

Fuzzy Volatility Networks for Enhanced Portfolio Optimization and Model Risk Forecasts for Deep Learning Models

by

Avanthi Saumyamala Moragammanna Gedara

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfilment of the requirements of the degree of

MASTER OF SCIENCE

Department of Statistics

University of Manitoba

Winnipeg

Copyright © 2025 by Avanthi Saumyamala Moragammanna Gedara

Abstract

This thesis explores advanced statistical and deep learning models for volatility forecasting and network inference for enhanced portfolio optimization. In the domain of risk forecasting, traditional time series models along with machine learning and deep learning models are evaluated for volatility forecasting. These models are also used to generate model risk forecasts for deep learning models, offering an advanced framework to assess financial system stability. Furthermore, this thesis demonstrates the superiority of the neural network autoregressive model (NNAR) with forecast linear augmented projection (FLAP) over the autoregressive integrated moving average (ARIMA) model with FLAP in forecasting multivariate price and volatility. This performance achievement leads to reduced forecast error variance. The thesis further introduces data-driven fuzzy volatility networks and nonlinear adaptive fuzzy adjacency matrices, which capture model uncertainty in volatility correlations and enable the construction of resilient network structures. These are employed in combination with clustering techniques, network-based community detection methods, and PageRank for portfolio construction, leading to diversified portfolios with improved cumulative returns. Another key innovation is introducing neuro correlation networks using nonlinear correlation measures derived from innovations of NNAR models. Dynamic price networks are used to account for the non-stationarity in financial time series, enabling improved insight into

market structure and systemic risk over time. Random matrix theory (RMT) is used to denoise covariance matrices, thereby improving the accuracy of estimation, demonstrating superior performance in portfolio optimization, especially with t -distributed returns. Moreover, the theorem of combined estimating functions, which introduces a new risk measure with a smaller variance, is studied and applied in both risk forecasting and portfolio optimization. Extensive simulations and empirical analysis using stocks, cryptocurrencies, and other assets validate the efficiency of these methods.

Acknowledgment

I would like to thank my supervisors, Dr. A. Thavaneswaran and Dr. R. Thulasiram, for their valuable advices, continuous guidance and financial support during my research and study in the MSc. program. I want to thank my committee members, Dr. Sumeet Kalia and Dr. Gyanendra Pokharel, for their valuable comments and insights in making this thesis successful. I am grateful to my parents and my loving husband, Sanka, son, Aveen for their commitments, encouragement, and love. Finally, I also thank the University of Manitoba, for providing me the opportunities and financial support during my research. I am also grateful to all members of the Department of Statistics for their support throughout this program.

Dedication

This thesis is dedicated to the unwavering pillars of my life, my son, husband, mother, father, family, friends, and teachers who have supported me with heart and soul.

Contents

Contents	iii
List of Tables	ix
List of Figures	xiii
Contribution of Author	xix
1 Introduction	1
2 Risk Forecasting	9
2.1 Linear and Non-linear Predictions and Predictive Models . . .	10
2.2 Models	12
2.2.1 Exponential Smoothing (ETS)	13
2.2.2 Prophet Model	13
2.2.3 ARIMA (p, d, q) Model	14

2.2.4	NNAR(p,k) Model	15
2.2.5	NNVol(p,k) Model	16
2.2.6	Recurrent Neural Network (RNN) Model	18
2.2.7	Long Short-Term Memory (LSTM) Model	19
2.2.8	Gated Recurrent Unit (GRU) Model	22
2.2.9	Model Risk	24
2.3	Univariate Volatility Forecasting	25
2.4	Multivariate Volatility Forecasting	30
2.4.1	VAR(p) Model	30
2.4.2	Forecast Linear Augmented Projection (FLAP)	32
2.4.3	Simulation Study	35
2.4.4	Real Data Experiments	40
2.5	Conclusion	44
3	Network Analysis	47
3.1	Adjacency Matrices and Networks	50
3.1.1	Empirical Correlation Matrix	51
3.1.2	DDEWMA Correlation Matrix	53
3.1.3	Cosine Similarity Matrix	55

3.2	Fuzzy Adjacency Matrices and Fuzzy Networks	57
3.2.1	Non-Linear Fuzzy Number and α -cuts	58
3.2.2	Fuzzy Cryptocurrency Networks Using Log Returns . .	60
3.3	Volatility Network	62
3.3.1	Covariance Relationship	64
3.3.2	Eigenvalue Distribution and the Marchenko-Pastur Law	70
3.3.3	Data Driven Volatility Correlation Networks	71
3.3.4	Effects of Data Driven Distribution of Returns on Volatility Networks	73
3.4	Autocorrelation Networks	73
3.5	Granger Causality Network	75
3.6	Networks for Big Data	81
3.6.1	Network Metrics	81
3.6.2	Stock Networks	85
3.7	Dynamic Price Correlation Networks	89
3.7.1	Price Correlation Network	90
3.7.2	ARIMA Innovation Correlation Network	91
3.7.3	Neuro Correlations based on NNAR Model	93
3.8	Conclusion	95

4	Portfolio Optimization	99
4.1	Clustering for Enhanced Portfolio Optimization	101
4.1.1	Distance-Based Clustering Methods	103
4.1.2	Cluster based Portfolio Optimization	108
4.1.3	Network based Clustering	111
4.1.4	Dynamic Clustering	113
4.1.5	Network based Portfolio Optimization	114
4.2	RMT Filtering for Enhanced Portfolio Optimization	118
4.2.1	Marchenko-Pastur Distribution	119
4.2.2	Covariance Models and Financial Networks	123
4.2.3	Simulation Study	132
4.2.4	Real Data Experiment	134
4.3	Conclusion	138
5	Applications of Combined Estimating Functions	149
5.1	Application 1 : Portfolio Optimization	159
5.2	Application 2: Generalized VaR and ES Forecasting	169
5.3	Conclusion	180
6	Conclusions and Contribution of the Thesis	183

Bibliography	187
Appendices	199
A Combined Estimating Function Theorem	199
B Trace Minimization in Estimating Functions	202
C Simulation Results of Moment Comparison	204
D Abbreviations used for Portfolios	205

List of Tables

2.1	Forecast accuracy of six cryptocurrencies and their model risk	26
2.2	One step ahead volatility forecast of BTC using DL models . .	27
2.3	One step ahead volatility forecast of ETH using DL models . .	28
2.4	One step ahead volatility forecast of BNB using DL models . .	28
2.5	One step ahead volatility forecast of XRP using DL models . .	28
2.6	One step ahead volatility forecast of DOGE using DL models .	28
2.7	One step ahead volatility forecast of ADA using DL models . .	29
2.8	Model risk of five deep learning models	29
2.9	Accuracy of 12-step-ahead VAR, ARIMA and ARIMA+FLAP forecasts from the simulated VAR(1) data with normally dis- tributed errors	37
2.10	Accuracy of 12-step-ahead VAR, NNAR and NNAR+FLAP fore- casts from the simulated VAR(1) data with normally distributed errors	37

2.11 Accuracy of 12-step-ahead VAR, ARIMA and ARIMA+FLAP forecasts from simulated VAR(1) data with t distributed errors ($\nu = 3$)	38
2.12 Accuracy of 12-step-ahead VAR, NNAR and NNAR+FLAP forecasts from simulated VAR(1) data with t distributed errors($\nu = 3$)	38
2.13 Accuracy of 12-step-ahead VAR,ARIMA and ARIMA+FLAP forecasts from simulated VAR(1) data with Laplace distributed errors	39
2.14 Accuracy of 12-step-ahead VAR,NNAR and NNAR+FLAP forecasts from simulated VAR(1) data with Laplace distributed errors	39
3.1 Parameters for minimally connectedness of fuzzy networks based on log returns of cryptocurrency	62
3.2 Parameters for minimally connectedness of fuzzy networks based on volatility of cryptocurrency	64
3.3 Scale factor values for 5 theoretical distributions and 12 data driven distributions	70
3.4 PageRank scores for cryptocurrency network nodes	84
3.5 Network metrics of return networks	88
3.6 Network metrics of price networks	95

4.1	Cluster based portfolio summary	110
4.2	Modularity score of communities detected in networks	113
4.3	Portfolio selected from correlation networks	117
4.4	Portfolio risk, return and Sharpe ratio results of 10 th window .	137
5.1	Networks and portfolio selection from identified communities .	167
5.2	Summary statistics of asset returns in a portfolio selected from sectors	167
5.3	Summary statistics of daily return for cryptocurrencies: Before pandemic	174
5.4	Summary statistics of daily return for cryptocurrencies: During pandemic	175
5.5	One step ahead volatility forecasts of Cryptocurrency	176
5.6	Non-regularized and regularized risk forecasts of cryptocurrency before the pandemic	180
5.7	Non-regularized and regularized risk forecasts of cryptocurrency during the pandemic	180
1	Moment comparison: theoretical vs. empirical Under normal and <i>t</i> -distributions	204
2	Abbreviations used for portfolios	205

List of Figures

2.1	<i>NNVol(p, k)</i> model	17
2.2	LSTM architecture	20
2.3	GRU architecture	22
2.4	Proportional and cumulative variance explained by principal components of original price data	41
2.5	MSE of ARIMA and NNAR volatility forecasts at different forecast horizons ($h = 3(blue), 12(red), 21(green), 30(yellow)$) for all 56 stocks	42
2.6	Comparison of out-of-sample total RMSE of of 12 step ahead VAR, ARIMA and NNAR price and volatility forecasts using different p number of components for FLAP projection (Green dashed line - VAR forecast, Blue dashed line - original ARIMA forecast, BLUE solid line - FLAP ARIMA forecast, Red dashed line - original NNAR forecast, Red solid line - FLAP NNAR forecast)	44

2.7	Comparison of Out-of-sample forecast accuracy of 5 stocks (Green dashed line - VAR forecast, Blue dashed line - original ARIMA forecast, BLUE solid line - FLAP+ARIMA forecast, Red dashed line - original NNAR forecast, Red solid line - FLAP+NNAR forecast)	46
3.1	Empirical correlation network based on log returns of cryptocurrencies	52
3.2	DDEWMA correlation network based on log returns of cryptocurrencies	56
3.3	Cosine similarity network based on log returns of cryptocurrencies	57
3.4	Plot of α -cuts of the non-linear adaptive asymmetric fuzzy number	59
3.5	Fuzzy networks of empirical correlation matrices based on log returns of cryptocurrency	63
3.6	Fuzzy networks of DDEWMA correlation matrices based on log returns of cryptocurrency	64
3.7	Fuzzy networks of cosine similarity matrices based on log returns of cryptocurrency	65
3.8	Empirical correlation network based on volatility of cryptocurrencies	66
3.9	Cosine similarity network based on volatility of cryptocurrencies	67

3.10 Fuzzy networks of empirical correlation matrices based on volatility of cryptocurrency	68
3.11 Fuzzy networks of cosine similarity matrices based on volatility of cryptocurrency	69
3.12 Comparison of theoretical MP density and empirical eigenvalue of t (df=3.5) distribution vs normal distribution	72
3.13 Cross-autocorrelation network	76
3.14 Granger causal network and heatmap of log returns of cryptocurrency	80
3.15 Granger causal network and heatmap of volatility of cryptocurrency	81
3.16 Effect of damping factor on Page rank of cryptocurrencies . . .	85
3.17 Volatility correlation network and eigenvalue distribution . . .	86
3.18 Data driven volatility correlation network and eigenvalue distribution	86
3.19 Return correlation network and eigenvalue distribution	87
3.20 Sign($\pm$) correlation network and eigenvalue distribution . . .	87
3.21 Price correlation network and eigenvalue distribution	91
3.22 ARIMA innovation correlation network and eigenvalue distribution	93
3.23 Neuro correlation network and eigenvalue distribution	94

3.24	Dynamic (monthly) price networks January - December 2024 .	97
4.1	k-means clustering of 50 technological stocks using standardised expected return and variance	105
4.2	Hierarchical clustering of 50 technological stocks using Sharpe ratio	107
4.3	Cumulative return plots obtained from five distance based clus- tering methods	109
4.4	Cumulative return plots obtained from two spectral clustering methods	111
4.5	Dynamic(monthly) price network and community graphs Jan- uary - December 2024	140
4.6	Three price correlation networks of window 1 at threshold=0.6,0.19,0.19 respectively	141
4.7	Communities detected from three price correlation networks .	141
4.8	Cumulative returns based on volatility correlation of the three portfolios in window 1	142
4.9	Cumulative returns based on volatility correlation of the three portfolios in window 2 -10	143
4.10	Portfolio weight distribution with RMT-filtered and non-filtered empirical covariance matrix based on normally and t distributed data	144

4.11 Side by side boxplots of portfolio weights with normal/ t distributed covariance matrices and RMT-filtered normal/ t covariance matrices and weights distributions across different degrees of freedom	144
4.12 Correlation networks based on normal and t distributed data and respective average node degrees	145
4.13 Empirical, data driven and neuro correlation networks of 10^{th} window	145
4.14 Portfolio risk and return plot for the 10^{th} window	146
4.15 Sharpe ratio of tangency portfolios based on empirical and data driven covariance matrices with and without RMT-filtering over the rolling windows	146
4.16 Cumulative returns of the portfolios - 10^{th} window	147
4.17 Cumulative returns of the portfolios - windows 1-10	148
5.1 Empirical correlation network graph and community graph . . .	164
5.2 Data-driven correlation network graph and community graph . . .	165
5.3 Partial correlation network graph and community graph	165
5.4 Volatility correlation network graph and community graph . . .	166
5.5 Cumulative returns of portfolios	168
5.6 Daily adjusted closing price series of cryptocurrencies	174

5.7	VaR and ES forecasts using MA and HS models	177
5.8	VaR and ES forecasts using NGARCH and tGARCH models .	178
5.9	VaR and ES forecasts using EWMA and DDEWMA models .	179
5.10	Model risk ratio of VaR before (Left) and after (Right) the pandemic (p values range from 0.001 to 0.01): non-regularized (green), regularized (blue)	181

Contribution of Authors

The thesis is structured with five chapters and Chapter 2 (Risk forecasting), Chapter 3 (Network Analysis), Chapter 4 (Portfolio Optimization) and Chapter 5 (Applications of Combined Estimating Functions) containing materials from research articles that have been peer reviewed and published. The following are the corresponding chapters in the thesis, titles of the research articles, and the authors' contributions:

1. Chapter 2 includes the content of the article “Superiority of the Neural Network Models for Multivariate Price and Volatility Forecasting”.
 - **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript
 - **You Liang**: Conceptualization, Methodology, Supervision, Writing - review & editing
 - **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.
 - **Alex Paseka**: Domain Knowledge, Writing - review & editing.

- **Sulalitha Bowala:** Methodology, Software, Writing - review & editing
- **Ruppa Thulasiram** (Co-Advisor): Conceptualization, Supervision, Writing - review & editing.

2. Chapter 3 includes the content of the articles:

(a) “Novel Non-linear Adaptive Fuzzy Adjacency Matrices for Financial Volatility Network Models”

- **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript
- **Sulalitha Bowala:** Conceptualization, Visualization, Methodology, Software, Writing - review & editing
- **You Liang:** Conceptualization, Methodology, Supervision, Writing - review & editing
- **Shanika Basnayaka:** Writing - review & editing
- **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.
- **Ruppa Thulasiram** (Co-Advisor): Conceptualization, Supervision, Writing - review & editing.

(b) “Novel Financial Network Models Using Neuro Correlations and Applications”

- **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript
- **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.
- **Sulalitha Bowala**: Methodology, Software, Writing - review & editing
- **Joy Das**: Methodology, Software
- **Ruppa Thulasiram** (Co-Advisor): Conceptualization, Supervision, Writing - review & editing.
- **Alex Paseka**: Domain Knowledge, Writing - review & editing.

(c) “Novel Data Driven High Dimensional Volatility Networks and Dynamic Price Networks”.

- **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript
- **Sulalitha Bowala**: Methodology, Software, Writing - review & editing
- **Erfanul Hoque**: Conceptualization, Methodology, Supervision, Writing - review & editing
- **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.

- **Alex Paseka**: Domain Knowledge, Writing - review & editing.
 - **Ruppa Thulasiram** (Co-Advisor): Supervision, Writing - review & editing.
3. Chapter 4 includes content from the article “Superiority of RMT Filtered Data-Driven Covariance Matrices for Resilient Networks and Portfolio Optimization”.
- **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript
 - **Sulalitha Bowala**: Methodology, Software, Writing - review & editing
 - **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.
 - **Alex Paseka**: Domain Knowledge, Writing - review & editing.
 - **You Liang**: Domain Knowledge, Writing - review & editing
 - **Ruppa Thulasiram** (Co-Advisor): Supervision, Writing - review & editing.
4. Chapter 5 includes the content of the article “Computational Finance Applications of Combined Estimating Functions”
- **Avanthi Saumyamala** (Thesis Author): Conceptualization, Data Collection, Visualization, Methodology, Software, Draft the Manuscript

- **Aerambamoorthy Thavaneswaran** (Advisor): Conceptualization, Methodology, Supervision, Writing - review & editing.
- **Ruppa Thulasiram** (Co-Advisor): Conceptualization, Supervision, Writing - review & editing.

Chapter 1

Introduction

The study of financial data for risk forecasting and investment optimization has been of continuous research interest among researchers and investors. Statistics plays a major role in financial data analysis and it has been further motivated by introduction of machine learning, deep learning and graphical network techniques in the recent past. This thesis focuses on three major research applications; risk forecasting, network inference and portfolio optimization. The experiments are based on stocks, cryptocurrencies, and other assets in the financial market.

Risk forecasting is a vast area of study where forecasting techniques become advanced and complex day by day. Several studies have been done in the area of volatility forecasting with continuous upgrading. In finance, volatility which is the standard deviation of log returns is commonly forecasted using variations of generalized autoregressive conditional heteroskedasticity (GARCH) models. Volatility can be considered an adequate risk measure when log returns

follow a normal distribution. However in [Thavaneswaran et al. \(2020\)](#) it is shown that log returns are t -distributed with degrees of freedom less than four. When the degrees of freedom of the t -distribution is less than four, the asymptotic variance of the conditional variance forecasts based on the t -GARCH model becomes infinity and cannot be used. Therefore, in [Thavaneswaran et al. \(2019a\)](#) a feed-forward neural network model has been used without any distributional assumptions to forecast the conditional volatility outperforming GARCH models. This model is a nonlinear data-driven model that enhances the practical usage. In [Singh et al. \(2022\)](#), data-driven volatility forecasting and neuro-volatility forecasting have been shown to be superior to the traditional volatility forecasting methods. The neural network autoregressive (NNAR) model ([Hyndman and Athanasopoulos, 2018](#)) is widely used for long term forecasting of non-linear, non-stationary data ([Hoque et al., 2024a](#)). Further, it is also known that when making decisions based on forecasts, relying on a single forecasting model increases risk. Hence, multiple models are used and model risk is calculated to examine the stability of the system. includes statistical, machine learning, and deep learning models to forecast volatility and compare the forecast performances.

In Chapter [2](#) (Risk forecasting), performance of time series models are compared with performance of machine learning and deep learning models for univariate volatility forecasting. Further multivariate time series forecasting is improved by combining non linear neural network model with forecast linear

augmented projection (FLAP) method. Chapter 2 captures basic models and prior work from the literature using these models relevant to the research problem and proposed approach. Also, experiments, results and discussions from this topic to my research study are presented.

Financial networks, commonly known as stock networks, illustrate the interconnections between stocks within a given market. By representing entities as nodes and their relationships as edges, graphs facilitate the study of various real-world phenomena across domains (Pitas, 2016). Among the various types of networks, correlation networks are widely utilized, leveraging statistical associations between stocks derived from their price fluctuations over time (Saha et al., 2022). One of the pivotal metrics defining risk is volatility. However, asset volatility has not been used in the literature to model financial networks using the network theory. Further, the existing studies do not capture the non-linear characteristics in log returns and uncertainty involved with the cross correlations of log returns.

Chapter 3 (Network Analysis) discusses class of networks: cross-correlation networks, similarity networks, autocorrelation networks, causal networks and inferring networks of financial assets to understand the systemic risk of the market. This chapter proposes a novel data-driven non-linear fuzzy volatility network model based on the observed volatility by incorporating uncertainty of the correlation structure in to the networks. Further, ARIMA innovation correlation networks and neuro-correlation networks are introduced. Consid-

ering the non stationarity of the stock prices, dynamic price networks are also discussed in some detail. Chapter 3 provides an overview of fundamental network models and reviews relevant literature that employs these models in relation to the research problem and the proposed methodology. In addition, the chapter presents experiments, results, and discussions that demonstrate the contribution of this topic to the overall research study.

Portfolio optimization faces several challenges, particularly in high-dimensional settings where the number of assets is large relative to the available historical observations. In such cases, the sample covariance matrix becomes poorly estimated, often containing a significant amount of noise that can distort risk assessment and lead to unstable portfolio weights. The problem of noise in the covariance matrix becomes even worse when the matrix is nearly singular, making its inversion unreliable. Moreover, achieving effective diversification becomes more difficult in high dimensions, as noisy correlations can give the false impression of diversification while actually concentrating risk. These issues can result in portfolios that perform well in-sample but fail to generalize out-of-sample, undermining the practical usefulness of optimization results. Addressing these problems often requires techniques such as dimensionality reduction, regularization, or noise-filtering methods like random matrix theory to obtain more robust covariance estimates and improve the stability of portfolio allocations. In literature Jaeger and Marinelli (2022) have shown that network-based portfolios outperform traditional portfolio optimization strategies due to the ability of networks to capture linear and nonlinear relationships

between stocks. Improving the existing methods, [Saumyamala et al. \(2024\)](#) introduced novel approach of network based portfolio optimization by combining community detection and Page ranking to attain better diversification.

Chapter 4 (Portfolio Optimization) includes two sub sections. Section 4.1 discuss on enhancing portfolio optimization using clustering techniques such as distance based clustering, hierarchical clustering, and spectral clustering. Cluster based portfolio optimization is proposed as a solution to high dimensional problem by considering clusters as portfolios. Section 4.1 also discusses on different network based community detection methods and proposes a novel strategy combining network communities and Page ranking algorithm to diversify portfolios. Moreover, Section 4.2 first discusses eigenvalue distribution, random matrix theory and the role of distributional assumptions of data with relevant to theories. Then, RMT filtering is applied to improve the covariance estimation (by denoising the covariance matrix) and is used in portfolio optimization context. Chapter 4 captures challenges in portfolio optimization, existing solutions, and prior work from the literature using these methods relevant to the research problem and the proposed approach. Also, experiments, results and discussions from this topic to my research study are presented.

Combining different estimating functions has been shown to improve efficiency by reducing the overall variance of the combined estimator compared to individual components ([Godambe, 1985](#)). Chapter 5 of the thesis discusses on

the theoretical background and applications in risk forecasting and portfolio optimization. In the first application, instead of minimizing the traditional portfolio variance to obtain portfolio weights as in minimum variance portfolio optimization, the focus is on minimizing the variance of the combined estimating function based on portfolio returns which has smaller variance. In the second application, risk forecasting, a common problem is that forecasting models perform well on the training data but fail to deliver the same performance on the test data (a phenomenon known as overfitting). Incorporating regularization techniques into risk forecasting can prevent overfitting, enabling the development of more robust models that generalize better to unseen data. A novel regularized risk forecasting model developed by [Liang et al. \(2020\)](#) was applied to a set of technology stocks, demonstrating that regularization improved forecasting accuracy. Accordingly, this chapter also discusses regularized Value at Risk (VaR) and Expected Shortfall (ES) forecasting based on sign correlation ([Thavaneswaran et al., 2015](#)), as well as the superiority of NNAR models in volatility forecasting. Chapter 5 captures theoretical background of combined estimating functions and prior work from the literature relevant to the research problem and two applications of the results derived from the theory in finance. Also, experiments, results and discussions from this topic to my research study are presented.

In summary, this thesis proposes the NNAR model for univariate risk forecasting and the NNAR model with FLAP for multivariate price and volatility forecasting, aiming to achieve better accuracy than existing models. Further-

more, it introduces novel fuzzy volatility networks that capture uncertainty in correlations, along with a class of networks that includes innovation correlation networks. It also presents a cluster-based portfolio optimization method for dimensionality reduction, a network-based portfolio optimization method to enhance diversification, and a random matrix theory filtering approach to improve high-dimensional covariance estimation by denoising the covariance matrix.

Chapter 2

Risk Forecasting

An accurate volatility/risk forecast of financial assets is essential for portfolio optimization and informed investment decisions (Zhu et al., 2020). Although deep learning models have shown promise in predicting stock returns (Avramov et al., 2023), their performance often declines during periods of high market volatility. Traditional statistical and machine learning models can overfit training data, reducing the predictive accuracy of new data. Incorporating regularization can mitigate overfitting and improve the accuracy of financial risk forecasting (Liang et al., 2020). Neural networks are increasingly used for time series forecasting due to their flexibility and accuracy. The NNAR model, proposed by Hyndman and Athanasopoulos (2018), is widely used for long-term forecasting. Feed-forward neural networks have outperformed GARCH models in forecasting conditional volatility (Thavaneswaran et al., 2019b), and data-driven or neuro-volatility methods continue to demonstrate

superior performance. Common non-linear models include NNAR, Prophet, recurrent neural network (RNN), and long short term memory (LSTM) models. Recently, these models have been applied to forecast non-linear, non-stationary data such as stock prices, volatility, and temperature (Hoque et al., 2024b).

2.1 Linear and Non-linear Predictions and Predictive Models

Linear prediction is optimal for Gaussian processes. Following Thavaneswaran and Heyde (1999) , given $Y = \theta + \varepsilon$ consider the problem of predicting θ , where θ with mean zero and known variance σ_θ^2 and ε with mean zero and known variance σ_ε^2 are independent. Here, θ is normal, and ε is non-normal with the density $f(\varepsilon)$. Among all linear predictions, the best linear prediction of θ is given by

$$\hat{\theta} = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \sigma_\varepsilon^2} Y$$

with mean square error:

$$P = E(\theta - \hat{\theta})^2 = \frac{\sigma_\theta^2 \sigma_\varepsilon^2}{\sigma_\theta^2 + \sigma_\varepsilon^2}$$

The optimal linear predictor for θ can also be derived as the solution of the

combined linear estimating function:

$$g_L^*(Y; \theta) = \frac{\theta - E(\theta)}{\sigma_\theta^2} + \frac{Y - \theta}{\sigma_\varepsilon^2},$$

which leads to:

$$\theta^* = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \sigma_\varepsilon^2} Y.$$

The information associated with this estimating function is:

$$I_{g_L^*} = \frac{1}{\sigma_\theta^2} + \frac{1}{\sigma_\varepsilon^2} = P^{-1}.$$

In general, the best prediction for θ using the mean square criterion is $E[\theta|Y]$.

Given certain regularity conditions, an application of Bayes' theorem combined with a Taylor series expansion yields the following result:

$$E[\theta|Y] = Y + \sigma_\varepsilon^2 \frac{\partial \ln f(\varepsilon)}{\partial \varepsilon} + o(\sigma_\varepsilon^2).$$

The conditional mean $E[\theta|Y]$ has the minimum mean squared error (MMSE),

$$E[(\theta - E[\theta|Y])^2|Y],$$

among all predictions, which is less than the MSE of the best linear prediction

$\hat{\theta}$. The MSE difference is given by

$$(\hat{\theta} - E[\theta|Y])^2 = \left((1 - \lambda)Y + \frac{\partial \ln f(\varepsilon)}{\partial \varepsilon} \sigma_\varepsilon^2 + o(\sigma_\varepsilon^2) \right)^2.$$

Thus, when $f(\varepsilon)$ is known, the best predictor ($E[\theta|Y]$) has a smaller MSE compared to the best linear predictor.

2.2 Models

All models have their specific benefits and fallbacks, and hence, it is important to compare different models to choose the best-fit model for the data considered. Common forecasting models for time series data are ARIMA, NNAR, Prophet, RNN, and LSTM models. Neural network models are becoming popular in time series forecasting due to their accuracy and capability to handle more complex real-world problems.

In computational finance, one of the most important risk measures is volatility, typically defined as the standard deviation of log returns (r_t) or simple returns (R_t). Let the price of an asset at time t is P_t , we consider the simple return

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

For given asset returns, the observed volatility (V_t) can be calculated as

$$V_t = \frac{|R_t - \bar{R}|}{\hat{\rho}_{sign}}.$$

where $\bar{R}$ is the mean of returns series and ρ_{sign} is the sign correlation, $\rho_{sign} = cor(R_t - \bar{R}, sign(R_t - \bar{R}))$ as in [Thavaneswaran et al. \(2020\)](#).

2.2.1 Exponential Smoothing (ETS)

Exponential smoothing generates forecasts using weighted averages of past observations, with the weights decreasing exponentially over the time. The oldest observations receive the smallest weight while most recent observation receive the highest weight:

$$\hat{y}_{t+1|t} = \alpha y_t + \alpha(1 - \alpha)y_{t-1} + \alpha(1 - \alpha)^2 y_{t-2} + \dots,$$

where $0 \leq \alpha \leq 1$ is the smoothing parameter. The one-step-ahead forecast for time $t + 1$ is a weighted average of all the observations in the series $y_1, \dots, y_t$. The diminishing rate of weight is determined by the parameter α . For any α , the weights assigned to the observations decrease exponentially as we move backward in time. A small α value close to 0 allocates more weight to observations from the distant past, while a large α value closer to 1 assigns more weight to recent observations. In the extreme case where $\alpha = 1$, forecasts reach the naïve forecasts $\hat{y}_{t+1|t} = y_t$.

2.2.2 Prophet Model

The prophet model introduced by Facebook is the most popular model for time series forecasting in financial markets. This model has special ability of forecasting daily data with weekly and yearly seasonality with holiday effects. This model is effective in handling datasets with strong seasonal patterns, multiple seasonality, and missing values ([Hyndman and Athanasopoulos, 2018](#)).

The prophet model is defined as:

$$y_t = g(t) + s(t) + h(t) + \epsilon_t,$$

where $g(t)$ denotes a piecewise-linear trend (growth term), $s(t)$ denotes the seasonal patterns, $h(t)$ captures the holiday effects, and ϵ_t is a white noise error term.

Prophet model addresses the forecasting task by treating it as a curve-fitting problem, modeling distinct components of the time series independently. The growth component $g(t)$ captures the overall trend in the data and uses a piecewise-linear structure, allowing the model to adapt to changes in the trend rate over time. Seasonality is modeled using a Fourier series with user-defined periods (e.g., 365 for yearly, 7 for weekly), effectively capturing recurring patterns. Holiday effects are introduced as discrete adjustments to the forecast at specific time points and are typically represented using dummy variables.

2.2.3 ARIMA (p, d, q) Model

ARIMA (p, d, q) model is a commonly used linear time series forecasting model. Following notations in [Hyndman and Athanasopoulos \(2018\)](#) an ARIMA (p, d, q) model is given by

$$\phi(B)(1 - B)^d y_t = \theta(B)\epsilon_t$$

where $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p$ and $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \dots + \theta_q B^q$. B is backward shift operator defined as $B^k y_t = y_{t-k}$, $k \in \mathbb{N}$, $\phi(B)$ is a p^{th} order polynomial in B , $\theta(B)$ is a q^{th} order polynomial in B and $\epsilon_t \sim WN(0, \sigma_\epsilon^2)$.

2.2.4 NNAR(p, k) Model

The feed-forward neural network is a popular way to approximate a given multivariate nonlinear function. The network consists of an input variable(s), an output variable, and a hidden layer(s), which join the input variable(s) and output variable. The hidden layer(s) of the neural network is created with units called neurons. Neurons are capable of identifying new features from the data. With a sufficient number of neurons in the hidden layer(s), a neural network can approximate any continuous function and translate inputs to desired output signals. NNAR is a popular neural network model with one hidden layer specifically designed to handle time series data which was proposed by [Hyndman and Athanasopoulos \(2018\)](#). NNAR model is denoted by NNAR(p, k) where p is the number of non-seasonal lagged observations and k denotes the number of neurons in the hidden layer. In the NNAR model, the inputs into each hidden neuron are linearly combined assigning weights to inputs given by

$$z_j = b_j + \sum_{i=1}^p w_{i,j} x_i \quad (2.1)$$

where z_j is the input to the j^{th} node in a hidden layer ($j = 1, \dots, k$), $w_{i,j}$ are the weights assigned for the i^{th} input for the j^{th} hidden node, and x_i is the i^{th} input variable. In the hidden layer, z_j is adjusted using a non-linear functions such as sigmoid function.

$$s(z_j) = \frac{1}{1 + e^{-z}} \quad (2.2)$$

The function `nnetar` from the R Package `forecast` is used to fit an $NNAR(p, k)$ model, where p is automatically selected if not specified. For non-seasonal time series, the default value of p is the optimal number of lags (according to the corrected Akaike Information Criterion (AICC)) for a linear $AR(p)$ model. Unlike other feed-forward neural networks, here the number of neurons in the hidden layer is determined as $k = (p + 1)/2$ (rounded to the nearest integer). Moreover, `nnetar` package runs 20 iterations and takes an average to give more accurate forecasts.

2.2.5 $NNVol(p, k)$ Model

In order to forecast volatility at time t , the lagged values of observed volatility $(V_{t-1}, V_{t-2}, \dots, V_{t-p})$ can be used as inputs to fit an NNAR model, with the forecast of volatility (V_t) as the output. This is called the neuro volatility model, denoted by $NNVol(p, k)$, where p is the lagged values of observed volatility and k is the neurons in the hidden layer (see Figure 2.1). The input

$z_j, j = 1, \dots, k$ to the j^{th} node in the hidden layer is

$$z_j = b_j + \sum_{i=1}^p w_{i,j} V_{t-i}$$

where V_{t-i} denotes the i^{th} input volatility at $(t-i)^{th}$ time point and $w_{i,j}$ denotes the weights assigned to the i^{th} input at j^{th} node in the hidden layer. In the hidden layer, z_j is adjusted using a non-linear activation functions such as sigmoid (as in equation 2.3) or hyperbolic tangent function.

$$s(z_j) = \frac{1}{1 + e^{-z}} \quad (2.3)$$

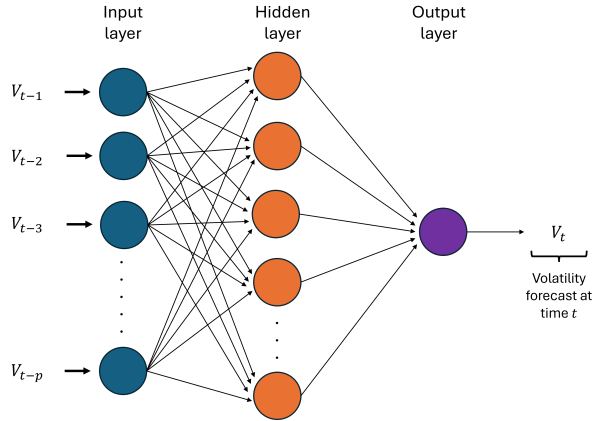


Figure 2.1: $NNVol(p, k)$ model

The NNVol model differs from the NNAR model in that the NNAR model forecasts the one-step-ahead value of the series, while the NNVol model forecasts the volatility of the series.

2.2.6 Recurrent Neural Network (RNN) Model

Recurrent Neural Networks (RNNs) are a class of neural networks designed for processing sequential data by maintaining a hidden state that captures information from previous time steps. At each time step t , the hidden state h_t is updated based on the current input x_t and the previous hidden state h_{t-1} .

$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (2.4)$$

$$y_t = W_{hy}h_t + b_y \quad (2.5)$$

where:

- t is the current time step in the sequence.
- $x_t \in \mathbb{R}^{n_x}$ is the input vector at time t , where n_x is the input dimension.
- $h_t \in \mathbb{R}^{n_h}$ is the hidden state vector at time t , where n_h is the number of hidden units.
- $y_t \in \mathbb{R}^{n_y}$ is the output vector at time t , where n_y is the output dimension.
- $W_{xh} \in \mathbb{R}^{n_h \times n_x}$ is the input-to-hidden weight matrix.
- $W_{hh} \in \mathbb{R}^{n_h \times n_h}$ is the hidden-to-hidden weight matrix.
- $W_{hy} \in \mathbb{R}^{n_y \times n_h}$ is the hidden-to-output weight matrix.
- $b_h \in \mathbb{R}^{n_h}$ is the bias vector for the hidden layer.

- $b_y \in \mathbb{R}^{n_y}$ is the bias vector for the output layer.
 - $\tanh(\cdot)$ is the hyperbolic tangent activation function, applied element-wise.
- Also, rectified linear unit (ReLU) can be used as an alternative.

Equation (2.4) defines how the hidden state h_t is updated based on the current input x_t and the previous hidden state h_{t-1} . The nonlinear $\tanh$ introduces the ability to model complex temporal patterns. Equation (2.5) maps the hidden state to the output space. However, RNNs suffer from vanishing and exploding gradients, which makes learning long-term dependencies challenging. Hoque et al. (2024b) have used RNN to obtain point forecasts of volatility of technology stocks and stock indices. Further Jayanetti et al. (2025) have used several deep learning models including RNN to predict volatility of renewable energy stocks.

2.2.7 Long Short-Term Memory (LSTM) Model

LSTM networks are a type of RNN specifically designed to overcome the vanishing gradient problem. They introduce memory cells and gating mechanisms (input, forget, and output gates) that regulate the flow of information, enabling the model to retain long-term dependencies effectively.

- Forget Gate: Identifies which information to remove from the cell and which information to preserve.

- Input Gate: Updates the cell with new information
- Output Gate helps determine the next hidden state.

These gates help LSTMs remember and forget information in long sequences, making them efficient for time series tasks like volatility forecasting (Hoque et al., 2024b) and stock index forecasting (Dip Das et al., 2024). The operations within a single LSTM cell at time step t are as follows¹:

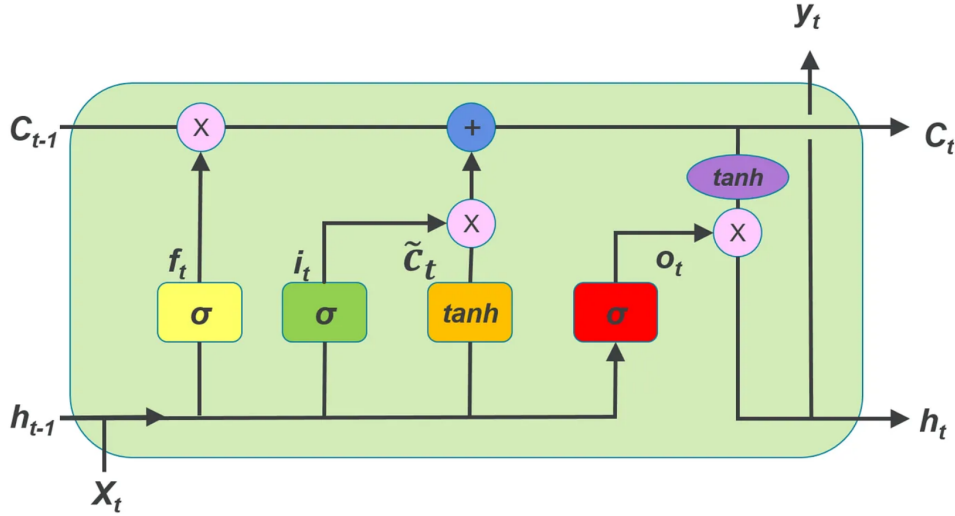


Figure 2.2: LSTM architecture

The LSTM functional units are described as follows:

¹<https://medium.com/@divyanshu132/lstm-and-its-equations-5ee9246d04af>

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

where

- x_t is the input vector at time t .
- h_{t-1} is the hidden state from the previous time step.
- c_{t-1} and c_t are the cell states at time $t - 1$ and t , respectively.
- $\sigma(\cdot)$ is the sigmoid activation function.
- $\tanh(\cdot)$ is the hyperbolic tangent function.
- $\odot$ and $\oplus$ denotes element-wise multiplication and addition respectively.
- W , U , and b are learnable weights and biases.

Bidirectional LSTM (BiLSTM) networks extend the LSTM by processing the input sequence in both forward and backward directions using two separate

LSTM layers. The outputs from both directions are then, concatenated, providing the model with both past and future context at each time step, which is beneficial in tasks where context from both directions improves performance.

2.2.8 Gated Recurrent Unit (GRU) Model

GRUs are a simplified variant of LSTMs that combine the forget and input gates into a single update gate and use a reset gate to control the incorporation of new input².

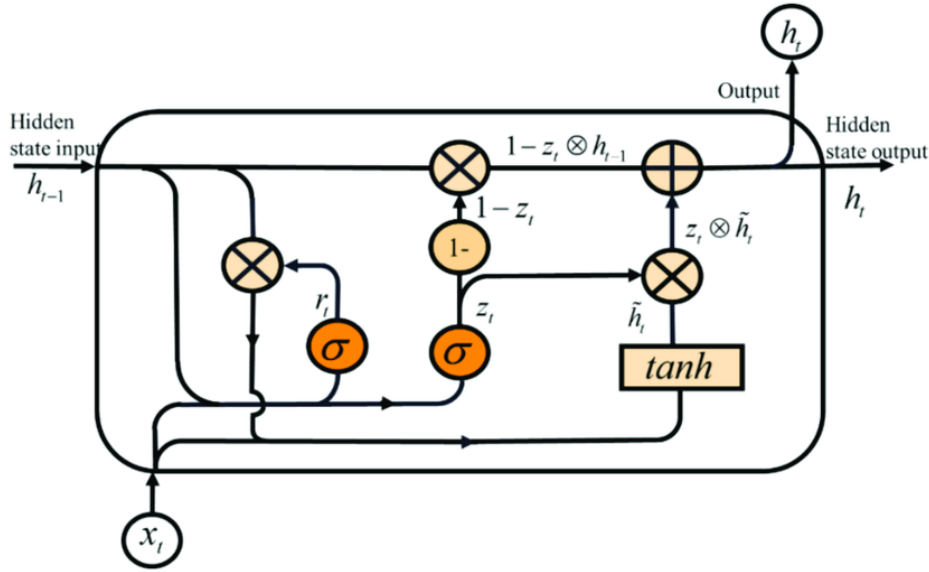


Figure 2.3: GRU architecture

²https://www.researchgate.net/figure/GRU-structure-diagram_fig2_355705028

The following equations can explain GRU:

$$r_t = \sigma(W_r x_t + U_r h_{t-1})$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1})$$

$$\tilde{h}_t = \tanh(r_t \odot U h_{t-1} + W x_t)$$

$$h_t = (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1}$$

where

- x_t is the input vector at time t
- h_{t-1} is the previous hidden state
- z_t is the update gate controlling how much past information is preserved
- r_t is the reset gate determining how to combine new input with past memory
- $\tilde{h}_t$ is the candidate activation (new memory content)
- h_t is the updated hidden state
- $\odot$ and $\oplus$ are element-wise multiplication and addition respectively.
- $\sigma(\cdot)$ is the sigmoid activation function
- $\tanh(\cdot)$ is the hyperbolic tangent function

GRUs require fewer parameters and are computationally more efficient than LSTMs, while still capturing dependencies over long sequences effectively. GRU was used to forecast stock indices in [Dip Das et al. \(2024\)](#) and to forecast volatility in [Jayanetti et al. \(2025\)](#).

