Inference in Switching Linear Dynamical Systems Applied to Noise Robust Speech Recognition of Isolated Digits

Bertrand Mesot [a,b]

IDIAP–RR 08–35

April 2008

[a] IDIAP Research Institute
[b] École Polytechnique Fédérale de Lausanne (EPFL)

IDIAP Research Report 08–35

Inference in Switching Linear Dynamical Systems Applied to Noise Robust Speech Recognition of Isolated Digits

Bertrand Mesot

April 2008

Abstract

Real world applications such as hands-free dialling in cars may have to perform recognition of spoken digits in potentially very noisy environments. Existing state-of-the-art solutions to this problem use feature-based Hidden Markov Models (HMMs), with a preprocessing stage to clean the noisy signal. However, the effect that the noise has on the induced HMM features is difficult to model exactly and limits the performance of the HMM system.

An alternative to feature-based HMMs is to model the clean speech waveform directly, which has the potential advantage that including an explicit model of additive noise is straightforward. One of the most simple model of the clean speech waveform is the autoregressive (AR) process. Being too simple to cope with the nonlinearity of the speech signal, the AR process is generally embedded into a more elaborate model, such as the Switching Autoregressive HMM (SAR-HMM).

In this thesis, we extend the SAR-HMM to jointly model the clean speech waveform and additive Gaussian white noise. This is achieved by using a Switching Linear Dynamical System (SLDS) whose internal dynamics is autoregressive. On an isolated digit recognition task where utterances have been corrupted by additive Gaussian white noise, the proposed SLDS outperforms a state-of-the-art HMM system. For more natural noise sources, at low signal to noise ratios (SNRs), it is also significantly more accurate than a feature-based HMM system.

Inferring the clean waveform from the observed noisy signal with a SLDS is formally intractable, resulting in many approximation strategies in the literature. In this thesis, we present the Expectation Correction (EC) approximation. The algorithm has excellent numerical performance compared to a wide range of competing techniques, and provides a stable and accurate linear-time approximation which scales well to long time series such as those found in acoustic modelling.

A fundamental issue faced by models based on AR processes is that they are sensitive to variations in the amplitude of the signal. One way to overcome this limitation is to use Gain Adaptation (GA) to adjust the amplitude by maximising the likelihood of the observed signal. However, adjusting model parameters without constraint may lead to overfitting when the models are sufficiently flexible. In this thesis, we propose a statistically principled alternative based on an exact Bayesian procedure in which priors are explicitly defined on the parameters of the underlying AR process. Compared to GA, the Bayesian approach enhances recognition accuracy at high SNRs, but is slightly less accurate at low SNRs.

Keywords Single Channel Source Separation, Signal Level Modelling, Autoregressive Process, Switching Linear Dynamical Systems, Approximate Inference, Bayesian Approaches, Variational Approximations, Noise Robustness, Isolated Digit Recognition.

Contents

Chapter 1

Introduction

1.1 Motivation

From the point of view of modelling, it is natural to see a speech signal as a mixture of discrete and continuous processes. On the one hand, we have a discrete number of words or subword units like phonemes, whilst, on the other hand, the continuous waveform results from the excitation signal generated by the vibration of the vocal cords, modulated by the vocal tract. Current state-of-the-art automatic speech recognition (ASR) systems are mainly centred on the modelling of the discrete part. The generic approach they follow can be schematically represented as

$$\begin{tikzcd} \boxed{\begin{array}{c}\text{Clean waveform}\\ y_{1:T}\end{array}} \arrow[r] & \boxed{\begin{array}{c}\text{Feature sequence}\\ \mathbf{o}_{1:N}\end{array}} \arrow[r] & \boxed{\begin{array}{c}\text{Model}\\ p(\mathbf{o}_{1:N})\end{array}} \end{tikzcd}$$

The information carried by the waveform—represented as a sequence $y_{1:T}$ of T samples—is compressed into a shorter sequence of feature vectors $\mathbf{o}_{1:N}$. The principal role of feature extraction is to remove redundancy by integrating information over a window of samples. For instance, the Mel-Frequency Cepstral Coefficients are obtained by performing, for each window, a filterbank analysis which mimics the pitch perception of the human ear. Another role of feature extraction is to remove information irrelevant for recognition; for example, the pitch is generally ignored. The sequence of feature vectors $\mathbf{o}_{1:N}$ is then generally modelled by a *Hidden Markov Model* (HMM)[1]. The HMM is a probabilistic temporal model which associates to a sequence of observations—in our case, a sequence of feature vectors—a sequence of discrete states. To each state corresponds a probability distribution, often called the *emission distribution*, which gives the probability of an observation. The evolution of the states with time is furthermore controlled by a *transition distribution* which gives the probability of switching to a certain state given the current state. The HMM approach is particularly appealing in the context of ASR because a discrete state is a natural representation of a subword unit, like a phoneme, and the transition distribution models the fact that a phoneme is more likely to follow certain phonemes than others.

1.1.1 Consistency

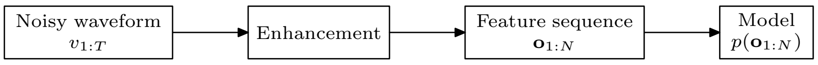
Although HMM-based systems have been very successful, from a modelling point of view, the use of feature makes them inconsistent. Indeed, feature vectors generally include dynamical information, like the first and second temporal derivatives of the static features. This induces correlation between consecutive feature vectors. However this correlation is typically neglected in the HMM. Various attempt have been made to address this issue [2, 16, 27, 36, 53, 56, 63, 65, 64], two notable examples

[1]Though alternatives exist, the most successful ASR systems to date use HMMs. In this thesis, unless stated otherwise, we will always refer to HMM-based ASR system when we talk about state-of-the-art or standard systems.

are the *Segmental HMM* [53] and the *Switching Autoregressive HMM* (SAR-HMM) [27]. The segmental HMM tries to capture the short term correlations which exists between consecutive feature vectors by modelling subsequences of observations; the sequence of feature vectors is split into segments which are then modelled as a whole. However, the segmental approach is not completely satisfactory since continuity is lost at segment boundaries. In contrast, the SAR-HMM does not use features, but models the waveform directly by means of autoregressive (AR) processes. This approach was first proposed by Poritz [56] and recently refined by Ephraim and Roberts [27]. Working with the waveform directly is interesting because it allows the continuous component of the signal to be explicitly modelled, while, with features, continuity must be artificially introduced by adding the temporal derivatives of the static features. From the perspective of ASR, the main drawback of working with waveforms is that a great amount of potentially redundant information is provided to the model. For example, when performing speaker independent speech recognition, the pitch is probably irrelevant. Also, in general, the amplitude of the signal can be ignored. In contrast, features are carefully designed by engineers to remove redundancies and discard information irrelevant for recognition.

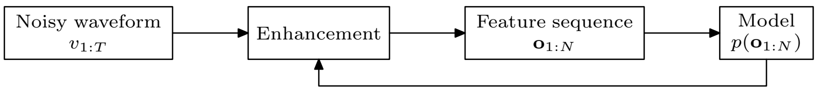
1.1.2 Noise Robustness

Another limitation of current feature-based models is that they are often particularly sensitive to noise [43, 62]. This is a well-known problem which has been extensively addressed in the literature, see for example [28, 14, 34, 43, 51, 62]. Noise in ASR is commonly defined as a source of degradation in the quality of the speech signal or as a difference in the acoustical properties of the signal between the training and testing environments. Typically, distortions caused by a transmission channel, interferences, change in recording conditions or, less evidently, a change of speaker, are all considered as sources of noise. It is also important to differentiate between *convolutional* and *additive* noise. Convolutional noise is generally produced when the signal goes through a linear distortion channel such as the telephone line, while additive noise is produced by the environment and is generally considered to be *additive* and *uncorrelated* with the original signal. The generic approach for dealing with noise in current state-of-the-art ASR systems can be schematically depicted as


Noisy waveform $v_{1:T}$
Enhancement
Feature sequence $\mathbf{o}_{1:N}$
Model $p(\mathbf{o}_{1:N})$

Before being transformed into feature vectors, the noisy speech waveform undergoes an enhancement step which tries to filter out noise. This kind of approach is typically taken by methods based on spectral subtraction [13, 14, 43] or Wiener filtering [4, 26], to deal with additive noise, and channel normalisation techniques [42], to deal with convolutional noise. For example, the *Unsupervised Spectral Subtraction* (USS) algorithm [43] filters out additive noise by exploiting the difference between the energy distributions of speech and noise in the frequency domain. Whilst successful for high ($> 15\,\text{dB}$) to moderate ($\leq 15\,\text{dB}$ and $> 5\,\text{dB}$) Signal to Noise Ratios (SNRs), the quality of filtering is significantly reduced at lower SNRs, where the two distributions overlap.

Current state-of-the-art feature-based models are generally trained on clean data and tested on potentially noisy data. An alternative approach to denoising could therefore be to use the trained model of clean speech during the pre-processing stage. This can be schematically depicted as


Noisy waveform $v_{1:T}$
Enhancement
Feature sequence $\mathbf{o}_{1:N}$
Model $p(\mathbf{o}_{1:N})$

A limitation of this procedure is that the denoising algorithm must deal with two different representations of the signal; the noisy waveform $v_{1:T}$ and the sequence of feature vectors $\mathbf{o}_{1:N}$. As a result, the denoising algorithm becomes dependent on the way features are extracted. This approach is not used in practice, but replaced by a more direct alternative in which the noisy signal is converted into a sequence of noisy features which is then modelled directly. Schematically, we have

$$\begin{tikzcd}\boxed{\begin{matrix}\text{Noisy waveform}\\ v_{1:T}\end{matrix}} \arrow[r] & \boxed{\begin{matrix}\text{Noisy features}\\ \mathbf{o}_{1:N}\end{matrix}} \arrow[r] & \boxed{\begin{matrix}\text{Model}\\ p(\mathbf{o}_{1:N})\end{matrix}}\end{tikzcd}$$

For example, this scheme is used by model-based compensation methods [29, 30, 31, 66] where the original model of clean speech is augmented with an explicit model of noise, and also by multi-stream approaches [15, 33] where many streams containing different types of features are combined to enhance recognition. More recently, this scheme has also been used in [59, 60, 70] where the uncertainty on the feature sequence is explicitly taken into account in the model. Such models extend the standard feature-based models by weighting each feature vector $\mathbf{o}_n$ according to how certain one is that it corresponds to speech. The common problem of most of the methods working with noisy features is that they require training a model of noise or, at least, a good estimation of the SNR [62]. Although in practice this can be done on signal segments where there is no speech signal—provided non-speech segments can be accurately detected—the amount of information required might not be always available. For example, in [60], models are trained on thirty seconds of noise. Multi-stream approaches do not suffer from this limitation. However, their accuracy is strongly dependent on the selection of an appropriate recombination criterion [33]. Another common practice, known as *multi-condition training* [35], is to train the model on noisy features directly. A limitation of this approach is that, in practice, the recognition accuracy depends on the availability of training data covering a sufficiently large number of noisy conditions.

1.1.3 Modelling the Speech Waveform

An alternative to modelling noise is to model the clean speech waveform with a SAR-HMM. The potential benefit of such an approach is that, since an observed *noisy* signal can be viewed as a *clean hidden* signal plus noise, a model trained on clean speech can be readily extended to deal with additive noise, without having to be retrained. This framework naturally leads to the *Switching Linear Dynamical System* (SLDS) [6] which represents the signal as a piecewise linear hidden variable model. Previous applications of the SLDS to ASR (see for example [24, 63]) have modelled the feature vectors and not the speech waveform directly. However, work in acoustic modelling [18] suggests that, provided the difficulties of performing inference and learning can be addressed, the SLDS is a potentially powerful tool for modelling the signal waveform.

1.2 Objectives

The aim of this thesis is to investigate the potential advantages of modelling the speech waveform directly for performing noise robust speech recognition. Our interest is to develop models which, once trained on *clean* data, can be readily used in noisy conditions. We will restrict ourselves to additive noise since this form is straightforward to include in the model. Convolutional noise may also be brought within the framework we discuss, although it is not explicitly considered in our work. Ideally, the proposed models should be more accurate than state-of-the-art feature-based HMM systems in noisy conditions while showing no performance degradation in clean conditions. Since separating additive noise from clean speech, given a noisy speech signal, is an instance of a single channel source separation problem, the models we will consider are potentially applicable to other domains as well, most notably in model guided source separation.

1.2.1 The Switching Autoregressive HMM

Models dealing with the speech waveform directly, like the SAR-HMM, are a potentially valuable alternative to the standard feature-based HMM used in most current ASR systems. The basic idea behind the SAR-HMM is to model the speech waveform as an AR process. The intrinsic non-stationarity of the speech signal is then dealt with by switching between a finite set of AR models (with different parameters). Whilst the SAR-HMM has comparable to state-of-the-art performance on clean speech, its accuracy degrades rapidly under noisy conditions. This is mainly because the underlying AR processes are defined on the potentially noisy signal directly.

1.2.2 Switching Linear Dynamical Systems

To deal with noise without having to train a new model, we go a step further in the modelling of the speech waveform. We propose to extend the SAR-HMM to include an explicit noise process whereby the observed noisy waveform is viewed as a corrupted version of a *clean hidden* waveform. This can be naturally expressed as a SLDS where the internal hidden dynamics is autoregressive. Instead of the noisy observations used in the SAR-HMM case, the proposed switching AR process only models a *clean hidden* counterpart of the observed noisy waveform and is therefore expected to enhance noise robustness. We start by making the simple assumption of additive Gaussian white noise and later extend the AR-SLDS to deal with more complex additive noise sources.

1.2.3 Approximate Inference in the SLDS

Contrary to the SAR-HMM, where inferring the posterior of the hidden variables can be carried out using a standard Forward-Backward algorithm, inference is formally intractable in the SLDS [6], scaling exponentially with the length of the speech waveform. Arguably, this is a possible reason why the SLDS has found relatively little support amongst the automatic speech recognition (ASR) community. Two well-known methods for performing approximate inference in the SLDS are *Expectation Propagation* (EP) [50] and *Generalised Pseudo Bayes* (GPB) [6, 52]. They both suffer from limitations which can be relaxed in the case of the SLDS—see [7] for a detailed explanation. We address the limitations in existing approximate inference procedures by using the *Expectation Correction* (EC) algorithm [7] which provides a stable, accurate approximation, and scales well to long time series like those found in ASR. We show that EC can be seen as an extension to the SLDS of the Rauch, Tung, Striebel (RTS) [61] inference algorithm for the *Linear Dynamical System* (LDS).

1.2.4 A Bayesian Alternative to Gain Adaptation

A fundamental issue faced by models based on AR processes—and linear models in general—is that they are very sensitive to variations in the amplitude of the signal. This is problematic since, when dealing with speech waveforms, the amplitude can vary significantly across speakers or because of changes in recording conditions. A common way to overcome this limitation is to use *Gain Adaptation* (GA) [25, 27] to automatically adjust the model. During training and testing, GA is performed by replacing the setting of some parameters of the model by the setting which maximises the likelihood of the observed signal. However, adjusting model parameters by maximising *test* likelihoods is fundamentally outside the framework of standard statistical approaches to machine learning, since this may lead to overfitting when the models are sufficiently flexible. We address this problem by devising a statistically principled alternative based on an exact Bayesian procedure in which priors are explicitly defined on the parameters of the AR process. This approach leads to two new models, the Bayesian SAR-HMM and the Bayesian AR-SLDS.

1.2.5 Bayesian Switching Linear Dynamical Systems

The Bayesian SAR-HMM is limited to clean speech. Following the same approach as for the AR-SLDS, to deal with noise without having to train a new model, we extend the Bayesian SAR-HMM to include an explicit model of additive Gaussian white noise. The introduction of priors on the parameters makes inference in the proposed Bayesian AR-SLDS significantly more complicated than in the AR-SLDS, preventing the direct use of standard approximation algorithms like EC. We address this issue by proposing two variational approximations for which inference can then be carried out using existing methods.

The Bayesian treatment of the AR-SLDS also provides an alternative to GA where the noisy observed signal is considered as a *scaled* and *corrupted* version of a clean hidden signal which is modelled by a SAR-HMM trained on clean data. Compared to the gain-adapted and Bayesian AR-SLDSs, the proposed scale-invariant AR-SLDS has the advantage of being able to model variations in the signal amplitude explicitly, without having to adjust the parameters of the underlying AR processes. The variance of the noise and the signal scaling factor are considered as random variables and are therefore automatically adapted, potentially enabling the robust identification of scale invariant clean signals in the presence of noise.

1.3 Contributions

The contributions of this thesis are the following:

- We propose a novel formulation of the inference in the SAR-HMM inspired from the Rauch, Tung, Striebel (RTS) algorithm for the Kalman Filter. This has been published in [49] and [48].

- We propose to use a Switching Linear Dynamical Systems (SLDS) to model the noisy speech waveform directly. Although SLDSs have been used before in a similar context [24, 63], their applications were limited to feature vectors. The proposed model is trained on clean data and can readily be used on noisy data. Experiments carried out with artificial and real noise sources showed that our approach significantly enhances recognition accuracy of isolated digits in noisy environments. This has been published in [49].

- We propose a new algorithm for performing approximate inference in the SLDS. Our method relaxes some of the limitations of current approximation techniques and proved to be stable, accurate and to scale well to long times series such as those found in ASR. This has been published in [9] and submitted to IEEE Signal Processing Letters.

- We propose a principled Bayesian alternative to Gain Adaptation (GA) in models based on AR processes. Our approach yields a performance comparable to that of GA for Signal to Noise Ratios (SNRs) above 5 dB. This was published in [48].

- We present a Bayesian treatment of the SLDS and devise two possible variational approximations which enable the intractable posterior distribution to be approximated using traditional inference methods. We also propose another probabilistic alternative to GA by devising a Bayesian SLDS which models scale invariance explicitly.

1.4 Limitations

In this thesis, we exclusively consider the task of recognising isolated digits spoken by different speakers. This relatively simple task will allow us to clearly present the underlying theoretical aspects of our approach to noise robust speech recognition. Compared to a feature-based HMM system, which essentially models the sequence of *discrete* subword units, the approach we will consider is more complex since it also incorporates a model of the *continuous* component, i.e., the waveform. Although, theoretically, nothing prevents the use of the proposed models to more challenging applications like large vocabulary ASR, in practice, the significant increase in computational time and memory consumption makes this impossible without the development of algorithmic surrogates. It is however beyond the scope of this thesis to investigate those techniques. The material presented here should therefore be considered as a tentative towards improving noise robustness in ASR, rather than a generic and fully applicable solution.

1.5 Structure of the Thesis

Chapter 2 This chapter introduces the common statistical framework used by most state-of-the-art ASR systems. It focuses on the problem of recognising isolated digits and explains how it can be addressed with HMMs. It shows that training can be reduced to an optimisation problem which can be solved by means of the Expectation Maximisation (EM) algorithm. Since EM requires inferring the posterior distribution of the hidden variables of the HMM, two inference procedure are presented; the well-known Forward-Backward (or Baum-Welch) algorithm and a version of the Rauch, Tung, Striebel (RTS) algorithm adapted to the HMM. This chapter also briefly presents two kinds of features commonly used in ASR, as well as a start-of-the-art denoising algorithm. Finally, we evaluate the recognition accuracy of a state-of-the-art HMM system on a isolated digit recognition task where utterances are artificially corrupted by additive Gaussian white noise.

Chapter 3 This chapter introduces the well-known and widely used AR process and its extension, the SAR-HMM, and shows how they can be used to model the speech waveform. Inference in the SAR-HMM can be performed with the RTS algorithm, and Gain Adaptation deals with variations in the signal amplitude. Update equations for the model parameters are derived and the recognition accuracy of the SAR-HMM on the isolated digit recognition task is evaluated and compared to that of the state-of-the-art HMM system.

Chapter 4 This chapter presents the AR-LDS and the AR-SLDS which extend the AR process and the SAR-HMM respectively to explicitly model additive Gaussian white noise. We explain how to perform Gain Adaptation in those models and briefly show the difficulty of performing exact inference in the AR-SLDS. The recognition accuracy of the AR-SLDS in clean and noisy conditions is evaluated and compared to that of the other models.

Chapter 5 This chapter presents the generic form of the LDS and the SLDS. We give a detailed derivation of the original RTS algorithm for the LDS and present the novel Expectation Algorithm for the SLDS. Pseudo-code for both algorithms is also provided.

Chapter 6 This chapter presents our proposed Bayesian alternative to Gain Adaptation. We introduce the Bayesian SAR-HMM which is an alternative form of SAR-HMM where the parameters are treated as random variables. An exact inference procedure is presented and the accuracy of the new model in clean and noisy conditions is evaluated and compared to that of the previous models.

Chapter 7 This chapter introduces two types of Bayesian AR-SLDS; the first one is a direct extension of the Bayesian SAR-HMM which includes an explicit model of additive Gaussian white noise and the second is a tentative to model scale invariance explicitly. Since exact inference is difficult in both models, two different variational procedures are presented. Update formulae for the scale-invariant model are also given. The accuracy of both models in clean and noisy conditions is compared to that of the previous models. Finally, the performance of the gain-adapted and Bayesian AR-SLDS is compared to that of a state-of-the-art HMM system on a subset of the Aurora database.

Chapter 8 This chapter summarises the main achievements of this thesis. We draw conclusions on the work presented in the previous chapters and outline possible future directions.

Chapter 2

Feature-Based Automatic Speech Recognition

2.1 Introduction

In this chapter we review the basic principles of classical speech recognition. Our exposition is mainly centred on the underlying theoretical aspects of speech recognition, rather than the implementation details. We consider the relatively simple task of recognising isolated spoken digits. This allows us to avoid difficulties with large vocabulary speech recognition and to easily introduce the probabilistic framework underlying current state-of-the-art ASR systems. Furthermore, we present only the traditional *maximum likelihood* (ML) approach where training is performed by maximising the total likelihood of the training examples. The more recent discriminative approaches to training [37, 68] are not considered because they are difficult to apply to more complex models like those we will consider in later chapters.

2.2 Isolated Digit Recognition

Our aim is to recognise digits spoken by different speakers. To achieve this, we possess a number of examples of each digits under the form of waveforms. A subset of the data will be used to train the ASR system and another to evaluate the recognition accuracy. We will denote by $y_{1:T}$ a waveform made of T samples and by y_t the t-th sample. In classical ASR systems, the waveform is not modelled directly, but converted into a shorter sequence of feature vectors $\mathbf{o}_{1:N}$ whose role is to retain the waveform characteristics essential for recognition. Each feature vector compresses the information over a window of samples and is therefore expected to present less variability than the original samples.

A statistical model with parameters Ψ defines the probability distribution $p(\mathbf{o}_{1:T} \mid \Psi)$ of a sequence $\mathbf{o}_{1:T}$. If we denote by Ψ_d the parameter setting for the model of the d-th digit and by $\{\mathbf{o}^1_{1:N}, \ldots, \mathbf{o}^M_{1:N}\}$ the set of M training sequences[1] for that same digit, then the goal of training is to find the parameter setting $\Psi^\star_d$ which maximises the total log-likelihood of the training sequences, i.e.,

$$\Psi^\star_d = \arg\max_{\Psi_d} \sum_{m=1}^{M} \log p(\mathbf{o}^m_{1:N} \mid \Psi_d). \tag{2.1}$$

Once the optimal parameter setting of each digit model has been found, the recognition accuracy of the system can be evaluated. For a given sequence of feature vectors $\mathbf{o}_{1:N}$ obtained from an utterance

[1]In practice, the training sequences have different length. To simplify notation, we assume that they all have the same length.

of the test set, the recognised digit $d^\star$ is the one which corresponds to the parameter setting for which the log-likelihood of the sequence is the highest, i.e.,

$$d^\star = \arg\max_d \ \log p(\mathbf{o}_{1:N} \mid \Psi_d). \tag{2.2}$$

We therefore translate the problems of training and evaluation to that of computing the optimal parameter setting $\Psi_d^\star$ and the log-likelihood $\log p(\mathbf{o}_{1:N} \mid \Psi_d)$ of a given sequence $\mathbf{o}_{1:N}$. In the following sections we describe in details the formal steps required to solve those two problems. Before doing so, we briefly review two common types of feature extraction techniques.

2.3 Feature Extraction

The material presented here is largely inspired from the HTK book, the reference manual of the HTK speech recognition toolkit [71]. We will focus on two types of features, the *Linear Predictive Cepstral Coefficients* (LPCCs) and the *Mel-Frequency Cepstral Coefficients* (MFCCs). Feature vectors integrate information over windows of samples which are generally overlapping. The raw samples $y_{1:T}$ are rarely used directly, but are generally pre-processed.

2.3.1 Pre-Emphasis & Hamming Windowing

Speech signals have most of their energy concentrated in the lower frequencies. This is sometime problematic because it makes the other frequencies less relevant. A common practice is therefore to perform pre-emphasis. If we denote by t_n the time index of the first sample of the n-th window and T_n the length of that same window, then pre-emphasis transforms the sequence of samples $x_{1:T_n} \equiv y_{t_n:t_n+T_n-1}$ into a new sequence $x'_{1:T_n}$ by applying the first order difference equation

$$x'_t = x_t - \alpha\, x_{t-1}$$

where $0 \leq \alpha < 1$ and typically $\alpha = 0.97$. Another common transformation, which is useful to avoid discontinuities at window boundaries, is to downweight the samples at the edge of the window. This is done by using an Hamming window

$$x''_t = \left(0.54 - 0.46 \cos \frac{2\pi(t-1)}{T_n - 1}\right) x'_t.$$

For each window $n \in [1, N]$, the corresponding sequence of pre-processed samples $x''_{1:T_n}$ is then transformed into a feature vector $\mathbf{o}_n$ of either LPCCs or MFCCs, as follows.

2.3.2 Linear Predictive Cepstral Coefficients

LPCCs are obtained by performing a linear prediction analysis; the vocal tract transfer function is modelled by an all-pole filter with transfer function

$$H(z) = \frac{1}{\sum_{i=0}^{p} a_i z^{-i}}$$

where p is the number of poles and $a_0 = 1$. The filter coefficients a_i are chosen to minimise the mean square filter prediction error summed over the analysis window. The LPCCs are computed with the recursion

$$c_k = -a_k + \frac{1}{k} \sum_{i=1}^{k-1} (k-i)\, a_i\, c_{k-i}.$$

and then used as features, i.e., $\mathbf{o}_n = \begin{bmatrix} c_0 & \dots & c_K \end{bmatrix}^\mathsf{T}$, where K is the number of coefficients. The advantage of using the cepstral coefficients c_k instead of the filter coefficients a_i as features is that the c_k are generally decorrelated. They can therefore be more conveniently modelled by mixture of Gaussians with diagonal covariance matrices.

2.3.3 Mel-Frequency Cepstral Coefficients

MFCCs are another common type of acoustic feature. They are generated from the result of a filterbank analysis whose goal is to mimic the pitch perception of the human ear. The basic idea behind filterbank analysis is to compute, for each window n, the Fourier transform of the corresponding sequence of samples $x''_{1:T_n}$ and then to apply p triangular filters of varying width on the resulting magnitudes. The filter centres are spaced non-linearly on the frequency scale in a way similar to the human ear. The result of the filterbank analysis on a window is a vector $\begin{bmatrix} m_1 & \dots & m_p \end{bmatrix}^\mathsf{T}$ of filterbank energies. The corresponding MFCCs are then computed from this by discrete cosine transform, i.e.,

$$c_k = \sqrt{\frac{2}{p}} \sum_{j=1}^{p} m_j \cos\left(\frac{\pi k}{p}(j-0.5)\right)$$

to form the feature vector $\mathbf{o}_n = \begin{bmatrix} c_0 & \dots & c_K \end{bmatrix}^\mathsf{T}$ of length K. The number of MFCCs typically used is $K = 13$. Feature vectors are also often extended to include information about the energy of the signal as well as the first and second temporal derivatives of the static coefficients. The typical dimension of an extended feature vector is 39 coefficients per window.

2.4 The Model

The role of the model is to define the probability distribution $p(\mathbf{o}_{1:N} \mid \Psi_d)$ needed by Equations 2.1 and 2.2. Since the recognition accuracy of an ASR system depends only on the likelihood, it is important to devise a model of appropriate complexity. The best model is of course the one for which $d^\star$ in Equation 2.2 is correct for all utterances of the test set. If the model is too simple, some utterances may not be correctly recognised. Similarly, if a model is too complex, overfitting may occur, and result in poor performance on the test data.

2.4.1 The Gaussian Distribution

For continuous valued feature vectors, one of the simplest models is probably the multivariate Gaussian distribution which defines the probability distribution of a feature vector $\mathbf{o}_n$ as

$$p(\mathbf{o}_n) = \frac{1}{|2\pi\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{o}_n - \boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{o}_n - \boldsymbol{\mu})\right\} \tag{2.3}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of $\mathbf{o}_n$, respectively. If we furthermore assume that the vectors are generated independently of each other, then the probability distribution of a sequence $\mathbf{o}_{1:N}$ is

$$p(\mathbf{o}_{1:N}) = \prod_{n=1}^{N} p(\mathbf{o}_n). \tag{2.4}$$

One limitation of this model is that the sequential order of the feature vectors is irrelevant. Another limitation is that the same probability distribution is used for all feature vectors. To better model the intrinsic non-stationarity of speech signals, it would be more appropriate to consider subword units, like phonemes, and to assume that to each unit is associated a potentially different probability distribution. A natural way to represent this is to use a mixture of Gaussians where each mixture component corresponds to a different unit.

2.4.2 The Mixture of Gaussians

We want to add a layer to our model which represents the fact that a feature vector $\mathbf{o}_n$ corresponds to the subword unit s_n. In classical ASR systems, it is generally assumed that there is a finite number

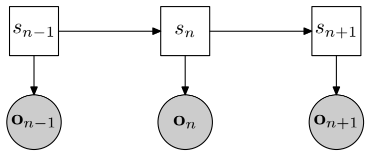


Figure 2.1: DBN representation of the HMM; s_n represents the hidden state and $\mathbf{o}_n$ the observed feature vector. Squares and circles depict discrete and continuous random variables respectively. The observed variables are darkened.

of subword units, each one having a certain probability $p(s_n)$ of occurring. One way to express this is by means of the mixture

$$p(\mathbf{o}_n) = \sum_{s_n} p(\mathbf{o}_n \,|\, s_n)\, p(s_n). \tag{2.5}$$

From a generative point of view, this means that we first pick a subword unit at random according to the distribution $p(s_n)$ and then generate a feature vector randomly from the distribution $p(\mathbf{o}_n \,|\, s_n)$. In a mixture of Gaussians, this latter distribution is the multivariate Gaussian distribution

$$p(\mathbf{o}_n \,|\, s_n) = \frac{1}{|2\pi\mathbf{\Sigma}_{s_n}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\big(\mathbf{o}_n - \boldsymbol{\mu}_{s_n}\big)^{\mathsf{T}}\mathbf{\Sigma}_{s_n}^{-1}\big(\mathbf{o}_n - \boldsymbol{\mu}_{s_n}\big)\right\}. \tag{2.6}$$

The main difference with Equation 2.3 is that the distribution is now conditioned on s_n, i.e., its parameters depends on s_n. The mixture approach addresses the problem of modelling subword units. However, if Equation 2.4 is used as the probability distribution of the whole sequence, the sequential order in which the subword units occur plays no role. To address this issue we need to consider a more elaborate distribution where the temporal dependency is taken into account.

2.4.3 The Hidden Markov Model

The hidden Markov model (HMM) is a key component of most classical ASR systems. It is attractive because it enables the temporal relationship between subword units to be expressed in a simple probabilistic way. The basic idea is to introduce a state transition distribution $p(s_n \,|\, s_{n-1})$ which gives the probability that the state (subword unit) s_n follows the state s_{n-1}. The model is (first-order) Markovian because the transition probability is conditioned on only the most recent state. Though it is possible to use distribution of higher orders, $p(s_n \,|\, s_{n-1}, s_{n-2})$ for example, this is rarely done because the number of possible state combinations to be considered grows exponentially with the order of the model. The HMM defines a joint probability distribution over the sequences of feature vectors and states of the form

$$p(\mathbf{o}_{1:N}, s_{1:N}) = p(\mathbf{o}_1 \,|\, s_1)\, p(s_1) \prod_{n=2}^{N} p(\mathbf{o}_n \,|\, s_n)\, p(s_n \,|\, s_{n-1}) \tag{2.7}$$

where $p(s_1)$ is a prior distribution which gives the probability of starting with state s_1 and $p(\mathbf{o}_n \,|\, s_n)$ is the emission distribution. The joint distribution given by Equation 2.7 can be represented as the Dynamical Bayesian Network (DBN) of Figure 2.1. The probability of a sequence of feature vectors $\mathbf{o}_{1:N}$ is obtained by integrating over all the possible state sequences

$$p(\mathbf{o}_{1:N}) = \sum_{s_{1:N}} p(\mathbf{o}_{1:N}, s_{1:N}). \tag{2.8}$$

This summation is more complicated to perform than that in Equation 2.5 since the factors in the product are no longer independent. Fortunately, the structure of the HMM allows the exact likelihood $p(\mathbf{o}_{1:N})$ to be efficiently computed, as explained in Section 2.7.

2.5 Parameter Optimisation

In our probabilistic framework, training a model on a set of M sequences $\{\mathbf{o}^1_{1:N}, \ldots, \mathbf{o}^M_{1:N}\}$ is equivalent to finding the parameter setting $\Psi^\star$ which maximises the total log-likelihood of the training data, i.e.,

$$\Psi^\star = \arg\max_{\Psi} \sum_{m=1}^{M} \log p(\mathbf{o}^m_{1:N} \,|\, \Psi). \tag{2.9}$$

A straightforward solution is to differentiate the sum on the right hand side (rhs) of Equation 2.9 with respect to Ψ and to look for the parameter setting for which the derivative is zero. In the case of a HMM, $p(\mathbf{o}_{1:N} \,|\, \Psi)$ is given by Equation 2.8 and Ψ corresponds to all the free parameters of the model[2]. For a HMM with a Gaussian emission probability given by Equation 2.6 for example, we would have

$$\Psi = \bigcup_s \Big\{ \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s, p(s_1 = s) \Big\} \,\cup\, \bigcup_{i,j} \Big\{ p(s_n = j \,|\, s_{n-1} = i) \Big\}.$$

Differentiating the rhs of Equation 2.9 with respect to Ψ and using Equation 2.8 gives

$$\frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N} \,|\, \Psi) = \frac{1}{p(\mathbf{o}_{1:N} \,|\, \Psi)} \sum_{s_{1:N}} \frac{\partial}{\partial \Psi} p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi).$$

To find the derivative we therefore have to differentiate $p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi)$ which, according to Equation 2.7, is a product. Things would be simpler if, instead of having to differentiate a product, we differentiate a sum. In that case the parameters would be isolated and the optimisation would be simplified. This is indeed possible if we consider the alternate differentiation

$$\frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N} \,|\, \Psi) = \sum_{s_{1:N}} \underbrace{\frac{p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi)}{p(\mathbf{o}_{1:N} \,|\, \Psi)}}_{p(s_{1:N} \,|\, \mathbf{o}_{1:N}, \Psi)} \frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi). \tag{2.10}$$

This corresponds to taking the average of the derivative of the joint distribution with respect to the posterior distribution $p(s_{1:N} \,|\, \mathbf{o}_{1:N}, \Psi)$. Equation 2.10 can be more conveniently written as

$$\frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N} \,|\, \Psi) = \Big\langle \frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi) \Big\rangle_{p(s_{1:N} \,|\, \mathbf{o}_{1:N}, \Psi)} \tag{2.11}$$

where $\langle \cdot \rangle_p$ denotes the average with respect to the distribution p. Whilst no closed form expression exists for finding zeros of this derivative, the right-hand-side (rhs) of Equation 2.11 is straightforward to compute numerically, and may be used as part of a gradient based optimisation routine. If we possess an estimate $\tilde{\Psi}$ of the optimal parameter setting $\Psi^\star$, an alternative strategy could be to approximate the average in the rhs of Equation 2.11 by

$$\Big\langle \frac{\partial}{\partial \Psi} \log p(\mathbf{o}_{1:N}, s_{1:N} \,|\, \Psi) \Big\rangle_{p(s_{1:N} \,|\, \mathbf{o}_{1:N}, \tilde{\Psi})}.$$

This approach forms the basis of the *Expectation Maximisation* (EM) algorithm [21].

