Hidden Markov Models for Time Series Analysis

Yiliu Cao

2025-03-25

1 Introduction

In probability theory and statistics, a Markov Chain is a stochastic process that models a sequence of random variables in which the distribution of the next state depends only on the current state or a limited number of past observations, and we call this structure a Markov process. This model was first introduced by Andrey Markov in the early 20th century (Rabiner, 1989). Building on this idea, Leonard E. Baum introduced the Hidden Markov Model (HMM), which extends the Markov chain framework by having two sequences of random variables, one to be hidden and another to be observed, with hidden variables modeling as the Markov process(Baum & Petrie, 1966). The distribution of the hidden variable at time t only depends on the hidden variable at time $t-1$, with an observed variable that depends on the hidden variable at the same time t. Since the hidden states can not be observed directly, the study of HMMs mainly focuses on two key tasks. Firstly, we want to compute the most likely sequence of hidden states given the observed data. Secondly, we want to estimate the parameters from the HMMs, such as transition distribution. This paper will discuss how to achieve the two goals and illustrate the relationship to the Time Series Analysis.

The Hidden Markov Models was firstly proposed by Baum through a series of papers between the 1960s and 1970s. The initial settings of HMMs follows that, suppose $\{X_t\}$ with stochatic matrix $\{a_{ij}\}$ is a s-state Markov Process and define $\{Y_t\}$ as a probabilistic function of $\{X_t\}$ (Baum & Petrie, 1966). The conditional probability of Y_t given all the past values of $\{X_t\}$ and $\{Y_t\}$ up to time t is defined as b_{jk}. In their first paper, they proved the consistency of the likelihood of the $\{Y_t\}$ and the smoothness property of the expectation of the log-likelihood of $\{Y_t\}$ wrt to the parameter $\boldsymbol{\theta}$ from $\{a_{ij}\}$ and $\{b_{ij}\}$ (Baum & Petrie, 1966). In their later paper in the 1970s, they further introduced an iterative method for estimating the model parameters and proved that the algorithm converges to a local maximum (Baum & Petrie, 1966). This method later became known as the Baum-Welch algorithm, which is the EM Algorithm in HMMs (Zucchini et al., 2016). The parameters $\{a_{ij}\}$ and $\{b_{ij}\}$ are now commonly referred to as the transition distribution and emission distribution, respectively (Zucchini et al., 2016).

One of the popular applications of HMMs in the Time Series Analysis is GARCH-HMMs or Regime-Switching GARCH (Cai, 1994). It captures the volatility process changes, which may depend on some unobserved regimes, and we assume the changes of hidden states following a Markov process. In this hybrid model, the HMMs divide the time series with different volatility levels and then use the GARCH model to model the time-varying variance within each regime (Zhuang & Chan, 2004). For example, in financial time series, this allows the model to shift between high and low volatility periods, such as market booms and crashes. We can, therefore, predict the state the next time and choose the corresponding local GARCH model to predict the volatility. This model is advantageous as it allows us to see how volatility changes between different levels and specific GARCH for each level.

Another significant development in HMMs is Automatic Speech Recognition (ASR). It is based on HMMs and became well-known by IBM and Bell Lab in the 1980s (Bahl et al., 1983). It has been considered an important tool to enhance the performance of human-human and human-machine communications. A famous one is Apple's Siri, which can be acted as a virtual assistant. The basic algorithm for HMMs in ASR is to set hidden states to be the phonetic characters and use the properties of HMMs to identify the true sentence that the speaker is saying. It can even be used to predict what the speaker is trying to say given what he already said. Moreover, one of the most recent studies of ASR uses Deep learning methods such as Deep Neural Networks to construct multiple hidden layers to recognize speech, which are the Deep Neural Networks Markov Models for ASR (Yu & Deng, 2015).

This paper is structured as follows. Section 2 will briefly review the Markov Chain, along with some properties. Then, Section 3 will introduce the Hidden Markov Model, where we will discuss its definitions and properties. In addition, Section 4 will talk about the Baum-Welch Algorithm, which is the application of the EM algorithm in HMMs. We will also discuss Viterbi Algorithm as well. Section 5 will discuss HMM-GARCH. After that, we will apply this algorithm to real-world scenarios and compare their performance on Section 6. We will also provide some discussion and conclusion in Section 7.

2 Markov Chains

2.1 Basics

Markov Chains models the structure of dependence for a sequence of variables. Let $\{X_t : t \in \mathbb{N}\}$ be a sequence of discrete random variables with values being one *state* from the state space $\mathbf{S} = \{1, \ldots, m\}$ s.t. $|\mathbf{S}| = m$, then such a sequence of variables is called a **Markov Chain (MC)** if, for all $t \in \mathbb{N}$, it satisfies the Markov Property such that

$$P(X_{t+1}|X_t, \ldots, X_1) = P(X_{t+1}|X_t)$$

The Markov Property says that the probability of a random variable in a MC only depends on the most recent one. With this property, the Markov Chain can be visualized as

$$\begin{tikzcd}
\cdots \arrow[r] & X_{t-1} \arrow[r] & X_t \arrow[r] & X_{t+1} \arrow[r] & X_{t+2} \arrow[r] & \cdots
\end{tikzcd}$$

Figure 1: First Order Markov Chain

Since each variable is discrete, the probability of transitioning from state i to state j at two consecutive time points is called the **transition probability**, denoted by

$$\gamma_{ij} = P(X_{t+1} = j|X_t = i)$$

If γ_{ij} does not change across time, $p(X_{t+1} = j|X_t = i) = p(X_{t+2} = j|X_{t+1} = i)\ \forall t \in \mathbb{N}$, then such markov chains are said to be **homogeneous (stationary)**. The matrix $\mathbf{\Gamma}(1) = \mathbf{\Gamma}$ is defined as the transition matrix with $(i,j)^{th}$ element as γ_{ij}. More generally, if the time points are not consecutive, the transition probability over t time steps is given by $\gamma_{ij}(t) = P(X_{s+t} = j|X_s = i)$ with transition matrix $\mathbf{\Gamma}(t)$. It can be shown that

$$\mathbf{\Gamma}(t+u) = \mathbf{\Gamma}(t)\mathbf{\Gamma}(u)$$

which is so-called the **Champman-Kolmogorov equations**. (Appendix).

2.2 Stationary Distribution

The unconditional probabilities $u_j(t) = p(X_t = j)$ is the probability of being in the state j at time t. It can be expressed as the row vector $\mathbf{u}(t)$ where

$$\mathbf{u}(t) = (p(X_t = 1), \ldots, p(X_t = m)) = (u_1(t), \ldots, u_m(t)) \in \mathbb{R}^m$$

It can be shown that $\mathbf{u}(t+1) = \mathbf{u}(t)\mathbf{\Gamma}$. If the Markov Chain runs for a long time, then we would expect that, starting from some point, the Markov Chain will reach out to a steady state. This means that the subsequent time steps have the same distribution, and we call the distribution at the steady state as the **stationary distribution** of Markov Chain, denoted by $\boldsymbol{\delta}$. Mathematically, $\boldsymbol{\delta}$ satisfies

$$\boldsymbol{\delta} = \boldsymbol{\delta}\mathbf{\Gamma}$$

Moreover, it is easy to see that $\boldsymbol{\delta}$ is the eigenvector of $\mathbf{\Gamma}$ with eigenvalue 1. This also implies that the stationary distribution $\boldsymbol{\delta}$ may not be unique. In fact, if a markov chain is irreducible (homogeneous, discrete-time, finite state space), then such markov chains has a unique, strictly positive stationary distribution.

In addition, a Markov Chain is said to be **stationary** if it starts at its stationary distribution and continue to have that distribution to all the subsequent time points, implying $\boldsymbol{\delta} = \mathbf{u}(1) = \cdots = \mathbf{u}(T)$. This is a stronger condition and usually to achieve in practice as it requires to know the exact transition matrix. Besides, the Markov Chain we observed in practice is something that is already running, and we do not have the ability to force it to have the initial state by $\boldsymbol{\delta}$.

It is important to distinguish between time-homogeneous and stationary Markov Chain. The time-homogeneous Markov Chain says the conditional distribution of hidden variables does not change over time. If a Markov Chain is time-homogeneous and $\mathbf{u}(1)$ is stationary distribution, then this Markov Chain is indeed stationary.

