Bayesian inference under hidden Markov models

by

Lewis Broderick-Gatrell

Student ID: 2182973

Supervised by: Prof. Xavier Didelot

Department of Statistics
University of Warwick
Coventry CV4 7AL
United Kingdom

Email: lewis.broderick-gatrell@warwick.ac.uk

7th September 2022

Report submitted in partial fulfillment of the requirements for the degree of MSc in Statistics in the University of Warwick

Abstract

Hidden Markov models (HMMs) have become commonly used in statistics since their introduction in the late 1960s and they are now applied across multiple fields of study. Traditionally, inference of HMMs has been conducted through point estimation techniques such as maximum likelihood estimation or recursive algorithms, such as the EM algorithm.

In recent years there has been an increasing interest in the Bayesian approach to HMM inference, typically done through Markov chain Monte Carlo algorithms used to sample from posterior distributions of unknown parameters. Current research that exists focuses on the comparison between the point estimation techniques and the Bayesian approach, but there has been little on the direct comparison between different MCMC techniques.

In this thesis, we applied two of the most popular algorithms to the HMM scenario, the Metropolis-Hastings and the Gibbs Sampler. Once these were successfully implemented and tested we generated Markov chains in order to estimate the posterior distributions of unknown parameters in the HMMs, given a number of assumptions about states, variance and prior distributions.

Acknowledgements

I would like to thank Prof. Xavier Didelot for his help, suggestions and guidance throughout this project. Prof. Xavier Didelot's contribution undoubtedly lead to the successful completion of this masters dissertation.

Contents

1 Introduction **5**

1.1 History of Hidden Markov Models 5

1.2 Scope 6

2 Hidden Markov Models **8**

2.1 Mixture Models 8

2.2 Markov Chains 8

2.3 Stationary Distributions 9

2.4 Hidden Markov Models 9

2.5 Likelihood Function 10

3 MCMC for Hidden Markov Models **12**

3.1 Model Assumptions 12

3.2 Markov chain Monte Carlo 13

3.3 Data Generation 13

3.4 Metropolis Hastings 14

3.5 Gibbs Sampler 15

3.6 MCMC Diagnostics 18

3.7 Trace-plots 18

3.8 Auto-Correlation Function 19

3.9 Effective Sample Size 20

3.10 Gelman-Rubin Statistic 20

4 Results **22**

4.1 Bayesian Inference of a 2-State Hidden Markov Model 22

4.2 Method Comparison 26

4.3 Analysis of Key Results 27

4.3.1 Effect of Observations 28

4.3.2 Effect of Variance 30

4.3.3 Effect of States 32

4.3.4 Label Switching 35

5 Discussion **38**

5.1 Key Findings 38

5.2 Limitations and Improvements 39

5.3 Further Research 39

6 Conclusion **41**

A Probability Distributions **42**

A.1 Univariate Normal Distribution 42

A.2 Multivariate Normal Distribution 42

A.3 Dirichlet Distribution 43

A.4 Continuous Uniform Distribution 43

B Propositions and Proofs **44**

B.1 A factorization needed for the forward probabilities 44

B.2 Forward probability result 45

Chapter 1

Introduction

1.1 History of Hidden Markov Models

Hidden Markov models (HMMs) are models in which the distribution that generates an observation depends on the state of an underlying and unobserved Markov process. They provide flexible general-purpose models for univariate and multivariate time series, especially for discrete-valued series, including categorical series and series of counts (Zucchini, MacDonald, and Langrock 2017). The theory of HMMs was first introduced in a series of papers in the late 1960s and early 1970s (Baum and Petrie 1966; Baum and Eagon 1967; Baum and Sell 1968; Baum, Petrie, et al. 1970; Baum 1972).

One of the first applications of HMMs was in signal processing, particularly speech recognition (Levinson, Rabiner, and Sondhi 1983; Sun and Jelinek 1999). Their applications have since been expanded to fields including bioinformatics (Fonzo, Aluffi-Pentini, and Parisi 2008), economics (Nurfalah et al. 2018), finance (Bhar and Hamori 2012) and weather forecasting (Khiatani and Ghose 2018).

The conditional distributions for each of the observed variables given the unobserved states are typically taken from a common family of distributions, for instance normal distributions with different means and/or variances, so that the marginal distribution of observed variables is over-dispersed relative to a single distribution in that family. A HMM can be viewed as a mixture model with dependence. One of the main problems of HMMs is the estimation of the parameters of each state as well as the transition probabilities between states. This is further complicated by the fact that the states remain hidden. As well as this, the observation at a given time-point must be influenced only by the hidden state at that time-point and no previously hidden state. Therefore inferences about the hidden states and their associated parameters has to be made through what is observed. Estimation of the parameters has most commonly been conducted through the use of a recursive algorithms, such as the **expectation-maximisation (EM) algorithm** (Dempster, Laird, and Rubin 1977), or by direct maximization of the likelihood. The EM algorithm in the case of HMMs is called the **Baum-Welch algorithm** (Baum, Petrie, et al. 1970) and its popularity is due to the efficient computational tool known as the **forward-backward algorithm** that is used to implement EM for HMMs.

Since then the MLE and the Baum-Welch algorithm have been the main vehicles for inference in HMMs. However, recursive algorithms are often viewed as black boxes. Moreover, both the MLE and the recursive algorithms are point estimation techniques thus they fail to generate sampling distributions for the unknown parameters and this can only be achieved through bootstrapping. Modern advances in computational capability have meant that **Markov chain Monte Carlo** (MCMC) techniques allow for the implementation of HMMs and inference of the parameters without having to rely on these recursive techniques. MCMC methods for HMMs have the advantage of being able to generate a sampling distribution of the parameters which can be explored, but come at the cost of greater computational expense. Numerous papers have been written exploring Bayesian inference of hidden Markov models, and examples of different techniques available can be found in Robert and Titterington 1998, Robert, Rydén, and Titterington 2000, Scott 2002, Cappé, Moulines, and Rydén 2005 and Congdon 2006 to name a few.

1.2 Scope

Current research into Bayesian inference for HMM has largely focused on comparing point estimation methods against one or several MCMC algorithms. For example, in Rydén 2008 the posterior distribution generated through either the Gibbs sampler or reversible jump MCMC sampler is compared against the bootstrapped point estimate obtained through the EM algorithm. They found that the Bayesian approach does show some advantages, particularly when the data was from complex HMMs where it was shown to tackle the parameter estimation problem more efficiently than the EM algorithm. It was also noted that the chains generated by the MCMC algorithms can spend most of their time in irrelevant modes of the posterior if the problem is ill-posed, and so great care must be take to ensure that the priors and proposal distributions are well suited.

However, little to no research has been conducted comparing Bayesian inference techniques against each other in the case of HMMs. In this thesis, we aim to compare the performance of the Metropolis Hastings and Gibbs Sampling algorithms across different data-sets from a range of HMMs. This will be done by measuring their computational efficiency and utilising the various diagnostics commonly used in this area of research.

Chapter 2

Hidden Markov Models

2.1 Mixture Models

Mixture models are commonly used to deal with over-dispersed, multi-modal distributions. They are designed to accommodate for unobserved heterogeneity in the population, whereby the data comes from a finite number of different groups each with a distinct probability distribution for the observed variable.

An independent mixture distribution consists of $m \in \mathbb{N}$ distinct component distributions and a **mixing distribution** which selects from these component distributions. They may be either discrete or continuous.

2.2 Markov Chains

A sequence of discrete random variables $\{C_t : t \in \mathbb{N}\}$ is said to be a (discrete-time) **Markov chain** if $\forall\, t \in \mathbb{N}$ it has the **Markov property**:

$$\mathbb{P}(C_t \,|\, C_1, C_2, \ldots, C_{t-1}) = \mathbb{P}(C_t \,|\, C_{t-1})$$

It is also important to define the **transition probabilities at time-step** $\boldsymbol{t}$, $\gamma_{ij}(t) = \mathbb{P}(C_{s+t} = j \,|\, C_s = i)$, which denotes the probability of moving to state j at time $s{+}t$ when the state is i at time s, where $i, j \in \{1, \ldots, m\}$ and $s \in \mathbb{N}$. We assume that these probabilities do not depend on s and therefore our Markov chain is **homogeneous** and depends only on the **lag** t. The transition probabilities are stored in an $m \times m$ matrix $\mathbf{\Gamma(t)}$, with the $(i, j)^{th}$ entry equal to $\gamma_{ij}(t)$. We primarily focus on $\mathbf{\Gamma(1)}$, abbreviated as $\mathbf{\Gamma}$, which is known as the **transition probability matrix** (t.p.m). An important property of all finite state-space homogeneous Markov chains is that they satisfy the **Chapman–Kolmogorov equation**:

$$\boldsymbol{\Gamma}(u+t) = \boldsymbol{\Gamma}(u)\,\boldsymbol{\Gamma}(t) \quad \forall\, u,t \in \mathbb{N} \tag{2.1}$$

This leads to the result that $\forall\; t \in \mathbb{N}$

$$\boldsymbol{\Gamma}(t) = \boldsymbol{\Gamma}(1)^t \tag{2.2}$$

Now, the unconditioned joint probability distributions of each C_t are denoted $\boldsymbol{u}(t) = \mathbb{P}(C_t = 1, \dots, C_t = m)$. A property of the t.p.m is that

$$\boldsymbol{u}(t+1) = \boldsymbol{u}(t)\boldsymbol{\Gamma} \quad \forall\; t \in \mathbb{N} \tag{2.3}$$

2.3 Stationary Distributions

A Markov chain with associated t.p.m $\boldsymbol{\Gamma}$ is said to have a stationary distribution $\boldsymbol{\delta}$, a row vector with non-negative entries, if $\exists\; \boldsymbol{\delta}$ such that $\boldsymbol{\delta\Gamma} = \boldsymbol{\delta}$ and $\boldsymbol{\delta 1} = \boldsymbol{1}$.

A Markov chain started from its stationary distribution will continue to have that distribution at all subsequent time points and we shall refer to such a process as a **stationary Markov chain**. Therefore we will set the initial distribution $\boldsymbol{u}(1) = \boldsymbol{\delta}$ by assuming that the hidden states are stationary. The stationary property combined with (2.3) implies that $\boldsymbol{u}(t) = \boldsymbol{\delta}\;\forall\; t = 1, \dots, T$.

2.4 Hidden Markov Models

A **hidden Markov model** (HMM) is a mixture model which in its simplest form can be written as

$$\mathbb{P}(C_t \,|\, \boldsymbol{C}^{(t-1)}) = \mathbb{P}(C_t \,|\, C_{t-1}),\; t \geq 2 \tag{2.4}$$

$$\mathbb{P}(X_t \,|\, \boldsymbol{X}^{(t-1)}, \boldsymbol{C}^{(t)}) = \mathbb{P}(X_t \,|\, C_t),\; t \in \mathbb{N} \tag{2.5}$$

where $\boldsymbol{X}^{(t)} = \{X_1, \dots, X_t\}$ and $\boldsymbol{C}^{(t)} = \{C_1, \dots, C_t\}$. The model is formed from two parts, an unobserved mixing distribution which specifies the parameters $\{C_t : t \in \mathbb{N}\}$ which satisfies the Markov property, and the observed state dependent process $\{X_t : t \in \mathbb{N}\}$ which depends only on the current state C_t.

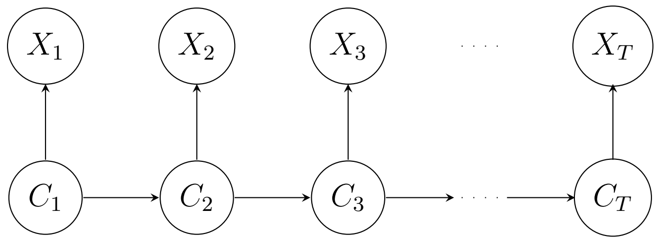


Figure 2.1: Directed graph of a simple hidden Markov model

This is represented in Figure 2.1. We will work with the basic form HMM: its state dependent process is univariate and it is based on a homogeneous Markov chain. We will also assume that it does not have trend or seasonal components and that the unobserved process has a finite number of states m, which is called an **m-state HMM**.