Similarly to BiLSTM, Bidirectional GRU (BiGRU) uses two GRU layers that process the sequence in forward and backward directions. It combines the efficiency of GRUs with the contextual advantages of bidirectional processing, making it suitable for many sequence-modeling tasks with reduced computational cost compared to BiLSTM.

2.2.9 Model Risk

[Danielsson et al. \(2016\)](#) first introduced the model risk ratio idea and applied the methodology to market risk measures, value at risk (VaR) and expected shortfall (ES), which depend on volatility based on the forecast of conditional variance. If there are N number of different models to forecast VaR or ES, each model results in a unique forecast given as

$$\{\text{Risk}_{t+1}^n\}_{n=1}^N. \quad (2.6)$$

The model risk is defined as the ratio between the highest VarR/ES forecasts to the lowest VarR/ES forecasts

$$\text{Model risk} = \text{Risk Ratio}_{t+1} = \frac{\max\{\text{Risk}_{t+1}^n\}_{n=1}^N}{\min\{\text{Risk}_{t+1}^n\}_{n=1}^N}. \quad (2.7)$$

When the model risk ratio value is close to 1, it indicates that all the models are forecasting in the same range resulting stable financial system. In addition, other risk metrics such as root mean squared error (RMSE) and mean absolute errors (MAE) can also be used to construct model risk. [Hoque et al. \(2024b\)](#) introduced neuro-model risk forecasts based on direct volatility forecasts, which allow more appropriate nonlinear nonstationary neuro-volatility forecasting models. The work investigates the stability of the forecasts of different models considering the proposed neuro-model risk.

2.3 Univariate Volatility Forecasting

This study considers the daily adjusted closing price of six cryptocurrencies (Bitcoin USD - BTC, BNB USD - BNB, Ethereum USD - ETH, XRP USD - XRP, Dogecoin USD - DOGE, Cardano USD - ADA) from 2018 to 2023 and computes the observed volatility $V_t = |r_t - \bar{r}| / \hat{\rho}$ where $\bar{r}$ is the mean of log returns and sign correlation between $r_t - \bar{r}$ and $\text{sign}(r_t - \bar{r})$ denoted as $\hat{\rho} = \text{corr}(r_t - \bar{r}, \text{sign}(r_t - \bar{r}))$. Then, obtain one step ahead volatility forecast using time series cross-validation (TSCV). TSCV considers the first 150 observed volatilities to forecast the 151st volatility and the first 151 observations to forecast the 152nd volatility (one step ahead volatility forecast) until forecasting the last volatility. In this study, we apply 4 models; Prophet, ETS, NNAR, and LSTM models followed by computing model risk for comparison.

Table 2.1 reports the average RMSE and MAE of volatility forecasts of six cryptocurrencies and their model risk ratios. When comparing the accuracy of the volatility forecasts, the NNAR model is superior to the other models considered. Considering RMSE and MAE values, it is evident that the Prophet and ETS models are not performing well for the selected cryptocurrencies during the study period, and this confirms the findings of [Hoque et al. \(2024b\)](#). According to [Hyndman and Athanasopoulos \(2018\)](#), the ETS and prophet models are more suitable for regular seasonality, while the prophet model has the special ability to capture the moving holiday effects. However, since the pandemic period shows fluctuations diverting from regular seasonality and holidays, in this study the prophet model marks a lower accuracy level confirming the above.

Table 2.1: Forecast accuracy of six cryptocurrencies and their model risk

Model	BTC		ETH		BNB		XRP		DOGE		ADA	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
ETS	0.50	0.37	0.53	0.4	0.64	0.45	0.62	0.43	0.74	0.50	0.61	0.42
Prophet	0.51	0.35	0.62	0.44	0.68	0.42	0.67	0.46	0.79	0.54	0.65	0.45
NNAR	0.19	0.35	0.08	0.21	0.22	0.36	0.20	0.33	0.25	0.39	0.21	0.34
LSTM	0.15	0.31	0.10	0.24	0.16	0.30	0.17	0.29	0.22	0.36	0.18	0.31
Model risk	3.4	1.19	6.2	2.0	4.25	1.5	3.94	1.58	3.59	1.50	3.61	1.45

Deep learning models are also used to obtain next-day volatility forecasts for six cryptocurrencies choosing different model configurations. First, the NNAR model is fitted to the observed volatility data for each cryptocurrency. The best-fitted model for BTC is identified as NNAR($p = 21$, $k = 11$), which uses the previous 21 volatility observations ($V_{t-1}, \dots, V_{t-21}$) as input to the neural network, with 11 neurons in the hidden layer. Similarly, for ETH,

the optimal model is NNAR($p = 12, k = 6$), indicating that the previous 12 volatility observations are used as input, with 6 neurons in the hidden layer. For the BNB, XRP, DOGE and ADA, NNAR($p = 29, k = 15$), NNAR($p = 14, k = 8$), NNAR($p = 28, k = 14$), NNAR($p = 15, k = 8$) models are fitted, respectively. Each NNAR model is selected based on its ability to best capture the underlying volatility patterns for the respective stock. The one-step-ahead volatility forecasts obtained from the NNAR models are as follows: the forecast for BTC is 0.0198, for ETH is 0.0350, for BNB is 0.0595, for XRP is 0.0393 ,for DOGE is 0.0513, and for ADA is 0.0332. Note that NNAR always fits a neural network with a single hidden layer. For comparison purposes, single-layer neural networks, two-layer neural networks, and three-layer neural networks are also used for RNN, LSTM, GRU, BiLSTM, and BiGRU. For each of the networks, the same number of neurons is used in the hidden layer for each stock. Furthermore, the number of neurons in the hidden layers increases twice and is used with the above three layers to forecast volatility for the next day.

Table 2.2: One step ahead volatility forecast of BTC using DL models

Model	1 Layer (k=11)	2 Layer (k=22)	3 Layer (k=33)	1 Layer (k=22)	2 Layer (k=44)	3 Layer (k=66)	Model Risk
RNN	0.0352	0.0414	0.0331	0.0430	0.0331	0.0368	1.3014
LSTM	0.0321	0.0273	0.0337	0.0368	0.0314	0.0362	1.3518
GRU	0.0396	0.0326	0.0311	0.0329	0.0326	0.0353	1.2721
BiLSTM	0.0389	0.0328	0.0294	0.0373	0.0382	0.0342	1.3221
BiGRU	0.0392	0.0338	0.0277	0.0360	0.0398	0.0391	1.4334

The model risk indicates the stability of DL models in forecasting the volatility of cryptocurrencies (Table 2.8). Under the RNN model, the highest

Table 2.3: One step ahead volatility forecast of ETH using DL models

Model	1 Layer (k=6)	2 Layer (k=12)	3 Layer (k=18)	1 Layer (k=12)	2 Layer (k=24)	3 Layer (k=36)	Model Risk
RNN	0.0439	0.0454	0.0568	0.0375	0.0664	0.0426	1.7715
LSTM	0.0516	0.0384	0.0429	0.0427	0.0486	0.0522	1.3599
GRU	0.0465	0.0441	0.0452	0.0449	0.0493	0.0460	1.1165
BiLSTM	0.0504	0.0497	0.0485	0.0455	0.0517	0.0493	1.1376
BiGRU	0.0455	0.0475	0.0358	0.0558	0.0522	0.0500	1.5580

Table 2.4: One step ahead volatility forecast of BNB using DL models

Model	1 Layer (k=15)	2 Layer (k=30)	3 Layer (k=45)	1 Layer (k=30)	2 Layer (k=60)	3 Layer (k=90)	Model Risk
RNN	0.0927	0.0430	0.0836	0.0689	0.0437	0.0766	2.1561
LSTM	0.0513	0.0582	0.0393	0.0463	0.0577	0.0530	1.4821
GRU	0.0536	0.0519	0.0514	0.0564	0.0518	0.0355	1.5892
BiLSTM	0.0537	0.0675	0.0478	0.0547	0.0572	0.0534	1.4110
BiGRU	0.0563	0.0538	0.0379	0.0413	0.0533	0.0489	1.4852

Table 2.5: One step ahead volatility forecast of XRP using DL models

Model	1 Layer (k=8)	2 Layer (k=16)	3 Layer (k=24)	1 Layer (k=16)	2 Layer (k=32)	3 Layer (k=48)	Model Risk
RNN	0.0513	0.0793	0.0430	0.0452	0.0610	0.0377	2.1060
LSTM	0.0548	0.0598	0.0522	0.0593	0.0591	0.0574	1.1460
GRU	0.0510	0.0669	0.0509	0.0531	0.0648	0.0542	1.3161
BiLSTM	0.0473	0.0604	0.0561	0.0526	0.0578	0.0486	1.2768
BiGRU	0.0631	0.0521	0.0604	0.0577	0.0570	0.0507	1.2435

Table 2.6: One step ahead volatility forecast of DOGE using DL models

Model	1 Layer (k=14)	2 Layer (k=28)	3 Layer (k=42)	1 Layer (k=28)	2 Layer (k=56)	3 Layer (k=48)	Model Risk
RNN	0.0399	0.0875	0.1525	0.0573	0.0898	0.0598	3.8206
LSTM	0.0717	0.0821	0.0799	0.0519	0.0843	0.0847	1.6331
GRU	0.0683	0.0983	0.0884	0.0587	0.0716	0.0730	1.6746
BiLSTM	0.0690	0.0928	0.0614	0.0864	0.0843	0.0661	1.5112
BiGRU	0.0799	0.1032	0.0683	0.0848	0.0714	0.1058	1.5488

Table 2.7: One step ahead volatility forecast of ADA using DL models

Model	1 Layer (k=8)	2 Layer (k=16)	3 Layer (k=24)	1 Layer (k=16)	2 Layer (k=32)	3 Layer (k=48)	Model Risk
RNN	0.0510	0.0640	0.0434	0.0633	0.0663	0.0419	1.5810
LSTM	0.0562	0.0559	0.0547	0.0565	0.0482	0.0543	1.1723
GRU	0.0611	0.0532	0.0444	0.0646	0.0603	0.0533	1.4531
BiLSTM	0.0572	0.0610	0.0600	0.0542	0.0560	0.0602	1.1252
BiGRU	0.0592	0.0580	0.0562	0.0486	0.0587	0.0584	1.2184

model risk is obtained by DOGE (3.8206), indicating substantial uncertainty in the volatility of the forecast (greater than 2), while BiLSTM (1.5112) and BiGRU (1.5488) improve the reliability of the forecast. ADA has the lowest model risk across all models, while the BiGRU (1.2184) and BiLSTM (1.1252) are also well-rated. The RNN (1.5810) for ADA indicates high risk. For the BTC, BiGRU gives higher model risk while for all other cryptocurrencies RNN is associated with significant risk. Except for BTC, BiLSTM and BiGRU considerably mitigate the uncertainty and become favorable for risk-sensitive predictions. These findings indicates that BiLSTM and BiGRU are the most reliable models for cryptocurrency forecasting.

Table 2.8: Model risk of five deep learning models

Model	Cryptocurrency					
	BTC	ETH	BNB	XRP	DOGE	ADA
RNN	1.3014	1.7715	2.1561	2.1060	3.8206	1.5810
LSTM	1.3518	1.3599	1.4821	1.1460	1.6331	1.1723
GRU	1.2721	1.1165	1.5892	1.3161	1.6746	1.4531
BiLSTM	1.3221	1.1376	1.4110	1.2768	1.5112	1.1252
BiGRU	1.4334	1.5580	1.4852	1.2435	1.5488	1.2184

Bi-directional models (Bi-LSTM and Bi-GRU) are often observed to perform better than their uni-directional counterparts in volatility forecasting. A uni-

directional model (LSTM or GRU) processes the sequence from past to present and uses this history to predict the next day volatility. A bi-directional model, on the other hand, processes the sequence both forward and backward. This means that at each time step, the model can consider both previous and subsequent volatilities in the sequence. This helps the model learn richer representations, especially when the full sequence is known during training (See Section 2.2.7 and Section 2.2.8 for more details).

2.4 Multivariate Volatility Forecasting

2.4.1 VAR(p) Model

The Vector Autoregressive (VAR) model is a fundamental multivariate linear time series model that extends the univariate AR model to handle multiple time-dependent variables. It is widely used in econometrics and finance to capture the dynamic relationship among several time series.

A VAR(p) model describes a k -dimensional time series $\mathbf{Y}_t = (Y_{1t}, Y_{2t}, \dots, Y_{kt})'$ as a linear function of its past values. The general form of the VAR(p) model is:

$$\mathbf{Y}_t = \mathbf{c} + A_1 \mathbf{Y}_{t-1} + A_2 \mathbf{Y}_{t-2} + \dots + A_p \mathbf{Y}_{t-p} + \mathbf{u}_t \quad (2.8)$$

where $\mathbf{Y}_t$ is a $k \times 1$ vector at each time t , $\mathbf{c}$ is a $k \times 1$ intercept vector, A_i are $k \times k$ coefficient matrices, p is the order of the VAR model, indicating the

number of lag values included and $\mathbf{u}_t$ is a $k \times 1$ error vector (white noise) with mean zero and covariance matrix Σ_u .

In a VAR(p) model, each time series $Y_{it}, i = 1, \dots, k$ depends on its own past values $(Y_{i,t-1}, \dots, Y_{i,t-p})$ and the past values of the other series $(Y_{j,t-1}, \dots, Y_{j,t-p})$. The appropriate lag order p can be determined using information criteria such as Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC).

In improving multivariate forecasting, [Wickramasuriya et al. \(2019\)](#) first introduces a novel method called MinT (Minimum Trace) to reconcile forecasts in hierarchical and grouped time series, then, shows that using shrinkage estimator with MinT for the covariance matrix outperform sample covariance matrix and other alternative structured and unstructured matrices. MinT adjusts forecasts to enforce the coherence while minimizing the total forecast error variance. MinT(Shrink) method was applied in forecasting big data consisting of 555 time series of domestic tourism. In [Saumyamala et al. \(2025\)](#), RMT filtering method was used to denoise the covariance matrix of 500 stock price series in S&P500 index and applied in the context of portfolio optimization. Forecast linear augmented projection (FLAP) is a forecast error variance reduction methodology using subspace projection. This method emphasizes enhanced prediction with higher forecast accuracy. In [Yang et al. \(2024\)](#) recently proposed FLAP is used to minimize the forecast error variance of any unbiased multivariate forecast while preserving its unbiasedness. FLAP

theoretically improves forecast accuracy as more components are added to the projection. Practical implementations using FLAP with shrinkage estimator of the covariance matrix and Principal Component Analysis (PCA) have demonstrated significant improvement in forecast accuracy.

FLAP method [Yang et al. \(2024\)](#) has demonstrated reduced forecast error variance under normally distributed errors. However, [Thavaneswaran et al. \(2020\)](#) shows that asset returns often exhibit high skewness and kurtosis, deviating from normality. In order to test FLAP's robustness under heavy-tailed distributions, this paper conducts a simulation study using multivariate t and Laplace distributions alongside normal distributions. FLAP has been primarily applied to linear models like ARIMA and dynamic factor models, but has not yet been explored with non-linear models such as neural networks. In this study, we extend FLAP to forecast asset prices and volatility using both ARIMA and NNAR models, and assess its effectiveness in reducing out-of-sample RMSE.

2.4.2 Forecast Linear Augmented Projection (FLAP)

FLAP works by augmenting data with linear combinations, forecasting these components, and then, projecting the augmented forecasts. The approach begins by creating new series of components as linear combinations of the original multivariate time series. Forecasts are then, produced for both the original series and the derived component series. Finally, the complete set of

both original series forecasts and component forecasts is projected onto a linear subspace that enforces the constraints imposed by the combination weights. The forecast accuracy is shown to be improved by combining direct original series and indirect component forecasts. Following the notation and method outlined in [Yang et al. \(2024\)](#), the projected h -step-ahead FLAP forecast $\hat{z}_{t+h}$ of the multivariate time series is obtained in **Algorithm 1** through several steps. The weight matrix Φ is obtained by using PCA and the number of principal components is selected based on the amount of variance explained by each component. PCA is a dimension reduction technique that transforms correlated variables into uncorrelated principal components, capturing maximum variance in the first components. Each subsequent component retains the next highest variance while remaining orthogonal to the previous ones. The forecast error covariance matrix W_h of the component-augmented projected h -step-ahead forecasts $\hat{z}_{t+h}$ is unknown, making its estimation crucial, as it directly impacts the accuracy of FLAP forecasts. In [Yang et al. \(2024\)](#), it is shown that the conventional estimator

$$\hat{W}_h = \frac{1}{n - h - 1} \sum_{t=h+1}^n \hat{e}_{t,h} \hat{e}_{t,h}^T$$

is not a good estimator when $m + h \approx n - h$ or $m + h > n - h$ where $\hat{e}_{t,h} = z_t - \hat{z}_{t|t-h}$ is the in-sample h -step-ahead base forecast error with forecast origin $t-h$ (in-sample residual). Moreover, [Yang et al. \(2024\)](#) applied covariance shrinkage method (`corpcor::cov.shrink` R function [Schafer et al. \(2021\)](#)) to

guarantee the positive definiteness of the forecast error covariance matrix.

In Algorithm 1, direct forecasts can be obtained by any univariate or multivariate forecasting method. In Yang et al. (2024), direct forecasts are obtained from ARIMA and dynamic factor model. In this paper, we use the linear VAR(p) model as the baseline, which captures the linear relationships between all time series but requires using the same lag values for all series when forecasting the future multiple time series. We will compare the linear ARIMA model for forecasting each univariate time series with the ARIMA+FLAP model, which incorporate the relationships between all linear ARIMA forecasts. The advantage of the ARIMA+FLAP model is that it allows the lag value for each ARIMA model to differ across the time series. Moreover, we will compare the non-linear NNAR model for forecasting each univariate time series with the NNAR+FLAP model, which incorporate the relationships between all non-linear NNAR forecasts.

In this thesis, we use h -step-ahead long-term forecasting approach with the expanding window model validation method. At each step, the training set expands by including one observation at a time and forecast the next h time steps. This repeats until the last observation of the test set. The mean forecast in expanding window validation is calculated as the average of the predicted values over a specific forecast horizon. Mean forecast is calculated by $\frac{1}{h} \sum_{i=1}^h \hat{y}_{n-h+i}$ where h is the forecast horizon (the number of steps ahead you are predicting), $\hat{y}_{n+i-h}$ is the predicted value at time $n + i - h$ and n is

the last observed time point in the test window.

Algorithm 1 FLAP Forecasting Algorithm

Require: Data: Observed series $y_t \in \mathbb{R}^m$, forecast horizon h , and base forecasting method.

- 1: Obtain component weight matrix Φ (either Principal components from PCA); $\Phi \in \mathbb{R}^{n_c \times m}$ (m - no.of series, n_c - no.of principal components)
- 2: Normalize Φ matrix $\frac{\phi_i}{\sqrt{\sum \phi_{ij}^2}}$ where ϕ_i is the i^{th} row vector and ϕ_{ij} is the i^{th} row j^{th} column weight.
- 3: Compute the components $c_t \in \mathbb{R}^{n_c}$ as linear combinations of y_t : $c_t = \Phi y_t$
- 4: Construct the combined matrix $z_t = [y_t^T \ c_t^T]^T \in \mathbb{R}^{m+n_c}$
- 5: Define the constraint matrix $C = [-\Phi \ I_{n_c}] \in \mathbb{R}^{n_c \times (m+n_c)}$
- 6: Ensure the constraint $Cz_t = c_t - \Phi y_t = 0$ holds.
- 7: Generate h -step-ahead base forecasts $\hat{z}_{t+h}$ for z_t using base forecasting models (either ARIMA or NNAR).
- 8: Obtain an estimate of the h -step-ahead forecast error covariance matrix $W_h = \text{Var}(z_{t+h} - \hat{z}_{t+h})$ (use covariance shrinkage method with cov.shrink function in `corpcor` R package s.t forecast error variance matrix is guaranteed to be positive definite)
- 9: Compute the projection matrix $M \in \mathbb{R}^{(m+n_c) \times (m+n_c)}$:

$$M = I_{m+n_c} - W_h C^T (C W_h C^T)^{-1} C$$

- 10: Obtain the projected forecasts $\tilde{z}_{t+h} = M \hat{z}_{t+h}$.
 - 11: Return the projected forecasts $\tilde{z}_{t+h}$.
-

2.4.3 Simulation Study

This section presents a simulation study to evaluate FLAP's effectiveness in reducing the multivariate forecast error variance. Using daily return data from 2017-01-01 to 2024-12-31 for 10 major assets (Apple Inc. - AAPL, NVIDIA

Corporation - NVDA, Microsoft Corporation - MSFT, JPMorgan Chase & Co. - JPM, Amazon.com, Inc. - AMZN, Tesla Inc. - TSLA, NIKE Inc. - NKE, Alphabet Inc. - GOOG, Meta Platforms Inc. - META, and Walmart Inc. - WMT), a VAR(1) model was fitted, with lag order $p=1$ selected via the BIC criterion. Simulated data were then, generated from the VAR(1) model, incorporating errors drawn from multivariate normal, t , and Laplace distributions based on the model's residual variance-covariance matrix.

First, the simulated 10 series with normal errors are modeled using VAR, ARIMA and NNAR models to obtain 12-step-ahead forecasts. Then, a PCA analysis was performed on the matrix of 10 simulated series. The resulting weight matrix Φ consists of principal component loadings for the 10 series from the PCA analysis.

The forecasts for the 10 component series were generated using VAR, ARIMA, and NNAR models. Next, the shrinkage estimator for the error covariance matrix (W) was computed using the `corpcor::cov.shrink` function in R (Schafer et al., 2021), and a projection matrix M was obtained. For ARIMA and NNAR forecasts, the FLAP method was then, applied by projecting the combined forecasts of original series and component series onto the subspace defined by the PCA components, achieved by multiplying the matrix M with the original ARIMA or NNAR forecasts. This process was repeated for different numbers of components, varying from 1 to 10, with the RMSE of the FLAP forecasts calculated for each number of components considered.

The accuracy of the VAR, ARIMA, ARIMA+FLAP forecasts is provided in Table 2.9 and that of VAR, NNAR and NNAR+FLAP is summarized in Table 2.10. It can be seen from Table 2.9 It was observed that RMSEs of the ARIMA+FLAP forecasts are lower than the RMSE of the original ARIMA forecasts for most of the components. Similarly, in Table 2.10, the minimum RMSE of the FLAP adjusted NNAR forecasts is less than the original NNAR forecasts. But in both situations, the VAR forecasts have lower RMSE values.

Table 2.9: Accuracy of 12-step-ahead VAR, ARIMA and ARIMA+FLAP forecasts from the simulated VAR(1) data with normally distributed errors

Stock	VAR RMSE	ARIMA model	ARIMA RMSE	RMSE of FLAP adjusted forecasts									
				No. of components									
				1	2	3	4	5	6	7	8	9	10
1	0.01394	ARIMA(0,0,0)	0.01409	0.01417	0.01453	0.01451	0.01450	0.01450	0.01449	0.01447	0.01446	0.01449	0.01448
2	0.04473	ARIMA(0,0,0)	0.04489	0.04488	0.04491	0.04507	0.04507	0.04508	0.04508	0.04507	0.04505	0.04504	0.04505
3	0.01879	ARIMA(0,0,0)	0.01895	0.01896	0.01906	0.01902	0.01903	0.01903	0.01903	0.01904	0.01904	0.01904	0.01903
4	0.01892	ARIMA(0,0,0)	0.01892	0.01889	0.01869	0.01864	0.01863	0.01864	0.01865	0.01865	0.01866	0.01867	0.01867
5	0.02651	ARIMA(0,0,0)	0.02656	0.02656	0.02693	0.02689	0.02693	0.02692	0.02692	0.02689	0.02687	0.02686	0.02685
6	0.03682	ARIMA(2,1,0)	0.03810	0.03679	0.03688	0.03677	0.03663	0.03665	0.03666	0.03667	0.03665	0.03665	0.03667
7	0.02403	ARIMA(0,0,0)	0.02408	0.02409	0.02386	0.02377	0.02377	0.02377	0.02377	0.02377	0.02378	0.02379	0.02379
8	0.01685	ARIMA(0,0,0)	0.01687	0.01685	0.01685	0.01685	0.01686	0.01685	0.01685	0.01685	0.01685	0.01686	0.01685
9	0.03103	ARIMA(0,0,0)	0.03119	0.03120	0.03128	0.03120	0.03121	0.03120	0.03120	0.03119	0.03118	0.03118	0.0311
10	0.01689	ARIMA(3,0,0)	0.01697	0.0169	0.01709	0.01707	0.01706	0.01706	0.01706	0.01707	0.01704	0.01704	0.01704

Table 2.10: Accuracy of 12-step-ahead VAR, NNAR and NNAR+FLAP forecasts from the simulated VAR(1) data with normally distributed errors

Series	VAR RMSE	NNAR model	NNAR RMSE	RMSE of FLAP adjusted forecasts									
				No. of components									
				1	2	3	4	5	6	7	8	9	10
1	0.01394	NNAR (1,0,1)	0.01404	0.01407	0.01407	0.01406	0.01408	0.01408	0.01406	0.01418	0.01423	0.01434	0.01436
2	0.04473	NNAR (1,0,1)	0.04496	0.04501	0.04504	0.04509	0.04509	0.04509	0.04506	0.04511	0.04517	0.04515	0.04514
3	0.01879	NNAR (1,0,1)	0.01898	0.01899	0.01900	0.01898	0.01898	0.01899	0.01898	0.01897	0.01897	0.01898	0.01897
4	0.01892	NNAR (1,0,1)	0.01866	0.01866	0.01867	0.01862	0.01860	0.01860	0.01856	0.01857	0.01857	0.01857	0.01858
5	0.02651	NNAR (1,0,1)	0.02667	0.02667	0.02667	0.02664	0.02661	0.02661	0.02661	0.02667	0.02666	0.02666	0.02666
6	0.03682	NNAR (1,0,1)	0.03683	0.03685	0.03683	0.03681	0.03679	0.03679	0.03678	0.03678	0.03679	0.03679	0.03679
7	0.02403	NNAR (1,0,1)	0.02404	0.02404	0.02405	0.02399	0.02400	0.02397	0.02396	0.02397	0.02404	0.02404	0.02404
8	0.01685	NNAR (1,0,1)	0.01689	0.01689	0.01689	0.01688	0.01688	0.01688	0.01688	0.01687	0.01688	0.01688	0.01688
9	0.03103	NNAR (1,0,1)	0.03117	0.03118	0.03120	0.03115	0.03117	0.03116	0.03115	0.03114	0.03113	0.03114	0.03114
10	0.01689	NNAR (3,0,2)	0.01704	0.01703	0.01704	0.01705	0.01707	0.01708	0.01707	0.01702	0.01706	0.01706	0.01705

Stock returns usually follow t -distribution; therefore, we further generate data from VAR(1) model with innovations simulated from multivariate t distribution with degrees of freedom $\nu = 3$. Unlike the normal distribution

with well-defined mean and covariance matrix, t -distribution's mean exists only if $\nu > 1$ and its covariance matrix exists only if $\nu > 2$. VAR, ARIMA and FLAP adjusted ARIMA forecast accuracies are given in Table 2.11 and VAR, NNAR and FLAP adjusted NNAR forecast accuracies are given in Table 2.12. The minimum RMSE among all the RMSE values from the FLAP forecasts for different numbers of components is highlighted in red. For t -distributed data, ARIMA+FLAP forecasts have lower RMSE than the original ARIMA forecasts and VAR forecasts. Similarly, FLAP+NNAR forecasts have lower RMSE than RMSE compared to the original NNAR forecasts and VAR forecasts.

Table 2.11: Accuracy of 12-step-ahead VAR, ARIMA and ARIMA+FLAP forecasts from simulated VAR(1) data with t distributed errors ($\nu = 3$)

Series	VAR RMSE	ARIMA model	ARIMA RMSE	RMSE of FLAP adjusted forecasts									
				No. of principal components									
				1	2	3	4	5	6	7	8	9	10
1	0.01269	ARIMA(0,0,1)	0.01237	0.01236	0.01244	0.01243	0.01243	0.01243	0.01242	0.01243	0.01245	0.01245	0.01248
2	0.03891	ARIMA(0,0,0)	0.03872	0.03874	0.03877	0.03874	0.03873	0.03872	0.03872	0.03872	0.03872	0.03872	0.03871
3	0.02677	ARIMA(0,0,0)	0.02641	0.02636	0.02617	0.02617	0.02620	0.02616	0.02618	0.02620	0.02628	0.02629	0.0261
4	0.02455	ARIMA(0,0,0)	0.02455	0.02454	0.02463	0.02463	0.02460	0.02458	0.02457	0.02457	0.02456	0.02456	0.02456
5	0.03053	ARIMA(0,0,0)	0.03009	0.03013	0.02952	0.02955	0.02960	0.02957	0.02960	0.02963	0.02965	0.02965	0.02966
6	0.04059	ARIMA(0,0,0)	0.04068	0.04045	0.04084	0.04059	0.04062	0.04061	0.04062	0.04063	0.04062	0.04063	0.04059
7	0.02740	ARIMA(0,0,0)	0.02736	0.02737	0.02737	0.02736	0.02738	0.02737	0.02737	0.02737	0.02737	0.02737	0.02738
8	0.02336	ARIMA(0,0,0)	0.02320	0.02318	0.02320	0.02319	0.02320	0.02319	0.02319	0.02321	0.02319	0.02319	0.02319
9	0.03059	ARIMA(0,0,1)	0.03032	0.03035	0.03088	0.03097	0.03095	0.03087	0.03085	0.03084	0.03082	0.03081	0.03081
10	0.02117	ARIMA(0,0,0)	0.02107	0.02107	0.02104	0.02105	0.02107	0.02106	0.02106	0.02106	0.02106	0.02107	0.02107

Table 2.12: Accuracy of 12-step-ahead VAR, NNAR and NNAR+FLAP forecasts from simulated VAR(1) data with t distributed errors($\nu = 3$)

Series	VAR RMSE	NNAR model	NNAR RMSE	RMSE of FLAP adjusted forecasts									
				No. of principal components									
				1	2	3	4	5	6	7	8	9	10
1	0.01269	NNAR (1,0,1)	0.01234	0.01241	0.01297	0.01323	0.01327	0.01383	0.01375	0.01383	0.01381	0.01380	0.01373
2	0.03891	NNAR (1,0,1)	0.03865	0.03867	0.03965	0.04089	0.04086	0.04146	0.04136	0.04141	0.04128	0.04125	0.04123
3	0.02677	NNAR (1,0,1)	0.02596	0.02598	0.02650	0.02663	0.02664	0.02710	0.02704	0.02707	0.02705	0.02704	0.02707
4	0.02455	NNAR (1,0,1)	0.02454	0.02449	0.02418	0.02410	0.02409	0.02403	0.02402	0.024	0.02401	0.02401	0.02401
5	0.03053	NNAR (1,0,1)	0.03009	0.03009	0.03090	0.03097	0.03089	0.03133	0.03129	0.03125	0.03119	0.03120	0.03114
6	0.04059	NNAR (1,0,1)	0.03967	0.03969	0.03999	0.03995	0.03995	0.04005	0.04004	0.04004	0.04001	0.04001	0.04001
7	0.02740	NNAR (1,0,1)	0.02746	0.02745	0.02758	0.02756	0.02749	0.02750	0.02749	0.02749	0.02749	0.02749	0.02750
8	0.02336	NNAR (1,0,1)	0.02318	0.02326	0.02378	0.02400	0.02406	0.02474	0.02466	0.02477	0.02466	0.02464	0.02462
9	0.03059	NNAR (9,0,5)	0.03461	0.03505	0.03708	0.03772	0.03773	0.03898	0.03868	0.03867	0.03836	0.03829	0.03815
10	0.02117	NNAR (1,0,1)	0.02104	0.02102	0.02100	0.02092	0.02085	0.02074	0.02076	0.02077	0.02078	0.02082	0.02083

The multivariate Laplace distribution extends the univariate Laplace dis-

tribution by incorporating a covariance structure and can be expressed as a scale mixture of multivariate normal distributions, where the scaling factor follows an exponential distribution. This distribution has heavy tails across all dimensions, making it particularly useful in applications in finance, where extreme events such as market crashes occur. Therefore, we also generate data from VAR(1) model with Laplace distributed innovations and use VAR, ARIMA and NNAR models to obtain 12-step-ahead forecasts.

Table 2.13: Accuracy of 12-step-ahead VAR,ARIMA and ARIMA+FLAP forecasts from simulated VAR(1) data with Laplace distributed errors

Series	VAR RMSE	ARIMA model	ARIMA RMSE	RMSE of FLAP adjusted forecasts									
				No. of components									
				1	2	3	4	5	6	7	8	9	10
1	0.01760	ARIMA(0,0,0)	0.01760	0.01759	0.01759	0.01759	0.01759	0.01759	0.01761	0.01763	0.01763	0.01758	0.01758
2	0.03541	ARIMA(0,0,0)	0.03546	0.03547	0.03549	0.03549	0.03548	0.03548	0.03543	0.03542	0.03543	0.03543	0.03543
3	0.01624	ARIMA(0,0,0)	0.01619	0.01619	0.01619	0.01619	0.01619	0.01619	0.01616	0.01616	0.01614	0.01612	0.01614
4	0.01433	ARIMA(1,0,2)	0.01457	0.01457	0.01456	0.01457	0.01456	0.01456	0.01446	0.01446	0.01446	0.01445	0.01445
5	0.01821	ARIMA(0,0,0)	0.01804	0.01803	0.01802	0.01803	0.01801	0.01801	0.01807	0.01805	0.01805	0.01802	0.01803
6	0.05047	ARIMA(0,0,0)	0.05041	0.05040	0.05034	0.05035	0.05035	0.05035	0.05041	0.05042	0.05044	0.05044	0.05044
7	0.02035	ARIMA(0,0,0)	0.02032	0.02033	0.02032	0.02032	0.02035	0.02034	0.02027	0.02025	0.02030	0.02030	0.02030
8	0.01685	ARIMA(0,0,0)	0.01712	0.01712	0.01713	0.01714	0.01713	0.01713	0.01713	0.01714	0.01716	0.01716	0.01716
9	0.03172	ARIMA(0,0,0)	0.03156	0.03156	0.03153	0.03153	0.03155	0.03155	0.03157	0.03157	0.03156	0.03156	0.03156
10	0.01607	ARIMA(0,0,2)	0.01604	0.01604	0.01604	0.01604	0.01603	0.01603	0.01604	0.01604	0.01612	0.01612	0.01613

Table 2.14: Accuracy of 12-step-ahead VAR,NNAR and NNAR+FLAP forecasts from simulated VAR(1) data with Laplace distributed errors

Series	VAR RMSE	NNAR model	NNAR RMSE	RMSE of FLAP adjusted forecasts									
				No. of components									
				1	2	3	4	5	6	7	8	9	10
1	0.01760	NNAR (1,0,1)	0.01767	0.01767	0.01770	0.01771	0.01771	0.01771	0.01770	0.01770	0.01771	0.01766	0.01767
2	0.03541	NNAR (1,0,1)	0.03572	0.03574	0.03583	0.03560	0.03558	0.03556	0.03553	0.03551	0.03551	0.03552	0.03552
3	0.01624	NNAR (1,0,1)	0.01628	0.01629	0.01632	0.01633	0.01633	0.01633	0.01625	0.01624	0.01621	0.0162	0.0162
4	0.01433	NNAR (4,0,2)	0.01453	0.01453	0.01453	0.01458	0.01460	0.01459	0.01453	0.01452	0.01453	0.01452	0.01452
5	0.01821	NNAR (1,0,1)	0.01813	0.01812	0.01818	0.01817	0.01816	0.01818	0.01811	0.01807	0.01809	0.01808	0.01808
6	0.05047	NNAR (1,0,1)	0.05044	0.05044	0.05024	0.05045	0.05041	0.05040	0.05060	0.05060	0.05066	0.05062	0.05061
7	0.02035	NNAR (1,0,1)	0.02033	0.02033	0.02034	0.02032	0.02030	0.02029	0.02019	0.02017	0.02020	0.02019	0.02019
8	0.01685	NNAR (1,0,1)	0.01712	0.01712	0.01717	0.01730	0.01731	0.01727	0.01736	0.01736	0.01741	0.01747	0.01748
9	0.03172	NNAR (1,0,1)	0.03166	0.03165	0.03164	0.03152	0.03148	0.03148	0.03148	0.03148	0.03146	0.03146	0.03146
10	0.01607	NNAR (1,0,1)	0.01594	0.01594	0.01593	0.01595	0.01594	0.01594	0.01596	0.01595	0.01607	0.01608	0.01609

It follows from Table 2.11 and 2.12 that FLAP adjusted ARIMA(NNAR) forecasts have lower RMSE than the original ARIMA(NNAR) forecasts but

have higher RMSE than the VAR forecasts.

The above results show the effectiveness of the FLAP projection method in improving the forecast accuracy of heavy tailed t -distributed data. Additionally, it was noted that, the FLAP method does not always reduce the RMSE of forecasts. There are several theoretical and practical reasons for this scenario. The FLAP method relies on the h -step-ahead error covariance matrix (W_h). If the covariance matrix is poorly estimated due to insufficient data or high dimensionality, the FLAP adjustment may not work as intended. In our study, the shrinkage estimators obtained from `corpcor::cov.shrink` R function (Schafer et al., 2021) were used to improve the estimation of the covariance matrix. RMT filtering is another possible alternative to improve covariance matrix estimation.

2.4.4 Real Data Experiments

Experiments are performed using the daily adjusted closing price of 55 stocks from 13 sectors and the S&P 500 market index (56 time series) for the period from 2021-01-01 to 2024-12-31, which are obtained from [Yahoo! Finance](#). Asset prices, and observed volatilities are forecasted using VAR, ARIMA, ARIMA+FLAP, NNAR and NNAR+FLAP models. Observed volatility is calculated using sign correlation.

For stock prices, h -step-ahead forecasts are obtained using VAR, ARIMA and NNAR models. Then, PCA is applied to the 56 return series, and Figure

2.4 shows that 35 components can explain more than 90% of the variance in the original series.

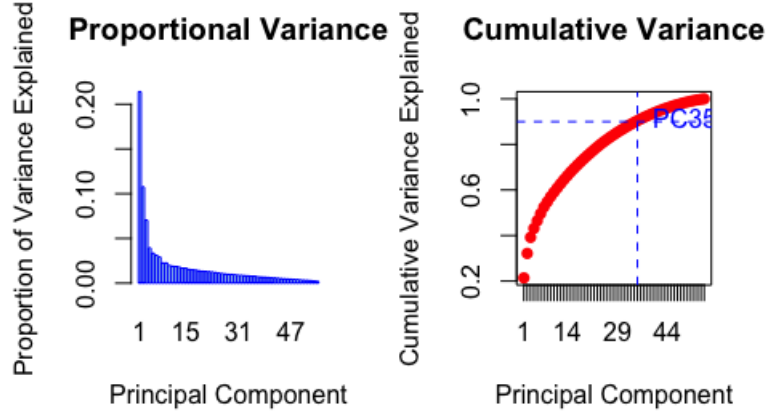


Figure 2.4: Proportional and cumulative variance explained by principal components of original price data

Next, for all 56 components, h -step-ahead forecasts are obtained using ARIMA and NNAR. Finally, the FLAP projection is applied and FLAP adjusted forecasts are obtained. The same procedure is applied to forecast volatilities. Figure 2.5 shows the MSE of $h = 3, 12, 21, 30$ step ahead FLAP adjusted ARIMA/NNAR forecasts of the 56 stock volatilities, respectively. It is evident that irrespective of the forecasting model (ARIMA/NNAR) or the data (price/volatility), long term FLAP forecasts have higher out-of-sample MSE compared to short term FLAP forecasts.

In Yang et al. (2024), it is mentioned that when the number of principal components used in the FLAP projection increases, forecast accuracy increases

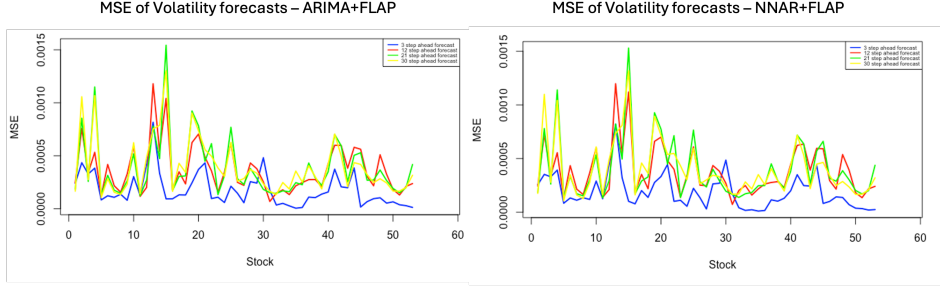


Figure 2.5: MSE of ARIMA and NNAR volatility forecasts at different forecast horizons ($h = 3(\text{blue}), 12(\text{red}), 21(\text{green}), 30(\text{yellow})$) for all 56 stocks

(MSE of forecasts decrease). In order to test this with stock prices and volatility data, RMSE of flap forecasts are calculated by considering different number of components ($n_c = 1, \dots, 56$). For each of 56 stocks, 2 data sets (price and volatility), $h = 3, 12, 21$ forecasts were plotted and out of $56 * 3 * 3 = 477$ plots, selected plots were reported in this section.

