April 2024

Comparative Analysis of Time Series Models on U.S. Stock and Exchange Rates: Bayesian Estimation of Time Series Error Term Model Versus Machine Learning Approaches

Young Keun Yang
University of South Florida

Comparative Analysis of Time Series Models on U.S. Stock and Exchange Rates: Bayesian Estimation of Time Series Error Term Model Versus Machine Learning Approaches

by

Young Keun Yang

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Arts
Department of Mathematics & Statistics
College of Arts and Sciences
University of South Florida

Major Professor: Jiwoong Kim, Ph.D.
Seung-Yeop Lee, Ph.D.
Lu Lu, Ph.D.

Date of Approval:
March 29, 2024

Keywords: Generalized linear model(GLM), Markov chain Monte Carlo(MCMC), Frequency domain, Spectral density, Recurrent Neural Network(RNN)

DEDICATION

I dedicate this thesis to my loving family, whose unwavering support and encouragement have supported me throughout this journey. I will forever be grateful for your unconditional love, patience, and understanding. This achievement is yours. Thank you for always being there for me.

ACKNOWLEDGMENTS

I am deeply grateful to my supervisor, Dr. Kim, whose invaluable guidance, support, and encouragement have been integral to the completion of this thesis. Without his expertise, insightful feedback, and unwavering dedication, I would not have been able to progress in this work.

I extend my thanks to my thesis committee members, Dr. Lee and Dr. Lu, for their valuable feedback and constructive criticism, which played a crucial role in enhancing the quality of this thesis.

This thesis would not have been possible without the contributions and support of these individuals. Thank you for being part of this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This study presents a comparative analysis of contemporary applications of time series models, focusing on the Bayesian approach. In contrast to many nonparametric studies, the Bayesian approach circumvents the common issue of bandwidth selection by offering systematic estimation and avoiding ad hoc methods. Specifically, we delve into the Bayesian approach for estimating the autocovariance function of a time series model's error term. Traditional time series models often make the unrealistic assumption of a constant error term. Furthermore, models such as autoregressive conditional heteroskedasticity (ARCH) and general autoregressive conditional heteroskedasticity (GARCH) address the limitation of constant variance by assuming an autoregressive error term. However, this may still fail to capture the actual error accurately. The Bayesian method overcomes these assumptions by leveraging actual sample data and employing the Markov Chain Monte Carlo (MCMC) method for parameter estimation. We elucidate the transformation of the time series error term into the frequency domain via spectral density. Spectral density, a population concept, is estimated using a periodogram with discrete Fourier transform. Utilizing sample data, we construct the periodogram in the frequency domain. Asymptotically, the periodogram follows an exponential distribution approximated by a mixture of five Gaussian distributions. This thesis thoroughly examines the Bayesian method and evaluates its efficacy using real-world time series data, such as exchange rates and stocks. Performance assessment involves comparing traditional time series autoregressive (AR) (1) models and machine learning recurrent neural network (RNN) models, which provide valuable reference points for analysis. The Bayesian model outperforms the AR model by effectively capturing and reflecting fluctuation patterns in the data. The Bayesian model's root mean squared error (RMSE) was consistently smaller than that of the AR(1) process. This indicates that the errors of the Bayesian method were smaller than those of AR(1), indicating superior performance of the Bayesian model compared to AR(1).

CHAPTER 1:
INTRODUCTION

1.1 Background

Time series data is one of the most prevalent data types in various domains. Any dataset involving time, regardless of the domain, falls under this category. From routine activities like tracking a baby's weight to analyzing complex phenomena such as minute-by-minute stock market behavior, time series data permeates numerous aspects of life. Given the integral role of time in our daily routines, time series data is omnipresent.

Among various topics within time series data, stock market data is prominent due to its substantial role in capitalist societies. The stock market consistently dominates daily news headlines, reflecting its importance in financial systems. Many companies and individuals actively seek financial gain from stock market investments, leading to a continuous influx of data. The allure of stock market investments, driven by capitalist principles, extends worldwide, with almost every country having its stock market, resulting in over 130 stock markets globally. [1] Despite the vast number, there are approximately 20 major stock markets worldwide. While every investor aims to predict individual stock prices accurately, inherent complexities make prediction challenging. Consequently, achieving near-precise stock price forecasting that closely aligns with actual outcomes would unlock the key to successful investment. The pursuit of financial gain continually attracts the interest of academia and industry in stocks, with thousands of financial analysts worldwide releasing reports on stock prices. Despite extensive analysis, successful prediction remains challenging, even at the company level, making it even more difficult for individuals.

The 2020 stock market chart events starkly demonstrated the challenge of forecasting stock prices. The sudden emergence of COVID-19 in December 2019 caught the world off guard and defied all prior predictions. This unexpected virus not only took a toll on human health but also had profound implications for global stock market activity. Throughout 2020, the influence of COVID-19 on the stock market was substantial, resulting in considerable fluctuations characterized by sharp declines and followed by rapid recoveries. The uncertainties stemming from the pandemic prompted heightened scrutiny of financial markets and spurred dynamic shifts in investment strategies and stock valuations. Adding complexity to this evolving landscape, ChatGPT, a generative language model (GLM) or large language model (LLM) that emerged on November 30, 2022, ushered in a new era of interest in the stock market intricacies infused with artificial intelligence

(AI). Constructed from a vast dataset of human language sentences, the LLM is adept at generating language tailored to specific contexts. Through exposure to billions of sentences, it has acquired the ability to respond with the most appropriate words and phrases based on its extensive training. This versatile tool can tackle various tasks, from sentence editing to answering general knowledge questions and coding assignments. While it may occasionally provide inaccurate responses in specialized domains, it generally offers reasonably accurate information, particularly concerning common, publicly available knowledge. Many anticipate substantial advancements in LLM technology, with its increasing sophistication expected to profoundly impact various aspects of human life, akin to the transformative effects of inventions such as the steam engine, cars, computers, the internet, and smartphones.

Consequently, there is a natural inclination for individuals to invest in AI-related companies, reflecting an inherent trend. The introduction of ChatGPT also resulted in a notable surge in specific stock prices. Notably, Nvidia, a prominent technology firm renowned for its graphics processing unit (GPU) innovations, witnessed a meteoric ascent. As it turned out, the GPU played a pivotal role in LLMs. Thus, within seven months of ChatGPT's inception, Nvidia's stock price had tripled. This underscores both the transformative influence of AI advancements and the inherent volatility in stock markets. Nvidia's extraordinary trajectory sparked heightened market interest in technology-driven sectors, serving as a motivation factor for this research.

While the emergence of AI has spurred the development of new machine learning techniques for predicting time series data, traditional time series models retain certain advantages over their modern counterparts. Despite the superior predictive accuracy often exhibited by machine learning, particularly neural networks, these models lack explicit insight into the influence of individual inputs. This characteristic, called "black-box" analysis, hinders researchers' ability to draw meaningful inferences from the data. In contrast, traditional time series models calculate distinct coefficients for each factor, enabling practitioners to interpret the impact of each variable. Consequently, there remains a continued interest in refining traditional models. A substantial research challenge lies in the underlying assumptions of basic conventional time series models. Due to computational or theoretical constraints, these models frequently rely on assumptions that may not align with real-world scenarios. For instance, the widely utilized autoregressive integrated moving average (ARIMA) model assumes constant variance in the error term; a condition rarely met in practice. More sophisticated models have been developed to address this limitation, such as autoregressive conditional heteroskedasticity (ARCH) or general autoregressive conditional heteroskedasticity (GARCH). Unlike ARIMA, ARCH, and GARCH do not assume constant variance but model variance autoregressively. However, they still presuppose that the error term follows a normal distribution, which may not hold for many real-world datasets, particularly those characterized by high volatility, such as stock prices. More-

over, ARCH and GARCH processes inherently assume uncorrelated errors, which may not reflect reality. Conversely, models incorporate because the variable of interest is not in the self-dependent structure. However, this assumption may not hold in the real world. Conversely, models incorporating heteroskedastic time-varying volatility imply that error terms are dependent and unconditionally correlated, as noted by Kim and Kim. [2] A nonparametric approach has emerged as an appealing, attractive alternative to address these limitations. "Nonparametric" signifies a departure from conventional parametric modeling, although some nonparametric methods may still entail weak parametric assumptions, leading to their classification as semi-parametric. Despite this nuance, "nonparametric" is commonly used for consistency. Nonparametric methods, including kernel density estimation (KDE), can derive probability density function (PDF) from data without rigid assumptions about their underlying distribution. Instead, they construct PDFs based on the collected data, allowing for flexible adaptation over time. KDE, a notable technique in this category, employs kernel smoothing to estimate PDF, as expressed below:

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)$$

In this equation, $h > 0$ is often referred to as the bandwidth, serving as a smoothing parameter for the function. K represents the kernel, a function possessing symmetric and normalization properties (such as its integral over the entire real line equaling 1), essential for shaping the estimation. Commonly utilized kernel functions comprise Gaussian, uniform, triangle, and quadratic.

However, even this nonparametric approach has its challenges. Selecting an appropriate bandwidth remains a critical concern, often requiring the application of rule-of-thumb techniques in practice. A rule-of-thumb method can be ad hoc, prompting the need for a more systematic approach for researchers' use. This study investigates a Bayesian approach for estimating the density function to overcome these limitations. By introducing the autocovariance error term from a Bayesian perspective, this research aims to provide novel insights into addressing bandwidth selection issues, thereby enhancing the accuracy and applicability of nonparametric estimations for time series data. This approach can introduce novel avenues for refining forecasting techniques in time series analysis.

1.2 Goal

The primary objective of this study is to investigate a time series model proposed by Dey [3], apply it to forecast future values in financial market data, and assess its performance compared to a current machine learning method. Existing time series models face a major challenge due to their reliance on parametric assumptions. By employing Bayesian methods to estimate the autocovariance of error terms, this model is expected to outperform traditional time series approaches, such as the AR process.

One of the key difficulties with current time series models stems from their parametric assumption, which categorizes them as parametric models. These models operate on the premise of a specific parametric distribution, necessitating the selection of a single probability distribution to construct the model. Typically, a normal distribution is chosen for computational ease or convenience. However, further exploration within the parametric model becomes futile if the underlying assumption is not met during analysis. Despite contributing to advancing statistical models based on robust theory, parametric models often prove unsuitable for real-life data, particularly those exhibiting high volatility. Nonparametric models have emerged as a promising alternative to address this limitation. They offer distinct advantages over parametric models, particularly in requiring fewer assumptions about the underlying populations derived from collected data. This contrasts with many traditional parametric models that assume normality in the underlying populations. [4]

Conventional time series models typically rely on assumptions of white noise (as seen in ARIMA models) or autoregressive conditional heteroscedasticity (ARCH, GARCH). Heteroskedasticity is frequently observed in financial return processes or macroeconomics and plays a vital role in economic time series. [5] However, when these assumptions do not hold for real-world data, the resulting predictions can deviate substantially from the actual data trends. In contrast, the Bayesian methods in this study operate semi-operatively, with weaker assumptions regarding autocovariance, enabling them to better adapt to the complexities inherent in real-world data. Similarly, the application of KDE presents challenges in selecting an appropriate bandwidth. With Bayesian estimation, this concern is mitigated, thereby enhancing the robustness of the estimation. Consequently, the primary aim of this study is to examine a model that outperforms the traditional time series model.

Furthermore, a secondary objective of this research is to perform a comparative analysis by utilizing machine learning techniques. In the era of burgeoning AI, machine learning algorithms have become increasingly accessible, and complex models can now be easily implemented. Neural networks have proven remarkably adept at forecasting tasks among machine learning algorithms. Specifically, for sequential data such as time series data, the use of RNNs has garnered attention for their efficacy.

Establishing a robust baseline is crucial for assessing the performance of a model. In this context, renowned for their predictive capabilities, neural network models are a suitable benchmark. Through a thorough comparison, our goal is to ascertain how the performance of our Bayesian-based time series model aligns with the predictive power demonstrated by advanced machine learning algorithms. This approach will clarify the model's effectiveness across various scenarios, contributing to a more comprehensive understanding of its potential applications. Bayesian estimation avoids assuming a specific parametric form for any time-dependent structure. Instead, it enables the incorporation of relevant information into covariance matrix estimation through the prior spectral density distribution. This method helps to mitigate the risk of model misspecification. [6]

CHAPTER 2:
LITERATURE REVIEW

This study aims to compare time series models with nonparametric Bayesian estimation of the error term autocovariance. Nonparametric research has been conducted on time series modeling. Jonestone and Silverman [7] examined an estimator based on wavelet thresholding designed for data with correlated noise. Robinson [8] explored nonparametric regression within the context of a linear process by incorporating semiparametric modeling of spectral density. Martins-Filho and Yao [9] derived the asymptotic distribution of a local linear estimator for nonparametric regression in the presence of dependent errors by incorporating a general parametric covariance model. Except for Su and Ullah [10], previous researchers assumed specific parametric and semiparametric autocovariance functions. In contrast, Su and Ullah [10] assumed that the error possesses a finite-order nonlinear structure. Given the considerable challenge of directly estimating the autocovariance function of the error term, an alternative approach is necessary. Our chosen method utilizes spectral density, analogous to the frequency domain probability density function.

Spectral density operates within the frequency domain as an analytical tool for frequency analysis. Frequency and time domain share numerous similarities, with frequency being a series that often exhibits cyclical or periodic behavior. Spectral analysis has been instrumental in identifying amplitudes of periodic frequencies and breaking the series down into distinct cycles or periods, similar to how a prism decomposes colors into primary colors (spectrum). The term "spectral" is derived from the light spectrum. Due to this parallel, researchers have extensively explored the use of this method for studying both time series and frequency, revealing that the Fourier transformation can convert the autocovariance of a stationary time series into a unique spectral distribution function in the frequency domain when the autocovariance function is summable. [11]

Furthermore, the inverse Fourier transformation allows the autocovariance function to be reconstructed from the spectral distribution function. Consequently, spectral density will be the primary tool for analyzing the autocovariance function of the error term in the model presented in this study. Additional steps are required before utilizing the Bayesian method. First, we estimate the prior log of the autocovariance function in the frequency domain. Subsequently, the prior is updated using the Whittle approximation for the likelihood function. [12] Due to the complexity of computing this posterior, we resort to an approximation.

This approximation uses a mixture of five known Gaussian distributions with known proportions, introduced by Carter et al. [13], to estimate the posterior method. We employ the periodogram, a nonparametric method for estimating spectral density, to facilitate this approximation. The detailed mathematical expression and process will be elucidated in the main body of the paper.

Once the model is constructed, evaluating its performance becomes crucial. This assessment can be conducted by comparing its performance against traditional statistics and time-series models, including, but not limited to, AR, OLS, and random walk. If the model demonstrates satisfactory performance in this comparison, it is evaluated against machine learning models. Previous efforts have utilized machine learning and traditional time modeling across various domains. Given the well-established history and comprehensive explanations of time series models, as detailed in [11], this section focuses on machine learning models for exploring time series data. Various models have been employed, and our investigation will explore the different models utilized for specific topics. Machine learning has become commoditized and can be applied across domains where time is a dependent variable, allowing prevalent models to be categorized into various industries.

Slater et al. [14] employed a hybrid model combining traditional methods and neural networks for climate prediction. Kisi et al. [15] utilized a combination of neural networks, an adaptive-neuro-fuzzy inference system (ANFIS), and gene expression programming (GEP) to forecast daily lake levels. Wang et al. [16] endeavored to predict water discharge using ARIMA, neural network, ANFIS, genetic programming (GP), and support vector machine (SVM). Liu et al. [17] utilized machine learning techniques to forecast natural gas consumption. Adil Masood et al. [18] applied AI techniques to predict air pollution, comparing artificial neural networks (ANN), deep neural networks (DNN), SVM, and fuzzy logic.

Yaseen et al. [19] conducted stream flow forecasting, which is crucial for water resource management. Predicting stream flow is challenging due to its substantial implications for human life. Traditional forecasting methods, such as ARIMA, ARIMAX, and linear regression, rely on linear assumptions and may not capture nonlinear, nonstationary relationships. Researchers have developed an innovative AI approach that can mimic hydrologic models to address these limitations.