2.6 The Expectation Maximisation Algorithm

The EM algorithm is an iterative method which provides an approximation to the solution of Equation 2.9. Given the estimate Ψ^i of the optimal parameter setting $\Psi^\star$ obtained at the i-th iteration, it finds the new estimate Ψ^{i+1} such that

$$\Psi^{i+1} = \arg\max_{\Psi} \sum_{m=1}^{M} \big\langle \log p(\mathbf{o}^m_{1:N}, s_{1:N} \,|\, \Psi) \big\rangle_{p(s_{1:N} \,|\, \mathbf{o}^m_{1:N}, \Psi^i)}. \tag{2.12}$$

[2]In this case, $p(\mathbf{o}_{1:N})$ and $p(\mathbf{o}_{1:N} \,|\, \Psi)$ are actually the same thing since we do not set any prior $p(\Psi)$ on the parameters of the model. If a prior is added, then $p(\mathbf{o}_{1:N}) = \sum_\Psi p(\mathbf{o}_{1:N} \,|\, \Psi)\, p(\Psi)$.

The initial estimate required to initialise the algorithm can be found heuristically or set randomly. An interesting property of the EM algorithm is that the total likelihood of the training sequences is guaranteed not to decrease after each iteration [21]. In the following, to simplify notation, we will mainly consider the case of a single training sequence $\mathbf{o}_{1:N}$. The extension to multiple sequences is easily obtained by summing the various contributions. If we define

$$q(s_{1:N}) \equiv p(s_{1:N} \,|\, \mathbf{o}_{1:N}, \Psi^i) \tag{2.13}$$

then, for a HMM, the average in the rhs of Equation 2.12 can be, according to Equation 2.7, rewritten as[3]

$$\sum_{n=1}^{N} \big\langle \log p(\mathbf{o}_n \,|\, s_n) \big\rangle_{q(s_n)} + \sum_{n=1}^{N} \big\langle \log p(s_n \,|\, s_{n-1}) \big\rangle_{q(s_{n-1},s_n)}. \tag{2.14}$$

Therefore, whilst the average in Equation 2.12 is carried out over all the possible state sequences $s_{1:N}$, for a HMM, only the marginal posterior distributions $q(s_n)$ and $q(s_{n-1}, s_n)$ are actually required. The computation of the marginal posterior distributions is often called *inference* because it infers from the sole knowledge of the feature vectors $\mathbf{o}_{1:N}$ the marginal distributions $q(s_n)$ and $q(s_{n-1}, s_n)$.

2.7 Inference

Inference in the HMM is traditionally carried out with the *Forward-Backward* algorithm [10, 58] which is a special case of the more general *Belief Propagation* algorithm [55]. An alternative way of performing inference in the HMM is to use a method similar to that proposed by Rauch, Tung and Striebel (RTS) for the Kalman Filter [61]. Though both the Forward-Backward and the RTS algorithms are applicable to the HMM, for more complicated models, like those we will consider in later chapters, the RTS method is more appropriate.

2.7.1 The Forward-Backward Algorithm

The Forward-Backward algorithm computes the posterior distribution $q(s_n) \equiv p(s_n \,|\, \mathbf{o}_{1:N})$ in two passes. The first computes the forward messages $\alpha_n(s_n)$ and the second computes the backward messages $\beta_n(s_n)$. The α and β messages are defined such that

$$p(s_n \,|\, \mathbf{o}_{1:N}) \propto \alpha_n(s_n)\, \beta_n(s_n). \tag{2.15}$$

In a HMM, the messages correspond to the probabilities:

$$\alpha_n(s_n) = p(\mathbf{o}_{1:n}, s_n) \quad \text{and} \quad \beta_n(s_n) = p(\mathbf{o}_{n+1:N} \,|\, s_n) \tag{2.16}$$

which satisfy Equation 2.15.

Forward Pass

The forward message $\alpha_n(s_n)$ at step n can be related to the message at the previous step by

$$\begin{aligned}
\alpha_n(s_n) = p(\mathbf{o}_{1:n}, s_n) &= \sum_{s_{n-1}} p(\mathbf{o}_{1:n}, s_n, s_{n-1}) \\
&= p(\mathbf{o}_n \,|\, s_n) \sum_{s_{n-1}} p(\mathbf{o}_{1:n-1}, s_n, s_{n-1}) \\
&= p(\mathbf{o}_n \,|\, s_n) \sum_{s_{n-1}} p(s_n \,|\, s_{n-1}) \underbrace{p(\mathbf{o}_{1:n-1}, s_{n-1})}_{\alpha_{n-1}(s_{n-1})}.
\end{aligned} \tag{2.17}$$

[3]To simplify notation we define $p(s_1 \,|\, s_0) \equiv p(s_1)$ and we do not write the dependency on Ψ explicitly.

Starting from $\alpha_1(s_1) = p(\mathbf{o}_1 \,|\, s_1)\, p(s_1)$, it is therefore possible to compute all the forward messages by recursively applying Equation 2.17. The forward pass is also useful for computing the likelihood of a sequence since, at the last step,

$$p(\mathbf{o}_{1:N}) = \sum_{s_N} p(\mathbf{o}_{1:N}, s_N) = \sum_{s_N} \alpha_N(s_N).$$

Backward Pass

The backward message $\beta_n(s_n)$ at step n can be related to the message at the next step by

$$\begin{aligned}
\beta_n(s_n) = p(\mathbf{o}_{n+1:N} \,|\, s_n) &= \sum_{s_{n+1}} p(\mathbf{o}_{n+1:N}, s_{n+1} \,|\, s_n) \\
&= \sum_{s_{n+1}} p(\mathbf{o}_{n+1} \,|\, s_{n+1}, s_n)\, p(\mathbf{o}_{n+2:N}, s_{n+1} \,|\, s_n) \\
&= \sum_{s_{n+1}} p(\mathbf{o}_{n+1} \,|\, s_{n+1})\, p(\mathbf{o}_{n+2:N} \,|\, s_{n+1}, s_n)\, p(s_{n+1} \,|\, s_n) \\
&= \sum_{s_{n+1}} p(\mathbf{o}_{n+1} \,|\, s_{n+1})\, \underbrace{p(\mathbf{o}_{n+2:N} \,|\, s_{n+1})}_{\beta_{n+1}(s_{n+1})}\, p(s_{n+1} \,|\, s_n) \qquad (2.18)
\end{aligned}$$

where, as can be seen in Figure 2.1, for the third equality, we used the fact that the emission distribution does not depend on s_n and, for the fourth equality, that $\mathbf{o}_{n+2:N}$ is independent of s_n once s_{n+1} is known. Starting from $\beta_N(s_N) = 1$, it is therefore possible to compute all the backward messages by iteratively applying Equation 2.18. The pairwise marginal $q(s_{n-1}, s_n) \equiv p(s_{n-1}, s_n \,|\, \mathbf{o}_{1:N})$ required by Equation 2.14 is given by

$$p(s_{n-1}, s_n \,|\, \mathbf{o}_{1:N}) \propto \alpha_{n-1}(s_{n-1})\, p(\mathbf{o}_n \,|\, s_n)\, p(s_n \,|\, s_{n-1})\, \beta_t(s_n).$$

2.7.2 The RTS Algorithm

The RTS algorithm [61] is a correction smoother; the forward pass computes the *filtered* posterior $p(s_n \,|\, \mathbf{o}_{1:n})$ and the backward pass corrects this to obtain the *smoothed* posterior $p(s_n \,|\, \mathbf{o}_{1:N})$. In the Forward-Backward algorithm the two passes are independent of each other, while in the RTS algorithm the backward pass requires the filtered posterior from the forward pass.

Forward Pass

The filtered posterior $p(s_n \,|\, \mathbf{o}_{1:n})$ computed by the RTS forward pass is equal to the normalised forward message $\alpha_n(s_n)$. The filtered posterior at step n can be related to the filtered posterior at the previous step by

$$\begin{aligned}
p(s_n \,|\, \mathbf{o}_{1:n}) &= \sum_{s_{n-1}} p(s_n, s_{n-1} \,|\, \mathbf{o}_{1:n}) \\
&\propto p(\mathbf{o}_n \,|\, s_n) \sum_{s_{n-1}} p(s_n, s_{n-1} \,|\, \mathbf{o}_{1:n-1}) \\
&\propto p(\mathbf{o}_n \,|\, s_n) \sum_{s_{n-1}} p(s_n \,|\, s_{n-1})\, p(s_{n-1} \,|\, \mathbf{o}_{1:n-1}). \qquad (2.19)
\end{aligned}$$

where $p(s_{n-1} \,|\, \mathbf{o}_{1:n-1})$ is the filtered posterior at the previous step. Equation 2.19 is similar to Equation 2.17, the only difference being that a normalised $\alpha_{n-1}(s_{n-1})$ is explicitly used. The filtered posterior can be used to compute the likelihood of a sequence since

$$p(\mathbf{o}_{1:N}) = \prod_{n=1}^{N} p(\mathbf{o}_n \,|\, \mathbf{o}_{1:n-1}) = \prod_{n=1}^{N} \sum_{s_n} p(\mathbf{o}_n, s_n \,|\, \mathbf{o}_{1:n-1})$$

and $p(\mathbf{o}_n, s_n \,|\, \mathbf{o}_{1:n-1})$ is given by the rhs of Equation 2.19.

Backward Pass

In the RTS backward pass, the smoothed posterior $p(s_n \,|\, \mathbf{o}_{1:N})$—which is given by Equation 2.15 in the Forward-Backward algorithm—is computed directly by means of the recursive formula

$$\begin{aligned} p(s_n \,|\, \mathbf{o}_{1:N}) &= \sum_{s_{n+1}} p(s_n, s_{n+1} \,|\, \mathbf{o}_{1:N}) \\ &= \sum_{s_{n+1}} p(s_n \,|\, s_{n+1}, \mathbf{o}_{1:N})\, p(s_{n+1} \,|\, \mathbf{o}_{1:N}) \\ &= \sum_{s_{n+1}} p(s_n \,|\, s_{n+1}, \mathbf{o}_{1:n})\, p(s_{n+1} \,|\, \mathbf{o}_{1:N}) \end{aligned} \tag{2.20}$$

where $p(s_{n+1} \,|\, \mathbf{o}_{1:N})$ is the smoothed posterior at the next step. For the third equation, we used the fact that s_n is independent of any future observations once s_{n+1} is known, as shown in Figure 2.1. The backward transition probability $p(s_n \,|\, s_{n+1}, \mathbf{o}_{1:n})$ is given by

$$p(s_n \,|\, s_{n+1}, \mathbf{o}_{1:n}) \propto p(s_n, s_{n+1} \,|\, \mathbf{o}_{1:n}) = p(s_{n+1} \,|\, s_n)\, p(s_n \,|\, \mathbf{o}_{1:n}) \tag{2.21}$$

where $p(s_n \,|\, \mathbf{o}_{1:n})$ is the filtered posterior at step n. The backward pass is initialised with the filtered posterior obtained at the N-th step, since both filtered and smoothed posteriors are equal at that point. The pairwise marginal $p(s_{n-1}, s_n \,|\, \mathbf{o}_{1:N})$ is given by the rhs of Equation 2.20 before summation.

2.8 Parameter Updating

To optimise Equation 2.14, we differentiate with respect to each of the parameters in Ψ and search for the zeros of the derivative. For the transition distribution, the function we need to optimise is[4]

$$\mathcal{F}_{\text{trans}} = \sum_{n=2}^{N} \big\langle \log p(s_n \,|\, s_{n-1}) \big\rangle_{q(s_{n-1}, s_n)} = \sum_{n=2}^{N} \sum_{s_{n-1}, s_n} q(s_{n-1}, s_n)\, \log p(s_n \,|\, s_{n-1}).$$

Optimising $\mathcal{F}_{\text{trans}}$ will not yield the correct answer since it misses the constraint that the transition distribution must sum to one. If we define $p(s_n = j \,|\, s_{n-1} = i) = a_{ij}$ and use the method of Lagrange multipliers then the correct function is

$$\mathcal{F}_{\text{trans}} = \sum_{n=2}^{N} \sum_{i,j} q(s_{n-1} = i, s_n = j)\, \log a_{ij} - \lambda \sum_{j} a_{ij}.$$

Differentiating with respect to a_{kl} and setting to zero, we obtain

$$\frac{\partial}{\partial a_{kl}} \mathcal{F}_{\text{trans}} = \sum_{n=2}^{N} q(s_{n-1} = k, s_n = l)\, \frac{1}{a_{kl}} - \lambda = 0.$$

Since $\sum_l a_{kl} = 1$, we end up with

$$p(s_n \,|\, s_{n-1}) = \frac{\sum_{n=2}^{N} q(s_{n-1}, s_n)}{\sum_{n=2}^{N} \sum_{s_n} q(s_{n-1}, s_n)} = \frac{\sum_{n=2}^{N} q(s_{n-1}, s_n)}{\sum_{n=2}^{N} q(s_{n-1})}. \tag{2.22}$$

[4]Note that, according to Equation 2.13, the posterior $q(s_{n-1}, s_n)$ is computed with the current estimate Ψ^i.

For the prior distribution, we simply have $p(s_1) = q(s_1)$. For the emission distribution, if we denote by ϑ_s the parameter of the emission distribution associated with state s, then the function to be optimised is

$$\mathcal{F}_{\text{emit}} = \sum_{n=1}^{N} \big\langle \log p(\mathbf{o}_n \,|\, s_n, \vartheta_{s_n}) \big\rangle_{q(s_n)} = \sum_{n=1}^{N} \sum_{s_n} q(s_n) \, \log p(\mathbf{o}_n \,|\, s_n, \vartheta_{s_n}).$$

Differentiating with respect to ϑ_s and setting the result equal to zero, yields

$$\frac{\partial}{\partial \vartheta_s} \mathcal{F}_{\text{emit}} = \sum_{n=1}^{N} q(s_n = s) \, \frac{\partial}{\partial \vartheta_s} \log p(\mathbf{o}_n \,|\, s_n = s, \vartheta_s) = 0. \tag{2.23}$$

This is the generic equation that we will need to solve to obtain update formulae for the parameters of the emission distribution.

2.9 Unsupervised Spectral Subtraction

The models used in current state-of-the-art ASR systems are often trained on clean data, but tested on noisy data. This problem is generally addressed by a pre-processing step which attempts to remove the effect of noise [28, 14, 34, 43, 62]. A good example of a pre-processing technique is the Unsupervised Spectral Subtraction (USS) [43]. The idea behind USS is to build a two-components mixture model of the power spectrum of a noisy speech signal. The mixture works as an activity (silence) detector; one component models regions of the power spectrum where no activity is detected and the other, regions where activity (speech) is detected. If we denote by m_{nf} the magnitude of the complex number obtained from the Fast Fourier Transform of a pre-emphasised signal $x'_{1:T_n}$ (see Section 2.3) at frequency f, then USS assumes that

$$p(m_{nf}) = p(m_{nf} \,|\, \mathsf{sil}) \, p(\mathsf{sil}) + p(m_{nf} \,|\, \mathsf{act}) \, p(\mathsf{act})$$

where $p(m_{nf} \,|\, \mathsf{sil})$ and $p(m_{nf} \,|\, \mathsf{act})$ are a Rayleigh and shifted Erlang distribution, respectively. This model is trained with EM so that the total log-likelihood of the magnitudes $\sum_{n,f} \log p(m_{nf})$ is maximised. An element of the power spectrum with magnitude m_{nf} is then filtered out if its posterior probability of being active

$$p(\mathsf{act} \,|\, m_{nf}) = \frac{p(m_{nf} \,|\, \mathsf{act}) \, p(\mathsf{act})}{p(m_{nf} \,|\, \mathsf{sil}) \, p(\mathsf{sil}) + p(m_{nf} \,|\, \mathsf{act}) \, p(\mathsf{act})}$$

is below a certain threshold. Although the approximations made by USS seem rather drastic—for example, the magnitudes m_{nf} are assumed to be independent of each other—this simple scheme has been shown to perform as well as other, more complex, denoising methods like [28]—see [42] for a comparison. In our experiments with feature-based HMMs, we will therefore only consider USS and implicitly assume it is a good representative of a state-of-the-art technique.

2.10 Experimental Setup

We evaluated the recognition accuracy of a state-of-the-art HMM system on a simple isolated digits recognition task on the TI-DIGITS database [44]. This was achieved by training a separate HMM for each of the eleven digits (0–9 and ‘oh’) found in the database. The training set for each digit was composed of 110 single digit utterances, spoken by 55 different male speakers. Similarly, the test set was composed of 112 single digit utterances, spoken by 56 different male speakers. The speakers in the test set were also different from those in the training set. The utterances were originally recorded at a sampling frequency of 20 kHz, but to shorten the processing time, we downsampled them to 8 kHz. To evaluate the noise robustness of the model, we corrupted the samples of the test utterances with additive Gaussian white noise.

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	**95.5**	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	**86.4**	**59.1**	**9.5**

Table 2.1: Comparison of the word accuracy (in percent), at various SNRs, of a state-of-the-art HMM system with and without USS. The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing.

2.11 Performance

Training and evaluation were performed with the HTK speech recognition toolkit [71], using the same setup as the baseline system used for the AURORA task [35], namely, 18 states, a left-to-right transition distribution—i.e., $p(s_n \mid s_{n-1}) \neq 0$ only for $s_n \in [s_{n-1}, s_{n-1}+1]$—a mixture of three Gaussians per state and feature vectors composed of 39 MFCCs, including first and second temporal derivatives as well as energy. The number of parameters to be trained per model was therefore: $18 \times 3 = 54$ mixture weights, $18 \times 3 \times 39 = 2106$ means and 2106 variances[5]—one for each of the 39 components of the feature vector—and 17 transition probabilities. This makes a total of 4283 parameters per digit model.

Table 2.1 compares the word accuracy, at various Signal to Noise Ratios (SNRs), of a trained HMM system with and without USS. As expected, the accuracy of the system without USS, at moderate to low SNRs, is significantly reduced. In general, USS enhances recognition accuracy, but fails when the SNR is low. The performance of the HMM is critically dependent on extracting effective noise free features from the noisy signal. As noise increases, this becomes increasingly difficult and standard ASR systems therefore break down.

2.12 Summary

We presented the essential ingredients of the probabilistic framework used in current state-of-the-art ASR systems. Fitting a model to a set of training data is equivalent to maximising the total log-likelihood of the training sequences and recognition is performed by finding the model for which the log-likelihood of a test sequence is the highest. The speech waveform is not modelled directly, but converted into a sequence of feature vectors which try to remove redundant information and to retain only the information essential to recognition.

The HMM is a temporal model which is particularly well suited for modelling sequences of feature vectors since its internal hidden state dynamics corresponds to the intuitive idea that speech can be seen as an ordered sequence of subword units. However, the presence of hidden variables in the HMM makes the direct optimisation of the total log-likelihood of the training sequences difficult. This problem can be addressed by using the EM algorithm. Inference of the posterior distribution of the hidden states given the observations can be efficiently performed by means of the Forward-Backward algorithm or, alternatively, with the RTS algorithm.

The accuracy of a feature-based HMM system degrades significantly in the presence of noise. At high SNRs, the use of pre-processing methods like USS generally makes the model more robust. However, at lower SNRs, this approach is probably too limited since the filtering is performed independently of the model. A potentially more fruitful approach is to model the noisy speech waveform directly as a *clean hidden* signal corrupted by additive Gaussian white noise. Since this approach *jointly* models speech and noise, it is therefore potentially more robust than the traditional approach where speech and noise are modelled separately. In the next chapter, we introduce the Switching

[5]MFCCs are generally considered as independent and are therefore modelled by a multivariate Gaussian distribution with a diagonal covariance matrix.

Autoregressive HMM (AR-HMM), a model of the clean speech waveform which we will later extend to include an explicit model of additive Gaussian white noise.

Chapter 3

Autoregressive Models

The novel formulation of the inference in the SAR-HMM by means of the RTS algorithm (Section 3.3.3) has been published in [49] and [48].

3.1 Introduction

The previous chapter focused on the modelling of sequences of feature vectors. In this chapter, with consider an alternative approach where the clean speech waveform is modelled directly by means of autoregressive (AR) processes. This will serve as a basis for the development of more refined models where speech and noise are *jointly* modelled.

We start by looking at the simple AR process and then show how it may be combined with a HMM to form the Switching AR-HMM (SAR-HMM) proposed by Ephraim and Roberts [27]. The aim of this chapter is to introduce the notation and the basic theory underlying the switching linear models. In [27], inference was originally performed with a modified version of the Forward-Backward algorithm. This modified algorithm actually corresponds to the RTS algorithm; presumably Ephraim and Roberts were unaware of this connection and their resulting derivation for inference in the SAR-HMM looks more complex. In contrast, our formulation of the inference procedure is directly inspired from the RTS algorithm.

3.2 The Autoregressive Process

A possible way to model a speech waveform—a sequence of unidimensional samples $y_{1:T}$—is by means of an *Autoregressive* (AR) process. An AR process models the sample y_t as the sum of a linear combination of the R previous samples and a random, normally distributed, innovation η_t:

$$y_t = \sum_{r=1}^{R} c_r y_{t-r} + \eta_t \quad \text{with} \quad \eta_t \sim \mathcal{N}(0, \sigma^2) \tag{3.1}$$

where $\mathcal{N}(\mu, \sigma^2)$ represents the normal (Gaussian) distribution with mean μ and variance σ^2, and c_r are the AR coefficients. Equation 3.1 defines the probability distribution of a single sample y_t as

$$p(y_t \mid y_{t-R:t-1}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2}\Big(y_t - \sum_{r=1}^{R} c_r y_{t-r}\Big)^2 \right\} \tag{3.2}$$

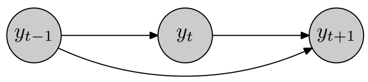


Figure 3.1: DBN representation of a second order AR process; y_t symbolises the observed speech sample at time t. The AR process does not contain any hidden variables.

which is a Gaussian whose mean is *conditioned* on the R previous samples. The probability distribution of the whole sequence $y_{1:T}$ is given by

$$p(y_{1:T}) = \prod_{t=1}^{T} p(y_t \mid y_{t-R:t-1}). \tag{3.3}$$

Since, at the beginning of the sequence, $y_{t-R:t-1}$ is not defined, we assume that $y_t = 0$ when $t \leq 0$. For a second order ($R = 2$) AR process, the joint distribution defined by Equation 3.3 can be graphically represented by the DBN of Figure 3.1.

3.2.1 Gain Adaptation

A fundamental limitation of the AR process is that the innovation variance σ^2 does not scale properly with the signal. In particular, if the samples $y_{1:T}$ are scaled by a factor α in Equation 3.1, then we would expect the innovation variance to scale by a factor α^2. In other words, the 'gain', i.e., the variance σ^2, needs to be properly adapted to each sequence $y_{1:T}$, and has a strong impact on the likelihood $p(y_{1:T})$. A solution to the gain problem is essential for the successful application of the AR process to acoustic signal analysis. A straightforward approach is to gain normalise the signal such that it always has unit variance. An alternate and, in practice, more effective solution is to replace σ^2 in Equation 3.1 by the variance which maximises the log-likelihood of the speech signal $y_{1:T}$, i.e.,

$$\sigma^2_{\mathsf{ML}} = \arg\max_{\sigma^2} \log p(y_{1:T} \mid \sigma^2). \tag{3.4}$$

This approach, known as *Gain Adaptation* (GA), has been successfully used for isolated digit recognition with AR models in clean and noisy environments [25, 27]. Solving Equation 3.4 is similar to train the model on a single sequence, the only difference being that only the variance σ^2 is modified, the AR coefficients are left untouched. Since GA must be performed for each individual signal, this means it has to be performed for *test* data as well. However, adapting model parameter on the basis of test data is dangerous because, in flexible models, it may lead to overfitting.

3.2.2 Parameter Optimisation

The generic approach to solve Equation 3.4 is to differentiate the log-likelihood with respect to σ^2 and to search for the zero of the derivative. For a Gaussian distribution however, σ^2_{ML} is equal to the empirical variance. If we define

$$\tilde{\mathbf{y}}_t = \begin{bmatrix} y_{t-1} & \dots & y_{t-R} \end{bmatrix}^{\mathsf{T}} \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} c_1 & \dots & c_R \end{bmatrix}^{\mathsf{T}}$$

then the most likely (ML) variance is given by

$$\sigma^2_{\mathsf{ML}} = \left\langle (y_t - \mathbf{c}^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \right\rangle = \frac{1}{T} \sum_{t=1}^{T} (y_t - \mathbf{c}^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2. \tag{3.5}$$

Later we will also have to find the optimal setting $\mathbf{c}_{\mathsf{ML}}$ of the AR coefficients. This can be obtained from the empirical covariance of y_t and $\tilde{\mathbf{y}}_t$. Multiplying both sides of Equation 3.1 by $\tilde{\mathbf{y}}_t^{\mathsf{T}}$ and taking the average, yields

$$\langle y_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle = \mathbf{c}_{\mathsf{ML}}^{\mathsf{T}} \langle \tilde{\mathbf{y}}_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle + \langle \eta_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle \quad \Rightarrow \quad \mathbf{c}_{\mathsf{ML}}^{\mathsf{T}} = \langle y_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle \langle \tilde{\mathbf{y}}_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle^{-1} \tag{3.6}$$

where we used the fact that $\langle \eta_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle = 0$ since η_t is independent of $\tilde{\mathbf{y}}_t$.

3.3 The Switching Autoregressive HMM

An AR process cannot model the strong non-stationarities typically encountered in speech signals. A possibly more fruitful approach is to consider that each sample y_t can be generated by one of S different AR processes. If the selected AR process is furthermore picked at random according to a Markovian transition distribution, then we naturally end up with a HMM. The idea of using the state of a HMM to switch between a finite number of AR processes is at the heart of the AR-HMM proposed by Poritz [56]. The AR-HMM is an instance of Segmental HMMs, where each segment is modelled by a potentially different AR process. But, like all models based on the Segmental HMM, continuity is lost at segment boundaries. The *Switching AR-HMM* (SAR-HMM), recently introduced by Ephraim and Roberts [27] improves Poritz's original model by restoring continuity at segment boundaries. Both models consider a speech signal as the concatenation of fixed-length segments, each being modelled by a potentially different AR process. The reason for considering segments instead of individual samples is that, in practice we expect the dynamics to last for a minimal amount of time. It is therefore not desirable to allow the model to switch between the AR processes too rapidly. If we denote by N the number of segments, by K their length, by $t_n = K(n-1)+1$ the starting time step of the n-th segment and by $\mathbf{y}_n \equiv y_{t_n:t_{n+1}-1}$ the samples belonging to that same segment, then, for a sequence of segments $\mathbf{y}_{1:N}$ and a sequence of state $s_{1:N}$, the SAR-HMM defines the joint distribution

$$p(\mathbf{y}_{1:N}, s_{1:N}) = p(\mathbf{y}_1 \,|\, s_1)\, p(s_1) \prod_{n=2}^{N} p(\mathbf{y}_n \,|\, s_n, \tilde{\mathbf{y}}_{t_n})\, p(s_n \,|\, s_{n-1}) \tag{3.7}$$

where $s_n \in [1, S]$ is a discrete switch variable which indicates which AR process has been used to generate the n-th segment. Since, inside a segment, the samples are modelled by the same AR process, the segment emission probability is given by

$$p(\mathbf{y}_n \,|\, s_n, \tilde{\mathbf{y}}_{t_n}) = \prod_{t=t_n}^{t_{n+1}-1} p(y_t \,|\, s_n, \tilde{\mathbf{y}}_t)$$

which is analogous to Equation 3.3. The sample emission distribution corresponds to an AR process whose parameters depend on the state s_n, i.e.,

$$p(y_t \,|\, s_n, \tilde{\mathbf{y}}_t) = \frac{1}{\sqrt{2\pi\sigma_{s_n}^2}} \exp\left\{ -\frac{1}{2\sigma_{s_n}^2} (y_t - \mathbf{c}_{s_n}^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \right\}. \tag{3.8}$$

The main difference between the AR-HMM and the SAR-HMM is that, in the former, the segment emission probability is not conditioned on the previous samples $\tilde{\mathbf{y}}_{t_n} \equiv y_{t_n-R:t_n-1}$. In the AR-HMM, continuity is therefore lost at segment boundaries. Graphically, the SAR-HMM can be represented by the DBN of Figure 3.2. When compared to the DBN of Figure 2.1, we see two differences: the switch state conditions the emission of a whole segment of samples, and the observations are coupled in a forward fashion. Despite the apparent complexity of the SAR-HMM, the model remains a specially constrained version of a HMM, for which inference is computationally straightforward if carried out with the RTS algorithm[1]. Since the SAR-HMM is based on an AR process, GA needs to be performed too.

[1]The inference algorithm used in [27] actually corresponds to the RTS algorithm. The authors were probably unaware of that and propose a complex derivation based on the Forward-Backward algorithm which ends up being equivalent to the RTS method.

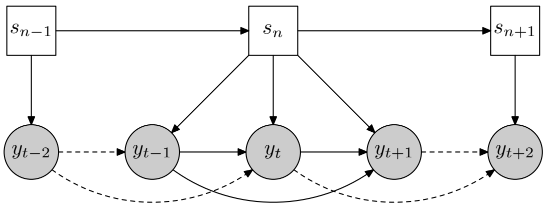


Figure 3.2: DBN representation of a second order SAR-HMM; s_n represents the hidden switch state and y_t the observed waveform sample. Compared to the SAR-HMM, Poritz's AR-HMM does not include the dashed links.

3.3.1 Gain Adaptation

Given a sequence of samples, GA is performed in the SAR-HMM by replacing the state innovation variance σ_s^2 in Equation 3.8 by the *per segment and state variance* σ_{ns}^2 which maximises the likelihood of the observed sequence $y_{1:T}$, i.e.,

$$\sigma_{ns}^2 = \frac{1}{T_n} \sum_{t=t_n}^{t_{n+1}-1} (y_t - \mathbf{c}_s^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \tag{3.9}$$

where $T_n = t_{n+1} - t_n$ is the length of the n-th segment.

3.3.2 Parameter Optimisation

The free parameters of the SAR-HMM are

$$\Psi = \bigcup_s \Big\{ \mathbf{c}_s, p(s_1 = s) \Big\} \;\cup\; \bigcup_{i,j} \Big\{ p(s_n = j \,|\, s_{n-1} = i) \Big\}.$$

The innovation variance σ_s^2 is not considered as a free parameter because it is replaced by σ_{ns}^2. Training a SAR-HMM consists in finding, for a given set of training sequences $\{\mathbf{y}_{1:N}^1, \dots \mathbf{y}_{1:N}^M\}$, the parameter setting $\Psi^\star$ such that

$$\Psi^\star = \arg\max_\Psi \sum_{m=1}^{M} \log p(\mathbf{y}_{1:N}^m \,|\, \Psi).$$

A straightforward optimisation of the total log-likelihood is difficult because

$$p(\mathbf{y}_{1:N}^m \,|\, \Psi) = \sum_{s_{1:N}} p(\mathbf{y}_{1:N}^m, s_{1:N} \,|\, \Psi).$$

A possible alternative is to use the EM algorithm (Section 2.6). This requires the computation of the marginal posterior $p(s_n \,|\, \mathbf{y}_{1:N})$.

