ASSESSMENT OF ARTIFICIAL NEURAL NETWORK TO IMPROVE HIDDEN
MARKOV MODEL FOR FINANCIAL DATA

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DİLEK AYDOĞAN-KILIÇ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
FINANCIAL MATHEMATICS

JULY 2022

Approval of the thesis:

## ASSESSMENT OF ARTIFICIAL NEURAL NETWORK TO IMPROVE HIDDEN MARKOV MODEL FOR FINANCIAL DATA

submitted by **DİLEK AYDOĞAN-KILIÇ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Financial Mathematics Department, Middle East Technical University** by,

Prof. Dr. Ayşe Sevtap Selçuk-Kestel
Dean, Graduate School of **Applied Mathematics**

Prof. Dr. Ayşe Sevtap Selçuk-Kestel
Head of Department, **Financial Mathematics**

Prof. Dr. Ayşe Sevtap Selçuk-Kestel
Supervisor, **Actuarial Sciences, IAM, METU**

**Examining Committee Members:**

Prof. Dr. Ceylan Yozgatlıgil
Department of Statistics, METU

Prof. Dr. Ömür Uğur
Scientific Computing, IAM, METU

Prof. Dr. A. Sevtap Selçuk-Kestel
Actuarial Sciences, IAM, METU

Assist. Prof. Dr. Uğur Karabey
Actuarial Sciences, Hacettepe University

Assist. Prof. Dr. Oytun Haçarız
Actuarial Sciences, Karabük University

**Date:**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name:    DİLEK AYDOĞAN-KILIÇ

Signature            :

# ABSTRACT

### ASSESSMENT OF ARTIFICIAL NEURAL NETWORK TO IMPROVE HIDDEN MARKOV MODEL FOR FINANCIAL DATA

Aydoğan-Kılıç, Dilek

Ph.D., Department of Financial Mathematics

Supervisor    : Prof. Dr. Ayşe Sevtap Selçuk-Kestel

July 2022, 89 pages

The aim of this thesis is to eliminate the possible weaknesses of HMMs, which is a successful statistical model that is frequently used in time series modeling. Depending on the selection of the initial parameters of the HMMs, RNN is used as a solution to the failure to reach the global maximum, and it is aimed to benefit from the classification power of this method. The hybrid model, which is developed with this motivation, is built in a way that is suitable for use in non-categorical data, contrary to the version generally used in the literature.

In this thesis, the hybrid model, which is effective in the development of speech recognition in the literature, is reconstructed and applied to financial data. Additionally, a multivariate comparison is conducted in order to identify the effect of the other variables in the model. Therefore, apart from univariate models, bivariate and trivariate models are also constructed. Moreover, classical HMM and RNN are applied and compared with the Hybrid model results. The applications use daily closing prices for the S&P 500 and Nasdaq and daily EUR/USD exchange rates from 2000 to 2021. In comparison to the single HMM and RNN methods, the accuracy in forecasting is significantly increased.

Keywords: Hidden Markov Model, Neural Network, stock prices, forecasting, EUR/USD exchange rates.

# ÖZ

## FİNANSAL VERİLERDE SAKLI MARKOV MODELİNİ GELİŞTİRMEK İÇİN YAPAY SİNİR AĞININ DEĞERLENDİRİLMESİ

Aydoğan-Kılıç, Dilek

Doktora, Finansal Matematik Bölümü

Tez Yöneticisi    : Prof. Dr. Ayşe Sevtap Selçuk-Kestel

Temmuz 2022, 89 sayfa

Bu tezin amacı zaman serisi modellemelerinde sıklıkla kullanılan başarılı bir istatistiksel model olan Gizli Markov Modeller'in (GMM) olası zayıf yönlerini gidermektir. GMM'lerin başlangıç parametrelerinin seçimine bağlı olarak global maksimuma ulaşamama durumuna çözüm olarak Tekrarlayan Sinir Ağı (TSA) kullanılmaktadır ve bu yöntemin sınıflandırma gücünden faydalanılması amaçlanmıştır. Bu motivasyonla geliştirilen Hibrit model literatürde genelde kullanılan halinin tersine kategorik olmayan verilerde kullanılmaya uygun şekilde inşa edilmiştir.

Bu çalışmada literatürde konuşma tanımanın geliştirilmesinde etkili olan hibrit model yeniden kurgulanarak finans verilerine uygulanmaktadır. Ek olarak, modeldeki farklı değişkenlerin etkisini belirlemek için çok değişkenli bir karşılaştırma yapılır. Bu nedenle, tek değişkenli modellerin yanında iki değişkenli ve üç değişkenli modeller de geliştirilmiştir. Uygulamada 2000'den 2021'e kadar S&P 500 ve Nasdaq için günlük kapanış fiyatları ve günlük EUR/USD döviz kurları kullanılmaktadır. Önerilen hibrit model tek başına GMM ve TSA yöntemlerine kıyasla tahminlemelerin doğruluk oranında önemli ölçüde bir artış sağlamıştır.

Anahtar Kelimeler: Saklı Markov Modeller, Yapay Sinir Ağları, hisse senedi fiyatları, tahminleme, EUR/USD döviz kuru.

# ACKNOWLEDGMENTS

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| AI | Artificial Intelligence |
| AIC | Akaike Information Criteria |
| ANN | Artificial Neural Network |
| AR | Autoregressive |
| ARMA | Autoregressive Moving Average |
| BHMM | Bivariate Hidden Markov Model |
| BIC | Bayesian Information Criteria |
| CI | Conditional Independence |
| CNN | Convolutional Neural Network |
| EUR/USD | Euro Dollar Exchange Rate |
| GA | Genetic Algorithm |
| GARCH | Generalized AutoRegressive Conditional Heteroskedasticity |
| HMM | Hidden Markov Model |
| LSTM | Long Short-Term Memory |
| LVQ | Learning Vector Quantization |
| MAE | Mean Absolute Error |
| MAPE | Mean Absolute Percentage Error |
| mllk | Minus log-likelihood |
| MSE | Mean Square Error |
| Nasdaq | National Association of Securities Dealers Automated Quotations |
| ReLU | Rectified Linear Unit |
| RMSE | Root Mean Square Error |
| S&P 500 | Standard and Poor's 500 |
| SVM | Support-vector machine |
| US | United States |

# CHAPTER 1

# INTRODUCTION

Machine learning (ML) is used in a wide variety of fields, including pattern recognition [8], human language technologies [15], computer vision [34], traffic prediction [9], image recognition [20] and medical diagnosis [35]. Through ML algorithms, researchers have a chance to develop ambitious techniques in finance that have been developing for nearly three decades [24], [61]. A novel application of Recurrent Neural Network (RNN) and Hidden Markov Model (HMM) methods is presented in this thesis that enables them to be applied to financial data in a more amenable way.

Several known and unknown factors affect the indicators in finance, making them complex and nonlinear. As an advanced statistical approach, HMMs are used in machine learning applications due to their fast and powerful learning algorithm [4]. However, this model may still have weaknesses due to the initial settings of the parameters. Our main goal in this study is to remove the possible weaknesses of the HMM algorithm by using the Artificial Neural Networks (ANN) classification power and thereby benefit from the algorithm's learning properties. More specifically, RNN, which is a type of ANN, is used in this thesis to benefit from the contribution of the information on the previous time steps. Although another approach in this study can be to use powerful methods such as Long-Short Term Memory (LSTM) and Convolution Neural Networks (CNN) in the hybrid model, RNN is used for its simplicity at the first stage, since the proposed nested structure has to be coded in detail without using package functions.

The idea of using this hybrid model comes from its promising results in the area of speech recognition in the literature. We reconstruct the hybrid model to be applicable to financial data.

Although finance is one of the most crucial and, therefore, one of the areas in need of accurate modeling, the number of studies benefit from ML applications is not significant. The modeling of stock prices is challenging due to time dependence, volatility, and other factors. HMM is a frequently preferred in dealing with these difficulties and therefore, beneficial in stock price prediction [26]. Although it successfully analyzes and predicts time depending phenomena, it still has some weaknesses. The correct determination of the initial variables is an important factor for the model to work well, and otherwise it can be stuck at local maximums

in the maximum likelihood search. One of the most important motivations of this thesis is to determine a model that provides flexibility in the selection of initial parameters and provide a significant benefit in this area.

In the literature, ANN is employed to transform the data by feeding the HMM model for the maximization of likelihood of the EM algorithm. ANN acts like a black-box in this structure. In the field of speech recognition, the aim is to increase the likelihood directly, and the cost function is chosen as a negative likelihood. Our motivation here is to modify this method for non-categorical (interval) data contrary the one in speech recognition. Financial observations whose developments in time is continuous is focused in this study, as the market with hidden factors has enormous influence on the price behaviour. The implementation and modification of such approach in non-categorical financial indicators are missing in the literature.

In addition, the influence of different variables and the number involved in the price behaviour can be well captured by HMM and the proposed hybrid approach. Multivariate time series models have become widely used in many real-world fields like weather data analysis [62], [55], health care [33], finance [13], [16], and others [63], [58], [21]. However, there are not a significant number of studies experimenting with multivariate HMMs.

## 1.1 Aim of the Study

In this study, the possible weak point of the HMM algorithm, which is resembling the selection of the initial parameter is improved by augmenting it with the classification power of the ANN and to benefit from its learning algorithm as well. As a basis for motivation, it is known that the ANN-HMM hybrid model improves the accuracy of speech recognition in the literature. To apply such hybrid model to finance data requires reconstruction and modification.

Since ANN has sufficient classification power due to the nature of the data, the hybrid model is used in speech recognition literature commonly. In the hybrid model used in the literature, the cost function of ANN is chosen as likelihood required in HMM. On the other hand, in this thesis Neural Network is used to optimize the whole model, not just HMM part, by comparing the HMM predictions to the actual values. Therefore, it is not just focusing on optimizing the parameters of the HMM. Namely, it uses the Mean Square Error (MSE) metric as a cost function to converge as close as possible to the original data. However, forecasting results, which are intended to approximate actual values, are calculated with HMM parameters as explained in the following sections of the thesis. Therefore, the HMM parameters are also indirectly optimized by RNN while converging to the real values.

Furthermore, while the left right HMM is commonly utilized in the literature that uses ANN-HMM, the HMM section in this thesis is built as an ergodic HMM, since it would be more appropriate to provide the ability for transition to previous states in financial data. The left right HMM is a type of HMM that is popularly used in speech analysis and models speech as a time sequence of distinct events that start at an initial state and end at a final state. It has a

left-to-right transition to the next state as well as a self-transition [32]. However, an ergodic HMM is a type of HMM that all the states are fully connected.

This study introduces a novel approach that can be more effective in especially non-categorical observations like financial data. In addition, an examination of variable selection in HMM and the hybrid model are performed. The purpose is to capture temporal and multivariate dependencies in the multivariate time series data by using HMM. Instead of modeling the financial time series solely, the advantage of using the dependencies between variables is considered. Hence, the model is designed for univariate, bivariate and trivariate RNN-HMMs to show its accuracy in univariate and multivariate components.

Additionally, an application of proposed approach to the real life problems is aimed to show the increase in the accuracy. S&P 500, Nasdaq daily closing prices and EUR/USD exchange rates are used in order to examine the performance of the model.

The EM-algorithm is used to estimate parameters at each step, and the gradients are generated using the cost function of the forecast values. The aim is to merge both models by increasing the classification capability of the EM method while keeping the RNN's learning from real observations, rather than by modifying the structure of the RNN portion.

## 1.2 Literature Review

In this part, the studies done in the area of univariate HMM are introduced briefly but the ones about the multivariate HMMs and the hybrid models including HMM and ANNs are discussed in more detail.

HMMs are general-purpose models and it is possible to use these models in a very wide range of applications. Some of the studies that use HMMs in speech recognition are [3], [60], [51] and [44]. It has also applications in genetics [37] , DNA sequence analysis [14], molecular biology [36] and economics [27]. Furthermore, there are special applications such as credit card fraud detection [7], customer relationship dynamics [45], signature verification [64] and real-time traffic sensing [25]. Moreover, various types of time series such as continuous-valued, circular, multivariate, as well as binary data, bounded and unbounded counts and categorical observations are modeled by HMMs [39].

A systematic review of HMM and its applications for 1982-2019 time period by [43]. According to this study HMMs are widely used in the area of speech recognition, human activity recognition and bioinformatics since they constitute 25%, 25% and 19% of all HMM studies, respectively.

Time series applications of HMMs in the literature can be summarized as follows. A Poisson hidden Markov model is applied by [40] mathematically to formulate the statistical interdependency among deterioration processes of pavement surfaces. It uses the panel data of road sections in different time intervals. HMM is used for modelling long-term persistence in

multi-site rainfall time series since it has an explicit mechanism to produce long-term wet and dry periods [59]. Univariate HMM is applied to forecast some of the airlines stock by [28]. Moreover, it also compares the forecasting results with an ANN application and it concludes that they have similar performance according to the MAPE results.

Most of the studies using HMM in time series analyzes and even in other analyzes are applied with univariate case. However, cylindrical time series are modeled by bivariate HMMs (BHMM). It is aimed in [38] to model cylindrical time series of intensities and angles which arise often in environmental research. It segments the data according to some specific environmental conditions. To deal with the complicated nature of identification of sea regimes by environmental multivariate time series because of the missing values in the data, skewness of some variables, and the temporal autocorrelation of the measurements HMM is also implemented [12]. Bivariate Mixed HMM with stochasticity with two hidden states remission and exacerbation and two observation sources patient reported outcomes and forced expiratory volume is used in [11]. Moreover, a Bivariate HMM is applied to model claim dependence with the assumption that claim numbers and aggregate claims are serially and mutually dependent through an underlying hidden state. It constructs three different BHMMs: Poisson–Normal HMM, Poisson–Gamma HMM, and Negative Binomial–Gamma HMM [47].

Moreover, there are several studies that combines HMMs with other methods in order to benefit from the advantages of this model such as Fuzzy-HMMs [30], AR-HMMs [54], ARMA-HMMs [41], GARCH-HMMs [65], SVM-HMM [23] and ANN-HMMs. Due to the focus on RNN-HMM, we skip details of literature for those.

ANN and HMM is combined in [53] to use in speech recognition system. It prefers Learning Vector Quantization (LVQ)-ANNs for the neural network part. It applies this hybrid model in two approaches: recognizing the entire command by HMM by using the recognized words by ANN and recognition of whole-word by HMM basing on the phonemes determined by the ANN. Hence, even if ANN misses some part or identifies wrong, HMM part can recover it.

Vector quantization techniques with recurrent neural networks are used in [4]. After using neural network it applies HMM part by feeding it with output of the previous part. In this study, the ANN and the HMM are trained separately. This method is determined to be a promising method. However, local optimization of the learning problem (separate training) is indicated as a possible weakness.

Multilayered and recurrent ANNs with HMMs [5] are integrated and applied for continuous speech recognition. The outputs of ANN constitute the observation vectors of HMM. It is one of the studies that aim to combine the classification power of ANNs with the time-series modeling capability of HMMs. Gradient of the optimization criterion of HMM with respect to the transformed observations is sent to ANN for reweighting. Actual observations for time are fed into ANN part and the output vector of this part used as the input of HMM part. Optimization criterion is the likelihood that is generated by an HMM. The optimized parameters here are the parameters of the HMM part. Namely, it estimates the parameters of

4

ANN and HMM with a joint global optimization. According to the application results 86% accuracy is obtained when the hybrid system is used, as opposed to 53% accuracy when an ANN is used.

A fusion model that combines ANN, HMM and Genetic Algorithms (GA) is proposed by [29] whose hybrid model is applied on two stock prices which is primarily taken as a guide in this thesis. It uses left-right HMM and ANN to optimize the maximum likelihood of the HMM part. In this model the aim is adding noise to the real data by applying ANN part and providing a better performance chance to HMMs by feeding it with transformed input data. Actual observations of four variables for time are fed into ANN part and the obtained output vector is given as input to HMM with 4-state. Actually, the reason why the model is selected as 4 states is to ensure that it is the same as attributes in the observation vectors and this is a limitation of this study. HMM and the Hybrid model results are compared with respect to MAPE results and it is shown that the forecast accuracy is raised up significantly.

Another study [57] that presents the use of a hybrid ANN-HMM for automatic speech recognition. ANN is trained to estimate the posterior probabilities of HMM. Training part of this application includes 20 exemplars and testing part includes 38 exemplars. 9-state HMM and one big ANN with 20 nodes in the hidden layer, 10 nodes at the output layer and 220 nodes in the input layer are combined and 13000 back-propagation training iterations are used. The Hybrid model provides a significant decrease in the percentage of misrecognized exemplars.

ANN and HMM are also combined by [56] for speech recognition. However, the methodology is totally different from the one in this thesis since it applies HMM part firstly to optimize the actual data. The output of HMM is given to feedforward ANN for further classification.

As the literature review reported in this section shows, HMM does not have as much work in time series and especially in multivariate time series modeling as it does in some specific areas. In addition to contributing to this field, this thesis also provides improvement in HMM by using the classification power of RNN. There is not a significant number of studies of the Hybrid model in the literature. Many of those present either optimizations of these methods separately or their joint optimizations only for categorical data.

## 1.3 Contributions of the Thesis

The main contributions can be listed in two parts. First, by presenting a new approach, HMM models and RNN models are combined for the use in non-categorical and interval data such as finance data since HMMs have an important place in time series modeling, and RNNs are successful in classification. Second, inferences on usage of different types and numbers of variables are made for both the HMM and the proposed model. The further contributions are listed as follows:

    i The hidden states under stock prices are examined by using the global decoding.

ii The proposed algorithm is presented by combining HMM and RNN models for the implementation to financial data.

iii Simple HMM and ANN models, and the proposed Hybrid model are applied for univariate, bivariate and trivariate cases. All these applications are compared with each other and inferences are made for the models and the number of different variables.

The thesis is comprised from five main chapters and two appendices. In Chapter 2, the HMM is explained for the univariate case in its most basic form, and then for the bivariate model to expand it to multivariate models. Additionally, the basics of Feedforward and Recurrent ANNs are explained. Chapter 3 introduces the financial data, S&P 500, Nasdaq and EUR/USD, implemented for the illustration of the proposed methodology. Then, the methodology of the RNN-HMM hybrid model together with the conventional models of HMM and ANN is explained in detail. The results of the HMM, RNN and Hybrid model applications are presented and discussed in Chapter 4. All cases, namely the traditional and newly proposed model, cases with different variable numbers are compared in graphs and tables in this section. In Chapter 5, the thesis is concluded by summarizing, discussing the results and proposing future studies.

# CHAPTER 2

# PRELIMINARIES

In this chapter, the mathematical background used in this thesis is provided. Firstly, the univariate and bivariate versions of Hidden Markov Models are explained. Then, the Recurrent Neural Network structure is overviewed briefly.

## 2.1   Hidden Markov Model

HMMs are the extended types of Markov models. While in Markov models, each state corresponds to an observable event, in HMMs observations are probabilistic functions of states [49]. An HMM is an embedded stochastic process with an underlying and unobservable stochastic process. It can be observed by only a stochastic process that produces the same occurrence sequence. Cheng-Der Fuh [19] states, "A hidden Markov model is defined as a parametrized Markov chain in a Markovian random environment, with the underlying environmental Markov chain viewed as missing data." The key idea of HMM is to describe a probability distribution over an infinite number of possible sequences of observations [18].

Since HMMs are general-purpose models, they can be used for modeling various types of time series such as continuous-valued, circular, multivariate, as well as binary data, bounded and unbounded counts and categorical observations [39]. Some of the applications of HMMs are speech recognition [48], DNA sequence analysis [14], molecular biology [36], stock market forecasting [28] and economics [27].

The EM algorithm and the Viterbi algorithm are the solutions to the main problems of HMM [49]. The EM algorithm is the most famous algorithm for getting the parameters that maximize the likelihood of the model. On the other hand, the best sequences of hidden states are estimated by the Viterbi algorithm.

The HMM is a tool for representing probability distributions over sequences of observations [22]. The model gets its name from two properties. The first one is that each observation at time $t$ is generated by a hidden state $X_t$. Second, each state $X_t$ is independent of all the states prior to state $X_{t-1}$, namely, the states satisfy the Markov property. This structure is presented in Figure 2.1 as well as it can be expressed as follows [22]:
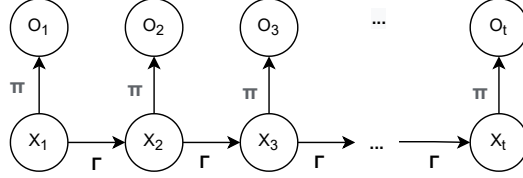
Figure 2.1: Dependence structure for a Hidden Markov Model

$$Pr(X_t|X^{(t-1)}) = Pr(X_t|X_{t-1}), \quad t = 2, 3, ... \tag{2.1}$$

$$Pr(O_t|O^{(t-1)}, X^{(t)}) = Pr(O_t|X_t), \quad t \in N, \tag{2.2}$$

where $O_t$ is the observation and $X_t$ is the state at time $t \in N$; $X^{(t)}$ and $O^{(t)}$ are history from time 1 to $t$; to illustrate $X^{(t)}$ denotes $\{X_0 = i_0, X_1 = i_1, ..., X_t = i_t\}$ for each $i_j \in E$ where $E$ is a discrete state space which is a set of values that each $X_t$ can take [2].

Before the explanation of EM Algorithm that is used to estimate the parameters of the HMM, some definitions are necessary to be presented.

**Definition 1.** *Transition Probabilities* , $\gamma_{ij}$, identify the probability of moving to state $j$ from the current state $i$, given as

$$Pr(X_t = j|X_{t-1} = i) = \gamma_{ij}, \quad t = 2, 3, \cdots \tag{2.3}$$

Moreover, the square matrix with $(i, j)$ element $\gamma_{ij}$ is the "Transition Probability Matrix":

$$\Gamma = \begin{pmatrix} \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & \ddots & \vdots \\ \gamma_{m1} & \cdots & \gamma_{mm} \end{pmatrix}, \tag{2.4}$$

where $m$ denotes the number states of the Markov chain. It is essential to note that the sum of rows is equal to 1.

**Definition 2.** *Unconditional Probabilities* are the elements of the row vectors that are denoted by

$$\pi(t) = (Pr(X_t = 1) \ldots Pr(X_t = m)), \tag{2.5}$$

where $Pr(X_t = j)$ is the probability of being in a given state $j$ at a given time $t$. $\pi(1)$ is the *initial distribution* of Markov chain which is denoted as $\delta$ in the rest of this thesis.

8

**Definition 3.** *Emission Probabilities* are the probabilities of observing a particular value provided that the system is in one of the hidden states [6].

$$p_i(o) = Pr(O_t = o | X_t = i) \tag{2.6}$$

Moreover, $P(o)$ refers to the diagonal matrix consisting of emission probabilities of observation $O_t = o$:

$$P(o) = \begin{pmatrix} p_1(o) & & 0 \\ & \ddots & \\ 0 & & p_m(o) \end{pmatrix}. \tag{2.7}$$

EM algorithm is used to estimate these observation distributions, the parameters initial probability $\delta$, transition probability matrix $\Gamma$ and the emission probabilities.

### 2.1.1 EM Algorithm

The EM algorithm is an iterative method that is used to find the maximum-likelihood estimation of the parameters of an underlying distribution from a given data set when the data is incomplete or has missing values [17]. The algorithm starts with a likelihood value and gets new likelihood values by iteration. At each iteration step, the likelihood of the model is better, or the same [2]. In order to apply this estimation method, we need the backward and forward probabilities.

Forward probability, $\alpha_t$, is to produce $O^{(t)}$ while ending up in state $j$ and it is presented for $t = 1, 2, \ldots, T$ and $j = 1, 2, \ldots, m$ as

$$\begin{aligned} \alpha_t(j) &= Pr(O^{(t)} = o^{(t)}, X_t = j), \\ \alpha_t &= \delta P(o_1)\Gamma P(o_2)\Gamma P(o_3)\ldots\Gamma P(o_t). \end{aligned} \tag{2.8}$$

Backward probability, $\beta_t$, is the probability of producing the observations $o_{t+1}, \ldots, o_T$ given that the system is at state $i$ at time $t$ for $t = 1, 2, \ldots, T-1$, $i = 1, 2, \ldots, m$. It is formulated in Equation (2.9) and for convenience, the vector $(O_k, O_{k+1}, \ldots, O_l)$ is denoted by $O_k^l$.

$$\begin{aligned} \beta_t(i) &= Pr(O_{t+1} = o_{t+1}, O_{t+2} = o_{t+2}, \ldots, O_T = o_T | X_T = i) \\ &= Pr(O_{t+1}^T = o_{t+1}^T | X_T = i) \\ \beta_t' &= \Gamma P(o_{t+1})\Gamma P(o_{t+2})\ldots\Gamma P(o_T)1' \end{aligned} \tag{2.9}$$

The log-likelihood of the complete data that consists of the observed and missing ones is

$$\begin{aligned} \log(Pr(o^{(T)}, x^{(T)})) &= \sum_{j=1}^{m} u_j(1) \log \delta_j + \sum_{j=1}^{m}\sum_{k=1}^{m}(\sum_{t=2}^{T} v_{jk}(t)) \log \gamma_{jk} \\ &+ \sum_{j=1}^{m}\sum_{t=1}^{T} u_j(t) \log p_j(o_t), \end{aligned} \tag{2.10}$$

9

where $u_j(t) = 1$ if and only if $X_t = j$, for $t = 1, \ldots, T$ and $v_{jk}(t) = 1$ if and only if $X_{t-1} = j$ and $X_t = k$ for $t = 2, \ldots, T$.

In the expectation step, instead of $v_{jk}(t)$ and $u_j(t)$, the conditional expectations of being in a state at a particular time given the observations is employed:

$$
\begin{aligned}
\hat{u}_j(t) &= Pr(X_t = j | o^{(T)}) \\
&= \frac{\alpha_t(j)\beta_t(j)}{L_T},
\end{aligned}
\tag{2.11}
$$

$$
\begin{aligned}
\hat{v}_{jk}(t) &= Pr(X_{t-1} = j, X_t = k | o^{(T)}) \\
&= \frac{\alpha_{t-1}(j)\gamma_{jk}p_k(o_t)\beta_t(k)}{L_T}.
\end{aligned}
\tag{2.12}
$$

Then, in the maximization step, the complete data likelihood is maximized by maximizing each term of Equation (2.10) and it is observed that the terms of the log-likelihood depend on initial distribution $\delta$, transition probability matrix $\Gamma$, and state-dependent distributions. When we maximize each component of the complete data log-likelihood, the maximizing values of relevant parameters are determined as follows:

i. $\delta_j = \frac{\hat{u}_j(1)}{\sum_{j=1}^m \hat{u}_j(1)} = \hat{u}_j(1),$

ii. $\gamma_{jk} = \frac{\sum_{t=2}^T \hat{v}_{jk}(t)}{\sum_{k=1}^m \sum_{t=2}^T \hat{v}_{jk}(t)},$

iii. Since the third term is on state-dependent distributions, this part varies by the type of the distribution. To give some examples, the maximizing value of this term for Poisson-HMM is $\hat{\lambda}_j = \frac{\sum_{t=1}^T \hat{u}_j(t)o_t}{\sum_{t=1}^T \hat{u}_j(t)}$ and the values for Normal-HMM are $\hat{\mu}_j = \frac{\sum_{t=1}^T \hat{u}_j(t)o_t}{\sum_{t=1}^T \hat{u}_j(t)}$ and $\hat{\sigma}_j^2 = \frac{\sum_{t=1}^T \hat{u}_j(t)(o_t - \mu_j)^2}{\sum_{t=1}^T \hat{u}_j(t)}.$

## 2.1.2 Forecast, State Decoding and State Prediction

Forecast distributions, i.e., conditional distribution of $O_{t+h}$ given observations $O^T$ where $h$ is the forecast horizon, can be represented as

$$
\begin{aligned}
Pr(O_{T+h} = o | O^{(T)} = o^{(T)}) &= \frac{Pr(O^{(T)} = o^{(T)}, O_{T+h} = o)}{Pr(O^{(T)} = o^{(T)})} \\
&= \frac{\alpha_T \Gamma^h P(o)1'}{\alpha_T 1'}.
\end{aligned}
\tag{2.13}
$$

For state decoding, one can use local decoding that maximize the conditional distribution of each state for given observations as follows,

$$
\begin{aligned}
i* &= \underset{i=1,\ldots,m}{argmax} Pr(X_t = i | O^{(T)} = o^{(T)}) \\
&= \underset{i=1,\ldots,m}{argmax} \frac{\alpha_t(i)\beta_t(i)}{L_T}.
\end{aligned}
\tag{2.14}
$$

or can use the more common method Viterbi algorithm that estimates the most likely sequence of states. To illustrate the methodology of the Viterbi algorithm, firstly, the following probabilities are defined.

$$\varphi_{1i} = Pr(X_1 = i, O_1 = o_1)$$
$$= \delta_i p_i(o_1) \tag{2.15}$$

and,

$$\varphi_{ti} = \max_{x_1, x_2, \ldots, x_{t-1}} Pr(X^{(t-1)} = x^{(t-1)}, X_t = i, O^{(T)} = o^{(T)}) \tag{2.16}$$

for $t = 2, 3, \ldots, T$.