Lipu at el. [20] investigated AI-based wind power generation forecasting. Electricity is undoubtedly essential for human life, and its demand is increasing. According to various studies, fossil fuels account for 63% of electricity production, while renewable energy comprises up to 25% of the world's energy production. This study focuses on wind energy among the few commercialized renewable energy sources. Because electricity generation from wind is subject to natural conditions, wind power is unstable and intermittent, posing challenges for accurate production forecasting. However, power grids must maintain reliability for practical usage. Otherwise, insufficient capacity may lead to power outages, while excess reserve capacity

could increase operational costs. Therefore, precise wind power forecasting is crucial for cost savings and promoting wind energy utilization. The article reviews previous research on wind power forecast models, summarizes their findings, and proposes a new hybridization approach.

Non-academic, industry-related studies have also been extensively conducted. Amirrkolaii et al. [21] employed neural networks for demand forecasting for irregular aircraft spare parts, while Kaushik et al. [22] utilized an ARIMA neural network, LSTM, and ensemble for predicting healthcare expenditures. With the ever-increasing volume of health data, healthcare research has focused on patient privacy concerns. Anonymization, which serves as the link between data and individuals, has become commonplace. Kaushik et al. [22] attempted to predict the cost of two pain medications, one of which is among the most popular medicines in the US. In contrast to previous research, this article aimed to construct an ensemble model comprising statistical and neural network models. Two models were utilized in the literature, comprising one statistical model and one neural network model. The study employed the persistence model (AR) among the statistical time-series models. The remaining model is recurrent neural network (RNN) models. The study aimed to identify the best model by combining the predictions of these two models, employing an ensembling architecture approach. A weighted average ensemble, which dynamically weights the best predictions of individual models, was utilized for the ensembling approach. Dropout, a technique of regularization and data shuffling, was employed to enhance the general learning of the data and address the common problem of overfitting. In separate research endeavors, Hadavandi et al. [23] investigated tourist arrival forecasting using a hybrid AI model consisting of a genetic algorithm (GA) and genetic fuzzy systems. Daut et al. [24] applied a hybrid AI model utilizing swarm intelligence (SI) and SVM to forecast electric consumption in a commercial building. Wang et al. [25] examined crude oil price prediction using a hybrid AI model and a data fluctuation network (DFN), along with several AI algorithms. Raza et al. [26] investigated electrical load forecasting for building a smart grid using hybrid ANN models, including wavelet-based neural networks (WNN), fuzzy logic, and SVM.

Studies on the stock market have also been conducted. Ding et al. [27] employed a convolutional neural network (CNN) combined with natural language processing (NLP) to predict stock prices. The model constituted an event-driven stock prediction model that analyzed real-life events affecting stock prices using NLP and processed them utilizing the CNN model. Wei Li et al. [28] utilized a hybrid LSTM and relational graph convolutional network model to investigate the relationship between previous close and open prices. They used a relational graph convolutional network to evaluate Reuters Financial News during stock market closure to evaluate stock connections. Akita et al. [29] also employed LSTM, paragraph vector, and deep learning to forecast financial time series. They utilized newspaper articles in paragraph vectors and modeled the effects of events on opening prices using LSTM. Soni [30] surveyed ANN for stock prediction using stock

market data. Hadavandi et al. [31] used genetic fuzzy systems and ANNs to forecast stock prices. They utilized stepwise regression to identify factors and divided them into k clusters using a self-organizing map. They independently fed those clusters into genetic fuzzy systems and utilized data from the IT and airline sectors. Hassan and Nath utilized the hidden Markov model for stock market forecasting. Xing et al. [32] employed natural language processing for forecasting stocks. Ren et al. [33] utilized sentiment analysis and SVM to predict the stock market, while Yan et al. [34] employed a Bayesian regularization neural network based on AI optimization.

Researchers have explored not only the US market but also other global markets. Tsang et al. [35] utilized a neural network and SVM to build a stock price forecasting model for the Hong Kong market. Nayak et al. [36] examined the impact of data normalization on an ANN model for stock index forecasting in the Indian market. Chen et al. [37] employed a backpropagation neural network, extreme learning machine, and radial basis function neural network for predicting the Chinese stock market. Wei et al. [38] applied an adaptive network fuzzy inference system to forecast the stock market in Taiwan. Huynh et al. [39] utilized AI for cryptocurrency forecasting.

Overall, the most commonly used models in previous studies included SVM, evolutionary algorithms families (evolution strategies (ES), genetic (evolutionary) programming, GA), fuzzy logic families (adaptive-neuro fuzzy inference systems and genetic fuzzy systems), and neural networks families (CNN, ANN, RNN, DNN, WNN, LSTM). In the following section, we will briefly outline the fundamental concepts of these commonly used models. An SVM is a generalized technique involving optimal separating hyperplanes commonly employed in classification or regression models. [40] The algorithm can be applied more generally to independent and dependent variable modeling rather than being a specialized method specifically for time series.

Another commonly used method in the literature is the evolutionary algorithm (EA). Paulo et al. [41] compared the performance of traditional time series models, such as exponential smoothing and ARIMA, and concluded that evolutionary algorithms outperformed these traditional models. The three main streams within evolutionary algorithms are evolutionary strategies, evolutionary programming, and GAs. [42] Evolutionary computation emerged in the 1950s and 1960s as early computer science researchers aimed to derive optimized solutions for engineering problems. The term "evolution" was incorporated into its name to mirror the natural process of expanding the population of living organisms on Earth, involving genetic variation and natural selection. [43] Rechenberg introduced "evolution strategies" as an optimization method for real-value device parameters like airfoils. [44]

In 1966, Fogel introduced "evolutionary programming," wherein candidate solutions for specific tasks were represented as finite-state machines. These machines evolved through random mutations in their

state-transition diagrams, with the fittest being selected. [45] GAs originated in the 1960s and 1970s, with John Holland playing a pivotal role in their development. Unlike evolution strategies and evolutionary programming, Holland's primary aim was not to create algorithms tailored to specific problems but to systematically explore natural adaptation and integrate the mechanisms of natural adaptation into computer systems.

Holland's influential book, "Adaptation in Natural and Artificial Systems," [46], introduced the GA as an abstraction of biological evolution, providing a theoretical framework for adaptation within GAs. The GA designed by Holland facilitates the transition from one population of "chromosomes" (e.g., sequences of ones and zeros or "bits") to a new population through a process of "natural selection" combined with genetics-inspired operations, such as crossover, mutation, and inversion. [47]

CHAPTER 3:
METHODOLOGY

3.1 Basic Linear Structure

A linear relationship is often considered a reasonable starting point when investigating the relationship between two variables. Utilizing a linear model is a prevalent approach for establishing the relationship between dependent and independent variables in statistical analysis. Notably, a time series model can also be conceptualized as linear. When representing a time series within a linear model framework, the relationship at a specific time point, denoted as t, can be expressed as follows:

$$y_t = f(\mathbf{x}_t) + \varepsilon_t, \quad t = 1, ..., T$$

where y_t represents the dependent variable of interest, $\mathbf{x}_t$ is a vector of n independent variables, $f : \mathbb{R}^n \to \mathbb{R}$ is a function defining the relationship between the dependent and independent variables, and ε_t denotes the error term.

Several nonparametric methods for estimating the function f such as spline smoothing, kernel smoothing, and wavelet methods, have been explored for independent errors ε_t. [48] We assume that $\{\varepsilon_t\}_{t=0}^{T}$ is a weakly stationary process with a mean of 0, constant variance, and an autocovariance function γ. Further investigation into the estimation of f has been carried out in cases where ε_t exhibits autocorrelation, as examined by Hall and Hart. [49] Then, for any positive integer k, γ can be explicitly expressed as:

$$\gamma(k) = Cov(\varepsilon_t, \varepsilon_{t+k}) = E[(\varepsilon_t - \mu_t)(\varepsilon_{t+k} - \mu_{t+k})]$$

where $\mu_t = E[\varepsilon_t]$.

A common assumption in time series modeling is that the model exhibits a constant variance. While this assumption holds theoretical significance, it proves inadequate for high-volatility data, such as stock prices, as it does not align with the characteristics of the data. Most high-volatility datasets deviate from constant variance, rendering the basic time series model less effective due to the mismatch between the model assumption and the data. Consequently, there arises a need for alternative models that do not presuppose

constant variance in error terms. This study proposes alternative nonparametric methods to address this issue. Specifically, this study investigates a Bayesian approach for estimating the autocovariance function of a stationary process $\{\varepsilon_t\}_{t=0}^{T}$ where the entire model can be expressed as:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{3.1}$$

where

$$\begin{aligned}
\boldsymbol{y} &= (y_1, ..., y_T)' \in \mathbb{R}^T, \\
\boldsymbol{\beta} &= (\beta_1, ..., \beta_p)' \in \mathbb{R}^p, \\
\boldsymbol{\varepsilon} &= (\varepsilon_1, ..., \varepsilon_T)' \in \mathbb{R}^T, \\
\boldsymbol{X} &= \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{T1} & x_{T2} & \dots & x_{Tp} \end{bmatrix} \in \mathbb{R}^{T \times p}.
\end{aligned}$$

Let $\boldsymbol{\Gamma}_{T\times T}$ denote an autocovariance matrix of $\boldsymbol{\varepsilon}$. For any independent error terms ε_s and ε_t, $Cov(\varepsilon_s, \varepsilon_t) = 0$. Therefore,

$$\boldsymbol{\Gamma}_{T\times T} = \begin{bmatrix} Cov(\varepsilon_1,\varepsilon_1), & Cov(\varepsilon_1,\varepsilon_2) & \dots & Cov(\varepsilon_1,\varepsilon_T) \\ Cov(\varepsilon_2,\varepsilon_1), & Cov(\varepsilon_2,\varepsilon_2) & \dots & Cov(\varepsilon_2,\varepsilon_T) \\ \vdots & \vdots & \ddots & \vdots \\ Cov(\varepsilon_T,\varepsilon_1), & Cov(\varepsilon_T,\varepsilon_2) & \dots & Cov(\varepsilon_T,\varepsilon_T) \end{bmatrix} = \begin{bmatrix} Var(\varepsilon_1) & 0 & \dots & 0 \\ 0 & Var(\varepsilon_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Var(\varepsilon_T) \end{bmatrix}$$

which is $\gamma(0)\boldsymbol{I}_{T\times T}$ where $\boldsymbol{I}_{T\times T}$ is a $T \times T$ identity matrix. Since $\mathbf{X}$ is known, we need to estimate $\boldsymbol{\beta}$ and $\gamma(0)$. We can use conjugate families for a clear calculation when applying the Bayesian approach to estimating $\boldsymbol{\beta}$ and $\gamma(0)$. Let $\gamma_0 := \gamma(0)$. Next, let the prior distributions of $\boldsymbol{\beta}$ and $\gamma(0)$ be a multivariate normal distribution with a mean vector of μ_β and a covariance of Σ and an inverse gamma distribution with a shape parameter a and scale parameter b. Then, by applying the conjugate property, the posterior distribution of $\boldsymbol{\beta}$ would also follow a normal distribution, and the posterior distribution of γ_0 would also follow the inverse-gamma distribution.

While the linear structure is a potent model for elucidating the relationship between independent and dependent variables, it often falls short of explaining time series data adequately. A linear model primarily identifies the relationships between the current dependent and independent variables. However, in many

cases, the dependent variable in time series data is intricately linked to past dependent and independent variables. For instance, if a company's stock price closed at $100 yesterday, today's stock price will likely start around $100; it would hardly commence from extremes such as $1 or $10,000. This demonstrates that the present value is influenced by past values. Therefore, it would be advantageous to incorporate past data into explanatory variables for modeling. This concept is known as an autoregressive format and will be elaborated on in the subsequent chapter.

3.2 Frequency Domain

If γ_0 is known, calculating the posterior distribution of $\boldsymbol{\beta}$ can be easily calculated. However, when γ_0 is unknown, $\boldsymbol{\beta}$ can be estimated by transforming the time domain of the series into the frequency domain. Both time and frequency domains exhibit similar characteristics, revealing the cycles in the data. In the frequency domain, a cycle is typically defined as one complete period of a sine or cosine function within a unit time interval. [11] Since the definition of a cycle involves sine or cosine functions, we can consider a sinusoidal waveform model, which can be conceptualized as a periodic process:

$$p_t = A\cos(2\pi\omega t + \phi), \tag{3.2}$$

where A represents the amplitude, ω signifies the oscillation frequency, ϕ denotes a phase shift, and t represents a specific time point. Since a sine function can be transformed into a cosine function and vice versa using the trigonometric identity $\sin(\frac{\pi}{2} - \theta) = \cos(\theta)$, representing a random sinusoidal waveform is effectively accomplished. Moreover, A, ω, and ϕ suffice to encapsulate all variations within any sinusoidal waveform. However, it is common for A and ϕ to be unknown. Therefore, expressing equation 3.2 in the following form proves to be useful:

$$A\cos(2\pi\omega t + \phi) = A\cos(\phi)\cos(2\pi\omega t) - A\sin(\phi)\sin(2\pi\omega t),$$

The equality follows from another trigonometric identity,

$$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta).$$

Substituting U_1 for $A\cos(\phi)$ and U_2 for $-A\sin(\phi)$, the equation becomes:

$$p_t = U_1\cos(2\pi\omega t) + U_2\sin(2\pi\omega t). \tag{3.3}$$

Then the amplitude A is given by $\sqrt{U_1^2 + U_2^2}$, and the phase ϕ is obtained by $\tan^{-1}\left(-\frac{U_2}{U_1}\right)$, where $\frac{U_2}{U_1} = \frac{-A\sin(\phi)}{A\cos(\phi)} = -\tan(\phi)$. Assuming that U_1 and U_2 are random variables with mean 0 and variance σ^2 and are not correlated, then p_t in (3.3) is a stationary process with $E(p_t) = 0$. Let $\gamma_p(h)$ be the autocovariance function of p_t. Notably,

$$\begin{aligned} \gamma_p(h) &= Cov(p_{t+h}, p_t), \\ &= \sigma^2(\cos(2\pi\omega(t+h))\cos(2\pi\omega t) + \sin(2\pi\omega(t+h))\sin(2\pi\omega t)), \\ &= \sigma^2\cos(2\pi\omega(t+h) - 2\pi\omega t), \\ &= \sigma^2\cos(2\pi\omega h), \end{aligned} \tag{3.4}$$

where the second-to-last inequality follows from the trigonometric identity. Note that $Var(p_t) = \gamma_p(0) = \sigma^2$. We can use the sample variance S^2 as an unbiased estimator to estimate σ^2. If we observe $U_1 = u$ and $U_2 = v$, then $S^2 = \frac{1}{2-1}(u^2 + v^2) = u^2 + v^2$, the estimate of σ^2. Because σ^2 was the variance of U_1 and U_2 and the mean of U_1 and U_2 is 0, the variance of the sample is $S^2 = \frac{1}{n-1}[(u-0)^2 + (v-0)^2]$, where n denotes the sample size, in this case 2. Because this applied to one periodic series, we can generalize it by summing a mixture of multiple sinusoidal amplitudes and frequencies as follows:

$$p_t = \sum_{k=1}^{q}[U_{1_k}\cos(2\pi\omega_k t) + U_{2_k}\sin(2\pi\omega_k t)],$$

where U_{1_k}, U_{2_k} for $k = 1, 2, \ldots, q$ are uncorrelated random variables with mean 0 and variance σ_k^2, and ω_k are distinct frequencies. Then, following the process of (3.4), the autocovariance function γ_p for a time difference h would be

$$\gamma_p(h) = \sum_{k=1}^{q}\sigma_k^2\cos(2\pi\omega_k h).$$

Therefore, the autocovariance function of a random periodic series p_t can be expressed as the sum of cosine functions with amplitude σ_k^2. Because the cosine function is centered at 0 without a phase, p_t is a stationary process with a mean of 0 and variance $\gamma_p(0)$, explicitly

$$\gamma_p(0) = cov(p_t, p_{t+0}) = var(p_t) = \sum_{k=1}^{q}\sigma_k^2$$

Hence, the variance of the whole process $\{p_t\}$ is the sum of each component k, σ_k^2. Similar to the single period case, when we observe $U_{1_k} = u_k$ and $U_{2_k} = v_k$ for $k = 1, \ldots, q$, the estimate for σ_k^2 would be $S_k^2 = u_k^2 + v_k^2$. The estimate of the total variance σ^2 would be

$$\hat{\gamma}_p(0) = \hat{Var}(p_t) = \sum_{k=1}^{q}(u_k^2 + v_k^2)$$

3.3 Spectral Density

The spectral density is a fundamental tool in the frequency domain, enabling a deeper understanding of stationary processes. Just as Wold decomposition [50] elucidates the application of linear regression in time series analysis, theorems of spectral representation provide a theoretical basis for decomposing stationary time series in the frequency domain. This decomposition unveils the inherent variance of the series. For a stationary process p_t with a constant frequency ω_0, it can be expressed as:

$$p_t = U_1 \cos(2\pi\omega_0 t) + U_2 \sin(2\pi\omega_0 t),$$

where U_1 and U_2 are random variables with a mean of 0 and are uncorrelated, both having equal variance σ^2. Since the cosine and sine functions exhibit a cycle of 2π, one cycle for p_t necessitates a time period of ω_0^{-1}. The process p_t completes ω_0 cycles for each time point t. With Euler's formula $e^{i\alpha} = \cos(\alpha) + i\sin(\alpha)$, $\gamma(h)$ can be expressed as:

$$\gamma(h) = \sigma^2 \cos(2\pi\omega_0 h) = \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{2\pi i\omega h} dF(\omega).$$

If an autocovariance function $\gamma(h)$ for a stationary process p_t satisfies the following condition:

$$\sum_{h=-\infty}^{\infty} |\gamma(h)| < \infty,$$

then according to the theorem by Shumway and Stoffer [11], the spectral density of p_t is given by

$$f(\omega) = \sum_{h=-\infty}^{\infty} \gamma(h) e^{-2\pi i\omega h}.$$

A periodogram is a sample-based concept, whereas spectral density is a population-based concept. When data $\{p_t\}_{t=1}^n$ are given, the discrete Fourier transform is defined as follows:

$$d(\omega_j) = n^{-\frac{1}{2}} \sum_{t=1}^{n} p_t e^{-2\pi i\omega_j t},$$

where $j = 0, 1, \ldots, n-1$, and $\frac{j}{n}$ is called the fundamental frequency or the Fourier frequency. The periodogram $I(\omega_j)$ is then defined as $I(\omega_j) = |d(\omega_j)|^2$. This representation indicates that any stationary process can be viewed as a random sum of cosines and sines.