Figure 2.7 illustrates the performance of 5 stocks (GOOG, NVDA, META, AAPL, NIKE) data by comparing the RMSE values of $h = 12$ steps ahead forecasts for VAR, ARIMA, and NNAR models against FLAP forecasts for price, and volatility with varying numbers of components. It is evident that the RMSE of NNAR and VAR price forecasts is lower than that of ARIMA forecasts. However, FLAP improves ARIMA price forecasts more significantly than it does for NNAR price forecasts. For volatility, FLAP enhances the accuracy of NNAR forecasts more than for ARIMA forecasts. The performance of NVDA data follows a similar pattern to GOOG. The RMSE of NNAR and VAR price forecasts is consistently lower than that of ARIMA across all

three cases, but FLAP improves ARIMA price forecasts more than NNAR forecasts. For volatility, FLAP again improves the forecast accuracy of NNAR more than that of ARIMA. In META data compared to GOOG and NVDA, the improvement in ARIMA price forecasts using FLAP remains consistent. However, for volatility, there is no significant improvement. Notably, when considering RMSE values with 56 components, the NNAR + FLAP forecast yields a smaller RMSE value. In APPLE it is clear that FLAP improves the accuracy of NNAR forecasts. Similarly, in NIKE for price and volatility, NNAR forecasts are improved by FLAP.

Finally, the total RMSE of price and volatility forecasts are plotted in Figure 2.6. It can be seen that total RMSE of ARIMA price forecasts improved by FLAP while NNAR FLAP forecasts are much accurate. In volatility forecasting total RMSE of NNAR and ARIMA are almost similar and FLAP improves NNAR forecast accuracy than that of ARIMA.

Overall, it is observable that in price forecasting FLAP improves ARIMA forecasts than NNAR forecasts. But in volatility forecasts, FLAP improves NNAR forecasts than the ARIMA forecasts. However in majority of the stocks out of 56 stocks, NNAR forecasts are much accurate than ARIMA forecasts. Also it is observed that when the number of principal components used in the FLAP projection increases, forecast accuracy increases (RMSE of forecasts decrease).

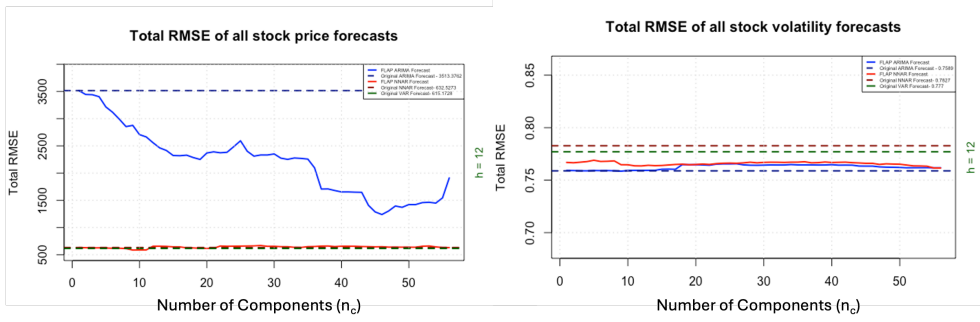


Figure 2.6: Comparison of out-of-sample total RMSE of of 12 step ahead VAR, ARIMA and NNAR price and volatility forecasts using different p number of components for FLAP projection (Green dashed line - VAR forecast, Blue dashed line - original ARIMA forecast, BLUE solid line - FLAP ARIMA forecast, Red dashed line - original NNAR forecast, Red solid line - FLAP NNAR forecast)

2.5 Conclusion

This chapter discusses univariate and multivariate time series forecasting methods and applied in volatility forecasting of cryptocurrencies and stocks. Among machine learning models, the NNAR model with a single hidden layer outperformed more complex architectures like RNN, LSTM, GRU, BiLSTM, and BiGRU. These findings highlight the significance of model selection in improving forecasting accuracy, offering valuable insights for investors. Further, this chapter extends the FLAP method to NNAR models and incorporates non-normal error distributions. A simulation study demonstrates the robustness of FLAP in reducing the Root Mean Square Error (RMSE) of forecasts for heavy-tailed distributions, such as the t -distribution and Laplace distribution, suggesting its applicability in return and volatility forecasting. An empirical

analysis of stock prices and volatility reveals that FLAP-adjusted NNAR forecasts outperform FLAP adjusted ARIMA forecasts, particularly in moderate and long-term forecasting. Additionally, the findings confirms that forecast accuracy improves as the number of components involved in the projection increases. A key contribution of this chapter is demonstrating that non-linear multivariate time series forecasts using the non-linear neural network autoregressive (NNAR) model with FLAP outperforms those obtained by the commonly used linear autoregressive integrated moving average (ARIMA) model forecasts with FLAP, resulting lower out-of-sample MSE in general. However, challenges such as estimating the forecast error covariance matrix remain areas for future research.

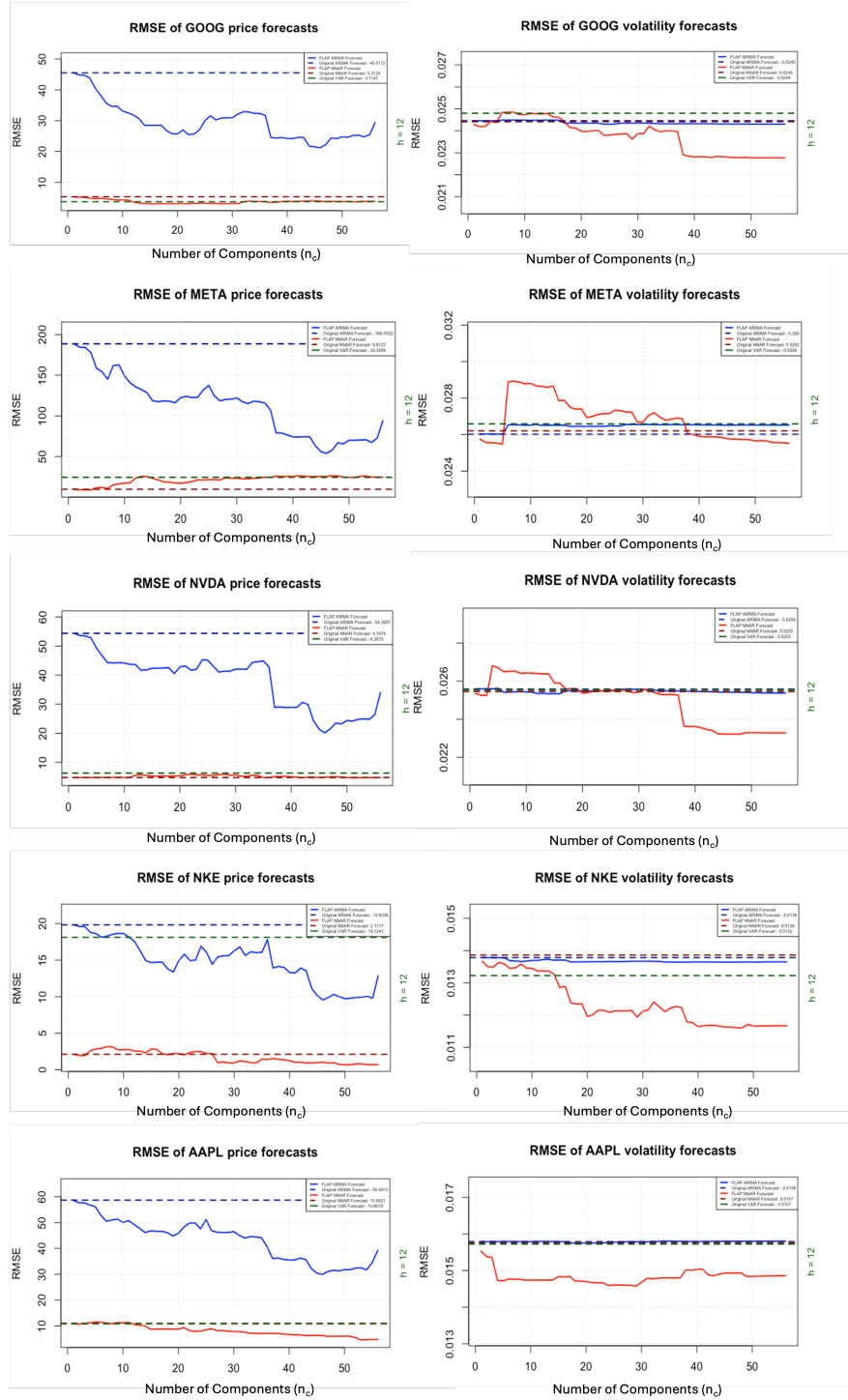


Figure 2.7: Comparison of Out-of-sample forecast accuracy of 5 stocks (Green dashed line - VAR forecast, Blue dashed line - original ARIMA forecast, BLUE solid line - FLAP+ARIMA forecast, Red dashed line - original NNAR forecast, Red solid line - FLAP+NNAR forecast)

Chapter 3

Network Analysis

Recently, there has been a growing interest in utilizing empirical correlations of log returns to examine financial network models. In financial networks, nodes represent assets (e.g., stocks or cryptocurrencies), and edges capture the relationship between two assets. The most common financial networks are based on the correlation of log returns (r_t) and are also used to reflect statistical dependencies between assets (Saha et al., 2022) (Mantziou et al., 2023). However, modeling and predicting financial risk plays a crucial role in risk management strategies for investment decisions. One of the pivotal metrics defining risk is volatility. But, asset volatility has not been used in the literature to model financial networks using the network theory. Therefore, networks based on correlation between observed volatilities (volatility networks) were introduced by Saumyamala et al. (2024). In Thavaneswaran et al. (2020) it has been shown that the log returns follow a t distribution and that the time series of log returns are also stationary. Hence, Saumyamala et al. (2025) have

shown that the covariance matrix of the absolute value of log returns (having data driven t -distribution) is a scaled version of the function of sign correlation [Thavaneswaran et al. \(2020\)](#) and the covariance matrix of log returns, and then, is used to construct the volatility network. That is, the data driven self adjacency matrix of the volatility network can be determined from the self adjacency matrix of the financial networks. Dynamic price (P_t) networks were introduced in [Ranathungage et al. \(2024\)](#) and recently in [Saumyamala et al. \(2025\)](#), price networks were extended to non-seasonal autoregressive integrated moving average (ARIMA) innovation correlation networks obtained by fitting ARIMA models to the stock prices and neuro innovation correlation networks obtained by fitting NNAR models to the stock prices. ARIMA/neuro innovation correlation networks can be used to describe the dynamic nature and capture the nonlinear structures in the market.

By representing entities as nodes and their relationships as edges, graphs facilitate the study of various real-world phenomena across domains ([Pitas, 2016](#)). In network science, a graph $G = (V, E)$ consists of a set of vertices V and a set of edges E . Every graph can be described mathematically through an adjacency matrix Q . The symmetric adjacency matrix Q of an unweighted, undirected graph is defined such that the element $Q_{ij} = 1$ denotes the presence of an edge/connection between vertex i and vertex j , while $Q_{ij} = 0$ denotes the absence of an edge/connection. Financial networks, commonly known as stock networks, illustrate the interconnections between stocks within a given market.

In such financial networks, individual stocks are depicted as nodes, while the relationships between them are represented as edges. The network or graph can be either directed or undirected with weighted or unweighted edges, reflecting the strength of connections between nodes. For unweighted, undirected graphs, a correlation threshold (ρ_{thres}) is utilized. If the correlation between nodes i and j , denoted as ρ_{ij} , exceeds ρ_{thres} , then, the nodes are connected.

In addition to correlation networks, an adjacency matrix using cosine similarity can also be used to construct networks. Cosine similarity determines whether two asset vectors point in the same direction, but not the magnitude. In portfolio optimization, research demonstrates that the data-driven exponential weighted moving average (DDEWMA) variance-covariance matrix model exhibits superior performance compared to models employing empirical variance-covariance matrices (Zhu et al., 2020). Thus, we propose the DDEWMA correlation matrix of asset prices or volatilities which is derived using the DDEWMA variance-covariance matrix (see Zhu et al. (2020) for more details). In Almog et al. (2015), stock price (P_t) fluctuations are represented using the sign of returns (R_t), and networks are constructed based on the correlation between these sign of R_t series. Further networks for big data based on correlations of log returns, volatility of log returns, data-driven volatility of log returns and $\text{sign}(\pm)$ of returns are also discussed in this chapter.

3.1 Adjacency Matrices and Networks

An adjacency matrix is used to represent a graph's structure, where the entries indicate the presence or absence of edges between nodes. Two necessary conditions must be met by the adjacency matrix of correlation or similarity to have a limiting distribution. The first condition is that graphs must be connected (connectedness). That is, for every pair of vertices/nodes, there is a sequence of paths connecting two nodes. The second condition is that the graph must be aperiodic. The connectedness of networks discussed in this chapter depends on the parameter threshold value. Multiple connected graphs can be obtained at different threshold values. We propose a data-driven approach by sequentially changing the threshold value to obtain threshold values that correspond to the minimally connected graph from adjacency matrices, which are based on the correlation or similarity of the asset (stocks/cryptocurrency) log returns.

In order to investigate how the networks behave in the cryptocurrency market, we consider the adjusted closing prices of six cryptocurrencies, Bitcoin (BTC), Ethereum (ETH), Binance Coin (BNB), Ripple (XRP), DogeCoin (DOGE), and Cardano (ADA), determined by their market capitalization and popularity sourced from [coinmarketcap](#). This datasets cover the period from January 2021 to December 2023. The study utilizes daily adjusted closing prices to compute log returns, which are downloaded from [Yahoo! Finance](#). Range of returns (simple net returns) are from -1 to $+\infty$. The lower bound

results from the fact that shareholders have limited liability and cannot lose more than 100% of their investment. This property also implies, obviously, that returns cannot be normal. But range of log returns are from $-\infty$ to $+\infty$. Therefore, it makes log returns are normal and more suitable for the network analysis. In this section networks for log returns of cryptocurrencies are constructed using three adjacency matrices (based on empirical correlation, DDEWMA correlation, and cosine similarity), and changes in fuzzy parameters and structure of minimally connected networks are analyzed. The importance of finding the minimally connected network is to determine the most linked (important) node (stock/cryptocurrency).

3.1.1 Empirical Correlation Matrix

Let an asset (stocks/cryptocurrency) price sequence be denoted by P_t where t represents a particular day or a time (frequency). The relative change in the price of a financial asset over a given time interval is usually expressed as a percentage. Continuously compounded (CC) return/log-return of i^{th} asset is calculated by the formula: $r_{i,t} = \log(P_{i,t}/P_{i,t-1}) = \log(P_{i,t}) - \log(P_{i,t-1})$.

The cross-correlation coefficient ρ_{r_i, r_j} between two log returns, r_i and r_j of asset i and j , is calculated by using the formula:

$$\rho_{r_i, r_j} = \frac{\sum (r_{i,t} - \bar{r}_i)(r_{j,t} - \bar{r}_j)}{\sqrt{\sum (r_{i,t} - \bar{r}_i)^2 \sum (r_{j,t} - \bar{r}_j)^2}}, \quad (3.1)$$

where $\bar{r}_i$ and $\bar{r}_j$ are the sample means of r_i and r_j , respectively. The empirical cross-correlation matrix for a dataset with k assets is a $k \times k$ square matrix, where each element ρ_{r_i, r_j} represents the cross-correlation coefficient between log returns of r_i and r_j . Then, a correlation threshold is applied to determine whether a pair of assets are highly correlated enough to be connected. The adjacency matrix Q is constructed such that the element $Q_{ij} = 1$ indicates that the correlation coefficient between an asset pair exceeds the threshold and $Q_{ij} = 0$ otherwise. From this adjacency matrix minimally connected network given in Figure 5.1 was obtained.

Emperical Correlation Network of log returns at Threshold = 0.11

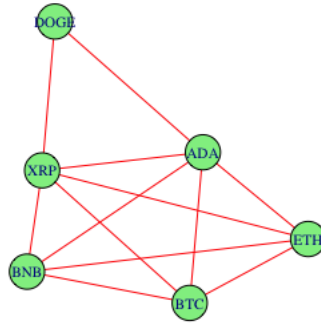


Figure 3.1: Empirical correlation network based on log returns of cryptocurrencies

3.1.2 DDEWMA Correlation Matrix

The DDEWMA algorithm for computing a one-step-ahead forecast of volatility and residuals proposed in [Zhu et al. \(2020\)](#) is summarized in Algorithm 2, and Algorithm 3 shows how to obtain the data-driven EWMA correlation matrix of the assets. Algorithm 2 provides the procedure of computing one step ahead volatility forecast and corresponding residuals using the data-driven EWMA algorithm. This algorithm begins with log returns and computes correlation, followed by computing the observed volatility, which is used to forecast the initial volatility. In Algorithms 2 and 3 we define the time window $[1, T_1]$ as the training period, and $[T_1 + 1, T_2]$ as the testing period. n number of observations in the training period are used for the initial volatility forecast and in practice we use $n = 30$. The observations in the training period are used to determine the smoothing constant ω , and we use the same ω value to forecast the volatility of log returns during the testing period. By introducing a smoothing constant ω , the algorithm then, refines the forecasted volatility. The sign correlation to obtain the DDEWMA volatility is first discussed in the [Thavaneswaran et al. \(2020\)](#). Further, details to obtain the corresponding degree of freedom of heavy tail distribution of log returns and limitations of traditional approaches of finding volatility can be found in the [Thavaneswaran et al. \(2020\)](#).

The covariance matrix of the standardized residuals is considered as the correlation matrix. The diagonal elements of the variance-covariance matrix of

Algorithm 2 Dynamic DDEWMA volatility forecasts of returns

Require: Data: Adjusted closing stock prices $P_{i,t}, i = 1, \dots, k, t = 1, \dots, T_2$. $[1, T_1]$ is the training period, and $[T_1 + 1, T_2]$ is the testing period.

- 1: $r_{i,t} \leftarrow \log(P_{i,t}) - \log(P_{i,t-1}), t = 2, \dots, T_1, \dots, T_2$ {Compute first difference of log returns}
- 2: $\hat{\rho}_i \leftarrow \text{Corr}(r_{i,t} - \bar{r}_i, \text{sign}(r_{i,t} - \bar{r}_i))$ {compute correlation between $r_{i,t} - \bar{r}_i$ and $\text{sign}(r_{i,t} - \bar{r}_i)$ from the first T_1 observations.}
- 3: $Z_{i,t} \leftarrow |r_{i,t} - \bar{r}_i| / \hat{\rho}_i$ {Compute observed volatility}
- 4: $S_{i,0} \leftarrow \frac{\sum_{t=1}^n Z_t}{n}$ {Initial volatility forecast}
- 5: $\omega_i \leftarrow (0, 1)$ {Set a range for smoothing constant}
- 6: $S_{i,t} \leftarrow \omega_i Z_{i,t} + (1 - \omega_i) S_{i,t-1}, t = 1, \dots, T_1$
- 7: $\omega_{i,opt} \leftarrow \min_{\omega_i} \sum_{t=n+1}^k (Z_{i,t} - S_{i,t-1})^2$ {Determine optimal ω_i by minimizing the one-step ahead Forecast Error Sum of Squares}
- 8: **for** $t \leftarrow 1, T_2$ **do**
- 9: $S_{i,t} \leftarrow \omega_{i,opt} Z_{i,t} + (1 - \omega_{i,opt}) S_{i,t-1}$
- 10: $res_{i,t} \leftarrow (r_{i,t} - \bar{r}_i) / S_{i,t-1}$ {Covariance of the residuals is the same as correlation of the residuals in this case.}
- 11: **end for**
- 12: **return** $S_{i,t}, t = T_1, \dots, T_2$
- 13: **return** $res_{i,t}, t = 1, 2, \dots, T_2$

log returns are squared values of data-driven EWMA volatilities, and the off-diagonal elements represent the product of correlations of DDEWMA residuals and the corresponding DDEWMA volatilities. That is, the data-driven variance matrix is: $\Sigma^{dd}[t-1]_{i,j} \leftarrow cor^{dd}[t-1]_{i,j} \hat{\sigma}_{i,t} \hat{\sigma}_{j,t}$

Figure 5.2 gives the minimally connected network obtained from corresponding adjacency matrix of DDEWMA correlation matrix.

Algorithm 3 Dynamic data-driven EWMA correlation matrix

Require: Data (adjusted closing price of stocks) $P_{i,t}$, $i = 1, \dots, k$, $t \in [1, \dots, T_2]$. $[1, T_1]$ is the training period, and $[T_1 + 1, T_2]$ is the testing period.

- 1: $r_{i,t} \leftarrow \log(P_{i,t}) - \log(P_{i,t-1})$, $t = 2, \dots, T_2$ {Compute first difference of log returns}
- 2: $\hat{\sigma}_{i,t}$ is the $S_{i,t-1}$ for $r_{i,t}$ in Algorithm 2, $t = T_1 + 1, \dots, T_2$
- 3: $\Sigma^{dd}[t-1]_{i,i} \leftarrow \hat{\sigma}_{i,t}^2$ {compute the diagonal elements of the covariance matrix of returns}
- 4: $cor^{dd}[t-1]$ is the estimated correlation matrix of recent standardized residuals before t .

3.1.3 Cosine Similarity Matrix

Besides using the correlation between assets to construct adjacency matrices, pairwise similarity between data points can also be used. First, a similarity matrix is computed based on a similarity measure such as cosine similarity. Let $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{w}}$ be two N -dimensional vectors. The cosine value of the angle between the two vectors $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{w}}$ is calculated as follows:

$$\cos(\tilde{\mathbf{v}}, \tilde{\mathbf{w}}) = \frac{\sum_{i=1}^N v_i w_i}{\sqrt{\sum_{i=1}^N v_i^2} \cdot \sqrt{\sum_{i=1}^N w_i^2}}. \quad (3.2)$$

The cosine similarity value ranges from -1 to $+1$. A value of $+1$ suggests vectors aligned in the same direction, 0 denotes orthogonal or perpendicular vectors, and -1 signifies vectors pointing diametrically opposite. Then, a similarity threshold is applied to determine which pairs of assets are considered similar enough to be connected. Finally, the adjacency matrix is obtained, where the entry $Q_{i,j} = 1$ indicates that the similarity between assets exceeds

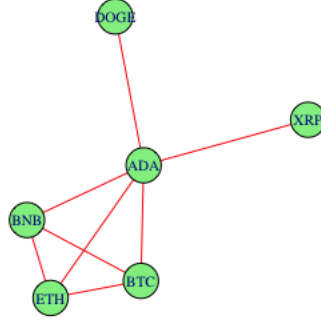
DDEWMA Correlation Network of log returns at Threshold = 0.15

Figure 3.2: DDEWMA correlation network based on log returns of cryptocurrencies

the threshold, and $Q_{ij} = 0$ otherwise. Figure 3.3 gives the minimally connected network obtained from corresponding adjacency matrix of Cosine similarity matrix.

The three correlation matrices in Figure 5.1, Figure 5.2 and Figure 3.3 are obtained at different threshold values. It implies that correlation structures are different from each other. As a result in each network, the links(connections) and mostly/least connected (most/least influential) stocks are different.

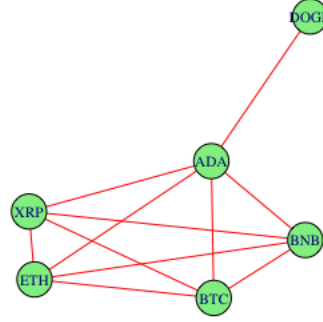
Cosine Similarity Network of log returns at Threshold = 0.13

Figure 3.3: Cosine similarity network based on log returns of cryptocurrencies

3.2 Fuzzy Adjacency Matrices and Fuzzy Networks

Real-world problems deal with uncertain, dynamic, and imprecise data. Since [Zadeh \(1965\)](#) introduced the fuzzy set theory in 1965, there has been an increasing interest in applying the fuzzy approach across various domains. [Thavaneswaran et al. \(2009\)](#) introduced non-linear adaptive fuzzy numbers and applied to GARCH modeling and option pricing. Fuzzy numbers are employed to depict uncertainty and imprecision present in real-world data, where exact boundaries or membership degrees are not clearly defined. The literature has extensively explored the concept of fuzzy sets to offer insights

into designing forecasting models. These models combining randomness and fuzziness are used to handle volatility forecasting problems and also to solve portfolio optimization problems (Thavaneswaran et al., 2021a). In practical scenarios, to construct networks, the probability of connectedness (correlation or similarity) between two nodes is not directly observed but should be estimated from available data. In order to address the uncertainty associated with the point estimates of connectedness, this study introduces data-driven non-linear adaptive symmetric fuzzy adjacency matrices, which can be identified as confidence matrices (interval estimates). Then, propose a novel data-driven volatility network model based on the observed volatility with a data-driven non-linear fuzzy volatility network model.

3.2.1 Non-Linear Fuzzy Number and α -cuts

A fuzzy number $Z(x)$, $x \in \mathbb{R}$ has the following form: $Z(x) = g(x)$ if $x \in [a, b]$, $Z(x) = 1$ if $x \in [b, c]$, $Z(x) = h(x)$ if $x \in [c, d]$, and $Z(x) = 0$ otherwise. In this context, both functions g and h are real-valued. Function g is right-continuous and increasing while function h is left-continuous and decreasing. The non-linear adaptive asymmetric fuzzy number Z can be represented as $Z = [a, b, c, d]_{u,m}$. The parameters a, b, c, d take real values satisfying $a < b < c < d$. Here, u and m represent positive real numbers ($u, m > 0$). When u and m both equal 1, the notation of the fuzzy number simplifies to $Z = [a, b, c, d]$, representing a linear fuzzy number often known as a trapezoidal fuzzy number. Functions g and h

are further specified as: $g(x) = ((x-a)/(b-a))^u$ and $h(x) = ((d-x)/(d-c))^m$. A comprehensive discussion of fuzzy numbers can be found in Zimmermann (2001).

The horizontal representation of a fuzzy number is known as α -cuts. Figure 3.4 illustrates the non-linear adaptive asymmetric fuzzy number and its corresponding α -cuts, and it is adaptable to the triangular fuzzy number by setting $b = c$. The α -cuts of the asymmetric adaptive non-linear fuzzy number $Z = [a, b, c, d]_{u,m}$ are given by $[a + \alpha^{\frac{1}{u}}(b-a), d - \alpha^{\frac{1}{m}}(d-c)]$ for all $\alpha \in [0, 1]$. Note that our study considers the case $u = m$ for simplicity, which is the non-linear adaptive symmetric fuzzy number. Detailed explanations of non-linear adaptive fuzzy numbers can be found in Thavaneswaran et al. (2009).

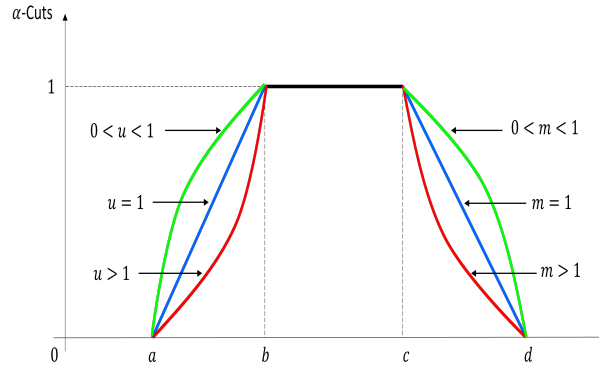


Figure 3.4: Plot of α -cuts of the non-linear adaptive asymmetric fuzzy number

R functions like `markovchainFit` are capable of generating confidence matrices (interval estimates for a matrix) known as lower and upper endpoint matrices. However, those confidence matrices are not suitable for subsequent

calculations. For example, when confidence matrices are obtained for transition probabilities (TP) of a Markov chain, they are not appropriate to represent TP matrices because the confidence matrices are not stochastic (Ranathungage et al., 2024). Therefore, this study proposes fuzzy adjacency matrices, which are stochastic and suitable for subsequent calculations, such as PageRank. The steps to obtain fuzzy adjacency matrices $Q(\alpha)$ (α -cuts) for a given stochastic matrix Q are provided in Algorithm 4.

3.2.2 Fuzzy Cryptocurrency Networks Using Log Returns

Following Algorithm 4, fuzzy financial networks for cryptocurrencies are obtained using the three correlation/similarity matrices. For selected values of m , by sequentially changing the threshold and α in the range $(0, 1)$, network graphs were obtained and the corresponding parameter values for minimally connected graphs are reported in Table 3.1. From the values given in Table 3.1, it is evident that for empirical and DDEWMA correlation, irrespective of the values of m , the minimally connected network is obtained at a constant threshold value. It can also be observed that as m increases, α decreases for empirical and data-driven correlations. But the cosine similarity matrices behave differently from correlation matrices.

Figures 3.5, 3.6, and 3.7 are examples of connected network graphs using

Algorithm 4 Data-driven non-linear adaptive symmetric fuzzy adjacency matrices for networks

Require: Data: Adjusted closing price of stocks/indexes/Cryptocurrencies ($P_{i,t}, i = 1, \dots, k, t = 1, \dots, N$). Parameters: Let k be the number of distinct stocks (states), $\alpha \in [0, 1]$, $m \in \mathbb{R}^+$.

- 1: $r_{i,t} \leftarrow \log P_{i,t} - \log P_{i,t-1}; t = 2, \dots, N$
- 2: Compute the cross-correlation (empirical/data-driven) matrix or the similarity matrix (C) of r_t . $\{c_{ij}$ represents cross-correlation between i^{th} and j^{th} asset log returns $\}$.
- 3: Compute adjacency matrix (Q) with ij^{th} element given by

$$q_{ij} = \frac{|c_{ij}|}{\sum_{j=1}^k |c_{ij}|},$$

which is a stochastic matrix.

- 4: Determine the threshold value (approximately $1/k$) for matrix Q and assign $q_{ij} = 0$ if $q_{ij} < \text{threshold value}$.
- 5: Construct a doubly stochastic matrix A where A is $k \times k$ matrix with $a_{ij} = 1/k$ for each $i, j = 1, \dots, k$.
- 6: Calculate the fuzzy adjacency matrices (α -cuts) of Q denoted by

$$Q(\alpha) = \alpha^{1/m} * Q + (1 - \alpha^{1/m}) * A$$

where $\alpha \in [0, 1]$ and $m \in \mathbb{R}^+$. $\{ij^{th}$ element of $Q(\alpha)$ is given by

$$q_{ij}(\alpha) = \alpha^{1/m} * q_{ij} + (1 - \alpha^{1/m}) * a_{ij}$$

empirical correlation, DDEWMA correlations, and cosine similarity at $m = 0.5$, $m = 1.0$ and $m = 2.0$. In all the fuzzy network graphs using correlations, ADA seems to be the most important cryptocurrency as it is connected (i.e., has a significant link) to all the other cryptocurrencies. Depending on the correlation matrix the network structure changes. Also, depending on the value of m , the α and threshold value (at which the minimally connected graph is obtained) changes.

Table 3.1: Parameters for minimally connectedness of fuzzy networks based on log returns of cryptocurrency

Adjacency Matrix	m	α	Threshold
Empirical correlation	0.5	0.97	0.14
	1.0	0.94	0.14
	2.0	0.89	0.14
DDEWMA correlation	0.5	0.70	0.16
	1.0	0.49	0.16
	2.0	0.24	0.16
Cosine similarity	0.5	0.88	0.15
	1.0	0.94	0.14
	2.0	0.89	0.14

3.3 Volatility Network

Most of the financial networks considered in the literature are based on empirical correlations between stocks calculated using log returns of the stock prices. The volatility networks based on the cross-correlation of the absolute centered

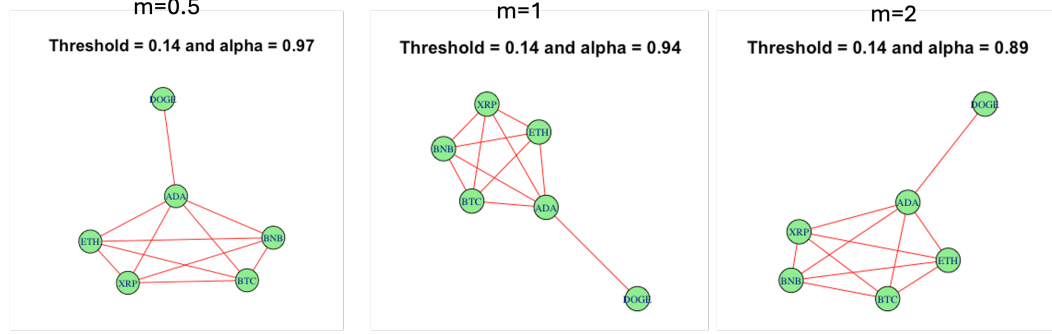


Figure 3.5: Fuzzy networks of empirical correlation matrices based on log returns of cryptocurrency

log returns of assets are introduced in [Saumyamala et al. \(2024\)](#). That is,

$$\hat{\rho}_{|r_i|,|r_j|}^{vol} = \frac{\sum_{k=1}^n (|r_{ik}| - |\bar{r}_i|) (|r_{jk}| - |\bar{r}_j|)}{\sqrt{Var(|r_{ik}| - |\bar{r}_i|) Var(|r_{jk}| - |\bar{r}_j|)}}.$$

The empirical correlation of volatility and cosine similarity of volatility networks are given in Figure 3.8 and 3.9.

The fuzzy volatility networks involves adapting Algorithm 4 with absolute centered log returns of stocks instead of stock returns in step 1 and computing a volatility matrix using $\hat{\rho}_{|r_i|,|r_j|}^{vol}$ in step 2. The resulting fuzzy adjacency matrix $Q(\alpha)$ from step 6 using volatility is called the data-driven non-linear adaptive symmetric fuzzy volatility adjacency matrix. Figure 3.10 shows the fuzzy networks of empirical correlation matrices based on volatility of cryptocurrency and Figure 3.11 shows the fuzzy networks of cosine similarity matrices based on volatility of cryptocurrency. Parameters for minimally connected networks are summarized in Table 3.2.

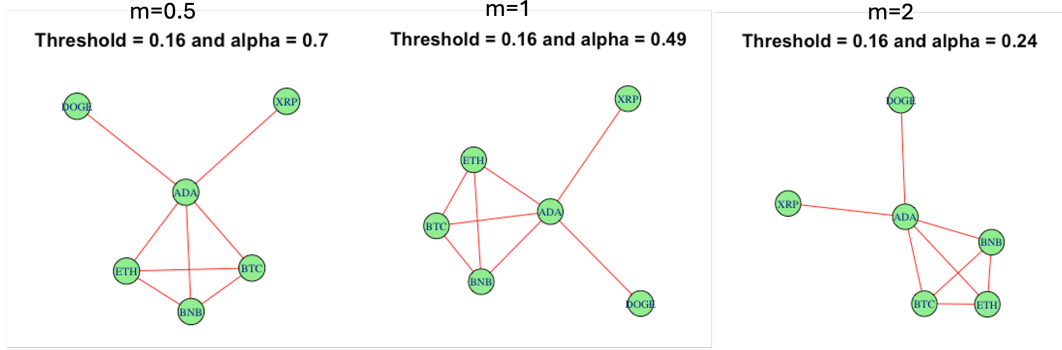


Figure 3.6: Fuzzy networks of DDEWMA correlation matrices based on log returns of cryptocurrency

Table 3.2: Parameters for minimally connectedness of fuzzy networks based on volatility of cryptocurrency

Adjacency Matrix	m	α	Threshold
Empirical correlation	0.5	0.88	0.13
	1.0	0.98	0.12
	2.0	0.97	0.12
Cosine similarity	0.5	0.88	0.15
	1.0	0.78	0.15
	2.0	0.61	0.15

3.3.1 Covariance Relationship

Given a set of log returns r_t , first the absolute returns $|r_t|$ are calculated as a simple measure of volatility. Then, covariance or correlation matrix which forms the basis for constructing a volatility network is calculated. The following theorem gives the covariance matrix of absolute returns $|r_t|$ as a scaled version of the covariance matrix of the returns r_t themselves.

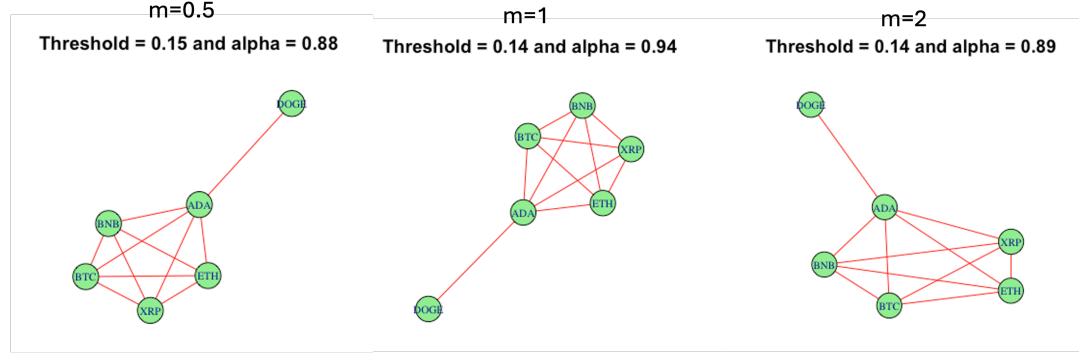


Figure 3.7: Fuzzy networks of cosine similarity matrices based on log returns of cryptocurrency

Theorem I:

a) For any two identically distributed symmetric random variables X, Y with mean zero, variances σ^2 and sign correlation ρ_{sign} ,

$$\text{Cov}(|X|, |Y|) = (1 - \rho_{sign}^2) \text{Cov}(X, Y).$$

b) The data-driven covariance matrix of the absolute value of log returns $|r_t|$ is

$$\Sigma_{|r|} = (1 - \rho_{sign}^2) \Sigma_r, \quad (3.3)$$

where $\Sigma_r = \mathbb{E}[r_t r_t^\top]$ is the covariance matrix of returns, $\Sigma_{|r|} = \mathbb{E}[|r_t| |r_t|^\top]$ is the covariance of the absolute returns and ρ_{sign} is the sign correlation, $\rho_{sign} = \text{cor}(r_t - \bar{r}, \text{sign}(r_t - \bar{r}))$ as in [Thavaneswaran et al. \(2020\)](#).

Proof: The proof of Theorem I is presented here.

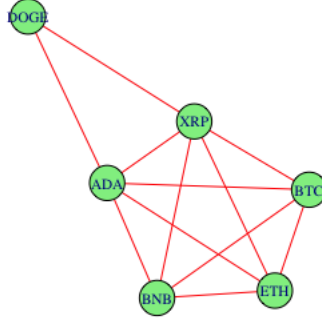
Empirical Correlation Network of volatility at Threshold = 0.11

Figure 3.8: Empirical correlation network based on volatility of cryptocurrencies

From first-order approximation, it follows that

$$Cov(|X|, |Y|) = \frac{Var(|X|)}{Var(X)} Cov(X, Y),$$

where,

$$\begin{aligned} Var(|X|) &= \mathbb{E}[|X|^2] - (\mathbb{E}[|X|])^2 \\ &= \mathbb{E}[X^2] - (\mathbb{E}[|X|])^2 \\ &= Var(X) - (\mathbb{E}[|X|])^2. \end{aligned}$$

Then,

$$\frac{Var(|X|)}{Var(X)} = 1 - \frac{(\mathbb{E}[|X|])^2}{Var(X)}.$$

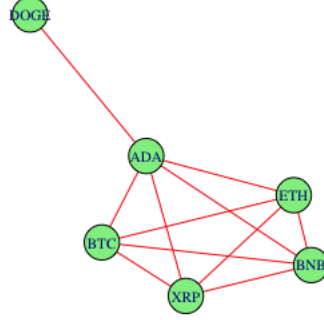
Cosine Similarity Network of volatility at Threshold = 0.14

Figure 3.9: Cosine similarity network based on volatility of cryptocurrencies

Using the fact, $\rho_{sign} = \frac{E|X|}{\sigma}$, we can write

$$\frac{Var(|X|)}{Var(X)} = 1 - \rho_{sign}^2.$$

Here,

$$Cov(|X|, |Y|) = (1 - \rho_{sign}^2)Cov(X, Y).$$

Similarly,

$$\Sigma_{|r|} = (1 - \rho_{sign}^2) \Sigma_r,$$

where $\rho_{sign} = cor(r_t - \bar{r}, sign(r_t - \bar{r}))$.

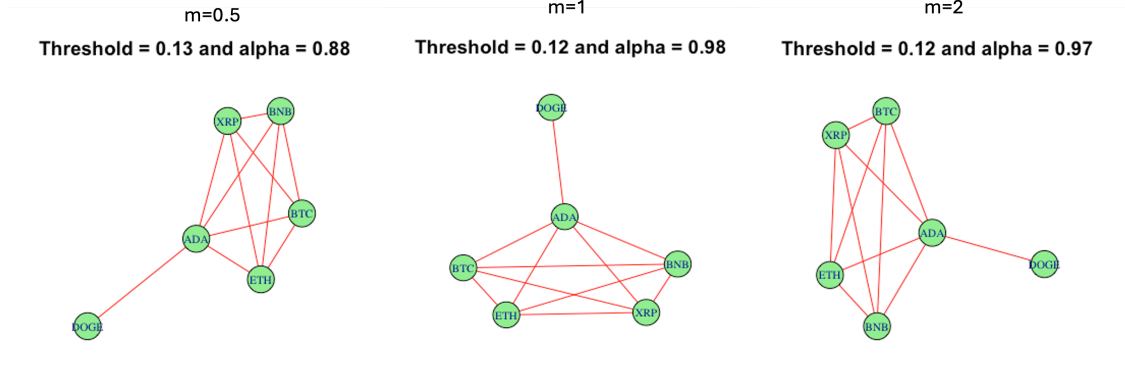


Figure 3.10: Fuzzy networks of empirical correlation matrices based on volatility of cryptocurrency

Moreover, the eigenvalues are related by

$$\lambda_k^{(|r|)} = (1 - \rho_{sign}^2) \lambda_k^{(r)}, \quad (3.4)$$

where $\lambda_k^{(|r|)}$ are the eigenvalues of $\Sigma_{|r|}$ and $\lambda_k^{(r)}$ are the eigenvalues of Σ_r .