2.5 Likelihood Function

After observing $\mathbf{x}^{(T)} = (x_1, ..., x_T)$ the likelihood function is given as $L_T = \mathbb{P}(X_1 = x_1, ..., X_T = x_t) = \sum_{c_1,..,c_T=1}^{m} \mathbb{P}(\mathbf{X}^{(T)} = \mathbf{x}^{(T)}, \mathbf{C}^{(T)} = \mathbf{c}^{(T)})$. A general result for sequential random variables $\{V_1, \ldots, V_n\}$ is that their joint distribution can be written as

$$\mathbb{P}(V_1, ..., V_n) = \mathbb{P}(V_1) \prod_{i=2}^{n} \mathbb{P}(V_i \,|\, V_1, ..., V_{i-1}) \tag{2.6}$$

By writing the likelihood of our HMM in the same conditional format as (2.6), and by using the properties stated in (2.4) and (2.5), we can write the likelihood as

$$L_T = \mathbb{P}(C_1) \prod_{k=2}^{T} \mathbb{P}(C_k \,|\, C_{k-1}) \prod_{k=1}^{T} \mathbb{P}(X_k \,|\, C_k). \tag{2.7}$$

This can be simplified further when we have the stationary distribution $\boldsymbol{\delta}$ of the Markov chain:

$$\begin{aligned} L_T &= \sum_{c_1,\ldots,c_T=1}^{m} (\delta_{c_1}\gamma_{c_1,c_2}\gamma_{c_2,c_3}\ldots\gamma_{c_{T-1},c_T})(p_{c_1}(x_1)p_{c_2}(x_2)\ldots p_{c_T}(x_T)) \\ &= \sum_{c_1,\ldots,c_T=1}^{m} \delta_{c_1}p_{c_1}(x_1)\gamma_{c_1,c_2}p_{c_2}(x_2)\gamma_{c_2,c_3}\ldots\gamma_{c_{T-1},c_T})\ldots p_{c_T}(x_T) \\ &= \boldsymbol{\delta}\mathbf{P}(x_1)\boldsymbol{\Gamma}\mathbf{P}(x_2)\boldsymbol{\Gamma}\mathbf{P}(x_3)...\boldsymbol{\Gamma}\mathbf{P}(x_T)\mathbf{1}^{'} \end{aligned}$$

As detailed in Zucchini, MacDonald, and Langrock 2017. Here, $\mathbf{P}(x_i)$ is a $m \times m$ diagonal matrix called an **emission matrix**, with the entries taking the form

$$(\mathbf{P}(x_i))_{jk} = \begin{cases} p_j(x_i) = \mathbb{P}(X_i = x_i \,|\, C_i = j) & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \tag{2.8}$$

Each entry along the diagonal is called an **emission probability**. A more computationally efficient way of computing the likelihood is to use the **forward algorithm**, where we define the **forward probability vector $\boldsymbol{\alpha}_t$**, for $t = 1, 2, ..., T$, by

$$\boldsymbol{\alpha}_t = \boldsymbol{\delta}\mathbf{P}(x_1)\prod_{i=2}^{t}\boldsymbol{\Gamma}\mathbf{P}(x_i) \tag{2.9}$$

Clearly, from this definition it follows that

$$L_T = \boldsymbol{\alpha}_T\mathbf{1}^{'} \quad \text{and} \quad \boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1}\boldsymbol{\Gamma}\mathbf{P}(x_t), \tag{2.10}$$
$$\forall\ t \geq 2,\ \text{where } \boldsymbol{\alpha}_1 = \boldsymbol{\delta}\mathbf{P}(x_1)$$

The use of the forward algorithm (2.9) allows for an exponentially faster computation time of the likelihood. Without it, integrating over all m states for T observations would require m^T steps, which would quickly become infeasible as T grew larger. However, through the forward algorithm this reduced to $T \times m^2$, which is computationally feasible even as T grows large.

Chapter 3

MCMC for Hidden Markov Models

3.1 Model Assumptions

The emission probabilities defined in (2.8) can come from any probability distribution, with the state it is conditioned on determining the parameters used. We will be working with **Normal-HMMs**, where $\forall\ i = 1, ..., T$

$$X_i \,|\, C_i = k \sim \mathcal{N}(\mu_k, \sigma^2), \quad k \in (1, ..., m) \tag{3.1}$$

with known variance σ^2. We also assume that the number of states, m, is known. The estimation of the number of states is another problem that can be explored separately. Another assumption made is that $\mu_1 < \ldots < \mu_m$.

We could have chosen to work with many other distributions when conducting this research, whether they be discrete or continuous. For example, in Zucchini, MacDonald, and Langrock 2017 they work with Poisson-HMMs throughout the book. The choice to work with the normal distribution was made because the Normal-HMM defined above is among the most commonly used due to the fact that it accurately captures the distribution of observations for many natural phenomena. Furthermore, since this thesis draws inspiration from Zucchini, MacDonald, and Langrock 2017, the Normal-HMM was also chosen to focus the research on a model not as well covered in it.

3.2 Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) are a class of techniques and algorithms used for sampling from probability distributions using Markov chains, typically when the distribution cannot be directly sampled from, often because it is unknown or it is complex. **Metropolis Hastings** and **Gibbs Sampling** are two of the most commonly used techniques and performance when sampling from the the posterior distributions HMM parameters will be the focus of this thesis. Given observations $x_1, ..., x_T$, a fixed number of states m, variance σ^2 and the prior distributions of parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Gamma}$ the objective is to estimate the corresponding posterior distributions using these techniques.

3.3 Data Generation

The difficulty of using HMM data is that the states are unobserved. To overcome this obstacle we have generated our own data from which we know what state has 'influenced' each observed outcome. We first specify the known parameters m and σ^2, as well as the unknown parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Gamma}$ for which we are aiming to generate Markov chains for in order to sample from their posterior distributions. We also specify the length of the time-series $T \in \mathbb{N}$.

Using these parameters as inputs the data is generated through the following process:

- Find the stationary distribution $\boldsymbol{\delta}$ such that $\boldsymbol{\delta\Gamma} = \boldsymbol{\delta}$
- Sample initial state c_1 from $\{1, \dots, m\}$, with each state's probability corresponding to its indexed position in $\boldsymbol{\delta}$
- Generate an initial observation x_1 from $\mathcal{N}(\mu_{c_1}, \sigma^2)$
- For $t = 2, \dots, T$
    1. Sample c_t from $\{1, \dots, m\}$, with each state's probability corresponding to its indexed position in the $c_{t-1}{}^{\text{th}}$ row of $\boldsymbol{\Gamma}$
    2. Generate an observation x_t from $\mathcal{N}(\mu_{c_t}, \sigma^2)$

The generated pairs $(c_t, x_t),\ t \in \{1, \dots, T\}$ are stored in a data-frame. Only the generated observations $x_1, \dots, x_T$ will be used in the MCMC algorithms.

3.4 Metropolis Hastings

The Metropolis-Hastings algorithm is named after Nicholas Metropolis, who proposed the algorithm for the case of symmetrical proposal distributions (Metropolis et al. 1953) and W. K. Hastings, who extended it to the more general case (Hastings 1970). It is one of the most common MCMC algorithms used and the general algorithm proceeds as follows:

- Find starting values for $\theta_i^{(1)},\ i = 1, \dots, k$
- Choose a proposal probability density $Q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta})$ that proposes the next sample $\boldsymbol{\theta}^*$ given current sample $\boldsymbol{\theta}$ and let $\pi(\boldsymbol{\theta} \mid \boldsymbol{x})$ be proportional to the target density.
- For $j = 1, \dots, N$
    1. Draw candidate sample $\boldsymbol{\theta}^*$ from $Q(\cdot \mid \boldsymbol{\theta}^{(j)})$
    2. Calculate acceptance probability $\alpha = \min\left(1, \frac{\pi(\boldsymbol{\theta}^* \mid \boldsymbol{x})\, Q(\boldsymbol{\theta}^{(j)} | \boldsymbol{\theta}^*)}{\pi(\boldsymbol{\theta}^{(j)} \mid \boldsymbol{x})\, Q(\boldsymbol{\theta}^* | \boldsymbol{\theta}^{(j)})}\right)$
    3. Draw $u \sim \mathcal{U}(0,1)$
    4. If $u \leq \alpha$ accept the candidate by setting $\boldsymbol{\theta}^{(j+1)} = \boldsymbol{\theta}^*$, otherwise reject the candidate by setting $\boldsymbol{\theta}^{(j+1)} = \boldsymbol{\theta}^{(j)}$

Here $\mathcal{U}$ represents the uniform distribution. Choosing the variance of the proposal distribution can be difficult. If its too small then the acceptance rate will be high but it will not explore the parameter space effectively. If its too large then it will explore the parameter space well but the acceptance rate will be small. Furthermore, as the number of parameters increases there is a decrease in the acceptance rate in the Metropolis Hastings algorithm. To overcome these issues, in this thesis we use a sequential Metropolis Hastings with an additional adaptive variance component (AMH) for the proposal distribution of the $\boldsymbol{\mu}$ parameters $\mathcal{N}(\boldsymbol{\mu}, \Sigma_j)$, where

$$\Sigma_j = \begin{cases} \Sigma_0 & \text{if } j \leq J \\ s_m\, Cov(\boldsymbol{\mu}^{(1)}, \dots, \boldsymbol{\mu}^{(j-1)}) + s_m\, \epsilon\, I_m & \text{if } j > J \end{cases} \tag{3.2}$$

as defined in Haario, Saksman, and Tamminen 2001. Here J is an index for the initial time period that Σ_0 is used for. The scaling parameter $s_m = (2.38)^2/m$ is taken from from Gelman, Roberts, and Gilks 1996, where it was shown that this choice optimizes the mixing properties of the Metropolis search in the case of Normal targets and Normal proposals. $\epsilon > 0$ is a free choice scaling parameter often chosen to be much smaller than s_m. We assume that $\boldsymbol{\mu}$ has multivariate normal prior distribution centered around the starting values $\boldsymbol{\mu}^{(1)}$ with variance Σ_0, which we define to be the identity matrix multiplied by σ_0^2. We also work in the log space, using the log-likelihood and log of the densities of the prior and proposal distributions as the standard likelihood quickly approaches 0 as the number of observations T increases. The formula for the log-likelihood of the HMM can be found in Zucchini, MacDonald, and Langrock 2017, Chapter 2.

For the $\boldsymbol{\Gamma}$ we treat each line as independent, with a Dirichlet prior distribution $\mathcal{D}(a_{i1}, \dots, a_{im})$ and a Dirichlet proposal distribution $\mathcal{D}(a_{i1}+\gamma_{i1}, \dots, a_{im}+\gamma_{im})$ for each row $i = 1, \dots, m$. Throughout the thesis we use $a_{ij} = 0.5$ for $i, j \in \{1, \dots, m\}$ in the prior and proposal distributions. The AMH algorithm follows this procedure:

- Find starting values $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\Gamma}^{(1)}$
- For $j = 1, \dots, N$
    1. Sample $\boldsymbol{\mu}^*$ from $\mathcal{N}(\boldsymbol{\mu}^{(j)}, \Sigma_j)$
    2. Calculate α and update $\boldsymbol{\mu}^{(j+1)}$
    3. For $i = 1, \dots, m$
        - Sample $\gamma_{i1}^*, \dots, \gamma_{im}^* \sim \mathcal{D}(a_{i1} + \gamma_{i1}^{(j)}, \dots, a_{im} + \gamma_{im}^{(j)})$
        - Calculate α and update the ith row of $\boldsymbol{\Gamma}^{(j+1)}$

3.5 Gibbs Sampler

The Gibbs Sampler is a specific case of the Metropolis-Hastings algorithm in which the newly proposed state is always accepted with probability 1. It is named after the physicist Josiah Willard Gibbs and was first described in S. Geman and D. Geman 1984. Let $q_i(\theta_i \,|\, \boldsymbol{\theta}_{-i}, \boldsymbol{x})$ denote the **full conditionals** (a conditional distributions being conditional upon everything except the variable being sampled at each step) of θ_i given data $\boldsymbol{x}$ and $\boldsymbol{\theta}_{-i} = \{\theta_1, \theta_2, \dots, \theta_{i-1}, \theta_{i+1}, \dots, \theta_k\}$ (where the index $-i$ denotes all components of that vector except i). To show that the acceptance rate is 1, first note that

$$\pi(\boldsymbol{\theta} \,|\boldsymbol{x}) = \pi(\theta_i, \boldsymbol{\theta}_{-i} \,|\, \boldsymbol{x}) = q_i(\theta_i \,|\, \boldsymbol{\theta}_{-i}, \boldsymbol{x})\pi(\boldsymbol{\theta}_{-i} \,|\, \boldsymbol{x}) \tag{3.3}$$

Therefore using this we see that the acceptance probability described for the Metropolis-Hastings algorithm becomes

$$\begin{aligned}\alpha &= \min\left(1,\ \frac{\pi(\boldsymbol{\theta}^* \mid \boldsymbol{x}) q_i(\theta_i \mid \boldsymbol{\theta}_{-i},\ \boldsymbol{x})}{\pi(\boldsymbol{\theta} \mid \boldsymbol{x}) q_i(\theta_i^* \mid \boldsymbol{\theta}^*_{-i},\ \boldsymbol{x})}\right)\\ &= \min\left(1,\ \frac{q_i(\theta_i^* \mid \boldsymbol{\theta}^*_{-i},\ \boldsymbol{x})\pi(\boldsymbol{\theta}^*_{-i} \mid \boldsymbol{x}) q_i(\theta_i \mid \boldsymbol{\theta}_{-i},\ \boldsymbol{x})}{q_i(\theta_i \mid \boldsymbol{\theta}_{-i},\ \boldsymbol{x})\pi(\boldsymbol{\theta}_{-i} \mid \boldsymbol{x}) q_i(\theta_i^* \mid \boldsymbol{\theta}^*_{-i},\ \boldsymbol{x})}\right) \qquad (3.4)\\ &= \min(1,\ 1) = 1\end{aligned}$$

where we use (3.3) to obtain (3.4). Also note that $\boldsymbol{\theta}_{-i} = \boldsymbol{\theta}^*_{-i}$. The general algorithm proceeds as follows:

- Find starting values for $\theta_i^{(1)},\ i = 1, \dots, k$
- For $j = 1, \dots, N$
  1. Draw $\theta_1^{(j+1)}$ from $q_1(\theta_1 \mid \boldsymbol{\theta}_{-1}, \boldsymbol{x})$; $\boldsymbol{\theta}_{-1} = \{\theta_2^{(j)}, \dots, \theta_k^{(j)}\}$
  2. Draw $\theta_2^{(j+1)}$ from $q_2(\theta_2 \mid \boldsymbol{\theta}_{-2}, \boldsymbol{x})$; $\boldsymbol{\theta}_{-2} = \{\theta_1^{(j+1)}, \theta_3^{(j)}, \dots, \theta_k^{(j)}\}$
  3. ...
  4. Draw $\theta_k^{(j+1)}$ from $q_k(\theta_k \mid \boldsymbol{\theta}_{-k}, \boldsymbol{x})$; $\boldsymbol{\theta}_{-k} = \{\theta_1^{(j+1)}, \theta_2^{(j+1)}, \dots, \theta_{k-1}^{(j+1)}\}$
  5. Set $\boldsymbol{\theta}^{(j+1)} = \{\theta_1^{(j+1)}, \dots, \theta_k^{(j+1)}\}$

In the Normal-HMM case, we assume that each row of $\boldsymbol{\Gamma}$ is a has an independent prior distribution $\mathcal{D}(1, \dots, 1)$ and that each μ_i has a independent prior distribution $\mathcal{N}(\xi, \kappa^{-1})$, where $\xi = (\max(x_i) + \min(x_i))/2$ and $\kappa = 1/R^2,\ R = \max(x_i) - \min(x_i)$ as used in Rydén 2008. As we do not observe the Markov chain of states $\boldsymbol{C}^{(T)}$ we first simulate the sample paths at each step of the iteration given the previous parameter updates. The algorithm then follows the procedure:

- Given the observed counts $\boldsymbol{x}^{(T)}$ and the current values of the parameters $\boldsymbol{\Gamma}$ and $\boldsymbol{\mu}$, we generate a sample path of the Markov chain of states.
- Use sample path to decompose the observed counts into (simulated) state contributions.
- With the MC sample path available and the state contributions, we can now update $\boldsymbol{\Gamma}$ and then $\boldsymbol{\mu}$

To obtain the probabilities used to simulate the Markov chain of states we make use of the forward probabilities (2.9, 2.10). The joint conditional probability is

$$\mathbb{P}(\boldsymbol{C}^{(T)} \,|\, \boldsymbol{x}^{(T)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu}) = \mathbb{P}(C_T \,|\, \boldsymbol{x}^{(T)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu}) \times \prod_{t=1}^{T-1} \mathbb{P}(C_t \,|\, \boldsymbol{x}^{(T)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu},\, \boldsymbol{C}_{t+1}^{(T)}) \tag{3.5}$$

The states are drawn in the order $C_T,\, C_{T-1}, \ldots,\, C_1$. Using that the forward probability components are $\alpha_t(j) = \mathbb{P}(\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}, C_t = j)$ (B.4), it can be shown that

$$\mathbb{P}(C_t \,|\, \boldsymbol{x}^{(T)}, \boldsymbol{\Gamma},\, \boldsymbol{\mu}) = \frac{\mathbb{P}(C_t,\, \boldsymbol{x}^{(T)},\, |\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})}{\mathbb{P}(\boldsymbol{x}^{(T)},\, |\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})} \propto \alpha_t(C_T), \text{ for } t = 1, \ldots, T \tag{3.6}$$

By first drawing C_T we can simulate the chain backwards by drawing $t = T-1, T-2, \ldots, 1$ by making use of the proportionality argument used in Chib 1996:

$$\begin{aligned}
&\mathbb{P}(C_t \,|\, \boldsymbol{x}^{(T)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu},\, \boldsymbol{C}_{t+1}^{(T)}) \\
&\quad\propto \mathbb{P}(C_t \,|\, \boldsymbol{x}^{(t)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})\, \mathbb{P}(\boldsymbol{x}_{t+1}^{(T)},\, \boldsymbol{C}_{t+1}^{(T)} \,|\, C_t,\, \boldsymbol{x}^{(t)}\, \boldsymbol{\Gamma},\, \boldsymbol{\mu}) \\
&\quad\propto \mathbb{P}(C_t \,|\, \boldsymbol{x}^{(t)},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})\, \mathbb{P}(C_{t+1} \,|\, C_t,\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})\, \mathbb{P}(\boldsymbol{x}_{t+1}^{(T)},\, \boldsymbol{C}_{t+2}^{(T)} \,|\, \boldsymbol{x}^{(t)},\, C_t,\, C_{t+1},\, \boldsymbol{\Gamma},\, \boldsymbol{\mu}) \\
&\quad\propto \alpha_t(C_t)\, \mathbb{P}(C_{t+1} \,|\, C_t,\, \boldsymbol{\Gamma},\, \boldsymbol{\mu})
\end{aligned} \tag{3.7}$$

Using this proportionality argument allows us to simulate the sample path. Once we have this, we are able to then sample from the Dirichlet distribution to generate the $\boldsymbol{\Gamma}$ Markov chain. We use the same updates used in Rydén 2008 that for each row $i = 1, \ldots, m$

$$\gamma_{i1}, \ldots, \gamma_{im} \,|\, \boldsymbol{C}^{(T)} \sim \mathcal{D}(n_{i1} + 1, \ldots, n_{im} + 1) \tag{3.8}$$

where $n_{ij} = \#\{1 < t \leq T \,:\, C_{t-1} = i,\, C_t = j\}$ is the number of transitions from state i to state j. Each row is conditionally independent of each other and $\boldsymbol{\mu}$ given the Markov chain of states. We also use the same updates for the $\boldsymbol{\mu}$ parameters as specified in Rydén 2008, such that for $i = 1, \ldots, m$

$$\mu_i \,|\, \boldsymbol{x}^{(T)}, \boldsymbol{C}^{(T)} \sim \mathcal{N}\left(\frac{S_i + \kappa\xi\sigma^2}{n_i + \kappa\sigma^2}, \frac{\sigma^2}{n_i + \kappa\sigma^2}\right) \tag{3.9}$$

where $S_i = \sum_{k\,:\,C_k=i} x_k$ is the sum of all observations generated whilst in state i and $n_i = \#\{1 < t \leq T \,:\, C_t = i\}$ is the number of times to state i occurs in the simulated Markov chain of states. Combining all these enables the use of the **backward recursion forward sampling** method (Chib 1996, Section 2.1).

3.6 MCMC Diagnostics

In order to assess how well our MCMC algorithms perform we use metrics and plots to check for **convergence** and **efficiency**. When checking for convergence we are simply looking for indication as to whether or not the chains have converged to a posterior distribution. To access this we look at histograms and **trace-plots**. We also use the **Gelman-Rubin statistic** to check for convergence. If the chains have not converged then the MCMC has failed to generate samples from the correct posterior distribution. When checking efficiency we are looking to see if we have run our algorithms for long enough after we have converged. To do so we check the **auto-correlation function** (ACF) and calculate the **effective sample size** (ESS).

Additionally, for the AMH algorithm we also need to check the acceptance rate. There are two competing forces at work here which we must try to balance. If the variance of the proposal distribution is too large then the acceptance rate will be small and the Markov chain will move very slowly and likely not converge. If the variance of the proposal is too small then the acceptance rate will be too large as the generated Markov chain will move quickly but not very far, resulting in it being highly correlated and thus reducing efficiency.

3.7 Trace-plots

Trace-plots simply plot the generated chain against the number of iterations. A good chain will have rapid mixing and the stationary distribution will be reached quickly from the initial starting point. If the chain shows a lack of mixing or does not move from the initial starting point to the correct posterior distribution then this is an indicator of something wrong within the algorithm, e.g. proposal variance either too large or too small.

Figure 3.1 shows two examples of trace-plots. Figure 3.1(a) shows that after

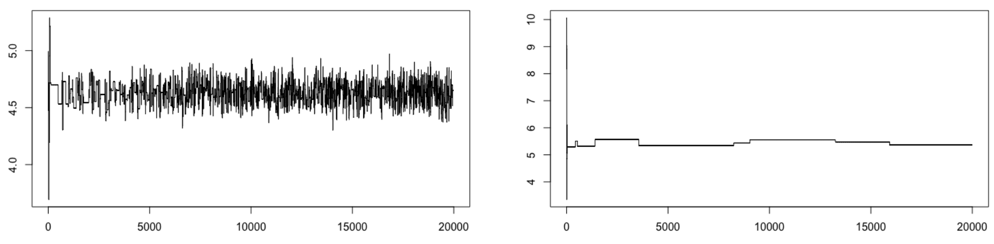

(a) Good Example (b) Bad Example

Figure 3.1: Trace-plot Examples

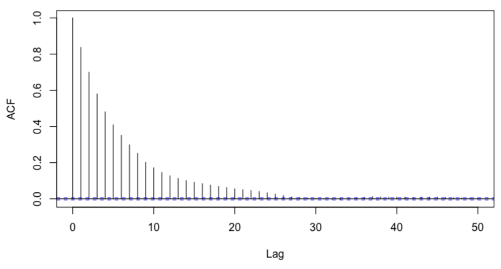
an initial burn-in period of around 10,000 the trace-plot begins mixing well and has moved far away from the initial value, indicating convergence. In this example the acceptance rate was around 23%. Figure 3.1 (b) shows that whilst the chain has moved away from the initial values, the chain is mixing incredibly slowly and is not converging well. This example had an acceptance rate of $< 1\%$.

When plotting the trace-plots we will often use a technique called **thinning** where we plot every i^{th} sample (e.g. 10^{th}, 100^{th} etc.) from the chain instead of all of the samples. When there is a large amount of samples generated the trace-plot can become too clustered to interpret and it also becomes computationally expensive to produce. Thinning resolves this issue. In this thesis we use thinning by plotting every 100^{th} sample in the trace-plots.

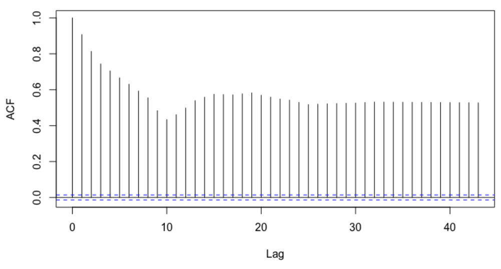
3.8 Auto-Correlation Function

The **auto-correlation function** (ACF) at lag k, denoted ρ_k, is the correlation in the Markov chain between points that are k iterations apart. The estimated auto-correlation function $\hat{\rho}_k$ from the chain is often plotted against k to access how large the within-chain correlation is, as shown in Figure 3.2.

Figure 3.2 (a) shows the ACF of a chain with low correlation, as the correlation between samples quickly decays and reaches 0 by lag 30. 3.2 (b) shows a highly correlated chain within which the correlation fails to decay significantly at all even after 40 lags.



(a) Good Example



(b) Bad Example

Figure 3.2: ACF Examples

3.9 Effective Sample Size

The **effective sample size** (ESS) was defined in Kish 1965 as the original sample size divided by the design effect to reflect the variance from the current sampling design as compared to what would be if the sample was a simple random sample. Given dependent samples N, the number of independent samples is replaced with the ESS $\mathrm{N}_{\mathrm{ess}}$, which is the number of independent samples with the same estimation power as the N auto-correlated samples.

Suppose we have independent samples $x_1, \ldots, x_N$ drawn from a distribution with mean μ and variance σ^2. The estimated mean is $\hat{\mu} = \sum_{i=1}^{N} x_i$ and the variance of $\hat{\mu}$ is $\mathbb{V}\mathrm{ar}(\hat{\mu}) = \frac{\sigma^2}{N}$. However, if the observations are correlated then the variance of $\hat{\mu}$ will be higher. $\mathrm{N}_{\mathrm{eff}}$ is defined to be the unique value such that $\mathbb{V}\mathrm{ar}(\hat{\mu}) = \frac{\sigma^2}{\mathrm{N}_{\mathrm{ess}}}$.

3.10 Gelman-Rubin Statistic

The **Gelman–Rubin statistic** (Gelman and Rubin 1992; Brooks and Gelman 1998) evaluates MCMC convergence by analyzing the difference between multiple Markov chains. The convergence is assessed by comparing the estimated between-chains and within-chain variances for each model parameter. Large differences between these variances indicate non-convergence.

Firstly, $L > 1$ parallel chains are run. Convergence is diagnosed after burn-in has been discarded and the output from all chains is indistinguishable. The initial values are chosen to be over-dispersed, i.e. larger than the target distributions variance. For a model parameter θ, let $\{\theta_t^{(l)}\}_{t=1}^{N}$ be the l^{th}

simulated chain, $l = 1, \ldots, L$. Let $\bar{\theta}^{(l)}$ and $(s^2)^{(l)}$ be the sample posterior mean and variance of the l^th chain, and let the overall sample posterior mean be $\bar{\theta} = \frac{1}{L}\sum_{l=1}^{L} \bar{\theta}^{(l)}$. The between-chains and within-chain variances are given by

$$B = \frac{N}{L-1}\sum_{l=1}^{L}(\bar{\theta} - \bar{\theta^{(l)}})^2$$

$$W = \frac{1}{N}\sum_{l=1}^{L}(s^2)^{(l)}$$

Under certain stationarity conditions, the pooled variance

$$\hat{V} = \frac{N-1}{N}W + \frac{L+1}{L \times N}B$$

is an unbiased estimator of the marginal posterior variance of θ (Gelman and Rubin 1992). The potential scale reduction factor (PSRF) is defined to be the ratio of $\hat{V}$ and W. If the L chains have converged to the target posterior distribution, then PSRF should be close to 1. Brooks and Gelman 1998 corrected the original PSRF by accounting for sampling variability as follows:

$$R = \sqrt{\frac{(d+3)\widehat{V}}{(d+1)W}} \tag{3.10}$$

where $d = \frac{2 \times \hat{V}}{\mathbb{V}\mathrm{ar}(\hat{V})}$ is the degrees of freedom estimate of a t-distribution. If R is much greater than 1 this indicates a lack of convergence. We tend to use the upper bound of the 95% credible interval (C.I) of R and the general rule-of-thumb is that the chains have converged if the upper bound is < 1.2.