3.4 Markov Chain Monte Carlo Procedure

3.4.1 Introduction

Due to the inherent self-dependency within the basic structure of time series models, it is imperative to incorporate the most recent data point into the model. Utilizing the latest data, such as yesterday's observations, is deemed more valuable than older data, such as those from a year ago. The most recent data point is considered paramount for forecasting the subsequent value. Consequently, data at time t holds the highest efficacy in estimating the data at time $t + 1$. This principle also holds for traditional time series models like ARIMA or GARCH, where coefficients associated with older data points are inherently smaller than those linked to more recent observations.

Time series data often lends itself to analysis using Bayesian statistics methods such as MCMC, particularly employing the Gibbs sampler. Within MCMC techniques, the Gibbs sampler is a specific instance of the Metropolis-Hastings algorithm, which has proven especially useful for estimating multivariate cases. The underlying principle of the Gibbs sampler lies in its ability to address challenges in sampling from the joint distributions. Instead of directly sampling from the joint distribution and then marginalizing other parameters, the Gibbs sampler samples from the conditional distribution for each parameter, facilitating the identification of specific parameters. Given the complexity of the model, employing Gibss sampling is a prudent choice. This study adopts the Bayesian estimation process outlined in [3]. Having a root in [3], following sections will borrow the notations and summarize the proposed method in detail: e.g., posterior probability distributions of parameters. In this model, two variables require estimation, and sampling from the joint distribution may be arduous, with marginalization potentially infeasible. Therefore, employing Gibbs sampling for $\boldsymbol{\beta}$ and $\gamma(0)$. The steps for Gibbs sampling are as follows:

1. The initial step involves estimating $\gamma(0)$ and $\boldsymbol{\beta}$. Given $\gamma(0)$, $\boldsymbol{\beta}$ can be determined. Therefore, we begin by assigning random values to the parameters a and b of inverse-Gamma distribution that $\gamma(0)$ follows. Let a and b be denoted as a_0 and b_0, respectively, with random values.

2. The subsequent step entails assuming that a_0 and b_0 represent the real parameter of $\gamma(0)$. With these values, it becomes feasible to determine Γ_{n*n}, indicating that $var(\varepsilon)$ is known, thereby facilitating the sampling of $\boldsymbol{\beta}$ values.

3. Repeat steps 1-2 until the parameters converge.

Given that γ is defined based on the expected statistical value, it exists for every lag h. Consequently, in the frequency domain ranging from $-\infty$ to ∞, there exists a monotonic function $s(\tau)$ such that

$$s(\tau) = \sum_{k=-\infty}^{\infty} \gamma(k) e^{-2\pi i k \tau} \qquad for \quad \tau \in [0, 1)$$

where $i = \sqrt{-1}$. This represents a commonly utilized transformation from the time domain to the frequency domain known as the Fourier transformation, with the function $s(\tau)$ referred to as the spectral density. Assuming that ε_t is invertible and γ is absolutely summable, we can express $s(\tau)$ in the inverse Fourier transform as follows:

$$\gamma(k) = \int_0^1 s(\tau) e^{2i\pi k \tau} d\tau$$

Therefore, we now obtain $\gamma(k)$ back from the spectral density. This implies we can derive the autocovariance function γ when modeling the spectral density using the inverse Fourier transform. Since $s(\tau) > 0$, and considering that $0 \leq \inf_\tau s \leq \sup_\tau s \leq \infty$, we can take logarithm of $s(\tau)$ and define $\theta(\tau)$ as $\theta(\tau) = \log(s(\tau))$. We assume that θ follows a Gaussian process with a mean function ι and a covariance kernel κ. These steps can be applied to the Bayesian framework. The same concept applies to $\gamma(0)$ and $\boldsymbol{\beta}$. Assuming that θ follows a Gaussian process, we can consider θ to have a Gaussian process prior. As assumed earlier, $\boldsymbol{\beta}$ in the Gaussian process is distributed as $\boldsymbol{\beta} \sim N(\mu_\beta, \sigma_0^2 \boldsymbol{I_n})$. However, calculating $\theta(\cdot|s, \gamma, \boldsymbol{\beta})$ is not straightforward. The variables are interrelated, making their integration complex. One approach to address this challenge is to leverage the discrete nature of the data. Since time data comprises a finite number of observations, we can utilize the discrete inverse Fourier transform, expressed as:

$$\gamma_m(k) = \frac{1}{m} \sum_{i=0}^{m-1} s(\tau_i) e^{2i\pi k \tau_i} \tag{3.5}$$

where $\tau_i = i/m$ represents the Fourier frequency. As $m \to \infty$, $\gamma_m(k) \to \gamma(k)$, so by obtaining $s(\tau)$ from $\theta(\tau)$ for $i = 0, 1, \cdots, m-1$, it becomes possible to approximate $\gamma(k)$ for $i = 0, 1, \cdots, m-1$.

To estimate the posterior density of θ, a nonparametric method for spectral density s, periodogram is utilized. The posterior density of θ can be estimated by the truncated form of the Fourier transform, known as the periodogram, which is defined as:

$$I_m(\tau) = \frac{1}{m} \left| \sum_{t=1}^{m} \varepsilon_t e^{-2\pi i t \tau} \right|^2 \qquad for \quad \tau \in [0, 1) \tag{3.6}$$

With the stationary assumption of ε, $I_m(\tau)$ asymptotically follows an exponential distribution with mean $s(\tau)$. Moreover, $I_m(\tau_0), I_m(\tau_1), I_m(\tau_2), \cdots, I_m(\tau_{\frac{m}{2}})$ are asymptotically independent for Fourier frequencies. [51] We may estimate the logarithm of a standard exponential random variable to estimate an exponential random variable. We can use a mixture of five Gaussian random variables to estimate standard exponential

random variables with known means and variances. Then, the relationship between the variables becomes:

$$\log I_m(\tau) = \log s(\tau) + \xi(\tau)$$

When assigning ξ for the mixture of five Gaussian random variables, then the distribution for π will be given as

$$\pi(\xi) \approx \sum_{l=1}^{5} p_l \phi_{v_l}(\xi - k_l)$$

where ϕ_{v_l} follows Gaussian distribution with mean zero and variance v_l^2 p_l, k_l, v_l for $l = 1, 2, \cdots, 5$ are known. According to [13], utilizing a Gaussian process before the logarithmic transformation of the spectral density captures the temporal dependence structure in the error process, enhancing the precision of the forecast by yielding a more accurate estimate of the model error. [3] Now, we elucidate the relationship between variables and determine how to estimate each, approximating the posterior distribution of θ. Subsequently, we can employ the MCMC method to ascertain the posterior distributions for these variables.

3.4.2 Posterior Distributions of Bayesian Estimators

Hyperparameters need to be established for the proposed MCMC approach, aligning with the methodology outlined in [3]. For the mean function ι and the covariance kernel κ, we assume $\iota \equiv 0$, $\kappa(\tau_x, \tau_y) = \frac{1}{\kappa_0^2} e^{-\rho||\tau_x - \tau_y||}$, where $||\cdot||$ denotes the Euclidean norm. For κ_0^2, we assume it follows **Gamma(a,b)** and ρ follows a non-informative prior.

We need to determine the posterior densities of unknown quantities from the data. The unknown variables in the model are β, θ, information on the compositions of the mixture in $\xi = (\xi_0, ..., \xi_{\frac{m}{2}})$, κ_0^2, and ρ. We introduce ψ_j for the label of the mixture component of ξ_j, which takes values in $\{1, ..., 5\}$. Introducing ψ_j simplifies the expression for posterior densities. The update steps for κ_0^2 and ρ_0 are skipped, because they are typical of conjugacy. Here is a description of how a Gibbs sampler works in sampling β, θ, and $\psi = (\psi_0, ..., \psi_m)$:

1. Update $\boldsymbol{\beta}$: Given $\boldsymbol{\theta}$, we approximate $\boldsymbol{\Gamma}_{n\times n}$ using equation 3.5 and denote it by $\widetilde{\boldsymbol{\Gamma}}_{n\times n}$. This will update $\boldsymbol{\beta}$. The conditional posterior density for $\boldsymbol{\beta}$ with a Gaussian prior, $\boldsymbol{\beta} \sim N(\boldsymbol{\beta}_0, \sigma_0^2 \boldsymbol{I_p})$, is given by:

$$\boldsymbol{\beta}|\cdots \sim N(\mu_\beta^*, \boldsymbol{\sigma}_*^2),$$

where $|\cdots$ means conditioning on all other variables are given.

$$\boldsymbol{\sigma}_*^2 = (\boldsymbol{X'}\widetilde{\boldsymbol{\Gamma}}_{n\times n}^{-1}\boldsymbol{X} + \sigma_0^{-2}\boldsymbol{I_p})^{-1},$$

$$\mu_\beta^* = \sigma_*^2(\boldsymbol{X'}\widetilde{\boldsymbol{\Gamma}}_{n\times n}^{-1}\boldsymbol{y} + \sigma_0^{-2}\boldsymbol{\beta_0}).$$

2. Update $\boldsymbol{\theta}$: Given $f(\boldsymbol{X_{t*n}}) = \boldsymbol{X\beta}$, we can compute $\varphi = (\varphi_0, ..., \varphi_{\frac{m}{2}})$, where $\varphi_j = \log(I_j(\omega_j))$ using equation 3.6 with $\varepsilon = y - \boldsymbol{X\beta}$. Then, we have the following conditional distribution for $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}|\cdots \sim N(\boldsymbol{\iota}^*, \boldsymbol{\Upsilon}^*),$$

where

$$\boldsymbol{\Upsilon}^* = (\boldsymbol{\Upsilon}^{-1} + \boldsymbol{V}_\psi^{-1})^{-1}, \quad \boldsymbol{\iota}^* = \boldsymbol{\Gamma}^*\boldsymbol{V}_\psi^{-1}(\boldsymbol{\varphi} - \boldsymbol{\kappa}_\psi - \boldsymbol{\iota}) + \boldsymbol{\iota},$$

with $\boldsymbol{\iota} = (\iota(\tau_0), ..., \iota(\tau_m))$, $\boldsymbol{\Upsilon} = (\kappa(\tau_x, \tau_y))_{x,y=0,...,\frac{m}{2}}$, $\boldsymbol{\kappa}_\psi = (\kappa_{\psi_0}, ..., \kappa_{\psi_m})'$, and $\boldsymbol{V}_\psi = \text{diag}\{v_{\psi_0}^2, ..., v_{\psi_m}^2\}$.

3. Update $\boldsymbol{\psi}$: Given $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, we can obtain the discrete posterior density for ψ_j such that, for $l = 1, ..., 5$:

$$P(\psi_j = l|\boldsymbol{\beta}, \boldsymbol{\theta}) = p_l\phi_{v_l}(\varphi_j - \theta(\omega_j) - \kappa_l).$$

The calculation of $\widetilde{\Gamma}_{n\times n}^{-1}$ can be obtained efficiently by modeling s.

$$\widetilde{\boldsymbol{\Gamma}}_{n\times n} = \boldsymbol{Q}_n\boldsymbol{\Lambda}_n\boldsymbol{Q}_n^*, \tag{3.7}$$

where $\boldsymbol{\Lambda}_n = \text{diag}\{\lambda(\omega_0), ..., \lambda(\omega_{n-1})\}$, $\boldsymbol{Q}_n$ is an $n\times n$ matrix with (u, v)-th entry being $q_{u,v} = \frac{1}{\sqrt{n}}e^{\frac{i(u-1)(v-1)2\pi}{n}}$, and $\boldsymbol{Q}_n^*$ is the complex conjugate matrix of $\boldsymbol{Q}_n$. Note that $\boldsymbol{\Lambda}_n$ is a real symmetric and positive definite matrix, even though $\boldsymbol{Q}_n$ and $\boldsymbol{Q}_n^*$ involve complex numbers. One can show the equality in 3.7 by using the expression of $\gamma_n(\cdot)$ discrete inverse Fourier transform. Because $\boldsymbol{Q}_n$ is a unitary matrix ($\boldsymbol{Q}_n\boldsymbol{Q}_n^* = \boldsymbol{I}$), we have $\widetilde{\boldsymbol{\Gamma}}_{n\times n}^{-1} = \boldsymbol{Q}_n\boldsymbol{\Lambda}_n^{-1}\boldsymbol{Q}_n^*$, where $\boldsymbol{\Lambda}_n^{-1} = \text{diag}\{\lambda^{-1}(\omega_0), ..., \lambda^{-1}(\omega_{n-1})\}$.