For normally distributed returns, $r_t \sim N(0, \sigma^2)$, $\rho_{sign}^2 = \frac{2}{\pi}$ and $(1 - \rho_{sign}^2) = 0.3634$ (see [Thavaneswaran et al. \(2020\)](#) for more details). Similarly for t distributed returns, $r_t \sim t_\nu(0, \sigma^2)$ with d.f. $\nu > 2$, the scale factor can be shown as

$$1 - \rho_{sign}^2 = 1 - \frac{\nu}{\nu - 2} \left(\frac{\Gamma(\frac{\nu-1}{2})}{\sqrt{\pi} \Gamma(\frac{\nu}{2})} \right)^2 \quad (3.5)$$

The following Table 3.3 provides the sign correlations and scale factors for a class of symmetric normal, t and data driven t -distributions. Normal distribution has the highest theoretical sign correlation and lowest scale factor

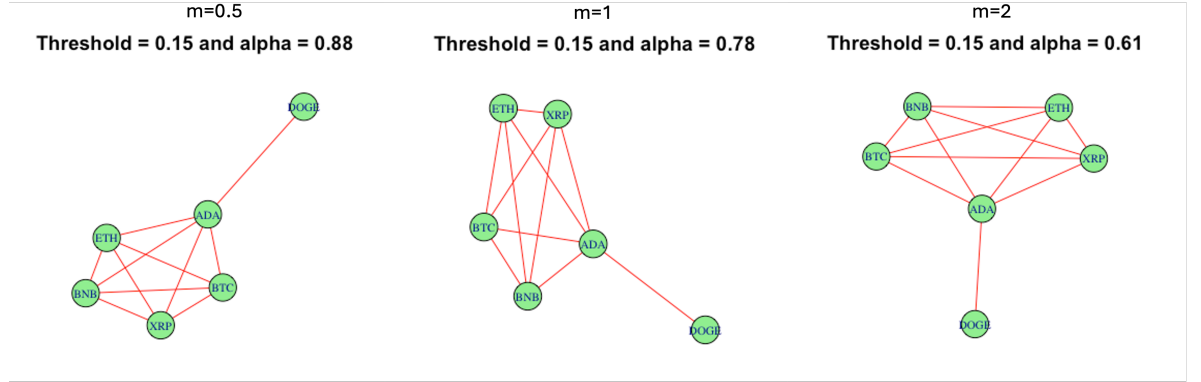


Figure 3.11: Fuzzy networks of cosine similarity matrices based on volatility of cryptocurrency

(0.3634). It follows from Table 3.3, when returns follow a t -distribution, the heavier tails result in a larger scale factor, reflecting higher variability in $|r_t|$. That is among the 12 listed stocks out of 55 stocks, ‘SBUX’ has the lowest sign correlation and the highest scale factor. Moreover, for S&P500 big data scale factor for almost all the stocks are approximately 0.5.

Volatility networks are constructed using the covariance matrix of absolute value of log returns $|r_t|$, where the weights of the edges reflect the strength of co-movement in volatility. The eigenvalues of this matrix represent the amount of variance explained by each mode or component. Under the normal return distributions, the scale is uniform, yielding stable networks with similar structure to those based on raw returns. However, when log returns follow a t -distribution, heavy tails increase variability in $|r_t|$, strengthening edge weights and enhancing network connectivity and clustering. This reflects

Table 3.3: Scale factor values for 5 theoretical distributions and 12 data driven distributions

Distribution	As.Var(s)	Theoretical ρ	Data driven ρ_{sign}	Scale factor $(1 - \rho_{sign}^2)$
Normal	$\sigma^2/2n$	0.7979	-	0.3634
t_4	∞	0.7071	-	0.5
t_5	$2\sigma^2/n$	0.7351	-	0.4596
t_6	$1.25\sigma^2/n$	0.7500	-	0.4375
t_7	σ^2/n	0.7592	-	0.4236
$t_{3.81}(AAPL)$	∞	-	0.6987	0.5118
$t_{3.76}(NVDA)$	∞	-	0.6964	0.5150
$t_{3.56}(MSFT)$	∞	-	0.6855	0.5300
$t_{3.25}(JPM)$	∞	-	0.6627	0.5608
$t_{3.66}(AMZN)$	∞	-	0.6911	0.5224
$t_{3.78}(TSLA)$	∞	-	0.6974	0.5136
$t_{3.29}(NKE)$	∞	-	0.6664	0.5559
$t_3(SBUX)$	∞	-	0.6356	0.5960
$t_{3.6}(GOOG)$	∞	-	0.6876	0.5272
$t_{3.07}(META)$	∞	-	0.6445	0.5846
$t_{3.14}(WMT)$	∞	-	0.6521	0.5747
$t_{3.6}(COST)$	∞	-	0.6876	0.5272

greater systemic risk, as extreme co-movements and spillover effects become more likely.

3.3.2 Eigenvalue Distribution and the Marchenko-Pastur Law

Equation 3.3.1 relates the eigenvalues of covariance matrices of absolute values of returns and returns. In here, we are visualizing the eigenvalues of t-distributed returns and normal returns. A random $n \times p$ matrix S with i.i.d. Gaussian entries is generated, and the sample covariance matrix is computed as:

$$C = \frac{1}{p}SS^T.$$

When $n, p \rightarrow \infty$ with aspect ratio $\lambda = \frac{n}{p}$, the eigenvalues of C follow the Marchenko-Pastur (MP) distribution:

$$\rho(\lambda) = \frac{1}{2\pi\sigma^2\lambda} \sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)},$$

where the MP bounds are given by: $\lambda_{\pm} = \sigma^2 \left(1 \pm \sqrt{\lambda}\right)^2$ and $\lambda \in [\lambda_-, \lambda_+]$.

However, when the data follows a t-distribution with heavy tails, the eigenvalue spectrum deviates significantly from the MP bounds. Unlike normal data, where eigenvalues are constrained by the MP bounds, t-distributed data has extreme values, which lead to eigenvalue outliers that exceed λ_+ . This occurs because: $P(|X| > x) \sim x^{-\nu}$, for large x . The presence of such large values perturbs the covariance structure, leading to a broader eigenvalue spectrum. For Normal data, the largest eigenvalue scales as: $\lambda_{max} \approx (1 + \sqrt{\lambda})^2$. However, for t-distributed data with $\nu < 4$, the largest eigenvalue follows a Fréchet-type extreme value distribution, $\lambda_{max} \sim p^{2/\nu}$ which grows beyond the MP bound λ_+ . This explains why the empirical eigenvalue distribution for t-distributed data has heavier tails compared to the MP law as observed in Figure 3.12.

3.3.3 Data Driven Volatility Correlation Networks

It follows from Theorem I, $\Sigma_{|r|} = (1 - \rho_{sign}^2) \Sigma_r$. Then,

$$Cov(|r_i|, |r_j|) = (1 - \rho_{sign}^2) Cov(r_i, r_j).$$

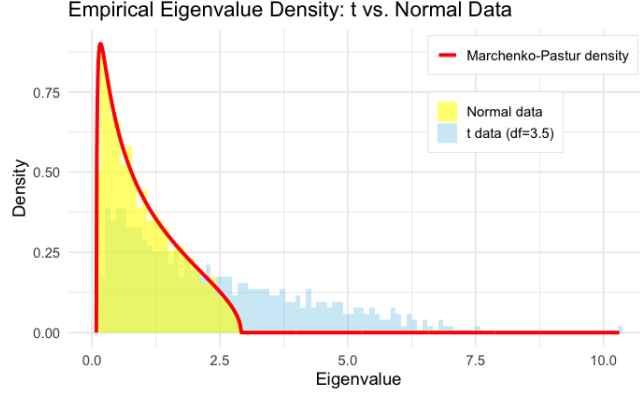


Figure 3.12: Comparison of theoretical MP density and empirical eigenvalue of t ($df=3.5$) distribution vs normal distribution

Then, the data driven volatility correlation can be expressed as

$$\rho_{|r_i|,|r_j|}^{ddvol} = \frac{(1 - \rho_{sign}^2) Cov(r_i, r_j)}{\sqrt{Var(|r_i|) Var(|r_j|)}}.$$

Theorem I assumes that the marginal distribution of all stocks has a common sign correlation. However, we can generalize the relationship under the assumption that X and Y are jointly symmetric (e.g., elliptically distributed with a t -copula), with different marginal distributions. However, there is no universal closed-form expression for this case. Equation 3.5 is used to calculate ρ_{sign}^2 . The self adjacency matrix represents the network structure, with 1 indicating an edge. Following [Saumyamala et al. \(2024\)](#), connected graphs were generated with sequence of thresholds from 0 to 0.75 (step size 0.01), and the minimally connected network was selected as the final network.

3.3.4 Effects of Data Driven Distribution of Returns on Volatility Networks

A volatility network is typically constructed by using the covariance (or correlation) matrix of $|r_t|$ to define the edge weights between nodes (assets). The eigenvalues of this matrix indicate the variance captured along different modes. When returns are normally distributed the scale factor is constant across assets. Therefore, the network structure such as clustering, degree distribution, and centrality remains similar to that obtained using the covariance matrix of r_t , scaled uniformly. Hence, the network is relatively stable, and extreme co-movements are infrequent. When returns follow a t -distribution the heavier tails result in a larger scale factor, reflecting higher variability in $|r_t|$. This leads to larger covariance values among absolute returns, which, in turn, results in stronger edge weights in the network. The network is likely to display higher connectivity and clustering, especially among assets that are prone to extreme movements. Such a network signals increased systemic risk as extreme events (or contagion effects) become more pronounced.

3.4 Autocorrelation Networks

Autocorrelation networks, particularly those constructed from autoregressive (AR) processes, have been widely studied for their ability to model temporal dependencies in time series data. The AR(1) model, which captures first-order

temporal correlation, serves as a foundational process for understanding the structure of such networks. In this context, each time point is treated as a node, and edges are drawn based on the strength of lagged correlations. Prior work has analyzed the spectral and topological properties of networks formed from AR(1) and higher-order processes, highlighting how the decay of autocorrelation influences the connectivity structure and eigenvalue distribution of the associated covariance matrices (see [Matsuda \(2000\)](#), [Bai and Yao \(2010\)](#)). These networks exhibit exponential degree distributions and reveal key properties about memory and persistence in time series. Furthermore, comparisons of autocorrelation networks derived from Gaussian and heavy-tailed (e.g., t) processes have shown significant differences in spectral behavior, underscoring the importance of distributional assumptions in network-based time series analysis. An autoregressive (AR) network graph is a directed graph that visualizes lagged dependencies among multiple time series under an AR(1) model. In this framework, nodes represent individual time series, and directed edges indicate significant lagged influences, where the value of one series at time $t - 1$ affects another (or itself) at time t . This section describes an AR(1) network graph generated from simulated data, detailing its conceptual basis, implementation, interpretation, and relevance to time-series analysis.

An AR(1) model for a single time series $X_i(t)$ is defined as:

$$X_i(t) = \phi_i X_i(t - 1) + \epsilon_t, \quad (3.6)$$

where ϕ_i is the autoregressive coefficient, and $\epsilon_t \sim N(0, \sigma^2)$ is white noise. In

a multivariate setting, the model extends to include cross-series lagged effects:

$$X_i(t) = \phi_i X_i(t-1) + \sum_{j \neq i} \beta_{ij} X_j(t-1) + \epsilon_t, \quad (3.7)$$

where β_{ij} quantifies the influence of $X_j(t-1)$ on $X_i(t)$.

The AR(1) network graph represents these relationships:

- **Nodes:** Each node corresponds to a time series (e.g., $X_1, X_2, \dots, X_5$).
- **Directed Edges:** An edge from X_j to X_i indicates that $X_j(t-1)$ significantly influences $X_i(t)$, determined by statistical significance (e.g., p-value < 0.05).
- **Self-Loops:** If included, a self-loop on X_i reflects significant autocorrelation ($\phi_i \neq 0$).

3.5 Granger Causality Network

Granger causality (GC), formalized by [Granger \(1969\)](#), tests whether one time series can predict another. If the inclusion of past values of time series X improves the prediction of Y beyond the information contained in the past values of Y alone, X is said to Granger cause Y .

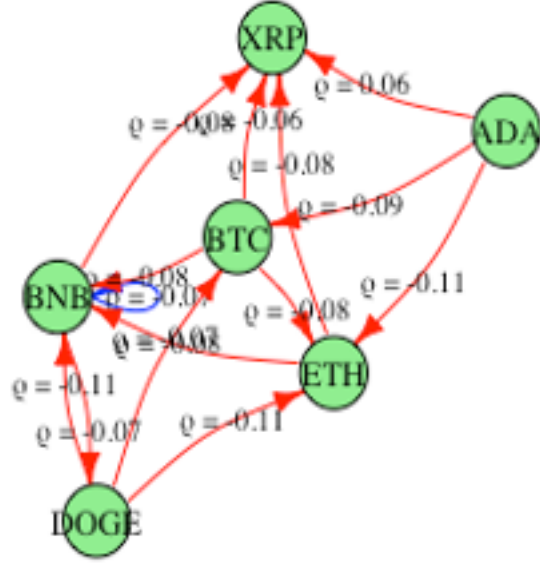


Figure 3.13: Cross-autocorrelation network

Let X_t and Y_t be two stationary time series. The reduced/restricted model can be fitted as a univariate autoregressive (AR) model for Y_t is:

$$Y_t = \sum_{i=1}^p a_i Y_{t-i} + \epsilon_t^{(1)} \quad (3.8)$$

The full/unrestricted model can be fitted as a bivariate AR model, including both X_t and Y_t , is given by:

$$Y_t = \sum_{i=1}^p a_i Y_{t-i} + \sum_{j=1}^p b_j X_{t-j} + \epsilon_t^{(2)} \quad (3.9)$$

The Granger test statistic (typically an F-statistic) relies on asymptotic

properties and does not assume that residuals are normally distributed. Based on the central limit theorem, the test is robust in large samples. However, in small samples non-normality can affect the accuracy of inference. Bootstrap versions of the Granger causality test are often used to address this issue. The variances of the residuals $\epsilon_t^{(1)}$ and $\epsilon_t^{(2)}$ from models (3.11) and (3.12) denoted as σ_X^2 and $\sigma_{X|Y}^2$ respectively, are compared. The Granger causality from Y to X is quantified as:

$$GC_{X \rightarrow Y} = \ln \left(\frac{\sigma_Y^2}{\sigma_{Y|X}^2} \right) \quad (3.10)$$

If $GC_{X \rightarrow Y} > 0$, X Granger-causes Y .

GC has been widely applied in neuroscience for inferring directed connectivity from electro physiological recordings. [Kostelecki \(2013\)](#) investigates the use of GC for identifying directional relationships between neural signals. A robust feature-based framework was proposed for classifying cognitive states by analyzing GC across trials and experimental conditions. Their work focuses on estimating GC using univariate and bivariate autoregressive (AR) models and demonstrated their application in differentiating free and forced motor tasks. Moreover, [Zou \(2010\)](#) explores methods for uncovering causal relationships in complex biological systems using time series data. It focuses primarily on GC, comparing its effectiveness with dynamic Bayesian network inference across both synthetic and real experimental data. It introduces the

concept of Global Granger Causality for large-scale network inference, especially in high-dimensional contexts like gene and protein networks. This work demonstrates Granger causality's strengths in detecting weak interactions when data length is sufficient, and provides computational tools for reconstructing and analyzing complex biological networks in both the time and frequency domains. Linear Granger causality (LGC) has been widely applied to detect causal relationships between continuous-valued time series, particularly using multivariate autoregressive models under the assumption of Gaussianity. To broaden its applicability to non-Gaussian time series, [Kim and Brown \(2010\)](#) propose a general statistical framework for evaluating causal interactions. In this approach, the GC from time series X to time series Y is quantified by examining the relative decrease in the likelihood of Y when X is excluded, compared to the full likelihood that incorporates all available time series.

A Granger causality network is a directed graph where an edge from node X to node Y indicates that past values of X improve the prediction of Y beyond past values of Y alone. In this study, the log returns of six cryptocurrencies (Bitcoin (BTC), Ethereum (ETH), Binance Coin (BNB), Ripple (XRP), Dogecoin (DOGE) and Cardano (ADA)) from January 2021 to December 2023 are used. Log returns are stationary processes. First, we fit a vector autoregressive (VAR) model and it was found that the selected lag order is 5 based on the AIC criterion. Then, we perform pairwise Granger causality tests.

Let r_t^X and r_t^Y be two stationary log return series of two cryptocurrencies.

The univariate autoregressive (AR) model for r_t^Y is:

$$r_t^Y = \sum_{i=1}^5 a_i r_{t-i}^Y + \epsilon_t^{(1)} \quad (3.11)$$

The bivariate AR model, including both r_t^X and r_t^Y , is given by:

$$r_t^Y = \sum_{i=1}^5 a_i r_{t-i}^Y + \sum_{j=1}^5 b_j r_{t-j}^X + \epsilon_t^{(2)} \quad (3.12)$$

Following similar notations the Granger causality from r^Y to r^X is quantified as:

$$GC_{r^X \rightarrow r^Y} = \ln \left(\frac{\sigma_{r^Y}^2}{\sigma_{r^Y|r^X}^2} \right) \quad (3.13)$$

If $GC_{r^X \rightarrow r^Y} > 0$, r^X Granger-causes r^Y . If $r^X \rightarrow r^Y$ with p-value < 0.05 , we consider r^X to Granger-cause r^Y . Then, we summarize these relationships in an adjacency matrix. The Granger causality network is constructed as a directed graph using the adjacency matrix, and plotted using `igraph` in R. In the Granger causality network nodes represent time series and directed edges represent Granger causal influence. Figure 3.14 shows the GC network of log returns and the respective heatmap where red tiles represent the p values less than 0.05 (significant causality). Significant causalities between the respective

stocks are denoted by arrows in the network (considering the cause to effect direction).

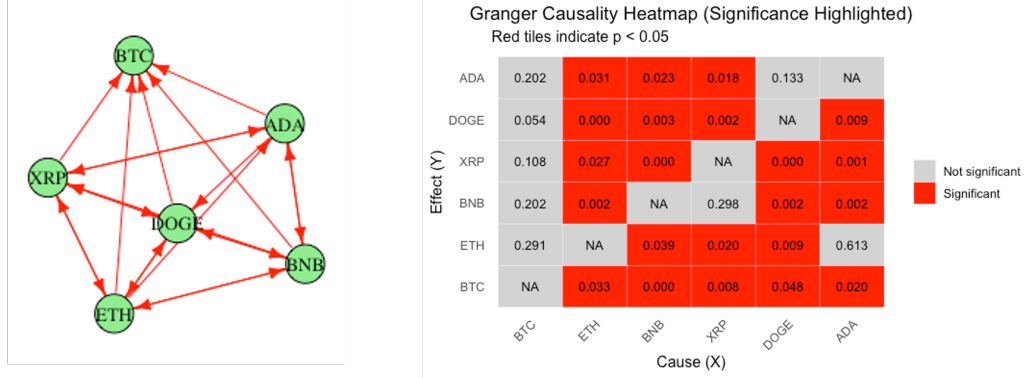


Figure 3.14: Granger causal network and heatmap of log returns of cryptocurrency

It follows from Figure 3.14 that log returns of all other 5 cryptocurrencies cause the log returns of BTC and BTC does not cause others. Also ETH and BNB can be considered as top influencers which cause for all other cryptocurrencies. This might be because since 2021, ETH, BNB, and altcoins have become stronger drivers in specific contexts, and the crypto market has become less BTC-centric. Following similar approach by replacing log returns (r^X and r^Y) with observed volatility (V^X and V^Y) we can obtain the GC network for volatility. GC network for observed volatility of cryptocurrencies is given in Figure 3.15. In the respective heatmap, blue tiles indicate the p values which are less than 0.05.

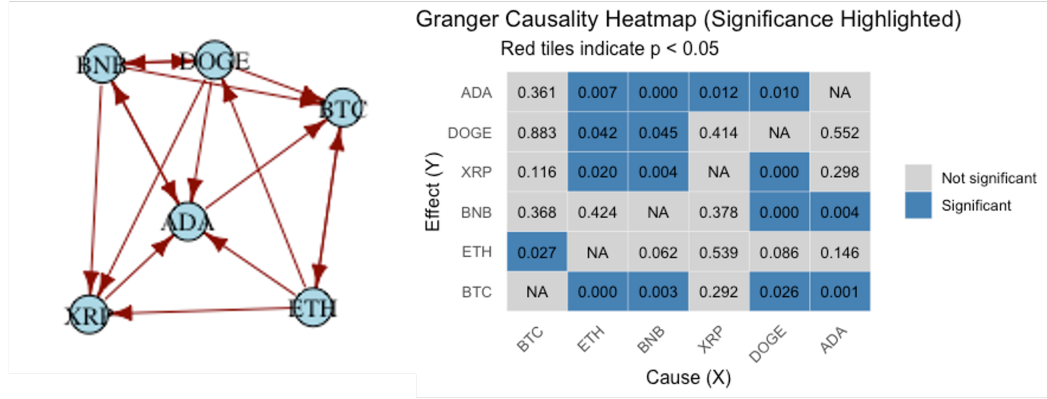


Figure 3.15: Granger causal network and heatmap of volatility of cryptocurrency

3.6 Networks for Big Data

The previous example was a simple example with only six nodes (cryptocurrencies) which is suitable for better visual representation and to observe behavior of networks according to the fuzzy parameters. However, in real world the networks are more complex and involve with many nodes. But when the number of nodes increases it becomes difficult to visually present the network and make conclusions. Therefore, we use network metrics as quantitative measures to compare the networks.

3.6.1 Network Metrics

Let a network be represented by an undirected, unweighted graph $G = (V, E)$, where:

- $|V| = N$ is the number of nodes (e.g., stocks),
- $|E|$ is the number of edges (i.e., pairwise connections),
- $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix of the graph.

Average Node Degree

The average node degree $\bar{k}$ is given by $\bar{k} = \frac{2|E|}{N}$ which measures the average number of connections per node.

Sparsity Level

The sparsity level S is defined as the proportion of missing edges relative to a fully connected graph:

$$S = 1 - \frac{2|E|}{N(N-1)} = 1 - \frac{\bar{k}}{N-1}.$$

A lower value of S indicates a denser network, while higher S implies a sparser structure. Sparsity level is computed for a given network and cannot be controlled.

Maximum Eigenvalue

Let $\lambda_{\max}$ denote the largest eigenvalue of the adjacency matrix A . The maximum eigenvalue reflects the level of overall connectivity and the presence of dominant

hubs or communities in the network. In dense and highly connected networks, $\lambda_{\max}$ tends to be large.

Page Rank

In the analysis of financial networks, particularly those involving cryptocurrencies, identifying the most influential assets is crucial. One widely recognized approach for assessing node importance in such networks is the PageRank algorithm, which is rooted in eigenvector centrality (Page, 1999). PageRank determines a node's relevance based not only on the number of its connections but also on the quality (i.e., importance) of those connections.

Originally developed to rank web pages in search engines, PageRank has since been adopted across multiple domains including financial market analysis, biological systems, and recommendation engines (Baumer et al., 2017). In financial contexts, it can be used to rank assets such as stocks, indices, or cryptocurrencies based on their systemic influence within a network constructed from statistical dependencies like correlation or co-integration.

For instance, Yang et al. (2014) demonstrated the effectiveness of PageRank in ranking 26 global stock indices within a network built from co-integration relationships. Further Ranathungage et al. (2024) used the Page Rank to rank the nodes of price and differenced price correlation networks of stocks. By applying PageRank to a directed, weighted network, they identified the most dominant market indices and quantified their influence over time.

When applied to a cryptocurrency network, where each node represents a distinct cryptocurrency and edges reflect interdependencies (e.g., correlations or co-integration), PageRank enables the identification of key influencers in the market. It follows from Table 3.4 Bitcoin (BTC) with highest PageRank value is interpreted as more central, indicating a stronger ability to affect the price dynamics of other cryptocurrencies in the network.

Table 3.4: PageRank scores for cryptocurrency network nodes

Node	Cryptocurrency	PageRank
1	BTC	0.2062
2	ETH	0.1797
6	BNB	0.1698
3	XRP	0.1682
4	DOGE	0.1625
5	ADA	0.1136

A critical parameter in the PageRank algorithm is the damping factor (teleport probability), which reflects the probability that a random walk will follow an outgoing edge rather than jumping to a random node. This parameter, typically ranging between 0 and 1, affects the algorithm’s convergence and sensitivity. Higher damping factors slow convergence but reduce sensitivity to local structure, while lower values enhance responsiveness to changes in network topology (Engström and Silvestrov, 2014). In practice, a fine-tuned range (e.g., from 0.85 to 0.95) is often used in conjunction with rolling windows to monitor the dynamic importance of assets over time.

In this study, we apply the PageRank algorithm to a network of six major

cryptocurrencies, using empirical correlation matrix and the converging nature of the Page Rank with the increase of damping factor is shown in Figure 3.16. It follows from Figure 3.16 that node with the highest page rank is consistent for all the damping factor values (BTC in this example) and when the damping factor increases, page rank of all the nodes converges to $1/\text{No.of nodes}$ (0.166 in this example).

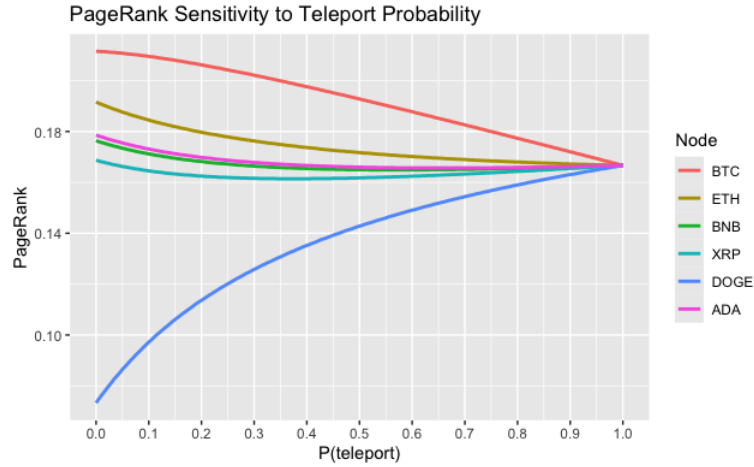


Figure 3.16: Effect of damping factor on Page rank of cryptocurrencies

The real world applications of PageRank is given in Chapter 4 and Chapter 5.

3.6.2 Stock Networks

This example is based on the US financial market and utilizes a diverse dataset consisting of 55 distinct assets from 13 sectors. Daily adjusted closing prices for these assets from January 1, 2015, to December 31, 2024, were sourced from

[Yahoo! Finance](#). In the data set considered in this study, degrees of freedom of returns of 55 stocks are calculated and it vary from 2.3 to 5.2, with the mean 3.4 and standard deviation of 0.43. First, four correlation networks which are based on stock returns, such as network based on Pearson correlation of stock returns (Return_cor), network based on Pearson correlation of volatility (Volatility), network based on data driven volatility correlation (DD Volatility) and network based on sign (positive or negative) of stock returns (Sign($\pm$)) are studied.

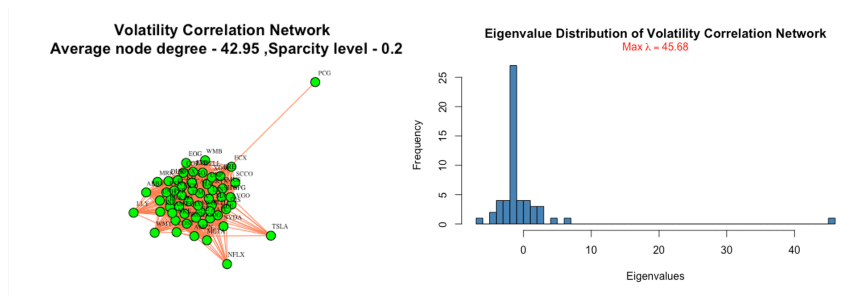


Figure 3.17: Volatility correlation network and eigenvalue distribution

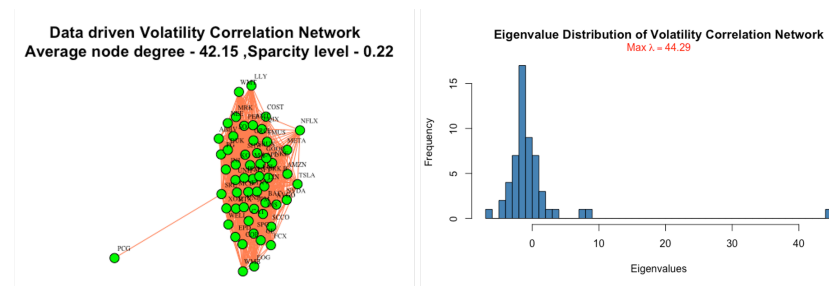


Figure 3.18: Data driven volatility correlation network and eigenvalue distribution

Figure 3.17 shows the volatility network and eigenvalue distribution based

on the empirical correlation of observed volatilities. Figure 3.18 shows the proposed data driven volatility network and its eigenvalue distribution based on the correlation defined using Theorem I. It follows from Table 3.5 that data driven volatility network is sparser than the volatility network, and the maximum eigenvalue is less than that of the volatility network. Figure 3.19 gives the log return network and Figure 3.20 gives the network based on the $\text{sign}(\pm)$ of log returns.

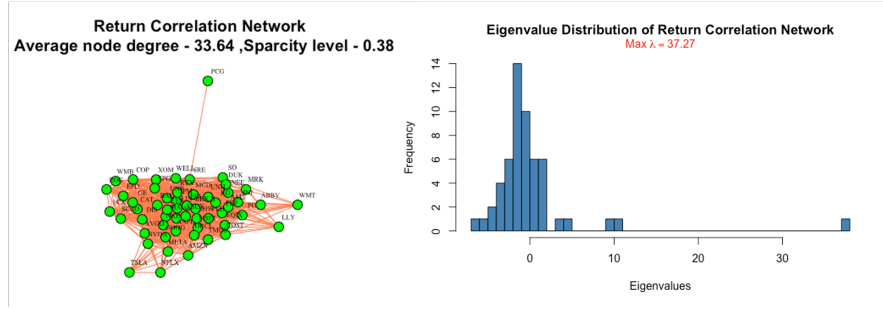
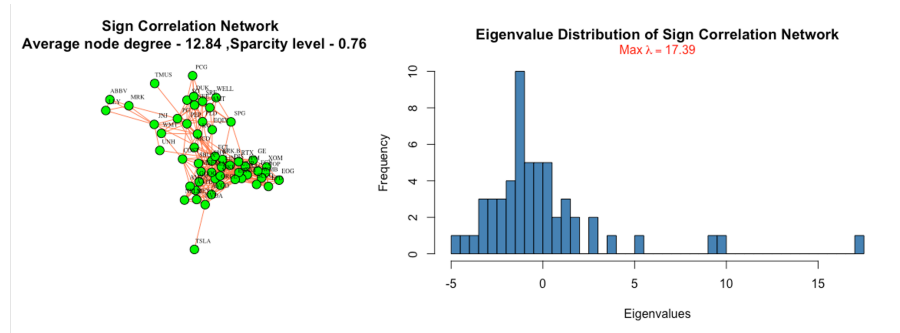


Figure 3.19: Return correlation network and eigenvalue distribution

Figure 3.20: $\text{Sign}(\pm)$ correlation network and eigenvalue distribution

Following Table 3.5 for 55 stocks, volatility network has the highest average node degree and highest maximum eigenvalue. This means volatility networks

Table 3.5: Network metrics of return networks

Correlation. Network	55 Stocks				S&P500 stocks		
	Thresh	Avg node degree	Sparsity level	Lambda max	Avg node degree	Sparsity level	Lambda max
Return_cor	0.30	33.40	0.38	37.27	265.27	0.43	311.87
Volatility	0.23	42.95	0.20	45.68	363.97	0.22	393.84
DD Volatility	0.26	42.15	0.22	44.29	296.67	0.36	335.73
Sign($\pm$)	0.27	12.84	0.76	17.39	175.44	0.62	232.95

are dense than other networks and volatilities of assets are more strongly related to each other than their returns. That is, more pairs of assets show significant co-movement in volatility. A dense volatility network may suggest a higher risk of spillover, where shocks to one asset's volatility could propagate more easily across the system.

The volatility, data driven volatility, return and sign($\pm$) correlation networks were drawn for the 500 stocks in S&P500 index daily adjusted closing prices from January 1, 2015, to December 31, 2024. After removing the missing values this data set is a 2515 (observations) x 468 (stocks) big data matrix. Due to the large number of nodes and edges the network visualization is not clear. Therefore, the network metrics are summarized in last three columns in Table 3.5. In S&P500 stocks, the volatility network has a higher average node degree and maximum eigenvalue. The lowest average node degree in sign($\pm$) network indicates that assets in the network show weaker or more independent directional behavior from each other.

3.7 Dynamic Price Correlation Networks

In financial network science, Pearson correlation (see for example, (Coscia, 2021)), a statistical tool used in quantifying the linear relationship between two variables, has been utilized in various applications. Its' most frequent use is the assessment of similarity between nodes in a network where nodes with high correlation are often considered to be structurally or functionally similar. Correlation matrices are sometimes used as adjacency matrices, in which weight of an edge between two nodes is determined by the correlation of their attributes. Functions of correlation are also used as distance measures in constructing financial networks (Saha et al., 2022)(Jothimani et al., 2018). Despite its popularity, the Pearson correlation has a major limitation as it only measures linear relationships between variables and highly sensitive to outliers. In financial market, asset prices do not follow simple, linear patterns. Instead, the market structure affected by economic indicators and other external factors such as geopolitical events as well as investor sentiment which often interact in non-linear ways. Correlation often fails to capture the full complexity of stock price movements (Coscia, 2021) .

Several studies have used correlation matrices in financial networks across various domains. For example, price correlation and differenced-price correlation networks were applied in algorithmic trading, showing superiority over co-integration methods in pairs trading (Ranathungage et al., 2024). Volatility correlation and fuzzy adjacency matrices were introduced for portfolio opti-

mization in [Saumyamala et al. \(2024\)](#), while the Toronto Stock Exchange's topological structure was analyzed using rolling correlations in minimum spanning tree method and planar maximally filtered graphs in [Jothimani et al. \(2018\)](#). However, these studies focus only on linear relationships. In order to address inherent non-linearity in financial markets, this study proposes a two-step approach: first fit linear (ARIMA) and neural network models (NNAR) to stock prices, and then, using their innovations to compute correlation matrices and construct networks.

Neural networks are widely used to capture complex, non-linear relationships in stock prices by learning from large data and detecting patterns missed by linear models. Previous studies, such as [Thavaneswaran et al. \(2019c\)](#), introduced neuro volatility models for forecasting log return volatility, while [Hoque et al. \(2024a\)](#) applied these models to forecast model risk using methods like Prophet, NNAR, and LSTM. The novelty of this study lies in introducing neuro correlations, which combine neural networks' ability to model non-linearity with financial networks' capacity to capture asset relationships.

3.7.1 Price Correlation Network

Pearson's correlation coefficient (ρ) quantifies the linear relationship between two variables in terms of strength and direction. Let $P_{i,t}$ and $P_{j,t}$ be the adjusted closing price of i^{th} and j^{th} assets respectively, which are time dependent series.

$$\rho_{P_i, P_j} = \frac{\sum (P_{i,t} - \bar{P}_i)(P_{j,t} - \bar{P}_j)}{\sqrt{\sum (P_{i,t} - \bar{P}_i)^2 \sum (P_{j,t} - \bar{P}_j)^2}}, \quad (3.14)$$

where $\bar{P}_i$ and $\bar{P}_j$ are the sample means of $P_{i,t}$ and $P_{j,t}$, respectively. Moreover, in equation (3.14) for each i , $(P_{i,t} - \bar{P}_i)$ are auto-correlated.

Price correlation network of the 55 stocks and its eigenvalue distribution is shown in Figure 3.21. It has a very high average node degree, low sparsity level and high maximum eigenvalue. Higher average node degree implies that network consists of large number of correlations including noisy correlations. The highly skewed eigenvalue distribution and the higher maximum eigenvalue means a few stocks drive the market risk.

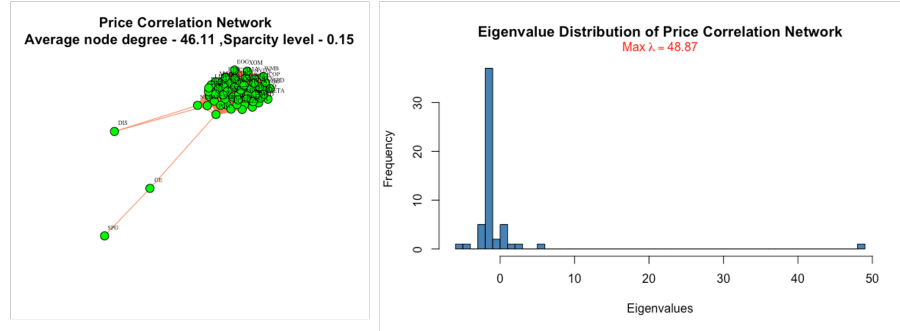


Figure 3.21: Price correlation network and eigenvalue distribution

3.7.2 ARIMA Innovation Correlation Network

The ARIMA innovation correlation network is obtained by first fitting ARIMA models to each series and then, obtaining the correlation of the resulting

innovation series. The innovation of the ARIMA model is obtained by the difference of the observed value and the fitted value ($P_{it} - \hat{P}_{it}^{ARIMA}$). These innovations of ARIMA models has four properties: First, innovations are not auto-correlated and if any auto-correlation presents that imply that the fitted model did not captured all the underlying patterns in the data. Secondly, innovation are normally distributed. Finally, innovations have constant variance and zero mean (Hyndman and Athanasopoulos, 2018).

Let $i_{X,t}$ and $i_{Y,t}$ are innovations of the two time series models X_t and Y_t with n observations. Then, the innovation correlation is given by ρ_{i_X, i_Y} (Wooldridge, 2010):

$$\rho_{i_X, i_Y} = \frac{\sum_{t=1}^n (i_{X,t} - \bar{i}_X)(i_{Y,t} - \bar{i}_Y)}{n\sigma_{i_X}\sigma_{i_Y}}$$

where $\bar{i}_X$ and $\bar{i}_Y$ are the mean of innovations obtained by fitting ARIMA models for non-stationary series of X and Y respectively.

The ARIMA innovation correlation network of the 55 stocks and its eigenvalue distribution is shown in Figure 3.22. It has a very low average node degree, high sparsity level and low maximum eigenvalue compared to price correlation networks. It implies that the network retains few but strong correlations among stocks.

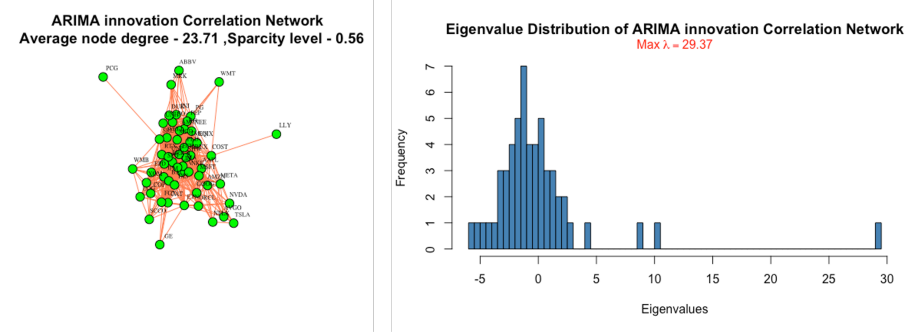


Figure 3.22: ARIMA innovation correlation network and eigenvalue distribution

3.7.3 Neuro Correlations based on NNAR Model

The neuro correlation network is obtained by first fitting NNAR models (Hyndman and Athanasopoulos, 2018) to each series and then, obtaining the correlation of the resulting innovation series. The correlation between innovations ($P_{it} - \hat{P}_{it}^{NNAR}$) of series X_t and Y_t is defined as the neuro correlation between X_t and Y_t . Neuro correlation network of the 55 stocks and its eigenvalue distribution is shown in Figure 3.23. It has the lowest average node degree, highest sparsity level and lowest maximum eigenvalue compared to price and ARIMA innovation correlation networks. It implies that the neuro correlation network retain most important correlations and hence, more informative.

The following Table 3.6 summarize the network metrics of three price networks of 55 stocks and 500 stocks from S&P500 stock index. It follows from Table 3.6 that for Price.cor network, the average node degree is very high, and sparsity is low. This means the network is densely connected. In 55 stocks, the average node degree of price network is 46.11, which means

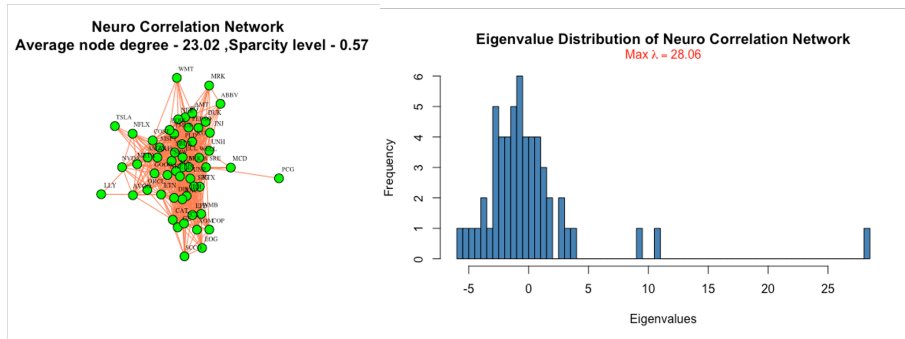


Figure 3.23: Neuro correlation network and eigenvalue distribution

on average, each stock is connected to 46 other stocks out of the 54 possible others (85% of possible links are present per node). So the network is densely connected suggesting strong overall market coherence. ARIMA innovation and neuro correlation networks have a lower average node degree than the price network, which means innovations are sparsely connected. That implies that most of the linear and nonlinear relationships in the market are captured by the ARIMA and NNAR models. However, in 500 stocks, average node degree of price correlation network is 296, which means each stock connects to 296 others out of a possible 467 (63%). The 500-stock price network based on *S&P500* stocks is significantly sparser, implying more structural diversity or selective correlation patterns.