The relation between successive $\varphi$ 's can be shown by

$$\varphi_{tj} = (\max_i(\varphi_{t-1,i}\gamma_{ij}))p_j(o_t) \tag{2.17}$$

for $t = 2, 3, \ldots, T$ and $i = 1, 2, \ldots, m$.

Afterwards the most likely state sequence is estimated by

$$i_T = \underset{i=1,\ldots,m}{argmax} \varphi_{Ti} \tag{2.18}$$

and,

$$i_t = \underset{i=1,\ldots,m}{argmax}(\varphi_{ti}\gamma_{i,i_{t+1}}). \tag{2.19}$$

Last, the state prediction can be done by the following probabilities,

$$Pr(X_{T+h} = i|O^{(T)} = o^{(T)}) = \frac{\alpha_T \Gamma^h(, i)}{L_T} \tag{2.20}$$

where $\Gamma^h(, i)$ is the $i^{th}$ column of the $h^{th}$ power of transition matrix $\Gamma, \Gamma^h$, and $h = t - T$.

## 2.2 Model Selection

As the state number $m$ increases, the model fitting improves. However, more parameters are needed, but parsimony theory recommends using fewer parameters. Box and Jenkins (1970) indicates that in statistical models, certain constants and parameters are needed, and they must be estimated from the data. For adequate representations, we should engage the smallest number of parameters possible in practice. The study predicts that parsimony plays a critical role in the use of parameters in future studies [10].

In order to decide the number of the states, $m$, of the model a criterion that consider the advantages and disadvantages of higher $m$ is needed. The commonly used criteria for determining the number of parameters are Akaike Information Criteria (AIC) and Bayesian Information Criteria (BIC), which are defined as

$$AIC = -2\log L + 2p, \tag{2.21}$$

11
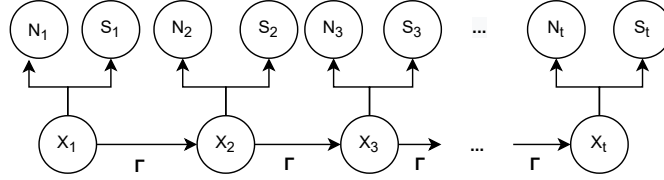
Figure 2.2: Dependence structure for a bivariate Hidden Markov Model

$$BIC = -2\log L + p\log T, \tag{2.22}$$

respectively. Here, $logL$ is the log-likelihood, $p$ is the number of parameters of the applied model, and $T$ is the number of observations.

AIC is a function of the model's likelihood and the number of parameters. This metric is negatively correlated with the likelihood of a model as well as positively correlated with the number of parameters. As a result, it makes sense to select the model with a lower AIC value. While the first term decreases, the second term, which is a penalty term, increases with the increase of $m$.

BIC is similar to AIC, but the penalty term is different. It can be seen from Equation (2.22) that when $T > e^2$, which is the case in most applications, the BIC gives more weight to the penalty term. Therefore, BIC generally tends to choose models with fewer parameters than AIC.

## 2.3 Bivariate Hidden Markov Models (BHMMs)

Let $N_t$ and $S_t$ ($t = 1, 2, \ldots$) be two variables that have dependencies between each other. Assume also that there is an unobservable background factor, which is defined by HMM. Moreover, as in the univariate HMMs each observation $N_t$ and $S_t$ at time $t$ is generated by a hidden state $X_t$ and each state is independent from all the states prior to state $X_{t-1}$. This dependence structure is shown in Figure 2.2 for the bivariate case.

Under the assumption of conditional independence (CI) although the series are still dependent conditional on the underlying hidden state $X_t := t = 1, 2, \ldots$ the variable $N$ at time $t$ and the variable $S$ at time $t$ are assumed to be independent [46]. Hence, the covariance between the variables are considered under the hidden states and a covariance matrix is not needed in the

calculations.

Now, we define a joint state-dependent distribution as

$$p_i(s_t, n_t) = P((S_t, N_t) = (s_t, n_t)|X_t = i) \tag{2.23}$$

where $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, m$.

Since we have CI assumption, we can write Equation (2.23) as

$$
\begin{aligned}
p_i(s_t, n_t) &= P((S_t, N_t) = (s_t, n_t)|X_t = i) \\
&= P(S_t = s_t|X_t = i)P(N_t = n_t|X_t = i).
\end{aligned} \tag{2.24}
$$

Different types of distributions lead to different joint distributions. For example, for Poisson-Normal HMM we can write

$$p_i(s_t, n_t) = (2\pi\sigma_i^2)^{-\frac{1}{2}} e^{-\frac{1}{2\sigma_i^2}(s_t - \mu_i)^2 - \lambda_i} \frac{\lambda_i^{n_t}}{n_t!}. \tag{2.25}$$

The most commonly used distributions in finance studies are assumed to follow normal distribution. Moreover, this thesis assumes normality in order to provide convenience in determining the initial parameters of multivariate HMMs. Bivariate Normal distribution is examined in this part in more detail. Under the assumption of conditional independence (CI) we have the following joint state-dependent distribution for time $t$ and state $i$,

$$
\begin{aligned}
p_i(s_t, n_t) &= P((S_t, N_t) = (s_t, n_t)|X_t = i) \\
&= P(S_t = s_t|X_t = i)P(N_t = n_t|X_t = i),
\end{aligned} \tag{2.26}
$$

where $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, m$.

For bivariate normal HMM, the joint distribution is expressed as

$$
\begin{aligned}
p_i(s_t, n_t) &= (2\pi\sigma_{1_i}^2)^{-\frac{1}{2}} e^{-\frac{1}{2\sigma_{1_i}^2}(s_t - \mu_{1_i})^2} (2\pi\sigma_{2_i}^2)^{-\frac{1}{2}} e^{-\frac{1}{2\sigma_{2_i}^2}(s_t - \mu_{2_i})^2} \\
&= (4\pi^2\sigma_{1_i}^2\sigma_{2_i}^2)^{-\frac{1}{2}} e^{-\frac{1}{2}\left(\frac{(S_t - \mu_{i_1})^2}{\sigma_{i_1}^2} + \frac{(N_t - \mu_{i_2})^2}{\sigma_{i_2}^2}\right)}.
\end{aligned} \tag{2.27}
$$

The likelihood function that will be maximized by using EM-algorithm is written by using the joint distributions as follows:

$$L_T = u(1)P(s_1, n_1)\Gamma P(s_2, n_2)\Gamma P(s_3, n_3)\ldots \Gamma P(s_T, n_T)1^T \tag{2.28}$$

where $u(1)$ is initial distribution function, and

$$
P(s, n) = \begin{pmatrix} p_1(s, n) & & 0 \\ & \ddots & \\ 0 & & p_m(s, n) \end{pmatrix}.
$$

13

Then, the complete data loglikelihood (CDLL) is defined as

$$\sum_{i=1}^{m} \sum_{l=1}^{T} \hat{u}_i(l) \log p_j(s_t, n_t)$$

where $\hat{u}_i(l) = \alpha_t(i)\beta_t(j)/L_T$.

For a bivariate normal HMM the log-likelihood, $F$, is

$$F = \sum_{l=1}^{T} \hat{u}_i(t) \left[ -\frac{1}{2} \log \left(4\pi^2 \sigma_{1_i}^2 \sigma_{2_i}^2\right) - \frac{1}{2} \left( \frac{(S_t - \mu_{i_1})^2}{\sigma_{i_1}^2} + \frac{(N_t - \mu_{i_2})^2}{\sigma_{i_2}^2} \right) \right].$$

Taking the derivative of $F$ with respect to the parameters of HMM and solving for the expressions maximizing $F$ yields

$$\hat{\mu}_{1j} = \frac{\sum_{t=1}^{T} \hat{u}_j(t) S_t}{\sum_{t=1}^{T} \hat{u}_j(t)},$$

$$\hat{\mu}_{2j} = \frac{\sum_{t=1}^{T} \hat{u}_j(t) N_t}{\sum_{t=1}^{T} \hat{u}_j(t)},$$

$$\hat{\sigma}_{1j}^2 = \frac{\sum_{t=1}^{T} \hat{u}_j(t) (S_t - \mu_j)^2}{\sum_{t=1}^{T} \hat{u}_j(t)},$$

$$\hat{\sigma}_{2j}^2 = \frac{\sum_{t=1}^{T} \hat{u}_j(t) (N_t - \mu_j)^2}{\sum_{t=1}^{T} \hat{u}_j(t)}.$$

## 2.4 Artificial Neural Network

Artificial intelligence (AI) started about the 1930s-1940s, and there have been significant developments since then. The aim of AI applications is to replicate the human and even animal intelligence behavior by simulating the brain structures and the thinking process by using computer models [31]. In basic ANNs, there are three types of neuron layers which are input, hidden, and output layers. While signal moves from input to output layers strictly without a feedback connection in feedforward neural networks, it also has a feedback connection in recurrent neural networks [1].

Feedforward neural networks is explained firstly to examine the RNN easily. According to the structure presented in Figure 2.3, each connection line has a connection weight. Although it is shown as all neurons have a connection to previous layer neurons, it might be the case of not having a connection with some neurons [1]. A one hidden layer RNN is explained in this section since it is used in the application part of the thesis.

Two weight matrices are included in one hidden layer ANN. The first weight matrix, $\mathbf{W}$, that connects the input layer with the hidden layer is a $n \times l$ matrix where $n$ and $l$ are the node size

14
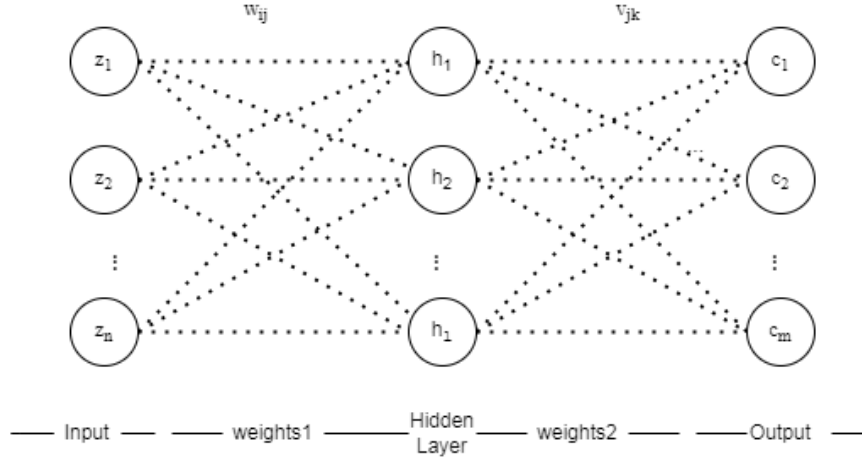
Figure 2.3: A feedforward neural network with a single hidden layer

of the input layer and hidden layer, respectively. The second weight matrix, $\mathbf{V}$, that connects the hidden layer with output layer is a $l \times m$ matrix where $m$ is the node number of output layer.

In Figure 2.3, $w_{ij}$ is the connection weight between the $i^{th}$ external neuron $z_i$ $(i = 1, \ldots, n)$ and the $j^{th}$ hidden neuron $h_j$ $(j = 1, \ldots, l)$, and $v_{jk}$ is the connection weight between $h_j$ and the $k^{th}$ output neuron $c_k$ $(k = 1, \ldots, m)$.

The nodes of hidden layer and output layer nodes are computed by Equations (2.29) and (2.30), respectively.

$$h_j = f\left(\sum_{i=1}^{n} w_{ij} z_i\right), \tag{2.29}$$

$$c_k = g\left(\sum_{j=1}^{l} v_{jk} h_j\right). \tag{2.30}$$

Here, $f$ and $g$ are the activation functions.

After this step in order to redefine the weights, it is needed to compare the output of the feedforward application to the actual observations. In this thesis backpropogation algorithm is used for reweighting. This algorithm uses gradients while determining the weights of the following iteration step, so the error function must be continuous and differentiable. Hence, activation functions, unlike the step function used in perceptron, are essential for the backpropagation learning algorithm [50]. Some of the most popular activation functions for backpropagation networks are sigmoid, tanh, ReLU, and leaky ReLU functions. Since sigmoid and tanh functions can cause vanishing problems and ReLU can not perform a backpropagation algorithm for negative values, leaky ReLU is preferred in this study.

Leaky ReLU and its derivative are expressed as follows:

$$f(x) = \begin{cases} 0.01x, & \text{if } x \leq 0 \\ x, & \text{if } x > 0 \end{cases} \tag{2.31}$$

$$\frac{d}{dx}f(x) = \begin{cases} 0.01, & \text{if } x < 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{2.32}$$

The optimal combinations of weights are searched by the learning algorithms. The weights are initialized firstly, and at each iteration, they are updated to minimize the error, defined as

$$E = \frac{1}{m} \sum_{k=1}^{m} (c_k - y_k)^2 \tag{2.33}$$

where $y_k$'s are the entries in the ground truth label $\vec{y}$ and $c_k$'s are the entries in the prediction matrix $\mathbf{C}$. In fact, Equation (3.7) is a Mean Square Error metric, but there are several loss function opportunities such as Mean Absolute Error, Mean Bias Error, Hinge Loss, and Cross-Entropy Loss.

By using Equation (2.29) and Equation (2.30) $E$ can be written as

$$\begin{aligned} E &= \frac{1}{m} \sum_{k=1}^{m} \left( g \left( \sum_{j=1}^{l} v_{jk} h_j \right) - y_k \right)^2 \\ &= \frac{1}{m} \sum_{k=1}^{m} \left( g \left( \sum_{j=1}^{l} v_{jk} f \left( \sum_{i=1}^{n} w_{ij} z_i \right) \right) - y_k \right)^2 \end{aligned} \tag{2.34}$$

The step that calculates the gradient of $E$ and the reweighing made is called as backpropagation. The error $E$ of the network is minimized by using an iterative process of gradient descent which is calculated as

$$\nabla E = \left( \frac{\partial E}{\partial \mathbf{W}}, \frac{\partial E}{\partial \mathbf{V}} \right) \tag{2.35}$$

Then, the increments in weight matrices are

$$\Delta \mathbf{W} \propto -\frac{\partial E}{\partial \mathbf{W}}, \tag{2.36}$$

$$\Delta \mathbf{V} \propto -\frac{\partial E}{\partial \mathbf{V}}, \tag{2.37}$$

and, more explicitly the increments of entries in weight matrices are

$$\Delta w_{ij} \propto -\frac{\partial E}{\partial w_{ij}}, \tag{2.38}$$

$$\Delta v_{jk} \propto -\frac{\partial E}{\partial v_{jk}}. \tag{2.39}$$

To write increments more explicitly, partial derivatives of $E$ should be written more explicitly. For this reason, the intermediate steps and prediction vector in the ANN structure are written in terms of weight matrices as follows:

$$\begin{aligned} \mathbf{C} &= f(g(\mathbf{Z} \times \mathbf{W}) \times \mathbf{V}) \\ &= f(\mathbf{L} \times \mathbf{V}) \end{aligned} \tag{2.40}$$

$$\theta = \mathbf{Z} \times \mathbf{W} \tag{2.41}$$

$$\mathbf{Q} = f(\mathbf{Z} \times \mathbf{W}) \times \mathbf{V} \tag{2.42}$$

where $\mathbf{Z}$ is the input matrix, $\mathbf{C}$ is the output matrix and $\mathbf{L}$ is the hidden layer matrix.

Then, the partial derivative of $E$ with respect to $\mathbf{W}$ can be written using the Chain rule:

$$\frac{\partial E}{\partial \mathbf{W}} = \frac{\partial E}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{W}} \tag{2.43}$$

and, we have

$$\frac{\partial E}{\partial \mathbf{L}} = \frac{\partial E}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial \mathbf{L}}. \tag{2.44}$$

Finally, the partial derivative of $E$ with respect to $\mathbf{W}$ is

$$\frac{\partial E}{\partial \mathbf{W}} = \frac{\partial E}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{W}} \tag{2.45}$$

Moreover, the partial derivative of $E$ with respect to $\mathbf{V}$ is written as follows

$$\frac{\partial E}{\partial \mathbf{V}} = \frac{\partial E}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{V}} \tag{2.46}$$

After finding the gradient for iteration step $i$ we recalculate $\mathbf{W}$ and $\mathbf{V}$ to use in the next iteration step $(i + 1)$ as follows:

$$\mathbf{W}(i + 1) = \mathbf{W}(i) + \lambda \frac{\partial E}{\partial \mathbf{W}(i)}, \tag{2.47}$$
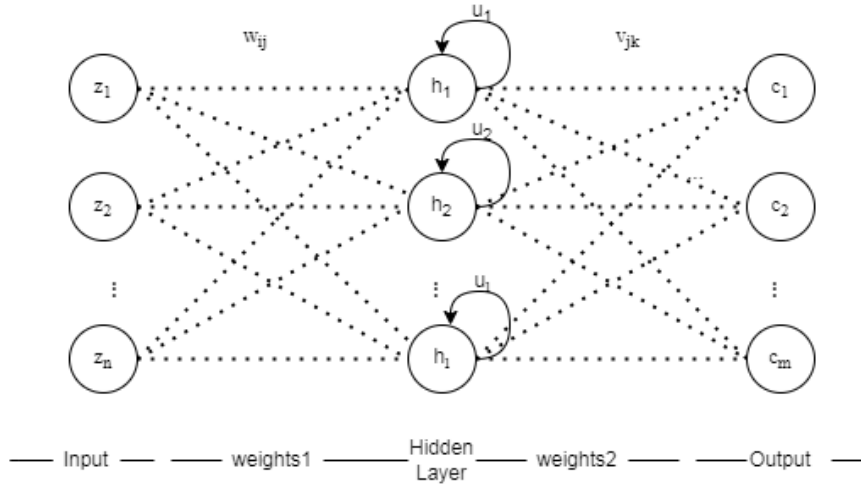
Figure 2.4: A recurrent neural network with a single hidden layer

$$\mathbf{V}(i+1) = \mathbf{V}(i) + \lambda \frac{\partial E}{\partial \mathbf{V}(i)}. \tag{2.48}$$

where $\lambda$ is the learning rate which determines the step size at each iteration while trying to reach the minimum loss function. The learning rate needs to be chosen small enough that the loss function can approximate 0. However, it should not be too small since the increment in weights would be so tiny, and the algorithm would work so slowly in such a case.

In the Feedforward Neural Network structure, incoming information is only processed forward. However, RNN is a type of ANNs that allows the result not only based on the current input but also on other inputs. According to the structure presented in Fig. 2.4 RNN additionally contains a link term with the previous layer in the calculation of the current hidden layer:

$$h_j{}^{(t)} = f\left(\sum_{i=1}^{n} w_{ij} z_i + u_j h_j{}^{(t-1)}\right) \tag{2.49}$$

where $h_j{}^{(t)}$ and $h_j{}^{(t-1)}$ are $j^{th}$ hidden layer nodes in time steps $t$ and $t-1$, respectively.

Then, the gradient of E becomes

$$\nabla E = \left(\frac{\partial E}{\partial \mathbf{W}}, \frac{\partial E}{\partial \mathbf{V}}, \frac{\partial E}{\partial \mathbf{U}}\right) \tag{2.50}$$

where $U$ is the hidden-to-hidden recurrent connections matrix, and

$$\frac{\partial E}{\partial \mathbf{U}} = \frac{\partial E}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{U}} \tag{2.51}$$

Then, $U$ weight matrix of the next iteration step is

$$\mathbf{U}(i+1) = \mathbf{U}(i) + \lambda \frac{\partial E}{\partial \mathbf{U}(i)}. \tag{2.52}$$

18

Up to here, batch gradient descent is explained. Namely, the gradient of the error function is computed for the whole training dataset. For one update calculating the gradients for the whole dataset can make the algorithm very slow [52]. Especially, for very large data set it is not preferable.

However, in stochastic gradient descent, one example is taken, and the feedforward step is applied for one epoch. After finding its gradients, the weights are updated by using these gradients. These steps are applied to all examples in the same epoch. Although it is much faster than the batch gradient descent, it causes fluctuations in the loss function. Hence, it cannot reach the minimum although it approaches to 0.

Mini-batch gradient descent is a mixture of batch and stochastic gradient descent. It splits data into specific mini-batches. In one epoch, it computes gradients of each mini-batch by taking the average of or summing up the gradients of the examples in the mini-batch. Then, the weights are updated for the related mini-batch. After applying all these steps to all mini-batches, the next epoch is started.

# CHAPTER 3

# EMPIRICAL APPROACH

This chapter contains the application methodologies of classical HMM and RNN for each of the univariate, bivariate, and trivariate cases in the frame of financial data. The selected assests are S&P 500, NASDAQ and EUR/USD exchange rate. The reasons for the choice of these are explained in Section 3.1. The methodology of the Hybrid HMM-RNN model is introduced. Since it would be more effective to explain the proposed model over the data set, first the data set is defined followed by the hybrid model.

## 3.1 Data and Descriptives

The Standards & Poor's 500 (S&P 500) stock market index measures the performance of 500 large companies listed on the US stock exchanges. It is a free float-adjusted market capitalization-weighted index. As the primary indicator of the overall stock market perfor-mance in the US, it is used to record and monitor daily changes in the largest companies in the American stock market. S&P 500 is considered as a very important global financial indicator since the United States is one of the biggest financial centers in the world [42]. National Association of Securities Dealers Automated Quotations (Nasdaq) Composite Index is also a US based stock market index. While S&P 500 covers different sectors like finance, health care, industry, energy, information technology, and many others, Nasdaq includes only the information technology sector. As a result, S&P 500 is risk free while Nasdaq is highly volatile. Since S&P 500 includes the stocks in Nasdaq Composite, the correlation between them is expected to be high which is also supported by the data set yielding 99.3% correla-tion (Table 3.1). The exchange rate EUR/USD is another critical indicator in international and global markets. It has indirect influence on the assets, especially during global crisis which fits well to HMM influencing the hidden states. It is interesting to observe that the correlation of the EUR/USD rate has low correlation to other two selected assets (Table 3.1). The European single market and the US market are world's two biggest economies. Therefore, EUR/USD is the most traded currency pair on the market and this parity is a crucial parameter in the US economy.

S&P 500 and Nasdaq data used in the application are presented in Figure 3.1a. It includes

Table 3.1: Correlation Matrix of Nasdaq, S&P 500 and EUR / USD data

|        | Nasdaq | S&P 500 | EUR/USD |
|--------|--------|---------|---------|
| Nasdaq | 1      | 0.973   | 0.044   |
| S&P 500| 0.973  | 1       | 0.046   |
| EUR/USD| 0.044  | 0.046   | 1       |
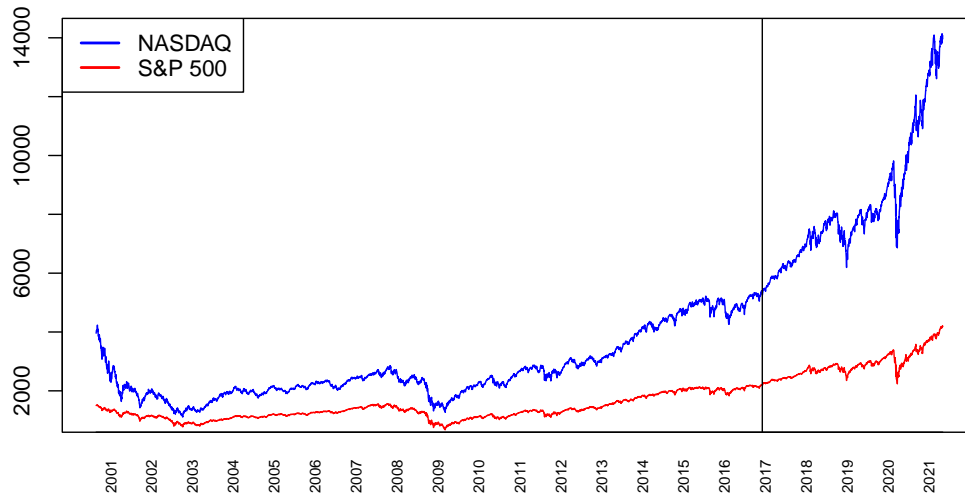


(a) Daily closing prices of S&P 500 and Nasdaq indexes
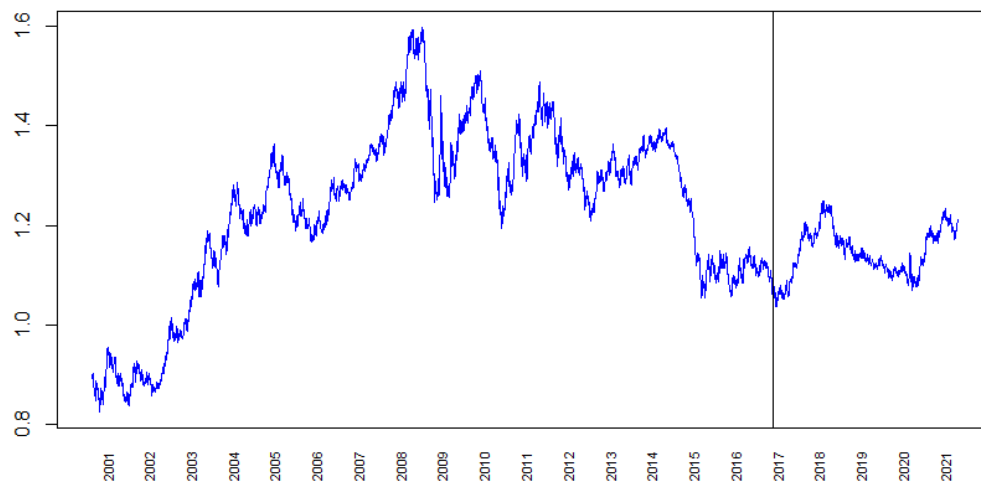


(b) Daily EUR/USD exchange rates

Figure 3.1: Time series of the variables

daily closing prices of S&P 500 and Nasdaq data. Figure 3.1b shows EUR/USD exchange rate index. They are retrieved from the database of Yahoo Finance for the time period between 22.08.2000 and 30.04.2021, yielding 5205-time points. The partition of training and test sets are made approximately 80%-20% in line with literature. While 4096 of them are selected as the training part, 1108 are used as the test part. The cut off lines in Figure 3.1a and Figure 3.1b correspond to the date of 02.12.2016 and separate the training and the test parts. In fact, 10% of the test data is used for validation and 10% for testing.

As can be seen from Figure 3.1a, there has been an upward trend in two stock prices since 2009 after the 2008 financial crisis. Large decreases are observed in 2019 and 2020. The first drop mentioned is due to a global economic slowdown, disruptive trade wars. The declines in 2020 are related to the effect of the pandemic on the economy. Figure 3.1b illustrates that there is no obvious trend in the exchange rate data. Also, it should be noted that test data differs from training data and follows a more stable pattern.

The variables are transformed by using Min-Max normalization technique and used in the applications in this form since it converts all variables into the same scale $(0 - 1)$ while preserving their behavior over time. Figure 3.2a shows the transformed data for the training part of the variables whereas Figure 3.2b represents the whole transformed sets. According to Figure 3.2b, while S&P 500 and Nasdaq follow a similar pattern from 2009, EUR/USD starts to follow a different pattern near the end of training data since it becomes more stable in test period.

Table 3.2 shows the descriptive statistics of the original data set and the training- test parts. For both of the cases Nasdaq and S&P 500 training data are positively skewed and exchange rate training data is negatively skewed. While the test part of Nasdaq and S&P 500 show similar features with the training part, this is not the case for the exchange rate data in terms of skewness and kurtosis. Moreover, Figure 3.3a, Figure 3.3c and Figure 3.3e represent the histogram of S&P 500, Nasdaq and exchange rate, respectively, which do not follow normal distribution, justified also by QQ plots and Shapiro-Wilk test (Figure 3.3b, Figure 3.3d and Figure 3.3f; Table 3.2). The financial data analysis assumes mostly normality as the log-returns in theoretical studies are taken as log-normal. For this reason, we stay with the normality for the rest of the analyses.

The bivariate histograms of the binary combinations of S&P 500 and Nasdaq indeces and exchange rate are represented in Figure 3.4 since they guide for the choice of the initial parameters of the bivariate and trivariate HMMs. State centers are tried to be selected according to the concentration regions.