3.3.3 Inference

The dependency of the samples $\mathbf{y}_n$ on the previous samples $\tilde{\mathbf{y}}_{t_n}$ in the segment emission distribution makes the Forward-Backward algorithm difficult to apply. In this case, the RTS algorithm is more appropriate because its backward pass does not involve the emission probability. The equivalent of Equation 2.19 for the filtered posterior $p(s_n \,|\, \mathbf{y}_{1:n})$ is

$$p(s_n \,|\, \mathbf{y}_{1:n}) \propto p(\mathbf{y}_n \,|\, s_n, \tilde{\mathbf{y}}_{t_n}) \sum_{s_{n-1}} p(s_n \,|\, s_{n-1})\, p(s_{n-1} \,|\, \mathbf{y}_{1:n}) \tag{3.10}$$

and the equivalent of Equation 2.20 for the smoothed posterior $p(s_n \,|\, \mathbf{y}_{1:N})$ is

$$p(s_n \,|\, \mathbf{y}_{1:N}) = \sum_{s_{n+1}} p(s_n \,|\, s_{n+1}, \mathbf{y}_{1:n})\, p(s_{n+1} \,|\, \mathbf{y}_{1:N}) \tag{3.11}$$

where the backward transition distribution (Equation 2.21) is given by

$$p(s_n \,|\, s_{n+1}, \mathbf{y}_{1:n}) \propto p(s_{n+1} \,|\, s_n)\, p(s_n \,|\, \mathbf{y}_{1:n}). \tag{3.12}$$

3.3.4 Parameter Updating

Using the same convention as that used in Section 2.8, i.e.,

$$q(s_{1:T}) \equiv p(s_{1:N} \,|\, \mathbf{y}_{1:N}, \Psi^i)$$

where Ψ^i is the parameter setting obtained at the end of the i-th iteration of the EM algorithm, the update formulae for the prior and transition probability are given by $p(s_1) = q(s_1)$ and Equation 2.22, respectively. The update formula for the AR coefficients $\mathbf{c}_s$ can be obtained from Equation 2.23 which, for the SAR-HMM, reads

$$\sum_{n=1}^{N} q(s_n = s) \frac{\partial}{\partial \mathbf{c}_s} \log p(\mathbf{y}_n \,|\, s_n, \tilde{\mathbf{y}}_{t_n}) = 0$$

with

$$\frac{\partial}{\partial \mathbf{c}_s} \log p(\mathbf{y}_n \,|\, s_n, \tilde{\mathbf{y}}_{t_n}) = \sum_{t=t_n}^{t_{n+1}-1} \frac{\partial}{\partial \mathbf{c}_s} \log p(y_t \,|\, s_n, \tilde{\mathbf{y}}_t).$$

The optimal $\mathbf{c}_s$ is given by Equation 3.6 which, in this case, reads

$$\mathbf{c}_s^{\mathsf{T}} = \left[\sum_{n=1}^{N} q(s_n = s) \sum_{t=t_n}^{t_{n+1}-1} y_t \tilde{\mathbf{y}}_t^{\mathsf{T}}\right] \left[\sum_{n=1}^{N} q(s_n = s) \sum_{t=t_n}^{t_{n+1}-1} \tilde{\mathbf{y}}_t \tilde{\mathbf{y}}_t^{\mathsf{T}}\right]^{-1} \tag{3.13}$$

3.4 Training & Evaluation

Following Section 2.10, we trained a separate SAR-HMM for each of the eleven digits (0–9 and ‘oh’) of the TI-DIGITS database [44]. The train and test sets were exactly the same as those used in Section 2.10. To facilitate comparison with the results presented in the literature, we chose the same experimental setup as that used in [27]. Each digit SAR-HMM was composed of ten states with a left-to-right transition matrix. Each state was associated with a 10-th order AR process and the model was constrained to stay an integer multiple of 140 time steps (0.0175 seconds) in the same state. The number of parameters to be trained was therefore: 10 AR coefficients per state and 9 transition probabilities. This makes a total of 109 free parameters, without counting the innovation variance which is automatically obtained from each utterance. For each training utterance the adapted gain σ^2_{ns} associated to each pair of segment and state was computed according to Equation 3.9. The parameters of the model—i.e., the transition matrix and the AR coefficients of each state—were then updated according to Equations 2.22 and 3.13. A new iteration of EM took place if the relative log-likelihood difference between two consecutive iterations was greater than 10^{-7}, i.e.,

$$\frac{\sum_m \left[\log p(\mathbf{y}^m_{1:N} \,|\, \Psi^{i+1}) - \log p(\mathbf{y}^m_{1:N} \,|\, \Psi^i)\right]}{\sum_m \log p(\mathbf{y}^m_{1:N} \,|\, \Psi^i)} > 10^{-7}. \tag{3.14}$$

Similarly to training, for each test sequence, the adapted gain σ^2_{ns} associated to each pair of segment and state was computed according to Equation 3.9, and recognition was then performed by selecting the digit model for which the likelihood of the given test sequence was the highest.

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	**95.5**	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	**86.4**	**59.1**	**9.5**
HMM (LPCC)	109	**100**	95.5	27.3	18.2	9.5	9.1
HMM (LPCC) + USS	109	**100**	86.4	77.3	18.2	13.6	10
SAR-HMM	119	88.3	25.5	9.7	8.6	9.3	9.4
SAR-HMM + GA	109	97.2*	79.8	56.7	22.2	9.7	9.1

Table 3.1: Comparison of the word accuracy (in percent), at various SNRs, of the SAR-HMM, with and without GA, and a state-of-the-art ASR system using MFCCs (copied from Table 2.1) or LPCCs, both with and without USS. The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing. For the SAR-HMM, the values result from experiments carried out with our own implementation of the model. *The original word accuracy, reported in [27], is 98.5.

3.5 Performance

Table 3.1 shows the recognition accuracy of the SAR-HMM with and without GA on the isolated digit recognition task described in Section 2.10. For comparison, the accuracy of the feature-based HMM reported in Table 2.1 is also reproduced. Table 3.1 also compares the SAR-HMM to a HMM using a single Gaussian per state and the same number of LPCCs features as the order of the SAR-HMM. Since LPCCs roughly correspond to the AR coefficients of an AR process fitted on the signal, they are more closely related to the SAR-HMM than the MFCCs. Furthermore, the number of parameters of the LPCC-based HMM is the same as that of the SAR-HMM (i.e., 109), while the number of parameters of the MFCC-based HMM is considerably higher (i.e., 4283). Comparing the SAR-HMM to a LPCC-based HMM is therefore more appropriate because their learning capacity are similar. The two models are not equivalent however since the AR coefficients used to compute the LPCCs are directly derived from the signal, whereas the SAR-HMM AR coefficients are part of the model. Reducing a segment of the signal to the AR coefficients which best reproduce that segment has a compressing effect which is not available to the SAR-HMM; matching the performance of the HMM is therefore difficult. A possible way to improve the accuracy of the SAR-HMM is to use a mixture of AR processes per state. Different mixture components might then be able to model different types of signal shape. Also, the order of the AR processes could be increased, thereby allowing a more accurate modelling of the signal. Although it might be interesting to investigate whether those two refinements improves the accuracy of the model, throughout this thesis, we will solely consider the case of a single AR process per state and simply assume that a 10-th order AR process is able to model the clean speech signal sufficiently accurately. A practical reason behind the choice of a 10-th order AR process is that, the CPU time and memory required to run the more complex models which we will consider later, are proportional to $S \times T \times R^3$ and $S \times T \times R^2$, respectively—where S is the number of states, T the number of speech samples and R the order of the AR process. Increasing the number of AR coefficients therefore quickly makes the running time of the experiments prohibitive.

Whilst the gain-adapted SAR-HMM has comparable to state-of-the-art performance on clean speech, the accuracy degrades rapidly under noisy conditions. This is because the AR process is defined on the potentially noisy observed signal directly; since the model forms predictions on the basis of past observations, the recognition accuracy of the SAR-HMM drops significantly if the speech signal is corrupted with noise. It is interesting to note however that GA significantly improves the accuracy of the SAR-HMM for low to moderate SNRs. This is expected since GA adapts the innovation variance in Equation 3.1 and adding Gaussian white noise is equivalent to increase that variance.

The improvement of USS when used with LPCCs is less significant than with MFCCs. This is probably because the signal after filtering is structurally much different from the clean signals the models were trained on. In contrast, MFCCs are much more robust since they are computed from the filtered power spectrum directly and encompass dynamic information as well. The poor performance of USS at high SNRs suggests that using a model of clean speech during filtering might be helpful, since it would allow USS to produce a denoised signal with a structure more similar to the training examples.

3.6 Summary

For isolated digits recognition, modelling the speech waveform directly is an alternative to conventional feature-based HMMs. The SAR-HMM represents the speech waveform as a sequence of segments which can be modelled by potentially different AR processes. Compared to the HMM commonly used in ASR, the SAR-HMM has two major differences; it uses the raw speech samples directly instead of feature vectors and the emission distribution depends on the previous samples. That dependency makes inference with the traditional Forward-Backward algorithm difficult. In contrast, inference is straightforward with the RTS algorithm.

A common problem with models based on an AR process is that they are sensitive to variations in signal amplitude. This problem is traditionally addressed by GA which consists in replacing the innovation variance by its most likely estimate. Although GA significantly improves the accuracy of the SAR-HMM in clean and noisy conditions, it is fundamentally outside the machine learning framework since it adapts model parameters on the basis of test data. This problem is addressed in Chapter 6 where we propose a Bayesian alternative to GA.

Although GA improves the performance of the SAR-HMM in noisy conditions, the SAR-HMM is only suitable for clean speech and does not model noise explicitly. In the next chapter, we extend the SAR-HMM to include an explicit model of additive Gaussian white noise. The proposed approach *jointly* models speech and noise and is therefore expected to be more robust than the traditional feature-based approach where speech and noise are modelled separately.

Chapter 4

Linear Dynamical Systems

The material presented in this chapter has been published in [49].

4.1 Introduction

To deal with noise, without having to train a new model, we extend the AR process and the SAR-HMM to include an explicit noise process whereby the observed *noisy* signal is viewed as a corrupted version of a *clean hidden* signal. This approach naturally leads to two well-known and widely used classes of models: the Linear Dynamical Systems (LDSs) and the Switching Linear Dynamical Systems (SLDSs) [6]. The LDS and the SLDS can be seen as both a generalisation and an extension of the AR process and the SAR-HMM respectively. Those models are expected to enhance noise robustness since the underlying AR/linear model is defined on a hidden clean counterpart of the noisy signal. Here we will make the simple assumption of independent Gaussian noise, although, as we will see in later chapters, the method may be extended to include more complex noise processes. Previous applications of the SLDS to ASR—see for example [24, 63]—have modelled the feature vectors, and not the raw signal directly. However, work in acoustic modelling [18] suggests that the SLDS is a potentially powerful tool for modelling the noisy speech waveform. In this chapter, we present the AR-LDS and the AR-SLDS, two instances of LDSs and SLDSs where the internal dynamics is autoregressive.

4.2 The AR-LDS

We would like to extend the AR process so that it can model additive Gaussian white noise explicitly. A possible approach is to assume that, to an observed sequence of *noisy* samples $v_{1:T}$ corresponds a *clean hidden* sequence of samples $y_{1:T}$. The clean sequence is modelled by an AR process (Equation 3.1) and the noisy samples v_t are related to the clean samples y_t by

$$v_t = y_t + \eta_t^{\nu} \quad \text{with} \quad \eta_t^{\nu} \sim \mathcal{N}(0, \sigma_{\nu}^2) \tag{4.1}$$

where η_t^{ν} represents additive Gaussian white noise with variance σ_{ν}^2. The corresponding joint distribution of the clean and noisy sequences is

$$p(v_{1:T}, y_{1:T}) = \prod_{t=1}^{T} p(v_t \,|\, y_t)\, p(y_t \,|\, \tilde{\mathbf{y}}_t)$$

where $\tilde{\mathbf{y}}_t = y_{t-R:t-1}$. Graphically, this model can be depicted by the DBN of Figure 4.1. It belongs to the class of Linear Dynamical Systems (LDSs) which represents the dynamics of an observed signal by means of linear hidden variable model. The major difference with the AR process (Figure 3.1)

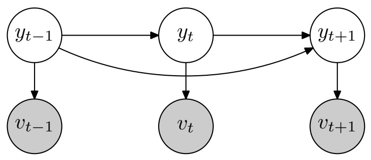


Figure 4.1: DBN representation of a second order AR-LDS; y_t represents the hidden clean speech sample and v_t the corresponding noisy sample.

is that the clean signal is now *hidden*. For $\sigma_\nu^2 \equiv 0$, the model is equivalent to an AR process and, when $\sigma_\nu^2 > 0$, it is effectively an AR model of clean speech plus a model of additive Gaussian white noise. The likelihood of an observed noisy signal $v_{1:T}$ is given by integrating over all possible hidden clean sequences, i.e.,

$$p(v_{1:T}) = \int_{y_{1:T}} p(v_{1:T}, y_{1:T}).$$

Since our ultimate goal is to use this model or one of its derivative to model speech waveforms, we need to address the problem of dealing with variations in the signal amplitude. A straightforward solution to that problem is to use GA.

4.2.1 Gain Adaptation

As for the AR process, GA in the AR-LDS consists in replacing the innovation variance σ^2 in Equation 3.1 by the variance which maximises the log-likelihood of the observed data $v_{1:T}$, i.e.,

$$\sigma^2_{\mathsf{ML}} = \arg\max_{\sigma} \log p(v_{1:T} \mid \sigma) = \arg\max_{\sigma} \log \int_{y_{1:T}} p(v_{1:T}, y_{1:T} \mid \sigma). \tag{4.2}$$

A major difference with GA in the AR process (Equation 3.5) is that the clean sequence $y_{1:T}$ is hidden. The presence of hidden variables makes the direct optimisation of Equation 4.2 difficult. A possible alternative is to use the EM algorithm. If we define

$$q(y_{1:T} \mid \sigma_i^2) \equiv p(y_{1:T} \mid v_{1:T}, \sigma_i^2)$$

where σ_i^2 is the estimate of σ^2_{ML} obtained after the i-th iteration, then the EM algorithm finds the next estimate σ^2_{i+1} such that

$$\begin{aligned}\sigma^2_{i+1} &= \arg\max_{\sigma^2} \left\langle \log p(v_{1:T}, y_{1:T} \mid \sigma^2) \right\rangle_{q(y_{1:T} \mid \sigma_i^2)}. \\ &= \arg\max_{\sigma^2} \sum_{t=1}^{T} \left\langle \log p(v_t, y_t \mid y_{t-1}, \sigma^2) \right\rangle_{q(y_{t-1}, y_t \mid \sigma_i^2)}. \end{aligned} \tag{4.3}$$

The solution to Equation 4.3 corresponds to the empirical variance, averaged over q, i.e.,

$$\sigma^2_{i+1} = \frac{1}{T} \sum_{t=1}^{T} \left\langle (y_t - \mathbf{c}^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \right\rangle_{q(y_t, \tilde{\mathbf{y}}_t \mid \sigma_i^2)}. \tag{4.4}$$

Since the amount of noise affecting an observed noisy sequence $v_{1:T}$ is unknown *a priori*, we need to find it automatically for each sequence. A possible approach is to use GA again and to replace σ_ν^2 in Equation 4.1 by the noise variance which maximises the likelihood of the observed sequence. Using EM, we obtain the update equation

$$\sigma^2_{\nu, i+1} = \frac{1}{T} \sum_{t=1}^{T} \left\langle (v_t - y_t)^2 \right\rangle_{q(y_t \mid \sigma^2_{\nu,i})}. \tag{4.5}$$

which is analogous to Equation 4.4. To evaluate Equations 4.4 and 4.5, we need to infer the posterior distribution $q(y_t, \tilde{\mathbf{y}}_t) \equiv p(y_t, \tilde{\mathbf{y}}_t \,|\, v_{1:T})$ and $q(y_t)$.

4.2.2 Inference

Inference in the AR-LDS can be performed with the RTS algorithm [61][1]. The forward pass of the RTS method computes the filtered posterior

$$p(y_t, \tilde{\mathbf{y}}_t \,|\, v_{1:t}) \equiv p(y_{t-R:t} \,|\, v_{1:t})$$

and the backward pass corrects this to obtain the smoothed posterior

$$p(y_t, \tilde{\mathbf{y}}_t \,|\, v_{1:T}) \equiv p(y_{t-R:t} \,|\, v_{1:T}).$$

The equivalent of Equation 2.19 for the filtered posterior is

$$p(y_{t-R:t} \,|\, v_{1:t}) \propto p(v_t \,|\, y_t) \int_{y_{t-R-1}} p(y_t \,|\, y_{t-R:t-1})\, p(y_{t-R-1:t-1} \,|\, v_{1:t-1}) \tag{4.6}$$

By analogy with Equations 2.19 and 2.20, we could have been tempted to compute $p(y_t \,|\, v_{1:t})$ directly instead of the joint $p(y_{t-R:t} \,|\, v_{1:t})$. However, since y_t depends on $y_{t-R:t-1}$, a proper recursion would not be achievable in that case. The equivalent of Equation 2.20 for the smoothed posterior is

$$p(y_{t-R:t} \,|\, v_{1:T}) = \int_{y_{t+1}} p(y_{t-R:t} \,|\, y_{t+1}, v_{1:t})\, p(y_{t-R+1:t+1} \,|\, v_{1:T}) \tag{4.7}$$

where the backward transition distribution (Equation 2.21) is given by

$$p(y_{t-R:t} \,|\, y_{t+1}, v_{1:t}) \propto p(y_{t+1} \,|\, y_{t-R+1:t})\, p(y_{t-R:t} \,|\, v_t).$$

Equations 4.6 and 4.7 are more complicated to evaluate than Equations 2.19 and 2.20 because they deal with a continuous variable instead of a discrete one. However, since all the distributions involved in the computation of both the filtered and smoothed posteriors are Gaussians, it is sufficient to keep track of the mean and covariance matrix of the vector $y_{t-R:t}$. Equations 4.6 and 4.7 can be written more concisely by using the change of variable $\mathbf{h}_t = y_{t-R:t}$. This corresponds to the generic form of the LDS for which inference is presented in the next chapter.

4.3 The AR-SLDS

As for the SAR-HMM, to deal with the intrinsic non-stationarities of the speech signal, it is useful to allow the model to switch between a finite number of potentially different AR-LDSs. This approach leads to the AR-SLDS, an extension of the SAR-HMM where additive Gaussian white noise is modelled explicitly. The clean sequence $y_{1:T}$ is considered as *hidden* and is modelled by a SAR-HMM (Equation 3.7). As for the LDS, the relationship between the noisy observed samples v_t and the clean hidden samples y_t is given by Equation 4.1. The AR-SLDS defines the joint distribution

$$p(v_{1:T}, y_{1:T}, s_{1:N}) = \prod_{n=1}^{N} p(s_n \,|\, s_{n-1}) \prod_{t=t_n}^{t_{n+1}-1} p(v_t \,|\, y_t)\, p(y_t \,|\, s_n, \tilde{\mathbf{y}}_t)$$

where $p(s_1 \,|\, s_0) \equiv p(s_1)$ is a given prior distribution. Graphically, this can be depicted by the DBN of Figure 4.2. If the noise variance is low, i.e., $\sigma_\mathcal{V}^2 \equiv 0$, then the AR-SLDS mimics the SAR-HMM.

[1] In Section 2.7.2, we used the RTS algorithm as a replacement for the Forward-Backward algorithm in the HMM, but it has originally been devised for the LDS. Although, formally, inference can also be carried out with the Forward-Backward algorithm, in practice, this approach is less stable because the backward message is not a probability distribution.

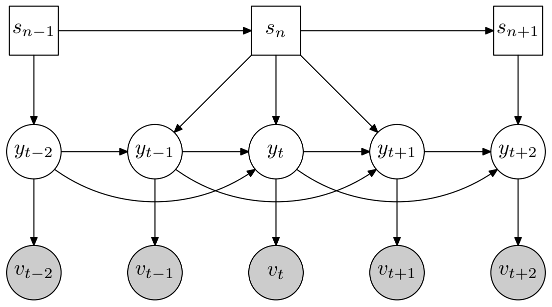


Figure 4.2: DBN representation of a second order AR-SLDS; s_n is the switch variable, y_t the clean hidden speech sample and v_t is the corresponding noisy sample. Compared to Figure 3.2, the clean speech samples are now hidden and only the noisy samples are observed.

For $\sigma_{\mathcal{V}}^2 > 0$, the model is effectively a SAR-HMM model of clean speech, plus a model of additive Gaussian white noise, and should thus provide a level of noise robustness. The likelihood of an observed noisy sequence $v_{1:T}$ is obtained by integrating out the hidden variables:

$$p(v_{1:T}) = \sum_{s_{1:N}} \int_{y_{1:T}} p(v_{1:T}, y_{1:T}, s_{1:N}).$$

As usual with AR-based models, we use GA to make the likelihood independent of the amplitude of the signal.

4.3.1 Gain Adaptation

Similarly to the SAR-HMM, the state dependent innovation variance σ_s^2 is replaced by the *segment-state* variance σ_{ns}^2 which maximises the likelihood of the observed sequence. Since inside a segment all the observations are modelled by the same AR-LDS, the update formula for σ_{ns}^2 is

$$\sigma_{i+1}^2 = \frac{1}{T_n} \sum_{t=t_n}^{t_{n+1}-1} \left\langle (y_t - \mathbf{c}_s^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \right\rangle_{q(y_t, \tilde{\mathbf{y}}_t \,|\, s_n = s, \sigma_i^2)} \tag{4.8}$$

where $T_n = t_{n+1} - t_n$ is the length of the n-th segment and

$$q(y_t, \tilde{\mathbf{y}}_t \,|\, s_n, \sigma_i^2) \equiv p(y_t, \tilde{\mathbf{y}}_t \,|\, s_n, v_{1:T}, \sigma_i^2).$$

Equation 4.8 is analogous to Equation 3.9 for the SAR-HMM, the main difference being that, since the clean signal on which the innovation acts on is not directly observed, we need to average with respect to the posterior probability of that hidden clean signal.

As for the AR-LDS, GA is also used for adapting the noise variance. Since we consider Gaussian white noise, we need to find a single noise variance common to all segments and states. Compared to σ^2, the formula for adapting $\sigma_{\mathcal{V}}^2$ therefore includes a sum over all segments and all possible switch states. Hence

$$\sigma_{\mathcal{V},i+1}^2 = \frac{1}{T} \sum_{n=1}^{N} \sum_{s_n} q(s_n) \sum_{t=t_n}^{t_{n+1}-1} \left\langle (v_t - y_t)^2 \right\rangle_{q(y_t \,|\, s_n, \sigma_{\mathcal{V},i}^2)}. \tag{4.9}$$

A useful alternative, in cases where the noise is not white, is to consider a segment dependent noise variance. This can be easily achieved by dropping the sum over n in Equation 4.9.

4.3.2 Inference

Solving Equations 4.8 and 4.9 requires computing the posterior distributions $p(y_t, \tilde{\mathbf{y}}_t \,|\, s_n, v_{1:T})$, $p(y_t \,|\, s_n, v_{1:T})$ and $p(s_n \,|\, v_{1:T})$. Contrary to the SAR-HMM and the LDS, where finding the posterior state of the hidden variables can be carried out exactly, inference is formally intractable in the SLDS. For example,

$$p(y_t, s_n \,|\, v_{1:T}) = \sum_{\substack{s_{1:n-1}\\ s_{n+1:N}}} p(y_t \,|\, s_{1:N}, v_{1:T})\, p(s_{1:N} \,|\, v_{1:T})$$

is a mixture of S^{N-1} Gaussians. Hence, the number of computations required to evaluate the posterior distribution exactly scales exponentially with the length of the sequence considered. In practice, approximations must therefore be used. Various algorithms exist for performing approximate inference in the SLDS—see [46, 72] for a review and [7] for a comparison. Two well-known examples are Expectation Propagation [50] and Generalised Pseudo Bayes [6, 52]. Both methods suffer from limitations which can be relaxed in the case of the SLDS—see [7] for a detailed explanation. To overcome limitations in existing approximate inference methods for the SLDS, we developed the Expectation Correction (EC) algorithm [7] which provides a stable, accurate approximation and scales well to long time series such as those found in ASR. A detailed description of EC is given in the next chapter. In this thesis, we use EC in all the experiments involving a SLDS.

4.4 Training & Evaluation

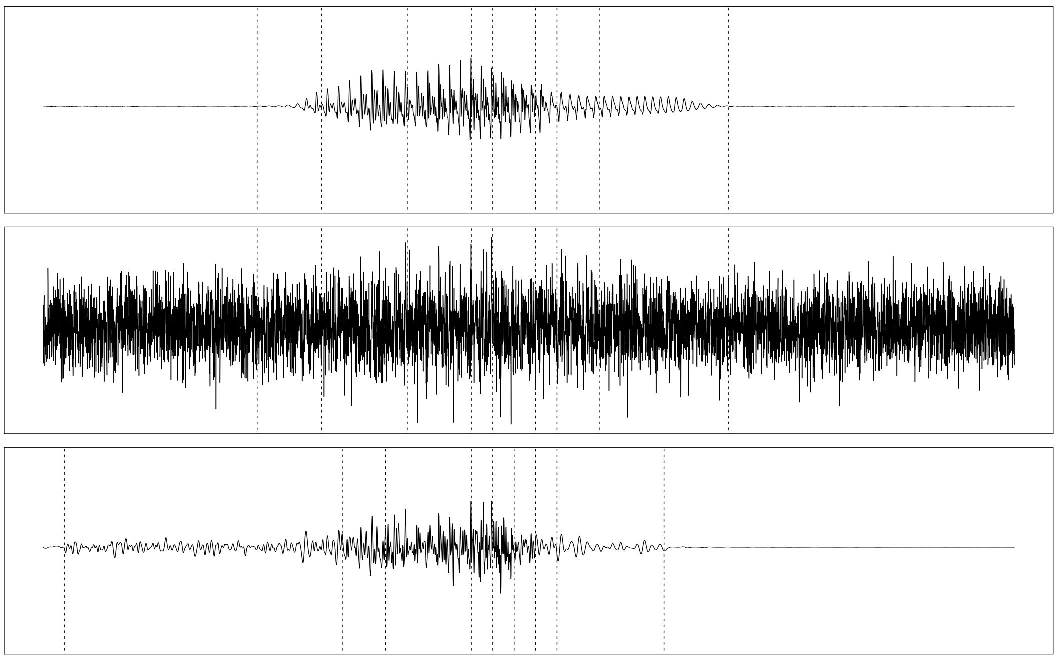
Following Section 2.10, we evaluated the recognition accuracy of the AR-SLDS on the isolated digits from the TI-DIGITS database. The AR-SLDS was not trained directly. Instead its parameters were simply set to the same value as in the corresponding trained SAR-HMM. The models were then tested on the same test set as used for the SAR-HMM. For each test utterance, the innovation and noise variance was iteratively adapted using Equations 4.8 and 4.9 until the relative log-likelihood difference between two consecutive iterations was less than 10^{-7} (Equation 3.14). The initial estimate σ_0^2 required by Equation 4.8 was obtained from the training set, by using the SAR-HMM maximum likelihood estimate of σ_s^2 for each state[2]. For the noise variance, Equation 4.9 was evaluated with a number of different values for $\sigma_{\nu,0}^2$ and the one which was resulting in the highest likelihood was used to seed the recursion. Since the noise variance is automatically adapted using GA, the AR-SLDS has the same number of parameters as the SAR-HMM, i.e., 109.

4.5 Examples of Signal Reconstruction

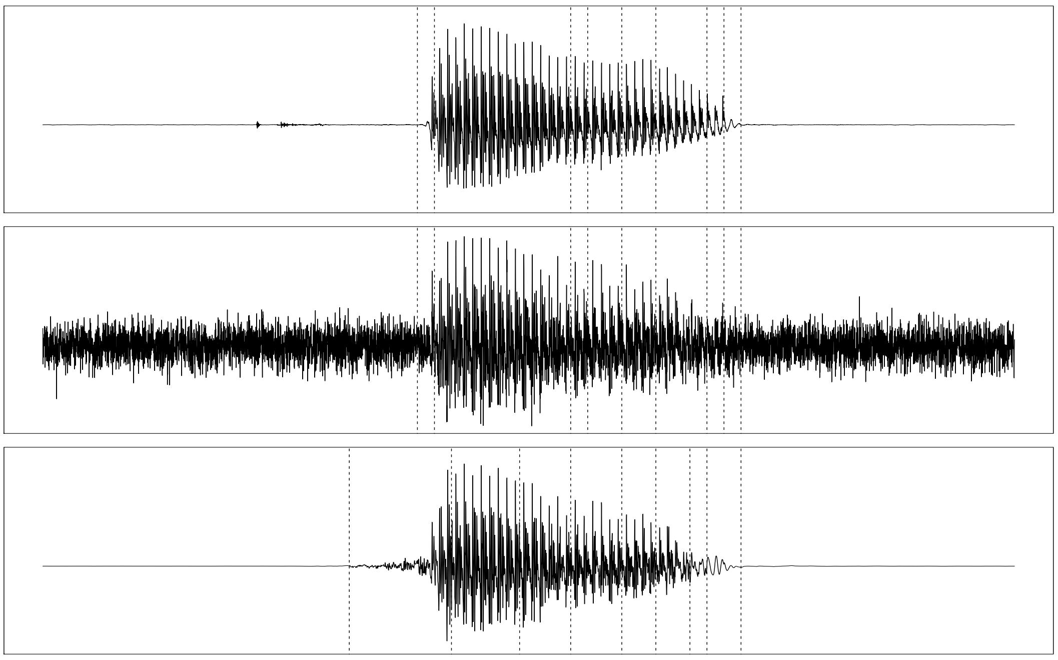
In order to demonstrate the noise robustness capabilities of the AR-SLDS, we plotted in Figure 4.3, for two examples: (top) the original clean waveform taken from the TI-DIGITS database, (middle) its artificially Gaussian white noise corrupted version and (bottom) the corresponding most likely reconstructed clean speech signal. The latter is obtained by taking, for each time step, the mean $\langle y_t \rangle$ of the smoothed posterior $p(y_t \,|\, s'_t, v_{1:T})$, with the state segmentation given by $s'_t = \arg\max_{s_t} p(s_t \,|\, v_{1:T})$. Figure 4.3 also shows, for each signal, the corresponding state segmentation given by the AR-SLDS.

In Figure 4.3 both noise-corrupted signals are correctly recognised by the AR-SLDS. This is encouraging since when the SNR is close to 0 dB, the shape of the original clean speech signal has almost disappeared and any denoising method which does not consider the dynamics of the clean signal will most likely fail. In this example, the reconstructed signal is reminiscent of a 'one' for the top example, and a 'five' for the bottom example. For the higher SNR the reconstruction is much closer to the original clean signal. The noisy 'one' shown on the top of Figure 4.3 has a log-likelihood of 2.0 when evaluated with the AR-SLDS corresponding to 'one' and a log-likelihood of 1.9995 with the model corresponding to 'oh'. This demonstrates that, under extremely noisy conditions, an accurate

[2]This value is not used in the gain-adapted SAR-HMM since it is replaced by the segment-state variance σ_{ns}^2.



‘one’ at SNR 0.7 dB



‘five’ at SNR 10.6 dB

Figure 4.3: Two examples of signal reconstruction using the AR-SLDS; (top) original clean waveform taken from the TI-DIGITS database, (middle) noisy signal, i.e., clean signal artificially corrupted by additive Gaussian white noise, (bottom) reconstructed clean signal. The dashed lines show the most-likely state segmentation. To facilitate comparison the clean segmentation is also reproduced on the noisy signal.

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	95.5	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	86.4	59.1	9.5
HMM (LPCC)	109	**100**	95.5	27.3	18.2	9.5	9.1
HMM (LPCC) + USS	109	**100**	86.4	77.3	18.2	13.6	10
SAR-HMM	119	88.3	25.5	9.7	8.6	9.3	9.4
SAR-HMM + GA	109	97.2	79.8	56.7	22.2	9.7	9.1
AR-SLDS	120	86.8	88.2	87.3	79.1	80	**63.6**
AR-SLDS + GA	109	96.8	96.8	**96.4**	**94.8**	**84**	61.2

Table 4.1: Comparison of the word accuracy (in percent), at various SNRs, of the AR-SLDS with and without GA, the SAR-HMM with and without GA (copied from Table 3.1), and a state-of-the-art ASR system with and without USS (copied from Table 2.1). The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing.

approximation of the likelihood is important, since many digit models are likely to have generated such a noisy signal. In both examples shown in Figure 4.3, the models stay in the second state for too long; this problem arises because the dynamics of the initial section of the speech signal is difficult to distinguish from silence. This suggests that a better reconstruction could potentially be achieved by explicitly modelling the state duration—in a way similar to [18], for example.

4.6 Performance

Table 4.1 shows the recognition accuracy of the AR-SLDS with and without GA on the isolated digit recognition task described in Section 2.10. When used without GA, the noise variance in the AR-SLDS was manually set to the correct value. In contrast, when GA was used, the noise variance was automatically adapted by iteratively applying Equation 4.9. For comparison, the performance of the SAR-HMM and the feature-based HMMs reported in Table 3.1 is also reproduced. Compared to the HMMs, the AR-SLDS significantly improves recognition accuracy for moderate to high SNRs. This is expected since USS does not use a model of clean speech and therefore cannot properly separate noise from speech. As for the SAR-HMM, apart from the lowest SNR, being able to adapt some of the model parameters to each particular example—with GA in this case—is essential for a successful use of the AR-SLDS for ASR. Although the AR-SLDS mimics the SAR-HMM when $\sigma_{\text{v}}^2 \equiv 0$, it is slightly less accurate than the SAR-HMM on clean data. This is mainly because, in the AR-SLDS, GA is performed via EM by iteratively applying Equations 4.8 and 4.9, while, in the SAR-HMM, GA is performed directly. It is therefore possible that the EM procedure stays stuck in a local minima which results in an innovation variance different from the one obtained with the SAR-HMM. Another potential explanation is that the true likelihood is sometimes wrongly estimated by EC.

4.7 Summary

The AR-SLDS is a joint model of speech and noise which extends the SAR-HMM to include an explicit model of additive Gaussian white noise. The potential benefit of this approach is that an AR-SLDS can be readily initialised with a SAR-HMM trained on clean speech, thereby providing a level of robustness to additive Gaussian white noise. On noisy data, the output noise variance, whose setting is unknown *a priori*, is automatically adjusted by means of GA. As in the SAR-HMM, GA is also used to adapt the innovation variance of the underlying AR process. Performing GA in the AR-SLDS is difficult because the clean signal is hidden, which prevents the direct optimisation of the likelihood. We proposed to use EM instead, which requires inferring the posterior distribution of the hidden variables. To overcome the difficulties of exactly computing the posterior distribution of the hidden variables in the AR-SLDS, we performed approximate inference with the EC algorithm. EC was preferred to other, more generic, methods found in the literature because it provides a fast, accurate and stable approximation which works well with long time series, as demonstrated in the next chapter.

Chapter 5

The Expectation Correction Algorithm

The material presented in this chapter has been published in [9] and submitted to IEEE Signal Processing Letters.

5.1 Introduction

In the previous chapter we focused on two particular cases of LDSs and SLDSs where the underlying linear model was an AR process. In this chapter we present the generic form of the LDS and the SLDS as well as two possible algorithms for performing inference with those models. We give a detailed derivation of the RTS algorithm for the LDS and show how it can be extended to form the Expectation Correction (EC) algorithm [7] for the SLDS.

