Bayesian Modelling of Music: Algorithmic Advances and Experimental Studies of Shift-Invariant Sparse Coding

Thesis submitted in partial fulfilment
of the requirements of the University of London
for the Degree of Doctor of Philosophy

Thomas Blumensath

Submitted: October 2005
Corrections: January 2006

Department of Electronic Engineering,
Queen Mary, University of London

I certify that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline. I acknowledge the helpful guidance and support of my supervisor, Dr Mike Davies.

Abstract

In order to perform many signal processing tasks such as classification, pattern recognition and coding, it is helpful to specify a signal model in terms of meaningful signal structures. In general, designing such a model is complicated and for many signals it is not feasible to specify the appropriate structure. Adaptive models overcome this problem by learning structures from a set of signals. Such adaptive models need to be general enough, so that they can represent relevant structures. However, more general models often require additional constraints to guide the learning procedure.

In this thesis a sparse coding model is used to model time-series. Relevant features can often occur at arbitrary locations and the model has to be able to reflect this uncertainty, which is achieved using a shift-invariant sparse coding formulation. In order to learn model parameters, we use Bayesian statistical methods, however, analytic solutions to this learning problem are not available and approximations have to be introduced. In this thesis we study three approximations, one based on an analytical integral approximation and two based on Monte Carlo approximations. But even with these approximations, a solution to the learning problem is computationally too expensive for the applications under investigation. Therefore, we introduce further approximations by subset selection.

Music signals are highly structured time-series and offer an ideal testbed for the studied model. We show the emergence of note- and score-like features from a polyphonic piano recording and compare the results to those obtained with a different model suggested in the literature. Furthermore, we show that the model finds structures that can be assigned to an individual source in a mixture. This is shown with an example of a mixture containing guitar and vocal parts for which blind source separation can be performed based on the shift-invariant sparse coding model.

To Rachel

Acknowledgements

I am very grateful to have been given the opportunity to spend the last three years dedicated to the research which led to this thesis. This would not have been possible had I not received tremendous support from my family, friends and colleagues. I would like to thank my father first of all, as his support enabled me to fully concentrate on my work without financial worries. I would also like to thank the rest of my family for believing in me, even though they could not always fully grasp all the issues involved in my work.

Probably the most influential person during my time as a postgraduate student was my supervisor Dr Mike Davies. I have been extremely lucky to be working with a person who shared the same outlook on many scientific topics with me. Without his guidance I would not have been able to achieve even a fraction of what I have. I would like to thank him for the freedom I was given and the inspiring discussions we had even though these were often sparsely distributed. These discussions can be seen as the foundations of this thesis.

The Centre for Digital Music has been a fantastic place to study and work. It offered the perfect environment for research providing ample opportunity to discuss research and learn from the experiences of a broad variety of persons with different backgrounds and ideas. Of all the researchers in the group, the greatest influence on my work had Samer Abdallah, whose thesis inspired me to follow the research which led to this thesis.

I also would like to thank Dan Ellis, who allowed me to spend two productive months during the Summer of 2004 in the Laboratory for the Recognition and Organisation of Sound and Audio at Columbia University in the City of New York. This stay was made possible by the generous support of the Royal Society of Engineering and the EPSRC Digital Music

Research Network. I would further like to thank the EPSRC ICA Research Network for their financial support.

From my university life before research, I would like to thank Iain Paterson-Stephens, who first introduced me to the world of digital signal processing during my undergraduate years and who was the first to encourage me to study for a PhD.

Most of my gratitude must go to the most important person in my life, Rachel, who kept me sane and well dressed during the last three years. I appreciate the sacrifices she made for me and my studies during our lives in separate parts of the country. Rachel had more direct influence on my work, spending much of her valuable time reading through many of my papers and in particular this thesis to criticise my often ‘unconventional’ use of the English language. As the reader can verify, I have often ignored her advice.

Contents

List of Figures

List of Tables

Nomenclature

$\mathbf{a}_n$	feature
$\mathbf{a}_{kl}$	feature k at shift l
c	term used to abbreviate equations
$\mathbf{d}$	double atoms
f	regularisation or error term
fn	false negative error
fp	false positive error
g	regularisation or error term
i	index of observations; also used as index of index, i.e. s_{n_i}
j	index of sample
k	index of function in shift-invariant model
l	index of function shift and with slight abuse of notation the amount of shift associated with this index
m	index of sample in the observation vector $\mathbf{x}$
n	index of column in matrix $\mathbf{A}$
p	$p = m + l$
$p(\cdot)$	probability density function of a variable
r	index of iteration in iterative algorithms and index in bridged transition sampler
$\mathbf{s}$	source vector or coefficient vector of length N
$\mathbf{s}^*$	fixed point of an iterative algorithm
u	binary indicator variable of mixture prior
w	weight in importance sampling algorithm
$\mathbf{x}$	observation vector of length M
$\mathbf{A}$	mixing matrix or dictionary

F	length of functions $\mathbf{a}_k$
H	Hessian
I	number of all observations
$\mathbf{I}$	identity matrix
$\mathcal{I}$	set of indices for all observations
J	number of samples in Monte Carlo approximation
K	number of all features
$\mathcal{K}$	set of indices for all features
L	number of all shifts
$\mathcal{L}$	set of indices for all shifts
L_1	L_1 norm, $\sum \lvert \cdot \rvert$
L_0	L_0 norm, number of nonzero values or numerosity
L_p	L_p norm, $(\sum \lvert \cdot \rvert^p)^{1/p}$
M	length of observation vector $\mathbf{x}$
N	length of coefficient vector $\mathbf{s}$
$N_{\emptyset}$	number of non-zero $\mathbf{s}$
$\mathcal{N}$	normal distribution
R_B	maximum number of annealing steps
T	temperature in annealing
$\mathbf{W}$	weighting matrix in IRLS
W	maximum number of elements in subset
Z	normalising constant
α	parameter in α-stable distribution
$\frac{\partial \cdot}{\partial \mathbf{A}}$	short for $\{\frac{\partial \cdot}{\partial \mathbf{a}_{mn}}\}_{mn}$, i.e. matrix of derivatives w.r.t. a_{mn}
ϵ	vector of observation noise
$\lambda = \frac{\lambda_c}{\lambda_\epsilon}$	parameter relating sparseness to reconstruction error
λ_c	function of prior variance
λ_u	parameter of hyper-prior in mixture model
λ_ϵ	inverse of error variance σ_ϵ^2
λ_{max}	maximum λ value

λ_G	scale parameter of Gaussian density in mixture prior
λ_R	scale parameter of Rayleigh density in mixture prior
σ_ϵ^2	variance of a scalar random variable or of a vector random variable with covariance matrix $\sigma_\epsilon^2\mathbf{I}$
η	location parameter in Rayleigh density
θ	vector of model parameters
μ	mean of Gaussian density
ν	learning rate
τ	prior variance
ψ	$\lambda_c + \lambda_R$
$\mathbf{\Sigma}_\epsilon$	$\sigma_\epsilon^2\mathbf{I}$
Δ	gradient
Σ_s	covariance matrix for Gaussian prior
$\emptyset$	index of non-zero coefficients
$\|\cdot\|$	norm (L_2 norm if not specified otherwise)
$<\cdot>$	expectation, the density, if not specified, should be clear from the context
$\sim$	is distributed as
$\hat{\cdot}$	approximated quantity
$\mathrm{diag}(\cdot)$	square matrix with the elements $\cdot$ along the diagonal

Abreviations

BP	basis pursuit
BSS	blind source separation
dB	deci Bell
EM	expectation maximisation
FFT	fast Fourier transform
fn	false negative
FOCUSS	focal underdetermined system solver
fp	false positive
ICA	independent component analysis
i.i.d	independent and identically distributed
IRLS	iterative re-weighted least squares
MA	moving average
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo
MIDI	musical instrument digital inteface
ML	maximum likelihood
MP	matching pursuit
OMP	orthogonal matching pursuit
PCA	principal component analysis
ROC	receiver operation curve
RVM	relevance vector machine
SAR	signal to artefact ratio
SDR	signal to distortion ratio
SIR	signal to interference ratio
w.r.t.	with respect to

"[...] we may consider the idea of building an induction machine. Placed in a simplified 'world' (for example, one of sequences of coloured counters) such a machine may through repetition 'learn', or even 'formulate' laws of succession which hold in its 'world'. [...]

In constructing an induction machine we, the architects of the machine, must decide a priori what constitutes its 'world'; what things are to be taken as similar or equal; and what kind of 'laws' we wish the machine to be able to 'discover' in its 'world'. In other words we build into the machine a framework determining what is relevant or interesting in its 'world': the machine will have its 'inborn' selection principles. The problem of similarity will have been solved for it by its makers who thus have interpreted the 'world' for the machine."
–K. R. Popper, *Conjectures and Refutations*

Chapter 1

Introduction

In this thesis we study an 'induction machine' to use the term offered by Popper in the quote above. How can such an 'induction machine' 'discover' and 'learn' 'laws' of nature, how can it find structure in its 'world' and what 'constraints' do we, the designer of such a machine, have to impose? First, the 'world' of such a machine needs to be specified. In this thesis we do not deal with the coloured counters of Popper's example but instead use music recordings. These signals have a great amount of structure and are, in fact, designed to contain such structures. However, this structure also shows unpredictable variation, for example the waves of sound pressure produced by a particular performance vary depending on many quantities of the instrument, the acoustics of the room, the temperature and the performer. Higher level structures such as timing and the acoustic energy of notes also vary between different performances. Our 'induction machine' must account for this variability which is done using probabilistic components and, in particular, Bayesian techniques.

Extracting information and structure from data seems a trivial exercise at first. Is it not easy for us to hear and understand spoken language even in noisy environments and is it not an effortless task to distinguish the face of your grandmother from most other faces? Only once we start to think about the computational processes required to achieve such tasks does it become clear how difficult these tasks are and what an extraordinary undertaking the human brain performs.

In this thesis we look at one small aspect of extracting and in fact discovering structure from data and investigate one possible approach based on Bayesian statistics and information theory. The data analysed is musical audio and we would like to address the following problems: how can we, with only minor prior knowledge of the actual structure of the data, extract information from musical signals such as individual notes, musical score, or distinguish and separate different sources in a mixture?

To progress in the direction of a solution to these questions we have to develop and study novel computational models. Bayesian theory promises to offer the best possible solution if we are able to model our data well and to specify the correct distributions. But even then, the computations required can only seldom be solved analytically and we are at best left with a sufficient approximation.

1.1 Thesis Outline

This thesis is divided into three main parts, part I, which introduces the sparse coding formulation and its extension to the shift-invariant model, part II, in which three computational strategies are discussed to solve the shift-invariant sparse coding problem and part III, in which experimental studies are presented.

The first part starts with a literature survey that introduces the sparse coding formulation and its relation to other signal processing methods. This is done in chapter 2, which starts with a more general discussion on the linear over-complete model before looking at sparse signal approximations and representations. This is followed by the introduction of adaptive sparse representations and the sparse coding model. This chapter finishes with a short overview of previous applications of the sparse coding method in areas such as image and audio analysis, blind source separation and biomedical signal processing. The second main chapter in part I is chapter 3, in which the shift-invariant sparse coding model is introduced and the learning rules for this model derived. This chapter concludes with an analysis of the advantages offered by the shift-invariant model and a discussion of the effects on the learned features sampling has if the original features can occur at arbitrary and continuous shifts.

Part II contains three chapters in which we introduce and study three different approximations to the sparse-coding learning rule. In chapter 4 we study analytic approximations and discuss in some detail an algorithm to find the MAP estimation required in these approximations. Furthermore, we discuss the use of a Gauss Seidel implementation of this algorithm. A subset selection step is introduced that allows the use of the method for the problems in music analysis studied in this thesis. Finally, the chapter concludes with a discussion of issues relating to the implementation of the method. In chapter 5 we introduce an importance sampling method for sparse coding. This method is based on a mixture prior, which is a mixture of a Gaussian and a delta function at zero. This method is faster than other approaches, however, bias is introduced in the learning rule. The third approach to sparse coding is developed in chapter 6 and is based on Gibbs sampling Monte Carlo approximations. In this chapter we introduce a novel mixture prior formulation. Here, a delta function is used together with a modified version of the Rayleigh distribution. This prior forces coefficients to be non-negative and is shown to be a conjugate prior for the Gaussian mean. A random subset selection procedure is introduced that guarantees the asymptotic convergence of the method.

