Bayesian Computation for Hidden Markov Models

Lindsey Anne Foreman
Imperial College
March 1994

Thesis submitted to the University of London for the degree of Doctor of Philosophy

Abstract

In order to motivate the use of hidden Markov models (HMMs) for classification tasks, a general background to the field of automatic speech recognition (ASR) is provided and used as a point of reference throughout the thesis. The search for improvements to classical template-matching procedures in ASR has led to the adoption of the HMM, which offers a more formal and flexible statistical framework within which to conduct pattern recognition. The HMM is represented by an observable process, associated with which is an underlying unobserved Markov chain state process. A detailed description of the HMM is given along with the problems to be addressed in its implementation, such as parameter estimation, classification procedures and recovery of the unobserved Markov chain state process. Current statistical methods for dealing with these problems are reviewed. These include a version of the EM algorithm used for maximum likelihood parameter estimation, a search procedure called the Viterbi algorithm, which uses dynamic programming techniques to recover the optimal unobserved state sequence underlying the observed data, and methods of comparing observed patterns and HMMs to find the best match for classification purposes. The Viterbi algorithm is then studied in more depth and a generalisation of the algorithm is established. Examples are given to illustrate its usefulness, particularly for classification purposes in handwritten text recognition and in a pharmaceutical application. The concepts underlying the Bayesian philosophy are introduced; in particular, a Bayesian approach to parameter estimation in HMMs. Markov chain Monte Carlo (MCMC) stochastic simulation techniques are discussed, including the Gibbs sampler and Hastings algorithm, which are useful computational tools in the Bayesian analysis of HMMs. Approaches to the design of efficient MCMC samplers are then critically reviewed. These MCMC methods are then implemented and various sampling designs investigated in a Bayesian approach to HMM parameter analysis.

Contents

Introduction 15

1 Introduction to hidden Markov models 18

1.1 Automatic speech recognition 18

1.1.1 What is speech recognition? 18

1.1.2 Knowledge-based versus data-based approaches 20

1.1.3 Template-matching recognition techniques 21

1.1.4 Dynamic time-warping 24

1.1.5 Locally constrained dynamic time-warping 29

1.2 Hidden Markov models 31

1.2.1 General introduction 31

1.2.2 Problems to be addressed 35

1.2.3 Choice of observation distribution 37

1.2.4 Time duration models 38

1.2.5 Other modifications of hidden Markov models 39

1.2.6 Neural networks 39

2 Standard analysis of hidden Markov models 41

2.1 Recovery of state sequence information 41

2.1.1 Introduction 41

2.1.2 The Viterbi algorithm 42

2.2 Classification/recognition 47

2.2.1 Introduction 47

2.2.2 The Baum-Welch algorithm 48

2.2.3 Viterbi scoring 50

2.2.4 Connected-class recognition 51

2.3 Parameter estimation 51

2.3.1 Maximum likelihood and the EM algorithm 51

2.3.2 The Baum-Welch re-estimation procedure 53

2.3.3 Viterbi re-estimation and alternative estimation procedures . . . 58

3 Generalisation of the Viterbi algorithm **60**

3.1 Why is generalisation useful? 60

3.2 The generalised Viterbi algorithm 61

3.3 Examples 68

3.3.1 General examples 68

3.3.2 Handwritten text recognition 72

3.3.3 A classification problem in pharmaceutical research 77

3.4 Conclusions 87

4 Bayesian inference and Markov chain Monte Carlo methods **88**

4.1 The Bayesian philosophy 88

4.2 Bayesian computations 90

4.3 Markov chain Monte Carlo stochastic simulation techniques 92

4.3.1 Introduction 92

4.3.2 Practical implementation of MCMC procedures 96

4.4 Output analysis 99

4.5 The design of MCMC samplers 105

4.5.1 Desirable properties of a MCMC scheme 106

4.5.2 Review of current suggestions in MCMC sampler design 110

4.6 Conclusions 120

5 Bayesian inference for hidden Markov models **121**

5.1 Introduction 121

5.2 Application of the Gibbs sampler algorithm 123

5.3 Practical issues when sampling a HMM 129

5.3.1 Choice of initial values 129

5.3.2 Problems with the Gibbs sampler design 130

5.4 Analysis of sweep strategies 135

5.5 Can utilisation of the generalised Viterbi algorithm improve sampling from $p(\mathbf{s}|\mathbf{y}, H)$? 144

5.6 Application of the Hastings algorithm 152

5.7 Analysis of sampler design in the case of unknown H 165

5.8 Time duration modelling 171

5.9 Conclusions 173

Conclusions **175**

A Appendix to Chapter 2 **180**

A.1 The B-W re-estimation procedure 180

A.2 The B-W scaling procedure 188

B Appendix to Chapter 3 **192**

B.1 Model parameters for the text recognition example 192

B.2 Model parameters for the EEG data example 193

C Appendix to Chapter 5 **196**

C.1 State sequence labels 196

C.2 Simulation of ordered Normal variables 197

References **205**

List of Tables

2.1 The value of S for various values of N and T under a L-R HMM 43

2.2 The value of S for various values of N and T under an ergodic HMM . . 43

2.3 Comparison of the order of calculations required under the Viterbi algorithm (with that under direct evaluation in brackets) 46

3.1 Posterior state sequence information underlying $\mathbf{y}$ = (1 2 1 5 4 2 5 5 5 4) for a L-R model 71

3.2 Posterior state sequence information underlying $\mathbf{y}$ = (4 5 4 4 4 5 4 2 3 3) for a L-R model 72

3.3 Posterior state sequence information underlying $\mathbf{y}$ = (1 1 5 5 2 4 5 5 3 4 2 3 3 2 5) for an ergodic model 73

3.4 Posterior state sequence information underlying $\mathbf{y}$ = (16 17 5) where the true word is BAD 75

3.5 Posterior state sequence information underlying $\mathbf{y}$ = (1 23 4 22 7) where the true word is CHIEF 76

3.6 Posterior state sequence information underlying $\mathbf{y}$ = (16 22 25 22 12 11) where the true word is BEHEAD 76

3.7 Breakdown of misclassified states 83

3.8 Most likely confused states 83

3.9 MAP state sequences underlying 2 sections of the full EEG observation sequence 86

3.10 Alternative classifications for those states varying in the top 20 sequences 86

4.1 Total number of HMM parameters under various specifications of N and K 91

5.1 Most probable state sequences underlying $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$ for Model 1 133

5.2 Eigenvalue comparison of sweep strategies for Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$ 141

5.3 Most probable state sequences underlying $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ for Model 2 142

5.4 Eigenvalue comparison of sweep strategies for Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ 142

5.5 Eigenvalue comparison of sweep strategies for Model 3 with $\mathbf{y} = (2\ 2\ 1\ 1)$ 143

5.6 Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$ 155

5.7 Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ 155

5.8 Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 3 with $\mathbf{y} = (2\ 2\ 1\ 1)$ 156

5.9 Acceptance and movement rates under Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$. 159

5.10 Acceptance and movement rates under Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$. 160

C.1 Comparison of Gibbs sampling and the Hastings algorithm for an ordered bivariate Normal target distribution 203

List of Figures

1.1 Format of a typical template-matching whole-word recogniser 22

1.2 Example of a template distortion 26

1.3 The DTW transition network 29

1.4 Transitions and the corresponding template distortion 30

1.5 Format of a typical HMM speech recogniser 36

2.1 The Viterbi algorithm 45

2.2 The Baum-Welch algorithm 50

3.1 The generalised Viterbi algorithm 69

3.2 Mean observation vectors for the sleep states of a rat 85

5.1 Comparison of histograms for the forward-cycling Gibbs sampler and independent sampling 134

5.2 Exact posterior distribution, $p(\mathbf{s}|\mathbf{y}, H)$, under Model 1 147

5.3 Histograms representing state sequence frequencies under the mixture Hastings algorithm 148

5.4 Autocorrelation structure exhibited under the mixture Hastings algorithm 149

5.5 Comparison of distance measures for the independence Hastings algorithms 161

5.6 Comparison of distance measures for Gibbs sampling and the independence Hastings algorithm 162

5.7 Comparison of distance measures for the Hastings algorithm under various specifications of r 163

5.8 Output sample autocorrelations for the various sampling strategies . . . 164

5.9 Mean ergodic averages for 4 parameters of the unknown model H 167

5.10 SDs about mean ergodic averages for 4 of the unknown model parameters 168

5.11 Autocorrelation plots for Trace(A) 169

5.12 Autocorrelation plots for $\sum_{i=1}^{N} b_{i1}$ 170

To Mum and Dad

Acknowledgements

I would like to thank my supervisor, Professor Adrian Smith, for his continued guidance and encouragement throughout my Ph.D research and for his many valuable suggestions which have helped shape my thesis.

I have also had many helpful discussions with colleagues which have assisted in the understanding and formulation of my ideas; in particular, I would like to thank David Buckle, David Phillips, Dr Jon Wakefield and Dr David Stephens.

Finally, many thanks are due to my family for their support, especially during the writing-up of my thesis.

Notation and terminology

We attempt to keep notation consistent throughout the thesis, although some symbols are used to denote different quantities in different chapters where there is thought to be no risk of ambiguity.

Bold type is used to denote vectors, e.g. $\mathbf{x} = (x_1, x_2, \ldots, x_k)$. In addition,

$$\begin{aligned} \mathbf{x}_{(i)} &= (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \\ \mathbf{x}_{[i]} &= (x_1, x_2, \ldots, x_i) \\ \mathbf{x}^{[i]} &= (x_i, x_{i+1}, \ldots, x_k), \end{aligned}$$

so that $\mathbf{x}_{(i)}$ denotes the vector $\mathbf{x}$ after removal of its i^{th} entry, $\mathbf{x}_{[i]}$ denotes the partial vector up to the i^{th} entry of $\mathbf{x}$ and $\mathbf{x}^{[i]}$ denotes the partial vector comprising entries i up to k of $\mathbf{x}$. Matrices are represented by upper-case Greek or Roman letters such as $A = (a_{ij})$, where a_{ij} denotes the $(i,j)^{th}$ entry of A.

The general notation $p(\cdot)$ is used to denote continuous probability density functions, discrete probability values or any other probability statements, the exact meaning being clear from the argument of $p(\cdot)$. Similarly, $p(\cdot|\cdot)$ is used to denote conditional probability densities, etc. In Chapter 4, the notation $\pi(\cdot)$ is adopted to denote probability densities, etc, associated with the stationary/target distribution of a general Markov chain Monte Carlo (MCMC) sampler in order to distinguish it from the general $p(\cdot)$ notation and maintain consistency with the bulk of the MCMC literature. Furthermore, clarity dictates that identical notation be used for both random quantities and their corresponding realisations, it being clear from the context which is intended.

Additional notation may be listed as follows:

$E(\cdot)$	denotes mathematical expectation,
$\lvert\cdot\rvert$	denotes either the absolute value (when the argument is a real number) or the cardinality (when the argument is a set),
$i!$	denotes the factorial of i, given by $i! = i \times (i-1) \times \ldots \times 1$,
$\sim$	denotes "is distributed as",
$\Re$	denotes the real line.

A summary of commonly used abbreviations is as follows:

ASR	—	automatic speech recognition

DTW	—	dynamic time-warping
TRP	—	time registration path
HMM	—	hidden Markov model
L-R	—	left-right
MLE	—	maximum likelihood estimate
MAP	—	maximum a posteriori
B-W	—	Baum-Welch
MCMC	—	Markov chain Monte Carlo
CGF	—	candidate generating function
SD	—	standard deviation

Introduction

The objective of this thesis is to investigate a general Bayesian approach for making inference when implementing hidden Markov models (HMMs) for use in classification problems.

In order to provide motivation and place HMMs in context, Chapter 1 outlines the background which has led to their use in the field of automatic speech recognition (ASR). Although this particular application is given special emphasis throughout the thesis, all the ideas and methodology are generally applicable and other examples will also be considered.

Within the ASR context, Chapter 1 details the template-matching procedures adopted by most commercial speech recognisers before the advent of HMM ideas and illustrates how the need for improved recognition accuracy and greater modelling flexibillty led naturally to the HMM framework for modelling the speech process. The HMM may be considered as a doubly stochastic process: an underlying unobserved state process, governed by a Markov chain mechanism, which drives an observable process through state-dependent probability distributions. The general format of a HMM is presented and the choices available in its complete specification for particular implementations are discussed.

When implementing HMM ideas, there are 3 main tasks to be addressed; namely, parameter estimation, techniques for classification and optimal recovery of the unobserved state process giving rise to observed data. Chapter 2 reviews standard procedures for dealing with these problems. These include the Viterbi algorithm, a search procedure which employs the recursive techniques of dynamic programming to recover the maximum a posteriori state sequence underlying observed data, and a version of the EM algorithm (called the Baum-Welch re-estimation procedure) for use in classical maximum

likelihood parameter estimation, which is applicable since the HMM may be thought of as an incomplete data model with the unobserved state structure corresponding to missing data.

In Chapter 3, the Viterbi algorithm is revisited and a useful generalisation established. For arbitrary L, the generalised algorithm recovers the set of the L most likely forms for the unobserved state structure underlying observed data, thus providing additional information about the corresponding Bayesian posterior distribution involved. The generalised algorithm's potential to provide more valid and flexible inference about the hidden state process is illustrated with two examples. The first is in the area of handwritten text recognition, where the state information is utilised directly for classification purposes; the second is a pharmaceutical application, where the Viterbi algorithm is compared with a sequential classification procedure on the basis of realised experimental data.

A Bayesian approach to drawing inference about HMM parameters for an unknown model, on the basis of observed data, typically requires the evaluation of multidimensional integrals and this is further complicated by the fact that direct computation of the posterior distribution in the HMM case is not feasible, due to its form as a sum over a large discrete space. The recently discovered (in statistical circles!) field of Markov chain Monte Carlo (MCMC) simulation techniques provides the powerful computational tools required for practical implementation of Bayesian HMM analysis. Chapter 4 introduces and discusses the issues raised when adopting MCMC samplers for inference purposes, with special reference to the two strategies most frequently employed: namely, the Gibbs sampler and the Hastings algorithms. Particular emphasis is placed on the need for designing efficient sampling strategies which are tailored to the problem of interest and a critical review of current suggestions in this area is provided.

Chapter 5 documents practical experience with the MCMC simulation approach in the case of Bayesian inference for unknown HMM parameters. It is shown that a Gibbs sampler strategy may be implemented to analyse the resulting posterior distribution in this HMM set-up. However, the sampling scheme advocated in the literature can exhibit certain inadequacies, in practice. This prompts a search for improvements in the Gibbs sampler design. Modifications to the original Gibbs sampling scheme are investigated by means of various performance measures, including application of appropriate Hastings

algorithms. The potential use of the resulting improved strategies for models other than the HMM are discussed.

Chapter 1

Introduction to hidden Markov models

Before introducing the general form of hidden Markov models (HMMs), we provide some background regarding their use. The original motivation behind this thesis was the application of HMMs in the area of automatic speech recognition (ASR) and in the next few sections we develop the ideas which have led to the adoption and successful implementation of HMMs in this field. However, all the ideas and approaches discussed here in relation to HMMs are generally applicable and other areas in which they are used will be discussed in greater detail throughout the thesis.

1.1 Automatic speech recognition

1.1.1 What is speech recognition?

The subject of ASR refers to the problem of how to train machines to interpret fluent speech signals and/or translate them into recognisable words and sentences which observe the proper grammatical constraints of the language. The potential for such voice input systems to increase efficiency, flexibility and security in many practical situations is already being commercially exploited, and further advances are continually being made in this field. Areas in which such systems are used to good effect include: inventory checking, where data is input verbally to the computer, thereby achieving

financial savings due to increased efficiency; military applications, e.g. the use of voice-activated instruments in the cockpit by pilots in combat; the "phonetic typewriter" for direct speech-to-text translation, thus removing the need for manual entry by keyboard, e.g. for composing business letters; voice-controlled equipment to assist the physically handicapped, e.g. voice-accessed telephone book, voice-activated domestic appliances. Bristow (1986) presents details of all the above applications, as well as a general introduction to the history of and current developments in commercial ASR.

The field of speech recognition is distinct from that of speech synthesis (see Chapters 5 and 6 of Holmes (1988)), which involves the simulation of the human voice through machines. The task of speech recognition involves the machine "listening" to a human voice and recognising the words and sentences which are uttered. The complexity of this task is governed by the particular application of the speech recogniser. For example, we can distinguish between the two categories of speaker-dependent recognition, where the machine is trained with utterances from one particular speaker in order to recognise speech uttered by that speaker alone, and speaker-independent recognition, where it is desired to train machines to recognise utterances from any potential speaker. For any set vocabulary of words, it is currently possible to achieve much higher degrees of recognition accuracy in speaker-dependent recognition than in speaker-independent recognition, as would be expected. Of course, both the size and content of the vocabulary itself (e.g. the number of similar and, thus, easily confused words) will necessarily have an impact on the complexity of the problem. Recognition can also be categorised into isolated word recognition, where sentences to be recognised are uttered with pauses inserted between adjacent words, the object being to recognise these individual words in relation to those around them, and continuous speech recognition, which requires the additional task of segmenting fluently spoken sentences into their component words. As one would expect, the former problem is more easily addressed than the latter.

Two further areas of speech recognition which are possible with today's technology are that of speaker identification, in which a particular speaker is identified from a list of potential candidates, and speaker verification, where a judgement is made as to whether a speaker is actually who (s)he says (s)he is. These two areas will not be discussed further in this thesis.

1.1.2 Knowledge-based versus data-based approaches

There are 2 distinct approaches currently being explored in the research area of ASR: knowledge-based and data-based.

Knowledge-based approaches attempt to exploit available knowledge of the structure of speech and the classification of vocal sounds. Speech signals are conceived of as consisting of a sequence of information-carrying elements called phonemes (vocal sounds) which are invariant and characteristic of particular parts of speech, this sequence being embedded in a mass of other irrelevant information. A typical classical recogniser of this type divides the speech signal into a sequence of time segments and extracts specific features from relevant information in the signal within each time segment (usually in the form of frequencies or transformations thereof). A segmentor then divides the signal into "phonemic segments", each individual phoneme in the sequence being classified according to its characteristic features. The resultant ordered sequence of phonemes is interpreted as a word, phrase etc, by consulting a dictionary of words, each represented as a specific string of phonemes. Zue (1985) demonstrates the use of speech knowledge in ASR and Allerhand (1987) attempts to combine ideas from both knowledge-based and data-based approaches.

This knowledge-based approach has, so far, proved inferior to data-based methods for many reasons; partly due to the implicit assumption that the whole essence of a sentence, say, may be characterised purely as a string of invariant vocal sounds, and partly due to the fact that there is still much that is unknown about the whole process of speech production and how it is perceived and understood by the human brain.

Data-based methods, however, perform recognition based mainly on evidence provided by sample speech signals and these have been used with success in many commercial speech recognisers (e.g. Jelinek (1985) describes an isolated-word recogniser intended for use in the dictation of office memos and Lee et al (1990) provide an overview of the SPHINX system, a speaker-independent continuous-speech recogniser trained to cope with large word vocabularies). The main difficulties facing any ASR approach can be summarised as follows:

Continuity In normal fluent speech, words flow into one another; pauses do not occur between every individual word and there are no obvious indications in the speech

signal itself to label the word boundaries.

Variability Speech signals exhibit various forms of variability which can be divided into two types: intra-speaker and inter-speaker. Inter-speaker variability is that which is attributable to differences between speakers, due to factors such as sex, age and local accent. However, restricting consideration to a single speaker does not eliminate all variability. Such intra-speaker variability comes in the form of differences in the volume or pitch at which words are spoken, or the health and mood of the speaker. Also, the context in which words are spoken governs which parts are accentuated and the length of time spent on pronouncing them. In fact, a single speaker pronouncing the same sentence on different occasions will never produce identical speech signals.

Context Basing recognition solely upon vocal sounds throws up problems in differentiating between words which sound similar, e.g. "to", "too" and "two". Thus, recognition must be performed in context — that of a sentence, say, under grammatical constraints — taking into account the broader meaning behind the communication (semantics) of which speech is only one part.

The development of adequate data-based recognisers therefore rests on our ability to overcome such problems.

1.1.3 Template-matching recognition techniques

Introduction

Template-matching approaches of the kind summarised in Figure 1.1 were predominant in the area of isolated whole-word recognition during the 1970's and early 1980's. Moore (1985) presents an overview of techniques used in this area. For the purposes of describing further the methods underlying speech recognisers of this type, we restrict consideration to isolated whole-word recognition, a task for which these template-matching techniques were originally designed.

In the training mode, a sample speech signal of each word in the target vocabulary is processed to obtain a useful representation of the information contained in the signal by extracting any relevant features. A segmentor then isolates that part of the signal

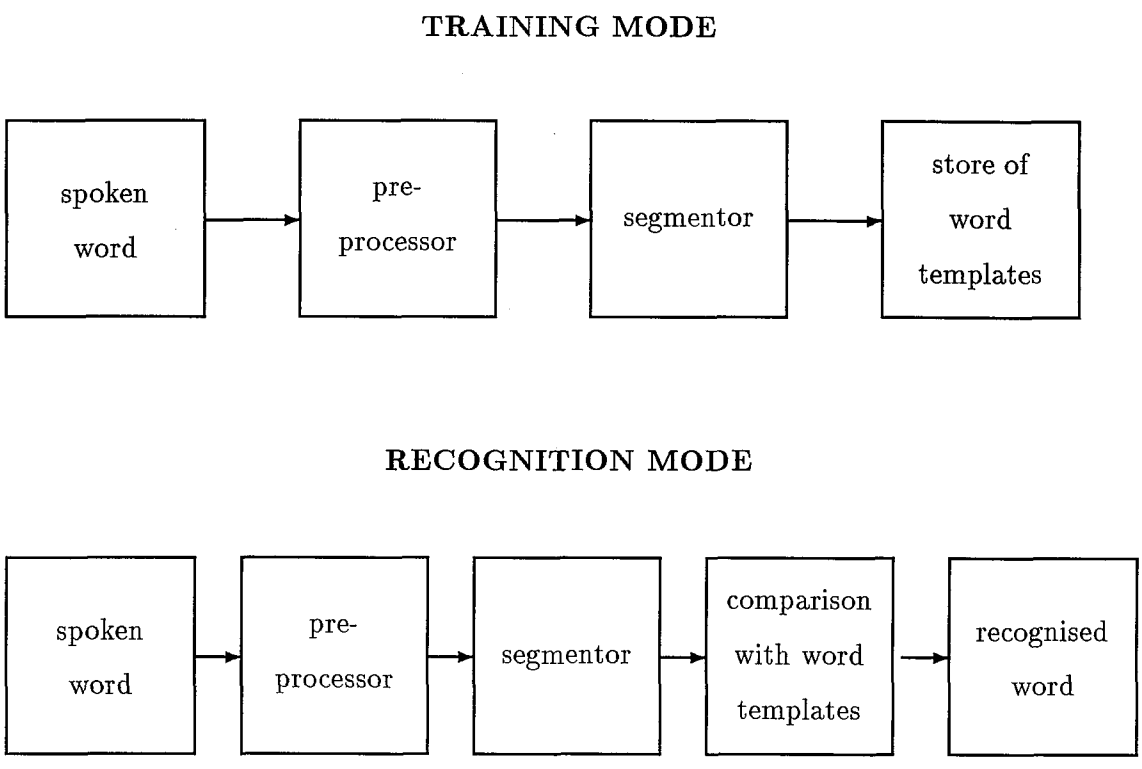


Figure 1.1: Format of a typical template-matching whole-word recogniser

attributable to the spoken word from the surrounding background noise. This resultant wave pattern, or template, then constitutes the reference pattern for the particular word being modelled. In the recognition mode, unclassified word signals are pre-processed in a similar manner and the resulting wave pattern compared with each template in the word vocabulary to find the best match.

Pre-processing and feature extraction

Any utterance to be analysed will normally be presented in the form of an acoustic, electronic signal received through a microphone. It is then the job of various signal processing techniques to condense this raw data by extracting relevant and characteristic information from the signal and representing it in some compact and useful form. In brief, the speech signal is segmented into short time intervals and the information from the signal within each of these intervals is condensed to form a vector at that timepoint. The entries in this vector are chosen to best represent the characteristic features of the signal, typically based on spectral properties. For an overview of those signal processing techniques most commonly employed in ASR, see Section 3.1 in Huang et al (1990).

Thus, ultimately, the speech signal is reduced to a sequence of vectors, ordered in time, which is referred to as the word pattern and denoted by $W = (W_1, W_2, \ldots, W_J)$, where each W_j is a vector in n-dimensional Euclidean space, say, and the length of the sequence of vectors is J. In the training mode, a word template consists simply of one or more such word patterns obtained from sample utterances of that particular word. These are then stored for comparison purposes in the task of classifying speech signals which represent unknown words.

Methods of pattern comparison

Once the templates for each word in the vocabulary have been obtained and stored, the main task of recognition has to be addressed. The quality of the recognition process is then wholly dependent on the method of comparing the signal from the unclassified word with each template in the vocabulary in order to find the best match. Consider, then, a word pattern of length I, denoted by $U = (U_1, U_2, \ldots, U_I)$, which represents the signal of an unknown target word we wish to identify. It is desired to define a procedure for comparing this with a particular template, or reference sequence, of length J which is denoted by $W = (W_1, W_2, \ldots, W_J)$. The most common way of defining "similarity" between two such patterns is via some form of distance measure; the smaller the distance, the closer the match. This distance will be denoted by $\Delta(U, W)$. Any suitable distance measure may be chosen, although the most common is the sum of Euclidean distances between some chosen component vectors of each sequence.

The simplest of such comparison techniques is absolute pattern matching. Due to the fact that patterns representing speech signals of the same word will vary in length (i.e. $I \neq J$), this direct comparison is achieved by aligning the beginnings of the two patterns, i.e. U_1 with W_1, and then calculating the distance between the parts of both sequences which overlap. So, if $d(\mathbf{u}, \mathbf{v})$ denotes some distance measure between vectors $\mathbf{u}$ and $\mathbf{v}$ and if $m = \min(I, J)$ denotes the length of the shortest pattern, then the distance between U and W is defined to be

$$\Delta(U, W) = \sum_{i=1}^{m} d(U_i, W_i).$$

A slight improvement to this method, called best absolute time-alignment, involves initially aligning the patterns at their beginnings, calculating the distance measure between the overlapping component vectors, and then repeatedly shifting the patterns relative

to one another by one vector and again calculating the appropriate distance measure, until the ends of each pattern are aligned. The distance between U and W is then taken to be the minimum of these $k - m + 1$ individual measures, where $k = \max(I, J)$ and $m = \min(I, J)$, so that

$$\Delta(U, W) = \begin{cases} \min\limits_{j=0,1,\ldots,k-m} \sum\limits_{i=1}^{m} d(U_i, W_{i+j}) & \text{, if } k = J, \\ \min\limits_{j=0,1,\ldots,k-m} \sum\limits_{i=1}^{m} d(U_{i+j}, W_i) & \text{, if } k = I. \end{cases}$$

Although these techniques are computationally very cheap and perform satisfactorily for simple tasks with small vocabularies, they do not remove the difficulties associated with temporal differences between signals representing the same word. Temporal or durational differences arise due to the varying lengths of time spent on parts of a particular word when it is pronounced on different occasions. A desirable property of a recognition process is its ability to highlight inter-word differences whilst ignoring the inherent variability arising from repeated pronunciation of the same word, one such type of variability being temporal variation. However, both pattern-matching procedures described above highlight such temporal differences as well as inter-word differences.

Time-normalization is one method for removing such irrelevant durational differences by aligning the two word patterns to make them the same length before evaluating the distance measure. This normalization is most commonly achieved by either stretching or compressing the template to the same length as the unclassified word pattern. One form of time-normalization is linear, where the template is stretched linearly to match the length of the observed word pattern. However, linear distortion does not adequately represent temporal variation in the pronunciation of words. In practice, different parts of words will be lengthened or shortened by varying amounts. A better approach allows non-linear time distortions and this technique of non-linear time-normalization is commonly referred to as dynamic time-warping (DTW).

1.1.4 Dynamic time-warping

The aim in DTW is to find the best non-linear distortion of the template; i.e. that distortion which minimizes the distance between U and W. At first glance, this would seem to involve a search over a very large space of all possible distortions. Fortunately, a solution exists using the mathematical technique of dynamic programming, which is a

computationally efficient recursive optimisation procedure, guaranteed to find the optimal distortion. An informative description of DTW in the context of speech recognition is given in Russell et al (1986) and summarised here.

Given the word patterns representing the template W and the unclassified word U, each component vector of which is of dimension n, a distance measure d may be defined on $\Re^n$. The distance matrix D between U and W is then defined to be the $I \times J$ matrix with $(i,j)^{th}$ element given by $d(U_i, W_j)$. Any path through the distance matrix, travelling left-to-right and top-to-bottom from $(1,1)$ to (I,J), defines a time registration path (TRP) between U and W, which corresponds to a distortion of the template to fit the length of the unclassified word pattern. The sum of distances between components of U and W along this TRP is then considered to be a measure of the dissimilarity exhibited between the unclassified word pattern and the distorted template. Now, the total number of admissible paths is usually reduced by imposing local constraints. These are chosen in order to preserve the underlying "shape" of the template and avoid over-distortion which does not reflect the main aim of the exercise, that of eliminating durational irrelevancies. For purposes of illustration here, we restrict consideration to the set of TRPs defined as follows.

Let the function $\phi : \{1,2,\ldots,I\} \rightarrow \{1,2,\ldots,J\}$ map each component of U to a component of W, so that a path through the distance matrix may be described by the sequence $\Phi = \{(1,\phi(1)), (2,\phi(2)), \ldots, (I,\phi(I))\}$, where $\phi(\cdot)$ satisfies the following common set of constraints:

(a) $\phi(1) = 1$,

(b) $\phi(I) = J$,

(c) for $i = 2,3,\ldots,I$, $\quad \phi(i) \in \{\phi(i-1),\ \phi(i-1)+1,\ \phi(i-1)+2\}$.

The first 2 constraints correspond to aligning the beginnings and ends of each word pattern, respectively. The third constraint restricts the local distortion of the template W:

$\phi(i) = \phi(i-1)$	represents local expansion of W
$\phi(i) = \phi(i-1)+1$	represents no local distortion of W
$\phi(i) = \phi(i-1)+2$	represents local contraction of W.

Figure 1.2 illustrates how a template is distorted along a particular TRP. Assuming $n = 1$, so that each word pattern consists of a sequence of scalars, the template and

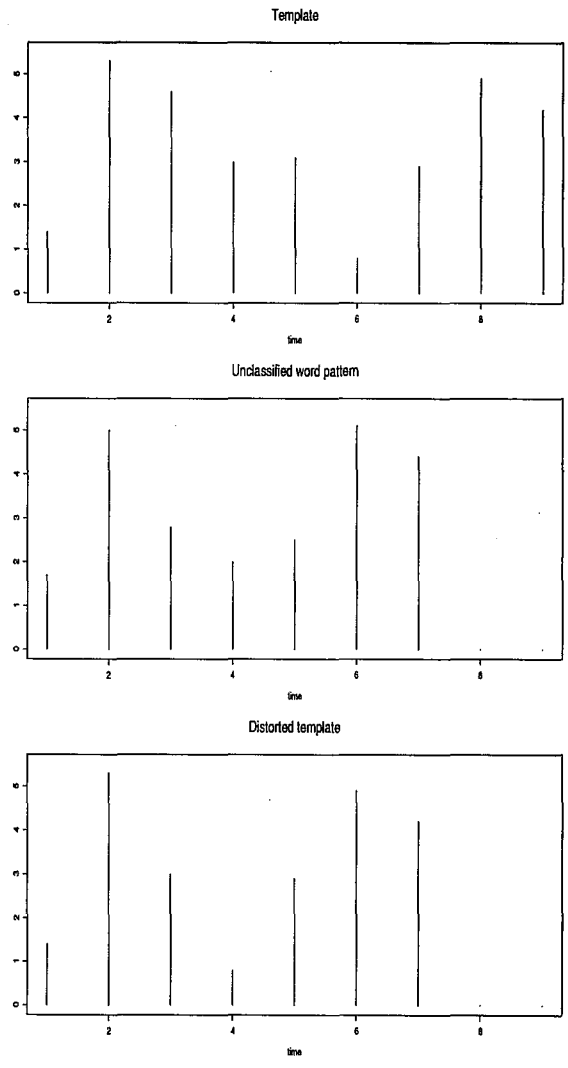


Figure 1.2: Example of a template distortion

unclassified word pattern in this example are

$$W = (1.4,\ 5.3,\ 4.6,\ 3.0,\ 3.1,\ 0.8,\ 2.9,\ 4.9,\ 4.2)$$
$$U = (1.7,\ 5.0,\ 2.8,\ 2.0,\ 2.5,\ 5.1,\ 4.4),$$

so that $I = 7$ and $J = 9$. The magnitude of each entry in the word pattern sequence is represented in Figure 1.2 by the length of its respective line. The distortion of the template W in this example corresponds to the TRP: {(1,1), (2,2), (3,4), (4,6), (5,7), (6,8), (7,9)}, so the distorted template is given by $W^* = (1.4,\ 5.3,\ 3.0,\ 0.8,\ 2.9,\ 4.9,\ 4.2)$, which is now very much closer in shape to the unclassified word pattern, U.

We may now define the accumulated distance between U and W along the path ϕ to be

$$A_\phi(U, W) = \sum_{i=1}^{I} d(U_i, W_{\phi(i)}).$$

The magnitude of this distance indicates the similarity of U to the template W, which has been distorted according to the path ϕ; the smaller the distance, the better the alignment of the two patterns. The aim is then to find the path ϕ which gives the best time-alignment of the patterns, i.e. the distortion of W which is "closest" to U. This optimal path has an associated accumulated distance called the cumulative distance between U and W, denoted by $C(U, W)$, where

$$C(U, W) = \min_{\phi} A_\phi(U, W).$$

This is, ultimately, the measure of comparison adopted in the template-matching procedure and the unknown word which gave rise to U is recognised by the class of the template in the full word vocabulary which minimises this cumulative distance.

The problem remains as to how to evaluate $C(U, W)$ since, if computed directly, this involves minimisation over a very large space of TRP's. However, subject to constraints (a)–(c) introduced earlier, the problem admits implementation of the principle of dynamic programming, which achieves global optimisation via repeated local optimisation through the use of recursive equations. Let $\gamma(i, j)$ denote the accumulated distance along the best partial path through the distance matrix from $(1, 1)$ to (i, j), i.e. the minimum value of this distance over all such admissible partial paths. Initially,

$$\gamma(1, 1) = d(U_1, W_1).$$

Then, for fixed $i = 2, 3, \ldots, I$, the quantities $\gamma(i, j)$, $j = 1, 2, \ldots, J$, may be evaluated according to the values $\gamma(i-1, j)$, $j = 1, 2, \ldots, J$, evaluated previously at the $(i-1)^{th}$

stage via the equation

$$\gamma(i,j) = \min_{x=0,1,2} \gamma(i-1, j-x) + d(U_i, W_j), \tag{1.1}$$

which is recursive in i. At the final stage, we obtain the desired quantity,

$$C(U,W) = \gamma(I,J).$$

The form of the recursive equation arises from the fact that the best partial paths up to row $i-1$ of the distance matrix are the only ones needed in order to construct those up to row i. In order to demonstrate the validity of (1.1) above, we introduce the notation $\boldsymbol{\phi}_{i,j} = (\phi_{i,j}(1),\ \phi_{i,j}(2),\ \ldots,\ \phi_{i,j}(i))$, with $\phi_{i,j}(i) = j$, to represent the best partial path from $(1,1)$ to (i,j) associated with $\gamma(i,j)$. Then

$$\begin{aligned}
\gamma(i,j) &= \min_{\boldsymbol{\phi}_{i,j}} \sum_{l=1}^{i} d(U_l, W_{\phi_{i,j}(l)}) \\
&= \min_{\boldsymbol{\phi}_{i,j}} \sum_{l=1}^{i-1} d(U_l, W_{\phi_{i,j}(l)}) + d(U_i, W_{\phi_{i,j}(i)}) \\
&= \min_{k=j-2,j-1,j} \sum_{l=1}^{i-1} d(U_l, W_{\phi_{i-1,k}(l)}) + d(U_i, W_j),
\end{aligned}$$

where the minimisation range for k is set to conform to the local constraint (c) on TRP's given above, which forces $\phi_{i,j}(i-1) \in \{(j-2), (j-1), j\}$. Hence,

$$\gamma(i,j) = \min_{x=0,1,2} \gamma(i-1, j-x) + d(U_i, W_j)$$

and, thus, the validity of (1.1) is proven.

This non-linear time-normalisation procedure was used extensively in speech recognisers before the advent of HMM-based recognisers. Alternative sets of TRP constraints may be employed to suit the classification problem of interest and there exist extensions of this basic algorithm which address the more complex problem of connected-word recognition. However, despite success for some recognition tasks, the technique of DTW has certain limitations. For instance, consider the two words "league" and "leek", the main discriminating factor between them being the length of the "ee" vowel sound. By its very nature, DTW treats such durational differences as irrelevant, removes them and may cause recognition errors. Moreover, it is possible that patterns from quite dissimilar words may be distorted on such a large scale as to produce a good match between them. Of course, the type and magnitude of admissible distortions is governed by the local constraints placed on TRP's and, thus, the accuracy of recognition is not only affected by the adequacy of the templates to characterise the underlying word, but also by the specification of the set of path constraints.

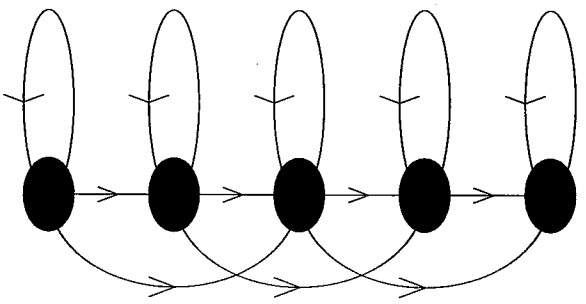


Figure 1.3: The DTW transition network

1.1.5 Locally constrained dynamic time-warping

In an effort to deal with these problems, an extension to the basic DTW algorithm was sought such that local constraints imposed on admissible paths might vary according to that part of the template currently being distorted. A visual aid to explaining the process of DTW exists in the form of a transition network, an example of which is given in Figure 1.3. Each node of this ordered network, labelled left-to-right from 1 through to J, represents a component vector of the template. A distortion of the template, for comparison with an unclassified word pattern, corresponds to a route through the network, from node 1 to node J, consisting of I transitions. Admissible transitions between nodes (indicated by arrows in Figure 1.3) correspond to the local constraints on admissible TRP's. The transitions indicated in Figure 1.4 correspond to the local constraints (a) – (c) of the basic algorithm discussed earlier. This network representation combines both the template itself with the constraints on admissible paths and, strictly, it is the combination of both which describes the word model. In terms of this transition network, an extension of basic DTW known as locally constrained dynamic time-warping can be described as follows.

Currently, the 3 local distortions admissible in the basic network of Figure 1.3 are given equal weight and remain constant throughout the sequence of nodes. It may, however, be desirable to give more weight to particular local distortions and to enable these weights to vary throughout the network. These template-dependent weights are chosen based on empirical evidence as to how sample patterns of the word under con-

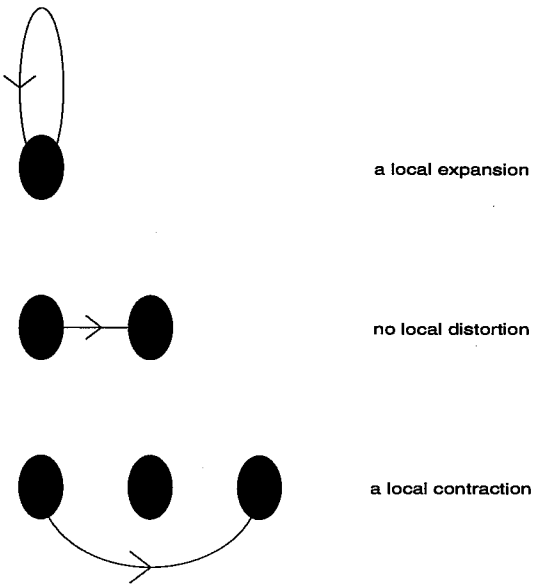


Figure 1.4: Transitions and the corresponding template distortion

sideration fluctuate with respect to the corresponding template. This is achieved by imposing node-dependent penalties on certain transitions to try and eliminate unnatural distortions of the template. If a transition is admissible between node i and node j, let $w_{i,j}$ denote the cost incurred by making this transition. The greater the likelihood of a particular transition, the lower the cost incurred. The expression for accumulated distance along a path ϕ now becomes

$$A_\phi(U,W) = d(U_1, W_{\phi(1)}) + \sum_{i=2}^{I} d(U_i, W_{\phi(i)}) + w_{\phi(i-1),\phi(i)}$$

and the cumulative distance can again be evaluated using dynamic programming via the recursive equation

$$\gamma(i,j) = \min_{x=0,1,2} [\gamma(i-1, j-x) + w_{j-x,j}] + d(U_i, W_j),$$

where

$$C(U,W) = \gamma(I,J).$$

As before, the path associated with this distance is the optimal TRP.

If we introduce the probability a_{ij} of a transition between node i and node j, one common choice for $w_{i,j}$ is $-c \log a_{ij}$, for some positive constant c. It then remains

to estimate the a_{ij}'s. As mentioned above, this is done empirically by employing the original DTW algorithm to find the optimal TRP between the template and each of a set of sample patterns representing the word being modelled. The probability a_{ij} is then simply estimated from the frequency of transitions made from node i to node j within these optimal paths.

Such locally constrained DTW can produce significant improvements over the original algorithm. However, it must be remembered that the word model is still simply a sample word pattern (often a set of templates are required to model adequately the variability present in speech signals of a single word) and that DTW only addresses the problem of temporal variation. Such an approach is not flexible enough to accommodate other important types of variability which exist. At the very least, however, the development of the technique of locally constrained DTW has provided ideas for the way forward. In particular, if one considers the transition network of nodes described above and in Figure 1.3 coupled with the matrix of transition probabilities, $A = (a_{ij})$, it is apparent that we are dealing with a first-order Markov chain with states represented by the nodes. This has prompted the idea of replacing the single pattern vector at each state with an entire probability distribution over $\Re^n$ (where n is the dimension of the component vectors) in order to allow greater modelling of variability. Hence, the genesis of hidden Markov models in ASR.

1.2 Hidden Markov models

Henceforth, we adopt the general convention that $p(\cdot)$ denotes a probability density function, when associated with a continuous variable, and a discrete probability mass function, when associated with a discrete variable. In either case, $p(\cdot|\cdot)$ denotes a conditional distribution.

1.2.1 General introduction

With the adoption of the stochastic hidden Markov model (HMM), the emphasis has now shifted from classical pattern recognition to that of statistical pattern recognition. Not only does the HMM, by virtue of its doubly stochastic nature, offer the chance to model the various sources of variability inherent to the speech process, but it also

establishes a more formal mathematical framework within which to model speech including, for example, extensions to continuous speech recognition and the incorporation of grammatical constraints. We now provide a general description of an HMM, including comments as to how the concepts introduced may be interpreted in the particular context of ASR.

Consider a set of N states, $\{1, 2, \ldots, N\}$, on which we impose a Markov chain structure. For a set of discrete timepoints, $t = 1, 2, \ldots, T$, we denote the state at time t by s_t. So, for an r^{th} order Markov chain, the Markov property is given by

$$p(s_t|s_{t-1}, s_{t-2}, \ldots, s_1) = p(s_t|s_{t-1}, s_{t-2}, \ldots, s_{t-r}), \tag{1.2}$$

i.e. given all previous states, the state at time t is dependent only on the preceding r states. In what follows, we shall restrict consideration to the first-order model (when $r = 1$) which is used most frequently in practice, although higher order chains are perfectly permissible. An observable stochastic process is now imposed upon this underlying state process, where the observation at time t, denoted by y_t, observed whilst the process occupies state s_t, satisfies a conditional independence assumption

$$p(\mathbf{y}|\mathbf{s}) = \prod_{t=1}^{T} p(y_t|s_t), \tag{1.3}$$

denoting the full observation sequence by $\mathbf{y} = (y_1, y_2, \ldots, y_T)$ and the corresponding unobserved state sequence by $\mathbf{s} = (s_1, s_2, \ldots, s_T)$. This assumption of state-dependent distributions could be relaxed to allow consideration of distributions of y_t dependent on the state-transition at time t (i.e. on both s_{t-1} and s_t). However, from now on we adopt the more common simplified assumption of Equation (1.3). The N-state HMM may be represented by the parameter set $H = (\pi, A, B)$, where

- $\pi = (\pi_i)$, $i = 1, 2, \ldots, N$, is the initial state distribution vector, with

$$\pi_i = p(s_1 = i), \quad \sum_i \pi_i = 1.$$

- $A = (a_{ij})$, $i, j = 1, 2, \ldots, N$, is the one-step state-transition matrix, with

$$a_{ij} = p(s_t = j|s_{t-1} = i), \quad \sum_j a_{ij} = 1, \ \forall i.$$

- $B = \{b_i(\cdot) : i = 1, 2, \ldots, N\}$ is the set of state-dependent observation distributions, with

$$b_i(y_t) = p(y_t | s_t = i), \quad \forall t,$$

and

$$\begin{cases} \sum_y b_i(y) = 1 & \text{, if } y \text{ is discrete} \\ \int_y b_i(y)\, dy = 1 & \text{, if } y \text{ is continuous.} \end{cases}$$

The above model generates an observation sequence, $\mathbf{y} = (y_1, y_2, \ldots, y_T)$, as follows:

1. Select an initial state, s_1, according to the initial state vector $\boldsymbol{\pi}$. Put t=1.
2. Generate an observation, y_t, according to the distribution $b_{s_t}(\cdot)$. If t=T, then STOP.
3. According to the probabilities $\{a_{s_t i} : i = 1, 2, \ldots, N\}$, make a transition from s_t to s_{t+1}=i. Put t=t+1 and go to Step 2.

The actual state sequence, $\mathbf{s} = (s_1\, s_2, \ldots\, s_T)$, which produces $\mathbf{y}$ is unobserved (or hidden) and cannot be recovered unambiguously, in general, as it is not unique.

Applications for HMMs are becoming increasingly widespread, including code deconvolution and digital communication (Forney, 1973), handwritten text recognition (Kundu et al (1989) and Nag et al (1986)), DNA sequence modelling (Churchill, 1989) and population analysis in biology (Dupuis, 1994), as well as the area of ASR. A very informative book on the subject of HMMs in the area of ASR is that of Huang et al (1990); shorter overviews have been written by Rabiner (1989) and Juang & Rabiner (1991).

In the context of ASR, HMMs are flexible enough to model statistically all parts of speech, from sub-word units such as phonemes (distinct vocal sounds) to entire phrases. For example, if we build an N-state model to represent a particular word, observation sequences in the HMM set-up correspond directly to word patterns in the template-matching approaches discussed in Section 1.1; observation sequences being elicited from the speech signal in an identical way to word patterns. The states of the HMM can be thought of as representing stationary regions of the word, e.g. distinct vocal sounds in the word corresponding to varying positions of the vocal tract. The transition matrix A then describes the structural and durational properties of the word in terms of its progression

through the stationary regions, this matrix playing an analogous role to that of the set of local constraints on TRPs in DTW. The set of observation distributions, $\{b_i(\cdot) : i = 1, 2, \ldots, N\}$, describes the variation within the stationary regions, corresponding to differences in pronunciation, pitch, accent, etc. These state-dependent distributions can be identified with sets of adjacent single vectors comprising the word pattern of the template-matching techniques, thus allowing more effective modelling of other types of variability besides temporal variation. The TRPs of DTW are replaced by state sequences, which indicate the path taken through the state network of the HMM and, as will be seen later, the similarity between unclassified observation sequences and word models is now measured according to a product of probabilities as opposed to a sum of distances. In fact, DTW may be considered as a special case of a HMM with Normal observation distributions (see Russell et al, 1986).

A special property of speech signals is the importance of the temporal order. To take account of this, consideration is often restricted to a subset of HMMs called left-right (L-R) models (analogous to the left-right constrained transition networks considered in the locally constrained DTW techniques of Section 1.1.5). These appear, at least in practice, to allow better modelling of the sequential nature of speech than do completely unconstrained, ergodic models (Rabiner et al, 1983). Ergodic models allow jumps from any one state to any other state with positive probability, whilst L-R models constrain the state process to proceed from left to right through the states $\{1, 2, \ldots, N\}$, never allowing a state to be revisited once it has been left, until the process reaches the final absorbing state N. The constraints of L-R models are represented by the upper-triangular structure of the transition matrix A (i.e. all entries below the diagonal are zero). Denoting by $\mathcal{S}$ the set of admissible state sequences under a particular HMM, the cardinality of $\mathcal{S}$ is $S = |\mathcal{S}|$, where

$$S = \begin{cases} \begin{pmatrix} N+T-1 \\ N-1 \end{pmatrix} & \text{, for L-R models} \\ N^T & \text{, for ergodic models,} \end{cases} \tag{1.4}$$

and

$$\begin{pmatrix} i \\ j \end{pmatrix} \text{ denotes the combinatorial quantity } \frac{i!}{j!\,(i-j)!}.$$

We see that the size of the space of admissible state sequences is radically reduced under the constraints of L-R modelling. However, depending on the application, other sets of constraints on the entries of the transition matrix may be considered more appropriate.

1.2.2 Problems to be addressed

Consider the case of a set of HMMs $\{H_1, H_2, \ldots, H_M\}$ which are chosen to model the corresponding set of classes $\{C_1, C_2, \ldots, C_M\}$ in a general pattern classification problem. There are 3 main problems, some or all of which we may wish to address, when implementing HMMs:

Training Given a particular class C_i, how may we best estimate its corresponding HMM H_i, based on training data in the form of a set of sample observation sequences obtained from this class? In the particular case of isolated word recognition, say, the HMM replaces the template in DTW as the word model and the observation sequences will take the form of sample signal patterns of the spoken word.

Recovery of state sequence information Given a known fixed HMM H_i and an observation sequence $\mathbf{y}$ which is known to have come from the class C_i being modelled, it is often informative to learn about the likely unobserved state sequence structure which gave rise to $\mathbf{y}$. In the context of ASR, this is a useful by-product of the HMM which aids understanding of the structure of words in terms of their stationary states. However, in other applications, recovery of underlying state sequence structure may be the main aim of the analysis (e.g. handwritten text recognition, which will be discussed in Chapter 3). In either case, it is desirable to find state sequences which best "explain" the observed sequence $\mathbf{y}$.

Classification Given that each class is modelled by a HMM in the set $\{H_i : i = 1, 2, \ldots, M\}$, classification of an observation sequence $\mathbf{y}$ from an unknown source corresponds to finding the HMM which offers the best match. Some "similarity" measure must, therefore, be specified to score each model in the set according to how well it matches $\mathbf{y}$, the model with the highest score being used for classification. In the context of ASR, this represents the actual process of speech recognition.

A summary of how the training and classification stages relate to ASR is illustrated in Figure 1.5. The speech units referred to in the figure are those elements in the structure of speech being modelled by the HMMs, e.g. phonemes, whole words.

In the next chapter, a review is given of the most common methods currently in use for addressing such problems. The remainder of this section expands on some of the

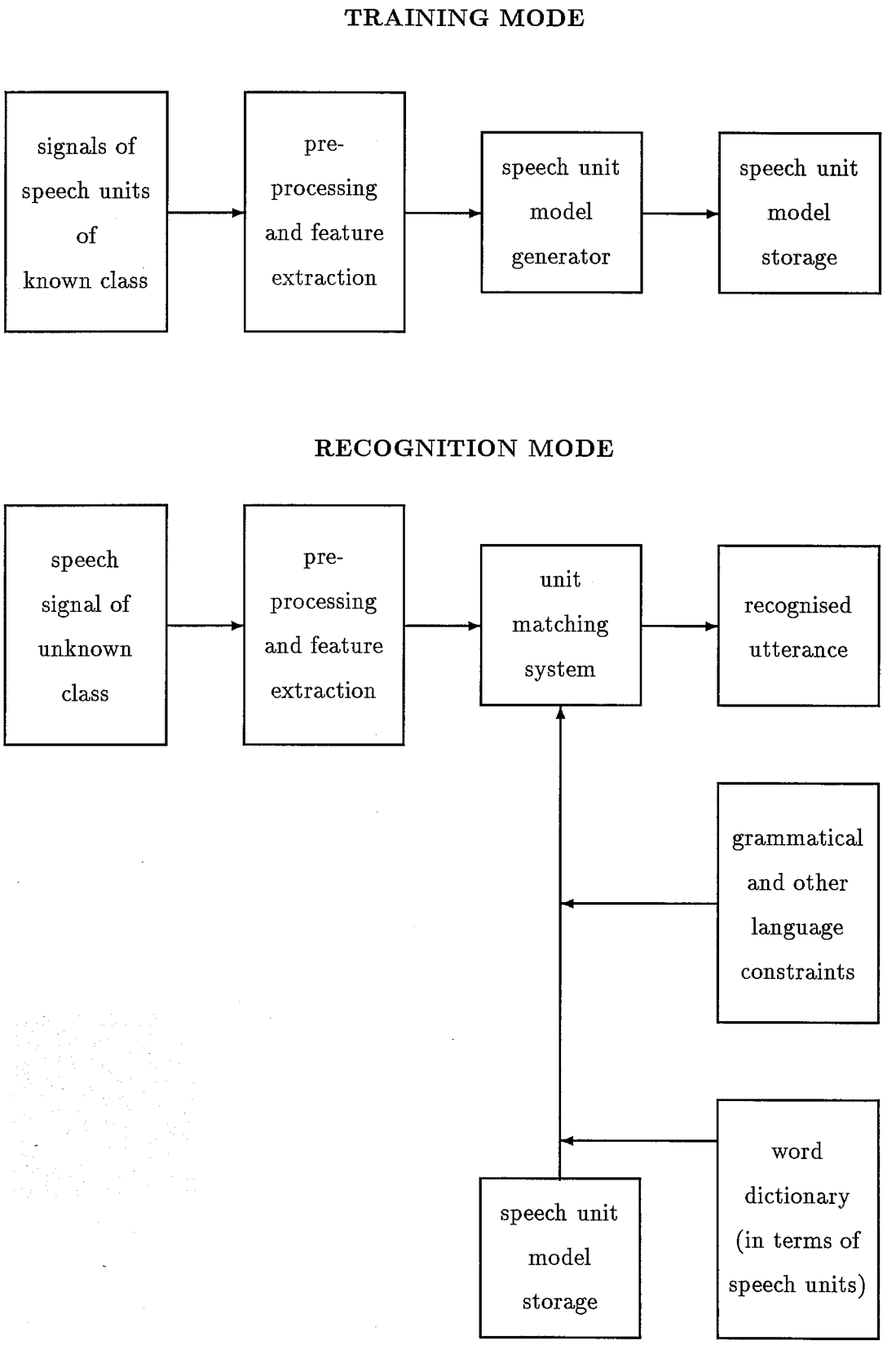


Figure 1.5: Format of a typical HMM speech recogniser

points already raised concerning the HMM and introduces some useful modifications of the basic model.

1.2.3 Choice of observation distribution

Each observation vector in the observation sequence is a realisation of some continuous distribution and consists of n real entries. It is usual to select standard parametric forms for the observation distributions, typically mixtures of multivariate Normal distributions whose properties are well-understood (e.g. Liporace (1982) and Rabiner et al (1985a)). However, in some applications, the observation distributions are assumed discrete over a finite set of symbols, $\{1, 2, \ldots, K\}$. Each of the K symbols corresponds to a distinct vector in $\Re^n$ and the procedure by which an optimal set of symbols (known as the symbol codebook) is chosen is known as vector quantisation (Rabiner et al (1983) and Chapter 4 in Huang et al (1990)). Each continuous vector in the observation sequence is then approximated by that symbol in the codebook to whose corresponding vector it is deemed "closest", by means of some similarity measure.

Both choices of continuous and discrete distribution have advantages and disadvantages. Adoption of vector quantisation and discrete distributions removes the assumption of a possibly unsuitable family of parametric models whilst introducing an extra degree of uncertainty due to the process of approximating continuous quantities with discrete ones. However, HMM training in this case involves the estimation of $N(K-1)$ quantities associated with the observation matrix B, which may grow very large in comparison with the set of parameters required to define a parametric family of distributions. The choice and adequacy of the form of observation distribution will therefore be determined by the particular application. For example, adoption of particular multivariate Normal mixtures as observation distributions in ASR may reduce recognition errors in comparison with vector quantisation (e.g. Rabiner et al (1985a), in the context of isolated digit recognition), whilst in handwritten text recognition (Kundu et al, 1989), a set of discrete observation symbols has been established which seem to characterise the underlying process well enough to perform successfully.

1.2.4 Time duration models

In the standard form of the HMM, introduced in Section 1.2.1, the inherent duration probability associated with state i is given by

$$p(\text{ duration of state } i = d\,) = a_{ii}^{d-1}(1 - a_{ii}), \quad d = 1, 2, \ldots$$

Unfortunately, modelling state durations with such distributions of a geometric nature often does not prove realistic. The basic HMM may be modified in order to explicitly model state durations (Guédon & Cocozza-Thivent (1990) and Levinson (1986)). This is achieved by setting the self-transition probabilities, $\{a_{ii} : i = 1, 2, \ldots, N\}$, of the transition matrix A to zero and replacing them by state duration densities, denoted by the set $\Delta = \{\delta_i(\cdot) : i = 1, 2, \ldots, N\}$, where

$$\delta_i(d) = p(\text{ duration of state } i = d\,), \quad d = 1, 2, \ldots, D,$$

where the duration of a state is usually assumed to be limited by a maximum duration, D. Thus, the parameter set is now denoted by $H = (\pi, A, B, \Delta)$. Under a modified notation, if we have a total of r state transitions, $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_r)$ represents the distinct state transitions made within the state sequences, these being governed by A, so that the actual state sequence $\mathbf{s}$ may be defined according to: $s_t = \sigma_1$, $t = 1, 2, \ldots, d_1$; $s_t = \sigma_2$, $t = d_1 + 1, d_1 + 2, \ldots, d_1 + d_2$; etc. We assume for simplicity that the distribution of d_i, given σ_i, is independent of d_j, $j \neq i$, and, as for the original HMM, we assume that y_t given s_t is independent of $y_{t'}$, $t' \neq t$. The mechanics of such a model are as follows. Initially, state σ_1 is selected according to $\boldsymbol{\pi}$. The duration of σ_1, denoted d_1, is generated according to the distribution $\delta_{\sigma_1}(\cdot)$. The observations $y_1, y_2, \ldots, y_{d_1}$ are then generated independently according to $b_{\sigma_1}(\cdot)$. A transition is then made from σ_1 to σ_2 according to row σ_1 of A, remembering that no transitions may be made to the current state, i.e. $\sigma_2 \neq \sigma_1$. The duration of state σ_2 is then generated from $\delta_{\sigma_2}(\cdot)$ and the process continues in this manner until an observation sequence of length T is obtained. Thus, letting $\mathbf{d} = (d_1, d_2, \ldots, d_r)$, the statistical model is given by

$$p(\boldsymbol{\sigma}, \mathbf{d}, \mathbf{y}|H) = \pi_{\sigma_1}\delta_{\sigma_1}(d_1)\prod_{t=1}^{d_1} b_{\sigma_1}(y_t)a_{\sigma_1\sigma_2}\delta_{\sigma_2}(d_2)\ldots a_{\sigma_{r-1}\sigma_r}\delta_{\sigma_r}(d_r)\prod_{t=d_1+\ldots+d_{r-1}+1}^{T} b_{\sigma_r}(y_t),$$

where $\sum_{i=1}^{r} d_i = T$. Specification of $(\boldsymbol{\sigma}, \mathbf{d})$ is equivalent to specification of $\mathbf{s}$, in the sense that one defines the other. Adopting this more sophisticated model may yield improved performance, but at the expense of increased complexity and computational burden due to the additional parameters.