## 3.2 Application of the Proposed Methodology

In this section, we illustrate the proposed approach on the selected variables. To elaborate the impact of hybrid model, i.e. RNN-HMM, the traditional models are applied firstly. The
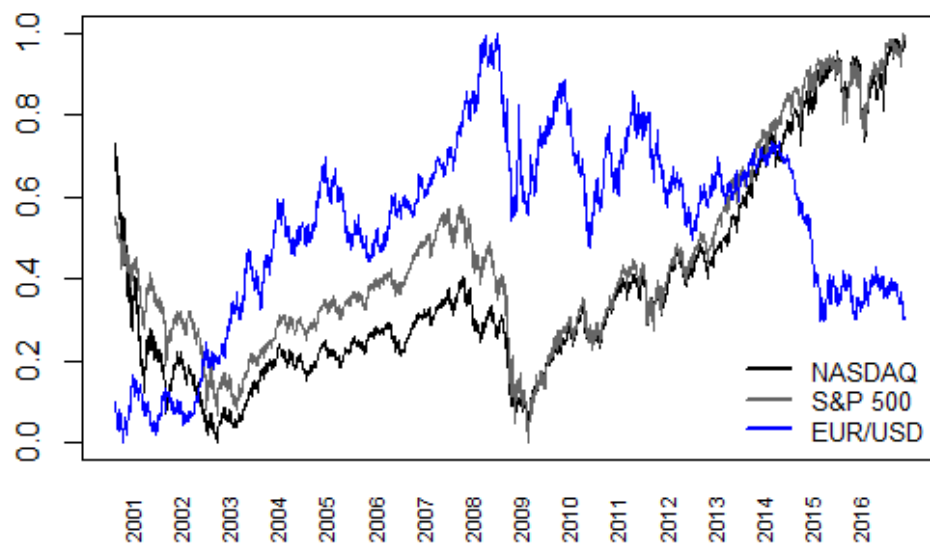
(a) For training set
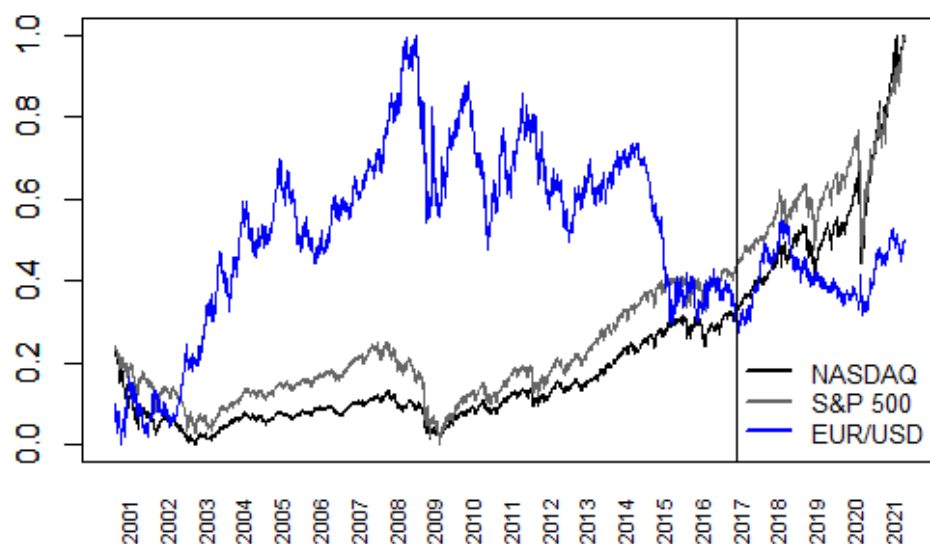


(b) For whole time frame

Figure 3.2: Min-Max transformed variables
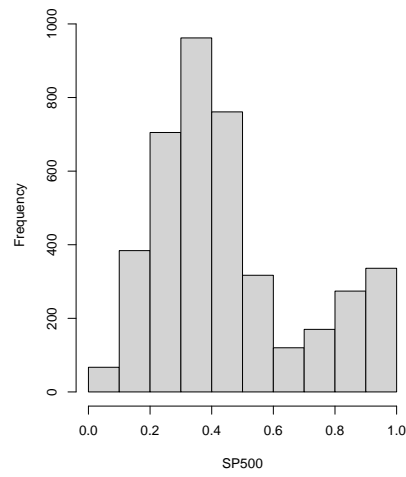
(a)
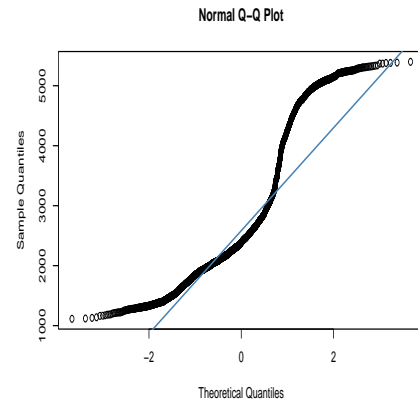
(b)
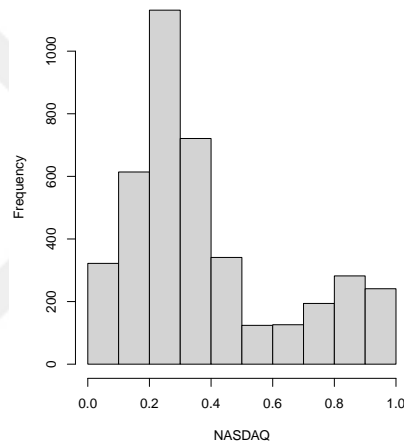
(c)
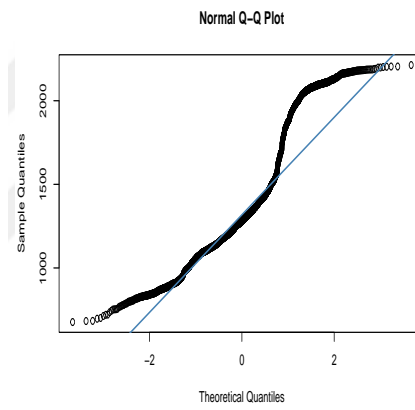
(d)
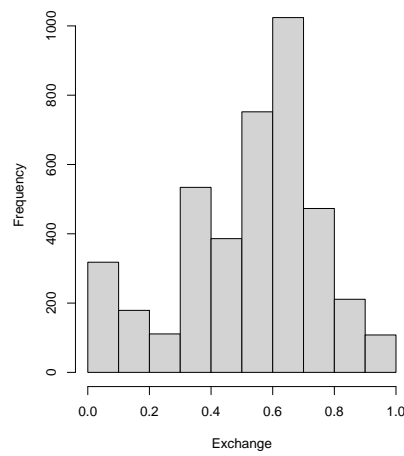
(e)
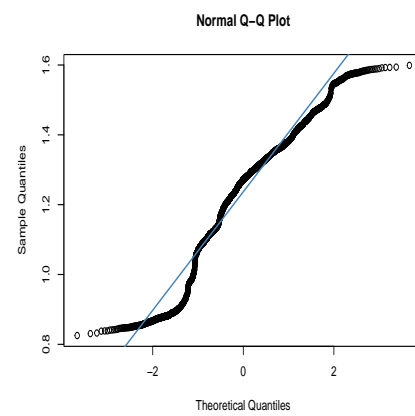
(f)

Figure 3.3: Histogram and QQ-plots of training sets

(a)



(b)



(c)
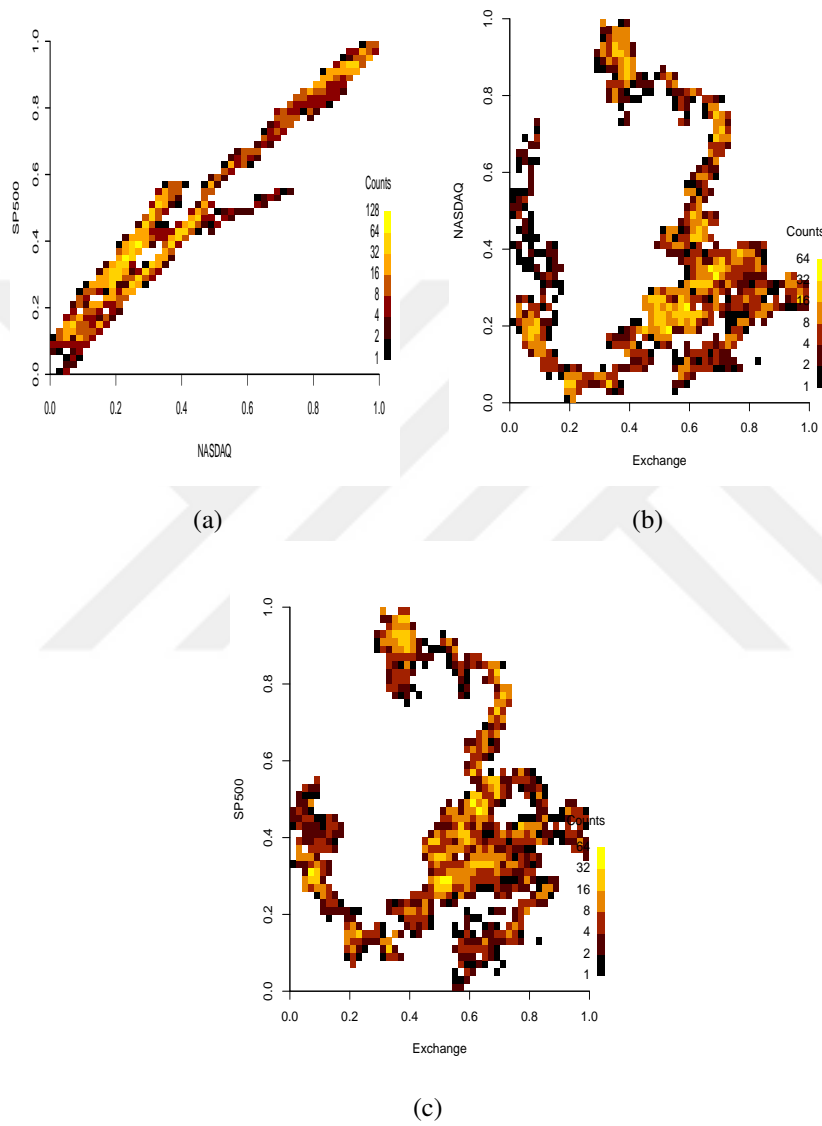
Figure 3.4: Bivariate histograms of S&P 500 and Nasdaq indexes and Exchange Rate data

Table 3.2: Descriptive statistics of daily S&P 500 and Nasdaq close prices and EUR/USD exchange rates ($n_A = 5205$, $n_{Tr} = 4096$, $n_T = 1108$)

| Variable | n | Mean | Std. Dev. | Min | Max | Skewness | Kurtosis | Shapiro-Wilk (p-value) |
|---|---|---|---|---|---|---|---|---|
| Nasdaq | $n_A$ | 3936.41 | 2673.81 | 1114.11 | 14138.78 | 1.57 | 5.15 | 0.00 |
| | $n_{Tr}$ | 2749.84 | 1090.01 | 1114.11 | 5398.92 | 0.95 | 2.77 | 0.00 |
| | $n_T$ | 8316.12 | 2182.48 | 5251.11 | 14138.78 | 1.12 | 0.38 | 0.00 |
| S&P 500 | $n_A$ | 1696.18 | 735.42 | 676.53 | 4211.47 | 1.11 | 3.42 | 0.00 |
| | $n_{Tr}$ | 1369.27 | 364.02 | 676.53 | 2213.35 | 0.77 | 2.64 | 0.00 |
| | $n_T$ | 2902.93 | 443.40 | 2191.08 | 4211.47 | 0.89 | 0.31 | 0.00 |
| EUR/USD | $n_A$ | 1.21 | 0.16 | 0.83 | 1.60 | -0.27 | -0.16 | 0.00 |
| | $n_{Tr}$ | 1.23 | 0.17 | 0.83 | 1.60 | -0.55 | -0.28 | 0.00 |
| | $n_T$ | 1.15 | 0.05 | 1.15 | 1.25 | 0.02 | -0.78 | 0.00 |

analysis contain three dimensions in the variables: Univariate, Bivariate and Trivariate cases. They are applied for HMM, RNN and Hybrid (RNN-HMM) approaches. The performance of the forecasting results are summarized in terms of the RMSE, MAPE and MAE measures.

### 3.2.1 HMM Methodology for Univariate, Bivariate and Trivariate Cases

Firstly, for the application of univariate HMM, the initial parameters of mean $\mu$, standard deviation $\sigma$, transition probability, $\Gamma$, and initial distribution $\delta$ are determined by trying to get rid of the local maximums. This step is applied to several univariate HMMs with different number of states to choose the most suitable one with respect to their AIC and BIC values.

At each step of EM algorithm $logL$ is computed by using the formula in Equation (2.10). Since there are $(m^2 - m)$ parameters for $\Gamma$, $m$ parameters for each of the $\mu$ and $\sigma$ variables and $m - 1$ parameters for $\delta$, $p$ can be written as $p = m^2 + 2m - 1$ where $m$ is the state number.

In the bivariate case, the mean, $\mu_1$, and standard deviation, $\sigma_1$, of the first variable and the mean, $\mu_2$, and standard deviation, $\sigma_2$, of the second variable are determined. Moreover, transition probability $\Gamma$ and initial distribution $\delta$ are initialized as in the univariate HMM application. Similarly, state number is determined according to AIC and BIC results. However, notice that the number of parameters is $p = m^2 + 4m - 1$ for the bivariate case and $p = m^2 + 6m - 1$ for the trivariate case. Covariance matrices are not included because of the conditional independence assumption. The effect of covariance is considered to be handled under hidden states, due to the assumption that the variables are bound to the same states. For each of the univariate, bivariate, and trivariate HMM applications, after performing the EM algorithm, the parameters of the models are estimated, and forecasting is achieved by using Equation (2.13).
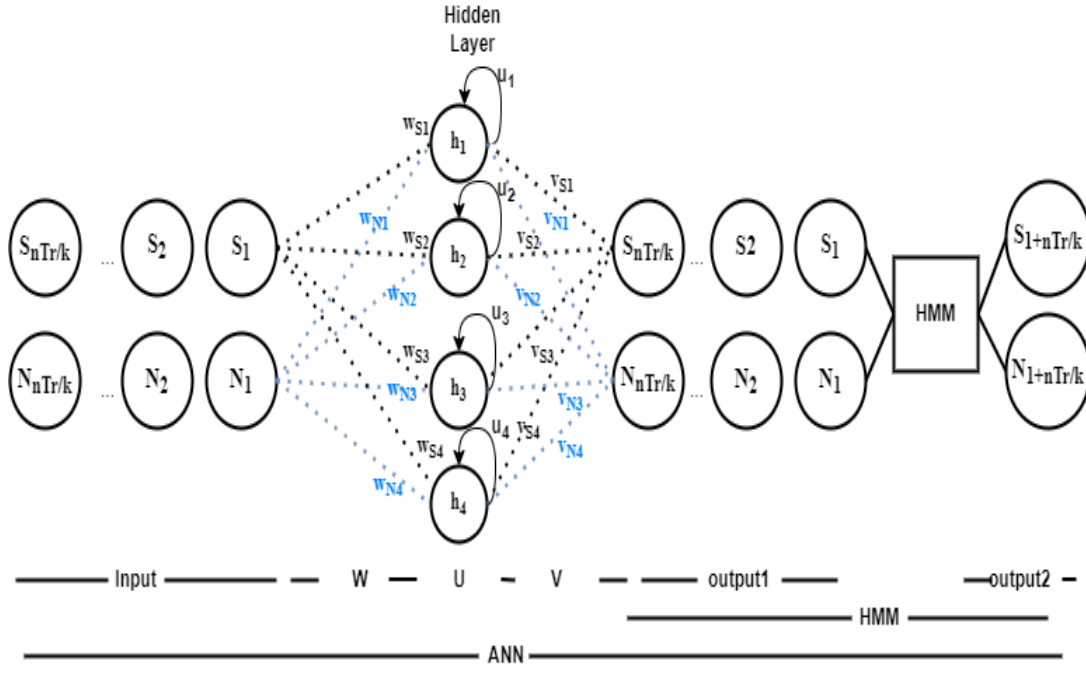
Figure 3.5: Structure of the Hybrid RNN-HMM

### 3.2.2 RNN-HMM Methodology

A hybrid model constructed in this thesis implements RNN and HMM algorithms sequentially and uses EM and discriminative algorithms of RNN to optimize the whole model. The proposed model is illustrated in Figure 3.5 whose details are given in Algorithm 1 and the flowchart in Figure 3.6. This algorithm can be extended for larger number of variables. Here, we illustrate the bivariate case for simplicity.

To elaborate more, we describe the process given in Figure 3.5 as follows. Data is divided into training and test sets as a requirement for RNNs. The training part $(n_{Tr})$ is then divided $k$ equal subsamples with size $n_{Tr}/k$. After that, initial weights are simultaneously given to each subsample in order to make forward propagation. The outputs of forward propagation are used as the input of HMM part. After applying the HMM part, final outputs of the first round are produced. Then, the loss function is calculated by taking the average of MSE results based on the actual values of the data at time $\frac{n_{Tr}}{k} + 1$. According to gradient results by using Back Propagation the weights are recalculated for use in the next step. All these steps are reapplied until reaching the tolerance criteria of the loss function.

28

---
**Algorithm 1** Algorithm of Hybrid RNN-HMM.

---

**Data:** Read $S$ and $N$ ($n_{\text{Tr}}$ data points for each variable)

**Output:** Forecasting results

  Divide data into $k$ equal parts for attaining $k$ subsamples

  Randomize initial weights and propose the learning rate of RNN

  Propose the learning rate and iteration number of RNN

  Set the forms of initial parameters of HMM

  Apply feedforward part of RNN to each subsample

  Use outputs of feedforward part as the input of HMM and apply EM algorithm

    E-step: Calculate likelihood

   M-step: Obtain the parameters that maximizes the likelihood

   **If** The convergence condition is not satisfied

   **Then** Return to E-step

   **Else** Obtain the optimal estimated parameters

   **End**

  Forecasting

  Compute the loss function and the gradient descent of the weights

  Update the weights of RNN part

   **If** The iteration number is not reached

   **Then** Return to feedforward application part

   **Else** Obtain the weights

   **End**

  Make validation

   **If** Validation does not reach to the minimum loss

   **Then** Restructure RNN

   **Else** Make final forecasting by using ultimate HMM and RNN parameters

   **End**

---

Nasdaq, S&P 500 and EUR/USD exchange observations starting at time 1 till $t$ are accepted as input depending on the number of variables of incorporated into RNN. Training set of 4096 is separated into 4 subsamples whose size is chosen to be 1024 attain to be the power of 2. There could be problems like overfitting and not being able to determine the appropriate number of states for the HMM part for smaller subsamples may occur.

The input-to-hidden, hidden-to-hidden and the hidden-to-output weights in RNN part are named as $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$, respectively. According to loss function results of the validation part 4-node case is preferred for the hidden layer of the RNN part. Therefore, for example, for the bivariate case the input data, $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$ are constructed as $2 \times 1024$, $2 \times 4$, $4 \times 4$ and $4 \times 2$ matrices, respectively. The weights $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$ are shown as

$$\mathbf{W} = \begin{bmatrix} w_{S1} & w_{S2} & w_{S3} & w_{S4} \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} \end{bmatrix}, \qquad (3.1a)$$
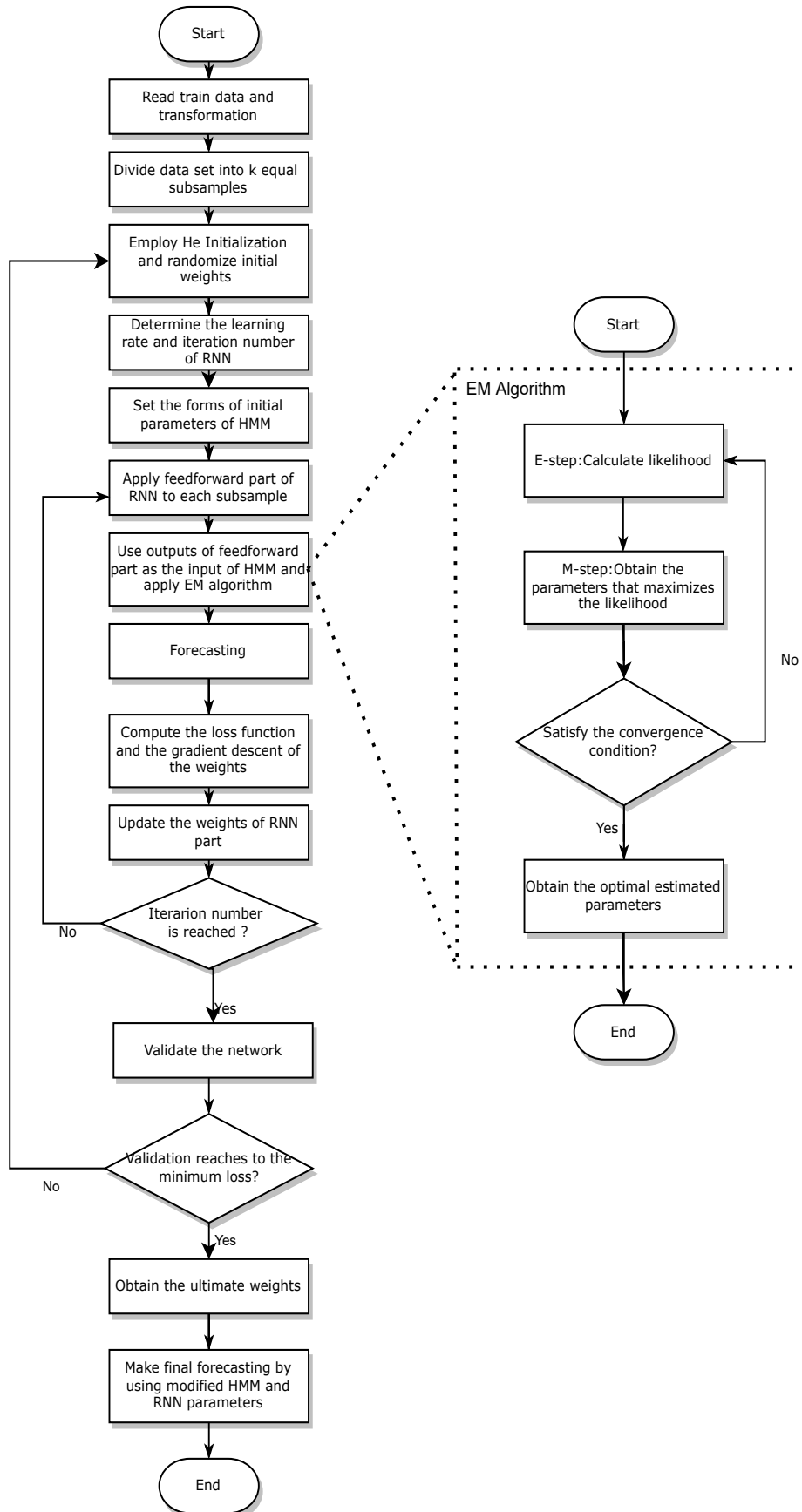
Figure 3.6: Flowchart of Hybrid RNN-HMM

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{21} & u_{31} & u_{41} \\ u_{12} & u_{22} & u_{32} & u_{42} \\ u_{13} & u_{23} & u_{33} & u_{43} \\ u_{14} & u_{24} & u_{34} & u_{44} \end{bmatrix}, \tag{3.1b}$$

$$\mathbf{V} = \begin{bmatrix} v_{S1} & v_{N1} \\ v_{S2} & v_{N2} \\ v_{S3} & v_{N3} \\ v_{S4} & v_{N4} \end{bmatrix}. \tag{3.1c}$$

This can be easily applied when the number of variables increase by simply adjusting the matrices accordingly. To illustrate, the input data, $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$ are constructed as $1 \times 1024$, $1 \times 4$, $4 \times 4$ and $4 \times 1$ matrices for the univariate case and $3 \times 1024$, $3 \times 4$, $4 \times 4$ and $4 \times 3$ matrices for the trivariate cases, respectively.

The activation function used in this model is leakyRelu ($LR$). Therefore, He initialization is preferred since it is the most suitable weight initialization technique for this activation function currently.

Then, for the first time step the hidden layer ($L^{(1)}$) is computed as

$$L^{(1)} = \mathrm{LR}\left( \begin{bmatrix} S_1 & N_1 \\ \vdots & \vdots \\ S_{1024} & N_{1024} \end{bmatrix} \begin{bmatrix} w_{S1} & w_{S2} & w_{S3} & w_{S4} \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} \end{bmatrix} \right) \tag{3.2}$$

and, for the $t^{th}$ time step the hidden layer ($L^{(t)}$, $t = 2, \ldots$) is as follows

$$\begin{aligned} L^{(t)} = \mathrm{LR}\left( \begin{bmatrix} S_1 & N_1 \\ \vdots & \vdots \\ S_{1024} & N_{1024} \end{bmatrix} \begin{bmatrix} w_{S1} & w_{S2} & w_{S3} & w_{S4} \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} \end{bmatrix} \right. \\ \left. + L^{(t-1)} \begin{bmatrix} u_{11} & u_{21} & u_{31} & u_{41} \\ u_{12} & u_{22} & u_{32} & u_{42} \\ u_{13} & u_{23} & u_{33} & u_{43} \\ u_{14} & u_{24} & u_{34} & u_{44} \end{bmatrix} \right). \end{aligned} \tag{3.3}$$

The output matrix of the RNN part $\mathbf{C}$ is calculated by using $\mathbf{V}$ presented in Equation (3.4) and is obtained as a $2 \times 1024$ matrix.

$$\mathbf{C}^{(t)} = \mathrm{LR}\left( \mathbf{L}^{(t)} \times \begin{bmatrix} v_{S1} & v_{N1} \\ v_{S2} & v_{N2} \\ v_{S3} & v_{N3} \\ v_{S4} & v_{N4} \end{bmatrix} \right) \tag{3.4}$$

31

Five-state is used for the HMM part and it is fed by $\mathbf{C}$, whose output matrix of this last part is named as $\mathbf{O}$. These two output matrices are formulated explicitly in Equations (3.5) and (3.6), respectively, where $H$ is the function representation of HMM part.

$$
\begin{aligned}
\mathbf{C}^{(1)} &= \text{LR}\left(\text{LR}\left(\mathbf{Z} \times \mathbf{W}\right) \times \mathbf{V}\right) \\
&= \text{LR}\left(\mathbf{L}^{(1)} \times \mathbf{V}\right)
\end{aligned}
\tag{3.5a}
$$

$$
\begin{aligned}
\mathbf{O}^{(1)} &= \text{H}\left(\text{LR}\left(\text{LR}\left(\mathbf{Z} \times \mathbf{W}\right) \times \mathbf{V}\right)\right) \\
&= \text{H}\left(\text{LR}\left(\mathbf{L}^{(1)} \times \mathbf{V}\right)\right) \\
&= \text{H}\left(\mathbf{C}^{(1)}\right)
\end{aligned}
\tag{3.5b}
$$

$$
\begin{aligned}
\mathbf{C}^{(t)} &= \text{LR}\left(\text{LR}\left(\mathbf{Z} \times \mathbf{W}\right) \times \mathbf{V} + L^{(t-1)}\mathbf{U}\right) \\
&= \text{LR}\left(\mathbf{L}^{(t)} \times \mathbf{V}\right)
\end{aligned}
\tag{3.6a}
$$

$$
\begin{aligned}
\mathbf{O}^{(t)} &= \text{H}\left(\text{LR}\left(\text{LR}\left(\mathbf{Z} \times \mathbf{W}\right) \times \mathbf{V} + \mathbf{L}^{(t-1)}\mathbf{U}\right)\right) \\
&= \text{H}\left(\text{LR}\left(\mathbf{L}^{(t)} \times \mathbf{V}\right)\right) \\
&= \text{H}\left(\mathbf{C}^{(t)}\right)
\end{aligned}
\tag{3.6b}
$$

Then the loss function $E$ is

$$
E = \frac{1}{m}\sum_{k=1}^{m}\left(o_k - y_k\right)^2
\tag{3.7}
$$

where $y_k$'s are the entries in the ground truth label $\vec{y}$ and $o_k$'s are the entries in the prediction matrix $\mathbf{O}$.

In order to recalculate the weights, we take the partial derivatives of loss function with respect to $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$. Before representing the derivatives following terms are needed to be defined. These are

$$
\theta^{(t)} = \mathbf{Z} \times \mathbf{W} + \mathbf{L}^{(t-1)} \times \mathbf{U}
\tag{3.8a}
$$

$$
\mathbf{Q}^{(t)} = \left(\text{LR}\left(\mathbf{Z} \times \mathbf{W} + \mathbf{L}^{(t-1)} \times \mathbf{U}\right)\right) \times \mathbf{V}
\tag{3.8b}
$$

Then, the derivative with respect to $\mathbf{W}$ can be written as follows:

$$
\frac{\partial E}{\partial \mathbf{W}} = \frac{\partial E}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \mathbf{W}}.
\tag{3.9}
$$

Based on Equation (3.9), we propose

$$\frac{\partial E}{\partial \mathbf{L}} = \frac{\partial E}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{L}} \tag{3.10}$$

Finally, the partial derivative of loss function according to $\mathbf{W}$ becomes

$$\frac{\partial E}{\partial \mathbf{W}} = \frac{\partial E}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{W}}. \tag{3.11}$$

The partial derivative of loss function according to $\mathbf{U}$ is

$$\frac{\partial E}{\partial \mathbf{U}} = \frac{\partial E}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \mathbf{Q}} \frac{\partial \mathbf{Q}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{U}}. \tag{3.12}$$

The derivative of loss function with respect to $\mathbf{V}$ is written as follows

$$\frac{\partial E}{\partial \mathbf{V}} = \frac{\partial E}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \theta} \frac{\partial \theta}{\partial \mathbf{V}} \tag{3.13}$$

When the matrix derivation is taken into account, the derivatives are fixed as follows

$$\frac{\partial E}{\partial \mathbf{W}} = Z^T \times \left[ \left[ \left( 2 \left( y - \mathbf{O}^{(t)} \right) H' \left( C^{(t)} \right) \mathrm{LR}' \left( C^{(t)} \right) \right) \right. \right. \\ \left. \left. \times \mathbf{V}^T \right] \mathrm{LR}' \left( L^{(t)} \right) \right] \tag{3.14}$$

$$\frac{\partial E}{\partial \mathbf{U}} = \left( L^{(t-1)} \right)^T \times \left[ \left[ \left( 2 \left( y - \mathbf{O}^{(t)} \right) H' \left( \mathbf{C}^{(t)} \right) \mathrm{LR}' \left( \mathbf{C}^{(t)} \right) \right) \right. \right. \\ \left. \left. \times \mathbf{V}^T \right] \mathrm{LR}' \left( L^{(t)} \right) \right] \tag{3.15}$$

$$\frac{\partial E}{\partial \mathbf{V}} = \left( L^{(t)} \right)^T \times \left[ 2 \left( y - \mathbf{O}^{(t)} \right) H' \left( \mathbf{O}^{(t)} \right) \mathrm{LR}' \left( \mathbf{C}^{(t)} \right) \right] \tag{3.16}$$

Since the HMM part is not feasible to take the derivative we need to use numerical derivative for $H'$ in above equations. It can be recognized that when all the input values increase at the same amount of $h$, the estimated means are expected to increase at this amount while the other parameters remain the same. Hence, the forecasting values are expected to increase about $h$. Therefore, in order to improve the time performance of the application derivative of HMM part is accepted as $1$.