3 Hidden Markov Models

Hidden Markov Models HMMs is a particular structure of dependence extending from Markov Chain. Besides the observed variables $\{X_t : t \in \mathbb{N}\}$, there is an extra sequence of hidden variables $\{Z_t : t \in \mathbb{N}\}$ modelling by Markov Chain and is linked to the observed variables. More specifically, define $\mathbf{Z}^{(t)}$ and $\mathbf{X}^{(t)}$ representing the histories for hidden and observed variables up to time t, the basic HMM follows

$$p\left(Z_t \mid \mathbf{Z}^{(t-1)}\right) = p\left(Z_t \mid Z_{t-1}\right), \quad t \in \{2, 3 \ldots\}$$
$$p\left(X_t \mid \mathbf{X}^{(t-1)}, \mathbf{Z}^{(t)}\right) = \operatorname{Pr}\left(X_t \mid Z_t\right), \quad t \in \mathbb{N}$$

A visualization of Basic Hidden Markov Model is shown on Figure 2. Each hidden variable only depends on the previous hidden variable, and the observed variable only depends on the hidden variable at the same time t.

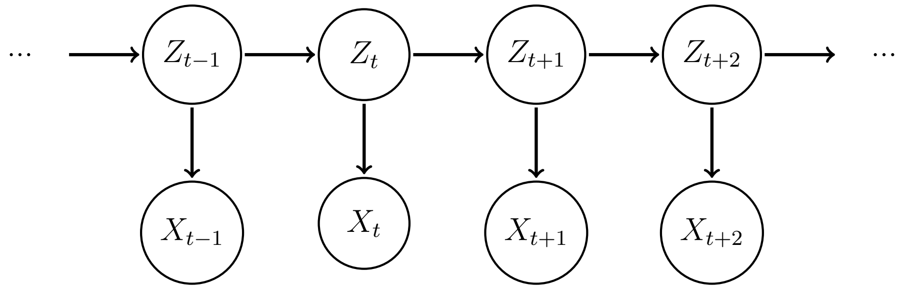


Figure 2: Basic Hidden Markov Model

All the properties of Markov Chain introduced on the Section 2 are now applied to the hidden variables $\{Z_t : t \in \mathbb{N}\}$ instead of the observed variables. Moreover, $\{X_t : t \in \mathbb{N}\}$ is called an m-state HMM if $|\mathbf{S}| = m$. In addition, since the observed variables only depends on the corresponding hidden variable, such conditional probability is called the **Emission (state-dependent) distribution** $p_i(x)$ such that

$$p_i(x) = p(X_t = x|Z_t = i)$$

Intuitively, the emission distribution is distribution of the observed variable that the hidden variable "emits" to it. Therefore, there are m emission distributions, corresponding to each state of the hidden states, whcih does not depend on t.

3.1 Marginal Distribution and Likelihood

To estimate the marginal distribution of the observed variables, the method we take is the marginalization that marginalize the hidden variable. Suppose in bivariate cases we have four variables $X_t, X_{t+k}, Z_t, Z_{t+k}$, the joint distribution of $p(X_t = a, X_{t+k} = b)$ can be expressed as

$$p(X_t = a, X_{t+k} = b) = \sum_{i}^{m}\sum_{j}^{m} p(X_t = a, X_{t+k} = b, Z_t = i, Z_{t+k} = j)$$

By the properties of HMMs, Z_{t+k} only depends on Z_t as we can marginalize the intermediate nodes out, the joint distribution of the four variables can be expressed as $p(Z_t = i)p(X_t = a|Z_t = i)p(Z_{t+k} = j|Z_t = i)p(X_{t+k} = b|Z_{t+k} = j)$. Therefore, the joint distribution of incomplete data can be written as

$$\begin{aligned}p(X_t = a, X_{t+k} = b) &= \sum_{i}^{m}\sum_{j}^{m} p(Z_t = i)p(X_t = a \mid Z_t = i)p(Z_{t+k} = j \mid Z_t = i)p(X_{t+k} = b \mid Z_{t+k} = j)\\ &= \sum_{i}^{m}\sum_{j}^{m} u_i(t)p_i(a)\gamma_{ij}(k)p_j(b)\\ &= \mathbf{u}(t)\mathbf{P}(a)\mathbf{\Gamma}^k\mathbf{P}(b)\mathbf{1'}\end{aligned}$$

where $\mathbf{P}(x) = \text{diag}(p_1(x), p_2(x), \ldots, p_m(x))$. This reduces to $\boldsymbol{\delta}\mathbf{P}(a)\mathbf{\Gamma}^k\mathbf{P}(b)\mathbf{1'}$ if the Markov Chain is stationary as we assume the distribution of hidden states at any time t is the stationary distribution $\boldsymbol{\delta}$

Using the bivariate cases, we can extend it to all observed variables $\{X_t : t \in \{1, \ldots, T\}\}$. Suppose we have initial distribution $\boldsymbol{\delta}$ and emission probability $p_i(x)$, the likelihood function of observed variables $L_T = p(X_1 = x_1, \ldots, X_T = x_T)$ can be written as

$$L_T = \boldsymbol{\delta} \mathbf{P}(x_1) \boldsymbol{\Gamma} \mathbf{P}(x_2) \boldsymbol{\Gamma} \mathbf{P}(x_3) \cdots \boldsymbol{\Gamma} \mathbf{P}(x_T) \mathbf{1}'$$

If the Markov Chain is stationary, we can replace $\boldsymbol{\delta}$ by $\boldsymbol{\delta\Gamma}$. Note that $\boldsymbol{\delta}$ usually represents the stationary distribution but however always hard to estimate. With that said, $\boldsymbol{\delta}$ here represents the distribution of the first hidden variable Z_1.

4 EM Algorithm in HMMs

The Expectation-Maximization (EM) algorithm is a recursive algorithm to perform the Maximum Likelihood Estimation when some values are missing. It is advantagenous when likelihood function is complicated and direct maximum likelihood estimation is inappropriate. It firstly calculate the expected likelihood of complete data given the incompleted (observed) data and current estimate of $\boldsymbol{\theta}$ (expectation). It then then maximize the expected likelihood to update $\hat{\boldsymbol{\theta}}$ (maximization). We run this for multiple time until convergence, and hence find the estimates. The resulting estimates of $\boldsymbol{\theta}$ is hence a stationary point of the observed data.

EM algorithm can also be applied to HMMs in a very straightforward way. As the hidden states can not be observed directly, which can be naturally treated as "missing." We can then apply the EM algorithm to recursively estimate $\boldsymbol{\theta}$, which is so-called the **Baum-Welch Algorithm**. This section will discuss both Expectation and Maximization step, respectively.

4.1 Forward-Backward Algorithm

The Forward-Backward Algorithm is one of the fundamental procedure to compute the posterior probability of the hidden states at each time step given the full sequence of observed data. In formula, it evaluates $p\left(Z_t = z_t \mid \mathbf{X}^T = \mathbf{x}^T\right)$, which is crucial as the hidden state are not directly observable. It relies on the two important quantities: **forward probabilities** and **backward probabilities**, defined as $\alpha_t(j) = p(\mathbf{X}^{(t)} = \mathbf{x}^{(t)}, Z_t = j)$ and $\beta_t(i) = p(\mathbf{X}_{t+1}^T = \mathbf{x}_{t+1}^T | Z_t = i)$. The names of "forward" and "backward" comes from the direction in which each quantity transfer the information. The forward probabilities accumulates the information from $t = 1$ to the present time t, while the backward probabilities integrate the information from the future $t = T$ to the present. In this part, we will discuss how the two quantities are defined in this way, and eventually show that the posterior probability is proportional to their products.

Both $\alpha_t(i)$ and $\beta_t(i)$ is came naturally from the factorization of the likelihood function. In matrix form, the forward probabilities α_t can be expressed as

$$\boldsymbol{\alpha}_t = \boldsymbol{\delta} \mathbf{P}(x_1) \prod_{s=2}^{t} \boldsymbol{\Gamma} \mathbf{P}(x_s)$$

which we can show that $\alpha_t(j) = p(\mathbf{X}^{(t)} = \mathbf{x}^{(t)}, Z_t = j)$. Similarly, the backward probabilities can be expressed as

$$\boldsymbol{\beta}_t' = \left(\prod_{s=t+1}^{T} \boldsymbol{\Gamma} \mathbf{P}(x_s) \right) \mathbf{1}'$$

implying $\beta_t(i) = p(\mathbf{X}_{t+1}^T = \mathbf{x}_{t+1}^T | Z_t = i)$ (Appendix). As we will show later, the posterior probability over the hidden state is proportional to the product $\alpha_t(j)\beta_t(i)$.