We can utilized the fitted model to forecast future values, particularly predicting a k step forward value of y given a value of $\boldsymbol{x}$ and the observed data. The prediction of $y_f = \boldsymbol{x}_f'\boldsymbol{\beta} + \varepsilon_f$ is given as $\boldsymbol{x}_f'\hat{\boldsymbol{\beta}} + E(\varepsilon_f|\boldsymbol{\varepsilon})$, where $\hat{\boldsymbol{\beta}}$ is the estimate obtained from the estimation procedure in the previous section. The conditional expectation of ε_f given $\boldsymbol{\varepsilon}$ is $E(\varepsilon_f|\boldsymbol{\varepsilon}) = \boldsymbol{h'}\boldsymbol{\Gamma}_n^{-1}\boldsymbol{\varepsilon}$, $h = Cov(\boldsymbol{\varepsilon}, \varepsilon_f) = (\gamma(k+n-1), \ldots, \gamma(k))'$, $E(\boldsymbol{\varepsilon}) = 0$ and $Var(\boldsymbol{\varepsilon}) = \boldsymbol{\Gamma}_n$. This prediction concept is integrated into the Gibbs sampling steps. In the *r-th* iteration, we obtain $\boldsymbol{\beta}^{(r)}$ and $\boldsymbol{\Gamma}_n^{(r)}$. From these, we derive $\boldsymbol{\varepsilon}^{(r)} = \boldsymbol{y} - \boldsymbol{x'}\boldsymbol{\beta}^{(r)}$. Then $y_f^{(r)}$ is computed as $y_f^{(r)} = \boldsymbol{x}_f'\boldsymbol{\beta}^{(r)} + \varepsilon_f^{(r)}$, where $\varepsilon_f^{(r)} = E(\varepsilon_f|\boldsymbol{\varepsilon}^{(r)}) = \boldsymbol{h}^{(r)'}\boldsymbol{\Gamma}_n^{-1^{(r)}}\boldsymbol{\varepsilon^{(r)}}$

To complete the forecasting step, we require $\boldsymbol{h}^{(r)}$, which comprises $\gamma(k+n-1), ..., \gamma(k)$ at the r-th iteration. $\gamma(n-1), ..., \gamma(0)$ are available from $\Gamma_n^{(r)}$ but we lack $\gamma(k+n-1), ..., \gamma(n)$. These quantities can be estimated as follows: we have $\lambda^{(r)}(\omega_0), ..., \lambda^{(r)}(\omega_{n-1})$, where $w_j = \frac{j}{n}$, representing Fourier frequencies with n. We interpolate these to obtain $\hat{\lambda}(\omega)$ for $\omega \in [0, 1)$, so that we obtain $\hat{\boldsymbol{\Lambda}}_f = \text{diag}\{\hat{\lambda}(\omega_0^*), ..., \hat{\lambda}(\omega_{f-1}^*)\}$, where $\omega_j^* = \frac{j}{f}$. Then, the estimate of $\boldsymbol{h}$, $\boldsymbol{h^{(r)}}$, is obtained from $\boldsymbol{\Gamma}_f = \boldsymbol{Q}_f \hat{\boldsymbol{\Lambda}}_f \boldsymbol{Q}_f^*$. The methodology outlined in this paper was applied to tackle the task of forecasting exchange rates. A notable demand in current macrofinance literature is developing a method or model to predict foreign exchange rates accurately. [52]

CHAPTER 4:
COMPARISON OF THE BAYESIAN METHOD WITH THE AUTOREGRESSIVE PROCESS

In this section, we will compare the Bayesian method described in the previous section with another time series approach using real data. To facilitate comparison, we will calculate prediction results for both methods.

4.1 Autoregressive Process

One of the fundamental models in time series analysis is the autoregressive (AR) process, utilized for analyzing and forecasting time series data. This model acknowledges the influence of previous values on the current value in the time series data. In the AR process, the current value is expressed as a linear combination of its past values, hence the term "autoregressive," indicating the dependence of the current value on its historical values. A crucial parameter in the AR process is its order, denoted by "p," which specifies the number of past values considered in the model. If x_t represents the time series data at time t, then the AR(p) process can be formulated as:

$$x_t = \sum_{i=1}^{p} \zeta_i x_{t-i} + \epsilon_t$$

where ζ_i are the parameters of the model and ϵ_t is the error term, which is assumed to be independent of x_{t-i}. If $p = 1$, then the model is simplified to:

$$x_t = \zeta_1 x_{t-1} + \epsilon_t, \tag{4.1}$$

which is the AR(1) process.

A fundamental assumption for the AR process is that it is weakly stationary (or simply stationary), making it possible to build a forecasting model for future values. Because time series data are random and not deterministic, predicting future values is extremely difficult or impossible if the random components

are not stationary. Therefore, a critical assumption is that the time series is stationary. For the weak stationarity, the time series is assumed to satisfy the following conditions:

(i) $E(x_t) = \mu < \infty$,

(ii) $Var(x_t) = E(x_t - \mu)^2 = \sigma^2 < \infty$,

(iii) and $Cov(x_t, x_{t+k}) = E(x_t - \mu)(x_{t+k} - \mu) = \sigma_k < \infty, \quad \forall k.$

In addition, it is assumed that the absolute value of coefficient ζ should be less than 1. Otherwise, it is no longer stationary. The particular case of a nonstationary model occurs when $\zeta = 1$: $x_t = x_{t-1} + \epsilon_t$, which is called a random walk. For the error term in a time series, one of the most popular models is the white noise process given the following conditions:

(i) $E(\epsilon_t) = 0$,

(ii) $E(\epsilon_t^2) = \gamma^2$,

(iii) and $E(\epsilon_t \epsilon_t) = 0, \quad \forall t \neq s.$

Order selection is crucial for AR model estimation. We need to include the relevant previous values to estimate the current value. To determine the order p, we use the autocorrelation function (ACF) or the partial autocorrelation function (PACF) to identify how the data relate to previous values. Additionally, it is essential to leverage domain knowledge and other statistical metrics, such as the Akaike information criterion (AIC), to select the appropriate p since the real p is often unknown.

This study will utilize the AR(1) model due to its computational simplicity. Additionally, real data have exhibited patterns consistent with AR(1), which further justifies its selection for comparison in the subsequent section. We risk overlooking discernible patterns in the data by opting for a higher-order p or a more complex model. Hence, we have opted for the simplest model for forecasting future values. It is important to note that AR models assume a linear relationship between past and current values, with the error term assumed to follow a normal distribution with constant variance. However, these models may not adequately capture complex nonlinear patterns or long-term dependencies within the data. To address this limitation, we will modify the assumption regarding error terms in this study.

4.2 Real Examples: Exchange Rate Data

Before introducing the neural network model, we aim to assess the effectiveness of the proposed methodology using actual exchange currency data. The currency exchange market is an ideal example of a time series data source characterized by a global market with rates fluctuating continuously. While the rate volatility in this market may not be as pronounced as that in the stock market, it provides valuable data for validating the performance of traditional time series models.

4.2.1 Forecasting

When forecasting future exchange rate values, we will utilize two distinct models: the forward premium regression of the exchange rate proposed by FAMA [53] and the AR model for both the Bayesian and AR processes. The FAMA model leverages data from the spot and 30-day forward exchange rates. The spot exchange rate represents the closing rate at time t, while the 30-day forward rate signifies the pre-agreed rate at time t to trade the currency at time $t+30$. The detailed format for the model is outlined as follows.

$$S_{t+k} - S_t = \beta_0 + \beta_1(F_t - S_t) + \epsilon_{t+1}$$

In this model, S_t represents the logarithm of the spot exchange rate at time t, while F_t represents the logarithm of the corresponding 30-day forward rate at time t. The prediction error at time t is denoted as ϵ_t. If we conceptualize this model linearly, $S_{t+k} - S_t$ serves as the dependent variable, whereas $F_t - S_t$ acts as the independent variable with coefficients β_1 and intercept β_0. Therefore, we will estimate β_0 and β_1 using the linear model in equation 3.1. It is worth noting that both β_0 and β_1 will vary upon the choice of different k, and hence, their estimators will depend on k. For the AR method, the model in equation 4.1 will be utilized, whereby,

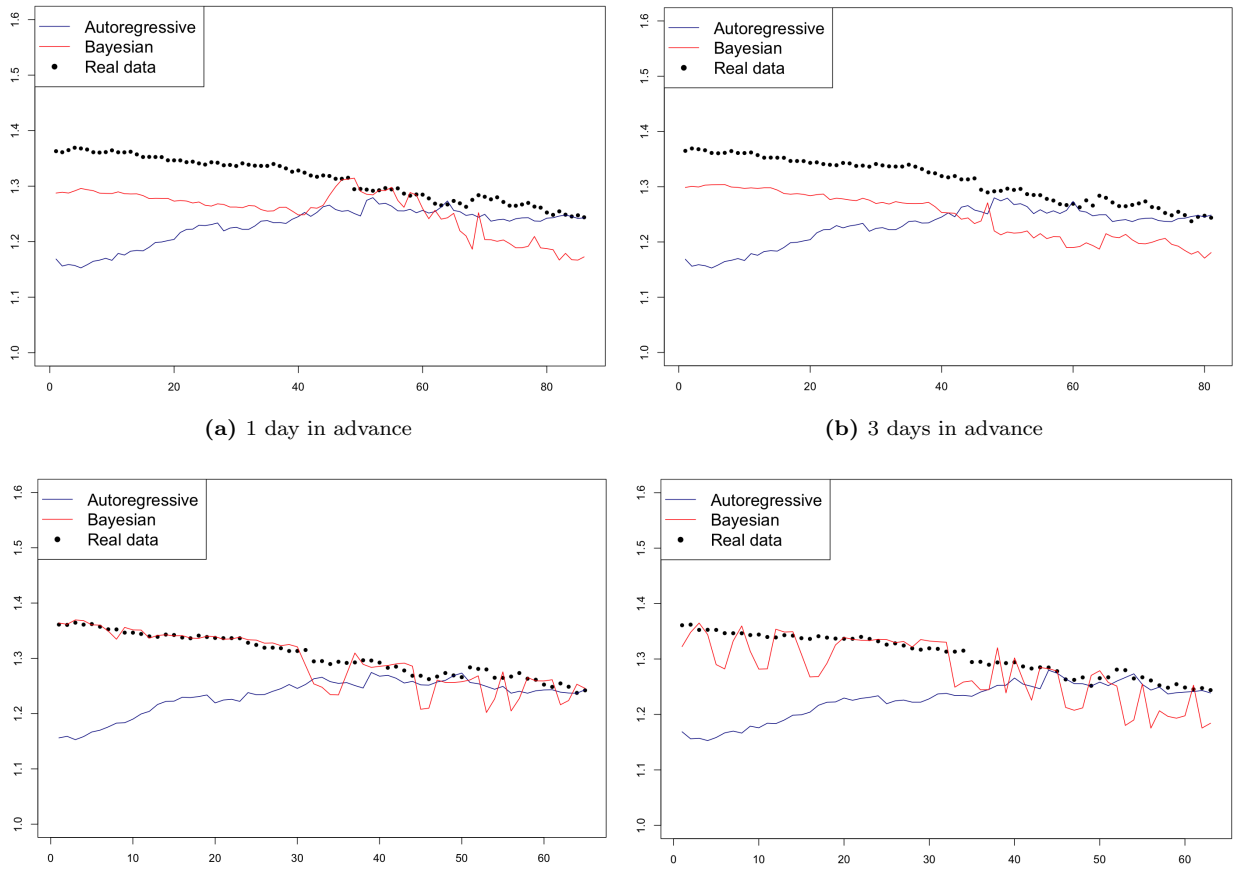
$$S_{t+k} = \beta_0 + \beta_1 S_t + \varepsilon_t,$$

and β_0 and β_1 will be estimated. Compared to the AR model, the FAMA model offers several advantages. First, it allows us to incorporate additional information into the model. Because the forward rate itself is already a prediction made by experts, its inclusion aids in rate forecasting. Additionally, by obtaining the difference between S_{t+k} and S_t, we can effectively eliminate any potential trends within the data.

We examined currency exchange rates for the US dollar against the Euro, Mexican Peso, and Indian Rupee. The analysis included the maximum possible period of currency data for each currency. The results of forecasting 1, 3, 6, and 12 days in advance are illustrated below and provide clear insight into the models' performance.

Four subplots were consolidated into a single figure, each corresponding to a distinct time horizon k in S_{t+k}: 1, 3, 6, and 12. The horizontal axis represents days, and the vertical axis represents the rate. The black dotted line in figures indicates the actual data. In contrast, the blue line represents the forecast generated by the AR model, and the red line denotes the prediction derived from the Bayesian method. Prediction involves forecasting S_{t+k}–S_t using the most accurate actual values up to time t. This approach anticipates the gap between the logarithm of the spot rate at time t and the rate at $t + k$, enabling the prediction of the logarithm of the spot rate for k days in advance. Upon analysis, a consistent trend was observed: the forecasts of currency exchange rates one day ($k = 1$) in advance were relatively close for both models. However, as the value of k increased, the forecasts diverged, and hence the forecasting performances for both models degrade. Another notable trend revealed through visualization is that the Bayesian model can detect changes in direction, whereas the AR model follows the trend.

The visualization below illustrates the predictions of the two models for the Euro, spanning from January 2014 to December 2023, the recent decade's data.



(a) 1 day in advance

(b) 3 days in advance

(c) 6 days in advance

(d) 12 days in advance

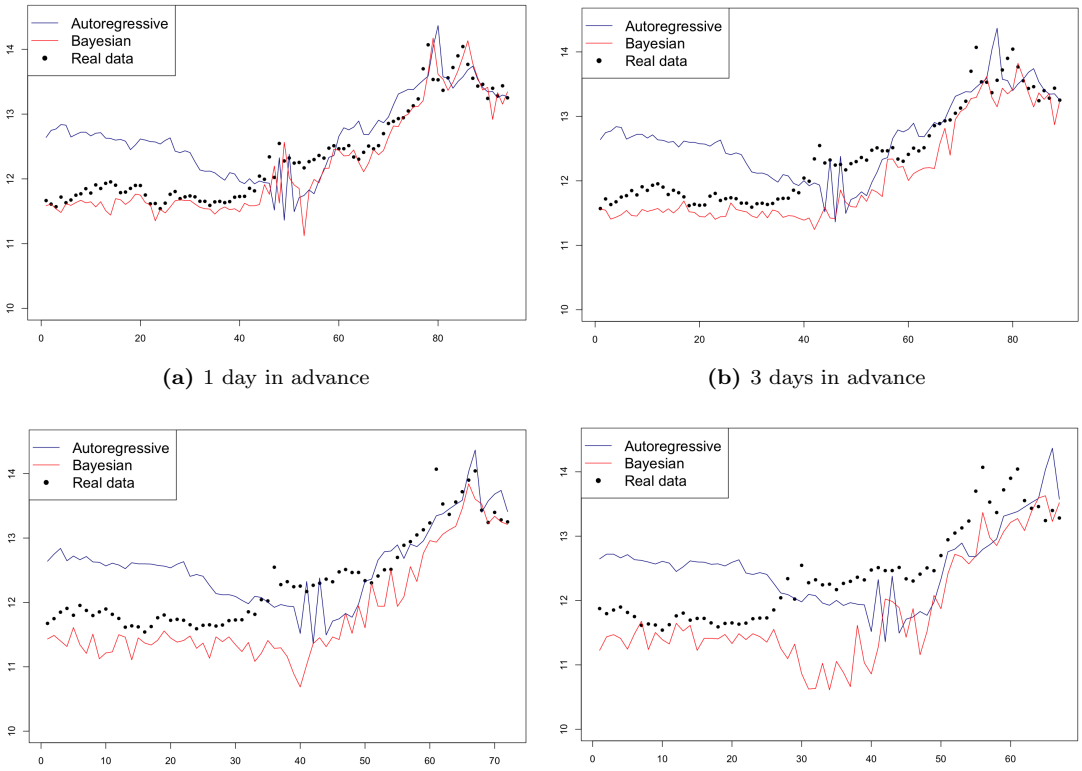
Figure 1. Euro prediction

Given the global significance of the US dollar and the Euro, their exchange rates exhibit more stability than other currencies depicted in the visualization. However, although generally stable, they are not consistently steady, leaving room for prediction. The Bayesian model consistently outperformed the AR model across at various time intervals ($k = 1$, 3, 6, and 12). Notably, the Bayesian model excels at capturing recent fluctuation patterns, enabling accurate prediction of future trends, particularly during periods of sharp deviations from previous patterns. The RMSE for each model at each time interval k is presented below, demonstrating the Bayesian method's superior performance over the AR model across all values of k.

Table 1. RMSE table of Euro

Interval	AR	Bayesian
1 day	0.0100	0.0041
3 days	0.0099	0.0043
6 days	0.0097	0.0006
12 days	0.0091	0.0017

Below is the visualization for the Mexican Peso. The format is similar to that of the Euro.