After finding the derivatives we recalculate $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$ as follows:

$$\mathbf{W}(i+1) = \mathbf{W}(i) + \lambda \frac{\partial E}{\partial \mathbf{W}(i)} \tag{3.17}$$

$$\mathbf{U}(i+1) = \mathbf{U}(i) + \lambda \frac{\partial E}{\partial \mathbf{U}(i)} \tag{3.18}$$

33

$$\mathbf{V}(i+1) = \mathbf{V}(i) + \lambda \frac{\partial E}{\partial \mathbf{V}(i)} \tag{3.19}$$

where $i$ is the iteration number and $\lambda$ is the learning rate.

Finally, the estimated weight values of RNN part and the parameters of HMM part attained from the training application are fixed. The forecasting is performed by using these fixed parameters while the input data is updated at each time point.

### 3.2.3 Basic RNN Methodology

Besides the basic HMM and Hybrid RNN-HMM methods, RNN is applied to training set to be able to make a comparison with the proposed method. Here, a basic RNN is used since the combined RNN of RNN-HMM is also a basic version of RNN. In classical RNN applications, one hidden layer is used and the loss function is chosen as MSE. Four subsamples with 1024 batch size are used and for each sample, the gradient is computed. The weights are updated by using their sum at the backpropagation step as in the Hybrid model to be comparable. LeakyRelu function is used as activation function. He initialization technique is used while initial weights are given. The iteration number and the learning rate are determined by controlling the RMSE on validation data. After determining the ultimate weights, forecasting is performed.

All the models mentioned in this section are applied on the training data and their results and comparisons are presented in the next chapter.

# CHAPTER 4

# APPLICATION RESULTS

In this chapter, the forecasting results of the Hybrid model are explained for each of the univariate, bivariate and trivariate cases. Moreover, basic HMMs and RNNs are applied, and the results are introduced to be able to evaluate the performance of the developed model relative to the classical models. All results in this section are one ahead forecast results. In other words, after the optimal parameters are fixed for each model, the input data is updated at each step and estimations are made.

## 4.1 Univariate Analyses

### 4.1.1 S&P 500 Analyses

The S&P 500 closing prices from 22.08.2000 to 30.04.2021 is presented once again in Figure 4.1. The first crisis of this period is the bubble in the stock market in 2000 (Dot-com bubble). Overvaluations, public enthusiasm for stocks, and speculation in the technology sector are the main reasons of this crisis. The S&P 500 is damaged when the bubble burst between 2000 and 2002. Moreover, during 2008 financial crisis and its recession, the S&P 500 dropped over 40% from late 2007 to early 2009, but recovered all its losses by 2013. The growth trend after this crisis is interrupted by the pandemic conditions that entered our lives at the beginning of 2020 and cause a sharp decline. The coronavirus pandemic causes a global recession and disrupts stock markets, including the S&P 500. However, it recovers in the second half of 2020 and hits all-time highs in 2021.

#### 4.1.1.1 HMM on S&P 500

The univariate HMM applied to S&P 500 requires the initial parameters to be determined by eliminating the local maximums with control of the AIC and BIC values. For three different state numbers, this model is compared and presented in Table 4.1. Since the high state numbers are not preferred in HMMs, it is set up to 5-state. In addition, with the higher number of variables, the number of parameters will increase and this will further increase the probability
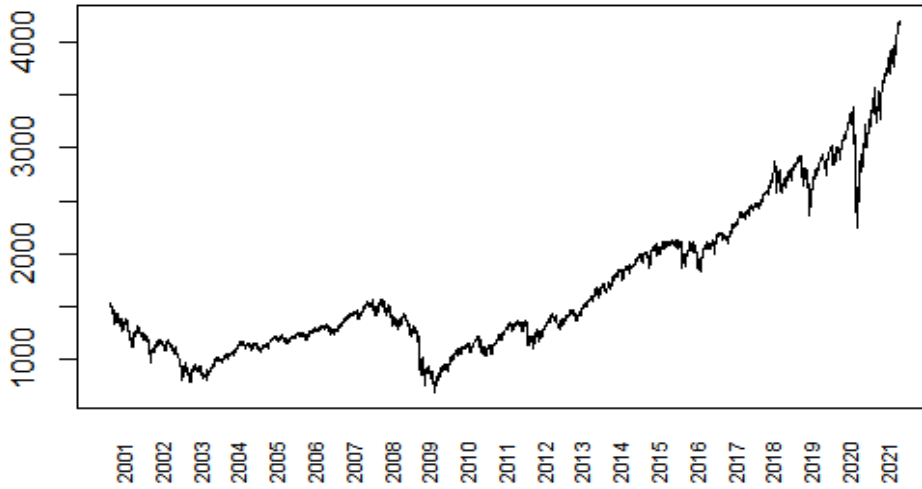
Figure 4.1: Daily closing prices of S&P 500

Table 4.1: Test statistics of 3-state, 4-state and 5-state univariate HMMs

|  | mllk | AIC | BIC |
|---|---|---|---|
| 3-state univariate HMM | -4666.964 | -9305.929 | -9217.48 |
| 4-state univariate HMM | -5732.058 | -11418.12 | -11272.81 |
| 5-state univariate HMM | -6856.77 | -13645.54 | -13430.74 |

of getting stuck at local minimums when setting the initial parameters. Based on the smallest AIC and BIC values, the 5-state model is chosen which is used for all HMM implementation.

Following a five-state specification, the choice of initial parameters is made by trial-and error using the histogram of the data (Figure 3.3a). Five normal distributions, whose means and standard deviations are given in initial parameter vectors $\mu$ and $\sigma$, are predicted to generate all S&P 500 data. After applying the EM algorithm to the initial mean and standard deviation predictions, AIC and BIC values are compared. The ones with lower criteria are chosen as the initial parameters of the HMM. The estimated parameters $\mu$, $\sigma$, $\delta$ and $\Gamma$ are derived from the initial ones by EM algorithm and they are presented with AIC, BIC and negative likelihood (mllk) results for fivestate univariate HMM as:

$$\mu = \begin{pmatrix} 0.1560 & 0.2993 & 0.4007 & 0.5258 & 0.8660 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} 0.0501 & 0.0310 & 0.0285 & 0.0607 & 0.0749 \end{pmatrix}$$

36

$$\Gamma = \begin{pmatrix} 0.9916 & 0.0056 & 0.0028 & 0 & 0 \\ 0.0069 & 0.9885 & 0 & 0.0046 & 0 \\ 0.0050 & 0 & 0.9933 & 0.0017 & 0 \\ 0 & 0.0053 & 0 & 0.9934 & 0.001 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$mllk = -6856.77, AIC = -13645.54, BIC = -13430.74$$

The transition probability matrix results show that the probability of being in the same state is very close to 1, which is expected to be due to the long time series. However, these high probability ratios prevent HMM from working actively, which behaves differently and is not close to 1 in the proposed approach. This is one of the strengths of the proposed model. Since it works by breaking up long data, it allows transitions between states even for long data.

Equation (2.13) is used to forecast the S&P 500 with the estimated parameters and the result is presented in Figure 4.2. The figure shows that it is in agreement with the trend and captures the sharp fall after 2020. Moreover, RMSE, MAPE and MAE results of this application are presented in Table 4.3. The results are compared with the estimation results obtained from the hybrid model application in the next section.

### 4.1.1.2 RNN-HMM on S&P 500

After the application of univariate HMM to S&P 500, the RNN-HMM model is performed. The weight matrices $\mathbf{W}$, $\mathbf{U}$ and $\mathbf{V}$ are initialized randomly using He normal initialization technique. In the HMM part, the structure of the output matrix $\mathbf{C}$ is analyzed to construct the parameters of the EM algorithm. For the proposed hybrid model, the mean and standard deviation parameters of the EM algorithm are constructed as functions, although they are numeric in the classical HMM. The reason is the input data changes at each iteration. As a result, the initial parameters $\mu$ and $\sigma$ of EM for the univariate case are chosen by trial and error as follows:

$$\mu = \begin{pmatrix} \mu(\mathbf{C}) - 2a_1 & \mu(\mathbf{C}) - a_1 & \mu(\mathbf{C}) & \mu(\mathbf{C}) + a_1 & \mu(\mathbf{C}) + 2a_1 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} \sigma(\mathbf{C}) - \sigma(\mathbf{C})/10 & \sigma(\mathbf{C}) - \sigma(\mathbf{C})/10 & \sigma(\mathbf{C}) & \sigma(\mathbf{C}) - \sigma(\mathbf{C})/9 & \sigma(\mathbf{C}) - \sigma(\mathbf{C})/8 \end{pmatrix}$$

where $\mathbf{C}$ is S&P 500 output of RNN and $a_1$ is given as

Table 4.2: Comparison of univariate HMMs with different initial mean and standard deviation values

| Initial Values | | | | |
|---|---|---|---|---|
| I | $\mu = (\mu(\mathbf{C}) - a_1 \ \ \mu(\mathbf{C}) \ \ \mu(\mathbf{C}) + a_1 \ \ \mu(\mathbf{C}) + 2a_1 \ \ \mu(\mathbf{C}) + 3a_1)$ <br> $\sigma = (\sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{10} \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{10} \ \ \sigma(\mathbf{C}) \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{9} \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{8})$ | | | | |
| | | mllk | AIC | BIC | |
| | 1st subsample | -2398.727 | -4729.454 | -4561.784 | |
| | 2nd subsample | -2607.466 | -5146.933 | -4979.263 | |
| | 3rd subsample | -2283.173 | -4498.346 | -4330.676 | |
| | 4th subsample | -2169.873 | -4271.745 | -4104.075 | |
| II | $\mu = (\mu(\mathbf{C}) - 2a_1 \ \ \mu(\mathbf{C}) - a_1 \ \ \mu(\mathbf{C}) \ \ \mu(\mathbf{C}) + a_1 \ \ \mu(\mathbf{C}) + 2a_1)$ <br> $\sigma = (\frac{\sigma(\mathbf{C})}{10} \ \ \frac{\sigma(\mathbf{C})}{10} \ \ \sigma(\mathbf{C}) \ \ \frac{\sigma(\mathbf{C})}{9} \ \ \frac{\sigma(\mathbf{C})}{8})$ | | | | |
| | | mllk | AIC | BIC | |
| | 1st subsample | -2405.307 | -4742.614 | -4574.943 | |
| | 2nd subsample | -2591.693 | -5115.386 | -4947.716 | |
| | 3rd subsample | -2250.923 | -4433.846 | -4266.176 | |
| | 4th subsample | -2137.562 | -4207.125 | -4039.455 | |
| III | $\mu = (\mu(\mathbf{C}) - 2a_1 \ \ \mu(\mathbf{C}) - a_1 \ \ \mu(\mathbf{C}) \ \ \mu(\mathbf{C}) + a_1 \ \ \mu(\mathbf{C}) + 2a_1)$ <br> $\sigma = (\sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{10} \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{10} \ \ \sigma(\mathbf{C}) \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{9} \ \ \sigma(\mathbf{C}) - \frac{\sigma(\mathbf{C})}{8})$ | | | | |
| | | mllk | AIC | BIC | |
| | 1st subsample | -2494.845 | -4921.69 | -4754.02 | |
| | 2nd subsample | -2580.526 | -5093.052 | -4925.382 | |
| | 3rd subsample | -2299.975 | -4531.95 | -4364.28 | |
| | 4th subsample | -2232.479 | -4396.957 | -4229.287 | |

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{8}.$$

Three different $(I, II, III)$ trials are represented in Table 4.2 to illustrate the choices of initial criteria determined by trial and error. Different mean and standard deviation formulas are chosen for each trial, and mllk, AIC, and BIC values are reported for each subsample of each trial. The III. initial parameters in Table 4.2 give the lower AIC and BIC values in general.

The estimated parameters of EM algorithm are presented in Appendix A (Table A.1). Using these parameters of HMM and initial weights of RNN forecasting is performed and presented in Figure 4.2. The obtained results capture the fluctuations as well as catch the trend. Moreover, despite the sharp declines in 2019 and 2021, it also yields successful results.

In addition to HMM and hybrid model application, a simple RNN is also applied to make comparisons and clarify the success of proposed model. The results are shown in Figure 4.2 with HMM and RNN-HMM. Basic RNN is just able to catch the trend but cannot capture the fluctuations of data.
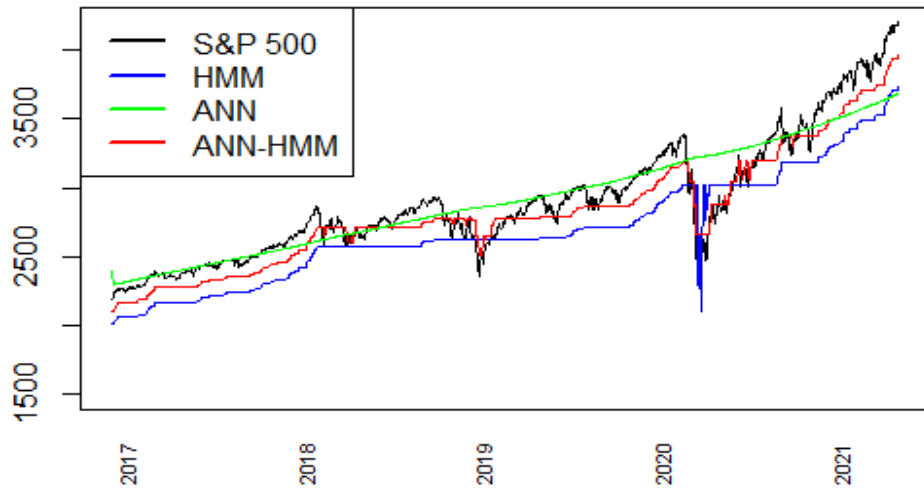
Figure 4.2: Forecasting results of S&P 500 for univariate HMM, RNN and RNN-HMM

Table 4.3: Univariate S&P 500 performance for each model

|  | RMSE | MAPE | MAE |
|---|---|---|---|
| HMM | 255.515 | 7.937 | 233.125 |
| RNN | 224.156 | 5.622 | 172.382 |
| RNN-HMM | **119.787** | **3.710** | **108.289** |

Figure 4.2 and Table 4.3 show that adding the learning algorithm of neural networks improves the forecasting performance of univariate basic HMM on S&P 500. Moreover, it gives much better results than the basic RNN. Among the univariate applications, the model with the best accuracy is the hybrid model with smallest RMSE (119.787) andd MAPE (3.710). In the second place is the HMM model, which performs better than the classical RNN application.

### 4.1.2 Nasdaq Analyses

Nasdaq prices which are reposted in Figure 4.3 also follow a very close path with the S&P 500. However, since the Dot-com bubble is actually a bubble about technology companies, it affects the Nasdaq significantly. As in the S&P 500, it has experienced sharp declines being affected by the 2008 recession and the 2020 pandemic conditions, but recent prices show increasing trend even faster than the S&P 500.

39
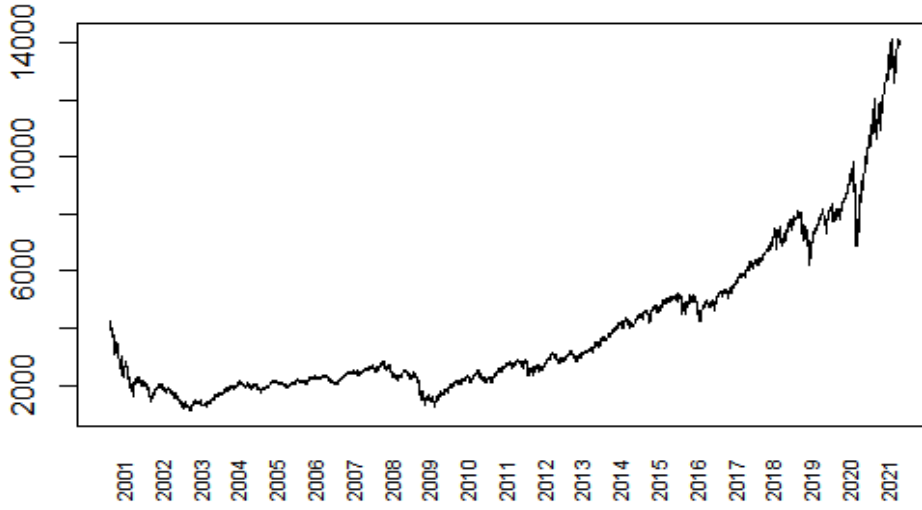
Figure 4.3: Daily closing prices of Nasdaq

#### 4.1.2.1 HMM on Nasdaq

Initial parameters are determined by comparing the AIC and BIC results of several trials for the basic univariate HMM application to Nasdaq as in the S&P 500 case. The histogram of Nasdaq in Figure 3.3c is used for this purpose and the parameters of the normally distributed five Nasdaq prices are found as:

$$\mu = \begin{pmatrix} 0.1012 & 0.2163 & 0.3070 & 0.4463 & 0.8248 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} 0.0429 & 0.0246 & 0.0345 & 0.0633 & 0.1002 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9906 & 0.0094 & 0 & 0 & 0 \\ 0.0061 & 0.9872 & 0 & 0.0066 & 0 \\ 0 & 0.0068 & 0.9878 & 0.0043 & 0 \\ 0 & 0 & 0.0062 & 0.9923 & 0.0015 \\ 0 & 0 & 0 & 0.0012 & 0.9988 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$mllk = -6713.109, AIC = -13358.22, BIC = -13143.41$$

40
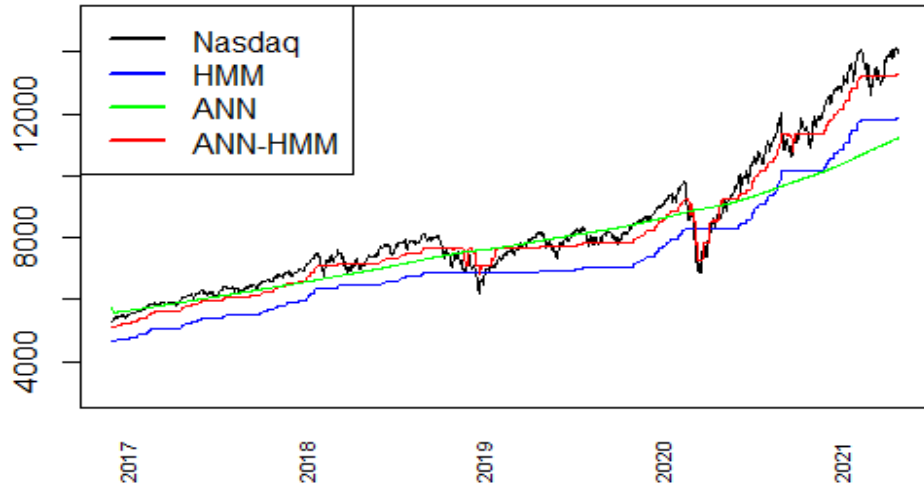
Figure 4.4: Forecasting results of Nasdaq for univariate HMM, RNN and RNN-HMM

Table 4.4:  Univariate Nasdaq performance for each model

|  | RMSE | MAPE | MAE |
|---|---|---|---|
| HMM | 1134.232 | 12.330 | 1038.582 |
| RNN | 1379.549 | 9.959 | 965.990 |
| RNN-HMM | **382.071** | **3.911** | **326.958** |

Forecasting result in Figure 4.4 shows that fluctuations could not be detected by HMM effectively. In order to examine its performance, RMSE, MAPE and MAE results are presented in Table 4.4.

#### 4.1.2.2   RNN-HMM on Nasdaq

Then, RNN-HMM Hybrid model is tested on Nasdaq for univariate case. The initial parameters of HMM part are determined by trial and error as given in Table 4.2 and they are as follows for the univariate Hybrid application to Nasdaq:

$$\mu = \Big( \mu(\mathbf{C}) - 2a_1 \quad \mu(\mathbf{C}) - a_1 \quad \mu(\mathbf{C}) \quad \mu(\mathbf{C}) + a_1 \quad \mu(\mathbf{C}) + 2a_1 \Big)$$

$$\sigma = \Big( \sigma(\mathbf{C}) - \sigma(\mathbf{C})/10 \quad \sigma(\mathbf{C}) - \sigma(\mathbf{C})/10 \quad \sigma(\mathbf{C}) \quad \sigma(\mathbf{C}) - \sigma(\mathbf{C})/9 \quad \sigma(\mathbf{C}) - \sigma(\mathbf{C})/8 \Big)$$

where $\mathbf{C}$ is Nasdaq output of RNN, and

41
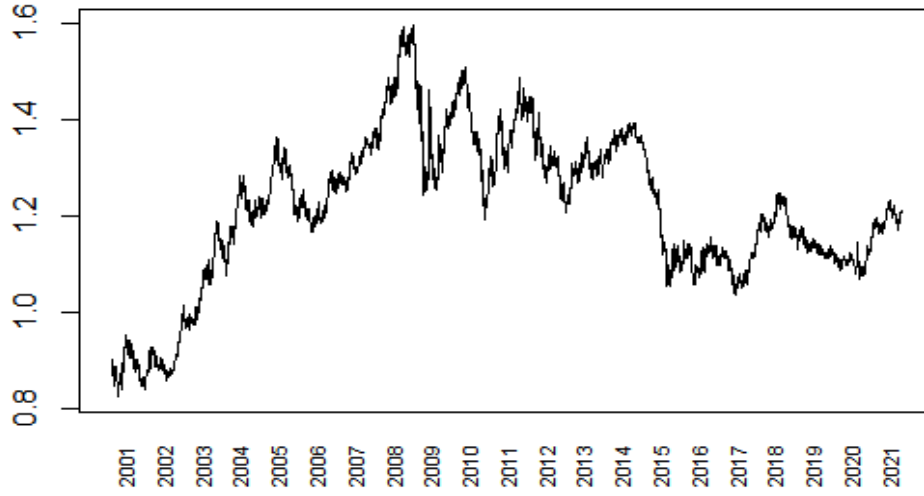
Figure 4.5: EUR/USD Exchange Rate

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{8}.$$

Figure 4.4 contains forecast results obtained using the optimized HMM parameters in Table A.2 and ultimate RNN parameters. Unlike the HMM implementation, this model is quite successful in capturing fluctuations on Nasdaq. On the other hand, RNN can only catch the trend. RMSE, MAPE and MAE results of HMM, RNN and the Hybrid applications (Table 4.4). These results reinforce that RNN-HMM gives the smallest errors (RMSE of 382.071 and MAPE 3.911).

### 4.1.3  EUR/USD Analyses

The EUR/USD exchange rates (reposted in Figure 4.5) between 22.08.2000 and 30.04.2021 does not have long history as Euro is launched in 1999 and used only in online applications for 3 years. It is started to be used as coins and banknotes in 2002. Due to the fact that Europe has a very large market, the Euro, the official currency of the European Union, has appreciated against the USD as well as against many currencies up to 2008. Due to the Greek economic crisis in 2008, the Euro experienced a sharp decline. The exchange rate follows a fluctuating course afterward. In 2014, one of the most significant decreases in the EUR/USD exchange rate takes place due to the strengthening of the US economy and the political instability in Greece.

#### 4.1.3.1   HMM on EUR/USD

After examining the estimation results of the univariate models applied to the S&P 500 and Nasdaq data, the models applied to the univariate EUR/USD data are shown in this section.

The histogram in Figure 3.3e is used for determining the initial parameters. Then, the optimized parameters are obtained by controlling the AIC and BIC results of EM algorithm as follows:

$$\mu = \begin{pmatrix} 0.0794 & 0.2030 & 0.3745 & 0.5749 & 0.7578 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} 0.0308 & 0.0302 & 0.0416 & 0.0555 & 0.0852 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9953 & 0.0047 & 0 & 0 & 0 \\ 0.0062 & 0.9876 & 0.0062 & 0 & 0 \\ 0 & 0 & 0.9973 & 0.0027 & 0 \\ 0 & 0 & 0.0013 & 0.9934 & 0.0053 \\ 0 & 0 & 0 & 0.0070 & 0.9930 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$mllk = -5974.19, AIC = -11880.27, BIC = -11665.57$$

Estimated parameters plugged in Equation (2.13) yield the forecasts of univariate HMM. Figure 4.6 includes this result and shows that forecasting values can follow actual values.

#### 4.1.3.2   RNN-HMM on EUR/USD

We obtain the initial parameter formulas of HMM by trial and error as follows

$$\mu = \begin{pmatrix} \mu(\mathbf{C}) - 3a_1 & \mu(\mathbf{C}) + a_1 & \mu(\mathbf{C}) + a_1 & \mu(\mathbf{C}) & \mu(\mathbf{C}) - a_1 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} \sigma(\mathbf{C})/4 & \sigma(\mathbf{C})/5 & \sigma(\mathbf{C})/4 & \sigma(\mathbf{C})/5 & \sigma(\mathbf{C})/5 \end{pmatrix}$$

where $\mathbf{C}$ is EUR/USD exchange rate output of RNN, and

43
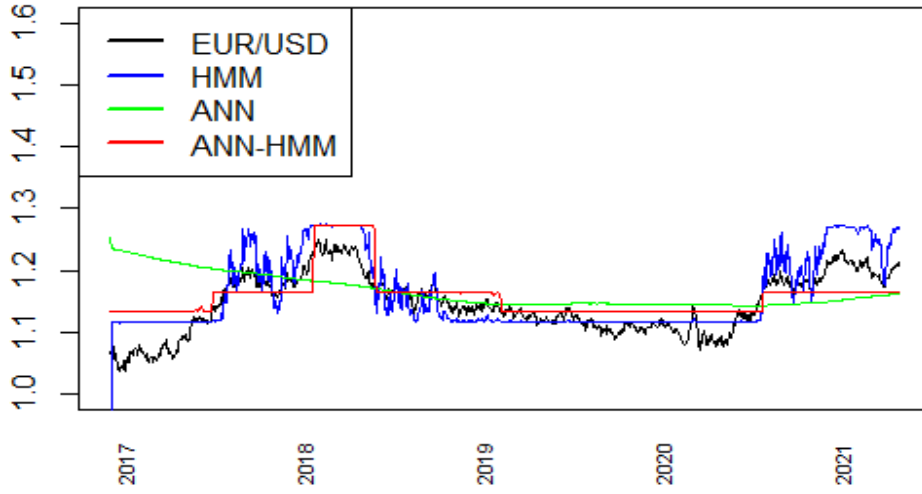
Figure 4.6: Forecasting results of EUR/USD for univariate HMM, RNN and RNN-HMM

Table 4.5: Univariate EUR/USD performance for each model

|  | RMSE | MAPE | MAE |
|---|---|---|---|
| HMM | 0.039 | **2.390** | **0.027** |
| RNN | 0.061 | 3.912 | 0.044 |
| RNN-HMM | **0.035** | 2.492 | 0.028 |

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{9}.$$

Forecasting is performed with the estimated parameters are presented in Table A.3. Although the test data of EUR/USD has a very different structure than its training data, the forecasting results are successful. However, the Hybrid model does not significantly contribute to the classical HMM. RMSE, MAPE, and MAE results in Table 4.5 depict that the simple HMM (RMSE of 0.039 and MAPE 2.390) and the proposed hybrid model (RMSE of 0.035 and MAPE 2.492) have similar accuracy and give better results than the basic RNN.

## 4.2 Bivariate Analyses

In this section, S&P 500, Nasdaq and EUR/USD variables are subjected to HMM, RNN and RNN-HMM applications in bivariate combinations.

44

### 4.2.1 Bivariate S&P 500 and Nasdaq Analyses

The bivariate case is applied first for the two highly correlated variables, S&P 500 and Nasdaq, for the classical and proposed Hybrid model.

