of stocks, when compared to other candidate models. Their work, and they themselves, suggest that there is scope for neural networks being combined with traditional economic models to enhance accuracy and give better predictions. The six-factor model outperforms the other candidate models in terms of explain current price movements and predict future price movements of stocks. Overall, by employing the LSTM algorithm, we successfully forecast the asset price using a limited number of high-quality pricing factors. However, 6th factor doesn't address the FF5's fundamental flaws of not being grounded in risk analysis, because of the way it is obtained, and the 'black-box' nature of the neural network.[33]

11.10 Conclusion

In this article, we have seen a summary of the development of the prominent Economic models for predicting the returns of securities on the stock exchange, and the criticism that have been levied against them. We have also explored the evolution of neural networks, from the basic ANN, to CNN and advanced cells such as GRU and LSTM. We have discussed how studies compared the economic models with the neural networks, as well as comparisons of combining them. We also briefly discussed the other uses of the neural networks in other financial matters,

such as stock trading, to consider if similar situations result in different results. We cannot say whether one type of neural network will perform conclusively better, as the evaluation above shows many situations where one performs better than the other and vice-versa. However, we saw a trend of LSTMs and GRUs being the best performers, which could in part be due to their superior memory structures. No doubt there will be future models that can improve upon these, or find a particular niche where they excel. However, a good understanding of the theory behind these models will be key to an effective implementation, and manage the difficulties of the training process. The research we have discussed points toward machine learning techniques and application of neural networks to the FF3 and the FF5 models as beneficial to improving the accuracy of their predictions. With the popularity of applying machine models increasing, not just in finance but in all fields, a good understanding in mathematical background of neural networks will be a useful tool to both future researchers and those work in finance.

Bibliography

[1] Abdulaziz A. Alsulami, Qasem Abu Al-Haija, Badraddin Alturki, Ali Alqahtani, Faisal Binzagr, Bandar Alghamdi, and Rayan A. Alsemmeari. Exploring the efficacy of GRU model in classifying the signal to noise ratio of microgrid model. *Sci. Rep.*, 14(15591):15591, July 2024. ISSN 2045-2322. doi: 10.1038/s41598-024-66387-1.

[2] Jan Bartholdy and Paula Peare. Estimation of expected return: Capm vs. fama and french. *International Review of Financial Analysis*, 14(4):407–427, 2005.

[3] Shun Chen and Lei Ge. Exploring the attention mechanism in LSTM-based Hong Kong stock price movement prediction. *Quantitative Finance*, September 2019. URL https://www.tandfonline.com/doi/full/10.1080/14697688.2019.1622287?scroll=top&needAccess=true.

[4] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL http://arxiv.org/abs/1406.1078.

[5] Google Developers. URL https://developers.google.com/machine-learning/glossary.

[6] Saul Dobilas. Lstm recurrent neural networks. *Towards Data Science*, January 2022. URL https://towardsdatascience.com/wp-content/uploads/2022/02/17cMfenu76BZCzdKWCfBABA.png.

[7] Saul Dobilas. Gru recurrent neural networks – a smart way to predict sequences in python, March 2022. URL https://towardsdatascience.com/wp-content/uploads/2022/02/13a8HnDUlzhhKcSpQzOyiCQ.png.

[8] Kenneth French Eugene Fama. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 1993. URL https://www.sciencedirect.com/science/article/pii/0304405X93900235.

[9] Kenneth French Eugene Fama. Multifactor explanations of asset pricing anomalies. *The Journal of Finance*, 1996. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1540-6261.1996.tb05202.x.

[10] Kenneth French Eugene Fama. A five-factor asset pricing model. *Journal of Financial Economics*, 2015. URL https://www.sciencedirect.com/science/article/pii/S0304405X14002323?via%3Dihub#s0010.

[11] Eugene F Fama and Kenneth R French. The cross-section of expected stock returns. *the Journal of Finance*, 47(2):427–465, 1992.

[12] Eugene F. Fama and Kenneth R. French. A five-factor asset pricing model. *Journal of Financial Economics*, 116(1):1–22, 2015. ISSN 0304-405X. doi: https://doi.org/10.1016/j.jfineco.2014.10.010. URL https://www.sciencedirect.com/science/article/pii/S0304405X14002323.

[13] fdeloche Own work. File:Recurrent neural network unfold.svg - Wikimedia Commons, December 2017. URL https://commons.wikimedia.org/w/index.php?curid=60109157#/media/File:Recurrent_neural_network_unfold.svg. [Online; accessed 4. Jan. 2026].

[14] Deborah Lewis Glen Arnold. Corporate financial management 3th edition.

[15] John W. Goodell, Satish Kumar, Weng Marc Lim, and Debidutta Pattnaik. Artificial intelligence and machine learning in finance: Identifying foundations, themes, and research clusters from bibliometric analysis. *Journal of Behavioral and Experimental Finance*, 32: 100577, 2021. ISSN 2214-6350. doi: https://doi.org/10.1016/j.jbef.2021.100577. URL https://www.sciencedirect.com/science/article/pii/S2214635021001210.

[16] Mohammad Hossain, Rezaul Karim, Ruppa Thulasiram, Neil Bruce, and Yang Wang. Hybrid deep learning model for stock price prediction. pages 1837–1844, 11 2018. doi: 10.1109/SSCI.2018.8628641.

[17] James M. Hutchinson, Andrew W. Lo, and Tomaso Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, 49(3):851–889, 1994. ISSN 00221082, 15406261. URL http://www.jstor.org/stable/2329209.