5.2 The Linear Dynamical System

The Linear Dynamical System (LDS) [6] is a key temporal model in which a latent linear process generates the observed time series. The observation (or 'visible' variable) $\mathbf{v}_t \in \mathbb{R}^V$ is linearly related to the hidden state $\mathbf{h}_t \in \mathbb{R}^H$ by

$$\mathbf{v}_t = \mathbf{B}\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{V}}, \quad \boldsymbol{\eta}_t^{\mathcal{V}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{V}}, \boldsymbol{\Sigma}_{\mathcal{V}}) \tag{5.1}$$

where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes a Normal (Gaussian) distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. The hidden state $\mathbf{h}_t$ at the t-th time step is linearly related to the state at the previous time step by

$$\mathbf{h}_t = \mathbf{A}\mathbf{h}_{t-1} + \boldsymbol{\eta}_t^{\mathcal{H}}, \quad \boldsymbol{\eta}_t^{\mathcal{H}} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{H}}, \boldsymbol{\Sigma}_{\mathcal{H}}). \tag{5.2}$$

From a probabilistic point of view, Equations 5.2 and 5.1 defines the multivariate normal distributions

$$\begin{aligned} p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}) &= \frac{1}{|2\pi\boldsymbol{\Sigma}_{\mathcal{H}}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{h}_t - \mathbf{A}\mathbf{h}_{t-1})^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathcal{H}}^{-1}(\mathbf{h}_t - \mathbf{A}\mathbf{h}_{t-1})\right\} \\ p(\mathbf{v}_t \,|\, \mathbf{h}_t) &= \frac{1}{|2\pi\boldsymbol{\Sigma}_{\mathcal{V}}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{v}_t - \mathbf{B}\mathbf{h}_t)^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathcal{V}}^{-1}(\mathbf{v}_t - \mathbf{B}\mathbf{h}_t)\right\}. \end{aligned}$$

and the joint probability distribution is

$$p(\mathbf{v}_{1:T}, \mathbf{h}_{1:T}) = p(\mathbf{v}_1 \,|\, \mathbf{h}_1)\, p(\mathbf{h}_1) \prod_{t=2}^{T} p(\mathbf{v}_t \,|\, \mathbf{h}_t)\, p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}) \tag{5.3}$$

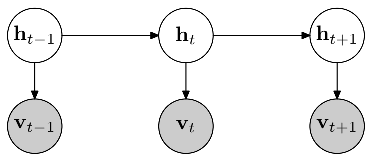


Figure 5.1: DBN representation of the LDS; $\mathbf{h}_t$ represents the hidden variable and $\mathbf{v}_t$ the observation. Graphically, the only difference with the DBN of a HMM (Figure 2.1) is the use of a continuous hidden variable instead of a discrete one.

where $p(\mathbf{h}_1)$ is a given prior distribution. Graphically, the LDS can be depicted by the DBN of Figure 5.1. The likelihood of an observed sequence $\mathbf{v}_{1:T}$ is given by integrating over all possible hidden sequences, i.e.,

$$p(\mathbf{v}_{1:T}) = \int_{\mathbf{h}_{1:T}} p(\mathbf{v}_{1:T}, \mathbf{h}_{1:T}).$$

5.2.1 The AR-LDS

For example, a third order AR-LDS with AR coefficients c_r, innovation variance σ^2 and noise variance $\sigma_\mathcal{V}^2$ can be written as a generic LDS by defining

$$\mathbf{h}_t = \begin{bmatrix} y_t \\ y_{t-1} \\ y_{t-2} \end{bmatrix}, \qquad \mathbf{A} = \begin{bmatrix} c_1 & c_2 & c_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{\Sigma}_\mathcal{H} = \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{5.4}$$

$$\mathbf{v}_t = \begin{bmatrix} v_t \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \qquad \mathbf{\Sigma}_\mathcal{V} = \begin{bmatrix} \sigma_\mathcal{V}^2 \end{bmatrix}. \tag{5.5}$$

5.2.2 The RTS Algorithm

The method of choice for performing inference in the LDS is the RTS algorithm [61]. In Section 2.7.2, we used the RTS algorithm as a replacement for the Forward-Backward algorithm in the HMM, but it has originally been devised for the LDS.

Forward and Backward Recursions

The forward pass computes the filtered posterior $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ and the backward pass corrects this to obtain the smoothed posterior $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:T})$. The equivalent of Equation 2.19 for the filtered posterior is

$$p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t}) \propto p(\mathbf{v}_t \,|\, \mathbf{h}_t) \int_{\mathbf{h}_{t-1}} p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1})\, p(\mathbf{h}_{t-1} \,|\, \mathbf{v}_{1:t-1}) \tag{5.6}$$

and the equivalent of Equation 2.20 for the smoothed posterior is

$$p(\mathbf{h}_t \,|\, \mathbf{v}_{1:T}) = \int_{\mathbf{h}_{t+1}} p(\mathbf{h}_t \,|\, \mathbf{h}_{t+1}, \mathbf{v}_{1:t})\, p(\mathbf{h}_{t+1} \,|\, \mathbf{v}_{1:T}) \tag{5.7}$$

where the backward transition distribution (Equation 2.21) is given by

$$p(\mathbf{h}_t \,|\, \mathbf{h}_{t+1}, \mathbf{v}_{1:t}) \propto p(\mathbf{h}_{t+1} \,|\, \mathbf{h}_t)\, p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t}). \tag{5.8}$$

Equations 5.6 and 5.7 are more complicated to evaluate than Equations 2.19 and 2.20 because they deal with a continuous variable instead of a discrete one. However, since all the distributions involved in the computation of both the filtered and smoothed posterior are Gaussians, it is sufficient to keep track of the mean and covariance of $\mathbf{h}_t$.

Finding the Forward Means and Covariances

To simplify exposition, we will consider the case where $\boldsymbol{\mu}_{\mathcal{H}}$ and $\boldsymbol{\mu}_{\mathcal{V}}$ are equal to zero. We will furthermore denote the mean and covariance of $\mathbf{h}_t$ corresponding to the filtered posterior $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ by $\mathbf{f}_t$ and $\mathbf{F}_t$ respectively. Evaluating the integral in Equation 5.6 yields $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t-1})$. If we define the operator Δ such that $\Delta\mathbf{x} = \mathbf{x} - \langle\mathbf{x}\rangle$, then the mean and delta of $\mathbf{h}_t$ are

$$\langle\mathbf{h}_t\rangle = \langle\mathbf{A}\mathbf{h}_{t-1} + \boldsymbol{\eta}_t^{\mathcal{H}}\rangle = \mathbf{A}\mathbf{f}_{t-1}, \qquad \Delta\mathbf{h}_t = \mathbf{h}_t - \langle\mathbf{h}_t\rangle = \mathbf{A}\Delta\mathbf{h}_{t-1} + \boldsymbol{\eta}_t^{\mathcal{H}}$$

and, by definition, the covariance matrix is

$$\langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle = \mathbf{A}\langle\Delta\mathbf{h}_{t-1}\Delta\mathbf{h}_{t-1}^{\mathsf{T}}\rangle\mathbf{A}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{H}} = \mathbf{A}\mathbf{F}_{t-1}\mathbf{A}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{H}}$$

since $\mathbf{h}_{t-1}$ and $\boldsymbol{\eta}_t^{\mathcal{H}}$ are uncorrelated and $\langle\boldsymbol{\eta}_t^{\mathcal{H}}\boldsymbol{\eta}_t^{\mathcal{H}\mathsf{T}}\rangle = \boldsymbol{\Sigma}_{\mathcal{H}}$. The filtered posterior $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ is obtained by conditioning on $\mathbf{v}_t$ the joint distribution

$$p(\mathbf{v}_t, \mathbf{h}_t \,|\, \mathbf{v}_{1:t-1}) = p(\mathbf{v}_t \,|\, \mathbf{h}_t)\, p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t-1}).$$

The mean, delta and covariance of $\mathbf{v}_t$, and the cross-covariance between $\mathbf{h}_t$ and $\mathbf{v}_t$ are

$$\begin{aligned} \langle\mathbf{v}_t\rangle &= \langle\mathbf{B}\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{V}}\rangle = \mathbf{B}\langle\mathbf{h}_t\rangle, & \Delta\mathbf{v}_t &= \mathbf{v}_t - \langle\mathbf{v}_t\rangle = \mathbf{B}\Delta\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{V}}, \\ \langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle &= \mathbf{B}\langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle\mathbf{B}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{V}}, & \langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle &= \langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle\mathbf{B}^{\mathsf{T}}. \end{aligned}$$

The joint distribution $p(\mathbf{v}_t, \mathbf{h}_t \,|\, \mathbf{v}_{1:t-1})$ is a Gaussian distribution proportional to

$$\exp\left\{-\frac{1}{2}\begin{bmatrix}\mathbf{h}_t - \langle\mathbf{h}_t\rangle \\ \mathbf{v}_t - \langle\mathbf{v}_t\rangle\end{bmatrix}^{\mathsf{T}} \begin{bmatrix}\langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle & \langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle \\ \langle\Delta\mathbf{v}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle & \langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle\end{bmatrix}^{-1} \begin{bmatrix}\mathbf{h}_t - \langle\mathbf{h}_t\rangle \\ \mathbf{v}_t - \langle\mathbf{v}_t\rangle\end{bmatrix}\right\}. \tag{5.9}$$

If we write the inverse covariance matrix as[1]

$$\begin{bmatrix}\langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle & \langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle \\ \langle\Delta\mathbf{v}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle & \langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle\end{bmatrix}^{-1} \equiv \begin{bmatrix}\mathbf{D}_{\mathcal{H}\mathcal{H}} & \mathbf{D}_{\mathcal{H}\mathcal{O}} \\ \mathbf{D}_{\mathcal{O}\mathcal{H}} & \mathbf{D}_{\mathcal{O}\mathcal{O}}\end{bmatrix}$$

then (5.9) can also be written as

$$\exp\left\{-\frac{1}{2}(\mathbf{h}_t - \mathbf{f}_t)^{\mathsf{T}}\mathbf{D}_{\mathcal{H}\mathcal{H}}(\mathbf{h}_t - \mathbf{f}_t)\right\} \exp\left\{-\frac{1}{2}(\mathbf{v}_t - \langle\mathbf{v}_t\rangle)^{\mathsf{T}}\mathbf{D}_{\mathcal{O}\mathcal{O}}(\mathbf{v}_t - \langle\mathbf{v}_t\rangle)\right\}$$

with $\mathbf{f}_t = \langle\mathbf{h}_t\rangle - \mathbf{D}_{\mathcal{H}\mathcal{H}}^{-1}\mathbf{D}_{\mathcal{H}\mathcal{O}}(\mathbf{v}_t - \langle\mathbf{v}_t\rangle)$. This corresponds to the factorisation

$$p(\mathbf{v}_t, \mathbf{h}_t \,|\, \mathbf{v}_{1:t-1}) = p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})\, p(\mathbf{v}_t \,|\, \mathbf{v}_{1:t-1}) \tag{5.10}$$

where $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ is the filtered posterior at time t, as given by Equation 5.6. Hence, $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ is a Gaussian with mean $\mathbf{f}_t$ and covariance $\mathbf{F}_t = \mathbf{D}_{\mathcal{H}\mathcal{H}}^{-1}$. Using the partionned matrix inverse [57], we obtain

$$\begin{aligned} \mathbf{D}_{\mathcal{H}\mathcal{H}}^{-1} &= \langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle - \langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle\langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle^{-1}\langle\Delta\mathbf{v}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle \\ \mathbf{D}_{\mathcal{H}\mathcal{O}} &= -\mathbf{D}_{\mathcal{H}\mathcal{H}}\langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle\langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle^{-1}. \end{aligned}$$

If we denote by $\mathbf{K} = \langle\Delta\mathbf{h}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle\langle\Delta\mathbf{v}_t\Delta\mathbf{v}_t^{\mathsf{T}}\rangle^{-1}$ the common factor in $\mathbf{D}_{\mathcal{H}\mathcal{H}}$ and $\mathbf{D}_{\mathcal{H}\mathcal{O}}$, then

$$\mathbf{f}_t = \langle\mathbf{h}_t\rangle + \mathbf{K}(\mathbf{v}_t - \langle\mathbf{v}_t\rangle) \quad \text{and} \quad \mathbf{F}_t = \langle\Delta\mathbf{h}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle - \mathbf{K}\langle\Delta\mathbf{v}_t\Delta\mathbf{h}_t^{\mathsf{T}}\rangle. \tag{5.11}$$

Since all the averaged quantities can be written as a function of the filtered mean $\mathbf{f}_{t-1}$ and covariance $\mathbf{F}_{t-1}$ at the previous time step, we therefore end up with recursive formulae for finding the mean and covariance at any given time step. The likelihood of an observed sequence $\mathbf{v}_{1:T}$ can be obtained by multiplying together the second factors in Equation 5.10 obtained at each time step, since

$$p(\mathbf{v}_{1:T}) = \prod_{t=1}^{T} p(\mathbf{v}_t \,|\, \mathbf{v}_{1:t-1}).$$

[1]Here $\mathcal{H}$ and $\mathcal{O}$ essentially mean *hidden* and *observed*, respectively.

Finding the Backward Means and Covariances

The backward transition distribution given by Equation 5.8 is obtained by conditioning the joint distribution $p(\mathbf{h}_{t+1}, \mathbf{h}_t \,|\, \mathbf{v}_{1:t})$ on $\mathbf{h}_{t+1}$. This is similar to the conditioning on $\mathbf{v}_t$ of the joint distribution $p(\mathbf{v}_t, \mathbf{h}_t \,|\, \mathbf{v}_{1:t-1})$ that we performed above. By analogy with Equation 5.11, the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ of $\mathbf{h}_t$ under $p(\mathbf{h}_t \,|\, \mathbf{h}_{t+1}, \mathbf{v}_{1:t})$ are therefore

$$\boldsymbol{\mu} = \langle \mathbf{h}_t \rangle + \mathbf{K}(\mathbf{h}_{t+1} - \langle \mathbf{h}_{t+1} \rangle) \quad \text{and} \quad \boldsymbol{\Sigma} = \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle - \mathbf{K} \langle \Delta\mathbf{h}_{t+1} \Delta\mathbf{h}_t^{\mathsf{T}} \rangle \tag{5.12}$$

with $\mathbf{K} = \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_{t+1}^{\mathsf{T}} \rangle \langle \Delta\mathbf{h}_{t+1} \Delta\mathbf{h}_{t+1}^{\mathsf{T}} \rangle^{-1}$ and

$$\begin{aligned} \langle \mathbf{h}_t \rangle &= \mathbf{f}_t & \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle &= \mathbf{F}_t \\ \langle \mathbf{h}_{t+1} \rangle &= \langle \mathbf{A}\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{H}} \rangle = \mathbf{A} \langle \mathbf{h}_t \rangle, & \Delta\mathbf{h}_{t+1} &= \mathbf{A}\Delta\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{H}}, \\ \langle \Delta\mathbf{h}_{t+1} \Delta\mathbf{h}_{t+1}^{\mathsf{T}} \rangle &= \mathbf{A} \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle \mathbf{A}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{H}}, & \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_{t+1}^{\mathsf{T}} \rangle &= \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle \mathbf{A}^{\mathsf{T}}. \end{aligned}$$

Equation 5.12 is equivalent to the backward linear equation

$$\mathbf{h}_t = \mathbf{K}\mathbf{h}_{t+1} + \langle \mathbf{h}_t \rangle - \mathbf{K} \langle \mathbf{h}_{t+1} \rangle + \boldsymbol{\eta}_t \quad \text{and} \quad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

which gives the state of $\mathbf{h}_t$ in function of $\mathbf{h}_{t+1}$. With the backward transition probability in hands we can evaluate the integral in Equation 5.7, which will yield the desired smoothed posterior. If we denote by $\mathbf{g}_t$ and $\mathbf{G}_t$ the mean and covariance of $\mathbf{h}_t$ under $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:T})$, then

$$\mathbf{g}_t = \langle \mathbf{h}_t \rangle + \mathbf{K}(\mathbf{g}_{t+1} - \langle \mathbf{h}_{t+1} \rangle) \quad \text{and} \quad \mathbf{G}_t = \mathbf{K}\mathbf{G}_{t+1}\mathbf{K}^{\mathsf{T}} + \boldsymbol{\Sigma}. \tag{5.13}$$

Implementation

By comparing Equations 5.11 and 5.13 (together with Equation 5.12), we see that the filtered and smoothed means have the same form and that the filtered and smoothed covariances are slightly different. The form of the filtered covariance is indeed the same as that of $\boldsymbol{\Sigma}$ (Equation 5.12) and the smoothed covariance includes the additional term $\mathbf{K}\mathbf{G}_{t+1}\mathbf{K}^{\mathsf{T}}$ which accounts for the fact that $\mathbf{h}_{t+1}$, contrary to $\mathbf{v}_t$, is not observed. Since both $\mathbf{F}_t$ and $\mathbf{G}_t$ are covariance matrices, they must be positive definite. In practice, numerical instabilities may lead to non-positive definite matrices. A slightly more stable way to compute the filtered and smoothed covariance matrices is by means of the so-called Josephson's formulae [6]. Josephson's formula for $\mathbf{F}_t$ reads

$$\mathbf{F}_t = (\mathbf{I}_H - \mathbf{K}\mathbf{B}) \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle (\mathbf{I}_H - \mathbf{K}\mathbf{B})^{\mathsf{T}} + \mathbf{K}\boldsymbol{\Sigma}_{\mathcal{V}}\mathbf{K}^{\mathsf{T}}$$

where $\mathbf{I}_H$ denotes the $H \times H$ identity matrix. Similarly, Josephson's formula for $\mathbf{G}_t$ reads[2]

$$\mathbf{G}_t = (\mathbf{I}_H - \mathbf{K}\mathbf{A}) \langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle (\mathbf{I}_H - \mathbf{K}\mathbf{A})^{\mathsf{T}} + \mathbf{K}(\mathbf{G}_{t+1} + \boldsymbol{\Sigma}_{\mathcal{H}})\mathbf{K}^{\mathsf{T}}.$$

Algorithms 4 and 5 give the pseudo-code for computing the filtered and smoothed means and covariances, respectively. The common part of both passes which consists in conditioning either the joint distribution $p(\mathbf{v}_t, \mathbf{h}_t \,|\, \mathbf{v}_{1:t-1})$ on $\mathbf{v}_t$ or $p(\mathbf{h}_t, \mathbf{h}_{t+1} \,|\, \mathbf{v}_{1:t})$ on $\mathbf{h}_{t+1}$ is implemented in Algorithm 1.

5.3 The Switching Linear Dynamical System

For time series which are not well described by a single LDS, we may model each observation by a potentially different LDS. This is the basis for the Switching LDS (SLDS) where, for each time step t,

[2]Although, to simplify notation, we use $\langle \Delta\mathbf{h}_t \Delta\mathbf{h}_t^{\mathsf{T}} \rangle$ and $\mathbf{K}$ in both the forward and the backward pass, their actual value are different.

Algorithm 1 COND $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{C}, \boldsymbol{\Sigma}_{\mathcal{O}}, \mathbf{w}, \mathbf{W}\}$.

```
$\boldsymbol{\mu}_{\mathcal{O}} \leftarrow \mathbf{C}\boldsymbol{\mu}$
$\boldsymbol{\Sigma}_{\mathcal{O}\mathcal{H}} \leftarrow \mathbf{C}\boldsymbol{\Sigma}$
$\boldsymbol{\Sigma}_{\mathcal{O}\mathcal{O}} \leftarrow \boldsymbol{\Sigma}_{\mathcal{O}\mathcal{H}}\,\mathbf{C}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{O}}$
$\mathbf{K} \leftarrow \boldsymbol{\Sigma}_{\mathcal{O}\mathcal{H}}\,\boldsymbol{\Sigma}_{\mathcal{O}\mathcal{O}}^{-1}$
$\mathbf{X} \leftarrow \mathbf{I} - \mathbf{K}\mathbf{C}$
$\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \mathbf{K}(\mathbf{w} - \boldsymbol{\mu}_{\mathcal{O}})$
$\boldsymbol{\Sigma} \leftarrow \mathbf{X}\,\boldsymbol{\Sigma}\,\mathbf{X}^{\mathsf{T}} + \mathbf{K}\,(\boldsymbol{\Sigma}_{\mathcal{O}} + \mathbf{W})\,\mathbf{K}^{\mathsf{T}}$
$p \leftarrow |\boldsymbol{\Sigma}_{\mathcal{O}\mathcal{O}}|^{-\frac{1}{2}}\ \exp\left\{-\frac{1}{2}(\mathbf{w} - \boldsymbol{\mu}_{\mathcal{O}})^{\mathsf{T}}\,\boldsymbol{\Sigma}_{\mathcal{O}\mathcal{O}}^{-1}\,(\mathbf{w} - \boldsymbol{\mu}_{\mathcal{O}})\right\}$
**return** $\{p, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$
```

Algorithm 2 RTS forward pass. This algorithm computes the mean $\mathbf{f}_t$ and the covariance $\mathbf{F}_t$ of $\mathbf{h}_t$ under $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$, as well as the log-likelihood $l \equiv \log p(\mathbf{v}_{1:T})$. The prior mean and covariance are denoted by $\boldsymbol{\mu}_{\mathcal{P}}$ and $\boldsymbol{\Sigma}_{\mathcal{P}}$ respectively.

```
$l \leftarrow 0$
**for** $t \leftarrow 1$ **to** T **do**
  **if** $t > 1$ **then**
    $\mathbf{x} \leftarrow \mathbf{A}\mathbf{f}_{t-1}$
    $\mathbf{X} \leftarrow \mathbf{A}\mathbf{F}_{t-1}\mathbf{A}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{H}}$
  **else**
    $\mathbf{x} \leftarrow \boldsymbol{\mu}_{\mathcal{P}}$
    $\mathbf{X} \leftarrow \boldsymbol{\Sigma}_{\mathcal{P}}$
  **end if**
  $\{p(\mathbf{v}_t \,|\, \mathbf{v}_{1:t-1}), \mathbf{f}_t, \mathbf{F}_t\} \leftarrow$ COND $\{\mathbf{x}, \mathbf{X}, \mathbf{B}, \boldsymbol{\Sigma}_{\mathcal{V}}, \mathbf{v}_t, \mathbf{0}\}$
  $l \leftarrow l + \log p(\mathbf{v}_t \,|\, \mathbf{v}_{1:t-1})$
**end for**
```

Algorithm 3 RTS backward pass. This algorithm computes the mean $\mathbf{g}_t$ and the covariance $\mathbf{G}_t$ of $\mathbf{h}_t$ under $p(\mathbf{h}_t \,|\, \mathbf{v}_{1:t})$.

```
$\mathbf{g}_T \leftarrow \mathbf{f}_T$
$\mathbf{G}_T \leftarrow \mathbf{F}_T$
**for** $t \leftarrow T - 1$ **to** 1 **do**
  $\{\alpha, \mathbf{g}_t, \mathbf{G}_t\} \leftarrow$ COND $\{\mathbf{f}_t, \mathbf{F}_t, \mathbf{A}, \boldsymbol{\Sigma}_{\mathcal{H}}, \mathbf{g}_{t+1}, \mathbf{G}_{t+1}\}$
**end for**
```

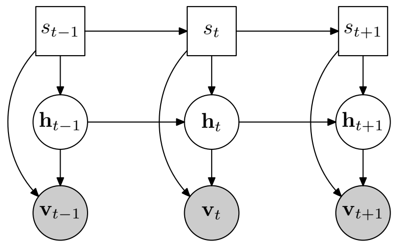


Figure 5.2: DBN representation of the SLDS; s_t and $\mathbf{h}_t$ represent the discrete and continuous hidden variables and $\mathbf{v}_t$ the observation.

a switch variable $s_t \in \{1, \ldots, S\}$ describes which of the LDSs is to be used. Formally, this is achieved by rewriting Equations 5.2 and 5.1 so that they depend on s_t:

$$\mathbf{h}_t = \mathbf{A}_{s_t}\mathbf{h}_{t-1} + \boldsymbol{\eta}_t^{\mathcal{H}} \quad \text{with} \quad \boldsymbol{\eta}_t^{\mathcal{H}} \sim \mathcal{N}\big(\mathbf{0}, \boldsymbol{\Sigma}_{\mathcal{H},s_t}\big) \tag{5.14}$$

$$\mathbf{v}_t = \mathbf{B}_{s_t}\mathbf{h}_t + \boldsymbol{\eta}_t^{\mathcal{V}} \quad \text{with} \quad \boldsymbol{\eta}_t^{\mathcal{V}} \sim \mathcal{N}\big(\mathbf{0}, \boldsymbol{\Sigma}_{\mathcal{V},s_t}\big). \tag{5.15}$$

The dynamics of the switch state is assumed Markovian and is modelled by a *discrete* transition distribution $p(s_t \,|\, s_{t-1})$. The addition of the *continuous* transition distribution $p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}, s_t)$ and the emission distribution $p(\mathbf{v}_t \,|\, \mathbf{h}_t, s_t)$, corresponding to Equations 5.14 and 5.15, yields the joint distribution

$$p(\mathbf{v}_{1:T}, \mathbf{h}_{1:T}, s_{1:T}) = \prod_{t=1}^{T} p(\mathbf{v}_t \,|\, \mathbf{h}_t, s_t)\, p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}, s_t)\, p(s_t \,|\, s_{t-1}) \tag{5.16}$$

where $p(\mathbf{h}_1 \,|\, \mathbf{h}_0, s_1) \equiv p(\mathbf{h}_1 \,|\, s_1)$ and $p(s_1 \,|\, s_0) \equiv p(s_1)$ are given prior distributions. Graphically, the SLDS can be represented as the DBN of Figure 5.2. The likelihood of an observed sequence $\mathbf{v}_{1:T}$ is obtained by integrating over all the possible hidden sequences of discrete and continuous states:

$$p(\mathbf{v}_{1:T}) = \sum_{s_{1:N}} \int_{\mathbf{h}_{1:T}} p(\mathbf{v}_{1:T}, \mathbf{h}_{1:T}, s_{1:N}).$$

If we use the particular parameters setting (5.4) and (5.5) and, as for the SAR-HMM, introduce segments over which the state cannot change, then the SLDS reduces to the AR-SLDS.

5.3.1 Inference in the SLDS

Computing the posterior distribution of the continuous and discrete hidden variables $\mathbf{h}_t$ and s_t is intractable in the SLDS [46]. For example, the posterior

$$p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:T}) = \sum_{\substack{s_{1:t-1} \\ s_{t+1:T}}} p(\mathbf{h}_t \,|\, s_{1:T}, \mathbf{v}_{1:T})\, p(s_{1:T} \,|\, \mathbf{v}_{1:T})$$

is a mixture of S^{T-1} Gaussians. Hence, the number of computations required to evaluate the posterior distribution exactly scales exponentially with the length T of the sequence considered. In practice, approximations are therefore considered. Performing approximate inference in the SLDS is a long-standing problem [6] for which an extensive literature exists. In the following section, we briefly review some of the most well-known approximation techniques and present their limitations.

5.3.2 Existing Approximate Inference Methods

Algorithms for performing approximate inference in the SLDS can be classified into four classes depending on whether they are generic or specific and deterministic or not. Specific and deterministic algorithms [6, 52, 45] are mainly based on a forward-backward recursion similar to the RTS method. Most of them share the common property of approximating the exponentially large number of mixture components by a mixture with fewer components. Those methods distinguish themselves from one another by the additional approximations they make.

A good example of a generic and deterministic approach is the *Expectation Propagation* (EP) algorithm [50]. Unlike the specific methods, which try to approximate the exact forward and backward passes, EP is based on a consistency criterion, collapsing to a single Gaussian at each stage [72]. Difficulties with EP are inherent numerical instabilities arising from frequent conversions between canonical and moment representations of Gaussians, and also the difficulty of generalising the approach simply to deal with mixture representations.

The most widely used non-deterministic algorithms are based on Monte-Carlo methods [22], also known as *Particle Filters/Smoothers*. Whilst potentially powerful, these non-analytic methods typically suffer in high-dimensional hidden spaces since they are often based on naive importance sampling, which restricts their practical use. Specific deterministic methods are preferred because the approximation is a mixture of non-trivial distributions, which is better at capturing the variability of the posterior. The Rao-Blackwellized particle filter [23] is a more specific algorithm which attempts to alleviate the difficulty of sampling in high-dimensional spaces by explicitly integrating over the continuous hidden variable. The mixture Kalman filters [19] is another technique which can potentially improve the performance of Monte-Carlo approaches in high dimensional space, since it represents the posterior as a mixture of non-trivial distributions. By increasing the number of mixture components, one can theoretically get increasingly better approximations. In practice, an appropriate resampling procedure must be used to avoid the degeneration of the estimate with time [41, 47, 22].

Gibbs sampling provides an alternative non-deterministic procedure [17]. For a fixed state sequence $s_{1:T}$, $p(\mathbf{v}_{1:T} \,|\, s_{1:T})$ is easily computable since this is just the likelihood of a LDS. We could therefore sample from the posterior $p(s_{1:T} \,|\, \mathbf{v}_{1:T}) \propto p(\mathbf{v}_{1:T} \,|\, s_{1:T})\, p(s_{1:T})$ directly. This procedure may work well provided that the initial setting of $s_{1:T}$ is in a region of high probability, otherwise sampling by such individual coordinate updates may be extremely inefficient. Another related Gibbs procedure consists in alternately sample from $p(s_{1:T} \,|\, \mathbf{h}_{1:T}, \mathbf{v}_{1:T})$ and $p(\mathbf{h}_{1:T} \,|\, s_{1:T}, \mathbf{v}_{1:T})$.

5.3.3 The Expectation Correction Algorithm

EC follows the same approach as the RTS algorithm. The forward pass computes the filtered posterior $p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:t})$ and the backward pass corrects this to form the smoothed posterior $p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:T})$. Without loss of generality, we may write the filtered and smoothed posterior as a product of a continuous and a discrete distribution:

$$p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:t}) = p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:t})\, p(s_t \,|\, \mathbf{v}_{1:t}),$$
$$p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:T}) = p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:T})\, p(s_t \,|\, \mathbf{v}_{1:T}).$$

Our approach will represent both the filtered and smoothed posteriors as a finite mixture of Gaussians. Formally, this can be achieved using,

$$p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:t}) = \sum_i p(\mathbf{h}_t \,|\, i_t, s_t, \mathbf{v}_{1:t})\, p(i_t \,|\, s_t, \mathbf{v}_{1:t})$$

see [3, 7], for example. In our exposition we use only a *single* Gaussian—extending EC to the mixture case is straightforward [7] and we prefer to present the central idea without the extra notational complexity of dealing with mixtures.

Forward Pass

If we denote by $\mathbf{x}_t = \{\mathbf{h}_t, s_t\}$ the hidden variables, the equivalent of Equation 5.6 for the SLDS, reads

$$p(\mathbf{x}_t \,|\, \mathbf{v}_{1:t}) \propto p(\mathbf{v}_t \,|\, \mathbf{x}_t) \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})\, p(\mathbf{x}_{t-1} \,|\, \mathbf{v}_{1:t-1}).$$

After expansion, the rhs becomes

$$\sum_{s_{t-1}} p(s_t \,|\, s_{t-1})\, p(s_{t-1} \,|\, \mathbf{v}_{1:t-1})\, p(\mathbf{v}_t \,|\, \mathbf{h}_t, s_t) \int_{\mathbf{h}_{t-1}} p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}, s_t)\, p(\mathbf{h}_{t-1} \,|\, s_{t-1}, \mathbf{v}_{1:t})$$

where $p(\mathbf{h}_{t-1} \,|\, s_{t-1}, \mathbf{v}_{1:t-1})$ and $p(s_{t-1} \,|\, \mathbf{v}_{1:t-1})$ are the continuous and discrete part of the filtered posterior at the previous time step. After carrying out the integration over $\mathbf{h}_{t-1}$ and grouping similar factors, we obtain

$$\begin{aligned} p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:t}) &\propto \sum_{s_{t-1}} p(s_{t-1}, s_t \,|\, \mathbf{v}_{1:t-1})\, p(\mathbf{v}_t, \mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t-1}) \\ &\propto \sum_{s_{t-1}} p(s_{t-1}, s_t \,|\, \mathbf{v}_{1:t})\, p(\mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t}) \end{aligned} \tag{5.17}$$

where $p(\mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t})$ corresponds to the filtered posterior of the LDS. It is given by Equation 5.6 which, in this case, reads

$$p(\mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t}) \propto p(\mathbf{v}_t \,|\, \mathbf{h}_t, s_t) \int_{\mathbf{h}_{t-1}} p(\mathbf{h}_t \,|\, \mathbf{h}_{t-1}, s_t)\, p(\mathbf{h}_{t-1} \,|\, s_{t-1}, \mathbf{v}_{1:t}). \tag{5.18}$$

The discrete component $p(s_{t-1}, s_t \,|\, \mathbf{v}_{1:t})$ is proportional to

$$p(\mathbf{v}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t-1})\, p(s_t \,|\, s_{t-1})\, p(s_{t-1} \,|\, \mathbf{v}_{1:t-1}) \tag{5.19}$$

where $p(\mathbf{v}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t-1})$ is obtained by integrating the rhs of Equation (5.18) over $\mathbf{h}_t$. The recursion is initialised with

$$p(\mathbf{h}_1, s_1 \,|\, \mathbf{v}_1) \propto p(\mathbf{v}_1 \,|\, \mathbf{h}_1, s_1)\, p(\mathbf{h}_1 \,|\, s_1)\, p(s_1).$$

The filtered posterior at time t, as given by Equation 5.17, is a mixture of Gaussians. At each time step the number of mixture components is multiplied by S and thus grows exponentially with t. A simple approximate remedy is to collapse the mixture obtained to a mixture with fewer components. This corresponds to the so-called *Gaussian Sum Approximation* (GSA) [3] which is a form of *Assumed Density Filtering* [50]. GSA is a common ingredient of most of the deterministic approximation algorithms based on a forward-backward recursion. It reduces the complexity of the forward pass to $\mathcal{O}(I \cdot S \cdot T)$, where I is the number of mixture components of the collapsed distribution.