Figure 3.24 shows the dynamic nature of the price network over the year 2024. It follows from the Figure 3.24 that in particular months the network is dense (May/Aug/Dec) and in particular months the network is sparse (Jan/Oct/Nov), which differentiates the months with strong and weak correlation and co-

Table 3.6: Network metrics of price networks

Network	55 Stocks				S&P500 stocks		
	Thresh	Avg node degree	Sparsity level	lambda max	Avg node degree	Sparsity level	lambda max
Price_cor.	0.53	46.11	0.15	48.87	296.08	0.37	347.98
ARIMA_cor.	0.29	23.71	0.56	29.37	–	–	–
Neuro_cor.	0.29	23.02	0.57	28.06	–	–	–

movement in the stock market.

3.8 Conclusion

In this chapter, first three different fuzzy networks were constructed using empirical correlation, DDEWMA correlation, and cosine similarity. Then, it is shown that same network can be obtained at different parameter values using interval of adjacency matrices of cryptocurrency data. The novelty of this study is introducing non-linear fuzzy volatility network. Moreover this chapter investigates data-driven financial correlation networks with a particular focus on modeling log returns using a data-driven t -distribution. A key contribution is showing that the covariance matrix of absolute log returns, used in volatility networks can be expressed as a function of the sign correlation and the covariance matrix of log returns. Using data from 55 stocks across 13 sectors and S&P500 (big data), we compare four different networks based on correlations of log returns, volatility of log returns, data-driven volatility of log returns and $\text{sign}(\pm)$ of returns. Three dynamic networks

based on price correlations, ARIMA innovation correlations of the price and neuro innovation correlations of the price are also considered. All seven networks are evaluated using metrics such as average node degree, sparsity, and maximum eigenvalue. Unlike the existing work, the novelty of this work is demonstrating a relationship between data driven volatility networks and recently proposed volatility correlation networks without using distributional properties. Autocorrelation networks and Granger causal networks are also discussed in some detail.

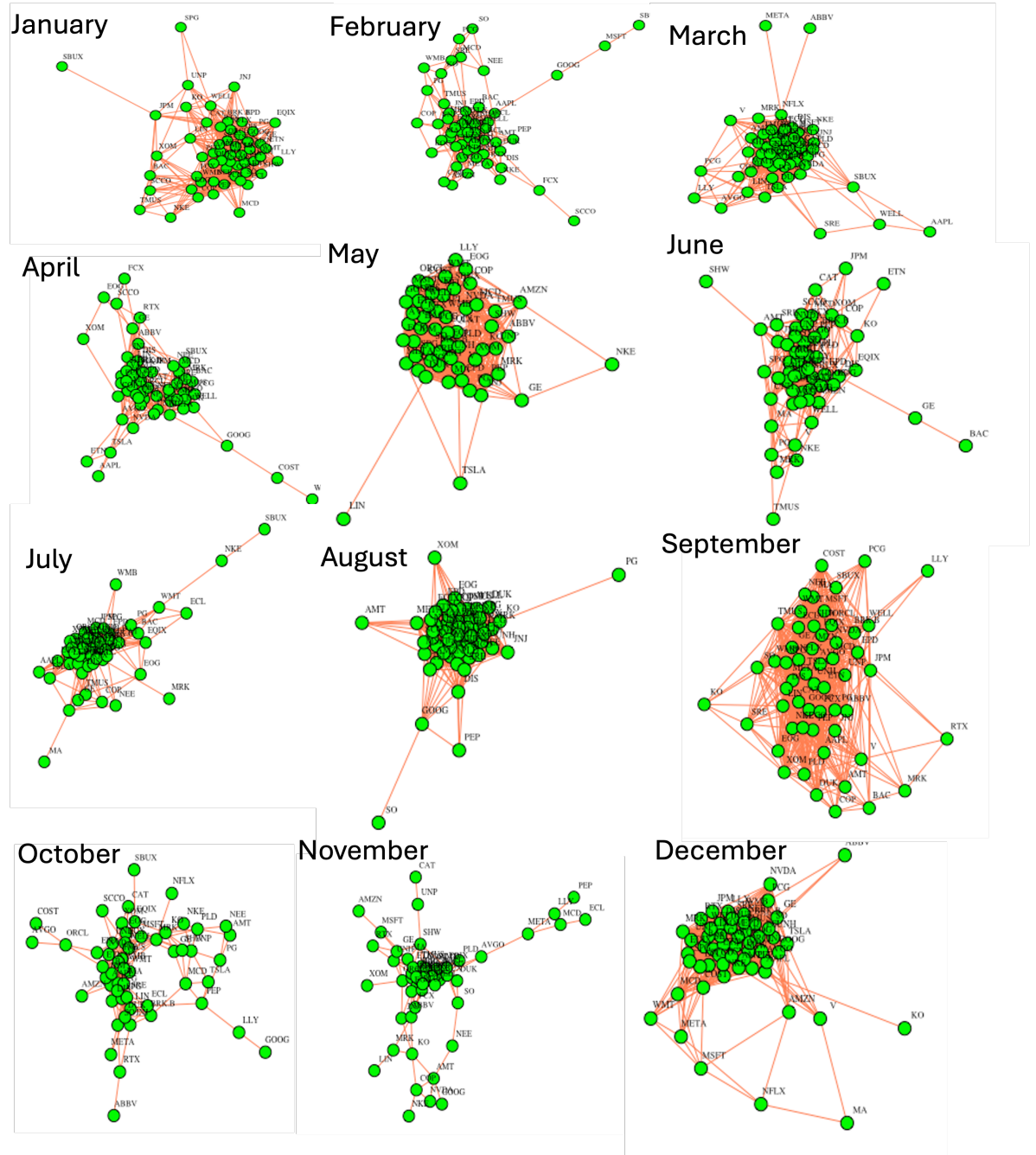


Figure 3.24: Dynamic (monthly) price networks January - December 2024

Chapter 4

Portfolio Optimization

Portfolio optimization is a fundamental task in finance that involves allocating capital across a set of assets to maximize expected return while minimizing risk. The classical framework for this problem is the mean-variance model introduced by [Markowitz \(1952\)](#), which laid the foundation for modern portfolio theory. This framework constructs portfolios that either maximize expected return for a given level of risk or minimize risk for a given level of expected return.

Let μ denote the vector of expected returns and Σ the covariance matrix of n number of asset returns. The standard mean-variance optimization problem can be formulated as:

$$\min_w \quad w^T \Sigma w \quad \text{subject to} \quad w^T \mu = \mu_p, \quad \text{and} \quad \sum_{i=1}^n w_i = 1, \quad (4.1)$$

where w is the vector of portfolio weights and μ_p is the target portfolio return.

Despite its theoretical appeal, the classical model is highly sensitive to input estimation errors, particularly in high-dimensional settings where the number of assets is large relative to the available historical observations. The sample covariance matrix used in practice tends to be noisy, leading to unstable portfolio weights and poor out-of-sample performance. Furthermore, the model assumes normally distributed and stationary returns, assumptions that often do not hold in real-world markets.

Another critical challenge in portfolio optimization is the reliable estimation of expected returns and the covariance matrix. While estimating returns is notoriously difficult, even the estimation of covariances is affected by noise in high dimensions, making the sample covariance matrix ill-conditioned or singular. Without regularization or dimension reduction, optimization under such conditions becomes unreliable.

To address these issues, modern portfolio optimization methods incorporate alternative risk measures, robust statistical estimation, and machine learning tools. One particularly promising approach is the integration of clustering techniques. Clustering groups assets based on similarity in return behavior, which can help reduce dimensionality and improve the stability of covariance estimates. Moreover, clustering captures latent structures such as sectoral behavior and asset groupings that are often ignored in traditional models. As financial markets grow in complexity and scale, clustering provides a data-driven, interpretable tool for enhancing portfolio construction and robustness.

Random Matrix Theory (RMT) provides a principled approach to filter

noise from the sample covariance matrix, especially in high-dimensional settings. By distinguishing signal from noise in the eigenvalue spectrum, RMT enhances the stability and reliability of covariance estimation. This leads to more robust portfolio allocations and improved out-of-sample performance.

Followed by this motivation this chapter includes two main sections:

- Clustering for enhanced portfolio Optimization
- RMT filtering for enhanced portfolio optimization

4.1 Clustering for Enhanced Portfolio Optimization

Clustering is a type of unsupervised machine learning that groups objects based on similarity without pre-labeled data. In finance, this similarity is often derived from return correlations or other distance measures. Clustering aligns well with the investor's need to diversify risk across uncorrelated or minimally correlated assets. Clustering, by identifying groups of assets with similar behavior, provides a way to summarize complex interdependencies among assets. It allows for more interpretable portfolio construction by enabling allocation at the group level, thereby fostering diversification and reducing estimation risk. Clustering can uncover hidden structures in financial markets, reduce dimensionality, and enhance the stability of covariance matrices by

grouping similar assets. Additionally, clustering facilitates dynamic portfolio strategies that adapt to evolving market conditions. These benefits make clustering a promising tool for improving portfolio optimization outcomes.

The global financial markets have witnessed rapid growth in data availability, diversity of asset classes, and computational capabilities. Yet, investors still grapple with problems such as overfitting in optimization, excessive turnover, and fragility under stress conditions. Traditional portfolio optimization models assume stationarity in correlations and returns an assumption that often fails during market crises. The market players often assume that stocks within the same industry sector exhibit similar behaviors, leading to the construction of network graphs within each sector (Jothimani et al., 2018; Saha et al., 2022). However, within a sprawling industry sector such as technology, where assets abound, their behaviors can vary significantly. Hence, instead of networking all stocks within an industry or sector, it is beneficial to identify the groups of stocks behaving similarly within a sector. Motivated by this, this study aims to combine the clustering techniques and financial networks, which can be used to unveil clusters of stocks exhibiting similar behaviors and risk-return profiles that can serve as valuable tools for portfolio diversification (Tumminello et al., 2007). In finance, clustering has been used to group stocks based on return correlations or co-movements, identify sectors or regimes in the market. Then, improve risk management and portfolio construction.

4.1.1 Distance-Based Clustering Methods

k-means Clustering

k-means is a widely used partitioning algorithm that aims to divide a set of n observations into k disjoint clusters $\{C_1, C_2, \dots, C_k\}$ such that the within-cluster variance is minimized.

Each cluster is represented by the centroid (mean) of its data points. The algorithm seeks to minimize the total squared Euclidean distance between each point and its assigned cluster centroid:

$$\min_{\{C_1, \dots, C_k\}} \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$$

where μ_i is the centroid of cluster C_i , computed as the mean of the points in C_i .

The k-means algorithm typically follows these steps:

1. Initialize k centroids randomly.
2. Assign each point to the nearest centroid based on Euclidean distance.
3. Recalculate centroids as the mean of assigned points.
4. Repeat steps 2 and 3 until convergence (no change in assignments or centroids).

k-means is efficient and easy to implement, but it is sensitive to initial centroid positions and outliers. It performs best with continuous, spherical, and similarly sized clusters. The input for the k-means clustering algorithm could be any distance based matrix or similarity matrix. The following are some possible distance matrices.

1. Distance matrix derived from correlation
2. Distance matrix derived from mean and standard deviation (SD)
3. Distance matrix derived from mean and mean absolute deviation (MAD)
4. Distance matrix derived from Mean, SD and kurtosis

Clustering based on distance matrix derived from Mean and SD considers two features: expected return and variance of stocks. The reason for choosing these two features is that investors have varying preferences for investment strategies, from risk-taking to risk aversion. Hence, it is important to identify stocks that provide high returns with high risk or high returns with moderate or low risk. As an example 50 technology stocks were clustered using k-means clustering method. Figure 4.1 depicts the 6 clusters identified with k-means clustering.

k-medoids Clustering

k-median and k-medoids are both robust clustering methods closely related to k-means that minimize dissimilarity between points and their assigned cluster

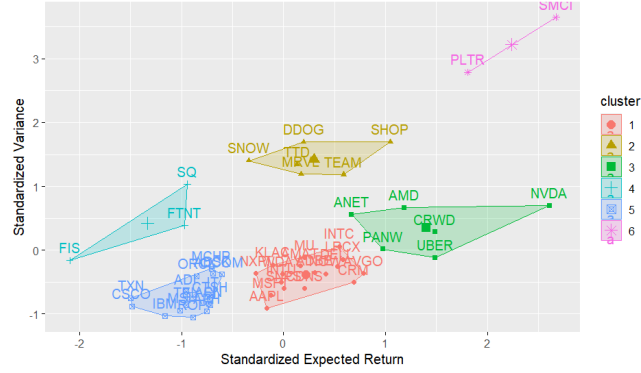


Figure 4.1: k-means clustering of 50 technological stocks using standardised expected return and variance

centers. Instead of using the mean of cluster points as a centroid (as in k-means), k-median and k-medoids selects centers of clusters. The key difference lies in how the cluster center is defined. k-median uses the coordinate-wise median, which may not correspond to an actual data point and is typically optimized using the Manhattan (L1) distance. In contrast, k-medoids selects actual data points (medoids) as cluster centers and can accommodate any distance or dissimilarity metric. k-medoids is particularly suitable for financial data because it is less sensitive to outliers and skewed distributions, and it naturally handles non-Euclidean distance measures such as correlation or return-based dissimilarities. By selecting representative assets as medoids, the method aligns well with interpretable portfolio construction and enhances robustness in high-dimensional financial applications.

Given a set of n observations, the goal of the algorithm is to partition the data into k clusters $\{C_1, C_2, \dots, C_k\}$ such that the sum of dissimilarities

between each point and its assigned medoid is minimized. Formally, the objective is:

$$\min_M \sum_{i=1}^k \sum_{x \in C_i} d(x, m_i)$$

where $M = \{m_1, m_2, \dots, m_k\}$ are the selected medoids and $d(x, m_i)$ is a distance metric (e.g., Euclidean, Manhattan) between data point x and medoid m_i .

The most common algorithm for k-medoids is the Partitioning Around Medoids (PAM) algorithm, which iteratively selects medoids and reallocates points to minimize the total cost. k-medoids is particularly useful for clustering categorical or mixed-type data where the mean is not a meaningful representation.

Hierarchical Clustering

The hierarchical clustering approach involves constructing a cluster hierarchy by iteratively merging or splitting clusters based on a similarity measure such as the Sharpe ratio until a desired number of clusters is reached. The Sharpe ratio measures the risk-adjusted return of an investment or portfolio, calculated as the ratio of the average excess return of the investment to the standard deviation of returns.

$$\text{Sharpe Ratio} = \frac{\mathbb{E}[R_p] - R_f}{\sigma_p}, \quad (4.2)$$

where R_p denotes the portfolio return, R_f is the risk-free rate, and σ_p is the standard deviation of the portfolio returns.

Figure 4.2 shows five optimal clusters based on the Sharpe ratio hierarchical clustering.

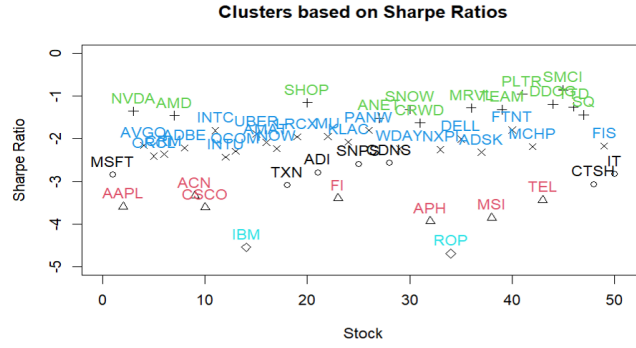


Figure 4.2: Hierarchical clustering of 50 technological stocks using Sharpe ratio

Spectral Clustering

Spectral clustering is a powerful method that leverages the eigen-decomposition of a similarity graph constructed from asset distances, making it effective for capturing complex and nonlinear relationships among assets.

The procedure begins by computing the Pearson correlation matrix of asset returns. This correlation matrix is then transformed into a similarity matrix using a Gaussian kernel. From the similarity matrix S , we construct the

unnormalized graph Laplacian as $L = D - S$, where D is the diagonal degree matrix. The top- k eigenvectors of the Laplacian matrix L are then, extracted to form a low-dimensional representation of the data. Clustering is performed in this low-dimensional space using the k -means algorithm. One representative stock is selected from each cluster to construct a diversified portfolio, and equal weights are assigned to each selected asset.

Spectral clustering effectively reveals latent structures in stock return behavior and provides a data-driven approach to portfolio diversification. Cluster-based portfolio can potentially reduce risk and enhance robustness by allocating capital across statistically distinct asset groups.

4.1.2 Cluster based Portfolio Optimization

In cluster based portfolio optimization, first optimal number of clusters is decided and then, different clustering approaches are used to cluster the observations. In the following example, we consider 55 stocks from different sectors and as the inputs we use different distance matrices and similarity matrices based on train data (from 2010-01-01 to 2023-12-31). Then, we consider each cluster as a portfolio and find the optimal weights by maximizing the Sharpe ratio. After obtaining the weights, we calculate the cumulative returns and Sharpe ratio for the test period (from 2024-01-01 to 2024-12-31). The cumulative return plots for the cluster portfolios are given in Figure 4.3 and Figure 4.4.

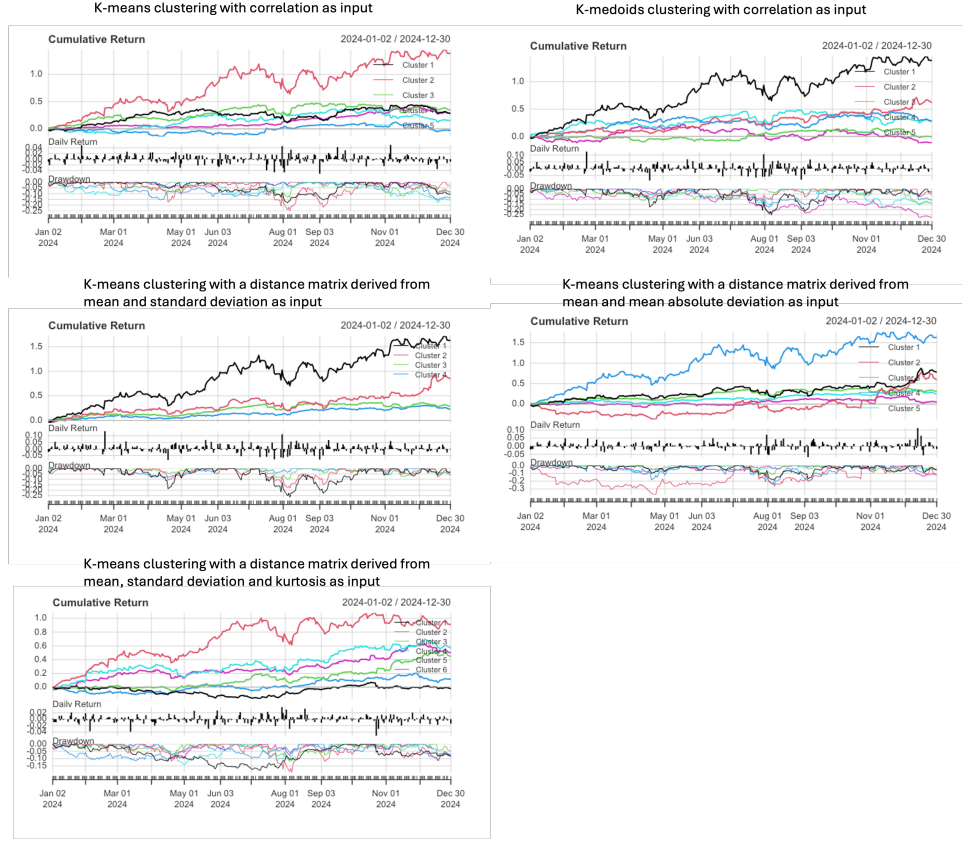


Figure 4.3: Cumulative return plots obtained from five distance based clustering methods

The portfolio which yield highest Sharpe under each clustering method is reported in Table 4.1. Equal weighted portfolio where equal weight is assigned to all 55 stocks is used as the benchmark portfolio.

Following Table 4.1 it can be observed that all the clustering approaches outperform the equal weighted portfolio in terms of Sharpe ratio.

Table 4.1: Cluster based portfolio summary

Clustering method	Algorithm	Input	Portfolio	Stocks in the portfolio	Test period - 2024		
					Annualized Return	Volatility	Sharpe Ratio
Distance based Clustering	k means	Distance matrix derived from correlation	Cluster 2	AAPL, NVDA, MSFT, AVGO, AMZN, TSLA, GOOG, META, NFLX	1.391	0.440	3.053
	k medoids	Distance matrix derived from correlation	Cluster 1	AAPL, NVDA, MSFT, AMZN, TSLA, GOOG, META, NFLX	1.398	0.442	3.052
	k means	Distance matrix derived from mean and SD	Cluster 1	NVDA, TSLA, FCX, PCG	1.634	0.466	3.399
	k means	Distance matrix derived from mean and SD	Cluster 4	NVDA, NFLX, COP, EOG, SCCO	1.638	0.459	3.459
	k means	Distance matrix derived from mean SD kurtosis	Cluster 2	NVDA, AVGO, BRK-B, V, MA, BAC, LLY, UNH, JNJ, ABBV, DIS, UNP, COST, PG, KO, PLD, AMT, NEE	0.914	0.317	2.726
Spectral clustering	k means	Eigen decomposition of similarity matrix	Cluster 4	AAPL, NVDA, AVGO, ORCL, AMZN, TSLA, MCD, NKE, SBUX, LLY, UNH, JNJ, ABBV, MRK, GOOG, META, NFLX, TMUS, DIS, CAT, GE, RTX, UNP, WMT, COST, PG, KO, XOM, COP, EOG, WMB, EPD, LIN, SHW, SCCO, FCX, PLD, AMT, EQIX, WELL, SPG, NEE, SO, DUK, SRE, PCG	0.912	0.319	2.705
	K medoids	Eigen decomposition of similarity matrix	Cluster 1	AAPL, NVDA, AVGO, ORCL, AMZN, TSLA, MCD, NKE, SBUX, LLY, UNH, JNJ, ABBV, MRK, GOOG, META, NFLX, TMUS, DIS, CAT, GE, RTX, UNP, WMT, COST, PG, KO, XOM, COP, EOG, WMB, EPD, LIN, SHW, SCCO, FCX, PLD, AMT, EQIX, WELL, SPG, NEE, SO, DUK, SRE, PCG	0.912	0.319	2.705
Equally weighted portfolio	All Stocks				0.248	0.105	1.905

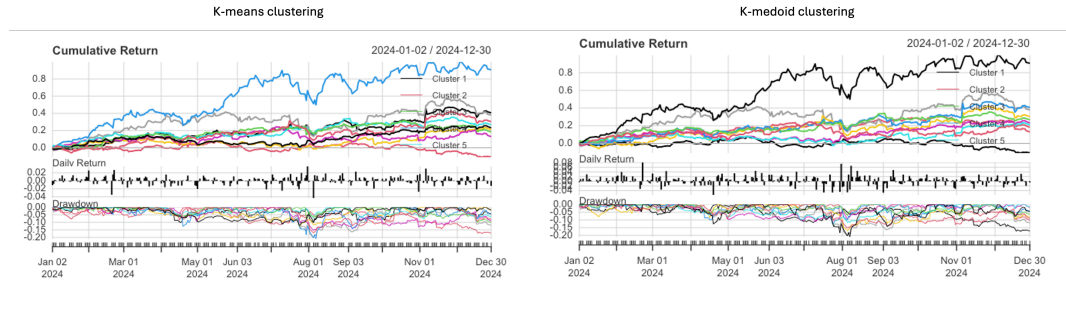


Figure 4.4: Cumulative return plots obtained from two spectral clustering methods

4.1.3 Network based Clustering

Financial assets can be represented as nodes in a network, where edges represent correlations or other similarity measures. Clustering in this context often involves community detection techniques. Network-based clustering enhances interpretability and allows for visual inspection of market structure. Network communities reveal underlying structures and offer insights into the behavior and dynamics of the system. Community detection is the process of identifying clusters of nodes that are more densely connected to each other than to the rest of the network. This study employs three community detection methods, all of which aim to maximize the modularity score to uncover meaningful groupings. Modularity is a measure of the strength of the division of a network into communities (modules). Modularity compares the density of edges inside communities to that between communities. High modularity indicates that the community structure is significantly better than the modularity score of

random separation. In other words, the network is well-separated into dense intra-community edges and sparse inter-community edges.

Cluster Fast Greedy (CFG) Method

The CFG method is a hierarchical, agglomerative algorithm that merges communities to maximize modularity, a measure comparing intra-community and inter-community edge densities. This method is computationally efficient for small to medium networks but may struggle with very large or complex structures due to its reliance on global optimization.

Louvain Method

The Louvain method is a two-phase modularity-based algorithm suitable for large networks. It builds communities through local modularity optimization and then, aggregates them into super-nodes for further refinement. While efficient and widely used, this method may overlook smaller communities due to resolution limits.

Leiden Method

The Leiden method improves upon Louvain by adding a refinement phase to ensure communities are well-connected and non-fragmented ([Anuar et al.](#),

2021). Leiden method offers better modularity optimization, faster convergence, and is more robust for large, complex networks.

The following Table 4.2 gives the modularity score of community structures calculated for seven networks discussed in Section 3.6 and Section 3.7 using three community detection methods. The networks are drawn from price and log returns of 55 stocks from January 1, 2015, to December 31, 2024.

Table 4.2: Modularity score of communities detected in networks

Algorithm	Price Networks			Return Networks			
	Price_cor	ARIMA_cor	Neuro_cor	Return_cor	Volatility	DD Volatility	Sign($\pm$)
CFG	0.014	0.180	0.190	0.080	0.028	0.140	0.380
Louvain	0.015	0.193	0.002	0.110	0.033	0.158	0.407
Leiden	0.002	0.131	0.141	0.040	0.000	0.048	-0.012

It follows from Table 4.2 that the DD Volatility network has a higher modularity score than the commonly used volatility network, implying stronger, well-separated communities. The Louvain method that yields the highest modularity score in majority of the networks is best at detecting strong community structures. Modularity score of Sign($\pm$) network using Leiden method is negative which indicates detected community structure is worse than random, that is, the grouping of nodes has fewer within-community edges than would be expected by chance in a random network with the same degree distribution.

4.1.4 Dynamic Clustering

Financial markets are dynamic, and static clustering may fail to capture temporal changes in asset relationships. Moreover stock prices are not stationary

and static clustering is less informative. Recent research has explored dynamic clustering methods, rolling window (weekly/monthly/annually) clustering to adapt to recent data and visualize using time-varying graphical models and regime-switching models with cluster-specific dynamics. These methods allow for more responsive portfolio strategies, especially in volatile environments.

Figure 3.24 given in Section 3.7 shows the monthly dynamic nature of the price network over the year 2024. For the same networks Figure 3.24 shows the respective community graphs which also differ from month to month. This shows that the stock clusters are dynamic and the cluster membership is time-dependent.

4.1.5 Network based Portfolio Optimization

Unlike the cluster based portfolio optimization where clusters are considered as portfolios, in network based portfolio optimization we follow a novel approach introduced by Saumyamala et al. (2024) where portfolio is constructed by combining one stock from each cluster. By this approach we aim to diversify the portfolio. Afterwards portfolio weights are optimized to minimize the risk (volatility estimates using sign correlation) while maximizing the returns. In order to decide which portfolio is more profitable we use the cumulative returns as the investors main concern is on the returns. Recently, Thavaneswaran et al. (2021b) introduced novel portfolio risk measure using Sign correlations and Volatility correlations and by Bowala and Singh (2022) applied these data

driven risk measures in portfolio optimization.

This thesis study is based on US financial market and data set consist of 30 assets:

- **5 Cryptocurrencies:** Bitcoin (BTC-USD), Ethereum (ETH-USD), Ripple (XRP-USD), Binance Coin (BNB-USD), Solana (SOL-USD)
- **5 Commodities:** Crude Oil Futures (CL=F), Corn Futures (ZC=F), Gold Futures (GC=F), Natural Gas Futures (NG=F), Soybean Futures (ZS=F).
- **20 stocks from different sectors and indices:** Apple Inc.(AAPL), Microsoft Corp(MSFT), NVIDIA corporation (NVDA), Broadcom Inc. (AVGO), Oracle Corporation (ORCL), Texas Gulf Energy (TXGE), Exxon Mobile corporation (XOM), Chevron Corporation (CVX), ConocoPhillips (COP), EOG resources (EOG), Berkshire Hathaway Inc. (BRK-B), JPMorgan Chase & Co. (JPM), S&P 500 index (SPY), eBay Inc (EBAY), Adobe Inc (ADBE), SPDR Gold Trust (GLD), VanEck Gold Miners ETF (GDX), International Business Machines Corporation (IBM), iShares MSCI Australia ETF (EWA), iShares North American Natural Resources ETF (IGE)

The daily adjusted closing price of assets from 2021-01-01 to 2023-12-31 was obtained from [Yahoo! Finance](#). In here, we consider rolling correlation

and hence, the data set divided into rolling windows. The full data set include 736 data points (3 years) and it is divided into 10 rolling windows such that length of a window is 180 (approximately 3 quarters) and rolling window size is 60 (approximately 1 quarter) as follows [1:180, 61:240, 121:300, 181:360, 241:420, 301:480, 361:540, 421:600, 481:660, 541:720]. Then, the experiments were done for each window separately. By following the equations and methods given in Section 3.7 we obtain price correlation network, ARIMA innovation correlation network and Neuro correlation network as given in Figure 4.6. Once the networks are obtained, using the cluster fast greedy algorithm communities of similarly behaving assets in the network were detected. The respective community graphs for window 1 are given in Figure 4.7. From Figure 4.7, it is clearly observed that structure of the correlation network is different from other two innovation correlation networks. Although the structure of the two innovation networks are quite similar, the communities identified are different from each other. In ARIMA network only two communities are detected with sizes 16 and 14 where both communities include atleast one cryptocurrency, commodity or stock. In contrast, NNAR network detect three communities where all the cryptocurrencies clustered to a one community.

As the next step, minimally connected network for each community was formed and page ranking algorithm was applied (Refer Section 3.6.1 for more details). The highest page rank node/asset was selected to the portfolio considering it as the most linked or influential asset in the community. In

networks where only two communities are detected the two highest page rank assets were selected as it is known that portfolio with only two assets are not profitable in practice. In communities containing two assets with same highest page rank both assets were selected. We applied the same procedure and constructed three portfolios each for all ten windows (Table 4.3).

Table 4.3: Portfolio selected from correlation networks

Window	Price correlation portfolio	ARIMA innovation correlation portfolio	Neuro correlation portfolio
1	NG.F,EOG,GC.F	COP,EWA,SPY,NVDA	IGE,EBAY,BTC.USD
2	GDX,BNB.USD,JPM	IGE,GDX,ETH.USD ,AAPL	IGE,EBAY,ETH.USD
3	EBAY,CVX,BTC.USD,ETH.USD	SPY,EBAY,COP,CVX	MSFT ,AAPL,EOG,IGE
4	SPY,BTC.USD,GC.F,BRK.B	SPY,BTC.USD,GC.F,BRK.B	SPY,AAPL,CL.F,COP
5	MSFT,XOM,XRP.USD	SPY,NVDA,CL.F,EOG	AAPL,EWA,CL.F,COP
6	XRP.USD,BTC.USD,JPM,XOM	CL.F,IGE,SPY,BRK.B	SPY,EOG,NG.F
7	JPM,BRK.B,BTC.USD,MSFT	IGE,EOG,SPY,MSFT	SPY,EWA,IGE,EOG
8	EWA,BTC.USD,IGE	BTC.USD,EWA,BRK.B,IGE	BRK.B,IGE,EWA,BNB.USD
9	ZC.F ,EOG,ETH.USD	EOG,SPY,GDX,ETH.USD	IGE,ADBE,GDX,ETH.USD
10	IGE,GDX,JPM	AAPL,IGE,GDX,XRP.USD	SPY,JPM,GDX,ETH.USD

The cumulative returns based on volatility correlation of the formed portfolios were calculated. In cumulative return calculation of window 1, asset returns of first three quarters (January - September) are used for training and the fourth quarter (October - December) are used for testing. When observing Figure 4.8, the portfolio selected from ARIMA and NNAR innovation correlation based networks has higher cumulative returns compared to the correlation network portfolio. To understand this behavior, the same cumulative return calculation was done for all 10 windows and plots are given in Figure 4.9. In window 2-4 (2022.01.01 -2022.09.30) neuro correlation network based portfolio outperform the other two portfolios and, in window 5-7 (2022.10.01 -2023.06.30) correlation network based portfolio outperforms. Again in window 8-9 (2023.07.01

- 2023.12.31) neuro correlation portfolio perform better. Interestingly, these turning points coincided with recent events, S&P 500 index which reflect the performance of the U.S. stock market entering a bull market in October 2022 and June 2023 [Forbes.com](https://www.forbes.com). However, The precise cause requires further investigation, but it can be assumed that during periods when the market exhibits a more non-linear structure, portfolios based on neuro correlations tend to perform better, whereas portfolios based on correlation networks outperform during more linear market conditions.

4.2 RMT Filtering for Enhanced Portfolio Optimization

Recently, a novel approach to financial networks and portfolio optimization is proposed in [Saumyamala et al. \(2025\)](#) by introducing pairwise neuro correlations, which capture non-linear relationships between assets using neural network autoregression (NNAR) models. [Zhu et al. \(2020\)](#) proposed the DDEWMA covariance model for estimating the covariance matrix of asset returns. It has been shown that the DDEWMA model, which accounts for volatility clustering, outperforms the empirical covariance model.

Covariance matrices are fundamental to financial network analysis but are often affected by the noise, especially in high-dimensional settings. RMT filtering offers a robust way to extract meaningful structure from noisy data

(Kyriakopoulos et al., 2009). Further Laloux et al. (2005) highlighted the use of the Marcenko-Pastur distribution to separate true correlations from noise, enhancing portfolio optimization. Filtered covariance matrices form the basis for Laplacian matrices, which encode network structure and enable spectral analysis. Makled and Cheng (2024) showed that eigenvalue dispersion in Wishart matrices varies with covariance structure and degrees of freedom, enhancing stability in financial analysis.

In this section, neuro covariance models Saumyamala et al. (2025) and data driven covariance models Zhu et al. (2020) are used to construct networks and discuss the average node degree reduction in details. Further RMT theory is used to denoise the covariance matrices and then, construct filtered covariance matrices. An algorithm to obtain RMT filtered covariance matrix for t distributed data is also given. Filtered and non filtered covariance matrices are used in portfolio optimization and the portfolios are compared based on Sharpe ratio. Experiments were conducted using simulated data as well as real stock price data.

4.2.1 Marchenko-Pastur Distribution

The Marchenko-Pastur (MP) law describes the asymptotic distribution of eigenvalues of a sample covariance matrix. Let X be an $n \times p$ random matrix whose entries are i.i.d. normal random variables with mean zero and variance

one or t -distributed random variable with mean zero and variance $\nu/\nu - 2$.

The sample covariance matrix is given by:

$$C = \frac{1}{p} X X^\top.$$

As $n, p \rightarrow \infty$ with a fixed aspect ratio $Q = \frac{n}{p}$, the eigenvalues of C follow the Marchenko-Pastur distribution with density function:

$$\rho_{\text{MP}}(\lambda) = \frac{1}{2\pi Q \lambda} \sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}, \quad \lambda \in [\lambda_-, \lambda_+],$$

where the support is defined by:

$$\lambda_{\pm} = \left(1 \pm \sqrt{Q}\right)^2.$$

Moments of the Marchenko-Pastur Distribution

The moments of the MP distribution are given by:

$$m_k = \sum_{j=0}^{k-1} \frac{1}{j+1} \binom{k}{j} \binom{k-1}{j} Q^j,$$

where:

- m_k is the k -th moment,

- $Q = \frac{n}{p}$ is the aspect ratio,
- $\binom{k}{j}$ is the binomial coefficient.
- First moment (mean): $m_1 = 1$
- Second moment (variance): $m_2 = 1 + Q$
- Third moment: $m_3 = 1 + 3Q + Q^2$
- Fourth moment: $m_4 = 1 + 6Q + 6Q^2 + Q^3$

$$\nu = Q^3 + 3Q^2 - 3Q \quad (\text{Skewness})$$

$$\kappa = Q^3 + 2Q^2 - 1 \quad (\text{Kurtosis})$$

The mean $m_1 = 1$ confirms that the MP law is centered around 1. The variance $m_2 - m_1^2$ depends on the aspect ratio Q , indicating the spread of eigenvalues. Higher-order moments determine properties such as skewness and kurtosis.

Modified Theoretical Moments for t -distributed Data

In order to account for the heavier tails of the t distribution, the MP moments are modified using a variance correction factor:

$$m_k = \sum_{m=0}^{k-1} \frac{1}{m+1} \binom{k}{m} \binom{k-1}{m} \left(\frac{\nu}{\nu-2} \right)^k Q^m,$$

where:

- ν is the degrees of freedom,
- $\frac{\nu}{\nu-2}$ is the variance correction factor.
- First moment:

$$m_1 = \left(\frac{\nu}{\nu-2} \right)$$

- Second moment:

$$m_2 = \left(\frac{\nu}{\nu-2} \right)^2 (1 + Q)$$

- Third moment:

$$m_3 = \left(\frac{\nu}{\nu-2} \right)^3 (1 + 3Q + Q^2)$$

- Fourth moment:

$$m_4 = \left(\frac{\nu}{\nu-2} \right)^4 (1 + 6Q + 6Q^2 + Q^3)$$

The simulation results of theoretical and empirical moment comparison for normally and t -distributed data are given in Appendix [C](#).

4.2.2 Covariance Models and Financial Networks

Covariance matrix estimation plays a crucial role in financial network modeling, particularly in portfolio optimization. The challenge lies in accurately estimating the data-driven covariance structure from finite and often noisy return data.

RMT Filtered Covariance Matrices

The following Algorithms 5 and 6 are used to obtain the RMT-filtered covariance matrices with normally distributed and data driven t distributed returns respectively. The RMT-filtered covariance matrix is constructed by first, removing eigenvalues that fall within the kurtosis dependent Marčenko-Pastur bounds and then, reconstructing the covariance matrix using the filtered eigenvalues and eigenvectors. This algorithm ensure that noise-dominated eigenvalues are replaced, improving stability and robustness in covariance estimation. Moreover, the networks constructed based on RMT-filtered covariance matrices turn out to be more resilient.

Neuro Covariance based on $\text{NNAR}(p, P, k)_m$ Model

The neural network autoregression (NNAR) model, used for time series forecasting, combines ARIMA and machine learning concepts [Thavaneswaran et al. \(2019c\)](#). It uses past data points as inputs, and the notation $\text{NNAR}(p, P, k)_m$

Algorithm 5 Random Matrix Theory (RMT) filtering for covariance matrix - normally distributed data

Require: Covariance matrix Σ , aspect ratio $Q = N/T$ (where N is the number of assets, T is the number of observations)

Compute eigendecomposition of Σ :

$$\Sigma = V\Lambda V^T$$

where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_N)$ are eigenvalues, and V contains eigenvectors.

Calculate Marčenko-Pastur bounds:

$$\lambda_{\min} = (1 - \sqrt{Q})^2, \quad \lambda_{\max} = (1 + \sqrt{Q})^2$$

Identify informative eigenvalues: Define an indicator function:

$$\mathbb{I}(\lambda_i > \lambda_{\max})$$

to select eigenvalues that exceed the theoretical upper bound.

Compute the bulk mean:

$$\mu_{\text{bulk}} = \frac{1}{|\{i : \lambda_i \leq \lambda_{\max}\}|} \sum_{\lambda_i \leq \lambda_{\max}} \lambda_i$$

Reconstruct the filtered covariance matrix:

$$\Sigma_{\text{filtered}} = V_{\text{inf}} \Lambda_{\text{inf}} V_{\text{inf}}^T + \mu_{\text{bulk}} I$$

where V_{inf} and Λ_{inf} correspond to eigenvectors and eigenvalues with $\lambda_i > \lambda_{\max}$.

Return Σ_{filtered} .

Algorithm 6 : RMT Filtering of Covariance Matrix for t distributed data

 htp

Require: Covariance matrix Σ , number of assets N , number of time periods T , degrees of freedom ν , and aspect ratio $q = N/T$

1: Perform eigen decomposition:

$$\Sigma = V\Lambda V^T$$

where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_N)$ contains the eigenvalues, and V consists of eigenvectors.

2: Extract eigenvalues λ_i and eigenvectors v_i

3: Compute theoretical scale factor for t-distribution:

$$\text{scale} \leftarrow \frac{\nu}{\nu - 2}$$

4: **if** $\nu > 4$ **then**

5: Compute kurtosis adjustment:

$$\text{kurtosis_factor} \leftarrow 3 + \frac{6}{(\nu - 4)}$$

6: **else**

7: Set default kurtosis factor: $\text{kurtosis_factor} \leftarrow 2$

8: **end if**

9: Adjust Marchenko-Pastur bounds:

$$\lambda_{\min} \leftarrow (1 - \sqrt{q \cdot \text{kurtosis_factor}})^2 \cdot \text{scale}$$

$$\lambda_{\max} \leftarrow (1 + \sqrt{q \cdot \text{kurtosis_factor}})^2 \cdot \text{scale}$$

10: Identify informative eigenvalues:

$$\mathcal{I} \leftarrow \{\lambda_i > \lambda_{\max}\}$$

11: Compute bulk mean of non-informative eigenvalues:

$$\mu_{\text{bulk}} \leftarrow \text{mean}(\{\lambda_i | \lambda_i \leq \lambda_{\max}\})$$

12: Construct filtered covariance matrix:

$$\Sigma_{\text{filtered}} \leftarrow \sum_{\lambda_i \in \mathcal{I}} \lambda_i v_i v_i^T + \mu_{\text{bulk}} \cdot I_N$$

13: **Return** Σ_{filtered}

describes a seasonal NNAR model with one hidden layer, where p is the number of lagged inputs, P is the AR order of the seasonal part, k is the number of nodes in the hidden layer, and m is the seasonal period (see [Hyndman and Athanasopoulos \(2018\)](#) for more details). The correlation between innovations of the fitted $\text{NNAR}(p, P, k)_m$ models for series X_t and Y_t is defined as the neuro correlation between X_t and Y_t [Saumyamala et al. \(2025\)](#). Similarly, the covariance between innovations of the fitted $\text{NNAR}(p, P, k)_m$ models is defined as the neuro-covariance.