[18] Foye James. A comprehensive test of the fama-french five-factor model in emerging markets. *Emerging Markets Review*, 37:199–222, 2018. URL https://www.sciencedirect.com/science/article/pii/S1566014117304089?via%3Dihub.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782. doi: 10.1145/3065386. URL https://doi.org/10.1145/3065386.

[20] John Lintner. The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. *Review of Economics and Statistics*, 1965. URL https://www.jstor.org/stable/1924119.

[21] Jan Mossin. Equilibrium in a capital asset market. *Econometrica*, 1966. URL https://www.jstor.org/stable/1910098.

[22] Nurdina Nurdina, Nurkholis Nurkholis, Noval Adib, and Sari Atmini. Evaluation of the resilience of real estate and property stocks to inflation and interest rate uncertainty: Implementation of two asset pricing models. *JRFM*, 17(12):1–18, November 2024. doi: None. URL https://ideas.repec.org/a/gam/jjrfmx/v17y2024i12p530-d1527108.html.

[23] Christopher Olah.

[24] Lin Qi. Noisy prices and the fama–french five-factor asset pricing model in china. *Emerging Markets Review*, 31:141–163, 2017.

[25] Richard Roll. A critique of the asset pricing theory's tests part i: On past and potential testability of the theory. *Journal of Financial Economics*, 1977. URL https://www.sciencedirect.com/science/article/pii/0304405X77900095.

[26] Fathi M. Salem. *Gated RNN: The Gated Recurrent Unit (GRU) RNN*, pages 85–100. Springer International Publishing, Cham, 2022. ISBN 978-3-030-89929-5. doi: 10.1007/978-3-030-89929-5_5. URL https://doi.org/10.1007/978-3-030-89929-5_5.

[27] Fathi M. Salem. *Gated RNN: The Long Short-Term Memory (LSTM) RNN*, pages 71–82. Springer International Publishing, Cham, 2022. ISBN 978-3-030-89929-5. doi: 10.1007/978-3-030-89929-5_4. URL https://doi.org/10.1007/978-3-030-89929-5_4.

[28] Chenjie Sang and Massimo Di Pierro. Improving trading technical analysis with tensorflow long short-term memory (lstm) neural network. *The Journal of Finance and Data Science*, 5(1):1–11, 2019. ISSN 2405-9188. doi: https://doi.org/10.1016/j.jfds.2018.10.003. URL https://www.sciencedirect.com/science/article/pii/S2405918818300539.

[29] William Sharpe. Capital asset prices: A theory of market equilibrium. *Journal of Finance*, 1964. URL https://www.jstor.org/stable/2977928.

[30] Kuang-Hsun Shih, Yi-Hsien Wang, I-Chen Kao, and Fu-Ming Lai. Forecasting etf performance: A comparative study of deep learning models and the fama-french three-factor model. *Mathematics*, 12(19), 2024. ISSN 2227-7390. doi: 10.3390/math12193158. URL https://www.mdpi.com/2227-7390/12/19/3158.

[31] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

[32] Bowen Yang. Trade-off analysis of efficiency and accuracy in gru vs lstm. *ITM Web of Conferences*, 80:01017, 12 2025. doi: 10.1051/itmconf/20258001017.

[33] Haixiang Yao, Shenghao Xia, and Hao Liu. Six-factor asset pricing and portfolio investment via deep learning: Evidence from chinese stock market. *Pacific-Basin Finance Journal*, 76:101886, 2022. ISSN 0927-538X. doi: https://doi.org/10.1016/j.pacfin.2022.101886. URL https://www.sciencedirect.com/science/article/pii/S0927538X22001810.

[34] Jingtao Yao, Yili Li, and Chew Lim Tan. Option price forecasting using neural networks. *Omega*, 28(4):455–466, August 2000. doi: None. URL https://ideas.repec.org/a/eee/jomega/v28y2000i4p455-466.html.

[35] Ariana Yunita, MHD Iqbal Pratama, Muhammad Zaki Almuzakki, Hani Ramadhan, Emelia Akashah P Akhir, Andi Besse Firdausiah Mansur, and Ahmad Hoirul Basori. Performance analysis of neural network architectures for time series forecasting: A comparative study of rnn, lstm, gru, and hybrid models. *MethodsX*, 15:103462, 2025.

[36] Liu Yuxiu. Explaining the book-to-market ratio effect from the perspective of behavioral finance. *Advances in Economics, Management and Political Sciences*, 131:20–26, 12 2024. doi: 10.54254/2754-1169/2024.18412.

[37] Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. *arXiv preprint arXiv:1905.11881*, 2019.

Chapter 12

Dynamic Cointegration: Kalman Filter for Pairs Trading

Ray Riyaz

Edited by: Adam Lewandowski

Abstract: This article examines the use of Kalman filter to estimate dynamic hedge ratios in pairs trading and clarifies its relationship to classical cointegration. The Engle–Granger framework is reviewed as a static equilibrium model, and is reformulated in a state-space setting with hedge ratio as a latent-process. The dynamic model improves adaptability under structural change, but it does not guarantee asymptotic spread stationarity. Simulations highlight a trade-off between stability and response in both models. Overall, Kalman-based estimation is best viewed as a control mechanism rather than a replacement for classical cointegration.