The task of hidden state inference from Forward-Backward Algorithm breaks down into three parts: **filtering**, **prediction** and **smoothing**. In filtering, the goal is to compute the posterior probability of the current hidden state at the current time t, given all observations up to that point. This is equivalent to *filtered* marginals $\tilde{\alpha}_t(j) = p(Z_t = j|\mathbf{X}^{(t)} = \mathbf{x}^{(t)}) \propto \alpha_t(j)$. For example, the posterior over Z_t given the observations up to $t-1$ can be expressed by

$$\begin{aligned}
P(Z_t = j \mid \mathbf{x}^{(t-1)}) &= \sum_i P(Z_{t+1} = i, Z_t = j \mid \mathbf{x}^{(t+1)}) \\
&= \sum_i P(Z_t = j \mid Z_{t+1} = i, \mathbf{x}^{(t+1)})\, P(Z_{t+1} = i \mid \mathbf{x}^{(t+1)}) \\
&= \sum_i P(Z_t = j \mid Z_{t+1} = i)\, \alpha_{t+1}(i) \\
&= \sum_i \gamma_{ij}\, \alpha_{t+1}(i) \\
&= \left(\alpha_{t+1}^T \Gamma\right)_j
\end{aligned}$$

This idea can be extended to estimate $P(Z_{t+k} = j \mid \mathbf{x}^{(t)})$ for $k > 1$.

Finally, the smoothing provides a refined estimate of the hidden state at time t by conditioning on the entire data $\mathbf{x}^{(T)}$. The smoothed posterior probability is given by

$$\begin{aligned}
p\left(z_t \mid \mathbf{x}^T\right) &\propto p\left(z_t, \mathbf{x}^T\right) \\
&= p\left(z_t, \mathbf{x}^t, \mathbf{x}_{t+1}^T\right) \\
&= p\left(z_t, \mathbf{x}^t\right) p\left(\mathbf{x}_{t+1}^T \mid z_t, \mathbf{x}^t\right) \\
&= p\left(z_t, \mathbf{x}^t\right) P\left(\mathbf{x}_{t+1}^T \mid z_t\right) \\
&= \alpha_t(z_t)\beta_t(z_t)
\end{aligned}$$

By combining the forward and backward probabilities, it provides us an effective way to estimate the hidden state for any time t given the entire observed data. It can be used to find the "beliefs" of the hidden state. We will show later that the Forward-Backward Algorithm is closely related to the E-step in the EM Algorithm and it can be also used as the local decoding on the next session.

4.2 Expectation Step

In the Expectation-Maximization (EM) algorithm for Hidden Markov Models, the E-step involves computing the expectation of the complete-data log-likelihood given the observed data and the current parameter estimates. In the context of HMMs, the complete data combines both the observed data x_t and the corresponding hidden states z_t. Since the hidden states are not directly observed, it naturally becomes the missing part. We will firstly express the complete-data likelihood, and then evaluate its expectation conditional on the observed data sequence.

Let the parameters be $\boldsymbol{\theta} = (\boldsymbol{\delta}, \boldsymbol{\Gamma}, \boldsymbol{\lambda})$, where $\boldsymbol{\delta}$ is the initial distribution, $\boldsymbol{\Gamma}$ is the transition matrix with $(i,j)th$ element γ_{ij}, and $\boldsymbol{\lambda}$ be the emission distribution specific parameters. Let the complete data be $\mathcal{D}_{complete} = \left\{\mathbf{X}^{(T)} = \mathbf{x}^{(T)}, \mathbf{Z}^{(T)} = \mathbf{z}^{(T)}\right\}$. Using the standard factorization of joint distribution of HMMs, the log-likelihood function can be written as

$$\begin{aligned}\ell\left(\boldsymbol{\theta}|\mathcal{D}_{complete}\right) &= \log\left(p\left(z_1\right)\prod_{t=2}^{T}p\left(z_t|z_{t-1}\right)\prod_{t=1}^{T}p\left(x_t|z_t\right)\right)\\ &= \log\delta_{z_1}+\sum_{t=2}^{T}\log\gamma_{z_{t-1},z_t}+\sum_{t=1}^{T}\log p_{z_t}\left(x_t\right)\end{aligned}$$

However, the log-likelihood expression above is written in terms of specific hidden states z_t, and does not explicitly leading to the parameters $\boldsymbol{\theta}$. To solve this, we introduce two new indicator variables $u_j(t) = \mathbf{1}\{Z_t = j\}$, $t \in \{1, \ldots, T\}$ and $v_{jk}(t) = \mathbf{1}\{Z_{t-1} = j, Z_t = k\}$, $t \in \{2, \ldots, T\}$. By plugging in the indicators, the log-likelihood function becomes

$$\ell\left(\boldsymbol{\theta}|\mathcal{D}_{complete}\right) = \sum_{j=1}^{m}u_j(1)\log\delta_j+\sum_{j=1}^{m}\sum_{k=1}^{m}\left(\sum_{t=2}^{T}v_{jk}(t)\right)\log\gamma_{jk}+\sum_{j=1}^{m}\sum_{t=1}^{T}u_j(t)\log p_j\left(x_t\right)$$

The two indicator variables depends only on the hidden states which is unobserved. On the other hand, the rest term on the log-likelihood are all parameters which is not random. Therefore, in the E-step, we only need to compute the expectation of $u_j(1), v_{jk}(t), u_j(t)$ conditioning on the observed data $\mathbf{x}^{(T)}$ and the current estimates of $\boldsymbol{\theta}$. It can be shown that these expectations are (Appendix)

$$\begin{aligned}\mathbb{E}[u_j(t)|\mathbf{x}^{(T)},\hat{\boldsymbol{\theta}}] &= p\left(C_t = j \mid \mathbf{x}^{(T)},\hat{\boldsymbol{\theta}}\right)\\ &= \hat{\alpha}_t(j)\hat{\beta}_t(j)/\hat{L}_T\\ &=: \hat{u}_j(t)\end{aligned}$$

$$\begin{aligned}\mathbb{E}[v_{jk}(t)|\mathbf{x}^{(T)},\hat{\boldsymbol{\theta}}] &= p\left(C_{t-1} = j, C_t = k \mid \mathbf{x}^{(T)},\hat{\boldsymbol{\theta}}\right)\\ &= \hat{\alpha}_{t-1}(j)\hat{\gamma}_{jk}\hat{p}_k\left(x_t\right)\hat{\beta}_t(k)/\hat{L}_T\\ &=: \hat{v}_{jk}(t)\end{aligned}$$

where the estimated forward and backward probabilities are estimated by

$$\begin{aligned}\hat{\alpha}_{t+1}(j) &= \sum_{i=1}^{m}\hat{\alpha}_t(i)\hat{\gamma}_{ij}\hat{p}_j\left(x_{t+1}\right)\ \text{ s.t. }\ \alpha_1(j) = \delta_j p_j\left(x_1\right)\\ \hat{\beta}_t(i) &= \sum_{j=1}^{m}\hat{\gamma}_{ij}\hat{p}_j\left(x_{t+1}\right)\hat{\beta}_{t+1}(j)\ \text{ s.t. }\ \beta_T(j) = 1\end{aligned}$$

And the estimated likelihood is given by

$$\hat{L}_T = \hat{\boldsymbol{\alpha}}_T\hat{\boldsymbol{\beta}}_T'$$

Putting all together, the expected log-likelihood function is written as

$$\mathbf{Q}(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}) = \sum_{j=1}^{m}\hat{u}_j(1)\log\delta_j+\sum_{j=1}^{m}\sum_{k=1}^{m}\left(\sum_{t=2}^{T}\hat{v}_{jk}(t)\right)\log\gamma_{jk}+\sum_{j=1}^{m}\sum_{t=1}^{T}\hat{u}_j(t)\log p_j\left(x_t\right)$$

4.3 Maximization Step

In M step, we maximize the expected log-likelihood $\mathbf{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$ wrt to $\boldsymbol{\theta}$ to update the estimates

$$\hat{\boldsymbol{\theta}}^{new} = \arg\max_{\theta} \mathbf{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$$

Conveniently, the subjective function separates to three terms with each depending on a component of $\boldsymbol{\theta}$: the initial distributions, transition matrix and emission parameters. This allows us to maximize each set of independent parameters. Taking the derivatives for each time, we can obtain the closed-form updates for $\boldsymbol{\delta}$ and $\boldsymbol{\Gamma}$ as

$$\delta_j = \frac{\widehat{u}_j(1)}{\sum_{j=1}^{m} \widehat{u}_j(1)} = \widehat{u}_j(1)$$

$$\gamma_{jk} = f_{jk} / \sum_{k=1}^{m} f_{jk}, \text{ where } f_{jk} = \sum_{t=2}^{T} \widehat{v}_{jk}(t)$$

However, the estimates for the emission parameters $\boldsymbol{\lambda}$ depends on the specific form of the emission model and we can not provide a general formula here.

We will run EM Algorithm recursively until convergence.