(a) 1 day in advance

(b) 3 days in advance

(c) 6 days in advance

(d) 12 days in advance

Figure 2. Mexican Peso prediction

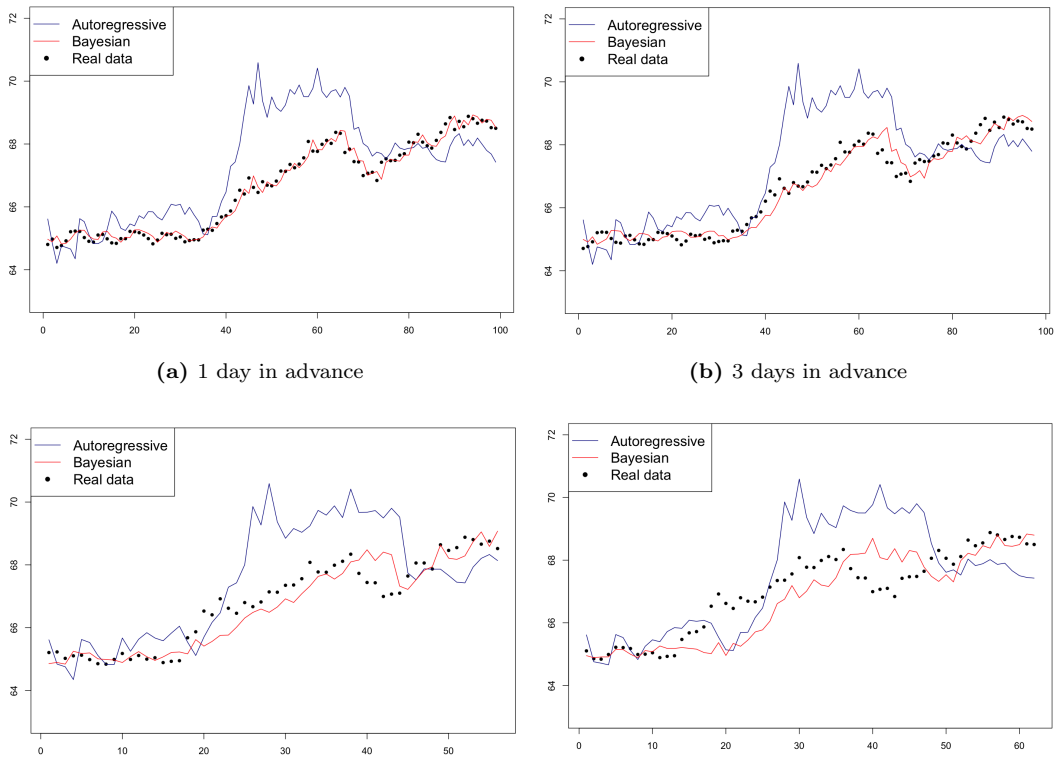
In contrast to the Euro, the Mexican Peso's exchange rate displayed substantial fluctuations. While the AR model struggled to capture the recent pattern, the Bayesian model excelled. The Bayesian method demonstrated strong performance for the 1-day and 3-day predictions but notably underestimated the difference for the 6-day and 12-day predictions.

Table 2. RMSE table of Mexican Peso

Interval	AR	Bayesian
1 day	0.3547	0.0743
3 days	0.3654	0.1709
6 days	0.3656	0.3827
12 days	0.4679	0.6037

The Bayesian method outperformed the AR model for k values of 1 and 3, but not for 6 or 12 days. During these periods, the rate fluctuated significantly, causing the Bayesian method to be overly sensitive to recent values when predicting rates, leading to deviations from the actual values.

Below is the prediction for the Indian Rupee. Data spans from September 2017 to December 2023.



(a) 1 day in advance

(b) 3 days in advance

(c) 6 days in advance

(d) 12 days in advance

Figure 3. Indian Rupee prediction

During the analysis period, the Indian Rupee fluctuated less than the Peso but more than the Euro. The Bayesian method performed superiorly in capturing the fluctuation pattern and showed minimal deviation from the actual values. The RMSE table below illustrates the effectiveness of the Bayesian method.

Table 3. RMSE table of Indian Rupee

Interval	AR	Bayesian
1 day	1.6144	0.0475
3 days	1.5400	0.1078
6 days	1.4948	0.2661
12 days	1.4671	0.5781

We conclude this section by comparing and summarizing the performances of both models. For the Euro and Indian Rupee, the Bayesian model outperformed the AR method. For the Euro, the AR model showed a better performance for larger k while the opposite was true for smaller k. The most prominent fact is that both Bayesian and AR reported relatively poor performance for forecasting the future value of the Euro. While there can be a number of reasons for the poor performance of the AR method, the breach of the assumption of (weak) stationarity of the Euro data is presumed to be a fundamental reason. On the contrary, the poor performance of Bayesian method for the case of Euro is not convincing. However, a simple remedy against the poor performance is to try different prior distributions for the parameters. To this end, non-informative prior distributions – e.g., a flat distribution – that are known to be more data-oriented can be suggested as a more fundamental solution. It should be admitted that lacking other prior distributions is one of the limitations of this study: it would, however, form future research.

CHAPTER 5:
COMPARISON OF THE BAYESIAN MODEL WITH RECURRENT OF NEURAL NETWORKS

5.1 RNN

Time series data possesses unique characteristics compared to other data types: data points are sequentially related and, therefore, not independent. Typical machine learning models rely on the assumption of independent and identically distributed input data, making them unsuitable for time series data. Instead, specialized models designed for sequential data are required. One of the most popular machine learning models for time series data is the RNN. RNN is a specialized variant of standard neural networks characterized by its recurrence property. This means that hidden states within a hidden layer influence themselves recursively. When configuring the parameters for the time series AR(1) model, the model ϕ is defined as follows:

$$y_t = \phi y_{t-1} + \varepsilon_t \tag{5.1}$$

The parameter ϕ iteratively impact the model as follows: If $\phi < 1$, then any y_t can be expressed as:

$$y_t = \sum_{j=0}^{\infty} \phi^j \varepsilon_{t-j} \tag{5.2}$$

Hence, it is imperative for ϕ to remain less than 1; otherwise, the equation will fail to converge.

Given the sequential nature of the data, the RNN model endeavors to retain information from previous steps by employing the same weight, denoted as W_{hh}, throughout the model. In contrast, standard neural networks require a continuous flow of data. For example, in a regression project to predict house prices using neural networks, the target value price would rely on explanatory variables such as house size and location. In this scenario, the model aims to estimate the price solely based on these explanatory variables. Let p_t represent the house price at time t; in this case, p_{t-1} is excluded from the equation for p, which is expressed as $p = X\beta + \epsilon$, while the time series model would incorporate p_{t-1}. Consequently, the regression variables are treated as independent and can be inputted into the neural networks and trained concurrently.

While considering a sequence model, specifically an RNN in our case, let x_t represent the input data and y_t denote the output data. Subsequently, we define a hidden state h_t and a pre-activation η_t for each time step t. Then, for the initial sequence data x_1, η_1 will be calculated as:

$$\eta_1 = \boldsymbol{W_{xh}} x_1 + b_h$$

where $\boldsymbol{W_{xh}}$ is the weight matrix between the input x_1 and the hidden state h_1, and b_h serves as the bias term for all hidden states h. Activation functions are employed in neural networks, including RNNs, to process the results of hidden states, such as the pre-activation η, based on the model's objective. Denoting the activation function for the hidden state as ψ_h, the expression for h_1 in an RNN would be:

$$h_1 = \psi_h(\eta_1) = \psi_h(\boldsymbol{W_{xh}} x_1 + b_h).$$

From the subsequent input x_2, we must incorporate the previous h_1 for the computatuin of h_2 as follows:

$$h_2 = \psi_h(\boldsymbol{W_{xh}} x_2 + \boldsymbol{W_{hh}} h_1 + b_h)$$

where $\boldsymbol{W_{hh}}$ represents the weight matrix corresponding to the recurrent edge, carrying data from h_{t-1} to h_t. This process can be generalized as

$$h_t = \psi_h(\boldsymbol{W_{xh}} x_t + \boldsymbol{W_{hh}} h_{t-1} + b_h).$$

The output y_t also requires an activation function of h_t to emerge from the hidden state. Denoting ψ_o as the output activation function and $\boldsymbol{W_{ho}}$ as the weight matrix between the hidden state and output state, the output y_t can be calculated as:

$$y_t = \psi_o(\boldsymbol{W_{ho}} h_t + b_o) = \psi_o(\boldsymbol{W_{ho}} * \psi_h(\boldsymbol{W_{xh}} x_t + \boldsymbol{W_{hh}} h_{t-1} + b_h) + b_o).$$

Figure 4 depicts the above procedure.

To find the weights W_{xh}, W_{hh}, W_{ho}, we utilize gradient descent, as in NNs. The loss function L is defined as the sum of squared errors:

$$L = \sum_{t=1}^{T} (y_t - \hat{y}_t)^2.$$

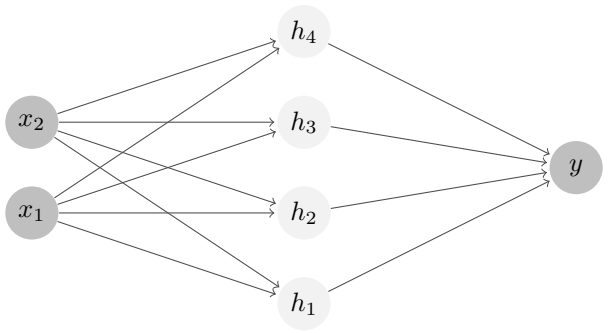


Figure 4. Basic structure of nueral network

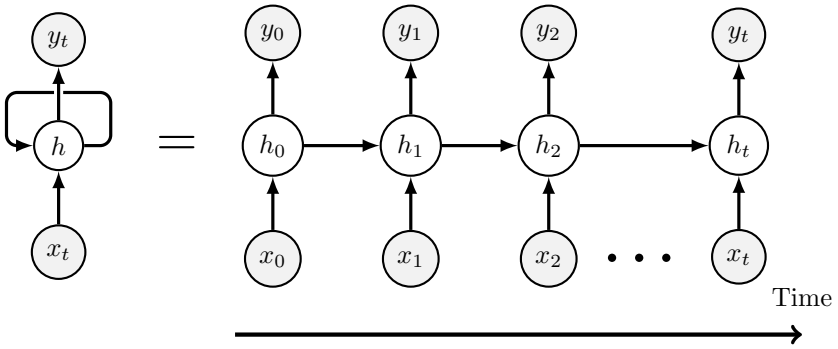


Figure 5. Structure of RNN

where $\hat{y}_t$ is the value predicted at time t. In time t, losses are dependent on previously hidden units at all previous time $1, ..., t-1$. Therefore, the gradient will be as follows:

$$\frac{\partial L_t}{\partial \boldsymbol{W_{hh}}} = \frac{\partial L_t}{\partial y_t} * \frac{y_t}{h_t} * (\sum_{k=1}^{t} \frac{\partial h_t}{\partial h_k} * \frac{\partial h_k}{\partial \boldsymbol{W_{hh}}})$$

where $\frac{\partial h_t}{\partial h_k}$ can be computed as a multiplication of adjacent time steps:

$$\frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}}$$

CHAPTER 6:
DATA ANALYSIS

6.1 Data Visualization

Understanding time series data comprehensively can be challenging, especially over long durations. Data visualization, which presents data graphically, is invaluable for gaining insights into the data. Therefore, this section utilizes data visualization to analyze various US stocks, focusing mainly on those related to Nvidia. Selecting stocks closely associated with Nvidia is crucial for analysis, as relevant data are pivotal in building effective time series models. Gathering a diverse range of related data enhances the model-building process, even if the effectiveness of some points may be uncertain. Hence, it is essential to include relevant stock data to ensure optimal model performance.

Moreover, incorporating additional data typically improves machine learning models without adverse effects, and collecting surrounding or related data can prove beneficial. For our analysis, we gathered and utilized various types of stock data, including AMD, Apple, Google, Microsoft, and Nvidia. The forthcoming visualizations illustrate stock data in candlestick chart format from September 1, 2022, to July 31, 2023. The x-axis denotes time, while the y-axis represents prices in the candlestick plot. Each business day is represented by a box plot, encapsulating five crucial pieces of information: open, close, high, low, and the day's price trend.

The color of each box plot indicates the price movement on the given day. In a bearish market, where prices decline, the color is red. Conversely, the color is green in a bullish market with rising prices. Thus, the color provides information on the price trend for the day. In the stock market, instead of whiskers, we use the term "wick" and refer to the chart as a candlestick chart.

Another critical aspect is the box's boundary. A red box indicates that the closing price is lower than the market opening price. In this scenario, the upper box boundary represents the opening price, while the lower boundary represents the closing price. Conversely, in a green box, the upper boundary represents the closing price, and the lower boundary represents the opening price. Therefore, the box boundary contains information about the opening and closing prices.

The whiskers represent the day's highest and lowest prices. The top of the upper wick denotes the highest price of the day, while the end of the lower wick indicates the lowest price.

Figure 6 illustrates the essential elements of the candlestick chart, offering clarity on its interpretation. A vertically elongated box indicates a substantial gap between the open and closed prices, indicating a substantial variation for that day. Likewise, the candle wick in the plot indicates a substantial deviation between the high and low prices from the open-close price, indicating high volatility for that day.

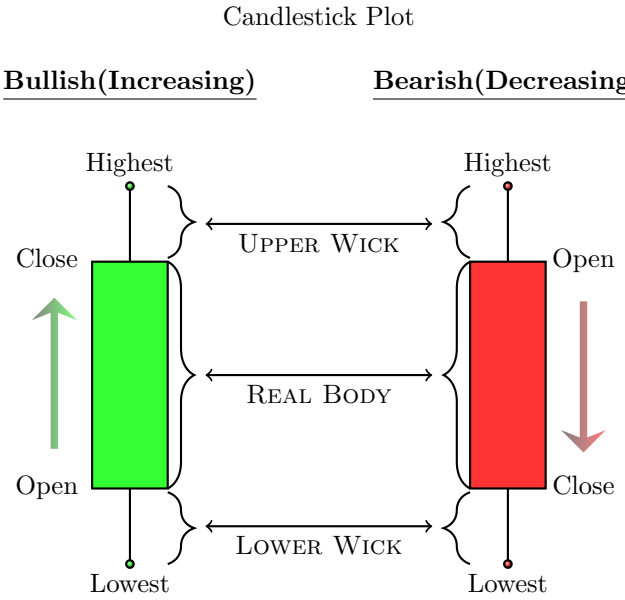


Figure 6. Candlestick chart explanation

The main focus of our study was the surge in Nvidia stock, making it imperative to examine Nvidia's candlestick chart, as depicted in Figure 7.

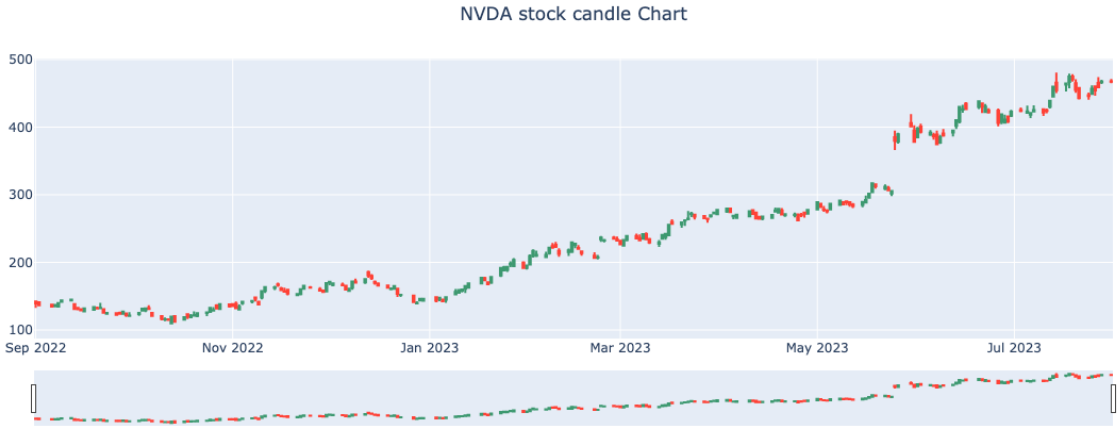


Figure 7. Nvidia candlestick chart

With the introduction of ChatGPT by OpenAI on November 30, 2022, there has been a surge in interest in LLMs, elevating them to a central focus in AI. This has led to substantial investments and exerted a global influence. AI has been a prominent field for over a decade, beginning with the emergence of AlphaGo, which ignited research into computation techniques for constructing AI models. These models are essentially programmed algorithms, and as they become more sophisticated, they demand extensive computational resources. Researchers have concentrated on methods to accelerate computation speed to enhance AI performance. Parallelization has emerged as a crucial approach for expediting computations compared to traditional methods. While computations were traditionally handled primarily by central processing units (CPUs), researchers found that graphics processing units (GPUs) outperform CPUs when parallelized computations. Initially designed for graphical processing, GPUs have become indispensable in AI research and development. The introduction of ChatGPT has sparked interest in LLMs, leading to widespread investment and global research by corporations and researchers alike. Constructing an LLM is computationally intensive and is expected to drive increased demand for GPUs. Consequently, Nvidia's leading GPU manufacturer is anticipated to experience substantial revenue growth, attracting investor interest in its stock.