12.1 Introduction

Pairs trading is a market-neutral strategy that seeks to profit from temporary mispricing between two related assets. Rather than forecasting price direction, the strategy takes opposing positions (buying one asset and selling another), so that returns depend on the convergence of their relative values rather than on overall market movements [12]. The *difference between these positions*, commonly referred to as the **spread**, is monitored for deviations from its typical behaviour, which are assumed to be short-lived. In practice, however, the relationship between assets is not always stable: structural changes, regime shifts, or evolving risk exposures can cause this spread to drift, reducing the effectiveness of classical pairs trading strategies. This motivates more adaptive approaches that allow trading relationships to evolve over time, such as those based on Kalman filtering (essentially a recursive prediction-update algorithm).[13].

12.2 Preliminaries

For completeness, we briefly recall the following definitions.

Definition 12.1 (Stochastic Process)**.** A stochastic process is a collection of random variables indexed by time, defined on a common probability space.

Definition 12.2 (Price Series)**.** A price series $\{P_t\}_{t\geq 1}$ is a discrete-time stochastic process representing the observed market price of a financial asset at each time t.

Definition 12.3 (Spread)**.** The spread is a linear combination of two asset prices, say P_1, P_2, weighted by a proportion (hedge ratio β) in which two assets are to be held to minimise risk and isolate their relative mispricing. Thus, the Spread $S_t = P_{2,t} - \beta P_{1,t}$.

Definition 12.4 (Weak-sense stationary process)**.** A weak-sense or wide-sense stationary (WSS) process is a stochastic process whose first moment (mean) and autocovariance do not vary with time, and whose second moment (variance) is finite for all times.

Definition 12.5 (Order of Integration)**.** The order of integration of a time series reports the minimum differences required to obtain a WSS series. If L is the lag operator ($LX_t = X_{t-1}$), a time series X_t is integrated of order d if $(1-L)^d X_t$ is a stationary process. This is denoted by $X_t \sim I(d)$.

Remark 12.1. In what follows, we use weak-sense stationarity as a practical proxy for I(0) behaviour in financial time series.

12.3 Static Cointegration

This section is expanded on the paper by Hossein et al. on profitability of various Pairs Trading strategies such as Cointegration [5]

12.3.1 Rules needed for Cointegration

Equilibrium-theories (mean-reversion trading) or Statistical arbitrage require the variables used to be $I(0)$, so that any deviation from the mean will be temporary. But, most real economic variables are non-stationary or $I(1)$. Stock-prices, for example can be modelled as random walks.

$$x_t = x_{t-1} + \epsilon_t$$

Here, $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ and $\epsilon_t \sim I(0)$. Thus, $x_t \sim I(1)$. Given constants a, b, c, the order of integration of a combination of random-variables is *generally* given by the following rule.

$$x_t \sim I(d), y_t \sim I(e) \implies ax_t + by_t + c \sim I(\max\{d, e\})$$

12.3.2 Cointegrated Assets

$$S_t = X_{2,t} - \beta X_{1,t} \tag{12.1}$$

Consider the spread S_t with hedge ratio β, where $X_{i,t} \sim I(1)$ is the price-series of a stock. Based on the rules, any arbitrary spread $S_t \sim I(1)$. But, this can be $I(0)$, under some conditions.

Definition 12.6 (Cointegrated Assets)**.** Two assets are said to be cointegrated, when there exists a linear combination or spread (S_t) that is stationary.

Remark 12.2. There is no guarantee that two assets are always cointegrated.

Intuition

Let two price-series be x_t, y_t. Ler v, ϵ be their non-stationary and stationary components, respectively. The spread is then:

$$\begin{aligned} S_t = x_t - \beta y_t &= (v_{x_t} + \epsilon_{x_t}) - \beta(v_{y_t} + \epsilon_{y_t}) \\ &= (v_{x_t} - \beta v_{y_t}) + (\epsilon_{x_t} - \beta \epsilon_{y_t}) \end{aligned} \tag{12.2}$$

For $S_t \sim I(0)$, we require $v_{x_t} = \beta v_{y_t}$ or no non-stationary component.

Implications of Cointegration

1. From equation 12.2, we can see that cointegrated assets share common non-stationary(I(1)) components [10]. They are affected by same trend, seasonal or stochastic parts.

2. Good candidates for cointegration are usually same-sector stocks: JPM and BAC, KO and PEP, INTC and AMD, etc. Even some currency relations are cointegrated: AUD/USD and NZD/USD.

3. The spread is asymptotically stationary. This is, thus, a long-run relationship between 2 assets that fails when the true equilibrium changes.

Correlation v/s Cointegration

Correlation and cointegration do not imply each other. Correlation measures short-term co-movement of returns ($I(0)$). However, applying correlation to non-stationary price levels ($I(1)$) can produce *misleading spurious correlations* [4]. Cointegration captures long-term equilibrium relationships between price levels. It applies to non-stationary asset prices (I(1)) that share a stable linear combination, even though individual prices may drift over time.