5 Forecast and Decoding

Forecast and decoding are two important implications from HMMs, which can help us to predict the

5.1 Forecasting

The forecasting in HMMs means we want to predict the values of the observed variables given the information we have. It describes the conditional distribution of the X_{T+h} given all the observed values we have up to time T. The conditional distribution can be expressed as

$$\begin{aligned} p(X_{T+h} = x | \mathbf{X}^{(T)} = \mathbf{x}^{(T)}) &= \frac{p(X_{T+h} = x, \mathbf{X}^{(T)} = \mathbf{x}^{(T)})}{p(\mathbf{X}^{(T)} = \mathbf{x}^{(T)})} \\ &= \frac{\boldsymbol{\delta}\mathbf{P}(x_1) \cdots \boldsymbol{\Gamma}\mathbf{P}(x_T)\boldsymbol{\Gamma}^h \mathbf{P}(x)\mathbf{1}'}{\boldsymbol{\delta}\mathbf{P}(x_1) \cdots \boldsymbol{\Gamma}\mathbf{P}(x_T)} \\ &= \frac{\boldsymbol{\alpha}_T \boldsymbol{\Gamma}^h \mathbf{P}(x)\mathbf{1}'}{\boldsymbol{\alpha}_T \mathbf{1}'} \end{aligned}$$

Let $\boldsymbol{\phi}_T = \frac{\boldsymbol{\alpha}_T}{\boldsymbol{\alpha}_T \mathbf{1}'}$, then it can be written as $\phi_T \boldsymbol{\Gamma}^h \mathbf{P}(x)\mathbf{1}'$. It can be also written in terms of a mixture of the emission distribution $p\left(X_{T+h} = x \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \sum_{i=1}^{m} \xi_i(h) p_i(x)$, where $\xi_i(h)$ is ith entry of the vector $\phi_T \boldsymbol{\Gamma}^h$.

One important implication of the forecasted conditional distribution is that, as h increases, the forecast distribution converges to the marginal distribution of the stationary HMM, that is

$$\lim_{h \to \infty} \Pr\left(X_{T+h} = x \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \lim_{h \to \infty} \boldsymbol{\phi}_T \boldsymbol{\Gamma}^h \mathbf{P}(x)\mathbf{1}' = \boldsymbol{\delta}^* \mathbf{P}(x)\mathbf{1}'$$

where $\boldsymbol{\delta}^*$ is the stationary distribution and $\boldsymbol{\delta}$ is the initial distribution. It shows that for any non-negative row vectors with row sum 1 ($\boldsymbol{\delta}^*$ here) approaches the stationary distribution of the Markov Chain. However, this property can be only applied to irreducible and aperiodic Markov Chain []. We call $\boldsymbol{\delta}^*\mathbf{P}(x)\mathbf{1}'$ as the limiting distribution of the Markov Chain. The speed of converging to the limiting distribution may various. We will discuss more about this in the application part.

5.2 Decoding

In HMMs, the term "forecasting" usually infers to predict the future values of the observed variables X_t. In contrast, "decoding" describes the way we use the observed values to compute the most likely hidden states. In this paper, we are going to discuss two kinds of decoding process: local and global decoding. The difference between these two is the number of hidden state we compute each time. If we only decode on hidden state, then it refers the local decoding. If we decode several hidden states simultaneously, it is refered as global decoding.

5.2.1 Local Decoding

The algorithm for local decoding is simple. Recall that the joint distribution $p\left(Z_t = i, \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)$ can be expressed as the product of forward and backward probabilities, $\alpha_t(i)\beta_t(i)$. Then the conditional distribution of being on state i at time t is

$$p\left(Z_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \frac{p\left(Z_t = i, \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)}{p\left(\mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)}$$

We have proved that $p\left(Z_t = i, \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \alpha_t(i)\beta_t(i)$ from the Forward-Backward Algorithm, then the joint distribution can be written as

$$p\left(Z_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \frac{\alpha_t(i)\beta_t(i)}{L_T}$$

Then for each $t \in \{1, \ldots, T\}$, we can compute the most probable hidden state $Z_t = i^*$ by

$$i_t^* = \underset{i=1,\ldots,m}{\operatorname{argmax}} p\left(Z_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)$$

By this approach, we can find the most likely hidden state at any time $t \leq T$, and this approach is called the local decoding.

5.2.2 Global Decoding

In contrast to local decoding, global decoding allows us to compute a sequence of hidden states together instead of a single hidden state. It is noticeble that global decoding is not equivalent to run the local decoding for multiple times. The difference between the two is that running local decoding multiple times may give us unrealistic sequence such as impossible transition. We prefer global decoding as it will give us a probable and reasonable sequence of hidden states, while still keep the realistic transition probabilities.

In global decoding, our goal is to to maximize

$$\begin{aligned} p(\mathbf{Z}^{(t)} = \mathbf{z}^{(t)} | \mathbf{X}^{(t)} = \mathbf{x}^{(t)}) &\propto p(\mathbf{Z}^{(t)} = \mathbf{z}^{(t)} | \mathbf{X}^{(t)} = \mathbf{x}^{(t)}) \\ &= \delta_{z_1} \prod_{t=2}^{T} \gamma_{z_{t-1}, z_t} \prod_{t=1}^{T} p_{z_t}(x_t) \end{aligned}$$

The complete global decoding works as follows

1. We firstly define $\xi_{1,i} = p(Z_1 = i, X_1 = x1) = \delta_i p_i(x_1)$, meaning that we are at state i at time 1.
2. For $t \in \{2, 3, \ldots, T\}$, we define

$$\xi_{ti} = \max_{z_1, z_2, \ldots, z_{t-1}} p\left(\mathbf{Z}^{(t-1)} = \mathbf{z}^{(t-1)}, Z_t = i, \mathbf{X}^{(t)} = \mathbf{x}^{(t)}\right)$$

   ξ_{ti} is the probability of the best sequence of $Z_1, \ldots, Z_t$ such that this sequence has to be ended on $Z_t = i$. It is computed by considering each possible path of $z_1, \ldots, z_t = i$ and return the highest probability. This equation is equivalent to

$$\xi_{tj} = \left(\max_i \left(\xi_{t-1,i} \gamma_{ij}\right)\right) p_j(x_t)$$

   By this step, it can give us a $T \times m$ matrix with each element storing the best score at in each time and state.
3. Given $t = T$, we have

$$i_T = \underset{i=1,\ldots,m}{\operatorname{argmax}} \, \xi_{Ti}$$

   This step helps us to find the most likely state at the final time T. It can be done simply by search the last row of the matrix we had and state corresponding to the highest score will be the choice of the value of the last hidden state.
4. For $t \in \{T-1, T-2, \ldots, 1\}$, find i_t such that

$$i_t = \underset{i=1,\ldots,m}{\operatorname{argmax}} \left(\xi_{ti} \gamma_{i, i_{t+1}}\right)$$

   Then can have the sequence of values of hidden states.

This algorithm is so-called the Viterbi Algorithm.

5.2.3 State Prediction

Finally, we are also concerned about how to predict the value of hidden states in the future time $t > T$, $p\left(Z_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)$. Since $\boldsymbol{\alpha}_T = p(Z_T, \mathbf{X}^{(T)} = \mathbf{x}^{(T)})$, then the joint distribution can be expressed as $p\left(Z_{T+1}, \mathbf{X}^{(T)}\right) = p\left(C_{T+1} = k \mid C_T = j\right) p\left(C_T = j \mid \mathbf{X}^{(T)}\right) = \boldsymbol{\alpha}_T \boldsymbol{\Gamma}$. Similarly, we also have $p\left(Z_{T+h}, \mathbf{X}^{(T)}\right) = \boldsymbol{\alpha}_T \boldsymbol{\Gamma}^h$. To have $p\left(Z_{T+h} = i, \mathbf{X}^{(T)}\right)$, it is equivalent to the i-th component of $\boldsymbol{\alpha}_T \boldsymbol{\Gamma}^h$, which is equivalent to $\boldsymbol{\alpha}_T \boldsymbol{\Gamma}^h \mathbf{e}_i'$, where $\mathbf{e}_i$ is a row vector that only the ith element is 1 and all other are zeros. Therefore the conditional distribution can be written as

$$p\left(Z_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) = \frac{p\left(Z_t = i, \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right)}{\boldsymbol{\alpha}_T \mathbf{1}'} = \frac{\boldsymbol{\alpha}_T \boldsymbol{\Gamma}^h \mathbf{e}_i'}{\boldsymbol{\alpha}_T \mathbf{1}'} = \phi_T \boldsymbol{\Gamma}^h \mathbf{e}_i'$$

To sum up, recall that in the Forward-Backward Algorithm, we have derived that $p(Z_t = z_t | \mathbf{X}^{(T)} = \mathbf{x}^{(T)}) = \alpha_t(z_t)\beta_t(z_t)$, the forecasting of hidden states can be summarized by

$$\begin{aligned} & L_T \operatorname{Pr}\left(C_t = i \mid \mathbf{X}^{(T)} = \mathbf{x}^{(T)}\right) \\ & = \begin{cases} \boldsymbol{\alpha}_T \boldsymbol{\Gamma}^{t-T} \mathbf{e}_i^{\prime} & \text{for } t > T \\ \alpha_T(i) & \text{for } t = T \\ \alpha_t(i)\beta_t(i) & \text{for } 1 \leq t < T \end{cases} \end{aligned}$$

6 Application

It is easy to performing HMMs in R. Although we can still write the recursive function manually, we can still the R package `HiddenMarkov` [] which performs the EM in Hidden Markov Models. In this section, we will firstly simulate the data for both discrete and continuous cases, to see how the EM Algorithm and decoding are performed. We will then apply it to a real-world scenarios such as the volatility of NVIDIA stock.