The last part of this thesis, part III, presents experimental studies and applications of the shift-invariant sparse coding method. Chapter 7 presents a detailed analysis and comparison of the three algorithms proposed in part II. Chapter 8 presents an account of different applications of shift-invariant sparse coding for music analysis. This chapter begins with an analysis of piano music and investigates, whether or not piano music can be represented as a linear combination of atomic, note-like features. This analysis is followed by the application of shift-invariant sparse coding to such a piano signal and we show the emergence of note- and score-like structures.

In chapter 9 a comparison between the shift-invariant sparse coding method and a phase blind spectral method is presented. This chapter looks in detail at the features and representations found with these different methods when analysing piano music. The shift-invariant sparse coding formulation is computationally more demanding, however, the representation found offers a sample accurate timing of the features, which is not found with the phase blind method. Furthermore, the phase blind method was found to find a number of atoms that could not be assigned to a single note, but which instead represented chord like structures. The shift-invariant sparse coding model was found to be less prone to this.

Chapter 10 studies the application of shift-invariant sparse coding to single channel blind source separation. In order to assign the different features to each of the sources an unsupervised clustering algorithm is proposed. The blind source separation performance based on this unsupervised clustering is compared to the performance based on clustering that utilised prior information. It is found that for the example studied, the unsupervised clustering method performs nearly as good as the method that uses prior source information.

1.2 Original Contributions

The main contributions presented in this thesis are discussed in parts II and III. The following list gives a short overview of the main four points.

- Subset selection

  Sparse coding and its extension to shift-invariant sparse coding studied in this thesis involves computationally extensive procedures. For the application of these methods to most real world data such as music, efficient approximations have to be employed. The main contribution in this respect is the introduction of a subset selection step in the sparse coding formulation. In this thesis we show that such a method is not only feasible, but also does not degrade the results significantly, so that the method can be used for the applications studied in part III.

- Importance sampling for shift-invariant sparse coding

  Driven by the need for efficient sparse coding formulations, we introduce and study an importance sampling approximation to the sparse coding learning rule. This method offers a fast computational method, however, for the problems of music signal analysis, the bias introduced with this method becomes significant.

- Gibbs sampling with a novel positive prior for shift-invariant sparse coding

  An unbiased estimate of the sparse coding learning rule can be developed using Gibbs sampling methods. Our main contribution here is the introduction of a novel conjugate prior formulation that enforces positivity. We propose the use of a random subset selection procedure that conserves the convergence properties of the Gibbs sampler and which offers good results in practice.

- Application to music

  Shift-invariant sparse coding methods have previously only been applied to video and image data. In this thesis different applications to audio are studied. In particular it can be shown that shift-invariant sparse coding leads to representations of a piano recording in terms of note- and score-like structures. To our knowledge, the work presented here is the first application of shift-invariant sparse coding to discover meaningful features that have a correspondence to real world objects from real world time-series. Furthermore, the applicability of the method to blind source separation is studied. In order to apply shift-invariant sparse coding to the blind source separation problem, we have developed a novel unsupervised clustering algorithm that assigns the features found to each of the sources.

The work on shift-invariant sparse coding presented in chapter 3 and the derivation of the learning rule in chapter 4 were developed during the early research which led to this thesis. However, it was found at a later stage that this model and the learning rule had previously been published independently. As some of these publications predate the publications of the author listed in the next section, these developments are not included in this section as novel contributions. The literature mentioned is referred to in the chapters in which the theory is developed.

1.3 Publications

Work presented in this thesis has previously been published in the following journal and conference papers.

- Journal Papers

  1. Thomas Blumensath and Mike Davies, "Sparse and shift-invariant representations of music," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 50-57, 2006.

     Parts of this paper have found their way into this thesis and contribute to the background in chapters 2 and 3. Some of the results in chapter 8 and the whole of chapter 10 have also been taken from this publication.

  2. Mark Plumbley, Samer Abdallah, Thomas Blumensath and Mike Davies, "Sparse representations of polyphonic music," to appear in *ELSEVIR Signal Processing*

     Material from this paper is here presented in chapter 9. This paper was a joint paper and the results presented for the phase blind methods have been supplied by the second author of the paper.

- Conference Papers

  1. Thomas Blumensath and Mike Davies, "Enforcing sparsity, shift-invariance and positivity in a Bayesian model of polyphonic music," in *Proc. IEEE Workshop in Statistical Signal Processing*, July, 2005

     This paper presents the work that can be found in chapter 6 of this thesis.

  2. Thomas Blumensath and Mike Davies, "A fast importance sampling algorithm for unsupervised learning of over-complete dictionaries," in *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, March, 2005

The work presented in chapter 5 has mainly been taken from this paper.

3. Thomas Blumensath and Mike Davies, "On shift-invariant sparse coding," in *Proc. Int. Conf. on Independent Component Analysis and Blind Source Separation*, September, 2004

   This paper discusses the issues which can be found at the end of chapter 3 in this thesis.

4. Thomas Blumensath and Mike Davies, "Unsupervised learning of sparse and shift-invariant decompositions of polyphonic music," in *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, May, 2004

   This early paper presents much of the work in chapter 4.

The work presented in chapter 7 and appendix B is currently unpublished.

Part I

Shift-Invariant Sparse Coding

Chapter 2

Sparse Coding

One of the fundamental tasks in signal processing is to find a representation of a signal in which the structures, patterns and dependencies of that signal are made more explicit. One approach is to find a representation that models a signal in terms of meaningful features or real-world objects. This can be done by assigning a small number of elementary objects from a large set of known components or objects to a certain observation. The number of components or objects assigned to each observation can be expected to be much smaller than the dimension of the observation itself, while the set of all known components or objects might be much larger. The problem is that neither do we know the set of components or objects a priori, nor do we have a simple 'linear' way to find the smallest number of these components or objects to explain an observation.

This problem is formalised in this chapter under the term of sparse coding. The concept of sparse coding is the main theoretic formalism used in this thesis and is therefore discussed in detail here. We start this chapter by introducing the linear generative model that is the basis for many signal processing applications and that forms the basis for sparse coding. Depending on the conditions on the model and the unknown parameters, different solutions have been presented in the literature, some of which are reviewed here.

The main concept required for this thesis is sparsity, which is defined in section 2.2. As the definition of sparsity introduced here leads to an NP hard optimisation problem, we also review several approximations to the sparsity measure and survey computational strategies that minimise these approximations.

Instead or in addition to sparsity, additional constraints can be used to solve over-complete linear models and we discuss the positivity constraint, which forms the basis of our work in chapter 6.

Section 2.3 presents a review of strategies used to find sparse representations and approximations in the case in which the other model parameters are known, while the fourth section in this chapter deals with the problem of adapting the linear model in order to find "optimal" (for a particular measure) sparse representations for a given class of signals. In this section we introduce a maximum likelihood formulation, which is the basis for all algorithms studied in this thesis. Analytic solutions to this maximum likelihood method are, however, not available and different approximations suggested in the literature are reviewed.

The current chapter concludes with a survey of different applications of adaptive sparse representations that had an influence on the research presented in this thesis. We discuss previous work on image and audio analysis and also give an interesting application to biomedical data analysis. In these examples we encounter one of the main problems addressed in this thesis: The problem of analysing time-series or images by blocking.

2.1 The Sparse Coding Model

This thesis studies methods that can discover features from music recordings *unsupervised*. This means that the salient structure in music is discovered from musical observations alone, without using training examples in which the structures are labelled a priori. However, such an approach needs the specification of a model that allows the emergence of structure and features. The approach taken here is based on a linear generative model that describes the observation as a linear combination of features. The model can be written algebraically as:

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \epsilon = \sum_k \mathbf{a}_k s_k + \epsilon. \tag{2.1}$$

The vector $\mathbf{x} \in \mathbb{R}^M$ is a block of data taken from the signal under study, which we want to explain or analyse. This observation block $\mathbf{x}$ is modelled as a linear combination of feature vectors $\mathbf{a}_k$, which we also call atoms, and which are the columns of the matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$. The mixing matrix $\mathbf{A}$ is also called dictionary. The vector $\mathbf{s} \in \mathbb{R}^N$ defines the multiplicative weights of these features. Therefore, each element s_k of $\mathbf{s}$ is associated with a single feature $\mathbf{a}_k$. The vector ϵ represents observation noise or, more generally, the error due to the inability of the model to describe the signal exactly.

The above linear model is used in a variety of signal processing applications with different constraints and dimensions. If we had training examples for which the vectors $\mathbf{x}$ as well as the vectors $\mathbf{s}$ are observed, this model would become the standard regression problem [93], while for a known square orthogonal matrix $\mathbf{A}$ and no observation noise, the problem of finding the coefficients $\mathbf{s}$ for any observation $\mathbf{x}$ is the one dealt with in standard orthogonal transforms [86]. In the case where $\mathbf{A}$ and $\mathbf{s}$ are unknown and $M \geq N$, both $\mathbf{A}$ and $\mathbf{s}$ can be chosen such that the elements in the vector $\mathbf{s}$ become uncorrelated. This is the well known method of *Principal Component Analysis* (PCA) [93]. Solving the same problem under the assumption of independence of the elements in the vector $\mathbf{s}$ leads to the standard problem of *Independent Component Analysis* (ICA) [58].

If $N > M$ we are faced with two problems. Apart from finding the matrix $\mathbf{A}$ as is the central problem in PCA and ICA, we also need to find a method to compute $\mathbf{s}$ for any estimate of $\mathbf{A}$. As there is no unique solution to this problem in general, additional conditions on the solution in form of regularisation terms or inequality constraints have to be used. The main condition used throughout this thesis are measures of sparsity to be defined later. In this thesis we use the term sparse coding to refer to the linear generative model in equation (2.1) in which both $\mathbf{A}$ and $\mathbf{s}$ are unknown and in which a regularisation term is used to measure and enforce sparseness in order to find optimal representations.

The problem of estimating $\mathbf{A}$ (i.e. of finding the ML estimate of $p(\mathbf{x}|\mathbf{A})$) requires as a sub-problem the estimation of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. Based on this estimate, an improved estimate of $\mathbf{A}$ can be found so that the developed approach alternates between estimation of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$ and $\mathbf{A}$. Once $\mathbf{A}$ has converged, an estimate of $\mathbf{s}$ can be found by finding the MAP or mean estimate of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$.

If the features $\mathbf{a}_k$ as well as their associated coefficients s_k are unknown, then the above model has an ambiguity in the norm of the features and the coefficients $\mathbf{s}$. A rescaling of the features together with an inverse scaling of the coefficients keeps the reconstruction unaltered. This ambiguity can be avoided by either restricting the features to unit length or by specifying the variance of the priors for either $\mathbf{s}$ or $\mathbf{A}$. In this thesis we keep the variance of the coefficients $\mathbf{s}$ adaptable and instead restrict the norm of the features to unit length.

2.2 Sparsity

Informally, the sparsity measure incorporates our belief, that an observation should be explained with as small a number of features as possible. A formal definition of sparsity as it is used in this thesis is given at the start of this section. This definition is followed by a discussion motivating the use of sparsity. As the definition of sparsity given here leads to an NP hard optimisation problem, we discuss a range of alternative sparsity measures, which can be used to approximate the measure of interest.

The algorithms developed in this thesis are based on Bayesian theory. In order to deal with sparsity in a Bayesian framework, we introduce different prior distributions for $\mathbf{s}$ that are equivalent to the sparsity measure. The Bayesian formulation further allows for an introduction of additional conditions on the solution. Positivity is such a constraint and its use for feature extraction is discussed.

2.2.1 Definition, Motivation and Measures for Sparsity

Definition of Sparse Approximation/Representation

Definition 2.1. *In the context of this thesis we use the term* sparse approximation/representation *to refer to an approximation/representation* $\mathbf{s}$ *with fewer non-zero coefficients than the dimension* M *of* $\mathbf{x}$.