### 4.2.1.1 HMM on S&P 500 and Nasdaq

The initial mean and standard deviation parameters of two variables are determined by examining the bivariate histogram in Figure 3.4a. According to the points where the data is concentrated, the mean and standard deviation initial parameters of the 5 states are determined. Then, the parameters are estimated by using EM algorithm and controlling the information criteria. The estimated mean, standard deviation, transition probability and initial distribution parameters and AIC, BIC, and mllk results of 5-state bivariate HMM application for S&P 500 and Nasdaq indices are selected. Because of the conditional independence assumption, covariance matrix is not among the estimated parameters in this part and in the multivariate applications after this part. The parameters are:

$$\mu_1 = \begin{pmatrix} 0.0984 & 0.2228 & 0.3584 & 0.6638 & 0.8916 \end{pmatrix}$$

$$\mu_2 = \begin{pmatrix} 0.1591 & 0.3104 & 0.4430 & 0.7072 & 0.9103 \end{pmatrix}$$

$$\sigma_1 = \begin{pmatrix} 0.0433 & 0.0343 & 0.0639 & 0.0928 & 0.0503 \end{pmatrix}$$

$$\sigma_2 = \begin{pmatrix} 0.0541 & 0.0445 & 0.0612 & 0.0964 & 0.0407 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9950 & 0.0050 & 0 & 0 & 0 \\ 0.0025 & 0.9933 & 0.0042 & 0 & 0 \\ 0 & 0.0004 & 0.9946 & 0.0015 & 0 \\ 0 & 0 & 0.0022 & 0.9898 & 0.0080 \\ 0 & 0 & 0 & 0.0050 & 0.9950 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$mllk = -12378.42, AIC = -24668.84, BIC = -24390.86$$

Figure 4.7a contains the forecasting results for the S&P 500 while Figure 4.7b exposes the forecasting results for Nasdaq. In both cases the results can follow the real observations and catch the two sharp drops in 2019 and 2021.
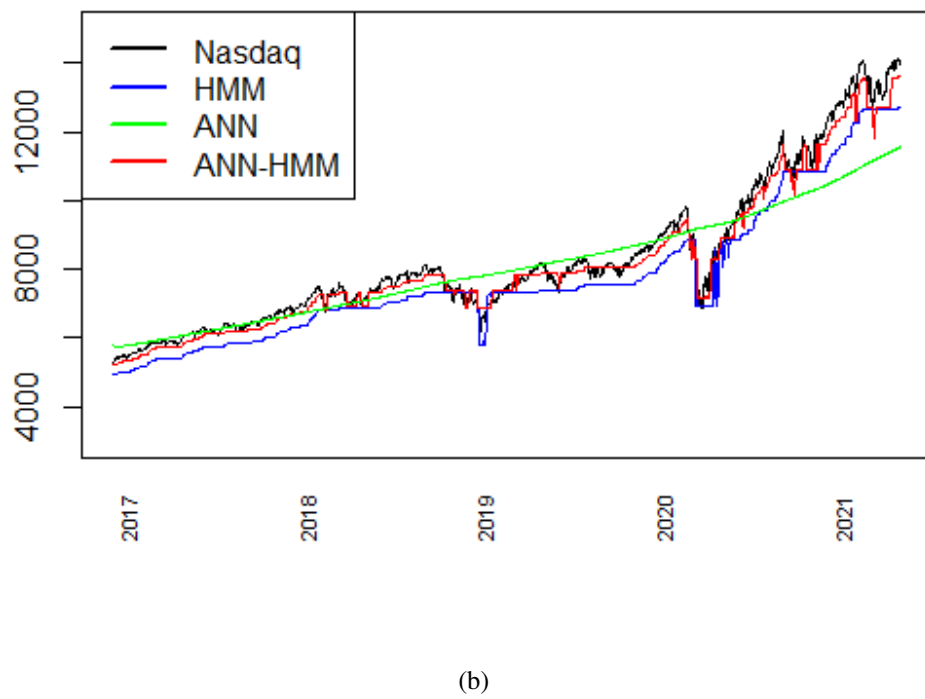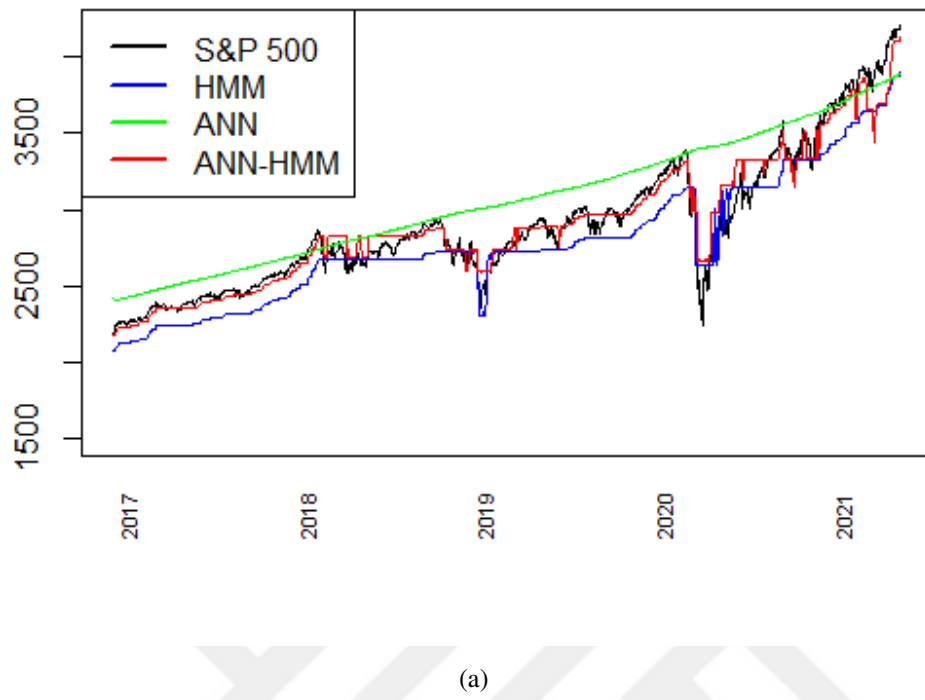
45

(a)



(b)

Figure 4.7: Forecasting results of (a) S&P 500 and (b) Nasdaq for Bivariate HMM, RNN and RNN-HMM

Table 4.6: Bivariate S&P 500 performances for each model

| | 2nd Variable | RMSE | MAPE | MAE |
|---|---|---|---|---|
| HMM | Nasdaq | 162.374 | 5.098 | 148.060 |
| | EUR/USD | 199.924 | 7.937 | 233.125 |
| RNN | Nasdaq | 180.957 | 3.937 | 120.541 |
| | EUR/USD | 259.345 | 6.992 | 212.478 |
| RNN-HMM | Nasdaq | **93.056** | **2.158** | **63.074** |
| | EUR/USD | **178.161** | **4.968** | **142.149** |

### 4.2.1.2   RNN-HMM on S&P 500 and Nasdaq

The parameters of EM algorithm part of bivariate RNN-HMM are chosen by trial and error as in the case of univariate RNN-HMMs. The initial parameters are determined as formulas after checking the selection criteria as follows:

$$\mu_1 = \Big( \mu(\mathbf{C}) - 2a_1 \quad \mu(\mathbf{C}_1) - a_1 \quad \mu(\mathbf{C}_1) \quad \mu(\mathbf{C}_1) + 2a_1 \quad \mu(\mathbf{C}_1) + 3a_1 \Big)$$

$$\mu_2 = \Big( \mu(\mathbf{C}_2) - 2a_2 \quad \mu(\mathbf{C}_2) - a_2 \quad \mu(\mathbf{C}_2) \quad \mu(\mathbf{C}_2) + 2a_2 \quad \mu(\mathbf{C}_2) + 3a_2 \Big)$$

$$\sigma_1 = \Big( \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/8 \Big)$$

$$\sigma_2 = \Big( \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/5 \Big)$$

where $\mathbf{C}_1$ is Nasdaq output and $\mathbf{C}_2$ is S&P 500 output of RNN, and

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{8}, \ a_2 = \frac{\max(\mathbf{C}_2) - \min(\mathbf{C}_2)}{7}.$$

After reaching the RNN part's ultimate weights and the EM algorithm's parameters in Table A.4, the forecasting is performed for both S&P 500 and Nasdaq and shown in Figure 4.7a and Figure 4.7b, respectively. Figure 4.7a and Table 4.6 illustrate that the Hybrid model (RMSE of 93.056 and MAPE 2.158) better fits the actual values of S&P 500 than the classical RNN (RMSE of 162.374 and MAPE 3.937) and HMM (RMSE of 180.957 and MAPE 7.937) in the case of bivariate S&P 500 and Nasdaq. Similarly, Figure 4.7b and Table 4.7 shows that Hybrid model on Nasdaq (RMSE of 374.436 and MAPE 3.607) provides better results than classical models.

Table 4.7: Bivariate Nasdaq performances for each model

| | 2nd Variable | RMSE | MAPE | MAE |
|---|---|---|---|---|
| HMM | S&P 500 | 676.588 | 7.146 | 598.548 |
| | EUR/USD | 911.658 | 8.647 | 1724.225 |
| RNN | S&P 500 | 1274.026 | 9.564 | 911.510 |
| | EUR/USD | 1649.092 | 14.301 | 1310.124 |
| RNN-HMM | S&P 500 | **374.436** | **3.607** | **302.828** |
| | EUR/USD | **371.790** | **3.306** | **268.968** |

## 4.2.2 Bivariate S&P 500 and EUR/USD Analyses

In this part, to examine the effect of the less correlated second variable on the proposed model, EUR/USD is modeled with S&P 500.

### 4.2.2.1 HMM on S&P 500 and EUR/USD

The initial mean and standard deviation parameters are determined as in previous applications by using bivariate histogram in Figure 3.4c. The estimated parameters, AIC, BIC, and mllk results of 5-state bivariate HMM application for S&P 500 index and EUR/USD exchange rate data are:

$$\mu_1 = \begin{pmatrix} 0.3218 & 0.2808 & 0.8233 & 0.6214 & 0.9256 \end{pmatrix}$$

$$\mu_2 = \begin{pmatrix} 0.6359 & 0.1387 & 0.3685 & 0.6521 & 0.3708 \end{pmatrix}$$

$$\sigma_1 = \begin{pmatrix} 0.0984 & 0.1209 & 0.0315 & 0.1395 & 0.0303 \end{pmatrix}$$

$$\sigma_2 = \begin{pmatrix} 0.1446 & 0.0907 & 0.0244 & 0.0505 & 0.0375 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9985 & 0 & 0 & 0 & 0.0015 \\ 0.0015 & 0.9985 & 0 & 0 & 0 \\ 0 & 0 & 0.9770 & 0 & 0.0230 \\ 0.0022 & 0 & 0 & 0.9967 & 0.0011 \\ 0 & 0 & 0 & 0.0050 & 0.9950 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

48

$$mllk = -7820.732, AIC = -15553.46, BIC = -15275.48$$

According to the forecasting results in Figure 4.8a and Figure 4.8b, the model follows the real values, but does not give as satisfying results as the application on two highly correlated variables.

### 4.2.2.2  RNN-HMM on S&P 500 and EUR/USD

The initial parameter formulas of the EM algorithm for the bivariate case with S&P 500 and EUR/USD are shown below.

$$\mu_1 = \Big( \mu(\mathbf{C}) + 1.5a_1 \quad \mu(\mathbf{C}_1) + 2.5a_1 \quad \mu(\mathbf{C}_1) \quad \mu(\mathbf{C}_1) - 2a_1 \quad \mu(\mathbf{C}_1) + 3a_1 \Big)$$

$$\mu_2 = \Big( \mu(\mathbf{C}_2) - 3a_2 \quad \mu(\mathbf{C}_2) + 2a_2 \quad \mu(\mathbf{C}_2) + a_2 \quad \mu(\mathbf{C}_2) \quad \mu(\mathbf{C}_2) \Big)$$

$$\sigma_1 = \Big( \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/5 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/5 \quad \sigma(\mathbf{C}_1)/5 \Big)$$

$$\sigma_2 = \Big( \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/5 \Big)$$

where $\mathbf{C}_1$ is S&P 500 output, $\mathbf{C}_2$ is EUR/USD output of RNN, and

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{9}, \ a_2 = \frac{\max(\mathbf{C}_2) - \min(\mathbf{C}_2)}{9}$$

The forecasting is performed by the help of estimated parameters of EM algorithm in Table A.5 and the results of the S&P 500 are displayed in Figure 4.8a and those for EUR/USD are presented in Figure 4.8b. Although the implementation where the second variable is EUR/USD provides less improvement than the implementation with Nasdaq, as it presented in Tables 4.6 and 4.8 for both variables S&P 500 and EUR/USD, RNN-HMM provides better accuracy estimation than HMM does.

The fact that the estimations are almost flat, which is seen in most of the bivariate applications made with EUR/USD, is also seen here. The reason is that as seen in Figure 3.2b, while S&P 500 and Nasdaq are increasing rapidly, EUR/USD is relatively stable. In this case, catching the mean of the EUR/USD is sufficient to keep the common RMSE low, while capturing the values of the S&P 500 and Nasdaq outweighs the low RMSE. Moreover, another possible reason is that EUR/USD starts to follow a different pattern towards the end of the training data.
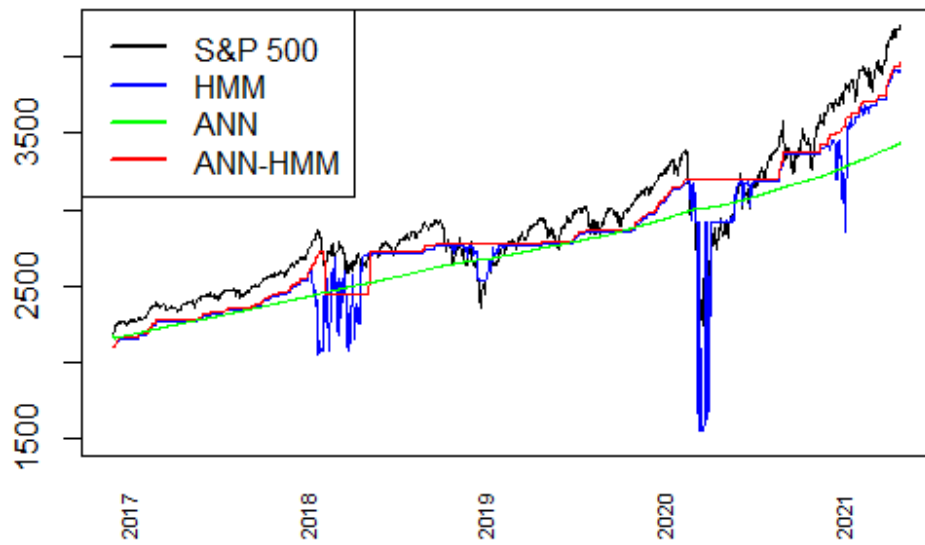
(a)
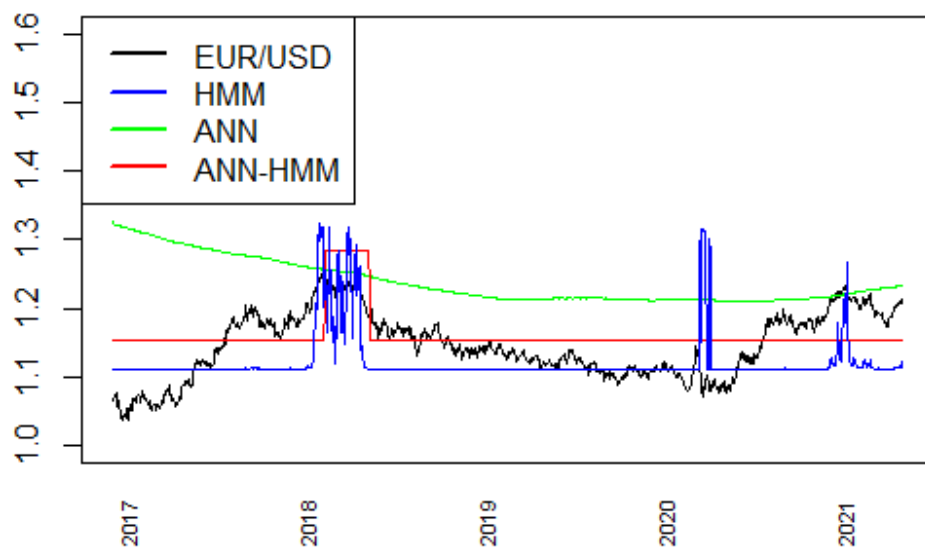


(b)

Figure 4.8: Forecasting results of (a) S&P 500 and (b) EUR/USD for Bivariate HMM, RNN and RNN-HMM

### 4.2.3 Bivariate Nasdaq and EUR/USD Analyses

#### 4.2.3.1 HMM on Nasdaq and EUR/USD

The initial mean and standard deviations of the states are determined by considering the bivariate histogram (Figure 3.4b). The parameters, AIC, BIC, and mllk results of 5-state are estimated for the bivariate HMM application to Nasdaq and EUR/USD data and given as:

$$\mu_1 = \begin{pmatrix} 0.2168 & 0.1928 & 0.6690 & 0.3700 & 0.8949 \end{pmatrix}$$

$$\mu_2 = \begin{pmatrix} 0.6155 & 0.1393 & 0.6499 & 0.6774 & 0.3683 \end{pmatrix}$$

$$\sigma_1 = \begin{pmatrix} 0.0589 & 0.1562 & 0.1124 & 0.0504 & 0.0522 \end{pmatrix}$$

$$\sigma_2 = \begin{pmatrix} 0.1526 & 0.0913 & 0.0553 & 0.0790 & 0.0316 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9980 & 0 & 0 & 0.0020 & 0 \\ 0.0015 & 0.9985 & 0 & 0 & 0 \\ 0 & 0 & 0.9980 & 0 & 0.0020 \\ 0.0022 & 0 & 0.0011 & 0.9967 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$mllk = -9077.304, AIC = -18066.61, BIC = -17788.63.$$

Figure 4.9a and Figure 4.9b include the forecasting results of HMM application to Nasdaq and EUR/USD, respectively. Moreover, the performance measures are presented in Table 4.7 and Table 4.8.

#### 4.2.3.2 RNN-HMM on Nasdaq and EUR/USD

The initial parameters for the bivariate case with Nasdaq and EUR/USD are shown below.

$$\mu_1 = \begin{pmatrix} \mu(\mathbf{C}) - a_1 & \mu(\mathbf{C}_1) + 2a_1 & \mu(\mathbf{C}_1) - 0.5a_1 & \mu(\mathbf{C}_1) - 2a_1 & \mu(\mathbf{C}_1) + 3.5a_1 \end{pmatrix}$$
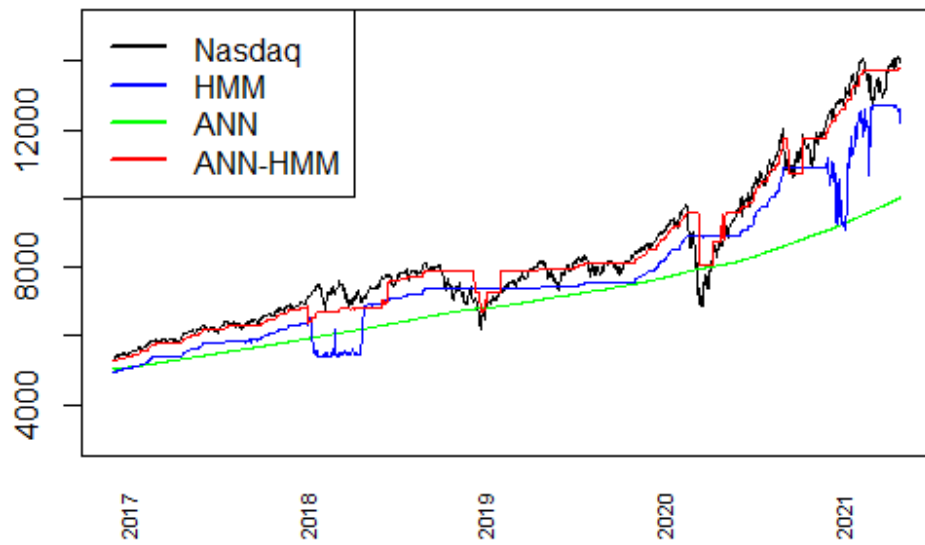
(a)
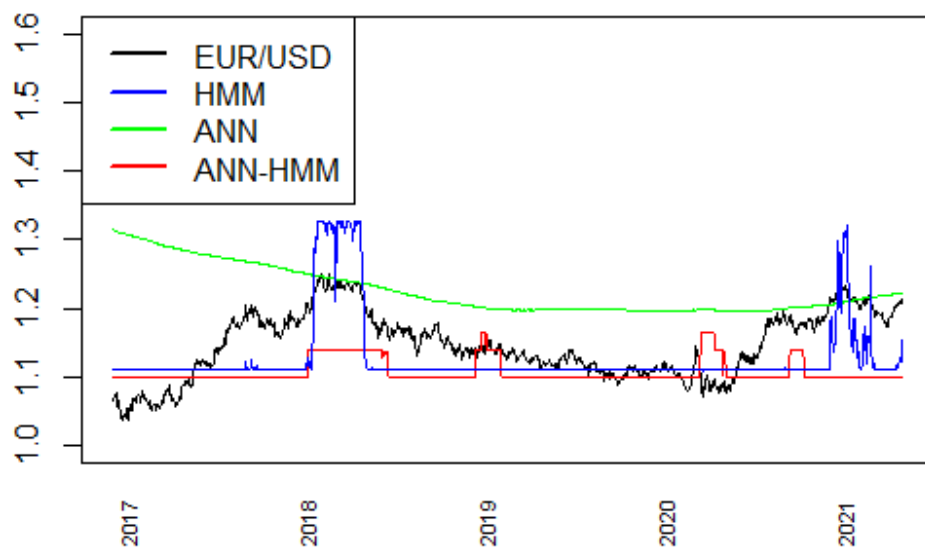


(b)

Figure 4.9: Forecasting results of (a) Nasdaq and (b) EUR/USD for Bivariate HMM, RNN and RNN-HMM

Table 4.8: Bivariate EUR/USD performances for each model

|  | 2nd Variable | RMSE | MAPE | MAE |
|---|---|---|---|---|
| HMM | S&P 500 | 0.054 | 3.738 | 0.043 |
|  | Nasdaq | **0.051** | **3.667** | **0.043** |
| RNN | S&P 500 | 0.111 | 8.294 | 0.080 |
|  | Nasdaq | 0.101 | 7.224 | 212.478 |
| RNN-HMM | S&P 500 | **0.047** | **3.551** | **0.040** |
|  | Nasdaq | 0.060 | 4.290 | 0.050 |

$$\mu_2 = \left( \mu(\mathbf{C}_2) - 3a_2 \quad \mu(\mathbf{C}_2) + 2a_2 \quad \mu(\mathbf{C}_2) + a_2 \quad \mu(\mathbf{C}_2) \quad \mu(\mathbf{C}_2) + a_2 \right)$$

$$\sigma_1 = \left( \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/8 \right)$$

$$\sigma_2 = \left( \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/5 \right)$$

where $\mathbf{C}_1$ is Nasdaq output and $\mathbf{C}_2$ is EUR/USD output of RNN, and

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{9}, \; a_2 = \frac{\max(\mathbf{C}_2) - \min(\mathbf{C}_2)}{9}$$

The estimated parameters of EM-algorithm part is presented in Table A.6. Figure 4.9a and Figure 4.9b includes the forecasting results of RNN-HMM application to Nasdaq and EUR/USD, respectively. Additionaly, the figures include RNN results. According to the figures, the Nasdaq estimation results show that the RNN-HMM model provides a significant improvement over the classical RNN and HMM models. The proposed model is more successful in catching fluctuations. As in the previous cases, the hybrid model cannot produce better results than the HMM application in EUR/USD. This is because the proposed model is built for S&P 500 and indirectly for Nasdaq modeling.

When the performance measures in Table 4.8 are compared, the EUR/USD estimations give the best results for the proposed model when the second variable is S&P 500. However, when the second variable is Nasdaq, although HMM gives the best results, these results are actually quite close to the RNN-HMM ones.

## 4.3 Trivariate Analyses

### 4.3.1 HMM on S&P 500, Nasdaq and EUR/USD

The parameters, AIC, BIC and mllk results of 5-state trivariate HMM application to S&P 500, Nasdaq and EUR/USD are:

$$\mu_1 = \begin{pmatrix} 0.2802 & 0.1848 & 0.8255 & 0.6313 & 0.9216 \end{pmatrix}$$

$$\mu_2 = \begin{pmatrix} 0.3578 & 0.2694 & 0.8565 & 0.6917 & 0.9322 \end{pmatrix}$$

$$\mu_3 = \begin{pmatrix} 0.6497 & 0.1764 & 0.4556 & 0.6675 & 0.3617 \end{pmatrix}$$

$$\sigma_1 = \begin{pmatrix} 0.0890 & 0.1467 & 0.0308 & 0.0996 & 0.0327 \end{pmatrix}$$

$$\sigma_2 = \begin{pmatrix} 0.1096 & 0.1157 & 0.0379 & 0.0948 & 0.0277 \end{pmatrix}$$

$$\sigma_3 = \begin{pmatrix} 0.1264 & 0.1260 & 0.0971 & 0.0391 & 0.0289 \end{pmatrix}$$

$$\Gamma = \begin{pmatrix} 0.9996 & 0.0004 & 0 & 0 & 0 \\ 0.0013 & 0.9987 & 0 & 0 & 0 \\ 0 & 0 & 0.9946 & 0.0004 & 0 \\ 0 & 0 & 0.0040 & 0.9744 & 0.0213 \\ 0 & 0 & 0.0118 & 0 & 0.9881 \end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$mllk = -11873, AIC = -23638.63, BIC = -23296.47$$

The forecasting results of trivariate HMM for S&P 500, Nasdaq and EUR/USD are presented in Figure 4.10, Figure 4.11 and Figure 4.12, respectively. Especially in the case of S&P 500 and Nasdaq, forecasting results are quite successful in capturing the actual data.
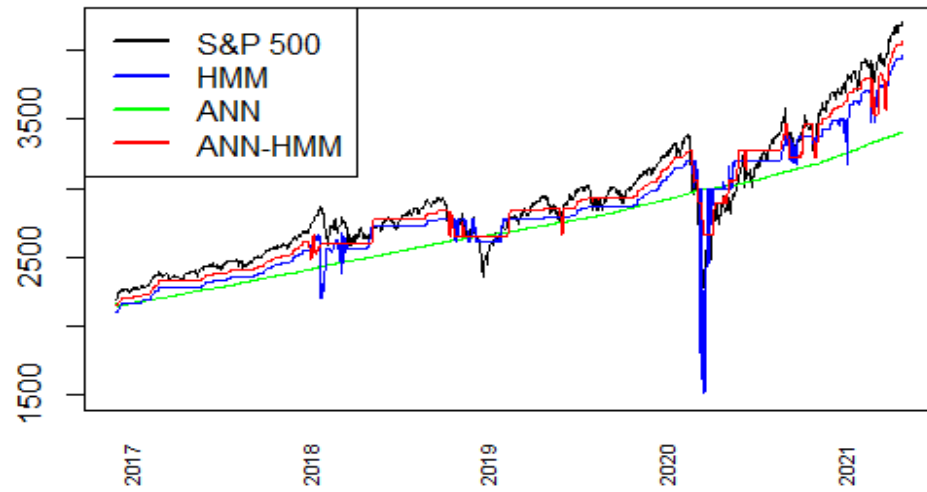
Figure 4.10: Forecasting results of S&P 500 for trivariate HMM, RNN and RNN-HMM
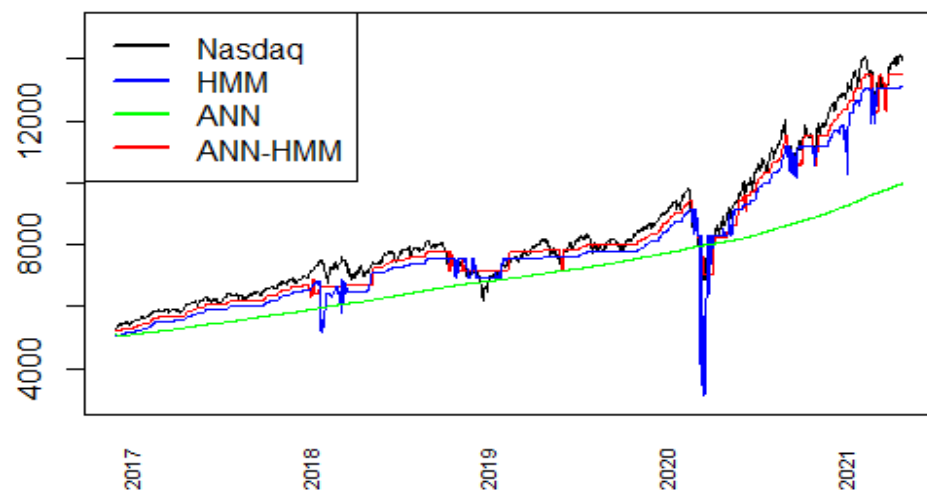


Figure 4.11: Forecasting results of Nasdaq for trivariate HMM, RNN and RNN-HMM
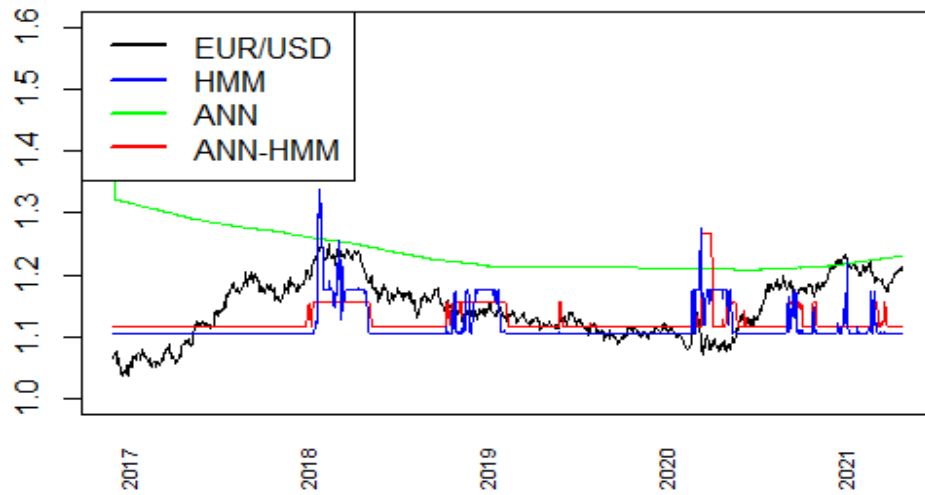
Figure 4.12: Forecasting results of EUR/USD for trivariate HMM, RNN and RNN-HMM

Table 4.9: Trivariate performance for each model

|         |         | RMSE     | MAPE   | MAE      |
|---------|---------|----------|--------|----------|
| S&P500  | HMM     | 159.496  | 4.350  | 126.360  |
|         | RNN     | 273.472  | 7.543  | 228.199  |
|         | RNN-HMM | **103.634** | **2.848** | **83.310**  |
| Nasdaq  | HMM     | 618.278  | 5.842  | 481.954  |
|         | RNN     | 1656.112 | 14.425 | 1314.052 |
|         | RNN-HMM | **363.918** | **3.641** | **301.135** |
| EUR/USD | HMM     | 0.057    | 4.097  | 0.048    |
|         | RNN     | 0.111    | 8.243  | 0.092    |
|         | RNN-HMM | **0.054** | **3.718** | **0.043**  |

56

### 4.3.2 RNN-HMM on S&P 500, Nasdaq and EUR/USD

The initial parameters of EM algorithm part in the form of formulas as are determined as follows for the trivariate model.

$$\mu_1 = \Big( \mu(\mathbf{C}) - a_1 \quad \mu(\mathbf{C}_1) + 2a_1 \quad \mu(\mathbf{C}_1) - 0.5a_1 \quad \mu(\mathbf{C}_1) - 2a_1 \quad \mu(\mathbf{C}_1) + 3.5a_1 \Big)$$

$$\mu_2 = \Big( \mu(\mathbf{C}_2) - a_2 \quad \mu(\mathbf{C}_2) + 2.5a_2 \quad \mu(\mathbf{C}_2) \quad \mu(\mathbf{C}_2) - 2a_2 \quad \mu(\mathbf{C}_2) + 3.5a_2 \Big)$$

$$\mu_3 = \Big( \mu(\mathbf{C}_3) - 3a_3 \quad \mu(\mathbf{C}_3) + a_3 \quad \mu(\mathbf{C}_3) + a_3 \quad \mu(\mathbf{C}_3) \quad \mu(\mathbf{C}_3) - a_3 \Big)$$

$$\sigma_1 = \Big( \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/8 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/4 \quad \sigma(\mathbf{C}_1)/8 \Big)$$

$$\sigma_2 = \Big( \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/4 \quad \sigma(\mathbf{C}_2)/5 \quad \sigma(\mathbf{C}_2)/5 \Big)$$

$$\sigma_3 = \Big( \sigma(\mathbf{C}_3)/4 \quad \sigma(\mathbf{C}_3)/5 \quad \sigma(\mathbf{C}_3)/4 \quad \sigma(\mathbf{C}_3)/5 \quad \sigma(\mathbf{C}_3)/5 \Big)$$

where $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$ are Nasdaq, S&P 500 and Exchange Rate output of RNN, respectively. Moreover,

$$a_1 = \frac{\max(\mathbf{C}_1) - \min(\mathbf{C}_1)}{9}, a_2 = \frac{\max(\mathbf{C}_2) - \min(\mathbf{C}_2)}{9}, \text{and} \quad a_3 = \frac{\max(\mathbf{C}_3) - \min(\mathbf{C}_3)}{9}$$

Then, the EM- algorithm parameters are estimated as in Table A.7. The forecasting results of trivariate RNN-HMM for S&P 500, Nasdaq and EUR/USD are presented in Figure 4.10, Figure 4.11 and Figure 4.12, respectively. Trivariate RNN-HMM application also gives successful results like Trivariate HMM. Although the results in the figures look very close to each other, when the error metrics in Table 4.9 are examined, it becomes clear that the RNN-HMM model performs better.