The primary impetus for our study stemmed from the extraordinary surge in Nvidia stock, prompting a natural inclination to scrutinize Nvidia's candlestick chart. A notable observation is that it expanded three-fold in just 231 days. The opening price on November 30, 2022, stood at $156.97; on July 19, 2023, it had soared to $470.77.

Figure 8 illustrates the candlestick chart of the NASDAQ Composite Index, identified by the ticker symbols IXIC or COMP.

Figure 8. NASDAQ index candlestick chart

The index was launched on February 5, 1971, with an initial value of 100. [54]. Comprising over 3500 stocks listed on the NASDAQ stock exchange, it functions as a weighted market capitalization index. [55] Although variations exist based on stock types (ordinary and preferred stock), the fundamental concept remains consistent. For instance, if today's COMP is 15,000, it implies that the total market capitalization of NASDAQ is 150 times larger than it was on February 5, 1971.

The COMP index is a pivotal indicator for comprehending the US stock market. This significance has been magnified with the ascent of major tech conglomerates like Apple, Microsoft, Google, Amazon, Nvidia, Meta, and Tesla. These entities experienced notable surges during the COVID-19 pandemic and the AI era. The pandemic-induced transition to remote work accentuated the importance of virtual systems, with these companies playing instrumental roles in facilitating remote operations. Furthermore, the normalization of remote work and virtual meetings post-COVID has cemented these companies' positions as trailblazers in the new virtual landscape. Moreover, AI has emerged as a central focal point, offering substantial advantages to major tech players. These companies are poised to maintain substantial advantages in the AI age as pioneering contributors to AI development and implementation.

Figure 9 illustrates the candlestick chart of the Dow Jones index, with the Bloomberg ticker being INDU.

Figure 9. Dow Jones industrial index candlestick chart

INDU represents a US stock market index comprising 30 blue-chip companies. As one of the oldest and most extensively monitored indexes, it is price-based and was inaugurated on May 26, 1896, with a base value 40.94. [56]

Figure 10 illustrates the candlestick chart of the S&P 500 index, identified by the ticker symbol SPX.

Figure 10. S&P 500 index candlestick chart

"S&P" stands for Standard & Poor's, a financial analysis company, and "500" includes 500 companies. Consequently, the S&P 500 functions as an index of 500 large public companies on the US stock market meticulously selected by S&P. Operating as a capitalization-weighted average, the S&P 500 was launched on March 4, 1957. It encompassed stocks from the NYSE and NASDAQ, utilizing a base of 10 from 1941 to 1943. Widely acknowledged as the premier indicator of large-cap U.S. equities, the S&P 500 comprises 500 leading companies, representing approximately 80% of the available market capitalization. [57]

Figure 11 illustrates Microsoft's candlestick chart.

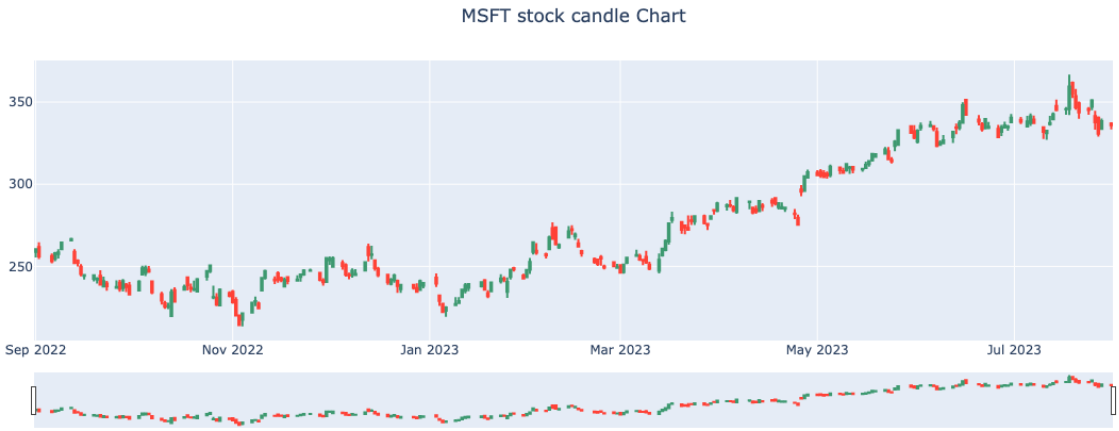


Figure 11. Microsoft candlestick chart

Microsoft currently has the second-largest market capitalization globally. Strategically, Microsoft has invested in OpenAI, acquiring a substantial 49% ownership stake. This effectively positions Microsoft as the

principal owner of OpenAI. As OpenAI is not publicly traded, individuals seeking to invest in It indirectly do so by purchasing Microsoft stock. This approach has proven remarkably successful in just a few months. Below is the candlestick chart of AMD stock data.

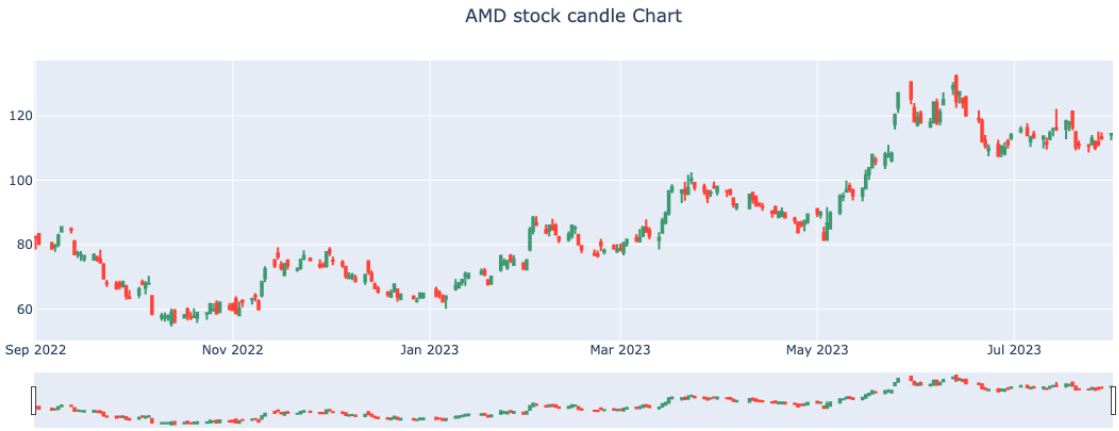


Figure 12. AMD candlestick chart

Given that one of the focal points of this article is understanding NVIDIA stock, comprehending related stocks is paramount. NVIDIA is a dominant GPU market force and is AMD's primary competitor. AMD, or Advanced Micro Devices, is a multinational semiconductor company renowned for producing computer processors and GPUs.

AMD holds a substantial position as one of NVIDIA's primary rivals in the GPU manufacturing sector. Together with Nvidia, AMD commands a considerable presence in this industry. AMD is prominently listed on the NASDAQ 100 and S&P 100 indices, making it a crucial subject for analysis in understanding NVIDIA. Following the release of ChatGPT in November 2022, AMD's stock price exhibited an upward trajectory, continuing this trend until the end of July 2023. The advent of ChatGPT sparked widespread interest in LLMs, prompting extensive investment and research endeavors by researchers and corporations worldwide. Given that LLMs necessitate substantial computational resources, GPU manufacturers are poised to reap substantial benefits from this trend. AMD's stock price reflects these expectations, underscoring its position as one of the leading GPU manufacturers.

Figure 13 illustrates Apple's candlestick chart.

Figure 13. Apple candlestick chart

Apple is currently the company with the highest total market capitalization globally. Throughout November 2022, the price showed lateral movement, but from January 2023 onwards, there has been a consistent upward trend.

Figure 14 displays Google's candlestick chart.

Figure 14. Google candlestick chart

Google did not directly benefit from ChatGPT's emergence. Instead, it faced competition from ChatGPT in the search engine market. While ChatGPT is not a full-fledged search engine, its ability to provide quick text responses has made it a favored choice for some users over traditional search engines like Google. As a

result, Google's stock price did not see the same increase as other related stocks; instead, it experienced a decline and remained relatively stable for a period.

Table 4. Descriptive statistics of stocks

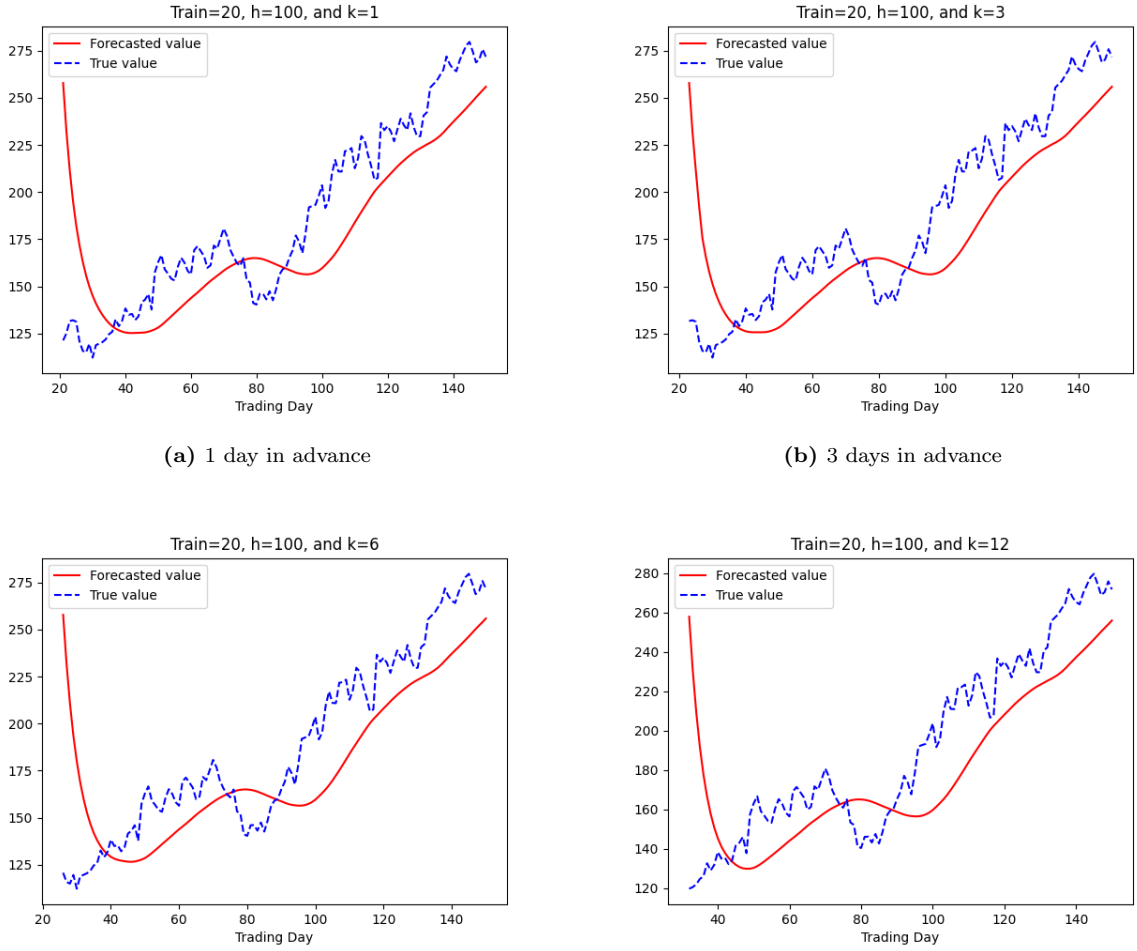
Ticker	avg_{price}	Std_{price}	avg_{return}	STd_{return}	avg_{volume}	Std_{volume}
NVDA	211.34	53.60	0.239%	0.03657	48,041,451	17,421,533
AMD	85.53	19.45	0.201%	0.03349	68,853,260	22,572,211
AAPL	157.97	18.22	0.113%	0.01838	70,896,956	24,079,656
GOOG	104.53	12.05	0.108%	0.00108	27,873,067	11,255,415
MSFT	273.69	39.06	0.133%	0.0204	30,044,626	11,057,702
SPX	4045.95	229.57	0.071%	0.0118	-	-
SOX	2951.44	426.98	0.195%	0.0224	-	-
COMP	11854.50	1069.18	0.098%	0.0150	-	-
INDU	33032.89	1413.17	0.056%	0.0100	-	-

6.1.1 Real Predictions

In this section, we train an RNN model using the time series data of the stocks analyzed in the previous section. We utilize the trained model to forecast the future values of these stocks and compare the predicted values with the real values. During the training of the RNN model, we experimented with various hyper-parameters, such as the number of hidden layers, and empirically selected the optimal ones that yield the closest predictions of the selected stocks.

RNN demonstrates remarkable effectiveness in capturing the intricate dynamics of time series models. One of its notable strengths lies in its ability to handle non-stationary processes, making it suitable for data exhibiting varying trends, cycles, or seasonality. The RNN adeptly replicates the overarching patterns observed in stock data movements by fine-tuning hyperparameters. However, our analysis has uncovered occasional discrepancies within the model, particularly noticeable when the market reacts unexpectedly to exceptional circumstances. In scenarios where stock movements are less pronounced than such outliers, the RNN consistently offers accurate predictions for the future of the time series. The following visualization illustrates the predictions made by the RNN model alongside the actual stock price of Nvidia. The predictions were generated to forecast the stock price using current data and anticipate the stock's future price at intervals of 1, 3, 6, and 12 days in advance—the data span from September 2022 to July 2023. The hyperparameters

used for training were set at 20, with a hidden state size of 100, representing the optimal combination of hyperparameters.



(a) 1 day in advance

(b) 3 days in advance

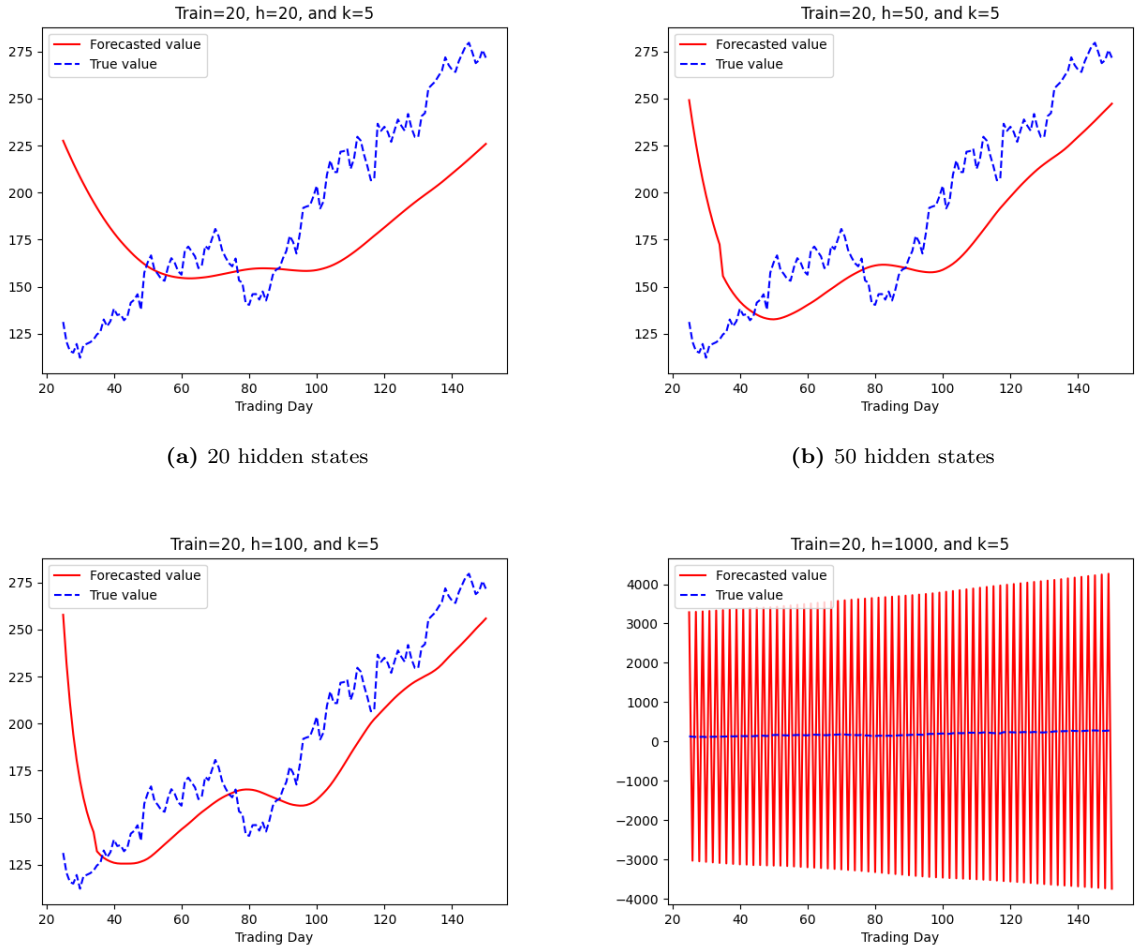
(c) 6 days in advance

(d) 12 days in advance

Figure 15. Predicted Nvidia stock prices using RNN

As depicted in the figure, the actual price of the stock, represented by the blue line, exhibited a notable increase from an initial $100 to surpass $400 within a few months. Conversely, the red line, indicating the RNN model prediction, initially started at 0. However, as more data became available, it rapidly captured the genuine trend of the stock price. Over recent years, Nvidia's stock has encountered substantial surges multiple times, rendering it one of the most volatile stocks on the NASDAQ 100. Although the model did not perfectly anticipate surges and drops on specific days, it generally reflected the most recent changes in predicting future values. However, as previously stated, this outcome was achieved using the best hyperparameters.