Chapter 4

Results

4.1 Bayesian Inference of a 2-State Hidden Markov Model

Starting with a relatively simple case, this HMM has parameters $\boldsymbol{\mu} = (5, 30)$ and $\boldsymbol{\Gamma} = \begin{bmatrix} 0.7 & 0.3 \\ 0.35 & 0.65 \end{bmatrix}$ from which we generate 200 observations. We will denote this *HMM 1* and its density can be observed in Figure 4.1. We run the AMH algorithm for 100,000 iterations to generate Markov chains of $\boldsymbol{\mu}$ and $\boldsymbol{\Gamma}$. We start with initial values of $\boldsymbol{\mu}^{(1)} = (10, 20)$ and $\gamma_{ij}^{(1)} = 1/2$, for $i, j \in \{1, 2\}$. For the AMH, we set $\sigma_0^2 = 5$, $\epsilon = 0.001$ and $J = 1000$. This achieves an acceptance rate of 25% for the $\boldsymbol{\mu}$ parameters and 20% for both rows of the $\boldsymbol{\Gamma}$ matrix. We use a burn-in of 10,000 iterations.

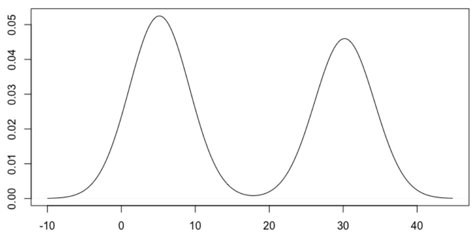
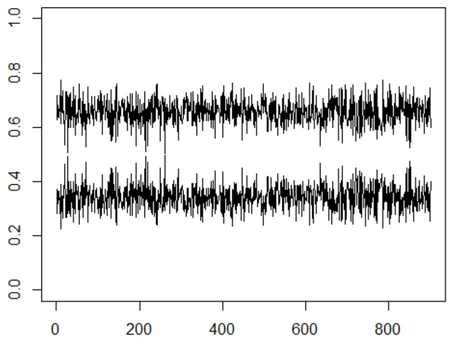
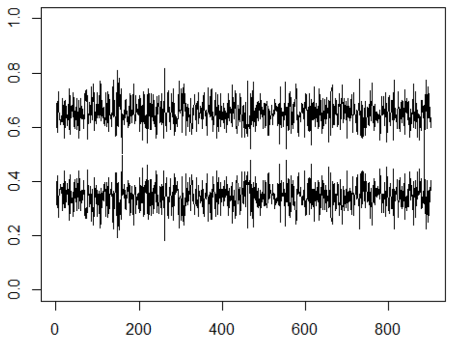


Figure 4.1: Density of *HMM 1*

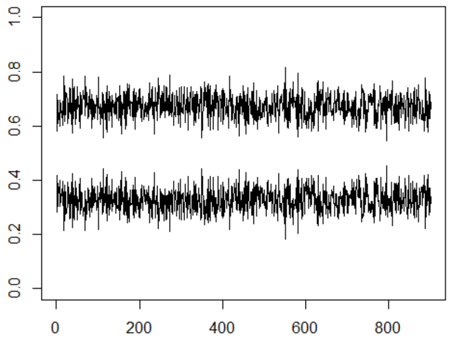
Figure 4.2 (a) shows that after discarding the burn-in, the γ_{11} and γ_{12} converge to their actual values very well, with good mixing and a low variance of 0.002. Similarly, μ_1 and μ_2 converge to 5 and 30 respectively with a very low variance of 0.01 and 0.02 respectively. The trace-plots in 4.2 (c) also show that γ_{21} and γ_{22} are centered around 0.35 and 0.65 respectively. All trace-plots show good mixing of the samples and indicate that the AMH has generated chains that have converged.



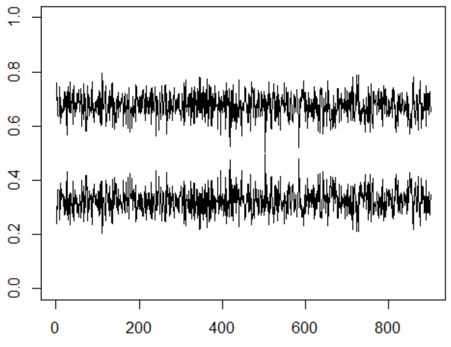
(a) AMH: γ_{11} and γ_{12}



(b) Gibbs: γ_{11} and γ_{12}



(c) AMH: γ_{21} and γ_{22}

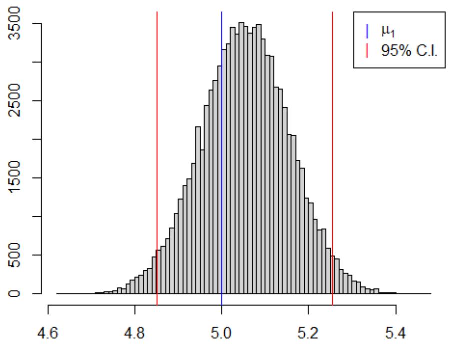


(d) Gibbs: γ_{21} and γ_{22}

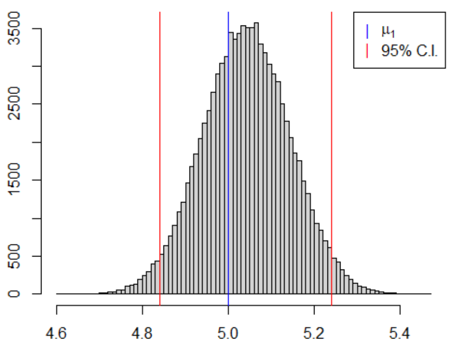
Figure 4.2: Thinned trace-plots of $\mathbf{\Gamma}$ for *HMM 1*

We also run the Gibbs sampler for 100,000 iterations and discard a burn-in of the same size. Figure 4.2 (b) shows that γ_{11} and γ_{12} have good mixing and are centered around the correct values. Figure 4.2 (d) also indicates that γ_{21} and γ_{22} have converged as they have good mixing and are centered around the correct values. The trace-plots of chains generated by both algorithms look very similar, indicating that they are each converging to the same posterior distributions. We see the same for the chains of $\boldsymbol{\mu}$ parameters generated by both MCMC algorithms, with good mixing in the trace-plots and convergence to the correct posterior distributions.

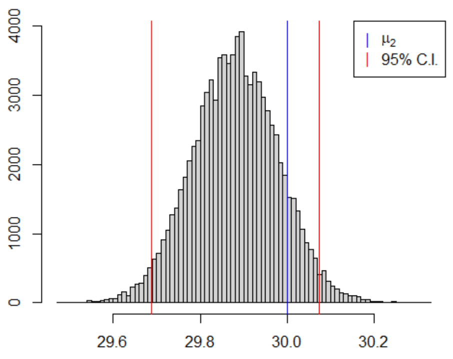
Figure 4.3 shows the histograms of the $\boldsymbol{\mu}$ parameters generated by the two



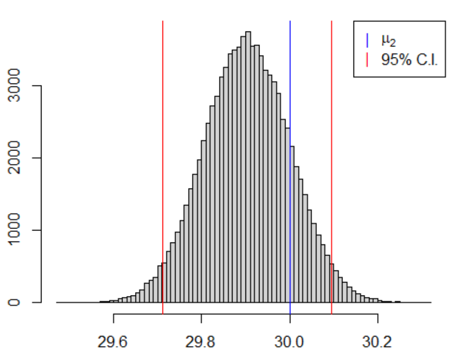
(a) AMH: μ_1



(b) Gibbs: μ_1



(c) AMH: μ_2

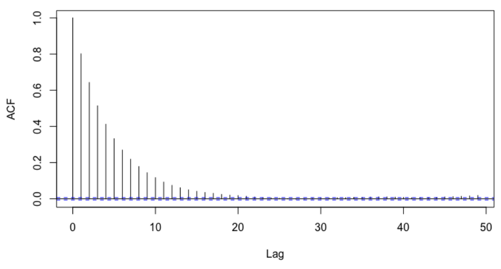


(d) Gibbs: μ_2

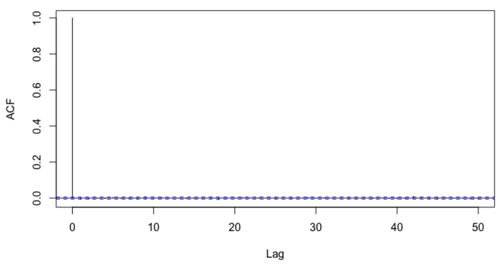
Figure 4.3: Histograms of $\boldsymbol{\mu}$ for *HMM 1* with 95% credible interval (C.I)

algorithms in *HMM 1*. All the actual values lie within the 95% C.I of the samples generated by both AMH and Gibbs. Figures 4.3 (a) and (b) show that the distributions for μ_1 generated by both algorithms are centered around values which are higher than the true value, whilst Figures 4.3 (c) and (d) show the opposite for μ_2. This is due to the initial value choice we made for $\boldsymbol{\mu}$ and the fact that the prior distribution is centered around it which is influencing this outcome.

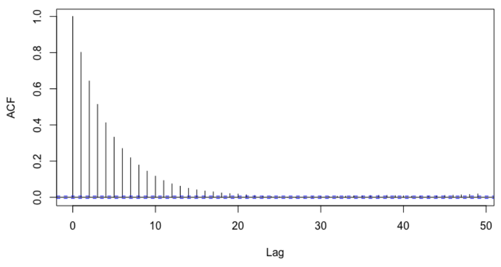
The histograms for the $\boldsymbol{\Gamma}$ parameters also show that the true values lie within the 95% C.I of the generated samples. We re-run each algorithm twice more in order to generate multiple-chains that enable the use of the Gelman-Rubin



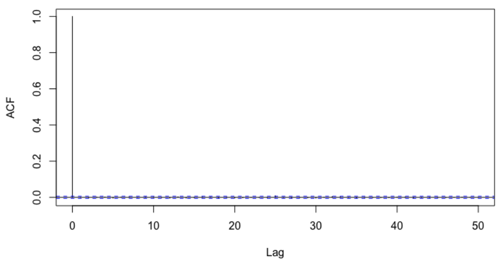
(a) AMH: μ_1



(b) Gibbs: μ_1



(c) AMH: μ_2



(d) Gibbs: μ_2

Figure 4.4: ACF of $\boldsymbol{\mu}$ for *HMM 1*

statistic. In doing so, we see that the upper bound of the 95% C.I for this statistic does not exceed 1 for any of the variables simulated by either the AMH or Gibbs sampler, providing further evidence of convergence.

Inspecting the auto-correlation function (ACF) of each of the samples we can see that there is low auto-correlation in the samples which indicates good efficiency in the samples produced by both algorithms. However, across all chains generated by the Gibbs sampler the auto-correlation disappears after 1 lag, whereas those generated by the AMH take longer to decrease, as shown in Figure 4.4. These figures demonstrate that the Gibbs sampler generates chains with much higher efficiency than the AMH in this case.

Another big difference between the two MCMC algorithm is shown when we look at their ESS and computational time, using the *proc.time()* function in R to monitor their run-time. For the *HMM 1* data-set, the AMH algorithm has a elapsed time of 1,076.08 seconds whereas the Gibbs sampler has a much smaller elapsed time of 400.48 seconds. The Gibbs sampler also generates samples with much higher ESS than the AMH, attaining an average ESS of 100,378.61 compared to the AMH's 11,074.93. In this scenario the Gibbs sampler has a much shorter runtime and a much larger ESS.

4.2 Method Comparison

We now generate various HMMs differing in number of states, number of observations and variance. The same methodology used in the previous section will be followed in order to check for convergence to the correct distribution and efficiency of the Markov chains generated. However, due to the number of different HMM data-sets we will be running the two algorithms on, it is not feasible to analyse every trace-plot and ACF plot in such depth as we previously did. Instead key results will be highlighted and explored.

We will generate data for 2, 3 and 4 state HMMs, with variances of 1 and 10. This will be done in order to access how well the two MCMC algorithms perform when the number of states (m), and thus number of parameters, increase as well as the impact of variance on their ability to sample from the posterior distributions. For each of these HMMs the number of data-points generated will be 100, 200 and 300 in order to access how the number of observations T affect their performance. In total there are 18 different data-sets which will be used. For each of these, we will run both MCMC algorithms for 100,000 iterations for 3 times in total to enable the use of the Gelman-Rubin statistic to test for convergence.