Formally, we distinguish between sparse approximations and sparse representations. A representation exactly describes a signal while an approximation has some non-zero error term. Sparse representations as defined here, i.e. representations with fewer non-zero coefficients than the signal dimension are only possible for a small number of signals for any dictionary $\mathbf{A}$. In this thesis we only deal with sparse approximations and the term representation is used on occasions to refer to the approximation coefficients $\mathbf{s}$.

Another often used definition of sparsity is one in which the coefficients are small with high probability. This idea is related to the different measures of sparsity discussed below. In Bayesian methods, this definition is common as these measures often correspond to probability densities that have most of their probability mass concentrated around zero, but not concentrated at zero. However, this definition of sparsity is less well defined and is not used here, instead the measures associated with these definitions of sparsity should be thought of as approximations to the sparsity measure as defined above.

Motivations for Sparsity

The main problem studied in this thesis is the extraction of features and structures from musical signals without the explicit modelling of expected musical relationships. The number of features and objects in these signals, such as notes, melodies, rhythms and chords, is small compared to the samples in the signals. For example, there are only a small number of notes played during any short time interval of a musical performance. This is also true for other signals such as images in which objects such as faces are much fewer than the samples in the signal, as it would be otherwise impossible for us to distinguish the objects. For example, a short excerpt of a musical recording of say one second, has, at a sampling rate of 8 000 Hz, 8 000 samples, however, if in this excerpt more than 8 000 different notes would be played, we would certainly not be able to distinguish between them. Therefore, it seems natural to look for approximations of a signal that only use a small number of objects to model the observation. If we assume the atoms $\mathbf{a}_k$ to represent objects in the signal, we would like to find a representation of $\mathbf{x}$ with as small a number of these atoms as possible.

Another motivation for sparse signal descriptions, which has led to the original work in this area, is the relationship between sparse representations and redundancy reduction. The idea of redundancy reduction has been advocated by Barlow in [3] as a fundamental principle underlying the primary processing in mammalian neural circuits for perception and has since been used in the neural coding literature by Fóldiak [37, 39] who linked the concept of redundancy reduction to sparse coding and by Harpur [53]. In these references sparse coding methods are used to find efficient codes. However, a sparse representation does not necessarily lead to an efficient coding strategy, as for very over-complete representations much of the coding capacity has to be spent on specifying which coefficients are non-zero. As stated by Fóldiak in [39], sparse coding relates to redundancy reduction only for code words of fixed length as found in neural circuitry in which the number of neurons used to code a signal is fixed. Fóldiak argues that sparse representations have, in fact, a lower capacity than distributed codes and that loss of information is only avoided by a reduction in redundancy in the sparse representation.

Measures of Sparsity

In general, the problem of sparse signal approximation can be solved by finding a solution to the optimisation problem:

$$\mathbf{s}_{sparse} = \arg\min_{\mathbf{s}} g(\mathbf{x}, \mathbf{A}, \mathbf{s}) + \lambda f(\mathbf{s}), \tag{2.2}$$

where $g(\cdot)$ is a term measuring the reconstruction accuracy or approximation error and $f(\cdot)$ is a measure of sparsity used as a regularisation term. Throughout this thesis, the approximation error measure is assumed to be the L_2 norm.

A measure of the sparsity of a representation as given in the definition at the beginning of this section would be the *numerosity* or L_0 norm. This norm is easy to compute, however, the problem of finding the minimum of equation (2.2) with this norm is NP-hard [19]. Different approximations to this norm have therefore been discussed [109, 71, 111, 70]. These papers argue that, when using a gradient optimisation method, the measure has to be concave in order to force coefficients to zero. A limiting case of these concave measures is the L_1 norm. The 'quasi norms' L_p for $0 < p < 1$ have also been proposed[1]. Other concave functions are possible.

From a Bayesian point of view, regularisation terms in the form of sparsity measures can be expressed as prior distributions. The optimisation problem in equation (2.2) can then be interpreted as the negative log posterior of $p(\mathbf{x}, \mathbf{s}|\mathbf{A})$. The L_2 reconstruction error term corresponds in this case to a Gaussian likelihood. If the sparsity measure is one of the L_p 'quasi norms' with $0 < p < 1$ or the L_1 norm, this leads to a generalised Gaussian prior on the coefficients $\mathbf{s}$. Probably the most commonly used sparse prior is the Laplacian (a generalised Gaussian with $p = 1$, i.e. the prior form of a L_1 norm) which leads to convex optimisation problems as discussed below.

2.2.2 Probabilistic Formulation

The sparse coding model introduced in this chapter is based on the assumption of sparsity of the coefficients $\mathbf{s}$. We have stated that this sparsity can be expressed using prior densities for the coefficients. In order to facilitate Bayesian analysis and for further development of the algorithm, we have to specify the exact form of this prior distribution, together with the distributions of other parameters of interest.

Throughout this thesis, the error term is assumed to be i.i.d Gaussian. This simplifies the used notation and computation, however, most algorithms can be extended to the case of non-white Gaussian noise. The noise variance is denoted by σ_ϵ^2 and its inverse by λ_ϵ.

In this thesis we use a number of different priors for $\mathbf{s}$ that can always be expressed as factorial priors of the form $p(\mathbf{s}|\theta) = \prod p(s|\theta)$. The used factorial form of the prior assumes that the representation coefficients are independent a priori. This independence links the sparse-coding model to the noisy and over-complete ICA model and is reasonable for the problems

[1]It is important to stress that L_p for $0 < p < 1$ are not norms as they do not satisfy the triangle inequality.

studied here. However, dependence structure can be incorporated into the prior if such information were available for a certain problem. This added complexity would then be reflected in more complex algorithms.

The EM algorithm discussed in chapter 4 uses an improper prior of the form $p(s) = s^{-2}$. This prior is justified in section 4.2 where it is shown that it is the marginalisation of the form $p(\mathbf{s}|\tau) = \prod p(s|\tau)$ with $p(s|\tau)$ being zero-mean normal distributions with variance τ having a Jeffrey's hyper-prior. The marginal distribution $p(s)$ can take on a variety of other distributions such as the generalised Gaussians if $\frac{1}{\tau}$ has a distribution proportional to $\tau^{0.5}p_\alpha(\alpha/2)$ where $p_\alpha(\alpha/2)$ is a symmetric alpha-stable distribution of $\frac{1}{\tau}$. The parameter α specifies the exponent in the generalised Gaussian. See [149] and [145] for more details.

In chapters 5 and 6 we use mixture priors $p(s) = u * p(s|u = 1) + (1 - u)p(s|u = 0)$ where u is a binary indicator variable with distribution $p(u) = \frac{1}{Z}e^{-\frac{\lambda_u}{2}u^2}$. To force coefficients to be exactly zero we use a delta function for $p(s|u = 0)$. The non-zero coefficients can then have different distributions. In chapter 5 we let $p(s|u = 1)$ be a zero mean Gaussian distribution, while in chapter 6 $p(s|u = 1)$ is a modified Rayleigh distribution, which is introduced in that chapter.

2.2.3 Additional Constraints

In order to find solutions to a particular problem it is of advantage to use all possible information available to guide the algorithm. Such prior information can be incorporated to further constrain the solutions of the algorithm. One powerful but still quite general constraint is the non-negativity constraint. In addition, we can for certain problems assume the features themselves to be strictly non-negative. This constraint has been proposed to extract salient features using the Non-Negative Matrix Factorisation algorithm of [74, 73]. For most signals, however, the constraint of non-negativity alone does not seem able to extract meaningful features. This led to the development of methods enforcing both, sparsity and positivity [55, 108, 7].

From a Bayesian point of view, both sparsity and non-negativity, are prior beliefs and can be incorporated into prior distributions as in [108]. This approach is taken in chapter 6 in which we introduce a non-negative sparseness enforcing prior.

2.3 Algorithms for Sparse Approximation/Representation

In this section we assume that $\mathbf{A}$ is known and that $N > M$ and concentrate on methods to compute sparse approximations $\mathbf{s}$ for given observations $\mathbf{x}$. Without the use of additional regularisation terms such as sparsity measures, a linear estimate of $\mathbf{s}$ can be calculated as $\mathbf{s} = \mathbf{W}\mathbf{x}$ where $\mathbf{W}$ is a general left inverse or the pseudo-inverse of $\mathbf{A}$. If $N \leq M$ such as in standard ICA and PCA, there is a unique minimum mean square error solution, however, if $N > M$ there is no unique solution in general. In fact, there is an infinite number of left inverses $\mathbf{W}$ [86] that map the space of observations $\mathbf{x}$ onto the space of coefficients $\mathbf{s}$. However, there is a unique pseudo-inverse. This is often used in frame theory to find a solution to the over-complete signal model [86]. However, this solution spreads the energy of the observation $\mathbf{x}$ over all coefficients s_k and does therefore not conform with our belief that an observation can be explained by a combination of a small selection of features.

Instead of using the pseudo-inverse we can impose additional regularisation terms on the solution of a linear, over-complete system. In particular we use the regularisation term that enforce sparsity as defined above. The problem of finding approximations or representations $\mathbf{s}$ with a small number of non-zero coefficients in the linear over-complete model with a known dictionary $\mathbf{A}$ has been studied in different areas of mathematics, statistics and signal processing. In the signal representation and approximation literature the problem of sparse signal representations with over-complete dictionaries has been studied as discussed in [86]. In the regression literature, a similar problem is that of *subset selection* [91]. This problem deals with the selection of a small number of regression variables from a larger set of possible regressors.

Several methods have been proposed for the problem in which $\mathbf{A}$ has known structure. If $\mathbf{A}$ is the union of orthogonal bases, it is possible to select the basis that leads to a minimum norm representation for some norm [17].

The solutions suggested in the literature can be roughly partitioned into search methods, greedy methods, Bayesian methods and optimisation methods. In the following, a short overview covering the most commonly used approaches is given.

2.3.1 Search Methods

If the matrix $\mathbf{A}$ is unconstrained, there is only one known method to find an approximation with the smallest number of non-zero coefficients. This method is exhaustive search, which is NP hard [19]. This method has been used in the subset-selection literature [91] for problems in which $N < 25$ but is impractical for larger problems. Random search methods have also been proposed for subset-selection as in [133]. These methods have so-far only been applied to problems of moderate size and are still too slow for large scale problems. However, the sampling method discussed in chapter 6 can be seen as a stochastic search and can offer good performance as is shown in this thesis, but these methods are not guaranteed to find the optimal solution.

2.3.2 Greedy Methods

One of the fastest and most widely used families of algorithms is the family of greedy methods. The Matching Pursuit (MP) algorithm is an iterative method that selects at each iteration the atom $\mathbf{a}_k$ closest (under some metric) to the residual reconstruction error [87, 60]. The error is then projected onto the direction of this atom to calculate s_k, after which the residual is reduced by $\mathbf{a}_k s_k$. An extension of this method, called Orthogonal Matching Pursuit (OMP) (also known as forward selection [91]), has been proposed [86], which, after each selection of an atom, updates the residual by calculating the difference between $\mathbf{x}$ and the projection of $\mathbf{x}$ onto the set of all previously selected atoms.

Instead of adding an atom to the set of atoms representing the signal at each iteration, a backwards selection is possible. In this method the atom is removed from the set of atoms that leads to the smallest increase in reconstruction error in each iteration. Hybrid methods of these strategies and incorporation of random selection is also possible [91, 96]. Prior information has been incorporated into these greedy algorithms by weighting the atoms, giving larger weights to atoms that are more likely [26]. These methods do not have guaranteed performance in general (certain exceptions are mentioned below) and it was found that these methods are not applicable to the learning task studied here.

2.3.3 Bayesian Methods

We have shown that the different sparsity measures do have an interpretation as prior probabilities. This allows us to deal with the problem using Bayesian theory. From a Bayesian point of view we are interested in either the mean or maximum of the posterior $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. The problem of learning the matrix $\mathbf{A}$ in the next section also requires the evaluation of expectations with respect to this probability. For the sparsity measures discussed above, this posterior does not have an analytic representation and expectations with respect to this distribution cannot be evaluated exactly and no closed form solution for the mean is available. Different Gibbs sampling methods have been proposed in [43, 105, 126] (See also [132, 25, 88, 52, 2] for similar approaches) that can be used to estimate the mean or maximum of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$.