1.2.5 Other modifications of hidden Markov models

We briefly mention here some other modifications and issues of interest when adopting a HMM approach.

The semi-continuous HMM combines ideas both from vector quantisation/discrete HMMs and continuous density HMMs. The discrete observation symbol codebook of vector quantisation (see Section 1.2.3) is replaced by a set of observation densities common to all states. In some comparison tests, improvement in performance over discrete or continuous HMMs alone has been reported, as well as reduction in the number of free parameters (see Chapter 7 in Huang et al (1990) and Huang (1992) for details).

Another possible model modification is to allow null transitions, i.e. transitions between states which produce no emission of an observation. Such transitions are useful in language modelling to allow for alternative pronunciation of words.

From a practical point of view, there are often problems associated with insufficient data for estimation purposes. These may be dealt with using techniques such as parameter tying, which equates certain parameters in the model, thus reducing the number of free parameters, e.g. if the observation distribution is known to be identical in 2 states. Another technique, called deleted interpolation, works by estimating parameters for the desired HMM along with a smaller model (one with fewer parameters allowing for more robust but less accurate estimates) and then weighting the resultant estimates according to some criterion which assesses how well trained each of the 2 models are.

The ideas of null transitions, parameter tying and deleted interpolation are all applied in Bahl et al (1983).

1.2.6 Neural networks

For completeness, we mention here an alternative pattern recognition/classification procedure currently being researched, employing neural network ideas which exploit the computational advantages of parallel processing.

Neural networks attempt to imitate what we know about the recognition processes of the brain. They are composed of layers of interconnected nodes (neural units) which can be in an active or inactive state according to the weights of their connections with

neighbouring nodes. Given a set of initial values, estimation involves repeatedly updating these weights, using information inputted from training data, until the difference between network outputs and the desired outputs under correct classification is minimised. Once the connection weights have been estimated, classification consists of parallel inputting of observed data which is then propagated through the network using parallel processing, eventually yielding outputs which provide a score that represents the quality of the match between the data and the network.

Such an alternative approach will not be discussed further in this thesis. A comprehensive guide to the appllcation of neural networks in classification problems may be found in Lippmann (1987). In particular, Woodland (1990) illustrates their use in a simple case of isolated word recognition.

Chapter 2

Standard analysis of hidden Markov models

2.1 Recovery of state sequence information

2.1.1 Introduction

Once the parameters of a particular HMM have been estimated (parameter estimation techniques will be discussed in Section 2.3) and the estimated model accepted as representing the process under study (e.g. in isolated word recognition, as representing a specific word), it is often of interest to understand the hidden state sequence structure underlying the observation sequences produced by it. In the example of isolated word recognition, if the states of the HMM word model are thought to correspond to the distinct vocal sounds comprising the word, then statistical study of the duration of these states and the transitions between them, for signals obtained from various speakers, can improve understanding not only of the state structure, but also of within-state behaviour. In some applications, the recovery of this state sequence information will be the main aim of the analysis. For example, in the recognition of handwritten text (Kundu et al, 1989), the observation sequence $\mathbf{y}$ consists of a sequence of vectors, each vector containing information characterising the physical features of a handwritten letter, the aim being to recognise the word which has been written. The state underlying each observation vector represents a letter of the alphabet and state sequences represent strings of letters. Thus, optimal state sequence recovery is used directly to recognise the

handwritten text in terms of proper words.

Before searching for an optimal estimate of the state sequence, $\mathbf{s}^* = (s_1^*, s_2^*, \ldots, s_T^*)$, underlying a given observation sequence, $\mathbf{y}$, we must first define what we mean by "optimal". Various locally optimal sequences might be considered useful, e.g. the sequence comprised of states which, for each individual $t = 1, 2, \ldots, T$, are the most likely at that timepoint, such that $\mathbf{s}^*$ is defined by

$$s_t^* = \arg \max_{i=1,2,\ldots,N} p(s_t = i \,|\, \mathbf{y}, H), \quad t = 1, 2, \ldots, T.$$

However, when the state s_t^* is evaluated as above, no account is taken of the values of nearby states, say s_{t-1}^* and s_{t+1}^*, or other more global behaviour and this can lead to invalid state sequences being proposed for $\mathbf{s}^*$ (e.g. in the case of L-R models, estimated sequences may not obey the L-R constraint). Therefore, a more global idea of optimisation is adopted, where $\mathbf{s}^*$ is chosen to be the single best sequence maximising the posterior probability, $p(\mathbf{s}|\mathbf{y}, H)$, i.e. the maximum a posteriori (MAP) sequence. The problem now becomes one of MAP decoding for which a solution exists by employing the recursive techniques of dynamic programming. This so-called Viterbi algorithm was developed in the electrical engineering literature for use in code deconvolution and other digital communication problems (Forney, 1973) from an original paper by Viterbi (1967).

2.1.2 The Viterbi algorithm

The Viterbi algorithm maximises the joint probability $p(\mathbf{s}, \mathbf{y}|H)$ with respect to $\mathbf{s}$, and the corresponding maximal state sequence is simply the MAP sequence $\mathbf{s}^*$, since

$$\begin{aligned} p(\mathbf{s}|\mathbf{y}, H) &= \frac{p(\mathbf{s}, \mathbf{y}|H)}{p(\mathbf{y}|H)} \\ \Longrightarrow \mathbf{s}^* &= \arg\max_{\mathbf{s}} p(\mathbf{s}|\mathbf{y}, H) \\ &= \arg\max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H), \end{aligned} \tag{2.1}$$

since the denominator of the right-hand side of Equation (2.1) is constant with respect to $\mathbf{s}$. We denote the maximal joint probability $p(\mathbf{s}^*, \mathbf{y}|H)$ by p^*. Since

$$p(\mathbf{s}, \mathbf{y}|H) = \pi_{s_1} b_{s_1}(y_1) a_{s_1 s_2} \ldots a_{s_{T-1} s_T} b_{s_T}(y_T) \tag{2.2}$$

and S is typically large (see Tables 2.1 and 2.2), naive searching for the MAP sequence, by evaluating $p(\mathbf{s}, \mathbf{y}|H)$ in (2.2) for all state sequences and identifying the maximum, is unfeasible in practice and there is a need for an efficient search algorithm.

	T	5	10	25	50	100
N						
5		126	1001	23,751	316,251	4.6×10^6
10		2002	92,378	5.2×10^7	1.3×10^{10}	4.3×10^{12}
20		42,504	2.0×10^7	1.4×10^{12}	4.6×10^{16}	4.9×10^{21}
30		278,256	6.4×10^8	1.7×10^{15}	3.3×10^{21}	6.0×10^{28}

Table 2.1: The value of S for various values of N and T under a L-R HMM

	T	5	10	25	50	100
N						
5		3125	9.8×10^6	3.0×10^{17}	8.9×10^{34}	7.9×10^{69}
10		100,000	1.0×10^{10}	1.0×10^{25}	1.0×10^{50}	1.0×10^{100}
20		3.2×10^6	1.0×10^{13}	3.4×10^{32}	1.1×10^{65}	1.3×10^{130}
30		2.4×10^7	5.9×10^{14}	8.5×10^{36}	7.2×10^{73}	5.1×10^{147}

Table 2.2: The value of S for various values of N and T under an ergodic HMM

Letting $\mathbf{s}_{[t]}$ and $\mathbf{y}_{[t]}$ denote the partial state sequence $(s_1, s_2, \ldots, s_t)$ and the partial observation sequence $(y_1, y_2, \ldots, y_t)$, respectively, we may define $p(\mathbf{s}_{[t-1]}, s_t = i; \mathbf{y}_{[t]})$ to be the probability of the partial state sequence $(\mathbf{s}_{[t-1]}, s_t = i) = (s_1, s_2, \ldots, s_{t-1}, i)$, up to time t and ending in state i, jointly with the partial observation sequence $\mathbf{y}_{[t]}$. To implement the Viterbi algorithm, we then define the probability $\phi_t(i)$, for $i = 1, 2, \ldots, N$ and $t = 2, 3, \ldots, T$, to be

$$\phi_t(i) = \max_{\mathbf{s}_{[t-1]}} p(\mathbf{s}_{[t-1]}, s_t = i; \mathbf{y}_{[t]}), \tag{2.3}$$

where implicit conditioning on the set of model parameters, H, is suppressed in the notation. Thus, $\phi_t(i)$ is the maximum value of the joint probability of the partial observation sequence $\mathbf{y}_{[t]}$, and the partial state sequence $\mathbf{s}_{[t]}$ up to time t which ends in state i. The particular maximal partial state sequence giving rise to $\phi_t(i)$ is denoted by $\mathbf{s}_t(i)$, so that

$$\phi_t(i) = p(\mathbf{s}_t(i), \mathbf{y}_{[t]}). \tag{2.4}$$

We provide here a full step-by-step proof of the Viterbi algorithm in terms of probabilities (which does not appear to have been documented elsewhere in the literature that we

have studied). Fixing t and i, we put t=t+1 in (2.3) to obtain, via use of the definition of conditional probability,

$$\begin{aligned}
\phi_{t+1}(i) &= \max_{\mathbf{s}_{[t]}} p(\mathbf{s}_{[t]}, s_{t+1} = i; \mathbf{y}_{[t+1]}) \\
&= \max_{\mathbf{s}_{[t]}} \left[p(s_{t+1} = i, y_{t+1} | \mathbf{s}_{[t]}, \mathbf{y}_{[t]}) \times p(\mathbf{s}_{[t]}, \mathbf{y}_{[t]}) \right] \\
&= \max_{\mathbf{s}_{[t]}} \left[p(y_{t+1} | s_{t+1} = i) \times p(s_{t+1} = i | s_t) \times p(\mathbf{s}_{[t]}, \mathbf{y}_{[t]}) \right],
\end{aligned}$$

where, by the Markov property of (1.2) and the conditional independence property of (1.3),

$$p(y_{t+1} | s_{t+1}, \mathbf{s}_{[t]}, \mathbf{y}_{[t]}) = p(y_{t+1} | s_{t+1})$$

and

$$p(s_{t+1} | \mathbf{s}_{[t]}, \mathbf{y}_{[t]}) = p(s_{t+1} | s_t).$$

It follows that $\phi_{t+1}(i)$ may be expressed as

$$\begin{aligned}
\phi_{t+1}(i) &= \max_{\mathbf{s}_{[t]}} \left[a_{s_t i} \, p(\mathbf{s}_{[t]}, \mathbf{y}_{[t]}) \right] \, b_i(y_{t+1}) \\
&= \max_{j=1,2,\ldots,N} \left[\max_{\mathbf{s}_{[t-1]}} \left\{ a_{ji} \, p(\mathbf{s}_{[t-1]}, s_t = j; \mathbf{y}_{[t]}) \right\} \right] \, b_i(y_{t+1}) \\
&= \max_{j=1,2,\ldots,N} \left[a_{ji} \, \phi_t(j) \right] \, b_i(y_{t+1}). \qquad (2.5)
\end{aligned}$$

The above recursive equation in terms of t is the basis for the Viterbi algorithm and leads to a corresponding recursive form for the maximal partial state sequences. If the quantity ψ is defined to be that value of j which maximises the term in square brackets in (2.5), i.e.

$$\psi = \arg \max_{j=1,2,\ldots,N} \left[a_{ji} \, \phi_t(j) \right],$$

then it can be seen that the corresponding maximal partial state sequence up to time t+1 can be expressed in terms of those up to time t as

$$\mathbf{s}_{t+1}(i) = (\mathbf{s}_t(\psi), \, i),$$

since, by (2.4), $\mathbf{s}_t(j)$ is the partial state sequence associated with $\phi_t(j)$.

So, at each timepoint $t = 1, 2, \ldots, T$ and for each state $i = 1, 2, \ldots, N$, we find the most probable partial state sequence up to t ending in state i and also its corresponding joint probability with the partial observation sequence $\mathbf{y}_{[t]}$. We can think of these maximal partial state sequences as survivors (using terminology from Forney (1973)), there being N of these at any timepoint t, each ending in a different state. These N survivors are the only sequences that need to be stored in order to construct the

Initialisation For $i = 1, 2, \ldots, N$,

$$\begin{aligned} \phi_1(i) &= \pi_i \, b_i(y_1) \\ \mathbf{s}_1(i) &= (i) \end{aligned}$$

Recursion For $t = 1, 2, \ldots, T-1$ and $i = 1, 2, \ldots, N$,

$$\begin{aligned} \phi_{t+1}(i) &= \max_{j=1,2,\ldots,N} \left[a_{ji} \, \phi_t(j)\right] \; b_i(y_{t+1}) \\ \mathbf{s}_{t+1}(i) &= (\mathbf{s}_t(\psi), i), \end{aligned}$$

$$\text{where} \quad \psi = \arg \max_{j=1,2,\ldots,N} \left[a_{ji} \, \phi_t(j)\right].$$

Termination

$$\begin{aligned} p^* &= \max_{j=1,2,\ldots,N} \phi_T(j) \\ \mathbf{s}^* &= \mathbf{s}_T(\psi), \end{aligned}$$

$$\text{where} \quad \psi = \arg \max_{j=1,2,\ldots,N} \phi_T(j)$$

Figure 2.1: The Viterbi algorithm

survivors at time $t+1$. It can then be seen that the final stage of the algorithm yields the desired maximal state sequence $\mathbf{s}^*$ and corresponding probability p^*, since

$$\begin{aligned} p^* &= \max_{\mathbf{s}} \; p(\mathbf{s}, \mathbf{y}|H) \\ &= \max_{j=1,2,\ldots,N} \max_{\mathbf{s}_{[T-1]}} \; p(\mathbf{s}_{[T-1]}, s_T = j; \mathbf{y}|H) \\ &= \max_{j=1,2,\ldots,N} \phi_T(j) \end{aligned}$$

and, thus,

$$\mathbf{s}^* = \mathbf{s}_T(\psi),$$

where

$$\psi = \arg \max_{j=1,2,\ldots,N} \phi_T(j).$$

The Viterbi algorithm in its entirety is set out in Figure 2.1. It can be seen that implementation of the Viterbi algorithm involves $N + N(N+1)(T-1)$ multiplications, of

	T	5	10	25	50	100
N						
5		125	250	625	1250	2500
		(31,250)	(2.0×10^8)	(1.5×10^{19})	(8.9×10^{36})	(1.6×10^{72})
10		500	1000	2500	5000	10,000
		(1.0×10^6)	(2.0×10^{11})	(5.0×10^{26})	(1.0×10^{52})	(2.0×10^{102})
20		2000	4000	10,000	20,000	40,000
		(3.2×10^7)	(2.0×10^{14})	(1.7×10^{34})	(1.1×10^{67})	(2.5×10^{132})
30		4500	9000	22,500	45,000	90,000
		(2.4×10^8)	(1.2×10^{16})	(4.2×10^{38})	(7.2×10^{75})	(1.0×10^{150})

Table 2.3: Comparison of the order of calculations required under the Viterbi algorithm (with that under direct evaluation in brackets)

the order of N^2T calculations whilst, from (2.2), direct evaluation of the MAP sequence involves $(2T-1)N^T$ multiplications, of the order of $2TN^T$ calculations. Table 2.3 indicates the computational savings the Viterbi algorithm brings for various values of N and T.

The algorithm can easily be modified when changes in the form of the model are made. For example, Huang et al (1990) provide extensions to semi-continuous models and time duration models (discussed in Section 1.2). In addition, when the observation distributions are state-transition dependent with

$$b_{ij}(y_t) = p(y_t | s_t = j, s_{t-1} = i),$$

then, following the same reasoning as was used to obtain (2.5), the recursive equation now becomes

$$\phi_{t+1}(i) = \max_{j=1,2,\ldots,N} [a_{ji}\, \phi_t(j)\, b_{ji}(y_{t+1})]$$

or, when the state process is an r^{th} order Markov chain with transition probabilities denoted by

$$a_{i_r i_{r-1} \ldots i_0} = p(s_t = i_0 \,|\, s_{t-1} = i_1, \ldots, s_{t-r} = i_r), \quad \forall i_0, i_1, \ldots, i_r = 1, 2, \ldots, N,$$

the recursive equation, for $t+1=r, r+1, \ldots, T$ and $i_0, i_1, \ldots, i_{r-1} \in \{1, 2, \ldots, N\}$, then

becomes

$$\phi_{t+1}(i_0, i_1, \ldots, i_{r-1}) = \max_{j=1,2,\ldots,N} \left[a_{j i_{r-1} \ldots i_0}\, \phi_{t-1}(i_1, i_2, \ldots, i_{r-1}, j)\right]\; b_{i_0}(y_{t+1}),$$

where $\phi_t(i_0, i_1, \ldots, i_{r-1})$ is now defined by

$$\phi_t(i_0, i_1, \ldots, i_{r-1}) = \max_{\mathbf{s}_{[t-r]}} p(\mathbf{s}_{[t-r]},\, s_{t-(r-1)} = i_{r-1},\, \ldots, s_t = i_0;\, \mathbf{y}_{[t]}).$$

Because the recursive equations involve products of small quantities less than 1, underflow occurs in practice even for moderate-sized problems. This can easily be overcome by re-scaling in terms of logarithms (Section 3.1 in Levinson et al (1983)), which also brings the advantage of operating with "additions" rather than more computationally expensive "multiplications". So, if we now define $\tilde{\phi}_t(i)$ to be

$$\tilde{\phi}_t(i) = \max_{\mathbf{s}_{[t-1]}} \left[\log p(\mathbf{s}_{[t-1]}, s_t = i; \mathbf{y}_{[t]})\right],$$

the re-scaled algorithm is given by

Initialisation $\tilde{\phi}_1(i) = \log \pi_i + \log b_i(y_1)$

Recursion $\tilde{\phi}_{t+1}(i) = \max_{j=1,2,\ldots,N} \left[\log a_{ji} + \tilde{\phi}_t(j)\right] + \log b_i(y_{t+1})$

Termination $\log p^* = \max_{j=1,2,\ldots,N} \tilde{\phi}_T(j)$

If $\tilde{\psi}$ is now defined to be the argument maximising the term inside square brackets in the above recursive equation, then the optimal state sequence, $\mathbf{s}^*$, is obtained as before, replacing ψ with $\tilde{\psi}$.

2.2 Classification/recognition

2.2.1 Introduction

We now consider the problem of recognition or classification with respect to a set of possible classes, e.g. in the case of isolated word recognition, we may wish to classify speech signals according to one of a vocabulary of words.

In the most general terms, consider the set $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$ of M classes, each of which is modelled by its respective HMM in the set $\mathcal{H} = \{H_1, H_2, \ldots, H_M\}$. The

recognition task consists of assigning incoming observation sequences from unknown sources to one of the M classes. Let $p(H_i)$ represent the probability of the class C_i occurring prior to observation of the data, e.g. in the example of text recognition, where each class represents a word in the vocabulary, the probability $p(H_i)$ represents how common the word of class C_i is in typical samples of text under consideration. Then a natural solution to the problem of classification is available using Bayes' theorem; i.e. if C^* is the classification of the unknown source, then

$$\begin{aligned} C^* = C_j \quad \text{if} \quad H_j &= \arg\max_{H \in \mathcal{H}} p(H|\mathbf{y}) \\ &= \arg\max_{H \in \mathcal{H}} p(\mathbf{y}|H)\, p(H), \quad \text{by Bayes' theorem.} \end{aligned}$$

This raises the question of how to compute the likelihood, $p(\mathbf{y}|H)$.

2.2.2 The Baum-Welch algorithm

Given a particular model H and observation sequence $\mathbf{y}$, the joint probability of $\mathbf{y}$ with the state sequence $\mathbf{s}$ may be expressed as

$$p(\mathbf{s}, \mathbf{y}|H) = \pi_{s_1} b_{s_1}(y_1) a_{s_1 s_2} b_{s_2}(y_2) \ldots a_{s_{T-1} s_T} b_{s_T}(y_T)$$

and, thus, by summing over all such state sequences, we obtain the marginal distribution for $\mathbf{y}$

$$p(\mathbf{y}|H) = \sum_{\mathbf{s} \in \mathcal{S}} \left\{ \pi_{s_1} \prod_{t=1}^{T-1} a_{s_t s_{t+1}} \prod_{t=1}^{T} b_{s_t}(y_t) \right\}. \tag{2.6}$$

However, the state sequence space, $\mathcal{S}$, is typically very large, making direct calculation of this sum computationally unfeasible, as in the case of evaluation of the MAP sequence.

Fortunately, dynamic programming techniques come to the rescue again in the form of the Baum-Welch (B-W) algorithm. This works in an analogous way to the Viterbi algorithm discussed in Section 2.1.2. The desired probabilities in both cases can be expressed in terms of the joint probability, $p(\mathbf{s}, \mathbf{y}|H)$:

$$p^* = \max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H), \quad p(\mathbf{y}|H) = \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H). \tag{2.7}$$

The B-W algorithm proceeds exactly as for the Viterbi algorithm, except that maximisation terms in the latter are replaced by summation terms in the former, as would seem reasonable in light of the definitions in Equation (2.7).

In order to implement the B-W algorithm, we introduce the quantity $\alpha_t(i)$, for $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, N$, defined by

$$\alpha_t(i) = p(s_t = i, \mathbf{y}_{[t]}|H), \tag{2.8}$$

the joint probability of the partial observation sequence up to time t, $\mathbf{y}_{[t]}$, with the state at time t being i. Fixing i and putting $t = t+1$ in Equation (2.8), we decompose $\alpha_{t+1}(i)$ into conditional probabilities to obtain

$$\begin{aligned}
\alpha_{t+1}(i) &= p(s_{t+1} = i, \mathbf{y}_{[t+1]}|H) \\
&= p(s_{t+1} = i, y_{t+1}, \mathbf{y}_{[t]}|H) \\
&= \sum_{j=1}^{N} p(s_t = j, s_{t+1} = i, y_{t+1}, \mathbf{y}_{[t]}|H) \\
&= \sum_{j=1}^{N} p(y_{t+1}|s_{t+1} = i, s_t = j, \mathbf{y}_{[t]}, H)\, p(s_{t+1} = i, s_t = j, \mathbf{y}_{[t]}|H) \\
&= \sum_{j=1}^{N} p(y_{t+1}|s_{t+1} = i, H)\, p(s_{t+1} = i|s_t = j, \mathbf{y}_{[t]}, H)\, p(s_t = j, \mathbf{y}_{[t]}|H), \\
&\qquad\qquad \text{by the conditional independence assumption of (1.3)}, \\
&= \sum_{j=1}^{N} b_i(y_{t+1})\, p(s_{t+1} = i|s_t = j, H)\, \alpha_t(j), \\
&\qquad\qquad \text{by Equations (1.2) and (1.3)}, \\
&= \sum_{j=1}^{N} [a_{ji}\, \alpha_t(j)]\; b_i(y_{t+1}).
\end{aligned}$$

This equation, recursive in t, is the basis of the B-W algorithm and exhibits a similar structure to the Viterbi algorithm, summations taking the place of maximisations (cf Equation (2.5)). The full algorithm is displayed in Figure 2.2. As with the Viterbi algorithm, the desired probability is obtained at the final stage.

Again, great computational savings are made by employing this dynamic programming algorithm. Implementation of the B-W algorithm requires $N(N+1)(T-1)+N$ multiplications and $N(N-1)(T-1)$ additions, so that the number of calculations is of the order N^2T. In contrast, direct evaluation of $p(\mathbf{y}|H)$ via (2.6) requires $(2T-1)N^T$ multiplications and $N^T - 1$ additions, so that the number of calculations is of the order $2TN^T$. Thus, the computational savings from using the B-W algorithm over direct evaluation are of the same order as those displayed in Table 2.3 for the Viterbi algorithm.

We have seen that the B-W algorithm makes great computational savings over direct "long-hand" evaluation of $p(\mathbf{y}|H)$. However, when it is required to compute

Initialisation For $i = 1, 2, \ldots, N$,

$$\alpha_1(i) = \pi_i \, b_i(y_1)$$

Recursion For $t = 1, 2, \ldots, T-1$ and $i = 1, 2, \ldots, N$,

$$\alpha_{t+1}(i) = \sum_{j=1}^{N} \left[a_{ji} \, \alpha_t(j) \right] \; b_i(y_{t+1})$$

Termination

$$p(\mathbf{y}|H) = \sum_{j=1}^{N} \alpha_T(j)$$

Figure 2.2: The Baum-Welch algorithm

$p(\mathbf{y}|H_i)$, $i = 1, 2, \ldots, M$, for large M (e.g. large-vocabulary ASR), M applications of the B-W algorithm may still be computationally burdensome. Thus, faster search techniques for identifying $H^* = \arg\max_{H_i} p(\mathbf{y}|H_i)$ are required; e.g. Bahl et al (1989) introduce an efficient method for specifying a shortlist of candidates from which H^* may be identified.

2.2.3 Viterbi scoring

The Viterbi scoring method is a non-optimal criterion which is sometimes used in classification, mainly due to its relative computational efficiency. A model H is scored according to the probability $p^* = \max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H)$ (calculated via the Viterbi algorithm) as opposed to the B-W scoring criterion of $p(\mathbf{y}|H)$. Thus, if we let p_i^* be defined by

$$p_i^* = \max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H_i),$$

the Viterbi scoring procedure classifies the unknown source to be C^*, where

$$C^* = C_j \quad \text{if} \quad j = \arg \max_{j=1,2,\ldots,M} p_j^*.$$

Under a particular model H, the likelihood used in the B-W scoring procedure can be expressed as the sum

$$p(\mathbf{y}|H) = \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H),$$

the largest term of which is p^*. So, if the maximal state sequence, $\mathbf{s}^*$, giving rise to p^* provides a dominant contribution to the summation, classifying according to this sequence alone should yield similar results to those obtained with the full sum. Equivalent recognition performance under both Viterbi and B-W scoring is reported in Rabiner et al (1983) in the context of isolated word recognition.

2.2.4 Connected-class recognition

In some applications, it may be that an optimal ordered string of classes is required for recognition. We confine ourselves here to a brief discussion of the case for ASR.

In connected-word (or continuous speech) recognition, the search space for finding an optimal string of words, say, (in terms of their corresponding HMMs) is greatly increased and will usually be accompanied by language/grammatical constraints. Section VII of Rabiner (1989) describes a level-building approach to connected-word recognition which employs the Viterbi algorithm to find optimal word strings. For large vocabularies, the Viterbi beam search may be employed (Picone, 1990). This is identical to the Viterbi algorithm except, at each timepoint t, only those survivors whose probabilities are above some specified lower limit are stored and extended at time $t+1$. This "pruning" reduces the computation time but the procedure is now non-optimal, no longer guaranteeing to find the MAP sequence. Heuristic search algorithms have also been used to good effect, e.g. Bahl et al (1983).

2.3 Parameter estimation

2.3.1 Maximum likelihood and the EM algorithm

Most of the current literature on HMM parameter estimation, certainly in the context of ASR, bases its estimation procedures on classical statistical ideas by adopting maximum likelihood (ML) methods. Consider a particular class C (e.g. a word, in ASR) for which we wish to construct a model, H, based on some training data in the form of a single observation sequence, $\mathbf{y}$. ML methods seek to find that H which maximises the likelihood, $l(H) = p(\mathbf{y}|H)$.

In the late 1960's and early 1970's, Baum produced a series of papers (Baum et al

(1970) and Baum (1972)) formulating a HMM parameter re-estimation procedure, consequently referred to as the Baum-Welch (B-W) re-estimation procedure, which guarantees to find a local maximum of $l(H)$ with respect to H. It transpires that this estimation procedure is, in fact, a special case of the EM algorithm, a general iterative local maximisation technique of especial use in finding ML estimates in the presence of incomplete data (Dempster et al, 1977). A brief description of the aims and implementation of the general EM algorithm is given here, before discussing the special case of the B-W procedure in depth.

The EM algorithm derives its name from its two component stages, the first being an expectation step (E-step) and the second a maximisation step (M-step). As mentioned above, the algorithm was formulated to deal specifically with statistical models where the observed data could be thought of as being incomplete. To formalise this, assume the observed data $\mathbf{y}$ to be a realization from the sample space $\mathcal{Y}$. The complete data $\mathbf{x}$, belonging to some sample space $\mathcal{X}$, is not observed directly, only indirectly through $\mathbf{y}$. There exists a many-to-one mapping, $\nu : \mathbf{x} \rightarrow \mathbf{y}$ from $\mathcal{X}$ to $\mathcal{Y}$, and the only available information about $\mathbf{x}$ is that $\mathbf{x} \in \mathcal{X}(\mathbf{y})$, where $\mathcal{X}(\mathbf{y}) = \{\mathbf{x} : \nu(\mathbf{x}) = \mathbf{y}\}$.

Suppose our model depends on a vector of parameters, $\boldsymbol{\theta}$, belonging to some parameter space, Θ, for which it is desired to find the ML estimate (MLE), where the observed data $\mathbf{y}$ is modelled by the pdf $g(\mathbf{y}|\boldsymbol{\theta})$ giving a likelihood function $l(\boldsymbol{\theta})$. There exists a corresponding model for the complete data $\mathbf{x}$, denoted $f(\mathbf{x}|\boldsymbol{\theta})$, where

$$g(\mathbf{y}|\boldsymbol{\theta}) = \int_{\mathcal{X}(\mathbf{y})} f(\mathbf{x}|\boldsymbol{\theta})\, d\mathbf{x}.$$

The ML approach seeks to find the value of $\boldsymbol{\theta}$ which maximises the likelihood $l(\boldsymbol{\theta})$ derived from $g(\mathbf{y}|\boldsymbol{\theta})$. This incomplete data likelihood will often be of an intractable form making direct ML methods impossible, whereas the form of the missing data is chosen to make the complete data specification, $f(\mathbf{x}|\boldsymbol{\theta})$, some "nice" density which can be dealt with simply. The EM algorithm, therefore, exploits this fact to find the MLE of $\boldsymbol{\theta}$ with respect to $g(\mathbf{y}|\boldsymbol{\theta})$, denoted $\hat{\boldsymbol{\theta}}$, via the complete data model, $f(\mathbf{x}|\boldsymbol{\theta})$.

Adopting the usual notation, $E(\cdot)$, to represent mathematical expectation, consider the function $Q(\cdot,\cdot)$ defined by

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}') = E_{\mathbf{x}} \left[\log f(\mathbf{x}|\boldsymbol{\theta}')|\mathbf{y}, \boldsymbol{\theta}\right],$$

where $E_{\mathbf{x}}\left[v(\mathbf{x})|\mathbf{y}, \boldsymbol{\theta}\right]$ denotes the expectation of some function $v(\mathbf{x})$ of $\mathbf{x}$ with respect to

the conditional density $h(\mathbf{x}|\mathbf{y},\boldsymbol{\theta})$ of $\mathbf{x}$ given $\mathbf{y}$ and $\boldsymbol{\theta}$, i.e.

$$E_{\mathbf{x}}[v(\mathbf{x})|\mathbf{y},\boldsymbol{\theta}] = \int v(\mathbf{x})\, h(\mathbf{x}|\mathbf{y},\boldsymbol{\theta})\, d\mathbf{x}.$$

The above definition of Q assumes that $f(\mathbf{x}|\boldsymbol{\theta}) > 0$ for almost all $\mathbf{x} \in \mathcal{X}(\mathbf{y})$ and $\forall \boldsymbol{\theta} \in \Theta$. The EM algorithm is an iterative procedure which, given a starting value, guarantees to find a local maximum of $l(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Thus, given the current value for $\boldsymbol{\theta}$ at the i^{th} iteration is $\boldsymbol{\theta}^{(i)}$, the $(i+1)^{th}$ iteration of the algorithm yields $\boldsymbol{\theta}^{(i+1)}$ as follows:

E-step Compute $Q(\boldsymbol{\theta}^{(i)}, \boldsymbol{\theta})$.

M-step $\boldsymbol{\theta}^{(i+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}^{(i)}, \boldsymbol{\theta})$.

The basic idea behind the algorithm is that we would like to find $\hat{\boldsymbol{\theta}}$ to maximise $f(\mathbf{x}|\boldsymbol{\theta})$ or, equivalently, to maximise $\log f(\mathbf{x}|\boldsymbol{\theta})$. However, since this cannot be achieved directly, because the form of the latter is not known explicitly, we instead maximise the expected value of $\log f(\mathbf{x}|\boldsymbol{\theta})$ with respect to $\mathbf{x}$, given $\mathbf{y}$ and the current estimate $\boldsymbol{\theta}^{(i)}$. Using this re-estimation procedure, it can be proved (Dempster et al, 1977) that

$$l(\boldsymbol{\theta}^{(i+1)}) \geq l(\boldsymbol{\theta}^{(i)}).$$

It is worth mentioning here that that the EM algorithm can easily be modified, in a Bayesian setting, to find the mode of the posterior distribution for $\boldsymbol{\theta}$ instead of the mode of the likelihood.

2.3.2 The Baum-Welch re-estimation procedure

We now study, in greater detail, the special form of the EM algorithm obtained for the HMM case; namely, the B-W re-estimation procedure. The form of the re-estimation equations will be derived in the special case of finite, discrete observation distributions. The extension of the proofs to more general and continuous distributions follows similar lines and is discussed later in this section. For now, consider the case where, for each $i = 1, 2, \ldots, N$, the observation distribution associated with state i, $b_i(\cdot)$, is defined over the finite set of observation symbols, $\{1, 2, \ldots, K\}$. So, we can define the $N \times K$ observation matrix $B = (b_{ij})$ where, for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, K$,

$$b_{ij} = p(y_t = j | s_t = i) = b_i(j), \quad \forall t.$$

In order to describe the re-estimation procedure, we must recall the B-W algorithm of Section 2.2.2 and the forward probabilities defined therein. To recap, for $t = 1, 2, \ldots, T$, $i = 1, 2, \ldots, N$, we defined $\alpha_t(i)$ by

$$\alpha_t(i) = p(\mathbf{y}_{[t]}, s_t = i|H),$$

where the $\alpha_t(i)$'s satisfy the following recursive relation in t:

$$\alpha_{t+1}(i) = \left[\sum_{j=1}^{N} \alpha_t(j)\, a_{ji}\right] b_i(y_{t+1}), \quad t = 1, 2, \ldots, T-1,$$

initialised by

$$\alpha_1(i) = \pi_i\, b_i(y_1), \quad i = 1, 2, \ldots, N.$$

A similar set of backward probabilities is now defined. Denoting the partial observation sequence $(y_t, y_{t+1}, \ldots, y_T)$ by $\mathbf{y}^{[t]}$, we can define the quantity $\beta_t(i)$, for $i = 1, 2, \ldots, N$, $t = 1, 2, \ldots, T-1$, by

$$\beta_t(i) = p(\mathbf{y}^{[t+1]}|s_t = i, H).$$

Initially, setting

$$\beta_T(i) = 1, \quad i = 1, 2, \ldots, N,$$

the backward recursion for $i = 1, 2, \ldots, N$ and $t = T-1, T-2, \ldots, 1$ is given by

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(y_{t+1})\, \beta_{t+1}(j), \tag{2.9}$$

from which the backward probabilities may be computed in an efficient manner. Equation (2.9) is obtained using the following argument, along similar lines to that for the forward probabilities:

$$\begin{aligned}
\beta_t(i) &= p(\mathbf{y}^{[t+1]}|s_t = i, H) \\
&= \sum_{j=1}^{N} p(\mathbf{y}^{[t+1]}, s_{t+1} = j|s_t = i, H) \\
&= \sum_{j=1}^{N} p(\mathbf{y}^{[t+1]}|s_{t+1} = j, s_t = i, H)\, p(s_{t+1} = j|s_t = i, H), \\
&\qquad \text{by the definition of conditional probability,} \\
&= \sum_{j=1}^{N} p(y_{t+1}|\mathbf{y}^{[t+2]}, s_{t+1} = j, s_t = i, H)\, p(\mathbf{y}^{[t+2]}|s_{t+1} = j, s_t = i, H)\, a_{ij}, \\
&\qquad \text{by the definition of conditional probability,} \\
&= \sum_{j=1}^{N} b_j(y_{t+1})\, \beta_{t+1}(j)\, a_{ij},
\end{aligned}$$

since, by the conditional independence assumption of Equation (1.3),

$$p(y_{t+1}|\mathbf{y}^{[t+2]}, s_{t+1} = j, s_t = i, H) = p(y_{t+1}|s_{t+1} = j, H),$$

and

$$p(\mathbf{y}^{[t+2]}|s_{t+1} = j, s_t = i, H) = p(\mathbf{y}^{[t+2]}|s_{t+1} = j, H).$$

The re-estimation procedure yields new parameter estimates, $H' = (\boldsymbol{\pi}', A', B')$, from the current values, $H = (\boldsymbol{\pi}, A, B)$, via the following equations:

$$\begin{aligned}
\pi'_i &= \frac{\alpha_1(i)\beta_1(i)}{\sum_{j=1}^{N}\alpha_T(i)}, \quad i = 1, 2, \ldots, N, \\
a'_{ij} &= \frac{\sum_{t=1}^{T-1}\alpha_t(i)a_{ij}b_j(y_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1}\alpha_t(i)\beta_t(i)}, \quad i = 1, 2, \ldots, N,\ j = 1, 2, \ldots, N \\
b'_{ij} &= \frac{\sum_{t\in\{t:y_t=j\}}\alpha_t(i)\beta_t(i)}{\sum_{t=1}^{T}\alpha_t(i)\beta_t(i)}, \quad i = 1, 2, \ldots, N,\ j = 1, 2, \ldots, K,
\end{aligned}$$

where each $\alpha_t(i)$ and $\beta_t(i)$ is evaluated based on the current parameter set H. Repeated use of the above re-estimation equations yields a local maximum, since $l(H') \geq l(H)$. A detailed proof is provided in Appendix A.1.

In terms of the general formulation of the EM algorithm, estimation of the parameters of the HMM may be interpreted as an incomplete data problem. In this context, the state sequence information can be thought of as missing data, so that $\mathbf{x} = (\mathbf{y}, \mathbf{s})$ is the complete data set and $\boldsymbol{\theta} = H$ is the parameter set. We assume $\mathbf{s} \in \mathcal{S}$, where $\mathcal{S}$ is the set of all state sequences admissible given $\mathbf{y}$, with the cardinality of $\mathcal{S}$ given by $|\mathcal{S}| = S$, S being defined as in (1.4). Therefore, in the notation of the EM algorithm of Section 2.3.1, the complete data model is specified by

$$\begin{aligned}
f(\mathbf{x}|\boldsymbol{\theta}) &= p(\mathbf{y}, \mathbf{s}|H) \\
&= \pi_{s_1} b_{s_1}(y_1) \prod_{t=2}^{T} a_{s_{t-1}s_t} b_{s_t}(y_t),
\end{aligned}$$

which, in turn, is used to specify the incomplete data model as

$$\begin{aligned}
g(\mathbf{y}|\boldsymbol{\theta}) &= p(\mathbf{y}|H) \\
&= \sum_{\mathbf{s}\in\mathcal{S}} p(\mathbf{y}, \mathbf{s}|H) \\
&= \sum_{\mathbf{s}\in\mathcal{S}} f(\mathbf{x}|\boldsymbol{\theta}).
\end{aligned}$$

The above specification of $g(\mathbf{y}|\boldsymbol{\theta})$ is in the form of a sum over a, typically large, state sequence space $\mathcal{S}$ and thus cannot easily be maximised directly. The much simpler form

of $f(\mathbf{x}|\boldsymbol{\theta})$ facilitates the use of the EM algorithm. In the proof of the B-W re-estimation procedure (see Appendix A.1), the function $Q(\cdot,\cdot)$, given by

$$Q(H, H') = E_{(\mathbf{s},\mathbf{y})}\left[\log p(\mathbf{s}, \mathbf{y}|H')|\mathbf{y}, H\right]$$

(with reference to Equation (A.9) of Appendix A.1), is identical to the function $Q(\cdot,\cdot)$ defined for the general EM algorithm in Section 2.3.1, replacing $\boldsymbol{\theta}$ by H and $\mathbf{x}$ by $(\mathbf{s},\mathbf{y})$. Thus, the B-W re-estimation procedure, which finds H' to maximise $Q(H, H')$, is equivalent to the EM algorithm.

In order to obtain estimates of the model parameters, the B-W re-estimation procedure is iterated until such time as the likelihood $l(H)$ (evaluated using the B-W algorithm) with respect to the current set of parameter estimates, H, makes only "very small" increases between adjacent iterations. There is no formal procedure by which to identify the point at which the local maximum is judged to have been attained but it is usual to stop the process when the likelihood remains constant rounded to a user-specified number of significant figures. It should be noted that no measures of error for the resulting point estimates are calculable using this ML procedure.

As for the Viterbi algorithm, the quantities involved in the forward-backward equations of the B-W re-estimation procedure are very small and underflow will occur, in practice, when using a computer. A scaling procedure exists to prevent this, although it is more complicated than that for the Viterbi algorithm. The scaling procedure is introduced in Section 3.1 of Levinson et al (1983) and briefly reviewed in Rabiner (1989) but, in both cases, a full explanation is not given (for example, the main result, equivalent to Lemma A.4 in Appendix A.2, is not proved). The notation in the latter paper is particularly misleading, the $\alpha_t(i)$'s appearing simultaneously to denote 2 distinct quantities: both the original forward probabilities plus their scaled versions. In addition, Equations (92a), (92b) and (93b) of Rabiner's account are incorrect, since the "$a_{ij}\, b_j(O_t)$" terms therein should in fact read "$a_{ji}\, b_i(O_t)$". For a detailed derivation of the scaling procedure, see Appendix A.2.

For some HMM applications, it may be possible to estimate an ergodic model on the basis of a single observation sequence, long enough to provide sufficient data for estimation of all model parameters. However, if observation sequences are relatively short in nature (as is the case in ASR), an independent sample of such sequences will typically be required for estimation purposes. This is certainly the case when L-R models

are adopted. Consider the most likely state sequences underlying a given observation sequence generated from some L-R model. Once the process has left any particular state, it will never revisit that particular state again and, therefore, no new information is gained about the parameters associated with that state for the remainder of the sequence, no matter how long it is.

In this case, the B-W procedure is easily modified to take account of multiple observation sequences. For a set of R independent observation sequences, $\{\mathbf{y}_r : r = 1, 2, \ldots, R\}$, of lengths $T_1, T_2, \ldots, T_R$, respectively, with $\mathbf{y}_r = (y_{r,1}, y_{r,2}, \ldots, y_{r,T_r})$, $r = 1, 2, \ldots, R$, the modified B-W re-estimation procedure now seeks to maximise

$$p(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_R | H) = \prod_{r=1}^{R} p(\mathbf{y}_r | H).$$

With this objective, the re-estimation procedure obtains a new model H' from the current estimate H according to:

$$\pi_i' = \sum_{r=1}^{R} \frac{1}{p(\mathbf{y}_r|H)} \alpha_1^r(i)\beta_1^r(i), \quad i = 1, 2, \ldots, N$$

$$a_{ij}' = \frac{\sum_{r=1}^{R} \frac{1}{p(\mathbf{y}_r|H)} \sum_{t=2}^{T_r} \alpha_{t-1}^r(i) a_{ij} b_j(y_{r,t}) \beta_t^r(j)}{\sum_{r=1}^{R} \frac{1}{p(\mathbf{y}_r|H)} \sum_{t=1}^{T_r-1} \alpha_t^r(i)\beta_t^r(i)}, \quad i = 1, 2, \ldots, N, \; j = 1, 2, \ldots, N,$$

and, for finite discrete observation distributions,

$$b_{ij}' = \frac{\sum_{r=1}^{R} \frac{1}{p(\mathbf{y}_r|H)} \sum_{t \in \{t : y_{r,t} = j\}} \alpha_t^r(i)\beta_t^r(i)}{\sum_{r=1}^{R} \frac{1}{p(\mathbf{y}_r|H)} \sum_{t=1}^{T_r} \alpha_t^r(i)\beta_t^r(i)}, \quad i = 1, 2, \ldots, N, \; j = 1, 2, \ldots, K,$$

where $p(\mathbf{y}_r|H)$ is computed via the B-W algorithm, i.e.

$$p(\mathbf{y}_r|H) = \sum_{j=1}^{N} \alpha_T^r(j),$$

and $\alpha_t^r(i)$, $\beta_t^r(i)$ denote the usual forward and backward probabilities, respectively, defined with respect to the r^{th} observation sequence, $\mathbf{y}_r$, under the current model estimate H. The scaling procedure mentioned earlier may easily be modified to take account of multiple observation sequences (see Chapter 5 of Huang et al (1990)).

Until now, observation distributions have been assumed finite and discrete. In the more general case, where the observation distribution may have a parametric form, the re-estimation equations for π and A remain the same (as can be seen by inspection of the proof in Appendix A.1) and the re-estimation of B alone is affected, where B is now defined in terms of the unknown observation parameters. The most common form of

continuous observation distribution adopted is a finite mixture of multivariate Normal distributions. Re-estimation equations have been formulated in this instance for the B-W procedure, as well as for the case when the observation sequence is modelled by an autoregressive process. Theoretically, any form of observation distribution may be dealt with in this way, although it may not always be possible to obtain an explicit form for the re-estimation equations, and the maximisation step may have to be achieved via analytical approximations. Furthermore, the re-estimation procedure is easily adapted to take account of other modifications in the model, e.g. time duration models. See Rabiner (1989) for details of all these extensions.

2.3.3 Viterbi re-estimation and alternative estimation procedures

Although the B-W re-estimation procedure estimates the HMM parameters according to the standard classical optimality criterion of maximising the likelihood, in practice this method may be computationally demanding with the need to calculate forward and backward probabilities at every re-estimation step and further complexity introduced by the need for scaling. The more computationally efficient Viterbi re-estimation procedure, so called because of its exploitation of the Viterbi algorithm, has a similar structure to the B-W procedure, seeking instead to maximise the joint probability, $l^*(H) = \max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H)$, with respect to H. Again, the Viterbi procedure is a local maximisation technique which, given some initial model, guarantees to increase $l^*(H)$ at each iteration, until a local maximum is attained.

Given a data set of R independent observation sequences, $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_R)$, of lengths $T_1, T_2, \ldots, T_R$ respectively, the Viterbi algorithm is employed, based on the current model H, to recover the MAP state sequence underlying each one of them, denoted $\mathbf{s}_1^*, \mathbf{s}_2^*, \ldots, \mathbf{s}_R^*$ respectively, and their corresponding joint probabilities, $p_1^*, p_2^*, \ldots, p_R^*$, where

$$p_r^* = \max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}_r|H) = p(\mathbf{s}_r^*, \mathbf{y}_r|H), \quad r = 1, 2, \ldots, R.$$

The model parameters are then re-estimated using state transition frequencies or observation symbol frequencies (depending on which parameters are being estimated) based on the R optimal state sequences, $\mathbf{s}_1^*, \mathbf{s}_2^*, \ldots, \mathbf{s}_R^*$. If we let $\mathbf{s}_r^* = (s_{r,1}^*, s_{r,2}^*, \ldots, s_{r,T_r}^*)$ and similarly define $\mathbf{y}_r$, $r = 1, 2, \ldots, R$, then the Viterbi re-estimation equations for obtaining a new model $H' = (\boldsymbol{\pi}', A', B')$ from the current model $H = (\boldsymbol{\pi}, A, B)$ may be

expressed as

$$\pi'_i = \frac{\sum_{r=1}^{R} I_{\{s^*_{r,1}=i\}}}{R}, \quad i = 1, 2, \ldots, N$$

$$a'_{ij} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r-1} I_{\{s^*_{r,t}=i, s^*_{r,t+1}=j\}}}{\sum_{r=1}^{R} \sum_{t=1}^{T_r-1} I_{\{s^*_{r,t}=i\}}}, \quad i = 1, 2, \ldots, N, \; j = 1, 2, \ldots, N$$

and, for finite discrete observation distributions,

$$b'_{ij} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} I_{\{s^*_{r,t}=i, y_{r,t}=j\}}}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} I_{\{s^*_{r,t}=i\}}}, \quad i = 1, 2, \ldots, N, \; j = 1, 2, \ldots, K,$$

where $I_{\{E\}}$ is the indicator function for the event E, such that

$$I_{\{E\}} = \begin{cases} 1, & \text{if } E \text{ occurs} \\ 0, & \text{otherwise.} \end{cases}$$

This procedure guarantees, at every iteration, to increase the joint probability

$$l^*(H) = \prod_{r=1}^{R} p^*_r,$$

until some local maximum is attained. The proof is analogous to that for B-W re-estimation, where instead of maximising $\sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H)$ with respect to H, we maximise $\max_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H)$, the single most dominant term in the aforementioned sum.

Merhav & Ephraim (1991) compare model estimation using the Viterbi and B-W re-estimation procedures and conclude that they often yield similar results. In these cases, use of the Viterbi procedure is preferred due to lower computational costs.

Other estimation criteria which have been considered in order to improve speech recognition performance are based on simultaneously estimating the models for the total set of M classes under consideration in the classification problem; e.g. corrective training finds those parameter estimates which minimise recognition errors for the training data (Lee & Mahajan, 1990). Other optimisation criteria are reviewed in Rabiner (1989), such as maximum mutual information which seeks to maximise the discrimination power of each model based on the training data.

Chapter 3

Generalisation of the Viterbi algorithm

3.1 Why is generalisation useful?

In this chapter, the Viterbi algorithm that was introduced in Section 2.1.2 is revisited and a significant generalisation established whereby the top L most probable state sequences underlying an observation sequence are recovered along with their associated maximal probabilities. Much of the work presented here can be found in Foreman (1993). However, we must first answer the question of why such a generalisation is necessary.

The Viterbi algorithm provides us with the MAP state sequence, $\mathbf{s}^*$, and its corresponding joint probability with the observation sequence, $p^* = p(\mathbf{s}^*, \mathbf{y}|H)$. Depending on the application, this information is used either to learn generally about the state structure of the process being modelled (e.g. ASR) or to specifically classify or recognise the observation sequence in some way (e.g. text recognition approach adopted in Kundu et al (1989)). In a Bayesian approach, all analysis is based on posterior distributions. For the MAP state sequence, $\mathbf{s}^*$,

$$p(\mathbf{s}^*|\mathbf{y}, H) = \frac{p^*}{p(\mathbf{y}|H)}, \tag{3.1}$$

where the normalising constant, $p(\mathbf{y}|\mathbf{H})$, is the likelihood of the observation sequence, $\mathbf{y}$. As seen in Section 2.2.2, the B-W algorithm provides an efficient method of computing $p(\mathbf{y}|H)$.

Thus, by combined use of both the Viterbi and the B-W algorithms in (3.1), we may obtain the MAP sequence, $\mathbf{s}^*$, and its corresponding posterior probability. However, a Bayesian wishing to learn about the state sequence structure underlying $\mathbf{y}$ is concerned with the entire posterior distribution, $p(\mathbf{s}|\mathbf{y}, H)$, not just the single sequence $\mathbf{s}^*$. Knowledge of $\mathbf{s}^*$ and $p(\mathbf{s}^*|\mathbf{y}, H)$ tells us virtually nothing about the remainder of the state sequence space (e.g. Is the state sequence $\mathbf{s}^*$ much more probable than all others or are there other sequences with near-optimal posterior probability? Do these near-optimal sequences have very similar state structure to $\mathbf{s}^*$ or do they differ greatly?). As recognised by Kundu et al (1989), there is also a need for specification of sub-optimal state sequences in cases where they are to be used for classification purposes.

For any specific state sequence, $\mathbf{s}$, of interest, direct computation of its posterior probability may be achieved through the expression

$$\begin{aligned} p(\mathbf{s}|\mathbf{y}, H) &= \frac{p(\mathbf{s}, \mathbf{y}|H)}{p(\mathbf{y}|H)} \\ &= \frac{p(\mathbf{y}|\mathbf{s}, H)\, p(\mathbf{s}|H)}{p(\mathbf{y}|H)} \\ &= \frac{1}{p(\mathbf{y}|H)} \left[\pi_{s_1} \prod_{t=1}^{T-1} a_{s_t s_{t+1}} \prod_{t=1}^{T} b_{s_t}(y_t) \right], \end{aligned} \tag{3.2}$$

where the normalising constant, $p(\mathbf{y}|H)$, can be evaluated using the B-W algorithm. If the size of the state sequence space is small, then (3.2) above may be used directly to evaluate the entire posterior distribution $p(\mathbf{s}|\mathbf{y}, H)$, although this will not, in general, be computationally feasible. However, obtaining the top L most probable state sequences and their posterior probabilities will provide much more information about state structure than the single sequence $\mathbf{s}^*$, helping to answer questions such as those posed in the previous paragraph. The potential for generalising the Viterbi algorithm in this way was recognised by Forney (1973) but does not seem to have been formally documented.

3.2 The generalised Viterbi algorithm

Suppose we wish to extract the L best state sequences, denoted $\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^L$ in descending order of posterior probability, and their respective probabilities joint with $\mathbf{y}$, denoted $p^1, p^2, \ldots, p^L$, i.e.

$$p^l = p(\mathbf{s}^l, \mathbf{y}|H), \quad l = 1, 2, \ldots, L.$$

If required, the corresponding posterior probabilities may be obtained from

$$p(\mathbf{s}^l|\mathbf{y}, H) = \frac{p(\mathbf{s}^l, \mathbf{y}|H)}{p(\mathbf{y}|H)}, \tag{3.3}$$

where, as before, the normalising constant $p(\mathbf{y}|H)$ may be computed via the B-W algorithm. The joint probabilities are sufficient, however, if interest lies only in ratios of posterior probabilities.

The generalised Viterbi algorithm seeks to find the top L most probable state sequences maximising the joint probability $p(\mathbf{s}, \mathbf{y}|H)$. The method by which this is done is exactly analogous to the case of the original Viterbi algorithm, although the notationally off-putting formal proof and documentation of the generalisation disguises the intuitiveness of the approach. In brief, at any particular timepoint $t \in \{1, 2, \ldots, T\}$, the generalised algorithm finds the top L most probable partial state sequences up to time t and ending in state i, for all states $i = 1, 2, \ldots, N$. Each of these survivors is then stored along with its corresponding joint probability with the partial observation sequence $\mathbf{y}_{[t]}$. These survivors and probabilities are then used recursively, via the generalised Viterbi algorithm, to construct the survivors at time $t+1$ and their corresponding joint probabilities with $\mathbf{y}_{[t+1]}$. So, at any particular timepoint t, we are only required to store LN survivors and LN corresponding probabilities. The recursive nature of the generalised Viterbi algorithm is based upon the fact that the initial t states of any survivor at time $t+1$ must constitute a survivor at time t. The original algorithm is then simply a special case of the generalised algorithm when $L = 1$.

In order to develop the generalisation, we re-introduce the quantities $\phi_t(i)$ and $\mathbf{s}_t(i)$, which now bear an additional superscript l, as

$$\phi_t^l(i) = \max_{\mathbf{s}_{[t-1]} \in R_t^l(i)} p(\mathbf{s}_{[t-1]}, s_t = i; \mathbf{y}_{[t]}) = p(\mathbf{s}_t^l(i), \mathbf{y}_{[t]}), \tag{3.4}$$

for $t = 1, 2, \ldots, T$, $i = 1, 2, \ldots, N$, $l = 1, 2, \ldots, L_t(i)$, where dependence on the model H is suppressed in order to simplify notation, and

$$L_t(i) = \begin{cases} \min\left\{L, \begin{pmatrix} i+t-2 \\ i-1 \end{pmatrix}\right\} & \text{, for L-R models} \\ \min\{L, N^{t-1}\} & \text{, for ergodic models,} \end{cases}$$

$$S_t^0(i) = \emptyset, \quad \forall t, i$$

$$S_t^l(i) = \{\mathbf{s}_t^1(i), \mathbf{s}_t^2(i), \ldots, \mathbf{s}_t^l(i)\}, \quad \text{for } l = 1, 2, \ldots, L_t(i),$$

$$R_t^l(i) = \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, i) \notin S_t^{l-1}(i)\}.$$

To clarify, $\phi_t^l(i)$ is now the probability of the partial observation sequence, $\mathbf{y}_{[t]}$, jointly with the l^{th} maximal partial state sequence up to time t and ending in state i (denoted by $\mathbf{s}_t^l(i)$); i.e. $\phi_t^l(i)$ is the maximum value of the probability of the partial observation sequence $\mathbf{y}_{[t]}$ jointly with the partial state sequence $(\mathbf{s}_{[t-1]}, i)$ where, in order to avoid duplication, the maximisation is constrained to exclude the top $l-1$ maximal partial state sequences that have already been recovered, constituting the set $S_t^{l-1}(i)$. The quantity $L_t(i)$ (the expression for which employs Equation (1.4)) indicates the number of partial state sequences up to time t and ending in state i which are to be stored at any one timepoint t ($L_t(i) = 1$ in original Viterbi algorithm); so, all such sequences are stored until such time as their total number exceeds L, in which case only the top L are stored. Putting $L = 1$ reduces us to the problem of MAP decoding and we can express $\phi_t(i)$ and $\mathbf{s}_t(i)$ from the original Viterbi algorithm, for all values of t and i, as

$$\begin{aligned} \phi_t(i) &= \phi_t^1(i) \\ \mathbf{s}_t(i) &= \mathbf{s}_t^1(i). \end{aligned}$$

Fixing t and i and decomposing $\phi_{t+1}^l(i)$ into conditional probabilities, as for the original algorithm, we obtain

$$\phi_{t+1}^l(i) = \max_{j=1,2,\ldots,N} \left[a_{ji} \max_{\mathbf{s}_{[t-1]} \in R_{t+1}^l(j,i)} p(\mathbf{s}_{[t-1]}, s_t = j; \mathbf{y}_{[t]}) \right] b_i(y_{t+1}), \tag{3.5}$$

where

$$R_{t+1}^l(j,i) = \left\{ \mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j, i) \notin S_{t+1}^{l-1}(i) \right\}.$$

Equation (3.5) is in exactly the same form as Equation (2.5) in the original algorithm, apart from the second maximisation term, which shall be denoted by

$$M_{t+1}^l(j,i) = \max_{\mathbf{s}_{[t-1]} \in R_{t+1}^l(j,i)} p(\mathbf{s}_{[t-1]}, s_t = j; \mathbf{y}_{[t]}).$$

Again, let ψ^l denote the argument in (3.5) which maximises the term in square brackets, so

$$\psi^l = \arg \max_{j=1,2,\ldots,N} \left[a_{ji} \, M_{t+1}^l(j,i) \right],$$

i.e. ψ^l is the value of the penultimate state (t^{th} state) in the partial sequence $\mathbf{s}_{t+1}^l(i)$. Then, it will be shown in Lemma 3.1 that

$$M_{t+1}^l(j,i) = \phi_t^k(j),$$

for some $k \in \{1, 2, \ldots, l\}$, where $k-1$ indicates the number of elements of $S_{t+1}^{l-1}(i)$ having penultimate state (i.e. t^{th} state) equal to j.