The 2-state HMM will use the same $\boldsymbol{\mu}$ and $\boldsymbol{\Gamma}$ parameters as in Section 3.1. The 3-state HMM will use parameters $\boldsymbol{\mu} = (5, 10, 20)$ and

$$\boldsymbol{\Gamma} = \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.35 & 0.5 & 0.15 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}.$$

The 4-state HMM will use parameters $\boldsymbol{\mu} = (2, 10, 20, 25)$ and

$$\boldsymbol{\Gamma} = \begin{bmatrix} 0.6 & 0.2 & 0.1 & 0.1 \\ 0.3 & 0.4 & 0.1 & 0.2 \\ 0.1 & 0.2 & 0.5 & 0.2 \\ 0.1 & 0.1 & 0.3 & 0.5 \end{bmatrix}.$$

To compare all of the combinations simultaneously the **effective sample size per second** metric is used, which is calculated by dividing the average ESS across all parameters by the elapsed run-time measured in seconds. If the metric is large this is an indication of a large ESS and short computation time. This metric allows us to compare the Markov chains generated by the MCMC algorithms for all combinations of HMMs. These are displayed in

Table 4.1.

Variables		AMH			Gibbs		
σ^2	T	2-states	3-states	4-states	2-states	3-states	4-states
1	100	18.38	7.75	2.99	725.76	638.21	629.88
	200	10.29	3.90	1.61	334.88	348.67	68.18
	300	8.8	2.13	0.56	228.35	235.17	10.18
10	100	20.74	6.63	2.81	575.36	327.99	7.53
	200	12.80	4.35	1.75	358.06	118.14	4.57
	300	9.0	2.72	0.73	256.04	84.11	3.80

Table 4.1: Effective sample size per second

For all data-sets, when running the AMH algorithm the prior variance is set at $\sigma_0 = 5$. We also use $\epsilon = 0.001$ and $J = 1000$. For all 2-state HMMs the intial values used are the same as defined before. For 3-state HMMs the initial values used are $\boldsymbol{\mu}^{(1)} = (10, 20, 30)$ and $\gamma_{ij}^{(1)} = 1/3$, for $i, j \in \{1, 2, 3\}$. For 4-state HMMs the initial values used are $\boldsymbol{\mu}^{(1)} = (10, 20, 30, 40)$ and $\gamma_{ij}^{(1)} = 1/4$, for $i, j \in \{1, 2, 3, 4\}$.

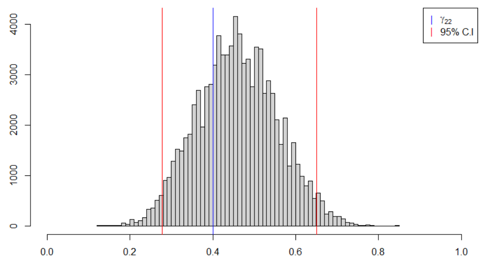
4.3 Analysis of Key Results

Table 4.1 shows that overall the Gibbs sampler tends to have a much higher ESS per second than the AMH algorithm. This is due to a combination of a faster CPU run-time and a much higher ESS in each of the Markov chains generated for the parameters. The degradation in performance is as severe for the AMH as it is for the Gibbs in relative terms. The AMH's range of performance is from 18.38 to 0.73, which is a approximately 96% decrease in ESS per second from its best to its worst performance. But this only amounts to a The Gibbs' range of performance is from 725.76 to 3.80, an approximately 99% decrease in ESS per second from its best to its worst performance. Whilst the decrease in performance is almost identical in percentage terms, the magnitude of the reduction is much larger for the Gibbs than the AMH.

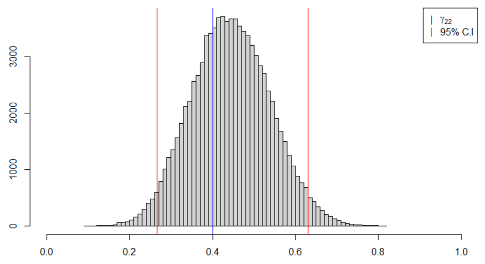
Focusing on the acceptance rate, due to the adaptive component the acceptance rate of the $\boldsymbol{\mu}$ parameters never falls below 20% even for the 4-state HMM data. However, for the $\boldsymbol{\Gamma}$ parameters it falls as low as 10% in the case of the 4-state HMM data with $T = 300$.

4.3.1 Effect of Observations

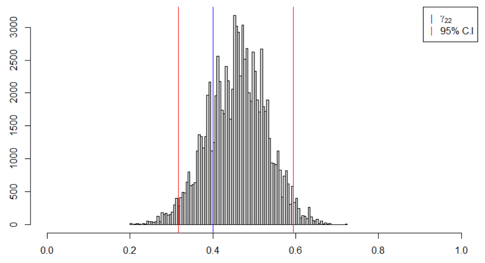
When T is increased the first obvious result is shown in Table 4.1, where the ESS per second decreases as T increases in all scenarios. For the computational



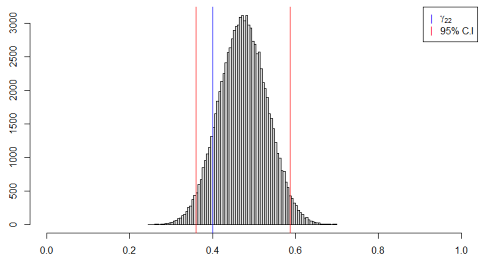
(a) AMH: $T = 100$



(b) Gibbs: $T = 100$



(c) AMH: $T = 300$

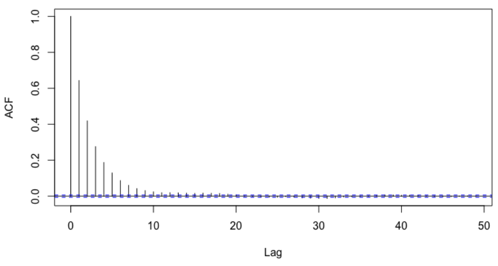


(d) Gibbs: $T = 300$

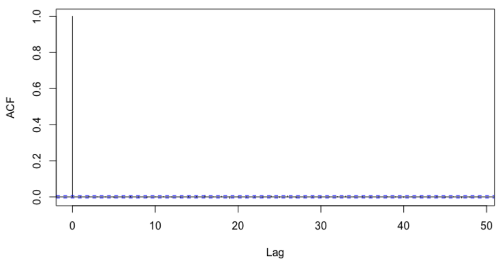
Figure 4.5: Histograms of γ_{22} for 4-state HMM with $\sigma^2 = 1$

time in particular, increasing the number of observations, T, increases the run-time of both algorithms. For example, in the 2-state HMM with $\sigma^2 = 1$, when $T = 100$ the computation time on the CPU to generate 100,000 samples in the Markov chain using the AMH takes $\sim$350 seconds and using the Gibbs sampler takes $\sim$130. Increasing T to 300 increases the computation time to $\sim$1250 and $\sim$430 for the AMH and Gibbs sampler, respectively. For the AMH this is because more samples means more iterations of the forward algorithm in the likelihood function. Furthermore, the increase in observations decreases the probability generated by the likelihood function, which in turn reduces the probability of acceptance in the AMH algorithm. For the Gibbs sampler this is because the sample path generator has a greater number of paths to generate for the hidden Markov chain.

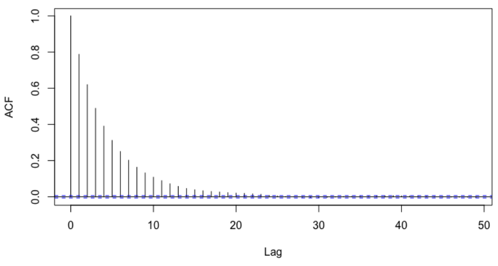
Another effect of increasing T is a smaller credible interval range. For example, looking at the 3-state HMM with $\sigma^2 = 1$ we see that for μ_1 the 95% C.I estimated by the AMH is 0.66 when $T = 100$, 0.41 when $T = 200$ and 0.32 when $T = 300$. We see a similar decrease in the C.I range estimated by the Gibbs sampler, with the 95% C.I ranges estimated to be 0.65, 0.45 and 0.36 when $T = 100$, $T = 200$ and $T = 300$ respectively. This is to be expected, since with a larger data-set the posterior distribution tends to have less 'uncertainty' around its estimates of the random variable, thus resulting in lower variance. This is a pattern we see across all of the HMM scenarios



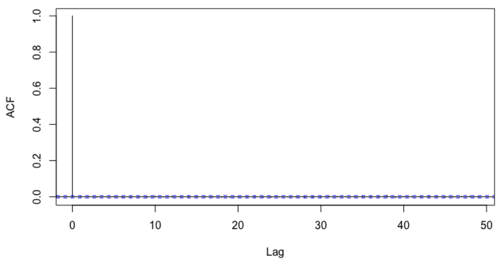
(a) AMH: $T = 100$



(b) Gibbs: $T = 100$



(c) AMH: $T = 300$



(d) Gibbs: $T = 300$

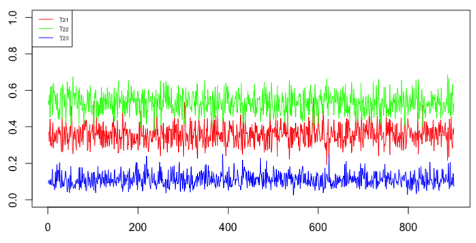
Figure 4.6: ACF of γ_{21} for 2-state HMM with $\sigma^2 = 10$

as T increases, as shown in Figure 4.5. By comparing the histograms on the same scale we see that both the AMH and Gibbs sampler return posterior distributions for γ_{22} with smaller variances as the number of observations, T, increases from 100 to 300, as would be expected if we could sample directly. The Gelman-Rubin statistic remains unchanged by an increase in T, with the upper bound of its 95% C.I going no higher than 1.04 for any parameter estimated by the AMH under both $T = 100$ and $T = 300$. The same is true for Gibbs, with the upper bound not exceeding 1.03 under both scenarios.

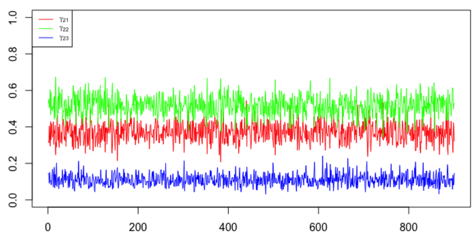
One of the major differences between the two MCMC algorithms is that the auto-correlation of the chains generated by the Gibbs sampler are mostly unaffected by changes in T, whilst those generated by the AMH are. Figure 4.6 shows an example of this, with the ACF of the γ_{21} chain generated by the AMH decaying much slower than the one generated by the Gibbs sampler in both the $T = 100$ and $T = 300$ scenarios. The increase in the number of observations also decreases the rate at which the auto-correlation decays in the chains generated by the AMH, as shown in Figures 4.6 (a) and (c). Conversely, Figures 4.6 (b) and (d) show that the ACF of Markov chains generated by the Gibbs sampler are unchanged by the increase in T. This is a pattern seen throughout all the samples we have generated through both of these MCMC methods.

4.3.2 Effect of Variance

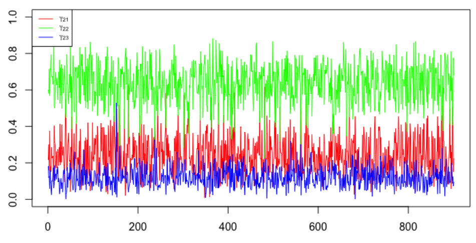
We have assumed that the variance is fixed and known for the HMMs and have used $\sigma^2 = 1$ and $\sigma^2 = 10$ in order to gather data for small and large variance scenarios. The increase in σ^2 results in an increase in the variance of the



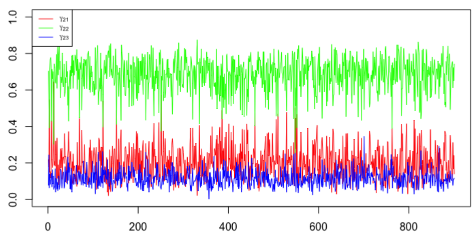
(a) AMH: $\sigma^2 = 1$



(b) Gibbs: $\sigma^2 = 1$



(c) AMH: $\sigma^2 = 10$



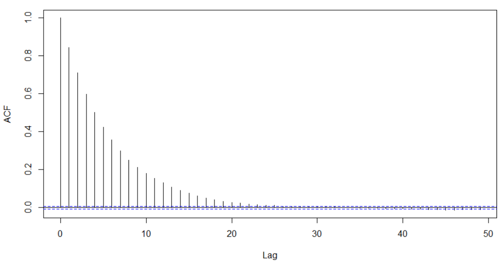
(d) Gibbs: $\sigma^2 = 10$

Figure 4.7: Thinned trace-plots of $\mathbf{\Gamma_2}$ for 3-state HMM with $T = 300$

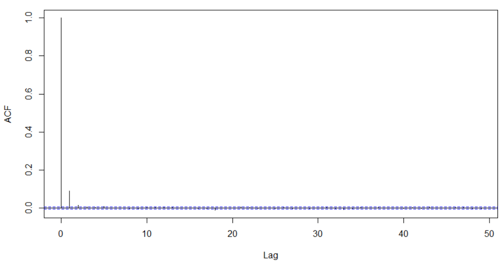
parameter's posterior distributions. For example, in the 3-state HMM with

$T = 200$, the 95% C.I range for μ_2 is estimated to be 0.39 by the AMH and 0.38 by the Gibbs sampler when $\sigma^2 = 1$. When σ^2 is increased to 10, the 95% C.I range estimate increases to 1.26 and 1.31 for chains generated by the AMH and Gibbs sampler, respectively. This is a pattern seen for all parameters under each different HMM scenario and is expected as increasing the variance of the data for each state is going to result in posterior distributions with higher variances. For the AMH we see that changes in variance does not impact the acceptance rate.