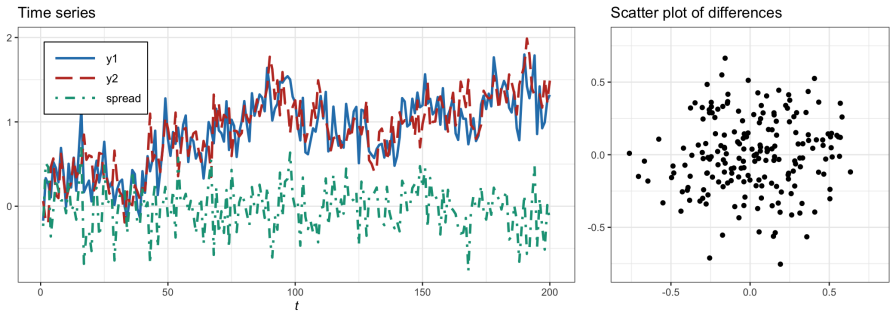


Figure 12.1: No Correlation but Cointegrated Assets [8]

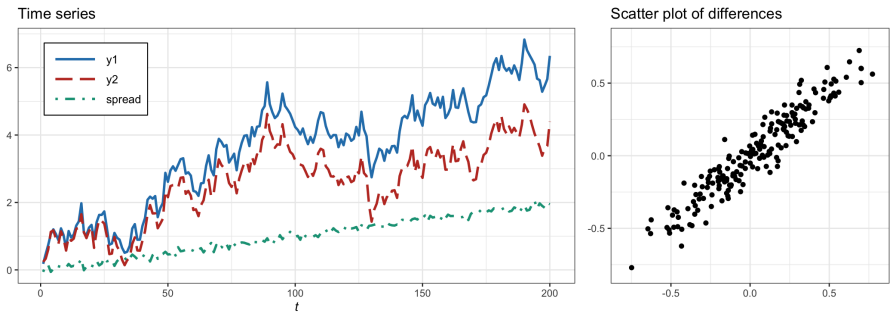


Figure 12.2: High Correlation but not Cointegrated Assets [8]

12.3.3 Engle–Granger Two-Step Procedure

Consider two price series x_t and y_t, such that $x_t, y_t \sim I(1)$. If these series share a long-run equilibrium (or equivalently, cointegration) relationship, deviations from this equilibrium should be temporary. The Engle–Granger two-step procedure provides a simple method to test for such cointegration [3].

Step 1: Estimation of the Long-Run Relationship: The long-run equilibrium relation is estimated via ordinary least squares (OLS):

$$y_t = \delta_0 + \delta_1 x_t + u_t \tag{12.3}$$

where u_t represents the equilibrium error. The estimated residuals ($\hat{u}_t$) measure deviations from the long-run relationship. Cointegration between x_t and y_t holds if the residual process $\hat{u}_t$ is stationary. This is typically assessed using an Augmented Dickey–Fuller (ADF) test applied to $\hat{u}_t$, with critical values adjusted for the cointegration setting MacKinnon critical values [9]. If cointegration is present, the OLS estimator $\hat{\delta}_1$ is *strongly-consistent*, converging to the true parameter faster than in standard stationary regressions.

Step 2: Error Correction Representation: If cointegration is established, the short-run dynamics of the system can be described by an error correction model (ECM):

$$\Delta y_t = \phi_0 + \sum_{j=1}^{p} \phi_j \Delta y_{t-j} + \sum_{h=0}^{q} \theta_h \Delta x_{t-h} + \alpha \hat{u}_{t-1} + \varepsilon_t, \tag{12.4}$$

The coefficient α captures the speed of adjustment toward the long-run equilibrium and is expected to be negative, ensuring that deviations from equilibrium are corrected over time. The ECM thus reconciles short-run fluctuations in y_t and x_t with their long-run cointegrating relationship.

Interpretation: The Engle–Granger procedure establishes an equivalence between cointegration and the existence of a valid error correction representation. In the context of pairs trading, $\hat{u}_t$ corresponds to the spread, whose mean-reverting behaviour underpins equilibrium-based trading strategies.

12.3.4 Trading Strategy

For any trading strategy involving pairs, we need a pair of 2 stocks and the timings at which they will be traded. Firstly Engle-Granger Test may be employed to check for Cointegration across a universe of stocks to find the pairs [2]. Then, the z-score of the spread ($S_t = X_{2,t} - \beta X_{1,t}$) is calculated for a pair found. SD denotes standard deviation and since the mean of the spread $\overline{S}_t = 0$,

$$(z - score)_t = z_t = \frac{S_t - \overline{S}_t}{\mathrm{SD}(S_t)} = \frac{S_t}{\mathrm{SD}(S_t)} \tag{12.5}$$

Then a threshold of $[-1, 1]$ is set for the z-score. If it's outside the threshold, we make profit from short-term deviation.

1. If $z_t < -1$ and no position is held, enter a *long spread* position by buying β unit of asset 2 and selling 1 unit of asset 1.
2. If $z_t > 1$ and no position is held, enter a *short spread* position by selling β unit of asset 2 and buying 1 unit of asset 1.
3. If $z_t \in [-1, 1]$ and a position is held, maintain it.
4. If $z_t = 0$, close all positions since the spread reverted to its mean.

12.3.5 Limitations of the Static Model

1. **Parameter constancy:** In practice, hedge ratios between assets may evolve due to structural changes such as shifts in business models, market regimes, or relative risk exposures. This leads to a non-stationary spread over time.
2. **Sensitivity to Structural Breaks:** A single regime change can invalidate the cointegration stationarity test. This is a problem in financial time series, where abrupt changes are common.
3. **Lag and Window Dependence:** Rolling-window implementations of static cointegration are often used, but these introduce a trade-off between responsiveness and stability.
4. **Lack of Uncertainty Quantification:** There is no way to distinguish genuine equilibrium shifts from transient noise (different types of noises).

Motivation for the Kalman filter: The limitations of static cointegration highlight the need for a method that allows for the underlying values (hedge-ratio) to evolve over time while accounting for uncertainty. The Kalman filter provides exactly that. It is a recursive estimator for a hidden or latent state observed through noisy measurements. At each step, it predicts the state based on its evolution model and then updates the prediction using new data. In this article, the Kalman filter is used to track a time-varying cointegration relationship by taking the hedge ratio to be the latent state variable.