We recall that the indicator $I_{\{E\}}$ of an event E may be defined by

$$I_{\{E\}} = \begin{cases} 1 & \text{, if } E \text{ occurs} \\ 0 & \text{, otherwise} \end{cases}$$

which is now used in the statement of Lemma 3.1.

Lemma 3.1

For $t = 1, 2, \ldots, T-1$, $i, j = 1, 2, \ldots, N$, $l = 1, 2, \ldots, L_{t+1}(i)$:

$$M^l_{t+1}(j,i) = \phi_t^{\eta^l(j)}(j) \tag{3.6}$$

$$\mathbf{s}^l_{t+1}(i) = \left(\mathbf{s}_t^{\xi^l}(\psi^l), i\right), \tag{3.7}$$

where $\psi^0 = 0$

$$\psi^k = \arg \max_{j=1,2,\ldots,N} \left[a_{ji}\, M^k_{t+1}(j,i)\right], \quad k = 1, 2, \ldots, L_{t+1}(i),$$

$$\eta^l(j) = 1 + \sum_{k=0}^{l-1} I_{\{\psi^k = j\}},$$

$$\xi^l = \eta^l(\psi^l).$$

To simplify notation, dependence on t and i is suppressed for the quantities ψ, η and ξ.

Proof

Fix t, j and i. We prove (3.6) and (3.7) by induction on l. First, consider the case $l = 1$:

$$\begin{aligned} S^0_{t+1}(i) &= \emptyset, \quad \text{by definition,} \\ \Longrightarrow R^1_{t+1}(j,i) &= \{\mathbf{s}_{[t-1]}\} \\ \Longrightarrow M^1_{t+1}(j,i) &= \max_{\mathbf{s}_{[t-1]}} p(\mathbf{s}_{[t-1]}, s_t = j; \mathbf{y}_{[t]}) \\ &= \phi^1_t(j) \\ &= \phi_t^{\eta^1(j)}(j), \end{aligned} \tag{3.8}$$

since

$$\begin{aligned} \psi^0 = 0 \implies \eta^1(j) &= 1 + I_{\{\psi^0 = j\}} \\ &= 1, \quad \forall j. \end{aligned}$$

Therefore, (3.6) is proven for the case $l = 1$.

Now, by definition (see Equation (3.4)),

$$\phi^1_{t+1}(i) = p(\mathbf{s}^1_{t+1}(i), \mathbf{y}_{[t+1]}),$$

where, by (3.5),

$$\phi^1_{t+1}(i) = \max_{j=1,2,\ldots,N} \left[a_{ji}\, M^1_{t+1}(j,i)\right] b_i(y_{t+1}). \tag{3.9}$$

Now, since

$$M^1_{t+1}(j,i) = \phi^1_t(j)$$

using (3.8) proved earlier, we can rewrite (3.9) above as

$$\phi^1_{t+1}(i) = \max_{j=1,2,\ldots,N} \left[a_{ji}\, \phi^1_t(j)\right] b_i(y_{t+1}). \tag{3.10}$$

Thus, $\mathbf{s}^1_{t+1}(i)$ can be expressed as $(\mathbf{s}_{[t]}, i)$, for some partial state sequence $\mathbf{s}_{[t]}$ of length t where, from (3.10),

$$\mathbf{s}_{[t]} = \mathbf{s}^1_t(j)$$

such that

$$j = \arg \max_{j=1,2,\ldots,N} \left[a_{ji}\, \phi^1_t(j)\right]. \tag{3.11}$$

But

$$\begin{aligned} \arg \max_{j=1,2,\ldots,N} \left[a_{ji}\, \phi^1_t(j)\right] &= \arg \max_{j=1,2,\ldots,N} \left[a_{ji}\, M^1_{t+1}(j,i)\right], \text{ by (3.8)}, \\ &= \psi^1, \text{ by definition.} \end{aligned}$$

Thus, substituting this in (3.11), we have that

$$\begin{aligned} \mathbf{s}^1_{t+1}(i) &= (\mathbf{s}^1_t(\psi^1), i) \\ &= (\mathbf{s}^{\xi^1}_t(\psi^1), i), \end{aligned}$$

since, by definition,

$$\xi^1 = \eta^1(\psi^1) = 1 + I_{\{\psi^0 = j\}} = 1.$$

Therefore, (3.7) is proven for the case $l = 1$.

For the inductive step, we assume Lemma 3.1 to be true for the value l, i.e.

$$M^l_{t+1}(j,i) = \phi^{\eta^l(j)}_t(j) \tag{3.12}$$

$$\mathbf{s}^l_{t+1}(i) = \left(\mathbf{s}^{\xi^l}_t(\psi^l), i\right). \tag{3.13}$$

Putting $l = l+1$ in (3.6) and (3.7) of the lemma, we are required now to prove that

$$M_{t+1}^{l+1}(j,i) = \phi_t^{\eta^{l+1}(j)}(j) \tag{3.14}$$

$$\mathbf{s}_{t+1}^{l+1}(i) = (\mathbf{s}_t^{\xi^{l+1}}(\psi^{l+1}), i). \tag{3.15}$$

By inspection of the definitions of both the M's and the ϕ's, proving (3.14) reduces to proving that

$$\{\mathbf{s}_{[t-1]} \in R_{t+1}^{l+1}(j,i)\} = \left\{\mathbf{s}_{[t-1]} \in R_t^{\eta^{l+1}(j)}(j)\right\}. \tag{3.16}$$

Now,

$$S_{t+1}^{l}(i) = S_{t+1}^{l-1}(i) \cup \{\mathbf{s}_{t+1}^{l}(i)\}$$

and, as for Equation (3.16), Equation (3.12) implies that

$$\begin{aligned} \{\mathbf{s}_{[t-1]} \in R_{t+1}^{l}(j,i)\} &= \left\{\mathbf{s}_{[t-1]} \in R_t^{\eta^{l}(j)}(j)\right\} \\ \Longrightarrow \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j, i) \notin S_{t+1}^{l-1}(i)\} &= \left\{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{\eta^{l}(j)-1}(j)\right\}. \end{aligned}$$

Thus, we obtain

$$\begin{aligned} &\{\mathbf{s}_{[t-1]} \in R_{t+1}^{l+1}(j,i)\} \\ &= \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j, i) \notin S_{t+1}^{l}(i)\} \\ &= \left\{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{\eta^{l}(j)-1}(j)\right\} \cap \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j, i) \neq \mathbf{s}_{t+1}^{l}(i)\}. \end{aligned} \tag{3.17}$$

But, (3.13) gives

$$\begin{aligned} \mathbf{s}_{t+1}^{l}(i) &= \left(\mathbf{s}_t^{\xi^{l}}(\psi^{l}), i\right) \\ \Longrightarrow \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j, i) \neq \mathbf{s}_{t+1}^{l}(i)\} &= \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \neq \mathbf{s}_t^{\xi^{l}}(\psi^{l})\}. \end{aligned}$$

Substituting this in (3.17),

$$\begin{aligned} \{\mathbf{s}_{[t-1]} \in R_{t+1}^{l+1}(j,i)\} &= \left\{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{\eta^{l}(j)-1}(j)\right\} \cap \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \neq \mathbf{s}_t^{\xi^{l}}(\psi^{l})\} \\ &= \begin{cases} \left\{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{\eta^{l}(j)-1}(j)\right\} & \text{, if } j \neq \psi^{l} \\ \left\{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{\eta^{l}(j)}(j)\right\} & \text{, if } j = \psi^{l}, \end{cases} \end{aligned}$$

since

$$S_t^{\eta^{l}(j)-1}(j) \cup \{\mathbf{s}_t^{\eta^{l}(j)}(j)\} = S_t^{\eta^{l}(j)}(j).$$

Thus

$$\{\mathbf{s}_{[t-1]} \in R_{t+1}^{l+1}(j,i)\} = \{\mathbf{s}_{[t-1]} : (\mathbf{s}_{[t-1]}, j) \notin S_t^{k}(j)\},$$

where

$$k \;=\; \eta^l(j) + I_{\{\psi^l = j\}} \;=\; \eta^{l+1}(j),$$

thereby proving (3.16), and thus proving (3.14) for $l+1$. To prove (3.15), consider Equation (3.5) given, for $l+1$, by

$$\phi_{t+1}^{l+1}(i) = \max_{j=1,2,\ldots,N} \left[a_{ji}\, M_{t+1}^{l+1}(j,i)\right]\; b_i(y_{t+1}).$$

We may now use the fact that (3.14) is true to rewrite this as

$$\begin{aligned}\phi_{t+1}^{l+1}(i) &= \max_{j=1,2,\ldots,N} \left[a_{ji}\, \phi_t^{\eta^{l+1}(j)}(j)\right]\; b_i(y_{t+1}) \\ &= \left[a_{ji}\, \phi_t^{\eta^{l+1}(j)}(j)\right]\; b_i(y_{t+1}), \quad \text{for } j = \psi^{l+1}.\end{aligned}$$

Thus, the corresponding maximal state sequence, $\mathbf{s}_{t+1}^{l+1}(i)$, can be expressed as

$$\mathbf{s}_{t+1}^{l+1}(i) = (\mathbf{s}_{[t-1]}, \psi^{l+1}, i),$$

for some partial state sequence $\mathbf{s}_{[t-1]}$, where $(\mathbf{s}_{[t-1]}, \psi^{l+1})$ is the partial state sequence up to time t and ending in state ψ^{l+1} which gives rise to $\phi_t^{\eta^{l+1}(j)}(j)$. By the definition of the ϕ's, this partial sequence is simply given by

$$\begin{aligned}(\mathbf{s}_{[t-1]}, \psi^{l+1}) &= \mathbf{s}_t^{\eta^{l+1}(\psi^{l+1})}(\psi^{l+1}) \\ &= \mathbf{s}_t^{\xi^{l+1}}(\psi^{l+1}) \\ \Longrightarrow\; \mathbf{s}_{t+1}^{l+1}(i) &= \left(\mathbf{s}_t^{\xi^{l+1}}(\psi^{l+1}),\, i\right).\end{aligned}$$

Hence, (3.15) is also true.

Thus, by induction on l, (3.6) and (3.7) are true in general and the lemma is proven. □

So, for fixed $j \in \{1, 2, \ldots, N\}$, if the full set of maximal sequences at time $t+1$, given by $S_{t+1}^{L_{t+1}(i)}(i)$, contains m sequences with penultimate state equal to j, then

$$M_{t+1}^{l}(j,i) = \begin{cases} \phi_t^1(j), & l = 1, \ldots, l_1 \\ \phi_t^2(j), & l = l_1+1, \ldots, l_2 \\ \vdots & \vdots \\ \phi_t^{m+1}(j), & l = l_m+1, \ldots, L_{t+1}(i), \end{cases}$$

where l_1 is the smallest k s.t. $\psi^k = j$, l_2 is the next smallest k s.t. $\psi^k = j$, etc. Therefore, $M_{t+1}^{l}(j,i) = \phi_t^1(j)$ until $l = l_1$, when $\mathbf{s}_t^1(j)$ is used in the construction of $\mathbf{s}_{t+1}^{l_1}(i)$ and so

must be excluded from further maximisations (in order to avoid duplication of $\mathbf{s}_{t+1}^{l_1}(i)$). Then, $M_{t+1}^{l}(j,i) = \phi_t^2(j)$ for $l > l_1$ until $\mathbf{s}_t^2(j)$ is used in the construction of $\mathbf{s}_{t+1}^{l_2}(i)$, etc., continuing in this manner until all $L_{t+1}(i)$ sequences at time $t+1$, ending in i, are collected.

It is worth mentioning that if ties occur between state sequences at any of the maximisation stages of the algorithm, it does not matter which of these maximal sequences is selected. Furthermore, if it happens that $R_{t+1}^{l}(j',i) = \emptyset$, for some $j' \in \{1,2,\ldots,N\}$ and $l \leq L_{t+1}(i)$, which implies that all $L_t(j')$ of the maximal state sequences up to time t and ending in state j' have been utilised in constructing $S_{t+1}^{l}(i)$, then j' is simply removed as an argument from the maximisation over values of j in (3.5).

As far as computational effort is concerned, the number of calculations in the generalised Viterbi algorithm is of the order LN^2T, indicating that computation times are increased by a factor of L in comparison with the original algorithm. For example, under an ergodic model with $N = 25$, $T = 25$, $L = 25$, computation time for the generalised Viterbi algorithm on a Sun SPARC Workstation is approximately 4 seconds; similarly, under an ergodic model with $N = 10$, $T = 100$, $L = 25$, computation time for the algorithm is approximately 7.5 seconds. Now, for all HMM applications that we have encountered in the literature, the number of states satisfies $N \leq 30$, and it seems likely that implementation of the generalised algorithm will be most useful when applied to observation sequences (or segments thereof) of reasonably short length ($T < 100$, say). Thus, the examples above may be considered as "worst-case" situations in terms of computation power required and we would expect most practical applications of the generalised Viterbi algorithm to be capable of real-time calculations.

Formal implementation of the generalised Viterbi algorithm is summarised in Figure 3.1.

3.3 Examples

3.3.1 General examples

To illustrate how the generalised Viterbi algorithm can improve on the original algorithm in terms of providing information about state structure underlying the modelled

Initialisation For $i = 1, 2, \ldots, N$,

$$\begin{aligned} \phi_1^1(i) &= \pi_i \, b_i(y_1) \\ \mathbf{s}_1^1(i) &= (i) \end{aligned}$$

Recursion For $t = 1, 2, \ldots, T-1$, $i = 1, 2, \ldots, N$, $l = 1, 2, \ldots, L_{t+1}(i)$,

$$\begin{aligned} \phi_{t+1}^l(i) &= \max_{j \in \mathcal{J}_t^l} \left[a_{ji} \, \phi_t^{\eta^l(j)}(j) \right] \, b_i(y_{t+1}), \\ \mathbf{s}_{t+1}^l(i) &= \left(\mathbf{s}_t^{\xi^l}(\psi^l), i \right), \end{aligned}$$

$$\begin{aligned} \text{where} \quad \mathcal{J}_t^l &= \{ j \in \{1, 2, \ldots, N\} : \eta^l(j) \leq L_t(j) \} \\ \psi^0 &= 0 \\ \psi^l &= \arg\max_{j \in \mathcal{J}_t^l} \left[a_{ji} \, \phi_t^{\eta^l(j)}(j) \right], \quad l = 1, 2, \ldots, L_{t+1}(i), \\ \eta^l(j) &= 1 + \sum_{k=0}^{l-1} I_{\{\psi^k = j\}}, \\ \xi^l &= \eta^l(\psi^l). \end{aligned}$$

Termination For $l = 1, 2, \ldots, L$,

$$\begin{aligned} p^l &= \max_{j \in \mathcal{J}_T^l} \phi_T^{\eta^l(j)}(j), \\ \mathbf{s}^l &= \mathbf{s}_T^{\xi^l}(\psi^l), \end{aligned}$$

$$\begin{aligned} \text{where} \quad \mathcal{J}_T^l &= \{ j \in \{1, 2, \ldots, N\} : \eta^l(j) \leq L_T(j) \} \\ \psi^0 &= 0 \\ \psi^l &= \arg\max_{j \in \mathcal{J}_T^l} \phi_T^{\eta^l(j)}(j), \quad l = 1, 2, \ldots, L_{t+1}(i), \\ \eta^l(j) &= 1 + \sum_{k=0}^{l-1} I_{\{\psi^k = j\}}, \\ \xi^l &= \eta^l(\psi^l). \end{aligned}$$

Figure 3.1: The generalised Viterbi algorithm

processes, we initially consider 3 particular HMMs which, for the sake of simplicity, are specified using finite discrete observation distributions, i.e.

$$b_i(y_t) = b_{ij}, \quad \text{if } y_t = j, \qquad j \in \{1, 2, \ldots, K\}$$

with

$$b_{ij} = p(\text{ symbol } j \text{ emitted } | s_t = i\,), \quad \forall t.$$

The conclusions drawn from the examples apply quite generally and are not specific to the choice of discrete observation distributions.

Before looking at the examples, some concept of "similarity of state sequences" needs to be introduced. Given sequences $\mathbf{s}$ and $\mathbf{s}'$ belonging to some state sequence space $\mathcal{S}$, one measure of their dissimilarity might be the number of states differing between them (or, in standardised form, the proportion of all T states which differ). In this way, the neighbourhood of a particular sequence can be thought of as containing all other sequences with very similar state structure, e.g. for some fixed $t' \in \{1, 2, \ldots, T\}$, if $s_t = s'_t$, $\forall t \neq t'$ and $s_{t'} \neq s'_{t'}$, then $\mathbf{s}$ and $\mathbf{s}'$ can be thought of as 1-state neighbours, etc.

We learn about state sequence structure through the posterior distribution, $p(\mathbf{s}|\mathbf{y}, H)$, which has many properties and features that are impossible to identify solely from knowledge of the global mode, $\mathbf{s}^1$, and its posterior probability. In the following examples, the generalised Viterbi algorithm is used to collect the top L sequences and these are tabulated along with their corresponding posterior probabilities obtained from (3.3) using the B-W algorithm.

This example is designed to show how the generalised Viterbi algorithm can provide important information about underlying state structure. The first model considered is a L-R model, specified by $N = 4$, $K = 5$ and $T = 10$, under which 286 distinct state sequences are possible. The model parameters are given by

$$\boldsymbol{\pi} = \begin{pmatrix} 0.60 & 0.20 & 0.10 & 0.10 \end{pmatrix}$$

$$A = \begin{pmatrix} 0.65 & 0.25 & 0.05 & 0.05 \\ 0.00 & 0.60 & 0.30 & 0.10 \\ 0.00 & 0.00 & 0.70 & 0.30 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{pmatrix}, \quad B = \begin{pmatrix} 0.60 & 0.20 & 0.10 & 0.05 & 0.05 \\ 0.30 & 0.20 & 0.10 & 0.05 & 0.35 \\ 0.10 & 0.10 & 0.10 & 0.20 & 0.50 \\ 0.05 & 0.10 & 0.30 & 0.30 & 0.25 \end{pmatrix}.$$

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	**1 1 1 2 3 3 3 3 3 3**	**0.0363**	1.000
2	1 1 1 4 4 4 4 4 4 4	0.0289	1.256
3	1 1 2 3 3 3 3 3 3 3	0.0279	1.301
4	1 2 2 3 3 3 3 3 3 3	0.0258	1.407
5	1 1 1 3 3 3 3 3 3 3	0.0242	1.500
6	1 1 1 2 3 3 3 3 3 4	0.0233	1.558
7	1 1 1 2 4 4 4 4 4 4	0.0202	1.797
8	1 1 2 3 4 4 4 4 4 4	0.0200	1.815
9	1 2 2 3 4 4 4 4 4 4	0.0185	1.962
	TOTAL	0.2251	

Table 3.1: Posterior state sequence information underlying $\mathbf{y} = (1\ 2\ 1\ 5\ 4\ 2\ 5\ 5\ 5\ 4)$ for a L-R model

Tables 3.1 and 3.2 display some results for 2 observation sequences simulated from this model. Figures in the final column of the tables give some idea of the spread of the distribution around the modes, values close to 1 indicating sequences with near-optimal posterior probability. In Table 3.1, the top 9 sequences appear to fall into 2 groups of very different state structure:

$$\left\{\begin{array}{c} 1\,1\,1\,2\,3\,3\,3\,3\,3\,4 \\ \mathbf{1\,1\,1\,2\,3\,3\,3\,3\,3\,3} \\ 1\,1\,1\,3\,3\,3\,3\,3\,3\,3 \\ 1\,1\,2\,3\,3\,3\,3\,3\,3\,3 \\ 1\,2\,2\,3\,3\,3\,3\,3\,3\,3 \end{array}\right\} \quad \text{and} \quad \left\{\begin{array}{c} 1\,1\,1\,2\,4\,4\,4\,4\,4\,4 \\ \mathbf{1\,1\,1\,4\,4\,4\,4\,4\,4\,4} \\ 1\,1\,2\,3\,4\,4\,4\,4\,4\,4 \\ 1\,2\,2\,3\,4\,4\,4\,4\,4\,4 \end{array}\right\}.$$

The first group constitutes a close neighbourhood of $\mathbf{s}^1$ and the second constitutes a close neighbourhood of $\mathbf{s}^2$, the latter having near-optimal posterior probability, where $\mathbf{s}^1$ and $\mathbf{s}^2$ are expressed in bold type. This would seem to indicate some sort of "multimodal" structure in the distribution (in terms of the state sequence neighbourhoods previously described) and any inference about state structure based solely on $\mathbf{s}^1$ would be misleading. Contrast this with the distribution represented in Table 3.2, where the top 4 sequences, carrying 72% of the total posterior probability, are sharply peaked around the mode, $\mathbf{s}^1$, revealing a strong indication of the likely state structure underlying the observation sequence.

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	**4 4 4 4 4 4 4 4 4 4**	**0.4492**	1.000
2	3 3 4 4 4 4 4 4 4 4	0.1258	3.571
3	3 4 4 4 4 4 4 4 4 4	0.0898	5.002
4	3 3 3 4 4 4 4 4 4 4	0.0587	7.652
	TOTAL	0.7235	

Table 3.2: Posterior state sequence information underlying $\mathbf{y} = (4\ 5\ 4\ 4\ 4\ 5\ 4\ 2\ 3\ 3)$ for a L-R model

Table 3.3 lists the top 15 sequences for a posterior distribution under a much larger, ergodic model, specified by $N = 4$, $K = 5$ and $T = 15$, with $S \simeq 10^9$. It illustrates the point that, as T and S increase, sub-optimal sequences will tend to have state structure and posterior probability more similar to that of the mode, and that the top L sequences will carry an ever-decreasing proportion of the total probability as S increases, for fixed L. However, the generalised Viterbi algorithm can still provide information about how the $L-1$ sub-optimal sequences differ from the mode, the location of these differences, etc. For example, in Table 3.3, where the model parameters are given by

$$\pi = \begin{pmatrix} 0.60 & 0.10 & 0.20 & 0.10 \end{pmatrix},$$

$$A = \begin{pmatrix} 0.45 & 0.25 & 0.15 & 0.15 \\ 0.20 & 0.50 & 0.20 & 0.10 \\ 0.20 & 0.10 & 0.20 & 0.50 \\ 0.30 & 0.35 & 0.05 & 0.30 \end{pmatrix}, \quad B = \begin{pmatrix} 0.60 & 0.20 & 0.10 & 0.05 & 0.05 \\ 0.30 & 0.20 & 0.10 & 0.05 & 0.35 \\ 0.10 & 0.10 & 0.10 & 0.20 & 0.50 \\ 0.05 & 0.10 & 0.30 & 0.30 & 0.25 \end{pmatrix},$$

each sequence differs at most by 4 states from any other sequence, the elements in bold type indicating the most stable regions which remain constant over the top 15 sequences. Locally optimal sequences may also be of value when T is very large (particularly for ergodic models), in which case the generalised Viterbi algorithm can be applied to shorter segments of interest in the observation sequence.

3.3.2 Handwritten text recognition

In the second instance, we consider the use of the generalised Viterbi algorithm for classification purposes. The example considered here is set up in Kundu et al (1989) for the

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	**1 1 2 2** 2 2 2 **3** 4 4 2 2 2 **2 2**	0.7335×10^{-4}	1.000
2	**1 1 2 2** 2 2 2 **3 4 4** 1 4 4 **2 2**	0.7129 "	1.029
3	**1 1 2 2** 2 2 2 **3** 4 4 4 4 4 **2 2**	0.7129 "	1.029
4	**1 1 2 2** 2 3 3 **3** 4 4 2 2 2 **2 2**	0.6706 "	1.094
5	**1 1 2 2** 2 3 3 **3** 4 4 1 4 4 **2 2**	0.6518 "	1.125
6	**1 1 2 2** 2 3 3 **3** 4 4 4 4 4 **2 2**	0.6518 "	1.125
7	**1 1 2 2** 2 4 2 **3** 4 4 2 2 2 **2 2**	0.6161 "	1.191
8	**1 1 2 2** 3 4 2 **3** 4 4 2 2 2 **2 2**	0.6161 "	1.191
9	**1 1 2 2** 2 2 2 **3** 4 4 2 3 4 **2 2**	0.6161 "	1.191
10	**1 1 2 2** 2 4 2 **3** 4 4 1 4 4 **2 2**	0.5989 "	1.225
11	**1 1 2 2** 3 4 2 **3** 4 4 1 4 4 **2 2**	0.5989 "	1.225
12	**1 1 2 2** 2 4 2 **3** 4 4 4 4 4 **2 2**	0.5989 "	1.225
13	**1 1 2 2** 3 4 2 **3** 4 4 4 4 4 **2 2**	0.5989 "	1.225
14	**1 1 2 2** 2 3 3 **3** 4 4 2 3 4 **2 2**	0.5633 "	1.302
15	**1 1 2 2** 2 2 2 **3** 4 4 2 4 4 **2 2**	0.5545 "	1.323
	TOTAL	0.9495×10^{-3}	

Table 3.3: Posterior state sequence information underlying $\mathbf{y}$ = (1 1 5 5 2 4 5 5 3 4 2 3 3 2 5) for an ergodic model

purposes of recognition of handwritten words, useful in applications such as writer identification and postal address recognition for the Mail Service. Nag et al (1986) describe a text recognition procedure using HMMs in an identical way to that discussed in earlier chapters for ASR. However, Kundu & Bahl (1988) and Kundu et al (1989) introduce an alternative application of HMMs in this area. Each state in the model represents a letter in the English alphabet, assigned to which is a discrete observation distribution over K symbols, as before. During classification, the handwritten word under study is segmented into individual letters, each letter being represented by an observation vector of its features which is then approximated by one of the K observation symbols using vector quantisation. The vector which is initially extracted from the raw handwritten letter under consideration is chosen to represent the geometrical features exhibited by that letter (based on distinctive properties such as the number of loops, relative vertical-to-horizontal distance and the existence of isolated dots). For further details, see Kundu et al (1989). It is then required to recover the original word from this resulting observation sequence. In this context, state sequences represent strings of letters and recognition will normally operate within the confines of a specified word vocabulary and under further constraints, due to grammar and semantics, when recognition is in the context of a sentence.

The model parameters $H = (\pi, A, B)$ used in Kundu et al (1989) were estimated from a large sample of data and then considered fixed. The values of π and A were estimated based on a vocabulary of English words consisting of 67,320 state-transitions between adjacent letters. A set of 90 discrete observation symbols were selected using vector quantisation techniques based on a sample of 2500 handwritten words. We restrict consideration, in this example, to words formed from the 9 letters A–I, the initial state vector, π, and one-step state transition matrix, A, being normalised versions of those given in Kundu et al (1989) and the observation symbols used being 1–25 from the full set of 90 symbols given in Kundu et al (1989). Any results obtained under this restricted observation distribution will be inferior to those obtained under the full distribution but will serve to illustrate some important points. The estimated model parameters are given in Appendix B.1.

Simulated data was obtained by generating observation sequences from the aforementioned observation distributions for the 3 words: BAD, CHIEF and BEHEAD. The generalised Viterbi algorithm was then used to find the top 10 sequences in each case,

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	**B E D**	**0.4132**	1.000
2	B E A	0.3445	1.199
3	B E F	0.0741	5.576
4	**B A D**	**0.0210**	19.676
5	D E D	0.0175	23.611
6	D E A	0.0146	28.301
7	**F E D**	**0.0117**	35.316
8	B E I	0.0113	36.566
9	**B E G**	**0.0111**	37.225
10	F E A	0.0097	42.598
	TOTAL	0.9287	

Table 3.4: Posterior state sequence information underlying $\mathbf{y} = (16\ 17\ 5)$ where the true word is BAD

the results being displayed in Tables 3.4–3.6. Sequences in bold type represent valid words from the English vocabulary.

The original Viterbi algorithm recognises the words by means of the MAP sequences causing misclassification to occur in all 3 cases. However, the generalised Viterbi algorithm may be used to extract the top L sequences ($L = 10$ in this example) and, thus, provide L candidate words to be considered for classification. Consideration may be constrained to a subset of this maximal set by imposing some criterion such as selecting:

- $\mathbf{s}^i$ s.t. $p(\mathbf{s}^1|\mathbf{y}, H)/p(\mathbf{s}^i|\mathbf{y}, H) < M$, for some upper limit M,
- $\mathbf{s}^i$, $i = 1, 2, \ldots, L'$, where L' is the least value of j s.t. $\sum_{i=1}^{j} p(\mathbf{s}^i|\mathbf{y}, H) \geq M$, for some lower limit M.

Isolated word recognition then consists of selecting the sequence which has highest posterior probability amongst those which are members of the vocabulary. If the maximal set contains no words present in the vocabulary, then the MAP sequence is chosen (useful if new, but valid, words not yet assigned to the vocabulary are met). Thus, in our example, both CHIEF and BEHEAD are correctly recognised, whilst BAD is misclassified

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	C H I E D	0.3356	1.000
2	C H I E A	0.2798	1.199
3	C A I E D	0.0837	4.010
4	C A I E A	0.0697	4.815
5	**C H I E F**	**0.0602**	5.575
6	C A I E F	0.0150	22.373
7	C H I A D	0.0109	30.789
8	C H I E I	0.0092	36.478
9	C H I E G	0.0090	37.289
10	C H I D A	0.0080	41.950
	TOTAL	0.8811	

Table 3.5: Posterior state sequence information underlying $\mathbf{y}$ = (1 23 4 22 7) where the true word is CHIEF

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$	$p(\mathbf{s}^1\|\mathbf{y}, H)/p(\mathbf{s}^i\|\mathbf{y}, H)$
1	B E D E D A	0.1390	1.000
2	B E D E A D	0.1268	1.096
3	B E H E D A	0.0650	2.138
4	**B E H E A D**	**0.0594**	2.340
5	B E D E A F	0.0408	3.407
6	B E D E F F	0.0333	4.174
7	B E D E D I	0.0330	4.212
8	B E D E F A	0.0302	4.603
9	B E D E A I	0.0269	5.167
10	B E D E D D	0.0192	7.240
	TOTAL	0.5736	

Table 3.6: Posterior state sequence information underlying $\mathbf{y}$ = (16 22 25 22 12 11) where the true word is BEHEAD

as BED. However, if the 3 words occur within the context of a sentence, e.g.

> "The CHIEF way for royalty, in the 16^{th} century, to rid themselves of BAD advisors was to BEHEAD them."

then suitable language constraints of grammar and semantics will cause the choice of BED to be abandoned in favour of BAD.

Thus, for the purposes of classification, it can be seen that the generalised Viterbi algorithm offers greater chance of correct recognition, especially in the face of "poor" handwriting which might otherwise give rise to meaningless or invalid words due to its inconsistency with the model parameters.

3.3.3 A classification problem in pharmaceutical research

The problem and data analysed in this example were provided by biological research at Ciba-Geigy, Switzerland (for full details of the experiment, from a biological perspective, see Ruigt et al (1989)). The problem is that of automatic classification of the sleeping status of rats based on data obtained from an electroencephalograph (EEG). The latter records electrical impulses in the brain which can characterise the sleep status of the subject under study. The underlying model adopted is a HMM with first-order Markov chain state structure. Sleep states (the underlying states of the HMM) fall into one of 7 classes ($N = 7$), represented as follows:

Sleep state	Description
1	wake
2	spindle
3	light sleep
4	deep sleep
5	pre-REM
6	REM
7	artifact

where REM denotes "rapid eye movement". Each Markov chain state corresponds to a physical sleep status, with state 7, representing "artifact", simply indicating signals registered due to external effects such as movement of the animals.

The data analysed was based on readings from a single rat. At each discrete time-point t, an EEG data point was recorded consisting of an observation vector of dimension 6, each component of which represents a frequency band. These 6 bands were selected based on a preliminary analysis which showed them to exhibit stable and characteristic properties of the electronic signal. It is assumed that each state-dependent observation distribution is multivariate Normal over 6 dimensions.

The rat was given a vehicle solution and, later, this was used to administer a drug. Its EEG signals were monitored in order to detect any effects on sleep status. The resulting data on the rat consists of 970 sequentially observed EEG observation vectors, $(y_1, y_2, \ldots, y_{970})$, with $y_t = (y_{t1}, y_{t2}, \ldots, y_{t6})$, $t = 1, 2, \ldots, 970$, measured at 2 second intervals. The actual sleep status underlying each vector was then classified according to an expert. Part of this data will be used as "learning data", to estimate the parameters of the state observation distributions, and the remainder will be used for validation purposes. The expert also provided a set of scores indicating the likelihood of the various transitions between sleep states, scores being on a range of 1–5 and increasing with increase in transition likelihood. This score matrix, denoted by $Z = (z_{ij})$, is presented below in the same format as the transition matrix A, so that z_{ij} denotes the score for the transition from state i to state j:

$$Z = \begin{pmatrix} 5 & 4 & 4 & 1 & 2 & 1 & 4 \\ 4 & 5 & 5 & 3 & 4 & 1 & 4 \\ 4 & 5 & 5 & 5 & 1 & 1 & 2 \\ 3 & 2 & 4 & 5 & 4 & 3 & 2 \\ 3 & 3 & 3 & 2 & 4 & 5 & 3 \\ 4 & 4 & 4 & 3 & 3 & 5 & 4 \\ 4 & 3 & 1 & 1 & 1 & 3 & 4 \end{pmatrix}.$$

This system of scores is converted into an estimate of the state transition matrix A by simply normalising each row to sum to 1, using the transformation $p(\text{transition}) \propto \text{score}$. Other score transformations yielded similar results. The initial state distribution is assumed to be uniform, as its specification should have negligible effect on very long state sequences from an unconstrained ergodic model of the type under consideration here. For each $i = 1, 2, \ldots, 7$, the observation distribution $b_i(\cdot)$ associated with state i is assumed to be multivariate Normal with dimension 6, mean vector $\boldsymbol{\mu}^i = (\mu_1^i, \mu_2^i, \ldots, \mu_6^i)$ and covariance matrix $\Sigma^i = (\Sigma_{jk}^i)$. Each $\boldsymbol{\mu}^i$ and Σ^i is approximated by its MLE based on

the training data $\{(y_t, s_t) : t = 501, 502, \ldots, 970\}$, where s_t is the expert classification of the sleep status underlying y_t. So, for $i = 1, 2, \ldots, 7$, the MLEs for $\boldsymbol{\mu}^i$ and Σ^i, respectively, are given by

$$\hat{\mu}^i_j = \frac{1}{n_i} \sum_{t \in \{t : s_t = i\}} y_{tj}, \qquad j = 1, 2, \ldots, 6,$$

and

$$\hat{\Sigma}^i_{jk} = \frac{1}{n_i} \sum_{t \in \{t : s_t = i\}} (y_{tj} - \hat{\mu}^i_j)(y_{tk} - \hat{\mu}^i_k), \qquad j = 1, 2, \ldots, 6, \; k = 1, 2, \ldots, 6,$$

where

$$n_i = |\{t : s_t = i\}|, \qquad T = \sum_{i=1}^{7} n_i.$$

These model estimates are then considered fixed.

The aim is to classify the remainder of the data (i.e. the observation sequence up to time $t = 500$), the expert classification of which is used to evaluate misclassification percentages for validation purposes. The original analysis performed by Ciba-Geigy (Racine-Poon, 1991) involved "on-line" classification of the state underlying each EEG data point received (i.e. sequential statistical procedures were employed). This sequential procedure classifies the state at time t, s_t, immediately after y_{t+1} has been observed, i.e. according to $p(s_t | \mathbf{y}_{[t-1]}, y_t, y_{t+1})$, where we recall that $\mathbf{y}_{[t]} = (y_1, y_2, \ldots, y_t)$ and we suppress dependence on the set of model parameters H in the notation since they are fixed. This latter distribution may be evaluated for all values of s_t via Lemma 3.2.

Lemma 3.2

For $t = 2, 3, , \ldots, T - 1$,

$$p(s_t | \mathbf{y}_{[t-1]}, y_t, y_{t+1}) \propto \sum_{s_{t+1}} \sum_{s_{t-1}} p(s_{t-1} | \mathbf{y}_{[t-1]})\, p(s_t | s_{t-1})\, p(y_t | s_t) \times p(s_{t+1} | s_t)\, p(y_{t+1} | s_{t+1}), \qquad (3.18)$$

with

$$p(s_1 | y_1, y_2) \propto \sum_{s_2} p(s_1)\, p(y_1 | s_1)\, p(s_2 | s_1)\, p(y_2 | s_2), \qquad (3.19)$$

at the initial timepoint, and

$$p(s_T | \mathbf{y}_{[T-1]}, y_T) \propto \sum_{s_{T-1}} p(s_{T-1} | \mathbf{y}_{[T-1]})\, p(s_T | s_{T-1})\, p(y_T | s_T), \qquad (3.20)$$

at the final timepoint. Furthermore,

$$p(s_1|y_1) \propto p(y_1|s_1)\, p(s_1) \tag{3.21}$$

and, for $t = 2, 3, \ldots, T$, $p(s_t|\mathbf{y}_{[t]})$ may be computed via a recursive equation in t, using $p(s_{t-1}|\mathbf{y}_{[t-1]})$ computed at the previous time $t-1$, given by

$$p(s_t|\mathbf{y}_{[t]}) \propto p(y_t|s_t) \sum_{s_{t-1}} p(s_{t-1}|\mathbf{y}_{[t-1]})\, p(s_t|s_{t-1}). \tag{3.22}$$

Proof

In the following proof, we use Bayes' theorem in the form

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

and, also, the definition of conditional probability in the form

$$p(x, y) = p(x|y)p(y).$$

So, to prove Equation (3.18) for $t = 2, 3, \ldots, T-1$:

$$\begin{aligned}
p(s_t|\mathbf{y}_{[t-1]}, y_t, y_{t+1}) &= \sum_{s_{t+1}} \sum_{s_{t-1}} p(s_{t-1}, s_t, s_{t+1}|\mathbf{y}_{[t-1]}, y_t, y_{t+1}) \\
&\propto \sum_{s_{t+1}} \sum_{s_{t-1}} p(y_{t+1}|s_{t-1}, s_t, s_{t+1}, \mathbf{y}_{[t-1]}, y_t)\, p(s_{t-1}, s_t, s_{t+1}|\mathbf{y}_{[t-1]}, y_t) \\
&\propto \sum_{s_{t+1}} \sum_{s_{t-1}} p(y_{t+1}|s_{t+1})\, p(y_t|s_{t-1}, s_t, s_{t+1}, \mathbf{y}_{[t-1]})\, p(s_{t-1}, s_t, s_{t+1}|\mathbf{y}_{[t-1]}), \\
&\qquad \text{by the conditional independence assumption of (1.3),} \\
&= \sum_{s_{t+1}} \sum_{s_{t-1}} p(y_{t+1}|s_{t+1})\, p(y_t|s_t)\, p(s_{t+1}|s_{t-1}, s_t, \mathbf{y}_{[t-1]})\, p(s_{t-1}, s_t|\mathbf{y}_{[t-1]}), \\
&\qquad \text{by the conditional independence assumption of (1.3),} \\
&= \sum_{s_{t+1}} \sum_{s_{t-1}} p(y_{t+1}|s_{t+1})\, p(y_t|s_t)\, p(s_{t+1}|s_t) \\
&\qquad \times p(s_t|s_{t-1}, \mathbf{y}_{[t-1]})\, p(s_{t-1}|\mathbf{y}_{[t-1]}),
\end{aligned}$$

since, by the Markov property of (1.2) and the conditional independence assumption of (1.3),

$$\begin{aligned}
p(s_{t+1}|s_{t-1}, s_t, \mathbf{y}_{[t-1]}) &= \frac{p(\mathbf{y}_{[t-1]}|s_{t+1}, s_{t-1}, s_t)\, p(s_{t+1}|s_{t-1}, s_t)}{p(\mathbf{y}_{[t-1]}|s_{t-1}, s_t)} \\
&= \frac{p(\mathbf{y}_{[t-1]}|s_{t-1})\, p(s_{t+1}|s_t)}{p(\mathbf{y}_{[t-1]}|s_{t-1})} \\
&= p(s_{t+1}|s_t).
\end{aligned}$$

Similarly,

$$p(s_t|s_{t-1}, \mathbf{y}_{[t-1]}) = p(s_t|s_{t-1})$$

and, thus,

$$p(s_t|\mathbf{y}_{[t-1]}, y_t, y_{t+1}) \propto \sum_{s_{t+1}} \sum_{s_{t-1}} p(y_{t+1}|s_{t+1})\, p(y_t|s_t)\, p(s_{t+1}|s_t)\, p(s_t|s_{t-1})\, p(s_{t-1}|\mathbf{y}_{[t-1]}).$$

Thus, (3.18) is proven. The proofs of (3.19) and (3.20) follow similarly. Now, Equation (3.21) may be obtained directly from the definition of conditional probability. Thus, it remains to prove Equation (3.22):

$$\begin{aligned}
p(s_t|\mathbf{y}_{[t]}) &= \sum_{s_{t-1}} p(s_t, s_{t-1}|\mathbf{y}_{[t-1]}, y_t) \\
&\propto \sum_{s_{t-1}} p(y_t|s_t, s_{t-1}, \mathbf{y}_{[t-1]})\, p(s_t, s_{t-1}|\mathbf{y}_{[t-1]}) \\
&= \sum_{s_{t-1}} p(y_t|s_t)\, p(s_t|s_{t-1}, \mathbf{y}_{[t-1]})\, p(s_{t-1}|\mathbf{y}_{[t-1]}), \\
&\qquad \text{by the conditional independence assumption of (1.3),} \\
&= p(y_t|s_t) \sum_{s_{t-1}} p(s_t|s_{t-1})\, p(s_{t-1}|\mathbf{y}_{[t-1]}).
\end{aligned}$$

Thus, (3.22) is proven. □

Noting that

$$\begin{aligned}
p(s_1) &= \pi_{s_1} \\
p(s_t|s_{t-1}) &= a_{s_{t-1}s_t}, \qquad t = 2, 3, \ldots, T \\
p(y_t|s_t) &= b_{s_t}(y_t), \qquad t = 1, 2, \ldots, T,
\end{aligned}$$

where the transition matrix A and the observation distributions $\{b_i(\cdot) : i = 1, 2, \ldots, 7\}$ have been estimated and are considered known, the probability $p(s_t|\mathbf{y}_{[t-1]}, y_t, y_{t+1})$ may be evaluated using Lemma 3.2 for $s_t = 1, 2, \ldots, 7$, and the sleep status at time t, s_t^*, classified by

$$s_t^* = \arg \max_{s_t=1,2,\ldots,7} p(s_t|\mathbf{y}_{[t-1]}, y_t, y_{t+1}).$$

The original Viterbi algorithm (with $L = 1$) was compared, via misclassification rates (assuming the expert to be expert enough!), to the above sequential procedure. Because the former recovers the MAP sequence given the complete observation sequence, it can be thought of as a retrospective classification procedure and would be expected to perform at least as well as the sequential classifier described above.

Of the sequence of 970 observations classified by the expert, we recall that the final 470 observations were used as the training data set from which the state observation distribution parameters were estimated. There was some imbalance in the amount of data available for each state distribution, which can be seen in the following breakdown of the training set, where n_i denotes the number of observations classified as state i in this set:

State, i	**Number,** n_i
1	169
2	13
3	204
4	29
5	25
6	22
7	8

Of course, the sparsity of data available for certain states may affect their classification potential and a larger training set may be required in practice. However, it will serve for the purposes of illustration and comparison in this example. The estimated observation distribution parameters are given in Appendix B.2.

Observations 1–500 were then classified, first using the sequential procedure and then using the Viterbi algorithm ($L = 1$). The sequential procedure misclassified approximately 16.2% of the states (validating with respect to the expert's classification) and the Viterbi algorithm misclassified approximately 16.6% (which corresponds to the misclassification of 2 more states than the sequential procedure). On closer inspection, the location and values of the misclassified states corresponded exactly for both procedures, bar a couple of states. This agreement is to be expected if both procedures are working well and shows that sequential classification for this particular problem does not result in any degradation of performance compared with retrospective classification. This may be due, in part, to the fact that the dynamic range of the transition probabilities is much smaller than that of the continuous observation probabilities so the transition matrix does not contribute significantly to the classification process and, in addition, the sequential procedure does not condition on the, possibly misclassified, value of s_{t-1} when classifying s_t. However, since the Viterbi algorithm classifies the states given the entire observation sequence, it would always be expected to perform at least as well as

State	Fraction of times state misclassified		Percentage misclassification
	Sequential procedure	Viterbi algorithm	
1	24/249	25/249	$\simeq$ 10%
2	5/6	5/6	$\simeq$ 83%
3	34/155	34/155	$\simeq$ 21%
4	4/31	4/31	$\simeq$ 12%
5	8/18	9/18	$\simeq$ 50%
6	6/21	6/21	$\simeq$ 28%
7	0/19	0/19	$\simeq$ 0%

Table 3.7: Breakdown of misclassified states

State	Most likely states to be confused with
1	3 & 6
2	5
3	4 & 5
4	3
5	3
6	1
7	—

Table 3.8: Most likely confused states

the sequential procedure and in the case of constrained models should perform much better.

A more detailed breakdown of the misclassified states is given in Table 3.7 which displays, for each state, the number of times it is misclassified as a fraction of its total number of occurrences. As would be expected, states 1 and 3 have low misclassification percentages due to the larger amounts of training data available for them and, conversely, state 2 has a high misclassification rate, partly due to the limited amount of training data associated with it. However, Table 3.8 breaks down even further the misclassifications for each state in order to find a more satisfactory explanation. This confusion between certain states (e.g. between states 1 and 6 or between states 3 and 4) can be explained

by studying Figure 3.2 which plots for each state $i = 1, 2, \ldots, 7$ the average value of its mean observation vector, $\hat{\boldsymbol{\mu}}^i$ (corresponding to — i — on the graph), based on the training data. Certain of the states, such as 1 and 6, have mean vectors of a very similar shape and, thus, some observation vectors actually emitted from state 1 may be more consistent with state 6 and vice versa. In contrast, the mean vector associated with state 7 is very distinctive and, thus, even though the available training data for state 7 is sparse, it was always classified correctly in the observation sequence we analysed. In the case of the other states, sparse training data will often result in the variance terms of the observation distribution covariance matrix being under-estimated. This will ultimately produce classification errors between all states with mean vectors of a similar shape, those with under-estimated variances being confused with those states which are better estimated and having mean vector of a similar shape. The solution, if possible, is to increase the amount of training data.

Now, suppose we are particularly interested in certain sections of the observation sequence, e.g. a short section of time directly after a certain drug has been administered. The generalised Viterbi algorithm locates the areas of uncertainty in the MAP sequence and gives likely alternative classifications of these regions. It also provides posterior probabilities which indicate how "optimal" the MAP sequence is in comparison to the other $L - 1$ maximal state sequences.

The generalised Viterbi algorithm with $L = 20$ was employed to analyse 2 short sections of the observation sequence, each comprising 50 observations (the equivalent of 1 minute and 40 seconds of EEG data), the first section being observations 1–50 and the second being observations 51–100. Table 3.9 specifies the MAP sequence in each case along with its respective posterior probability. As previously noted, the size of posterior probabilities will necessarily decrease with increasing sequence length due to the increased size of $\mathcal{S}$, but these probabilites may be used for comparison purposes, e.g. the posterior probability of the sequence for the second section is an order of magnitude greater than that for the first section, leading us to be much more confident in the classification provided for the second section. Thus, studying the state sequence structure underlying observations 1–50 in more depth, by inspection of the top 20 maximal sequences, reveals the location of the uncertain areas of the MAP sequence, $\mathbf{s}^1$, given by

$$\begin{aligned}\mathbf{s}^1 = (&4\,3\,3\,3\,3\,6\,1\,3\,1\,3\,3\,3\,3\,3\,3\,\mathbf{5}\,4\,3\,3\,3\,4\,4\,4\,\mathbf{3}\,\mathbf{3}\\ &4\,\mathbf{3}\,3\,3\,3\,3\,3\,5\,5\,5\,3\,5\,5\,\mathbf{5}\,6\,6\,5\,1\,\mathbf{6}\,3\,3\,3\,3\,3\,3),\end{aligned}$$

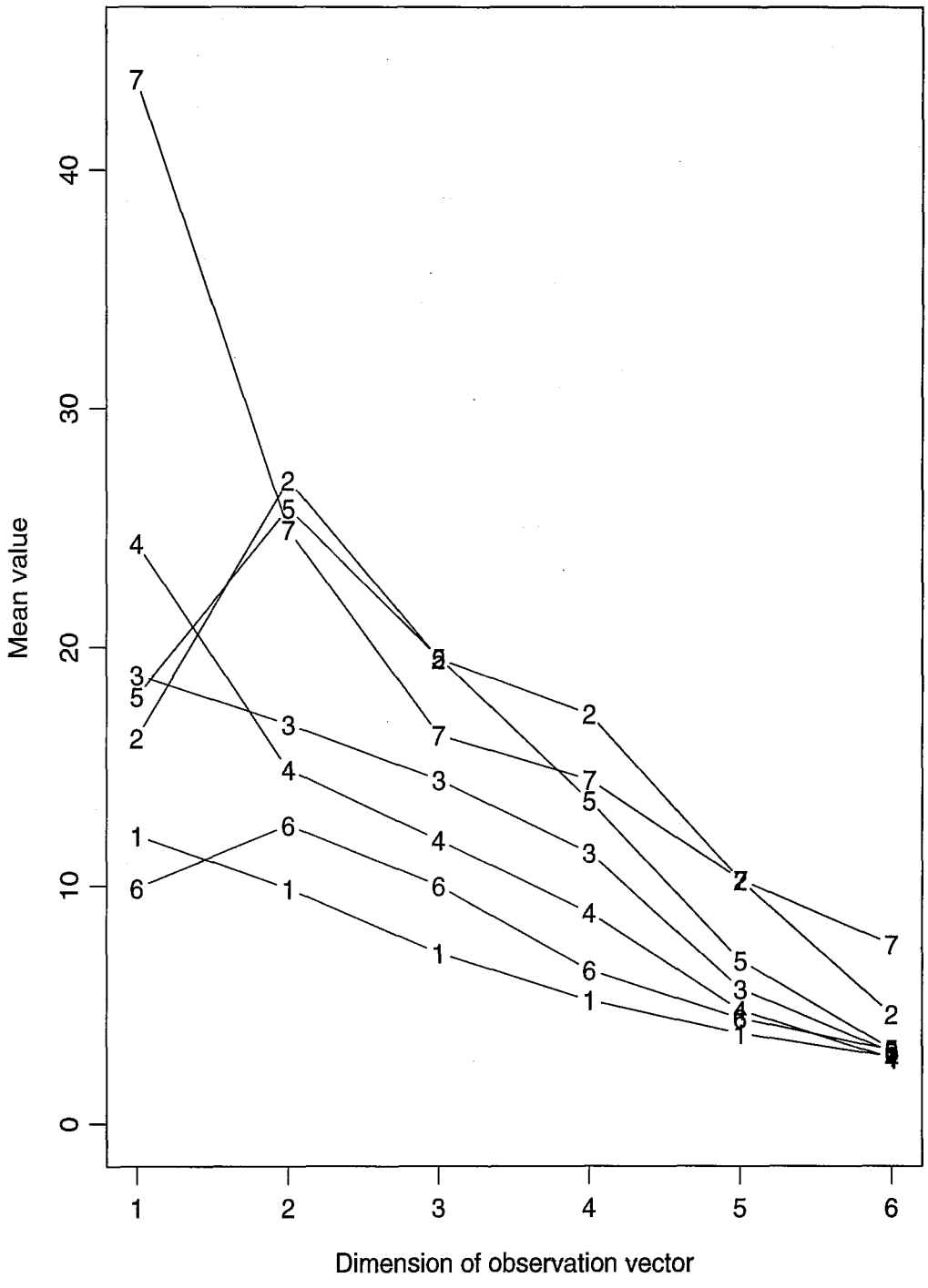


Figure 3.2: Mean observation vectors for the sleep states of a rat

Section	MAP state sequence	Posterior probability
s_1–s_{50}	4 3 3 3 3 6 1 3 1 3 3 3 3 3 3 5 4 3 3 3	
	4 4 4 3 3 4 3 3 3 3 3 3 5 5 5 3 5 5 5 6	
	6 5 1 6 3 3 3 3 3 3	0.222×10^{-3}
s_{51}–s_{100}	4 3 3 3 3 3 4 4 3 4 4 4 3 4 4 3 4 4 4 4	
	4 4 4 4 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3	
	4 4 4 4 4 4 4 4 1 1	0.193×10^{-2}

Table 3.9: MAP state sequences underlying 2 sections of the full EEG observation sequence

State	Alternative classifications
s_{24}	3, 4
s_{25}	3, 4
s_{44}	6, 3
s_{27}	3, 4
s_{39}	5, 3
s_{16}	5, 3

Table 3.10: Alternative classifications for those states varying in the top 20 sequences

where the location of the "uncertain" states within the set of top 20 sequences are indicated in bold. Alternative classifications of these uncertain states are given in Table 3.10, ordered by their "likelihood" (i.e. the most likely classification first, etc). The locations of the uncertain states are also given in descending order of their "uncertainty", i.e. $\mathbf{s}^2$ differs from $\mathbf{s}^1$ at states s_{24} and s_{25} listed first, etc. By comparing Table 3.10 with Table 3.8, it is clear that the alternative classifications arise mainly amongst the most likely confused states whose mean vectors are of similar shape. As to the usefulness of these sub-optimal state sequences, alternative classifications may be more sensible in light of the surrounding states or in the experience of the investigator. If the locations of uncertain states are isolated this may prove insignificant as opposed to the uncertainty of large connected regions of the state sequence. At the very least, they may give an indication as to the confusability between certain states.

3.4 Conclusions

The Viterbi algorithm may be generalised to extract the top L most probable state sequences of the particular posterior distribution under consideration. Knowledge of the mode and its posterior probability alone is often inadequate for drawing any useful inference about the posterior and gives little information about important features or properties of the distribution. Even for large models, it is preferable to use the algorithm with $L > 1$, since it can provide valuable extra information about state structure. When used for purposes of classification, it also offers the flexibility of making sub-optimal choices which may be more meaningful than the MAP sequence. The generalised algorithm may also be applied locally to sections of the observation sequence of particular interest, in order to gain greater insight into the underlying state structure.

Chapter 4

Bayesian inference and Markov chain Monte Carlo methods

4.1 The Bayesian philosophy

Prior to re-addressing the problem of HMM parameter estimation, we introduce the concepts of the Bayesian approach to statistical analysis. The entire Bayesian philosophy can be summarised in the statement of Bayes' theorem. Suppose data $\mathbf{y}$ is obtained from an experiment, where $\mathbf{y}$ is a realisation from the sample space $\mathcal{Y}$, the set of all possible experimental outcomes. This data is collected in order to investigate some particular process of interest and is modelled statistically by $p(\mathbf{y}|\boldsymbol{\theta})$, a probability distribution dependent on some unknown parameter $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_k)$ belonging to the parameter space Θ. The aim is to make inference about $\boldsymbol{\theta}$ based on the evidence provided by the data $\mathbf{y}$. Prior to observing $\mathbf{y}$, a distribution $p(\boldsymbol{\theta})$ is specified, known as the prior distribution, which represents prior information available as to the likely values $\boldsymbol{\theta}$ assumes. This may be in the form of previous empirical evidence, an expert's personal beliefs or any other relevant information. The process of updating these beliefs, in view of the data $\mathbf{y}$, is achieved via Bayes' theorem:

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{y}) &= \frac{p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}} \\ &\propto p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta}), \end{aligned} \tag{4.1}$$

where $p(\boldsymbol{\theta}|\mathbf{y})$ represents the posterior distribution of $\boldsymbol{\theta}$ given $\mathbf{y}$. When treating the model $p(\mathbf{y}|\boldsymbol{\theta})$ as a function of $\boldsymbol{\theta}$, it is usual to refer to it as the likelihood and, thus, Bayes' theorem may be expressed as

$$\text{"posterior"} \propto \text{"likelihood"} \times \text{"prior"}.$$

The posterior distribution can, thus, be thought of as the combination of prior information about $\boldsymbol{\theta}$ with the evidence provided by the data $\mathbf{y}$.

The posterior distribution, specified via Bayes' theorem in (4.1), contains all the information that is required to make inference about $\boldsymbol{\theta}$, based on $\mathbf{y}$. It may be that the posterior distribution in its entirety is the desired outcome of the analysis but, often, one of a wide variety of summary statistics available from the full distribution will prove sufficient. For example, if simple point estimates are desired, then relevant location properties of the distribution such as mean, mode or median may be extracted, along with any other useful properties such as measures of scale or percentiles of the distribution.

In comparison with the classical approach to inference (see Cox & Hinkley (1974) for an overview), which is based solely on the likelihood $p(\mathbf{y}|\boldsymbol{\theta})$, the Bayesian approach requires specification of an additional component: the prior distribution, $p(\boldsymbol{\theta})$ (Chapter 3 of Berger (1985) reviews techniques for constructing priors). It is the, sometimes very subjective, specification and use of this distribution which proves central to the controversy which exists between classical and Bayesian statisticians. Classical statisticians argue that there is no place for subjectivity in the scientific and data-driven world of statistical analysis. However, the subjective preferences and experience of a classical statistician are always involved in any analysis that (s)he carries out; e.g. specification of the form of parametric model for the data, $p(\mathbf{y}|\boldsymbol{\theta})$, and any assumptions that accompany it, such as independence between parameters; choice of loss functions when conducting decision analysis; specification of "acceptable" significance levels when testing hypotheses. Choices such as these, and many more, must be made subjectively by classical statisticians every time an analysis is performed and will, ultimately, affect any results obtained. The Bayesian statistician accepts the fact that incorporating prior information about the process under study into the analysis is not only desirable but inherent in any investigation.