HMM, RNN and RNN-HMM models based on variations in dimension are compared and the ones giving best accuracy results are summarized in Table 4.10. It is seen that the proposed model gives better results than the classical models for most of the cases based on performance measures. However, the proposed model only gives similar results with the HMM in univariate EUR/USD application and does not give better results than the HMM for the bivariate application of EUR/USD applied with the Nasdaq.

Table 4.10: Summary of the best fitting model

| | | HMM | RNN-HMM | RNN |
|---|---|---|---|---|
| Univariate | SP500 | | X | |
| | Nasdaq | | X | |
| | EUR/USD | X | X | |
| Bivariate | SP500 Nasdaq | | X | |
| | SP500 EUR/USD | | X | |
| | Nasdaq EUR/USD | X | X | |
| Trivariate | SP500 | | X | |
| | Nasdaq | | X | |
| | EUR/USD | | X | |

## 4.4 Comparison of models with respect to the Number of Variables

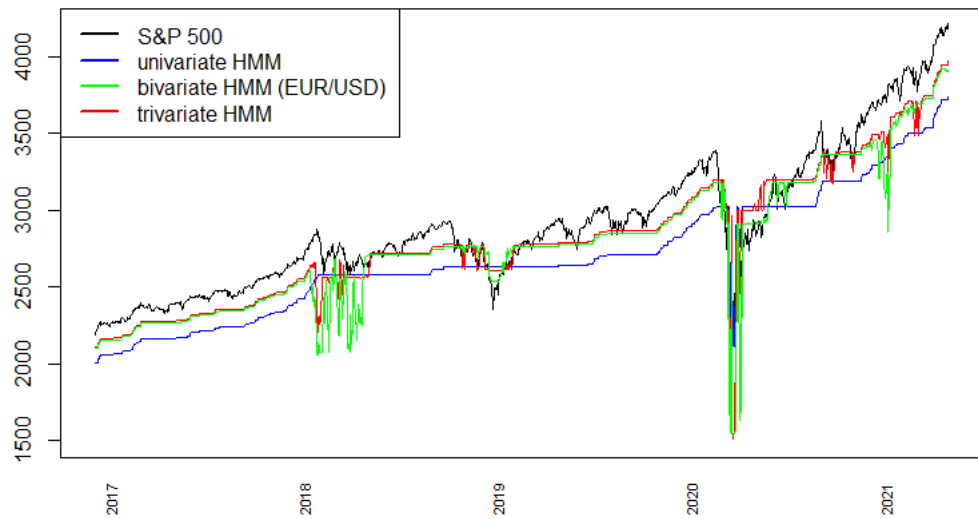### 4.4.1 Univariate, Bivariate and Trivariate comparisons of HMM

All the univariate, bivariate and trivariate HMM application results of S&P 500 are shown in Figure 4.13a and Figure 4.13b to see the differences between behavior of different numbers and type of variable choice. Figures are shown twice to reflect the two different combinations in the bivariate case. As the second variable, one shows the Nasdaq and the other shows the EUR/USD. These figures clearly show that trivariate application gives the best results among the HMM applications. Moreover, it is seen that each of the bivariate HMMs that uses Nasdaq and EUR/USD as the second variable gives better results than the univariate case. However, highly correlated variables increase accuracy more than those with low correlation.

The inferences made on the Nasdaq data are quite similar to those in the S&P 500 application, as the two variables are highly correlated. For basic HMM applications, all variable applications are shown in Figure 4.14a and Figure 4.14b for comparison. It is clear that the bivariate HMM forecasting results are better than the univariate HMM. The trivariate model provides the best estimations among HMM applications.

In Figure 4.15a and Figure 4.15b, the forecasting results of EUR/USD including for all variable number choice are displayed. In S&P 500 and Nasdaq applications, the accuracy increases as the number of variables increases in HMM applications. In contrast, the increase in the number of variables in the applications made on the exchange rate decreases the accuracy. The closest results are obtained from the univariate model. This is because the models are built for S&P 500 and, therefore, for highly correlated Nasdaq data. The addition of one of these data may cause the model to be suppressed.
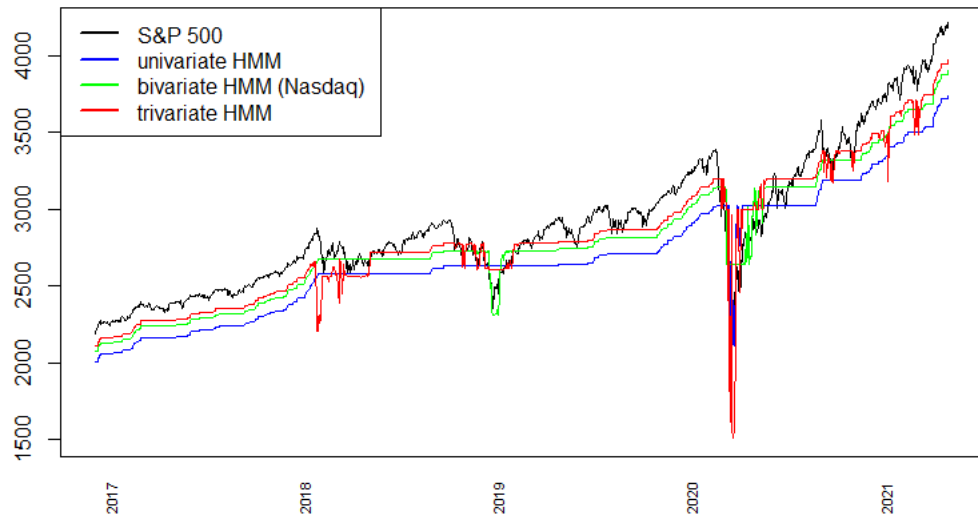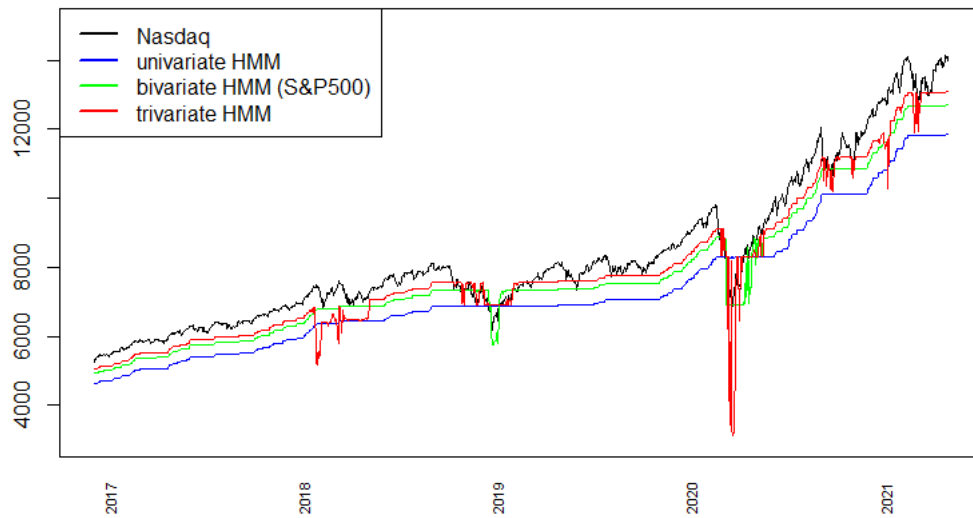
(a)



(b)

Figure 4.13: Forecasting results of S&P 500 index for univariate, bivariate and trivariate HMMs

(a)



(b)

Figure 4.14: Forecasting results of Nasdaq index for univariate, bivariate and trivariate HMMs

(a)



(b)

Figure 4.15: Forecasting results of EUR/USD index for univariate, bivariate and trivariate HMMs

### 4.4.2 Univariate, Bivariate and Trivariate Comparisons of RNN-HMM

All the number of variable cases for RNN-HMM on S&P 500 are presented together in Figure 4.16a and Figure 4.16b. These figures show that the univariate model has the lowest accuracy among the hybrid models. Moreover, as expected, the bivariate RNN-HMM application with Nasdaq data gives better results than the bivariate application with the exchange rate data. The trivariate model does not significantly improve the accuracy of the hybrid model. Although it gives better results than the case using the EUR/USD as the second variable, it does not have an advantage over the case using the Nasdaq as the second variable.

The forecasting results of RNN-HMM on Nasdaq obtained according to all variable numbers are compared in Figure 4.17a and Figure 4.17b. Although bivariate and trivariate results have higher accuracy than the univariate case, the estimation results for all variables are close to each other.

Estimation results for all variable number selections are shown in Figure 4.18a and Figure 4.18b. Accordingly, adding variables to RNN-HMM models on EUR/USD data does not improve their accuracy, as in the HMM applications on them.

The tables (Table 4.3 - 4.9) are presented to summarize all the applications of all the variable choices. For S&P 500 data we see that increasing the number of variables provides better accuracy. However, bivariate RNN-HMM that uses Nasdaq as the second variable performs better than the trivariate RNN-HMM. For all of the methods applied on Nasdaq data shows that while the increase in the number of variables in HMMs significantly contributes to the accuracy, this is not the case for the hybrid case. Bivariate RNN-HMMs perform better than the univariate RNN-HMM as in the applications of S&P 500. However, when two and three variables are compared, it is not possible to say which one is more successful. Because according to the RMSE results, the trivariate model seems less successful, whereas it performs better according to the MAPE and MAE results. To compare the models, the best model here is the RNN-HMM Hybrid model, and the HMM follows it as in S&P 500 applications. For EUR/USD although the test and training part of the EUR/USD are structurally different, the proposed hybrid model gives good results. However, while it provides a clear advantage to the RNN model, this is not the case against the HMM model.

## 4.5 Global Decoding

The best sequences of hidden states are determined by the Viterbi Algorithm explained in Section 2.1.2. The global decoding results of HMM are presented in Appendix B.

Figure B.1 shows the state decoding results of univariate HMM application on S&P 500. In this figure, the 1st state is the representative of recession in US economy. The 5th state belongs to the last peak of the training set, while the 4th state belongs to the peak periods in the US economy until 2014. Since the US economy's longest draw-down period is between August
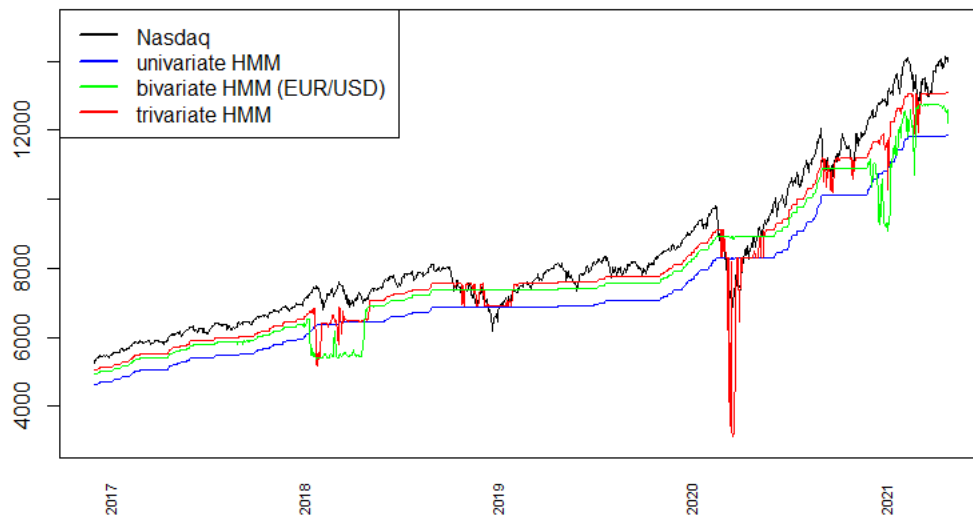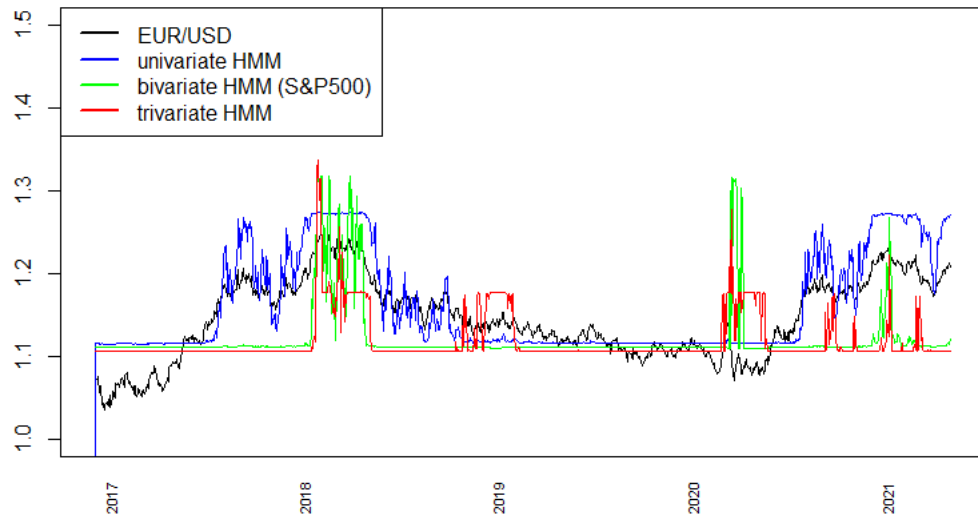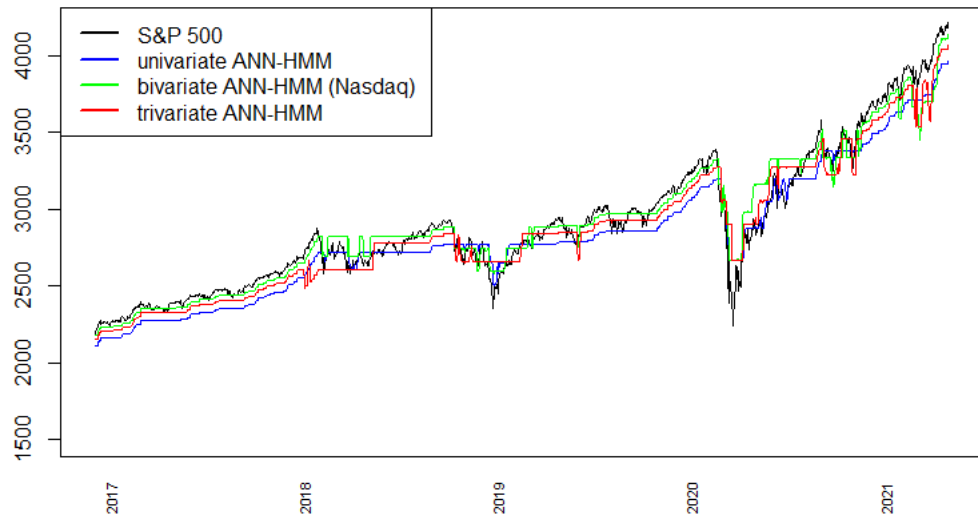
(a)
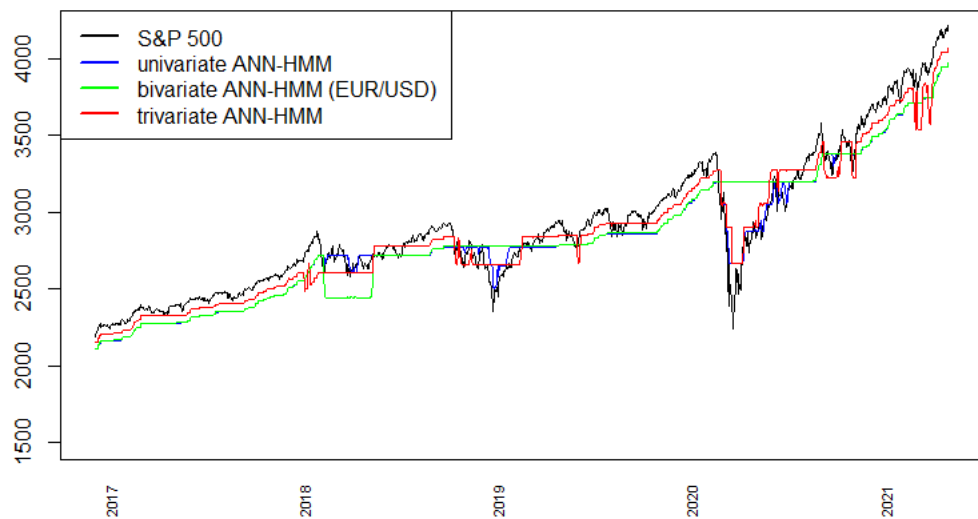


(b)

Figure 4.16: Forecasting results of S&P 500 index for univariate, bivariate and trivariate RNN-HMMs
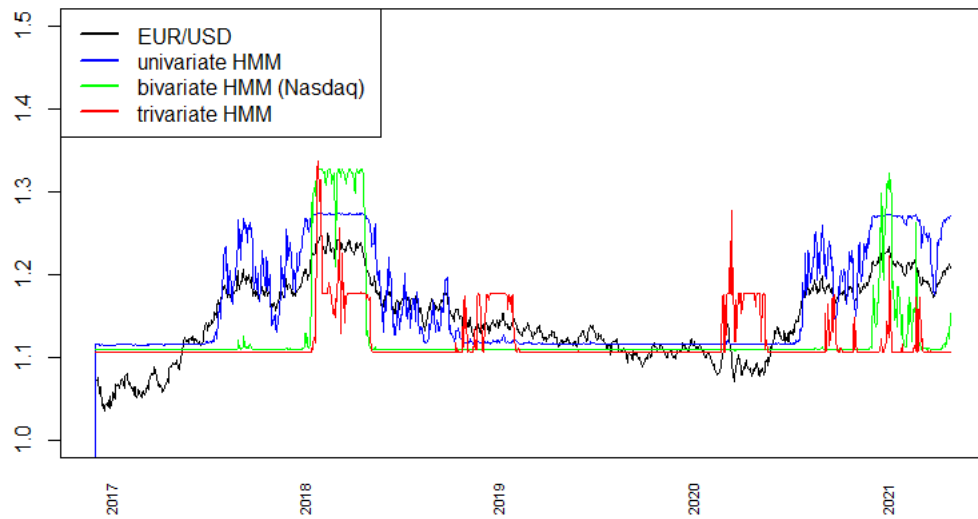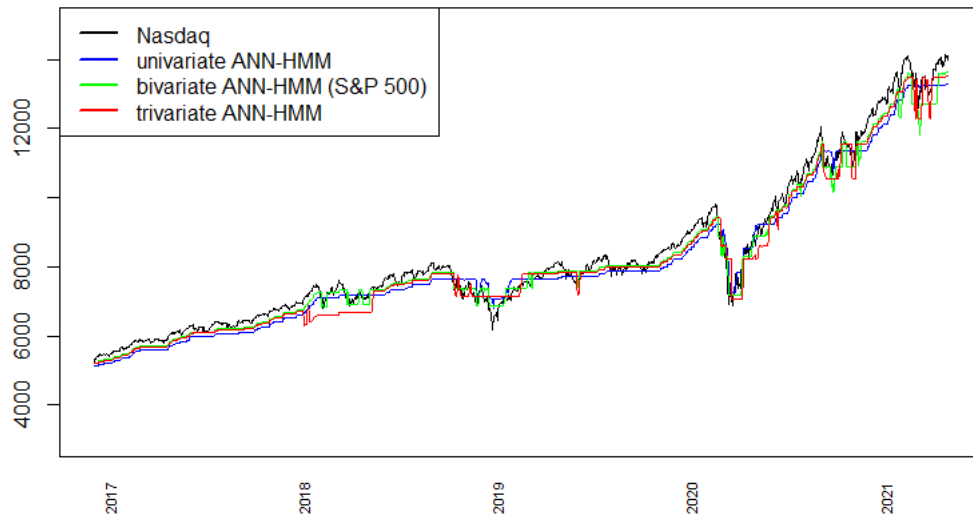
(a)
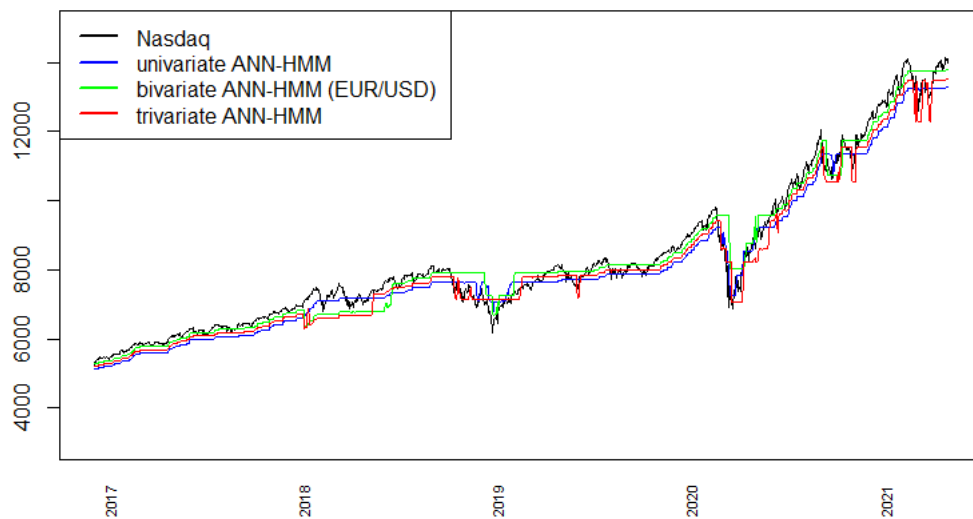


(b)

Figure 4.17: Forecasting results of Nasdaq index for univariate, bivariate and trivariate RNN-HMMs

(a)



(b)

Figure 4.18: Forecasting results of EUR/USD index for univariate, bivariate and trivariate RNN-HMMs

65

2000 and December 2013, the model assigns a state specific to the period after 2013. The 2nd and 3rd states are the stationary periods in the expansion and contraction periods of the business cycle. Moreover, the decoding results of univariate HMM on Nasdaq is presented in Figure B.2. Since, Nasdaq is highly correlated with S&P 500, the states are the representative of same business cycles. In both figures, the states fit the real data successfully.

The state decoding of univariate HMM on EUR/USD is shown in Figure B.3. It is seen that the first circulation period of the EUR is determined as the first state by the model. A turning point in the US economy in 2014 caused the EUR to depreciate against the USD and to enter a new era. This period is represented as 3rd state together with the 2003-2004 time interval in the first appreciation periods of the EUR. For the remaining part outside these time intervals, the periods when the EUR peaks are represented by 5th state, and the periods when it troughs are represented by 4th state.

When the bivariate S&P 500 and Nasdaq application in Figure B.4 and Figure B.5 are examined, the number of states has increased especially after 2014. This section is represented by two states. The 4th and 5th states represent the peaks and troughs of this period, and the 1st and 3rd states represent the peaks and troughs before 2014. The second state represents the stationary periods in the expansion and contraction periods.

When the bivariate HMM application of Nasdaq and S&P 500 with EUR/USD is examined as shown in Figure B.6 and Figure B.8, one state in both variables is dominated by the low value of EUR/USD in the initial period. However, while peak periods are still represented by a state, a separate state is not formed for the lowest points. Instead, draw down periods are represented by a state. However, looking at the forecast results, this situation still provides better results than univariate applications. Since EUR/USD is the second variables of these bivariate applications the decoding results in Figure B.7 and Figure B.9 represent the same states with above mentioned ones. However, an extra conclusion that can be drawn from the figures is that the S&P 500 and Nasdaq dominate the EUR/USD states. The means of the states in 2004-2015 and 2015-2017 time periods are very close to each other. This explains why the EUR/USD forecast results are mostly straight lines.

S&P 500, Nasdaq and EUR/USD results for trivariate HMM are shown in Figure B.10, Figure B.11 and Figure B.12, respectively. Since S&P 500 and Nasdaq are used together in this application, they dominate EUR/USD more and ensure that the states are more on stock prices. However, it should be noted that the state boundaries are different for each variable dimension and combination. The EUR/USD variable contributes to determine the boundaries of the states of the stock variables. However, as expected, such a combination causes the inaccuracy of the EUR/USD forecast results.

# CHAPTER 5

# CONCLUSION AND OUTLOOK

The starting point of this thesis is to benefit from the possibilities of machine learning in the financial sector, which has a vital place in today's world. HMM, which is a probability model used in machine learning area also, gives successful results in time series. However, HMM still can suffer from local maximum points without finding the global maximum since it only guarantees to converge to a zero gradient point. With this study, it is aimed to take advantage of the classification power using RNNs and to maximize the likelihood of HMM. Classification power is used to classify the hidden states of HMM. It is also desired to obtain the closest estimation results to the stock prices. This does not only give the RNN the function of approximating the real result, but it can also indirectly optimize the EM parameters as the estimation is done with the parameters of the EM algorithm. To elaborate more, in our study at each step parameter estimation is made by EM-algorithm and the gradients are computed according to the cost function of the forecast values. Hence, we investigate whether combining RNNs and HMMs in the proposed way described leads to improved performance.