12.4 Kalman Filter: Likelihood Interpretation

12.4.1 Mean-Squared Error

Kalman filter aims to act as a minimiser for MSE (Mean-Squared Error) [7]. Consider the relation

$$z_t = g_t x_t + v_t$$

where z_t is a time-dependant observed signal, g_t is a gain, x_t is the information-bearing signal and v_t is some additive noise. We first attempt to estimate x_t; let this estimate be $\hat{x}_t$. The error term is then:

$$e_t = x_t - \hat{x}_t$$

The "error-measuring" function $f(e_t)$ is application-dependent, but it should be positive always and monotically increasing. One example is the squared error function.

$$\epsilon(t) = \mathbb{E}[f(e_t)] = \mathbb{E}[e_t^2] \tag{12.6}$$

Remark 12.3. It is important to note that equation 12.6 only locally minimises the MSE.

12.4.2 Maximum Likelihood Statistics

To rigorously define Kalman Filter, we want to maximise the probability of finding z_t given $\hat{x}_t$ (maximum a posteriori estimation). Assume an additive random noise is Gaussian with standard deviaion σ_t and a normalisation constant K_t.

$$\mathbb{P}(z \mid \hat{x}_t) = \prod_t \mathbb{P}(z_t \mid \hat{x}_t) = \prod_t K_t \exp\left(-\frac{(z_t - g_t\hat{x}_t)^2}{2\sigma_t^2}\right) \tag{12.7}$$

$$\log \mathbb{P}(z \mid \hat{x}_t) = -\frac{1}{2}\sum_t \frac{(z_t - g_t\hat{x}_t)^2}{2\sigma_t^2} + constant \tag{12.8}$$

We see that whenever there is a Gaussian noise, MSE appears naturally. The Kalman filter is optimal under linear dynamics and Gaussian noise; under non-Gaussian noise it remains the best linear unbiased estimator.[6]

12.4.3 Kalman Filter: State-space based Derivation

When the Kalman filter uses State-Space techniques, it can be used as a smoother, filter or predictor [11]. The predictor is of special interest for cointegration purposes. State-space models are based on discrete time-steps. Matrices computed for the algorithms estimate the errors. We will have two main “systems": the process that happens and the measurement.

Process

$$x_{t+1} = \Phi x_t + \omega_t \tag{12.9}$$

x_t is the state-vector at time t, Φ is the (stationary) transition matrix, ω_t is some white-noise associated with the process with known covariance.

Measurement

$$z_t = H x_t + \nu_t \tag{12.10}$$

z_t is the actual-measurement of x_t at time k, H is the noiseless (stationary) connection between the state and measurement, ν_t is some white-noise associated with the measurement with known covariance.

To minimise MSE for the optimal filter, we must be able to model errors using Gaussian distributions. Covariances can be assumed stationary for the noise-models. Note that ω_t and ν_t have 0 correlation.

$$Q = \mathbb{E}[\omega_t \omega_t^\top] \tag{12.11}$$

$$R = \mathbb{E}[\nu_t \nu_t^\top] \tag{12.12}$$

The error-covariance matrix at time k is the expectancy value of MSE.

$$P_t = \mathbb{E}[e_t e_t^\top] = \mathbb{E}[(x_t - \hat{x}_t)(x_t - \hat{x}_t)^\top] \tag{12.13}$$

If the “prior estimate" of $\hat{x}_t$ ($\hat{x}'_t$) was gained by system knowledge, measured data can be used for new estimate.

$$\hat{x}_t = \hat{x}'_t + K_t(z_t - H\hat{x}'_t) \tag{12.14}$$

K_t is the Kalman gain and $i_t = (z_t - H\hat{x}'_t)$ is the innovation or measurement residual. Using the equation for z_t and subtracting $\hat{x}_t$ from x_t we get:

$$\begin{aligned} x_t - \hat{x}_t &= x_t - \hat{x}'_t - K_t(Hx_t + \nu_t - H\hat{x}'_t) \\ &= (I - K_t H)(x_t - \hat{x}'_t) - K_t \nu_t \end{aligned} \tag{12.15}$$

Let the prior estimate of P_t be P'_t.

$$P'_t = \mathbb{E}[(x_t - \hat{x}'_t)(x_t - \hat{x}'_t)^\top] \tag{12.16}$$

Also, the error of the prior estimate $(x_t - \hat{x}'_t)$ is uncorrelated with the measurement noise. The error covariance update equation can be written as:

$$\begin{aligned} P_t &= (I - K_t H)\mathbb{E}[(x_t - \hat{x}'_t)(x_t - \hat{x}'_t)^\top](I - K_t H)^\top + K_t \mathbb{E}[\nu_t \nu_t^\top] K_t^\top \\ &= (I - K_t H) P'_t (I - K_t H)^\top + K_t R K_t^\top \end{aligned} \tag{12.17}$$

To minimise the error-covariance matrix P_{tt}, we must minimise the $\text{tr}(P_{tt})$ and thus the $\text{tr}(P_t)$ with respect to K_t.