Box & Tiao (1973) and Berger (1985) both provide a comprehensive guide to the world of Bayesian theory and methodology.

4.2 Bayesian computations

The Bayesian approach to statistical analysis is attractive not only in its conceptual ability to combine vital prior information with current empirical evidence but also in its straightforward and intuitive theoretical implementation. This is in stark contrast to the classical statistical approach, where the method of analysis is governed by its ultimate function (such as hypothesis testing, point estimation, etc). Bayesian analysis is achieved, in every situation, through the derivation of the posterior distribution from which all desired inference questions may be answered. However, practical difficulties arise due to the complexity of computations required within the Bayesian framework, thus precluding its widespread and automatic use. This stems from the frequent need to compute complicated multidimensional integrals in order to obtain statistical summaries of interest from the posterior distribution.

Given a parameter $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_k)$ and data $\mathbf{y}$, it is often the case that the posterior distribution $p(\boldsymbol{\theta}|\mathbf{y})$ is only available up to proportionality, i.e.

$$p(\boldsymbol{\theta}|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta})\, p(\boldsymbol{\theta}).$$

Then, for example, the marginal distribution for one particular component of interest, θ_i say, is obtained by integrating over the remaining components and dividing by the normalising constant for the posterior distribution:

$$p(\theta_i|\mathbf{y}) = \frac{\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\theta_1 d\theta_2 \ldots d\theta_{i-1} d\theta_{i+1} \ldots d\theta_k}{\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}.$$

Similarly, if the expected posterior value of some function $g(\boldsymbol{\theta})$ of interest is desired, this again requires multidimensional integrals of the form

$$E[g(\boldsymbol{\theta})] = \frac{\int g(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}{\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}.$$

In general, such integrals will be difficult to compute directly and will have to be approximated using techniques such as numerical integration, analytic approximation and Monte Carlo integration (see Smith (1991) for a summary and comparison of these techniques).

	N	5	10	15
K				
5		49	149	299
10		74	199	374
20		124	299	524
50		274	599	974

Table 4.1: Total number of HMM parameters under various specifications of N and K

In the HMM parameter estimation problem, we are required to make inference about the posterior distribution $p(H|\mathbf{y})$. This will typically involve a very large number of parameters, e.g. for an ergodic model, with each state-dependent observation distribution completely specified by K parameters, there are $N-1$ parameters defining $\boldsymbol{\pi}$, $N(N-1)$ parameters defining A and NK parameters defining B. Table 4.1 illustrates the total number of parameters required to be estimated under an ergodic model for various values of N and K. Even for small values of the latter, the total number of model parameters to be estimated is large and, in general, there will be too many for the standard integral approximation techniques mentioned above to cope with (as these tend only to perform well in low dimensions, up to 10 say). Not only that but, as shall be seen in Section 5.2, the form of the likelihood as a sum over a very large state sequence space means that, in general, it is not possible to write down an explicit expression for the form of the posterior, $p(H|\mathbf{y})$.

In fact, most Bayesian problems of interest will involve many parameters for which marginal inference will require integrating a complex distribution over, possibly, hundreds of dimensions. The arrival of the iterative simulation techniques of Markov chain Monte Carlo (MCMC) methods on the statistical scene has, therefore, been of great importance in this respect, offering the chance to analyse simple and highly complicated processes alike. The basic idea involves simulating a Markov chain, $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots)$, which tends, in the long run, to settle into a steady state of equilibrium such that $\mathbf{x}^{(n)}$, for n large enough, is distributed as the target distribution about which it is desired to make inference (e.g. the posterior distribution, in a Bayesian analysis). The simulated output chain constitutes a dependent sample that is approximately distributed according to the target distribution and which may then be used as a basis for estimating

target distribution summaries of interest.

These powerful computational techniques will be employed in Chapter 5 to re-address the problem of HMM parameter inference from a fully Bayesian perspective. The remainder of this chapter is devoted to both theoretical and practical issues concerning MCMC simulation methods.

4.3 Markov chain Monte Carlo stochastic simulation techniques

4.3.1 Introduction

The appearance of the paper by Geman and Geman (1984) in the image processing literature, introducing the Gibbs sampler as a tool for simulation in a statistical context, has encouraged a flood of further work in the area of MCMC samplers, the general class of which offers the chance to simulate from complicated multivariate distributions. Although these methods are not new and have been employed for many years in the physics literature, only recently has their potential been recognised within the field of statistics. In particular, they provide a powerful computational tool when adopting a Bayesian approach (Gelfand & Smith, 1990), where the need for evaluating highly complicated, multidimensional integrals has previously caused problems in practice.

We now provide a brief introduction to some basic Markov chain theory which will be useful when discussing MCMC techniques. Recall from Section 1.2.1 that a first-order discrete-time Markov chain, defined on some target space $\mathcal{X}$, is driven by a mechanism such that the state occupied by the process at time t, denoted $\mathbf{x}^{(t)}$, is governed only by the state $\mathbf{x}^{(t-1)}$ occupied at the previous timepoint $t-1$, for $t = 1, 2, \ldots$. In the context of modelling the unobserved HMM state process, the matrix A describes the transition mechanism driving the Markov chain state process over the discrete state space $\{1, 2, \ldots, N\}$. We extend our definition here to include Markov chains defined on continuous target state spaces. Thus, consider a Markov chain defined on the target space $\mathcal{X}$ and driven by the transition function $P : \mathcal{X} \times \mathcal{X} \rightarrow [0, 1]$, where $P(\mathbf{x}, \mathbf{x}')$ governs

the value of $\mathbf{x}^{(t+1)} = \mathbf{x}'$ given $\mathbf{x}^{(t)} = \mathbf{x}$, $\forall t$; i.e.

$$\begin{cases} p(\mathbf{x}^{(t+1)} = \mathbf{x}' | \mathbf{x}^{(t)} = \mathbf{x}) = P(\mathbf{x}, \mathbf{x}') & \text{, for discrete } \mathcal{X}, \\ p(\mathbf{x}^{(t+1)} \in \mathcal{A} | \mathbf{x}^{(t)} = \mathbf{x}) = \int_{\mathbf{x}' \in \mathcal{A}} P(\mathbf{x}, \mathbf{x}')\, d\mathbf{x}', \quad \mathcal{A} \subseteq \mathcal{X} & \text{, for continuous } \mathcal{X}, \end{cases}$$

and

$$\begin{cases} \sum_{\mathbf{x}' \in \mathcal{X}} P(\mathbf{x}, \mathbf{x}') = 1 & \text{, if } \mathcal{X} \text{ is discrete,} \\ \int_{\mathbf{x}' \in \mathcal{X}} P(\mathbf{x}, \mathbf{x}')\, d\mathbf{x}' = 1 & \text{, if } \mathcal{X} \text{ is continuous.} \end{cases}$$

We now introduce two properties which may be exhibited by a Markov chain. A Markov chain is said to be irreducible if $\forall \mathbf{x}' \in \mathcal{X}$ and $\forall \mathcal{A} \subseteq \mathcal{X}$ such that $\pi(\mathbf{x} \in \mathcal{A}) > 0$, then $p(\mathbf{x}^{(t)} \in \mathcal{A} | \mathbf{x}^{(0)} = \mathbf{x}') > 0$ for some $t \geq 1$. This effectively means that, wherever the Markov chain is started, it has positive probability of visiting any region $\mathcal{A}$ of the target space and, thus, can be thought of as a "connectedness" property. Furthermore, a Markov chain is said to be aperiodic if there does not exist a measurable partition of $\mathcal{X} = \{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_{r-1}\}$, for some $r \geq 2$, such that $p(\mathbf{x}^{(t)} \in \mathcal{A}_{t \bmod r} | \mathbf{x}^{(0)} \in \mathcal{A}_0) = 1$ for all t, where $t \bmod r$ denotes the value of t modulo r. Essentially, aperiodicity implies that there are no regions of the target state space $\mathcal{X}$ which can only be visited at certain regularly-spaced times, a sufficient condition for this being that $p(\mathbf{x}^{(t)} = \mathbf{x}^{(t-1)}) > 0$ for some t. We may also introduce a stationary (or invariant) distribution $\pi(\cdot)$ for the Markov chain (not to be confused with the initial state vector $\boldsymbol{\pi}$ of the HMM) as some distribution satisfying

$$\begin{cases} \pi(\mathbf{x}') = \sum_{\mathbf{x} \in \mathcal{X}} \pi(\mathbf{x}) P(\mathbf{x}, \mathbf{x}') & \text{, if } \mathcal{X} \text{ is discrete,} \\ \pi(\mathcal{A}) = \int_{\mathbf{x} \in \mathcal{X}} p(\mathbf{x}^{(1)} \in \mathcal{A} | \mathbf{x}^{(0)} = \mathbf{x})\, \pi(\mathbf{x})\, d\mathbf{x}, \quad \mathcal{A} \subseteq \mathcal{X} & \text{, if } \mathcal{X} \text{ is continuous,} \end{cases}$$

where $p(\mathbf{x}^{(1)} \in \mathcal{A} | \mathbf{x}^{(0)} = \mathbf{x}) = \int_{\mathbf{x}' \in \mathcal{A}} P(\mathbf{x}, \mathbf{x}')\, d\mathbf{x}'$. The stationary distribution π is an equilibrium distribution for the Markov chain process if, for almost all $\mathbf{x} \in \mathcal{X}$ and sets $\mathcal{A} \subseteq \mathcal{X}$,

$$\lim_{t \to \infty} p(\mathbf{x}^{(t)} \in \mathcal{A} | \mathbf{x}^{(0)} = \mathbf{x}) = \pi(\mathcal{A}).$$

Theorem 1 of Tierney (1991) provides the result that if a Markov chain is irreducible, aperiodic and has a stationary distribution π, then π is the unique stationary distribution and is also the equilibrium distribution of the chain. In simple terms, if such conditions are satisfied, then the Markov chain process passes through an initial transient phase, eventually settling into a steady state of equilibrium, where the distribution of $\mathbf{x}^{(t)}$,

given any starting point of the chain, is the stationary distribution $\pi(\cdot)$ for all times t occurring after the process has reached equilibrium.

Against this brief background to Markov chain theory, we may now formulate the ideas behind MCMC simulation techniques as follows.

Suppose we wish to obtain a sample from a target joint probability distribution with density function denoted by $\pi(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}$, the target space $\mathcal{X}$ denotes the set $\{\mathbf{x} : \pi(\mathbf{x}) > 0\}$ and we adopt the common notation of $\pi(\cdot)$ to denote all probabilites and densities associated with the particular target distribution of interest. The idea behind a MCMC sampling scheme is to construct a suitable transition function defining a Markov chain which has π as its stationary (and equilibrium) distribution. Given some starting value for the chain, $\mathbf{x}^{(0)}$, these methods iteratively simulate $\mathbf{x}^{(t)}$ based on the current state of the chain $\mathbf{x}^{(t-1)}$ via the transition function $P(\mathbf{x}^{(t-1)}, \mathbf{x}^{(t)})$, for $t = 1, 2, \ldots, n$, to obtain a realisation of the underlying Markov chain $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)})$ which, after a run of sufficient length, may be thought of as a dependent sample drawn, approximately, from the target distribution. Currently, the two most widely used members of this broad class of MCMC schemes are the Gibbs sampler (Geman & Geman, 1984) and the Hastings algorithm (Hastings, 1970):

The Gibbs sampler

Suppose $\mathbf{x}$ can be decomposed into k components as $(x_1, x_2, \ldots, x_k)$, where the x_i's may be multivariate. Then, introducing the notation

$$\mathbf{x}_{(i)} = (x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k)$$

to denote the vector $\mathbf{x}$ excluding its i^{th} component, the Gibbs sampler may be employed if all full conditional distributions of the form $\pi(x_i|\mathbf{x}_{(i)})$ are "available", in the sense that they can be readily sampled from using standard simulation techniques.

Given the current state of the chain $\mathbf{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \ldots, x_k^{(t)})$ after t iterations, $\mathbf{x}^{(t+1)}$ is generated at the next iteration, component by component from the appropriate full conditional distributions, in the following way:

$$\begin{aligned} x_1^{(t+1)} &\sim \pi(x_1|x_2^{(t)}, x_3^{(t)}, \ldots, x_k^{(t)}) \\ x_2^{(t+1)} &\sim \pi(x_2|x_1^{(t+1)}, x_3^{(t)}, x_4^{(t)}, \ldots, x_k^{(t)}) \end{aligned}$$

$$\vdots$$

$$x_k^{(t+1)} \sim \pi(x_k | x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{k-1}^{(t+1)}).$$

So, $x_1^{(t+1)}$ is updated from $x_1^{(t)}$ by sampling from the full conditional of x_1 given the current values of the remaining x_j's; $x_2^{(t+1)}$ is then sampled from its full conditional given the updated value of x_1 and the current values of the other x_j's, etc.

The Hastings algorithm

Let q denote the Hastings candidate generating function (CGF), $q : \mathcal{X} \times \mathcal{X} \to [0, 1]$, where $q(\mathbf{x}, \mathbf{x}')$ defines the distribution (density function or discrete mass function) of the proposal for the next state $\mathbf{x}'$ in the Markov chain, given the current state $\mathbf{x}$. So

$$\begin{cases} \int_{\mathbf{x}'} q(\mathbf{x}, \mathbf{x}')\, d\mathbf{x}' = 1 & \text{, if } q \text{ is continuous,} \\ \sum_{\mathbf{x}'} q(\mathbf{x}, \mathbf{x}') = 1 & \text{, if } q \text{ is discrete.} \end{cases}$$

The acceptance probability α is then defined by

$$\alpha(\mathbf{x}, \mathbf{x}') = \begin{cases} \min\left\{ \dfrac{\pi(\mathbf{x}')\, q(\mathbf{x}', \mathbf{x})}{\pi(\mathbf{x})\, q(\mathbf{x}, \mathbf{x}')}, 1 \right\} & \text{, if } \pi(\mathbf{x})\, q(\mathbf{x}, \mathbf{x}') > 0 \\ 1 & \text{, if } \pi(\mathbf{x})\, q(\mathbf{x}, \mathbf{x}') = 0. \end{cases}$$

The $(t+1)^{th}$ iteration of the Hastings algorithm proceeds as follows:

1. If the Markov chain is currently at $\mathbf{x}^{(t)} = \mathbf{x}$, then generate a candidate point $\mathbf{x}'$ according to $q(\mathbf{x}, \mathbf{x}')$.

2. Accept $\mathbf{x}'$ with probability $\alpha(\mathbf{x}, \mathbf{x}')$, setting $\mathbf{x}^{(t+1)} = \mathbf{x}'$; otherwise, set $\mathbf{x}^{(t+1)} = \mathbf{x}$.

The Hastings algorithm is the generalisation of an original algorithm introduced by Metropolis et al (1953) and reduces to this original Metropolis algorithm when q is a symmetric function such that $q(\mathbf{x}, \mathbf{x}') = q(\mathbf{x}', \mathbf{x})$.

Under mild regularity conditions, the following results hold for both the aforementioned sampling schemes:

Convergence Theorem $\mathbf{x}^{(t)} \xrightarrow{d} \mathbf{x} \sim \pi(\mathbf{x})$ as $t \to \infty$

Ergodic Theorem For a function $f(\mathbf{x})$ whose expectation, $E_\pi[f(\mathbf{x})]$, exists with respect to the stationary distribution $\pi(\mathbf{x})$,

$$\lim_{t\to\infty} \frac{1}{t}\sum_{i=1}^{t} f(\mathbf{x}^{(i)}) \xrightarrow{a.s.} E_\pi[f(\mathbf{x})],$$

where $\xrightarrow{d}$ denotes convergence in distribution and $\xrightarrow{a.s.}$ denotes convergence almost surely. Roberts & Smith (1994) provide proofs and establish some mild, and easily checked, conditions which are sufficient to ensure the Convergence Theorem holds. In particular, for the Hastings algorithm, this reduces to checking that the CGF q is

1. aperiodic,
2. irreducible,
3. such that $q(\mathbf{x},\mathbf{x}') = 0 \iff q(\mathbf{x}',\mathbf{x}) = 0$.

With the emergence of these simulation techniques on the statistical scene, there have been countless papers written illustrating their usefulness in many different contexts. Smith & Roberts (1993), Besag & Green (1993) and Gilks et al (1993), plus the discussion from the RSS meeting at which they were presented, provide an extensive review of current applications, including those in spatial statistics, medical applications and incomplete data problems. We look now in further detail at both of the above sampling schemes and how they may be implemented.

4.3.2 Practical implementation of MCMC procedures

Practical implementation of the Gibbs sampler can be achieved provided it is possible to decompose the target distribution into a set of full conditional distributions which are easily simulated from. The functional form of these conditionals, up to proportionality, may easily be determined from the full target distribution by noting that

$$\begin{aligned}\pi(x_i|x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_k) &= \frac{\pi(x_1,x_2,\ldots,x_k)}{\pi(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_k)} \\ &\propto \pi(x_1,x_2,\ldots,x_k).\end{aligned}$$

Thus, the full conditional distribution for x_i is simply proportional to the full target distribution considered as a function of x_i alone. Once the form of each full conditional distribution is established, the remaining task is to specify techniques by which to simulate from them.

The area of univariate stochastic simulation is well-documented, e.g. Ripley (1986), Devroye (1986) and Morgan (1984). For standard distributions (e.g. Normal, Beta, Gamma), tailored simulation algorithms have been devised which efficiently simulate from the desired distribution. In addition, there are a host of more general-purpose techniques which may be employed (e.g. cdf (cumulative distribution function) inversion, rejection sampling and ratio-of-uniforms), many of which only require the target distribution to be specified up to proportionality. The recent emergence of the Gibbs sampler as a statistical tool has itself prompted further developments in the simulation literature, including a generalisation of the ratio-of-uniforms method (Wakefield et al, 1991) and an efficient adaptive-rejection scheme for log-concave densities (Gilks & Wild, 1992). The variety and general applicability of such techniques admits simulation from just about any univariate distribution of interest. As for simulating from distributions of more than one variable, again techniques have been devised for standard multivariate distributions (e.g. multivariate Normal, Dirichlet). If no other suitable simulation technique is available for sampling from a particular conditional distribution, then the Hastings algorithm may be employed if an appropriate CGF (candidate generating function) can be specified, a technique which has been adopted by Stephens & Smith (1993) to sample from a discrete conditional distribution defined on a large discrete space. Thus, implementation of the Gibbs sampler, if the target distribution admits decomposition into suitable full conditionals, basically requires some experience with and knowledge of stochastic simulation techniques.

The format of the Hastings algorithm is more general in the sense that the Gibbs sampler can be thought of as a special case; i.e. the Gibbs sampler is simply a cycle of k Hastings steps, one for each of the full conditional distributions (as noted by Gelman & Rubin (1993)). The major task in its implementation is the specification of a suitable CGF, $q(\mathbf{x}, \mathbf{x}')$. In general, there will be a huge choice and it will be necessary to identify some subclass which has "good" properties for the class of target distributions under study. For example, it is usually desirable to choose $q(\mathbf{x}, \mathbf{x}')$ to have some standard parametric form to facilitate easy and efficient simulation of candidate points. Complete specification of the CGF then reduces to choosing particular values for the density parameters. The issue of designing MCMC samplers is discussed in greater depth in Section 4.5 and, in particular, the choice of Hastings CGF. As recognised by Tierney (1991), Hastings algorithm steps and/or Gibbs sampling steps may be combined in

cycles or mixtures and the Hastings algorithm may be used within the Gibbs sampler to generate from "nasty" conditional distributions or to reduce computation time (the Hastings algorithm is often more computationally efficient than direct generation from conditional distributions).

The output of a MCMC sampler is an approximate, dependent sample from the target distribution $\pi(\cdot)$. The sample is approximately a sample from $\pi(\cdot)$ since, if $\pi^{(t)}(\mathbf{x})$ denotes the distribution after t iterations then, by the Convergence Theorem, $\pi^{(t)}$ tends to π as $t \to \infty$; so, for finite t, $\pi^{(t)}$ will only be an approximation to π, albeit a, hopefully, very close approximation. The sample is dependent since, by the very nature of its generation, it is the realization of a first-order Markov chain, where the value of each state is governed by the value of the previous state. The availability of an output sample admits analysis similar to the usual methods of inference applied to data samples (e.g. point estimation with accuracy estimates, quantile estimation, density estimation). For example, if it is desired to make marginal inferences about a single component of $\mathbf{x}$, x_i say, then this is achieved simply be analysing the i^{th} component of each sample point, $(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)})$.

The flexibility of analysis that such an output sample provides (i.e. the ability of a single sample from the target distribution to answer multiple questions of interest, from simple moment estimation to approximation of entire marginal densities, without the need for re-analysis) is one of the main attractions of MCMC sampling methods. Another is the wide applicability of such schemes and their ability to tackle complicated and highly multidimensional Bayesian problems which have, until now, remained intractable. Not only this, but concessions such as adopting unrealistic models or priors, in order to ensure simplified calculations, are no longer necessary. For example, sampling from a k-dimensional target distribution via the Gibbs sampler may be reduced to sampling from k univariate conditional distributions at each iteration, a fairly simple task given the multitude of univariate simulation techniques available. For these reasons, the strategies of MCMC simulation have offered a real breakthrough in the field of practical application of Bayesian methods. However, once the MCMC sample $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)})$ has been generated, the question of how best to utilise it for inference remains one for which there is, as yet, no definitive answer. The fact that the sample is dependent and only approximately drawn from the target distribution complicates the analysis. The subject of output analysis for MCMC samples is one of ongoing research and ranges

from completely ad hoc methods to those which attempt to introduce some theoretical formality.

4.4 Output analysis

There are two properties of an output sample from a MCMC simulation which make its analysis more complicated than that of the standard independent and identically distributed data samples usually dealt with in statistics. By the Convergence Theorem which governs the theory of MCMC simulation techniques, given some initial distribution $\pi^{(0)}(\mathbf{x})$ from which the initial state $\mathbf{x}^{(0)}$ of the Markov chain is generated, the realised Markov chain passes through an initial transient period before "converging" to its stationary distribution π. Of course, the speed at which such convergence takes place is governed, in part, by the choice of initial distribution, $\pi^{(0)}(\mathbf{x})$; the closer this is to $\pi(\mathbf{x})$, the more rapidly convergence will occur (in the extreme case of $\pi^{(0)}(\mathbf{x}) \equiv \pi(\mathbf{x})$, convergence is instantaneous). Thus, convergence diagnostics are required to identify the point at which $\pi^{(t)}(\mathbf{x})$ is "close enough" to $\pi(\mathbf{x})$ to treat the chain as if convergence has occurred. The second property exhibited is dependence of the sample which affects estimation procedures.

This now raises one of the main points of contention in output analysis; whether to base inference on the sample generated by a single long run of the MCMC sampler or to obtain multiple samples via independently replicated shorter runs from different starting points (see Gelman & Rubin (1992) and Geyer (1992) plus the accompanying discussion of these papers). The main argument for the latter is that it may identify when convergence has not occurred; i.e. diagnosis of convergence at iteration m, say, on the basis of a single chain of length n may often give a false sense of security in the sense that diagnosis based on multiple chains may show convergence to occur much later than iteration m. This phenomenon most often occurs if the MCMC sampler "gets stuck" in restricted regions of the target space for long periods of time, giving the appearance that stationarity of the process has been reached; e.g. in the case of a multimodal target distribution, certain MCMC samplers may remain in the neighbourhood of a single mode for many iterations before visiting another mode. This type of behaviour may be identified if multiple chains are used, initialised by starting points which are well-dispersed about the target space; e.g. for the multimodal example, if multiple chains are

initialised by points in the neighbourhood of different modes. Exponents of replication put forward this argument as sufficient endorsement for its use. However, although multiple runs may identify non-convergence, the converse argument does not hold; i.e. if convergence is diagnosed on the basis of multiple chains then it is not necessarily the case that convergence has actually occurred — how many replications are "enough" to ensure valid output inference? In the multimodal example, initialising the multiple runs with points which lie in the neighbourhood of only a strict subset of the full set of modes will have the same problems as a single chain (and we note again the importance of initial values). This is the basis of the argument used for the "single long chain" philosophy, that if convergence cannot be guaranteed by the use of multiple runs, then one may as well base inference on a single long chain which has a greater chance of visiting all regions of the target space carrying non-negligible probability. Also, it is usual to discard an initial part of the output chain when utilising it for estimation purposes, in order to reduce the bias introduced by the transient period. Exponents of the single chain approach point out that this makes the multiple run approach less efficient and more wasteful.

It seems that, if a single long run will not suffice, then this surely indicates an inadequacy in the design of the sampler rather than a need to adopt multiple runs which will not solve the problem, only identify that there is one! The only way to increase confidence in the inference obtained from any output chain is to address the factors which might cause problems by performing some sort of preliminary analysis of the target distribution and designing sampling schemes which perform well in light of this knowledge. A further reason for concentrating our attentions away from the "single chain" versus "multiple chain" debate and onto designing better sampling strategies is a further factor of slow convergence — high dependence between successive chain iterates. This latter property is, again, governed by how well-suited the design of the sampler is to generate from the particular form of target distribution under consideration. If there is too much dependence within chains, then the issue of using a single chain or multiple runs is irrelevant, as the output will generally be untrustworthy and unsuitable for drawing inferences. The design of good, target-distribution-guided MCMC samplers will be discussed in Section 4.5 and it will be seen that there is a place for both multiple runs, to explore the target distribution in pilot analyses and test the adequacy of proposed sampling schemes, and a single long chain, to actually carry out the final analysis once

a reliable sampling strategy to suit the target distribution has been devised.

One of the most ad hoc and yet widely used convergence diagnostics is the monitoring of ergodic averages. Suppose it is desired to draw inference about a function, $f(\mathbf{x})$. Then the ergodic average of f, after t iterations, is defined to be

$$\bar{f}_t(\mathbf{x}) = \frac{1}{t}\sum_{i=1}^{t} f(\mathbf{x}^{(i)}) \tag{4.2}$$

and, by the Ergodic Theorem, this can be considered as an estimate of $E_\pi[f(\mathbf{x})]$ with respect to $\pi(\mathbf{x})$, based on the iterations up to t. Graphs may then be produced, plotting $\bar{f}_t(\mathbf{x})$ against t, and convergence is considered to have occurred once the average settles down and assumes some constant value for an adequate length of time (although this by no means indicates that the full distribution has converged and is only valid when the relevant expectation, $E_\pi[f(\mathbf{x})]$, exists). Of course, the phrase "an adequate length of time" is one which opens itself to ad hoc interpretation. How long is adequate? If the sampler remains in one restricted region of the target space for too long (either through poor sampler design or "badly-behaved" target distribution or both), then the ergodic average will tend to level out, falsely giving the appearance of stationarity. However, to be fair, most of the diagnostics to be discussed, which are used in practice, exhibit similar inadequacies in the face of such unforeseen problems. Monitoring ergodic averages from multiple runs may identify the existence of such problems and indicate that a modification of sampler design may be required. It should be mentioned that when monitoring ergodic averages, functions do not necessarily have means which stabilise at the same iteration as the original model variables and, thus, one should monitor only those functions which are ultimately of interest. Similarly, the ergodic averages for some variables of the model will take longer to stabilise than others.

Once convergence has been diagnosed, from iteration m say, the output sample may be used to estimate posterior quantities of interest. For instance, the ergodic average, $\bar{f}_n(\mathbf{x})$ (defined in (4.2)) of some function $f(\mathbf{x})$ may be used as a point estimate of $E_\pi[f(\mathbf{x})]$ with respect to the target distribution $\pi(\mathbf{x})$. The Ergodic Theorem ensures that, in the long run, this estimate tends to the correct expectation. Thus, any posterior quantity which can be expressed in the form of an expectation may be estimated in this way, e.g. distribution moments, cumulative distribution probabilities.

If ergodic averages are to be used for estimation, then a measure of their accuracy must be evaluated in the form of a variance estimate, say. Although the Ergodic Theorem

tells us that the entire chain may be utilised for expectation estimates, since the bias caused by the initial transient phase will have negligible effect for sufficiently long runs, most procedures for estimating the precision of such estimates require the simulated Markov chain to have achieved stationarity. Thus, it is usual to discard the initial m transient iterations before proceeding with estimation.

We are, therefore, left with a stationary (or as good as stationary) and dependent series of points. Now, the variance of an ergodic average for the function $f(\mathbf{x})$ is defined by

$$\text{Var}\left[\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}^{(i)})\right] = \frac{1}{n^2}\sum_{i=1}^{n}\text{Var}\left[f(\mathbf{x}^{(i)})\right] + \frac{2}{n^2}\sum_{j=1}^{n}\sum_{i=1}^{j-1}\text{Cov}\left[f(\mathbf{x}^{(i)}),\, f(\mathbf{x}^{(j)})\right], \qquad (4.3)$$

where $\text{Cov}[\cdot,\cdot]$ denotes covariance. Since the process is assumed to have attained stationarity, all variances are assumed equal to the target variance $\sigma_f^2 = \text{Var}_\pi[f(\mathbf{x})]$ of $f(\mathbf{x})$ with respect to $\pi(\mathbf{x})$ and the covariances are assumed stationary, so that

$$\text{Cov}\left[f(\mathbf{x}^{(i)}),\, f(\mathbf{x}^{(j)})\right] = \text{Cov}\left[f(\mathbf{x}^{(0)}),\, f(\mathbf{x}^{(j-i)})\right] = \gamma_{j-i},\ \text{say;}$$

i.e. the covariances are a function of the lags only and $\gamma_0 = \sigma_f^2$. One approach which is often adopted is to try and obtain an approximately uncorrelated sample by trimming the series of iterations, thus allowing computation of the variance of the ergodic estimator via an estimate of σ_f^2. The autocorrelation at lag t is defined by γ_t/γ_0 and it effectively represents the amount of dependence which exists between points in the output sample lying t iterates apart. Thus, if estimates of the autocorrelations can be obtained from the output chain, then the time-lag at which autocorrelations first become negligible, l say, may be identified. If a subsample of the original chain (minus the m transient iterations) is formed by taking every l^{th} iterate, then this resultant trimmed sample should be approximately uncorrelated. Thus, if μ_f and σ_f^2 are the mean and variance, respectively, of some function $f(\mathbf{x})$ and if $n^* = \frac{n-m}{l}$, then a run of n iterations will yield the estimates

$$\begin{aligned}
\hat{\mu}_f &= \frac{1}{n^*}\sum_{i=1}^{n^*} f(\mathbf{x}^{(m+il)}) \\
\hat{\sigma}_f^2 &= \frac{1}{n^*}\sum_{i=1}^{n^*}(f(\mathbf{x}^{(m+il)}) - \hat{\mu}_f)^2 \\
\text{Var}(\hat{\mu}_f) &\simeq \frac{\hat{\sigma}_f^2}{n^*},
\end{aligned}$$

where $\hat{\mu}_f$ and $\hat{\sigma}_f^2$ are the estimates of μ_f and σ_f^2, respectively, based on the trimmed sample and $\text{Var}(\hat{\mu}_f)$ can be taken as a measure of the accuracy of the average estimate.

If multiple runs are used, r being the number of replications, then it is sometimes suggested that only the final iterate of each run be recorded so that a truly independent sample of size r may be obtained on which to base precision estimates. However, it seems difficult to justify wasting most of the iterations generated simply to ensure an independent sample at the end.

This method is very ad hoc and there is a need to provide a more formal framework in which to conduct output analysis, an area of ongoing research. Ripley (1986), Bratley et al (1983) and Schmeiser (1989) each devote a section to the subject of output analysis for a general process considered to be in equilibrium or tending to equilibrium asymptotically. They discuss procedures by which to estimate σ_f^2 above, incorporating the dependence of the chain and, thus, attempting to avoid wasteful discarding of iterations; e.g. batch means, spectral analysis methods, autoregressive modelling.

The above methods are presented in the general context of a stochastic process tending asymptotically to a steady state. However, since the emergence of MCMC sampling methods in Bayesian analysis, efforts have been made to provide more formal procedures in output analysis for this particular case of steady-state simulation. Various theoretical results, based on Markov chain theory, have been obtained pertaining to convergence rates. For example, Frigessi et al (1993) investigate the convergence rates for Markov random fields and Ingrassia (1991) derives bounds on the rate of convergence for a Metropolis algorithm. These and other papers form part of the ongoing research into the area of Markov chain theory which may help, in the future, to provide a complete formal framework in which to implement MCMC sampling strategies.

At present, however, there is a growing literature which attempts to use such theoretical ideas to provide a more formal basis on which to implement practical MCMC analyses. Whitt (1989) addresses the ongoing debate of a single long chain versus multiple runs in terms of efficiency, a quantity related to the variance of the ergodic average estimator. He draws the conclusion, as one would expect for such a complex problem, that there is no simple answer and that different sampling strategies may be optimal, depending on both the properties of the sampling design and those of the target distribution; e.g. the convergence rate of the sampler and its correlation structure once in steady state. Fishman (1991) addresses the problem of choosing the "warm-up" interval (i.e. the number of initial transient iterations to be discarded) and the steady-state sample

size in order to attain a required degree of accuracy in the ergodic average estimates for expected quantites of interest. This is discussed in the context of a Markov chain taking values in a discrete, finite set and is based on minimising computational cost. Again, he reaches the conclusion that optimality of sampling strategy is a function of both the target distribution and the properties of the MCMC design employed. Glynn & Heidelberger (1992) also discuss, on a more theoretical plane, the beneficial effect of inital transient deletions on the resultant bias of steady-state summaries of interest, such as Bayesian confidence intervals. These, and other, results attempt to perform comparisons of general strategies in a general context and tend to conclude that there is no optimal strategy — it all depends on the form of target distribution and the properties of the chosen MCMC sampler when applied to this distribution.

However, there still remains a gulf between the theory and techniques which can be applied in practice. Consequently, there have been attempts to advance the area of practical MCMC output analysis of realised chains whilst incorporating some approximate theoretical ideas. We outline here a selection of the various analysis procedures which have been proposed.

Gelman & Rubin (1992) have designed a convergence diagnostic based on replicated chains initiated from a widely dispersed starting distribution. Approximating with Normal theory, their technique takes into account the fact that, at each iteration, we sample from an approximation to the target distribution and seek to identify the point at which this difference proves insignificant. One disadvantage is its requirement for constructing an initial distribution which is over-dispersed with respect to the target distribution, necessary for the ensuing theory to be valid, which assumes some quite detailed prior knowledge of the target distribution.

Raftery & Lewis (1992) provide a sampling plan (i.e. "warm-up" period, sample spacing and chain length) for estimating the cumulative distribution probabilities of univariate posterior quantiles to a specified accuracy, based on a preliminary run of the MCMC sampler. Its success is dependent upon the adequacy of the preliminary chain as a general guide to the sampler properties and can run into problems if the target space is not adequately covered. Wakefield (1993) highlights this sensitivity with respect to alternative model parameterisations.

Menzefricke (1991) conducts analysis based on exact calculations for the special

case of a multivariate Normal distribution, illustrating how the speed of convergence of mean and covariance estimates to their true values decreases with increasing model correlations and increasing distribution dimension, properties which we may expect to apply in general. A similar analysis is performed for the simple bivariate Normal case in Hills & Smith (1992).

Ritter & Tanner (1992) introduce the idea of the "Gibbs stopper", a technique which monitors the distribution of a set of weights evaluated over independent replications at each iteration of a MCMC run. Each weight represents the ratio of values of the target density and an estimate of the current sampling density evaluated at the current sample point. As the current sampling density from which iterates are drawn tends to the target density, the distribution of weights should tend to a constant spike. This method may prove computationally cumbersome as many replications will be required to provide useful estimates of the current sampling density.

Mykland et al (1992) propose methods for designing chains with regeneration times; i.e. points in the MCMC chain at which future values of the process are independent of past values. Standard regenerative methods for analysing the output may then be employed.

Jennison (1992) also suggests sampling from several "heated" target distributions, at various temperatures, and comparing output ergodic averages. If $\pi(\mathbf{x})$ is the target distribution, then $\pi^{\frac{1}{\beta}}(\mathbf{x})$, $\beta > 1$, is referred to as the heated target distribution with temperature β. As β increases, so the heated distribution tends to flatten out, thus reducing the effect of multimodality and allowing freer movement of the sampler about the target space. Therefore, we may sample from several heated target distributions, with temperatures greater than 1, and reweight the outputs using importance sampling ideas to provide approximate samples from the desired "unheated" distribution. This allows comparison of ergodic averages obtained from the different "heated" output samples with those from the "unheated" output sample in order to check agreement.

4.5 The design of MCMC samplers

The remainder of this chapter is devoted to the subject of designing MCMC samplers with the target distribution in mind, in order to improve their performance properties.

This subject will be given particular emphasis when re-analysing the HMM from a Bayesian viewpoint in Chapter 5.

The design of a MCMC sampling strategy, with respect to some class of target distributions under consideration, is of great importance in order to achieve reliable inference from the output chain. Unfortunately, no general-purpose MCMC sampler is optimal for every problem or target distribution of interest and it is naive to think that an arbitrary sampling scheme may be directly applied to any target distribution and automatically produce valid results; at the very least, there is a conflict in designing chains with both optimal rates of convergence and from which the most efficient parameter estimation is possible (see Section 2 of Besag & Green (1993)). The way forward now is to utilise specific knowledge of the particular class of distributions and problems which is of interest to the user in order to design chains which perform well and are robust within this specified class.

On one level, we have available general-purpose simulation techniques such as the Gibbs sampler and Hastings algorithm described earlier, which offer the chance to sample from virtually any target distribution π. On a higher level, we now want to incorporate problem-specific information in chain designs to produce a more tailored approach with a view to improving sampler performance. So, what properties, in general, are desirable in a well-designed class of MCMC samplers?

4.5.1 Desirable properties of a MCMC scheme

Simplicity

The basic concepts of both the Gibbs sampler and the Hastings algorithm are quite straightforward to the average statistician and, in their general format, require little effort to program. In some ways, they are fairly automatic, although a little extra thought is required to decide how best to generate from the full conditionals when using the Gibbs sampler (requiring knowledge of stochastic simulation techniques) and to specify a CGF when using the Hastings algorithm. Too much fine tailoring to suit individual distributions which destroys this simplicity and over-complicates computer coding is neither desirable nor necessary and, when dealing with univariate distributions in particular, will invariably not yield such improvement in performance as to justify

the extra effort and skill involved. If computer software is ever to be designed with a non-expert in mind, a degree of automation must be retained in the methodology and implementation, so the design of problem-dependent strategies should not stray too much from the original, simple format of the chosen MCMC samplers.

Rapid mixing and adequate target space coverage

The property of rapid mixing relates to the existence of low dependence between successive iterates of the chain and linked with this is the property of greater, and more rapid, coverage of the target space under consideration. High autocorrelations within individual chains (which indicates slow movement around the target space) increases variance estimates of ergodic averages (see Equation (4.3)) so that even if a chain has "converged" to the target distribution, its output may be of no practical use if autocorrelations are so high that parameter estimates, based on ergodic averages, are very imprecise within a fixed run length. This, in turn, has caused users to resort to subsampling, i.e. discarding large parts of individual chains to try and obtain approximately uncorrelated samples, and also to use multiple runs and average over final independent iterates in order to give parameter estimates for which precision estimates may be easily calculated using standard independent sampling theory. These approaches all seem rather wasteful and the way forward is to try and design efficient chains which individually mix rapidly and have the potential to visit all regions of the target space carrying non-negligible probability with the correct frequency in a finite run (i.e. in real time).

One particular related issue that has received a lot of attention is how to deal with problems of multimodality, both within univariate conditional distributions and in much higher dimensions in complicated systems. Problems occur since moving between two modes of the target distribution, say, may necessitate the chain passing through regions of low probability; this will be a lengthy task for many of the commonly used sampling strategies which make local transitions about the current state of the chain. If the sampling strategy does not take account of multimodality when it exists or does not adequately visit all regions of the target space of non-negligible probability, then any inference based on the output from such a chain will be misleading. If the target distribution is multimodal, say, and the chain never visits some of the modes in a finite run, then the inadequacy of the chain design will probably not be evident from the output of

the sampler which will appear to have come from a distribution confined to the region of the target space visited, so leading to invalid inference about the underlying target distribution. For example, the output sample may appear to come from a unimodal distribution when the true distribution is multimodal.

Exploitation of problem-dependent information

Incorporation of relevant prior information is central to the Bayesian approach and any relevant information obtainable about the class of target distributions of interest should be put to good use when designing sampling strategies. This idea is implicit to the selection of initial distributions and of CGFs for the Hastings algorithm — poor choices of these will often prove detrimental to the performance of the sampling strategy. For example, specification of a Hastings CGF which is too "tight" for the target distribution will hinder free movement about the target space and slow mixing, whereas one which has a spread much greater than the target distribution will incur very low acceptance rates for candidate points in the chain. Rarely are we in the position of knowing nothing about the model of interest and, in any case, a preliminary analysis is always advisable to attempt to locate regions of the target space carrying non-negligible density, identify features such as multimodality with the potential to cause difficulties in sampling, etc, and so guide our choice of sampler design, e.g. Hastings CGFs and initial distributions.

Recognition of ultimate function of the chain output

The design of the sampling strategy should always take account of the objectives of the analysis and of how the output is to be used. For example, chains which are to be used in searching large spaces for maxima and minima using simulated annealing techniques (Aarts & Korst, 1989) may require different design properties to those which are employed for purposes of obtaining an uncorrelated sample from the target distribution to be stored for repeated future use. Also, in the case of parameter estimation, ergodic averages for different parameters in a particular chain may not stabilise at the same point or exhibit similar autocorrelations. Thus, our design should focus on improving the performance of the chain with respect to the parameters of particular interest, or functions thereof.

Confidence of user in chain output

The user should have confidence that the design of the MCMC sampler will provide an output sample from which valid inference may be made about the target distribution and that the performance of the designed sampler is not sensitive to varying specifications within the class of distributions of interest. The sampling strategy should be able to deal with properties of the target distribution which might cause difficulties, such as multimodality or high dependence between certain variables of the distribution which are notorious for causing restricted movement of certain sampling strategies about the target space. A preliminary analysis would certainly help to identify any potentially problem-causing properties of the class of target distributions. Of course, the Ergodic and Convergence Theorems are based on asymptotics (i.e. infinite runs), so the user will never be 100% certain that a finite-run chain has converged or has visited all regions of the target space of non-negligible probability. However, good sampler design will help to increase confidence that chains have a good chance of adequate target space coverage and achieving approximate convergence in a finite run.

Efficiency

Useful sampler output should be obtainable in real time, where constraints on computation time will be determined by the user in the context of the particular class of problems of interest. Such constraints will be determined by issues such as whether the analysis is on- or off-line, one-off or repeatable, whether the strategy admits use of parallel processing techniques, etc. Also, a little extra thought in designing the sampling strategy could save time and effort in the long run, since a poorly chosen sampler may yield output from which no useful inference can be made, thus requiring a complete re-analysis.

A combination of good properties

Many of the above aims are interlinked, e.g. rapidly mixing chains require less computation time to achieve parameter estimates of a specified precision and cover the target space more quickly, thus increasing user confidence in the output. Therefore, designing chains with at least some of these properties should be possible in many problems. The

emphasis, therefore, is on approaching the design of the sampling strategy with some thought, as opposed to simply selecting the most obvious or straightforward scheme or, worse still, a purely arbitrary scheme.

There has been much recent interest in methods of dealing with properties of target distributions which cause slow convergence and high autocorrelations within output chains, e.g. multimodality and high dependence between model variables. The following section documents some current suggestions for dealing with such problems.

4.5.2 Review of current suggestions in MCMC sampler design

The heated target distribution

Much mention has been made recently of the potential uses of the "powered-down" target distribution, $\pi^{\frac{1}{\beta}}, \beta > 1$. It was discussed briefly in Section 4.4 in the context of convergence diagnostics but may also offer the possibility of overcoming multimodality difficulties in some situations, since replacing the desired target distribution with its β^{th} root serves to flatten the distribution and reduce the effects of any modes.

The heated distribution has been successfully used by Lin et al (1993) as a Hastings CGF, where their particular target distribution is a highly multimodal surface. So, instead of generating variables directly from the full conditional distributions within the Gibbs sampler, i.e.

$$x_i \sim \pi(x_i|\mathbf{x}_{(i)}), \qquad i = 1, 2, \ldots, k,$$

the Hastings algorithm is used to sample from each of the full conditionals, with a heated conditional distribution as the CGF. Thus, if the current point is $\mathbf{x}$ and the i^{th} site, x_i, is to be updated, then the candidate point $\mathbf{x}'$ is generated through $q(\mathbf{x}, \mathbf{x}')$ as

$$q(\mathbf{x}, \mathbf{x}') \propto \begin{cases} \pi(x_i'|\mathbf{x}_{(i)})^{\frac{1}{\beta}}, & \text{if } x_j' = x_j,\ j \neq i \\ 0, & \text{otherwise.} \end{cases}$$

For the case when $\mathbf{x}' = (x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_k)$, the acceptance probability, $\alpha(\mathbf{x}, \mathbf{x}')$, is given by

$$\alpha(\mathbf{x}, \mathbf{x}') = \min\left\{1, \frac{\pi(x_i'|\mathbf{x}_{(i)})^{1-\frac{1}{\beta}}}{\pi(x_i|\mathbf{x}_{(i)})^{1-\frac{1}{\beta}}}\right\}.$$

Now, since α decreases with increasing β for $\pi(x_i'|\mathbf{x}_{(i)}) < \pi(x_i|\mathbf{x}_{(i)})$, this would suggest that the acceptance probability is low for large β in this case. However, Lin et al (1993) claim improved performance over other sampling schemes in their highly multimodal application.

In particular, for discrete $\pi(\cdot)$, suppose that at the updating stage for the i^{th} site, $i \in \{1, 2, \ldots, k\}$, the candidate point $\mathbf{x}' = (x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_k)$ has low density relative to the current point $\mathbf{x} = (x_1, x_2, \ldots, x_k)$. Then, we may compute the probability of passing from the current point to the proposed point as

$$\begin{cases} \pi(x_i'|\mathbf{x}_{(i)}), & \text{, with the Gibbs sampler,} \\ \min\left\{\pi(x_i'|\mathbf{x}_{(i)})^{\frac{1}{\beta}}, \dfrac{\pi(x_i'|\mathbf{x}_{(i)})}{\pi(x_i|\mathbf{x}_{(i)})^{1-\frac{1}{\beta}}}\right\} & \text{, with the heated Hastings algorithm.} \end{cases}$$

Now, since $\pi(\cdot|\cdot) \in [0, 1]$ for discrete distributions, then

$$\pi(x_i'|\mathbf{x}_{(i)})^{\frac{1}{\beta}} \geq \pi(x_i'|\mathbf{x}_{(i)}),\ \beta > 1$$

and

$$\frac{\pi(x_i'|\mathbf{x}_{(i)})}{\pi(x_i|\mathbf{x}_{(i)})^{1-\frac{1}{\beta}}} \geq \pi(x_i'|\mathbf{x}_{(i)}).$$

Thus, certainly in the case of a discrete distribution, the heated Hastings algorithm seems to increase the chances of the chain moving to a much less probable point, which eases movement between local modes where this may require passing through areas of comparatively low probability.

Lin et al (1993) also consider simulations using a scheme with the heated version of the desired distribution as the stationary target distribution. Reweighting the output sample, using importance sampling ideas (see, for example, Section 5.2 in Ripley (1986)), provides an approximate sample from the desired distribution. Thus, if it is desired to sample from the "correct" target distribution $\pi(\cdot)$ and we have available a sample $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)})$ distributed according to the "wrong" distribution, $\pi^*(\cdot)$, then, using the fact that

$$E_\pi[\mathbf{x}] = E_{\pi^*}\left[\frac{\pi(\mathbf{x})}{\pi^*(\mathbf{x})}\mathbf{x}\right],$$

we may, for example, estimate the expected value of $\mathbf{x}$ with respect to π, $E_\pi[\mathbf{x}]$, by the ergodic average

$$\frac{1}{n}\sum_{t=1}^{n}\frac{\pi(\mathbf{x}^{(t)})}{\pi^*(\mathbf{x}^{(t)})}\mathbf{x}^{(t)}.$$

The objective, easing movement about the target space, remains the same and this idea of sampling from the "wrong" distribution and reweighting, when suitable MCMC sampling schemes cannot be found for the "correct" distribution of interest, has also been proposed elsewhere (see Green (1992), which appears in the discussion of a paper by Geyer & Thompson, plus the authors' replies to his comments). As with importance sampling, though, the more dissimilar the two distributions, π and π^*, the more unstable the resulting ergodic average estimates.

Because heated distributions are more dispersed than the original target distribution, they may also be useful for specifying initial distributions in the Gelman & Rubin method discussed in Section 4.4. If an approximation of the target distribution π can be constructed, then "heating" it will provide a distribution which is over-dispersed with respect to π. Alternatively, short pilot runs may be conducted with $\pi^{\frac{1}{\beta}}$, $\beta > 1$, as the target distribution to obtain an approximate sample from the over-dispersed distribution.

Exploitation of symmetries

Exploitation of any symmetries (or approximate symmetries) existing in the target distribution is another method for speeding up coverage of the target space and reducing autocorrelations. At specific time intervals of the MCMC sampler chain, a Hastings algorithm step may be performed to generate a candidate point, which is the symmetrical counterpart of the current point, by reflection with respect to the appropriate symmetry. If the target distribution is exactly symmetric, this candidate point will be accepted with probability 1; otherwise, if it is only approximately symmetric, then the acceptance probability should still be high, although not 1.

This idea has been used to good effect in sampling from many physical systems, where symmetries frequently occur. For example, the Ising model is a lattice of binary sites, where each site can be either black or white. The model is defined by a small number of parameters, the value of one of which determines whether the distribution is unimodal or bimodal. In the case of a bimodal distribution, symmetric switching is equivalent to jumping between the neighbourhoods of the 2 modes, where the symmetrical counterpart of a particular lattice state is obtained by changing all white sites in the lattice to black and vice versa (see Section 6 of Geyer (1991) for further details). When symmetrical

properties exist this idea can work well, although this is rarely the case when dealing with statistical applications.

Hybrid MCMC schemes

As stated previously, hybrid chains may be devised using mixtures of Hastings CGFs or cycles of Gibbs sampling steps and/or Hastings steps. For example, let $q_1, q_2, \ldots, q_m$ be Hastings CGFs defining Markov chains with stationary distribution π. If at least one is aperiodic and irreducible, then q given by the mixture

$$q = \varepsilon_1 q_1 + \varepsilon_2 q_2 + \ldots + \varepsilon_m q_m, \qquad \sum_i \varepsilon_i = 1, \;\; \varepsilon_i > 0, \; i = 1, 2, \ldots, m,$$

is also an aperiodic, irreducible Hastings CGF defining a Markov chain with stationary distribution π. It may be possible to construct combinations of suitable "local" and "global" CGFs which perform better than any one component CGF alone (e.g. mix more rapidly) and which offer the chance to make larger jumps around the target space at the possible expense of reduced Hastings candidate-point acceptance rates. Local CGFs (proposing candidate points in some neighbourhood of the current state) ensure high acceptance rates whilst more global CGFs (proposing candidate points from within a much larger area of the target space) safeguard against the event of getting stuck in certain regions of the target space. An example of a global CGF might be some appropriate choice of independence CGF of the form

$$q(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}'),$$

which permits candidate points to be proposed from any part of the entire target space, thus allowing movement away from the current modal neighbourhood.

Hybrid sampling algorithms also offer the chance to combine Hastings and Gibbs sampling steps in order to obtain a strategy with improved convergence properties. Now, a Markov chain $(\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots)$ is said to be uniformly ergodic if there exist constants $C > 0$ and $0 \leq \rho < 1$ such that, for any initial value $\mathbf{x}^{(0)}$,

$$\begin{cases} \int_{\mathbf{x}} |\pi^{(t)}(\mathbf{x}) - \pi(\mathbf{x})| \, d\mathbf{x} \;\leq\; C\rho^t & \text{, for continuous } \pi \\ \sum_{\mathbf{x}} |\pi^{(t)}(\mathbf{x}) - \pi(\mathbf{x})| \;\leq\; C\rho^t & \text{, for discrete } \pi, \end{cases}$$

and this implies that identical convergence properties hold, regardless of where the chain is started. It can be shown that if one of the component CGFs, q_i say, of a Hastings

mixture defines a uniformly ergodic Markov chain, then this "nice" convergence property of the chain is automatically transferred to the mixture. Now, if $f(\mathbf{x})$ is an independence CGF, a sufficient condition for uniform ergodicity is that the weight function

$$w(\mathbf{x}) = \frac{\pi(\mathbf{x})}{f(\mathbf{x})}$$

be bounded (see Corollary 4 of Tierney (1991)). Thus, the use of appropriate independence chains in mixtures can not only reduce autocorrelations within individual chains (since candidate points are generated independently of the current point), but also confer good convergence properties on mixtures in which they feature. See Tierney (1991) for a more detailed review of possible Hastings CGF choices.

Re-parameterisation of model variables

An important factor affecting the rate of mixing of a chain is the amount of dependence between component variables of the model. If the vector $(x_1, x_2, \ldots, x_k)$ is to be sampled using a MCMC sampling strategy, where each x_i (possibly multivariate) is to be updated separately from the rest, then a desirable property is that the x_i's be approximately uncorrelated. If x_1 and x_2 are highly dependent, say, small changes in x_1 will induce small changes in x_2, which in turn will induce small changes in x_1, etc; thus, single-site updating through the Gibbs sampler will cause x_1 and x_2 to move very slowly about their respective marginal target spaces. Section 3.2 of Hills & Smith (1992) provides an illustration of the effect of inter-variable correlation on convergence rates and within-chain dependence, in the special case of a bivariate Normal distribution. Therefore, if possible, a suitable re-parameterisation should be found, transforming the original set of variables to a new set which is approximately uncorrelated. Müller (1991) describes such an approximate orthogonalisation procedure (the "accelerated Gibbs sampler") which reduces model variable dependence. The re-parameterised model is then simulated via the Gibbs sampler, using the Hastings algorithm with Normal CGFs, to sample from full conditionals. Of course, such re-parameterisations cannot be applied in discrete situations and require the model variables to be defined on the entire real line (which may necessitate an initial transformation of the original variables to ensure this condition is satisfied).

Random-direction sampling

In the case of the Gibbs sampler, it is usual to sample in each of the x_i directions separately; so, if component variables are highly correlated, this could cause slow mixing of the chain. However, random-direction sampling (Schmeiser & Chen, 1991), where many parameters are updated simultaneously, may offer a more robust strategy for continuous distributions in this case.

For $\mathbf{x} = (x_1, x_2, \ldots, x_k)$, the standard Gibbs sampler updates cyclically in each of the k orthogonal directions of $\Re^k$, $\{(1,0,\ldots,0),\ (0,1,0,\ldots,0),\ \ldots,\ (0,\ldots,0,1)\}$. The idea behind random-direction sampling is to select k random directions in $\Re^k$, $\{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_k\}$, and cyclically sample in each of these directions, conditional on the current point. This may necessitate using the general form of the Hastings algorithm, since the format of generating from simple full conditionals is now lost. A variation on this theme, introduced by Phillips & Smith (1993), chooses only $\mathbf{d}_1$ randomly and the remaining $k-1$ directions are fixed to be orthogonal to it, a strategy which, they report, seems to improve on purely random directions.

When model variables are correlated, the directions in $\Re^k$ along which the target distribution varies most will not be the standard orthogonal co-ordinate directions. For example, in a bivariate situation with x_1 and x_2 highly correlated, the distribution varies most in directions parallel to the line $x_1 = x_2$. This accounts for the restricted movement associated with the single-site updating Gibbs sampler in this case. Sampling in alternative directions may, thus, offer the chance of more rapid mixing and it may even be possible to find "optimal" directions in specific examples, a task similar to identifying principal components (see Phillips & Smith (1993) for details).

Auxiliary variables

Besag & Green (1993) introduce another method which attempts to induce a suitable blocking system onto the set of model variables (in order to facilitate simultaneous updating), this being achieved by the introduction of further auxiliary variables into the model. These extra variables, $\mathbf{u}$, are to be chosen so that $\pi(\mathbf{x}|\mathbf{u})$ is some simple distribution (e.g. under which the x_i's are independent), thus facilitating simultaneous block updating of the component variables of $\mathbf{x}$. When a suitable $\mathbf{u}$ can be defined, this

method may perform extremely well; e.g. the Swendsen-Wang algorithm (Swendsen & Wang, 1987) which is applied in the physics literature to a generalisation of the Ising model described earlier in this section in "Exploitation of symmetries". However, there is no guarantee that suitable auxiliary variables exist for a particular model and, even if they do, it may require much ingenuity to find them. Careless choices of $\mathbf{u}$ may not only decrease the rate of mixing but will almost certainly increase computation time, due to the increased number of variables to be updated. Also, the auxiliary variables will often be quite abstract and of no other implicit interest, their inclusion in the sampling strategy being purely to speed the rate of mixing. This is unlike the situation where missing data are included in the updating schedule, in order to facilitate the use of Gibbs sampling (e.g. for the HMM, as we shall see in Chapter 5). Such "missing data"-type variables will often have some physical interpretation within the model under study and information gained about them is a useful by-product of the output.

Sweep strategies

Another issue to bear in mind when designing Gibbs sampling schemes is that of the choice of sweep strategy, which defines the order in which the k sites, $x_1, x_2, \ldots, x_k$, are to be updated. One sweep or cycle through the sites defines the sequence of updates which occurs between successive iterates of the Gibbs sampler (iterates being those timepoints at which the current value of $\mathbf{x}$ is recorded in the output sample). Common examples of sweep strategies include random updating, where k updates occur in each cycle at randomly chosen sites, and fixed-order cycles, where the k sites are updated in some pre-specified order. Any choice of strategy is valid provided it allows, in an infinite run, each site to be updated infinitely often. Study of such strategies (e.g. Amit & Grenander (1991) in the context of multivariate Normal distributions) reveals no evidence of a single, uniformly optimal scheme, again stressing the need to look at the particular structure of the class of problems of interest to guide this choice.