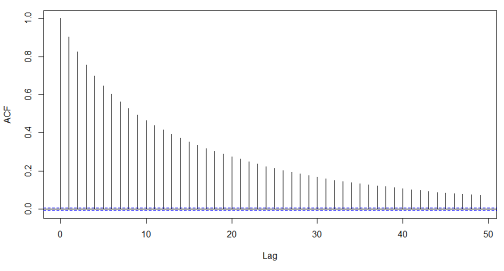
For most scenarios, the increase in variance does not negatively effect the convergence of the Markov chains. For example, in Figure 4.7 the trace-plots still show good mixing around the target value, hence good sampling from the posterior distribution. Here $\mathbf{\Gamma_2}$ represents the second row of the t.p.m $\mathbf{\Gamma}$. The increased variance has resulted in wider mixing as shown in Figure 4.7 (c) and (d), which we expect. The Gelman-Rubin statistics of these chains confirms that they still converge, with the upper bound of its 95% C.I not exceeding 1.03 across all chains generated by both Gibbs and AMH in both the $\sigma^2 = 1$ and $\sigma^2 = 10$ scenarios. Convergence becomes an issue when there is a combination of high variance and $m > 4$.



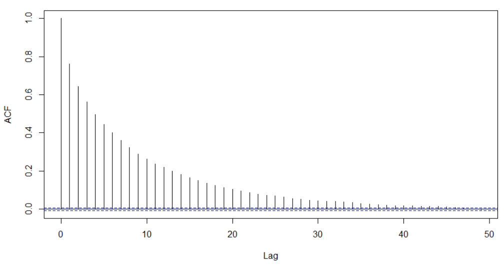
(a) AMH: $\sigma^2 = 1$



(b) Gibbs: $\sigma^2 = 1$



(c) AMH: $\sigma^2 = 10$



(d) Gibbs: $\sigma^2 = 10$

Figure 4.8: ACF of μ_2 for 3-state HMM with $T = 300$

Looking at the effect of variance on the ESS and computation time Table 4.1 shows that for the AMH algorithm the ESS per second is very similar whether the variance is $\sigma^2 = 1$ or $\sigma^2 = 10$. For the Gibbs sampler variance has a greater effect on the ESS and computation time. For the 2-state HMM, when $T = 100$ the increase of σ^2 from 1 to 10 decreases the ESS per second by 150, but for $T = 200$ or $T = 300$ it hardly changes it. However, for the 3 and 4-state HMMs increasing σ^2 from 1 to 10 decreases the ESS per second for all levels of T tested. For the 4-state HMM in particular an increase in the variance results in a very large decrease in the ESS per second.

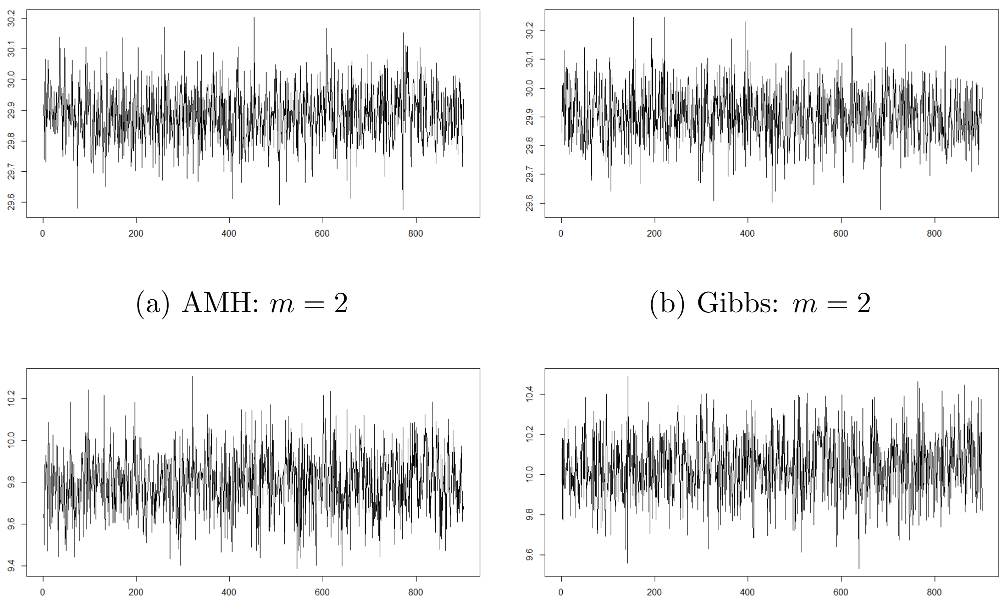
Looking at the auto-correlation, the ACFs for Markov chains generated by both MCMC methods show that the auto-correlation decays at a slower rate when σ^2 increases. Figure 4.8 shows a large increase in the auto-correlation under both MCMC algorithms when σ^2 is increased from 1 to 10. The increase from Figure 4.8 (b) to (d) is especially large. When $\sigma^2 = 10$ the ACFs of the chains generated by the AMH and Gibbs look more similar than they did when $\sigma^2 = 1$. The ACF under the Gibbs sampler still decays at faster rate than the AMH when variance is higher in the $m = 2$ and $m = 3$ cases. However, when $m > 4$ and $\sigma^2 = 10$ the ACF under the AMH decays faster than the Gibbs sampler, which is discussed in Section 4.3.4.

4.3.3 Effect of States

Variable	AMH			Gibbs		
	2-states	3-states	4-states	2-states	3-states	4-states
μ_1	0.30	0.36	0.45	0.31	0.38	0.45
γ_{11}	0.14	0.16	0.21	0.14	0.18	0.21

Table 4.2: Variable 95% C.I range when $T = 300$ and $\sigma^2 = 1$

Table 4.1 shows that increasing the number of states reduces the ESS per second for both AMH and Gibbs. Compared to changing T or σ^2, changing m results in the biggest change of the ESS per second. This is expected, since changing the number of states changes the number of posterior distributions the MCMC algorithms will need to sample from. Changing the number of states increases the computational time for the AMH algorithm but it does not effect the computational time of the Gibbs as much. The run-time of the AMH for a 2-state HMM with $T = 200$ and $\sigma^2 = 10$ is $\sim$1020 seconds and increasing m to 4 increases the computational time to $\sim$1800 seconds.



(a) AMH: $m = 2$

(b) Gibbs: $m = 2$

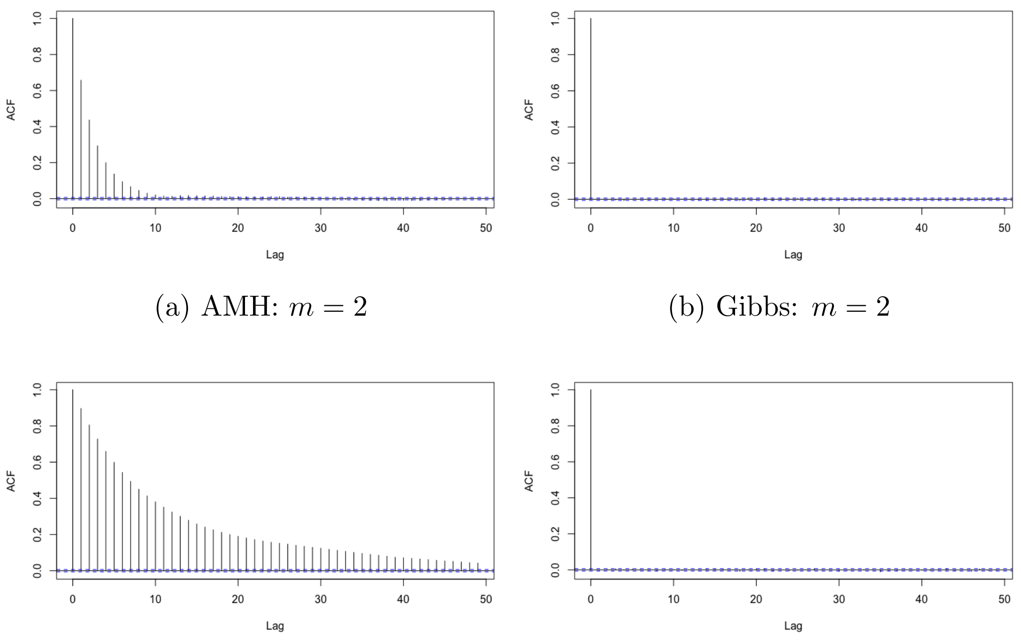
(c) AMH: $m = 4$

(d) Gibbs: $m = 4$

Figure 4.9: Thinned trace-plots of μ_2 for m-state HMMs with $T = 200$ and $\sigma^2 = 1$

For the Gibbs sampler under the same conditions the computational time is ~280 and ~310 seconds for $m = 2$ and $m = 4$, respectively. Increasing the number of states also decreases the effective sample size of each of the chains generated by both MCMC algorithms. Furthermore, for the AMH in particular it decreases the acceptance rate of the $\mathbf{\Gamma}$ parameters, from around 20% to as low as 10%. The acceptance rates of $\boldsymbol{\mu}$ do not decrease due to the adaptive component and remain at around 20% throughout. We also see that the variance in the posterior distributions increase as the number of states increase. Table 4.2 shows the 95% C.I range of μ_1 and γ_{11} for the 2, 3 and 4-state HMMs with $T = 300$ and $\sigma^2 = 1$. As the number of states increase the range increases for both variables, whether they are simulated by Gibbs sampling or AMH, but they increase by approximately the same amount, hence they're both converging to the same distributions.

In most scenarios, increasing the number of states does not impact the convergence of the Markov chains. Figure 4.9 shows the trace-plots of samples of μ_1 for the 2 and 4-state HMMs with $T = 200$ and $\sigma^2 = 1$. Whilst the actual value of μ_1 differs in these scenarios, this example shows that increasing the number of states does not affect the convergence, as both plots are mixing well and centered around the correct values. Checking the Gelman-Rubin statistic also confirms this, as the upper bound of the 95% C.I for each variables statistics does not exceed 1.03, whether generated by Gibbs or AMH. However, the notable exception is the 4-state HMM with $\sigma^2 = 10$ where there is a lack of convergence in the chains generated by the Gibbs sampler. As mentioned previously, this is explored further in Section 4.3.4.



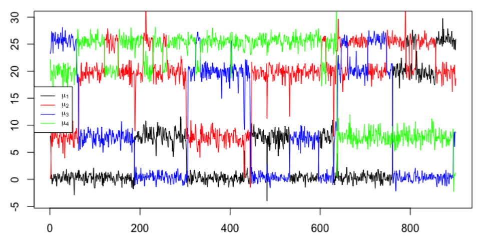
(a) AMH: $m = 2$ (b) Gibbs: $m = 2$

(c) AMH: $m = 4$ (d) Gibbs: $m = 4$

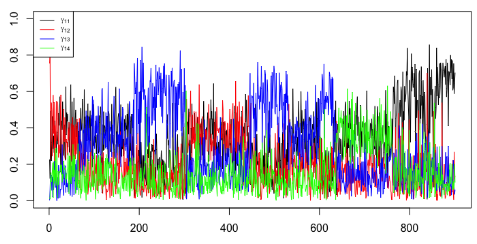
Figure 4.10: ACF of γ_{22} for m-state HMMs with $T = 100$ and $\sigma^2 = 1$

The ACFs of the two MCMC algorithms differ a lot when changing the number of states. Figures 4.10 (b) and (d) show that changing the number of states from $m = 2$ to $m = 4$ has no effect on the ACF of the Markov chain of γ_{22} generated by the Gibbs sampler. However, Figures 4.10 (a) and (c) show that this same change increases the number of lags that it takes the ACF to decay in the chain generated by the AMH, with the ACF reaching 0 at $\sim$15 lags when $m = 2$, but reaching 0 after 50 lags when $m = 4$. Therefore, the AMH is more sensitive to changes in m than the Gibbs sampler, with larger m resulting in longer computational time, higher auto-correlation and lower ESS. However this is only the case when $\sigma^2 = 1$ or when $\sigma^2 = 10$ and $m < 4$. When we look at the scenarios where $\sigma^2 = 10$ and $m = 4$ the chains generated by the Gibbs sampler begin to suffer from **label switching**.