In order to enforce sparsity as defined above we would like to exactly set coefficients to zero. This can be achieved by a mixture prior of a Gaussian and a delta mass at zero as proposed in [88, 153, 126]. This model is discussed further in chapter 5, while in 6 we develop a similar model based on a mixture of a non-negative and delta distribution. In [45] a similar method is proposed in the context of model selection and Reversible Jump Markov chain Monte Carlo methods [47]. The problem of sparse signal approximations therefore has strong links to problems in model order selection.

Instead of finding the mean of the posterior $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$ we can also find a local maximum of the distribution. Annealing methods could be used to find these maxima when using Markov chain sampler. Another approach based on sampling methods, which are a form of stochastic search, is to select the sample for which the posterior is highest. This approach can be used in the sampling strategies used in this thesis to select the non-zero coefficients. Based on this selection, the MAP estimation of $\mathbf{s}$ can be evaluated analytically. Another method would be to use maximisation methods as discussed in subsection 2.2.1. For this optimisation we can use the methods discussed in the next paragraph. These optimisation methods use a scale mixture of Gaussians as a prior on $\mathbf{s}$. This model is discussed further in chapter 4.

2.3.4 Optimisation Methods

If the L_1 norm is used (or equivalently if we use a Laplacian prior), linear programming techniques can be used to solve the optimisation problem. This leads to the method of Basis Pursuit (BP) [15]. The advantage of using the L_1 norm is that the optimisation problem has a unique optimum (apart from pathological cases). However, whether this optimum coincides with a solution of the optimisation problem based on the L_0 norm cannot be assumed in general. However, for dictionaries with certain structures and certain signals this is true as discussed below.

For general sparsity enforcing norms, other optimisation methods have been proposed, the simplest of which are gradient descent type algorithms. More involved optimisation algorithms have been developed such as the Focal Underdetermined System Solver (FOCUSS) proposed in [110, 112, 69, 95] and [30]. This algorithm is a flavour of the Iterative Re-Weighted Least Squares algorithm by Nelder [101] (see also [93]) which was used for sparse signal representations in [21]. The work by Figueiredo [35, 33, 34, 36] shows the equivalence of this method to a family of EM algorithms. This method is used extensively in this thesis and is therefore discussed in detail in chapter 4.

A different approach to MAP estimation of the posterior $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$

is to use methods similar to Type 2 maximum likelihood estimates as discussed in [85]. These methods led to the development of the Relevance Vector Machine (RVM) [137, 29] and similar methods for sparse signal representations [150].

2.3.5 Uniqueness and L_0 Optimality of BP and OMP

There now exists a number of papers studying the conditions on the matrix $\mathbf{A}$ under which OMP and BP find unique solutions and under which conditions these solutions identify the elements of the optimal solution to the L_0 optimisation problem [140, 139, 138, 41, 42, 49]. The bounds derived depend on the distance between the atoms $\mathbf{a}_k$ and the number of atoms required to represent the signal. In [48] a practical test was proposed to determine whether any given sparse approximation identifies the non-zero elements of the optimal L_0 optimisation problem. Even though these results are promising, for many common dictionaries and signals the results do not apply. This is in particular true for the problem studied in this thesis, in which the necessary conditions on $\mathbf{A}$ cannot be guaranteed, so that these results are not discussed further.

2.4 Adaptive Sparse Approximations

For a given class of signals, the question arises how to choose or adapt the matrix $\mathbf{A}$ such that we can find an optimal sparse approximation of signals from this particular class. A review of current methods that can be used to find such adaptive sparse approximations is presented in this section. Particular attention is given to maximum likelihood (ML) estimation of the matrix $\mathbf{A}$ and other model parameters excluding the coefficients $\mathbf{s}$. We concentrate here and in the rest of this thesis on maximum likelihood estimates and do not investigate possible maximum a posterior (MAP) estimates. For the problems studied here we justify this choice by the assumption that our prior knowledge of the parameters (in particular of $\mathbf{A}$) would lead to relatively flat priors. Under these conditions we assume that the MAP estimate and the ML estimate only differ slightly and an introduction of additional priors would unnecessarily complicate the development and distract from the main focus, which is the use of sparseness. The inclusion of priors for the parameters, if these are assumed to be independent from the prior on **s**, is straightforward and only requires the addition of the gradient of the log of this prior to the learning rules developed in this thesis.

2.4.1 Possible Strategies

Different methods have been proposed to adapt the matrix **A** and a good overview is presented in [68].

The first adaptive sparse approximations have been studied in the Neural Network literature. Inspired by Barlow's [3] idea of redundancy reduction, Fóldiak [37, 39] and Harpur [53] developed artificial Neural Networks able to find sparse approximations of input signals. This work led to the work on sparse coding in [103, 104] and [80, 78], which is further discussed below.

Another approach was taken recently in [8] based on the assumption that vectors in high dimensional spaces are likely to be 'nearly' orthogonal. However, enforcing near orthogonality might not necessarily lead to the emergence of salient features. Other approaches based on geometric considerations are those in [134, 144, 82] and [135] whilst a method based on histograms can be found in [136]. These methods exploit the assumption that observations **x** are clustered around the directions specified by the features $\mathbf{a}_k$ when the coefficients **s** are very sparse. However, for very over-complete models and for high dimensional observation spaces, these assumptions are not met in general and the applicability of these methods to such problems is questionable.

The work in [22] is based on the same assumptions but uses a Gaussian mixture data model to describe the sparse prior distribution of **s**. A simplified inference algorithm is then developed making the assumption that each observation coefficient is due to a single source. This leads to a model in which the data follows a mixture of Gaussian distribution, where each source is directly related to a Gaussian in the data distribution.

2.4.2 ML Estimation of the Model Parameters

The problem of learning the matrix $\mathbf{A}$ can be formulated from a probabilistic point of view as the problem of finding the maximum likelihood estimate of the marginal likelihood [80]:

$$p(\{\mathbf{x}\}|\mathbf{A}) = \int p(\{\mathbf{x}\}|\mathbf{A}, \mathbf{s})p(\mathbf{s})\ d\mathbf{s},$$

where we use $\{\mathbf{x}\}$ to denote the set of all available data vectors $\mathbf{x}$.

Unfortunately, for the sparseness inducing priors discussed above, this integral cannot be solved analytically and approximations are required. If we assume the observations $\mathbf{x}$ to be independent we can use the factorisation $p(\{\mathbf{x}\}|\mathbf{A}) = \prod p(\mathbf{x}|\mathbf{A})$ where the product is over all observations. Instead of maximising this joint distribution, it is possible to use stochastic gradient descent optimisation. This procedure has the advantage that not all data needs to be taken into account in each step, reducing the memory demands of the algorithm. Furthermore, it is then possible to update model parameters 'on-line' as new data becomes available. Furthermore, for the maximisation studied here, the gradient, whether with respect to all data or with respect to a single observation, is not available analytically. The approximations introduced below can only offer noisy estimates and naturally lead to stochastic gradients.

In the stochastic gradient descent procedure used here, the matrix $\mathbf{A}$ is updated iteratively using a single data-point in each iteration to calculate an approximation of the gradient. If the gradient with respect to a single data point is unbiased, then this method converges to a local maximum of the likelihood [72].

In order to derive a stochastic gradient learning rule and in order to gain a better understanding of the problem we rewrite the required gradient by following [78] and use the notation:

$$\mathcal{Z} = p(\mathbf{x}|\mathbf{A}) = \int p(\mathbf{x}|\mathbf{A}, \mathbf{s})p(\mathbf{s})ds$$

and the abbreviation:

$$\mathcal{E}(\mathbf{s}) = \log p(\mathbf{x}|\mathbf{A}, \mathbf{s}) + \log p(\mathbf{s}). \tag{2.3}$$

An expression for the gradient of the log-likelihood:

$$\mathcal{L} = \log p(\{\mathbf{x}\}|\mathbf{A})$$

can be found as:

$$\frac{\partial \log p(\{\mathbf{x}\}|\mathbf{A})}{\partial \mathbf{A}},$$

where the derivative is w.r.t. the individual elements of the matrix $\mathbf{A}$.

The learning algorithm is derived as a stochastic gradient algorithm for which in each iteration the gradient has to be evaluated for a single observation vector $\mathbf{x}$ and not for the set of all available observations $\{\mathbf{x}\}$. This gradient can be written as:

$$\begin{aligned}\frac{\partial \log \mathcal{Z}}{\partial \mathbf{A}} &= \frac{1}{p(\mathbf{x}|\mathbf{A})}\frac{\partial}{\partial \mathbf{A}}p(\mathbf{x}|\mathbf{A}) \\ &= \int \frac{1}{\mathcal{Z}}e^{\mathcal{E}(\mathbf{s})}\frac{\partial}{\partial \mathbf{A}}\mathcal{E}(\mathbf{s})\ ds \\ &= \int p(\mathbf{s}|\mathbf{A},\mathbf{x})\frac{\partial}{\partial \mathbf{A}}\mathcal{E}(\mathbf{s})ds \\ &= \left\langle \frac{\partial}{\partial \mathbf{A}}\mathcal{E}(\mathbf{s})\right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})} \qquad (2.4)\end{aligned}$$

where $< \cdot >$ denotes expectation.

So the gradient can be written as an expectation of the derivative of equation (2.3) with respect to $p(\mathbf{s}|\mathbf{A},\mathbf{x})$. Taking the derivative of equation (2.3) and assuming $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2\mathbf{I})$ the negative of the gradient can be written as:

$$-\frac{\partial \log \mathcal{Z}}{\partial \mathbf{A}} = \left\langle \sigma_\epsilon^2(\mathbf{x} - \mathbf{A}\mathbf{s})\mathbf{s}^T\right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})}, \qquad (2.5)$$

where the derivative is again with respect to the individual elements of the matrix $\mathbf{A}$.

2.4.3 Approximations to ML Learning

As the expectation w.r.t. $p(\mathbf{s}|\mathbf{A},\mathbf{x})$ cannot be evaluated analytically, different strategies have been proposed. In [72] different conditions on the estimation of the gradient w.r.t. a single data-point are given that ensure convergence to a local maximum. One important condition is the (asymptotic) unbiasedness of the gradient estimate. The first two methods discussed below do not take this bias into account. The Gibbs sampling method in chapter 6, however, does offer such an unbiased estimate (at least asymptotically). The importance sampling method developed in chapter 5 also address this problem and is also asymptotically unbiased, however, for finite samples, the bias can be significant.

Delta Approximation

The simplest approximation of the integral in equation (2.5) is to approximate the posterior $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$ with a delta function at its maximum as suggested in [103]. In [57] this approximation was shown to lead to the joint maximum likelihood estimation of $\mathbf{s}$ and $\mathbf{A}$ for the complete likelihood function in a missing data problem, in which the missing data is $\mathbf{s}$. In this case the gradient estimate becomes:

$$\frac{\partial \log \mathcal{Z}}{\partial \mathbf{A}} \approx \sigma_\epsilon^2 (\mathbf{x} - \mathbf{A}\hat{\mathbf{s}})\hat{\mathbf{s}}^T,$$

where we use $\hat{\mathbf{s}}$ to denote the MAP estimate of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. This method requires the estimation of $\hat{\mathbf{s}}$, which can be done using the methods discussed in the previous section.

Gaussian Approximation

Lewicki [80] proposed a Gaussian approximation of the posterior around the MAP estimate of $\mathbf{s}$ which leads to the approximation:

$$\frac{\partial \log \mathcal{Z}}{\partial \mathbf{A}} \approx \sigma_\epsilon^{-2}((\mathbf{x} - \mathbf{A}\mathbf{s}) - \mathbf{A}\mathbf{H}^{-1}),$$

where $\mathbf{H}$ is the Hessian of the log-posterior evaluated at the current MAP estimate of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. Further approximations can be made [80] leading to:

$$\frac{\partial \log \mathcal{Z}}{\partial \mathbf{A}} \approx -\mu \mathbf{A}(-\frac{\partial}{\partial \mathbf{s}} \log p(\mathbf{s})\hat{\mathbf{s}}^T + \mathbf{I}).$$

This method also requires the evaluation of $\hat{\mathbf{s}}$, which can again be done using methods introduced in the previous section.