As mentioned earlier in this chapter, simultaneous block updating of variables is advisable whenever possible to avoid "sticking" in local neighbourhoods, since single-site updating in conjunction with the Gibbs sampler can often suffer slow convergence and high dependence induced within chains due to correlated model variables. So, for example, when sampling from a Bayesian posterior distribution, if the likelihood

admits the use of natural conjugate priors (i.e. ones which have the same functional form as the likelihood, yielding a posterior distribution of the same, usually simple, form as the prior; see Section 4.2 of Berger (1985) for further details), then the full conditionals of certain blocks of variables may have the form of simple and/or standard multivariate distributions. This may allow the simultaneous updating of many variables via simulation techniques for such standard distributions.

Coupled chains

Another idea to aid more rapid mixing of chains is a technique called Metropolis-coupling of Markov chains which was developed by Geyer (1991) and has been applied in Geyer & Thompson (1993). Suppose it is desired to sample from a distribution denoted $\pi^1(\cdot)$. Instead of running a single MCMC sampling strategy with stationary distribution $\pi^1(\cdot)$, m such processes are run in parallel, each tending to a different stationary distribution, $\pi^i(\cdot)$, $i = 1, 2, \ldots, m$, respectively. After each full scan of the variables within each chain, a swap is proposed between the states of two of the chains, and this swap is accepted according to the appropriate Hastings acceptance probability in order to maintain the correct stationary distribution for both chains. The idea behind this strategy is that some of the constituent chains will mix faster than others and that by coupling the initial chain, associated with the desired target distribution $\pi^1(\cdot)$, with chains which individually mix more rapidly, the coupled "target" chain may mix faster than if it was uncoupled.

The following example is used to illustrate how suitable choices of coupling chains may be made. Suppose it is desired to sample from a target distribution taken from some class of distributions, $\{\pi(\mathbf{x}|\gamma) : \gamma \in \Re\}$, indexed by a parameter γ, where the sampling scheme is to be the Gibbs sampler. Furthermore, suppose that the value of γ determines how correlated the components of $\mathbf{x}$ are; the larger the value of γ, the larger the correlations. As already mentioned, Gibbs sampler chains mix slowly when correlations between component variables are high. Thus, chains for which γ is low will mix more rapidly than those for which γ is high. Therefore, if the desired target distribution $\pi^1(\cdot)$ is specified by a large value for γ, couple it with chains for which the target distribution, $\pi^i(\cdot)$, $i \neq 1$, is specified by a smaller γ; i.e. specify each of $\pi^i(\cdot)$, for $i = 2, 3, \ldots, m$, by an increasingly smaller value of γ than the previous one. Then, using the Gibbs sampler,

we may couple chains which have γ values that are close so that swaps between these chains will have a high probability of acceptance.

Individual chains are no longer Markov and the question arises as to whether this affects our ability to analyse the output, any techniques employed which depend on the output being a Markov chain now being useless. Also, no mention is made in Geyer's paper about what to do with the remaining $m-1$ output chains — do we simply discard them or can we reweight them, using importance sampling ideas, to obtain a sample from the desired target distribution? Otherwise, the added computational burden may not justify the improved mixing properties.

Also, it may not be obvious as to how to find rapidly mixing chains associated with the desired target distribution. One solution may be to appeal to the ideas of heated target distributions, since we would expect chains associated with distributions at higher temperatures to mix more rapidly than those at lower temperatures (Lin et al, 1993).

Use of problem-specific knowledge

It seems unadvisable to approach any statistical problem without first obtaining some insight into the process being studied. Thus, a preliminary analysis of the class of target distributions of interest is essential. This may be done theoretically, by inspection and interpretation of the functional form and structure of the target distributions, or empirically, using computationally cheap "mode-finding" algorithms or multiple pilot runs, from various starting points, of an initially chosen MCMC strategy (such as a heated strategy). This information can then be used to specify Hastings CGFs and initial distributions, possibly also employing further information provided by the data.

For example, when specifying Hastings CGFs, it is usual to choose from standard, well-understood distributions, e.g. uniform, Normal, Student's t, for which standard simulation techniques are available, and it is general practice to centre these CGFs at the current state of the Markov chain. However, the scale of the CGF must then be fixed (e.g. the variance) and we are advised (Tierney, 1991) to adopt a distribution with heavier tails than the target distribution. This assumes some prior knowledge of the underlying target distribution. Furthermore, when the Hastings algorithm is being used to sample from full conditional distributions within the Gibbs sampler, the scale of these

target distributions may well vary greatly between iterations. In this case, there will not exist a fixed value for the scale parameter of the Hastings CGF which is optimum at each iteration — if the scale parameter is fixed too large, the acceptance rate of the algorithm will be very low and if it is fixed too small, movement around the target space will be very slow, in both cases leading to high autocorrelations between successive iterates. One solution may simply be to run several pilot chains with different values for the scale parameter. A suitable choice may then be arrived at by trial-and-error based on empirical acceptance rates for the Hastings algorithm (where the acceptance rate measures the percentage of candidate points which are accepted over a typical run of the sampler). A reasonable acceptance rate might be one in the range 60–70% (Vounatsou & Smith, 1993). Müller (1991) suggests a similar idea using a multivariate Normal Hastings CGF with some initial estimate of the covariance matrix. During an initial run of the sampler, the covariance matrix is constantly updated based on the output so far and adjusted to maintain a reasonable acceptance rate. In this way, a suitable choice for the covariance matrix may be specified and fixed before implementing the final MCMC run from which the output sample is collected.

Another instance where specific information about the target distribution may be incorporated in the sampling strategy involves use of the Hastings algorithm. If it is computationally cheap to evaluate the target density function at particular points, then a modification of the griddy Gibbs sampler idea (Ritter & Tanner, 1992) may be used to specify a Hastings CGF in the form of an approximation to the target distribution (see Tierney (1991)). The CGF is constructed to propose a candidate in some neighbourhood of the current location $\mathbf{x}$ of the chain. For a fixed grid increment $\mathbf{h}$ and density f defined on the target space, the points at which π is evaluated are $\{\mathbf{x} \pm i\mathbf{h} : i = 0, 1, \ldots, m\}$, for some pre-specified integer m. The Hastings CGF is then defined to be

$$q(\mathbf{x}, \mathbf{x}') = \frac{\sum_{i=0}^{m} \pi(\mathbf{x} \pm i\mathbf{h}) f(\mathbf{x}' - \mathbf{x} \mp i\mathbf{h})}{\sum_{i=0}^{m} \pi(\mathbf{x} \pm i\mathbf{h})}.$$

In this way, structure from the target distribution may be incorporated in the CGF specification. The task of specifying $\mathbf{h}$, m and f again raises the same problems as that of specifying functional forms and associated scale parameters for any Hastings CGF. It should be noted that Ritter & Tanner suggested using the above CGF directly to sample from the target distribution, without use of the Hastings acceptance probability adjustment. However, since it is only an approximation to the true target density, this

will result in convergence to the wrong stationary distribution, if convergence occurs at all.

4.6 Conclusions

MCMC simulation techniques offer the powerful computational tools required to address the task of Bayesian inference for HMMs. In Section 4.5, we stressed the need for designing MCMC sampling strategies with regard to the class of target distributions of interest, in order to ensure output chains from which valid inference may be drawn.

In light of the issues raised here, particular emphasis will be placed in Chapter 5 on how suitable designs for the sampling scheme may be derived which give improved performance properties in the case of HMMs.

Chapter 5

Bayesian inference for hidden Markov models

5.1 Introduction

In light of the issues raised in the previous chapter, we look in closer detail at the particular case of interest to us — the HMM and how to draw inference about unknown model parameters. Classical solutions to the parameter estimation problem usually involve use of either Viterbi re-estimation or B-W re-estimation, both of which were discussed in Chapter 2. Even from a classical viewpoint, these procedures are not perfect since they only guarantee convergence to a local maximum in the neighbourhood of the starting value and neither procedure provides accuracy measures for the estimates obtained. Furthermore, B-W re-estimation is often abandoned (certainly in the area of speech recognition) for Viterbi re-estimation in practice, since the forward-backward equations and necessary scaling procedure of the former can often be computationally cumbersome for reasonable-sized problems.

As pointed out in Dempster et al (1977), the techniques of the EM algorithm (of which B-W re-estimation is the special case for HMMs) may be adapted in the Bayesian framework to obtain local maxima of the posterior distribution (i.e. local modes) instead of the likelihood, although similar problems exist with respect to accuracy measurements and the desire for a global maximum, quite apart from the fact that a posterior mode may not be the required Bayesian estimator. The MCMC methods described in the previous

chapter have the power to address the highly parametric HMM and, at the very least, can provide accuracy measures for parameter estimates. In addition, provided long enough chains are used, global (as opposed to local) quantities should be obtainable from a MCMC output chain. This is in contrast to the results provided by the B-W re-estimation procedure. Moreover, we have seen in the examples of Section 3.3 that there is often prior information available which may be incorporated as part of a Bayesian approach. For example, in the pharmaceutical application of Section 3.3.3, an expert provides approximate scores for the likely HMM state transitions which, in an estimation problem, could be used in the construction of a prior distribution.

It is worth mentioning here that the HMM (often referred to in the "hidden mixture model" literature as the hidden Markov chain, in order to distinguish it from its multi-dimensional generalisation, the Markov random field) defines a particular model in the more general class of finite mixture distributions. We recall the form of the data model (the data being equivalent to the observation sequence, in our case) as

$$\begin{aligned} p(\mathbf{y}|H) &= \sum_{\mathbf{s}\in\mathcal{S}} p(\mathbf{y},\mathbf{s}|H) \\ &= \sum_{i=1}^{S} p(\mathbf{s}^i|H)\, p(\mathbf{y}|\mathbf{s}^i,H), \end{aligned} \tag{5.1}$$

where the S components of $\mathcal{S}$ are labelled $\{\mathbf{s}^1,\mathbf{s}^2,\ldots,\mathbf{s}^S\}$. Expressing $p(\mathbf{y}|H)$ in the form of a finite sum over the state sequence space, as in (5.1), $p(\mathbf{s}^i|H)$ may be considered as the mixing probability of the i^{th} mixture component and $p(\mathbf{y}|\mathbf{s}^i,H)$ may be considered as modelling the data given that it belongs to the i^{th} mixture component. Titterington (1990) gives a review of issues and current methods concerning estimation problems in the general class of mixture models and illustrates the place occupied by HMMs in this setting. Qian & Titterington (1991) then look in greater detail at parameter estimation for hidden Markov models (in their most general sense) using hybrid schemes which incorporate both ML techniques and stochastic restoration methods. Robert (1990) points out that any model which can be expressed in the form of a mixture distribution (as for the HMM) will admit Bayesian treatment via the Gibbs sampler or data augmentation (another iterative simulation procedure — see Tanner & Wong (1987)). The analysis of finite mixture distributions is then studied in further detail from a Bayesian viewpoint in Diebolt & Robert (1994). The authors make empirical comparisons of Gibbs sampling and data augmentation with ML techniques, such as the EM algorithm.

5.2 Application of the Gibbs sampler algorithm

A Bayesian approach to analysis of the model parameter set H, given the observation sequence $\mathbf{y}$, requires specification of the posterior distribution $p(H|\mathbf{y})$ and extraction from this distribution of any posterior quantities of interest. Now,

$$p(H|\mathbf{y}) \propto p(\mathbf{y}|H)\, p(H),$$

where $p(H)$ denotes the prior distribution on the model parameters. The normalising constant, $p(\mathbf{y})$, is effectively unknown since its computation would involve integrating over the complete HMM parameter space and summing over the entire state sequence space. We recall that

$$\begin{aligned} p(\mathbf{y}|H) &= \sum_{\mathbf{s}} p(\mathbf{y}, \mathbf{s}|H) \\ &= \sum_{\mathbf{s}} \pi_{s_1} b_{s_1}(y_1) \prod_{t=1}^{T-1} a_{s_t s_{t+1}} b_{s_{t+1}}(y_{t+1}), \end{aligned} \tag{5.2}$$

i.e. the likelihood can only be explicitly expressed as a summation over all possible state sequences, $\mathbf{s}$. As in the classical approach, calculation of the sum in (5.2) above, at every iteration of the sampler, is unfeasible because of the typically large size of the state sequence space, $\mathcal{S}$ (see Tables 2.1 and 2.2). Therefore, in order to be able to sample from the posterior distribution of H, it is necessary to include the missing data (i.e. the state sequence information $\mathbf{s}$) in the model and sample instead from the joint distribution

$$\begin{aligned} p(\mathbf{s}, H|\mathbf{y}) &\propto p(\mathbf{y}|\mathbf{s}, H)\, p(\mathbf{s}|H)\, p(H) \\ &= \left\{ \pi_{s_1} b_{s_1}(y_1) \prod_{t=1}^{T-1} a_{s_t s_{t+1}} b_{s_{t+1}}(y_{t+1}) \right\} p(H), \end{aligned} \tag{5.3}$$

which is of a more tractable form. By the theory of MCMC samplers discussed in Chapter 4, if we obtain a sample from the joint posterior distribution of $p(\mathbf{s}, H|\mathbf{y})$, denoted $\{(\mathbf{s}^{(1)}, H^{(1)}), (\mathbf{s}^{(2)}, H^{(2)}), \ldots, (\mathbf{s}^{(n)}, H^{(n)})\}$, then the sample $\{H^{(1)}, H^{(2)}, \ldots, H^{(n)}\}$, obtained by selecting only the realisations of the model parameters H, represents an approximate sample from the marginal posterior distribution $p(H|\mathbf{y})$ which is, ultimately, the distribution we wish to sample from. Thus, incorporation of the state sequence information into the sampling scheme facilitates the obtaining of a sample from $p(H|\mathbf{y})$, whilst being of no real importance from an inference viewpoint (although, in some applications, it may be that this information could provide useful insight into the underlying marginal state process).

Now, let $\mathbf{a}_i$ denote the i^{th} row of the transition matrix A. If we assume a priori that π, A and B are mutually independent and, furthermore, that the rows of A are also mutually independent, then $p(\mathbf{s}, H|\mathbf{y})$ may be decomposed into full conditional distributions which facilitate implementation of the Gibbs sampler. Non-zero correlation between the rows of A may also be dealt with using a generalisation of the naturally conjugate Dirichlet priors which apply in the independent case (Martin, 1975), although we confine ourselves to the independent case here. On one level, we can represent the $(i+1)^{th}$ iteration of the Gibbs sampling scheme in 2 parts, with respect to the following 2 conditionals:

1. Generate $H^{(i+1)}$ according to $p(H|\mathbf{s}^{(i)}, \mathbf{y})$.

2. Generate $\mathbf{s}^{(i+1)}$ according to $p(\mathbf{s}|H^{(i+1)}, \mathbf{y})$.

Henceforth, implicit conditioning on $\mathbf{y}$ is assumed in all full conditional distributions. To implement Step 1 above, according to the Gibbs sampler, requires the further breakdown of $p(H|\mathbf{s}, \mathbf{y})$ into its component full conditional distributions, as follows:

$$\begin{aligned}
\pi^{(i+1)} &\sim p(\pi|A^{(i)}, B^{(i)}, \mathbf{s}^{(i)}) \\
&= p(\pi|\mathbf{s}^{(i)}), \qquad (5.4) \\
\mathbf{a}_j^{(i+1)} &\sim p(\mathbf{a}_j|\pi^{(i+1)}, \mathbf{a}_1^{(i+1)}, \ldots, \mathbf{a}_{j-1}^{(i+1)}, \mathbf{a}_{j+1}^{(i)}, \ldots, \mathbf{a}_N^{(i)}, B^{(i)}, \mathbf{s}^{(i)}), \quad j = 1, 2, \ldots, N \\
&= p(\mathbf{a}_j|\mathbf{s}^{(i)}), \quad j = 1, 2, \ldots, N, \qquad (5.5) \\
\mathbf{b}_j^{(i+1)} &\sim p(\mathbf{b}_j|\pi^{(i+1)}, A^{(i+1)}, \mathbf{b}_1^{(i+1)}, \ldots, \mathbf{b}_{j-1}^{(i+1)}, \mathbf{b}_{j+1}^{(i)}, \ldots, \mathbf{b}_N^{(i)}, \mathbf{s}^{(i)}), \quad j = 1, 2, \ldots, N \\
&= p(\mathbf{b}_j|\mathbf{s}^{(i)}), \quad j = 1, 2, \ldots, N, \qquad (5.6)
\end{aligned}$$

where, for completeness, we assume the state observation distributions to be discrete and to take values in the finite set $\{1, 2, \ldots, K\}$, so that $B = (b_{ij})$ is the $N \times K$ state observation matrix with j^{th} row denoted by $\mathbf{b}_j$. The generalisation to continuous observation distributions is straightforward and will be addressed later. The simplifications in the above conditionals arise from the prior independence assumptions. As mentioned in Section 4.3.2, the functional form of conditional distributions may be extracted from the full joint distribution $p(\mathbf{s}, H|\mathbf{y})$ given in (5.3). Thus, recalling the prior independence assumption which means that we can express $p(H)$ as

$$p(H) = p(\pi) \prod_{j=1}^{N} p(\mathbf{a}_j)\, p(\mathbf{b}_j),$$

the conditionals in (5.4), (5.5) and (5.6) above may be written as

$$\begin{aligned} p(\boldsymbol{\pi}|\mathbf{s}) &\propto p(\boldsymbol{\pi}) \prod_{i=1}^{N} \pi_i^{v_i} \\ p(\mathbf{a}_i|\mathbf{s}) &\propto p(\mathbf{a}_i) \prod_{j=1}^{N} a_{ij}^{w_{ij}}, \quad i = 1, 2, \ldots, N \\ p(\mathbf{b}_i|\mathbf{s}) &\propto p(\mathbf{b}_i) \prod_{j=1}^{K} b_{ij}^{z_{ij}}, \quad i = 1, 2, \ldots, N \end{aligned} \tag{5.7}$$

where, for $i = 1, 2, \ldots, N$,

$$\begin{aligned} v_i &= I_{\{s_1=i\}} \\ w_{ij} &= \sum_{t=1}^{T-1} I_{\{s_t=i, s_{t+1}=j\}}, \quad j = 1, 2, \ldots, N \\ z_{ij} &= \sum_{t=1}^{T} I_{\{s_t=i, y_t=j\}}, \quad j = 1, 2, \ldots, K \end{aligned}$$

and $I_{\{E\}}$ denotes the indicator function for the event E. So, for instance, w_{ij} indicates the number of transitions from state i to state j in the state sequence $\mathbf{s}$ and z_{ij} indicates the number of times observation symbol j is emitted from state i in $\mathbf{s}$.

Now, the Dirichlet distribution (a multivariate generalisation of the bivariate Beta distribution) with parameter $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_k)$, denoted as $\text{Dir}(\gamma_1, \gamma_2, \ldots, \gamma_k)$, has density function given by

$$p(\mathbf{x}) = B_k(\gamma) \prod_{j=1}^{k} x_j^{\gamma_j - 1}, \qquad \sum_i x_i = 1,$$

where

$$B_k(\gamma) = \frac{\Gamma\left(\sum_{i=1}^{k} \gamma_i\right)}{\prod_{i=1}^{k} \Gamma(\gamma_i)}$$

and $\Gamma(\omega)$ denotes the Gamma function defined by

$$\Gamma(\omega) = \int_0^\infty x^{\omega-1} e^{-x}\, dx, \quad \omega > 0.$$

Thus, closer inspection of the functional forms for each of the HMM parameter conditional distributions given above reveals the existence of conjugate priors in the form of Dirichlet distributions (Martin, 1975). So, for example, if $p(\boldsymbol{\pi}) = \text{Dir}(\gamma_1, \gamma_2, \ldots, \gamma_N)$, so that we assume a conjugate Dirichlet prior for $\boldsymbol{\pi}$, then

$$p(\boldsymbol{\pi}|\mathbf{s}) = \text{Dir}(\gamma_1 + v_1, \gamma_2 + v_2, \ldots, \gamma_N + v_N),$$

and similarly for the other conditionals; i.e. for each $i \in \{1, 2, \ldots, N\}$, if $p(\mathbf{a}_i) = \text{Dir}(\gamma_1^i, \gamma_2^i, \ldots, \gamma_N^i)$, then $p(\mathbf{a}_i|\mathbf{s}) = \text{Dir}(\gamma_1^i + w_{i1}, \gamma_2^i + w_{i2}, \ldots, \gamma_N^i + w_{iN})$ and if $p(\mathbf{b}_i) =$

$\text{Dir}(\delta_1^i, \delta_2^i, \ldots, \delta_K^i)$, then $p(\mathbf{b}_i|\mathbf{s}) = \text{Dir}(\delta_1^i + z_{i1}, \delta_2^i + z_{i2}, \ldots, \delta_K^i + z_{iK})$. Thus, adopting conjugate Dirichlet priors for the model parameters leads to simple standard forms for their full conditional distributions which will aid simulation (Theorem 4.1 of Section XI.4 in Devroye (1986) describes a simple method for simulating from a Dirichlet distribution in terms of Gamma variates).

In the case of more general observation distributions, the only stage of the Gibbs sampling scheme which alters is the generation of the observation parameters. If $b_i(\cdot|\boldsymbol{\theta}_i)$ denotes the observation distribution associated with state i, which is dependent on the parameter vector $\boldsymbol{\theta}_i = (\theta_{i1}, \theta_{i2}, \ldots, \theta_{iK})$ then, if we again assume independence between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$ for $i \neq j$, the set of conditional distributions given by (5.7) in the Gibbs sampler set-up is replaced by

$$p(\boldsymbol{\theta}_i|\mathbf{s}) \propto p(\boldsymbol{\theta}_i) \prod_{t=1}^{T} b_i^{z_{ti}}(y_t|\boldsymbol{\theta}_i), \qquad i = 1, 2, \ldots, N,$$

where

$$z_{ti} = I_{\{s_t = i\}}.$$

The form of $b_i(\cdot|\boldsymbol{\theta}_i)$ will often admit the use of conjugate priors, especially if standard observation distributions are employed. For example, if the observation distributions are univariate Normal with $\boldsymbol{\theta}_i = (\mu_i, \sigma_i^2)$, where μ_i and σ_i^2 are the mean and variance, respectively, then

$$b_i(y|\mu_i, \sigma_i^2) = \frac{1}{(2\pi\sigma_i^2)^{\frac{1}{2}}} \exp\left[-\frac{1}{2\sigma_i^2}(y - \mu_i)^2\right], \quad i = 1, 2, \ldots, N$$

and, assuming μ_i and σ_i^2 to be independent prior to observing the data, standard conjugate priors can be used for $p(\mu_i)$ and $p(\sigma_i^2)$ in the form of Normal and inverse Gamma distributions, respectively (see Chapter 10 of Cox & Hinkley (1974)). The posteriors of $p(\mu_i|\mathbf{s})$ and $p(\sigma_i^2|\mathbf{s})$ will also then be of Normal and inverse Gamma forms, respectively. Diebolt & Robert (1994) raise the interesting point that improper priors (i.e. ones which are not normalisable) on observation distribution parameters are not permissible. Assuming a Gibbs sampler analysis of the HMM, this characteristic may be illustrated by considering the value of $\mathbf{s}^{(i)}$ at iteration i. Then, for any state $j \in \{1, 2, \ldots, N\}$ not appearing in the current sequence $\mathbf{s}^{(i)}$, $p(\boldsymbol{\theta}_j|\mathbf{s}^{(i)}) = p(\boldsymbol{\theta}_j)$; i.e. the full conditional distribution of the observation parameter vector $\boldsymbol{\theta}_j$ at iteration $(i + 1)$ will simply be equal to its prior. Thus, if $p(\boldsymbol{\theta}_j)$ is improper, simulations from this full conditional, in the above eventuality, will be impossible. Care must, therefore, be taken when specifying

non-informative/ignorance priors for parameters, in particular for those from unbounded parameter spaces, as these may be represented by improper distributions (Section 3.3 of Berger (1985) details the construction of non-informative priors, which attempt to express prior ignorance of parameter values).

Stage 2 of the Gibbs sampling scheme involves the generation of $\mathbf{s}$ from $p(\mathbf{s}|H,\mathbf{y})$. Generating $\mathbf{s}$ directly from this distribution would involve evaluating S probabilities and generating from a discrete distribution of size S. However, the state sequence space $\mathcal{S}$ will typically be very large, making direct simulation unfeasible. The most obvious way to deal with this in the Gibbs sampler setting is to further decompose $p(\mathbf{s}|H,\mathbf{y})$ into T full conditionals, one for each component of the state sequence, and generate from these T univariate distributions instead. Thus, at the $(i+1)^{th}$ iteration,

$$\begin{aligned}
s_1^{(i+1)} &\sim p(s_1|s_2^{(i)}, s_3^{(i)}, \ldots, s_T^{(i)}, H^{(i+1)}) \\
s_2^{(i+1)} &\sim p(s_2|s_1^{(i+1)}, s_3^{(i)}, s_4^{(i)}, \ldots, s_T^{(i)}, H^{(i+1)}) \\
&\vdots \quad\quad \vdots \\
s_T^{(i+1)} &\sim p(s_T|s_1^{(i+1)}, s_2^{(i+1)}, \ldots, s_{T-1}^{(i+1)}, H^{(i+1)}),
\end{aligned}$$

where dependence on $\mathbf{y}$ is, again, suppressed in the notation. Each of the T full conditionals is represented by a discrete distribution over the N possible states and the form of these can be extracted from the full joint distribution, as before. For the initial state,

$$\begin{aligned}
p(s_1|s_2, s_3, \ldots, s_T, H) &= p(s_1|s_2, H) \\
&\propto \pi_{s_1}\, b_{s_1}(y_1)\, a_{s_1 s_2}, \qquad (5.8)
\end{aligned}$$

for $t = 2, 3, \ldots, T-1$,

$$\begin{aligned}
p(s_t|s_1, s_2, \ldots, s_{t-1}, s_{t+1}, s_{t+2}, \ldots, s_T, H) &= p(s_t|s_{t-1}, s_{t+1}, H) \\
&\propto a_{s_{t-1}s_t}\, b_{s_t}(y_t)\, a_{s_t s_{t+1}} \qquad (5.9)
\end{aligned}$$

and, for the final state,

$$\begin{aligned}
p(s_T|s_1, s_2, \ldots, s_{T-1}, H) &= p(s_T|s_{T-1}, H) \\
&\propto a_{s_{T-1}s_T}\, b_{s_T}(y_T), \qquad (5.10)
\end{aligned}$$

where simplifications in the full conditionals above can be made due to the first-order Markov property of the state sequences. Thus, (5.8) implies that at the $(i+1)^{th}$ iteration,

$s_1^{(i+1)}$ is generated according to the discrete probability distribution defined by

$$p(s_1^{(i+1)} = j | s_2^{(i)}, H^{(i+1)}) = \frac{\pi_j^{(i+1)} b_j^{(i+1)}(y_1) a_{j s_2^{(i)}}^{(i+1)}}{\sum_{k=1}^{N} \pi_k^{(i+1)} b_k^{(i+1)}(y_1) a_{k s_2^{(i)}}^{(i+1)}}, \quad j = 1, 2, \ldots, N$$

and similarly for the other components of the state sequence, by normalising (5.9) and (5.10).

The case of multiple observation sequences can be dealt with using an obvious extension of the single sequence case described above. Thus, suppose we have R independent observation sequences, $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_R\}$, of lengths $T_1, T_2, \ldots, T_R$, respectively, so that $\mathbf{y}_r = (y_{r,1}, y_{r,2}, \ldots, y_{r,T_r})$, $r = 1, 2, \ldots, R$, and $T = \sum_{r=1}^{R} T_r$. We let $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_R)$ denote the vector of observation sequences and, again, the aim is to make inference about H based on the posterior $p(H|\mathbf{y})$. If $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_R)$ denotes the vector of state sequences underlying the respective observation sequences in $\mathbf{y}$, then the independence of the $\mathbf{y}_r$'s allows both the conditional distribution $p(\mathbf{y}|\mathbf{s}, H)$ and $p(\mathbf{s}|H)$ to be expressed as products of the form

$$p(\mathbf{y}|\mathbf{s}, H) = \prod_{r=1}^{R} p(\mathbf{y}_r|\mathbf{s}, H) = \prod_{r=1}^{R} p(\mathbf{y}_r|\mathbf{s}_r, H)$$

and

$$p(\mathbf{s}|H) = \prod_{r=1}^{R} p(\mathbf{s}_r|H).$$

Thus, substituting these values in

$$p(\mathbf{s}, \mathbf{y}|H) \propto p(\mathbf{y}|\mathbf{s}, H)\, p(\mathbf{s}|H)\, p(H)$$

and expressing each component of the products in terms of π, A and B as for the single sequence case in (5.3), it can be seen that using independent conjugate Dirichlet priors for π and the rows of A again yields independent Dirichlet conditional distributions for these quantities; similarly for B in the case of discrete finite observation distributions. So, for example, if the prior for π is taken to be a Dirichlet distribution, $p(\pi) = \mathrm{Dir}(\gamma_1, \gamma_2, \ldots, \gamma_N)$, then the full conditional for π is also Dirichlet with $p(\pi|\mathbf{s}) = \mathrm{Dir}(\gamma_1 + v_1, \gamma_2 + v_2, \ldots, \gamma_N + v_N)$, where

$$v_i = \sum_{r=1}^{R} I_{\{s_{r,1}=i\}}, \quad i = 1, 2, \ldots, N,$$

i.e. v_i indicates the number of state sequences in the set $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_R\}$ which begin in state i. Conditionals for A and B may be dealt with similarly, as in the single

sequence case, so that for each $i \in \{1, 2, \ldots, N\}$, if $p(\mathbf{a}_i) = \text{Dir}(\gamma_1^i, \gamma_2^i, \ldots, \gamma_N^i)$, then $p(\mathbf{a}_i|\mathbf{s}) = \text{Dir}(\gamma_1^i + w_{i1}, \gamma_2^i + w_{i2}, \ldots, \gamma_N^i + w_{iN})$, where

$$w_{jk} = \sum_{r=1}^{R} \sum_{t=1}^{T-1} I_{\{s_{r,t}=j,\ s_{r,t+1}=k\}}, \qquad j = 1, 2, \ldots, N,\ k = 1, 2, \ldots, N,$$

and if $p(\mathbf{b}_i) = \text{Dir}(\delta_1^i, \delta_2^i, \ldots, \delta_K^i)$, then $p(\mathbf{b}_i|\mathbf{s}) = \text{Dir}(\delta_1^i + z_{i1}, \delta_2^i + z_{i2}, \ldots, \delta_K^i + z_{iK})$, where

$$z_{jk} = \sum_{r=1}^{R} \sum_{t=1}^{T} I_{\{s_{r,t}=j,\ y_{r,t}=k\}}, \qquad j = 1, 2, \ldots, N,\ k = 1, 2, \ldots, K.$$

This deals with the first Gibbs sampler stage of generating H from $p(H|\mathbf{s}, \mathbf{y})$. The second stage involves the generation of $\mathbf{s}$ from $p(\mathbf{s}|H, \mathbf{y})$ and, since the R component state sequences of $\mathbf{s}$ are independent, this can be achieved in R independent stages, each of which is identical to the single sequence case in terms of simulating T_r univariate full conditionals with respect to the particular sequence $\mathbf{s}_r$ being generated, as given by Equations (5.8)–(5.10).

Robert et al (1993) study the Gibbs sampler approach to estimating HMM parameters described in this section and prove some strong theoretical convergence results. These are based on a duality principle which transfers the convergence properties of the chain of state sequences $(\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \ldots)$ to the chain of model parameters $(H^{(1)}, H^{(2)}, \ldots)$. They prove that the distribution of $H^{(i)}$ converges uniformly ergodically to the stationary distribution $p(H|\mathbf{y})$ (recalling the uniformly ergodic convergence property defined in "Hybrid MCMC schemes" of Section 4.5.2) and similar convergence results for the first moment of the stationary distribution. In addition, they prove that the sequence $(H^{(1)}, H^{(2)}, \ldots)$ exhibits a geometric mixing rate so that, essentially, the within-chain autocorrelations decay exponentially.

5.3 Practical issues when sampling a HMM

5.3.1 Choice of initial values

One of the first issues to consider when applying the Gibbs sampling scheme outlined in Section 5.2 is how to specify initial values or an initial distribution. As noted by Rabiner et al (1985b), the performance of the EM algorithm, in terms of its rate of convergence and the optimality of the local maximum to which it converges, tends to be very sensitive to poor initial values for the observation parameters whilst being relatively

insensitive to initial values for π and A. Of course, although good starting values will speed convergence for the Gibbs sampler, the theory of MCMC methods guarantees convergence to the true target distribution, eventually; in this sense, their choice is not as significant as for the EM algorithm.

The guidelines given for specifying initial parameter values in the case of the EM algorithm may well be suitable for the Gibbs sampler when adopting vague priors or when information provided by the data overwhelms the prior information. Rabiner et al (1985a) suggest specifying discrete uniform distributions for π and the rows of A but basing initial values for the observation parameters on the observed data, $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_R)$. This is done using an iterative procedure which employs the Viterbi algorithm to find the optimal underlying state sequences on which to base the initial estimates (calculated as ratios of frequencies). In fact, given some rough initial estimate of the entire model, it may be worth implementing the computationally efficient Viterbi re-estimation procedure, described in Section 2.3.3, to obtain an improved initial estimate $H^{(0)}$ for the Gibbs sampler, thereby reducing the time to convergence (e.g. for the example considered later in Section 5.7, the computation time for 500 iterations of the Viterbi re-estimation procedure is of the order of 10 seconds on a Sun SPARC Workstation, making it a very efficient technique for arriving at improved initial values for the Gibbs sampler).

Once $H^{(0)}$ has been specified, the best initial state sequence, $\mathbf{s}^{(0)}$, given $H^{(0)}$ may be obtained via the Viterbi algorithm; i.e. we choose $\mathbf{s}^{(0)} = \max_{\mathbf{s}} p(\mathbf{s}|H^{(0)}, \mathbf{y})$.

5.3.2 Problems with the Gibbs sampler design

We now address problems which may arise when implementing the Gibbs sampling scheme of Section 5.2 in the case of HMMs, making reference to some of the points raised in Chapter 4 about aims for good sampler designs. By inspection of the structure of the HMM Gibbs sampler scheme proposed, with the adoption of the natural conjugate priors, it can be seen that π and the rows of A are generated independently from Dirichlet distributions and,if suitable independent conjugate priors are again adopted for the observation parameters, these may also be updated independently. Thus, the only dependence which exists within the updating scheme occurs, on one level, between $H^{(i)}$ and $\mathbf{s}^{(i)}$ and, on the second level, between the components of $\mathbf{s}^{(i)}$ due to its single-state updating strategy. The dependence which exists between values of H and $\mathbf{s}$ is

an implicit property of the model and the only way it could be removed would be to sample H and $\mathbf{s}$ simultaneously (which would effectively require us to be able to sample from the full target distribution directly!). However, it may be possible to address the problem of dependence arising from the single-state updating of the state sequence $\mathbf{s}$.

We recall that Stage 2 of the HMM Gibbs sampling scheme presented earlier comprises the generation of $\mathbf{s}^{(i)}$ from $p(\mathbf{s}|\mathbf{y}, H^{(i)})$. It has been proposed that simulation from $p(\mathbf{s}|\mathbf{y}, H)$ is achieved by generating each s_t in turn from its full conditional distribution (according to the theory of Gibbs sampling). By Equations (5.8), (5.9) and (5.10), we can see that the distribution for s_t is highly dependent on the values of s_{t-1} and s_{t+1} (due to the first-order Markov property of the state sequences).

For the L-R HMM in particular, the s_t's are highly correlated due to the constraints and this will occur for any set of constraints imposed on state transitions. This induces high autocorrelations between successive iterates of $\mathbf{s}$, within individual chains of the Gibbs sampler, when the single-site updating schedule of Equations (5.8)–(5.10) is used. This phenomenon can be illustrated by looking at a couple of sample state sequences.

For example, consider a L-R HMM with $N = 3$, $T = 6$, and suppose we are employing the Gibbs sampler through the T full conditional distributions, as outlined in Section 5.2. Given the current state sequence $\mathbf{s}$, it will not, in general, be possible to update all T states of $\mathbf{s}$ in one sweep of the states because of the L-R constraints on state sequences. Suppose the current state sequence within a particular Gibbs sampler cycle is $\mathbf{s} = (1\ \mathbf{1} \vdots \mathbf{2}\ 2\ \mathbf{2} \vdots \mathbf{3})$, say, where we have segmented the full sequence into subsequences, one for each of the $N = 3$ possible states. Then, if a single state is to be updated, either within a fixed cycle or at random, it is only possible to make a transition to a distinct state sequence if it is one of the sites in bold type that is to be updated; the remaining 2 sites (s_1 and s_4) are constrained by the value of their neighbours to stay unchanged. So, the current value of s_1 is 1 and inspection of the full conditional for s_1 reveals that

$$\begin{aligned} p(s_1 = i|s_2, s_3, \ldots, s_6, \mathbf{y}, H) &= p(s_1 = i|s_2 = 1, \mathbf{y}, H) \\ &= \begin{cases} 1 & \text{, if } i = 1 \\ 0 & \text{, otherwise} \end{cases} \end{aligned}$$

i.e. no change in the value of s_1 is possible if this is the site required to be updated under the adopted sweep strategy. If we assume the fixed forward-cycling sweep strategy of Section 5.2, then the problem is especially worsened if the current sequence happens to

be $\mathbf{s} = (1\ 1\ 1\ 1\ 1\ \mathbf{1})$. Then, in one complete sweep of the sites, forward-cycling offers the potential to alter the final state alone so that, if successive iterates are taken to be those state sequences obtained after successive sweeps, a transition can only be made to one of the following 3 state sequences after one iteration: (1 1 1 1 1 1), (1 1 1 1 1 2) or (1 1 1 1 1 3). Successive iterates will therefore be highly correlated and, consequently, movement about the state sequence space will be slow. Of course, in one full sweep of the sites, backward-cycling would offer the potential to alter many more states but would suffer similar problems with sequences such as $\mathbf{s} = (\mathbf{3}\ 3\ 3\ 3\ 3\ 3)$. This illustrates the importance of an informed choice of sweep strategy to suit the model of interest.

We now recall the idea of a neighbourhood system in the context of state sequences, which was introduced in Section 3.3.1; i.e. a neighbourhood of a particular state sequence is considered to be a set of sequences which differ from it in only a few sites. For example, sequences differing by exactly one state constitute 1-state neighbours, etc. Experience then shows that L-R constrained state sequence distributions often exhibit multimodality in terms of this neighbourhood system and/or restricted movement of sampler chains causing "sticking" within particular regions of the state sequence space. This is borne out in the following example.

Consider the L-R model with discrete observation distributions, which shall be named Model 1, given by the parameter set

$$\pi = \begin{pmatrix} 0.8 & 0.1 & 0.1 \end{pmatrix},$$

$$A = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0 & 0.8 & 0.2 \\ 0 & 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \\ 0.5 & 0.5 \end{pmatrix},$$

from which an observation sequence, $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$, was generated. Thus, $N = 3$, $T = 6$, $K = 2$ and $S = 28$. Because of the small size of the state sequence space, we may compute the full posterior distribution $p(\mathbf{s}|\mathbf{y}, H)$ via the generalised Viterbi algorithm (with $L = S = 28$). Table 5.1 displays the top 8 most probable state sequences, carrying nearly 60% of the total probability, and it can be seen that the distribution is fairly multimodal in terms of the widely differing structure of these sequences. Considering H as fixed and defined by Model 1, it is desired to sample from $p(\mathbf{s}|\mathbf{y}, H)$.

The forward-cycling Gibbs sampler strategy for simulating from $p(\mathbf{s}|\mathbf{y}, H)$, as out-

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$
1	3 3 3 3 3 3	0.1413
2	1 1 1 1 1 1	0.1126
3	1 3 3 3 3 3	0.0678
4	1 1 3 3 3 3	0.0665
5	1 1 1 1 1 2	0.0643
6	1 1 2 2 2 2	0.0627
7	1 1 1 1 2 2	0.0420
8	2 2 2 2 2 2	0.0410

Table 5.1: Most probable state sequences underlying $\mathbf{y}$ = (1 2 1 2 2 1) for Model 1

lined in Equations (5.8)–(5.10) of Section 5.2, was run for 50 iterations, initialised by the modal state sequence (3 3 3 3 3 3). Figure 5.1 plots the resulting histogram obtained from state sequence frequencies observed in the output sample, using the labelling scheme for state sequences set out in Appendix C.1. This is compared with the true distribution, $p(\mathbf{s}|\mathbf{y}, H)$, and with a similar histogram plotting empirical state sequence frequencies based on an independent sample of size 50 from the true distribution. It can be seen that 20 distinct state sequences ($\simeq$ 71.5% of the full target space) have been realised under independent sampling and with close to their correct frequencies, as can be seen by comparing the similar histogram "shapes" for both independent sampling and the true distribution. In contrast, the Gibbs sampler has visited only 13 distinct state sequences ($\simeq$ 46.5% of the full target space) in the run of 50 iterations, which indicates restricted coverage of the state sequence space. In addition, the second most probable state sequence under the true distribution, (1 1 1 1 1 1), has not been visited at all and the histogram exhibits disproportionately high frequencies for the 3 closely neighbouring sequences: (1 1 2 2 2 2), (1 2 2 2 2 2) and (2 2 2 2 2 2). This indicates that the Gibbs sampler has got "stuck" in this region of the state sequence space.

To summarise, when the Gibbs sampling scheme for the HMM was introduced, the most obvious sweep strategy for generating the state sequence $\mathbf{s}$ was described; i.e. forward-scanning, single-site updating. However, this is not necessarily the most effective strategy. The potential problems of slow movement about the state sequence space have already been highlighted, particularly for constrained models such as the

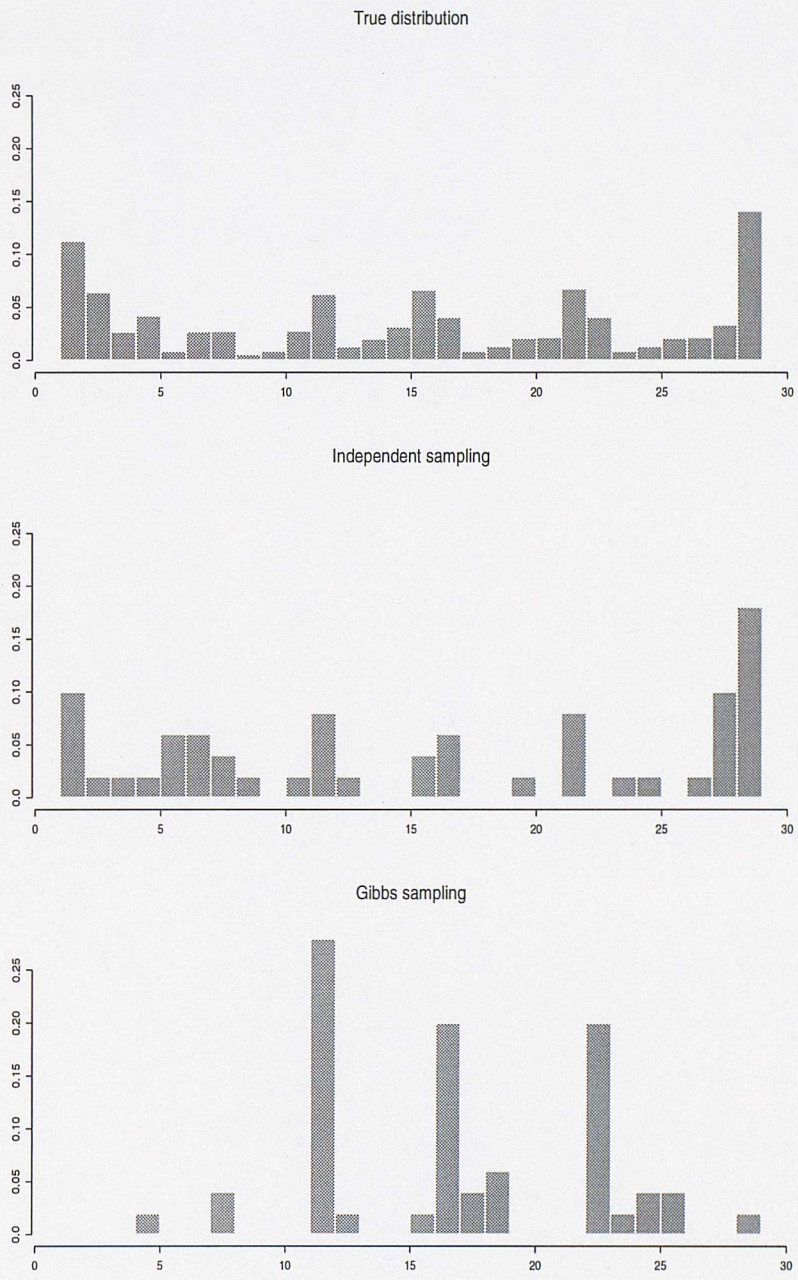


Figure 5.1: Comparison of histograms for the forward-cycling Gibbs sampler and independent sampling

L-R HMM. Because of the dependence which exists between the state sequences and the HMM parameters, restricted coverage of the state sequence space in the sequence updating steps will cause greater autocorrelations between successive parameter estimates. Improving the updating strategy of $\mathbf{s}$ should, thus, improve the properties of the output chain of ultimate interest, $(H^{(1)}, H^{(2)}, \ldots)$. We, therefore, wish to find an improved design for sampling from $p(\mathbf{s}|\mathbf{y}, H)$.

5.4 Analysis of sweep strategies

We discussed the issue of choice of sweep strategy in Section 4.5.2 and now analyse the relative merits of these choices in the context of generating from $p(\mathbf{s}|\mathbf{y}, H)$. In this case, $(s_1, s_2, \ldots, s_T)$ is the vector of T sites to be updated. We recall that the sweep strategy is the mechanism governing the order in which the sites are to be updated in any particular cycle comprising one iteration of the MCMC sampler. Particular deterministic choices include a forward-backward scan, in which one cycle consists of scanning forwards and then backwards through the sites or, alternatively, choosing equally between a forward and backward scan. Some common random update mechanisms, in which a scan consists of exactly T updates, include either choosing each of the T sites in a purely random order (with replacement) or by choosing a random permutation of the T sites (so that every site is updated once in each scan).

The issues of single-site updating versus block updating of many sites simultaneously also fall into the category of sweep strategy. Amit & Grenander (1991) compare deterministic and random sweep strategies in a Gaussian setting and conclude that neither strategy dominates the other and a mixture of both mechanisms may, in some cases, be preferable. They do, however, reach the firm conclusion (borne out in much of the current literature) that block updating is generally preferable to single-site updating, whenever possible.

In the following analysis, a fixed known HMM is considered in order to facilitate easy comparison of sweep strategies for the generation of state sequences. Several sweep strategies are suggested for comparison in some simple examples, each scan comprising of the updating of each of the T component states of $\mathbf{s}$ exactly once:

(a) Mixture of forward and backward scans: the scan is either forwards ($t = 1, 2, \ldots, T$) or backwards ($t = T, T-1, \ldots, 1$) through the states, each of these scans being chosen with equal probability $\frac{1}{2}$.

(b) Random permutation scan: the order in which the T states are updated is chosen randomly from the full set of $T!$ permutations. Because of the problems already discussed concerning the fact that, certainly for constrained models, only some of the states in a sequence have the potential to be changed, the completely random updating mechanism would probably be unsuitable for our model.

(c) Heated Hastings algorithm: in the forward-backward mixture of Strategy (a), instead of generating each component state from its full conditional distribution directly, a heated Hastings step, with temperature $\beta = 2$, is implemented. Thus, to update s_t, propose a candidate s'_t from $p(s'_t|s_{(t)}, \mathbf{y}, H)^{\frac{1}{2}}$ and accept or reject according to the Hastings acceptance probability.

(d) Block updating: the sequence is updated in blocks of 2 states at a time by generating exactly from the appropriate bivariate full conditional distributions, with the states being scanned according to a mixture of forward and backward cycles as in Strategy (a).

(e) Independent sampling: because of the small size of the state sequence space in this example, independent sampling from the true distribution $p(\mathbf{s}|\mathbf{y}, H)$ may be used as the standard by which the other strategies are judged.

The measure of comparison adopted, based on the spectral theory of Markov chains, involves inspection of the eigenvalues of the transition matrix governing each MCMC strategy. It is not usually possible to compute eigenvalues for models used in practice since they are typically high-dimensional and very complex. However, for the simple examples we will be considering, such a computation is feasible (e.g. finding the eigenvalues of a 28×28 matrix) and the conclusions will, hopefully, be indicative of performances for more realistic HMMs. We describe the theory in terms of our finite set of S state sequences, labelled $\mathcal{S} = \{\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^S\}$, where $\mathbf{s}^i = (s^i_1, s^i_2, \ldots, s^i_T)$ for $i = 1, 2, \ldots, S$. Let P be the transition matrix associated with the Markov chain $(\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \ldots)$ defined on $\mathcal{S}$, so that

$$P_{ij} = p(\mathbf{s}^{(t+1)} = \mathbf{s}^j | \mathbf{s}^{(t)} = \mathbf{s}^i), \qquad t = 0, 1, 2, \ldots$$

So, for example, if the Markov chain is generated through a Hastings algorithm with CGF $q(\cdot,\cdot)$ and acceptance probability $\alpha(\cdot,\cdot)$ then, for every $i,j = 1,2,\ldots,S$,

$$P_{ij} = q(\mathbf{s}^i,\mathbf{s}^j)\,\alpha(\mathbf{s}^i,\mathbf{s}^j), \quad i \neq j \tag{5.11}$$

$$P_{ii} = 1 - \sum_{j\neq i} P_{ij}. \tag{5.12}$$

Let $\lambda_1,\lambda_2,\ldots,\lambda_S$ denote the eigenvalues of P.

We now refer to a result discussed in Besag & Green (1993) and derived in greater detail in Frigessi et al (1992) which describes how the eigenvalues of P are related to both the rate of convergence of the MCMC sampler and its efficiency for estimation purposes. In general, for a finite state space of size V, consider a reversible (defined in the following paragraph), irreducible, aperiodic Markov chain with transition matrix P which has eigenvalues $1 = \lambda_1 > \lambda_2 \geq \ldots \geq \lambda_V > -1$. Then, the rate of convergence of the chain to its stationary distribution is governed by

$$R = \max_{i\geq 2} |\lambda_i|$$

and efficiency of estimation is governed by the ratios

$$\Lambda_i = \frac{1+\lambda_i}{1-\lambda_i}, \qquad i \neq 1.$$

Specifically, the smaller the value of R, the faster the rate of convergence. In addition, the smaller the values of Λ_i $(i \neq 1)$, the more efficient are the ergodic average estimators in terms of their accuracy. Thus, fast convergence rates are achieved by having all the eigenvalues small in absolute value, whereas efficient estimation is achieved by having eigenvalues as small as possible (preferably negative and large in absolute value). It can be seen that these ideals are at odds with each other but that, in terms of comparison between sampling designs, it may be possible to improve on any given MCMC strategy with respect to both of these performance measures.

All the strategies analysed here satisfy the conditions stated in the above result. The state sequence space is finite and all the sweep strategies considered are reversible; i.e. $p_iP_{ij} = p_jP_{ji}$, $i,j = 1,2,\ldots,S$, where $p_i = p(\mathbf{s} = \mathbf{s}^i|\mathbf{y},H)$, $i = 1,2,\ldots,S$, denotes the probability of $\mathbf{s}^i$ under the stationary distribution. Reversibility is proved for Strategy (a) in Lemma 5.1 and follows similarly for the other strategies.

Lemma 5.1

The Gibbs sampler strategy which employs a mixture of forward and backward cycling is reversible; i.e. $p_i P_{ij} = p_j P_{ji}$.

Proof

Let P^t be the transition matrix associated with updating the t^{th} site s_t of the state sequence. Then, for $i = 1, 2, \ldots, S,\ j = 1, 2, \ldots, S$,

$$P_{ij}^t = \begin{cases} p(s_t = s_t^j | \mathbf{s}_{(t)} = \mathbf{s}_{(t)}^i, \mathbf{y}, H) & \text{, if } s_{t'}^i = s_{t'}^j,\ \forall t' \neq t \\ 0 & \text{, otherwise} \end{cases}$$

i.e. if $\mathbf{s}^i$ and $\mathbf{s}^j$ differ only in their t^{th} state, then P_{ij}^t is simply the full conditional probability of s_t^j given the remaining states. Otherwise, $P_{ij}^t = 0$. Thus, if a full forward sweep of all T states is undertaken in each cycle, the associated transition matrix is

$$P^F = P^1 P^2 \ldots P^T.$$

Likewise, the transition matrix for a full backward sweep through the T states is

$$P^B = P^T P^{T-1} \ldots P^1.$$

Thus, for our mixture of forward and backward steps, the transition matrix is

$$P = \frac{1}{2}(P^F + P^B).$$

Now, the target probability p_i may be decomposed as

$$\begin{aligned} p_i &= p(s_1^i, s_2^i, \ldots, s_T^i | \mathbf{y}, H) \\ &= p(s_t^i | \mathbf{s}_{(t)}^i, \mathbf{y}, H)\, p(\mathbf{s}_{(t)}^i | \mathbf{y}, H), \end{aligned}$$

so that, for $t = 1, 2, \ldots, T$, when $\mathbf{s}_{(t)}^i = \mathbf{s}_{(t)}^j$, then

$$\begin{aligned} p_i P_{ij}^t &= p(s_t^i | \mathbf{s}_{(t)}^i, \mathbf{y}, H)\, p(\mathbf{s}_{(t)}^i | \mathbf{y}, H)\, P_{ij}^t \\ &= p(s_t^i | \mathbf{s}_{(t)}^i, \mathbf{y}, H)\, p(\mathbf{s}_{(t)}^i | \mathbf{y}, H)\, p(s_t^j | \mathbf{s}_{(t)}^i, \mathbf{y}, H), && \text{by the definition of } P_{ij}^t \text{ above,} \\ &= p(s_t^i | \mathbf{s}_{(t)}^j, \mathbf{y}, H)\, p(\mathbf{s}_{(t)}^j | \mathbf{y}, H)\, p(s_t^j | \mathbf{s}_{(t)}^j, \mathbf{y}, H), && \text{since } \mathbf{s}_{(t)}^i = \mathbf{s}_{(t)}^j, \\ &= p(s_t^j | \mathbf{s}_{(t)}^j, \mathbf{y}, H)\, p(\mathbf{s}_{(t)}^j | \mathbf{y}, H)\, p(s_t^i | \mathbf{s}_{(t)}^j, \mathbf{y}, H), && \text{by rearrangement,} \\ &= p_j P_{ji}^t, && \text{by the definitions of } p_j \text{ and } P_{ji}^t \text{ given above.} \end{aligned}$$

Similarly, when $\mathbf{s}^i_{(t)} \neq \mathbf{s}^j_{(t)}$ and $P^t_{ij} = 0$, then this implies that $P^t_{ji} = 0$, by symmetry, so that $p_i P^t_{ij} = p_j P^t_{ji} = 0$. Thus, each state update of the Gibbs sampler is locally reversible. Now, for P^F_{ij} and P^B_{ij} defined above, we have that, for all $i = 1, 2, \ldots, S$, $j = 1, 2, \ldots, S$,

$$\begin{aligned}
p_i P^F_{ij} &= p_i \sum_{k_1, k_2, \ldots, k_{T-1}} P^1_{ik_1} P^2_{k_1 k_2} \cdots P^T_{k_{T-1} j} \\
&= \sum_{k_1, k_2, \ldots, k_{T-1}} (p_i P^1_{ik_1}) P^2_{k_1 k_2} \cdots P^T_{k_{T-1} j} \\
&= \sum_{k_1, k_2, \ldots, k_{T-1}} P^1_{k_1 i} (p_{k_1} P^2_{k_1 k_2}) P^3_{k_2 k_3} \cdots P^T_{k_{T-1} j}, \\
&\qquad \text{by the local reversibility condition proved above,} \\
&= \sum_{k_1, k_2, \ldots, k_{T-1}} P^1_{k_1 i} P^2_{k_2 k_1} (p_{k_2} P^3_{k_2 k_3}) \cdots P^T_{k_{T-1} j}, \\
&\qquad \text{by the local reversibility condition,} \\
&\vdots \\
&= \sum_{k_1, k_2, \ldots, k_{T-1}} P^1_{k_1 i} P^2_{k_2 k_1} \cdots P^{T-1}_{k_{T-1} k_{T-2}} P^T_{j k_{T-1}} p_j \\
&= p_j \sum_{k_1, k_2, \ldots, k_{T-1}} P^T_{j k_{T-1}} P^{T-1}_{k_{T-1} k_{T-2}} \cdots P^1_{k_1 i} \\
&= p_j P^B_{ji}. \qquad (5.13)
\end{aligned}$$

Thus, to prove reversibility of P for the full Gibbs sampler strategy:

$$\begin{aligned}
p_i P_{ij} &= p_i \frac{1}{2}(P^F_{ij} + P^B_{ij}) \\
&= \frac{1}{2}(p_i P^F_{ij} + p_i P^B_{ij}) \\
&= \frac{1}{2}(p_j P^B_{ji} + p_j P^F_{ji}), \quad \text{by (5.13),} \\
&= p_j \frac{1}{2}(P^F_{ji} + P^B_{ji}) \\
&= p_j P_{ji}
\end{aligned}$$

and, thus, the lemma is proved. □

Analogous proofs may be obtained for reversibility of Strategies (b) and (d). For Strategy (c), each state update is locally reversible by construction of the Hastings algorithm and, hence, proof of reversibility for the full scheme is similar to that in Lemma 5.1. In addition, all strategies are clearly irreducible (since they allow chains to move between all valid state sequences) and aperiodic (since generating from full conditional distributions, or their heated versions, ensures strictly positive probability is assigned to the event of

"no transition made", hence implying aperiodicity). Thus, it is valid to employ the aforementioned Besag & Green eigenvalue result in our analysis.

In order to compare the sweep strategies, we compute the eigenvalues of the transition matrix driving each particular MCMC scheme. The transition matrix for the Gibbs sampler using a mixture of forward and backward cycling was derived in Lemma 5.1 above. The forward and backward sweeps may be thought of as elements of the entire class of $T!$ permutations of the T states. Thus, in the case of the random permutation sweep strategy, we may label the i^{th} permutation as $\sigma^i = (\sigma_1^i \ \sigma_2^i \ \ldots \ \sigma_T^i)$, where each $\sigma_t^i \in \{1, 2, \ldots, T\}$ and $\sigma_t^i \neq \sigma_{t'}^i$ for $t \neq t'$. The transition matrix P^{σ^i} associated with the i^{th} permutation is then given by

$$P^{\sigma^i} = P^{\sigma_1^i} P^{\sigma_2^i} \ldots P^{\sigma_T^i}.$$

Thus, the transition matrix corresponding to the full random permutation sweep strategy is given by

$$P = \frac{1}{T!} \sum_{i=1}^{T!} P^{\sigma^i}.$$

Similar calculations may be performed for the block updating Gibbs sampler and the transition matrix for the heated Hastings algorithm may be evaluated using the general format for the Hastings algorithm presented earlier in Equations (5.11) and (5.12).