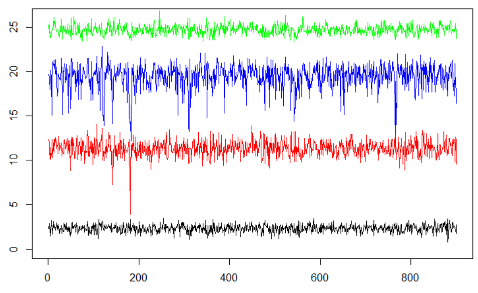
4.3.4 Label Switching



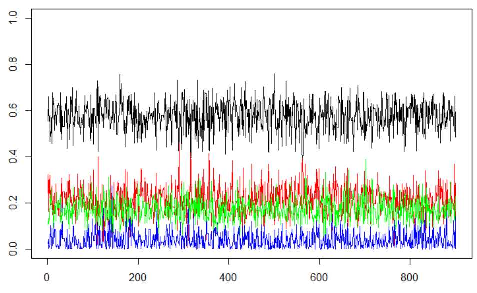
(a) Gibbs: $\boldsymbol{\mu}$



(b) Gibbs: $\boldsymbol{\Gamma_1}$



(c) AMH: $\boldsymbol{\mu}$



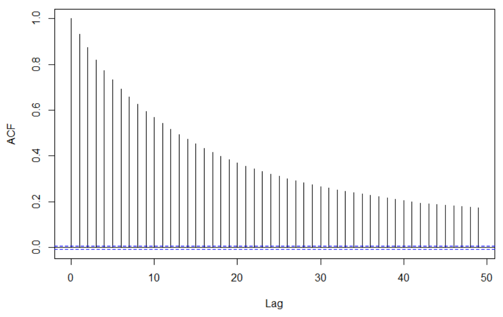
(d) AMH: $\boldsymbol{\Gamma_1}$

Figure 4.11: Thinned trace-plots for 4-state HMM with $\sigma^2 = 10$ and $T = 100$

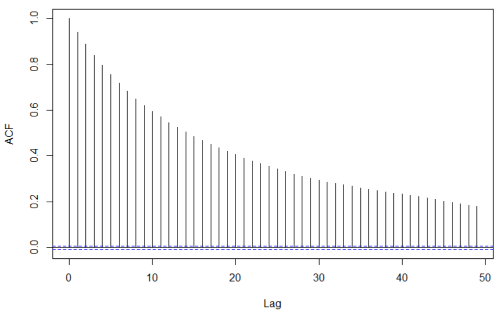
Label switching is a well documented problem that commonly occurs when taking a Bayesian approach to the estimation of hidden Markov model parameters. It was first defined in Redner and Walker 1984 and examples of it can be seen in Frühwirth-Schnatter 2006, Section 3.5.5. In the case that the prior distribution of the model parameters is the same for all states, then both the likelihood and posterior distribution are invariant to permutations of the parameters. This property makes Markov chain Monte Carlo (MCMC) samples simulated from the posterior distribution non-identifiable.

Table 4.1 shows that the Gibbs sampler's ESS per second drops significantly when the data is from 4-state HMM with $\sigma^2 = 10$. Figures 4.11 (a) and (b) show that label switching has occurred, with the trace-plots of the $\boldsymbol{\mu}$ and $\boldsymbol{\Gamma_1}$

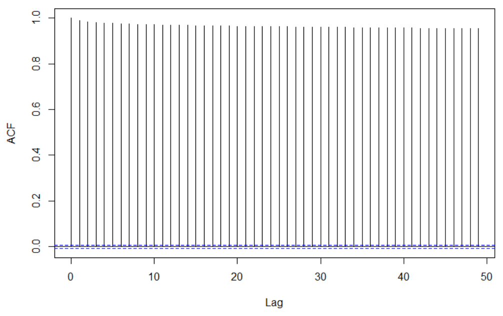
components overlapping. $\mathbf{\Gamma_1}$ represents the first row of the t.p.m $\mathbf{\Gamma}$. Figures 4.11 (c) and (d) show how the trace-plots should look when convergence is achieved and come from the AMH algorithm.



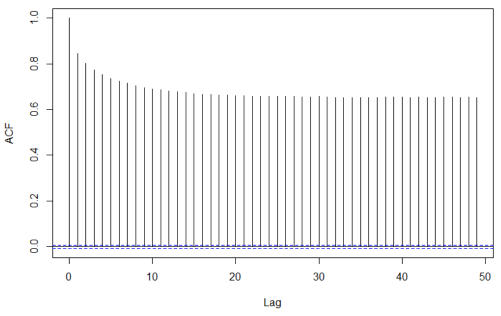
(a) AMH: μ_2



(b) AMH: γ_{44}



(c) Gibbs: μ_2

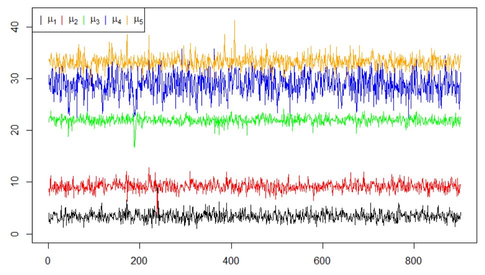


(d) Gibbs: γ_{44}

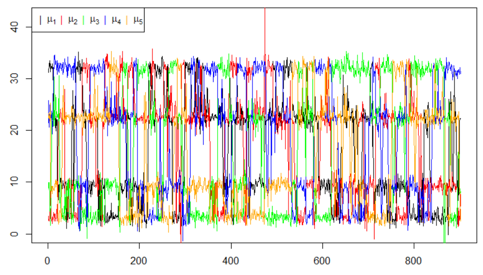
Figure 4.12: ACF for 4-State HMM with $\sigma^2 = 10$ and $T = 100$

The Gelman-Rubin statistic's 95% C.I upper-bounds do not exceed 1.03 for the chains generated by the AMH using data from the $\sigma^2 = 10$, 4-state HMM across the $T = 100$, $T = 200$ and $T = 300$ scenarios. For the Gibbs sampler, the upper-bound does not fall below 8 in any of these cases.

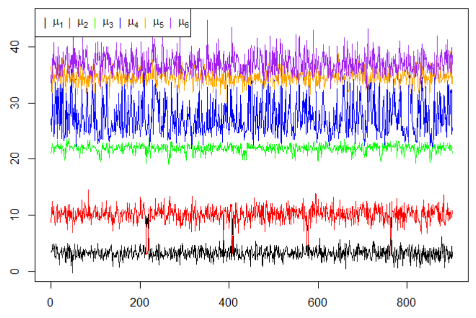
This label switching also has a very adverse impact on the ACF of the affected Markov chains, as shown in Figure 4.12. In the samples of μ_2 and γ_{44} generated by the AMH the auto-correlation decays much faster than those generated by the Gibbs sampler. Figures 4.12 (c) and (d) show that the generated chains are highly inefficient when label switching has occurred, thus confirming that we cannot gather useful information from these samples if they have suffered a label switching problem. However, it is important to note that the auto-correlation is still quite high for the AMH, therefore the Gibbs sampler would likely have outperformed in this area had the label switching problem not occurred.



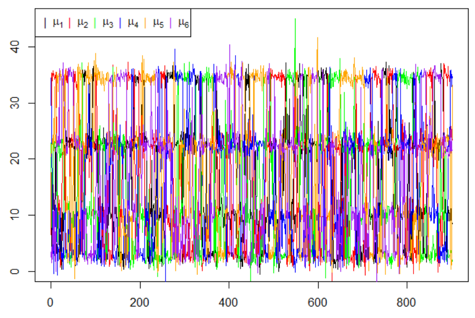
(a) AMH: $m = 5$



(b) Gibbs: $m = 5$



(c) AMH: $m = 6$



(d) Gibbs: $m = 6$

Figure 4.13: Thinned trace-plots for $\boldsymbol{\mu}$. Data is from m-state HMM with $\sigma^2 = 10$ and $T = 100$

To see if the the Gibbs sampler is susceptible to label-switching in high variance high state HMMs, 100 observations were generated from a 5-state and 6-state HMM with $\sigma^2 = 1$ and $\sigma^2 = 10$ and arbitrary unknown parameter values that meet the assumptions. We see again that when $\sigma^2 = 1$ no label switching occurs in either algorithm, but when $\sigma^2 = 10$ label switching occurs in the Gibbs sampler, but not the AMH, as shown in Figure 4.13. This occurred in the repeated re-runs of both MCMC algorithms and the parameters simulated by the AMH algorithm all attain Gelman-Rubin statistics with 95% C.I upper bounds that do not exceed 1.05. This supports the hypothesis that the Gibbs sampler is more sensitive to label-switching in high m and high σ^2 environments, whereas the AMH is able to generate samples which converge.

Chapter 5

Discussion

5.1 Key Findings

In the results section it was shown that for most of the data-sets the Gibbs sampler outperformed the AMH in a number of areas. Whilst both generated chains that converged to very similar posterior distributions regardless of changes in state, variance or observations, the computational time to generate the 100,000 samples was much quicker when using the Gibbs Sampler than the AMH. Furthermore, the auto-correlation within the chains generated by the Gibbs sampler decayed faster than those generated by the AMH under most circumstances tested.

The results also showed the only thing that increased the auto-correlation within the chains generated by the Gibbs sampler was increasing σ^2, as shown in Figure 4.8, whilst the ACF of chains generated by the AMH was shown to be sensitive to changes in m, T and σ^2 (Figures 4.6 and 4.10). The lower auto-correlation resulted in a larger ESS for samples generated by the Gibbs sampler compared to the AMH.

However, label-switching became a problem for the Gibbs sampler when the data was from a 4-state HMM with $\sigma^2 = 10$. When this happened the samples became non-identifiable and the auto-correlation increased to a point where the samples generated were inefficient and unusable. Further testing showed that the Gibbs sampler is susceptible to label-switching when the data is from high variance and high state HMMs, whereas the AMH did not suffer from label-switching under the same scenarios as it possesses a surprisingly positive resistance to label-switching when the data is from high state, high variance

HMMs. This is because the adaptive component of the AMH optimises the proposal distribution to explore a single mode, thus preventing it from exploring multiple modes as we see happening in the Gibbs sampler. Whilst this sounds good, this is also a failure of the AMH to detect that there were multiple modes in the data-sets as the adaptive variance is preventing it from exploring very far. It may also have benefited from the choice of initial values, which enforced the assumption that the $\boldsymbol{\mu}$ components are strictly increasing. Had we used poor initial starting values the adaptive variance may not have prevented the AMH from label-switching.

5.2 Limitations and Improvements

One of the key improvements that could be made is to prevent the label-switching from happening by redesigning how the parameters that are sampled. For instance, one method explored in Zucchini, MacDonald, and Langrock 2017 is to parameterize the model in terms of increments $\boldsymbol{\tau} = \{\tau_1, \dots, \tau_m\}$, where $\mu_i = \sum_{j=1}^{i} \tau_j$. This forces the $\boldsymbol{\mu}$ parameters to be in increasing order and prevents the occurrence of label-switching.

Improvements could also be made by increasing the CPU power. This research was conducted on a 1.6 GHz Dual-Core Intel Core i5 CPU. In order to improve on the computational speed of the AMH algorithm a more powerful CPU could be used in order to offset its slower computational speed. However, this would also improve the speed of the Gibbs sampler and would not decrease the auto-correlation seen in its chains, thus its ESS per second would still be higher in most scenarios tested.

A limitation of this research that must be noted is that the results are conditional on the implementation of the MCMC algorithms. This is impossible to avoid, but important to note since small deviations in the code of these algorithms will likely result in small changes in the measurements we have recorded, such as computational time or ESS.

5.3 Further Research

One obvious research extension is to expand the range of data-sets used to see if the patterns identified are applicable across even more HMMs. We limited the range of data to 18 different scenarios, with an additional 4 scenarios to explore the label switching further. This was done due to time and computational constraints, with data from HMMs with much higher m and T resulting in much longer computational times for the MCMC algorithms. With more time and greater CPU power such scenarios could be explored. HMMs with different underlying distributions could also be explored, such as a Poisson or multivariate normal distribution.

Another expansion of this research could be the inclusion of more MCMC algorithms. The adaptive component was included in order to combat the curse of dimensionality seen as the number of states increase which in turn leads to a decrease in the acceptance rate. Other MCMC algorithms that are designed for large parameter problems could be considered, such as the Hamiltonian Monte Carlo algorithm (Duane et al. 1987). These may scale better when the number of states grows significantly larger.

Additional research could focus on expanding the parameters being estimated. Removing the assumption that σ^2 and m are known parameters could alter the performance of the MCMC algorithms tested. In particular, the posterior distribution of the number of states m has been described as 'notoriously difficult to calculate' (Scott, James, and Sugar 2005). Assessing the best approach through comparison of MCMC algorithms could lead to greater insights into the best way to approach this problem.

Chapter 6

Conclusion

To summarise, we have shown that both of the algorithms generate chains that achieve convergence and efficiency and thus are able to sample from their respective parameter posterior distributions. In particular, we have shown that when the data is not from a high state, high variance HMM, the Gibbs sampler performs better across all of the performance metrics we have looked at. In all of the tested scenarios in which it did not suffer from label-switching we showed that it had a much higher ESS per second than and significantly lower within-chain auto-correlation than the AMH. However, in the scenarios that the data is from a HMM with high variance and number of states $m \geq 4$ we saw that it was susceptible to the label-switching problem. In these cases the AMH algorithm outperforms as it does not suffer from label-switching and whilst it has a much longer computational time, its parameter chains converge to the correct posterior distributions and the auto-correlation decays.

If computational speed is not a concern then the AMH does produce samples that converge to the correct posterior distributions in all scenarios, albeit with higher auto-correlation and lower ESS than the Gibbs. It is simpler to implement compared to the Gibbs since it does not require the simulation of the hidden Markov chain. It also has the unexpected benefit of being resistant to label-switching. But as we have discussed this is because it does not explore as well as the Gibbs sampler due to the adaptive component. If the label-switching can be avoided, e.g. by reparameterisation of $\boldsymbol{\mu}$, then the Gibbs sampler will likely be the superior algorithm in the scenarios in which it previously suffered from label-switching, with faster computational time, higher ESS and lower within-chain auto-correlation than the AMH.