6.1 Simulation: Three-state Poisson-HMM

We will firstly see how the HMMs perform when the emission distribution is Poisson. Suppose there are three discrete hidden states $Z_t \in \{1, 2, 3\}$, following a three-state Poisson HMM, where the initial state distribution is given by $\boldsymbol{\delta} = (0.5, 0.3, 0.2)$, and the transition probability matrix is

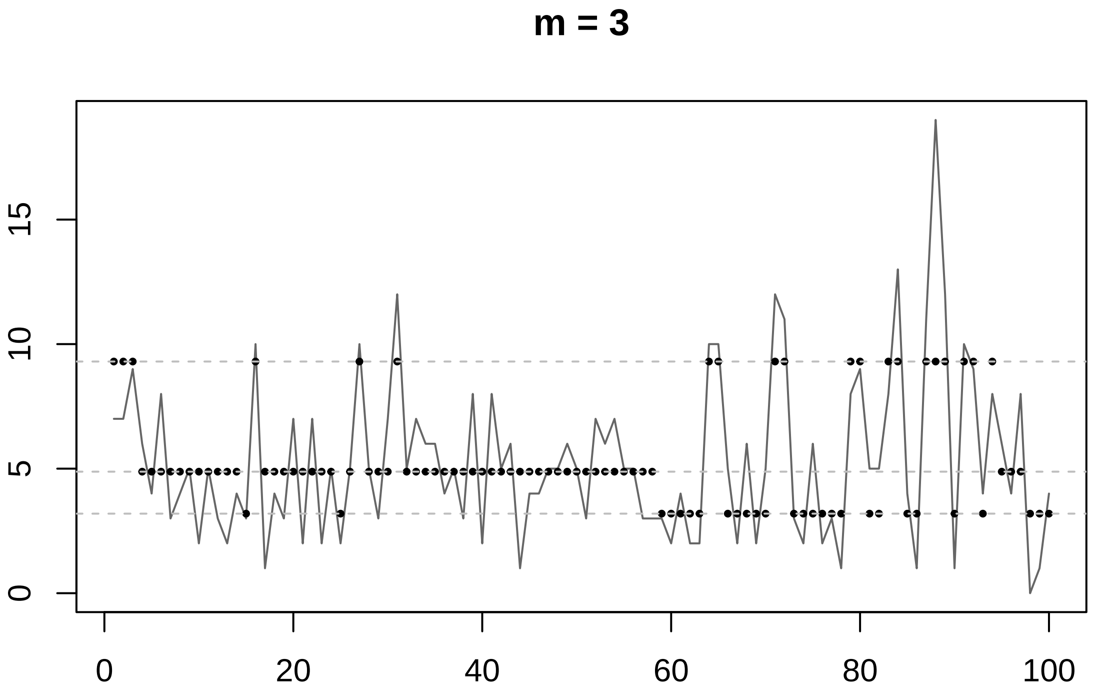
$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.5 & 0.3 & 0.2 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.3 & 0.5 \end{pmatrix}$$

The emission distribution in each state is Poisson, with state-specific $\boldsymbol{\lambda} = (3, 4, 5)$, so that conditional on the hidden state $Z_t = j$, the observation X_t is distributed as Poisson$(\boldsymbol{\lambda}_j)$. A sequence of $n = 100$ observations was generated by first sampling the initial hidden state from $\boldsymbol{\delta}$, then recursively generating subsequent states using the transition matrix $\boldsymbol{\Gamma}$, and finally sampling each observation from the corresponding Poisson distribution based on the current hidden state.

We will firstly estimate the parameters $\boldsymbol{\theta} = (\boldsymbol{\delta}, \boldsymbol{\Gamma}, \boldsymbol{\lambda})$ using the EM algorithm. To initialize the algorithm, we specify the initial Poisson means as $\boldsymbol{\lambda}^{(0)} = (1, 2, 3)$. The initial transition probability matrix $\boldsymbol{\Gamma}^{(0)}$ is chosen to have 0.8 on the diagonals and all 0.1 off the diagonals. We assume the initial state distribution $\boldsymbol{\delta}^{(0)}$ to be uniform over the three states, $\boldsymbol{\delta}^{(0)} = (1/3, 1/3, 1/3)$. Given these starting values, the EM algorithm is then applied to maximize the likelihood of the observed sequence $\{X_t\}$ to estimate the parameters.

The estimated $\hat{\boldsymbol{\lambda}} = (3.194, 4.879, 9.301)$ with initial distribution $\boldsymbol{\delta} = (0, 0, 1)$. The estimated transition matrix is

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.664 & 0 & 0.3366 \\ 0.053 & 0.947 & 0 \\ 0.369 & 0.128 & 0.503 \end{pmatrix}$$



6.2 Simulation: Three-State Gaussian-HMM

We will firstly see how the HMMs perform when the emission distribution is Gaussian. Suppose there are three discrete hidden states $Z_t \in \{1, 2, 3\}$, following a three-state Poisson HMM, where the initial state distribution is given by $\boldsymbol{\delta} = (0.5, 0.3, 0.2)$, and the transition probability matrix is

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0.3 & 0.5 & 0.2 \\ 0.4 & 0.3 & 0.3 \end{pmatrix}$$

The emission distribution in each state is Gaussian, with state-specific $\boldsymbol{\mu} = (0, 5, 10)$ and $\boldsymbol{\sigma} = (1, 2, 3)$, so that conditional on the hidden state $Z_t = i$, the observation X_t is distributed as $\mathcal{N}(\mu_i, \sigma_i)$. A sequence of $n = 150$ observations was generated by first sampling the initial hidden state from $\boldsymbol{\delta}$, then recursively generating subsequent states using the transition matrix $\boldsymbol{\Gamma}$, and finally sampling each observation from the corresponding Gaussian distribution based on the current hidden state.

We will firstly estimate the parameters $\boldsymbol{\theta} = (\boldsymbol{\delta}, \boldsymbol{\Gamma}, \boldsymbol{\mu}, \boldsymbol{\sigma})$ using the EM algorithm. To initialize the algorithm, we specify the initial Gaussian parameters as $\boldsymbol{\mu}^{(0)} = (1, 2, 3)$ and $\boldsymbol{\sigma}^{(0)} = (1, 1, 1)$. The initial transition probability matrix $\boldsymbol{\Gamma}^{(0)}$ is chosen to have 0.8 on the diagonals and all 0.1 off the diagonals. We assume the initial state distribution $\boldsymbol{\delta}^{(0)}$ to be uniform over the three states, $\boldsymbol{\delta}^{(0)} = (1/3, 1/3, 1/3)$. Given these starting values, the EM algorithm is then applied to maximize the likelihood of the observed sequence $\{X_t\}$ to estimate the parameters.

After applying the EM algorithm, the estimated transition matrix is

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.496 & 0.248 & 0.256 \\ 0.137 & 0.734 & 0.129 \\ 0.417 & 0.098 & 0.485 \end{pmatrix}$$

and the estimated initial distribution is

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

which indicated the hidden markov process begins in state 2. Moreover, the estimated parameters of the emission distributions are closely aligned with the true values: $\mu_1 = -0.254, \sigma_1 = 0.906$, $\mu_2 = 4.615, \sigma_2 = 2.925$ and $\mu_3 = 9.892, \sigma_3 = 3.976$. These all demonstrate the robustness of EM Algorithm in HMMs.

Furthermore, we can plot the estimated hidden states for all time and compare them to the original time series.