Monte Carlo Approximation

Using sampling methods to sample from $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$ does not only allow us to estimate the mean or maximum of the posterior as discussed in section 2.3, it also allows us to use Monte Carlo approximations of the expectation in equation (2.5). This method was proposed in [126, 105]. This approximation is extensively used in chapters 5 where we develop an importance sampling method and in 6 where we study a Markov chain sampler. More details on previous methods based on Monte Carlo approximations are given in these chapters.

Other Approximations

For completeness we mention two other solutions suggested in the literature. One of these approaches is to approximate equation (2.5) with the help of variational methods (see for example [44, 59, 92, 116]). The other approach was proposed by Engan in [30]. This batch method (Method of Optimized Directions) is similar to the solution of the standard Wiener Filter [63], because once the vector $\mathbf{s}$ or its correlations with $\mathbf{x}$ are known, or assumed to be known, the model reduces to the standard linear model with Gaussian noise.

2.5 Applications of Sparse Coding

There are two main areas for which sparse coding ideas have been used: feature extraction and Blind Source Separation (BSS). BSS based on sparse signal representations uses the realisation that most signals can be transformed with an orthogonal transform into a representation in which expected features occur sparsely. For example, the time domain representation of a spoken word is not sparse, however, the frequency domain representation has only a small number of significant coefficients.

Feature extraction based on sparse coding ideas uses the assumption that most features do not occur most of the time in any one observation. This is the assumption used in this thesis. A general overview of previous applications based on this approach to feature extraction can be found in [16]. Possible applications include audio, image, and biomedical data analysis. Previous contributions to these areas as well as applications to BSS are given below.

2.5.1 Sparse Image Representations

Analysis of image data was the main application area of the early papers by Olshausen and Lewicki [103, 104, 80, 78]. This work was motivated by neurological signal processing mechanisms and it was shown that sparse image representations share many similarities to the representations found in the primary visual pathways in the mammalian brain. This work showed the similarity between the features learned from images of natural scenes and the receptive fields of simple cells in the primary visual cortex V1. Further applications to images can be found in [94, 78, 106, 126]. In general, the features found from image data were localised in space, had a narrow frequency support and showed a clear orientation. However, in this work features occurred at fixed positions of the analysed data blocks and only a small number of features was learned (about twice the number of input block dimensions). We argue in section 3.3.1 that such representations are not able to learn reoccurring features in the signal and do in general lead to features with small space (or time) and frequency support.

2.5.2 Sparse Audio Representations

Audio signals were analysed in the time domain by Lewicki in [76]. In this work, the signal was analysed in blocks and only a relatively small number of features was learned. Three types of audio signals were used and the features learned from each of the signals compared. The used signals were animal vocalisation, music and natural sounds. The features found had different time support, with the features learned from music having the longest time support and the features learned from natural sounds having the shortest time support. The features were again localised in frequency. The number of features learned in these experiments was again only slightly larger than the dimension of the observation space, leading to the same problems as mentioned in the above section on image analysis.

Features, such as particular sounds in a mixture, can occur at arbitrary time locations. This is not reflected in the time domain sparse coding method. For this reason, sparse coding has been used on audio spectra in [1, 129, 66]. This representation is phase blind and is less affected by the arbitrary location of events in audio. Harmonic sounds lead to similar spectral features if they occur at shifted positions. This was used in [1] to learn note spectra. These spectral features could then be used to find a music representation which was similar to the score of the performance. This method assumes that spectral features add linearly. Furthermore, as the learned features are phase blind, Wiener filtering has to be used to reconstruct the original signal from the representation. This can lead to artefacts when using the method for source separation. In chapter 7 we compare such a phase blind spectral method to the shift-invariant method studied in this thesis.

2.5.3 Applications to Biomedical Data

Spare coding can be used for other application domains such as biomedical data analysis. One interesting example of this is the work in [18], which studies the activation of different muscles in the frog leg. The question posed is whether the complex movements of the frog leg can be modelled as a scaled mixture of a small number of activation patterns. In this work it was found that different combinations of a small number of features can explain the activation of the different muscles in the frog leg leading to a wide range of different leg movements. This problem not only used sparseness, but also used a positivity constraint as muscle activations are necessarily positive. Furthermore, muscle activations can occur at arbitrary time locations and the algorithm used a shift-invariant formulation similar to the one introduced in the next chapter.

2.5.4 Sparse Representations for Blind Source Separation

There is a large amount of publications studying the separation of a small number of sources from a smaller number of observations. In this work, the number of observations is typically larger than one. The solution to this problem is often based on the fact that the representation of the signal is sparse in some transform domain. A decomposition of the signal can then be found by constraining the individual sources to have a sparse distribution. The extensive work by Zibulevsky and co-workers [157, 156, 158, 84, 159, 155] is a good example of this approach. Further examples can be found in [75, 81].

A Bayesian view of the problem is taken by Rowe in his work [124, 125, 123] and by Févotte in [32] while the work in [22] proposes a Gaussian mixture model as mentioned previously to solve this problem.

This work requires that more than one observation of the mixture is available. In chapter 10 we study the use of the model proposed in chapter 3 for the problem of single channel source separation. Models for single channel source separation were previously introduced in [142] and [61, 62]. These models, however, incorporate prior knowledge of the sources in the form of source models. Shift-invariant spectral methods for single channel source separation have been studied in [143] and [130]. However, these papers assume a linear combination of features in the spectral domain and do not address the problem of clustering features into individual sources.

Conclusions

In this chapter we have introduced a linear generative model to describe observations. In order to discover salient structures, restrictive conditions are forced upon the representation. Sparsity has been shown to be a very powerful assumption in this context, but positivity, where applicable, can also produce good results.

We formulated the sparse coding model in a probabilistic framework. This allows us to specify sparseness measures as prior distributions. Maximum likelihood methods can then be used to adapt model parameters and in particular the set of features. This method leads to (at least local) optimal solutions. However, exact solutions are not possible and approximations have to be introduced.

From the section on applications, it is clear that time-series and images pose additional difficulties. Features can often occur at arbitrary locations.

The standard method of processing time-series in blocks leads to a model that has to learn features at all possible shifts. This not only increases the number of parameters to be adapted, it also requires the number of features to be large enough to cope with this repetition of features at different shifts. These effects and their influence on the extraction of features are studied in the next chapter, in which a shift-invariant sparse coding formulation is introduced. In this formulation, the model is explicitly constrained so that features can be used at arbitrary locations to describe the signal.

Chapter 3

Shift-Invariant Sparse Coding[1]

In the standard sparse coding formulation introduced in the previous chapter, the observations $\mathbf{x}$ are vectors. However, many signals of interest in engineering, such as audio signals, are time-series. In order to deal with these time-series, it is customary to partition the sequence into smaller blocks. These blocks can then be used as the observations $\mathbf{x}$ in the sparse coding model. However, one motivation for the use of the sparse coding model is to represent the observations as a linear combination of salient features. In time-series such as audio, it is not generally known a priori at which time-locations features occur. The features present in a particular observation block are then randomly shifted with respect to the beginning of the block. In order to model this uncertainty, the standard sparse coding model has to include several copies of each feature at all possible time-locations.

This structure can be learned from the observations themselves, which requires that the model includes enough free parameters so that the features can be learned at different locations. It is, however, of advantage to keep the number of free parameters low, which can be done by explicitly enforcing the shift-invariant structure in the dictionary as suggested in [113, 79, 14, 102, 126, 147]. In this chapter we introduce this shift-invariant sparse coding model, which explicitly takes possible feature shifts into account.

The first section defines and clarifies the notion of shift-invariance used in this thesis and distinguishes the concept of shift-invariance used here

[1]Some of the material in this chapter has previously been published in [10]

from a concept that we call shift-consistency. With this terminology in place, section 3.2 introduces the shift-invariant model studied and used throughout this work. This model is based on the linear model of the previous chapter, however, additional structure is imposed on the matrix **A**. The inclusion of these structures then leads to a modification of the learning rule used to update the features.

The number of features learned and the number of features in the signal are important parameters in the learning process. In section 3.3.1 we analyse this relationship and discuss the advantages offered by the shift-invariant sparse coding model introduced here when compared to the standard sparse coding formulation. In digital signal processing we are dealing with discretised time-series. Often the original time-series is a mixture of features that can occur at continuously shifted time-locations. The effect that sampling of such signals has on the features learned is analysed in section 3.3.2.

3.1 Shift-Invariant Approximations/Representations

We can distinguish two cases of 'shift-invariance'; one in which the representation remains unchanged for a shift in the input of the system and one for which the representation is shifted linearly with a shift in the input. The second definition is the standard definition for shift-invariant systems in engineering and is the definition used throughout this thesis. As there has been work on systems that show 'shift-invariance' with either of the definitions, we first summarise some work for which the representation does not shift with a shift in the input. We call such a representation *shift-consistent*. A formal definition is given below.

To motivate the notion of shift-invariance and shift-consistency let us think of an illustrative example. Imagine we have a picture and would like to know whether the picture contains a face. So we are looking for a representation that has an element that signifies the presence of a face. If this representation remains unchanged when we shift the face in the image, i.e. the same element in the representation is active due to the presence of the face, then we have a shift-consistent representation. The representations we are dealing with in this thesis also give information of the location of the face in the image. This representation has an element for a face at every possible location, so if the face is moved in the picture, the representation also moves, and a different element in the representation becomes active. This behaviour of the representation is called shift-invariance here.

3.1.1 Shift-Consistent Approximations/Representations

Definition 3.1. *A map, such that $x(t) \mapsto s(t)$ and $x(t+t_0) \mapsto s(t)$ $\forall t_o \in \mathcal{T}$ is called shift-consistent for the set of* admissible shifts $\mathcal{T}$.

Phase blind methods are approximately shift-consistent for small shifts. Another example of a shift-consistent representation can be found in neuroscience. Here the problem of shift-invariant vision is assumed to be dealt with in two distinct systems, the 'where' system that deals with the localisation of a feature and the 'what' system that deals with the identification of a feature [117]. The system that deals with the identification of the feature is then shift-consistent.

Computational models of this have first been proposed in [38] and extensive studies on invariant vision can be found in [121, 117, 118, 122, 120, 119]. These methods learn a representation based on time constraints. It is assumed that features change slowly during observation. The representations then enforce features to be active over longer time periods. This was formalised as Slow Feature Analysis in [151].

Another method for shift-consistent representations are the averaging of filter coefficients in [28]. Here transforms such as discrete wavelet-transforms or the discrete Fourier transform are implemented using filter banks and instead of downsampling the filter bank output, averages over the filter coefficients are taken.

In [56], independent subspaces were learned and it was found that the representations were shift-consistent to a certain degree, i.e. for small shifts. This behaviour is also known from complex wavelets [64].

In [40] a Bayesian model is studied that uses a Gaussian translation prior and an EM algorithm for clustering to find shift-consistent representations. This model is not additive as the model studied in this thesis but assumes that each observation is produced by an individual feature represented by the mean of a Gaussian distribution. The realisation of this distribution is assumed to be transformed and noise is again added to the transformed signal. This model can be used to learn and infer the feature as well as the transform, which is not restricted to be a shift.

3.1.2 Shift-Invariant Approximations/Representations

Definition 3.2. *A map, such that $x(t) \mapsto s(t)$ and $x(t+t_0) \mapsto s(t+t_0)$ $\forall t_o$ is called* shift-invariant.

An often encountered problem in engineering in which a single feature has to be learned in a shift-invariant model is the blind equalisation problem in which both, the impulse response of a linear and shift-invariant system as well as the input signal of this system are unknown [54]. An early approach to shift-invariant feature learning which uses a linear combination of features was developed by Zazula and can be found in [107, 154, 67]. In these papers a mixture of MA models plus noise is studied. There, the functional relationship of the fourth order cumulant of the input of a general filter to the fourth order cumulant of the output of the filter is used. It can be assumed that the fourth order cumulant of a sparse and independent source is given and the fourth order cumulant of the output is easily estimated and not influenced by the additive Gaussian noise. This then leads to a set of quadratic equations of the individual transfer functions, which can be solved using standard optimisation methods. Unfortunately for the problems studied in this thesis this method is not feasible due to its computational complexity and memory demands. The fourth order cumulant of a stationary process can be expressed using a third order tensor if we assume a non-time-varying system (see [23]). This third order tensor has N^3 values. With $N = 1024$ as used in some of the experiments described in this thesis, the computation of this tensor is computationally taxing. It should be noted that in order to estimate the impulse responses of K MA systems with an impulse response of length F, we need to specify at least FK equations so that we have to estimate at least FK different fourth order cumulants. However, due to the estimation error in the fourth order cumulants it is of advantage to specify a higher number of equations and to use non-linear optimisation methods to find a minimum mean square error solution. Unfortunately, the solution to the set of non-linear equations is computationally taxing and not feasible for the problem dimension used in some of the examples of interest in this thesis.