$$\text{tr}(P_t) = \text{tr}(P'_t) - 2\text{tr}(K_t H P'_t) + \text{tr}(K_t(HP'_t H^\top + R)K_t^\top) \tag{12.18}$$

$$\frac{\partial \text{tr}(P_t)}{\partial K_t} = -2(HP'_t)^\top + 2K_t(HP'_t H^\top + R) = 0 \tag{12.19}$$

$$K_t = P'_t H^\top (HP'_t H^\top + R)^{-1} \tag{12.20}$$

Putting the Kalman-gain back into equation 12.17:

$$\begin{aligned} P_t &= P'_t - P'_t H^\top (HP'_t H^\top + R)^{-1} HP'_t = P'_t - K_t HP'_t \\ &= (I - K_t H)P'_t \end{aligned} \tag{12.21}$$

This is the update equation for error covariance matrix in optimal gain. Finally, we require the state-projection onto the next time-step

$$\hat{x}'_{t+1} = \Phi \hat{x}_t \tag{12.22}$$

We also need the error covariance matrix projection onto t+1

$$\begin{aligned} e'_{t+1} &= x_{t+1} - \hat{x}'_{t+1} = (\Phi x_t + \omega_t) - (\Phi \hat{x}_t) \\ &= \Phi e_t + \omega_t \end{aligned} \tag{12.23}$$

Since e_t and ω_t have 0 cross-correlation:

$$\begin{aligned} P'_{t+1} &= \mathbb{E}[(\Phi e_t + \omega_t)(\Phi e_t + \omega_t)^\top] = \mathbb{E}[\Phi e_t (\Phi e_t)^\top] + \mathbb{E}[\omega_t \omega_t^\top] \\ &= \Phi P_t \Phi^\top + Q \end{aligned} \tag{12.24}$$

Thus, the state-space derivation of the Recursive Linear Kalman Filter is done.

1. **Kalman Gain:** $K_t = P'_t H^\top (HP'_t H^\top + R)^{-1}$
2. **Update Estimate:** $\hat{x}_t = \hat{x}'_t + K_t(z_t - H\hat{x}'_t)$
3. **Update Covariance:** $P_t = (I - K_t H)P'_t$
4. **Project onto k+1:** $\hat{x}'_{t+1} = \Phi \hat{x}_t$ and $P'_{t+1} = \Phi P_t \Phi^\top + Q$

12.5 Dynamic Pairs Trading

The following dynamic pairs trading model is based on Jia Yu's implementation of Kalman Filter for Cointegration [13].

12.5.1 Finding beta

Dynamic pairs-trading is a market neutral strategy. The spread is:

$$S_t = X_{2,t} - \beta X_{1,t} - \alpha \sim I(0)$$

β or the hedge ratio can be found by minimising the Spread with respect to α, β. This is very common in OLS regressions and is given by

$$\beta = \frac{\text{Cov}(X_1, X_2)}{\text{Var}(X_1)} \tag{12.25}$$

This is the form of β when X_1 is chosen as the independent stock, i.e., X_2 is traded at some multiple of X_1.

12.5.2 State-Space model

Applying the state-space equations from Section 4 on the Cointegration relationships: The measurement equation can be derived from the spread equation and process equations can be modelled with hedge ratio being a random walk (I is identity).

$$\begin{aligned} X_{2,t} &= \beta_t X_{1,t} + S_t \\ S_t &\sim \mathcal{N}(0, R) \end{aligned} \tag{12.26}$$

$$\begin{aligned} \beta_t &= I\beta_{t-1} + \nu_t \\ \nu_t &\sim \mathcal{N}(0, Q) \end{aligned} \tag{12.27}$$

The projection equations are:

$$\begin{aligned} \hat{\beta}'_t &= \hat{\beta}_{t-1} \\ P'_t &= P_{t-1} + Q \end{aligned} \tag{12.28}$$

The state-space equations are:

$$\begin{aligned} \hat{X}_{2,t} &= \hat{\beta}'_t X_{1,t} \\ S_t &= X_{2,t} - \hat{\beta}'_t X_{1,t} \\ \hat{\beta}_t &= \hat{\beta}'_t + K_t S_t \end{aligned} \tag{12.29}$$

$$\begin{aligned} P_t &= (I - K_t X_{1,t}) P'_t \\ K_t &= P'_t X_{1,t} (X_{1,t} P'_t X_{1,t} + R)^{-1} \end{aligned} \tag{12.30}$$

β_0 is calculated from historical prices. $R = \mathrm{Cov}(S_t)$ is calculated by actual prices and Q (covariance of volatility of β) is calculated by equilibrium prices of stocks.

Equilibrium price via Kalman Filter

Stock-prices are signals with noise and short-term deviation, but Long-term prices will be price with noise removed, which can be done with Kalman filters. Stock-prices (X_t with noise x_t) are random walks; and Y_t with noise y_t is another auxiliary value, such that V_y(covariance of y_t) is the variance of period volatilites. These are calculated from historical prices. From this Kalman filter, equilibrium prices are outputted as $Y_{1,t}, Y_{2,t}$.

$$\begin{aligned} \beta_{e(t)} &= \mathrm{regress}(Y_{1,t}, Y_{2,t}) \\ Q &= \mathrm{Cov}(\Delta\beta_{e(t)}) \end{aligned} \tag{12.31}$$

12.5.3 Crucial Implication

Since, Kalman filter minimises MSE only locally, and since cointegration is supposed to be a long-run relationship, a dynamic β implies that the Spread is no longer asymptotically stationary. Allowing β_t to evolve destroys the notion of a fixed cointegrating relation and replaces it with a time-varying projection. Now, β acts as a control parameter: it minimises prediction error rather than representing a fixed structural relationship. It is thus also prone to overfitting the noise. As $Q \to 0$, $\beta_t \to$ constant (static EG). As $Q \to \infty$, β_t tracks noise.