Backward Pass

The generic form of Equation 5.7 for the SLDS reads

$$p(\mathbf{x}_t \,|\, \mathbf{v}_{1:T}) = \int_{\mathbf{x}_{t+1}} p(\mathbf{x}_t \,|\, \mathbf{x}_{t+1}, \mathbf{v}_{1:t})\, p(\mathbf{x}_{t+1} \,|\, \mathbf{v}_{1:T}).$$

After expansion, the rhs becomes

$$\sum_{s_{t+1}} p(s_{t+1} \,|\, \mathbf{v}_{1:T}) \int_{\mathbf{h}_{t+1}} p(\mathbf{h}_t, s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t})\, p(\mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T}) \tag{5.20}$$

where $p(\mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T})$ and $p(s_{t+1} \,|\, \mathbf{v}_{1:T})$ are the continuous and discrete parts of the smoothed posterior at the next time step. The integral in (5.20) can also be written as

$$\big\langle p(\mathbf{h}_t, s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle = \big\langle p(\mathbf{h}_t \,|\, \mathbf{h}_{t+1}, s_t, s_{t+1}, \mathbf{v}_{1:t})\, p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle$$

where the average is taken with respect to $p(\mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T})$. This is difficult to evaluate because of the dependency of s_t on $\mathbf{h}_{t+1}$. In its most simple form EC approximates the average by

$$\big\langle p(\mathbf{h}_t \,|\, \mathbf{h}_{t+1}, s_t, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle \big\langle p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle. \tag{5.21}$$

This is particularly appealing since the first factor corresponds to the smoothed posterior of the LDS, as given by Equation 5.7, and can be evaluated by conditioning on $\mathbf{h}_{t+1}$ the joint distribution

$$p(\mathbf{h}_t, \mathbf{h}_{t+1} \,|\, s_t, s_{t+1}, \mathbf{v}_{1:t}) = p(\mathbf{h}_{t+1} \,|\, \mathbf{h}_t, s_{t+1})\, p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:t}) \tag{5.22}$$

which is obtained by forward propagation. The second factor in (5.21) is still difficult to evaluate exactly. Formally, this term corresponds to

$$\big\langle p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle \equiv p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:T}).$$

The distinguishing feature of EC from other methods, such as *Generalised Pseudo Bayes* (GPB) [6], is in the approximation of $p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:T})$. GPB uses Kim's approximation [38, 39], $p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:T}) \approx p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:t})$, which depends only on the filtered posterior for s_t and does not include any information coming from the continuous variable $\mathbf{h}_{t+1}$. Since $p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:t}) \propto p(s_{t+1} \,|\, s_t)\, p(s_t \,|\, \mathbf{v}_{1:t})$ computing the smoothed recursion for the switch states in GPB is equivalent to running the RTS backward pass on a HMM. This represents a potentially severe loss of information from the future and means any information from the continuous variables cannot be used when 'correcting' the filtered results $p(s_t \,|\, \mathbf{v}_{1:t})$ into smoothed posteriors $p(s_t \,|\, \mathbf{v}_{1:T})$. In contrast, EC attempts to preserve future information passing through the continuous variables. The simplest approach within EC is to use the approximation

$$\begin{aligned} p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:T}) &\equiv \big\langle p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1} \mathbf{v}_{1:t}) \big\rangle \\ &\approx p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big|_{\mathbf{h}_{t+1} = \langle \mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T} \rangle} \end{aligned} \tag{5.23}$$

where $\langle \mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T} \rangle$ is the mean of $\mathbf{h}_{t+1}$ with respect to $p(\mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T})$. More sophisticated approximations schemes—which take into account the covariance of $\mathbf{h}_{t+1}$, for example—may be applied, but practically the proposed one has proven to be accurate enough [7]. Finally, the rhs of Equation 5.23 can be evaluated by considering the joint distribution

$$p(\mathbf{h}_{t+1}, s_t \,|\, s_{t+1}, \mathbf{v}_{1:t}) \propto p(\mathbf{h}_{t+1} \,|\, s_t, s_{t+1}, \mathbf{v}_{1:t})\, p(s_{t+1} \,|\, s_t)\, p(s_t \,|\, \mathbf{v}_{1:t})$$

where $p(\mathbf{h}_{t+1} \,|\, s_t, s_{t+1}, \mathbf{v}_{1:t})$ is obtained by marginalising Equation 5.22 over $\mathbf{h}_t$. In summary, the smoothed posterior, as given by (5.20), is a mixture of Gaussians of the form

$$p(\mathbf{h}_t, s_t \,|\, \mathbf{v}_{1:T}) = \sum_{s_{t+1}} p(s_t, s_{t+1} \,|\, \mathbf{v}_{1:T})\, p(\mathbf{h}_t \,|\, s_t, s_{t+1}, \mathbf{v}_{1:T}). \tag{5.24}$$

In its most generic form, EC approximates each factor by

$$\begin{aligned} p(s_t, s_{t+1} \,|\, \mathbf{v}_{1:T}) &\approx p(s_{t+1} \,|\, s_t) \big\langle p(s_t \,|\, \mathbf{h}_{t+1}, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle \\ p(\mathbf{h}_t \,|\, s_t, s_{t+1}, \mathbf{v}_{1:T}) &\approx \big\langle p(\mathbf{h}_t, \mathbf{h}_{t+1}, s_t, s_{t+1}, \mathbf{v}_{1:t}) \big\rangle \end{aligned}$$

where the averages are taken with respect to $p(\mathbf{h}_{t+1} \,|\, s_{t+1}, \mathbf{v}_{1:T})$. The backward recursion is initialised with the filtered posterior obtained at the T-th step, since both filtered and smoothed posteriors are equal at that point. As in the forward pass, the number of mixture components is multiplied by S at each iteration. Hence, to retain tractability, the mixture in Equation 5.24 is collapsed to a mixture with fewer components.

Implementation

Algorithms 4 and 5 give the pseudo-code of EC forward and backward passes. In Algorithm 5, the prefactor α in the expression $p_{s_t, s_{t+1}} \leftarrow \alpha\, p(s_{t+1} \,|\, s_t)\, p(s_t \,|\, \mathbf{v}_{1:t})$ differentiates EC from GPB; this corresponds to the approximation (5.23) which, in GPB, is replaced by $p(s_t \,|\, s_{t+1}, \mathbf{v}_{1:t})$. The COLLAPSE

Algorithm 4 EC forward pass. This algorithm computes the filtered posterior $p(s_t \mid \mathbf{v}_{1:t})$, the mean $\mathbf{f}_{s_t}$ and the covariance $\mathbf{F}_{s_t}$ of $\mathbf{h}_t$ under $p(\mathbf{h}_t \mid s_t, \mathbf{v}_{1:t})$, as well as the log-likelihood $l \equiv \log p(\mathbf{v}_{1:T})$. The prior mean and covariance are denoted by $\boldsymbol{\mu}_{\mathcal{P}}$ and $\boldsymbol{\Sigma}_{\mathcal{P}}$.

```
$l \leftarrow 0$

for $t \leftarrow 1$ to T do
  for all $(s_{t-1}, s_t)$ do
    if $t > 1$ then
      $\mathbf{x} \leftarrow \mathbf{A}_{s_t}\mathbf{f}_{s_{t-1}}$
      $\mathbf{X} \leftarrow \mathbf{A}_{s_t}\mathbf{F}_{s_{t-1}}\mathbf{A}_{s_t}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathcal{H},s_t}$
    else
      $\mathbf{x} \leftarrow \boldsymbol{\mu}_{\mathcal{P}}(s_1)$
      $\mathbf{X} \leftarrow \boldsymbol{\Sigma}_{\mathcal{P}}(s_1)$
    end if
    $\left\{\alpha, \boldsymbol{\mu}_{s_{t-1},s_t}, \boldsymbol{\Sigma}_{s_{t-1},s_t}\right\} \leftarrow \textsc{cond}\left\{\mathbf{x}, \mathbf{X}, \mathbf{B}_{s_t}, \boldsymbol{\Sigma}_{\mathcal{V},s_t}, \mathbf{v}_t, \mathbf{0}\right\}$
    $p_{s_{t-1},s_t} \leftarrow \alpha\, p(s_t \mid s_{t-1})\, p(s_{t-1} \mid \mathbf{v}_{1:t-1})$
  end for
  $p(\mathbf{v}_t \mid \mathbf{v}_{1:t-1}) \leftarrow \sum_{s_{t-1},s_t} p_{s_{t-1},s_t}$
  for all $(s_{t-1}, s_t)$ do
    $p_{s_{t-1},s_t} \leftarrow p_{s_{t-1},s_t} \,/\, p(\mathbf{v}_t \mid \mathbf{v}_{1:t-1})$
  end for
  for all $s_t$ do
    $p(s_t \mid \mathbf{v}_{1:t}) \leftarrow \sum_{s_{t-1}} p_{s_{t-1},s_t}$
    $p(s_{t-1} \mid s_t, \mathbf{v}_{1:t}) \leftarrow p_{s_{t-1},s_t} \,/\, p(s_t \mid \mathbf{v}_{1:t})$
    $\left\{\mathbf{f}_{s_t}, \mathbf{F}_{s_t}\right\} \leftarrow \textsc{collapse}\left\{p(s_{t-1} \mid s_t, \mathbf{v}_{1:t}), \boldsymbol{\mu}_{s_{t-1},s_t}, \boldsymbol{\Sigma}_{s_{t-1},s_t}\right\}$
  end for
  $l \leftarrow l + \log p(\mathbf{v}_t \mid \mathbf{v}_{1:t-1})$
end for
```

routine collapses the mixture of S Gaussians passed as arguments to a mixture with less components. For example, in the forward pass, for each s_t, the mixture we want to collapse is

$$p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:t}) = \sum_{s_{t-1}} p(\mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t})\, p(s_{t-1} \,|\, s_t, \mathbf{v}_{1:t}).$$

The case of collapsing to a single Gaussian can be formalised by considering the minimisation of the KL divergence

$$\mathrm{KL}\big(q(\mathbf{h}_t \,|\, s_t) \,||\, p(\mathbf{h}_t \,|\, s_t)\big)$$

where q is a Gaussian distribution with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$. Differentiating the KL divergence with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and setting the result equal to zero, yields the moment matching formulae

$$\boldsymbol{\mu} = \sum_{s_{t-1}} p(s_{t-1} \,|\, s_t)\, \boldsymbol{\mu}_{s_{t-1},s_t}$$
$$\boldsymbol{\Sigma} = \sum_{s_{t-1}} p(s_{t-1} \,|\, s_t)\, \big(\boldsymbol{\Sigma}_{s_{t-1},s_t} + \boldsymbol{\mu}_{s_{t-1},s_t}\boldsymbol{\mu}^{\mathsf{T}}_{s_{t-1},s_t}\big) - \boldsymbol{\mu}\boldsymbol{\mu}^{\mathsf{T}}$$

where $\boldsymbol{\mu}_{s_{t-1},s_t}$ and $\boldsymbol{\Sigma}_{s_{t-1},s_t}$ are the mean and covariance of $\mathbf{h}_t$ under the posterior $p(\mathbf{h}_t \,|\, s_{t-1}, s_t, \mathbf{v}_{1:t})$. The KL approach is difficult to apply to the case of collapsing to a mixture and, in practice, heuristics are therefore considered instead. For example, amongst the S components, one may retain the $I-1$ most likely and merge the rest into a single Gaussian. To simplify exposition, Algorithms 4 and 5 presents the case where the mixture is collapsed to a single Gaussian.

Algorithm 5 EC backward pass. This algorithm computes the smoothed posterior $p(s_t \,|\, \mathbf{v}_{1:T})$, the mean $\mathbf{g}_{s_t}$ and the covariance $\mathbf{G}_{s_t}$ of $\mathbf{h}_t$ under $p(\mathbf{h}_t \,|\, s_t, \mathbf{v}_{1:t})$.

```
for all $s_T$ do
    $\mathbf{g}_{s_T} \leftarrow \mathbf{f}_{s_T}$
    $\mathbf{G}_{s_T} \leftarrow \mathbf{F}_{s_T}$
end for
for $t \leftarrow T-1$ to 1 do
    for all $(s_t, s_{t+1})$ do
        $\{\alpha, \boldsymbol{\mu}_{s_t,s_{t+1}}, \boldsymbol{\Sigma}_{s_t,s_{t+1}}\} \leftarrow \textsc{cond}\,\{\mathbf{f}_{s_t}, \mathbf{F}_{s_t}, \mathbf{A}_{s_{t+1}}, \boldsymbol{\Sigma}_{\mathcal{H},s_{t+1}}, \mathbf{g}_{s_{t+1}}, \mathbf{G}_{s_{t+1}}\}$
        $p_{s_t,s_{t+1}} \leftarrow \alpha\, p(s_{t+1} \,|\, s_t)\, p(s_t \,|\, \mathbf{v}_{1:t})$
    end for
    for all $(s_t, s_{t+1})$ do
        $p_{s_t,s_{t+1}} \leftarrow p_{s_t,s_{t+1}} \,/\, \sum_{s_t} p_{s_t,s_{t+1}}$
    end for
    for all $(s_t, s_{t+1})$ do
        $p_{s_t,s_{t+1}} \leftarrow p_{s_t,s_{t+1}} \cdot p(s_{t+1} \,|\, \mathbf{v}_{1:T})$
    end for
    for all $s_t$ do
        $p(s_t \,|\, \mathbf{v}_{1:T}) \leftarrow \sum_{s_{t+1}} p_{s_t,s_{t+1}}$
        $p(s_{t+1} \,|\, s_t, \mathbf{v}_{1:T}) \leftarrow p_{s_t,s_{t+1}} \,/\, p(s_t \,|\, \mathbf{v}_{1:T})$
        $\{\mathbf{g}_{s_t}, \mathbf{G}_{s_t}\} \leftarrow \textsc{collapse}\,\{p(s_{t+1} \,|\, s_t, \mathbf{v}_{1:T}), \boldsymbol{\mu}_{s_t,s_{t+1}}, \boldsymbol{\Sigma}_{s_t,s_{t+1}}\}$
    end for
end for
```

5.3.4 Comparison with Existing Methods

To compare the quality of the estimate provided by EC and the various other approximation algorithms, we sequentially generated hidden states $\mathbf{h}_t$, s_t and observations $\mathbf{v}_t$ from a known model. Given

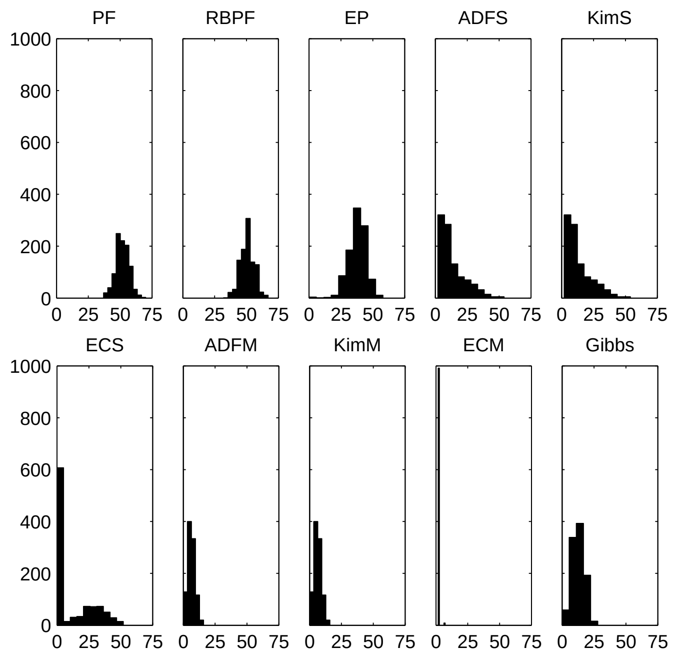


Figure 5.3: Number of errors in estimating the state sequence $s_{1:T}$ for $S = 2$ and $T = 100$. Hence 50 errors corresponds to random guessing. Plotted are the histograms of the errors over 1000 experiments. (PF) Particle Filter with 1000 particles, (RBF) Rao-Blackwellised PF with 500 particles, (EP) Expectation Propagation. (ADFS/M) Assumed Density Filtering using collapse to a single/mixture of Gaussians. (KimS/M) Kim's smoother using the results from ADFS/M. (ECS/M) Expectation Correction using a single/mixture of Gaussians. (Gibbs) Gibbs sampling initialised with ADFM. Models able to deal with mixtures were using four components.

only the model and the observations, the task was then to infer $p(\mathbf{h}_t, s_t \mid \mathbf{v}_{1:T})$. Since the exact computation is exponential in T, a formally exact evaluation of the method is infeasible. A possible alternative is to assume that the sampled state sequence is the most likely sequence and to perform a Viterbi decoding [69, 58]. However, since our goal was to compare the quality of the posterior estimate provided by the various algorithms, we preferred using a simpler alternative where, for each time step, the sample state s_t was compared to the most probable posterior smoothed estimate $\arg\max_{s_t} p(s_t \mid v_{1:T})$. We considered a two states ($S = 2$) model where the dimension of the hidden and observed variables were $H = 30$ and $V = 1$ respectively, and high output noise ($\mathbf{\Sigma}_\mathcal{V} = 30\,\mathbf{I}_V$) was used. The transition matrices $\mathbf{A}_s$ and the projection matrices $\mathbf{B}_s$ were sampled at random (see [7] for details); each random sampling of parameters generated a problem instance on which the rival algorithms were evaluated.

A histogram of inference performance over 1000 problem instances was constructed, as displayed in Figure 5.3. All deterministic algorithms were initialised using the corresponding filtered results, as was Gibbs sampling. For the Particle Filter methods, 1000 particles were used, with Kitagawa resampling scheme [41]. For the Rao-Blackwellized Particle Filter [23], 500 particles were used, again with Kitagawa resampling. We include the Particle Filter methods merely for a point of comparison with ADF, since they are not designed to approximate the smoothed estimate. Gibbs sampling was initialised with the most likely switch states $s_{1:T}$ obtained from the filtered results of ADFM. The sampling was first carried out forwards in time until the end of the chain and then backward up to the first time step. This procedure was repeated 100 times and the mean over the last 80 sweeps was finally used as the posterior mean approximation.

As expected Kim's GPB method does not improve much on the filtered results, nor is Gibbs sampling able to escape the sharp local maxima found by filtering. The poor performance of EP is most likely due to severe numerical instabilities, particularly when a single Gaussian is insufficient to represent the posterior well. EC performs admirably using a single Gaussian and improves dramatically when using a mixture of four Gaussians in this example. The Particle Filter methods most likely failed since the hidden dimension is too high to be explored well with only 1000 particles. The running time of all algorithms was set to be roughly equal to that of ECM.

5.4 Summary

The SLDS extends the LDS by introducing a switch variable which allows each observed sample to be modelled by a potentially different LDS. Contrary to the LDS, where inference can be carried out exactly by means of the RTS algorithm, finding the posterior distribution of the hidden variable in the SLDS is computationally intractable. We presented the EC algorithm, an extension of the RTS procedure to the SLDS which addresses some of the limitations of current approximation methods. By being specifically designed for the SLDS, EC better captures the variability of the posterior distribution and circumvents the difficulties inherent to more generic methods such as EP and Particles Filters/Smoothers. EC is similar to GPB; both algorithms use the same forward pass, but EC backward pass is more accurate because it better preserves the information carried by the continuous variable. EC is not limited to the simple approximations (5.21) and (5.23), but can readily be extended to use more elaborate schemes [7]. In thesis, for all the experiments involving the use of SLDSs, we used EC in its most simple form—with collapse to a single Gaussian—in order to save computational time and memory. This scheme proved accurate enough and no numerical instabilities were encountered. It was also significantly much faster than particle-based approaches.

Chapter 6

A Bayesian Alternative to Gain Adaptation

The material presented in this chapter has been published in [48].

6.1 Introduction

Whilst useful in practice, Gain Adaptation (GA) does not fit into the usual machine learning framework since, formally, model parameters may only be set on the basis of training data. Otherwise, in flexible models, setting model parameters on the basis of test data may lead to overfitting. In this chapter, we consider a statistically principled alternative Bayesian approach to GA which consists in specifying a prior probability distribution on the model parameters. This approach has two potential benefits over GA: the variation of the gain can be explicitly controlled and the AR coefficients are allowed to change, which may be useful to model inter and intra speaker variations for example.

6.2 The Bayesian AR Process

In the formulation of the AR process presented in Section 3.2, the AR coefficients $\mathbf{c}$ and the innovation variance σ^2 were considered as free parameters whose setting had to be learned from data. Although theoretically the optimal setting can be found from training data only, in practice, the amplitude of the speech signal can vary significantly and better accuracy can often be obtained by allowing the parameters—in the case of GA, the innovation variance—to be slightly adjusted to a sequence $y_{1:T}$. A possible principled way to allow variability into the AR process is to treat the parameters as *random variables* whose probability distributions are controlled by hyper-parameters. If we define $\nu = 1/\sigma^2$ and write the dependency of $p(y_t \,|\, \tilde{\mathbf{y}}_t)$ on $\mathbf{c}$ and ν explicitly, then the emission probability given by Equation 3.2 becomes

$$p(y_t \,|\, \mathbf{c}, \nu, \tilde{\mathbf{y}}_t) = \frac{\nu^{1/2}}{\sqrt{2\pi}} \exp\left\{-\frac{\nu}{2}(y_t - \mathbf{c}^{\mathsf{T}}\tilde{\mathbf{y}}_t)^2\right\}. \tag{6.1}$$

After writing the priors on the parameters explicitly, Equation 3.3 reads

$$p(y_{1:T}, \mathbf{c}, \nu) = p(\mathbf{c}, \nu) \prod_{t=1}^{T} p(y_t \,|\, \mathbf{c}, \nu, \tilde{\mathbf{y}}_t).$$

The new factor $p(\mathbf{c}, \nu)$ defines a prior on the AR coefficients $\mathbf{c}$ and the inverse variance ν. In order to keep the model tractable, we choose *conjugate* priors

$$\mathbf{c} \,|\, \nu \sim \mathcal{N}(\boldsymbol{\mu}, \nu^{-1}\boldsymbol{\Sigma}) \quad \text{and} \quad \nu \sim \mathcal{G}(\alpha, \beta)$$

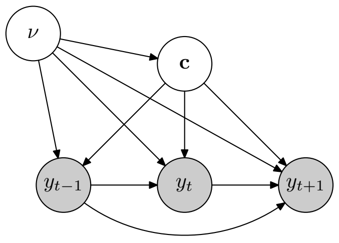


Figure 6.1: DBN representation of a second order Bayesian AR process; $\mathbf{c}$ represents the AR coefficients, $\nu = 1/\sigma^2$ the inverse innovation variance and y_t the observed waveform sample.

where $\mathcal{G}(\alpha, \beta)$ is the Gamma distribution defined as

$$\nu \sim \mathcal{G}(\alpha, \beta) \quad \Leftrightarrow \quad p(\nu) = \frac{\beta^\alpha}{\Gamma(\alpha)} \nu^{\alpha-1} e^{-\beta\nu}$$

and $\Gamma(\cdot)$ is the Gamma function [1]. Choosing conjugate priors is particularly useful in this case because it will allow us to later compute the posterior distribution of the parameters $p(\mathbf{c}, \nu \,|\, y_{1:T})$ exactly. Graphically, the Bayesian AR process can be depicted by the DBN of Figure 6.1. The likelihood of an observed sequence $y_{1:T}$ is given by integrating out the parameters in the joint distribution $p(y_{1:T}, \mathbf{c}, \nu)$:

$$p(\mathbf{y}_{1:T}) = \int_{\mathbf{c},\nu} p(\mathbf{y}_{1:T} \,|\, \mathbf{c}, \nu)\, p(\mathbf{c} \,|\, \nu)\, p(\nu).$$

Contrary to the standard AR process, no GA is required since the priors already account for possible variations of the parameters. Instead of optimising the AR coefficients and the innovation variance directly, we optimise the hyper-parameters $\Psi = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \beta\}$.

6.2.1 Parameter Optimisation & Inference

Following our usual approach in the presence of hidden variables, we use the EM algorithm to find update equations for the hyper-parameters Ψ. In this case, Equation 2.12 reads[1]

$$\Psi^{i+1} = \arg\max_{\Psi} \big\langle \log p(y_{1:T}, \mathbf{c}, \nu \,|\, \Psi) \big\rangle_{p(\mathbf{c},\nu \,|\, y_{1:T}, \Psi^i)}. \tag{6.2}$$

This requires inferring the posterior distribution of the parameters $p(\mathbf{c}, \nu \,|\, y_{1:T})$. Since we chose conjugate priors, the posterior distribution has the same structure as the prior, i.e.,

$$\mathbf{c} \,|\, \nu, y_{1:T} \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \nu^{-1}\hat{\boldsymbol{\Sigma}}) \quad \text{and} \quad \nu \,|\, y_{1:T} \sim \mathcal{G}(\hat{\alpha}, \hat{\beta})$$

where $\hat{\boldsymbol{\mu}}$, $\hat{\boldsymbol{\Sigma}}$, $\hat{\alpha}$ and $\hat{\beta}$ are parameters we need to calculate[2]. A possible way to compute $p(\mathbf{c} \,|\, \nu, y_{1:T})$ is by means of the forward recursion

$$p(\mathbf{c} \,|\, \nu, y_{1:t}) \propto p(y_t \,|\, \mathbf{c}, \nu)\, p(\mathbf{c} \,|\, \nu, y_{1:t-1}).$$

This is analogous to the forward pass for the LDS (Section 4.2.2). If we denote by $\boldsymbol{\mu}_t$ and $\nu^{-1}\boldsymbol{\Sigma}_t$ the mean and covariance of $\mathbf{c}$ under $p(\mathbf{c} \,|\, \nu, y_{1:t})$, then, since the joint distribution $p(y_t, \mathbf{c} \,|\, \nu, y_{1:t-1})$ is Gaussian, after conditioning on y_t, we obtain

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \langle \Delta\mathbf{c}^{\mathsf{T}} \Delta y_t \rangle \langle \Delta y_t^2 \rangle^{-1} (y_t - \langle y_t \rangle) \tag{6.3}$$

$$\nu^{-1}\boldsymbol{\Sigma}_t = \nu^{-1}\boldsymbol{\Sigma}_{t-1} - \langle \Delta\mathbf{c}^{\mathsf{T}} \Delta y_t \rangle \langle \Delta y_t^2 \rangle^{-1} \langle \Delta y_t \Delta\mathbf{c} \rangle \tag{6.4}$$

[1]To simplify notation we consider the case of a single sequence.
[2]Unless otherwise specified, we will differentiate the prior from the posterior parameters be adding an hat on top of the latter.

where Δx is a shorthand for $x - \langle x \rangle$. $\langle y_t \rangle$ and Δy_t are given by

$$\langle y_t \rangle = \langle \mathbf{c}^\mathsf{T} \rangle \tilde{\mathbf{y}}_t + \langle \eta_t \rangle = \boldsymbol{\mu}_{t-1}^\mathsf{T} \tilde{\mathbf{y}}_t, \qquad \Delta y_t = y_t - \langle y_t \rangle = (\mathbf{c} - \boldsymbol{\mu}_{t-1})^\mathsf{T} \tilde{\mathbf{y}}_t + \eta_t.$$

The variance of y_t and the cross-covariance $\langle \Delta \mathbf{c}^\mathsf{T} \Delta y_t \rangle$ are therefore

$$\begin{aligned} \langle \Delta y_t^2 \rangle &= \tilde{\mathbf{y}}_t^\mathsf{T} \nu^{-1} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t + \nu^{-1}, \qquad\qquad & \langle \Delta \mathbf{c}^\mathsf{T} \Delta y_t \rangle = \nu^{-1} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t. \\ &= \nu^{-1} \big(\tilde{\mathbf{y}}_t^\mathsf{T} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t + 1 \big) & \end{aligned}$$

Hence, Equations 6.3 and 6.4 become

$$\begin{aligned} \boldsymbol{\mu}_t &= \boldsymbol{\mu}_{t-1} + \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t \big(\tilde{\mathbf{y}}_t^\mathsf{T} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t + 1 \big)^{-1} \big(y_t - \boldsymbol{\mu}_{t-1}^\mathsf{T} \tilde{\mathbf{y}}_t \big) \\ \boldsymbol{\Sigma}_t &= \boldsymbol{\Sigma}_{t-1} - \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t \big(\tilde{\mathbf{y}}_t^\mathsf{T} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t + 1 \big)^{-1} \tilde{\mathbf{y}}_t^\mathsf{T} \boldsymbol{\Sigma}_{t-1}. \end{aligned}$$

Those formulae can also be written more compactly as

$$\begin{aligned} \sigma_t^2 &= \tilde{\mathbf{y}}_t^\mathsf{T} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t + 1, & \mathbf{K}_t &= \frac{1}{\sigma_t^2} \boldsymbol{\Sigma}_{t-1} \tilde{\mathbf{y}}_t, \\ \boldsymbol{\mu}_t &= \big(\mathbf{I}_{R \times R} - \mathbf{K}_t \tilde{\mathbf{y}}_t^\mathsf{T} \big) \boldsymbol{\mu}_{t-1} + \mathbf{K}_t y_t, & \boldsymbol{\Sigma}_t &= \big(\mathbf{I}_{R \times R} - \mathbf{K}_t \tilde{\mathbf{y}}_t^\mathsf{T} \big) \boldsymbol{\Sigma}_{t-1} \end{aligned}$$

where $\mathbf{I}_{R \times R}$ denotes the $R \times R$ identity matrix. The recursion is initialised with $\boldsymbol{\mu}_0 \equiv \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_0 \equiv \boldsymbol{\Sigma}$ and, at the end of the recursion, we have $\hat{\boldsymbol{\mu}} \equiv \boldsymbol{\mu}_T$ and $\hat{\boldsymbol{\Sigma}} \equiv \boldsymbol{\Sigma}_T$. The parameters $\hat{\alpha}$ and $\hat{\beta}$ of $p(\nu \,|\, y_{1:T})$ can be obtained by considering

$$p(\nu \,|\, y_{1:T}) = p(\nu)\, p(y_{1:T} \,|\, \nu) \propto p(\nu) \prod_{t=1}^{T} p(y_t \,|\, \nu, y_{1:t-1})$$

where $p(y_t \,|\, \nu, y_{1:t-1})$ is a Gaussian distribution with mean $\langle y_t \rangle = \boldsymbol{\mu}_{t-1}^\mathsf{T} \tilde{\mathbf{y}}_t$ and variance $\langle \Delta y_t^2 \rangle = \nu^{-1} \sigma_t^2$. Hence

$$p(\nu \,|\, y_{1:T}) \propto \nu^{\alpha-1} e^{-\beta\nu} \prod_{t=1}^{T} \frac{\nu^{\frac{1}{2}}}{\sqrt{2\pi\sigma_t^2}} \exp\left\{ -\frac{\nu}{2\sigma_t^2} \big(y_t - \boldsymbol{\mu}_{t-1}^\mathsf{T} \tilde{\mathbf{y}}_t \big)^2 \right\} \tag{6.5}$$

which yields

$$\hat{\alpha} = \alpha + \frac{T}{2} \quad \text{and} \quad \hat{\beta} = \beta + \sum_{t=1}^{T} \frac{1}{2\sigma_t^2} \big(y_t - \boldsymbol{\mu}_{t-1}^\mathsf{T} \tilde{\mathbf{y}}_t \big)^2.$$

6.2.2 Parameter Updating

Update formulae for the parameters in Ψ can be obtained by differentiating Equation 6.2 with respect to each variable in Ψ. Since $p(\mathbf{c} \,|\, \nu, \Psi)$ is Gaussian, the update formulae for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ correspond to the empirical mean and covariance:

$$\boldsymbol{\mu} = \langle \mathbf{c} \rangle_q = \hat{\boldsymbol{\mu}} \quad \text{and} \quad \boldsymbol{\Sigma} = \big\langle \nu (\mathbf{c} - \boldsymbol{\mu})(\mathbf{c} - \boldsymbol{\mu})^\mathsf{T} \big\rangle_q = \hat{\boldsymbol{\Sigma}} \tag{6.6}$$

where $q(\mathbf{c}, \nu) \equiv p(\mathbf{c}, \nu \,|\, y_{1:T}, \Psi^i)$. For α and β, the function we need to optimise is

$$\mathcal{F}_\nu(\alpha, \beta) = \big\langle \alpha \log \beta - \log \Gamma(\alpha) + (\alpha - 1) \log \nu - \beta\nu \big\rangle_q.$$

Differentiating with respect to β and setting the result equal to zero yields $\beta = \alpha / \langle \nu \rangle_q$. Using this in $\mathcal{F}_\nu(\alpha, \beta)$, differentiating with respect to α and setting the result equal to zero, gives

$$\log \alpha - \psi(\alpha) = \log \langle \nu \rangle_q - \langle \log \nu \rangle_q \tag{6.7}$$

where $\psi(\cdot)$ is the Digamma function [1]. Whilst no explicit formula for α exists, Equation 6.7 is well-behaved and can be solved using Newton-Raphson's method, for example; see Appendix B for a detailed explanation. Although Equation 6.7 can be simplified by writing $\langle \nu \rangle$ and $\langle \log \nu \rangle$ in function of $\hat{\alpha}$ and $\hat{\beta}$, this form is more generic and generalises better, as we will see in the next section. A detailed explanation on how to compute $\langle \nu \rangle$ and $\langle \log \nu \rangle$ is given in Appendix A.

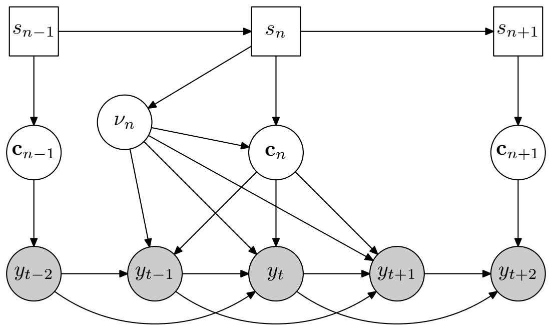


Figure 6.2: DBN representation of a second order Bayesian SAR-HMM; s_n represents the hidden switch state, $\mathbf{c}_n$ the AR coefficients, $\nu_n = 1/\sigma_n^2$ the inverse innovation variance and y_t the observed waveform sample.

6.3 The Bayesian SAR-HMM

A Bayesian treatment of the SAR-HMM can be achieved by introducing a *state dependent* prior distribution on the parameters of the underlying AR process. The sequence of observations $\mathbf{y}_n \equiv y_{t_n:t_{n+1}-1}$ belonging to the n-th segment is then modelled by an AR process whose coefficients $\mathbf{c}_n$ and inverse innovation variance ν_n are drawn randomly from a prior distribution conditioned on the switch state s_n. Furthermore, the dependency on the segment index n implies that the AR coefficients and the innovation variance can be different for each pair of segment and state. This scheme is similar to GA where a different gain was computed for each couple of segment and state. Formally the Bayesian SAR-HMM defines the joint distribution

$$p(\mathbf{y}_{1:N}, \mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N}) = \prod_{n=1}^{N} p(\mathbf{y}_n \mid \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_{t_n})\, p(\mathbf{c}_n, \nu_n \mid s_n)\, p(s_n \mid s_{n-1}) \tag{6.8}$$

where $p(s_1 \mid s_0) \equiv p(s_1)$ is a given prior distribution. Graphically, this can be depicted by the DBN of Figure 6.2. The main difference with the SAR-HMM (Figure 3.2) is that the *segment* emission distribution

$$p(\mathbf{y}_n \mid \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_{t_n}) = \prod_{t=t_n}^{t_{n+1}-1} p(y_t \mid \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t)$$

indirectly depends on s_n, through the prior $p(\mathbf{c}_n, \nu_n \mid s_n)$. The sample emission distribution $p(y_t \mid \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t)$ is given by Equation 6.1. As for the Bayesian LDS, choosing a Normal-Gamma conjugate prior of the form

$$\mathbf{c}_n \mid \nu_n, s_n \sim \mathcal{N}(\boldsymbol{\mu}_{s_n}, \nu_n^{-1}\boldsymbol{\Sigma}_{s_n}) \quad \text{and} \quad \nu_n \mid s_n \sim \mathcal{G}(\alpha_{s_n}, \beta_{s_n})$$

will later allow us to perform exact inference.

6.3.1 Parameter Optimisation & Inference

The free parameters of the Bayesian SAR-HMM are

$$\Psi = \bigcup_s \Big\{\boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s, \alpha_s, \beta_s, p(s_1 = s)\Big\} \cup \bigcup_{i,j} \Big\{p(s_t = j \mid s_{t-1} = i)\Big\}.$$

As for the SAR-HMM, training is achieved by means of the EM algorithm. This time however, the average must be carried out over all the possible sequences of parameters $\mathbf{c}_{1:N}$ and $\nu_{1:N}$ as well. Hence, the equivalent of Equation 2.12 for the Bayesian SAR-HMM reads[3]

$$\Psi^{i+1} = \arg\max_{\Psi} \left\langle \log p(\mathbf{y}_{1:N}, \mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N} \,|\, \Psi) \right\rangle_q \tag{6.9}$$

where $q(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N}) \equiv p(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N} \,|\, \mathbf{y}_{1:N}, \Psi^i)$. The required posterior distribution q can be computed with the RTS algorithm.