The 3 simple examples analysed here include 2 L-R models and an ergodic model. Both L-R models have 3 states with discrete observation distributions over the 2-symbol set $\{1, 2\}$. Observation sequences of length $T = 6$ were generated so that the state sequence space is of size $S = 28$ (requiring the eigenvalues of a 28×28 transition matrix to be computed). The first model analysed is Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$, which was introduced in Section 5.3.2 and shown to yield a multimodal posterior distribution, $p(\mathbf{s}|\mathbf{y}, H)$. In order to summarise the information provided by the set of 28 transition matrix eigenvalues for each sweep strategy, we record the value $R = \sup_{i \neq 1} |\lambda_i|$, where $\lambda_1 = 1$, which guides the convergence rate, and then tabulate the number of the set of 27 eigenvalues (not including $\lambda_1 = 1$) which lie in various sub-intervals of $(-1, 1)$ (since $\lambda_i \in (-1, 1)$, $i \neq 1$). We recall that efficiency of estimation is governed by having eigenvalues as small as possible, preferably negative, so we want the sum of all eigenvalues to be small. As a baseline standard for comparison, we recall that the transition matrix associated with independent sampling has $\lambda_1 = 1$ and $\lambda_i = 0$, $i \neq 1$. Table 5.2 displays the results. In terms of convergence rate, block updating converges

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.922	9	1	2	—	5	10	4.142
(b)	0.930	5	7	2	2	2	9	4.142
(c)	0.935	4	6	—	2	4	11	5.267
(d)	0.843	3	2	14	3	—	5	2.860

Table 5.2: Eigenvalue comparison of sweep strategies for Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$

more rapidly, as would be expected. The heated Hastings strategy converges slowest and further analysis indicated that its performance worsened with increasing temperature. The forward-backward Gibbs sampler compares slightly more favourably with that of a purely random permutation strategy, possibly due to the natural sequential order in which state sequences are generated. The efficiency estimation problem is slightly more tricky since it depends on the function being estimated. However, the heated Hastings strategy again appears inferior whilst block updating seems the best option. There is little to choose between the random permutation strategy and the forward-backward mixture.

The second L-R model studied, Model 2, is defined by the parameter set

$$\pi = \begin{pmatrix} 0.5 & 0.3 & 0.2 \end{pmatrix},$$

$$A = \begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} 0.2 & 0.8 \\ 0.75 & 0.25 \\ 0.3 & 0.7 \end{pmatrix},$$

from which the observation sequence $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ was generated. The posterior distribution $p(\mathbf{s}|\mathbf{y}, H)$ in this case is much more unimodal, as can be seen from Table 5.3 by noting the similar state structure of the top 8 most probable sequences, which carry nearly 82% of the total probability. The eigenvalue summary is displayed in Table 5.4. The results here reinforce those of the previous example: the inferiority of the heated Hastings strategy and the superiority of block updating. Although the forward-backward sampler has a slightly faster convergence rate than the random permutation strategy, again they both have equivalent estimation properties. Also, Tables 5.2 and 5.4 illustrate the fact that, as would be expected, MCMC sampler designs converge

i	State sequence, $\mathbf{s}^i$	$p(\mathbf{s}^i\|\mathbf{y}, H)$
1	3 3 3 3 3 3	0.2021
2	1 3 3 3 3 3	0.1733
3	1 1 2 2 3 3	0.1238
4	1 1 2 3 3 3	0.0990
5	1 1 3 3 3 3	0.0792
6	2 3 3 3 3 3	0.0541
7	1 2 2 2 3 3	0.0483
8	1 2 2 3 3 3	0.0387

Table 5.3: Most probable state sequences underlying $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ for Model 2

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.810	6	4	1	3	4	9	3.120
(b)	0.823	4	8	1	3	4	7	3.120
(c)	0.849	2	5	3	2	4	11	4.806
(d)	0.716	2	3	15	2	1	4	2.271

Table 5.4: Eigenvalue comparison of sweep strategies for Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.567	3	15	23	21	12	6	1.935
(b)	0.586	2	9	29	30	6	4	1.935
(c)	0.660	—	13	14	29	17	7	3.592
(d)	0.308	1	1	74	2	—	2	1.250

Table 5.5: Eigenvalue comparison of sweep strategies for Model 3 with $\mathbf{y} = (2\ 2\ 1\ 1)$

faster the more unimodal the target distribution and exhibit more efficient estimation properties.

The final example is an ergodic model, Model 3, with $N = 3$ and $K = 2$ as before, only sequences of length $T = 4$ are now generated to maintain a state sequence space of manageable size ($S = 81$). The model parameters are given by

$$\pi = \begin{pmatrix} 0.6 & 0.1 & 0.3 \end{pmatrix},$$

$$A = \begin{pmatrix} 0.7 & 0.1 & 0.2 \\ 0.2 & 0.5 & 0.3 \\ 0.05 & 0.2 & 0.75 \end{pmatrix}, \qquad B = \begin{pmatrix} 0.3 & 0.7 \\ 0.75 & 0.25 \\ 0.65 & 0.35 \end{pmatrix},$$

and the observation sequence generated was $\mathbf{y} = (2\ 2\ 1\ 1)$. Table 5.5 displays the eigenvalue summary. Similar conclusions can be drawn as for the L-R models with respect to sweep strategy comparisons. Tables 5.4 and 5.5 also illustrate the fact that convergence rates are much faster and estimation properties better for unconstrained models, as would be expected.

To summarise, all 3 examples seem to indicate that the heated Hastings strategy is of no benefit in our particular class of models. They also illustrate the improvement in performance achieved by updating blocks of r variables simultaneously according to the appropriate r-variate full conditional distribution, which may be feasible in the case of constrained HMMs. For a L-R HMM, if we update sequences in blocks of r and it is desired to generate new values of states in the block $\{s_{t+1}, s_{t+2}, \ldots, s_{t+r}\}$, for some $t \in \{0, 1, \ldots, T-r\}$, then we can use the form of (1.4) to establish that the number of partial state sequences for which we are required to evaluate the full conditional

distribution is

$$\begin{pmatrix} s_{t+r+1} - s_t + r \\ s_{t+r+1} - s_t \end{pmatrix}$$

(where we adopt the convention that $s_0 = 1$, $s_{T+1} = N$) due to the L-R constraints. So, for example, if $N = 4$, $T = 8$, $r = 4$ and the current state sequence is $\mathbf{s} = (1\ 1\ 2\ 2 \vdots 2\ 2\ 4\ 4)$, we are required to update in 2 stages the segments of $\mathbf{s}$ separated by " $\vdots$ ". Updating the first block requires evaluating full conditional probabilities for only 5 partial state sequences, whilst the second block requires evaluations for just 15 sequences. Thus, the increase in computation is not too great and may well be justified by the improvement in performance. If simultaneous updating of more than 1 state is feasible, it is not necessary to implement it at every iteration — a fixed cycle with standard univariate Gibbs sampling may be enough to improve the properties of the latter whilst keeping the increase in computational burden to a minimum. Coupled with this, the mixture of forward and backward cycling through the states offers a more robust strategy than forward- or backward-cycling alone.

5.5 Can utilisation of the generalised Viterbi algorithm improve sampling from $p(\mathbf{s}|\mathbf{y}, H)$?

Section 5.3.2 highlights the problems which can occur when using single-site updating strategies to sample from $p(\mathbf{s}|\mathbf{y}, H)$, due to the "local" nature of changes to the state sequences. It is, thus, desired to amend the sampling scheme to allow for the possibility of making more "global" changes, especially in the face of multimodality which can occur under constrained models in particular. The generalised Viterbi algorithm provides an efficient technique for evaluating the set of L most probable sequences under $p(\mathbf{s}|\mathbf{y}, H)$, denoted $\mathcal{S}_L = \{\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^L\}$, which we may term the L "modes". So, can this information be put to good use in designing an improved sampling scheme?

Figure 5.1 illustrates the phenomenon of "sticking" within particular neighbourhoods of the state sequence space which can occur when using the single-site updating Gibbs sampler, so it would be desirable to find a sampling scheme which permits jumps to be made between the most probable state sequences, in order to sample from all regions of high probability. One possibility is to construct a mixture CGF and apply the Hastings

algorithm. For $\epsilon \in (0, 1]$, consider the CGF given by

$$q(\mathbf{s}, \mathbf{s}') = \begin{cases} q^1(\mathbf{s}, \mathbf{s}') & , \text{if } \mathbf{s} \notin \mathcal{S}_L \\ \epsilon q^1(\mathbf{s}, \mathbf{s}') + (1 - \epsilon) q^2(\mathbf{s}, \mathbf{s}') & , \text{if } \mathbf{s} \in \mathcal{S}_L \end{cases}$$

where $q^1(\mathbf{s}, \mathbf{s}')$ denotes some CGF which makes local proposals for $\mathbf{s}'$ in the neighbourhood of $\mathbf{s}$ and $q^2(\mathbf{s}, \mathbf{s}')$ denotes a more global CGF which proposes candidates in $\mathcal{S}_L$ and may be defined by

$$q^2(\mathbf{s}, \mathbf{s}') = \begin{cases} q_{ij} & , \text{if } \mathbf{s} = \mathbf{s}^i \text{ and } \mathbf{s}' = \mathbf{s}^j \\ 0 & , \text{otherwise.} \end{cases}$$

The q_{ij}'s may be chosen to be uniform over $\mathcal{S}_L$ or proportional to $p(\mathbf{s}^j|\mathbf{y}, H)$ (which is provided by the generalised Viterbi algorithm) or even such that $q_{ii} = 0$, $\forall i$, to encourage transitions away from the current region of the state sequence space. The value of ϵ will reflect the relative importance of local and global proposals, perhaps guided by how unimodal the distribution is judged to be. It can be seen that the CGF $q^2(\cdot, \cdot)$ used on its own (i.e. with $\epsilon = 0$) is not valid because it is not irreducible. However, the full mixture will be irreducible (and thus ensure valid convergence of the sampling scheme) provided $q^1(\cdot, \cdot)$ is constructed to be so. The Hastings algorithm above may be cycled with the Gibbs sampler strategy to generate from $p(\mathbf{s}|\mathbf{y}, H)$, in order to incorporate the information provided by the generalised Viterbi algorithm in the sampling scheme and, hopefully, ease movement between the most probable regions of the state sequence space.

We illustrate the possible benefits of such a sampling strategy by considering the simple multimodal example of $p(\mathbf{s}|\mathbf{y}, H)$ introduced in Section 5.3.2, where H is Model 1 and $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$. The mixture Hastings algorithm described above is compared for values of $\epsilon = 0.1, 0.5, 0.9, 1$, where $\epsilon = 1$ corresponds to implementation of the CGF $q^1(\cdot, \cdot)$ alone. If $\mathcal{N}_\mathbf{s}$ denotes the neighbourhood of $\mathbf{s}$ defined by the set of all 1-state neighbours of $\mathbf{s}$, then the local CGF $q^1(\mathbf{s}, \mathbf{s}')$ considered in this example may be defined by a uniform distribution over the neighbourhood of the current point; i.e.

$$q^1(\mathbf{s}, \mathbf{s}') = \begin{cases} \frac{1}{|\mathcal{N}_\mathbf{s}|} & , \text{if } \mathbf{s}' \in \mathcal{N}_\mathbf{s} \\ 0 & , \text{otherwise.} \end{cases}$$

Now, under a L-R HMM, the number of 1-state neighbours of any state sequence $\mathbf{s}$ which differ only at the t^{th} state is clearly $(s_{t+1} - s_{t-1})$, for $t = 1, 2, \ldots, T$, with the convention that $s_0 = 1$ and $s_{T+1} = N$. Thus, in total, the number of 1-state neighbours

of s, denoted by $|\mathcal{N}_{\mathbf{s}}|$, can be computed as

$$\begin{aligned} |\mathcal{N}_{\mathbf{s}}| &= \sum_{t=1}^{T}(s_{t+1} - s_{t-1}) \\ &= \sum_{t=2}^{T+1} s_t - \sum_{t=0}^{T-1} s_t \\ &= s_T + s_{T+1} - s_0 - s_1 \\ &= (s_T - s_1) + (N-1). \end{aligned}$$

In this example, we take $L = 2$ and $\mathcal{S}_2 = \{(1\ 1\ 1\ 1\ 1\ 1),\ (3\ 3\ 3\ 3\ 3\ 3)\}$, so that $\mathcal{S}_2$ comprises the 2 modes of $p(\mathbf{s}|\mathbf{y}, H)$, as given in Table 5.1. Then, $q^2(\cdot,\cdot)$ is defined to be

$$q^2(\mathbf{s},\mathbf{s}') = \begin{cases} 1 & \text{, if } \mathbf{s} \in \mathcal{S}_2 \ \&\ \mathbf{s}' \in \mathcal{S}_2 \text{ and } \mathbf{s} \neq \mathbf{s}' \\ 0 & \text{, otherwise} \end{cases}$$

which prevents the proposed sequence being the current state sequence in an attempt to encourage movement between modes. The average acceptance rates for each strategy, computed over a run of 25,000 iterations, are given by

Value of ϵ	**Acceptance rate**
1	58.3%
0.9	60.3%
0.5	69.8%
0.1	73.0%

illustrating a trend for the acceptance rate to increase with decreasing value of ϵ. From an initial run of 100 iterations for each of the strategies, initialised by the global mode (3 3 3 3 3 3), histograms were plotted of the output frequencies for the entire state sequence space (where sequences are labelled according to the scheme described in Appendix C.1). Figure 5.2 displays the true distribution, $p(\mathbf{s}|\mathbf{y}, H)$, whilst Figure 5.3 displays the histograms obtained from the mixture Hastings algorithm under various specifications of ϵ. When $\epsilon = 1$ and the local neighbourhood CGF alone is implemented, only 10 of the entire space of 28 state sequences are visited in 100 iterations. These are

(3 3 3 3 3 3)	(2 2 3 3 3 3)
(2 3 3 3 3 3)	(1 2 3 3 3 3)
(1 3 3 3 3 3)	(1 2 2 3 3 3)
(1 1 3 3 3 3)	(1 1 2 3 3 3)
	(1 1 1 3 3 3)
	(1 1 1 2 3 3)

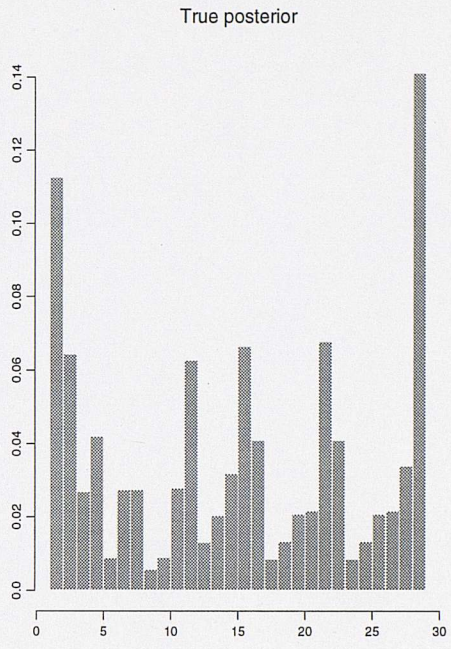


Figure 5.2: Exact posterior distribution, $p(\mathbf{s}|\mathbf{y}, H)$, under Model 1

with disproportionately high frequencies being exhibited by those sequences indicated in bold type. These all lie in a very close neighbourhood of the initial state sequence (3 3 3 3 3 3) and the other mode, (1 1 1 1 1 1), is not visited at all, indicating restricted movement about the state sequence space; i.e. the sampler has got "stuck" in the neighbourhood of the global mode. However, when $\epsilon = 0.5$, say, 24 of the state sequences are visited with frequencies which agree more closely with those of the true distribution. Inclusion of some "modal" information, therefore, seems to speed coverage of the state sequence space.

Now, autocorrelations indicate the dependence which exists in the chain; the faster autocorrelations tend to zero with increasing lag, the more rapid the mixing within chains. Figure 5.4 displays the within-chain autocorrelation structure, calculated from runs of length 10,000, exhibited under the mixture Hastings algorithm for different values of ϵ. The unusual form exhibited when $\epsilon = 0.1$ is probably due to the fact that chains are jumping too frequently, back and forth, between (1 1 1 1 1 1) and (3 3 3 3 3 3) without fully exploring the modal neighbourhoods (i.e. ϵ is too low and insufficient weight is placed on local proposals). However, even when $\epsilon = 0.9$ and the global CGF is implemented only occasionally, autocorrelations are reduced dramatically in comparison with those exhibited when using the local CGF alone. In addition, for mixture Hastings

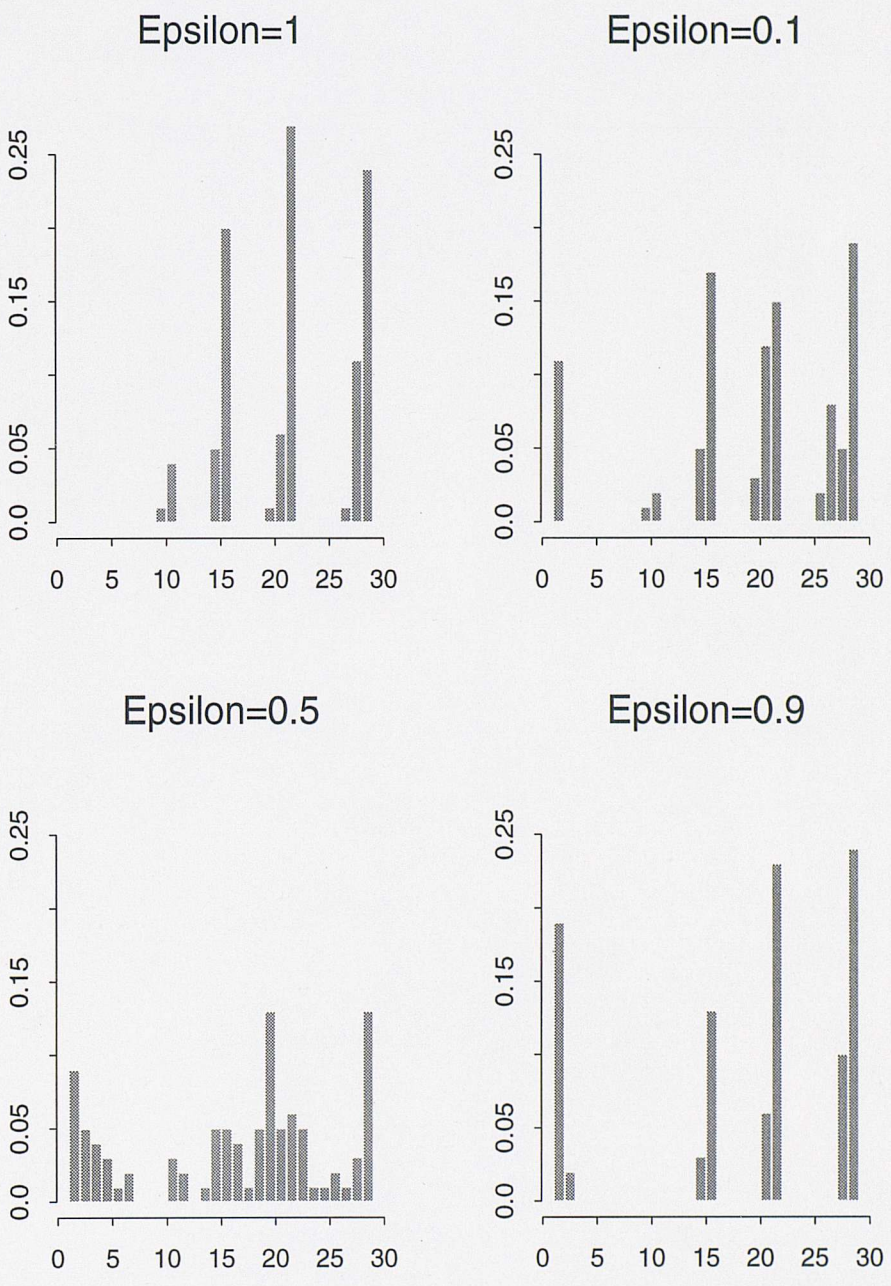


Figure 5.3: Histograms representing state sequence frequencies under the mixture Hastings algorithm

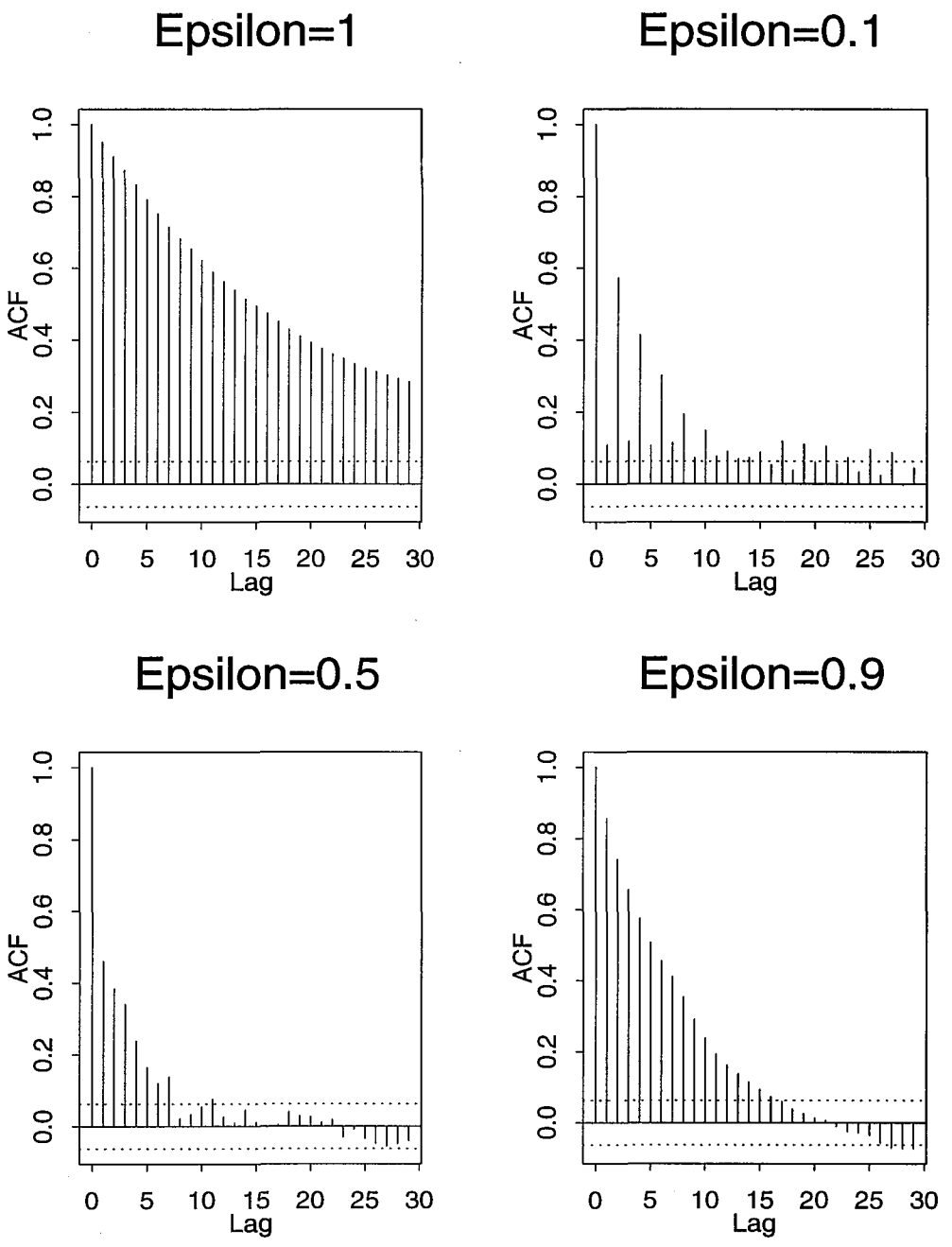


Figure 5.4: Autocorrelation structure exhibited under the mixture Hastings algorithm

algorithms with $\epsilon = 0.1, 0.5, 0.9, 1$, distance measures between the true distribution and the empirical distribution (based on output sample frequencies) were monitored with increasing iteration. The results reinforce those obtained above. Distance measures for $\epsilon < 1$ were uniformly lower than for $\epsilon = 1$, indicating that convergence of output-based empirical distributions to the true distribution is more rapid when the global information is incorporated in the Hastings CGF. Therefore, inclusion of the global CGF, even for large values of ϵ, seems to improve the sampling properties of the Hastings algorithm in this simple example.

However, as the size S of the state sequence space increases for fixed L, the effect of the global CGF will become less significant, since the probability of the chain lying within $\mathcal{S}_L$ at any point (which was equal to

$$p(\mathbf{s} = (3\ 3\ 3\ 3\ 3\ 3)\,|\,\mathbf{y}, H) + p(\mathbf{s} = (1\ 1\ 1\ 1\ 1\ 1)\,|\,\mathbf{y}, H) \ \simeq 0.25,$$

in the above simple example) will decrease, thus decreasing the instances at which $q^2(\cdot,\cdot)$ is implemented. So, for more realistic models, it will be required to specify a distribution over a large discrete set $\mathcal{S}_L$ in order for the global CGF to play a significant role in sampling. However, this is not computationally feasible. Not only that, but when the model H is unknown and it is required to sample from $p(\mathbf{s}, H|\mathbf{y})$, the set $\mathcal{S}_L$ specified for Stage 2 sampling of $p(\mathbf{s}|\mathbf{y}, H)$ must be fixed throughout any sampler run, otherwise the correct stationary distribution will not be maintained. However, the generalised Viterbi algorithm only calculates $\mathcal{S}_L$ for fixed H and this set will change as H changes at each iteration. Therefore, the mixture Hastings algorithm described above will not be useful in the case of unknown H.

However, the ideas described in this section may prove to be useful when considering continuous distributions. For example, suppose the target distribution, $\pi(\mathbf{x})$, for $\mathbf{x} = (x_1, x_2, \ldots, x_k) \in \mathcal{X}$, is known to be multimodal or highly correlated and other methods for dealing with these problems (such as auxiliary variables, re-parameterisation, etc) are either inapplicable or ineffective. Then, if there exist computationally cheap procedures for approximately locating modes, say, as with the generalised Viterbi algorithm in the case of a HMM, this information can be included in a mixture Hastings CGF. So, if $q^1(\mathbf{x}, \mathbf{x}')$ is specified to be a local CGF proposing candidates in some neighbourhood of the current point and there are L "modal" regions available, $\{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_L\}$, with $\mathcal{R} = \bigcup_i \mathcal{R}_i$, that have been constructed about the mode approximations (k-dimensional

cubes, say), then the Hastings CGF may be defined by

$$q(\mathbf{x},\mathbf{x}') = \begin{cases} q^1(\mathbf{x},\mathbf{x}') & , \text{if } \mathbf{x} \notin \mathcal{R} \\ \epsilon\, q^1(\mathbf{x},\mathbf{x}') + (1-\epsilon)\, q^2(\mathbf{x},\mathbf{x}') & , \text{if } \mathbf{x} \in \mathcal{R}, \end{cases}$$

where

$$q^2(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{L} p_i\, q_i^2(\mathbf{x},\mathbf{x}'), \qquad \sum_{i=1}^{L} p_i = 1,$$

and $q_i^2(\mathbf{x},\mathbf{x}')$ proposes candidates in region $\mathcal{R}_i$, for $i = 1,2,\ldots,L$. The p_i's may be chosen as uniform or proportional to the height of the i^{th} mode and the $q_i^2(\mathbf{x},\mathbf{x}')$'s may be specified by any suitable distribution, e.g. uniform, constrained Normal centred at the mode. The choice of ϵ could be made based on analysis of pilot runs of the sampler and guided by average acceptance rates, as is suggested in the general case of specification of scale parameters for Hastings CGFs. Use of both local and global CGFs in this way should ensure the algorithm remains efficient in terms of maintaining reasonable acceptance rates.

The discrete nature of the state sequence distribution, $p(\mathbf{s}|\mathbf{y},H)$, precludes use of such a mixture Hastings algorithm under realistic models. Such problems no longer arise when considering continuous distributions, since the union of modal regions, $\mathcal{R}$, may be chosen to carry a large percentage of the target probability, thus ensuring the global CGF contributes significantly to the mixture Hastings algorithm (e.g. if 2 modal regions carry about 25% of the total target probability, we would expect results similar to those obtained for the L-R HMM example considered earlier). Thus, if the target distribution is multimodal, such a Hastings algorithm may ease movement between modes if the mode-location procedures are reliable. In fact, we ran a mixture Hastings algorithm, of the form specified above, in the case of a multimodal bivariate Normal target distribution and obtained similar results as for the HMM example; i.e. use of a local Hastings CGF alone caused "sticking" within the initial modal neighbourhood, whilst incorporation of the global CGF allowed movement between the modes. As an alternative method of incorporating prior "modal" information in the Hastings algorithm, standard continuous distributions over the entire target space may be specified for the global CGF; e.g. $q^2(\mathbf{x},\cdot)$ could be defined by a mixture of L multivariate Normals, each being centred at one of the mode approximations.

5.6 Application of the Hastings algorithm

When sampling from $p(\mathbf{s}|\mathbf{y}, H)$ using single-state updating, the dependence which exists between states induces high autocorrelations between successive iterates of $\mathbf{s}$, thus restricting movement about the state sequence space. Since we are dealing with a discrete distribution, techniques such as re-parameterisation and random-direction sampling are not available for use in attempting to overcome this problem.

Now, we can express any distribution $p(x_1, x_2, \ldots, x_k)$ in terms of conditional distributions as

$$p(x_1, x_2, \ldots, x_k) = p(x_1)\, p(x_2|x_1) \ldots p(x_k|x_{k-1}, x_{k-2}, \ldots, x_1),$$

by repeated use of the definition of conditional probability. To maintain clarity of notation, we suppress dependence on $\mathbf{y}$ and H (which are fixed) so that the target distribution is $p(\mathbf{s}) \equiv p(\mathbf{s}|\mathbf{y}, H)$ for now. Then, if it were possible to sample $\mathbf{s}$, state-by-state, as

$$\begin{aligned} 1) \quad & s_1 \sim p(s_1) \\ 2) \quad & s_2 \sim p(s_2|s_1) \\ & \vdots \quad\quad \vdots \\ T) \quad & s_T \sim p(s_T|s_{T-1}, s_{T-2}, \ldots, s_1), \end{aligned}$$

this would be equivalent to exact and independent sampling from the target distribution $p(\mathbf{s})$. This is not feasible in most practical applications due to the size of the state sequence space $\mathcal{S}$. However, we may use a similar idea to specify an independence Hastings CGF for proposing candidate sequences of length T, thus allowing simultaneous updating of all T states. The target distribution in this case can be specified as

$$p(s_1, s_2, \ldots, s_T) \propto \pi_{s_1}\, b_{s_1}(y_1)\, a_{s_1 s_2}\, b_{s_2}(y_2) \ldots a_{s_{T-1} s_T}\, b_{s_T}(y_T).$$

Therefore, we suggest the Hastings independence CGF, $q(\mathbf{s})$, given by

$$q(\mathbf{s}) = q_1(s_1)\, q_2(s_2|s_1) \ldots q_T(s_T|s_{T-1}, s_{T-2}, \ldots, s_1),$$

where we take $q_t(s_t|s_{t-1}, s_{t-2}, \ldots, s_1) = q_t(s_t|s_{t-1})$, $t = 2, 3, \ldots, T$, due to the Markov property of the state sequences, and the q_t's are defined by

$$\begin{aligned} q_1(s_1) &\propto \pi_{s_1}\, b_{s_1}(y_1) \\ q_t(s_t|s_{t-1}) &\propto a_{s_{t-1} s_t}\, b_{s_t}(y_t), \qquad t = 2, 3, \ldots, T. \end{aligned}$$

Thus, a candidate state sequence, s′, is generated state-by-state as

$$\begin{aligned} 1)\quad s'_1 &\sim q_1(s'_1) \\ 2)\quad s'_2 &\sim q_2(s'_2|s'_1) \\ &\vdots \\ T)\quad s'_T &\sim q_T(s'_T|s'_{T-1}). \end{aligned}$$

Note the similarity between the q_t's and the corresponding full conditional distributions for the s_t's given in (5.8)–(5.10). In this way, structure from the model $p(\mathbf{s})$ may be incorporated in the specification of the Hastings CGF. Now, given the q_t's as defined above, the Hastings acceptance probability can be computed as

$$\alpha(\mathbf{s}, \mathbf{s}') = \min\left\{1, \frac{p(\mathbf{s}')q(\mathbf{s})}{p(\mathbf{s})q(\mathbf{s}')}\right\},$$

where

$$\frac{p(\mathbf{s}')}{p(\mathbf{s})} = \frac{\pi_{s'_1} b_{s'_1}(y_1) a_{s'_1 s'_2} \ldots a_{s'_{T-1} s'_T} b_{s'_T}(y_T)}{\pi_{s_1} b_{s_1}(y_1) a_{s_1 s_2} \ldots a_{s_{T-1} s_T} b_{s_T}(y_T)}$$

and

$$\frac{q(\mathbf{s})}{q(\mathbf{s}')} = \frac{\pi_{s_1} b_{s_1}(y_1)}{\pi_{s'_1} b_{s'_1}(y_1)} \frac{a_{s_1 s_2} b_{s_2}(y_2)/c_2}{a_{s'_1 s'_2} b_{s'_2}(y_2)/c'_2} \cdots \frac{a_{s_{T-1} s_T} b_{s_T}(y_T)/c_T}{a_{s'_{T-1} s'_T} b_{s'_T}(y_T)/c'_T}.$$

For $t = 2, 3, \ldots, T$, the normalising constants, c_t and c'_t, associated with $q_t(s_t|s_{t-1})$ and $q_t(s'_t|s'_{t-1})$ respectively, are defined by

$$c_t = \sum_{j=1}^{N} a_{s_{t-1} j} b_j(y_t), \qquad c'_t = \sum_{j=1}^{N} a_{s'_{t-1} j} b_j(y_t).$$

Thus, the acceptance probability is computed as

$$\alpha(\mathbf{s}, \mathbf{s}') = \min\left\{1, \prod_{t=2}^{T} \frac{c'_t}{c_t}\right\}.$$

The ratio c'_t/c_t can be thought of as measuring how likely the $(t-1)^{th}$ state in the sequence is to be s'_{t-1} as opposed to s_{t-1}, given y_t.

The above independence Hastings CGF, when used in the context of a fixed HMM, H, is uniformly ergodic since, by virtue of the discrete nature of the state sequence space, p/q is bounded and, thus, q satisfies the sufficient condition for uniform ergodic convergence stated in "Hybrid MCMC schemes" of Section 4.5.2. In fact, this strong convergence property is transferred to the chain $(H^{(1)}, H^{(2)}, \ldots)$ in the case of unknown H, using a similar argument to that presented in Robert et al (1993) for the forward-cycling, single-site updating Gibbs sampler.

Furthermore, the independence Hastings algorithm described above compares very favourably with the forward-backward cycling Gibbs sampler in terms of computation time. When generating from the full conditional distributions, the Gibbs sampler updates each state according to a discrete distribution specified by probabilities of the form

$$\frac{a_{s_{t-1}s_t} b_{s_t}(y_t) a_{s_t s_{t+1}}}{\sum_{j=1}^{N} a_{s_{t-1}j} b_j(y_t) a_{j s_{t+1}}},$$

whilst the Hastings algorithm proposes each state according to probabilities of the form

$$\frac{a_{s_{t-1}s_t} b_{s_t}(y_t)}{\sum_{j=1}^{N} a_{s_{t-1}j} b_j(y_t)},$$

which requires less calculations. However, the Hastings algorithm requires additional computation of the normalising constants, c_t, $t = 2, 3, \ldots, T$, in order to evaluate the acceptance probability. Now, by definition, $c_t = c'_t$ if $s_{t-1} = s'_{t-1}$ so that, in this case, $\frac{c'_t}{c_t} = 1$ in the acceptance probability α. Thus, it is only necessary to evaluate c_t for those t such that $s_{t-1} \neq s'_{t-1}$; so, if $\mathbf{s}$ and $\mathbf{s}'$ agree at many states, additional computations for evaluating α will be low, whereas if $\mathbf{s}$ and $\mathbf{s}'$ are structurally very different, the additional computation may be justified by the fact that a global transition may be made. In our experience, practical computation times for both strategies tend to be equivalent.

There are many more sophisticated independence CGFs, following similar ideas to those outlined above, which could be formulated, the only constraint being computational burden. For example, an alternative independence CGF may generate s_t given s_{t-1} conditional not only on y_t (as before) but also on y_{t+1}; i.e. propose a candidate sequence $\mathbf{s}'$ according to the density

$$q(\mathbf{s}') = q_1(s'_1)\, q_2(s'_2|s'_1) \ldots q_T(s'_T|s'_{T-1}),$$

where

$$\begin{aligned}
q_1(s'_1) &\propto \pi_{s'_1} b_{s'_1}(y_1) \sum_{j=1}^{N} a_{s'_1 j}\, b_j(y_2) \\
q_2(s'_2|s'_1) &\propto a_{s'_1 s'_2}\, b_{s'_2}(y_2) \sum_{j=1}^{N} a_{s'_2 j}\, b_j(y_3) \\
&\vdots \qquad \vdots \\
q_T(s'_T|s'_{T-1}) &\propto a_{s'_{T-1} s'_T}\, b_{s'_T}(y_T).
\end{aligned}$$

Taking into account the extra information provided by y_{t+1} when proposing s'_t should improve on the original independence Hastings algorithm. There is a slight modification

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.922	9	1	2	—	5	10	4.142
(b)	0.398	—	—	—	3	10	14	4.379
(c)	0.246	—	—	—	4	16	7	3.077
(d)	0.849	—	—	1	7	13	6	4.113
(e)	0.159	—	—	—	13	12	2	1.770
(f)	0.150	—	6	5	8	6	2	1.336

Table 5.6: Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.810	6	4	1	3	4	9	3.120
(b)	0.575	—	—	—	2	9	16	5.546
(c)	0.333	—	—	—	—	20	7	4.085
(d)	0.657	—	—	—	12	11	4	2.920
(e)	0.330	—	—	—	11	12	4	2.385
(f)	0.187	—	6	5	8	7	1	1.348

Table 5.7: Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$

to the acceptance rate when using this improved version and, of course, an increase in the computational complexity. However, it may not be necessary to apply such Hastings steps at every iteration in order to improve the properties of the sampler design.

We shall now re-analyse the examples considered in the sweep strategy comparison of Section 5.4. We compare the performance of the Gibbs sampler, using a mixture of forward and backward cycling, with both of the aforementioned independence Hastings algorithms and also consider a hybrid scheme in the form of a cycle of Gibbs sampler and Hastings algorithm steps (using the original independence Hastings CGF). Tables 5.6, 5.7 and 5.8 summarise the transition matrix eigenvalues for comparison under each

		Number of eigenvalues in the interval:						
	R	(-1,-0.1]	(-0.1,-0.01]	(-0.01,0]	(0,0.01]	(0.01,0.1]	(0.1,1)	Sum
(a)	0.567	3	15	23	21	12	6	1.935
(b)	0.423	—	—	—	32	—	48	10.886
(c)	0.325	—	—	—	8	9	63	15.804
(d)	0.322	—	—	19	52	7	2	1.720
(e)	0.179	—	—	—	32	36	12	3.633
(f)	0.084	—	6	31	36	7	—	1.125

Table 5.8: Eigenvalue comparison of Gibbs sampling and the Hastings algorithm under Model 3 with $\mathbf{y} = (2\ 2\ 1\ 1)$

of these strategies, where the latter are labelled according to:

(a) Gibbs sampler

(b) original Hastings algorithm

(c) alternative Hastings algorithm

(d) 2 sweeps of the Gibbs sampler

(e) 2 sweeps of the original Hastings algorithm

(f) a Hastings/Gibbs cycle

It is clear that both independence Hastings algorithms exhibit substantially faster convergence rates than the Gibbs sampler and that the more sophisticated algorithm does indeed improve on the original. However, the Gibbs sampler appears to exhibit better general estimation properties in terms of admitting some negative eigenvalues and positive eigenvalues of a smaller magnitude than those under the Hastings algorithm. It would seem reasonable, therefore, to expect a combination of both sampling schemes within a cycle to inherit good convergence properties from the Hastings algorithm and good estimation properties from the Gibbs sampler. This is borne out in all 3 examples: the Hastings/Gibbs cycle improves on both Gibbs sampling and the Hastings algorithm alone in terms of both convergence rate and efficiency of estimation.

When observation sequences are very long, it may not be reasonable to attempt to update entire state sequences simultaneously in the Hastings algorithm, since accompanying acceptance rates may be low. Thus, the original independence Hastings algorithm may be modified to update simultaneously in blocks of r states, one full cycle being achieved once all T states have been updated. When $r = T$, we are reduced to the independence Hastings algorithm, and when $r = 1$, we are reduced to the Gibbs sampler. For $1 < r < T$, it seems sensible to choose values of r which divide T exactly (although this is not necessary) so that one sweep of the sites $\{s_1, s_2, \ldots, s_T\}$ consists of $\frac{T}{r}$ applications of the Hastings algorithm — the first of which updates sites $\{s_1, s_2, \ldots, s_r\}$, the second of which updates sites $\{s_{r+1}, s_{r+2}, \ldots, s_{2r}\}$, etc. As in the general case of sweep strategy choice, it is not necessary for the blocks to be fixed and updated in a prespecified order. For example, the r states to be updated simultaneously may be chosen at random. As a further generalisation, mixtures or cycles of Hastings steps with different values of r may be employed in order to introduce a balance between both local (small r) and more global (large r) proposals; for example, the cycle of independence Hastings algorithm ($r = T$) and Gibbs sampler ($r = 1$) steps.

Additional comparison measures which shall be employed here to illustrate differences in the performance of various sampler designs are distance measures and within-chain autocorrelations. Distance measures may be computed between the true distribution (feasible to calculate in these small examples) and the empirical sample distribution at each iteration t (obtained by estimating each state sequence probability according to its relative frequency within the MCMC output chain up to iteration t, for the sampling strategy under consideration) and these can be monitored throughout the sampler run. If independent replications of the MCMC strategy are to be performed, then average distance measures over all replications may be monitored, along with the corresponding standard deviations. The smaller the average distance measures, the closer the output-based empirical distribution is to the true distribution, and smaller standard deviations about these average distances should reflect more rapid mixing within each of the individual output chains. Autocorrelations may also be plotted, lower correlations indicating more rapid mixing and better estimating properties of the chain in terms of higher precision in ergodic average estimates.

We briefly return to Models 1 and 2 and monitor the above performance indicators for the sampler designs already discussed, as well as for 2 new designs comprising the

Hastings algorithm with $r = 2$ and $r = 3$, respectively. It should be noted that all the Hastings CGFs considered here (apart from when $r = 1$, equivalent to the Gibbs sampler) update blocks of states in a forward-cycling manner and, thus, a more robust strategy may be to mix forward- and backward-cycling sweeps of the states, as for the Gibbs sampling scheme. For brevity in all tables and figures, the sampler designs are labelled as:

(a) mixture of forward- and backward-cycling Gibbs samplers ($r = 1$)

(b) original independence Hastings algorithm ($r = T = 6$)

(c) alternative independence Hastings algorithm

(d) Hastings/Gibbs cycle

(e) Hastings algorithm ($r = 2$)

(f) Hastings algorithm ($r = 3$)

(g) independent sampling from the true distribution

We recall that Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$ yields a fairly multimodal posterior distribution for $\mathbf{s}$ and Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$ yields a reasonably unimodal posterior distribution for $\mathbf{s}$. All the sampler designs were run for 1000 iterations and replicated 28 times, each replication being initialised by a different state sequence.

The distance measures employed are the averaged L_2-norm and the total variation distance (although other distance measures were investigated and yielded similar results). Over a finite set of size V, let discrete distributions p and p' be denoted by $\{p_1, p_2, \ldots, p_V\}$ and $\{p'_1, p'_2, \ldots, p'_V\}$, respectively. The averaged L_2-norm distance between p and p' may then be defined by

$$d(p, p') = \sum_{i=1}^{V} (p_i - p'_i)^2 p_i.$$

If p represents the true target distribution and p' represents an empirical distribution based on the output from a MCMC sampler after some specified number of iterations, then the averaged L_2-norm places emphasis on the empirical distribution doing well in the most probable regions of the state sequence space. If $\mathcal{V}$ is the set of all subsets of $\{1, 2, \ldots, V\}$, then the total variation distance between p and p' is defined to be

$$d(p, p') = 2 \max_{v \in \mathcal{V}} |\sum_{i \in v} p_i - \sum_{i \in v} p'_i|. \tag{5.14}$$

Strategy	Acceptance rate	Movement rate
(a)	100%	73.9%
(b)	88.5%	82.4%
(c)	93.3%	87.2%
(d)	94.2%	78.0%
(e)	91.8%	70.3%
(f)	88.1%	67.2%

Table 5.9: Acceptance and movement rates under Model 1 with $\mathbf{y} = (1\ 2\ 1\ 2\ 2\ 1)$

So, when p represents the true target distribution and p' the empirical distribution based on a MCMC output sample, then the variation distance corresponds to twice the maximum discrepancy which can be made for any probability statement about the true distribution based on the estimate provided by the empirical distribution. It can easily be seen that (5.14) may be re-expressed as

$$\begin{aligned} d(p,p') &= 2\sum_{i=1}^{V}\max(p_i - p'_i, 0) \\ &= \sum_{i=1}^{V}\max(p_i - p'_i, 0) + \sum_{i=1}^{V}\max(p'_i - p_i, 0), \qquad \text{since } d(p,p') = d(p',p), \\ &= \sum_{i=1}^{V}|p_i - p'_i|, \end{aligned}$$

so the variation distance may easily be computed. Autocorrelations were calculated based on the last 5000 iterations of a single run of 10,000 iterations, in order to ensure approximate stationarity of the chain.

Tables 5.9 and 5.10 display the acceptance and movement rates for Models 1 and 2, respectively. As mentioned previously, the acceptance rate for a Hastings algorithm represents the percentage of candidate points which are accepted on average and, thus, it provides some measure of the "efficiency" of the algorithm and the suitability of the choice of CGF. The movement rate, however, measures the percentage of iterates which differ from the previous one, giving an indication of how freely the chain is moving about the target space. In terms of acceptance rates, the more sophisticated independence Hastings algorithm indeed improves on the original algorithm in terms of efficiency and, as would be expected, the rate for the Hastings/Gibbs cycle lies approximately halfway between that for the Hastings algorithm ($r = 6$) and that for the Gibbs sampler

Strategy	Acceptance rate	Movement rate
(a)	100%	77.4%
(b)	82.7%	70.6%
(c)	90.1%	78.0%
(d)	91.3%	74.2%
(e)	95.5%	73.0%
(f)	92.6%	76.1%

Table 5.10: Acceptance and movement rates under Model 2 with $\mathbf{y} = (2\ 2\ 1\ 1\ 2\ 2)$

considered individually. The acceptance rate also appears to show a trend for increasing with decreasing r, which is reasonable since a decreasing number of states are being proposed for simultaneous updating within each block. However, the movement rate is in many ways more revealing. In the multimodal case of Model 1, both independence Hastings algorithms encourage freer movement about the state sequence space than the Gibbs sampler. In the unimodal case of Model 2, it can be seen that the Gibbs sampler exhibits less restricted movement.

We present here the distance measure and autocorrelation results for Model 1 only, since those for Model 2 are similar. Figure 5.5 monitors mean distance measures and corresponding standard deviations (SDs) and illustrates the fact that both independence Hastings algorithms perform nearly as well as independent sampling from the true distribution. Figure 5.6 illustrates the huge improvement achieved by the independence Hastings algorithm over the Gibbs sampler in terms of "closeness" of empirical distributions to the target distribution. Not only are the average distance measures much smaller for the Hastings algorithm ($r = 6$), but the SDs about these averages are also smaller, indicating more rapid mixing within individual chains. The performance of the Hastings/Gibbs cycle seems to mirror that of the dominant component — the independence Hastings algorithm. Figure 5.7 illustrates the increasing improvement in performance achieved with increasing r. However, in models with much larger values of T, we would expect the optimal value of r to be less than T, since acceptance rates for the independence Hastings algorithm will decrease with increasing T making its use at every iteration inefficient. Figure 5.8 displays the autocorrelations at various lags for each of the sampling strategies. The results reinforce the conclusions drawn from the

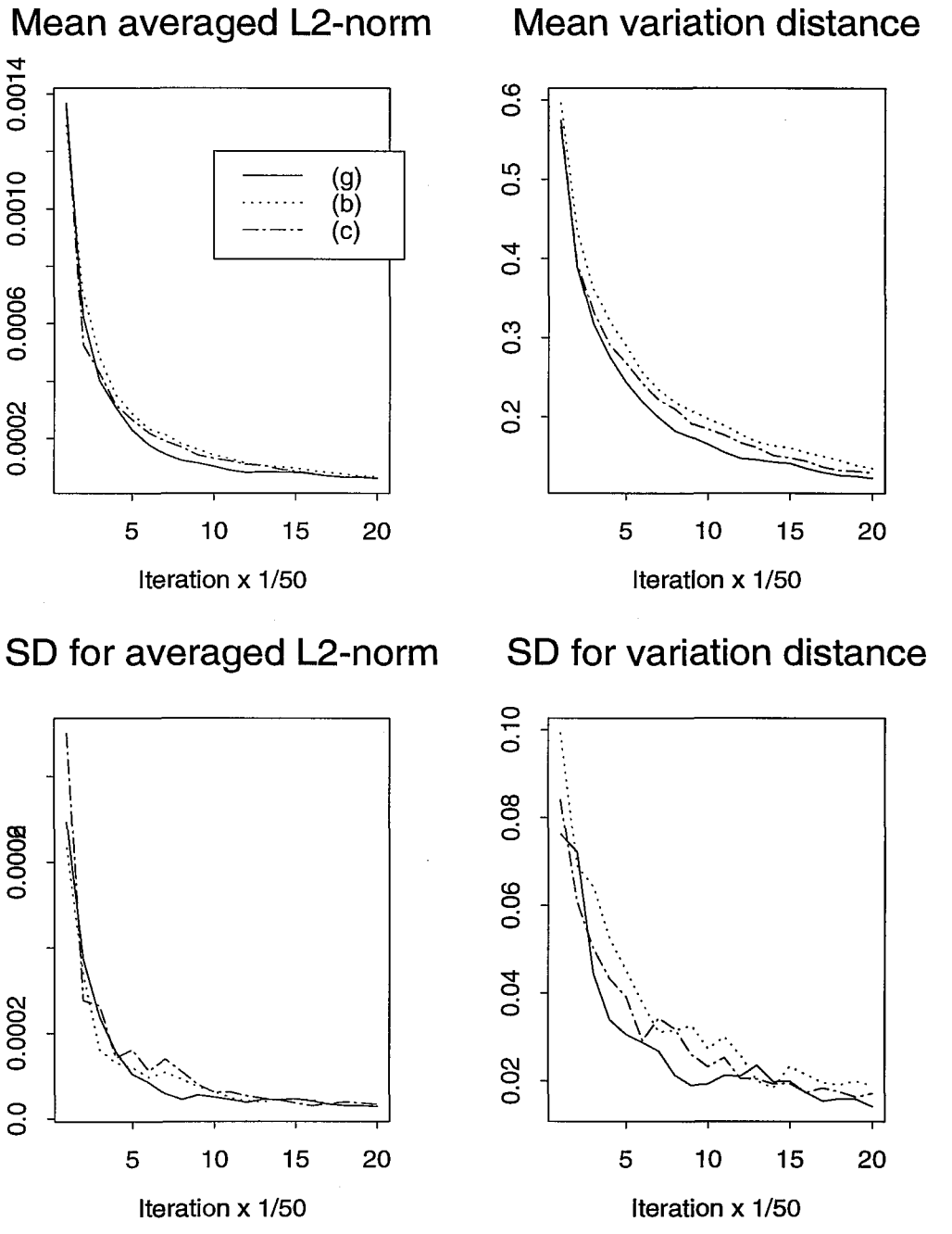


Figure 5.5: Comparison of distance measures for the independence Hastings algorithms

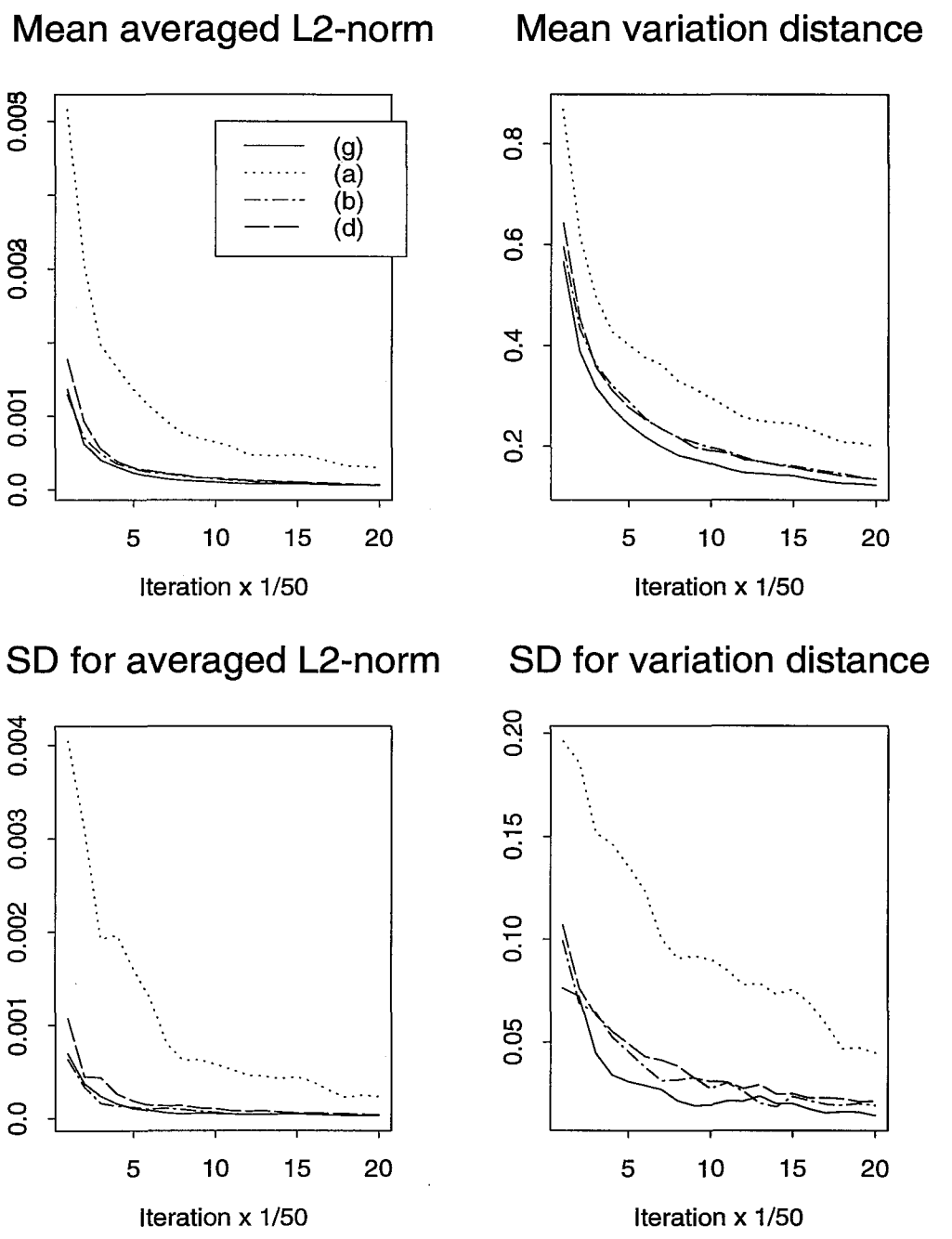


Figure 5.6: Comparison of distance measures for Gibbs sampling and the independence Hastings algorithm

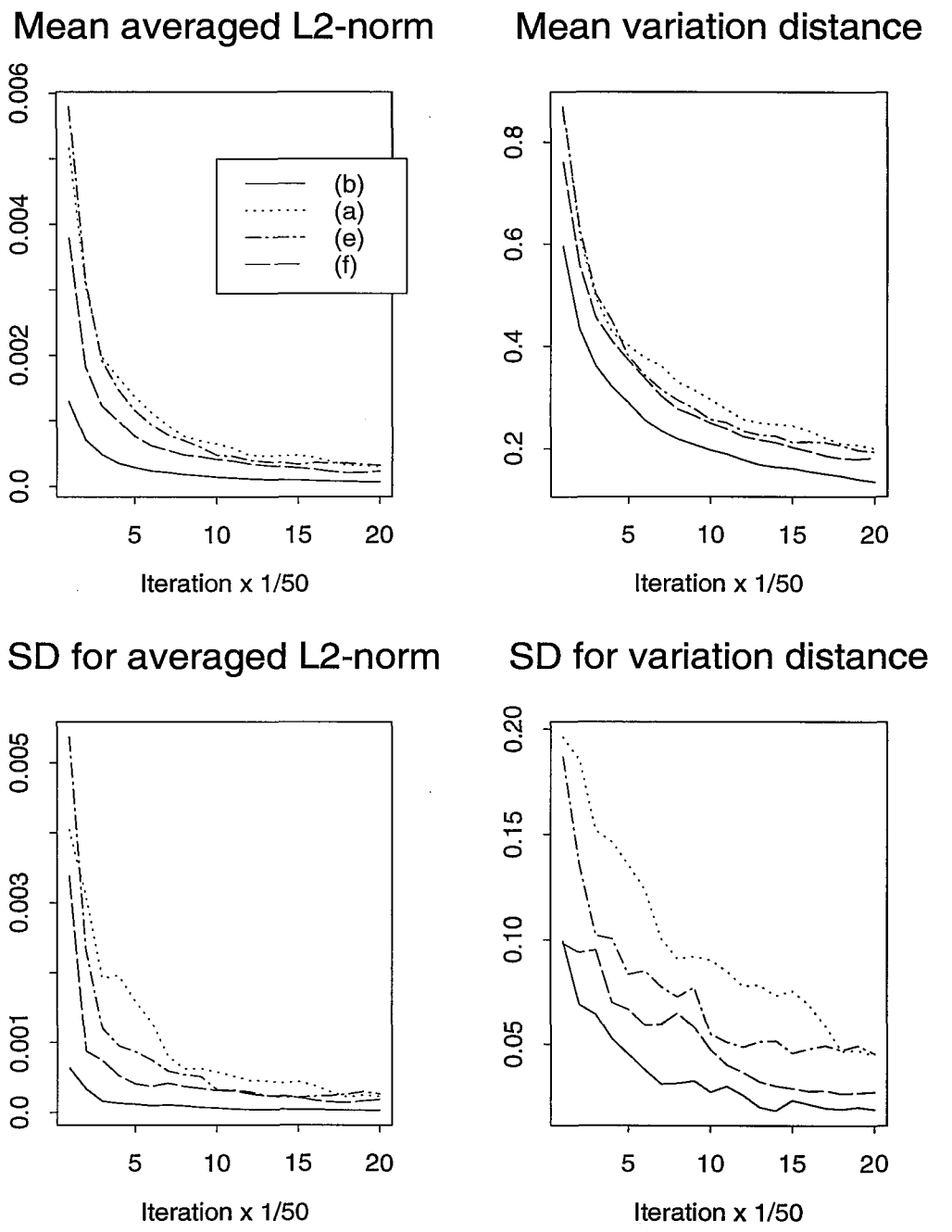


Figure 5.7: Comparison of distance measures for the Hastings algorithm under various specifications of r

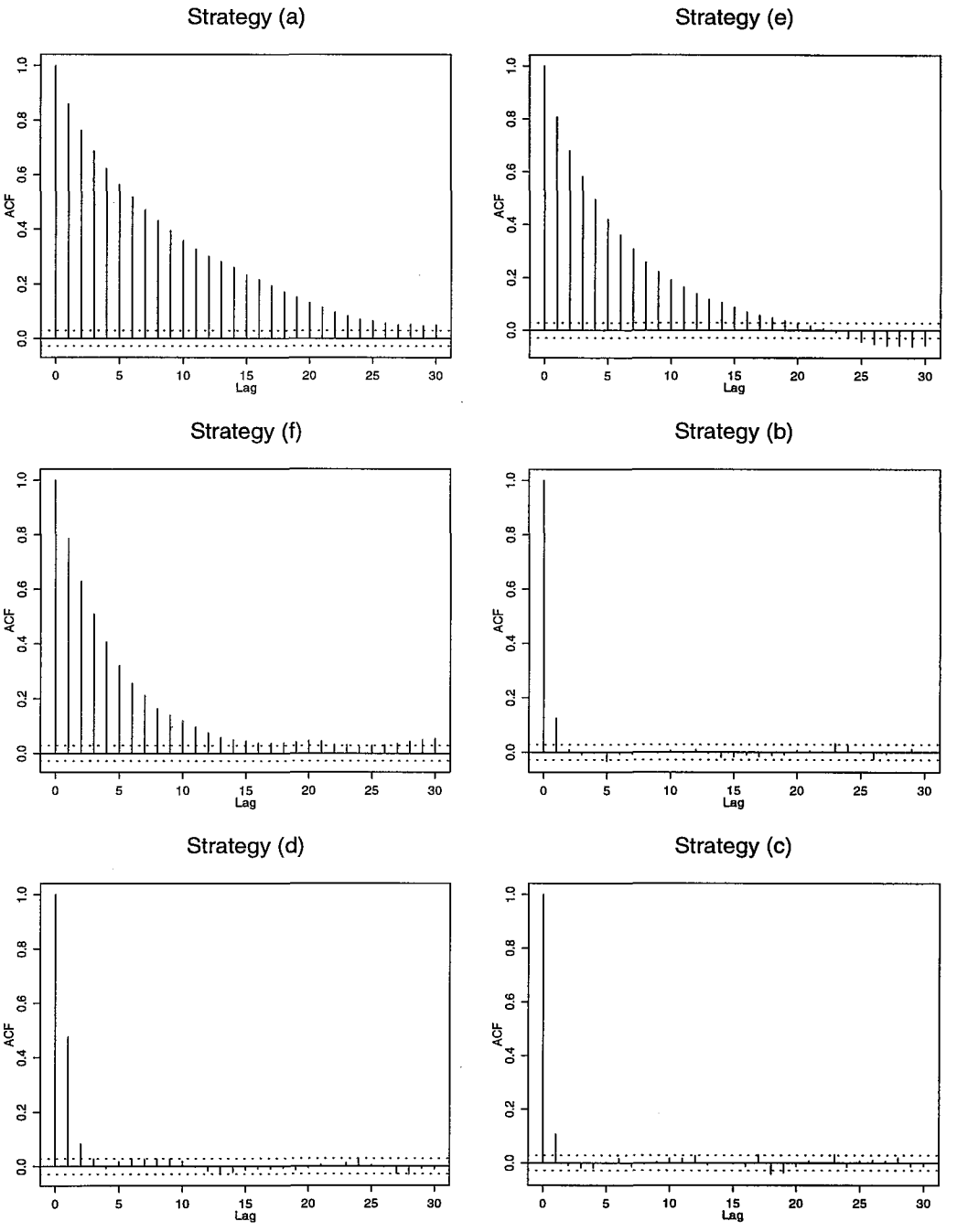


Figure 5.8: Output sample autocorrelations for the various sampling strategies

other performance measures studied. Both independence Hastings algorithms provide virtually uncorrelated output samples, whilst within-chain dependence for the Gibbs sampler ($r = 1$) only becomes negligible after lag 30. For values of r greater than 1, the point at which output sample autocorrelations become negligible is reduced to lag 20 or less.