Another possible approach to shift-invariant feature learning is the work in [128, 127] that deals with frame design, i.e. with the design of synthesis frames similar to the ones used in sparse coding. The overlapping frames learned in this work do not occur at all possible locations. However, the model developed here can be seen as an extension of this idea to frame design in which the overlapping frames are located at all possible shifts.

In neuroscience, shift-invariant representations have been studied recently in [79, 77, 131]. In this work, the observation is modelled as a linear combination of convolutions of the sparse representation, (i.e. the coefficients $\mathbf{s}$) with a set of known features. This convolutive model is the same generative model introduced in the next section. This work deals with the problem of finding the sparse representation $\mathbf{s}$ for a given model matrix $\mathbf{A}$ and does not investigate methods to adapt the matrix $\mathbf{A}$. In [113] the same convolutive model was suggested to reconstruct the observation signal from the recording of neurons. In this example the features are defined by each neuron and the sparse representation is the action potential of this neuron. It was shown that such a convolutive model can be used to reconstruct observation signals from neural activations. However, in this work, both, the input signal, i.e. the observation, as well as the neural activations were known and only the impulse response of the filter, which defines the feature, had to be found.

The model introduced in this chapter as well as some of the learning rules developed in the next chapter have recently been published in [148, 147]. A similar model was also used in [126], however, here the features were constrained to be wavelets. The learning rule used in this paper uses a Gibbs sampling method, which is discussed in chapter 6 where we extend this method further.

Another method proposed for invariant sparse representations is the method proposed in [51], which uses a bilinear transformation to model shift invariance. This model is also of a similar form to the model introduced here and so is the learning rule. The specification of the prior distribution however differs.

An extension of Non-Negative Matrix Factorisation to a shift-invariant model was also recently proposed in [130], where it was used to learn audio features such as words in the spectral domain. This method models audio spectra as a summation of spectral features at all possible shifts. A similar shift-invariant model is proposed in [18] to study frog leg muscle activations and enforces both sparsity and positivity.

3.2 Shift-Invariant Sparse Coding Model

In this section we introduce an extension of the general sparse coding model for time-series and similar data in which features can occur at arbitrary locations. First, we define a generative model that includes all shifted features. We then derive learning rules for this extended model by finding approximations to the gradient of the marginal likelihood w.r.t. the feature parameters.

In time-series, features can often occur at arbitrary time locations. The application of the sparse coding model described in the previous chapter to such time-series has previously been achieved by arbitrarily cutting the time-series into blocks $\mathbf{x}$. However, these blocks seldom align with the features in the time-series such that the shifts of the different instances of a feature relative to the block positions vary arbitrarily. We therefore modify the model to account for these arbitrary shifts of features relative to a selected block position. This can be achieved by enforcing structure on the matrix $\mathbf{A}$. From now on we use the notation $\mathbf{A}$ to refer to this structured matrix. To state the model used in [113, 79, 14, 102, 126, 147] we introduce the following notation:

The index k labels a particular feature whilst the index l denotes the corresponding shift relative to the beginning of the data-block analysed. $\mathcal{K}$ and $\mathcal{L}$ are the sets of indices of all features and shifts respectively, while we denote the length of the features $\mathbf{a}_k$ as L. From now on we let l be zero to denote no shift[2], i.e. $\mathbf{a}_{k0} = [a_{k1}, a_{k2}, \cdots, a_{kL}, 0, \cdots, 0]^T$ whilst, for example, $\mathbf{a}_{k-4} = [a_{k5}, a_{k6}, \cdots, a_{kL}, 0, \cdots, 0]^T$ and so forth. Note that for all $p - l \notin [0, L]$ the elements of $\mathbf{a}_{kl}$ are set to zero and that for $l < 0$ and $l > M - L$ the features $\mathbf{a}_k$ have to be truncated. We use a_{kp} to denote the p^{th} component of a feature, which should not be confused with the notation $\mathbf{a}_{kl}$ that refers to a shifted feature. With this notation we can write $\mathbf{A} = [\mathbf{a}_{1,-L}, \mathbf{a}_{1,-L+1}, \ldots, \mathbf{a}_{k,M-1}, \mathbf{a}_{k+1,-L}, \ldots, \mathbf{a}_{K,M-1}]$.

$\mathbf{A}$ is shown graphically for $M = 4, N = 12, L = 3, K = 2$ below:

$$\begin{bmatrix}
\star_3 & \star_2 & \star_1 & 0 & 0 & 0 & \circ_3 & \circ_2 & \circ_1 & 0 & 0 & 0 \\
0 & \star_3 & \star_2 & \star_1 & 0 & 0 & 0 & \circ_3 & \circ_2 & \circ_1 & 0 & 0 \\
0 & 0 & \star_3 & \star_2 & \star_1 & 0 & 0 & 0 & \circ_3 & \circ_2 & \circ_1 & 0 \\
0 & 0 & 0 & \star_3 & \star_2 & \star_1 & 0 & 0 & 0 & \circ_3 & \circ_2 & \circ_1
\end{bmatrix}$$

Here the two features are shown as stars $\star$ and circles $\circ$ respectively with the subscripts labelling the sample.

If we use s_{kl} as the coefficient multiplying feature $\mathbf{a}_{kl}$, then the data model can be written as

$$\mathbf{x} = \sum_{k \in \mathcal{K}, l \in \mathcal{L}} \mathbf{a}_{kl} s_{kl} + \epsilon = \mathbf{A}\mathbf{s} + \epsilon.$$

The observation block $\mathbf{x}$ is modelled with features and all their possible shifts. Note that this model is a mixture of convolutions and that the matrix $\mathbf{A}$ is the concatenation of convolution matrices. The coefficient vector $\mathbf{s}$ is now a concatenation of the signals being convolved.

This is shown more clearly by writing the model in the familiar form of a discrete system:

$$x[t] = \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}} a_k[L + 1 - l] s_k[t + 1 - l] + \epsilon[t],$$

which shows the equivalence of the model to a mixture of linear shift-invariant filters with added noise.

The above model can be used to describe data blocks of arbitrary length and it would be possible to model the complete observation sequence.

[2]Note that the observation $\mathbf{x} \in \mathbb{R}^M$ with $M \geq L$ so that the features $\mathbf{a}_k$ have to be zero-padded accordingly.

However, for many time-series of interest, it is infeasible to deal with the complete observation at once. Nevertheless, it is possible to randomly select blocks of data from the time-series of interest and to use stochastic gradient descent to learn the model parameters by using a similar method to the one introduced in the previous chapter. In this case, the length M of the observation vector $\mathbf{x}$ can be chosen arbitrarily to be at least L. In the experiment reported later this vector was chosen to be twice the size of the feature length.

3.2.1 Learning Rule

The model introduced above requires a revision of the learning rules introduced in the previous chapter. The elements of the features $\mathbf{a}_k$ are now repeated along the diagonals of the matrix $\mathbf{A}$. The values of $\mathbf{A}$ cannot be updated individually without taking this repetition into account, which is achieved by calculating the gradient of $\log p(\mathbf{x}|\mathbf{A},\mathbf{s})$ (which is required in equation (2.4)) w.r.t. the p^{th} component of the feature $\mathbf{a}_k$.

Using $\epsilon_m = x_m - \sum_{k\in\mathcal{K},l\in\mathcal{L}} a_{km+l}s_{kl}$ we can write the log likelihood as:

$$\log p(\mathbf{x}|\mathbf{A},\mathbf{s}) \propto -\frac{0.5}{\sigma_\epsilon^2}\sum_m \epsilon_m^2.$$

We can now calculate the derivative of this w.r.t. a_{kp} and write:

$$\frac{\partial \log p(\mathbf{x}|\mathbf{A},\mathbf{s})}{\partial a_{kp}} = -\frac{1}{\sigma_\epsilon^2}\sum_m \epsilon_m \frac{\partial \epsilon_m}{\partial a_{kp}}.$$

The derivative on the right only leaves those $a_{k,m+l}$ for which $m+l=p$. The gradient then becomes:

$$\Delta a_{kp} = \frac{1}{\sigma_\epsilon^2}\left\langle \sum_m \epsilon_m s_{k,m-p}\right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})}. \tag{3.1}$$

If $\mathbf{x}$ and $\mathbf{a}_k$ are both in $\mathbb{R}^L$, we can write this expression as a convolution and derive a gradient update rule for the set of features $\{\mathbf{a}_k\}$ as:

$$\Delta\{\mathbf{a}_k\} \propto \sigma_\epsilon^{-2}\left\langle \epsilon \star \{\mathbf{s}_k\}\right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})} ,$$

where $\star$ is the convolution operator and $\epsilon = \{\epsilon_m\}$.

This gradient then leads to an update of the features of the form:

$$\{\mathbf{a}_k\}^{r+1} = \frac{\{\mathbf{a}_k\}^r + \nu\Delta\{\mathbf{a}_k\}}{\|\{\mathbf{a}_k\}^r + \nu\Delta\{\mathbf{a}_k\}\|_2}.$$

Due to the scale ambiguity in the model, in each update the features $\mathbf{a}_k$ are normalised to unit L_2 norm.

This learning rule again requires the evaluation of an expectation w.r.t. $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$, which cannot be solved analytically, so that approximations are required that are similar to the methods discussed in the previous chapter. We introduce and study several possible approximations to the learning rule for the shift-invariant sparse coding model in the next part of this thesis. Chapter 4 studies analytic approximations to the required integration whilst chapter 5 develops an importance sampling method. Chapter 6 developes and studies a Markov chain Monte Carlo approximations.

3.3 Theoretical Analysis of Feature Extraction with the Shift-Invariant Sparse Coding Model

3.3.1 Sensitivity to the Model Size

If the analysed signal consists of a superposition of features at arbitrary locations, then the model used to learn these features has to have enough free parameters to represent these features. In general this means that at least one feature has to be learned for each feature present. However, in the standard sparse coding model, features have to be learned at all possible shifts, so that the number of features to be learned is much larger than the number of features in the signal. If the standard sparse coding model does not have enough free parameters to represent the features in the signal, not all features are learned. Instead, some features have to be used to model more than one feature in the observation.

In this section we study the influence of the number of features used in the traditional sparse coding model, when this number is smaller than the number of features in the signal. We assume here that the observed signal follows the model $\mathbf{x} = \sum \mathbf{a}_k s_k + \epsilon$. $\hat{a}_{\hat{k}}$ and $\hat{s}_{\hat{k}}$ are used to denote the estimated features and coefficient respectively. We now use the indices k

to denote the features and the associated shifts of the underlying process, while $\hat{k}$ indexes the learned features.