Forward Pass

The forward pass computes the filtered posterior $p(\mathbf{c}_n, \nu_n, s_n \,|\, \mathbf{y}_{1:n})$ by means of the recursion (compare with Equation 3.10)

$$p(\mathbf{c}_n, \nu_n, s_n \,|\, \mathbf{y}_{1:n}) \propto p(\mathbf{y}_n \,|\, \mathbf{c}_n, \nu_n)\, p(\mathbf{c}_n, \nu_n \,|\, s_n) \sum_{s_{n-1}} p(s_n \,|\, s_{n-1})\, p(s_{n-1} \,|\, y_{1:n-1})$$

where $p(s_n \,|\, y_{1:n-1})$ is obtained by integrating the filtered posterior at the previous segment over $\mathbf{c}_{n-1}$ and ν_n. The filtered posterior for the n-th segment can be written as

$$\begin{aligned} p(\mathbf{c}_n, \nu_n, s_n \,|\, \mathbf{y}_{1:n}) &= p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_{1:n})\, p(s_n \,|\, \mathbf{y}_{1:n}) \\ &= p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)\, p(s_n \,|\, \mathbf{y}_{1:n}) \end{aligned}$$

because, as can be seen in Figure 6.2, $\mathbf{c}_n$ and ν_n are independent of past and future observations once s_n is known. Since we chose conjugate priors, the posterior $p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)$ is a Normal-Gamma distribution of the form

$$\mathbf{c}_n \,|\, \nu_n, s_n, \mathbf{y}_n \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{s_n}, \nu_n^{-1}\hat{\boldsymbol{\Sigma}}_{s_n}) \quad \text{and} \quad \nu_n \,|\, s_n, \mathbf{y}_n \sim \mathcal{G}(\hat{\alpha}_{s_n}, \hat{\beta}_{s_n}).$$

This corresponds to the posterior distribution of a Bayesian AR process defined on the n-th segment. The posterior parameters $\hat{\boldsymbol{\mu}}_{s_n}$, $\hat{\boldsymbol{\Sigma}}_{s_n}$, $\hat{\alpha}_{s_n}$ and $\hat{\beta}_{s_n}$ are therefore given by the formulae derived in Section 6.2.1. The switch state posterior is given by the recursion

$$p(s_n \,|\, \mathbf{y}_{1:n}) \propto p(\mathbf{y}_n \,|\, s_n) \sum_{s_{n-1}} p(s_n \,|\, s_{n-1})\, p(s_{n-1} \,|\, \mathbf{y}_{1:n-1})$$

where $p(\mathbf{y}_n \,|\, s_n)$ is obtained by considering the decomposition

$$p(\mathbf{y}_n, \mathbf{c}_n, \nu_n \,|\, s_n) = p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)\, p(\mathbf{y}_n \,|\, s_n).$$

When we computed $p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)$ in Section 6.2.1, some factors which were not depending on $\mathbf{c}_n$ or ν_n have been left aside. In the Bayesian SAR-HMM, those factors actually depend on s_n and have to be included in $p(\mathbf{y}_n \,|\, s_n)$. We must also compensate for the introduction of the normalisation constant of the posterior distribution $p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)$ which also depends on s_n. Grouping the various contributions therefore yields[4]

$$p(\mathbf{y}_n \,|\, s) \propto \frac{|\hat{\boldsymbol{\Sigma}}_s|^{\frac{1}{2}}}{|\boldsymbol{\Sigma}_s|^{\frac{1}{2}}} \frac{\Gamma(\hat{\alpha}_s)}{\Gamma(\alpha_s)} \frac{\beta_s^{\alpha_s}}{\hat{\beta}_s^{\hat{\alpha}_s}} \prod_{t=t_n}^{t_{n+1}-1} \frac{1}{\sqrt{2\pi\sigma_t^2}}.$$

The first factor is the ratio of the normalisation constant of the prior and posterior distribution of $\mathbf{c}_n$, the second and third are the ratio of the normalisation constant of the prior and posterior distribution of ν_n[5], and the last factors occur in Equation 6.5, but are left aside since they do not depend on ν—σ_t^2 depends on s_n since it is initialised with the prior covariance $\boldsymbol{\Sigma}_{s_n}$.

[3]As usual, to simplify notation, we consider the case of a single training sequence.

[4]To simplify notation, we use s instead of s_n.

[5]Numerically, it is more stable to compute the ratio of the Gamma functions by using the Pochammer symbol $(x)_n$, defined as $(x)_n = \Gamma(x+n)/\Gamma(x)$.

Backward Pass

The backward pass corrects the filtered posterior $p(\mathbf{c}_n, \nu_n, s_n \,|\, \mathbf{y}_{1:n})$ by including the future information coming from $\mathbf{y}_{n+1:N}$. Since $\mathbf{c}_n$ and ν_n depend only on $\mathbf{y}_n$ once s_n is known, the filtered posterior $p(\mathbf{c}_n, \nu_n \,|\, s_n, \mathbf{y}_n)$ is therefore not modified by the backward pass. The smoothed posterior $p(s_n \,|\, \mathbf{y}_{1:N})$ can be obtained my means of Equations 3.11 and 3.12.

6.3.2 Parameter Updating

Equation 6.9 is similar to Equation 6.2 for the Bayesian AR process, the only difference being that the former has an additional average over the switch variable. The update formulae therefore have the same form as Equations 6.6 and 6.7. For the mean and covariance, we have

$$\boldsymbol{\mu}_s = \frac{\sum_n q(s_n = s)\, \langle \mathbf{c} \rangle_{q(\mathbf{c}_n \,|\, s_n)}}{\sum_{n=1}^{N} q(s_n = s)} = \frac{\sum_n q(s_n = s)\, \hat{\boldsymbol{\mu}}_{s_n}}{\sum_n q(s_n = s)} \tag{6.10}$$

$$\boldsymbol{\Sigma}_s = \frac{\sum_n q(s_n = s)\, \left\langle \nu_n (\mathbf{c}_n - \mu_s)(\mathbf{c}_n - \mu_s)^{\mathsf{T}} \right\rangle_{q(\mathbf{c}_n, \nu_n \,|\, s_n)}}{\sum_n q(s_n = s)} \tag{6.11}$$

where the sums are over $n \in [1, N]$. The formula for $\boldsymbol{\Sigma}_s$ is slightly more complicated than Equation 6.6 because, in general, $\boldsymbol{\mu}_s \neq \hat{\boldsymbol{\mu}}_{s_n}$. The average in Equation 6.11 therefore does not reduce to $\hat{\boldsymbol{\Sigma}}_{s_n}$, but is equal to

$$\begin{aligned}
&\left\langle \nu_n \big(\mathbf{c}_n \mathbf{c}_n^{\mathsf{T}} - \mathbf{c}_n \boldsymbol{\mu}_s^{\mathsf{T}} - \boldsymbol{\mu}_s \mathbf{c}_n^{\mathsf{T}} + \boldsymbol{\mu}_s \boldsymbol{\mu}_s^{\mathsf{T}} \big) \right\rangle_q \\
&\quad = \left\langle \nu_n (\mathbf{c}_n - \hat{\boldsymbol{\mu}}_{s_n})(\mathbf{c}_n - \hat{\boldsymbol{\mu}}_{s_n})^{\mathsf{T}} \right\rangle_q + \left\langle \nu_n \big(\hat{\boldsymbol{\mu}}_{s_n} \hat{\boldsymbol{\mu}}_{s_n}^{\mathsf{T}} - \mathbf{c}_n \boldsymbol{\mu}_s^{\mathsf{T}} - \boldsymbol{\mu}_s \mathbf{c}_n^{\mathsf{T}} + \boldsymbol{\mu}_s \boldsymbol{\mu}_s^{\mathsf{T}} \big) \right\rangle_q \\
&\quad = \hat{\boldsymbol{\Sigma}}_{s_n} + \langle \nu_n \rangle_q \, (\hat{\boldsymbol{\mu}}_{s_n} - \boldsymbol{\mu}_s)(\hat{\boldsymbol{\mu}}_{s_n} - \boldsymbol{\mu}_s)^{\mathsf{T}}
\end{aligned}$$

since $\langle \mathbf{c}_n \rangle_q = \hat{\boldsymbol{\mu}}_{s_n}$. The update formula for β_s is

$$\beta_s = \frac{\alpha_s \sum_n q(s_n = s)}{\sum_n q(s_n = s)\, \langle \nu_n \rangle_q} \tag{6.12}$$

and α_s is given by the implicit equation

$$\log \alpha_s - \psi(\alpha_s) = \log \frac{\sum_n q(s_n = s)\, \langle \nu_n \rangle_q}{\sum_n q(s_n = s)} - \frac{\sum_n q(s_n = s)\, \langle \log \nu_n \rangle_q}{\sum_n q(s_n = s)} \tag{6.13}$$

which can be solved in the same way as Equation 6.7, using Newton-Raphson's method for example—see Appendix B for a detailed explanation. The update equation for the prior is $p(s_1) = q(s_1)$ and the updated transition distribution is given by Equation 2.22.

6.4 Training & Evaluation

Following Section 3.4, we trained a separate Bayesian SAR-HMM for each of the eleven digits of the TI-DIGITS database. The Bayesian SAR-HMM of each digit was composed of ten states with a left-to-right transition matrix. To each state was associated a 10-th order Bayesian AR process and the model was constrained to stay an integer multiple of 140 time steps (0.0175 seconds) in the same state. The number of parameters to be trained where therefore 9 transition probabilities and, for each state: 10 means ($\boldsymbol{\mu}_s$), 10×10 covariance matrix elements ($\boldsymbol{\Sigma}_s$), and the parameters α_s and β_s of the Gamma distribution. This makes a total of 1129 free parameters. The parameters where updated by iteratively applying Equations 6.10, 6.11, 6.12 and 6.13 until the relative log-likelihood difference

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	95.5	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	86.4	59.1	9.5
HMM (LPCC)	109	**100**	95.5	27.3	18.2	9.5	9.1
HMM (LPCC) + USS	109	**100**	86.4	77.3	18.2	13.6	10
SAR-HMM	119	88.3	25.5	9.7	8.6	9.3	9.4
SAR-HMM + GA	109	97.2	79.8	56.7	22.2	9.7	9.1
AR-SLDS	120	86.8	88.2	87.3	79.1	80	**63.6**
AR-SLDS + GA	109	96.8	96.8	**96.4**	**94.8**	**84**	61.2
Bayesian SAR-HMM	1129	98.7	38.2	22.7	9.1	9.1	13.4

Table 6.1: Comparison of the word accuracy (in percent), at various SNRs, of the Bayesian SAR-HMM, the AR-SLDS with and without GA, the SAR-HMM with and without GA (copied from Table 3.1), and a state-of-the-art ASR system with and without USS (copied from Table 2.1). The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing.

between two consecutive iterations was less then 10^{-7} (Equation 3.14). For a given test utterance, recognition was performed by selecting the model with the highest likelihood.

For each state s, the model parameters were initialised as follows: (i) each speech utterance of the training set was split into S sequences of equal length, (ii) all the s-th sequences were gathered together and used to train an AR process for state s, (iii) the shape of the Gamma prior was arbitrarily set to $\alpha_s = 10$ and β_s was set such that the mean of the Gamma distribution matched the inverse innovation variance $\nu_s = 1/\sigma_s^2$ obtained by training the AR process, i.e., $\beta_s = \alpha_s \sigma_s^2$, (iv) the AR coefficients $\mathbf{c}_s$ obtained were used as the mean in the Gaussian prior, i.e., $\boldsymbol{\mu}_s = \mathbf{c}_s$, (v) the covariance of the AR coefficients was set to the identity matrix, i.e, $\nu_s^{-1}\boldsymbol{\Sigma}_s = \mathbf{I}$, (vi) a new state segmentation was obtained by doing a Viterbi decoding [69, 58] with the so-defined Bayesian SAR-HMM, and steps (ii) to (vi) were then repeated three times.

6.5 Performance

Table 6.1 shows the recognition accuracy of the Bayesian SAR-HMM on the isolated digit recognition task described in Section 2.11. For comparison, the performance of the AR-SLDS, the SAR-HMM and the feature-based HMMs reported in Table 4.1 is also reproduced. In clean conditions, the slightly better accuracy of the Bayesian SAR-HMM over its gain-adapted counterpart demonstrates the soundness of the Bayesian alternative to ad-hoc maximum likelihood gain adaptation. Although both types of SAR-HMM cannot handle noise explicitly, it is nevertheless interesting to compare their performance in the presence of noise. In noisy conditions, the Bayesian SAR-HMM is less accurate probably because it explicitly constrains the parameters to change within the limits of what has been observed in the training data. On the other hand, with GA, no such constraint exists and the innovation variance can become large to accommodate variations in the signal not observed in the training data. This can be better seen by looking at the KL divergence

$$\mathrm{KL}\big(q(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N}) \,||\, p(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N} \,|\, y_{1:T})\big) \geq 0$$

which yields

$$\begin{aligned}&\log p(y_{1:T}) \geq \\ &\quad - \big\langle \log q(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N})\big\rangle_q + \big\langle \log p(y_{1:T}, \mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N})\big\rangle_q \end{aligned} \tag{6.14}$$

where the averages are taken with respect to $q(\mathbf{c}_{1:N}, \nu_{1:N}, s_{1:N})$. Since the posterior distribution q can be computed exactly in the Bayesian SAR-HMM, the bound in Equation 6.14 is exact. The first term in Equation 6.14 is the entropy of the posterior distribution and the second is called the energy. The entropy measures the uncertainty on the setting of the parameters. It has a regularising effect which prevents the parameters from taking values which are too uncertain, i.e., too different from those observed in the training data. In contrast, the gain-adapted SAR-HMM does not include an entropy term and only maximises the energy. A possible explanation for the better accuracy of the Bayesian SAR-HMM at SNR 0.7 dB is that, for high noise levels, the innovation variance in the gain-adapted SAR-HMM is significantly adapted, whilst, in the Bayesian case, such a large modification is prevented by the entropy term. The Bayesian SAR-HMM is therefore more conservative and, as a result, performs better in conditions where the structure of the speech signal has almost completely disappeared.

6.6 Summary

A Bayesian treatment of the parameters of the AR process is a statistically principled alternative to the more traditional GA. The parameters are considered as random hidden variables whose states have to be inferred from the observed signal. Provided appropriate priors are chosen, inference can be performed exactly. The proposed Bayesian approach results in simple update formulae which correctly deal with the uncertainty in the parameter estimates. Contrary to GA, the variability of the parameter setting is explicitly constrained by prior distributions fitted on training data and no further adjustment is required. Whilst the performance of the Bayesian SAR-HMM and the SAR-HMM are comparable on clean data, the additional constraint imposed by the priors makes the Bayesian SAR-HMM less accurate in noisy conditions. This is expected and desired, since the SAR-HMM is essentially a model of clean speech. In the next chapter, we propose a Bayesian treatment of the AR-SLDS. Like the gain-adapted AR-SLDS, the Bayesian AR-SLDS jointly models speech and noise, thereby providing a level of noise robustness.

Chapter 7

Bayesian Switching Linear Dynamical Systems

7.1 Introduction

In the previous chapter, we saw that a Bayesian approach is a valuable alternative to GA. However, the application of the proposed Bayesian SAR-HMM was limited to clean speech. To deal with noise explicitly without having to train a new model, we devise the Bayesian AR-SLDS, an extension of the Bayesian SAR-HMM which includes an explicit model of additive Gaussian white noise. In this model, the *a priori* unknown noise variance, which was adjusted with GA in the AR-SLDS, is also considered as a random variable.

Inferring the posterior distribution of the hidden variables in the Bayesian AR-SLDS is considerably more complex than in the AR-SLDS because, due to the nonlinearities introduced by the priors, the structure of the posterior distribution is in general not a mixture of Gaussians. To address this issue we propose a simple variational approximation which yields a two-steps inference procedure where the posterior of the parameters and the switch states is inferred from a HMM, and the posterior of the clean waveform is inferred from a LDS.

We also consider an alternative form of Bayesian AR-SLDS where the noisy observed signal is considered as a *scaled* and *corrupted* version of a clean hidden signal which is modelled by a standard SAR-HMM. Compared to the gain-adapted and Bayesian AR-SLDS, the proposed scale-invariant AR-SLDS has the advantage of being able to model variations in the signal amplitude explicitly, without adjusting the parameters of the underlying AR process. The variance and signal scaling factor are considered as random variables and are therefore automatically adapted. Contrary to the Bayesian AR-SLDS which can be initialised from a trained Bayesian SAR-HMM, the scale-invariant AR-SLDS has to be trained directly. To achieve this, we propose a more elaborate variational approximation where the posterior of the clean waveform and switch states is inferred from a SLDS by means of EC.

So far we have been considering simple additive Gaussian white noise. In this chapter, we also compare the recognition accuracy of the gain-adapted and Bayesian AR-SLDS to that of a state-of-the-art HMM system on a more natural source of noise provided by the Aurora database [35].

7.2 The Bayesian AR-SLDS

A Bayesian treatment of the AR-SLDS can be achieved by setting priors on the parameters of the AR-SLDS. As with the Bayesian SAR-HMM, for the AR coefficients and the inverse innovation variance, we choose a Normal-Gamma prior conditioned on the switch state s_n:

$$\mathbf{c}_n \,|\, \nu_n, s_n \sim \mathcal{N}(\boldsymbol{\mu}_{s_n}, \nu_n^{-1}\boldsymbol{\Sigma}_{s_n}) \quad \text{and} \quad \nu_n \,|\, s_n \sim \mathcal{G}(\alpha_{s_n}, \beta_{s_n}).$$

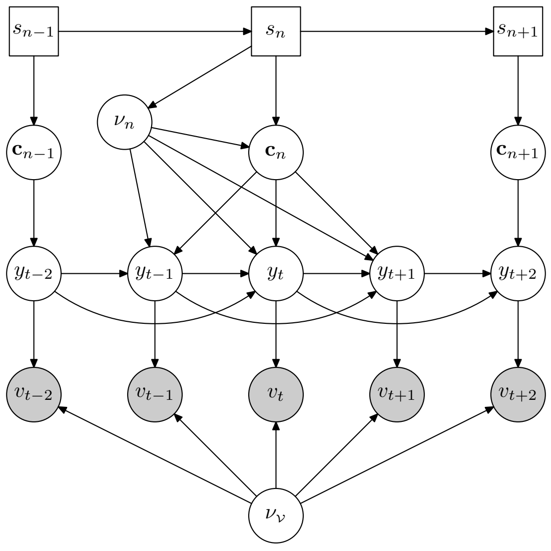


Figure 7.1: DBN representation of a second order Bayesian AR-SLDS; s_n represents the hidden switch state, $\mathbf{c}_n$ the AR coefficients, ν_n the inverse innovation variance, y_t the hidden clean waveform sample, v_t the observed noisy waveform sample and $\nu_\mathcal{V}$ the inverse output noise variance.

To deal with variable levels of additive Gaussian white noise without having to train a new model, we also introduce a Gamma prior on the inverse noise variance:

$$\nu_\mathcal{V} \sim \mathcal{G}(\alpha_\mathcal{V}, \beta_\mathcal{V}).$$

Compared to the Bayesian SAR-HMM, the Bayesian AR-SLDS therefore contains two additional free parameters, $\alpha_\mathcal{V}$ and $\beta_\mathcal{V}$. As for the AR-SLDS, an observed noisy sample v_t is seen as a clean hidden sample y_t corrupted by additive Gaussian white noise. Hence

$$p(v_t \,|\, y_t, \nu_\mathcal{V}) = \frac{\nu_\mathcal{V}^{1/2}}{\sqrt{2\pi}} \exp\left\{-\frac{\nu_\mathcal{V}}{2}(v_t - y_t)^2\right\}.$$

The joint distribution defined by the Bayesian AR-SLDS is

$$\begin{aligned} p(v_{1:T}, y_{1:T}, \mathbf{c}_{1:N}, \nu_{1:N}, \nu_\mathcal{V}, s_{1:N}) = p(\nu_\mathcal{V}) \prod_{n=1}^{N} & p(\mathbf{c}_n \,|\, \nu_n, s_n)\, p(\nu_n \,|\, s_n)\, p(s_n \,|\, s_{n-1}) \\ & \times \prod_{t=t_n}^{t_{n+1}-1} p(v_t \,|\, y_t, \nu_\mathcal{V})\, p(y_t \,|\, \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t) \end{aligned} \tag{7.1}$$

where $p(y_t \,|\, \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t)$ is given by Equation 6.1. Graphically, this can be depicted by the DBN of Figure 7.1. The likelihood $p(v_{1:T})$ of an observed noisy sequence $v_{1:T}$ is given by integrating Equation 7.1 over the hidden variables. In temporal models such as the Bayesian SLDS the likelihood can generally be efficiently computed by means of the recursion

$$p(v_{1:T}) = \prod_{t=1}^{T} p(v_t \,|\, v_{1:t-1})$$

where $p(v_t \,|\, v_{1:t-1})$ is obtained by integrating the *unnormalised* filtered posterior $p(v_t, y_t, \tilde{\mathbf{y}}_t, \mathbf{c}_n, \nu_n, \nu_\mathcal{V}, s_n \,|\, v_{1:t-1})$ over the hidden variables y_t, $\tilde{\mathbf{y}}_t$, $\mathbf{c}_n$, ν_n, $\nu_\mathcal{V}$ and s_n. The filtered posterior of the Bayesian AR-SLDS is difficult to compute because of the additional complexity induced by $p(\mathbf{c}_n \,|\, \nu_n, s_n)$. Indeed, according to Equation 3.1, the predicted sample y_t depends on the product of two normally distributed random variables which, in general, is not normally distributed. An alternative is to consider a variational Bayesian approach [5] where the lower-bound, provided by the KL divergence between an approximate posterior distribution q and the true posterior distribution, is used instead of the true likelihood. If we define $\vartheta_n = \{\mathbf{c}_n, \nu_n, \nu_v\}$, then

$$\mathrm{KL}\big(q(y_{1:T}, \vartheta_{1:N}, s_{1:N}) \,||\, p(y_{1:T}, \vartheta_{1:N}, s_{1:N} \,|\, v_{1:T})\big) \geq 0$$

implies

$$\log p(v_{1:T}) \geq -\big\langle \log q(y_{1:T}, \vartheta_{1:N}, s_{1:N})\big\rangle_q + \big\langle \log p(v_{1:T}, y_{1:T}, \vartheta_{1:N}, s_{1:N})\big\rangle_q \tag{7.2}$$

with equality if and only if q is the true posterior. The central idea behind the variational approach is to approximate the true, intractable posterior distribution by a simpler, tractable distribution q such that the KL divergence between the two is as small as possible. If the KL divergence is small, then the lower-bound will be close to the true likelihood.

7.2.1 Variational Inference

We propose to approximate the true posterior distribution by a simpler q distribution for which the problematic nonlinear interaction between $\mathbf{c}_n$ and $\tilde{\mathbf{y}}_t$ is removed. Two alternatives can be considered; the first is

$$q(y_{1:T}, \vartheta_{1:N}, s_{1:N}) = q(y_{1:T} \,|\, s_{1:N})\, q(\vartheta_{1:N} \,|\, s_{1:N})\, q(s_{1:N}) \tag{7.3}$$

and the second is

$$q(y_{1:T}, \vartheta_{1:N}, s_{1:N}) = q(y_{1:T})\, q(\vartheta_{1:N} \,|\, s_{1:N})\, q(s_{1:N}). \tag{7.4}$$

The first approximation is less crude than the second, but the dependency of $y_{1:T}$ on $s_{1:N}$ is problematic because the marginal posterior

$$q(y_t, s_n) = \sum_{\substack{s_{1:n-1}\\ s_{n+1:N}}} q(y_t \,|\, s_{1:N})\, q(s_{1:N})$$

required to evaluate the lower-bound given by the rhs of Equation 7.2, is a mixture with $\mathcal{O}(S^{N-1})$ components and, in practice, this is generally computationally intractable. The second approximation is easier to deal with because the posterior distribution of $y_{1:T}$ is completely decoupled from that of $s_{1:T}$ and, since we chose conjugate priors,

$$q(\vartheta_{1:N} \,|\, s_{1:N}) = \prod_{n=1}^{N} q(\vartheta_n \,|\, s_n).$$

The individual marginals $q(y_t)$, $q(\vartheta_n \,|\, s_n)$ and $q(s_n)$ can therefore be computed exactly. Combining Equations 7.2 and 7.4 yields the lower-bound

$$\begin{aligned}\log p(v_{1:T}) \geq & - \big\langle \log q(y_{1:T})\big\rangle_q - \big\langle \log q(\vartheta_{1:N}, s_{1:N})\big\rangle_q \\ & + \big\langle \log p(v_{1:T}, y_{1:T} \,|\, \vartheta_{1:N}, s_{1:N})\big\rangle_q + \big\langle \log p(\vartheta_{1:N}, s_{1:N})\big\rangle_q\end{aligned} \tag{7.5}$$

The optimal q distribution which satisfies Equation 7.4 is the one which maximises this lower-bound. After differentiating the rhs of Equation 7.5 with respect to $q(y_{1:T})$ and $q(\vartheta_{1:N}, s_{1:N})$ and setting the result equal to zero, we obtain

$$q(y_{1:T}) \propto \exp\Big\{\big\langle \log p(v_{1:T}, y_{1:T} \,|\, \vartheta_{1:N})\big\rangle_{q(\vartheta_{1:N}, s_{1:N})}\Big\} \tag{7.6}$$

$$q(\vartheta_{1:N}, s_{1:N}) \propto \exp\Big\{\big\langle \log p(v_{1:T}, y_{1:T} \,|\, \vartheta_{1:N})\big\rangle_{q(y_{1:T})}\Big\}\, p(\vartheta_{1:N}, s_{1:N}). \tag{7.7}$$

Starting with a first approximation of either $q(\vartheta_{1:N}, s_{1:N})$ or $q(y_{1:T})$, the posterior distribution $q(y_{1:T}, \vartheta_{1:N}, s_{1:N})$ can thus be computed by iteratively applying Equations 7.6 and 7.7. Since this scheme guarantees that the lower-bound on the log-likelihood does not decrease [5], convergence is therefore achieved when the lower-bound does not change for two consecutive iterations.

Finding $q(\vartheta_{1:N} \,|\, s_{1:N})$

Isolating the factors related to ϑ_n in Equation 7.7, yields

$$q(\vartheta_n \,|\, s_n) \propto \exp\left\{ \sum_{t=t_n}^{t_{n+1}-1} \big\langle \log p(v_t, y_t \,|\, \tilde{\mathbf{y}}_t, \vartheta_n) \big\rangle_{q(y_t, \tilde{\mathbf{y}}_t)} \right\} p(\vartheta_n \,|\, s_n). \tag{7.8}$$

Due to conjugacy, $q(\vartheta_n \,|\, s_n)$ has the same form as the prior $p(\vartheta_n \,|\, s_n)$ and is therefore a product of Normal and Gamma distributions, i.e.,

$$\mathbf{c}_n \,|\, \nu_n, s_n \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{s_n}, \nu_n^{-1}\hat{\boldsymbol{\Sigma}}_{s_n}), \quad \nu_n \sim \mathcal{G}(\hat{\alpha}_{s_n}, \hat{\beta}_{s_n}) \quad \text{and} \quad \nu_{\mathcal{V}} \sim \mathcal{G}(\hat{\alpha}_{\mathcal{V}}, \hat{\beta}_{\mathcal{V}}).$$

To find the posterior value of the hyper-parameters, we expand the rhs of Equation 7.8 and group the factors according to the parameters they contain.

By grouping all the factors in the rhs of Equation 7.8 which involves $\mathbf{c}_n$, we find that, up to an irrelevant constant, the logarithm of $q(\mathbf{c}_n \,|\, \nu_n, s_n)$ is equal to

$$-\frac{\nu_n}{2} \sum_{t=t_n}^{t_{n+1}-1} \Big[\langle y_t^2 \rangle - 2\langle y_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle \mathbf{c}_n + \mathbf{c}_n^{\mathsf{T}} \langle \tilde{\mathbf{y}}_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle \mathbf{c}_n \Big]$$
$$- \frac{\nu_n}{2} \sum_{t=t_n}^{t_{n+1}-1} \Big[\mathbf{c}_n^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \mathbf{c}_n - 2\boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \mathbf{c}_n + \boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} \Big]$$

where the first term comes from the logarithm of $p(y_t \,|\, \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t)$ averaged over $q(y_{1:T})$ and the second from the logarithm of the prior $p(\mathbf{c}_n \,|\, \nu_n, s_n)$. After grouping the quadratic and linear terms together, this can be written as

$$-\frac{\nu_n}{2} (\mathbf{c}_n - \hat{\boldsymbol{\mu}}_{s_n})^{\mathsf{T}} \hat{\boldsymbol{\Sigma}}_{s_n}^{-1} (\mathbf{c}_n - \hat{\boldsymbol{\mu}}_{s_n}) - \frac{\nu_n}{2} \sum_{t=t_n}^{t_{n+1}-1} \Big[\langle y_t^2 \rangle + \boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} - \hat{\boldsymbol{\mu}}_{s_n} \hat{\boldsymbol{\Sigma}}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} \Big].$$

The first term defines a Normal distribution $\mathcal{N}(\hat{\boldsymbol{\mu}}_{s_n}, \nu_n^{-1}\hat{\boldsymbol{\Sigma}}_{s_n})$ with

$$\hat{\boldsymbol{\Sigma}}_{s_n} = \Big[\langle \tilde{\mathbf{y}}_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle + \boldsymbol{\Sigma}_{s_n}^{-1} \Big]^{-1} \quad \text{and} \quad \hat{\boldsymbol{\mu}}_{s_n} = \Big[\langle y_t \tilde{\mathbf{y}}_t^{\mathsf{T}} \rangle + \boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \Big] \hat{\boldsymbol{\Sigma}}_{s_n}.$$

The second depends only on ν_n and will be absorbed into $q(\nu_n \,|\, s_n)$.

After grouping together the factors which depend on ν_n and which have been left aside during the computation of $q(\mathbf{c}_n \,|\, \nu_n, s_n)$, we find that, up to an irrelevant constant, the logarithm of $q(\nu_n \,|\, s_n)$ is equal to

$$\log \nu_n^{\frac{1}{2}} + \log \nu_n^{\alpha_{s_n}-1} - \beta_{s_n}\nu_n - \frac{\nu_n}{2} \sum_{t=t_n}^{t_{n+1}-1} \Big[\langle y_t^2 \rangle + \boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} - \hat{\boldsymbol{\mu}}_{s_n}^{\mathsf{T}} \hat{\boldsymbol{\Sigma}}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} \Big]$$

where the first term comes from the normalisation constant of $p(y_t \,|\, \mathbf{c}_n, \nu_n, \tilde{\mathbf{y}}_t)$, the second and third come from the prior $p(\nu_n \,|\, s_n)$ and the last is the term which has been left aside during the computation of $q(\mathbf{c}_n \,|\, \nu_n, s_n)$. This defines a Gamma distribution $\mathcal{G}(\hat{\alpha}_{s_n}, \hat{\beta}_{s_n})$ with

$$\hat{\alpha}_{s_n} = \alpha_{s_n} + \frac{T_n}{2} \quad \text{and} \quad \hat{\beta}_{s_n} = \beta_{s_n} + \frac{1}{2} \sum_{t=t_n}^{t_{n+1}-1} \Big[\langle y_t^2 \rangle + \boldsymbol{\mu}_{s_n}^{\mathsf{T}} \boldsymbol{\Sigma}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} - \hat{\boldsymbol{\mu}}_{s_n} \hat{\boldsymbol{\Sigma}}_{s_n}^{-1} \boldsymbol{\mu}_{s_n} \Big].$$

where $T_n = t_{n+1} - t_n$ is the length of the n-th segment.

For $\nu_\mathcal{V}$, we have that, up to an irrelevant constant, the logarithm of $p(\nu_\mathcal{V} \mid s_n)$ is equal to

$$\frac{T}{2} \log \nu_\mathcal{V} - \frac{\nu_\mathcal{V}}{2} \sum_{t=1}^{T} \Big[v_t^2 - 2 v_t \langle y_t \rangle + \langle y_t^2 \rangle \Big] + (\alpha_\mathcal{V} - 1) \log \nu_\mathcal{V} - \beta_\mathcal{V} \nu_\mathcal{V}$$

where the first two terms correspond to the logarithm of the emission distribution $p(v_t \mid y_t, \nu_v)$ averaged over $q(y_{1:T})$ and the others come from the prior $p(\nu_\mathcal{V})$. This defines a Gamma distribution $\mathcal{G}(\hat{\alpha}_\mathcal{V}, \hat{\beta}_\mathcal{V})$ with

$$\hat{\alpha}_\mathcal{V} = \alpha_\mathcal{V} + \frac{T}{2} \quad \text{and} \quad \hat{\beta}_\mathcal{V} = \beta_\mathcal{V} + \frac{1}{2} \sum_{t=1}^{T} \Big[v_t^2 - 2 v_t \langle y_t \rangle + \langle y_t^2 \rangle \Big].$$

Finding $q(s_{1:N})$

Equation 7.7 can be alternatively written as a HMM of the form

$$p(\mathbf{o}_{1:N}, \vartheta_{1:N}, s_{1:N}) = \prod_{n=1}^{N} p(\mathbf{o}_n \mid \vartheta_n)\, p(\vartheta_n \mid s_n)\, p(s_n \mid s_{n-1})$$

where $\mathbf{o}_n$ is an auxiliary observation of dimension $T_n = t_{n+1} - t_n$, such that

$$p(\mathbf{o}_n \mid \vartheta_n) \propto \exp \left\{ \sum_{t=t_n}^{t_{n+1}-1} \big\langle \log p(v_t, y_t \mid \vartheta_n, \tilde{\mathbf{y}}_t) \big\rangle_{q(\vartheta_n)} \right\}.$$

With this notation, we have $q(\vartheta_n \mid s_n) \propto p(\mathbf{o}_t \mid \vartheta_n)\, p(\vartheta_n \mid s_n)$ and

$$q(s_{1:N}) \equiv p(s_{1:N} \mid \mathbf{o}_{1:N}) \propto \prod_{n=1}^{N} p(\mathbf{o}_n \mid s_n)\, p(s_n \mid s_{n-1}).$$

This is exactly Equation 2.7. The posterior $q(s_{1:N})$ can therefore be computed using either the traditional Forward-Backward algorithm or the RTS method. The emission distribution is given by integrating out ϑ_n, i.e.,

$$p(\mathbf{o}_n \mid s_n) = \int_{\vartheta_n} p(\mathbf{o}_n \mid \vartheta_n)\, p(\vartheta_n \mid s_n) = \prod_{t=t_n}^{t_{n+1}-1} \frac{|\hat{\boldsymbol{\Sigma}}_{s_n}|^{\frac{1}{2}}}{|\boldsymbol{\Sigma}_{s_n}|^{\frac{1}{2}}} \frac{\beta_{s_n}^{\alpha_{s_n}}}{\hat{\beta}_{s_n}^{\hat{\alpha}_{s_n}}} \frac{\Gamma(\hat{\alpha}_{s_n})}{\Gamma(\alpha_{s_n})} \frac{\beta_\mathcal{V}^{\alpha_\mathcal{V}}}{\hat{\beta}_\mathcal{V}^{\hat{\alpha}_\mathcal{V}}} \frac{\Gamma(\hat{\alpha}_\mathcal{V})}{\Gamma(\alpha_\mathcal{V})}.$$

This corresponds to the ratio of the normalisation constants of the prior and posterior distributions. The normalisation constants of the priors have been left aside during the computation of $q(\mathbf{c}_n, \nu_n \mid s_n)$ and those of the posteriors are introduced by the integral over ϑ_n. Since, for a given pair of segment and state, the prior and posterior parameters are constant, the emission distribution actually reduces to the ratio of the normalisation constants raised to the power T_n.