Data-driven Covariance Model

In [Zhu et al. \(2020\)](#) first, Sign correlation [Thavaneswaran et al. \(2020\)](#) and the autocorrelations of the absolute return values are employed to demonstrate the non-normality of returns and the presence of time-varying volatility. Then, [Zhu et al. \(2020\)](#) proposed the data-driven exponentially weighted moving average (DDEWMA) model to estimate the covariance matrix of asset returns and provided evidence that DDEWMA covariance model outperforms the commonly used empirical covariance model. The novelty of this paper is to show that the RMT-filtered DDEWMA covariance matrix model outperforms the RMT-filtered empirical covariance as well as the RMT-filtered neuro-covariance model.

Mean Squared Error of Covariance Matrix under Normal and t Distributions

The mean squared error (MSE) of an estimator $\hat{\Sigma}$ of the covariance matrix Σ is given by

$$\text{MSE}(\hat{\Sigma}) = \mathbb{E}[\|\hat{\Sigma} - \Sigma\|^2],$$

where $\|\cdot\|^2$ is a matrix norm (Frobenius norm). Let the sample covariance matrix based on normally distributed data be denoted by S_N and sample covariance matrix based on t -distributed data be denoted by S_t

$$E[S_N] = \Sigma \tag{4.3}$$

Then, MSE of the sample covariance matrix with normal data is given by

$$\text{MSE}(S_N) = \mathbb{E}[\|S_N - \Sigma\|^2] = \text{Var}(S_N). \tag{4.4}$$

For the t -distribution with ν degrees of freedom, the sample covariance matrix, S_t , is given by

$$S_t = \frac{\nu}{\nu - 2} S_N. \tag{4.5}$$

From 4.3 and 4.5, the expectation of the sample covariance matrix is:

$$\mathbb{E}[S_t] = \frac{\nu}{\nu - 2} \Sigma \quad \text{for } \nu > 2.$$

This implies that correlations from t -distributed data exhibit higher variance, leading to more extreme values compared to normally distributed data.

For the multivariate t -distribution, the MSE of S_t is:

$$\text{MSE}(S_t) = \mathbb{E}[\|S_t - \Sigma\|^2]$$

$\text{MSE}(S_t)$ can be decomposed into bias and variance as follows:

$$\text{MSE}(S_t) = \left\| \frac{\nu}{\nu - 2} \Sigma - \Sigma \right\|_F^2 + \text{Var}(S_t). \quad (4.6)$$

It follows from equation (4.4) and (4.6), for a t -distribution with finite variance and degrees of freedom $\nu > 2$, MSE of the covariance matrix (S_t) based on t -distributed data is greater than the MSE of the covariance matrix (S_N) based on normally distributed data.

$$\text{MSE}(S_t) > \text{MSE}(S_N).$$

When the degrees of freedom tends to infinity (the excess kurtosis becomes zero), $\text{MSE}(S_t) = \text{MSE}(S_N)$.

Average Node Degree Reduction in Financial Networks

Networks built from t distributed data tend to have fewer but stronger connections due to the suppression of weaker edges in threshold adjacency matrices.

The correlation matrix with k assets is a $k \times k$ square matrix, where each element ρ_{ij} denotes the correlation coefficient between i^{th} and j^{th} assets.

$$\hat{\rho}_{ij} = \frac{S_{ij}}{\sqrt{S_{ii}S_{jj}}}.$$

where S_{ij} 's are i^{th} and j^{th} entries of sample covariance matrix. First we construct an adjacency matrix A using a threshold τ such that $A_{ij} = 1$ if $|\hat{\rho}_{ij}| > \tau$ and 0 otherwise. In this study, once the adjacency matrix is obtained, following [Saumyamala et al. \(2024\)](#) connected graphs were generated for a sequence of threshold values from 0 to 0.75 with 0.01 time steps and obtained the minimally connected network as the final network.

For normally distributed data, the sample correlation coefficient ($\hat{\rho}_{ij}$), after Fisher transformation, approximately follows a standard normal distribution, resulting in most values clustering around zero (within three standard deviations of the mean). In contrast, for t -distributed data, the sample correlation follows a heavy-tailed t -distribution, which increases the likelihood of extreme values (i.e., large positive or negative correlations) and decreases the occurrence of values near zero.

Thresholding the correlation matrix removes weak correlations while retaining strong ones. Since weaker correlations (near zero) are less frequent in t -distributed data, thresholding tends to retain fewer but stronger connections.

Consequently, the adjacency matrix derived from t -distributed returns exhibits sparser yet more pronounced edges.

The expected node degree k_i (the number of edges per node) in a d -dimensional return space is given by:

$$\mathbb{E}[k_i] = (d - 1)P(|\hat{\rho}_{ij}| > \tau).$$

where τ is the correlation threshold. For t -distributed data, although the tails are heavier, the suppression of weak correlations results in a lower probability $P(|\hat{\rho}_{ij}| > \tau)$ for moderate thresholds. This leads to networks with a smaller average node degree. However, the retained connections are stronger, reflecting the heavy-tailed nature of the t -distribution. As a result, networks constructed from t -distributed returns tend to have fewer but stronger connections, contributing to enhanced resilience.

Laplacian Matrix in Financial Networks

The Laplacian matrix, a core concept in graph theory, is increasingly used in financial network analysis to capture connectivity and structural dependencies among assets. Unlike the covariance matrix, which emphasizes risk and optimization, the Laplacian highlights network topology, offering insights into systemic risk, stability, and clustering [Saumyamala et al. \(2024\)](#), [Saumyamala et al. \(2025\)](#).

In an undirected weighted network, the Laplacian matrix L is given by:

$$L = D - A,$$

where A represents the adjacency matrix of the financial network, and D is the degree matrix a diagonal matrix with each entry D_{ii} equal to the total weight of all edges connected to node i . The adjacency matrix A ensures that all relationships are positive, reflecting the strength of connections between assets. The degree matrix D , on the other hand, captures the total influence of each node by summing its connections.

The Laplacian matrix is symmetric and positive semi definite, with non-negative eigenvalues. Its smallest eigenvalue $\lambda_1 = 0$ and the second smallest, the Fiedler value λ_2 , measures the connectivity of the network. Small λ_2 implies weakly connected or fragmented networks, while large λ_2 indicates strong integration.

Unlike the covariance matrix, which captures co-movement via return variances and covariances, the Laplacian emphasizes structural connectivity. In the covariance matrix, the largest eigenvalue reflects dominant risk factors and in the Laplacian matrix, the smallest nonzero eigenvalue (λ_2) signals network fragility. During market stress, λ_2 approaching zero indicates higher spillover risk, while the covariance matrix shows increased market-wide exposure. Thus, while the covariance matrix aids in risk estimation and portfolio optimization, the Laplacian provides a network-based view of market stability.

4.2.3 Simulation Study

In order to evaluate the impact of Random Matrix Theory (RMT) filtering, simulated data from normal and t -distributions ($df = 3$) are used. First a symmetric, positive definite empirical covariance matrix was generated from simulate normally distributed returns and then, t -distributed returns. The empirical covariance matrix was decomposed to extract eigenvalues and eigenvectors, and the Marchenko-Pastur bounds were applied to filter out noise by identifying informative eigenvalues. For portfolio optimization, minimum variance portfolio weights were computed using the inverse of both original and filtered covariance matrices. Figure 4.10 illustrates the change in portfolio weight dispersion after filtering. The analysis was extended to t -distributed data, adjusting for kurtosis dependent scale, to assess RMT filtering effects on covariance estimation and portfolio construction.

Figure 4.11 displays box plots comparing the distribution of portfolio weights obtained from empirical covariance matrices and their RMT-filtered versions, using both normally distributed and t -distributed data. The results show that RMT filtering significantly reduces the spread of portfolio weights. Figure 4.11 supports this observation by providing a comparative analysis of portfolio weight distributions at different levels of kurtosis. The RMT-filtered covariance matrices consistently produce narrower weight distributions than their unfiltered counterparts.

It follows from Figure 4.10 that the dispersion of the weights obtained

from the filtered covariance matrix based on t -distributed data has the least dispersion. Less dispersed portfolio weights generally indicate that the portfolio is more balanced, avoiding over concentration in a few assets. This reduces the risk of large losses from any single asset's poor performance. With less dispersion in the portfolio weights, the portfolio may achieve better diversification. By smoothing out extreme weightings, the portfolio is less sensitive to individual asset volatility, leading to more predictable returns.

Since we are working with t -distributed returns, it is useful to examine the stability of networks constructed from such data. Theoretically, financial networks built from data-driven t -distributed returns tend to eliminate weaker edges, leading to smaller average node degrees. This was explored using simulated data for 50 assets over 300 periods. A symmetric, positive definite covariance matrix was generated, and two sets of returns; normal and t -distributed ($df = 3$) were simulated. Correlation matrices were computed, and a dynamic thresholding method was applied, where the threshold equals the mean of absolute correlations plus 1.5 times the standard deviation of the lower triangular correlations. This threshold was used to construct adjacency matrices: values above the threshold were set to 1 (edges), others to 0, with self-loops removed. Two undirected graphs were generated, and their average node degrees calculated. As shown in Figure 4.12, the network from t -distributed returns had a slightly lower average degree (4.4) than the normal one (4.44), due to the removal of weak edges. However, the retained edges were stronger, reflecting the heavier tails of the t -distribution, suggesting enhanced network

stability.

4.2.4 Real Data Experiment

This example is based on the US financial market and utilizes a diverse dataset consisting of 55 distinct assets from 13 sectors. Daily adjusted closing prices for these assets from January 1, 2015, to December 31, 2024, were sourced from [Yahoo! Finance](#). In order to facilitate rolling correlation analysis, the dataset was divided into rolling windows. The complete dataset contains 2478 data points for each asset spanning ten years, which were segmented into 10 overlapping rolling windows. Each window consists of 2289 observations, 2268 observations (approximately nine years) for training and 21 observations (approximately one month) for testing, with a rolling window size of 21 data points (approximately one month). The windows are defined as follows in terms of indices: [1:2289], [22:2310], [43:2331], [64:2352], [85:2373], [106:2394], [127:2415], [148:2436], [169:2457], [190:2478]. Separate experiments were conducted for each window to capture dynamic relationships over time. In this section we use the stock returns data for 55 stocks and analyze the correlation networks, covariance matrices and portfolios constructed from different covariance matrices for each window. Due to the space constraint all the results corresponding to the last window 10 are reported here.

Figure 4.13 include the empirical correlation, data driven correlation and neuro correlation networks for returns data. It can be observed that depending

on the correlation network the network structure varies. Also out of the three correlation networks, data driven correlation network has the smallest average node degree. This means data driven correlation network is sparse with fewer but strong connections. These correlation networks were constructed for all 10 windows and dynamic nature of the networks was observed.

The Laplacian matrix of all three covariance matrices was obtained and analyzed the eigenvalues. The largest eigenvalue of the data driven covariance matrix is 0.0032 considerably large compared to other eigenvalues, it suggests that most assets move together, indicating a highly correlated market. The Fiedler value (second smallest eigenvalue) of the Laplacian matrix is 35.03 which suggests that the network graph is strongly connected and stocks are more interdependent. This further emphasizes the suitability of a data-driven covariance matrix for portfolio optimization.

Next, the analysis was continued with 6 covariance matrices; empirical covariance (emp), filtered empirical covariance (emp_f), data driven covariance (dd), filtered data driven covariance (dd_f), neuro covariance (neuro) and filtered neuro covariance matrices (neuro_f). Inverse of above six matrices were used in portfolio optimization. In minimum variance portfolio optimization (MVP), weights were obtained by minimizing the portfolio variance. In tangency portfolio optimization (TP), weights were obtained by maximizing the Sharpe ratio. Next, the eigen values of the six matrices were calculated and eigen vector corresponding to the minimum eigen values was extracted as the weights

for each portfolio which is known as Eigen portfolios (EP). Using the obtained portfolio weights for 18 scenarios, portfolio returns, portfolio standard deviation and Sharpe ratio was calculated. Figure 4.14 shows the portfolio risk return plot of 18 portfolios along with equally weighted portfolio (EQW).

It follows from the Figure 4.14 that tangency portfolio based on filtered data driven covariance matrix has higher return and risk compared to tangency portfolio based on non filtered data driven covariance matrix. Similarly, tangency portfolio based on filtered empirical covariance matrix has higher return and risk compared to tangency portfolio based on non filtered empirical covariance matrix. The rest of the dots overlap and the Table 4.4 contains the exact return standard deviation and Sharpe ratio values. This procedure was repeated for all 10 windows and the respective plots were not reported due to space constraints. Abbreviations used for portfolio are given in Appendix D. Overall, it can be observed that portfolios based on filtered covariance matrices yield high return with high risks compared to the portfolios based on non-filtered covariance matrices.

The Sharpe ratio of portfolios based on filtered and non-filtered, empirical and data driven covariance matrices for 10 windows are plotted in Figure 4.15. According to this plot, in most of the windows Sharpe ratio of portfolio based on filtered covariance matrix is higher than the portfolio based on non-filtered covariance matrix. Moreover, in most of the windows, portfolio based on data-driven or filtered data-driven covariance matrix has higher Sharpe ratio than that of empirical covariance matrices.

Table 4.4: Portfolio risk, return and Sharpe ratio results of 10th window

Portfolio	Return	SD	Sharpe_Ratio
MVP_emp	0.000399	0.008047	0.029059
MVP_emp_f	0.001251	0.002669	0.406708
MVP_dd	0.001413	0.003771	0.330754
MVP_dd_f	0.001251	0.002285	0.474933
MVP_neuro	0.001604	0.468406	0.003071
MVP_neuro_f	0.000941	0.452177	0.001715
TP_emp	0.014669	0.121671	0.119206
TP_emp_f	0.067395	0.154045	0.436432
TP_dd	0.034693	0.128386	0.268937
TP_dd_f	0.090639	0.17826	0.507538
TP_neuro	0.001621	0.468416	0.003107
TP_neuro_f	0.00098	0.452193	0.001802
EP_emp	0.001749	0.011993	0.132025
EP_emp_f	0.000407	0.019792	0.012223
EP_dd	0.001756	0.007005	0.227105
EP_dd_f	0.000407	0.016949	0.014274
EP_neuro	0.002609	0.756778	0.003229
EP_neuro_f	0.001285	0.612125	0.00183

The following Figure 4.16 shows the cumulative returns in the test period of 10^{th} window. It indicate that during this period, Tangency portfolio based on weights obtained by filtered data driven covariance matrix (denoted by red solid line) yields higher cumulative return than other portfolios. Similarly cumulative returns were calculated for all windows and plots are given in Figure 4.17. It is observable that in majority of the windows, portfolio with filtered DDEWMA covariance matrix (denoted by red solid line) yields higher cumulative return.

4.3 Conclusion

This chapter includes two main sections which are enhancing portfolio optimization using clustering and RMT filtered covariance matrices. In first section different clustering methods are discussed and different portfolio construction methods are followed. Second section examines the resilient financial networks and network connectivity based on RMT-filtered covariance matrices of t -distributed returns. This paper discusses the stability of the networks based on average node degree and analyzes the underlying market structure based on second smallest eigenvalue (Fiedler value) of the Laplacian matrix. Experimental results show that networks based on t -distributed data have a smaller average node degree, leading to more resilient networks. Sharpe ratio and cumulative returns of portfolio based on the filtered DDEWMA covariance matrix are higher than the corresponding Sharpe ratio and cumulative returns of other portfolios. The key contribution of this study is demonstrating the

superiority of the RMT-filtered DDEWMA model for portfolio optimization and the enhanced resilience of networks derived from t -distributed returns.

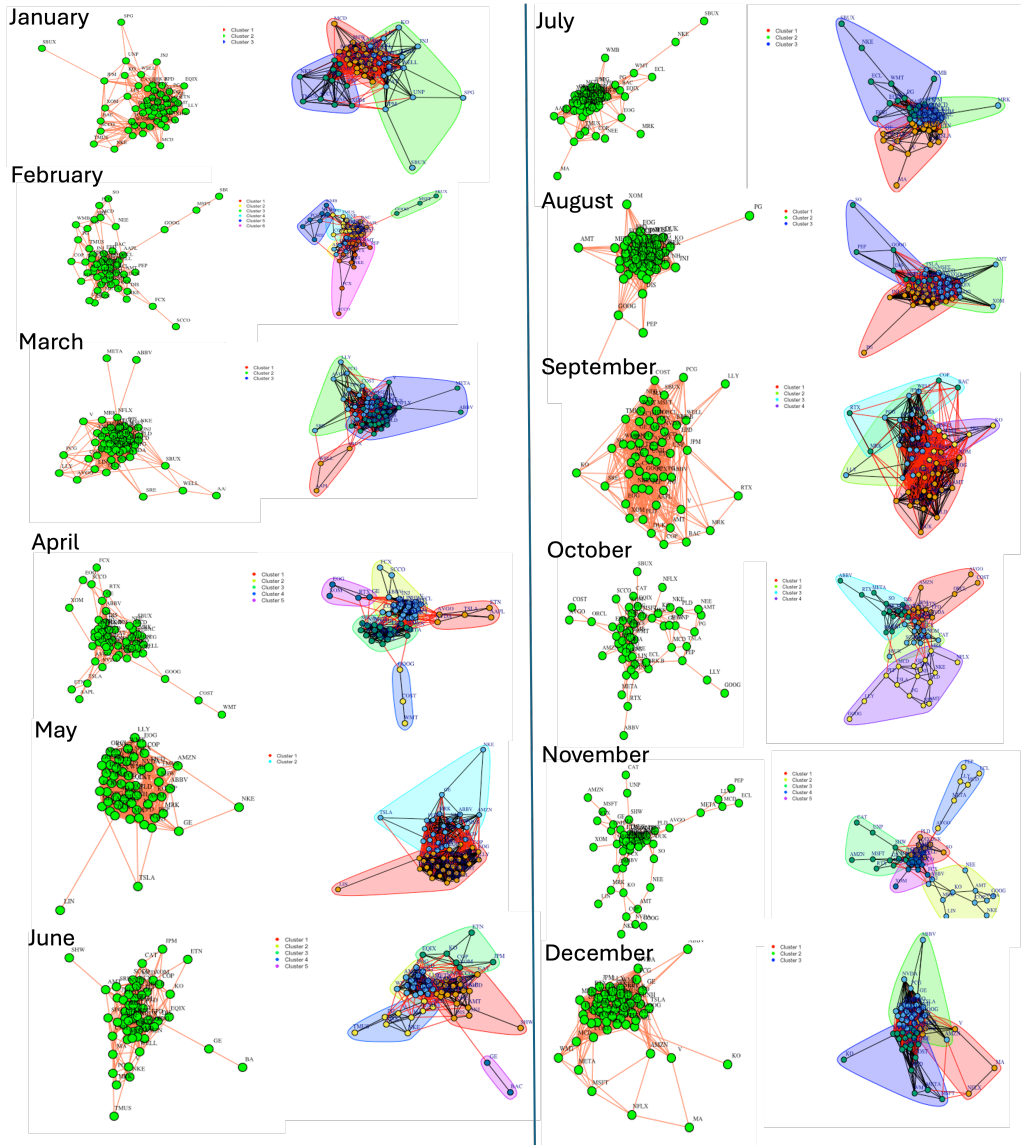


Figure 4.5: Dynamic(monthly) price network and community graphs January - December 2024

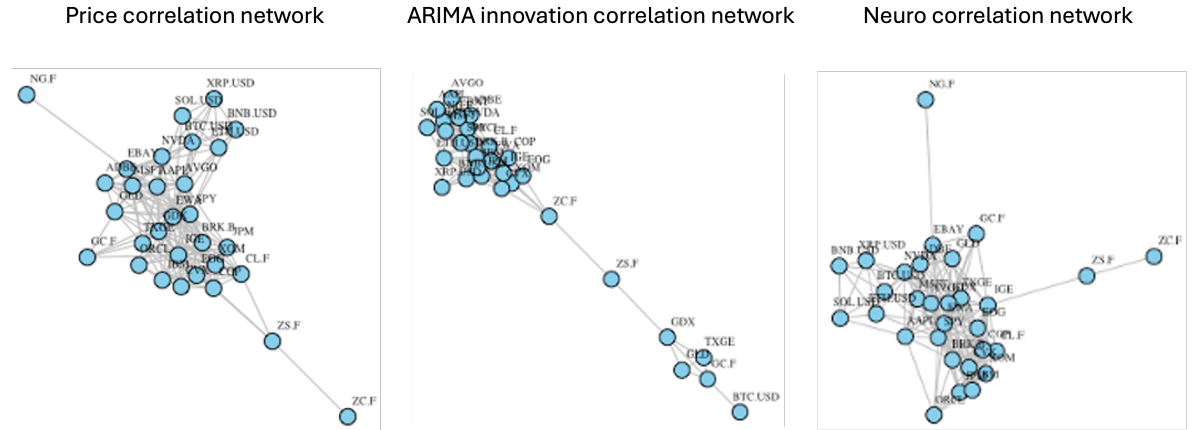


Figure 4.6: Three price correlation networks of window 1 at threshold=0.6,0.19,0.19 respectively

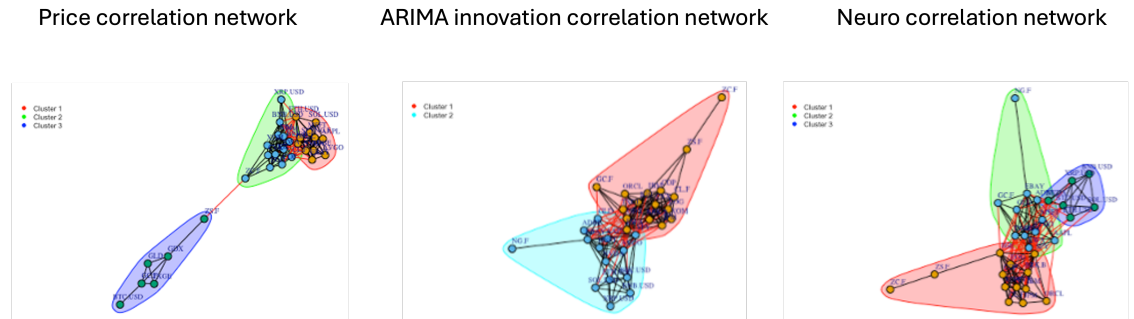


Figure 4.7: Communities detected from three price correlation networks

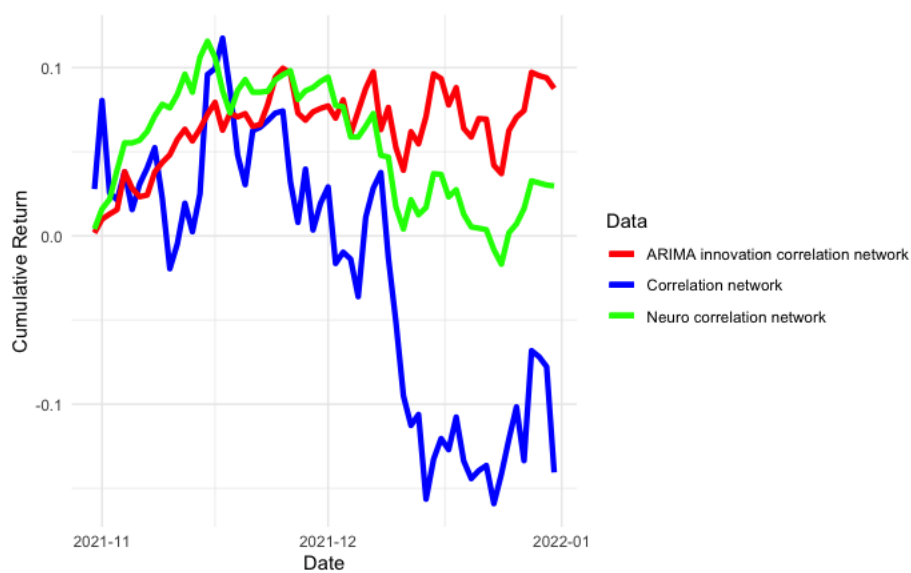


Figure 4.8: Cumulative returns based on volatility correlation of the three portfolios in window 1

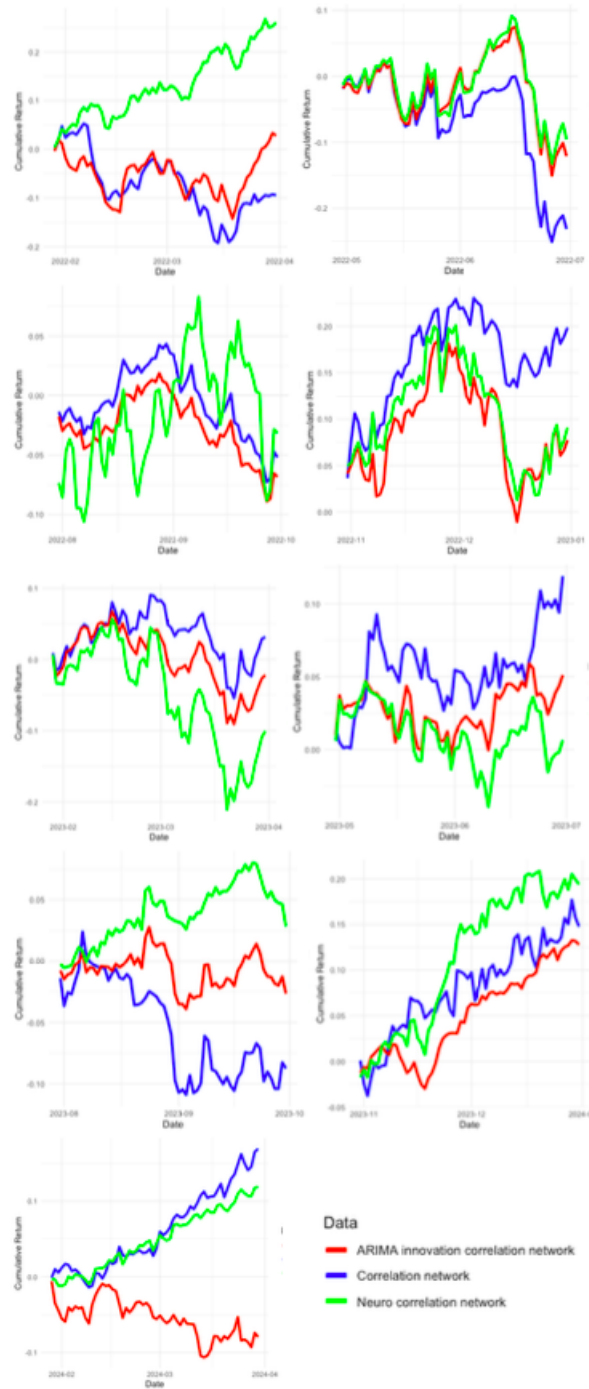


Figure 4.9: Cumulative returns based on volatility correlation of the three portfolios in window 2-10

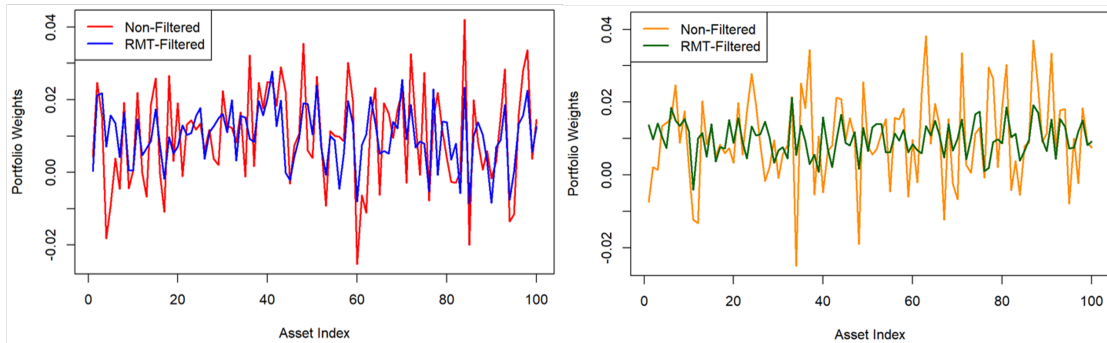


Figure 4.10: Portfolio weight distribution with RMT-filtered and non-filtered empirical covariance matrix based on normally and t distributed data

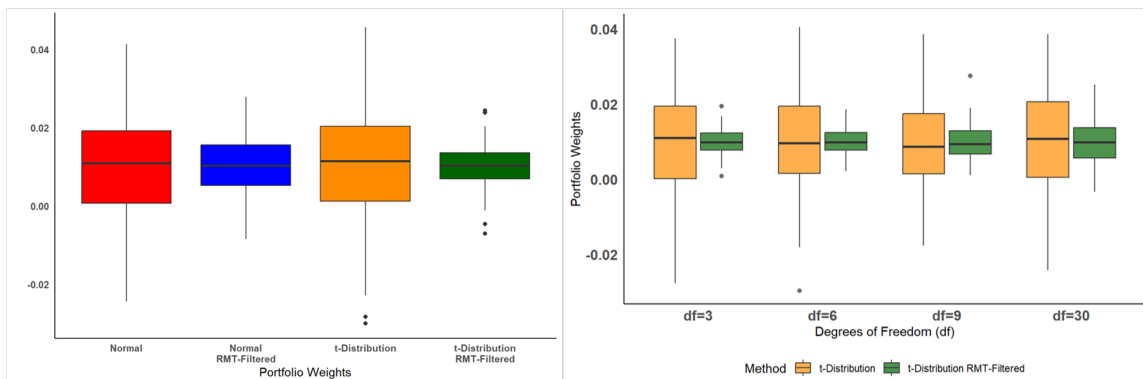


Figure 4.11: Side by side boxplots of portfolio weights with normal/ t distributed covariance matrices and RMT-filtered normal/ t covariance matrices and weights distributions across different degrees of freedom

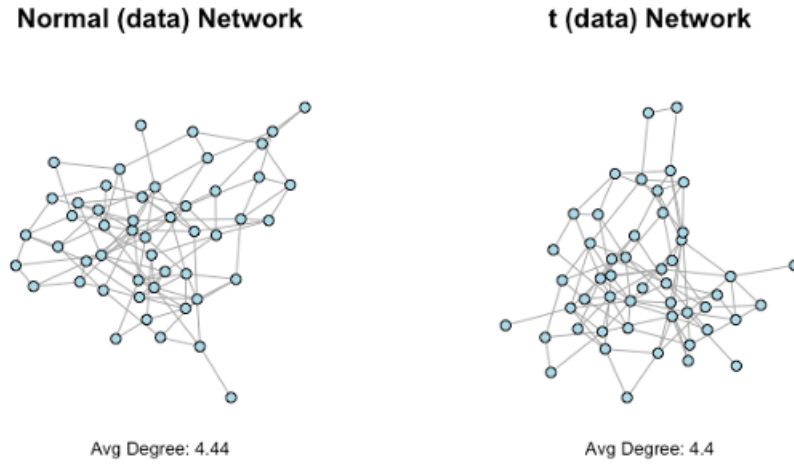


Figure 4.12: Correlation networks based on normal and t distributed data and respective average node degrees

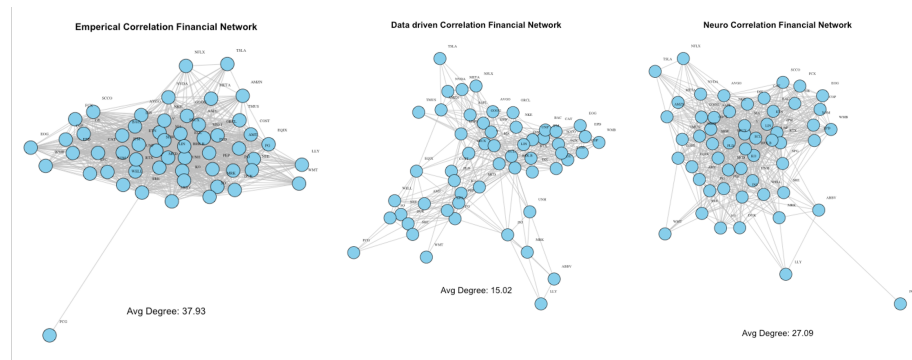


Figure 4.13: Empirical, data driven and neuro correlation networks of 10^{th} window

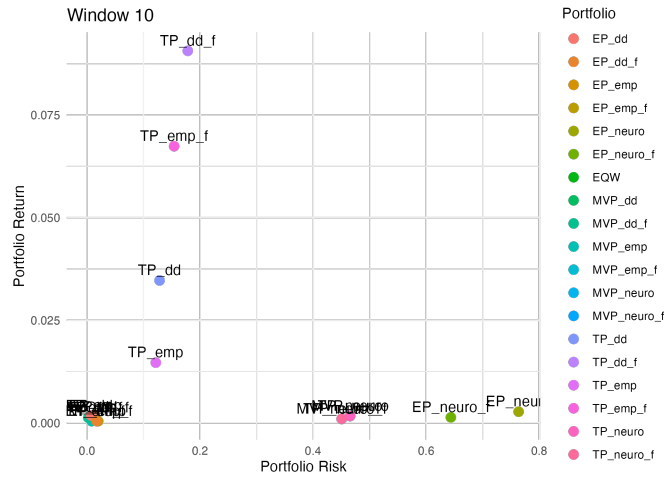
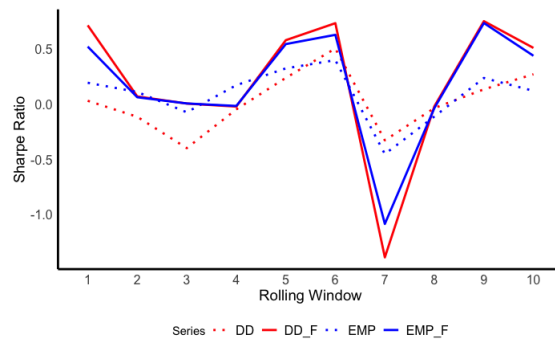
Figure 4.14: Portfolio risk and return plot for the 10th window

Figure 4.15: Sharpe ratio of tangency portfolios based on empirical and data driven covariance matrices with and without RMT-filtering over the rolling windows

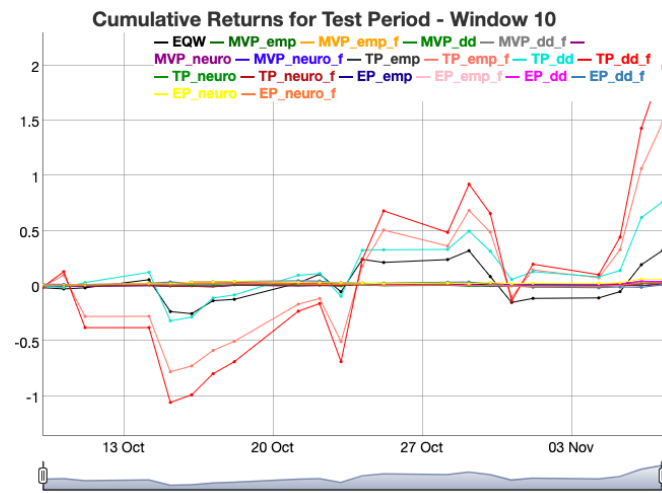
Figure 4.16: Cumulative returns of the portfolios - 10th window



Figure 4.17: Cumulative returns of the portfolios - windows 1-10

Chapter 5

Applications of Combined Estimating Functions

Estimating function theory (EFT) involves estimating parameters of interest from observed data by solving the associated optimal estimating equation. [Liang et al. \(2011\)](#) studied combined estimating functions for time series models where the conditional mean and conditional variance are functions of the same parameter of interest by using the first four conditional moments. They have demonstrated the superiority of the optimal estimate over the conditional least square estimator. Further it is shown that the information associated with the combined estimating function is always larger than the information associated with the component estimating functions. In other words, variance of combined estimating function is always smaller than variances of component estimating functions.

The applications of the estimating function theory can be found in the

literature with different aspects and motivations. [Liang et al. \(2011\)](#) developed a general framework for joint estimation of conditional mean and variance parameters in time series models using quadratic estimating functions and applied for random coefficient autoregressive (RCA) models, autoregressive conditional duration (ACD) models, doubly stochastic models and regression models with ARCH errors. [Thavaneswaran et al. \(2012\)](#) have extended the research for RCA model with generalized autoregressive conditional heteroscedasticity (GARCH) errors using martingale estimating functions and generalized duration models in [Thavaneswaran and Ravishanker \(2015\)](#). While estimating functions are defined for different models, ACD models were well applied for high frequency financial data, which are usually not bounded by distributional assumptions using penalized estimating functions as discussed in [Zhang et al. \(2019\)](#) and applied for circular time series models in [Thavaneswaran and Ravishanker \(2023\)](#).

The theorem on combining different estimating functions demonstrates that the variance of the combined function is lower than that of the individual functions. While traditional portfolio optimization focuses on minimizing the portfolio's overall variance, we followed a different approach by minimizing the variance of the combined estimating function of portfolio returns.

Following [Liang et al. \(2011\)](#) consider a discrete time stochastic process $\{y_t, t = 1, 2, \dots\}$ with conditional moments as follows.

- First moment (Mean)

$$\mu_t(\boldsymbol{\theta}) = \text{E} [y_t | \mathcal{F}_{t-1}^y], \quad (5.1)$$

- Second moment (Variance)

$$\sigma_t^2(\boldsymbol{\theta}) = \text{Var} (y_t | \mathcal{F}_{t-1}^y), \quad (5.2)$$

- Third moment (Skewness)

$$\gamma_t(\boldsymbol{\theta}) = \frac{1}{\sigma_t^3(\boldsymbol{\theta})} \text{E} [(y_t - \mu_t(\boldsymbol{\theta}))^3 | \mathcal{F}_{t-1}^y], \quad (5.3)$$

- Fourth moment (Excess kurtosis)

$$\kappa_t(\boldsymbol{\theta}) = \frac{1}{\sigma_t^4(\boldsymbol{\theta})} \text{E} [(y_t - \mu_t(\boldsymbol{\theta}))^4 | \mathcal{F}_{t-1}^y] - 3. \quad (5.4)$$

In here $\mathcal{F}_{t-1}^y = \sigma(y_1, y_2, \dots, y_{t-1})$ is the sigma-field generated by $y_1, y_2, \dots, y_{t-1}$ also defined as the information set (filtration) generated by the history of the process. Further we assume that the skewness and the excess kurtosis of the standardized variable y_t does not contain any additional parameters. In order to estimate the parameter $\boldsymbol{\theta}$ based on the observations $y_1, \dots, y_n$, we consider two classes of martingale differences $\{m_t(\boldsymbol{\theta}) = y_t - \mu_t(\boldsymbol{\theta}), t = 1, \dots, n\}$ and $\{s_t(\boldsymbol{\theta}) = m_t^2(\boldsymbol{\theta}) - \sigma_t^2(\boldsymbol{\theta}), t = 1, \dots, n\}$ such that conditional variance of $m_t(\langle m \rangle_t)$, conditional variance of $s_t(\langle s \rangle_t)$ and covariance between m_t and $s_t(\langle m, s \rangle_t)$ given as

$$\langle m \rangle_t = E[m_t^2 | \mathcal{F}_{t-1}^y] = E[(y_t - \mu_t)^2 | \mathcal{F}_{t-1}^y] = \sigma_t^2, \quad (5.5)$$

$$\langle s \rangle_t = E[s_t^2 | \mathcal{F}_{t-1}^y] = E[(y_t - \mu_t)^4 + \sigma_t^4 - 2\sigma_t^2(y_t - \mu_t)^2 | \mathcal{F}_{t-1}^y] = \sigma_t^4(\kappa_t + 2), \quad (5.6)$$

$$\langle m, s \rangle_t = E[m_t s_t | \mathcal{F}_{t-1}^y] = E[(y_t - \mu_t)^3 - \sigma_t^2(y_t - \mu_t) | \mathcal{F}_{t-1}^y] = \sigma_t^3 \gamma_t. \quad (5.7)$$

The optimal estimating functions based on the martingale differences m_t and s_t are $\mathbf{g}_M^*(\boldsymbol{\theta}) = -\sum_{t=1}^n \frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{m_t}{\langle m \rangle_t}$ and $\mathbf{g}_S^*(\boldsymbol{\theta}) = -\sum_{t=1}^n \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{s_t}{\langle s \rangle_t}$, respectively. Then, the information associated with $\mathbf{g}_M^*(\boldsymbol{\theta})$ and $\mathbf{g}_S^*(\boldsymbol{\theta})$ are $\mathbf{I}_{\mathbf{g}_M^*}(\boldsymbol{\theta}) = \sum_{t=1}^n \frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{1}{\langle m \rangle_t}$ and $\mathbf{I}_{\mathbf{g}_S^*}(\boldsymbol{\theta}) = \sum_{t=1}^n \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{1}{\langle s \rangle_t}$, respectively.

Theorem 5.0.1 *For the general model in (5.1) to (5.4), in the class of all quadratic estimating functions of the form $\mathcal{G}_Q = \{\mathbf{g}_Q(\boldsymbol{\theta}) : \mathbf{g}_Q(\boldsymbol{\theta}) = \sum_{t=1}^n (\mathbf{a}_{t-1} m_t + \mathbf{b}_{t-1} s_t)\}$,*

(a) the optimal estimating function is given by $\mathbf{g}_Q^(\boldsymbol{\theta}) = \sum_{t=1}^n (\mathbf{a}_{t-1}^* m_t + \mathbf{b}_{t-1}^* s_t)$,*

where

$$\mathbf{a}_{t-1}^* = \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(-\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t}\right)$$

and

$$\mathbf{b}_{t-1}^* = \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{1}{\langle s \rangle_t}\right);$$

(b) the information $\mathbf{I}_{g_Q^*}(\boldsymbol{\theta})$ is given by

$$\begin{aligned} \mathbf{I}_{g_Q^*}(\boldsymbol{\theta}) = & \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t} \right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{1}{\langle s \rangle_t} \right. \\ & \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right); \end{aligned}$$

(c) the gain in information $\mathbf{I}_{g_Q^*}(\boldsymbol{\theta}) - \mathbf{I}_{g_M^*}(\boldsymbol{\theta})$ is given by

$$\begin{aligned} & \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t} \right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t^2 \langle s \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{1}{\langle s \rangle_t} \right. \\ & \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right); \end{aligned}$$

(d) the gain in information $\mathbf{I}_{g_Q^*}(\boldsymbol{\theta}) - \mathbf{I}_{g_S^*}(\boldsymbol{\theta})$ is given by

$$\begin{aligned} & \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t} \right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t - \langle m, s \rangle_t^2} \right. \\ & \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right). \end{aligned}$$

The proof of the theorem is provided in Appendix [A](#).