The figure below illustrates the critical role of hyperparameter tuning in the performance of RNN. While it may seem intuitive that a more complex model with higher hyperparameters would yield better results, this is not always the case. As depicted, the performance increases from 20 to 100 hidden states, but with 1000 hidden states, the model oscillates and fails to make accurate predictions. This underscores the challenge of hyperparameter selection in RNN, necessitating careful tuning akin to ad hoc methods like bandwidth selection in KDE.



(a) 20 hidden states

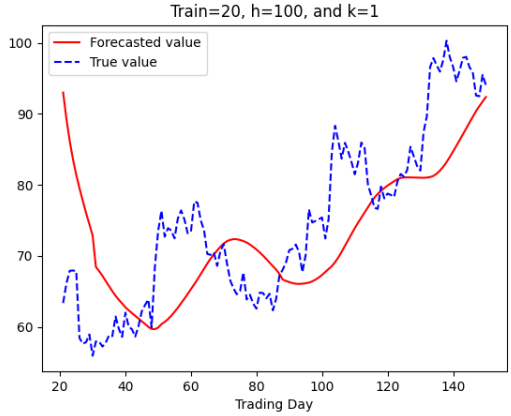
(b) 50 hidden states

(c) 100 hidden states

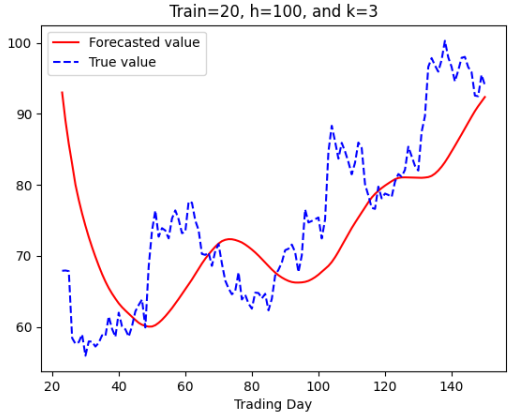
(d) 100 hidden states

Figure 16. Predicted Nvidia stock prices on the number of hidden states in RNN

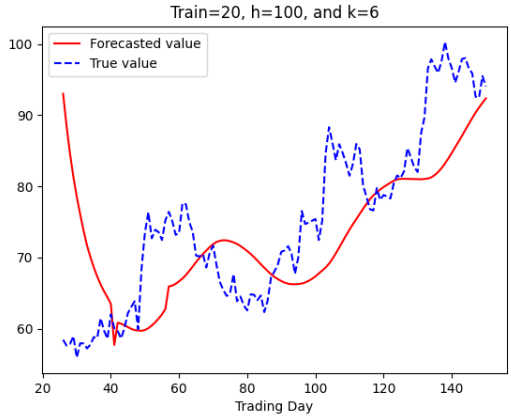
The figure below illustrates the RNN prediction of AMD's stock price using the same set of data dates.



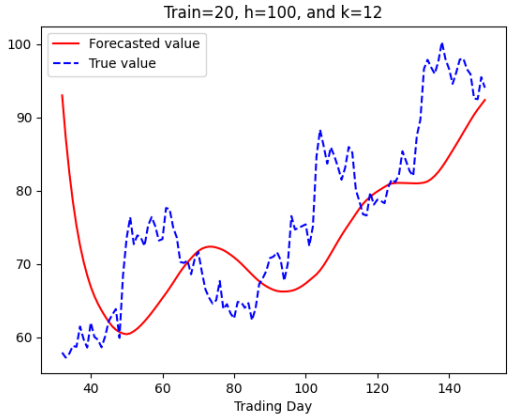
(a) 1 day in advance



(b) 3 days in advance



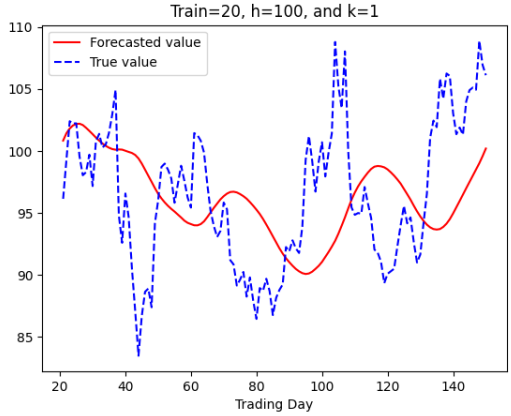
(c) 6 days in advance



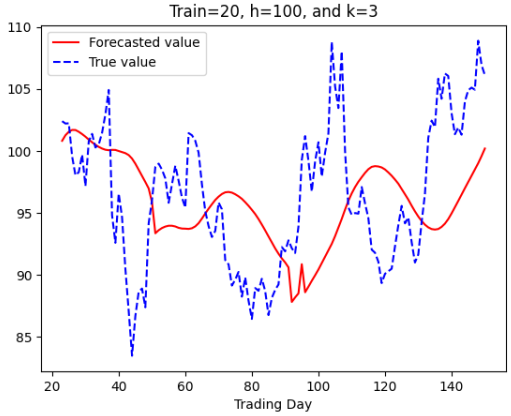
(d) 12 days in advance

Figure 17. Predicted AMD stock prices using RNN

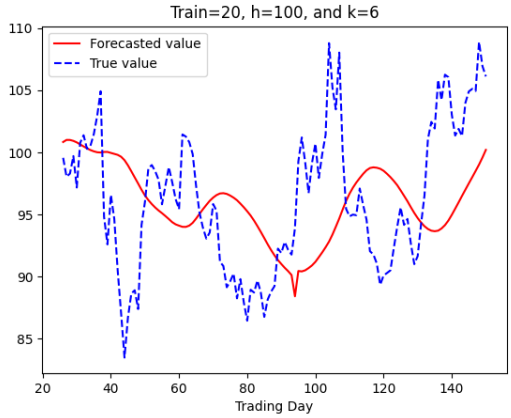
Like Nvidia, AMD's stock price has experienced substantial fluctuations, spiking several times and doubling within a few months. Consequently, AMD's time series data are far from steady and characterized by considerable volatility. After digesting the data for a few days, the RNN forecast started at 0 but quickly caught up with the actual value. Figure 18 shows Google's stock price.



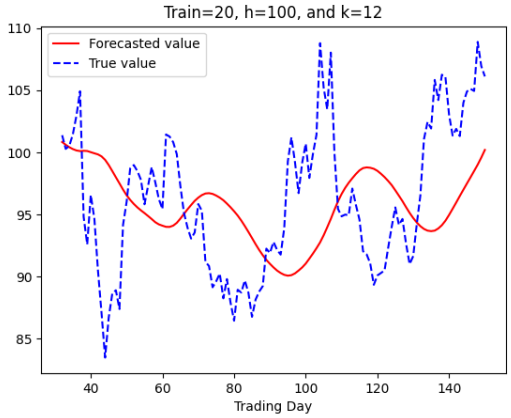
(a) 1 day in advance



(b) 3 days in advance



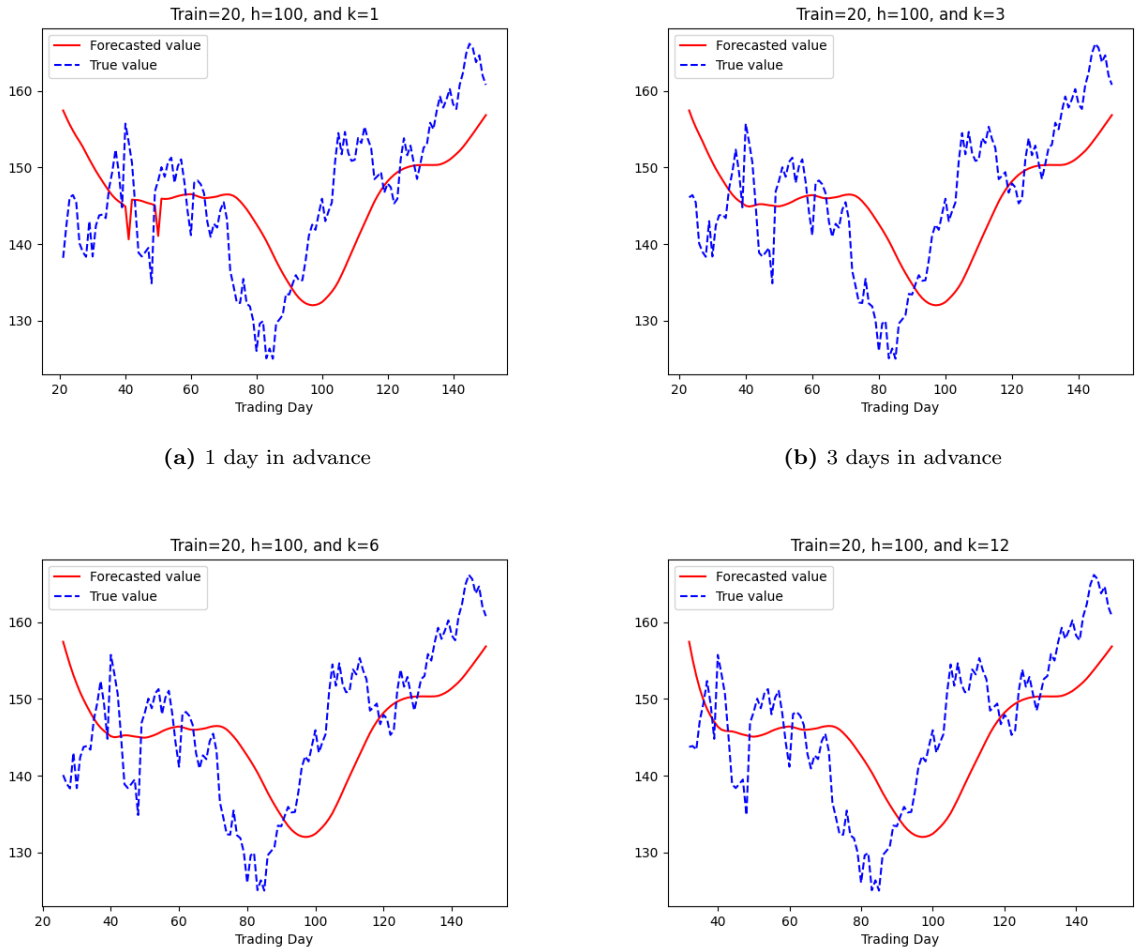
(c) 6 days in advance



(d) 12 days in advance

Figure 18. Predicted Google stock prices using RNN

Google, with two stock tickers, is represented by GOOG in this chart. While primarily known for its search engine, Google offers a diverse range of services in the tech industry. Concerns arose following the introduction of ChatGPT that Google might lose market share in the search engine industry due to ChatGPT's capabilities. This briefly impacted Google's stock price, but it swiftly rebounded. Despite ChatGPT potentially diverting some search traffic, Google remains the most popular search engine. Additionally, Google acknowledges the significance of AI and launched Bard four months after ChatGPT's debut. Many anticipate that major technology companies will play a pivotal role in the AI era, ensuring Google's stock price remains robust. However, while RNN effectively captured price patterns, it struggled to account for sudden spikes. The figure below illustrates Apple's stock price.



(a) 1 day in advance

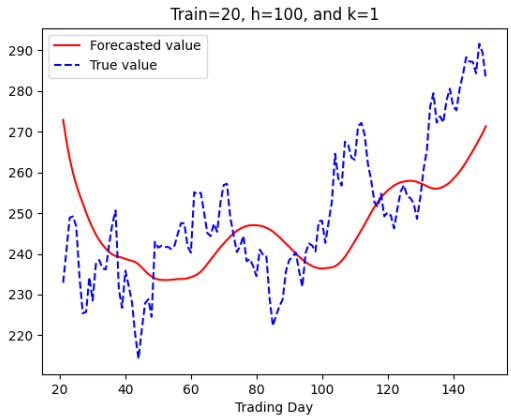
(b) 3 days in advance

(c) 6 days in advance

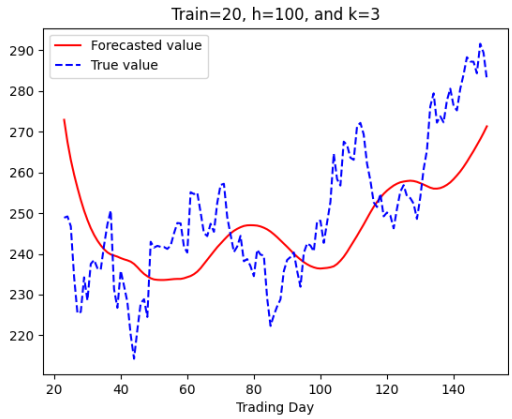
(d) 12 days in advance

Figure 19. Predicted Apple stock prices using RNN

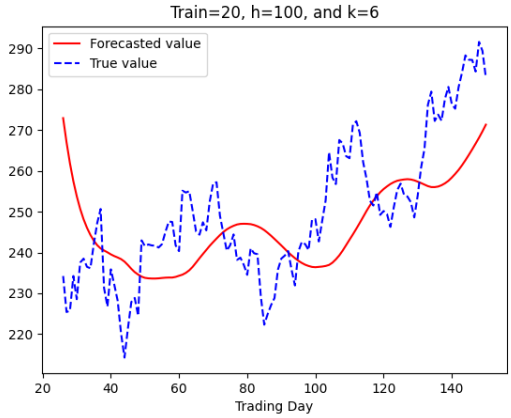
Apple, boasting the largest market capitalization, offers an AI service named Siri, primarily functioning as a digital assistant for mobile devices rather than as an LLM akin to ChatGPT. Unlike Nvidia or AMD, Apple has not demonstrated a significant response to LLM technology since the emergence of ChatGPT. This has raised doubts about Apple's AI capabilities, which is evident in its stock price. This has raised doubts about Apple's AI capabilities, which is evident in its stock price. Apple's stock price surge is more closely linked to its product sales than LLM developments. Nonetheless, as one of the premier tech companies globally, optimism surrounds Apple's potential unveiling of innovative products, thus preventing a significant decline in its stock price. Below is the depiction of Microsoft's stock price.