Finding $q(y_{1:T})$

For the t-th step, the logarithm of the rhs of Equation 7.6 is, up to an irrelevant constant, equal to

$$-\frac{1}{2} \big\langle \nu_\mathcal{V} (v_t - y_t)^2 \big\rangle_{q(\vartheta_n, s_n)} - \frac{1}{2} \big\langle \nu_n (y_t - \mathbf{c}_n^\mathsf{T} \tilde{\mathbf{y}}_t)^2 \big\rangle_{q(\vartheta_n, s_n)}. \tag{7.9}$$

Since the state of y_t depends only on the state of the previous variables, $q(y_{1:T})$ is clearly a chain. Belief Propagation [55] may then be applied to find the required statistics—like the mean and variance of y_t, for example. However, Belief Propagation is potentially numerically unstable and we therefore prefer to convert the distribution into the form of a LDS, for which inference is straightforward. Following the same approach as used in [8], we rewrite (7.9) as a mean plus fluctuation term

$$-\frac{1}{2}\langle\nu_{\mathcal{V}}\rangle(v_t - y_t)^2 - \frac{1}{2}\langle\nu_n\rangle\big(y_t - \langle\mathbf{c}_n\rangle\tilde{\mathbf{y}}_t\big)^2 + \tilde{\mathbf{y}}_t^{\mathsf{T}}\underbrace{\langle\nu_n\Delta\mathbf{c}_n\Delta\mathbf{c}_n^{\mathsf{T}}\rangle}_{\mathbf{S}_n}\tilde{\mathbf{y}}_t \tag{7.10}$$

where the averages are taken with respect to $q(\vartheta_n, s_n)$ and $\Delta\mathbf{c}_n = \mathbf{c}_n - \langle\mathbf{c}_n\rangle$. Explicitly, we therefore have

$$\langle\nu_{\mathcal{V}}\rangle = \frac{\hat{\alpha}_{\mathcal{V}}}{\hat{\beta}_{\mathcal{V}}}, \quad \langle\nu_n\rangle = \sum_{s_n} q(s_n)\,\frac{\hat{\alpha}_{s_n}}{\hat{\beta}_{s_n}} \quad \text{and} \quad \langle\mathbf{c}_n\rangle = \sum_{s_n} q(s_n)\,\hat{\boldsymbol{\mu}}_{s_n}.$$

The computation of $\mathbf{S}_n$ is a slightly more complicated and is presented in Appendix C. Expression 7.10 is equivalent to the LDS:

$$\begin{bmatrix} y_t \\ \tilde{\mathbf{y}}_t \end{bmatrix} = \begin{bmatrix} \langle\mathbf{c}_n\rangle & 0 \\ \mathbf{I}_R & \mathbf{0} \end{bmatrix}\begin{bmatrix} y_{t-1} \\ \tilde{\mathbf{y}}_{t-1} \end{bmatrix} + \boldsymbol{\eta}_t^{\mathcal{H}} \quad \text{and} \quad \boldsymbol{\eta}_t^{\mathcal{H}} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \langle\nu_n\rangle^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}\right) \tag{7.11}$$

$$\begin{bmatrix} v_t \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_t \end{bmatrix}\begin{bmatrix} y_t \\ \tilde{\mathbf{y}}_t \end{bmatrix} + \boldsymbol{\eta}_t^{\mathcal{V}} \quad \text{and} \quad \boldsymbol{\eta}_t^{\mathcal{V}} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \langle\nu_{\mathcal{V}}\rangle^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_R \end{bmatrix}\right) \tag{7.12}$$

where $\mathbf{I}_R$ is the $R \times R$ identity matrix and $\mathbf{U}_n$ is the upper triangular matrix obtained from the Cholesky decomposition of $\mathbf{S}_n$. Equations 7.11 and 7.12 are similar to Equations 5.2 and 5.1, the only difference being that the parameters are different for each segment n. The marginal posterior $q(y_t, \tilde{\mathbf{y}}_t)$ can therefore be computed by means of the RTS algorithm, as described in Section 5.2.2. The form of the RTS algorithm presented in Section 5.2.2 is particularly well suited for cases where the dimension of the observed variable is smaller than that of the hidden variable. In the case of Equations 7.11 and 7.12, the alternate version of the RTS method proposed in [8] is more appropriate.

Initialisation

A possible way to initialise the variational procedure is to start with an approximation of $q(y_{1:T})$ and then to compute $q(\vartheta_{1:N}, s_{1:N})$ with Equation 7.7. A good approximation of $q(y_{1:T})$ can be obtained by performing inference with a SLDS whose parameters are set to the average parameter setting of the Bayesian SLDS. If the EC algorithm is used for example, this would yield the mixture of Gaussians:

$$p(y_t, \tilde{\mathbf{y}}_t \,|\, v_{1:T}) = \sum_{s_n} p(y_t, \tilde{\mathbf{y}}_t \,|\, s_n, v_{1:T})\, p(s_n \,|\, v_{1:T}).$$

An initial estimate of $q(y_t, \tilde{\mathbf{y}}_t)$ can then be obtained by collapsing this mixture to a single Gaussian.

Convergence

The central idea in the variational approximation is to push up the lower-bound, given by the rhs of Equation 7.5, so that it gets as close as possible to the true log-likelihood $p(v_{1:T})$. Computing the value of the bound is important because it is an indicator of the convergence of the variational approximation. Since applying Equations 7.6 and 7.7 cannot decrease the lower-bound [5, 11], convergence will eventually be achieved when no significant change occurred between two consecutive iterations.

7.2.2 Performance

We tested the Bayesian AR-SLDS on the isolated digit recognition task described in Section 2.10. As for the gain-adapted AR-SLDS, the models were not trained but simply initialised with the corresponding Bayesian SAR-HMM. To allow the models to automatically adjust to the various noise levels, we manually set a prior on $\nu_{\mathcal{V}}$ such that the variance of $\nu_{\mathcal{V}}$ was large enough (10^{10} in our case). Inference was then carried out using an AR-SLDS initialised with the average parameter setting. Since

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	95.5	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	86.4	59.1	9.5
HMM (LPCC)	109	**100**	95.5	27.3	18.2	9.5	9.1
HMM (LPCC) + USS	109	**100**	86.4	77.3	18.2	13.6	10
SAR-HMM	119	88.3	25.5	9.7	8.6	9.3	9.4
SAR-HMM + GA	109	97.2	79.8	56.7	22.2	9.7	9.1
AR-SLDS	120	86.8	88.2	87.3	79.1	80	**63.6**
AR-SLDS + GA	109	96.8	96.8	**96.4**	**94.8**	84	61.2
Bayesian SAR-HMM	1129	98.7	38.2	22.7	9.1	9.1	13.4
Bayesian AR-SLDS	1131	98.5	—	95.5	94.2	**89.5**	52.3

Table 7.1: Comparison of the word accuracy (in percent), at various SNRs, of the Bayesian AR-SLDS and all the other models considered so far. The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing.

the noise level in unknown *a priori*, we tested three different noise variances (10^{-10}, 10^{-5} and 10^{-3}) and the posterior resulting from the most likely AR-SLDS was then used to initialise the variational procedure. For a given test utterance, recognition was performed by picking the model for which the lower-bound on the log-likelihood, as given by Equation 7.5, was the highest. The lower-bound is used as a replacement for the true log-likelihood which, for the Bayesian AR-SLDS, cannot be obtained.

Table 7.1 shows the recognition accuracy of the Bayesian AR-SLDS compared to all the other models. On clean data, the performance of the Bayesian AR-SLDS is close to that of the Bayesian SAR-HMM. Although theoretically the Bayesian AR-SLDS reduces to the Bayesian SAR-HMM in clean conditions, in practice, the accuracy may differ because the lower-bound is used instead of the true log-likelihood. In general, the accuracy of the gain-adapted and Bayesian AR-SLDS are comparable. On clean data, the Bayesian SLDS is slightly more accurate probably because it has the advantage of being able to slightly adapt the AR coefficients while the gain-adapted SLDS cannot. At SNR 10.6 dB, we would expect the accuracy of the Bayesian AR-SLDS to be close to that of the gain-adapted AR-SLDS. The fact that, to the contrary, it is more than 5% higher would suggest that the gain-adapted AR-SLDS has overfitted. At SNR 0.7 dB, the accuracy of the Bayesian AR-SLDS is significantly lower than that of the gain-adapted AR-SLDS. A possible explanation is that, at a low SNR, the proposed variational approximation fails to properly localise the position of the clean speech waveform in the noisy signal. This is likely to happen since the posterior $q(y_{1:T})$ essentially relies on the segmentation given by $q(s_{1:N})$ which is computed *independently*, on the basis of the difference between the value of the prior and posterior parameters. This effect is particularly visible in the case of the digit 'zero' which, at SNR 0.7 dB, is correctly recognised only 1.79% of the time by the Bayesian AR-SLDS. Figure 7.2 compares the most likely segmentations and reconstructed signals provided by the gain-adapted and Bayesian AR-SLDSs. Clearly, the latter is not able to properly localise the clean waveform. The gain-adapted AR-SLDS is more accurate and is able to correctly recognise a 'zero' 64.3% of the time. In contrast, in the same conditions, the digit 'eight' is correctly recognised 70.5% of the time by the Bayesian AR-SLDS and only 0.9% of the time by the gain-adapted AR-SLDS. Each model therefore has its pros and cons. On the one hand, the gain-adapted AR-SLDS is more conservative since it does not allow the AR coefficient to change; however GA is potentially dangerous because it may lead to overfitting. On the other hand, the Bayesian AR-SLDS is more versatile and provides a principled alternative to GA, but has the drawback of being too complex to be dealt with exactly.

An alternative is to use the less crude variational approximation given by Equation 7.3. The main difference with Equation 7.4 is that the posterior $q(y_{1:T}, s_{1:N})$ is computed by running EC on a SLDS

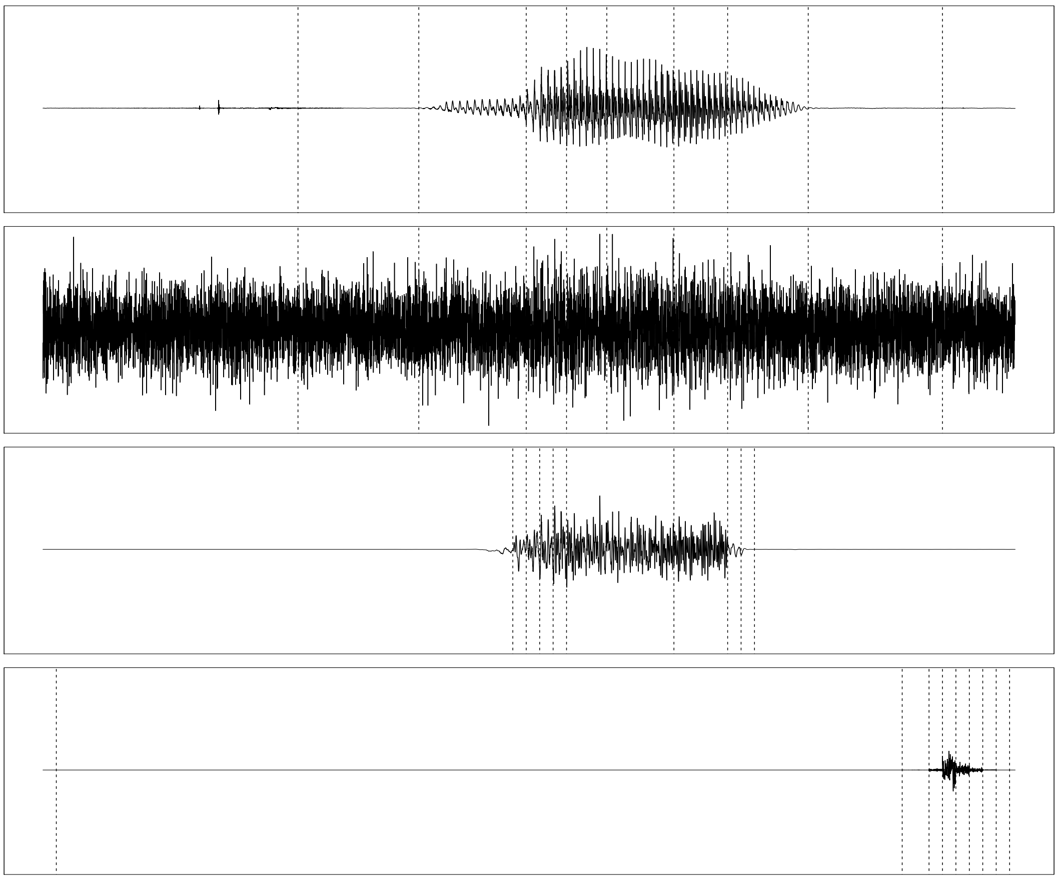


Figure 7.2: Comparison of the reconstructed signals provided by the gain-adapted and Bayesian AR-SLDSs. From top to bottom, the original clean waveform of a 'zero' taken from the TI-DIGITS database, the noisy signal obtained by adding Gaussian white noise at SNR 0.7 dB, the reconstructed clean signals provided by the gain-adapted AR-SLDS and the Bayesian AR-SLDS, respectively. The dashed lines show the most-likely state segmentation. To facilitate comparison the clean segmentation is also reproduced on the noisy signal.

defined in a way similar to Equations 7.11 and 7.12. Unfortunately, in practice, when EC is used with collapse to a single Gaussian, this approach performs much worse, reaching a recognition accuracy of only 79.2% on clean data. A likely explanation is that the more complex variational approximation is trapped into a local maximum which the simpler method can avoid. This is confirmed by the fact that initialising the more complex variational approximation (Equation 7.3) with the optimal simpler distribution (Equation 7.4) leads to a comparable performance. This is expected since setting $q(y_t \,|\, s_n)$ in the more complex approximation to $q(y_t)$, obtained from the simpler one, is equivalent to assuming independence between y_t and s_n. A possible way to avoid local maxima is to collapse the posterior to a mixture of Gaussians. However, in practice, this approach is considerably slower than the single Gaussian case and the large amount of memory required to store the posteriors generally prevents its use on long time series.

7.3 Modelling Scale Invariance Explicitly

So far we have been addressing the problem of dealing with variations in the signal amplitude by adapting the innovation variance. A more direct alternative is to consider the observed noisy sample v_t as a scaled version of a *scale-invariant clean hidden* sample y_t corrupted by Gaussian white noise:

$$v_t = b y_t + \eta_t^{\mathcal{V}} \quad \text{with} \quad \eta_t^{\mathcal{V}} \sim \mathcal{N}(0, \sigma_{\mathcal{V}}^2) \tag{7.13}$$

and to model the clean hidden samples y_t with a switching AR process:

$$y_t = \mathbf{c}_s^{\mathsf{T}} \tilde{\mathbf{y}}_t + \eta_t^{\mathcal{H}} \quad \text{with} \quad \eta_t^{\mathcal{H}} \sim \mathcal{N}(0, \sigma_{\mathcal{H},s}^2). \tag{7.14}$$

where s denotes the state of the switch variable. In this manner, no innovation-inflation is required, provided that the observed signal is simply a scaled, noisy version of an underlying AR process. For a given observed sequence, the setting of b and $\sigma_{\mathcal{V}}^2$ is unknown *a priori* and needs to be determined. To solve this problem we treat both parameters as random variables and introduce a Normal-Gamma prior on b and the inverse noise variance $\nu = 1/\sigma_{\mathcal{V}}^2$:

$$b \,|\, \nu, s \sim \mathcal{N}\big(\mu_s, \nu^{-1}\sigma_s^2\big) \quad \text{and} \quad \nu \,|\, s \sim \mathcal{G}(\alpha_s, \beta_s). \tag{7.15}$$

Similarly to the SAR-HMM, we consider a segmental approach where the state, scaling factor and noise variance are kept constant over a segment. Using $\vartheta_n = \{b_n, \nu_n\}$, Equations 7.13, 7.14 and 7.15 correspond to the distributions $p(v_t \,|\, y_t, \vartheta_n)$, $p(y_t \,|\, \tilde{\mathbf{y}}_t, s_n)$ and

$$p(\vartheta_n \,|\, s_n) = p(b_n \,|\, \nu_n, s_n)\, p(\nu_n \,|\, s_n)$$

respectively. The joint distribution defined by this model is

$$p(v_{1:T}, y_{1:T}, \vartheta_{1:N}, s_{1:N}) = \prod_{n=1}^{N} p(\vartheta_n \,|\, s_n)\, p(s_n \,|\, s_{n-1}) \prod_{t=t_n}^{t_{n+1}-1} p(v_t \,|\, y_t, \vartheta_n)\, p(y_t \,|\, \tilde{\mathbf{y}}_t, s_n). \tag{7.16}$$

Since the internal dynamics is autoregressive and priors are set on some of the parameters, this model is also a form of Bayesian AR-SLDS. The main difference is that the AR dynamics is learned on the basis of training data and is not allowed to adapt thereafter. This approach may improve the recognition accuracy since having a more constrained model will potentially help in noisy conditions where the speech signal is difficult to localise.

7.3.1 Parameter Optimisation

Compared to the AR-SLDS and the Bayesian AR-SLDS which can be initialised with a SAR-HMM and a Bayesian SAR-HMM respectively, the proposed scale-invariant AR-SLDS must be trained directly. Given a set of M training sequences[1] $\{v_{1:T}^1, \ldots, v_{1:T}^M\}$, we want to find the parameter setting $\Psi^\star$ which

[1]For simplicity, we assume that they all have the same length.

maximises the total log-likelihood of the training sequences, i.e.,

$$\Psi^{\star} = \arg\max_{\Psi} \sum_{m=1}^{M} \log p(v_{1:T}^{m} \,|\, \Psi) \tag{7.17}$$

where Ψ are the free parameters of the model, i.e.,

$$\Psi = \bigcup_{s} \Big\{ \mathbf{c}_s, \sigma_{\mathcal{H},s}^2, \mu_s, \sigma_s^2 \Big\} \cup \bigcup_{i,j} \Big\{ p(s_n = j \,|\, s_{n-1} = j) \Big\}.$$

Since our aim is to train the model on clean signals and to later test it on noisy data, we do not use a prior on ν during training and manually set appropriate values for α_s and β_s during testing. The likelihood of a sequence $v_{1:T}$ is

$$p(v_{1:T} \,|\, \Psi) = \sum_{s_{1:N}} \int_{\substack{\vartheta_{1:N} \\ y_{1:T}}} p(v_{1:T}, y_{1:T}, \vartheta_{1:N}, s_{1:N} \,|\, \Psi). \tag{7.18}$$

The sum and integral in Equation. 7.18 make an explicit solution to Equation 7.17 difficult to obtain. The usual approach would therefore be to use the EM algorithm. However, the non-linear interaction between y_t and ϑ_n in Equation 7.13 renders computing the required EM posterior $p(y_{1:T}, \vartheta_{1:N}, s_{1:N} \,|\, v_{1:T})$ intractable.

7.3.2 Variational Inference

An alternative to EM is to use a variational Bayesian approach where the true posterior distribution is approximated by the simpler distribution (Equation 7.3)

$$q(y_{1:T}, \vartheta_{1:N}, s_{1:N}) = q(y_{1:T} \,|\, s_{1:N})\, q(\vartheta_{1:N} \,|\, s_{1:N})\, q(s_{1:N}).$$

The other variational distribution, given by Equation 7.4, is not appropriate in this case because, in order to train the internal switching AR process, the dependency of $y_{1:T}$ on $s_{1:N}$ must be retained. By considering the KL divergence between the approximate and true posterior, we obtain the lower-bound

$$\begin{aligned} \log p(v_{1:T}) \geq & - \big\langle \log q(y_{1:T}, \vartheta_{1:N}, s_{1:N}) \big\rangle_q \\ & + \big\langle \log p(v_{1:T} \,|\, y_{1:T}, \vartheta_{1:N}, s_{1:N}) \big\rangle_q + \big\langle \log p(y_{1:T}, \vartheta_{1:N}, s_{1:N}) \big\rangle_q . \end{aligned} \tag{7.19}$$

Maximising the bound with respect to $q(\vartheta_{1:N} \,|\, s_{1:N})$ yields

$$\begin{aligned} & q(\vartheta_{1:N} \,|\, s_{1:N}) \propto \\ & \qquad \exp \Big\{ \big\langle \log p(v_{1:T} \,|\, y_{1:T}, \vartheta_{1:N}, s_{1:N}) \big\rangle_{q(y_{1:T} \,|\, s_{1:N})} \Big\}\, p(\vartheta_{1:N}, s_{1:N}) \end{aligned} \tag{7.20}$$

and maximising with respect to $q(y_{1:T}, s_{1:N})$ yields

$$\begin{aligned} q(y_{1:T}, s_{1:N}) \propto & \exp \Big\{ \big\langle \log p(v_{1:T} \,|\, y_{1:T}, \vartheta_{1:N}, s_{1:N}) \big\rangle_{q(\vartheta_{1:N} \,|\, s_{1:N})} \Big\} \\ & \times \exp \Big\{ \Big\langle \log \frac{p(\vartheta_{1:N} \,|\, s_{1:N})}{q(\vartheta_{1:N} \,|\, s_{1:N})} \Big\rangle_{q(\vartheta_{1:N} \,|\, s_{1:N})} \Big\}\, p(y_{1:T}, s_{1:N}). \end{aligned} \tag{7.21}$$

Finding $q(\vartheta_{1:N} \,|\, s_{1:N})$

Isolating the factors related to ϑ_n in Equation 7.20 gives

$$q(\vartheta_n \,|\, s_n) \propto \exp \Bigg\{ \sum_{t=t_n}^{t_{n+1}-1} \big\langle \log p(v_t \,|\, y_t, b_n, \nu_n) \big\rangle_{q(y_t \,|\, s_n)} \Bigg\}\, p(b_n \,|\, \nu_n, s_n)\, p(\nu_n \,|\, s_n)$$

which is similar to Equation 7.8. Since we chose conjugate priors, the posterior distribution has the same form as the prior, hence

$$b_n \,|\, \nu_n, s_n \sim \mathcal{N}(\hat{\mu}_{s_n}, \nu_n^{-1}\hat{\sigma}^2_{s_n}) \quad \text{and} \quad \nu_n \,|\, s_n \sim \mathcal{G}(\hat{\alpha}_{s_n}, \hat{\beta}_{s_n}).$$

Following the same derivation as in Section 7.2.1, for the variance and mean of b_n, we obtain

$$\hat{\sigma}^2_{s_n} = \sigma^2_{s_n}\left[1 + \sigma^2_{s_n}\sum_t \langle y_t^2\rangle\right]^{-1} \quad \text{and} \quad \hat{\mu}_{s_n} = \hat{\sigma}^2_{s_n}\left[\frac{\mu_{s_n}}{\sigma^2_{s_n}} + \sum_t v_t\langle y_t\rangle\right]$$

and, for the shape and inverse scale of ν_n, we have

$$\hat{\alpha}_{s_n} = \alpha_{s_n} + \frac{1}{2}T_n \quad \text{and} \quad \hat{\beta}_{s_n} = \beta_{s_n} + \frac{1}{2}\sum_t\left[v_t^2 + \frac{\mu_{s_n}}{\sigma^2_{s_n}} - \frac{\hat{\mu}_{s_n}}{\hat{\sigma}^2_{s_n}}\right]$$

where the averages are taken with respect to $q(y_t \,|\, s_n)$, $T_n = t_{n+1} - t_n$ and the sums are carried out from t_n to $t_{n+1} - 1$.

Finding $q(y_{1:T}, s_{1:N})$

Equation 7.21 can be written as

$$q(y_{1:T}, s_{1:N}) \propto \prod_{n=1}^{N} q(s_n \,|\, s_{n-1}) \prod_{t=t_n}^{t_{n+1}-1} q(v_t \,|\, y_t, s_n)\, p(y_t \,|\, \tilde{\mathbf{y}}_t, s_n) \tag{7.22}$$

where the logarithm of $q(v_t \,|\, y_t, s_n)$ is equal to

$$\begin{aligned}\big\langle \log p(v_t \,|\, y_t, \vartheta_n, s_n)\big\rangle &= \frac{1}{2}\big\langle \log\nu\big\rangle - \frac{1}{2}\big\langle \nu(v_t - by_t)^2\big\rangle \\ &= \frac{1}{2}\big\langle \log\nu\big\rangle - \frac{1}{2}\langle\nu\rangle\big(v_t - \langle b\rangle y_t\big)^2 - \frac{1}{2}\underbrace{\big\langle \nu\big(b - \langle b\rangle\big)^2\big\rangle}_{\hat{\sigma}^2_{s_n}} y_t^2\end{aligned}$$

and

$$q(s_n \,|\, s_{n-1}) \propto \exp\left\{\left\langle \log\frac{p(b_n, \nu_n \,|\, s_n)}{q(b_n, \nu_n \,|\, s_n)}\right\rangle\right\} p(s_n \,|\, s_{n-1})$$

where the averages are over $q(b_n, \nu_n \,|\, s_n)$. Since, $\langle b\rangle = \hat{\mu}_{s_n}$ and $\langle\nu\rangle = \hat{\alpha}_{s_n}/\hat{\beta}_{s_n}$, we have

$$q(v_t \,|\, y_t, s_n) \propto \exp\left\{-\frac{1}{2}\begin{bmatrix} v_t - \hat{\mu}_{s_n}y_t \\ \hat{\sigma}_{s_n}y_t\end{bmatrix}^{\mathsf{T}}\begin{bmatrix}\frac{\hat{\beta}_{s_n}}{\hat{\alpha}_{s_n}} & 0 \\ 0 & 1\end{bmatrix}^{-1}\begin{bmatrix} v_t - \hat{\mu}_{s_n}y_t \\ \sigma_{s_n}y_t\end{bmatrix}\right\}.$$

This can equivalently be written as a stochastic linear equation defined on an augmented observation,

$$\begin{bmatrix} v_t \\ 0\end{bmatrix} = \begin{bmatrix}\hat{\mu}_{s_n} \\ \hat{\sigma}_{s_n}\end{bmatrix} y_t + \boldsymbol{\eta}_t \quad \text{with} \quad \boldsymbol{\eta}_t \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\frac{\hat{\beta}_{s_n}}{\hat{\alpha}_{s_n}} & 0 \\ 0 & 1\end{bmatrix}\right). \tag{7.23}$$

Together with Equations 7.14 and 7.22, this defines a standard AR-SLDS for which the marginal posteriors $q(y_t, \tilde{\mathbf{y}}_t \,|\, s_n)$ and $q(s_n)$ can be computed using any of the numerous available algorithms found in the literature—see [7] for a review and comparison. For the experiments presented in this chapter, we performed approximate inference with the EC algorithm with collapse to a single Gaussian. We also used EC to find a first estimate of $q(y_t, \tilde{\mathbf{y}}_t \,|\, s_n)$ and $q(s_n)$ by running the algorithm on a AR-SLDS where the parameters where set to their mean value. Variational inference was then carried out by iteratively applying Equations 7.20 and 7.21.

7.3.3 Parameter Updating

With the posterior distribution in hands, update formulae for the parameters in Ψ can be obtained by means of the Variational Bayesian EM algorithm [12]. This corresponds to maximising the lower-bound given by Equation 7.19 with respect to Ψ. The update formula for the transition distribution is given by Equation 2.22. For the AR coefficients $\mathbf{c}_s$, the update formula is given by Equation 3.13. The formulae for the mean μ_s and variance σ_s^2 of the scaling parameter b_n are analogous to Equations 6.10 and 6.11. In this case, they read

$$\mu_s = \frac{\sum_n q(s_n = s)\,\hat{\mu}_{s_n}}{\sum_n q(s_n = s)} \quad \text{and} \quad \sigma_s^2 = \frac{\sum_n q(s_n = s)\left\langle \nu_n (b_n - \mu_s)^2 \right\rangle_{q(b_n,\nu_n \,|\, s_n)}}{\sum_n q(s_n = s)}.$$

Finally, the update formula for the innovation variance is

$$\sigma_{\mathcal{H},s}^2 = \frac{1}{\langle T_n \rangle} \sum_n q(s_n = s) \sum_{t=t_n}^{t_{n+1}-1} \left\langle (y_t - \mathbf{c}_s^{\mathsf{T}} \tilde{\mathbf{y}}_t)^2 \right\rangle_{q(y_t,\tilde{\mathbf{y}}_t \,|\, s_n)}$$

where $\langle T_n \rangle = \sum_n q(s_n = s)\, T_n$.

7.4 Training & Evaluation

Following Section 3.4, we trained a separate scale-invariant AR-SLDS for each of the eleven digits of the TI-DIGITS database. For each digit the model was composed of ten states with a left-to-right transition matrix. To each switch state was associated a 10-th order AR process and the segment length was of 140 samples (1.75 ms). The parameters to be trained were therefore: 9 transition probabilities and, for each state, the 10 AR coefficients, the innovation variance and the mean and variance of b. Since the models were trained on clean data, no prior was set on ν_n. The training was stopped after convergence of the lower bound given by Equation 7.19. Recognition was performed by picking the digit model for which the likelihood of the corresponding augmented observation (Equation 7.23) was the highest. Contrary to the Bayesian AR-SLDS, where the lower-bound is used for recognition, the bound was not used here because we did not want to penalise models for which the value of the scaling parameter was too different from the values observed in the training data. To give the model the opportunity to remove noise, we manually specified a prior on ν_n with a mean of 1 and a large variance. If we also add the two free parameters α and β introduced by the prior on $\nu_{\mathcal{V}}$, the model therefore contains $10 \times 13 + 9 + 2 = 141$ free parameters.

7.5 Example of Signal Reconstruction

To test the potential benefit of the proposed scale-invariant model, we compared the reconstructions of scaled noisy signals provided by the scale-invariant and the gain-adapted AR-SLDSs. As a demonstration, a clean utterance of a ‘one’ was taken, from which a scaled noisy version of the signal was then formed and corrupted by additive Gaussian white noise at SNR 0 dB. Given this scaled-noisy signal, the posterior $q(y_t, s_n)$ was then used to reconstruct the most likely (ML) signal. Figure 7.3 shows the ML reconstructed clean signal given by the gain-adapted AR-SLDS and the scaled-invariant AR-SLDS. The latter does not allow the innovation to change, resulting in less variability in the underlying signal and a more accurate denoising, particularly at the edges where the signal level is low. On the other hand, the gain-adapted AR-SLDS provides a reasonable reconstruction but, as a result of the extra innovation required to explain the change in signal level, allows the reconstructed signal too much freedom, particularly in the low signal level areas, as anticipated.

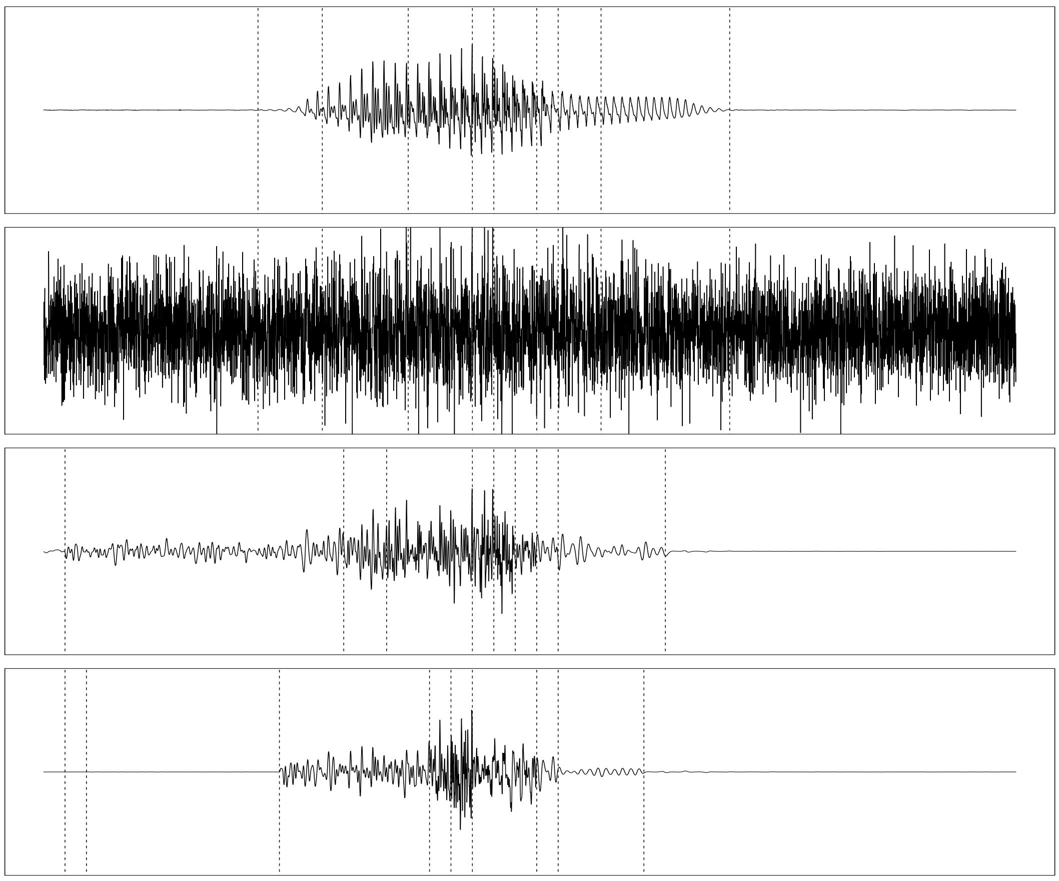


Figure 7.3: Comparison of signal reconstruction. From top to bottom: original waveform of a 'one', same waveform corrupted by additive Gaussian white noise at SNR 0.7 dB, ML reconstruction as given by a gain-adapted AR-SLDS and a scale-invariant AR-SLDS. The dashed lines indicate the most likely state segmentation. The state segmentation of the clean signal is shown on the noisy signal as well.