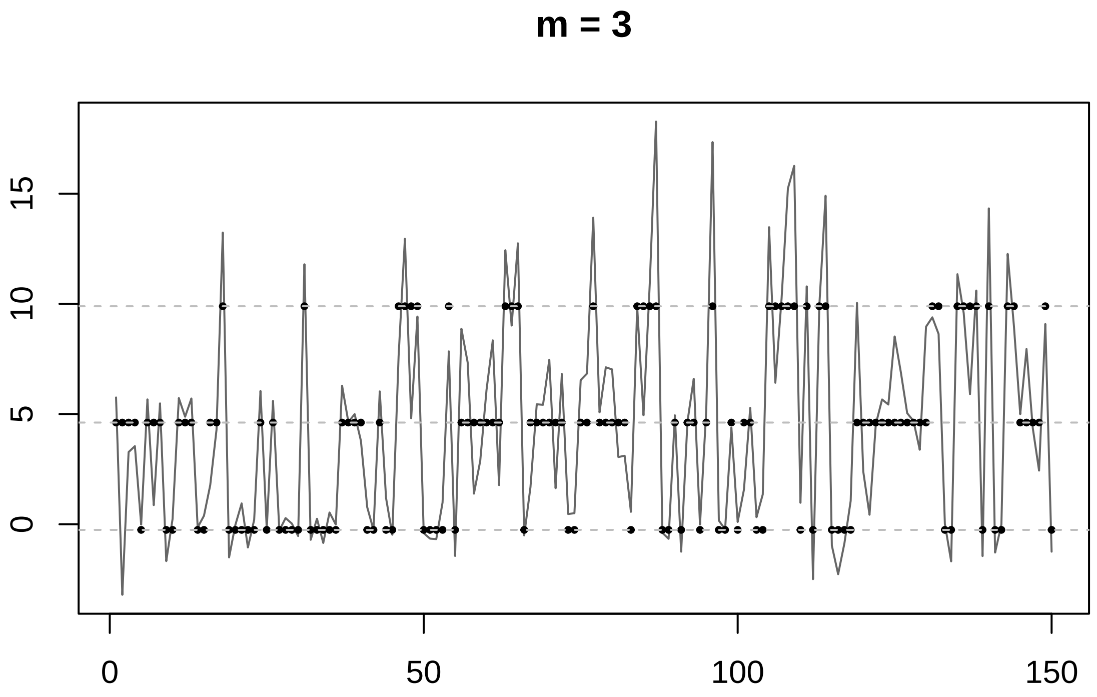
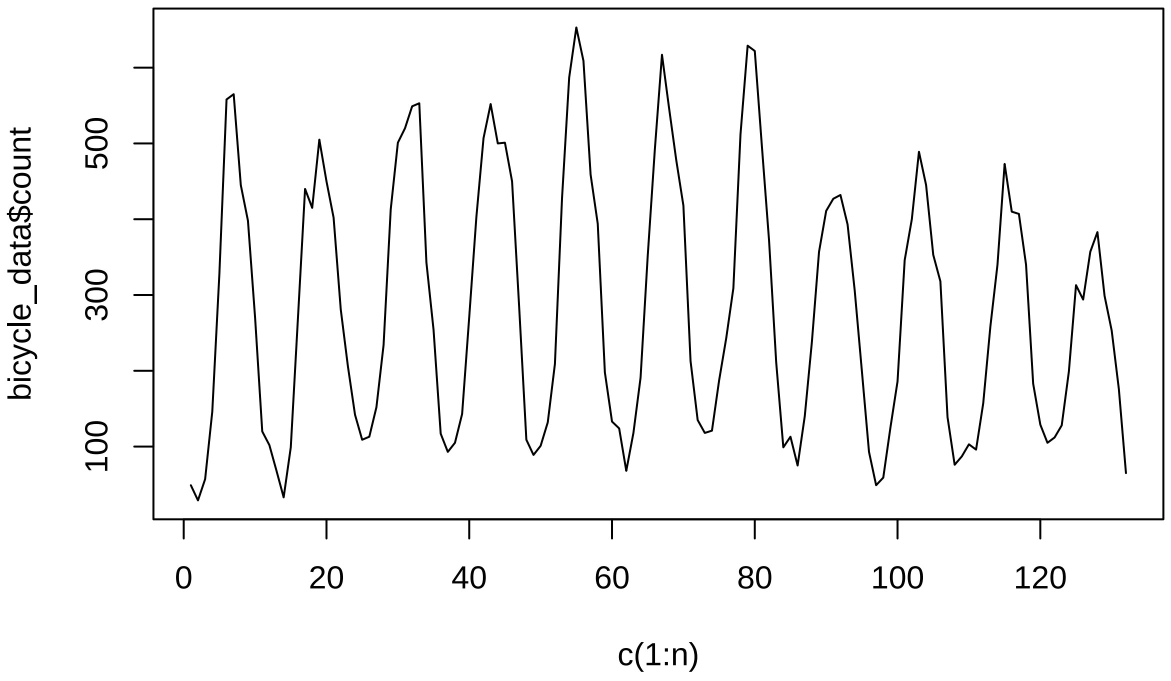


Figure 3: Local decoding of 3-state Gaussian HMM

6.3 Toronto Bicycle Thefts



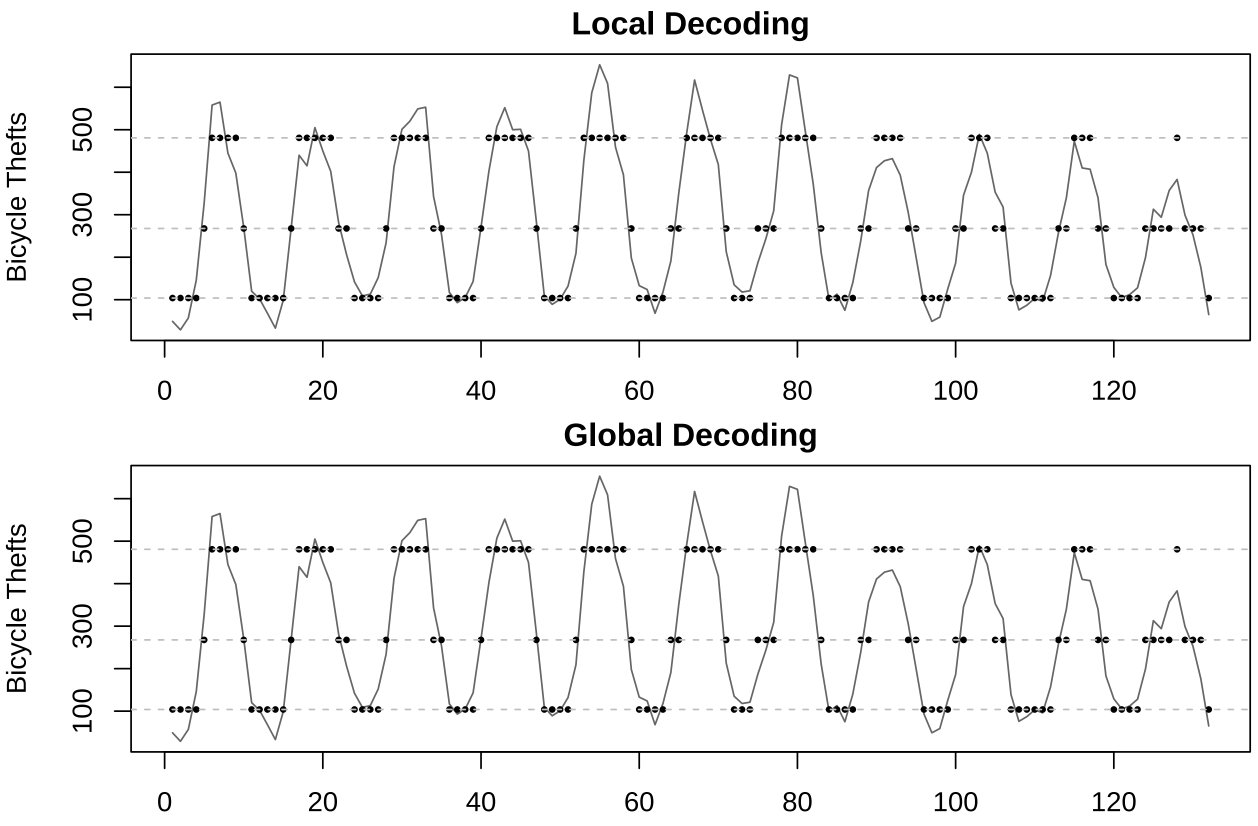


Figure 4: Local decoding (top) and glocal decoding (bottom) for the number of monthly bicycle thefts at Toronto since 2014 (3 hidden states)