Theorem 5.0.2 *Under the suitable regularity conditions, in the class of all unbiased estimating functions of the form $\mathcal{G} = \{g : g = g_1 + Cg_2\}$ where g_1 and g_2 are unbiased, the function which minimizes $\text{Var}(g)$ is given by $g^* = g_1 + C^*g_2$, where*

$$C^* = -\frac{\text{Cov}(g_1, g_2)}{\text{Var}(g_2)}. \quad (5.8)$$

Portfolio optimization aims to minimize the variance and maximize the returns. Minimum variance portfolio weights are obtained by minimizing the variance of the portfolio returns and tangency portfolio weights are obtained by maximizing the risk adjusted returns (Sharpe ratio). However, these tangency portfolios based on existing risk measures such as volatility, value at risk (VaR) and mean absolute deviation (MAD) are heavily affected by skewness and kurtosis. The following theorem on various combinations of estimating functions and the corresponding variances of the combined estimating functions are given for portfolio returns ([Thavaneswaran et al., 2021a](#)).

Theorem 5.0.3 *Let R_P be the portfolio return with mean μ_P , variance σ_P^2 , skewness $\tilde{\mu}_3$, and excess kurtosis κ . $F(\mu_P)$ is the cumulative density function of portfolio return evaluated at mean μ_P .*

- (i) *Combining the EFs $R_P - \mu_P$ and $(R_P - \mu_P)^2 - \sigma_P^2$, the variance of the combined EF is given by $\sigma_P^2(1 - \rho_P^2)$, where ρ_P is skewness correlation.*
- (ii) *Combining the EFs $R_P - \mu_P$ and $\text{sign}(R_P - \mu_P)$, the variance of the combined EF is given by $\sigma_P^2(1 - \rho_{P,\text{sign}}^2)$, where $\rho_{P,\text{sign}}$ is sign correlation.*

(iii) Combining the EFs $|R_P - \mu_P| - 2\rho_{P,\text{sign}}\sigma_P\sqrt{F(\mu_P)(1 - F(\mu_P))}$ and $(R_P - \mu_P)^2 - \sigma_P^2$, the variance of the combined EF is given by $\sigma_P^2(1 - \rho_{P,\text{sign}}^2)(1 - \rho_{P,\text{vol}}^2)$ where $\rho_{P,\text{vol}}$ is volatility correlation.

For any random variable X with mean μ and finite variance, ρ is defined as the sign correlation, where $\rho = \text{cor}(X - \mu, \text{sign}(X - \mu))$, and $\text{sign}(x) = \frac{\max(x, 0)}{x}$. Then, ρ is given by:

$$\rho = \frac{E|X - \mu|}{2\sigma\sqrt{F(\mu)(1 - F(\mu))}}. \quad (5.9)$$

where, F is the cumulative density function of variable X . The optimal estimate of σ is given by:

$$\hat{\sigma}_n = \frac{1}{n} \sum_{t=1}^n |r_t - \mu| \quad (5.10)$$

and the asymptotic variance of $\hat{\sigma}$ is given by:

$$\text{As. var}(\hat{\sigma}) = \frac{1}{n} \left(\frac{1 - \rho^2}{\rho^2} \right) \sigma^2. \quad (5.11)$$

Then, using the delta method, the asymptotic variance of the sample standard deviation is given by:

$$\text{As. var}(s) = \frac{\kappa + 2}{4} \frac{\sigma^2}{n} \quad (5.12)$$

with known mean. Variance of the combined estimating function by combining $R_P - \mu_P$ and $\text{sign}(R_P - \mu_P)$, is $\sigma_P^2(1 - \rho_{P,\text{sign}}^2)$, where $\rho_{P,\text{sign}}$ is defined as the sign correlation given by:

$$\rho_{P,\text{sign}} = \text{Corr}(R_P - \mu_P, \text{sign}(R_P - \mu_P)) = \frac{E|R_P - \mu_P|}{2\sigma_P \sqrt{F(\mu_P)(1 - F(\mu_P))}}. \quad (5.13)$$

Similarly variance of the combined estimating function by combining $R_P - \mu_P$ and $(R_P - \mu_P)^2 - \sigma_P^2$ is

$$\sigma_P^2(1 - \rho_P^2) \quad (5.14)$$

where ρ_P is defined as the skewness correlation given by

$$\rho_P = \text{Corr}(R_P - \mu_P, (R_P - \mu_P)^2 - \sigma_P^2) = \frac{\mu_3}{\sqrt{(\kappa + 2)}}. \quad (5.15)$$

Consider $|R_p - \mu_p| - E|R_p - \mu_p|$ as g_1 . Then, substituting mean absolute deviation $E|R_p - \mu_p|$ derived from sign correlation function we get g_1 as $|R_p - \mu_p| - 2\rho_{p,\text{sign}}\sigma_p\sqrt{F(\mu_p)(1 - F(\mu_p))}$. Combining this function and $(R_p - \mu_p)^2 - \sigma_p^2$ variance of the combined estimating function can be derived as

$$\sigma_P^2(1 - \rho_{P,\text{sign}}^2)(1 - \rho_{P,\text{vol}}^2). \quad (5.16)$$

where $\rho_{P,\text{vol}}$ is the volatility correlation defined as

$$\rho_{P,\text{vol}} = \text{Corr}(|R_p - \mu_p|, (R_p - \mu_p)^2). \quad (5.17)$$

The square root of the variance of the combined estimating function is a risk measure for portfolio returns and it incorporates skewness and kurtosis.

Existing risk measures do not consider information on higher-order moments (skewness and kurtosis) while new risk measures consider them. Most of the log returns violate normality assumption and follows heavy-tailed distributions with infinite kurtosis and degrees of freedom less than four which makes proposed risk measures more appropriate. In this study it is evident from the high skewness and kurtosis values obtained for asset returns given in Table 5.2. A simulation study done by Thavaneswaran et al. (2021a) based on 8000 portfolio returns and standardized weights obtained from uniform distributions, shows that by increasing skewness and decreasing kurtosis portfolio returns can be increased. Hence, in our applications variance of the combined estimating function based on portfolio returns is minimized to get optimum portfolio weights. That is instead of minimizing portfolio variance (σ_P^2), in this study minimizes smaller variance, $(\sigma_P^2 (1 - \rho_{P,\text{sign}}^2) (1 - \rho_{P,\text{vol}}^2))$ which is a variance of combined estimating functions.

Portfolio selection and diversification is another challenging topic in computational finance. Jaeger and Marinelli (2022) have shown that network-based portfolios outperform traditional portfolio optimization strategies due to the ability of networks to capture linear and nonlinear relationships between stocks.

They have applied network inverse degree/eigenvector centrality measures on planar maximally filtered graphs based on Pearson correlation for their study. A graph-based approach for network formulation, filtering, clustering, stock movement prediction, and portfolio optimization has been discussed in the paper [Saha et al. \(2022\)](#). Clustering techniques used here are hierarchical clustering, role-based clustering, spectral clustering, etc., and peripherality and centrality measures used for portfolio formation and optimization. [Saumyamala et al. \(2024\)](#) have followed a different approach using k-means clustering and cluster fast greedy algorithm for clustering, page ranking algorithm to select stocks from each cluster, and volatility correlation-based cumulative returns for portfolio optimization. [Jothimani et al. \(2018\)](#) study the Toronto Stock Exchange using the peripherality and centrality measures and it considers pre-defined sectors in the stock market. The models combining randomness and fuzziness were used in [Thavaneswaran et al. \(2021a\)](#) to handle volatility forecasting problems and also to solve portfolio optimization problems. In portfolio optimization, [Zhu et al. \(2020\)](#) demonstrate that the data-driven exponential weighted moving average (DDEWMA) variance covariance matrix model exhibits superior performance compared to models employing empirical variance-covariance matrices.

Risk forecasting is a vast area of study where the techniques become advanced and complex day by day. In traditional risk forecasting, statistical or machine learning models are used to predict the likelihood of certain events or

losses occurring. However, these models perform well with the training data but fail to perform as expected with test data which is known to be overfitting. Overfitting is a common problem in predictive modelling, especially for small noisy data sets. Incorporating regularization techniques into risk forecasting prevent overfitting and provides the capability to create more robust models and generalize better to unseen data. Several studies have been conducted in the area of volatility forecasting with continuous upgrading. A novel regularized risk forecasting model was developed in [Liang et al. \(2020\)](#) and tested for a set of technology stocks concluding that the regularization has improved the forecasting accuracy. In finance volatility is commonly forecasted using GARCH or variations of GARCH models. However when the degrees of freedom of the t -distribution is less than four, the variance of the conditional variance forecasts based on the t -GARCH model becomes infinity and cannot be used. The advantage of the proposed model is that an estimate of ρ can be used to identify t -distribution with appropriate degrees of freedom.

The following two sections discusses two different applications of estimation function theory combined with the methods discussed in Chapter [2](#), Chapter [3](#) and Chapter [4](#).

5.1 Application 1 : Portfolio Optimization

Portfolio optimization involves selecting the best mix of investments to achieve specific goals, such as maximizing returns or minimizing risk. The usual

procedure is predicting the future returns of the assets based on historical data while assessing the risk and correlations between the assets followed by using mathematical models and optimization algorithms to find the optimal asset allocation.

In a recent study fuzzy numbers were used to capture the uncertainty associated with the point estimates of correlation or similarity in adjacency matrices. Three different fuzzy networks were constructed using empirical correlation, data-driven correlation, and cosine similarity and a novel non-linear fuzzy volatility network was introduced, and the connectedness of the network was discussed based on 50 technology stocks. This study emphasizes that higher cumulative returns can be obtained by incorporating networking, clustering, page ranking, and calculation of cumulative returns using volatility correlation.

In this study, the following approach was followed. Well diversified portfolio should contain stocks, cryptocurrencies and commodities. Therefore, we considered 30 assets, including 5 cryptocurrencies (Bitcoin (BTC-USD), Ethereum (ETH-USD), Tether USDt (USDT-USD), Binance coin (BNB-USD), Solana (SOL-USD)), 5 commodities (Crude oil Futures (CL=F), Corn Futures (ZC=F), Gold Futures (GC=F), Natural Gas (NG=F), Soybean Futures (ZS=F)), 5 stocks each from the largest four sectors; technology (Apple Inc. (AAPL), NVIDIA Corporation (NVDA), Microsoft Corporation (MSFT), Broadcom Inc.(AVGO), Oracle Corporation (ORCL)), financial services (Berkshire Hath-

away Inc.(BRK-B), JPMorgan Chase & Co.(JPM), Visa Inc.(V), Mastercard Incorporated (MA), Bank of America Corporation (BAC)), healthcare (Eli Lilly and Company (LLY), United Health Group Incorporated (UNH), Johnson & Johnson (JNJ), AbbVie Inc.(ABBV), Merck & Co., Inc.(MRK)), and consumer cyclical (Amazon.com, Inc.(AMZN), Tesla, Inc.(TSLA), McDonald's Corporation (MCD), NIKE Inc (NKE), Starbucks Corporation (SBUX)), with the highest market capitalization over the period from January 2021 to April 2024. Also, we considered adjusted closing prices to calculate log returns, $r_t = \log\left(\frac{P_t}{P_{t-1}}\right) = \log(P_t) - \log(P_{t-1})$. Then, 4 correlation matrices were calculated.

1. Empirical correlation matrix

This is the most common Pearson correlation coefficient used in literature which is calculated by

$$\rho_{r_i, r_j} = \frac{\sum (r_i - \bar{r}_i)(r_j - \bar{r}_j)}{\sqrt{\sum (r_i - \bar{r}_i)^2 \sum (r_j - \bar{r}_j)^2}} \quad (5.18)$$

where r_i and r_j are return of two assets i and j , $\bar{r}_i$ and $\bar{r}_j$ are the sample means of r_i and r_j , respectively. Each element in the matrix represents the correlation between two variables without controlling of the influence from any other variables.

2. Data-driven correlation matrix

Data driven exponential weighted moving average (DDEWMA) correla-

tion (data-driven correlation) matrix of assets' returns is proposed using the DDEWMA variance-covariance matrix (see [Zhu et al. \(2020\)](#) for more details).

3. Partial correlation matrix

Unlike the empirical correlation network partial correlation coefficients between pairs of variables, control the influence of all other variables in the dataset. Each element in the partial correlation matrix reflects the direct linear relationship between two variables after removing the effects of the other variables in the dataset. This can reveal the underlying direct relationship that might be masked in the correlation matrix.

Consider returns of three stocks: r_h, r_i and r_j . The partial correlation coefficient ρ_{r_i, r_j, r_h} between stock i and j based on the stock h is the Pearson's correlation coefficient between the residuals of stock i and j that are uncorrelated with h . Both returns of stocks i and j are regressed on stock h to obtain these residuals. Thus, [Saha et al. \(2022\)](#) define the partial correlation coefficient in terms of three Pearson's correlation coefficients as follows.

$$\rho_{r_i, r_j, h(t)} = \frac{\rho_{r_i, r_j(t)} - \rho_{r_i, r_h(t)} \rho_{r_j, r_h(t)}}{\sqrt{(1 - \rho_{r_i, r_h(t)}^2) (1 - \rho_{r_j, r_h(t)}^2)}} \quad (5.19)$$

In R, partial correlation is obtained by computing the covariance matrix, followed by inverting the matrix to get a precision matrix and then, normalizing.

4. Volatility correlation matrix

Calculate observed volatility $V_t = |r_t - \bar{r}| / \bar{\rho}$ where correlation between $r_t - \bar{r}$ and $\text{sign}(r_t - \bar{r})$ denoted as $\bar{\rho} = \text{corr}(r_t - \bar{r}, \text{sign}(r_t - \bar{r}))$. The observed volatility shown to be an unbiased estimator of the standard deviation in [Thavaneswaran et al. \(2012\)](#). Then, calculate the Pearson correlation between volatilities.

The above correlation matrices are used to obtain adjacency matrices (Refer Section 3.1) such that if the correlation between nodes i and j (ρ_{ij}) exceeds a threshold value (ρ_{thresh}) then, the two nodes are connected. To construct a network, the connectedness (correlation) probability between two nodes is not directly observed but should be estimated from available data. To address the uncertainty associated with the point estimates of connectedness, [Saumyamala et al. \(2024\)](#) introduce data-driven non-linear adaptive symmetric fuzzy adjacency matrices (Refer Section 3.2 for more details), which can be identified as confidence matrices (interval estimates). The detailed steps to obtain fuzzy adjacency matrices are given in [Saumyamala et al. \(2024\)](#).

For community detection, cluster-fast-greedy algorithm (see Section 4.1.3 for more details), which optimizes a specific cost function (modularity) was used to determine the optimal number of communities as in [Kolaczyk and Csárdi \(2014\)](#). The input for this algorithm is the fuzzy network graphs proposed in this study. The algorithm follows a hierarchical agglomerative approach, meaning it starts with each node as a separate cluster and iteratively merges

the clusters in pairs. This process creates a hierarchy of clusters, where larger clusters are built by merging smaller ones. At each step, the goal is to merge the pair of clusters that leads to the largest increase (or smallest decrease) in modularity. The algorithm works to maximize modularity by iteratively merging communities. A high modularity value indicates that more edges are concentrated within communities than expected in a random graph, meaning the clustering reflects a meaningful division (Newman, 2003).

The minimally connected network graphs based on different fuzzy adjacency correlation matrices and their community structure with the optimal number of communities resulting from cluster fast greedy algorithm are given in Figures 5.1-5.4 respectively.

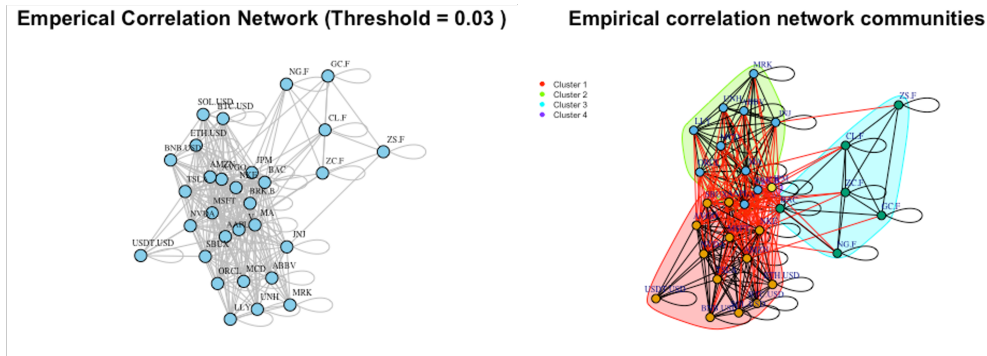


Figure 5.1: Empirical correlation network graph and community graph

The empirical correlation network is divided into 4 communities, and it is observed that all the commodities and cryptocurrencies are clustered separately. When observing cluster membership, it was noted that JPM alone forms a separate cluster. The data-driven correlation network is also

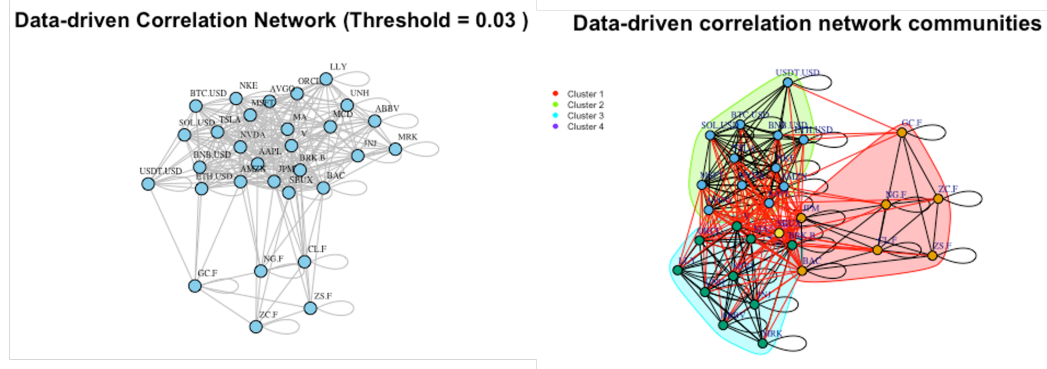


Figure 5.2: Data-driven correlation network graph and community graph

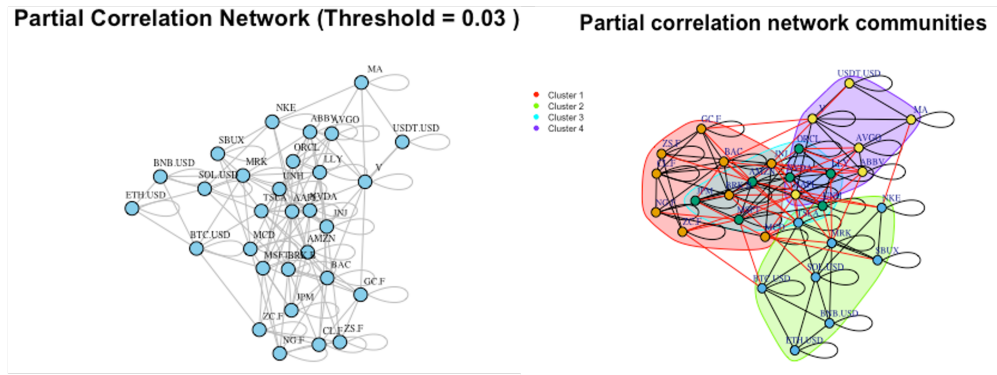


Figure 5.3: Partial correlation network graph and community graph

divided into 4 communities but with different asset combinations. Similar to the above network, commodities are clustered together but SBUX itself forms a separate cluster. The partial correlation network is clustered into 4 communities and commodities form a separate community. Cryptocurrencies also clustered together except USDT which include in a separate cluster with some stocks. The volatility correlation network is clustered into 5 clusters, and its cluster memberships are different from the above three networks. In here all the cryptocurrencies in cluster 3, all technology stocks in cluster 2, all

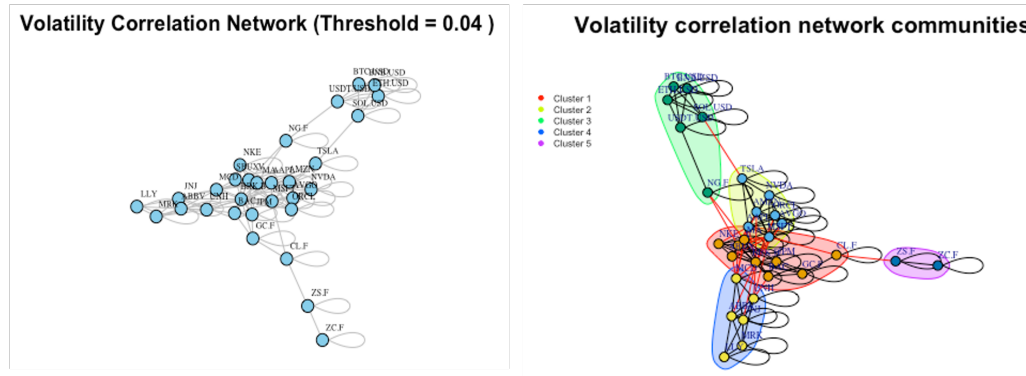


Figure 5.4: Volatility correlation network graph and community graph

financial services stocks in cluster 1, and all healthcare stocks in cluster 4 while commodities and consumer cyclical stocks are scattered in different clusters.

After identifying communities in all four networks, the PageRank algorithm discussed in Section 3.6.1 was applied to each community to identify the asset with the highest PageRank. The asset with highest PageRank is considered as the most linked/important asset in a community, which can have significant impact on the overall network dynamics. Four portfolios are created for each network and finally, create another portfolio by considering the highest page rank asset from each sector, cryptocurrency, commodity and stock.

After applying the page rank algorithm to each cluster, the following portfolios were selected. The portfolios selected from each sector are also included in Table 1.

When observing 5 portfolios it is noted that each portfolio consists of at least one cryptocurrency, one commodity, and one stock which makes the portfolio well diversified. It is also notable that different cryptocurrencies are chosen as

Table 5.1: Networks and portfolio selection from identified communities

Network	Portfolio with stocks (with highest page rank) selected from each community or sector				
	1	2	3	4	5
Empirical correlation Network	AMZN	BRKB	CLF	JPM	
Data driven correlation network	CLF	BTC	BRKB	SBUX	
Partial correlation network	CLF	SOL	NVDA	V	
Volatility correlation network	BRKB	MSFT	USDT	JNJ	ZCF
Sectors	BTC	CLF	AAPL	BRKB	AMZN

the most important/connected asset in different portfolios. However, although Gold is the most active commodity in the market, in most of portfolios, crude oil (CLF) has been chosen as the most important asset in the community. In the case of stocks, the selection is unique for each network. This implies that rather than dividing assets by default it is also possible to define meaningful communities with assets following similar behavior.

Table 5.2: Summary statistics of asset returns in a portfolio selected from sectors

	Mean	SD	$\tilde{\mu}_3$	κ	$\hat{\rho}_{sign}$	$\hat{\rho}_{vol}$
BTC	0.0011	0.0405	-0.0546	3.8997	0.6978	0.8903
CLF	0.0008	0.0249	-0.5249	2.0848	0.7615	0.8912
AAPL	0.0007	0.0175	0.1013	1.7145	0.7482	0.8992
BRK-B	0.0007	0.0111	0.1990	1.3515	0.7566	0.9137
AMZN	0.0002	0.0236	0.0930	4.7015	0.7161	0.8690
JNJ	0.0001	0.0102	0.2694	3.2348	0.7371	0.8768

Summary statistics of all 5 portfolios were computed, but due to a space constraint, only the results of the portfolio selected from sectors are given in Table 5.2. The high skewness and kurtosis values suggest that stock returns

follow heavy-tailed distributions, suggesting the appropriateness of minimizing the variance of combined estimating functions instead of portfolio variance.

In order to evaluate the performance of five portfolios the optimal portfolio weights were determined using a recently proposed risk measure by minimizing the variance of combined estimating functions following equation 5.16. For model validation 80% of the data used as training and 20% of the data used for testing purpose. A detailed introduction and applications of the new risk measure are discussed in [Thavaneswaran et al. \(2021b\)](#).

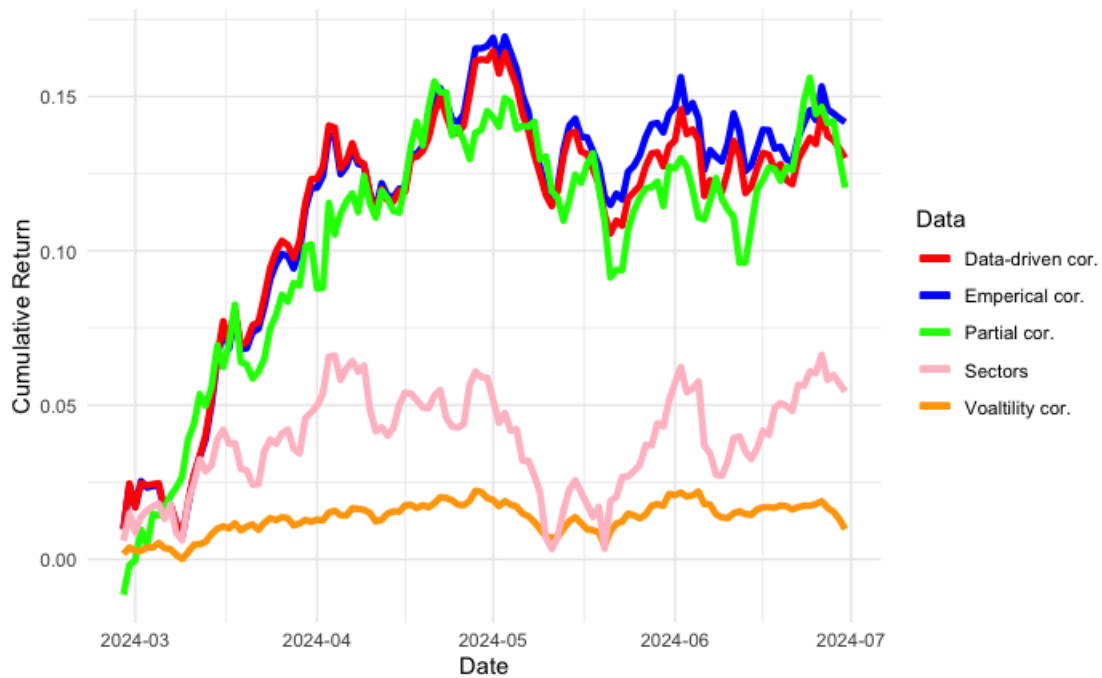


Figure 5.5: Cumulative returns of portfolios

Observing the graph in Figure 5.5, it can be clearly observed that the

portfolios obtained from the communities detected by the empirical correlation networks outperform the other approaches. Also, it is evident that the usual diversification strategy of the formation of portfolios from sectors is less profitable than suggested approaches with empirical correlation and data-driven correlation networks which can be considered as a novel contribution to the literature.

5.2 Application 2: Generalized VaR and ES Forecasting

In this section we consider the three most common risk measures; volatility, VaR, and ES. The volatility of the returns is represented by the standard deviation, but is more appropriate only when the expected returns of the asset are normally distributed. In the real world, returns being normal is highly unlikely due to the outliers and extreme cases. Hence, the forecasts using the standard deviation may lead to inaccurate information. VaR is a more common statistic that quantifies the risk of potential losses in an investment. Investors use this measure to control the level of risk exposure. It is more convenient to use because of its theoretical properties, implementation, and ease of backtest. VaR is denoted by $\text{VaR}(p)$ or $\text{VaR}^{100 \times p\%}$, which means there is a p probability of losses exceeding VaR and $(1 - p)$ probability of losses less than VaR. The most common probability(p) values are 1% or 5%. ES

is another measure to quantify the risk conservatively. ES at $p\%$ level is the expected return of the portfolio at the worst $p\%$ of cases, which is calculated by taking a weighted average of losses beyond the VaR cutoff point in the tail of the loss distribution. In this study, the input of the risk forecasting model would be the series of logarithmic returns $r_t = \log P_t - \log P_{t-1}$. Following [Liang et al. \(2020\)](#), equations (3) and (4) are used to calculate the one step ahead forecasts of VaR and ES.

$$\text{VaR}_{t+1}(p) = -\hat{\sigma}_{t+1} F_r^{-1}(p), \quad (5.20)$$

$$\text{ES}_{t+1}(p) = -\frac{\hat{\sigma}_{t+1}}{p} \int_{\infty}^{F_r^{-1}(p)} x f(x) dx. \quad (5.21)$$

where $f(x)$ is the density function for the conditional distribution of logarithmic returns r_t , and $F_t^{-1}(p)$ is the inverse cumulative density function (CDF) of r_t at a given tail probability p , while $\hat{\sigma}_{t+1}$ represents the volatility forecast at time $t + 1$ given the past t observations are known.

The commonly used volatility forecasting models of log returns are Historical Simulation (HS), Moving Average (MA), Exponentially Weighted Moving Average (EWMA), Generalized Autoregressive Conditional Heteroskedasticity (GARCH) and Extreme Value Theory (EVT) models. From these models, GARCH models have the most practical usage in volatility analysis such that major financial forecasting websites such as Financial Risk Forecasting ([Danielsson, 2024](#)) and Volatility Laboratory (V-Lab) ([Engle et al., 2024](#))

use the family of GARCH models for forecasting volatility (see <https://www.financialriskforecasting.com/> and <https://vlab.stern.nyu.edu/> for further details).

A novel data-driven generalized EWMA volatility forecasting model proposed by [Thavaneswaran et al. \(2012\)](#) to forecast the volatility directly and obtain VaR forecasts which are given by

$$\hat{\sigma}_{t+1} = (1 - \alpha)\hat{\sigma}_t + \alpha \frac{|r_t - \bar{r}|}{\hat{\rho}}, \quad 0 < \alpha < 1, \quad (5.22)$$

where, $\hat{\rho}$ is the sample sign correlation between $(r_t - \bar{r})$ and $\text{sign}(r_t - \bar{r})$, defined as $\rho = \text{Corr}((r_t - \bar{r}), \text{sign}(r_t - \bar{r}))$. This volatility model is defined as a data-driven model by considering the two facts that are; the optimal value of α is obtained by minimizing the one-step ahead forecast error sum of squares (FESS) and the conditional distribution of r_t is calculated using sign correlation. [Liang et al. \(2020\)](#) proposed that regularized risk forecasting using volatility models outperforms the existing risk forecasts, including usual EWMA forecasts for time series models.

First calculate the sample mean $\bar{r}$ and sample sign correlation $\hat{\rho}$ of data, and then, use the recursive equation 5.22 to find volatility forecasts. We obtain the recursive form of regularized volatility forecasts based on the ridge, LASSO, and elastic net penalties as given below.

$$\hat{\sigma}_{t+1}^{\text{ridge}} = \frac{\hat{\sigma}_{t+1} - s}{1 + \lambda} + s, \quad (5.23)$$

$$\hat{\sigma}_{t+1}^{\text{lasso}} = \text{sign}(\hat{\sigma}_{t+1} - s) (|\hat{\sigma}_{t+1} - s| - \lambda)_+ + s, \quad (5.24)$$

$$\hat{\sigma}_{t+1}^{\text{en}} = \frac{\text{sign}(\hat{\sigma}_{t+1} - s) (|\hat{\sigma}_{t+1} - s| - \omega^{\text{en}} \lambda)_+}{1 + (1 - \omega^{\text{en}}) \lambda} + s, \quad (5.25)$$

where, $\lambda, \omega^{\text{en}} \in (0, 1)$ are tuning parameters. s is the sample standard deviation of estimated volatilities and we smooth extreme volatility values by penalizing the centred estimated volatilities at sample standard deviation to improve the forecast.

$\hat{\sigma}_{t+1}^{\text{ridge}}$ is a special case of $\hat{\sigma}_{t+1}^{\text{en}}$ when $\omega^{\text{en}} = 0$.

$\hat{\sigma}_{t+1}^{\text{lasso}}$ is a special case of $\hat{\sigma}_{t+1}^{\text{en}}$ when $\omega^{\text{en}} = 1$.

When $\lambda = 0$ and $\omega^{\text{en}} = 0$ in equation 5.25, regularized volatility forecasts become usual EWMA volatility forecasts.

The following steps were followed to obtain regularized volatility forecasts.

1. Calculate the sample sign correlation ($\hat{\rho}$) and estimated volatility ($|r_t - \bar{r}| / \hat{\rho}$) using log returns as input data.
2. Set an initial value S_0 and a range for the optimal smoothing constant α to $(0, 1)$. For each value, compute $S_t = \alpha Z_t + (1 - \alpha) Z_{t-1}$ and determine optimal α such that one step ahead FESS $\sigma(Z_t - S_{t-1})^2$ is minimum.

3. Regularize $S_t = \alpha_{\text{opt}} Z_t + (1 - \alpha_{\text{opt}}) Z_{t-1}$ by adding elastic net penalty S_t^{en} .
4. Recursively compute forecast S_t using α_{opt} and regularized forecast S_t^{en} .
5. Minimize the one step ahead FESS computed from regularized forecasts to determine optimal λ and ω^{en} .
6. Compute σ_{t+1}^{en} using equation (7) and use it to compute the one step ahead VaR (equation (4)) and ES (equation (5)).

The dataset comprises the adjusted closing price of six cryptocurrencies (BTC, BNB, ETH, XRP, DOGE and ADA) from 2018 to 2023. The time series plot of the daily adjusted closing price of six cryptocurrencies in Figure 5.6 clearly shows that there was a remarkable shift during the beginning of the COVID-19 pandemic, which depicts the importance of dividing the data set into two periods. The pre-COVID period is 2018-2019 and the pandemic period is 2020-2023.

The summary statistics of log returns including mean, standard deviation (SD), excess kurtosis (κ), lag 1 sample ACF of log returns (r_t), squared returns (r_t^2), absolute returns ($|r_t|$), sample sign correlation ($\hat{\rho}$) and estimated degrees of freedom (df.est) for before and during the pandemic are shown in Table 5.3 and Table 5.4 respectively. Here, df.est is the degrees of freedom of the t -distribution using the estimated $\hat{\rho}$.

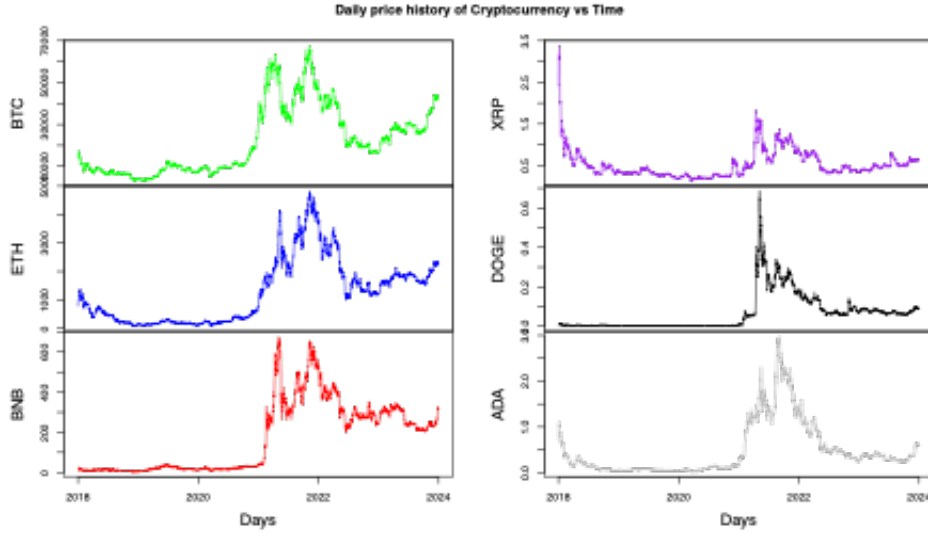


Figure 5.6: Daily adjusted closing price series of cryptocurrencies

Table 5.3: Summary statistics of daily return for cryptocurrencies: Before pandemic

Crypto	Mean	SD	κ	Skewness	Lag 1 sample acf			$\hat{\rho}$	df.est
					r_t	r_t^2	$ r_t $		
BTC	-0.0005	0.0378	6.0151	-0.1268	-0.0387	0.2212	0.1762	0.6756	3.4140
ETH	-0.0030	0.0479	5.3837	-0.4297	-0.0532	0.1272	0.1059	0.6937	3.7086
BNB	0.0003	0.0507	5.8348	0.0530	-0.0486	0.1585	0.2098	0.7013	3.8645
XRP	-0.0025	0.0494	8.4945	0.6914	-0.0018	0.3010	0.2717	0.6645	3.2723
DOGE	-0.0016	0.0510	12.1565	1.1280	0.1070	0.4553	0.5368	0.6389	3.0187
ADA	-0.0040	0.0556	5.0624	0.1060	-0.0173	0.1415	0.0706	0.7286	4.6955

Table 5.4: Summary statistics of daily return for cryptocurrencies: During pandemic

Crypto	Mean	SD	κ	Skewness	Lag 1 sample acf			$\hat{\rho}$	df.est
					r_t	r_t^2	$ r_t $		
BTC	0.0011	0.0335	6.4981	-0.1999	-0.0166	0.1243	0.0993	0.6851	3.5576
ETH	0.0015	0.0437	8.3121	-0.3992	-0.0403	0.1828	0.1346	0.6907	3.6530
BNB	0.0020	0.0488	22.6427	0.8092	-0.0722	0.3624	0.2782	0.6132	2.8332
XRP	0.0007	0.0605	22.2968	0.6965	-0.0441	0.2738	0.1297	0.5891	2.6992
DOGE	0.0029	0.0825	106.7780	6.0311	-0.0084	0.3757	0.0756	0.4990	2.3864
ADA	0.0015	0.0520	6.9001	0.3898	-0.0431	0.2476	0.2012	0.6912	3.6625

Among all the cryptocurrencies, the expected returns before the pandemic are negative and during the pandemic are positive. However, the standard deviation has decreased during the pandemic only in XRP and DOGE. Kurtosis increases for all the cryptocurrencies, which implies more data in the tail than that of normally distributed data. This behavior is extreme in DOGE. However, in both periods, the sample sign correlation value is less than 0.72, resulting in the usual normality assumption being invalid. Further, when the degrees of freedom of the t -distribution is less than four, the variance of the conditional variance forecasts based on the t -GARCH model becomes infinity and cannot be used (Thavaneswaran et al., 2012). In here for all cryptocurrencies degrees of freedom are also less than 4, except in ADA before the pandemic. Hence, log returns with infinite kurtosis were used in modeling.

The Table 5.5 shows one step ahead volatility forecasts of six cryptocurrencies across different models. For the DDEWMA model, ADA exhibits the highest volatility (0.0456), while BTC show lowest values volatility (0.0241). Similarly, EWMA model forecasts higher volatility in ADA and lowest volatil-

ity in BTC, but less than the DDEWMA volatility forecasts. This pattern suggests that the DDEWMA model approach offers a more comprehensive risk capture by directly estimating volatility (σ). Notably, the EWMA model shows substantially lower volatility estimates for all cryptocurrencies except for ADA, potentially underestimating market risks. The NGARCH and tGARCH models illustrate relatively consistent volatility estimates across all cryptocurrencies.

Table 5.5: One step ahead volatility forecasts of Cryptocurrency

Model	BTC	ETH	BNB	XRP	DOGE	ADA
NGARCH	0.0236	0.0282	0.0429	0.0401	0.0321	0.0343
tGARCH	0.0253	0.0280	0.0457	0.0316	0.0331	0.0332
EWMA	0.0220	0.0259	0.0393	0.0257	0.0298	0.0400
DDEWMA	0.0241	0.0287	0.0423	0.0261	0.0294	0.0456
Model Risk	1.15	1.11	1.16	1.56	1.09	1.37

Figure 5.7 shows VAR and ES forecasts of MA and HS models at different tail probabilities. MA model forecasts lower VAR and ES for BTC and higher values for XRP across all p values. HS model show step-like behavior and limited responsiveness, underestimating recent risk changes.

Figure 5.8 shows VAR and ES forecasts of NGARCH and tGARCH models at different tail probabilities. Both models forecasts lower VAR and ES for BTC and higher values for BNB across all p values.