SNR (dB)	#prms	clean	26.3	25.1	19.7	10.6	0.7
HMM (MFCC)	4283	**100**	**100**	95.5	50	13.6	9.1
HMM (MFCC) + USS	4283	**100**	**100**	90.9	86.4	59.1	9.5
HMM (LPCC)	109	**100**	95.5	27.3	18.2	9.5	9.1
HMM (LPCC) + USS	109	**100**	86.4	77.3	18.2	13.6	10
SAR-HMM	119	88.3	25.5	9.7	8.6	9.3	9.4
SAR-HMM + GA	109	97.2	79.8	56.7	22.2	9.7	9.1
AR-SLDS	120	86.8	88.2	87.3	79.1	80	63.6
AR-SLDS + GA	109	96.8	96.8	**96.4**	**94.8**	84	61.2
Bayesian SAR-HMM	1129	98.7	38.2	22.7	9.1	9.1	13.4
Bayesian AR-SLDS	1131	98.5	—	95.5	94.2	**89.5**	52.3
Scale-Invariant AR-SLDS	141	87	—	—	83.3	78.3	**64**

Table 7.2: Comparison of the word accuracy (in percent), at various SNRs, of the scale-invariant AR-SLDS with all the other models considered so far. The best performance for each column is indicated in **bold**. The second column indicates the number of free parameters in the model. A word accuracy of 9.1% corresponds to random guessing.

7.6 Performance

We tested the scale-invariant AR-SLDS on the isolated digit recognition task described in Section 2.10. Table 7.2 compares the recognition accuracy of the scale-invariant AR-SLDS with all the models considered so far. Although there is a slight improvement at SNR 0.7 dB, the scale-invariant AR-SLDS is otherwise less accurate than its gain-adapted counterpart. This clearly shows that gain-adaptation does more than dealing with variations in the amplitude of the signal. It adds a limited form of variability which also improves accuracy. Furthermore, the similar performance of the scale-invariant AR-SLDS and the non-gain-adapted AR-SLDS tends to show that modelling scale invariance explicitly does not significantly help in improving the accuracy.

The proposed scale-invariant AR-SLDS does not adapt the innovation variance and uses only the scale to allow for changes in the signal. A natural extension would be to add priors on the AR coefficients and the innovation variance as well. Such a model should have the benefit that the additional variability will be required only in those cases that cannot be well explained by a simple rescaling of the underlying clean signal. Unfortunately, in practice, this approach does not perform well on clean data, reaching a recognition accuracy similar to that of the Bayesian AR-SLDS when used with the variational approximation given by Equation 7.3. A possible alternative is to use the simpler variational approximation given by Equation 7.4. However, we did not implement this model because we did not expect that modelling scale invariance explicitly would significantly improve the accuracy.

7.7 Experiments with Natural Noise Sources

So far we have been considering simple additive Gaussian white noise. To test the potential advantage of modelling the speech waveform directly over the more classical feature-based approach, we compared the recognition accuracy of the gain-adapted and Bayesian AR-SLDS against two state-of-the-art HMM systems, on the Aurora task [35]. The Aurora database is a modified version of the TI-DIGITS database where the utterances are convolved with a filter which mimics the frequency characteristic of a mobile phone, and are then artificially corrupted with real-world type noises recorded in a subway, car or exhibition hall for example. The training set we used was composed of 220 instances of each of the eleven digits (0–9 and 'oh') found in the database, pronounced by 110 different male or female

	Subway Noise						
SNR (dB)	clean	20	15	10	5	0	-5
HMM (MFCC) + USS	**99.3**	**88.3**	**68.8**	37.6	13.4	9.2	9.4
HMM (LPCC) + USS	96	58.4	44.6	29.9	19.8	9.1	9.1
AR-SLDS + GA	92.2	64.4	56	**40.3**	**30.9**	**20.1**	**13.1**
Bayesian AR-SLDS	96.1	52.7	34.2	27.2	19.9	19.8	9.1

Table 7.3: Comparison of the word accuracy (in percent) of two standard HMM systems against the gain-adapted and Bayesian AR-SLDS, on the 'subway noise' part of the Aurora database. The best performance for each column is indicated in **bold**. A word accuracy of 9.1% corresponds to random guessing.

speakers. The clean/noisy test sets were composed of 102/28 instances of each digit pronounced by 51/14 speakers, different from those of the training set. We trained two different HMM systems, one using MFCCs and the other LPCCs, following the setup and procedure described in Sections 2.10 and 3.5. The gain-adapted and Bayesian AR-SLDS were not trained directly but initialised with a gain-adapted and Bayesian SAR-HMM respectively, trained as explained in Sections 3.4 and 6.4. All the models were trained on clean data and tested on noisy data corrupted with convolutional and additive noise, apart from the clean test set for which only convolutional noise was used. Since our primary interest is in modelling additive noise, in our experiments, we did not use any technique which can potentially deal with convolutional noise—like channel normalisation [43] for example.

Table 7.3 shows the recognition accuracy of the various models on the 'subway noise' part of the Aurora database. For high SNRs, both HMM systems are superior to the AR-SLDSs. For moderate to low SNRs, the gain-adapted AR-SLDS performs better. The better accuracy of the AR-SLDS at low SNRs is expected since it uses an internal model of clean speech while, in the case of the HMMs, features are filtered independently of the model. Although the AR-SLDS can theoretically only deal with additive Gaussian white noise, in practice, this is sufficient since, at low SNRs, the original structure of the clean waveform is almost completely lost. Therefore, being able to roughly infer the clean waveform already provides a significant advantage over a pre-processing method like USS. In contrast, for high SNRs, the structure of the speech waveform is more evident and it becomes important to detect it accurately. The performance of the AR-SLDS is worse in this case because the part of the noise which is not properly filtered out by the Gaussian white noise model is considered as speech. USS is more accurate in this situation because it is better able to discriminate speech from noise.

A possible way to deal with more elaborate noise sources in the AR-SLDS is to consider that a noisy sample v_t is a corrupted version of the sum of a clean hidden sample y_t and a hidden pure noise sample u_t:

$$v_t = y_t + u_t + \eta_t^{\nu} \quad \text{and} \quad \eta_t^{\nu} \sim \mathcal{N}(0, \sigma_{\nu}^2).$$

The dynamics of the noise can then be modelled by a Bayesian AR process of the form:

$$u_t = \mathbf{c}_{\mathcal{U}}^{\mathsf{T}} \tilde{\mathbf{u}}_t + \eta_t^{\mathcal{U}} \quad \text{and} \quad \eta_t^{\mathcal{U}} \sim \mathcal{N}(0, \sigma_{\mathcal{U}}^2) \tag{7.24}$$

with a zero mean prior on $\mathbf{c}_{\mathcal{U}}$ and a prior with a small mean on $\sigma_{\mathcal{U}}^2$. The central idea behind this approach is to allow the AR-SLDS to model non-white noise explicitly by means of the AR process defined by Equation 7.24. A Bayesian treatment is essential in this case since otherwise, with a GA-type approach, adapting the AR coefficients $\mathbf{c}_{\mathcal{U}}$ and variance $\sigma_{\mathcal{U}}^2$ for each utterance, without constraint, would be susceptible to overfitting. In contrast, setting a zero mean prior on $\mathbf{c}_{\mathcal{U}}$ tells the model that a bit of correlation is possible between the noise samples u_t, but ideally this should not be needed. Although this approach sounds appealing, in practice, the preliminary experiments we carried out with this model showed that the noise model is rarely used. Since the accuracy was therefore expected to be similar to that of the standard Bayesian AR-SLDS, we did not evaluate the performance of this model on the complete database. A likely explanation for the fact that an explicit AR model of noise does not improve the accuracy is that, compared to speech waveforms, most noise sources—in this case, subway noise—are poorly modelled by an AR process.

Apart from the clean case, the Bayesian approach does not improve the accuracy of the AR-SLDS. A possible explanation is that the Bayesian AR-SLDS is more conservative than its gain-adapted counterpart and therefore cannot deal with non-white noise properly. However, on clean data, the accuracy of the Bayesian AR-SLDS is better than that of the gain-adapted AR-SLDS and comparable to that of the LPCC-based HMM system. This suggests that the gain-adapted AR-SLDS might be overfitting.

7.8 Summary

The Bayesian AR-SLDS extends the Bayesian SAR-HMM to include an explicit model of additive Gaussian white noise. Inference in the Bayesian AR-SLDS is significantly more complicated than in the original AR-SLDS, preventing the direct use of standard approximation algorithms like EC. To address this problem we proposed a variational approximation for which inference can be carried out with traditional methods. In clean conditions, the accuracies of the Bayesian AR-SLDS and SAR-HMM are comparable and both Bayesian models perform slightly better than their gain-adapted counterpart. This suggests that the Bayesian approach is a valuable alternative which can potentially replace GA. The accuracies of the Bayesian and gain-adapted AR-SLDS are comparable for high to moderate SNRs. However, at lower SNRs, the assumptions made in the proposed variational approximation may fail to properly localise the position of the clean waveform in the noisy signal and potentially prevent correct recognition.

We also considered an alternative Bayesian AR-SLDS where scale invariance is explicitly modelled by representing a noisy signal as a scaled version of a clean hidden signal corrupted with additive Gaussian white noise. This approach generally results in cleaner reconstructions than approaches based on GA since it allows for less variability in the underlying signal and a more accurate denoising, particularly at the edges where the signal level is low. Furthermore, at low SNRs, it is also more accurate than all the other models considered. However, at high to moderate SNRs, the accuracy of the scale-invariant AR-SLDS is significantly lower than that of the other models. Furthermore, the similar performance of the scale-invariant and the non-gain adapted AR-SLDS tends to show that modelling scale invariance explicitly does not significantly help in improving recognition accuracy.

Although, theoretically, the AR-SLDS can only deal with additive Gaussian white noise, in practice, this model can also be effective on more natural sources of noise. At high SNRs, the performance of the AR-SLDS is worse than that of feature-based HMM systems because the part of the noise which is not properly filtered by the Gaussian white noise model is considered as speech. In contrast, for moderate to low SNRs, the gain-adapted AR-SLDS is superior because it can recover, thanks to its internal model of clean speech, a rough approximation of the clean waveform, while the filtering quality of pre-processing techniques drops significantly in adverse conditions since the structure of the original clean waveform is almost completely lost.

Chapter 8

Discussion & Future Directions

8.1 Conclusions

This thesis investigated the potential advantages of modelling the speech waveform directly for performing noise robust recognition of isolated digits. The central idea was to use the simple and well-known AR process as a basic ingredient upon which more refined models were devised. Two contributions of this thesis were to improve the accuracy of current AR models in noisy conditions by introducing an explicit model of additive Gaussian white noise, and to propose a probabilistic principled solution to the problem of dealing with variations in the signal amplitude. Compared to existing models, where computing the posterior distribution of the hidden variables can be carried out exactly, inference in the proposed models is formally intractable. Another contribution of this thesis was therefore to propose a new, stable and accurate approximate inference algorithm which removes some of the limitations of existing approximation procedures.

8.1.1 Switching Linear Dynamical Systems

We started by investigating the problem of making the original SAR-HMM [27] more robust to additive noise. This was achieved by viewing an observed noisy sample as a clean hidden sample corrupted by additive Gaussian white noise. This approach naturally led us to the class of Switching Linear Dynamical Systems (SLDSs). To deal with noise without having to train a new model, we devised the AR-SLDS, a special case of SLDSs where the internal dynamics is autoregressive. The advantage of this approach is that a SAR-HMM trained on clean data can readily be used on data corrupted by additive noise. Thanks to its internal AR model of clean speech, for moderate to low SNRs, the AR-SLDS is more robust than a state-of-the-art HMM system. However, for more natural noise sources, at high SNRs, the HMM system is in general more accurate because the AR-SLDS is not always able to properly filter out non-Gaussian white noise. A limitation of the AR-SLDS is that its accuracy crucially depends on the use of Gain Adaptation (GA). GA is problematic because it allows the parameters to be adjusted on test data and this may lead to overfitting. The Bayesian approach is an alternative which yields a performance comparable to that of a state-of-the-art HMM system in clean conditions. However, the difficulty of accurately approximating the likelihood in the Bayesian AR-SLDS makes its use in noisy conditions problematic.

8.1.2 Approximate Inference in the SLDS

We presented the EC algorithm, a novel approximation procedure which addresses some of the limitations of current approximation techniques. EC is based on the RTS method which was originally developed for the Linear Dynamical System (LDS). It is a forward-backward-type algorithm where the backward pass corrects the result from the forward pass to form the desired posterior. Both passes produce posteriors under the form a mixture of Gaussians whose number of components grows exponentially with the length of the time series. Like the Generalised Pseudo Bayes (GPB) technique, the main approximation within EC is to collapse the mixture to a mixture with less components or, even more drastically, to a single Gaussian. Compared to GPB, EC's backward pass is more accurate since more of the available future information is used to update the posterior of the switch state. In this thesis, for computational and memory efficiency, we used EC in its most simple form—with collapse to a single Gaussian—for all the experiments involving a SLDS. In general, EC proved to be more stable than Expectation Propagation while being more accurate and faster than Monte-Carlo methods.

8.1.3 A Bayesian Alternative to Gain Adaptation

Another issue we investigated in this thesis is how to deal with variations in the signal amplitude in AR models. We proposed a Bayesian alternative to GA where the parameters were considered as random hidden variables. A potential advantage of the Bayesian approach is that the variability of the parameter setting is explicitly controlled by prior distributions fitted on training data and no further adjustment is required. In clean conditions, the performance of the proposed Bayesian SAR-HMM was comparable to that of the gain-adapted SAR-HMM. To deal with noise without having to train a new model, we devised the Bayesian AR-SLDS, an extension of the Bayesian SAR-HMM which includes an explicit model of additive Gaussian white noise. We addressed the problem of performing approximate inference in the Bayesian AR-SLDS by devising a variational approximation for which inference could be carried out with existing inference methods. For data artificially corrupted with additive Gaussian white noise at high to moderate SNRs, the performance of the Bayesian AR-SLDS was comparable to that of its gain-adapted counterpart. At low SNRs, the assumption that the clean waveform and the switch states could be inferred independently was probably too limiting and prevented the correct identification of the clean waveform. We tried to address this problem by proposing a model where the observed signal is considered as a scaled and corrupted version of a clean hidden signal, which is modelled by a standard SAR-HMM. Although this approach is better able to localise the clean waveform at low SNRs, alone it is too limited to deal with variabilities which cannot be considered as simple rescalings of signals. Nevertheless, in conjunction with innovation noise, rescaling should, in principle, aid the extraction of clean waveforms in noisy environments.

8.1.4 Computational Tractability

The models proposed in this thesis are computationally very expensive. For example, processing a noisy utterance of one second takes, on a 2.6 GHz dual core Pentium 4, about one minute and a half. Since there are eleven digits, recognition is therefore $11 \times 90 = 990$ time slower than real time. A more problematic issue is that, in general, the complexity of performing approximate inference in the SLDS is $\mathcal{O}(S^2 \cdot R^3 \cdot T)$, and the memory required to store the various statistics is $\mathcal{O}(S^2 \cdot R^2 \cdot T)$, where S is the number of states, R the number of AR coefficients and T the number of samples. Although the complexity can be reduced by using a segmental approach, a left-to-right transition matrix and AR dynamics, the computational time and memory consumption quickly becomes prohibitive in more generic cases. For example, on a single machine, for the gain-adapted AR-SLDS, the time required to obtain a single result reported in the third line of Table 7.3, was about 22 days. Testing the proposed models on more complicated tasks, like the recognition of connected digits, or in a large vocabulary application, will therefore require algorithmic surrogates. In the future, with the improvement of computational resources, the models we considered might nevertheless become much more attractive and might also be applicable beyond ASR.

8.2 Application to Other Fields

The application of the SLDS is of course not limited to speech since this class of models is used in many disciplines concerned with time series modelling or prediction, including econometrics, biology, brain-computer interfaces and machine learning in general—see for example [6, 20, 32, 39, 40, 45, 54] and also [46, 72] for a review of recent work. The work done during this thesis is therefore potentially useful to other fields as well. For example, though it might not be useful for ASR, the proposed Bayesian treatment of the SLDS is transferable to other applications where allowing the model to adapt to changing conditions is important. The EC algorithm is another example, since it is a generic algorithm for performing inference in any kind of SLDSs, it can potentially be applied in any place where a SLDS is used.

8.3 Limitations & Future Directions

In the context of isolated digits recognition, the experiments we performed suggest that modelling clean speech can potentially significantly enhance noise robustness. Although we took the extreme choice of modelling the clean waveform directly, less drastic approaches might potentially be more successful and might scale better to applications such as large vocabulary ASR. Our choice of working with the waveform was motivated by the fact that, at this level, modelling additive noise is easy. However, this approach introduces a number of problems which are difficult to address at the waveform level. We mainly considered the problem of dealing with variations in the signal amplitude, but other problems like the modelling of subword unit duration or non-stationary sources of noise are also worth considering. Modelling duration is expected to enhance the quality of the signal reconstruction because it can potentially prevent the appearance of too long segments at the edges of the signal where the signal level is low. As we have seen in our experiments on the Aurora task, properly modelling non-stationary and non-Gaussian white noise is also important to improve accuracy on real-world noise sources. Our approach to this problem was to include an explicit Bayesian AR model of noise. However, this strategy did not work in practice, most likely because an AR model is not appropriate to model the kinds of complex noise environments in real world scenarios which may express greater variability than can be reasonably modelled by an AR process.

The choice of the AR process to model the speech waveform is also debatable since it is known to have difficulties to properly model fricatives and voiced speech [67]. AR models are nevertheless appealing because they have a small number of parameters and can generally be dealt with exactly. This is the main reason why, in this thesis, we limited ourselves to modelling the speech waveform with an AR model. A possible future direction could be to consider a generic SLDS and to see what performance it has on ASR. A potentially fruitful direction could be to work in the Fourier domain and to model the harmonic components of the signal instead of the waveform. A good example of this type of approach applied to acoustic modelling can be found in [18], where a SLDS is used for music transcription. Working in the Fourier domain is however considerably more demanding in terms of computational power because the dimensionality is much bigger than at the waveform level. Another problem with working with a generic SLDS is that, in general, it does not reduce to a simpler model—like the AR-SLDS which reduces to the SAR-HMM in clean conditions—and the problem of training such a complex model must therefore be considered.

Another clear limitation of the proposed approach is its computational complexity. This limits its practical use to relatively simple tasks like the recognition of isolated digits and prevents its application to more challenging tasks like large vocabulary ASR. For example, given current computational resources, performing continuous speech recognition, which requires modelling the speech signal at the phoneme level, is almost impossible without the development of algorithmic surrogates. Furthermore, to keep the running time of the experiments acceptable, we only considered the setting proposed in [27] for the segment length, the number of states and the order of the AR processes. In the future, when more computational power will be available, it might be interesting to investigate how those parameters affect the accuracy of the proposed models. Another potentially fruitful future direction might also be to consider a discriminative approach to training the models. Discriminative training becomes increasingly popular in classical feature-based speech recognition and has been shown to significantly enhance the accuracy of the models [68]. Discriminative training approaches such as those described in [37] could be applied to the SAR-HMM and potentially yield similar performance improvements. In many situations, the proposed maximum likelihood approach was nevertheless capable to identify and reconstruct an approximate clean signal even in very noisy conditions. This suggests a possible future application where, instead of being used for recognition, the AR-SLDS would be used as an elaborate pre-processing step which would provide filtered clean speech signals to a standard feature-based HMM system. A potential advantage of this approach over a more traditional one, like USS, is that the dynamics of the resulting reconstructed clean waveform will be closer to that of the training examples, thereby facilitating recognition by the HMM.

Appendices

A Properties of the Gamma Distribution

Given $\nu \sim \mathcal{G}(\alpha, \beta)$, then

$$\langle \nu \rangle = \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^{+\infty} \nu \, \nu^{\alpha-1} e^{-\beta\nu} d\nu = \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(\alpha+1)}{\beta^{\alpha+1}} = \frac{\alpha}{\beta}$$

where, for the last equality, we used the fact that $\Gamma(\alpha+1) = \alpha\Gamma(\alpha)$. The average of the logarithm of ν is given by

$$\langle \log \nu \rangle = \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^{+\infty} \log(\nu) \, \nu^{\alpha-1} e^{-\beta x} d\nu.$$

This integral can be evaluated by considering the partial derivative, with respect to α, of the normalisation constant of the Gamma distribution[1]. Hence

$$\langle \log \nu \rangle = \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\partial}{\partial \alpha} \int_0^{+\infty} \nu^{\alpha-1} e^{-\beta\nu} d\alpha = \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\partial}{\partial \alpha} \frac{\Gamma(\alpha)}{\beta^\alpha}$$

and since

$$\frac{\partial}{\partial \alpha} \frac{\Gamma(\alpha)}{\beta^\alpha} = \Gamma'(\alpha) \frac{1}{\beta^\alpha} - \Gamma(\alpha) \log(\beta) \frac{1}{\beta^\alpha}$$

we end up with

$$\langle \log \nu \rangle = \psi(\alpha) - \log \beta$$

where $\psi(\cdot) = \Gamma'(\alpha)/\Gamma(\alpha)$ is the Digamma function [1].

B Solution to Equations 6.7 and 6.13

Our goal is to find $\hat{\alpha}$ such that

$$\log \hat{\alpha} - \psi(\hat{\alpha}) = \Omega$$

where Ω denotes the rhs of either Equation 6.7 or Equation 6.13. A solution can be efficiently obtained by using the Newton-Raphson's method. If we define

$$f(\alpha) = \log \alpha - \psi(\alpha) - \Omega$$

then, given the i-th estimate α_i of $\hat{\alpha}$, the next estimate is given by

$$\alpha_{i+1} = \alpha_i - \frac{f(\alpha_i)}{f'(\alpha_i)} \tag{1}$$

[1]Thanks to David Barber for pointing this out.

where $f'(\cdot)$ denotes the derivative of f with respect to α. A good initial estimate α_0 can be obtained by considering the asymptotic expansion [1]:

$$\psi(\alpha) = \log\alpha - \frac{1}{2\alpha} - \frac{1}{12\alpha^2} + \mathcal{O}(\alpha^4).$$

The first derivative of $f(\cdot)$ required by Equation 1 is given by

$$\frac{d}{d\alpha}\Big[\log\alpha - \psi(\alpha)\Big] = \frac{1}{\alpha} - \psi_1(\alpha)$$

where $\psi_1(\cdot)$ is the Trigamma function [1].

C Computing $\mathbf{S}_n$

Using the convention that, unless explicitly specified, the averages are taken with respect to $q(\vartheta_n, s_n)$, we have

$$\begin{aligned}\mathbf{S}_n &= \big\langle \nu_n(\mathbf{c}_n - \langle\mathbf{c}_n\rangle)(\mathbf{c}_n - \langle\mathbf{c}_n\rangle)^{\mathsf{T}}\big\rangle \\ &= \langle\nu_n\mathbf{c}_n\mathbf{c}_n^{\mathsf{T}}\rangle - \langle\nu_n\mathbf{c}_n\rangle\mathbf{c}_n^{\mathsf{T}} - \mathbf{c}_n\langle\mathbf{c}_n^{\mathsf{T}}\nu_n\rangle + \langle\mathbf{c}_n\rangle\langle\nu_n\rangle\langle\mathbf{c}_n^{\mathsf{T}}\rangle\end{aligned}$$

with

$$\begin{aligned}\big\langle\nu_n\mathbf{c}_n\mathbf{c}_n^{\mathsf{T}}\big\rangle &= \sum_{s_n} q(s_n)\,\langle\nu_n\mathbf{c}_n\mathbf{c}_n^{\mathsf{T}}\rangle_{q(\mathbf{c}_n,\nu_n\,|\,s_n)} \\ &= \sum_{s_n} q(s_n)\,\Big[\big\langle\nu_n(\mathbf{c}_n - \hat{\boldsymbol{\mu}}_s)(\mathbf{c}_n - \hat{\boldsymbol{\mu}}_s)^{\mathsf{T}}\big\rangle_{q(\mathbf{c}_n,\nu_n\,|\,s_n)} + \langle\nu_n\rangle_{q(\nu_n\,|\,s_n)}\,\hat{\boldsymbol{\mu}}_s\hat{\boldsymbol{\mu}}_s^{\mathsf{T}}\Big] \\ &= \sum_{s_n} q(s_n)\left[\hat{\boldsymbol{\Sigma}}_{s_n} + \frac{\hat{\alpha}_{s_n}}{\hat{\beta}_{s_n}}\hat{\boldsymbol{\mu}}_{s_n}\hat{\boldsymbol{\mu}}_{s_n}^{\mathsf{T}}\right]\end{aligned}$$

and

$$\big\langle\nu_n\mathbf{c}_n\big\rangle = \sum_{s_n} q(s_n)\,\langle\nu_n\rangle_{q(\nu_n\,|\,s_n)}\langle\mathbf{c}_n\rangle_{q(\mathbf{c}_n\,|\,\nu_n,s_n)} = \sum_{s_n} q(s_n)\,\frac{\alpha_{s_n}}{\beta_{s_n}}\hat{\boldsymbol{\mu}}_{s_n}.$$

References

[1] M. Abramowitz and I. A. Stegun, editors. *Handbook of mathematical functions with formulas, graphs and mathematical tables, tenth printing*, chapter 6, Gamma functions and related functions, pages 253–266. National Bureau of Standards, 1972.

[2] K. Achan, S. T. Roweis, and B. J. Frey. A segmental HMM for speech waveforms. Technical Report UTML-TR-2004-001, University of Toronto, January 2004.

[3] D. L. Alspach and H. W. Sorensen. Nonlinear Bayesian estimation using Gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17(4):439–448, 1972.

[4] L. M. Arslan. Modified wiener filtering. *Signal Processing*, 86(2):267–272, February 2006.

[5] H. Attias. Variational Bayesian framework for graphical models. In *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 209–215, 2000.

[6] Y. Bar-Shalom and X.-R. Li. *Estimation and tracking: principles, techniques and software.* Artech House, Norwood, MA, 1998.

[7] D. Barber. Expectation correction for smoothed inference in switching linear dynamical systems. *Journal of Machine Learning Research*, 7:2515–2540, November 2006.

[8] D. Barber and S. Chiappa. Unified inference for variational Bayesian linear Gaussian state-space models. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, 2006.

[9] D. Barber and B. Mesot. A novel Gaussian sum smoother for approximate inference in switching linear dynamical systems. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, 2006.

[10] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.

[11] M. J. Beal and Z. Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Statistics*, 7:453–464, 2002.

[12] M. J. Beal and Z. Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In Oxford University Press, editor, *Bayesian Statistics 7*, pages 453–464, 2003.

[13] M. Berouti, R. Schwartz, and J. Makhoul. Enhancement of speech corrupted by acoustic noise. In *Proceedings of ICASSP'79*, pages 208–211, April 1979.

[14] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transaction on Acoustic, Speech and Signal Processing*, 27(2):113–120, April 1979.

[15] H. Bourlard. Non-stationary multi-channel (multi-stream) processing towards robust and adaptive ASR. In *Proceedings of the ESCA Workshop on Robust Methods for Speech Recognition in Adverse Conditions*, pages 1–9, 1999.

[16] J. S. Bridle. Towards better understanding of the model implied by the use of dynamic features in HMMs. In *Proceedings of ICSLP*, volume 1, pages 725–728, October 2004.

[17] C. Carter and R. Kohn. Markov chain Monte Carlo in conditionally Gaussian state space models. *Biometrika*, 83:589–601, 1996.

[18] A. T. Cemgil, B. Kappen, and D. Barber. A generative model for music transcription. *IEEE Transactions on Audio, Speech and Language Processing*, 14(2):679–694, 2006.

[19] R. Chen and J. S. Liu. Mixture Kalman filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(3):493–508, 2000.

[20] S. Chiappa. *Analysis and Classification of EEG Signals using Probabilistic Models for Brain Computer Interfaces*. PhD thesis, IDIAP Research Institute, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland, 2006.

[21] A. P. Dempster. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[22] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte-Carlo Methods in Practice*. Springer, 2001.

[23] A. Doucet, N. de Freitas, K. Murphy, and S. Russel. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2000.

[24] J. Droppo and A. Acero. Noise robust speech recognition with a switching linear dynamical model. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, volume 1, May 2004.

[25] Y. Ephraim. Gain–adapted hidden Markov models for recognition of clean and noisy speech. *IEEE Transaction on Signal Processing*, 40(6):1303–1316, June 1992.

[26] Y. Ephraim and D. Malah. Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 33:443–445, 1985.

[27] Y. Ephraim and W. J. J. Roberts. Revisiting autoregressive hidden Markov modeling of speech signals. *IEEE Signal Processing Letters*, 12(2):166–169, February 2005.

[28] ETSI. Speech processing, transmission and quality aspects (STQ); distributed speech recognition; advanced frond-end feature extraction algorithm; compression algorithms. ETSI standard doc. ES 202 050 V1.1.1, October 2002.

[29] M. J. F. Gales. *Model-based techniques for noise robust speech recognition*. PhD thesis, Cambridge University, UK, 1995.

[30] M. J. F. Gales. Nice mode-based compensation schemes for robust speech recognition. In *Proceedings of ESCA/NATO Workshop on Robust Speech Recognition for Unknown Communication Channels*, pages 55–64, France, April 1997.

[31] M. J. F. Gales. Predictive model-based compensation schemes for robust speech recognition. *Speech Communication*, 25(1–3):49–74, 1998.

[32] Z. Ghahramani and G. E. Hinton. Variational learning for switching state-space models. *Neural Computation*, 12(4):963–996, 1998.

[33] A. Hagen. *Robust Speech Recognition Based on Multi-Stream Processing*. PhD thesis, École Polytechnique Fédérale de Lausanne (EPFL), 2001.

[34] H. Hermansky. RASTA processing of speech. *IEEE Transaction on Speech and Audio Processing*, 2(4):578–589, October 1994.

[35] H.-G. Hirsch and D. Pearce. The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ASR-2000*, pages 181–188, 2000.

[36] W. J. Holmes and M. J. Russel. Probabilistic-trajectory segmental HMMs. *Computer Speech and Language*, 13(1):3–37, January 1999.

[37] S. Kapadia. *Discriminative Training of Hidden Markov Models.* PhD thesis, University of Cambridge, 1998.

[38] C.-J. Kim. Dynamic linear models with markov-switching. *Journal of Econometrics*, 60(1–2):1–22, 1994.

[39] C.-J. Kim and C. R. Nelson. *State-Space Models with Regime Switching.* MIT Press, 1999.

[40] G. Kitagawa. The two-filter formula for smoothing and an implementation of the Gaussian-sum smoother. *Annals of the Institute of Statistical Mathematics*, 46(4):605–623, 1994.

[41] G. Kitagawa. Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, March 1996.

[42] G. Lathoud, M. Magimai-Doss, and H. Bourlard. Channel normalization for unsupervised spectral subtraction. Technical Report IDIAP-RR-06-09, IDIAP Research Institute, September 2006.

[43] G. Lathoud, M. Magimai-Doss, B. Mesot, and H. Bourlard. Unsupervised spectral subtraction for noise-robust ASR. In *Proceedings of ASRU 2005*, pages 189–194, November 2005.

[44] R.G. Leonard. A database for speaker independent digit recognition. In *Proceedings of ICASSP84*, volume 3, 1984.

[45] U. Lerner, R. Parr, D. Koller, and G. Biswas. Bayesian fault detection and diagnosis in dynamic systems. In *Proceedings of the Seventeenth National Converence on Artificial Intelligence (AIII-00)*, pages 531–537, 2000.

[46] U. N. Lerner. *Hybrid Bayesian networks for reasoning about complex systems.* PhD thesis, Stanford University, 2002.

[47] J. Liu and R. Chen. Sequential Monte-Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[48] B. Mesot and D. Barber. A Bayesian alternative to gain adaptation in autoregressive hidden Markov models. In *Proceedings of ICASSP 2007*, April 2007.

[49] B. Mesot and D. Barber. Switching linear dynamical systems for noise robust speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 15(6):1850–1858, August 2007.

[50] T. Minka. *A family of algorithms for approximate Bayesian inference.* PhD thesis, MIT Media Lab, 2001.

[51] A. Morris, A. Hagen, H. Glotin, and H. Bourlard. Multi-stream adaptive evidence combination for noise robust ASR. *Speech Communication*, 34(1–2):25–40, April 2001.

[52] K. Murphy. Learning switching Kalman filter models. Technical Report 98-10, Compaq Cambridge Research Lab, 1998.

[53] M. Ostendorf, V. V. Digalakis, and O. A. Kimball. From HMM's to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(5):360–378, September 1996.

[54] V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems (NIPS 13)*, pages 981–987, 2001.

[55] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[56] A. B. Poritz. Linear predictive hidden Markov models and the speech signal. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 1291–1294, May 1982.

[57] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*, chapter 2.7 Sparse Linear Systems. Cambridge University Press, 1992.

[58] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[59] B. Raj and R. Singh. Tracking noise via dynamical systems with a continuum of states. In *Proceedings of ICASSP 2003*, 2003.

[60] B. Raj, R. Singh, and R. Stern. On tracking noise with linear dynamical system models. In *Proceedings of ICASSP 2004*, 2004.

[61] H. E. Rauch, G. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *Journal of American Institute of Aeronautics and Astronautics*, 3(8):1445–1450, 1965.

[62] C. Ris and S. Dupont. Assessing local noise level estimation methods: Application to noise robust ASR. *Speech Communication*, 34(1–2):141–158, April 2001.

[63] A.-V. I. Rosti. *Linear Gaussian Models for Speech Recognition.* PhD thesis, University of Cambridge, EN, May 2004.

[64] K. Tokuda, H. Zen, and T. Kitamura. Reformulating the HMM as a trajectory model. In *Proceedings of Beyond HMM Workshop on statistical modeling approach for speech recognition*, December 2004.

[65] K. Tokuda, H. Zen, and T. Kitamura. Trajectory modeling based on HMMs with the explicit relationship between static and dynamic features. In *Proceedings of Eurospeech*, 2004.

[66] A. P. Varga and R. K. Moore. Hidden Markov model decomposition of speech and noise. In *Proceedings of ICASSP'90*, pages 845–848, 1990.

[67] J. Vermaak, M. Niranjan, and S. J. Godsill. An improved speech production model for voiced speech utilising a seasonal AR-AR model and Markov Chain Monte Carlo simulation. Technical Report CUED/F-INFENG/TR.325, Speech, Vision and Robotics Group; Department of Engineering, University of Cambridge, June 1998.

[68] K. Vertanen. An overview of discriminative training for speech recognition. Technical report, Computer Speech, Text and Internet Technology, University of Cambridge, UK, 2004.

[69] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transations on Information Theory*, IT-13:260–269, 1967.

[70] M. Wölfel and F. Faubel. Considering uncertainty by particle filter enhanced speech features in large vocabulary continuous speech recognition. In *Proceedings of ICASSP 2007*, 2007.

[71] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book*. 2002.

[72] O. Zoeter. *Monitoring non-linear and switching dynamical systems*. PhD thesis, Radboud University, Nijmegen, 2005.