5.7 Analysis of sampler design in the case of unknown H

Until now, we have concentrated on investigating the mechanism by which to sample from $p(\mathbf{s}|\mathbf{y}, H)$ for a fixed model H. However, the suitability of such a sampling design is ultimately judged by how it improves the properties of the output chain $(H^{(1)}, H^{(2)}, \ldots)$ in the case of an unknown model H. In order to analyse the effect of state sequence sampling scheme on the corresponding output sample for H, a simple example is considered here.

The 3-state L-R model H has finite, discrete observation distributions over 2 symbols. The "true" model H is given by

$$\pi = \begin{pmatrix} 0.70 & 0.20 & 0.10 \end{pmatrix}$$

$$A = \begin{pmatrix} 0.70 & 0.20 & 0.10 \\ 0.00 & 0.70 & 0.30 \\ 0.00 & 0.00 & 1.00 \end{pmatrix}, \qquad B = \begin{pmatrix} 0.20 & 0.80 \\ 0.40 & 0.60 \\ 0.75 & 0.25 \end{pmatrix},$$

which is an 8-parameter model (excluding those parameters fixed by the constraints of row vectors summing to 1, etc). 40 observation sequences of length 6 were generated according to the above model. The model H is now considered "unknown" and it is desired to draw inference about H on the basis of this simulated data.

Thus, we wish to sample from the target distribution, $p(\mathbf{s}, \mathbf{y}|H)$. The sampling scheme to be followed for the parameters is exactly that set out in Section 5.2. Each full sampling strategy to be compared, however, varies in its mechanism for generating the state sequences from $p(\mathbf{s}|\mathbf{y}, H)$ and is labelled accordingly, as follows:

(a) mixture of forward- and backward-cycling Gibbs samplers

(b) independence Hastings algorithm

(c) Hastings/Gibbs cycle

(d) independent sampling from the true distribution

The strategy which employs independent sampling from the true distribution of $p(\mathbf{s}|\mathbf{y}, H)$ at the state sequence generation stage is used as a standard by which the other sampling schemes are judged. Each of the 4 strategies for sampling from $p(\mathbf{s}, H|\mathbf{y})$ was run for 20,000 iterations and this process was replicated 25 times, with all chains being initialised by starting the 40 state sequences at (1 1 1 1 1 1). At each iteration i, 4 full sweeps of the sampling scheme for generating from $p(\mathbf{s}|\mathbf{y}, H^{(i)})$ are completed before $\mathbf{s}^{(i)}$ is recorded, in order to obtain an iterate which is closer to a true drawing from $p(\mathbf{s}|\mathbf{y}, H^{(i)})$. Jeffreys priors (which attempt to represent "neutrality" with regard to prior parameter values) are adopted for all the model parameters, and this corresponds to using Dirichlet priors of the form $\text{Dir}(\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$ (for a derivation, see Ex. 5.21 of Smith & Bernardo (1994)).

Figures 5.9 and 5.10 monitor mean ergodic averages and associated SDs, respectively, over all replications for each of the sampling strategies. Averages for 4 of the 8 parameters are given, although similar conclusions can be drawn from the remaining parameters. In the case of π_1 and a_{11}, the mean ergodic average for both the Hastings algorithm and the Hastings/Gibbs cycle stabilises very quickly, whereas that for the Gibbs sampler does not stabilise before iteration 15,000. The ergodic averages for b_{11} seem to stabilise much faster for all the sampling strategies, although that for the Gibbs sampler still does not quite converge to the average obtained from independent sampling (which we assume to be the "best" strategy). Not only that, but the standard deviations for the Gibbs sampler in this case are much larger than for the other strategies, indicating that individual chains are not converging to this mean as quickly. The graph plotting ergodic averages for b_{31} illustrates a situation where the Hastings algorithm and Gibbs sampler taken individually stabilise at approximately the same point (around iteration 10,000), although the SDs about these averages are again much larger for the Gibbs sampler. However, the Hastings/Gibbs cycle stabilises almost immediately and, therefore, may offer a more robust sampling strategy than any one of its components taken individually.

Figures 5.11 and 5.12 display the within-chain autocorrelation structure for the quantities $\text{Trace}(A) = \sum_{i=1}^{N} a_{ii}$ and $\sum_{i=1}^{N} b_{i1}$, respectively. These quantities are chosen in or-

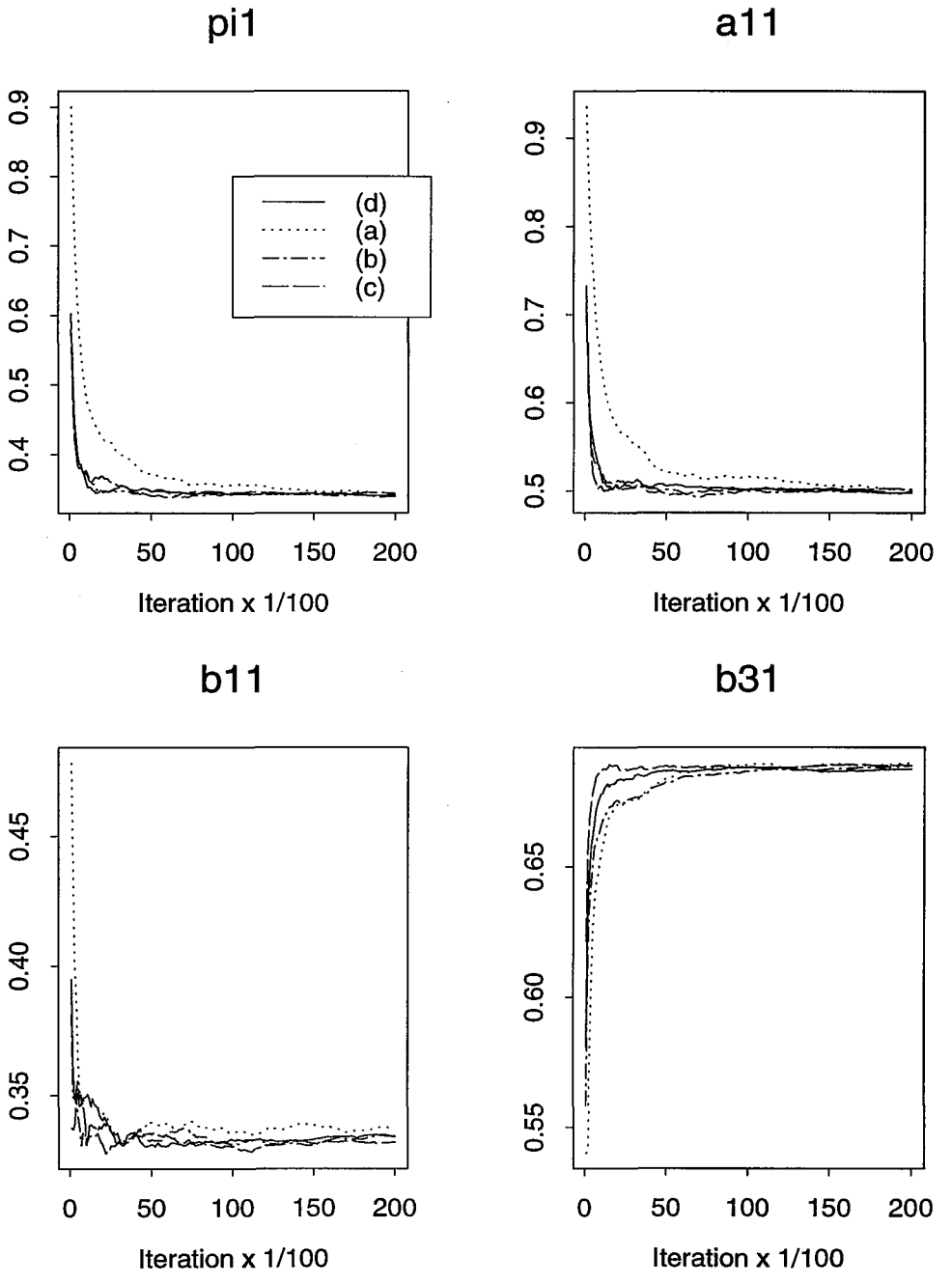


Figure 5.9: Mean ergodic averages for 4 parameters of the unknown model H

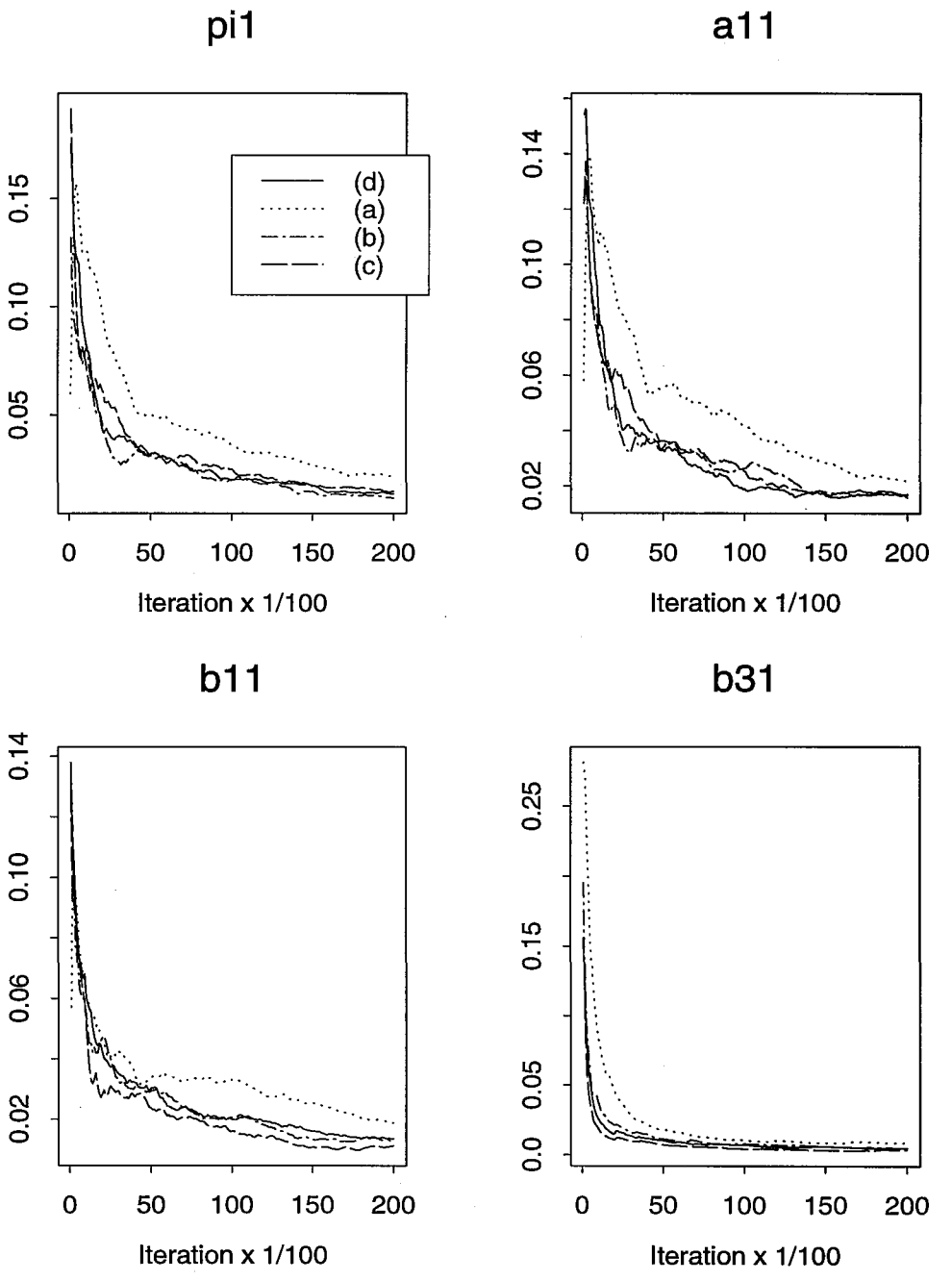


Figure 5.10: SDs about mean ergodic averages for 4 of the unknown model parameters

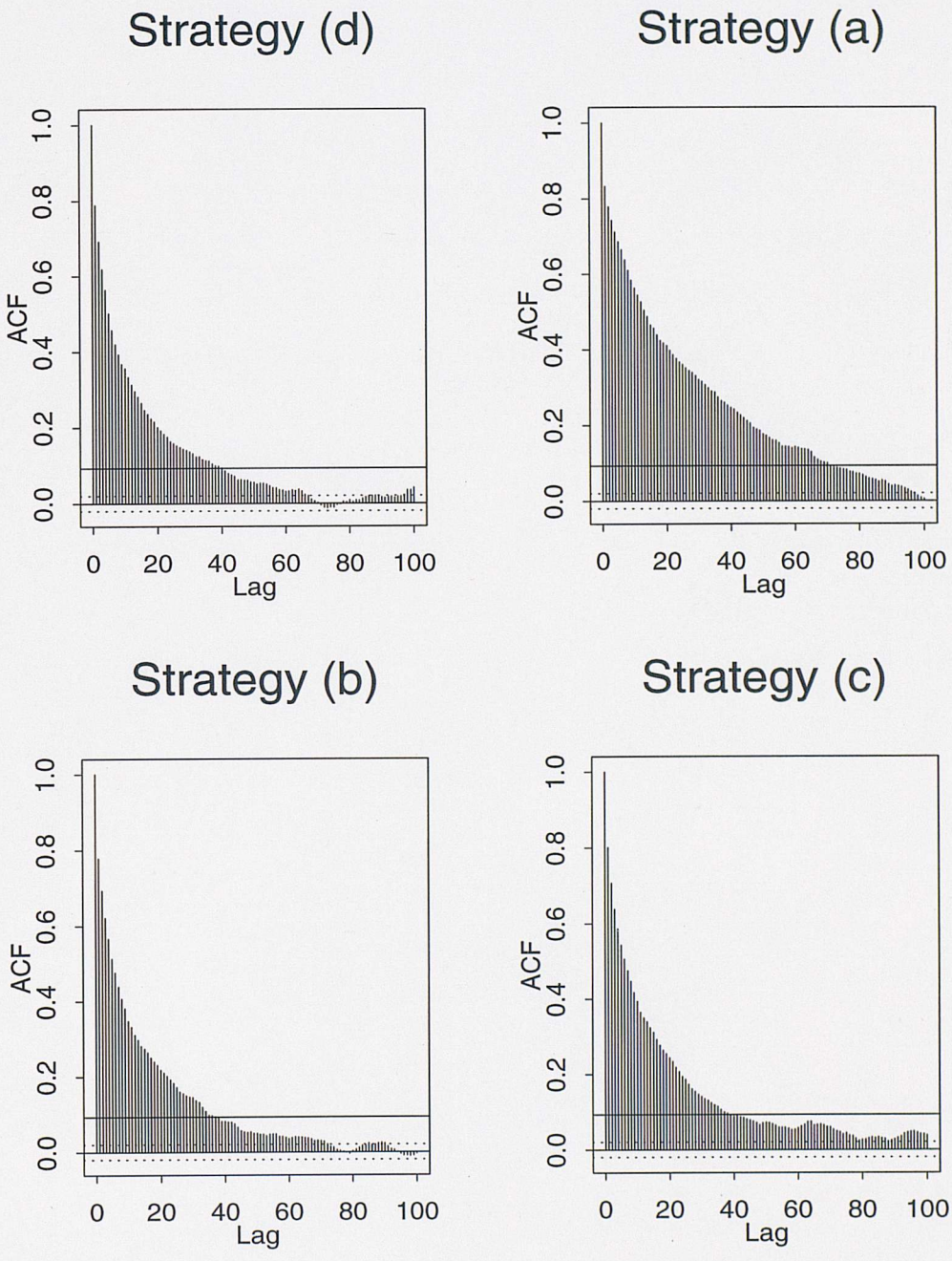


Figure 5.11: Autocorrelation plots for Trace(A)

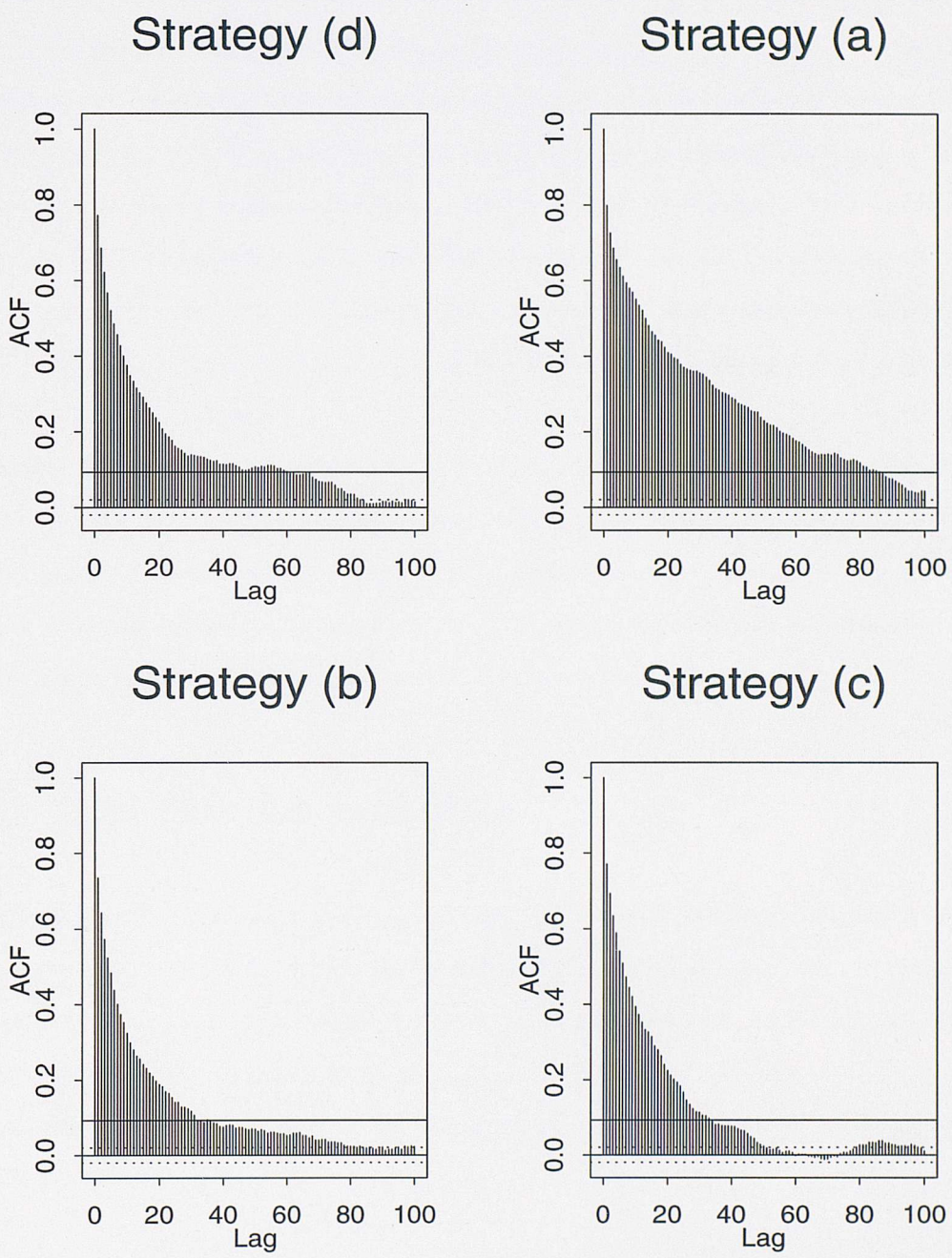


Figure 5.12: Autocorrelation plots for $\sum_{i=1}^{N} b_{i1}$

der to summarise the information provided by the output samples from all parameters of A and B. When adopting Strategy (d) which generates $\mathbf{s}$ directly and independently from $p(\mathbf{s}|\mathbf{y}, H)$, the dependence exhibited between successive parameter iterates is wholly attributable to the dependence existing between $\mathbf{s}$ and H. Thus, by comparison with the autocorrelations exhibited under Strategy (d), we may gauge how much extra dependence is introduced into the chains as a result of using the other "approximate" samplers of $p(\mathbf{s}|\mathbf{y}, H)$. It is clear that employing the Hastings algorithm alone or as part of a cycle can result in autocorrelations for the HMM parameters which are close to those obtained under independent sampling of $p(\mathbf{s}|\mathbf{y}, H)$, whilst the dependence within chains using the single-state updating Gibbs sampler for generating from $p(\mathbf{s}|\mathbf{y}, H)$ is much higher. In Figures 5.11 and 5.12, horizontal lines have been added at autocorrelations of 0.1 in order to ease the task of comparison. Thus, for Trace(A), autocorrelations fall below 0.1 after lag 40 when using the Hastings algorithm or the Hastings/Gibbs cycle to sample $p(\mathbf{s}|\mathbf{y}, H)$. However, under the single-state updating Gibbs sampler, the equivalent point is not reached until lag 70. Similarly, for the output of $\sum_{i=1}^{N} b_{i1}$ values, autocorrelations fall below 0.1 after lag 35 for both strategies which use Hastings steps, whilst this occurs only after lag 85 for the Gibbs sampling strategy.

5.8 Time duration modelling

All the ideas which have been discussed in this chapter may be modified to take account of changes in model specification. For example, for the time duration model introduced in Section 1.2.4, the Gibbs sampler scheme of Section 5.2 may be implemented after some minor alterations. To our knowledge, Gibbs sampler analysis of time duration models has not been addressed in the literature and a brief discussion of the procedure involved is presented here.

In the case of a time duration model, there is an additional set of model parameters to be updated, namely the set of duration parameters specifying the state duration distributions, Δ. In this case, $\boldsymbol{\pi}$, A and B are updated as described in Section 5.2 for the standard HMM and, if the prior for Δ is assumed independent of the other parameters, the generation of Δ from its full conditional distribution $p(\Delta|\mathbf{s}, \boldsymbol{\pi}, A, B, \mathbf{y})$ is similar to that for B. For example, if all state duration distributions are discrete and finite over the set $\{1, 2, \ldots, D\}$, then $\Delta = (\delta_{ij})$ denotes the $N \times D$ state duration matrix,

with

$$\delta_{ij} = p(\text{ duration of the } i^{th} \text{ state} = j\ ).$$

As in the finite discrete case for B, if $\boldsymbol{\delta}_i$ denotes row i of Δ and all rows are modelled by independent Dirichlet priors, then the full conditional distribution for $\boldsymbol{\delta}_i$ is also Dirichlet and is generated independently of all other parameters. In brief, for each $i \in \{1, 2, \ldots, N\}$, if $p(\boldsymbol{\delta}_i) = \text{Dir}(\gamma_1^i, \gamma_2^i, \ldots, \gamma_D^i)$ a priori, then $p(\boldsymbol{\delta}_i|\mathbf{s}) \equiv p(\boldsymbol{\delta}_i|\boldsymbol{\sigma}, \mathbf{d})$ is $\text{Dir}(\gamma_1^i + u_{i1}, \gamma_2^i + u_{i2}, \ldots, \gamma_D^i + u_{iD})$, where

$$u_{jk} = \sum_{l=1}^{r} I_{\{\sigma_l = j,\, d_l = k\}}, \qquad j = 1, 2, \ldots, N,\ k = 1, 2, \ldots, D,$$

assuming that $\mathbf{s}$ consists of r distinct state transitions.

Generation of the state sequence $\mathbf{s}$ from $p(\mathbf{s}|\mathbf{y}, H)$ is slightly more complicated than for the original HMM. As in Section 5.2, $\mathbf{s}$ may be generated in sequence, state-by-state, according to

$$\begin{aligned}
s_1 &\sim p(s_1|\mathbf{s}_{(1)}, \mathbf{y}, H) \propto p(\mathbf{s}, \mathbf{y}|H) \\
s_2 &\sim p(s_2|\mathbf{s}_{(2)}, \mathbf{y}, H) \propto p(\mathbf{s}, \mathbf{y}|H) \\
&\vdots \qquad \vdots \\
s_T &\sim p(s_T|\mathbf{s}_{(T)}, \mathbf{y}, H) \propto p(\mathbf{s}, \mathbf{y}|H).
\end{aligned}$$

It is difficult to write out explicitly and compactly the general forms taken by these full conditionals, since they depend on the structure of the current state sequence and the particular state to be updated, but we can illustrate with a specific example.

For a L-R HMM with $N = 4$, $T = 6$ and current state sequence $\mathbf{s} = (1\ 1\ 2\ 2\ 2\ 4)$, the full conditional distributions for s_2 and s_6, say, are proportional to $p(\boldsymbol{\sigma}, \mathbf{d}, \mathbf{y}|H)$ given in Section 1.2.4, where $p(\boldsymbol{\sigma}, \mathbf{d}, \mathbf{y}|H)$ is expressed in terms of $\mathbf{s}$ rather than $(\boldsymbol{\sigma}, \mathbf{d})$, so that:

$$p(s_2 = i|\mathbf{s}_{(2)}, \mathbf{y}, H) \propto \begin{cases} \delta_1(2)\, b_1(y_2)\, \delta_2(3) & ,\ i = 1 \\ \delta_1(1)\, b_2(y_2)\, \delta_2(4) & ,\ i = 2 \\ 0 & ,\ i = 3\ \&\ 4 \end{cases}$$

and

$$p(s_6 = i|\mathbf{s}_{(6)}, \mathbf{y}, H) \propto \begin{cases} 0 & ,\ i = 1 \\ \delta_2(4)\, b_2(y_6) & ,\ i = 2 \\ \delta_2(3)\, a_{23}\, \delta_3(1)\, b_3(y_6) & ,\ i = 3 \\ \delta_2(3)\, a_{24}\, \delta_4(1)\, b_4(y_6) & ,\ i = 4 \end{cases}$$

Thus, by evaluating the full conditional distributions in this way for each state s_t, $t = 1, 2, \ldots, T$, the Gibbs sampler may be implemented to sample from $p(\mathbf{s}, \mathbf{y}|H)$ when H represents a time duration HMM. The problems associated with single-site updating, and demonstrated under the original HMM in Section 5.3.2, are compounded when explicit state durations are modelled. Typically, state durations much greater than 1 are given most weight so that passing through sequences which contain states of short duration, in order to move freely around the state sequence space, may cause difficulties. Issues such as choice of sweep strategy and use of the Hastings algorithm as an alternative for sampling from $p(\mathbf{s}|\mathbf{y}, H)$ may also be addressed for this modified model.

For example, an independence Hastings algorithm may be constructed to generate $\mathbf{s}$ from $p(\mathbf{s}|\mathbf{y}, H)$ as follows. Since specification of $(\boldsymbol{\sigma}, \mathbf{d})$ completely defines $\mathbf{s}$ and vice versa, it may be possible to propose a candidate for $(\boldsymbol{\sigma}, \mathbf{d})$, taking into account the form of $p(\boldsymbol{\sigma}, \mathbf{d}, \mathbf{y}|H)$ given in Section 1.2.4, according to a mechanism of the form

$$
\begin{aligned}
1)\ \sigma_1 &\sim q_1^{\boldsymbol{\sigma}}(\sigma_1) \quad \propto \pi_{\sigma_1} b_{\sigma_1}(y_1) \\
d_1 &\sim q_1^{\mathbf{d}}(d_1|\sigma_1) \quad \propto \delta_{\sigma_1}(d_1) \prod_{t=1}^{d_1} b_{\sigma_1}(y_t), \quad 1 \leq d_1 \leq \min\{D, T\}, \\
2)\ \sigma_2 &\sim q_2^{\boldsymbol{\sigma}}(\sigma_2|\sigma_1, d_1) \quad \propto a_{\sigma_1\sigma_2} b_{\sigma_2}(y_{d_1+1}) \\
d_2 &\sim q_2^{\mathbf{d}}(d_2|\sigma_1, \sigma_2, d_1) \quad \propto \delta_{\sigma_2}(d_2) \prod_{t=1}^{d_2} b_{\sigma_2}(y_{d_1+t}), \quad 1 \leq d_2 \leq \min\{D, T - d_1\}, \\
&\vdots \qquad \vdots
\end{aligned}
$$

Some extra thought may be required when generalising such an approach either to constrained models or to updating in blocks of length $< T$. This may be an area for future research.

5.9 Conclusions

In Chapter 5, particular emphasis has been placed on improving the design of the sampling scheme for simulating from $p(\mathbf{s}, H|\mathbf{y})$. Adopting independent conjugate priors for the model parameters leads to $\boldsymbol{\pi}$ and the rows of A being independently updated according to Dirichlet distributions. Similar independent block updating is possible for the observation distribution parameters if appropriate conjugate priors are adopted which yield standard full conditional distributions that may readily be simulated from.

However, the forward-cycling, single-site updating Gibbs sampler strategy for simulating $p(\mathbf{s}|\mathbf{y}, H)$, which was outlined in Section 5.2 and is advocated in the literature (Robert et al, 1993), can often exhibit slow convergence and high within-chain dependencies, especially for constrained models (as seen in Section 5.3.2). Choosing equally between forward and backward cycling (according to a mixture) should offer a more robust strategy than simple forward- or backward-cycling alone, and updating blocks of more than 1 state simultaneously from their full conditional distributions should improve sampler performance. This may be computationally feasible for constrained models where the number of valid state sequences is restricted. If simultaneous block updating by direct simulation of the exact full conditionals is not an option, then the Hastings algorithm of Section 5.6, which incorporates structure of the HMM in the specification of its CGF, offers the chance for simultaneous updating of many states. This improves sampling from $p(\mathbf{s}|\mathbf{y}, H)$ and, ultimately, yields improved properties for the chain $(H^{(1)}, H^{(2)}, \ldots)$ of interest. Such ideas may also be of use under modified model structures, e.g. the time duration model.

Finally, simultaneously updating blocks of states via the Hastings algorithm seemed particularly useful in the case of the constrained L-R HMM. It may be that such an idea can be utilised in more general ordered parameter models. Appendix C.2 considers an analogous independence Hastings algorithm in the case of an ordered multivariate Normal model.

Conclusions

We provide here a summary of what has been achieved, with regard to the objective of approaching HMM analysis from a fully Bayesian viewpoint, and indicate possible areas requiring further research.

Motivation for the consideration of HMMs in general classification tasks was provided in the context of ASR. Further HMM applications which were considered in detail included handwritten text recognition and a classification problem in pharmaceutical research. However, all the ideas and methodology presented within this thesis are applicable in general.

A comprehensive review was provided of issues arising when HMMs are adopted and standard techniques used in their analysis. Step-by-step proofs of the Viterbi and B-W algorithms were presented, adding detail to various outline "proofs" provided throughout the HMM literature. For the B-W re-estimation procedure, a concatenation of proofs by different authors was presented plus additional material omitted in these papers (such as the maximisation proof of Lemma A.3). A clear description detailing the necessary scaling procedure for B-W re-estimation was then provided, including the main induction proof of Lemma A.4. To our knowledge, such a full documentation has not been presented in the literature. Some mistakes and ambiguity of notation were also identified in the brief derivation of this scaling procedure provided in Rabiner (1989).

The Viterbi algorithm, described in Chapter 2, recovers the MAP state sequence $\mathbf{s}^1$ and its associated joint probability with $\mathbf{y}$, $p^1 = p(\mathbf{s}^1, \mathbf{y}|H)$. In terms of learning about the full posterior distribution $p(\mathbf{s}|\mathbf{y}, H)$ and drawing reliable inference about the unobserved state sequence underlying $\mathbf{y}$, knowledge of $\mathbf{s}^1$ and p^1 alone is inadequate. Thus, a generalisation is necessary, if only to confirm the "optimality" of the MAP sequence (e.g. unimodality of $p(\mathbf{s}|\mathbf{y}, H)$). With reference to Foreman (1993), a generalisation of

the Viterbi algorithm was documented in Chapter 3 and a detailed induction proof given (Lemma 3.1). For arbitrary L, the generalised algorithm extracts the L most probable state sequences maximising $p(\mathbf{s}, \mathbf{y}|H)$ and their assocated joint probabilities with $\mathbf{y}$. It has been shown how combined use of the generalised Viterbi algorithm and the B-W algorithm allows calculation of posterior state sequence probabilities, and the usefulness of the generalised algorithm was demonstrated via some examples. In particular, it offers the flexibility of making sub-optimal choices in classification problems which are more meaningful than the MAP classification; e.g. in the context of handwritten text recognition. A pharmaceutical application of HMMs was also considered and analysed using realised experimental data. A sequential classification procedure was documented and proved in Lemma 3.2 for comparison with the retrospective Viterbi algorithm, which provides the MAP classification. The benefits of the generalised Viterbi algorithm were illustrated; for example, its ability to provide additional posterior information, to indicate the state locations at which MAP classification is most uncertain, to be applied locally to regions of the observation sequence of particular interest, etc.

We turn our attention now to the problem of HMM parameter inference for unknown models. Application of the EM algorithm (through the B-W re-estimation procedure) for ML estimation of HMM parameters is not completely satisfactory even from a classical viewpoint: it is a local maximisation technique, whose performance is highly dependent on initial values, and accuracy measures are unavailable for the resulting point estimates. Although, as part of a Bayesian approach, the EM algorithm may be modified to compute local posterior modes, the procedure suffers the same inadequacies as outlined above and, moreover, modes may not be the desired posterior summary of interest. Now, HMMs are typically highly parametric. In addition, the form of $p(H|\mathbf{y})$ as a sum $\sum_{\mathbf{s}} p(\mathbf{s}, H|\mathbf{y})$ over a very large discrete state sequence space means that its direct computation is unfeasible and, thus, we do not have an explicit expression for $p(H|\mathbf{y})$. Thus, standard integral approximation techniques are unsuitable for implementing Bayesian analysis of $p(H|\mathbf{y})$, and this leads to consideration of the computationally more powerful methods of MCMC simulation.

In Chapter 4, the pitfalls of adopting such MCMC techniques without careful thought about sampler design were discussed with reference to examples provided in the MCMC literature. A detailed account was provided of objectives to be achieved when designing suitable MCMC sampling strategies, geared towards the class of target distributions of

interest, the aim being to improve properties exhibited by sampler output chains. A guide to current design suggestions in the literature was then presented and critically reviewed.

Returning to the question of Bayesian HMM analysis, it was seen in Chapter 5 that inclusion of the state sequence information in the estimation process facilitates implementation of the Gibbs sampler to simulate $p(\mathbf{s}, H|\mathbf{y})$, and, thus, a marginal output sample $(H^{(1)}, H^{(2)}, \ldots, H^{(n)})$ may be obtained. Based on this sample, any posterior summary of interest may be obtained, along with precision estimates, making this approach more flexible than B-W re-estimation. Extensions to the cases of multiple observation sequences and time duration models were also outlined, these being areas which we have not seen addressed elsewhere in the literature.

With regard to practical issues of implementation, the choice of initial values was discussed, one suggestion being to make use of the efficient Viterbi re-estimation procedure when priors are vague or the data is overwhelming ("equivalent to" the classical case). Furthermore, the stage of the Gibbs sampler at which values of $\mathbf{s}$ are generated from $p(\mathbf{s}|\mathbf{y}, H)$ was investigated in depth. This was due to certain inadequacies exhibited in practice by the forward-cycling, single-site updating strategy advocated in the literature, such as slow convergence and restricted movement about the target space, such problems being particularly evident for constrained models. We investigated alternative sweep strategies for the Gibbs sampler updating of state sequences and found a mixture of both forward- and backward-cycling to be more robust than either forward- or backward-cycling used alone. Further improvements were gained by updating blocks of states simultaneously from their exact full conditional distributions. This may be computationally feasible when constraints are imposed and when single-site updating presents most difficulties, e.g. under L-R models. Comparisons were made in terms of the transition matrix eigenvalues of each strategy, a technique which was available due to the simplicity of the examples considered. The form of transition matrix associated with each strategy was derived and reversibility was proved in Lemma 5.1 (a necessary condition to be satisfied for application of the eigenvalue result).

In an effort to incorporate posterior information provided by the generalised Viterbi algorithm in the MCMC sampling strategy, a mixture Hastings algorithm was considered. This was achieved through specification of both local and global CGFs for a

more balanced sampling strategy, the global CGF being constructed using any available distribution-specific information such as mode locations (available through the generalised Viterbi algorithm, in the case of HMMs). Use of the mixture CGF, as opposed to a single local CGF, yielded improved performance when applied to a simple discrete multimodal HMM example; e.g. increased state sequence space coverage and reduced within-chain dependence. However, for more realistic models, improvements will be negligible due to the discrete nature of the target distribution $p(\mathbf{s}|\mathbf{y}, H)$. In a more general setting, though, if the target distribution is continuous and presents difficulties for local updating strategies (e.g. if it is multimodal or highly correlated), then such a mixture Hastings algorithm may be potentially useful. This is a possible area for future research.

Hence, an alternative strategy for sampling from $p(\mathbf{s}|\mathbf{y}, H)$ was sought, leading to the consideration of an independence Hastings algorithm. This incorporates structure from the model, $p(\mathbf{s}, \mathbf{y}|H)$, in the specification of its CGF and generates candidate sequences in a sequential fashion, independent of the current state sequence. It was shown that, for fixed H, use of this independence Hastings algorithm to sample state sequences from $p(\mathbf{s}|\mathbf{y}, H)$ yielded improved properties of the output chain; i.e. more rapid convergence and lower within-chain dependence. Furthermore, in the case of an unknown model H, improvements in sampling from $p(\mathbf{s}|\mathbf{y}, H)$ were shown, ultimately, to improve sampling from $p(H|\mathbf{y})$, as was desired. Now, the independence algorithm and Gibbs sampler are simply special cases of a general Hastings algorithm which simultaneously updates blocks of r states at a time. Thus, more robust strategies may be achieved by cycling schemes with different values of r, as was illustrated by the scheme cycling independence Hastings steps ($r = T$) and Gibbs sampler steps ($r = 1$). For large values of T, the problems of single-site updating will be further compounded; however, it will probably not be efficient to use the independence CGF at every iteration and greater use of values of $r < T$ will be necessary. The importance of alternative sampling schemes in the area of duration modelling was also highlighted, possibly an area for further research. Thus, in Chapter 5, we illustrated how to improve sampler performance in the particular case of HMMs by giving extra thought to the design of strategies, as was hypothesised in Chapter 4.

Problems analogous to those exhibited when sampling state sequences under L-R HMMs exist for any continuous model with ordered variables. Appendix C.2 indicates that the block-updating Hastings algorithm idea may be more widely applicable; e.g. for

multivariate Normal distributions with ordered variables. Uniform ergodic convergence of the proposed independence Hastings algorithm was proved when implemented with this constrained multivariate Normal model. Application of such ideas under more general models may be an area worthy of future investigation.

Appendix A

Appendix to Chapter 2

A.1 The B-W re-estimation procedure

We present here one proof of the B-W re-estimation procedure in the case of discrete, finite observation distributions. The proof follows closely that outlined in Levinson et al (1983), but extra details are provided which are not fully explained in their paper. First, we state some results which are subsequently required for the proof.

Lemma A.1

Let u_i, $i = 1, 2, \ldots, k$, be positive real numbers and let v_i, $i = 1, 2, \ldots, k$, be non-negative real numbers such that $\sum_i v_i > 0$. Then

$$\log\left(\frac{\sum_i v_i}{\sum_i u_i}\right) \geq \frac{1}{\sum_j u_j}\left(\sum_i u_i \log v_i - u_i \log u_i\right).$$

Proof

This result is proved in Lemma 1 of Levinson et al (1983) using Jensen's inequality for concave functions (a fact which the authors fail to include in their proof). □

Lemma A.2

Let the function $f(x_1, x_2, \ldots, x_k)$ have a stationary point at $(x_1^*, x_2^*, \ldots, x_k^*)$ and consider the quadratic form $F(z_1, z_2, \ldots, z_k)$ given by

$$F(z_1, z_2, \ldots, z_k) = \sum_{i=1}^{k} \sum_{j=1}^{k} \delta_{ij} z_i z_j,$$

where

$$\delta_{ij} = \frac{\partial^2 f}{\partial x_i \, \partial x_j}, \quad \text{evaluated at the stationary point } (x_1^*, x_2^*, \ldots, x_k^*).$$

Then, if F is a negative definite quadratic form in the variables $(z_1, z_2, \ldots, z_k)$, i.e.

$$F(z_1, z_2, \ldots, z_k) \leq 0, \quad \forall z_i, \qquad i = 1, 2, \ldots, k,$$

and

$$F(z_1, z_2, \ldots, z_k) = 0 \iff z_i = 0, \qquad i = 1, 2, \ldots, k$$

this implies that $(x_1^*, x_2^*, \ldots, x_k^*)$ gives a maximum for f.

Proof

The proof of this result is straightforward and is provided in Sections 12 and 13 of Hancock (1917). □

Lemma A.3

If $\{c_i : i = 1, 2, \ldots, k\}$ is a set of positive real numbers then, subject to the constraint that $\sum_i x_i = 1$, the function

$$f(x_1, x_2, \ldots, x_k) = \sum_i c_i \log x_i$$

attains its maximum when

$$x_i = \frac{c_i}{\sum_j c_j}, \qquad i = 1, 2, \ldots, k.$$

Proof

We expand here on the proof given for Lemma 2 in Levinson et al (1983) and, in addition, prove the critical point to be a maximum, which the authors omit in their paper.

Proof is by the method of Lagrange multipliers. The constraint may be expressed as

$$\sum_i x_i - 1 = 0.$$

Thus,

$$f(x_1, x_2, \ldots, x_k) = \sum_i c_i \log x_i - \lambda \left(\sum_i x_i - 1 \right), \tag{A.1}$$

where λ denotes the Lagrange multiplier. At stationary points,

$$\frac{\partial}{\partial x_i} f(x_1, x_2, \ldots, x_k) = 0, \qquad i = 1, 2, \ldots, k.$$

Therefore, differentiating f as expressed in Equation (A.1) with respect to x_i, for each value of $i = 1, 2, \ldots, k$, and equating to zero gives

$$\begin{aligned} \frac{c_i}{x_i} - \lambda &= 0 \qquad (A.2) \\ \Longrightarrow c_i - \lambda x_i &= 0 \\ \Longrightarrow \sum_i c_i - \lambda \sum_i x_i &= 0, \\ \Longrightarrow \lambda &= \sum_i c_i, \qquad \text{by the constraint that } \sum_i x_i = 1. \end{aligned}$$

Substituting this value for λ in Equation (A.2) establishes that a stationary point of $f(x_1, x_2, \ldots, x_k)$ occurs at

$$x_i = \frac{c_i}{\sum_j c_j}, \qquad i = 1, 2, \ldots, k.$$

To prove that this stationary point is a maximum, we apply the result stated in Lemma A.2. Now,

$$\frac{\partial^2 f}{\partial x_i \, \partial x_j} = \begin{cases} 0 & , \; i \neq j \\ -\frac{c_j}{x_j^2} & , \; i = j. \end{cases}$$

Thus, at the stationary point when $x_i = c_i / \sum c_j$, $i = 1, 2, \ldots, k$, we have that, for $i = 1, 2, \ldots, k$, $j = 1, 2, \ldots, k$, the δ_{ij}'s (adopting the notation of Lemma A.2) are given by

$$\delta_{ij} = \begin{cases} 0 & , \; i \neq j \\ -\frac{\left(\sum c_j\right)^2}{c_i} & , \; i = j. \end{cases}$$

Thus, we may express the quadratic form F as follows:

$$F(z_1, z_2, \ldots, z_k) = \sum_{i=1}^{k} -\frac{\left(\sum c_j\right)^2}{c_i} z_i^2.$$

It is clear that $F \leq 0$, since it is the negative sum of non-negative quantities (recalling that $c_i > 0,\ i = 1, 2, \ldots, k$). Thus, due to this latter strict positivity condition satisfied by the c_i's, $F = 0 \iff z_i = 0,\ i = 1, 2, \ldots, k$. So, by Lemma A.2, the stationary point identified yields a maximum for f and, hence, Lemma A.3 is proved. □

Theorem (B-W re-estimation procedure)

If current parameter estimates $H = (\pi, A, B)$ are re-estimated as $H' = (\pi', A', B')$ according to

$$\pi'_i = \frac{\alpha_1(i)\beta_1(i)}{\sum_{j=1}^{N} \alpha_T(j)}, \qquad i = 1, 2, \ldots, N, \tag{A.3}$$

$$a'_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(y_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i)\beta_t(i)}, \qquad i = 1, 2, \ldots, N,\ j = 1, 2, \ldots, N \tag{A.4}$$

$$b'_{ij} = \frac{\sum_{t \in \{t : y_t = j\}} \alpha_t(i)\beta_t(i)}{\sum_{t=1}^{T} \alpha_t(i)\beta_t(i)}, \qquad i = 1, 2, \ldots, N,\ j = 1, 2, \ldots, K, \tag{A.5}$$

then

$$l(H') \geq l(H).$$

Proof

In the notation of Lemma A.1, put $k = S$, the number of distinct state sequences of length T admissible under an N-state HMM H, so that $\mathcal{S} = \{\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^S\}$. Then, let u_i be the joint probability of the i^{th} state sequence with the observation sequence $\mathbf{y}$, given model H; so

$$u_i = p(\mathbf{s}^i, \mathbf{y} | H).$$

Similarly, let v_i denote the same joint probability but, instead, given model H', i.e.

$$v_i = p(\mathbf{s}^i, \mathbf{y} | H').$$

Then,

$$l(H) = p(\mathbf{y}|H) = \sum_i p(\mathbf{s}^i, \mathbf{y}|H) = \sum_i u_i$$

and, similarly,

$$l(H') = \sum_i v_i.$$

Substituting all these quantities into Lemma A.1 gives

$$\log \frac{l(H')}{l(H)} \geq \frac{1}{l(H)} [Q(H, H') - Q(H, H)], \tag{A.6}$$

where $Q(\cdot,\cdot)$ is defined by

$$Q(H,H') = \sum_i u_i \log v_i, \tag{A.7}$$

$$Q(H,H) = \sum_i u_i \log u_i. \tag{A.8}$$

Thus, if H is the current model estimate and H' is a new model estimate satisfying

$$Q(H,H') - Q(H,H) \geq 0,$$

then Equation (A.6) implies that

$$\log \frac{l(H')}{l(H)} \geq 0$$

and, thus, that $l(H') \geq l(H)$; i.e. any procedure which re-estimates the current model H to be H' in such a way that the models satisfy

$$Q(H,H') \geq Q(H,H)$$

ensures that the likelihood increases at every re-estimation step, until a critical point is attained when $Q(H,H') = Q(H,H)$. To prove this, we substitute our definitions of u_i and v_i in Equations (A.7) and (A.8) above and we see that $Q(H,H')$ is the quantity

$$Q(H,H') = \sum_{\mathbf{s}} p(\mathbf{s},\mathbf{y}|H)\ \log p(\mathbf{s},\mathbf{y}|H'). \tag{A.9}$$

Thus, at the point when $Q(H,H') = Q(H,H)$,

$$\sum_{\mathbf{s}} p(\mathbf{s},\mathbf{y}|H)\ [\log p(\mathbf{s},\mathbf{y}|H') - \log p(\mathbf{s},\mathbf{y}|H] = 0,$$

substituting from (A.9). This is true for all $\mathbf{y}$ and, since log is a one-to-one function, this implies that

$$\begin{aligned}
p(\mathbf{s},\mathbf{y}|H) &= p(\mathbf{s},\mathbf{y}|H'), \quad \forall \mathbf{s} \\
\Longrightarrow p(\mathbf{s}|\mathbf{y},H)\,p(\mathbf{y}|H) &= p(\mathbf{s}|\mathbf{y},H')\,p(\mathbf{y}|H'), \quad \forall \mathbf{s} \\
\Longrightarrow p(\mathbf{y}|H)\sum_{\mathbf{s}} p(\mathbf{s}|\mathbf{y},H) &= p(\mathbf{y}|H')\sum_{\mathbf{s}} p(\mathbf{s}|\mathbf{y},H'), \quad \text{summing over all } \mathbf{s}, \\
\Longrightarrow l(H) &= l(H').
\end{aligned}$$

Thus, we have shown that if H' satisfies $Q(H,H') = Q(H,H)$, then $l(H') = l(H)$ (which Levinson et al (1983) fail to check in their proof). In Section 3 of Baum et al (1970), it is proved that, if H' is specified to maximise $Q(H,H')$, then the equality $l(H') = l(H)$ holds if and only if $H = H'$ is a critical point (i.e. local maximum, in our case) of $l(\cdot)$.

It, therefore, remains for us to find that specification of H' which maximises $Q(H, H')$, given H.

The sum in (A.9) may be interpreted as the expectation of a function of $(\mathbf{s}, \mathbf{y})$ with respect to $p(\mathbf{s}, \mathbf{y}|H)$, in order to compare it with the quantity Q in the general specification of the EM algorithm (as is done in Section 2.3.2), i.e.

$$Q(H, H') = E_{(\mathbf{s},\mathbf{y})}\left[\log p(\mathbf{s}, \mathbf{y}|H')|\mathbf{y}, H\right].$$

Now, the joint probability $p(\mathbf{s}, \mathbf{y}|H)$ may be expressed as

$$p(\mathbf{s}, \mathbf{y}|H) = \pi_{s_1} b_{s_1}(y_1) \prod_{t=2}^{T} a_{s_{t-1}s_t} b_{s_t}(y_t).$$

Thus, substituting this expansion in Equation (A.9) gives

$$\begin{aligned} Q(H, H') &= \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \log \left\{ \pi'_{s_1} \prod_{t=1}^{T-1} a'_{s_t s_{t+1}} \prod_{t=1}^{T} b'_{s_t}(y_t) \right\} \\ &= \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \log \pi'_{s_1} + \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \sum_{t=1}^{T-1} \log a'_{s_t s_{t+1}} \\ &\quad + \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \sum_{t=1}^{T} \log b'_{s_t}(y_t). \end{aligned} \tag{A.10}$$

Thus, maximisation of $Q(H, H')$ with respect to H' reduces to maximisation of the 3 component terms of Equation (A.10) separately, the first with respect to $\boldsymbol{\pi}'$, the second with respect to A' and the third with respect to B'. We consider the proof for maximising the second term involving the a'_{ij}'s. The other terms are dealt with similarly.

It is required to maximise the function

$$\begin{aligned} \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \sum_{t=1}^{T-1} \log a'_{s_t s_{t+1}} &= \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \left\{ \sum_{i=1}^{N} \sum_{t \in \{t<T : s_t = i\}} \log a'_{i s_{t+1}} \right\} \\ &= \sum_{i=1}^{N} \left\{ \sum_{\mathbf{s}} p(\mathbf{s}, \mathbf{y}|H) \sum_{t \in \{t<T : s_t = i\}} \log a'_{i s_{t+1}} \right\} \end{aligned} \tag{A.11}$$

with respect to A'. So, maximisation of the term involving only A' again reduces to the individual maximisation of N separate components, each corresponding to a value for i in Equation (A.11) above, i.e. each corresponding to a row in the transition matrix.

Consider a fixed value of i and denote by $n_{ij}(\mathbf{s})$ the number of transitions from state i to j occurring in the sequence $\mathbf{s}$. The i^{th} component in the sum of Equation (A.11) may be written as

$$f(a'_{i1}, a'_{i2}, \ldots, a'_{iN}) = \sum_{j=1}^{N} \left\{ \sum_{\mathbf{s}} n_{ij}(\mathbf{s}) p(\mathbf{s}, \mathbf{y}|H) \right\} \log a'_{ij}$$

and, therefore, maximisation of the i^{th} component in Equation (A.11) with respect to $(a'_{i1}, a'_{i2}, \ldots, a'_{iN})$ reduces to maximising f above subject to the constraint that

$$\sum_{j=1}^{N} a'_{ij} = 1, \tag{A.12}$$

since each row in the transition matrix defines a probability distribution. This maximisation may be recognised as being in the form of that considered in Lemma A.3. Thus, substituting

$$\begin{aligned} k &= N \\ x_j &= a'_{ij}, \quad j = 1, 2, \ldots, N \\ c_j &= \sum_{\mathbf{s}} n_{ij}(\mathbf{s})\, p(\mathbf{s}, \mathbf{y}|H) \end{aligned}$$

in Lemma A.3, the maximum of f above, subject to the constraint of Equation (A.12), occurs when

$$a'_{ij} = \frac{\sum_{\mathbf{s}} n_{ij}(\mathbf{s}) p(\mathbf{s}, \mathbf{y}|H)}{\sum_k \sum_{\mathbf{s}} n_{ik}(\mathbf{s}) p(\mathbf{s}, \mathbf{y}|H)}, \qquad j = 1, 2, \ldots, N. \tag{A.13}$$

Up to this point, we have followed the outline of Proof 2.1 in Levinson et al (1983), filling in details they have omitted. The remainder of our proof derives in full the form of re-estimation equations from (A.13) above which is not covered in the aforementioned paper.