This study introduces a novel approach by modifying an existing approach on non-categorical data sets such as financial data. Moreover, several variable choices are considered to see the effect of the number and the correlation factor of the variables involved in. Therefore, three variables are chosen to practice the model and compare it with the classical HMMs and RNNs. Firstly, S&P 500 data is determined to model. Then, in order to measure the performance of the model on data with different characteristics, the Nasdaq, which is highly correlated with the S&P 500, and the EUR/USD, which has a very important place in the US economy but has a very low correlation with the S&P 500, are selected for modeling. The results show that S&P 500 and Nasdaq bivariate applications gives better results than the univariate ones. Morever, adding a low correlated variable EUR/USD increases the accuracy although the contribution of it is not as high as an correlated variable. This is because a second variable contributes when determining state boundaries. Since the hidden states of the highly correlated variables is expected to be very similar, they contribute more than the second variable with low correlation.

Another point to be remarked in this thesis is that the exchange rate data has very different features for the training and test sections, but the proposed model still provides promising results. According to the results, in almost all applications, the hybrid model introduced

in this study offers better performance metrics. Moreover, bivariate applications that uses highly correlated S&P 500 and Nasdaq variables give better results than the univariate cases. Trivariate applications provide a significant improvement in the classical model, while it is shown to be not proper for the Hybrid models since these models have already made sufficient improvement. Moreover, the difficulty in determining the initial values chosen as the formulas is another possible reason. On the other hand, increasing the number of variables up to three variables does not increase the accuracy of the exchange rate variable. The reason is that the structure of the models is adjusted according to the S&P 500 and Nasdaq variables.

As a future work, it is planned to replace the basic RNN part of this model with Neural Network types such as LSTM and CNN, which generally provide more successful results to increase the performance even more.

# REFERENCES

[1] A. Abraham, Artificial neural networks, Handbook of Measuring System Design, 2005.

[2] D. Aydoğan, *Determination of Inflation Rate in a Hidden Markov Model Framework: Turkey Case*, Master's thesis, Middle East Technical University, Financial Mathematics, Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey, 2015.

[3] L. Bahl, P. Brown, P. De Souza, and R. Mercer, Maximum mutual information estimation of hidden markov model parameters for speech recognition, in *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 11, pp. 49–52, IEEE, 1986.

[4] Y. Bengio, R. Cardin, R. De Mori, and Y. Normandin, A hybrid coder for hidden markov models using a recurrent neural networks, in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 537–540, IEEE, 1990.

[5] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, Global optimization of a neural network-hidden markov model hybrid, IEEE Transactions on Neural Networks, 3(2), pp. 252–259, 1992.

[6] R. Bhar and S. Hamori, *Hidden Markov models: applications to financial economics*, volume 40, Springer Science & Business Media, 2004.

[7] V. Bhusari and S. Patil, Application of hidden markov model in credit card fraud detection, International Journal of Distributed and Parallel Systems, 2(6), p. 203, 2011.

[8] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, volume 4, Springer, 2006.

[9] A. Boukerche and J. Wang, Machine learning-based traffic prediction models for intelligent transportation systems, Computer Networks, 181, p. 107530, 2020.

[10] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*, John Wiley & Sons, 2015.

[11] A. Brekkan, S. Jönsson, M. O. Karlsson, and E. L. Plan, Handling underlying discrete variables with bivariate mixed hidden markov models in NONMEM, Journal of Pharmacokinetics and Pharmacodynamics, 46(6), pp. 591–604, 2019.

[12] J. Bulla, F. Lagona, A. Maruotti, and M. Picone, A multivariate hidden markov model for the identification of sea regimes from incomplete skewed and circular time series,

Journal of Agricultural, Biological, and Environmental Statistics, 17(4), pp. 544–567, 2012.

[13] J. Byrne, J. Taminiau, K. N. Kim, J. Lee, and J. Seo, Multivariate analysis of solar city economics: Impact of energy prices, policy, finance, and cost on urban photovoltaic power plant implementation, Advances in Energy Systems: The Large-scale Renewable Energy Integration Challenge, pp. 491–506, 2019.

[14] L. W.-K. Cheung, Use of run statistics for pattern recognation in genomic DNA sequences, Journal of Computational Biology, 11(1), pp. 107–124, 2004.

[15] R. Collobert and J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167, 2008.

[16] G. D'Amico and F. Petroni, Copula based multivariate semi-markov models with applications in high-frequency finance, European Journal of Operational Research, 267(2), pp. 765–777, 2018.

[17] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society. Series B (methodological), pp. 1–38, 1977.

[18] S. R. Eddy, Hidden markov models, Current Opinion in Structural Biology, 6(3), pp. 361–365, 1996.

[19] C.-D. Fuh, Sprt and cusum in hidden markov models, The Annals of Statistics, 31(3), pp. 942–977, 2003.

[20] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, Deep learning-based image recognition for autonomous driving, IATSS Research, 43(4), pp. 244–252, 2019.

[21] J. Gao, Y. L. Murphey, and H. Zhu, Multivariate time series prediction of lane changing behavior using deep neural network, Applied Intelligence, 48(10), pp. 3523–3537, 2018.

[22] Z. Ghahramani, An introduction to hidden markov models and bayesian networks, International Journal of Pattern Recognition and Artificial Intelligence, 15(01), pp. 9–42, 2001.

[23] M. F. Ghalwash, D. Ramljak, and Z. Obradović, Early classification of multivariate time series using a hybrid HMM/SVM model, in *Bioinformatics and Biomedicine (BIBM), 2012 IEEE International Conference on*, pp. 1–6, IEEE, 2012.

[24] P. Gogas and T. Papadimitriou, Machine learning in economics and finance, Computational Economics, 57(1), pp. 1–4, 2021.

[25] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet, Online map-matching based on hidden markov model for real-time traffic sensing applications, in

*2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 776–781, IEEE, 2012.

[26] A. Gupta and B. Dhingra, Stock market prediction using hidden markov models, in *2012 Students Conference on Engineering and Systems*, pp. 1–4, IEEE, 2012.

[27] J. D. Hamilton, A new approach to the economic analysis of nonstationary time series and the business cycle, Econometrica: Journal of the Econometric Society, pp. 357–384, 1989.

[28] M. R. Hassan and B. Nath, Stock market forecasting using hidden markov model: a new approach, in *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*, pp. 192–196, IEEE, 2005.

[29] M. R. Hassan, B. Nath, and M. Kirley, A fusion model of HMM, ANN and GA for stock market forecasting, Expert Systems with Applications, 33(1), pp. 171–180, 2007.

[30] M. R. Hassan, B. Nath, M. Kirley, and J. Kamruzzaman, A hybrid of multiobjective evolutionary algorithm and HMM-fuzzy model for time series prediction, Neurocomputing, 81, pp. 1–11, 2012.

[31] X. He and S. Xu, *Process neural networks: Theory and applications*, Springer Science & Business Media, 2010.

[32] O. Ibe, *Markov Processes for Stochastic Modeling*, Newnes, 2013.

[33] D. C. Kale, D. Gong, Z. Che, Y. Liu, G. Medioni, R. Wetzel, and P. Ross, An examination of multivariate time series hashing with applications to health care, in *2014 IEEE International Conference on Data Mining*, pp. 260–269, IEEE, 2014.

[34] A. I. Khan and S. Al-Habsi, Machine learning in computer vision, Procedia Computer Science, 167, pp. 1444–1451, 2020.

[35] I. Kononenko, Machine learning for medical diagnosis: history, state of the art and perspective, Artificial Intelligence in Medicine, 23(1), pp. 89–109, 2001.

[36] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler, Hidden markov models in computational biology: Applications to protein modeling, Journal of Molecular Biology, 235(5), pp. 1501–1531, 1994.

[37] A. Krogh, B. Larsson, G. Von Heijne, and E. L. Sonnhammer, Predicting transmembrane protein topology with a hidden markov model: application to complete genomes, Journal of Molecular Biology, 305(3), pp. 567–580, 2001.

[38] F. Lagona, M. Picone, and A. Maruotti, A hidden markov model for the analysis of cylindrical time series, Environmetrics, 26(8), pp. 534–544, 2015.

[39] R. Langrock, Some applications of nonlinear and non-gaussian state space modelling by means of hidden markov model, Journal of Applied Statistics, 38(12), pp. 2955–2970, 2011.

[40] N. Lethanh, K. Kaito, and K. Kobayashi, Infrastructure deterioration prediction with a poisson hidden markov model on time series data, Journal of Infrastructure Systems, 21(3), p. 04014051, 2015.

[41] K. Li and Y. Fu, Arma-hmm: A new approach for early recognition of human activity, in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 1779–1782, IEEE, 2012.

[42] C. Liu, J. Wang, D. Xiao, and Q. Liang, Forecasting S&P 500 stock index using statistical learning models, Open Journal of Statistics, 6(6), pp. 1067–1075, 2016.

[43] B. Mor, S. Garhwal, and A. Kumar, A systematic review of hidden markov models and their applications., Archives of Computational Methods in Engineering, 28(3), 2021.

[44] H. Z. Muhammad, M. Nasrun, C. Setianingsih, and M. A. Murti, Speech recognition for english to indonesian translator using hidden markov model, in *2018 International Conference on Signals and Systems (ICSigSys)*, pp. 255–260, IEEE, 2018.

[45] O. Netzer, J. M. Lattin, and V. Srinivasan, A hidden markov model of customer relationship dynamics, Marketing Science, 27(2), pp. 185–204, 2008.

[46] Z. Oflaz, *Bivariate Hidden Markov Model to Capture the Dependency in Claim Estimate*, Ph.D. thesis, Middle East Technical University, Ankara, 2016.

[47] Z. N. Oflaz, C. Yozgatligil, and A. S. Selcuk-Kestel, Aggregate claim estimation using bivariate hidden markov model, ASTIN Bulletin: The Journal of the IAA, 49(1), pp. 189–215, 2019.

[48] L. Rabiner and B.-H. Juang, *Fundamentals of speech recognition*, Prentice-Hall, Inc., 1993.

[49] L. R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, Proceedings of the IEEE, 77(2), pp. 257–286, 1989.

[50] R. Rojas, *Neural networks: a systematic introduction*, Springer Science & Business Media, 2013.

[51] R. C. Rose and D. B. Paul, A hidden markov model based keyword recognition system, in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 129–132, IEEE, 1990.

[52] S. Ruder, An overview of gradient descent optimization algorithms, arXiv preprint arXiv:1609.04747, 2016.

[53] P. Sokólski and T. Rutkowski, Hybrid of neural networks and hidden markov models as a modern approach to speech recognition systems, Pomiary Automatyka Robotyka, 17, pp. 449–455, 2013.

[54] I. Stanculescu, C. K. Williams, and Y. Freer, Autoregressive hidden markov models for the early detection of neonatal sepsis, IEEE Journal of Biomedical and Health Informatics, 18(5), pp. 1560–1570, 2013.

[55] S. Steinschneider and C. Brown, A semiparametric multivariate, multisite weather generator with low-frequency variability for use in climate risk assessments, Water Resources Research, 49(11), pp. 7205–7220, 2013.

[56] S. S. Surwade and Y. Angal, Speech recognition using HMM/ANN hybrid model, International Journal on Recent and Innovation Trends in Computing and Communication, 3(6), pp. 4155–4157, 2015.

[57] X. Tang, Hybrid hidden markov model and artificial neural network for automatic speech recognition, in *2009 Pacific-Asia Conference on Circuits, Communications and Systems*, pp. 682–685, IEEE, 2009.

[58] X. Tang, H. Yao, Y. Sun, C. Aggarwal, P. Mitra, and S. Wang, Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values, in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5956–5963, 2020.

[59] M. Thyer and G. Kuczera, Modeling long-term persistence in hydroclimatic time series using a hidden state markov model, Water Resources Research, 36(11), pp. 3301–3310, 2000.

[60] A. Varga and R. K. Moore, Hidden markov model decomposition of speech and noise, in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 845–848, IEEE, 1990.

[61] H. Wang, C. Li, B. Gu, and W. Min, Does AI-based credit scoring improve financial inclusion? Evidence from online payday lending, in *40th International Conference on Information Systems, ICIS 2019*, Association for Information Systems, 1984.

[62] S. Wibisono, M. Anwar, A. Supriyanto, and I. Amin, Multivariate weather anomaly detection using DBSCAN clustering algorithm, in *Journal of Physics: Conference Series*, volume 1869, p. 012077, IOP Publishing, 2021.

[63] C.-L. Yang, Z.-X. Chen, and C.-Y. Yang, Sensor classification using convolutional neural network by encoding multivariate time series as two-dimensional colored images, Sensors, 20(1), p. 168, 2019.

[64] L. Yang, B. Widjaja, and R. Prasad, Application of hidden markov models for signature verification, Pattern Recognition, 28(2), pp. 161–170, 1995.

[65] X.-F. Zhuang and L.-W. Chan, Volatility forecasts in financial time series with HMM-GARCH models, in *International Conference on Intelligent Data Engineering and Automated Learning*, pp. 807–812, Springer, 2004.

# APPENDIX A

# ESTIMATED PARAMETERS OF SUBSAMPLES OF RNN-HMM APPLICATIONS

In the hybrid models proposed in this thesis, the initial parameters of the HMM section are determined in the form of formulas. These initial parameters converge to specific parameters for each subsample after the EM algorithm. Since there are 4 subsamples for each application, the details are presented in this section. The results of these converged parameters are presented for all variable number selections in the order of S&P 500, Nasdaq, and EUR/USD exchange rates.

Table A.1: Estimated Parameters of Each Subsample of Univariate RNN-HMM application to S&P 500 index

|  | 1st subsample | | | | | 2nd subsample | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\mu_{SP}$ | 0.0840 | 0.1349 | 0.1663 | 0.1861 | 0.2578 | 0.3138 | 0.3493 | 0.3774 | 0.4248 | 0.4798 |
| $\sigma_{SP}$ | 0.0159 | 0.0109 | 0.0067 | 0.0067 | 0.0368 | 0.0287 | 0.0074 | 0.0116 | 0.0147 | 0.0189 |
| $\Gamma$ | 0.9955 | 0.0045 | 0 | 0 | 0 | 0.9691 | 0.0309 | 0 | 0 | 0 |
|  | 0.0061 | 0.9816 | 0.0123 | 0 | 0 | 0.0153 | 0.9730 | 0.0117 | 0 | 0 |
|  | 0 | 0.0129 | 0.947 | 0.0398 | 0 | 0 | 0.0135 | 0.9778 | 0.0088 | 0 |
|  | 0 | 0 | 0.0247 | 0.9714 | 0.0040 | 0 | 0 | 0.0094 | 0.9831 | 0.0008 |
|  | 0 | 0 | 0 | 0.0088 | 0.9912 | 0 | 0 | 0 | 0.0009 | 0.9909 |
| $\delta$ | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
|  | 3rd subsample | | | | | 4th subsample | | | | |
| $\mu_{SP}$ | 0.2030 | 0.3197 | 0.3898 | 0.4549 | 0.4906 | 0.5685 | 0.7335 | 0.8105 | 0.8755 | 0.9309 |
| $\sigma_{SP}$ | 0.0382 | 0.0195 | 0.0201 | 0.0125 | 0.0157 | 0.0593 | 0.0270 | 0.0175 | 0.0160 | 0.0225 |
| $\Gamma$ | 0.9953 | 0.0047 | 0 | 0 | 0 | 0.9958 | 0.0042 | 0 | 0 | 0 |
|  | 0 | 0.9915 | 0.0085 | 0 | 0 | 0 | 0.9858 | 0.0014 | 0 | 0 |
|  | 0 | 0.0057 | 0.9718 | 0.0225 | 0 | 0 | 0.0053 | 0.9748 | 0.0198 | 0 |
|  | 0 | 0 | 0.0162 | 0.9598 | 0.0240 | 0 | 0 | 0.0200 | 0.9448 | 0.0352 |
|  | 0 | 0 | 0 | 0.0212 | 0.9788 | 0 | 0 | 0 | 0.0117 | 0.9883 |
| $\delta$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Table A.2: Estimated Parameters of Each Subsample of Univariate RNN-HMM application to Nasdaq index

|  | 1st subsample | | | | | 2nd subsample | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\mu_{N}$ | 0.0932 | 0.2134 | 0.2857 | 0.3442 | 0.0721 | 0.1972 | 0.2275 | 0.2581 | 0.2877 | 0.3302 |
| $\sigma_{N}$ | 0.0337 | 0.0245 | 0.0171 | 0.0282 | 0.0201 | 0.0170 | 0.0084 | 0.0086 | 0.0105 | 0.0185 |
| $\Gamma$ | 0.9922 | 0.0078 | 0 | 0 | 0 | 0.9699 | 0.0300 | 0 | 0 | 0 |
|  | 0.0119 | 0.9581 | 0.0300 | 0 | 0 | 0.0147 | 0.9739 | 0.0114 | 0 | 0 |
|  | 0 | 0.0203 | 0.9578 | 0.0219 | 0 | 0 | 0.0115 | 0.9696 | 0.0192 | 0 |
|  | 0 | 0 | 0.0292 | 0.9708 | 0 | 0 | 0 | 0.0264 | 0.9640 | 0.0097 |
|  | 0 | 0 | 0 | 0.0078 | 0.9922 | 0 | 0 | 0 | 0.0129 | 0.9871 |
| $\delta$ | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
|  | 3rd subsample | | | | | 4th subsample | | | | |
| $\mu_{N}$ | 0.1604 | 0.2795 | 0.3504 | 0.4058 | 0.4521 | 0.5118 | 0.7034 | 0.7702 | 0.8511 | 0.9355 |
| $\sigma$ | 0.0285 | 0.0220 | 0.0194 | 0.0129 | 0.0199 | 0.0626 | 0.0291 | 0.0162 | 0.0244 | 0.0311 |
| $\Gamma$ | 0.9948 | 0.0052 | 0 | 0 | 0 | 0.9958 | 0.0042 | 0 | 0 | 0 |
|  | 0 | 0.9919 | 0.0081 | 0 | 0 | 0 | 0.9858 | 0.0014 | 0 | 0 |
|  | 0 | 0.0068 | 0.9780 | 0.0152 | 0 | 0 | 0.0053 | 0.9748 | 0.0198 | 0 |
|  | 0 | 0 | 0.0068 | 0.9780 | 0.0156 | 0 | 0 | 0.0200 | 0.9448 | 0.0352 |
|  | 0 | 0 | 0 | 0.0089 | 0.9911 | 0 | 0 | 0 | 0.0117 | 0.9883 |
| $\delta$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Table A.3: Estimated Parameters of Each Subsample of Univariate RNN-HMM application to EUR/USD data

|  | 1st subsample | | | | | 2nd subsample | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\mu_{ex}$ | 0.0539 | 0.2886 | 0.4095 | 0.1719 | 0.1027 | 0.4169 | 0.6172 | 0.7833 | 0.5482 | 0.4988 |
| $\sigma_{ex}$ | 0.0181 | 0.0377 | 0.0342 | 0.0138 | 0.0161 | 0.0205 | 0.0210 | 0.0642 | 0.0136 | 0.0152 |
| $\Gamma$ | 0.9894 | 0 | 0 | 0 | 0.0106 | 0.9921 | 0 | 0 | 0 | 0.0079 |
|  | 0 | 0.9876 | 0.0124 | 0 | 0 | 0 | 0.9928 | 0.0072 | 0 | 0 |
|  | 0 | 0.0043 | 0.9957 | 0 | 0 | 0 | 0.0039 | 0.9961 | 0 | 0 |
|  | 0 | 0.0081 | 0 | 0.9919 | 0 | 0 | 0.0074 | 0 | 0.9776 | 0.0150 |
|  | 0.017 | 0 | 0 | 0.0071 | 0.9762 | 0.0042 | 0 | 0 | 0.0128 | 0.9830 |
| $\delta$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
|  | 3rd subsample | | | | | 4th subsample | | | | |
| $\mu_{ex}$ | 0.5502 | 0.6959 | 0.7546 | 0.6531 | 0.6091 | 0.3976 | 0.6406 | 0.6930 | 0.5775 | 0.4397 |
| $\sigma_{ex}$ | 0.0264 | 0.0167 | 0.0194 | 0.0139 | 0.0155 | 0.0174 | 0.0168 | 0.0113 | 0.0245 | 0.0133 |
| $\Gamma$ | 0.9702 | 0 | 0 | 0 | 0.0298 | 0.9880 | 0 | 0 | 0 | 0.0120 |
|  | 0 | 0.9630 | 0.0109 | 0.0260 | 0 | 0 | 0.9785 | 0.0048 | 0.0167 | 0 |
|  | 0 | 0.0127 | 0.9873 | 0 | 0 | 0 | 0.0059 | 0.9941 | 0 | 0 |
|  | 0 | 0.0252 | 0 | 0.9411 | 0.0337 | 0 | 0.0221 | 0 | 0.9717 | 0.0063 |
|  | 0.0227 | 0 | 0 | 0.0302 | 0.9470 | 0.0200 | 0 | 0 | 0.0302 | 0.9800 |
| $\delta$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

Table A.4: Estimated Parameters of Each Subsample of Bivariate RNN-HMM application to S&P 500 and Nasdaq index

|  | 1st subsample | | | | | 2nd subsample | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\mu_N$ | 0.0950 | 0.1741 | 0.2206 | 0.2974 | 0.4249 | 0.1502 | 0.1833 | 0.2097 | 0.2357 | 0.2766 |
| $\mu_{SP}$ | 0.0802 | 0.1561 | 0.1994 | 0.2740 | 0.4285 | 0.2463 | 0.2831 | 0.3116 | 0.3406 | 0.3862 |
| $\sigma_N$ | 0.0204 | 0.0186 | 0.0112 | 0.0366 | 0.0392 | 0.0128 | 0.0096 | 0.0080 | 0.0093 | 0.0164 |
| $\sigma_{SP}$ | 0.0186 | 0.0183 | 0.0114 | 0.0409 | 0.0461 | 0.0018 | 0.0088 | 0.0089 | 0.0129 | 0.0152 |
| $\Gamma$ | 0.9957 | 0.0043 | 0 | 0 | 0 | 0.9781 | 0.0219 | 0 | 0 | 0 |
|  | 0.0040 | 0.9717 | 0.0243 | 0 | 0 | 0.0180 | 0.9584 | 0.0236 | 0 | 0 |
|  | 0 | 0.0188 | 0.9780 | 0.0031 | 0 | 0 | 0.0185 | 0.9619 | 0.0195 | 0 |
|  | 0 | 0 | 0.0131 | 0.9869 | 0 | 0 | 0 | 0.0291 | 0.9621 | 0.0088 |
|  | 0 | 0 | 0 | 0.0158 | 0.9842 | 0 | 0 | 0 | 0.0216 | 0.9784 |
| $\delta$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
|  | 3rd subsample | | | | | 4th subsample | | | | |
| $\mu_N$ | 0.1614 | 0.2667 | 0.3358 | 0.3880 | 0.4358 | 0.4792 | 0.6953 | 0.8233 | 0.8948 | 0.9608 |
| $\mu_{SP}$ | 0.1922 | 0.2996 | 0.3672 | 0.4259 | 0.4745 | 0.5218 | 0.7344 | 0.8492 | 0.9175 | 0.9782 |
| $\sigma_N$ | 0.0239 | 0.0211 | 0.0185 | 0.0126 | 0.0179 | 0.0619 | 0.0391 | 0.0240 | 0.0216 | 0.0166 |
| $\sigma_{SP}$ | 0.0263 | 0.0214 | 0.0167 | 0.0145 | 0.0175 | 0.0608 | 0.0396 | 0.0226 | 0.0172 | 0.0115 |
| $\Gamma$ | 0.9953 | 0.0047 | 0 | 0 | 0 | 0.9948 | 0.0052 | 0 | 0 | 0 |
|  | 0 | 0.9915 | 0.0085 | 0 | 0 | 0 | 0.9921 | 0.0079 | 0 | 0 |
|  | 0 | 0.0057 | 0.9718 | 0.0225 | 0 | 0 | 0.0073 | 0.9696 | 0.0230 | 0 |
|  | 0 | 0 | 0.0162 | 0.9598 | 0.0240 | 0 | 0 | 0.0092 | 0.9750 | 0.0158 |
|  | 0 | 0 | 0 | 0.0212 | 0.9788 | 0 | 0 | 0 | 0.0181 | 0.9819 |
| $\delta$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Table A.5: Estimated Parameters of Each Subsample of Bivariate RNN-HMM application to S&P 500 and EUR/USD

**1st subsample**

$$\begin{aligned}
\mu_{SP} &= \begin{pmatrix} 0.2303 & 0.1806 & 0.1329 & 0.0752 & 0.3256 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.0770 & 0.1117 & 0.4230 & 0.2296 & 0.0791 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{SP} &= \begin{pmatrix} 0.0150 & 0.0190 & 0.0237 & 0.0197 & 0.0455 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0149 & 0.0188 & 0.0527 & 0.0513 & 0.0358 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9826 & 0.0116 & 0 & 0 & 0.0058 \\
0.0133 & 0.9738 & 0 & 0.0129 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0.0045 & 0.9955 & 0 \\
0.0097 & 0 & 0 & 0 & 0.9903
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

**2nd subsample**

$$\begin{aligned}
\mu_{SP} &= \begin{pmatrix} 0.2927 & 0.3840 & 0.3064 & 0.2574 & 0.3663 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.4688 & 0.7526 & 0.8435 & 0.5450 & 0.6196 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{SP} &= \begin{pmatrix} 0.0151 & 0.0227 & 0.0228 & 0.0175 & 0.0293 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0363 & 0.0309 & 0.0587 & 0.0454 & 0.0367 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9887 & 0 & 0 & 0.0075 & 0.0037 \\
0 & 0.9875 & 0.0125 & 0 & 0 \\
0 & 0 & 0.9945 & 0.0056 & 0 \\
0.0107 & 0 & 0 & 0.9893 & 0 \\
0 & 0.0047 & 0 & 0 & 0.9953
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

**3rd subsample**

$$\begin{aligned}
\mu_{SP} &= \begin{pmatrix} 0.2556 & 0.4047 & 0.3021 & 0.2424 & 0.4920 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.5585 & 0.7157 & 0.7115 & 0.6717 & 0.6135 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{SP} &= \begin{pmatrix} 0.0331 & 0.0157 & 0.0323 & 0.0080 & 0.0207 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0311 & 0.0534 & 0.0583 & 0.0395 & 0.0372 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9775 & 0 & 0.0056 & 0.0169 & 0 \\
0 & 0.9907 & 0.0047 & 0 & 0.0046 \\
0.0035 & 0.0071 & 0.9894 & 0 & 0 \\
0.0237 & 0 & 0.0078 & 0.9685 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

**4th subsample**

$$\begin{aligned}
\mu_{SP} &= \begin{pmatrix} 0.9310 & 0.8044 & 0.7054 & 0.5509 & 0.8585 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.4238 & 0.5955 & 0.7010 & 0.6318 & 0.5126 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{SP} &= \begin{pmatrix} 0.0451 & 0.0366 & 0.0348 & 0.0542 & 0.0103 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0254 & 0.0263 & 0.0182 & 0.0258 & 0.0274 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0.9882 & 0 & 0 & 0.1184 \\
0 & 0.0043 & 0.99957 & 0 & 0 \\
0 & 0 & 0.0047 & 0.9953 & 0 \\
0.0444 & 0 & 0 & 0.0181 & 0.9556
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Table A.6: Estimated Parameters of Each Subsample of Bivariate RNN-HMM application to Nasdaq and EUR/USD

**1st subsample**

$$\begin{aligned}
\mu_{N} &= \begin{pmatrix} 0.1647 & 0.2903 & 0.1247 & 0.0581 & 0.4994 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.0493 & 0.0821 & 0.3949 & 0.1841 & 0.0031 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{N} &= \begin{pmatrix} 0.0380 & 0.0464 & 0.0166 & 0.0127 & 0.0985 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0250 & 0.0179 & 0.0526 & 0.0461 & 0.0078 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9931 & 0 & 0 & 0.0069 & 0 \\
0.0177 & 0.9823 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0.0041 & 0 & 0.0041 & 0.9919 & 0 \\
0 & 0.0124 & 0 & 0 & 0.9876
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

**2nd subsample**

$$\begin{aligned}
\mu_{N} &= \begin{pmatrix} 0.2477 & 0.2680 & 0.2411 & 0.1973 & 0.3460 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.3960 & 0.6383 & 0.7915 & 0.5354 & 0.4030 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{N} &= \begin{pmatrix} 0.0169 & 0.0150 & 0.0125 & 0.0261 & 0.0235 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0222 & 0.0993 & 0.0794 & 0.0524 & 0.0606 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9905 & 0 & 0 & 0.0095 & 0.0037 \\
0 & 0.9651 & 0.0176 & 0.0174 & 0 \\
0 & 0.0070 & 0.9896 & 0.0035 & 0 \\
0.0059 & 0.0030 & 0.0029 & 0.9882 & 0 \\
0 & 0.0148 & 0 & 0 & 0.9852
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

**3rd subsample**

$$\begin{aligned}
\mu_{N} &= \begin{pmatrix} 0.1816 & 0.2964 & 0.2249 & 0.1809 & 0.3687 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.5552 & 0.7456 & 0.7573 & 0.6512 & 0.6985 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{N} &= \begin{pmatrix} 0.0471 & 0.0204 & 0.0149 & 0.0147 & 0.0196 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0220 & 0.0661 & 0.0455 & 0.0332 & 0.0386 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9706 & 0 & 0 & 0.0294 & 0 \\
0.0031 & 0.9874 & 0.0064 & 0 & 0.0031 \\
0 & 0.0019 & 0.9805 & 0 & 0 \\
0.0136 & 0 & 0.0112 & 0.9752 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