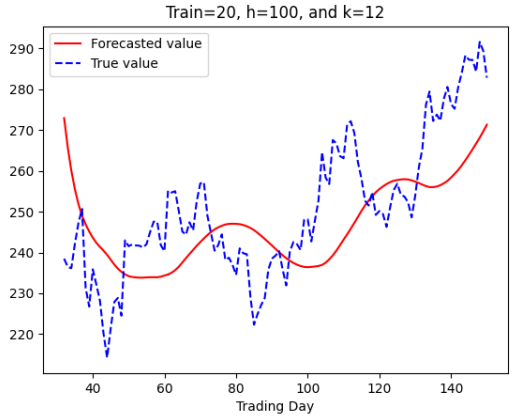
(a) 1 day in advance



(b) 3 days in advance



(c) 6 days in advance

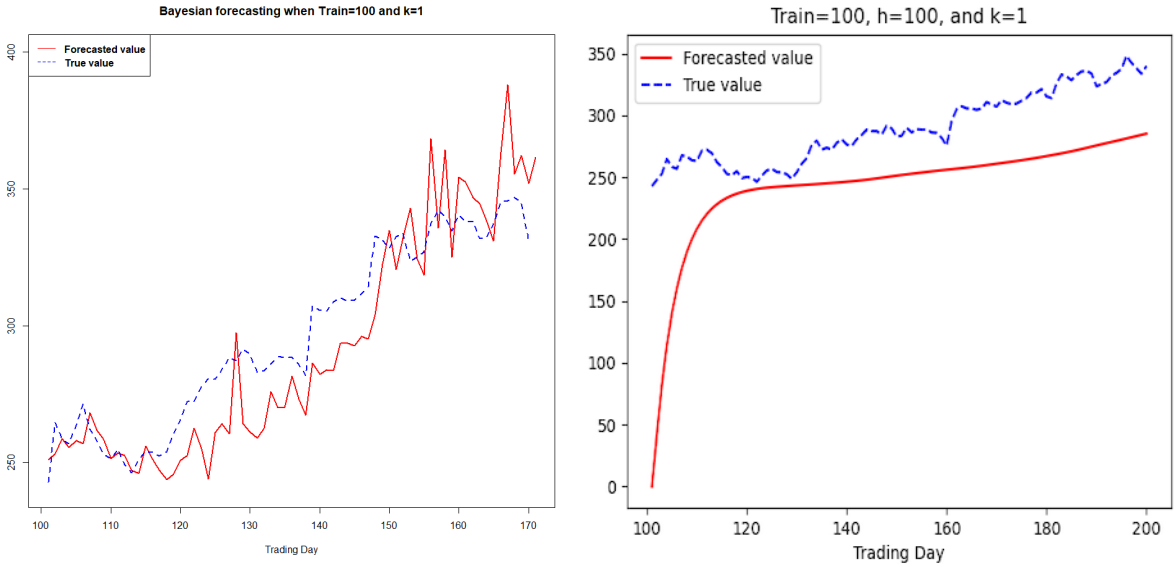


(d) 12 days in advance

Figure 20. Predicted Microsoft stock prices using RNN

Microsoft recently invested over $11 billion in OpenAI, establishing a robust partnership between the two entities. While the specific terms of their agreement remain undisclosed, their collaboration is evident. Following the successful launch of chatGPT by OpenAI, Microsoft has announced its intention to integrate the platform into its web browser, Bing. Despite Bing's relatively modest market share in the search engine industry, this integration is poised to drive its growth. Microsoft's substantial investment in OpenAI also indicates potential benefits from the latter's success. As a result, Microsoft's stock price has surged by more than 40% in recent months. Given Microsoft's prominence, substantial fluctuations in its stock price typically occur only in response to notable events, highlighting the transformative impact of ChatGPT's innovation.

We conclude this section by comparing RNN with the Bayesian method. For the comparison purpose, both methods will use MS stock data for forecasts. The Bayesian methos will use the NASDAQ index as an independent variable while RNN doesn't require any. Figure 21 depicts the Microsoft stock price and forecasts by the Bayesian method and RNN. For the fair comparison, both used the same 100 lagged data for training.



(a) Bayesian method: utilizing NASDAQ index as $\boldsymbol{X}$ **(b)** RNN: model trained with Microsoft 100 previous days

Figure 21. Microsoft stock price forecast

A quick glance reveals that the Bayesian method outperformed RNN. In the previous analysis, RNN suffered from tuning hyperparameters (especially, h the number of hidden states). When $h = 100$ – RNN with which showed the best performance – was tried, RNN showed a huge discrepancy and tried to catch up the true values as shown in Figures 15–20. However, RNN tended to show worse performance as the training data is getting large, and it displayed only a increasing, smooth trend without showing any fluctuation while the Bayesian method displayed a constant up and down pattern together with a increasing trend, which closely accords with true values. Consequently, the figure serves to demonstrate effectiveness of the Bayesian method in capturing and tracking the movement of the real data. The inherent structural variances between RNN and the Bayesian method lead to differences in input data, thereby complicating direct performance comparisons. Nevertheless, the graph effectively illustrates the capability of the Bayesian method.

CHAPTER 7:
CONCLUSION

Bayesian statistics offer several advantages over frequentist statistics. One significant advantage is the Bayesian method's ability to update its posterior distribution as new samples are obtained. In contrast, frequentist statistics require all samples to be collected simultaneously. This fundamental characteristic allows the Bayesian approach to demonstrate a remarkable ability to quickly adjust its predictions based on recent trends, a capability lacking in the AR model. The comparison of prediction figures from the two models clearly illustrates the superior performance of the Bayesian method over the AR model. While the AR model may not be the most suitable method with the given dataset, it is commonly used due to its constant-variance assumption. However, the results demonstrate how this assumption can lead to different results, contradicting the Bayesian approach. Additionally, RNNs, with their multiple hidden states to capture relations, typically perform well in prediction. However, the model's performance relies heavily on selecting hyperparameters, presenting a challenge for RNNs. Moreover, the parameterization of the RNN model remains a black box, so we do not have insight into its internal workings. For specific application cases, inference is crucial rather than simply obtaining the prediction result.

Through this study, we have confirmed that the Bayesian method adeptly predicts time series data and estimates the coefficients of each explanatory variable, making it a valuable tool for inferential studies. However, the study was not designed to directly compare the Bayesian method's and RNN's performance due to differences in their model structures. The Bayesian method relies on a linear model base, while the RNN utilizes an AR method-based self-dependent algorithm.

Future research should delve deeper into comparing Bayesian methods with machine learning techniques to enhance Bayesian models and align them with machine learning methods' predictive capabilities. This endeavor can potentially enhance comprehension and application of Bayesian approaches across diverse domains.

REFERENCES

[1] NASDAQ. "Our story." (2023), [Online]. Available: https://www.nasdaq.com/about#story. Jan 23, 2024.

[2] K. H. Kim and T. Kim, "Capital asset pricing model: A time-varying volatility approach," *Journal of Empirical Finance*, vol. 37, pp. 268–281, 2016.

[3] T. Dey, K. H. Kim, and C. Y. Lim, "Bayesian time series regression with nonparametric modeling of autocorrelation," *Computational Statistics*, vol. 33, pp. 1715–1731, 2018.

[4] M. Hollander, D. A. Wolfe, and E. Chicken, *Nonparametric statistical methods*. John Wiley & Sons, 2013.

[5] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation," *Econometrica: Journal of the econometric society*, pp. 987–1007, 1982.

[6] Y. B. Jun, C. Y. Lim, and K. H. Kim, "Bayesian estimation of the autocovariance of a model error in time series," *arXiv preprint arXiv:2210.07457*, 2022.

[7] I. M. Johnstone and B. W. Silverman, "Wavelet threshold estimators for data with correlated noise," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 59, no. 2, pp. 319–351, 1997.

[8] P. M. Robinson, "Large-sample inference for nonparametric regression with dependent errors," *The Annals of Statistics*, vol. 25, no. 5, pp. 2054–2083, 1997.

[9] C. Martins-Filho and F. Yao, "Nonparametric regression estimation with general parametric error covariance," *Journal of Multivariate Analysis*, vol. 100, no. 3, pp. 309–333, 2009.

[10] L. Su and A. Ullah, "More efficient estimation in nonparametric regression with nonparametric autocorrelated errors," *Econometric Theory*, vol. 22, no. 1, pp. 98–126, 2006.

[11] R. H. Shumway and D. S. Stoffer, *Time Series Analysis and Its Applications: With R Examples*, 4th ed. Springer, 2017.

[12] P. Whittle, "On stationary processes in the plane," *Biometrika*, pp. 434–449, 1954.

[13] C. K. Carter and R. Kohn, "Semiparametric bayesian inference for time series with mixed spectra," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 59, no. 1, pp. 255–268, 1997.

[14] L. J. Slater, L. Arnal, M.-A. Boucher, *et al.*, "Hybrid forecasting: Blending climate predictions with ai models," *Hydrology and earth system sciences*, vol. 27, no. 9, pp. 1865–1889, 2023.

[15] O. Kisi, J. Shiri, and B. Nikoofar, "Forecasting daily lake levels using artificial intelligence approaches," *Computers & Geosciences*, vol. 41, pp. 169–180, 2012.

[16] W.-C. Wang, K.-W. Chau, C.-T. Cheng, and L. Qiu, "A comparison of performance of several artificial intelligence methods for forecasting monthly discharge time series," *Journal of hydrology*, vol. 374, no. 3-4, pp. 294–306, 2009.

[17] J. Liu, S. Wang, N. Wei, X. Chen, H. Xie, and J. Wang, "Natural gas consumption forecasting: A discussion on forecasting history and future challenges," *Journal of Natural Gas Science and Engineering*, vol. 90, p. 103 930, 2021.

[18] A. Masood and K. Ahmad, "A review on emerging artificial intelligence (ai) techniques for air pollution forecasting: Fundamentals, application and performance," *Journal of Cleaner Production*, vol. 322, p. 129 072, 2021.

[19] Z. M. Yaseen, A. El-Shafie, O. Jaafar, H. A. Afan, and K. N. Sayl, "Artificial intelligence based models for stream-flow forecasting: 2000–2015," *Journal of Hydrology*, vol. 530, pp. 829–844, 2015.

[20] M. H. Lipu, M. S. Miah, M. Hannan, *et al.*, "Artificial intelligence based hybrid forecasting approaches for wind power generation: Progress, challenges and prospects," *IEEE Access*, vol. 9, pp. 102 460–102 489, 2021.

[21] K. N. Amirkolaii, A. Baboli, M. Shahzad, and R. Tonadre, "Demand forecasting for irregular demands in business aircraft spare parts supply chains by using artificial intelligence (ai)," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 15 221–15 226, 2017.

[22] S. Kaushik, A. Choudhury, P. K. Sheron, *et al.*, "Ai in healthcare: Time-series forecasting using statistical, neural, and ensemble architectures," *Frontiers in big data*, vol. 3, p. 4, 2020.

[23] E. Hadavandi, A. Ghanbari, K. Shahanaghi, and S. Abbasian-Naghneh, "Tourist arrival forecasting by evolutionary fuzzy systems," *Tourism Management*, vol. 32, no. 5, pp. 1196–1203, 2011.

[24] M. A. M. Daut, M. Y. Hassan, H. Abdullah, H. A. Rahman, M. P. Abdullah, and F. Hussin, "Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: A review," *Renewable and Sustainable Energy Reviews*, vol. 70, pp. 1108–1118, 2017.

[25] M. Wang, L. Zhao, R. Du, *et al.*, "A novel hybrid method of forecasting crude oil prices using complex network science and artificial intelligence algorithms," *Applied energy*, vol. 220, pp. 480–495, 2018.

[26] M. Q. Raza and A. Khosravi, "A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings," *Renewable and Sustainable Energy Reviews*, vol. 50, pp. 1352–1372, 2015.

[27] X. Ding, Y. Zhang, T. Liu, and J. Duan, "Deep learning for event-driven stock prediction," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[28] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, "Modeling the stock relation with graph network for overnight stock movement prediction," in *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, 2021, pp. 4541–4547.

[29] R. Akita, A. Yoshihara, T. Matsubara, and K. Uehara, "Deep learning for stock prediction using numerical and textual information," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, IEEE, 2016, pp. 1–6.

[30] S. Soni, "Applications of anns in stock market prediction: A survey," *International Journal of Computer Science & Engineering Technology*, vol. 2, no. 3, pp. 71–83, 2011.

[31] E. Hadavandi, H. Shavandi, and A. Ghanbari, "Integration of genetic fuzzy systems and artificial neural networks for stock price forecasting," *Knowledge-Based Systems*, vol. 23, no. 8, pp. 800–808, 2010.

[32] F. Z. Xing, E. Cambria, and R. E. Welsch, "Natural language based financial forecasting: A survey," *Artificial Intelligence Review*, vol. 50, no. 1, pp. 49–73, 2018.

[33] R. Ren, D. D. Wu, and T. Liu, "Forecasting stock market movement direction using sentiment analysis and support vector machine," *IEEE Systems Journal*, vol. 13, no. 1, pp. 760–770, 2018.

[34] D. Yan, Q. Zhou, J. Wang, and N. Zhang, "Bayesian regularisation neural network based on artificial intelligence optimisation," *International Journal of Production Research*, vol. 55, no. 8, pp. 2266–2287, 2017.

[35] P. M. Tsang, P. Kwok, S. O. Choy, *et al.*, "Design and implementation of nn5 for hong kong stock price forecasting," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 4, pp. 453–461, 2007.

[36] S. Nayak, B. B. Misra, and H. S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 6, no. 2014, pp. 257–269, 2014.

[37] L. Chen, Z. Qiao, M. Wang, C. Wang, R. Du, and H. E. Stanley, "Which artificial intelligence algorithm better predicts the chinese stock market?" *IEEE Access*, vol. 6, pp. 48 625–48 633, 2018.

[38] L.-Y. Wei, T.-L. Chen, and T.-H. Ho, "A hybrid model based on adaptive-network-based fuzzy inference system to forecast taiwan stock market," *Expert Systems with Applications*, vol. 38, no. 11, pp. 13 625–13 631, 2011.

[39] T. L. D. Huynh, E. Hille, and M. A. Nasir, "Diversification in the age of the 4th industrial revolution: The role of artificial intelligence, green bonds and cryptocurrencies," *Technological Forecasting and Social Change*, vol. 159, p. 120 188, 2020.

[40] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[41] P. Cortez, M. Rocha, and J. Neves, "Genetic and evolutionary algorithms for time series forecasting," in *Engineering of Intelligent Systems: 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2001 Budapest, Hungary, June 4–7, 2001 Proceedings 14*, Springer, 2001, pp. 393–402.

[42] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary computation*, vol. 1, no. 1, pp. 1–23, 1993.

[43] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[44] I. Rechenberg, "The evolution strategy. a mathematical model of darwinian evolution," in *Synergetics—From Microscopic to Macroscopic Order: Proceedings of the International Symposium on Synergetics at Berlin, July 4–8, 1983*, Springer, 1984, pp. 122–132.

[45] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Adaption of evolutionary programming to the prediction of solar flares," Tech. Rep., 1966.

[46] J. R. Sampson, *Adaptation in natural and artificial systems (john h. holland)*, 1976.

[47] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," *SIAM journal on computing*, vol. 2, no. 2, pp. 88–105, 1973.

[48] W. Härdle, *Applied nonparametric regression*. Cambridge university press, 1990.

[49] P. Hall and J. D. Hart, "Nonparametric regression with long-range dependence," *Stochastic Processes and Their Applications*, vol. 36, no. 2, pp. 339–351, 1990.

[50] G. M. Jenkins and M. Priestley, "The spectral analysis of time-series," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 19, no. 1, pp. 1–12, 1957.

[51] P. J. Brockwell and R. A. Davis, *Time series: theory and methods*. Springer science & business media, 1991.

[52] B. Rossi, "Exchange rate predictability," *Journal of economic literature*, vol. 51, no. 4, pp. 1063–1119, 2013.

[53] E. F. Fama, "Forward and spot exchange rates," *Journal of monetary economics*, vol. 14, no. 3, pp. 319–338, 1984.

[54] *Index version*, NASDAQ, 2020. [Online]. Available: https://indexes.nasdaqomx.com/Index/Overview/COMP, Jan 23, 2024.

[55] S. Mehle, "Nasdaq composite: Benchmark for the 21st century," NASDAQ, Tech. Rep., 2023.

[56] *Dow jones averages methodology*, S&P Global, 2023. [Online]. Available: https://www.spglobal.com/spdji/en/methodology/article/dow-jones-averages-methodology/, Jan 23, 2024.

[57] *S&p u.s. indices methodology*, S&P Global, 2024. [Online]. Available: https://www.spglobal.com/spdji/en/indices/equity/sp-500/#overview/, Jan 23, 2024.