The expected ML estimate of a feature $\hat{a}_{\check{k}}$ w.r.t. the distribution of the data, i.e. w.r.t. the distribution of ϵ and $\mathbf{s}$, is the value for which the expected gradient is zero. We can write this expected gradient as:

$$\langle \Delta \hat{a}_{\check{k}} \rangle_{p(\epsilon,\mathbf{s})} = \left\langle \mu\sigma_\epsilon^{-2} \int \left(\sum_k a_k s_k - \sum_{\hat{k}} \hat{a}_{\hat{k}} \hat{s}_{\hat{k}} + \epsilon \right) \hat{s}_{\check{k}} \; p(\hat{\mathbf{s}}|\mathbf{x}, \hat{\mathbf{A}}) \; d\hat{\mathbf{s}} \right\rangle_{p(\epsilon,\mathbf{s})} .$$

Note the use of $\check{k}$ to index the particular feature for which we evaluate the gradient and the corresponding coefficient $s_{\check{k}}$, while k indexes the true features in the generative model. $\hat{k}$ indexes all of the estimated features and coefficients. Using the abbreviation

$$\mathcal{T} = \mu\sigma_\epsilon^{-2} \left(\sum_k a_k s_k - \sum_{\hat{k}} \hat{a}_{\hat{k}} \hat{s}_{\hat{k}} + \epsilon \right) \hat{s}_{\check{k}}$$

we can write this as:

$$\begin{aligned} & \int\int\int \mathcal{T} p(\hat{\mathbf{s}}|\mathbf{x}, \hat{\mathbf{A}}) p(\epsilon, \mathbf{s}) \; d\hat{\mathbf{s}} \; d\mathbf{s} \; d\epsilon \\ = & \int\int\int \mathcal{T} p(\hat{\mathbf{s}}|\epsilon, \hat{\mathbf{A}}, \mathbf{A}, \mathbf{s}) p(\epsilon) p(\mathbf{s}) \; d\hat{\mathbf{s}} \; d\mathbf{s} \; d\epsilon \; , \end{aligned}$$

where the last step is possible as $\mathbf{s}, \mathbf{A}$ and ϵ define $\mathbf{x}$ and as ϵ is assumed to be independent of $\mathbf{s}$. Setting the gradient to zero and rearranging gives:

$$\begin{aligned} \hat{\mathbf{a}}_{\check{k}} \langle \hat{s}_{\check{k}} \hat{s}_{\check{k}} \rangle_{p(\hat{\mathbf{s}}|\hat{\mathbf{A}}\mathbf{A})} &= \mathbf{a}_{\overline{k}} \langle s_{\overline{k}} \hat{s}_{\check{k}} \rangle_{p(\hat{\mathbf{s}},\mathbf{s}|\hat{\mathbf{A}}\mathbf{A})} \\ &+ \sum_{k \neq \overline{k}} \mathbf{a}_k \langle s_k \hat{s}_{\check{k}} \rangle_{p(\hat{\mathbf{s}},\mathbf{s}|\hat{\mathbf{A}}\mathbf{A})} \\ &- \sum_{\hat{k} \neq \check{k}} \hat{\mathbf{a}}_{\hat{k}} \langle \hat{s}_{\hat{k}} \hat{s}_{\check{k}} \rangle_{p(\hat{\mathbf{s}}|\hat{\mathbf{A}}\mathbf{A})} \\ &+ \langle \epsilon \hat{\mathbf{s}}_{\check{k}} \rangle_{p(\hat{\mathbf{s}},\epsilon|\hat{\mathbf{A}}\mathbf{A})} . \end{aligned}$$

where we have introduced the index $\overline{k}$ to label the true feature and coefficient associated with the feature and coefficient to be learned, i.e. we assume that feature $\hat{\mathbf{a}}_{\check{k}}$ converges to feature $\mathbf{a}_{\overline{k}}$. If we assume that $\hat{s}_{\check{k}}$ is uncorrelated to ϵ then the last term is zero. $\langle \hat{s}_{\hat{k}} \hat{s}_{\check{k}} \rangle_{p(\hat{\mathbf{s}}|\hat{\mathbf{A}}\mathbf{A})}$ is also zero due to the assumed independence of the individual $\hat{s}_{\hat{k}}$. So we are left with:

$$\begin{aligned}\hat{\mathbf{a}}_{\check{k}} \left\langle \hat{s}_{\check{k}} \hat{s}_{\check{k}} \right\rangle_{p(\hat{\mathbf{s}}|\hat{\mathbf{A}}\mathbf{A})} &= \mathbf{a}_{\overline{k}} \left\langle s_{\overline{k}} \hat{s}_{\check{k}} \right\rangle_{p(\hat{\mathbf{s}},\mathbf{s}|\hat{\mathbf{A}}\mathbf{A})} \\ &+ \sum_{k \neq \overline{k}} \mathbf{a}_k \left\langle s_k \hat{s}_{\check{k}} \right\rangle_{p(\hat{\mathbf{s}},\mathbf{s}|\hat{\mathbf{A}}\mathbf{A})}\end{aligned}$$

In order for a feature $\hat{\mathbf{a}}_{\check{k}}$ to converge to a feature $\mathbf{a}_{\overline{k}}$ we require the correlation between $\hat{s}_{\check{k}}$ and s_k to be zero for all $k \neq \overline{k}$.

If the number of features used to model a signal is less than the number of features in the signal at all locations, then dependencies between $\hat{s}_{\check{k}}$ and several s_k have to occur. Dependencies can also occur as a result of the inference process or the approximations to the learning rule used.

To analyse the possible dependencies which can occur due to the incorrect model size, we assume that all learned features have converged to some of the true features. The dependency between $\hat{s}_{\check{k}}$ and s_k (and therefore the exact form of the averaging process described above) then depends on which of the features $\mathbf{a}_k$ are modelled by each feature $\hat{\mathbf{a}}_{\check{k}}$. The feature chosen to model a feature which has not been learned, depends on the decrease in reconstruction error when using this feature. The highest decrease in this error is achieved by modelling a feature in the signal with the same feature at the exact location. If this feature is not available at this location, a feature at a different location or a different feature has to be used.

In the following list three forms of dependencies which can occur are given together with the influence they have on the learned features:

- A feature can be modelled with a slightly shifted version of itself. If several slightly shifted features are modelled by a single feature, then the average update of this feature is a low-pass filtered version of the true feature.

- A windowed periodic feature can be modelled with a version of itself which is shifted by multiples of the period. A weighted averaging over such feature shifts leads to a windowing of the learned feature.

- A missing feature can also be modelled with a different feature. The chosen feature is likely to share a strong frequency component and is at a location at which both features have the same phase for this component. Averaging then increases this frequency component but might decrease other frequency components, as the phase for those other components might not match.

This seems to suggest that if the number of features to be learned is less than the number of features in the signal, windowed and filtered features emerge. However, the above derivation uses the traditional sparse coding formulation. If shift-invariance is explicitly enforced and if the inference process is working correctly (i.e. the s_k are uncorrelated to $\hat{s}_{\hat{k}}$ for all but one pair of coefficients) then the first two effects (i.e. the filtering and the windowing) cannot occur.

3.3.2 Sampling and Shift-Invariant Sparse Coding

Many time-series encountered in signal processing are sampled versions of continuous signals. Often the features that contribute to the original signal can occur at arbitrary and continuous time locations. Such a continuously shifted feature occupies a 1-dimensional manifold in the sampled space. This path is shown in the left panel of figure 3.1 for a 3-dimensional signal. (The circles show the location of the sampled signal and its shifted versions. The dotted lines are the projections onto the xy, xz and yz planes.)

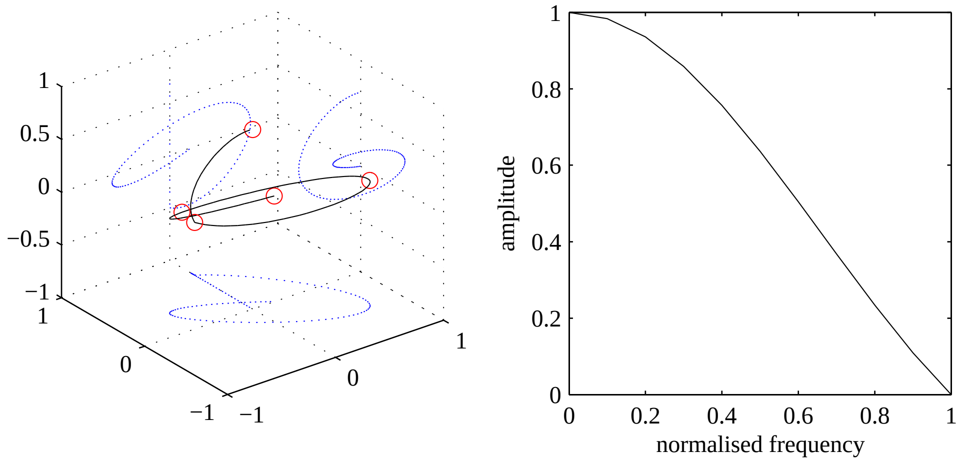

Figure 3.1: Path of a shifted feature (left) and filtering due to sampling (right).

If we assume that during learning each true feature in the signal is assigned to the single shift position that is closest to the true position of the feature, then the updates of a feature to be learned are averages of the true feature over shifts of one sample. The frequency response of this averaging process can be found as the Fourier transform of a rectangular window. The amplitude response is $\frac{\mathrm{sinc}(\pi \mathrm{f})}{\pi \mathrm{f}}$ where f is the normalised frequency. This sinc function is plotted in the right panel of figure 3.1. It can be seen that the effect of continuously shifted features in the original signal leads to the emergence of low-pass filtered features. This effect cannot easily be overcome without substantially increasing computational complexity.

Conclusions

The sparse coding model can be used for time-series data. However, for such data, features can occur at arbitrary time locations, which requires the model to learn features at different shifts. In this chapter we have extended the sparse coding model to explicitly take these shifts into account. In this formulation, the model is explicitly constrained so that features can be used at arbitrary locations to describe the signal.

The theoretical analysis presented suggests several advantages of this model. However, a practical implementation can only approximate the learning rule proposed and practical results have to be analysed for such approximations.

In the next part of this thesis three different approximations to the learning problem are proposed. The first method is an extension of previously proposed methods used for the standard sparse coding model. The problem size requires the implementation of further approximations discussed in the next chapter. The other two methods are based around Monte Carlo approximations of the learning rule. In chapter 5 an importance sampling strategy is proposed and in chapters 6 a Markov chain sampler is developed and studied.

Part II

Algorithmic Advances

Chapter 4

Analytic Approximation[1]

The integral in equation (2.5) cannot be solved analytically and approximations are required. In chapter 2 two analytic approximations for the sparse coding learning rule have been introduced. These learning rules approximate the integral with either a delta function or a Gaussian around the MAP estimation of **s**.

In this chapter we derive a learning rule similar to the delta approximation learning rule introduced in chapter 2. This approximation of the learning rule can again be interpreted as a joint ML estimation of both **A** and **s** in the missing data problem where **s** can be interpreted as missing data. This learning rule requires the MAP estimate for **s** conditioned on **x** and **A**. One possible method of evaluating this estimate is discussed and derived as an Iterative Re-weighted Least Squares algorithm. We also show that this algorithm has an interpretation as an EM algorithm. This realisation allows us to use the well developed theoretical results for EM algorithms that guarantee convergence of the method. This also allows implementations of the optimisation based on generalised EM methods.

The size of the problems studied in chapters 8, 9 and 10 does not allow for the straightforward use of the optimisation method introduced. We therefore describe a strategy to select a small number of features to reduce the problem size. This subset selection is based on the distance of the features from the signal. Finally, we discuss issues relating to the implementation of this method.

[1]This chapter is partly based on material published in [9]

4.1 The Delta Approximation Learning Rule

In the shift-invariant sparse coding model introduced in the previous chapter the main problem is the learning of the features $\mathbf{a}_k$. This requires an estimation of the gradient:

$$\Delta a_{kp} = \frac{1}{\sigma_\epsilon^2} \left\langle \sum_m \epsilon_m s_{k,p-m} \right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})}.$$

For the standard sparse coding model, this gradient cannot be evaluated analytically. If we drop the expectation in the learning rule and use the MAP estimate of $p(\mathbf{s}|\mathbf{x},\mathbf{A})$, we get the following delta approximation to the learning rule that is similar to the learning rule in [103]:

$$\Delta a_{kp} = \frac{1}{\sigma_\epsilon^2} \sum_m \hat{\epsilon}_m \tilde{s}_{k,p-m}. \tag{4.1}$$

Here we use $\hat{\epsilon}_m$ to denote the m^{th} element of the vector $\mathbf{x}-\mathbf{A}\hat{\mathbf{s}}$. We have introduced the notation $\tilde{s}_{k,p-m}$. Intuitively one would choose $\tilde{s}_{k,p-m} = \hat{s}_{k,p-m}$, i.e. use the MAP estimate of the coefficients $\mathbf{s}$. However, by using the delta approximation of the posterior $p(\mathbf{s}|\mathbf{A},\mathbf{x})$, information about the distribution is lost. This is especially critical for those feature shifts for which only part of the feature contributes to the current observation $\mathbf{x}$ (in the example in the previous chapter these are columns one, two, five to eight, eleven and twelve.). For example, at the extreme shift positions, where a feature only overlaps with the observation block by one sample (i.e. columns one, six, seven and twelve in the example in chapter 3), there is no information in the observation block to guide the selection of a specific feature. Any error in modelling the first and last sample in the observation block can therefore be reduced to exactly zero by selecting any feature at such an extreme shift with an appropriate coefficient value. This uncertainty would be reflected in the full posterior $p(\mathbf{s}|\hat{\mathbf{A}},\mathbf{x})$ by an increased variance for the coefficients associated with these features. This information is not available in the delta approximation in Eq. (4.1) and, as suggested in [14], only those coefficients are used in the feature update for which the entire feature contributes to the observation. We therefore use:

$$\tilde{s}_{k,p-m} = \begin{cases} \hat{s}_{k,p-m} & \text{if } 0 \leq p-m \leq M-L \\ 0 & \text{otherwise.} \end{cases}$$

In the example in the previous chapter, we would therefore only use the coefficients associated with the third, fourth, ninth and tenth columns. This does not bias the estimate of the features if the data blocks $\mathbf{x}$ are selected at random locations during learning.