Thus, it remains to prove that the form of the re-estimation equations given in Equation (A.13) above is the same as that given in Equation (A.4) of the formal definition of the procedure. Now

$$\sum_{\mathbf{s}} n_{ij}(\mathbf{s})\, p(\mathbf{s}, \mathbf{y}|H) = p(\mathbf{y}|H) \sum_{\mathbf{s}} n_{ij}(\mathbf{s})\, p(\mathbf{s}|\mathbf{y}, H), \tag{A.14}$$

where the summation term in Equation (A.14) by definition denotes the expected number of transitions from i to j in a state sequence of length T, given $\mathbf{y}$ and H; i.e.

$$\sum_{\mathbf{s}} n_{ij}(\mathbf{s})\, p(\mathbf{s}|\mathbf{y}, H) = E(\text{number of transitions from } i \text{ to } j|\mathbf{y}, H),$$

so that

$$a'_{ij} = \frac{E(\text{number of transitions from } i \text{ to } j|\mathbf{y}, H)}{\sum_{k=1}^{N} E(\text{number of transitions from } i \text{ to } k|\mathbf{y}, H)}. \tag{A.15}$$

Now, the expectation $E(\text{number of transitions from } i \text{ to } j|\mathbf{y}, H)$ may be expressed alternatively in terms of the indicator random variable $I_{E_t}(\mathbf{s})$ defined, for $t = 1, 2, \ldots, T-1$, by

$$I_{E_t}(\mathbf{s}) = \begin{cases} 1 & , \text{ if } s_t = i \text{ and } s_{t+1} = j \\ 0 & , \text{ otherwise.} \end{cases}$$

So, fixing i and j,

$$\begin{aligned}
&E(\text{number of transitions from } i \text{ to } j|\mathbf{y}, H) \\
&= \sum_{t=1}^{T-1} E(\text{number of transitions from } i \text{ to } j \text{ at time } t|\mathbf{y}, H) \\
&= \sum_{t=1}^{T-1} \sum_{\mathbf{s}} I_{E_t}(\mathbf{s})\, p(\mathbf{s}|\mathbf{y}, H) \\
&= \sum_{t=1}^{T-1} \sum_{\mathbf{s}\in\{\mathbf{s}:I_{E_t}(\mathbf{s})=1\}} p(\mathbf{s}|\mathbf{y}, H) \\
&= \sum_{t=1}^{T-1} p(s_t = i, s_{t+1} = j|\mathbf{y}, H).
\end{aligned}$$

Now, substituting this in (A.15), we are reduced to proving that the form of a'_{ij} in the re-estimation equations can be expressed as

$$\frac{\sum_{t=1}^{T-1} p(s_t = i, s_{t+1} = j|\mathbf{y}, H)}{\sum_{k=1}^{N}\sum_{t=1}^{T-1} p(s_t = i, s_{t+1} = k|\mathbf{y}, H)}.$$

Multiplying this quantity by $p(\mathbf{y}|H)/p(\mathbf{y}|H)$, we are thus reduced to proving that

$$\frac{\sum_{t=1}^{T-1} \alpha_t(i)a_{ij}b_j(y_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i)\beta_t(i)} = \frac{\sum_{t=1}^{T-1} p(s_t = i, s_{t+1} = j, \mathbf{y}|H)}{\sum_{k=1}^{N}\sum_{t=1}^{T-1} p(s_t = i, s_{t+1} = k, \mathbf{y}|H)}. \tag{A.16}$$

Suppressing dependence on the model H in the notation, we recall from Section 2.3.2 that

$$\begin{aligned}
\alpha_t(i) &= p(\mathbf{y}_{[t]}, s_t = i) \\
\beta_t(i) &= p(\mathbf{y}^{[t+1]}|s_t = i).
\end{aligned}$$

Now, for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, N$,

$$\begin{aligned}
p(s_t = i, s_{t+1} = j, \mathbf{y}) &= p(\mathbf{y}|s_t = i, s_{t+1} = j) \times p(s_t = i, s_{t+1} = j) \\
&= p(\mathbf{y}_{[t]}, y_{t+1}, \mathbf{y}^{[t+2]}|s_t = i, s_{t+1} = j) \\
&\qquad \times p(s_{t+1} = j|s_t = i) \times p(s_t = i) \\
&= p(y_{t+1}|\mathbf{y}_{[t]}, \mathbf{y}^{[t+2]}, s_t = i, s_{t+1} = j) \\
&\qquad \times p(\mathbf{y}_{[t]}, \mathbf{y}^{[t+2]}|s_t = i, s_{t+1} = j) \times a_{ij} \times p(s_t = i) \\
&= p(y_{t+1}|s_{t+1} = j) \times p(\mathbf{y}_{[t]}|\mathbf{y}^{[t+2]}, s_t = i, s_{t+1} = j) \\
&\qquad \times p(\mathbf{y}^{[t+2]}|s_t = i, s_{t+1} = j) \times a_{ij} \times p(s_t = i),
\end{aligned}$$

by the conditional independence assumption of (1.3),

$$
\begin{aligned}
&= b_j(y_{t+1}) \times \frac{p(\mathbf{y}^{[t+2]}, s_{t+1}=j|\mathbf{y}_{[t]}, s_t=i)\; p(\mathbf{y}_{[t]}|s_t=i)}{p(\mathbf{y}^{[t+2]}, s_{t+1}=j|s_t=i)} \\
&\quad \times \frac{p(s_t=i|\mathbf{y}^{[t+2]}, s_{t+1}=j)\; p(\mathbf{y}^{[t+2]}|s_{t+1}=j)}{p(s_t=i|s_{t+1}=j)} \times a_{ij} \times p(s_t=i) \\
&= b_j(y_{t+1}) \times a_{ij} \times \alpha_t(i) \times \beta_{t+1}(j) \\
&\quad \times \frac{p(\mathbf{y}^{[t+2]}, s_{t+1}=j|\mathbf{y}_{[t]}, s_t=i)\; p(s_t=i|\mathbf{y}^{[t+2]}, s_{t+1}=j)}{p(\mathbf{y}^{[t+2]}, s_{t+1}=j|s_t=i)\; p(s_t=i|s_{t+1}=j)}.
\end{aligned}
$$

Now, by the Markov property of (1.2) and the conditional independence assumption of (1.3),

$$
\begin{aligned}
p(\mathbf{y}^{[t+2]}, s_{t+1}=j|\mathbf{y}_{[t]}, s_t=i) &= p(\mathbf{y}^{[t+2]}, s_{t+1}=j|s_t=i) \\
p(s_t=i|\mathbf{y}^{[t+2]}, s_{t+1}=j) &= p(s_t=i|s_{t+1}=j).
\end{aligned}
$$

Therefore,

$$
p(s_t=i, s_{t+1}=j, \mathbf{y}) = \alpha_t(i)\, a_{ij}\, b_j(y_{t+1})\, \beta_{t+1}(j). \tag{A.17}
$$

So, by (A.17) above, proving (A.16) reduces to proving that

$$
\sum_{k=1}^{N} p(s_t=i, s_{t+1}=k, \mathbf{y}) = \alpha_t(i)\beta_t(i). \tag{A.18}
$$

Now,

$$
\begin{aligned}
\sum_{k=1}^{N} p(s_t=i, s_{t+1}=k, \mathbf{y}) &= p(s_t=i, \mathbf{y}) \\
&= p(s_t=i, \mathbf{y}_{[t]}, \mathbf{y}^{[t+1]}) \\
&= p(\mathbf{y}^{[t+1]}|s_t=i, \mathbf{y}_{[t]})\, p(s_t=i, \mathbf{y}_{[t]}) \\
&= p(\mathbf{y}^{[t+1]}|s_t=i)\, p(s_t=i, \mathbf{y}_{[t]}),
\end{aligned}
$$

since, by the conditional independence assumption of (1.3), $\mathbf{y}^{[t+1]}$ is independent of $\mathbf{y}_{[t]}$ given s_t. Hence,

$$
\sum_{k=1}^{N} p(s_t=i, s_{t+1}=k, \mathbf{y}) = \beta_t(i)\alpha_t(i).
$$

Thus, (A.18) is proven and, hence, (A.4) is proven. The re-estimation equations for π and B, (A.3) and (A.5) respectively, are established in a similar fashion. □

A.2 The B-W scaling procedure

Here, we outline the scaling procedure employed to prevent underflow in the B-W re-estimation equations. Let $\tilde{\alpha}_t(i)$ and $\tilde{\beta}_t(i)$ denote the rescaled values of $\alpha_t(i)$ and $\beta_t(i)$,

respectively, which are chosen with the aim of keeping all recursive quantities within the dynamic range of the computer.

We first recall the recursive equation of the B-W algorithm given, for $i = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T-1$, by

$$\alpha_{t+1}(i) = \sum_{j=1}^{N} [\alpha_t(j)\, a_{ji}]\ b_i(y_{t+1}). \tag{A.19}$$

At each timepoint t in the scaled algorithm, instead of calculating $\alpha_{t+1}(i)$ recursively through (A.19) directly, the sum in (A.19) is evaluated based on the scaled $\tilde{\alpha}_t(j)$'s and then weighted. Thus, we may define the scaled $\alpha_t(i)$'s recursively, for $i = 1, 2, \ldots, N$, by

$$\begin{aligned} \tilde{\alpha}_1(i) &= \frac{\alpha_1(i)}{\sum_{j=1}^{N} \alpha_1(j)}, \\ \tilde{\alpha}_t(i) &= \frac{\sum_{j=1}^{N} \tilde{\alpha}_{t-1}(j) a_{ji} b_i(y_t)}{\sum_{k=1}^{N} \sum_{j=1}^{N} \tilde{\alpha}_{t-1}(j) a_{jk} b_k(y_t)}, \qquad t = 2, 3, \ldots, T, \end{aligned}$$

so that, if c_t denotes the scale factor at time t, then

$$c_1 = \frac{1}{\sum_{j=1}^{N} \alpha_1(j)} \tag{A.20}$$

$$c_t = \frac{1}{\sum_{k=1}^{N} \sum_{j=1}^{N} \tilde{\alpha}_{t-1}(j) a_{jk} b_k(y_t)}, \qquad t = 2, 3, \ldots, T. \tag{A.21}$$

Thus, the re-scaled quantities satisfy

$$\sum_{j=1}^{N} \tilde{\alpha}_t(j) = 1, \quad \forall t. \tag{A.22}$$

We now prove a useful lemma.

Lemma A.4

For $t = 1, 2, \ldots, T$ and $i = 1, 2, \ldots, N$,

$$\tilde{\alpha}_t(i) = \left[\prod_{j=1}^{t} c_j \right] \alpha_t(i). \tag{A.23}$$

Proof

Proof is by induction on t. For $i = 1, 2, \ldots, N$,

$$\tilde{\alpha}_1(i) = c_1\, \alpha_1(i), \qquad \text{by definition,}$$

so the result is true for $t = 1$. For the inductive step, we assume Lemma A.4 to be true for time t, i.e.

$$\tilde{\alpha}_t(i) = \left[\prod_{j=1}^{t} c_j\right] \alpha_t(i), \qquad i = 1, 2, \ldots, N. \tag{A.24}$$

Putting $t = t + 1$, we are required to prove that

$$\tilde{\alpha}_{t+1}(i) = \left[\prod_{j=1}^{t+1} c_j\right] \alpha_{t+1}(i), \qquad i = 1, 2, \ldots, N. \tag{A.25}$$

So, for $i = 1, 2, \ldots, N$,

$$\begin{aligned}
\tilde{\alpha}_{t+1}(i) &= c_{t+1} \sum_{j=1}^{N} \tilde{\alpha}_t(j) a_{ji} b_i(y_{t+1}), \qquad \text{by definition,} \\
&= c_{t+1} \left[\prod_{k=1}^{t} c_k\right] \sum_{j=1}^{N} \alpha_t(j) a_{ji} b_i(y_{t+1}), \qquad \text{by the induction hypothesis (A.24),} \\
&= \left[\prod_{j=1}^{t+1} c_j\right] \alpha_{t+1}(i), \qquad \text{by (A.19).}
\end{aligned}$$

Thus, (A.25) is proved for $t + 1$ and, hence, by induction on t, Lemma A.4 is proved. □

A similar set of equations exist for the backward probabilities, employing the same scale factors as in (A.20) and (A.21). So, for $i = 1, 2, \ldots, N$:

$$\begin{aligned}
\tilde{\beta}_T(i) &= c_T \\
\tilde{\beta}_t(i) &= c_t \sum_{j=1}^{N} a_{ij} b_j(y_{t+1}) \tilde{\beta}_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1 \\
\tilde{\beta}_t(i) &= \left[\prod_{j=t}^{T} c_j\right] \beta_t(i), \qquad t = 1, 2, \ldots, T
\end{aligned} \tag{A.26}$$

The proof of (A.26) is analogous to that for (A.23), except by backward induction on t, proving the hypothesis to be true for $t - 1$ if it is true for t, etc.

We may then derive expressions for the B-W re-estimation procedure using the scaled α's and β's as follows. For $i = 1, 2, \ldots, N$,

$$\begin{aligned}
\pi_i' &= \frac{\alpha_1(i)\beta_1(i)}{\sum_{j=1}^{N} \alpha_T(j)}, \qquad \text{from (A.3) of the re-estimation procedure,} \\
&= \frac{\alpha_1(i) \sum_{j=1}^{N} a_{ij} b_j(y_2) \beta_2(j)}{\sum_{j=1}^{N} \alpha_T(j)}, \qquad \text{by (2.9) when } t = 1,
\end{aligned}$$

$$= \tilde{\alpha}_1(i) \sum_{j=1}^{N} a_{ij} b_j(y_2) \tilde{\beta}_2(j), \tag{A.27}$$

where (A.27) is obtained by substituting expressions for the original α's and β's in terms of their corresponding re-scaled quantities and recalling (A.22). Scaled computations for the transition matrix and observation probabilities can be obtained by a similar substitution giving, for $i = 1, 2, \ldots, N$,

$$a'_{ij} = \frac{\sum_{t=1}^{T-1} \tilde{\alpha}_t(i) a_{ij} b_j(y_{t+1}) \tilde{\beta}_{t+1}(j)}{\sum_{k=1}^{N} \sum_{t=1}^{T-1} \tilde{\alpha}_t(i) a_{ik} b_k(y_{t+1}) \tilde{\beta}_{t+1}(k)}, \qquad j = 1, 2, , \ldots, N \tag{A.28}$$

$$b'_{ij} = \frac{\sum_{k=1}^{N} \sum_{t \in \{t: y_t = j\}} \tilde{\alpha}_t(i) a_{ik} b_k(y_{t+1}) \tilde{\beta}_{t+1}(k)}{\sum_{k=1}^{N} \sum_{t=1}^{T} \tilde{\alpha}_t(i) a_{ik} b_k(y_{t+1}) \tilde{\beta}_{t+1}(k)}, \qquad j = 1, 2, , \ldots, K. \tag{A.29}$$

Thus, the re-estimation equations under the scaling procedure may be calculated via (A.27), (A.28) and (A.29).

Now,

$$1 = \sum_{i=1}^{N} \tilde{\alpha}_T(i), \qquad \text{by (A.22)},$$

$$= \left[\prod_{j=1}^{T} c_j \right] \sum_{i=1}^{N} \alpha_T(i), \qquad \text{by (A.23) when } t = T,$$

$$= \left[\prod_{j=1}^{T} c_j \right] p(\mathbf{y}|H), \qquad \text{by the B-W algorithm (see Figure 2.2).}$$

Therefore, we obtain an expression from which $p(\mathbf{y}|H)$ may be evaluated:

$$p(\mathbf{y}|H) = \frac{1}{\prod_{t=1}^{T} c_t}.$$

This quantity is then monitored at each iteration of the re-estimation procedure in order to assess convergence. In practice, the small numbers involved will necessitate the monitoring of the equivalent quantity, $\log p(\mathbf{y}|H)$, instead.

Appendix B

Appendix to Chapter 3

B.1 Model parameters for the text recognition example

Given below are the estimated HMM parameters for the text recognition example of Section 3.3.2. If states 1–9 are taken to correspond respectively to the letters A–I in alphabetical order, then the state-dependent observation symbol probabilities are given by

i	b_{ij}	EXCEPTIONS
1	0.0400	
2	0.0053	0.51155 (j=16), 0.36565 (j=20).
3	0.0011	0.91245 (j=1), 0.06225 (j=19).
4	0.0265	0.3640 (j=20).
5	0.0011	0.12013 (j=2), 0.15014 (j=17), 0.70553 (j=22).
6	0.0400	
7	0.0068	0.09395 (j=3), 0.74965 (j=20).
8	0.0016	0.1071 (j=6), 0.2786 (j=23), 0.5791 (j=25).
9	0.0055	0.8680 (j=4).

and π and A are given by

$$\pi = \begin{pmatrix} 0.1252 & 0.1197 & 0.2118 & 0.1210 & 0.0770 & 0.1045 & 0.0688 & 0.0770 & 0.0950 \end{pmatrix}$$

$$A = \begin{pmatrix} 0.0057 & 0.1008 & 0.2027 & 0.2450 & 0.0104 & 0.0522 & 0.1217 & 0.0104 & 0.2511 \\ 0.1925 & 0.0118 & 0.0033 & 0.0017 & 0.6656 & 0.0000 & 0.0000 & 0.0000 & 0.1251 \\ 0.2335 & 0.0000 & 0.0381 & 0.0008 & 0.3316 & 0.0000 & 0.0000 & 0.2481 & 0.1479 \\ 0.1484 & 0.0028 & 0.0037 & 0.0310 & 0.5370 & 0.0010 & 0.0188 & 0.0010 & 0.2563 \\ 0.2058 & 0.0112 & 0.1350 & 0.3723 & 0.1366 & 0.0443 & 0.0390 & 0.0065 & 0.0493 \\ 0.1801 & 0.0000 & 0.0000 & 0.0000 & 0.2757 & 0.1986 & 0.0000 & 0.0000 & 0.3456 \\ 0.1863 & 0.0000 & 0.0000 & 0.0031 & 0.4137 & 0.0000 & 0.0306 & 0.2214 & 0.1450 \\ 0.2059 & 0.0006 & 0.0016 & 0.0009 & 0.6545 & 0.0000 & 0.0000 & 0.0006 & 0.1359 \\ 0.1489 & 0.0321 & 0.3005 & 0.1799 & 0.1712 & 0.0505 & 0.1097 & 0.0008 & 0.0064 \end{pmatrix}.$$

B.2 Model parameters for the EEG data example

The observation distribution parameters used in the EEG data example of Section 3.3.3 are given here (rounded to 3 decimal places). The mean vectors are given by:

$$\begin{aligned} \mu^1 &= (12.100 \;\; 9.881 \;\; 7.172 \;\; 5.193 \;\; 3.790 \;\; 2.810) \\ \mu^2 &= (16.198 \;\; 27.001 \;\; 19.514 \;\; 17.200 \;\; 10.239 \;\; 4.599) \\ \mu^3 &= (18.839 \;\; 16.769 \;\; 14.439 \;\; 11.392 \;\; 5.671 \;\; 3.010) \\ \mu^4 &= (24.344 \;\; 14.861 \;\; 11.918 \;\; 8.886 \;\; 4.769 \;\; 2.734) \\ \mu^5 &= (17.934 \;\; 25.822 \;\; 19.564 \;\; 13.587 \;\; 6.860 \;\; 3.135) \\ \mu^6 &= (9.914 \;\; 12.540 \;\; 10.011 \;\; 6.459 \;\; 4.429 \;\; 3.092) \\ \mu^7 &= (43.804 \;\; 24.930 \;\; 16.319 \;\; 14.451 \;\; 10.257 \;\; 7.546). \end{aligned}$$

In order to display the covariance matrices in a concise and neat fashion, we provide, for each $i = 1, 2, \ldots, 7$, the vector $\Lambda^i = (\Sigma^i_{11}, \Sigma^i_{22}, \ldots, \Sigma^i_{66})$ of variances and the correlation matrix $\Gamma^i = (\Gamma^i_{jk})$ with entries

$$\Gamma^i_{jk} = \frac{\Sigma^i_{jk}}{(\Sigma^i_{jj}\Sigma^i_{kk})^{\frac{1}{2}}}, \quad j = 1, 2, \ldots, 6, \; k = 1, 2, \ldots, 6.$$

Thus, the variance vectors are given by:

$$\begin{aligned} \Lambda^1 &= (4.728 \;\; 5.583 \;\; 3.650 \;\; 1.085 \;\; 0.327 \;\; 0.125) \\ \Lambda^2 &= (9.620 \;\; 23.301 \;\; 23.472 \;\; 21.744 \;\; 4.501 \;\; 0.581) \end{aligned}$$

$$\Lambda^3 = (11.132 \quad 14.215 \quad 10.829 \quad 7.890 \quad 1.196 \quad 0.200)$$

$$\Lambda^4 = (6.067 \quad 6.864 \quad 3.312 \quad 2.790 \quad 0.379 \quad 0.094)$$

$$\Lambda^5 = (10.019 \quad 31.909 \quad 26.260 \quad 9.380 \quad 1.931 \quad 0.219)$$

$$\Lambda^6 = (3.527 \quad 7.921 \quad 8.151 \quad 3.023 \quad 0.369 \quad 0.171)$$

$$\Lambda^7 = (128.798 \quad 129.889 \quad 48.059 \quad 29.716 \quad 14.980 \quad 8.662).$$

Since the correlation matrices are symmetric with all diagonal elements equal to 1, we need only record the above-diagonal entries of the Γ^i's, as follows:

$$\Gamma^1 = \begin{pmatrix} 0.167 & 0.147 & 0.125 & -0.035 & -0.025 \\ & 0.612 & 0.249 & 0.054 & -0.026 \\ & & 0.426 & 0.440 & 0.204 \\ & & & 0.481 & 0.290 \\ & & & & 0.421 \end{pmatrix}$$

$$\Gamma^2 = \begin{pmatrix} 0.447 & 0.236 & 0.779 & 0.347 & -0.127 \\ & 0.333 & 0.753 & 0.330 & 0.615 \\ & & 0.049 & 0.745 & 0.284 \\ & & & 0.247 & 0.242 \\ & & & & 0.249 \end{pmatrix}$$

$$\Gamma^3 = \begin{pmatrix} 0.048 & 0.099 & 0.133 & 0.110 & 0.025 \\ & 0.674 & 0.587 & 0.595 & 0.491 \\ & & 0.630 & 0.541 & 0.329 \\ & & & 0.629 & 0.392 \\ & & & & 0.535 \end{pmatrix}$$

$$\Gamma^4 = \begin{pmatrix} 0.048 & 0.443 & 0.590 & 0.212 & 0.072 \\ & 0.514 & 0.337 & 0.589 & 0.225 \\ & & 0.590 & 0.427 & 0.116 \\ & & & 0.114 & 0.185 \\ & & & & 0.053 \end{pmatrix}$$

$$\Gamma^5 = \begin{pmatrix} 0.313 & 0.284 & 0.335 & 0.397 & 0.281 \\ & 0.490 & 0.331 & 0.715 & 0.522 \\ & & 0.608 & 0.652 & 0.450 \\ & & & 0.270 & 0.468 \\ & & & & 0.531 \end{pmatrix}$$

$$\Gamma^6 = \begin{pmatrix} 0.286 & 0.544 & 0.556 & 0.332 & 0.390 \\ & 0.761 & 0.745 & 0.684 & 0.362 \\ & & 0.674 & 0.727 & 0.285 \\ & & & 0.634 & 0.626 \\ & & & & 0.510 \end{pmatrix}$$

$$\Gamma^7 = \begin{pmatrix} 0.570 & 0.576 & 0.670 & 0.510 & 0.426 \\ & 0.715 & 0.934 & 0.655 & 0.733 \\ & & 0.636 & 0.725 & 0.831 \\ & & & 0.758 & 0.647 \\ & & & & 0.768 \end{pmatrix}$$

Appendix C

Appendix to Chapter 5

C.1 State sequence labels

In order to display analysis results from Chapter 5 in histograms etc, the following labelling scheme is adopted for those state sequences admissible under a L-R HMM with $N = 3$ and $T = 6$:

Label	State sequence	Label	State sequence
1	1 1 1 1 1 1	15	1 1 3 3 3 3
2	1 1 1 1 1 2	16	1 2 2 2 2 2
3	1 1 1 1 1 3	17	1 2 2 2 2 3
4	1 1 1 1 2 2	18	1 2 2 2 3 3
5	1 1 1 1 2 3	19	1 2 2 3 3 3
6	1 1 1 1 3 3	20	1 2 3 3 3 3
7	1 1 1 2 2 2	21	1 3 3 3 3 3
8	1 1 1 2 2 3	22	2 2 2 2 2 2
9	1 1 1 2 3 3	23	2 2 2 2 2 3
10	1 1 1 3 3 3	24	2 2 2 2 3 3
11	1 1 2 2 2 2	25	2 2 2 3 3 3
12	1 1 2 2 2 3	26	2 2 3 3 3 3
13	1 1 2 2 3 3	27	2 3 3 3 3 3
14	1 1 2 3 3 3	28	3 3 3 3 3 3

C.2 Simulation of ordered Normal variables

In Section 5.6, the use of an independence Hastings algorithm, or one which updates blocks of r states simultaneously, seemed to improve sampling from $p(\mathbf{s}|\mathbf{y}, H)$, particularly in the case of L-R HMMs. It may be possible, therefore, to apply these ideas in other settings where variables are ordered. The example considered here is that of simulating a multivariate Normal distribution with ordered variables.

Let T denote "transpose" when applied to a matrix or vector (not to be confused with the " T " of the HMM). We suppose the target distribution $\pi(\mathbf{x})$, where $\mathbf{x}^T = (x_1, x_2, \ldots, x_k)$, to be a multivariate Normal distribution, i.e. $\mathbf{x} \sim N_k(\boldsymbol{\mu}, \Sigma)$, restricted to the region $\mathcal{R} = \{\mathbf{x} : x_1 < x_2 < \ldots < x_k\}$, where $\boldsymbol{\mu}$ and Σ are the mean vector and covariance matrix, respectively, of the corresponding unconstrained distribution and are denoted by $\boldsymbol{\mu}^T = (\mu_1, \mu_2, \ldots, \mu_k)$ and $\Sigma = (\sigma_{ij}^2)$. Robert (1992) suggests using the Gibbs sampler to generate from such a distribution, since standard rejection sampling, which generates from the unconstrained distribution and only accepts points which lie in the region $\mathcal{R}$, may be very inefficient when $p(\mathbf{x} \in \mathcal{R})$ is small under the unconstrained distribution. The Gibbs sampler strategy that Robert proposes comprises the sampling of each of the x_i's in order from its full conditional distribution. This reduces to k generations from constrained univariate Normal distributions. Thus, the $(i+1)^{th}$ iteration proceeds as follows:

$$
\begin{array}{lrcll}
1) & x_1^{(i+1)} & \sim & N(\theta_1^{(i+1)}, \sigma_1^2), & x_1^{(i+1)} < x_2^{(i)} \\
2) & x_2^{(i+1)} & \sim & N(\theta_2^{(i+1)}, \sigma_2^2), & x_1^{(i+1)} < x_2^{(i+1)} < x_3^{(i)} \\
 & \vdots & & \vdots & \\
k) & x_k^{(i+1)} & \sim & N(\theta_k^{(i+1)}, \sigma_k^2), & x_{k-1}^{(i+1)} < x_k^{(i+1)}
\end{array}
$$

where, for $j = 1, 2, \ldots, k$,

$$
\begin{aligned}
\theta_j^{(i+1)} &= \mu_j + \Sigma_{j(j)}^T \Sigma_{(j)(j)}^{-1} (\mathbf{x}_{(j)}^{(i+1)} - \boldsymbol{\mu}_{(j)}) \\
\sigma_j^2 &= \sigma_{jj}^2 - \Sigma_{j(j)}^T \Sigma_{(j)(j)}^{-1} \Sigma_{j(j)},
\end{aligned}
$$

letting $\Sigma_{(j)(j)}$ denote the $(k-1) \times (k-1)$ matrix derived from Σ by eliminating its j^{th} row and its j^{th} column, $\Sigma_{j(j)}$ denote the vector of length $(k-1)$ derived from the j^{th} column of Σ by removing the j^{th} row term and letting the transpose of $\mathbf{x}_{(j)}^{(i+1)}$ be denoted by $(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_k^{(i)})$.

Thus, in order to implement this Gibbs sampler strategy, we are required only to be able to sample from constrained univariate Normal distributions. Gelfand et al (1992) describe a method for generating from any constrained univariate disribution, $p(x)$, $x \in [a, b]$, where the cumulative distribution function, $F(x) = \int_{-\infty}^{x} p(z)\, dz$, is available. If a uniform variate, u, is generated from the range [0,1], then $x = F^{-1}[F(a)+u(F(b)-F(a))]$ is a drawing from $p(x)$, $x \in [a, b]$. In this way, the Gibbs sampler may be implemented in the constrained multivariate Normal case of interest here, provided computational procedures (e.g. NAG library routines) are available for evaluating Normal percentiles and Φ-values, where $F \equiv \Phi$ for the Normal distribution.

However, when correlations between the x_j's are high or the probability $p(\mathbf{x} \in \mathcal{R})$ is small under the corresponding unconstrained multivariate Normal distribution, this single-site updating strategy may suffer restricted movement and, thus, slow convergence and/or high within-chain dependence. Possible solutions may be to seek a suitable re-parameterisation or implement random-direction sampling. Alternatively, it may be possible to apply the same ideas as in Section 5.6 to construct an independence Hastings algorithm in order to allow simultaneous updating of all k variables.

Now, if $\pi^*(\mathbf{x})$ denotes the unconstrained multivariate Normal distribution associated with $\pi(\mathbf{x})$, then $\pi^*(\mathbf{x})$ may be expressed in terms of conditional distributions as

$$\pi^*(\mathbf{x}) = \pi^*(x_1)\, \pi^*(x_2|x_1) \ldots \pi^*(x_k|x_{k-1}, x_{k-2}, \ldots, x_1),$$

where all conditional distributions in the above equation are Normal (by standard Normal theory; for instance, see Chapter 3 of Press (1972)). Therefore, under the constrained target distribution $\pi(\mathbf{x})$, an appropriate choice of independence Hastings CGF, $q(\mathbf{x})$, may be one which proposes a candidate $\mathbf{x}$ by

$$q(\mathbf{x}) = q_1(x_1)\, q_2(x_2|x_1) \ldots q_k(x_k|x_{k-1}, x_{k-2}, \ldots, x_1),$$

with

$$\begin{aligned}
q_1(x_1) &\propto \pi^*(x_1), \qquad x_1 \in \Re, \\
q_2(x_2|x_1) &\propto \pi^*(x_2|x_1), \qquad x_2 > x_1, \\
q_3(x_3|x_2, x_1) &\propto \pi^*(x_3|x_2, x_1), \qquad x_3 > x_2, \\
&\vdots \qquad\quad \vdots
\end{aligned}$$

For the purposes of illustration here, a constrained bivariate Normal distribution ($k = 2$) is considered.

For an unconstrained bivariate Normal distribution, the marginal distribution of x_1 is univariate Normal with mean μ_1 and variance σ_{11}^2, denoted $\mathrm{N}(\mu_1, \sigma_{11}^2)$, and the conditional distribution of x_2 given x_1 is $\mathrm{N}\Big(\mu_2 + \rho\frac{\sigma_{22}}{\sigma_{11}}(x_1 - \mu_1),\ \sigma_{22}^2(1-\rho^2)\Big)$, where

$$\rho = \frac{\sigma_{12}^2}{(\sigma_{11}^2\,\sigma_{22}^2)^{\frac{1}{2}}}$$

denotes the correlation between x_1 and x_2. In this case, the Hastings algorithm may be defined to propose a candidate $\mathbf{x}'$ as follows:

$$\begin{aligned}
&1) \quad x_1' \ \sim\ \mathrm{N}(\mu_1, \sigma_{11}^2), \quad x_1' \in \Re, \\
&2) \quad x_2' \ \sim\ \mathrm{N}\left(\mu_2 + \rho\frac{\sigma_{22}}{\sigma_{11}}(x_1' - \mu_1),\ \sigma_{22}^2(1-\rho^2)\right), \qquad x_2' > x_1'.
\end{aligned}$$

However, when $\mu_1 > \mu_2$, the expected value of x_1 under the constrained target distribution will be less than μ_1 and practical experience has shown that proposing x_1' as in Step 1 above, according to a Normal distribution located at μ_1, can result in very low acceptance rates for this Hastings algorithm. Thus, in order to maintain reasonable acceptance rates, it is necessary to amend the above Hastings CGF so that a candidate $\mathbf{x}'$ is proposed as follows:

$$\begin{aligned}
&1) \quad x_1' \ \sim\ \mathrm{N}(\mu_1^*, \sigma_{11}^2), \quad x_1' \in \Re, \\
&2) \quad x_2' \ \sim\ \mathrm{N}\left(\mu_2 + \rho\frac{\sigma_{22}}{\sigma_{11}}(x_1' - \mu_1),\ \sigma_{22}^2(1-\rho^2)\right), \qquad x_2' > x_1',
\end{aligned}$$

where a suitable choice of μ_1^* was found to be

$$\mu_1^* = \min\left(\mu_1,\ \frac{\mu_1 + \mu_2}{2}\right),$$

which locates the Normal distribution proposing x_1' at a point less than μ_1 when $\mu_1 > \mu_2$; otherwise, the location remains at μ_1. This choice of μ_1^*, as a function of μ_1 and μ_2, means that the above Hastings algorithm may be used in a full Gibbs sampler scheme where parameters $\boldsymbol{\mu}$ and Σ are unknown. For a fixed model, as considered here, it may be possible to arrive at a more suitable value for μ_1^* by performing a pilot run and adapting μ_1^* until reasonable acceptance rates for the Hastings algorithm are achieved.

It is now shown that the strategy associated with this Hastings algorithm is uniformly ergodic and, thus, exhibits a strong convergence property.

Lemma C.1

Let $q(\mathbf{x})$ be defined by

$$q(\mathbf{x}) = q_1(x_1)\, q_2(x_2|x_1),$$

where

$$q_1(x_1) = \frac{1}{(2\pi\sigma_{11}^2)^{\frac{1}{2}}} \exp\left[-\frac{1}{2\sigma_{11}^2}(x_1 - \mu_1^*)^2\right], \quad x_1 \in \Re,$$

$$q_2(x_2|x_1) = \frac{g(x_2|x_1)}{\int_{x_1}^{\infty} g(x|x_1)\, dx}, \quad x_2 > x_1,$$

where

$$g(x_2|x_1) = \exp\left[-\frac{1}{2\sigma_{22}^2(1-\rho^2)}\left(x_2 - \left(\mu_2 + \rho\frac{\sigma_{22}}{\sigma_{11}}(x_1 - \mu_1)\right)\right)^2\right].$$

Then, the Hastings algorithm defined by $q(\cdot)$ drives a Markov chain which exhibits uniform ergodic convergence to its stationary distribution.

Proof

We first recall the sufficient condition for uniform ergodic convergence of an independence Hastings algorithm presented in "Hybrid MCMC schemes" of Section 4.5.2; i.e. if the independence Hastings CGF, $f(\mathbf{x})$, is defined such that $\pi(\mathbf{x})/f(\mathbf{x})$ is bounded on the target space $\mathcal{X}$, then this is sufficient to ensure uniform ergodic convergence of the associated MCMC strategy. Now, since $q(\cdot)$, as defined in the statement of the lemma, is clearly an independence Hastings CGF, it remains for us to prove that π/q is bounded on $\mathcal{R}$.

Now, by definition of the bivariate Normal density function,

$$\pi(\mathbf{x}) \propto \exp\left[-\frac{1}{2(1-\rho^2)}\left\{\left(\frac{x_1-\mu_1}{\sigma_{11}}\right)^2 - 2\rho\left(\frac{x_1-\mu_1}{\sigma_{11}}\right)\left(\frac{x_2-\mu_2}{\sigma_{22}}\right) + \left(\frac{x_2-\mu_2}{\sigma_{22}}\right)^2\right\}\right].$$

By simple manipulation of this exponential form, we have that

$$\pi(\mathbf{x}) \propto \exp\left[-\frac{1}{2\sigma_{11}^2}(x_1-\mu_1)^2\right]\; g(x_2|x_1),$$

for $g(x_2|x_1)$ as defined in the statement of the lemma. In addition,

$$q(\mathbf{x}) \propto \exp\left[-\frac{1}{2\sigma_{11}^2}(x_1-\mu_1^*)^2\right]\; \frac{g(x_2|x_1)}{\int_{x_1}^{\infty} g(x|x_1)\, dx}.$$

Therefore,

$$\frac{\pi(\mathbf{x})}{q(\mathbf{x})} \propto \frac{\exp\left[-\frac{1}{2\sigma_{11}^2}(x_1-\mu_1)^2\right]}{\exp\left[-\frac{1}{2\sigma_{11}^2}(x_1-\mu_1^*)^2\right]} \int_{x_1}^{\infty} g(x|x_1)\, dx.$$

This is a function of x_1 alone and to check boundedness, we need only investigate π/q as $x_1 \to \pm\infty$ (since, when x_1 is finite, π/q is also finite). Thus, for positive constants c_1 and c_2, as $x_1 \to \pm\infty$, then

$$\begin{aligned}\frac{\pi(\mathbf{x})}{q(\mathbf{x})} &\to c_1 \frac{\exp\left[-\frac{1}{2\sigma_{11}^2}x_1^2\right]}{\exp\left[-\frac{1}{2\sigma_{11}^2}x_1^2\right]} \int_{x_1}^{\infty} g(x|x_1)\,dx \\ &\to \begin{cases} 0, & \text{as } x_1 \to +\infty \\ c_2, & \text{as } x_1 \to -\infty \end{cases}\end{aligned}$$

using the fact that $g(x_2|x_1)$ is a normalisable function of x_2 over $\Re$ (since it is proportional to a univariate Normal density function) and, thus, its integral over $\Re$ is some finite positive constant. Hence, $\forall \mathbf{x} \in \mathcal{X}$, $\pi(\mathbf{x})/q(\mathbf{x})$ remains finite, thus ensuring the existence of an upper bound.

Furthermore, π/q is clearly bounded below by zero. Thus, π/q is bounded and, hence, the lemma is proved. □

It is easy to see that this result generalises to the k-variate ordered Normal distribution, provided the variance associated with each $q_i(x_i|x_{i-1}, x_{i-2}, \ldots, x_1)$ is equal to (or greater than) the corresponding variance associated with the unconstrained Normal conditional distribution of $\pi(x_i|x_{i-1}, x_{i-2}, \ldots, x_1)$, since this is the factor which ensures the boundedness condition is satisfied.

We now briefly illustrate the beneficial effect of including such independence Hastings steps into the sampling strategy when the target distribution is ordered bivariate Normal. In order to compare the performance of this independence Hastings algorithm with the Gibbs sampling strategy described earlier in this section, the method of Raftery & Lewis (1992), mentioned in Section 4.4, is employed. Let m denote the value of the "warm-up" period (i.e. number of initial transient iterations to discard), let l denote the output sample spacing required in order to obtain an approximately uncorrelated sample and let n denote the required run length of the sampling strategy (minus m) on which to base inference about the target distribution. Then, Raftery & Lewis's method specifies the values of m, l and n required in order to estimate the cumulative distribution function of a univariate target quantile to some specified accuracy (see Raftery & Lewis (1992) for further details). As recommended by the authors, we compute m, l and n for each sampling strategy from a pilot run of 5000 iterations, based on the requirement that

the cumulative probability of the 0.025 target distribution quantile is to be estimated to within ± 0.005 of its true value with probability 0.95.

To simplify notation, we label the sampling strategies being compared as:

(a) Gibbs sampler

(b) independence Hastings algorithm

(c) Hastings/Gibbs cycle.

Small values of m, l and n indicate that the sampling strategy is performing well. A comparison of the strategies, for different values of model correlation ρ, mean $\boldsymbol{\mu}$ and variances $\sigma_{11}^2 = \sigma_{22}^2 = \sigma^2$, is given in Table C.1. Computation times for the Hastings algorithm were slightly higher than for the Gibbs sampler.

It can be seen from the first 3 sub-tables that the acceptance rate of the Hastings algorithm, and thus its efficiency, seems to increase with model correlation ρ. In addition, these 3 sub-tables indicate that the Gibbs sampler performance degrades with increasing ρ (as would be expected), whilst the Hastings algorithm performance improves. It is also evident that the Hastings/Gibbs cycle offers a more robust strategy in terms of yielding reasonable values for l, m and n for all values of ρ; i.e. it provides a balance between the Gibbs sampler and the Hastings algorithm, since when the former performs badly then the latter performs well and vice versa. By comparison of the first and fourth sub-tables, as the ratio of $\mu_1 : \mu_2$ increases (and, thus, the probability $p(\mathbf{x} \in \mathcal{R})$ under the unconstrained bivariate Normal distribution decreases), so the correlation between x_1 and x_2 increases and the Gibbs sampler performs very poorly in comparison with the Hastings algorithm. This point is further illustrated by comparing the second and last sub-tables, when $\rho = 0.9$. The final sub-table also shows that when $p(\mathbf{x} \in \mathcal{R})$ under the unconstrained model is large (as is the case if $\mu_1 << \mu_2$ and the bulk of the unconstrained probability density lies in the constrained region $x_1 < x_2$), then the Hastings algorithm is very efficient in terms of the actual acceptance rates being almost 100%. This is because both the constrained and unconstrained distributions in this case are very similar and we recall that the Hastings CGF is constructed based on the form of conditional distributions under an unconstrained multivariate Normal model. Thus, the closer the target distribution $\pi(\mathbf{x})$, $\mathbf{x} \in \mathcal{R}$, is to an unconstrained bivariate Normal distribution, the better the "approximation" made by $q(\mathbf{x})$ and the closer we are to direct independent sampling from the target distribution.

	Acceptance rate	l	m	n
$\rho = 0,\ \sigma^2 = 1$ or $0.1,\ \boldsymbol{\mu}^T = (1,1)$				
(a)	100%	1	4	4334
(b)	66.1%	1	12	8389
(c)	83.6%	1	6	5292
$\rho = 0.5,\ \sigma^2 = 1$ or $0.1,\ \boldsymbol{\mu}^T = (1,1)$				
(a)	100%	1	7	5725
(b)	78.2%	1	6	5330
(c)	89.2%	1	7	5945
$\rho = 0.9,\ \sigma^2 = 1$ or $0.1,\ \boldsymbol{\mu}^T = (1,1)$				
(a)	100%	2	26	18,394
(b)	89.5%	1	7	5626
(c)	94.8%	1	11	7729
$\rho = 0,\ \sigma^2 = 0.1,\ \boldsymbol{\mu}^T = (4,1)$				
(a)	100%	4	32	26,580
(b)	78.0%	1	3	4104
(c)	89.7%	2	4	7400
$\rho = 0.9,\ \sigma^2 = 1,\ \boldsymbol{\mu}^T = (1,4)$				
(a)	100%	2	14	11,462
(b)	100.0%	1	2	3867
(c)	100.0%	2	4	7714

Table C.1: Comparison of Gibbs sampling and the Hastings algorithm for an ordered bivariate Normal target distribution

This simple example serves to illustrate a few points. Firstly, worst-case performances for the Gibbs sampler are always much worse than worst-case performances for the Hastings algorithm. Furthermore, the Hastings algorithm performs well when the Gibbs sampler performs badly and vice versa. Thus, it is not surprising that the strategy which combines both Gibbs and Hastings steps in a cycle inherits good properties from both its constituent sampling schemes and offers a more robust procedure when the model parameters (ρ, σ^2 and $\boldsymbol{\mu}$) are changing at every iteration; i.e. when the true model is unknown and the parameters are included in the MCMC updating mechanism.

Such ideas were applied further to ordered Normal models of dimension 3 and 4 and some encouraging results (not reported here) were obtained. Performance of the Hastings algorithm is highly dependent on the mechanism by which μ_i^*, the mean of $q_i(x_i|x_{i-1}, x_{i-2}, \ldots, x_1)$, is chosen, since the quality of this determines the efficiency of the Hastings algorithm in terms of its acceptance rate. For larger dimensions, it may be necessary to update $\mathbf{x}$ in blocks of $r < k$, as in the case of long HMM state sequences. Further investigations in this area may be worthwhile.

For other non-Normal ordered-parameter models, where normalising constants for conditional distributions are unknown and/or the full conditionals are not easily sampled from, it may still be appropriate to use the above Normal-based Hastings algorithm in order to achieve simultaneous updating. In this case, we would expect acceptance rates to decrease due to the added discrepancy between constrained univariate Normal CGFs (the $q_i(\cdot)$'s) and non-Normal target distribution.

References

Aarts, E. & Korst, J. (1989), Simulated annealing and Boltzmann machines. *John Wiley & Sons*

Allerhand, M. (1987), Knowledge-based speech pattern recognition. *Kogan Page*

Amit, Y. & Grenander, U. (1991), *Comparing sweep strategies for stochastic relaxation.* Journal of Multivariate Analysis, **37**(2)

Bahl, L., Gopalakrishnan, P.S., Kanevsky, D. & Nahamoo, D. (1989), *Matrix fast match: a fast method for identifying a short list of candidate words for decoding.* Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Glasgow

Bahl, L.R., Jelinek, F. & Mercer, R. (1983), *A maximum likelihood approach to continuous speech recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence, **5**(2)

Baum, L.E. (1972), *An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes.* Inequalities III (edited by O. Shisha), New York Academic Press

Baum, L.E., Petrie, T., Soules, G. & Weiss, N. (1970), *A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains.* Annals of Mathematical Statistics, **41**(1)

Berger, J.O. (1985), Statistical decision theory and Bayesian analysis. *Springer-Verlag*

Besag, J. & Green, P.J. (1993), *Spatial statistics and Bayesian computation.* Journal

of the Royal Statistical Society, B, **55**(1)

Box, G.E.P. & Tiao, G.C. (1973), Bayesian inference for statistical analysis. *Addison-Wesley*

Bratley, P., Fox, B.L. & Schrage, L.E. (1983), A guide to simulation. *Springer-Verlag*

Bristow, G. (1986), Electronic speech recognition: techniques, technology and applications. *Collins*

Churchill, G.A. (1989), *Stochastic models for heterogeneous DNA sequences.* Bulletin of Mathematical Biology, **51**(1)

Cox, D.R. & Hinkley, D.V. (1974), Theoretical statistics. *Chapman & Hall*

Dempster, A.P., Laird, N.M. & Rubin, D.B. (1977), *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society, B, **39**(1)

Devroye, L. (1986), Non-uniform random variate generation. *Springer-Verlag*

Diebolt, J. & Robert, C.P. (1994), *Estimation of finite mixture distributions through Bayesian sampling.* Journal of the Royal Statistical Society, B, **56**(2)

Dupuis, J.A. (1994), *Bayesian estimation of movement probabilities in open populations using hidden Markov chains.* Biometrika (submitted for publication)

Fishman, G.S. (1991), *Choosing warm-up interval and sample size when generating Monte Carlo data from a Markov chain.* Technical Report, Department of Operations Research, University of North Carolina, USA

Foreman, L.A. (1993), *Generalization of the Viterbi algorithm.* IMA Journal of Mathematics Applied in Business & Industry, **4**(4)

Forney, G.D. (1973), *The Viterbi algorithm.* Proceedings of the IEEE, **61**(3)

Frigessi, A., di Stefano, P., Hwang, C.R. & Sheu, S.J. (1993), *Convergence rates of the Gibbs sampler, the Metropolis algorithm and other single-site updating dynamics.* Journal of the Royal Statistical Society, B, **55**(1)

Frigessi, A., Hwang, C.R. & Younes, L. (1992), *Optimal spectral structure of reversible stochastic matrices, Monte Carlo methods and the simulation of Markov random*

fields. Annals of Applied Probability, **2**(3)

Gelfand, A.E. & Smith, A.F.M. (1990), *Sampling-based approaches to calculating marginal densities.* Journal of the American Statistical Association, **85**(410)

Gelfand, A.E., Smith, A.F.M. & Lee, T.M. (1992), *Bayesian analysis of constrained parameter and truncated data problems using Gibbs sampling.* Journal of the American Statistical Association, **87**(418)

Gelman, A. & Rubin, D.B. (1992), *Inference from iterative simulation using multiple sequences.* Statistical Science, **7**(4)

Gelman, A. & Rubin, D.B. (1993), *Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods.* Journal of the Royal Statistical Society, B, **55**(1)

Geman, S. & Geman, D. (1984), *Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images.* IEEE Transactions on Pattern Analysis and Machine Intelligence, **6**(6)

Geyer, C.J. (1991), *Markov chain Monte Carlo maximum likelihood.* Computing Science and Statistics: Proceedings of the 23^{rd} Symposium on the Interface (edited by E.M. Keramides), Interface Foundation, Fairfax Station, Virginia

Geyer, C.J. (1992), *Practical Markov chain Monte Carlo.* Statistical Science, **7**(4)

Geyer, C.J. & Thompson, E.A. (1993), *Annealing Markov chain Monte Carlo with applications to pedigree analysis.* Technical Report, School of Statistics, University of Minnesota, Minneapolis

Gilks, W.R., Clayton, D.G., Spiegelhalter, D.J., Best, N.G., McNeil, A.J., Sharples, L.D. & Kirby, A.J. (1993), *Modelling complexity: applications of Gibbs sampling in medicine.* Journal of the Royal Statistical Society, B, **55**(1)

Gilks, W.R. & Wild, P. (1992), *Adaptive rejection sampling for Gibbs sampling.* Applied Statistics, **41**(2)

Glynn, P.W. & Heidelberger, P. (1992), *Analysis of initial transient deletion for parallel steady-state simulations.* SIAM Journal on Scientific and Statistical Computing, **13**(4)

Green, P.J. (1992), *Discussion of the paper "Constrained Monte Carlo maximum likelihood" by Geyer & Thompson.* Journal of the Royal Statistical Society, B, **54**(3)

Guédon, Y. & Cocozza-Thivent, C. (1990), *Explicit state occupancy modelling by hidden semi-Markov models: application of Derin's scheme.* Computer Speech and Language, 4(2)

Hancock, H. (1917), Theory of maxima and minima. *Ginn & Company*

Hastings, W.K. (1970), *Monte Carlo sampling methods using Markov chains and their applications.* Biometrika, **57**(1)

Hills, S.E. & Smith, A.F.M. (1992), *Parameterization issues in Bayesian inference.* Bayesian Statistics 4 (edited by J.M. Bernardo, J. Berger, A.P. Dawid & A.F.M. Smith), Oxford University Press

Holmes, J.N. (1988), Speech synthesis and recognition. *Van Nostrand Reinhold (UK)*

Huang, X.D. (1992), *Phoneme classification using semi-continuous hidden Markov models.* IEEE Transactions on Signal Processing, **40**(5)

Huang, X.D., Ariki, Y. & Jack, M.A. (1990), Hidden Markov models for speech recognition. *Edinburgh University Press*

Ingrassia, S. (1991), *A geometric bound on the rate of convergence of a Metropolis algorithm.* Technical Report, Department of Mathematics, University of Catania, Italy

Jelinek, F. (1985), *The development of an experimental discrete dictation recognizer.* Proceedings of the IEEE, **73**(11)

Jennison, C. (1992), *Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods.* Journal of the Royal Statistical Society, B, **55**(1)

Juang, B.H. & Rabiner, L.R. (1991), *Hidden Markov models for speech recognition.* Technometrics, **33**(3)

Kundu, A. & Bahl, P. (1988), *Recognition of handwritten script: a hidden Markov model based approach.* Proceedings of the International Conference on Acoustics, Speech and Signal Processing, New York City

Kundu, A., He, Y. & Bahl, P. (1989), *Recognition of handwritten word: first and*

second order hidden Markov model based approach. Pattern Recognition, **22**(3)

Lee, K.F., Hon, H.W. & Reddy, R. (1990), *An overview of the SPHINX speech recognition system.* IEEE Transactions on Acoustics, Speech and Signal Processing, **38**(1)

Lee, K.F. & Mahajan, S. (1990), *Corrective and reinforcement learning for speaker-independent continuous speech recognition.* Computer Speech and Language, **4**(3)

Levinson, S.E. (1986), *Continuously variable duration hidden Markov models for automatic speech recognition.* Computer Speech and Language, **1**(1)

Levinson, S.E., Rabiner, L.R. & Sondhi, M.M. (1983), *An introduction to the applications of the theory of probabilistic functions of a Markov process to automatic speech recognition.* Bell System Technical Journal, **62**(4)

Lin, S., Thompson, E. & Wijsman, E. (1993), *Achieving irreducibility of the Markov chain Monte Carlo method applied to pedigree data.* IMA Journal of Mathematics Applied in Medicine & Biology, **10**(1)

Liporace, L.A. (1982), *Maximum likelihood estimation for multivariate observations of Markov sources.* IEEE Transactions on Information Theory, **28**(5)

Lippmann, R.P. (1987), *An introduction to computing with neural nets.* IEEE ASSP Magazine, **4**(2)

Martin, J.J. (1975), Bayesian decision problems and Markov chains. *Robert E. Krieger*

Menzefricke, U. (1991), *On the performance of the Gibbs sampler for the multivariate normal distribution.* Technical Report, Faculty of Management, University of Toronto, Canada

Merhav, N. & Ephraim, Y. (1991), *Hidden Markov modelling using a dominant state sequence with application to speech recognition.* Computer Speech and Language, **5**(4)

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. & Teller, E. (1953), *Equation of state calculations by fast computing machines.* Journal of Chemical Physics, **21**(6)

Moore, R.K. (1985), *Systems for isolated and connected word recognition.* New Systems and Architectures for Automatic Speech Recognition and Synthesis (edited by R. de Mori & C.Y. Suen), Springer-Verlag

Morgan, B.J.T. (1984), Elements of simulation. *Chapman & Hall*

Müller, P. (1991), *A generic approach to posterior integration and Gibbs sampling.* Technical Report, Department of Statistics, Purdue University, West Lafayette

Mykland, P., Tierney, L. & Yu, B. (1992), *Regeneration in Markov chain samplers.* Technical Report, School of Statistics, University of Minnesota, Minneapolis

Nag, R., Wong, K.H. & Fallside, F. (1986), *Script recognition using hidden Markov models.* Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Tokyo

Phillips, D.B. & Smith, A.F.M. (1993), *Orthogonal random-direction sampling in Markov chain Monte Carlo.* Technical Report, Statistics Section, Department of Mathematics, Imperial College, London

Picone, J. (1990), *Continuous speech recognition using hidden Markov models.* IEEE ASSP Magazine, **7**(3)

Press, S.J. (1972), Applied multivariate analysis. *Holt, Rinehart & Winston*

Qian, W. & Titterington, D.M. (1991), *Estimation of parameters in hidden Markov models.* Philosophical Transactions of the Royal Society of London, A, **337**(1647)

Rabiner, L.R. (1989), *A tutorial on hidden Markov models and selected applications in speech recognition.* Proceedings of the IEEE, **77**(2)

Rabiner, L.R., Juang, B.H., Levinson, S.E. & Sondhi, M.M. (1985a), *Recognition of isolated digits using hidden Markov models with continuous mixture densities.* AT & T Technical Journal, **64**(6)

Rabiner, L.R., Juang, B.H., Levinson, S.E. & Sondhi, M.M. (1985b), *Some properties of continuous hidden Markov model representations.* AT & T Technical Journal, **64**(6)

Rabiner, L.R., Levinson, S.E. & Sondhi, M.M. (1983), *On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition.*

Bell System Technical Journal, **62**(4)

Racine-Poon, A. (1991), *Bayesian methods in medical research.* Unpublished talk presented at the 13^{th} meeting of the International Society for Clinical Biostatistics, Brussels

Raftery, A.E. & Lewis, S. (1992), *How many iterations in the Gibbs sampler?* Bayesian Statistics 4 (edited by J.M. Bernardo, J. Berger, A.P. Dawid & A.F.M. Smith), Oxford University Press

Ripley, B. (1986), Stochastic simulation. *John Wiley & Sons*

Ritter, C. & Tanner, M.A. (1992), *Facilitating the Gibbs sampler: the Gibbs stopper and the griddy-Gibbs sampler.* Journal of the American Statistical Association, **87**(419)

Robert, C. (1990), *Hidden mixtures and Bayesian sampling.* Technical Report, LSTA, Université Paris VI, France

Robert, C.P. (1992), *Simulation of truncated normal variables.* Technical Report, LSTA, Université Paris VI, France

Robert, C.P., Celeux, G. & Diebolt, J. (1993), *Bayesian estimation of hidden Markov chains: a stochastic implementation.* Statistics & Probability Letters, **16**(1)

Roberts, G.O. & Smith, A.F.M. (1994), *Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms.* Stochastic Processes and their Applications, **49**(2)

Ruigt, G.S.F., Van Prosdij, J.N. & Van Delft, A.M.I. (1989), *A large scale high resolution, automated system for rat sleep staging. I: Methodological and technical aspects.* Electroencephalography and Clinical Neurophysiology, **72**, pp. 52–63

Russell, M.J., Moore, R.K. & Tomlinson, M.J. (1986), *Dynamic programming and statistical modelling in automatic speech recognition.* Journal of the Operational Research Society, **37**(1)

Schmeiser, B. (1989), *Simulation experiments.* Technical Report, School of Industrial Engineering, Purdue University, West Lafayette

Schmeiser, B. & Chen, M.H. (1991), *On random-direction Monte Carlo sampling*

for evaluating multidimensional integrals. Technical Report, Department of Statistics, Purdue University, West Lafayette

Smith, A.F.M. (1991), *Bayesian computational methods.* Philosophical Transactions of the Royal Society of London, A, **337**, pp. 369–386

Smith, A.F.M. & Bernardo, J.M. (1994), Bayesian Theory. *John Wiley & Sons*

Smith, A.F.M. & Roberts, G.O. (1993), *Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods.* Journal of the Royal Statistical Society, B, **55**(1)

Stephens, D.A. & Smith, A.F.M. (1993), *Bayesian inference in multipoint gene-mapping.* Annals of Human Genetics, **57**(1)

Swendsen, R.H. & Wang, J.S. (1987), *Nonuniversal critical dynamics in Monte Carlo simulations.* Physical Review Letters, **58**(2)

Tanner, M.A. & Wong, W.H. (1987), *The calculation of posterior distributions by data augmentation.* Journal of the American Statistical Association, **82**(398)

Tierney, L. (1991), *Markov chains for exploring posterior distributions.* Technical Report, School of Statistics, University of Minnesota, Minneapolis

Titterington, D.M. (1990), *Some recent research in the analysis of mixture distributions.* Statistics, **21**(4)

Viterbi, A.J. (1967), *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm.* IEEE Transactions on Information Theory, **13**(2)

Vounatsou, P. & Smith, A.F.M. (1993), *Bayesian analysis of ring-recovery data via Markov chain Monte Carlo simulation.* Technical Report, Statistics Section, Department of Mathematics, Imperial College, London

Wakefield, J. (1993), *Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods.* Journal of the Royal Statistical Society, B, **55**(1)

Wakefield, J.C., Gelfand, A.E. & Smith, A.F.M. (1991), *Efficient generation of random variates via the ratio-of-uniforms method.* Statistics and Computing, **1**, pp. 129–133

Whitt, W. (1989), *The efficiency of one long run versus independent replications in steady-state simulation.* Technical Report, AT & T Bell Laboratories, Murray Hill, New Jersey

Woodland, P.C. (1990), *Isolated word speech recognition based on connectionist techniques.* British Telecom Technology Journal, **8**(2)

Zue, V.W. (1985), *The use of speech knowledge in automatic speech recognition.* Proceedings of the IEEE, **73**(11)