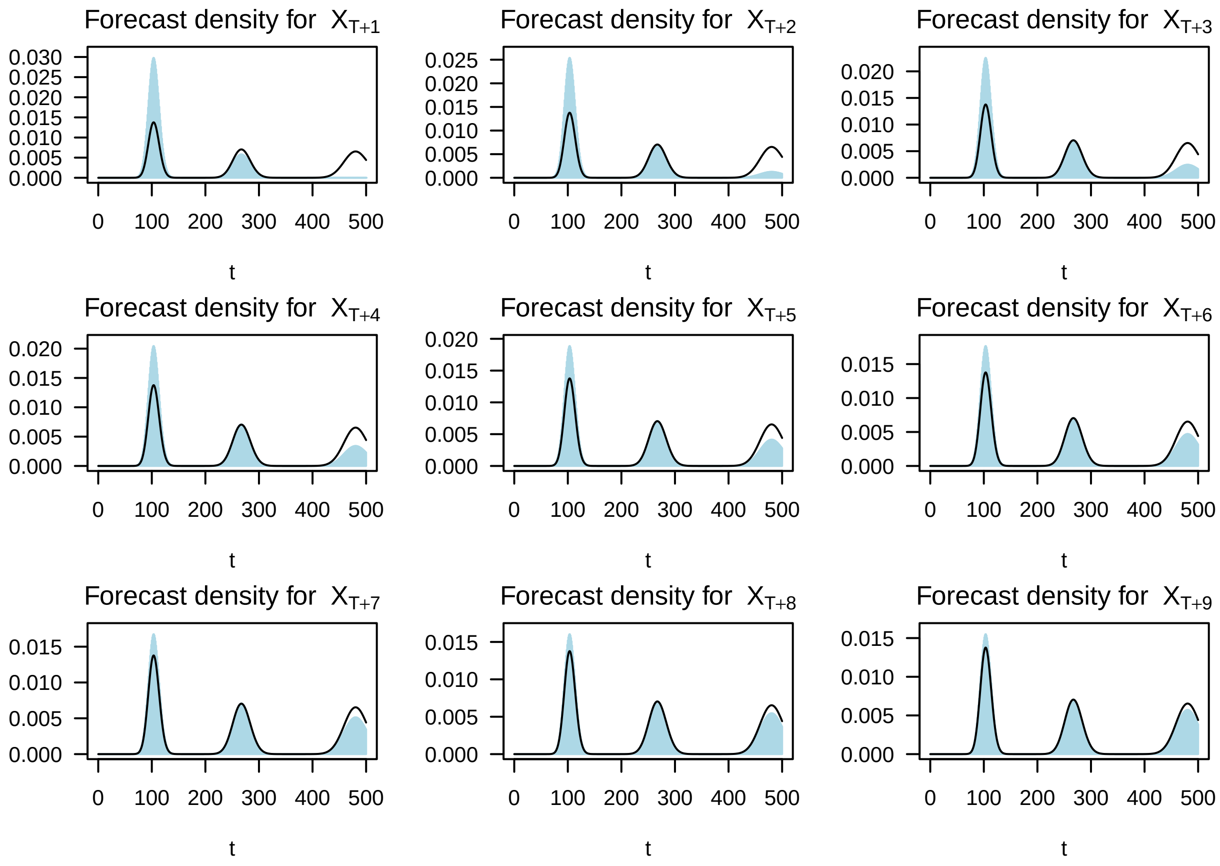


Figure 5: Forecasted distribution of monthly bicycle theft counts in Toronto for the next nine months

```
##              [,1]       [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 7.613847e-01 0.64907207 0.5760438 0.5225899 0.4821846 0.4513992 0.4278984
## [2,] 2.386153e-01 0.28156274 0.2889544 0.2899647 0.2898908 0.2896801 0.2894910
## [3,] 1.508727e-45 0.06936519 0.1350018 0.1874454 0.2279246 0.2589207 0.2826106
##           [,8]      [,9]     [,10]
## [1,] 0.4099502 0.3962413 0.3857701
## [2,] 0.2893415 0.2892263 0.2891382
## [3,] 0.3007083 0.3145324 0.3250917
```

6.4 NVIDIA stock price

The previous simulation example illustrate that the perform well. To further illustrate its performance, let us apply it to the NVIDIA stock price. The data used here is through the R package `quantmod` to access the NVIDIA stock price from January 2nd, 2024 to April 8th, 2025. The goal of this study is to use HMM to estimate the volatility level behind each price and assumed hidden. The volatility level here is calculated by absolute log returns. We divide the volatility levels into three states: low, medium and high, and the emission distribution of volatility is assumed to follow a normal distribution.

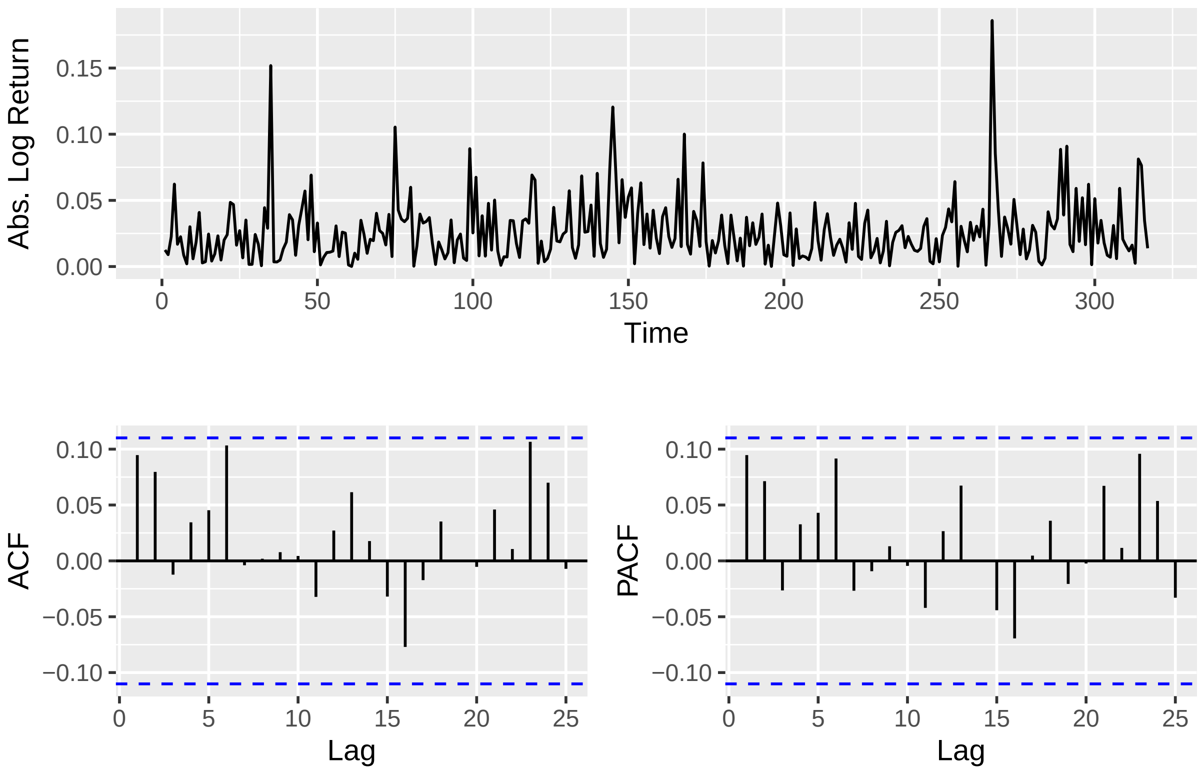


Figure 6: The absolute log return (top), ACF plot (bottom left) and PACF (bottom right)

Descriptions here.

By performing the EM Algorithm, the estimated initial distribution $\boldsymbol{\delta} = (1, 0, 0)$, and the estimated transition matrix is

$$\hat{\mathbf{\Gamma}} = \begin{pmatrix} 0.254 & 0.678 & 0.068 \\ 0.264 & 0.558 & 0.178 \\ 0.157 & 0.613 & 0.23 \end{pmatrix}$$

The estimated parameters are $\mu_1 = 0.006, \sigma_1 = 0.004$, $\mu_2 = 0.025, \sigma_2 = 0.012$ and $\mu_3 = 0.062, \sigma_3 = 0.033$.