Figure 5.9 shows VAR and ES forecasts of EWMA and DDEWMA models at different tail probabilities. The EWMA model predicts lower VAR and ES for BTC and higher values for ADA for all values p . Similarly, the DDEWMA model forecasts lower VAR and ES for BTC but higher forecast values for BNB

5.2. APPLICATION 2: GENERALIZED VAR AND ES FORECASTING 177

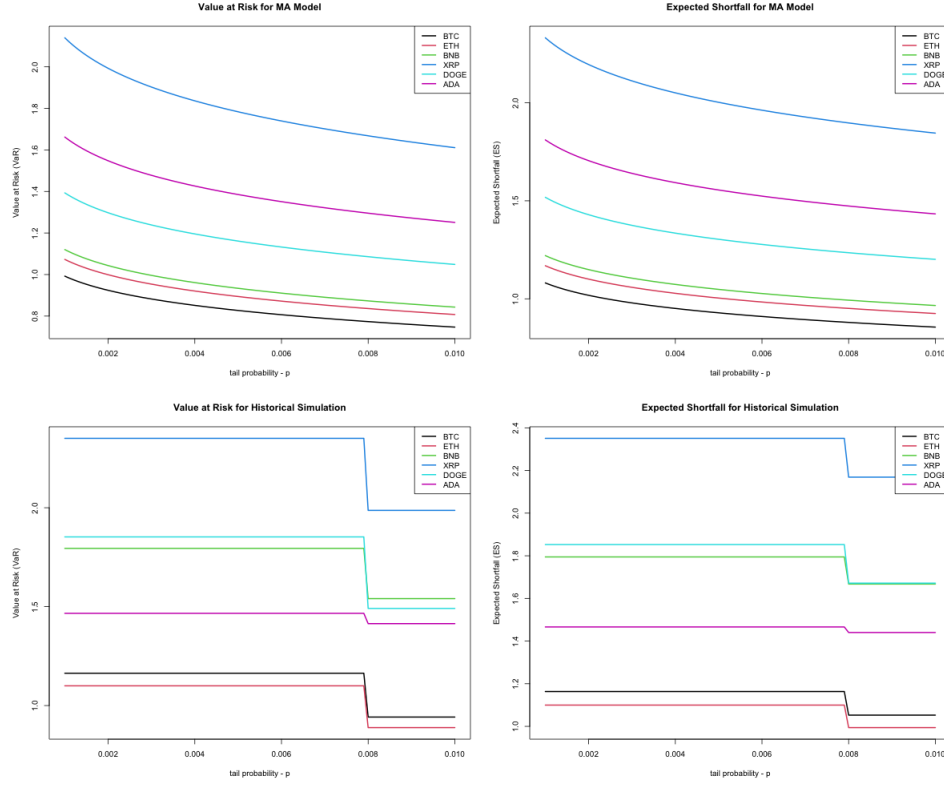


Figure 5.7: VaR and ES forecasts using MA and HS models

across all p values.

Tables 5.6 and 5.7 show the risk forecasts with regularization and without regularization before and during the pandemic period. From Table 5.6, ADA has the highest volatility during the pandemic, implying it is the highest risky asset while DOGE is the lowest risky asset. From Table 5.7, ADA is the highest risky asset while BTC is the lowest risky asset. This observation is similar with regularization as well as without regularization. When observing RMSE values of volatility forecasts in both periods without regularization and with

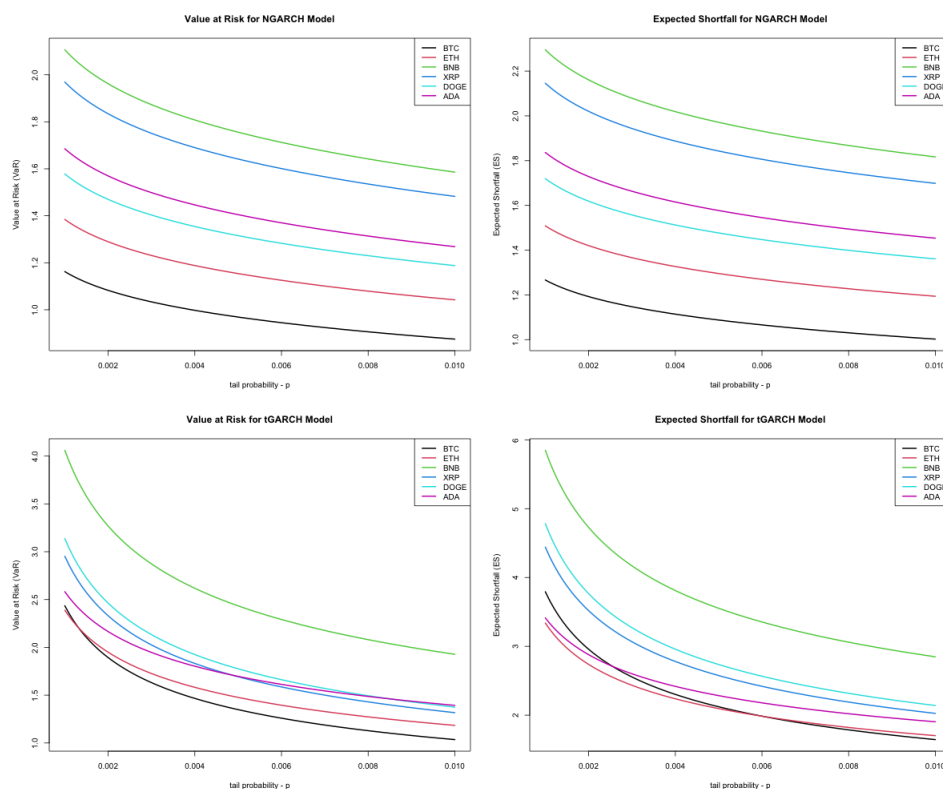


Figure 5.8: VaR and ES forecasts using NGARCH and tGARCH models

regularization using elastic net penalty at $p = 0.01$, it only shows a very slight improvement.

Since it is not sufficient to consider only one p-value and make a conclusion, for a sequence of p-values from 0.001 to 0.01, regularized VaR and ES forecasts using NGARCH, tGARCH, EWMA and DDEWMA models were recorded, and calculated the model risk ratio between highest risk and lowest risk. Figure 5.10 show the box plots of the model risk ratio of VaR forecasts before the pandemic and during the pandemic.

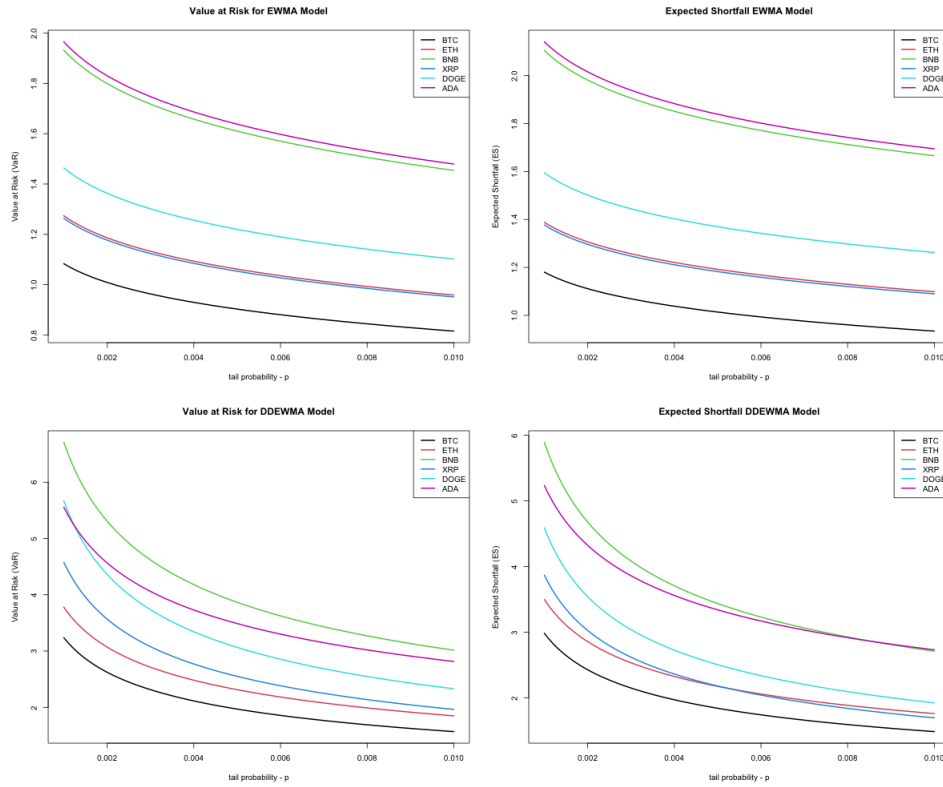


Figure 5.9: VaR and ES forecasts using EWMA and DDEWMA models

Before the pandemic, model risk of regularized VaR forecasts are smaller than non-regularized VaR forecasts for all cryptocurrencies. However, during the pandemic, only for ETH and BNB the model risk VaR and ES forecast was reduced after regularization. It is also the same observation for the model risk of ES and model risk of VaR and ES forecasts before the pandemic. By comparing the two figures we can conclude that regularization made an impact on model risk before the pandemic than during the pandemic.

Table 5.6: Non-regularized and regularized risk forecasts of cryptocurrency before the pandemic

Crypto	α_{opt}	$\omega_{\text{opt}}^{\text{en}}$	$\lambda_{\text{opt}}^{\text{en}}$	RMSE		Volatility		VaR		ES	
				Non-Reg	Reg	Non-Reg	Reg	Non-Reg	Reg	Non-Reg	Reg
BTC	0.10	1.00	0.004	0.0382	0.0380	0.022	0.026	59.354	69.982	86.921	102.485
ETH	0.10	1.00	0.04	0.048	0.048	0.030	0.0349	82.294	92.923	117.229	132.370
BNB	0.07	1.00	0.004	0.048	0.0488	0.032	0.036	85.908	96.523	120.868	135.803
XRP	0.12	1.00	0.004	0.052	0.0525	0.023	0.028	63.989	74.594	95.195	110.972
DOGE	0.25	1.00	0.004	0.055	0.055	0.013	0.017	34.286	44.785	52.733	68.881
ADA	0.09	1.00	0.004	0.051	0.050	0.035	0.039	92.863	103.343	117.229	138.497

Table 5.7: Non-regularized and regularized risk forecasts of cryptocurrency during the pandemic

Crypto	α_{opt}	$\omega_{\text{opt}}^{\text{en}}$	$\lambda_{\text{opt}}^{\text{en}}$	RMSE		Volatility		VaR		ES	
				Non-Reg	Reg	Non-Reg	Reg	Non-Reg	Reg	Non-Reg	Reg
BTC	0.03	0	0.004	0.0222	0.0222	0.0227	0.0267	60.4862	60.5998	87.3264	87.4905
ETH	0.03	0	0.004	0.0233	0.0234	0.02661	0.0268	70.7455	70.9263	101.2589	101.5176
BNB	0.04	0	0.004	0.0242	0.0264	0.0348	0.0349	90.0395	90.1823	142.6555	142.8818
XRP	0.01	0	0.004	0.0242	0.0262	0.0344	0.0344	84.6201	84.6201	137.4676	137.4676
DOGE	0.09	0	0.004	0.0396	0.0395	0.0424	0.0426	96.0049	96.3661	168.0487	168.6809
ADA	0.09	0	0.004	0.0364	0.0364	0.0477	0.0497	126.9402	132.2680	181.5403	189.1598

5.3 Conclusion

This Chapter discusses the new risk measures (standard deviation of combined estimating functions of returns) and its applications in computational finance. The first application of portfolio optimization suggests that the approach of obtaining fuzzy correlation networks followed by community detection and portfolio formation using page ranking results in well-diversified portfolios. The portfolio cumulative returns calculated from weights obtained by minimizing the variance of the combined estimating functions suggest that portfolios obtained from empirical, data-driven, and partial correlation networks outperform the portfolios based on predefined sectors in the financial market. Secondly,

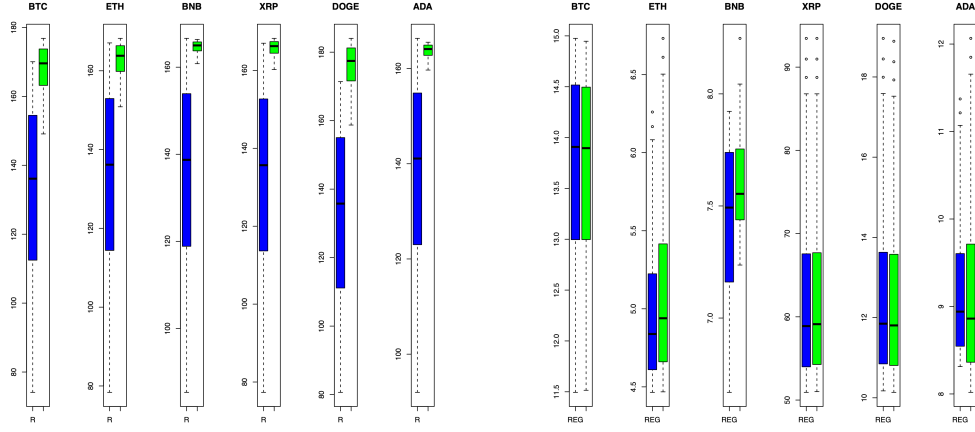


Figure 5.10: Model risk ratio of VaR before (Left) and after (Right) the pandemic (p values range from 0.001 to 0.01): non-regularized (green), regularized (blue)

regularized VaR and ES forecasts of cryptocurrencies based on sign correlation confirm that regularization has slightly improved the forecasting accuracy in pre- and post-COVID periods. However, the model risk was improved by regularization before the pandemic than during the pandemic.

Heyde (1997) made foundational contributions to statistical inference using estimating functions, focusing particularly on minimizing the trace of the asymptotic covariance matrix of estimators. This work advanced the theory of quasi-likelihood and generalized estimating equations and the details are included in Appendix B. The future work involve using Trace Minimization method in covariance estimation and apply in portfolio optimization and risk forecasting.

Chapter 6

Conclusions and Contribution of the Thesis

This thesis presents contributions across three key areas in statistics and computational finance: risk forecasting, network analysis, and portfolio optimization.

In Chapter 2 (Risk Forecasting), we propose the neural network autoregressive (NNAR) model for univariate volatility forecasting, offering improved guidance for investors in making forward-looking investment decisions. For multivariate price and volatility forecasting, NNAR model combined with FLAP method outperforms the existing models achieving higher forecast accuracy. Key contribution of this thesis chapter is combining FLAP with NNAR model which is non linear and demonstrating the robustness of NNAR+FLAP method for forecasting non normally distributed data.

In Chapter 3 (Network Analysis) we include a discussion of fuzzy volatility networks, class of networks with network metrics, data driven volatility networks

and dynamic price correlation networks. A significant contribution of the thesis is introducing fuzzy volatility networks which effectively capture uncertainty in correlation structures which is an aspect often overlooked in traditional networks. Moreover, deriving volatility networks from financial networks and introducing data driven volatility network is a novel addition to the literature. Another novel contribution of the thesis is introduction of Neuro-innovation correlation networks, derived from innovations (residuals) of the NNAR model, providing a deeper view of nonlinear dependencies in market dynamics.

In Chapter 4 (Portfolio Optimization) there are three sections which address three limitations in portfolio optimization of high dimensional data. Firstly, it discusses cluster based portfolio optimization method which proposed as a method of dimensional reduction. Secondly, network based community detection approach combined with Page rank algorithm is suggested as a novel approach for better diversification of portfolio. Thirdly, the thesis presents a Data-Driven Exponentially Weighted Moving Average (DDEWMA) covariance matrix, combined with Random Matrix Theory (RMT) filtering which outperforms the existing covariance matrices in portfolio optimization. This approach effectively denoises high-dimensional covariance matrices, improving matrix invertibility and ultimately leading to higher Sharpe ratios and cumulative returns.

In summary, this thesis contributes to the advancement of computational finance by introducing novel models and methodologies with practical implications for stock market analysts, brokers, and investors. The proposed techniques

enhance forecasting accuracy, capture complex financial relationships through networks, and provide more robust solutions for portfolio construction under uncertainty.

Bibliography

- Almog, A., F. Besamusca, M. MacMahon, and D. Garlaschelli (2015). Mesoscopic community structure of financial markets revealed by price and sign fluctuations. *PLOS one* 10(7), e0133679. (Cited on page 49.)
- Anuar, S. H. H., Z. A. Abas, N. M. Yunus, N. H. M. Zaki, N. A. Hashim, M. F. Mokhtar, and A. F. Nizam (2021). Comparison between louvain and leiden algorithm for network structure: a review. *Journal of Physics: Conference Series* 2129(1), 012028. (Cited on page 112.)
- Avramov, D., S. Cheng, and L. Metzker (2023). Machine learning vs. economic restrictions: Evidence from stock return predictability. *Management Science* 69(5), 2587–2619. (Cited on page 9.)
- Bai, Z. and J. Yao (2010). Spectral analysis of large dimensional random matrices and its applications to wireless communications and finance. *Statistics and Its Interface* 3(3), 345–378. (Cited on page 74.)
- Baumer, B. S., D. T. Kaplan, and N. J. Horton (2017). *Modern Data Science with R*. CRC Press. (Cited on page 83.)

- Bowala, S. and J. Singh (2022). Optimizing portfolio risk of cryptocurrencies using data-driven risk measures. *Journal of Risk and Financial Management* 15(10), 427. (Cited on page 114.)
- Coscia, M. (2021). Pearson correlations on complex networks. *Journal of Complex Networks* 9(6), cnab036. (Cited on page 89.)
- Danielsson, J. (2024). Financial risk forecasting. Accessed: 10 January 2024. (Cited on page 170.)
- Danielsson, J., K. R. James, M. Valenzuela, and I. Zer (2016). Model risk of risk models. *Journal of Financial Stability* 23, 79–91. (Cited on page 24.)
- Dip Das, J., R. K. Thulasiram, C. Henry, and A. Thavaneswaran (2024). Encoder–decoder based lstm and gru architectures for stocks and cryptocurrency prediction. *Journal of Risk and Financial Management* 17(5), 200. (Cited on pages 20 and 24.)
- Engle, R. et al. (2024). V-Lab. Accessed: 10 January 2024. (Cited on page 170.)
- Engström, C. and S. Silvestrov (2014). Generalisation of the damping factor in pagerank for weighted networks. In N. Bäuerle, M. Bladt, and M. Steffensen (Eds.), *Modern Problems in Insurance Mathematics*, pp. 313–333. Cham: Springer. (Cited on page 84.)
- Godambe, V. P. (1985). The foundations of finite sample estimation in stochastic processes. *Biometrika* 72(2), 419–428. (Cited on page 5.)

Granger, C. W. (1969). Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society* 37(3), 424–438. (Cited on page 75.)

Heyde, C. C. (1997). *Quasi-Likelihood And Its Application: A General Approach to Optimal Parameter Estimation*. Springer. (Cited on page 181.)

Hoque, M. E., S. Bowala, A. Saumyamala, A. Thavaneswaran, and R. Thulasiram (2024a). Novel resilient model risk forecasts based on neuro volatility models. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 210–219. IEEE. (Cited on pages 2 and 90.)

Hoque, M. E., S. Bowala, A. Saumyamala, A. Thavaneswaran, and R. Thulasiram (2024b). Novel resilient model risk forecasts based on neuro volatility models. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 210–219. IEEE. (Cited on pages 10, 19, 20, 25 and 26.)

Hyndman, R. J. and G. Athanasopoulos (2018). *Forecasting: Principles and Practice* (3rd ed.). OTexts. Accessed: 17 August 2024. (Cited on pages 2, 9, 13, 14, 15, 26, 92, 93 and 126.)

Jaeger, M. and D. Marinelli (2022). Network diversification for a robust portfolio allocation. *SSRN Electronic Journal* (4068889). Available at SSRN. (Cited on pages 4 and 157.)

Jayanetti, S., S. Bowala, S. Kalia, A. Thavaneswaran, M. E. Hoque, and G. Razvan (2025). Statistical and deep learning approaches for risk assessment and volatility forecasting in renewable energy investments. In *IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE. (Cited on pages 19 and 24.)

Jothimani, D., C. Kavaklioğlu, and A. Başar (2018). Financial networks: A study of the toronto stock exchange. In *2018 IEEE International Conference on Big Data (Big Data)*, pp. 4684–4691. IEEE. Accessed: 17 August 2024. (Cited on pages 89, 90, 102 and 158.)

Kim, S. and E. N. Brown (2010). A general statistical framework for assessing granger causality. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2222–2225. IEEE. (Cited on page 78.)

Kolaczyk, E. D. and G. Csárdi (2014). *Statistical Analysis of Network Data with R*. Springer. Accessed: 17 August 2024. (Cited on page 163.)

Kostecki, W. (2013). *Granger Causality Estimation and Classification of Neural Time Series*. Phd thesis, University of Reading, Reading, UK. (Cited on page 77.)

Kyriakopoulos, F., S. Thurner, C. Puh, and S. W. Schmitz (2009). Network and eigenvalue analysis of financial transaction networks. *The European Physical Journal B* 71, 523–531. (Cited on page 119.)

- Laloux, L., P. Cizeau, J.-P. Bouchaud, and M. Potters (2005). Financial applications of random matrix theory: Old laces and new pieces. *Acta Physica Polonica B* 36(9), 2767–2784. (Cited on page 119.)
- Liang, Y., A. Thavaneswaran, and B. Abraham (2011). Joint estimation using quadratic estimating function. *Journal of Probability and Statistics* 2011, 372512. (Cited on pages 149 and 150.)
- Liang, Y., A. Thavaneswaran, Z. Zhu, R. K. Thulasiram, and M. E. Hoque (2020). Data-driven adaptive regularized risk forecasting. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1296–1301. IEEE. Accessed: 17 August 2024. (Cited on pages 6, 9, 159, 170 and 171.)
- Makled, R. A. and W. Cheng (2024). Exploring multivariate statistics: Unveiling the power of eigenvalues in wishart distribution analysis. *Contemporary Mathematics*, 4054–4063. (Cited on page 119.)
- Mantziou, A., M. Cucuringu, V. Meirinhos, and G. Reinert (2023). The gnar-edge model: a network autoregressive model for networks with time-varying edge weights. *Journal of Complex Networks* 11(6), cnad039. (Cited on page 47.)
- Markowitz, H. (1952). Portfolio selection. *The Journal of Finance* 7(1), 77–91. (Cited on page 99.)

Matsuda, H. (2000). Tests for the structure of the covariance matrix of a stationary gaussian time series. *The Annals of Statistics* 28(6), 2609–2635. (Cited on page 74.)

Newman, M. E. J. (2003). Fast algorithm for detecting community structure in networks. *Physical Review E* 69(6), 066133. (Cited on page 164.)

Page, L. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. (Cited on page 83.)

Pitas, I. (Ed.) (2016). *Graph-Based Social Media Analysis*, Volume 39. CRC Press. (Cited on pages 3 and 48.)

Ranathungage, T., S. Bowala, M. E. Hoque, A. Thavaneswaran, and R. Thulasiram (2024). Application of a novel fuzzy pattern mining algorithm for sequence data. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE. In press. (Cited on pages 48, 60 and 83.)

Ranathungage, T., A. Thavaneswaran, Y. Liang, R. Thulasiram, and A. Paseka (2024). Novel data-driven dynamic network science application in algorithmic trading. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 396–401. IEEE. (Cited on page 89.)

Saha, S., J. Gao, and R. Gerlach (2022). A survey of the application of graph-based approaches in stock market analysis and prediction. *International*

Journal of Data Science and Analytics 14(1), 1–15. (Cited on pages 3, 47, 89, 102, 158 and 162.)

Saumyamala, A., S. Bowala, Y. Liang, S. Basnayake, A. Thavaneswaran, and R. Thulasiram (2024). Novel non-linear adaptive fuzzy adjacency matrices for financial volatility network models. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1813–1818. IEEE. (Cited on pages 5, 47, 63, 72, 90, 114, 129, 130, 158 and 163.)

Saumyamala, A., S. Bowala, A. Thavaneswaran, A. Paseka, Y. Liang, and R. K. Thulasiram (2025). Superiority of rmt filtered data-driven covariance matrices for resilient networks and portfolio optimization. In *5th IEEE International Workshop on Dynamic Data Science and Big Data Analytics in Finance (DDS-BDAF 2025) of IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE. (Cited on page 31.)

Saumyamala, A., A. Thavaneswaran, S. Bowala, J. D. Das, R. Thulasiram, and A. Paseka (2025). Novel financial network models using neuro correlations and applications. In *2025 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics Companion (CiFer Companion)*, pp. 1–5. IEEE. (Cited on pages 47, 48, 118, 119, 126 and 130.)

Schafer, J., R. Opgen-Rhein, V. Zuber, M. Ahdesmaki, A. P. D. Silva, and K. Strimmer (2021). corpcor: Efficient estimation of covariance and (partial) correlation. R package version 1.6.10. (Cited on pages 33, 36 and 40.)

Singh, J., S. Bowala, A. Thavaneswaran, R. Thulasiram, and S. Mandal (2022). Data-driven and neuro-volatility fuzzy forecasts for cryptocurrencies. In *2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–8. IEEE. Accessed: 17 August 2024. (Cited on page 2.)

Thavaneswaran, A., S. S. Appadoo, and A. Paseka (2009). Weighted possibilistic moments of fuzzy numbers with applications to garch modeling and option pricing. *Mathematical and Computer Modelling* 49(1-2), 352–368. (Cited on pages 57 and 59.)

Thavaneswaran, A. and C. C. Heyde (1999). Prediction via estimating functions. *Journal of Statistical Planning and Inference* 77(1), 89–101. (Cited on page 10.)

Thavaneswaran, A., Y. Liang, and J. Frank (2012). Inference for random coefficient volatility models. *Statistics and Probability Letters* 82(12), 2086–2090. (Cited on pages 150, 163, 171 and 175.)

Thavaneswaran, A., Y. Liang, N. Yu, A. Paseka, and R. K. Thulasiram (2021a). Novel data-driven resilient portfolio risk measures using sign and volatility correlations. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1742–1747. IEEE. (Cited on pages 58, 154, 157 and 158.)

Thavaneswaran, A., Y. Liang, N. Yu, A. Paseka, and R. K. Thulasiram (2021b).

- Novel data-driven resilient portfolio risk measures using sign and volatility correlations. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1742–1747. IEEE. (Cited on pages 114 and 168.)
- Thavaneswaran, A., A. Paseka, and J. Frank (2020). Generalized value at risk forecasting. *Communications in Statistics - Theory and Methods* 49(20), 4988–4995. (Cited on pages 2, 12, 32, 47, 48, 53, 65, 68 and 126.)
- Thavaneswaran, A. and N. Ravishanker (2015). Estimating functions for analogy models. Available at SSRN 2643256. (Cited on page 150.)
- Thavaneswaran, A. and N. Ravishanker (2023). Estimating functions for circular time series models. *Sankhya A* 85(1), 198–213. (Cited on page 150.)
- Thavaneswaran, A., N. Ravishanker, and Y. Liang (2015). Generalized duration models and optimal estimation using estimating functions. *Annals of the Institute of Statistical Mathematics* 67(1), 129–156. (Cited on page 6.)
- Thavaneswaran, A., R. K. Thulasiram, Z. Zhu, M. E. Hoque, and N. Ravishanker (2019a, July). Fuzzy value-at-risk forecasts using a novel data-driven neuro volatility predictive model. In *Proceedings of the IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, Milwaukee, WI, USA, pp. 221–226. (Cited on page 2.)

- Thavaneswaran, A., R. K. Thulasiram, Z. Zhu, M. E. Hoque, and N. Ravishanker (2019b). Fuzzy value-at-risk forecasts using a novel data-driven neuro volatility predictive model. In *2019 IEEE 43rd Annual Computer Software and Applications Conference*, Volume 2, pp. 221–226. IEEE. Accessed: 17 August 2024. (Cited on page 9.)
- Thavaneswaran, A., R. K. Thulasiram, Z. Zhu, M. E. Hoque, and N. Ravishanker (2019c). Fuzzy value-at-risk forecasts using a novel data-driven neuro volatility predictive model. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, Volume 2, pp. 221–226. IEEE. (Cited on pages 90 and 123.)
- Tumminello, M., T. Di Matteo, T. Aste, and R. N. Mantegna (2007). Correlation based networks of equity returns sampled at different time horizons. *The European Physical Journal B* 55, 209–217. (Cited on page 102.)
- Wickramasuriya, S. L., G. Athanasopoulos, and R. J. Hyndman (2019). Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association* 114(526), 804–819. (Cited on page 31.)
- Wooldridge, J. M. (2010). *Econometric analysis of cross section and panel data*. MIT press. (Cited on page 92.)
- Yang, C.-X., Y. Chen, L. Niu, and Q. Li (2014). Cointegration analysis and

influence rank—a network approach to global stock markets. *Physica A: Statistical Mechanics and its Applications* 400, 168–185. (Cited on page 83.)

Yang, Y. F., G. Athanasopoulos, R. J. Hyndman, and A. Panagiotelis (2024). Forecast linear augmented projection (flap): A free lunch to reduce forecast error variance. arXiv preprint arXiv:2407.01866. (Cited on pages 31, 32, 33, 34 and 41.)

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control* 8(3), 338–353. (Cited on page 57.)

Zhang, Y., J. Zou, N. Ravishanker, and A. Thavaneswaran (2019). Modeling financial durations using penalized estimating functions. *Computational Statistics Data Analysis* 131, 145–158. (Cited on page 150.)

Zhu, Z., A. Thavaneswaran, A. Paseka, J. Frank, and R. Thulasiram (2020). Portfolio optimization using a novel data-driven ewma covariance model with big data. *incom*, 1308–1313. (Cited on pages 9, 49, 53, 118, 119, 126, 158 and 162.)

Zimmermann, H.-J. (2001). *Fuzzy Set Theory and Its Applications* (4th ed.). Nowell: Kluwer Academic Publishers. (Cited on page 59.)

Zou, C. (2010). *Applications of Granger Causality to Biological Data*. PhD thesis, University of Warwick, Coventry, UK. Available from Warwick Research Archive Portal (WRAP). (Cited on page 77.)

Appendices

A Combined Estimating Function Theorem

Theorem A.1 *For the general model in (5.1) to (5.4), in the class of all quadratic estimating functions of the form $\mathcal{G}_Q = \{\mathbf{g}_Q(\boldsymbol{\theta}) : \mathbf{g}_Q(\boldsymbol{\theta}) = \sum_{t=1}^n (\mathbf{a}_{t-1}m_t + \mathbf{b}_{t-1}s_t)\}$,*

(a) the optimal estimating function is given by $\mathbf{g}_Q^(\boldsymbol{\theta}) = \sum_{t=1}^n (\mathbf{a}_{t-1}^*m_t + \mathbf{b}_{t-1}^*s_t)$,*

where

$$\mathbf{a}_{t-1}^* = \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(-\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t}\right)$$

and

$$\mathbf{b}_{t-1}^* = \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{1}{\langle s \rangle_t}\right);$$

(b) the information $\mathbf{I}_{g_Q^}(\boldsymbol{\theta})$ is given by*

$$\begin{aligned} \mathbf{I}_{g_Q^*}(\boldsymbol{\theta}) &= \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{1}{\langle s \rangle_t} \right. \\ &\quad \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'}\right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t}\right); \end{aligned}$$

(c) the gain in information $\mathbf{I}_{\mathbf{g}_Q^*}(\boldsymbol{\theta}) - \mathbf{I}_{\mathbf{g}_M^*}(\boldsymbol{\theta})$ is given by

$$\sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t^2 \langle s \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{1}{\langle s \rangle_t} \right. \\ \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right);$$

(d) the gain in information $\mathbf{I}_{\mathbf{g}_Q^*}(\boldsymbol{\theta}) - \mathbf{I}_{\mathbf{g}_S^*}(\boldsymbol{\theta})$ is given by

$$\sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t - \langle m, s \rangle_t^2} \right. \\ \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right).$$

We choose two orthogonal martingale differences m_t and $\psi_t = s_t - \sigma_t \gamma_t m_t$, where the conditional variance of ψ_t is given by $\langle \psi \rangle_t = \frac{\langle m \rangle_t \langle s \rangle_t - \langle m, s \rangle_t^2}{\langle m \rangle_t} = \sigma_t^4 (\kappa_t + 2 - \gamma_t^2)$. That is, m_t and ψ_t are uncorrelated with conditional variance $\langle m \rangle_t$ and $\langle \psi \rangle_t$, respectively. Moreover, the optimal martingale estimating function and associated information based on the martingale differences ψ_t are

$$\mathbf{g}_\Psi^*(\boldsymbol{\theta}) = \sum_{t=1}^n \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \right) \frac{\psi_t}{\langle \psi \rangle_t} \\ = \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t}\right)^{-1} \left(\left(-\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t^2 \langle s \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right) m_t \right. \\ \left. + \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{1}{\langle s \rangle_t} \right) s_t \right)$$

and

$$\begin{aligned}
\mathbf{I}_{\mathbf{g}_\Psi^*}(\boldsymbol{\theta}) &= \sum_{t=1}^n \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \right) \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t}{\langle m \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \right) \frac{1}{\langle \psi \rangle_t} \\
&= \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t} \right)^{-1} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t^2 \langle s \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} \frac{1}{\langle s \rangle_t} \right. \\
&\quad \left. - \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right).
\end{aligned}$$

Then, the quadratic estimating function based on m_t and ψ_t becomes

$$\begin{aligned}
\mathbf{g}_Q^*(\boldsymbol{\theta}) &= \sum_{t=1}^n \left(1 - \frac{\langle m, s \rangle_t^2}{\langle m \rangle_t \langle s \rangle_t} \right)^{-1} \left(\left(-\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{1}{\langle m \rangle_t} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} \right) m_t \right. \\
&\quad \left. + \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\langle m, s \rangle_t}{\langle m \rangle_t \langle s \rangle_t} - \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{1}{\langle s \rangle_t} \right) s_t \right),
\end{aligned}$$

and satisfies the sufficient condition for optimality

$$\mathbb{E} \left[\frac{\partial \mathbf{g}_Q(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \middle| \mathcal{F}_{t-1}^y \right] = \text{Cov}(\mathbf{g}_Q(\boldsymbol{\theta}), \mathbf{g}_Q^*(\boldsymbol{\theta}) | \mathcal{F}_{t-1}^y) K, \forall \mathbf{g}_Q(\boldsymbol{\theta}) \in \mathcal{G}_Q,$$

where K is a constant matrix. Hence, $\mathbf{g}_Q^*(\boldsymbol{\theta})$ is optimal in the class $\mathcal{G}_Q$, and part (a) follows. Since m_t and ψ_t are orthogonal, the information $\mathbf{I}_{\mathbf{g}_Q^*}(\boldsymbol{\theta}) = \mathbf{I}_{\mathbf{g}_M^*}(\boldsymbol{\theta}) + \mathbf{I}_{\mathbf{g}_\Psi^*}(\boldsymbol{\theta})$ and part (b) follows. Hence, for each component $\theta_i, i = 1, \dots, p$, neither $g_M^*(\theta_i)$ nor $g_S^*(\theta_i)$ is fully informative, that is, $I_{g_Q^*}(\theta_i) \geq I_{g_M^*}(\theta_i)$ and $I_{g_Q^*}(\theta_i) \geq I_{g_S^*}(\theta_i)$.

Corollary A.2 *When the conditional skewness γ and kurtosis κ are constants, the optimal quadratic estimating function and associated information, based on the martingale differences $m_t = y_t - \mu_t$ and $s_t = m_t^2 - \sigma_t^2$, are given by*

$$\mathbf{g}_Q^*(\boldsymbol{\theta}) = \left(1 - \frac{\gamma^2}{\kappa + 2}\right)^{-1} \sum_{t=1}^n \frac{1}{\sigma_t^3} \left(\left(-\sigma_t \frac{\partial \mu_t}{\partial \boldsymbol{\theta}} + \frac{\gamma}{\kappa + 2} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \right) m_t + \frac{1}{\kappa + 2} \left(\gamma \frac{\partial \mu_t}{\partial \boldsymbol{\theta}} - \frac{1}{\sigma_t} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \right) s_t \right)$$

and

$$\mathbf{I}_{\mathbf{g}_Q^*}(\boldsymbol{\theta}) = \left(1 - \frac{\gamma^2}{\kappa + 2}\right)^{-1} \left(\mathbf{I}_{\mathbf{g}_M^*}(\boldsymbol{\theta}) + \mathbf{I}_{\mathbf{g}_S^*}(\boldsymbol{\theta}) - \frac{\gamma}{\kappa + 2} \sum_{t=1}^n \frac{1}{\sigma_t^3} \left(\frac{\partial \mu_t}{\partial \boldsymbol{\theta}} \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}'} + \frac{\partial \sigma_t^2}{\partial \boldsymbol{\theta}} \frac{\partial \mu_t}{\partial \boldsymbol{\theta}'} \right) \right).$$

B Trace Minimization in Estimating Functions

Estimating Functions

An estimating function $g(X, \theta)$ satisfies:

$$E[g(X, \theta_0)] = 0 \tag{1}$$

for the true parameter θ_0 . The estimator $\hat{\theta}$ is obtained by solving $g(\hat{\theta}) = 0$.

These generalize likelihood score functions.

Asymptotic Covariance and Trace Minimization

The asymptotic covariance of $\hat{\theta}$ is given by the Godambe information:

$$\text{Cov}(\hat{\theta}) \approx A^{-1} B A^{-T} \tag{2}$$

where:

$$A = E \left[\frac{\partial g}{\partial \theta} \right],$$

$$B = \text{Var}(g) = E[gg^T].$$

Heyde proposed minimizing the scalar measure of uncertainty via:

$$\min \text{tr}(A^{-1}BA^{-T}) \tag{3}$$

which corresponds to minimizing the total variance across all parameters. Further Heyde established **optimality conditions** for estimating functions minimizing $\text{tr}(A^{-1}BA^{-T})$ and applied this criterion to **martingale estimating functions**, particularly in time series and stochastic process models. Also she provided a general framework aligning with later developments in **GMM (Generalized Method of Moments)** and pioneered connections between statistical efficiency and quasi-likelihood estimation.

C Simulation Results of Moment Comparison

Table 1: Moment comparison: theoretical vs. empirical Under normal and t -distributions

df	Moment	Theoretical Normal	Empirical Normal	Maximum eigenvalue Normal	Theoretical t	Empirical t	Maximum eigenvalue t
3	1	1	1.000	5.784	3.000	2.940	37.869
3	2	3	2.998	5.784	27.000	28.297	37.869
3	3	11	10.998	5.784	297.000	385.544	37.869
3	4	45	45.045	5.784	3645.000	7292.908	37.869
4	1	1	1.003	5.747	2.000	2.014	41.860
4	2	3	3.024	5.747	12.000	13.537	41.860
4	3	11	11.153	5.747	88.000	159.808	41.860
4	4	45	45.915	5.747	720.000	3780.282	41.860
5	1	1	1.000	5.767	1.667	1.665	9.597
5	2	3	3.008	5.767	8.333	8.340	9.597
5	3	11	11.066	5.767	50.926	51.105	9.597
5	4	45	45.449	5.767	347.222	349.839	9.597
6	1	1	1.000	5.754	1.500	1.501	8.623
6	2	3	2.995	5.754	6.750	6.773	8.623
6	3	11	10.968	5.754	37.125	37.400	8.623
6	4	45	44.852	5.754	227.812	230.546	8.623
7	1	1	1.001	5.768	1.400	1.401	8.203
7	2	3	3.004	5.768	5.880	5.907	8.203
7	3	11	11.015	5.768	30.184	30.510	8.203
7	4	45	45.094	5.768	172.872	176.084	8.203
8	1	1	0.998	5.754	1.333	1.336	7.742
8	2	3	2.990	5.754	5.333	5.361	7.742
8	3	11	10.935	5.754	26.074	26.301	7.742
8	4	45	44.576	5.754	142.222	144.059	7.742
9	1	1	0.997	5.646	1.286	1.284	7.328
9	2	3	2.981	5.646	4.959	4.953	7.328
9	3	11	10.885	5.646	23.379	23.376	7.328
9	4	45	44.311	5.646	122.968	123.072	7.328
10	1	1	0.997	5.714	1.250	1.245	7.216
10	2	3	2.978	5.714	4.688	4.657	7.216
10	3	11	10.872	5.714	21.484	21.301	7.216
10	4	45	44.295	5.714	109.863	108.782	7.216
11	1	1	0.998	5.727	1.222	1.217	7.013
11	2	3	2.986	5.727	4.481	4.446	7.013
11	3	11	10.907	5.727	20.084	19.866	7.013
11	4	45	44.390	5.727	100.418	99.101	7.013
12	1	1	0.998	5.773	1.200	1.205	6.951
12	2	3	2.988	5.773	4.320	4.363	6.951
12	3	11	10.937	5.773	19.008	19.307	6.951
12	4	45	44.667	5.773	93.312	95.274	6.951
13	1	1	1.001	5.794	1.182	1.185	6.858
13	2	3	3.006	5.794	4.190	4.209	6.858
13	3	11	11.036	5.794	18.157	18.283	6.858
13	4	45	45.273	5.794	87.784	88.663	6.858
14	1	1	1.000	5.786	1.167	1.165	6.700
14	2	3	2.998	5.786	4.083	4.082	6.700
14	3	11	10.984	5.786	17.468	17.508	6.700
14	4	45	44.922	5.786	83.368	83.835	6.700
15	1	1	0.998	5.708	1.154	1.155	6.607
15	2	3	2.981	5.708	3.994	4.015	6.607
15	3	11	10.879	5.708	16.898	17.067	6.607
15	4	45	44.302	5.708	79.763	80.948	6.607

D Abbreviations used for Portfolios

Table 2: Abbreviations used for portfolios

Abbreviation	Detailed name of portfolio
MVP_emp	Minimum variance portfolio with weights calculated from empirical covariance matrix
MVP_emp_f	Minimum variance portfolio with weights calculated from RMT-filtered empirical covariance matrix
MVP_dd	Minimum variance portfolio with weights calculated from data driven covariance matrix
MVP_dd_f	Minimum variance portfolio with weights calculated from RMT-filtered data driven covariance matrix
MVP_neuro	Minimum variance portfolio with weights calculated from neuro covariance matrix
MVP_neuro_f	Minimum variance portfolio with weights calculated from RMT-filtered neuro covariance matrix
EP_emp	Tangency portfolio with weights calculated from empirical covariance matrix
TP_emp_f	Tangency portfolio with weights calculated from RMT-filtered empirical covariance matrix
TP_dd	Tangency portfolio with weights calculated from data driven covariance matrix
TP_dd_f	Tangency portfolio with weights calculated from RMT-filtered data driven covariance matrix
TP_neuro	Tangency portfolio with weights calculated from neuro covariance matrix
TP_neuro_f	Tangency portfolio with weights calculated from RMT-filtered neuro covariance matrix
EP_emp	Eigen portfolio with weights calculated from empirical covariance matrix
EP_emp_f	Eigen portfolio with weights calculated from RMT-filtered empirical covariance matrix
EP_dd	Eigen portfolio with weights calculated from data driven covariance matrix
EP_dd_f	Eigen portfolio with weights calculated from RMT-filtered data driven covariance matrix
EP_neuro	Eigen portfolio with weights calculated from neuro covariance matrix
EP_neuro_f	Eigen portfolio with weights calculated from RMT-filtered neuro covariance matrix