Appendix A

Probability Distributions

We list the density functions and parameters of the probability distributions used in this paper.

A.1 Univariate Normal Distribution

The Normal Distribution is a type of continuous probability distribution for a real-valued random variable.

Support	$x \in \mathbb{R}$
Density	$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$
Parameters	$\mu \in \mathbb{R}$ $\sigma^2 \in \mathbb{R}_{>0}$

A.2 Multivariate Normal Distribution

The Multivariate Normal Distribution (MVN) is a generalization of the one-dimensional (univariate) normal distribution to higher dimensions. Let k denote the number of dimensions.

Support	$\boldsymbol{x} \in \mathbb{R}^k$
Density	$f(\boldsymbol{x}) = det(2\pi\Sigma)^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$
Parameters	$\boldsymbol{\mu} \in \mathbb{R}^k$ $\Sigma \in \mathbb{R}^{k\times k}$

Additionally, it must be noted that Σ must be positive semi-definite, i.e. $\boldsymbol{z}^T\Sigma\boldsymbol{z} \geq 0 \; \forall \; \boldsymbol{z} \in \mathbb{R}$.

A.3 Dirichlet Distribution

The Dirichlet distribution is a multivariate probability distributions parameterized by a vector $\boldsymbol{a} = (a_1, \ldots, a_K) \in \mathbb{R}^K_{>0}$, where K is the number of categories. It is a multivariate generalization of the beta distribution.

Support	$x_1, \ldots, x_K$ where $x_i \in (0,1)$ and $\sum_{i=1}^K x_i = 1$
Density	$f(\boldsymbol{x}) = \frac{1}{B(\boldsymbol{a})} \prod_{i=1}^K x_i^{a_i-1}$
Parameters	$a_1, \ldots, a_K > 0$

A.4 Continuous Uniform Distribution

The continuous uniform distribution describes an experiment where there is an arbitrary outcome that lies between certain bounds. In-between the bounds all outcomes are equally probable.

Support	$x \in [a,b]$
Density	$f(x) = \begin{cases} \frac{1}{b-a} & \text{if } x \in [a,b] \\ 0 & \text{otherwise} \end{cases}$
Parameters	$-\infty < a < b < \infty$

Appendix B

Propositions and Proofs

B.1 A factorization needed for the forward probabilities

First defining and proving a factorisation needed to prove that the forward probability $\alpha_t(i) = \mathbb{P}(\boldsymbol{X}^{(t)},\, C_t = i)$

Proposition 1. *For $t \in \mathbb{N}$,*

$$\mathbb{P}(\boldsymbol{X}^{(t+1)},\, C_t,\, C_{t+1}) = \mathbb{P}(\boldsymbol{X}^{(t)},\, C_t)\, \mathbb{P}(C_{t+1} \,|\, C_t)\, \mathbb{P}(X_{t+1} \,|\, C_{t+1}) \tag{B.1}$$

Proof:
Using equations (2.4), (2.5) and (2.6) from Chapter 2, we have that the joint distribution

$$\mathbb{P}(\boldsymbol{X}^{(t)},\, \boldsymbol{C}^{(t)}) = \mathbb{P}(C_1) \prod_{k=2}^{T} \mathbb{P}(C_k \,|\, C_{k-1}) \prod_{k=1}^{T} \mathbb{P}(X_k \,|\, C_k) \tag{B.2}$$

Now, using (B.2) it clearly follows that

$$\mathbb{P}(\boldsymbol{X}^{(t+1)},\, \boldsymbol{C}^{(t+1)}) = \mathbb{P}(\boldsymbol{X}^{(t)},\, \boldsymbol{C}^{(t)})\, \mathbb{P}(C_{t+1} \,|\, C_t)\, \mathbb{P}(X_{t+1} \,|\, C_{t+1})$$

and summing over $\boldsymbol{C}^{(t-2)}$ we obtain (B.1).

$\square$

B.2 Forward probability result

Now using the above section to prove our desired result about the forward probability by induction. In order to do so we must first note that for $t = 1, 2, \dots, T-1$, $\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t \boldsymbol{\Gamma} \mathbf{P}(x_{t+1})$, where $\mathbf{P}(x_{t+1})$ is the emission probability matrix defined in (2.8). In scalar form each component can be written as:

$$\alpha_{t+1}(j) = \left(\sum_{i=1}^{m} \alpha_t(i) \gamma_{ij} \right) p_j(x_{t+1}) \tag{B.3}$$

Proposition 2. *For $t = 1, 2, \dots, T$ and $j = 1, 2, \dots, m$,*

$$\alpha_t(j) = \mathbb{P}(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)},\, C_t = j) \tag{B.4}$$

Proof:
Since $\boldsymbol{\alpha}_1 = \boldsymbol{\delta}\mathbf{P}(x_1)$ we have that for each component

$$\alpha_1(j) = \delta_j\, p_j(x_t) = \mathbb{P}(C_1 = j)\mathbb{P}(X_1 = x_1 \,|\, C_1 = j) = \mathbb{P}(X_1 = x_1,\, C_1 = j)$$

hence it holds for $t = 1$. Now we show that if it holds for some $t \in \mathbb{N}$ then it also holds for $t + 1$:

$$\begin{aligned}
\alpha_{t+1}(j) &= \sum_{i=1}^{m} \alpha_t(i) \gamma_{ij} p_j(x_{t+1}) && \text{(B.5)}\\
&= \sum_{i} \mathbb{P}(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)},\, C_t = i)\, \mathbb{P}(C_{t+1} = j \,|\, C_t = i)\\
&\times \mathbb{P}(X_{t+1} = x_{t+1} \,|\, C_{t+1} = j)\\
&= \sum_{i} \mathbb{P}(\boldsymbol{X}^{(t+1)} = \boldsymbol{x}^{(t+1)},\, C_t = i,\, C_{t+1} = j) && \text{(B.6)}\\
&= \mathbb{P}(\boldsymbol{X}^{(t+1)} = \boldsymbol{x}^{(t+1)},\, C_{t+1} = j)
\end{aligned}$$

which is our desired result. The line B.5 is justified by B.3 and line B.6 is justified by B.1.

□

Bibliography

Baum, Leonard E. “An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process”. In: 1972.

Baum, Leonard E. and J. A. Eagon. “An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology”. In: *Bulletin of the American Mathematical Society* 73 (3 1967). ISSN: 02730979. DOI: 10.1090/S0002-9904-1967-11751-8.

Baum, Leonard E. and Ted Petrie. “Statistical Inference for Probabilistic Functions of Finite State Markov Chains”. In: *The Annals of Mathematical Statistics* 37 (6 1966). ISSN: 0003-4851. DOI: 10.1214/aoms/1177699147.

Baum, Leonard E., Ted Petrie, George Soules, and Norman Weiss. “A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains”. In: *The Annals of Mathematical Statistics* 41 (1 1970). ISSN: 0003-4851. DOI: 10.1214/aoms/1177697196.

Baum, Leonard E. and George R. Sell. “Growth transformations for functions on manifolds”. In: *Pacific Journal of Mathematics* 27 (2 1968). ISSN: 00308730. DOI: 10.2140/pjm.1968.27.211.

Bhar, Ramaprasad and Shigeyuki Hamori. *Hidden Markov Models Applications to Financial Economics.* Vol. 91. 2012.

Brooks, Stephen P. and Andrew Gelman. “General methods for monitoring convergence of iterative simulations)?” In: *Journal of Computational and Graphical Statistics* 7 (4 1998). ISSN: 15372715. DOI: 10.1080/10618600.1998.10474787.

Cappé, O., E. Moulines, and Tobias Rydén. *Inference in Hidden Markov Models.* English. Germany: Springer, 2005.

Chib, Siddhartha. “Calculating posterior distributions and modal estimates in Markov mixture models”. In: *Journal of Econometrics* 75 (1 1996). ISSN: 03044076. DOI: 10.1016/0304-4076(95)01770-4.

Congdon, Peter. “Bayesian model choice based on Monte Carlo estimates of posterior model probabilities”. In: *Computational Statistics and Data Analysis* 50 (2 2006). ISSN: 01679473. DOI: 10.1016/j.csda.2004.08.001.

Dempster, A. P., N. M. Laird, and Donald B. Rubin. “Maximum Likelihood from Incomplete Data Via the EM Algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39 (1 1977). DOI: `10.1111/j.2517-6161.1977.tb01600.x`.

Duane, Simon, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. “Hybrid Monte Carlo”. In: *Physics Letters B* 195.2 (1987), pp. 216–222. ISSN: 0370-2693. DOI: `https://doi.org/10.1016/0370-2693(87)91197-X`. URL: `https://www.sciencedirect.com/science/article/pii/037026938791197X`.

Fonzo, Valeria De, Filippo Aluffi-Pentini, and Valerio Parisi. “Hidden Markov Models in Bioinformatics”. In: *Current Bioinformatics* 2 (1 2008). ISSN: 15748936. DOI: `10.2174/157489307779314348`.

Frühwirth-Schnatter, Sylvia. *Finite Mixture and Markov Switching Models*. 2006. DOI: `10.1007/978-0-387-35768-3`.

Gelman, Andrew, O. G. Roberts, and R. W. Gilks. “Efficient Metropolis jumping rules”. In: *Bayesian Statistics* 5 (1996).

Gelman, Andrew and Donald B. Rubin. “Inference from iterative simulation using multiple sequences”. In: *Statistical Science* 7 (4 1992). ISSN: 08834237. DOI: `10.1214/ss/1177011136`.

Geman, Stuart and Donald Geman. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6 (6 1984). ISSN: 01628828. DOI: `10.1109/TPAMI.1984.4767596`.

Haario, Heikki, Eero Saksman, and Johanna Tamminen. “An adaptive Metropolis algorithm”. In: *Bernoulli* 7 (2 2001). ISSN: 13507265. DOI: `10.2307/3318737`.

Hastings, W. K. “Monte carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57 (1 1970). ISSN: 00063444. DOI: `10.1093/biomet/57.1.97`.

Khiatani, Diksha and Udayan Ghose. “Weather forecasting using Hidden Markov Model”. In: *2017 International Conference on Computing and Communication Technologies for Smart Nation, IC3TSN 2017* 2017-October (2018). DOI: `10.1109/IC3TSN.2017.8284480`.

Kish, Leslie. “Survey Sampling. By Leslie Kish. (New York: John Wiley & Sons, Inc., 1965. Pp. xvi, 643)”. In: *American Political Science Review* 59 (4 1965). ISSN: 0003-0554. DOI: `10.1017/s0003055400132113`.

Levinson, S. E., L. R. Rabiner, and M. M. Sondhi. “An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition”. In: *Bell System Technical Journal* 62 (4 1983). ISSN: 15387305. DOI: `10.1002/j.1538-7305.1983.tb03114.x`.

Metropolis, Nicholas, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. “Equation of state calculations by fast computing machines”. In: *The Journal of Chemical Physics* 21 (6 1953). ISSN: 00219606. DOI: 10.1063/1.1699114.

Nurfalah, Irfan, Aam Slamet Rusydiana, Nisful Laila, and Eko Fajar Cahyono. “Early warning to banking crises in the dual financial system in Indonesia: The Markov switching approach”. In: *Journal of King Abdulaziz University, Islamic Economics* 31 (2 2018). ISSN: 16584244. DOI: 10.4197/Islec.31-2.10.

Redner, Richard A. and Homer F. Walker. “Mixture Densities, Maximum Likelihood and the EM algorithm.” In: *SIAM Review* 26 (2 1984). ISSN: 00361445. DOI: 10.1137/1026034.

Robert, Christian P., Tobias Rydén, and D. M. Titterington. “Bayesian inference in hidden Markov models through the reversible jump Markov chain Monte Carlo method”. In: *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 62 (1 2000). ISSN: 13697412. DOI: 10.1111/1467-9868.00219.

Robert, Christian P. and D. M. Titterington. “Reparameterization strategies for hidden Markov models and Bayesian approaches to maximum likelihood estimation”. In: *Statistics and Computing* 8 (2 1998). ISSN: 09603174. DOI: 10.1023/A:1008938201645.

Rydén, Tobias. “Em versus Markov chain monte carlo for estimation of hidden Markov models: A computational perspective”. In: *Bayesian Analysis* 3 (4 2008). ISSN: 19360975. DOI: 10.1214/08-BA326.

Scott, Steven L. “Bayesian Methods for Hidden Markov Models”. In: *Journal of the American Statistical Association* 97 (457 2002). ISSN: 0162-1459. DOI: 10.1198/016214502753479464.

Scott, Steven L., Gareth M. James, and Catherine A. Sugar. “Hidden Markov models for longitudinal comparisons”. In: *Journal of the American Statistical Association* 100 (470 2005). ISSN: 01621459. DOI: 10.1198/01621450-4000001592.

Sun, Don X. and Frederick Jelinek. “Statistical Methods for Speech Recognition”. In: *Journal of the American Statistical Association* 94 (446 1999). ISSN: 01621459. DOI: 10.2307/2670189.

Zucchini, Walter, Iain L. MacDonald, and Roland Langrock. *Hidden Markov Models for Time Series*. 2017. DOI: 10.1201/b20790.