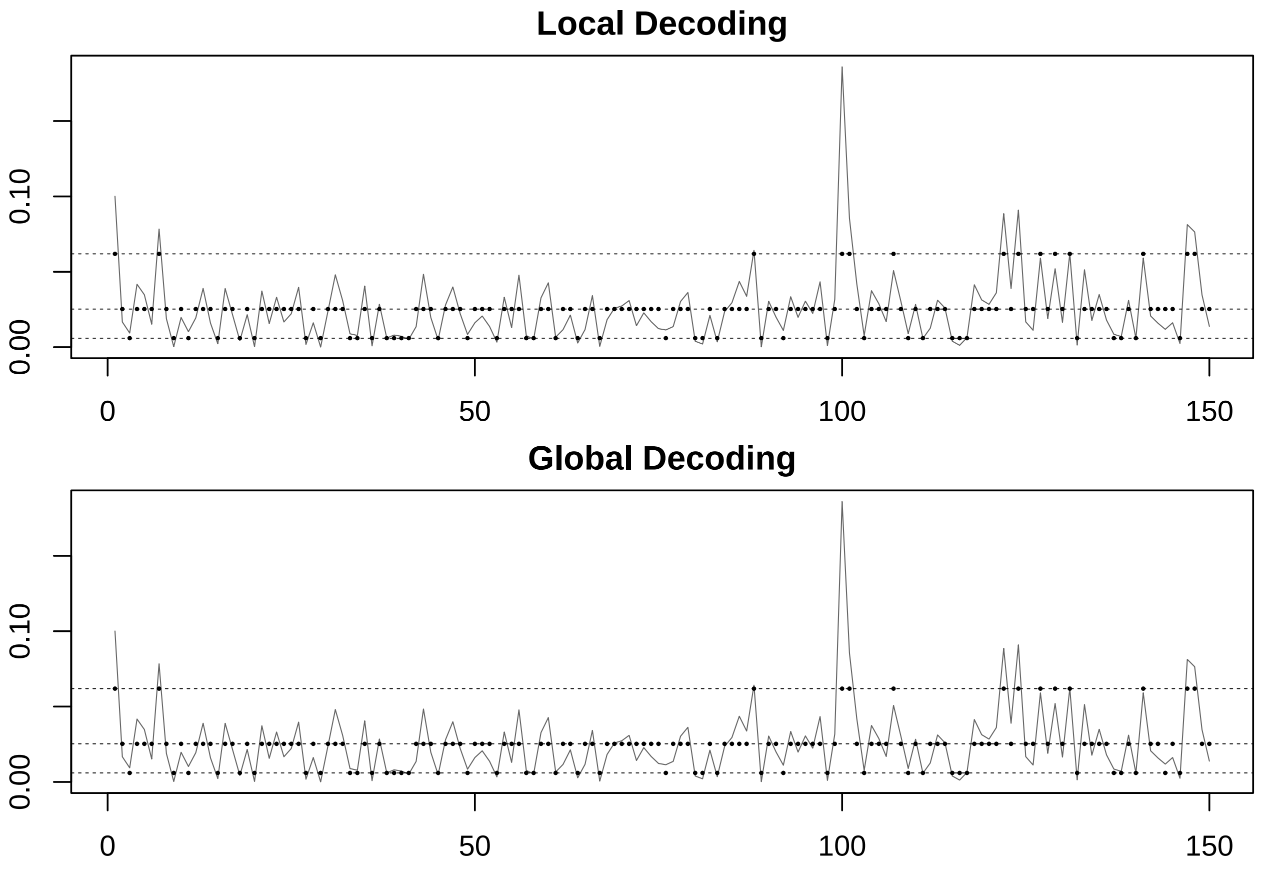


Figure 7: Local decoding (top) and glocal decoding (bottom) for the volatility of NVIDIA stock (the most recent 150 observations due to visualbility) (3 hidden states)

Figure 5 shows the estimated hidden states for each time point on NVIDIA stock prices. We can see that the estimates are reasonable, which almost correctly identify the points with high, medium and low volatilities.

```
##   step       mean    lower_95   upper_95
## 1    1 0.02701487 0.001571192 0.09541011
## 2    2 0.02727076 0.001621442 0.09550260
## 3    3 0.02732287 0.001621506 0.09568861
## 4    4 0.02732820 0.001622131 0.09569745
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.2568054 0.2446599 0.2447034 0.2445557 0.2445500 0.2445480 0.2445478
```

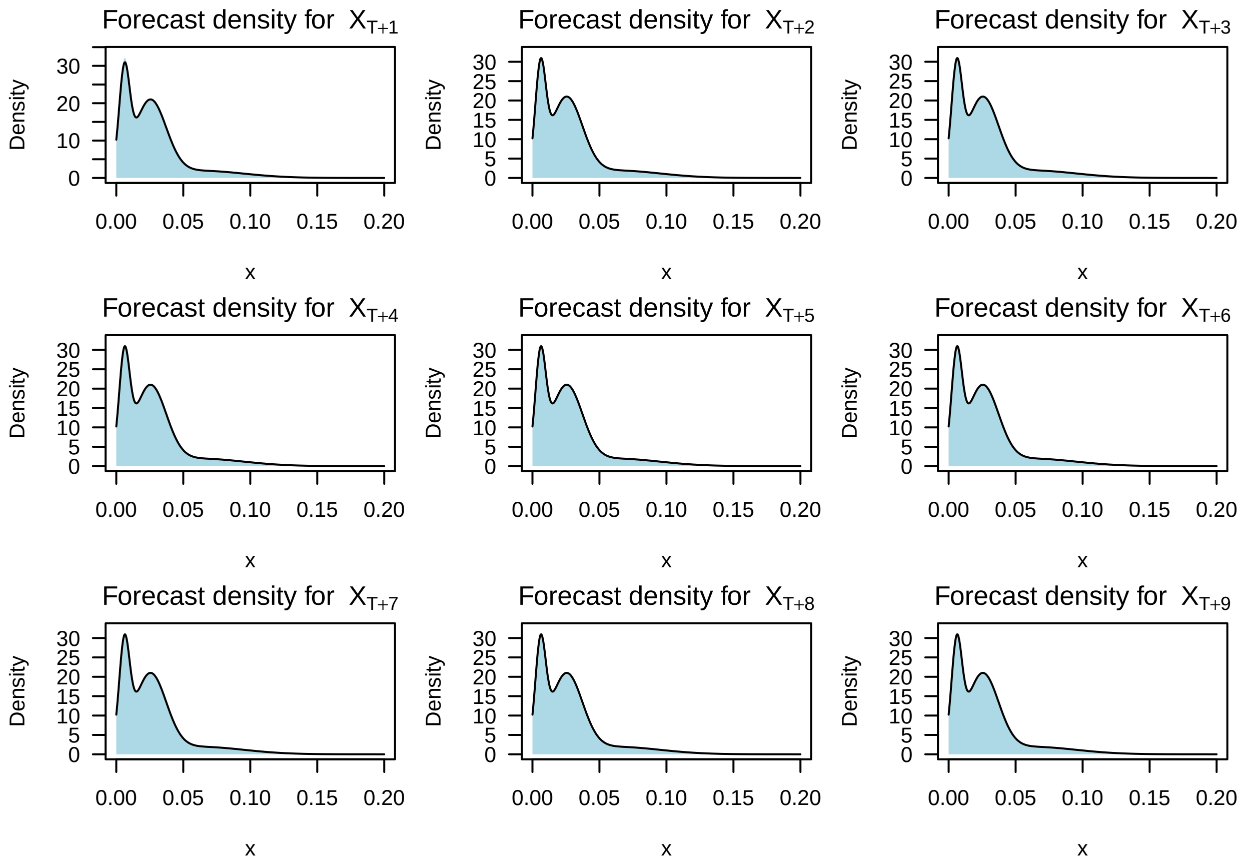


Figure 8: Forecasted distribution of daily volatility (by absolute log returns) of NVIDIA stock for the next nine days

```
## [2,] 0.5860006 0.5973801 0.5959597 0.5960404 0.5960263 0.5960267 0.5960265
## [3,] 0.1571940 0.1579600 0.1593370 0.1594040 0.1594237 0.1594254 0.1594257
##           [,8]      [,9]     [,10]
## [1,] 0.2445478 0.2445478 0.2445478
## [2,] 0.5960265 0.5960265 0.5960265
## [3,] 0.1594257 0.1594257 0.1594257
```

6.5 HMM_GARCH

7 Discussion

I will finish this part in the final version.

8 Appendix

I will show all the necessary proofs and coding on the final version.

References

Bahl, L. R., Jelinek, F., & Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-5*(2), 179–190. https://doi.org/10.1109/TPAMI.1983.4767370

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, *37*(6), 1554–1563. https://doi.org/10.1214/aoms/1177699147

Cai, J. (1994). A markov model of switching-regime ARCH. *Journal of Business & Economic Statistics*, *12*(3), 309–316. https://doi.org/10.2307/1392087

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286. https://doi.org/10.1109/5.18626

Yu, D., & Deng, L. (2015). *Automatic speech recognition: A deep learning approach* (1st ed., pp. XXVI, 321). Springer London. https://doi.org/10.1007/978-1-4471-5779-3

Zhuang, X. F., & Chan, L. W. (2004). Volatility forecasts in financial time series with HMM-GARCH models. In Z. R. Yang, H. Yin, & R. M. Everson (Eds.), *Intelligent data engineering and automated learning – IDEAL 2004* (Vol. 3177, pp. 995–1001). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-28651-6_120

Zucchini, W., MacDonald, I. L., & Langrock, R. (2016). *Hidden markov models for time series: An introduction using r* (2nd ed.). Chapman; Hall/CRC. https://doi.org/10.1201/b20790