12.5.4 Kalman filter as an Instability Indicator

In the equation for Kalman gain, $K_t \propto R^{-1}$. R is the measurement noise; thus, a higher R means lesser trust in newer observations. A stable system (β does not change fast) is in equilibrium and thus has high R and low K_t. Similarly, an unstable system (β changes rapidly) has low R and high K_t.The code for the simulations can be found on Github.[1]

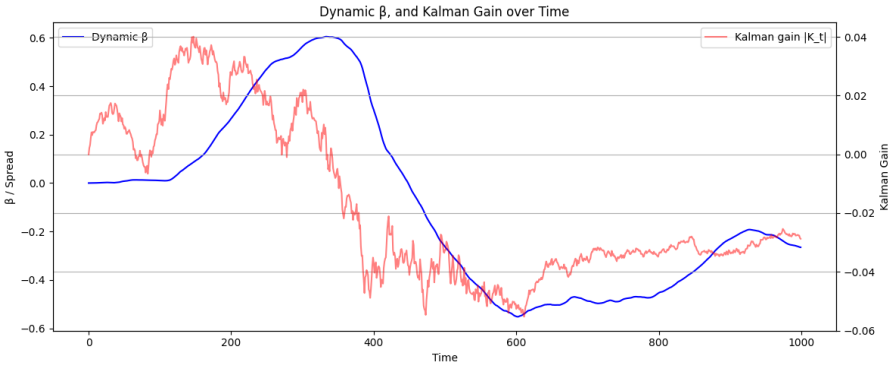


Figure 12.3: Dynamic β and Kalman Gain over Time for $R = 10$

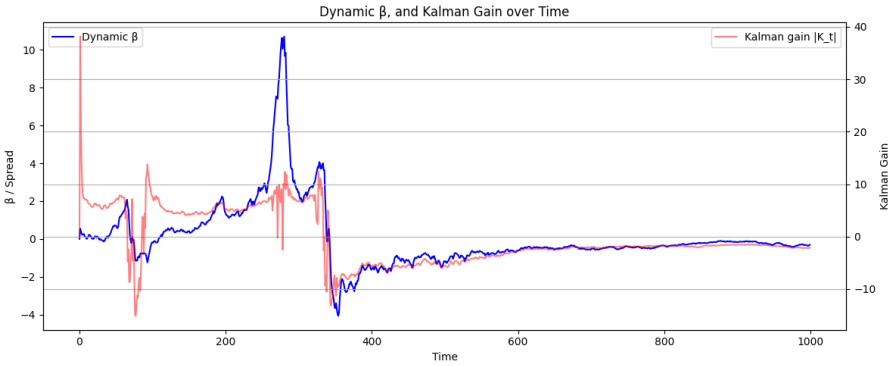


Figure 12.4: Dynamic β and Kalman Gain over Time for $R = 10^{-4}$

12.5.5 Simulation: Comparison of Static/Dynamic

The spread is traded based on mean-reversion. For the static case, we obtain one data-point of Sharpe Ratio and an Average Turnover of 0. But, in the dynamic case, depending on the turnover, the Sharpe Ratio changes. A random series of prices are generated and then the corresponding data is plotted below.[2]

[1] https://github.com/rayray404/dyn-ci/blob/main/instability_indicator.py
[2] https://github.com/rayray404/dyn-ci/blob/main/turnover_sharpe.py

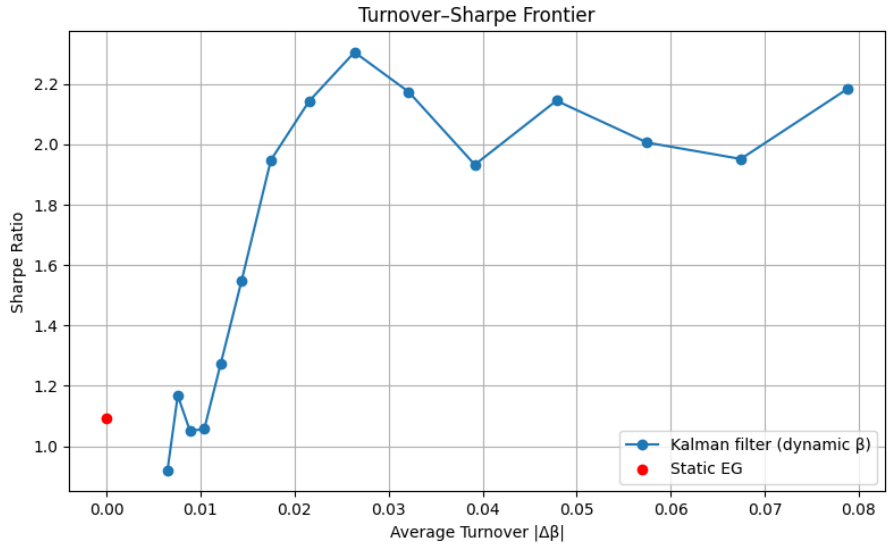


Figure 12.5: Sharpe-Ratio v/s Turnover for Static EG and Kalman Filter based dynamic β

It is observed that for static EG, there is lesser profits, but at the same time lesser trades done; it is confident in its structural constancy. For, the dynamic case, the model tries to fit the equilibrium noise and thus it is very reactive to small changes.