**4th subsample**

$$\begin{aligned}
\mu_{N} &= \begin{pmatrix} 0.7944 & 0.8779 & 0.7052 & 0.5034 & 0.9721 \end{pmatrix} \\
\mu_{ex} &= \begin{pmatrix} 0.4416 & 0.4044 & 0.6313 & 0.7069 & 0.3553 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\sigma_{N} &= \begin{pmatrix} 0.0247 & 0.0220 & 0.0358 & 0.0987 & 0.0189 \end{pmatrix} \\
\sigma_{ex} &= \begin{pmatrix} 0.0408 & 0.0282 & 0.0379 & 0.0400 & 0.0220 \end{pmatrix}
\end{aligned}$$

$$\Gamma = \begin{pmatrix}
0.9738 & 0.0262 & 0 & 0 & 0 \\
0.0071 & 0.9893 & 0 & 0 & 0.0035 \\
0.0068 & 0.0043 & 0.9932 & 0 & 0 \\
0 & 0 & 0.0026 & 0.9974 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

$$\delta = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Table A.7: Estimated Parameters of Each Subsample of Trivariate RNN-HMM application

**1st subsample**

$$
\begin{matrix}
\mu_N \\ \mu_{SP} \\ \mu_{ex}
\end{matrix}
\begin{pmatrix}
0.1785 & 0.2870 & 0.1316 & 0.0531 & 0.5282 \\
0.1915 & 0.2987 & 0.1837 & 0.0785 & 0.5451 \\
0.0774 & 0.0446 & 0.2911 & 0.1759 & -0.0003
\end{pmatrix}
$$

$$
\begin{matrix}
\sigma_N \\ \sigma_{SP} \\ \sigma_{ex}
\end{matrix}
\begin{pmatrix}
0.0284 & 0.0575 & 0.0217 & 0.0209 & 0.0602 \\
0.0268 & 0.0603 & 0.0277 & 0.0225 & 0.0644 \\
0.0159 & 0.0180 & 0.369 & 0.0430 & 0.0001
\end{pmatrix}
$$

$$
\Gamma
\begin{pmatrix}
0.9886 & 0.0208 & 0 & 0.0107 & 0 \\
0.0100 & 0.9900 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0.0042 & 0.9958 & 0 \\
0 & 0.0145 & 0 & 0 & 0.9855
\end{pmatrix}
$$

$$
\delta \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \end{pmatrix}
$$

**2nd subsample**

$$
\begin{matrix}
\mu_N \\ \mu_{SP} \\ \mu_{ex}
\end{matrix}
\begin{pmatrix}
0.2349 & 0.2891 & 0.2846 & 0.2127 & 0.3313 \\
0.2670 & 0.3615 & 0.3195 & 0.2878 & 0.4070 \\
0.3697 & 0.4969 & 0.3259 & 0.6683 & 0.5398
\end{pmatrix}
$$

$$
\begin{matrix}
\sigma_N \\ \sigma_{SP} \\ \sigma_{ex}
\end{matrix}
\begin{pmatrix}
0.0222 & 0.0144 & 0.0124 & 0.0348 & 0.0141 \\
0.0195 & 0.0155 & 0.0116 & 0.0425 & 0.0148 \\
0.0404 & 0.0616 & 0.0368 & 0.0476 & 0.0284
\end{pmatrix}
$$

$$
\Gamma
\begin{pmatrix}
0.9920 & 0 & 0 & 0.0080 & 0 \\
0 & 0.9631 & 0 & 0.0057 & 0.0312 \\
0.0125 & 0.0062 & 0.9813 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0.0476 & 0 & 0 & 0.9524
\end{pmatrix}
$$

$$
\delta \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}
$$

**3rd subsample**

$$
\begin{matrix}
\mu_N \\ \mu_{SP} \\ \mu_{ex}
\end{matrix}
\begin{pmatrix}
0.2177 & 0.2976 & 0.1963 & 0.0852 & 0.3678 \\
0.2513 & 0.3476 & 0.2347 & 0.0906 & 0.4142 \\
0.5044 & 0.6152 & 0.6099 & 0.5347 & 0.5947
\end{pmatrix}
$$

$$
\begin{matrix}
\sigma_N \\ \sigma_{SP} \\ \sigma_{ex}
\end{matrix}
\begin{pmatrix}
0.0256 & 0.0163 & 0.0404 & 0.0264 & 0.0228 \\
0.0280 & 0.0195 & 0.0458 & 0.0322 & 0.0254 \\
0.0372 & 0.0446 & 0.0308 & 0.0424 & 0.0279
\end{pmatrix}
$$

$$
\Gamma
\begin{pmatrix}
0.9895 & 0.0053 & 0.0053 & 0 & 0 \\
0 & 0.9869 & 0.0087 & 0 & 0.0044 \\
0.0097 & 0.0097 & 0.9806 & 0 & 0 \\
0 & 0 & 0.0052 & 0.9948 & 0 \\
0 & 0 & 0 & 0 & 1
\end{pmatrix}
$$

$$
\delta \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}
$$

**4th subsample**

$$
\begin{matrix}
\mu_N \\ \mu_{SP} \\ \mu_{ex}
\end{matrix}
\begin{pmatrix}
0.8165 & 0.8576 & 0.6824 & 0.4578 & 0.9546 \\
0.8181 & 0.8752 & 0.7343 & 0.5120 & 0.9601 \\
0.3766 & 0.4286 & 0.5723 & 0.5719 & 0.3763
\end{pmatrix}
$$

$$
\begin{matrix}
\sigma_N \\ \sigma_{SP} \\ \sigma_{ex}
\end{matrix}
\begin{pmatrix}
0.0241 & 0.0310 & 0.0575 & 0.0635 & 0.0344 \\
0.0243 & 0.0241 & 0.0539 & 0.0662 & 0.0326 \\
0.0114 & 0.0394 & 0.0435 & 0.0276 & 0.0250
\end{pmatrix}
$$

$$
\Gamma
\begin{pmatrix}
0.9719 & 0.0281 & 0 & 0 & 0 \\
0.0065 & 0.9616 & 0 & 0 & 0.0320 \\
0 & 0.0038 & 0.9962 & 0 & 0 \\
0 & 0 & 0.0042 & 0.9958 & 0 \\
0 & 0.0119 & 0 & 0 & 0.9881
\end{pmatrix}
$$

$$
\delta \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \end{pmatrix}
$$

# APPENDIX B

# GLOBAL DECODING OF HMM APPLICATIONS

Global decoding is performed to determine the best sequences of hidden states. Viterbi algorithm explained in Section 2.1.2 is used for this purpose. The states are computed by using Equation (2.17). The hidden states decoding of HMM applications are presented in this section.
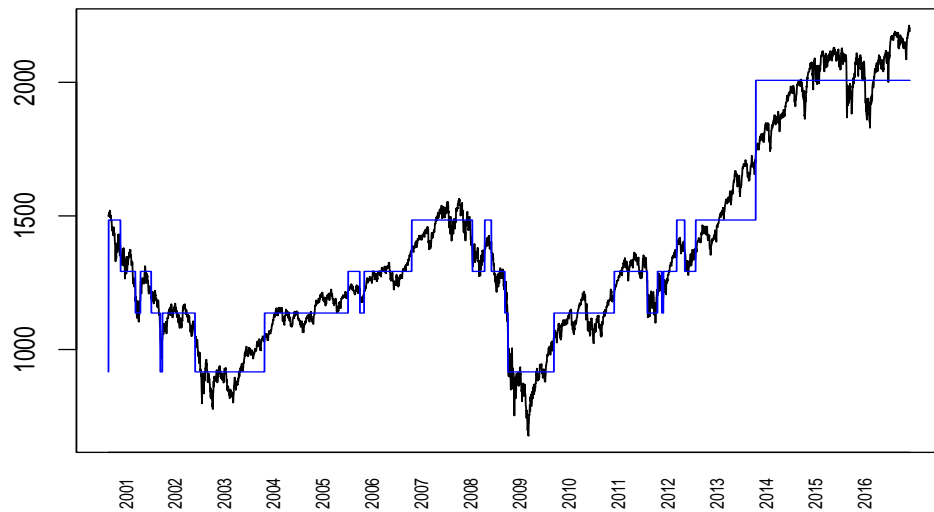


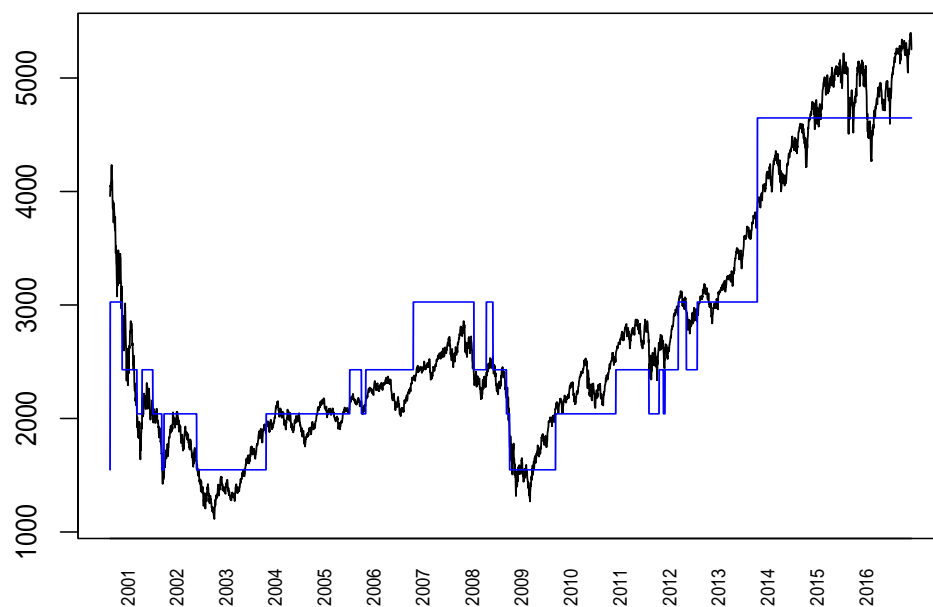Figure B.1: Global Decoding of Univariate S&P 500 HMM

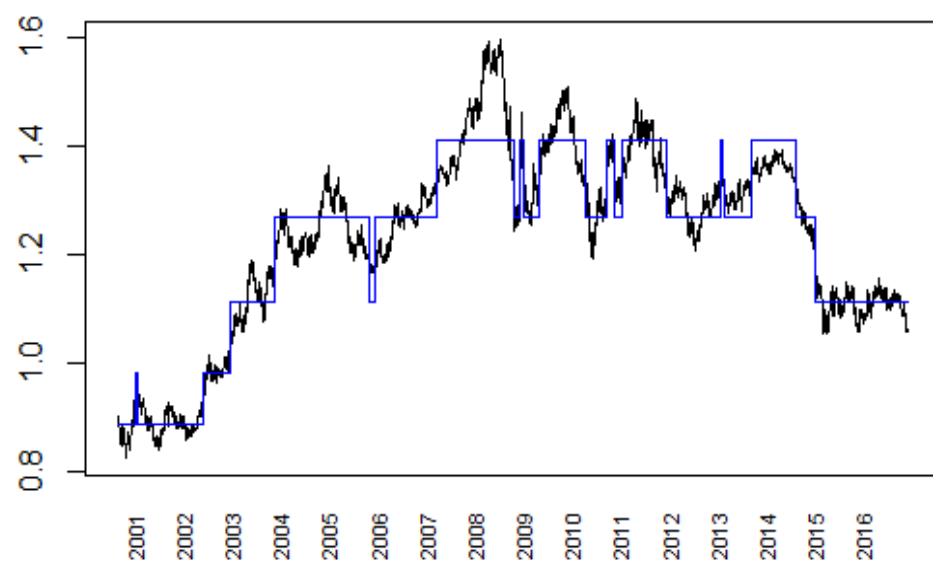Figure B.2: Global Decoding of Univariate Nasdaq HMM
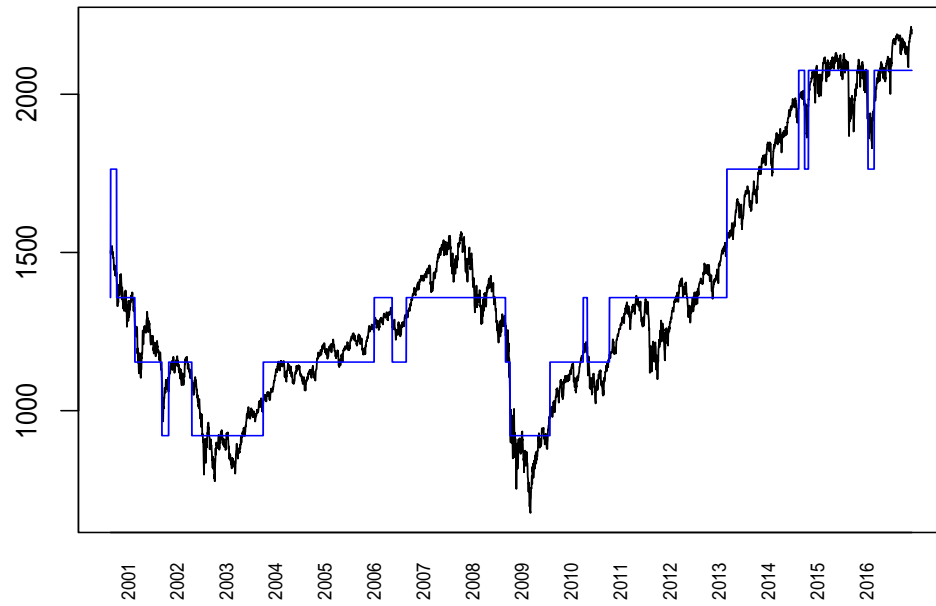


Figure B.3: Global Decoding of Univariate EUR/USD HMM

Figure B.4: Global Decoding of S&P 500 for Bivariate S&P 500 and Nasdaq HMM
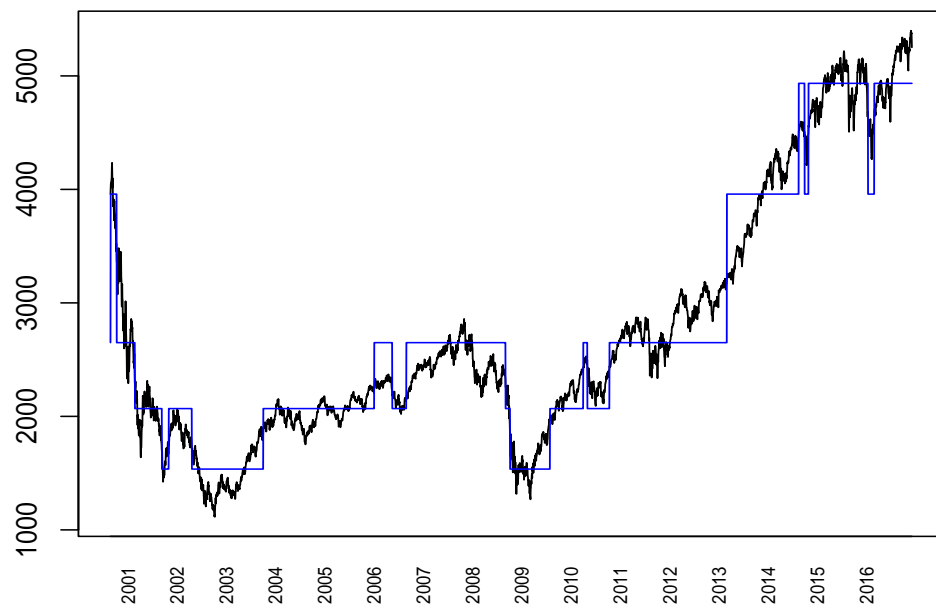


Figure B.5: Global Decoding of Nasdaq for Bivariate S&P 500 and Nasdaq HMM
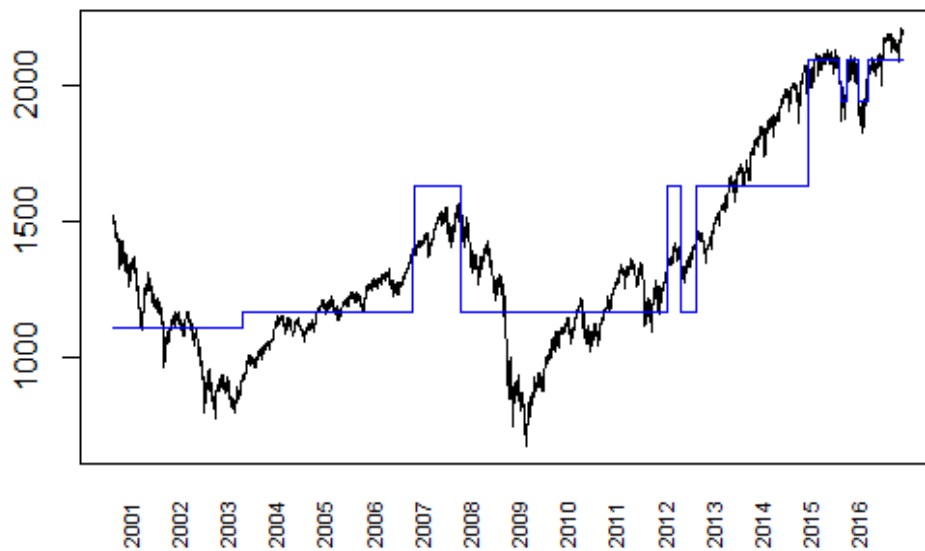
Figure B.6: Global Decoding of S&P 500 for Bivariate S&P 500 and EUR/USD HMM
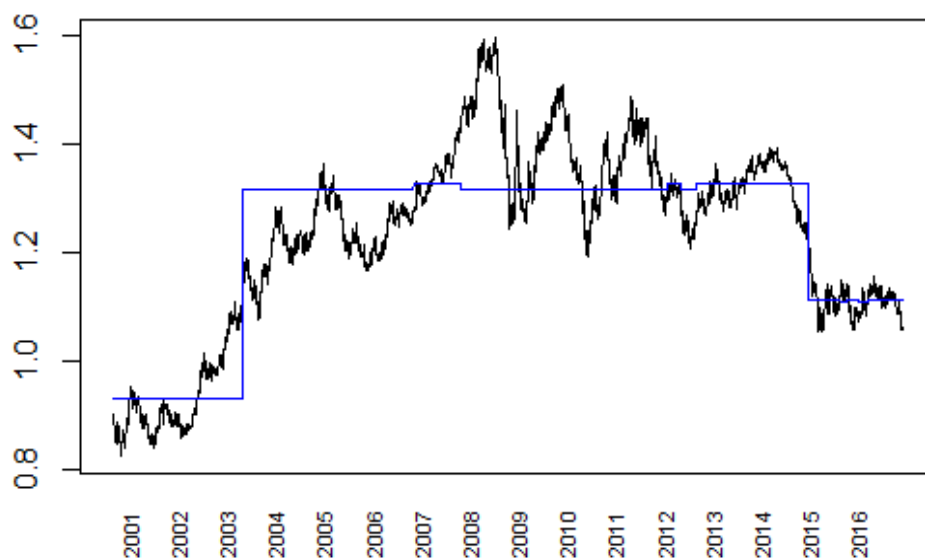


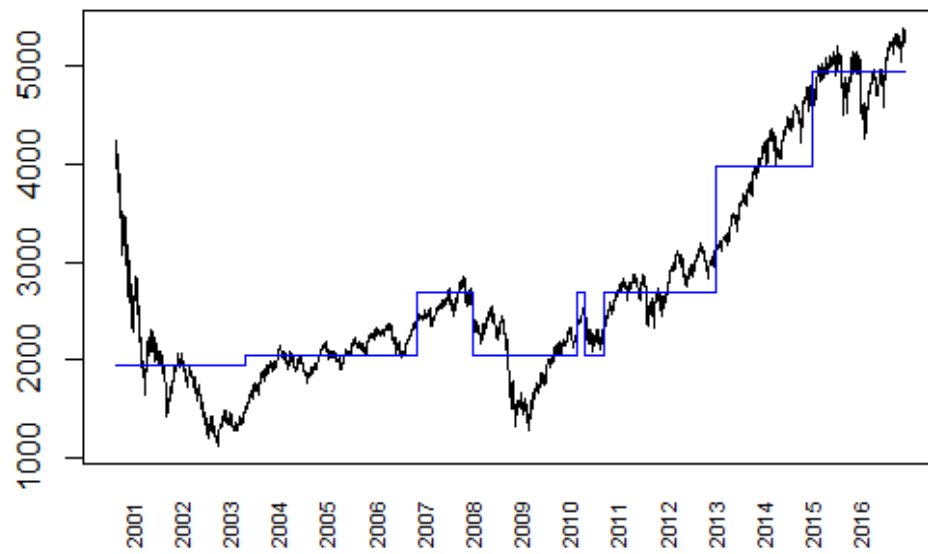Figure B.7: Global Decoding of EUR/USD for Bivariate S&P 500 and EUR/USD HMM

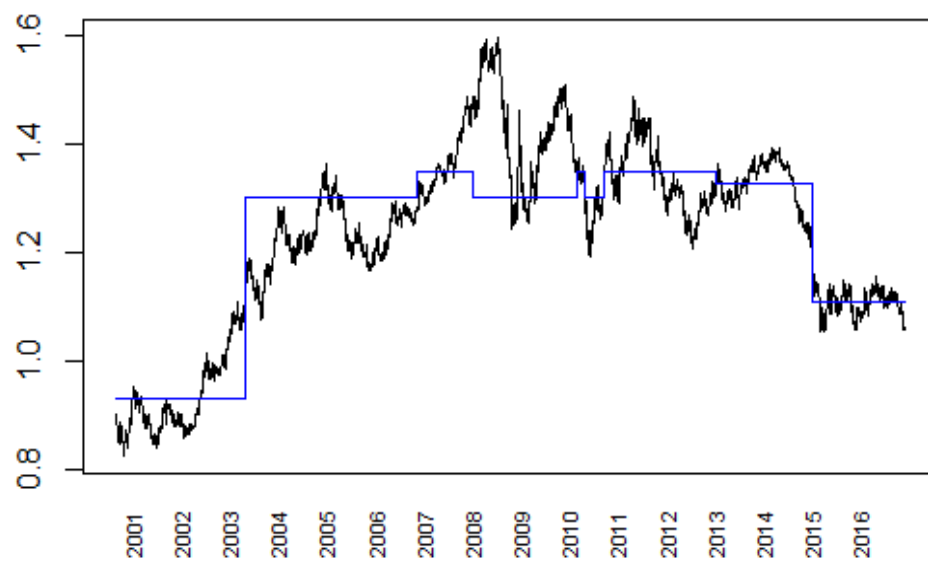Figure B.8: Global Decoding of Nasdaq for Bivariate Nasdaq and EUR/USD HMM



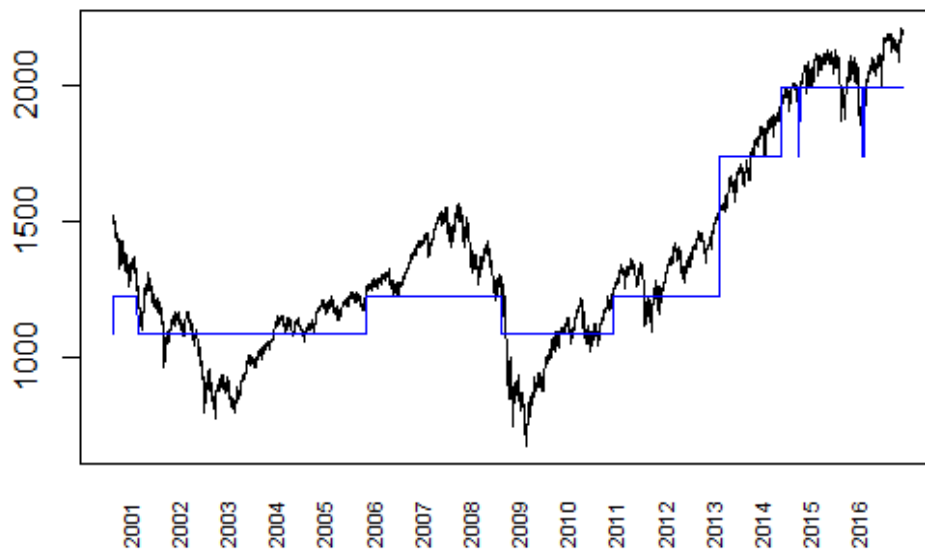Figure B.9: Global Decoding of EUR/USD for Bivariate Nasdaq and EUR/USD HMM

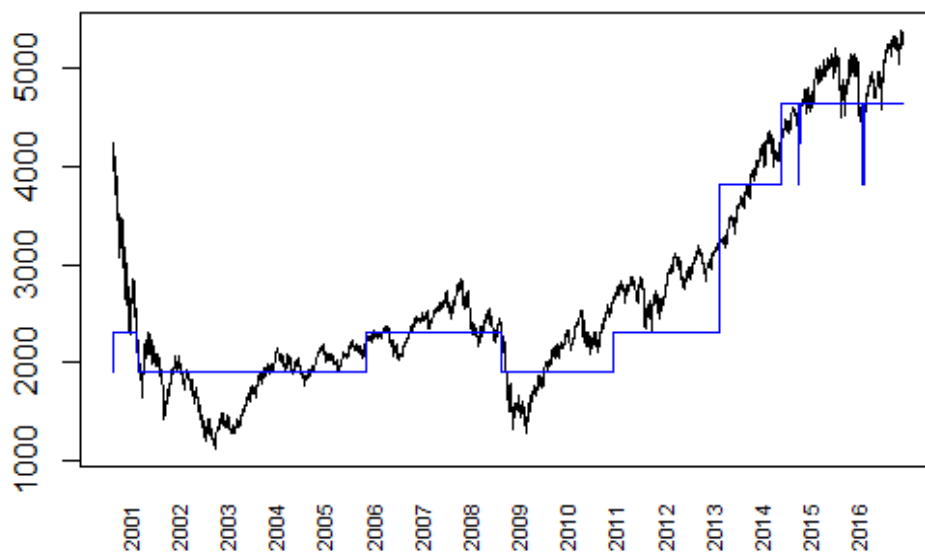Figure B.10: Global Decoding of S&P 500 for Trivariate HMM



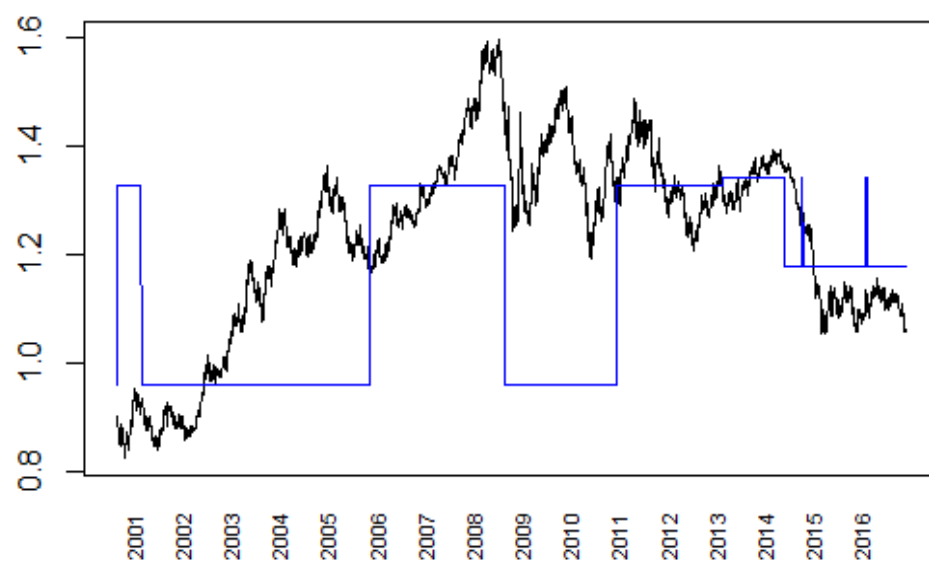Figure B.11: Global Decoding of Nasdaq for Trivariate HMM

Figure B.12: Global Decoding of EUR/USD for Trivariate HMM

# CURRICULUM VITAE

## PERSONAL INFORMATION

**Surname, Name:** Aydoğan Kılıç, Dilek
**Nationality:** Turkish (TC)
**Date and Place of Birth:** XXXX, XX
**Marital Status:** Married

## EDUCATION

| Degree | Institution | Year of Graduation |
|---|---|---|
| M.S. | Financial Mathematics, Middle East Technical University | 2015 |
| B.S. | Mathematics, Middle East Technical University | 2012 |
| High School | Milli Piyango Anatolian High School | 2007 |

## PROFESSIONAL EXPERIENCE

| Year | Place | Enrollment |
|---|---|---|
| 2014- | Ministry of Health, Turkey | Health Expert |

## PUBLICATIONS

Performansa Dayalı Ödeme Sistemi: Terminoloji, Çeşitli Ülkelerde Uygulama Örnekleri, Türkiye Durumu, Uzmanlık Tezi, 2017

Determination of Inflation Rate in a Hidden Markov Model Framework: Turkey Case, MSc. Thesis, 2015

Health Statistics Yearbook 2015, Ministry of Health, Turkey

Health Statistics Yearbook 2016, Ministry of Health, Turkey