12.6 Extensions to the Dynamic Model

12.6.1 Regime-Switching Between Static and Dynamic models

An alternative extension is to allow the hedge ratio β_t to evolve under **distinct regimes**, reflecting different market conditions. Rather than confining β_t to be always static or always dynamic, the stability determines which model is to be applied. In this interpretation, the static/stable regime captures periods where cointegration-like behaviour dominates, while the dynamic/unstable regime allows β_t to adapt during structural shifts, volatility spikes, or regime changes. This framework reconciles static and dynamic models within a single structure, emphasising that neither approach is universally superior; rather, their relevance is regime-dependent.

12.6.2 Mean-Reverting Model: Disciplined Kalman Filter

While the standard Kalman filter treats the hedge ratio β_t as a random walk, this can lead to excessive sensitivity to short-term noise. To impose economic discipline, we model β_t as a **mean-reverting process**. This would be similar to a disciplined/constrained Kalman filter [1].

$$\beta_t = \beta_{t-1} + \kappa(\bar{\beta} - \beta_{t-1}) + \nu_t, \tag{12.32}$$

where $\bar{\beta}$ represents an economically anchored long-term level (static EG hedge ratio or a sectoral beta), $\kappa \in [0, 1]$ controls the speed of mean reversion, and ν_t is the process noise. This specification ensures that β_t fluctuates around a plausible economic equilibrium rather than drifting freely.

12.6.3 Volatility-Scaled Noise

Furthermore, we allow the process noise to be **volatility-scaled**:

$$\nu_t \sim \mathcal{N}(0, \sigma_{x,t}^2 Q), \tag{12.33}$$

where $\sigma_{x,t}^2$ is the estimated variance of the driving series x_t, and Q is a baseline noise parameter. By scaling the noise with market volatility, the model adapts more rapidly in turbulent periods and remains conservative in calm markets.

12.7 Conclusion

The Kalman filter-based dynamic β is **not a superior form of cointegration** compared to the static Engle–Granger approach; rather, it represents a *different object* with distinct objectives and failure modes. While static β captures the long-run structural equilibrium, dynamic β adapts locally to minimise prediction error, overfitting short-term noise if interpreted as cointegration. The simulations and analysis demonstrate that **both static and dynamic approaches can outperform each other depending on the context**:

- Static Engle–Granger is more reliable when the underlying equilibrium is stable and structural relationships are constant.
- Dynamic Kalman-filter β is advantageous under structural changes or regime shifts, particularly when combined with mean-reverting and volatility-scaled noise.
- The Kalman gain K_t serves as an *instability indicator*: high K_t reflects low confidence in equilibrium, signalling structural breaks or turbulent market conditions.
- Incorporating mean reversion and regime-switching models in β_t ensures economic plausibility and avoids excessive reactivity to noise.

Applications: These findings align with real-world practices: firms often use hybrid or adaptive models that balance responsiveness to changing market conditions while adhering to structural equilibrium. In summary, **dynamic cointegration should be treated as a control system for equilibrium tracking**, not as a replacement for classical cointegration. Recognizing its distinct objective and monitoring the Kalman gain allows for more informed modelling choices, better risk management, and mitigation of overfitting in pairs trading strategies.

Bibliography

[1] Nesrine Amor, Ghulam Rasool, and Nidhal C. Bouaynaya. Constrained state estimation – a review, 2022. URL https://arxiv.org/abs/1807.03463.

[2] Richard M Bookstaber. *A demon of our own design : markets, hedge funds, and the perils of financial innovation*. J. Wiley, 2007.

[3] Robert F. Engle and C. W. J. Granger. Co-integration and error correction: Representation, estimation, and testing. *Econometrica*, 55:251–276, 1987.

[4] C.W.J. Granger and P. Newbold. Spurious regressions in econometrics. *Journal of Econometrics*, 2:111–120, 07 1974. doi: 10.1016/0304-4076(74)90034-7.

[5] Rad Hossein, Rand Kwong Yew Low, and Robert W. Faff. The profitability of pairs trading strategies: Distance, cointegration, and copula methods. *SSRN Electronic Journal*, 2015. doi: 10.2139/ssrn.2614233.

[6] Jeffrey Humpherys, Preston Redd, and Jeremy West. A fresh look at the kalman filter. *SIAM Review*, 54(4):801–823, 2012. doi: 10.1137/100799666.

[7] R E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. doi: 10.1115/1.3662552.

[8] Daniel P Palomar. 15.2 cointegration and correlation | portfolio optimization, 02 2025.

[9] Said E Said and David A Dickey. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3):599–607, 12 1984. doi: 10.1093/biomet/71.3.599.

[10] James Stock and Mark Watson. The evolution of national and regional factors in us housing construction. *Volatility and time series econometrics: essays in honor of Robert F. Engle*, 2008.

[11] N Thacker and A Lacey. Tutorial: The likelihood interpretation of the kalman filter, 01 1996.

[12] Ganapathy Vidyamurthy. *Pairs Trading*. John Wiley Sons, 02 2011.

[13] Jia Yu. Cointegration approach for the pair trading based on the kalman filter. *Atlantis Highlights in Computer Sciences/Atlantis highlights in computer sciences*, pages 633–642, 01 2023. doi: 10.2991/978-94-6463-102-9_66.

End of Sixth Edition

Thank you for reading the sixth edition of the Mathematical Finance Magazine!

For more resources and information, visit the WFS Linktree.

Edited by the WFS Magazine Editorial Team


WARWICK
FINANCE SOCIETIES