4.2 Inference by MAP Estimation via the EM Algorithm

Inference of the coefficients $\mathbf{s}$ can be done using the same methods employed in the non-shift-invariant case. For example, the gradient rule used by Lewicki as well as Olshausen and their co-workers [103, 104, 80, 78] can be used by simply replacing their matrix $\mathbf{A}$ with the version containing all shifts. In this section, however, we derive an EM algorithm to find the MAP estimate of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. This algorithm has been derived independently by different researchers and we present the different derivations here.

We use the method discussed in this subsection in many of the experiments reported in this thesis and therefore discuss it in more detail. The explanation of the FOCUSS algorithm (defined below) as a form of IRLS (also defined below) and its equivalence to an EM algorithm for certain prior formulations has also not been given in the literature before.

4.2.1 Prior Formulations

In order to use the optimisation method developed in this chapter, a continuous prior formulation is required. This means, that we cannot use a formulation in which a discrete probability mass is placed at zero, as gradient optimisation methods cannot be used for such models and more sophisticated methods such as those developed in the next chapters are required.

In this chapter we therefore use a scale mixture of Gaussians prior of the form $p(s_n|\sigma^2_{s_n}) \sim \mathcal{N}(0, \sigma^2_{s_n})$. By using different hyper priors for the variance term we can model a range of different distributions for $p(s_n)$. For hyper priors of $\frac{1}{\sigma^2_{s_n}}$ of the form $\sigma_{s_n} p_\alpha(\alpha/2)$ where $p_\alpha(\alpha/2)$ is a symmetric alpha-stable distribution of $\frac{1}{\sigma^2_{s_n}}$ [149, 145] the marginalised prior $p(s_n)$ become generalised Gaussians of which the Laplacian as well as the Gaussian distributions are special cases. The generalised Gaussian distribution is given as:

$$\frac{1}{Z} e^{-\frac{\lambda_c}{2}|s|^p}.$$

This prior leads to sparseness measures in the optimisation method that are of the L_p (quasi) norm type. For $p > 1$, the distribution has weaker tails than the Laplacian, i.e. has a non-convex logarithm and as discussed in [71] and cannot be used as a sparsifying prior. Another possible hyper prior is the inverse gamma. The marginal likelihood is then of Student t form. Other hyper priors are possible and an extensive list for hyper priors for scale mixtures of Gaussian models and the associated marginal distributions are given in [31].

4.2.2 The EM Interpretation of the Algorithm

In the EM algorithm by Figueiredo et al. in [36, 33, 35, 34] a parameter free model was proposed. The algorithm is based on a hierarchical prior $p(s_n|\sigma^2_{s_n}) \sim \mathcal{N}(0, \sigma^2_{s_n})$ with the uninformative and parameter free Jeffrey's hyper prior $p(\sigma^2_{s_n}) = \frac{1}{\sigma^2_{s_n}}$. This hyper-prior is improper and leads to an extremely super-Gaussian prior.

For the Gaussian noise model the likelihood is:

$$p(\mathbf{x}|\mathbf{A}, \mathbf{s}) \sim \mathcal{N}(\mathbf{x} - \mathbf{A}\mathbf{s}, \sigma^2_\epsilon \mathbf{I}).$$

Using the prior

$$p(\mathbf{s}|\mathbf{\Sigma_s}) \sim \mathcal{N}(0, \mathbf{\Sigma_s}),$$

where we use $\mathbf{\Sigma_s} = \text{diag}\{\sigma^2_{s_n}\}$. This leads to the well known expression for the mode of the posterior $p(\mathbf{s}|\mathbf{A}, \mathbf{x}, \mathbf{\Sigma_s})$ of:

$$(\sigma^2_\epsilon {\mathbf{\Sigma_s}}^{-1} + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}.$$

$\mathbf{\Sigma_s}$ can be thought of as missing data and thus the EM algorithm can be used. The logarithm of the posterior can now be written as:

$$\begin{aligned}\log p(\mathbf{s}|\mathbf{A},\mathbf{x},\mathbf{\Sigma_s}) &\propto \log p(\mathbf{x}|\mathbf{A},\mathbf{s}) + \log p(\mathbf{s}|\mathbf{\Sigma_s}) \\ &\propto -n\log\sigma_\epsilon^2 - \|\mathbf{x}-\mathbf{As}\| - \mathbf{s}^T\mathbf{W}(\mathbf{\Sigma_s})\mathbf{s},\end{aligned} \tag{4.2}$$

where $\mathbf{W} = diag(\sigma_{s_1}^{-2}, \cdots, \sigma_{s_n}^{-2})$ is the inverse of the covariance matrix $\mathbf{\Sigma_s}$ of the prior. The E step is then:

$$\hat{\mathbf{W}}^{[r]} = \langle \mathbf{W}|\mathbf{x}, \hat{\mathbf{s}}\rangle,$$

where $\hat{\mathbf{s}}$ is the current estimation of $\mathbf{s}$ and $\hat{\mathbf{W}}^{[k]}$ is the expectation of the inverse of the prior covariance matrix. As $p(\sigma_{s_n}^2|\mathbf{A},\mathbf{x},\mathbf{s}) = p(\sigma_{s_n}^2|s_n) \propto p(s_n|\sigma_{s_n}^2)p(\sigma_{s_n}^2)$ the expectation of the inverse of each $\sigma_{s_n}^2$ can be calculated individually. For the Jeffrey's hyper prior this expectation can be evaluated and is $|\hat{s}_n|^{-2}$, so that $W^{[k]} = diag\{|\hat{s}_1|^{-2}, |\hat{s}_2|^{-2}, \cdots\}$. The expectation can also be evaluated for a hyper prior of the form $\frac{\lambda_p}{2}e^{-\lambda_p\sigma_{s_n}^{-2}}$ with $\sigma_{s_n}^{-2} \geq 0$. This exponential prior is a special form of the gamma distribution and also a particular case of the hyper prior of the form $\sigma_{s_n}p_\alpha(\alpha/2)$, where $p_\alpha(\alpha/2)$ is a symmetric alpha-stable distribution of $\frac{1}{\sigma_{s_n}^2}$. For this hyper prior the marginal distribution for $p(s_n)$ is a Laplacian (see [31] and [34]) and the E step can be written as $W^{[k]} = diag\{|\hat{s}_1|^{-1}, |\hat{s}_2|^{-1}, \cdots\}$ (Note the change in the exponent).

In the M-step the noise variance can be estimated as the argument σ_ϵ^2 that maximises equation (4.2) which is:

$$\hat{\sigma}_\epsilon^2 = \frac{\|\mathbf{x}-\mathbf{As}\|}{M}.$$

An estimate of the coefficient $\mathbf{s}$ can be found by:

$$\begin{aligned}\hat{\mathbf{s}} &= \arg\max\left(-\frac{\|\mathbf{x}-\mathbf{As}\|}{\sigma_\epsilon^2} - \mathbf{s}^T\hat{\mathbf{W}}^{[r]}\mathbf{s}\right) \\ &= \left(\hat{\sigma}_\epsilon^2\hat{\mathbf{W}}^{[r]} + \mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{x}.\end{aligned} \tag{4.3}$$

4.2.3 The FOCUSS Derivation of the Algorithm

The above iterative method is the same as the FOCUSS algorithm for noisy observations proposed in a series of papers [110, 111, 112] by Rao et al.. A good overview of the algorithm and underlying theory can be found in [68].

To derive the algorithm, we write the prior for $\mathbf{s}$ in the general form:

$$p(\mathbf{s}) = Z^{-1} e^{-\lambda_c f(\mathbf{s})}.$$

This density is further assumed to be zero mean and symmetric. Using this general distribution together with an assumed i.i.d. Gaussian error term, the MAP estimate of $\mathbf{s}$ can be found as:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} p(\mathbf{x}|\mathbf{A}, \mathbf{s})p(\mathbf{s}),$$

which leads to:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \frac{1}{2}\|\mathbf{x} - \mathbf{A}\mathbf{s}\|^2 + \sigma_\epsilon^2 \lambda_c f(\mathbf{s}). \tag{4.4}$$

The FOCUSS algorithm can now be derived by looking at the fixed points of the above minimisation problem. The derivative at the fixed point $\mathbf{s}^*$ can be written as:

$$\mathbf{A}^T(\mathbf{A}\mathbf{s}^* - \mathbf{x}) + \sigma_\epsilon^2 \lambda_c \Delta_s f(\mathbf{s}^*) = 0. \tag{4.5}$$

In general this cannot be solved but if the following factorisation is assumed:

$$\Delta_s f(\mathbf{s}) = \alpha(\mathbf{s})\mathbf{W}(\mathbf{s})\mathbf{s},$$

where $\alpha(\mathbf{s})$ is a scalar function of $\mathbf{s}$ and $\mathbf{W}(\mathbf{s})$ symmetric, positive definite and diagonal a solution becomes feasible. The diagonal form of $\mathbf{W}(\mathbf{s})$ follows from the assumption of independence of the different s_i and the other assumptions can generally be assumed for convex/schur convex functions as discussed in [112].

Equation (4.5) can now be written as:

$$\mathbf{A}^T(\mathbf{A}\mathbf{s}^* - \mathbf{x}) + \sigma_\epsilon^2 \lambda_c \alpha(\mathbf{s}^*)\mathbf{W}(\mathbf{s}^*)\mathbf{s} = 0,$$

which has a solution at:

$$\mathbf{s}^* = (\sigma_\epsilon^2 \lambda_c \alpha(\mathbf{s}^*)\mathbf{W}(\mathbf{s}^*) + \mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}.$$

As the fixed points $\mathbf{s}^*$ are not known, the matrix $\mathbf{W}$ has to be recalculated iteratively for the current estimation of $\hat{\mathbf{s}}$ and this estimate has then to be used in the above equation to find a better estimate of $\hat{\mathbf{s}}$. In the EM algorithm this step corresponds to the E step or the estimation of the prior variance in the hierarchical model. If $f(s)$ is one of the L_p (quasi) norms, then W can be calculated iteratively as $\mathbf{W}(\mathbf{s}) = diag(|s_1^{p-2}|, \cdots, |s_N^{p-2}|)$.

For $p = 1$, i.e. for a Laplacian prior, this is exactly the same expression as the EM algorithm with the same marginal prior distribution. For $p = 0$, i.e. for a prior which can be thought of as the limiting case of the generalised Gaussian, the expression is exactly the same as for the EM algorithm with a Jeffrey's hyper prior.

4.2.4 The IRLS Interpretation of the Algorithm

Here we derive the FOCUSS algorithm and the algorithm proposed by Figueiredo in the light of the well known Iterative Reweighted Least Squares (IRLS) approach used in statistics to solve generalised linear models (See Nelder's original paper [101]). As already noted by Dempster in [24], the EM algorithm often leads to IRLS solutions, a fact also mentioned in the thorough discussion paper by Green [46].

Equation (4.4) can be written as the weighted least squares problem:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \frac{1}{2}\|\mathbf{x} - \mathbf{A}\mathbf{s}\|^2 + \lambda \mathbf{s}^T\mathbf{W}\mathbf{s}, \tag{4.6}$$

where we use $\lambda = \sigma_\epsilon^2 \lambda_c$, with λ_c being the scale parameter of the prior. In order for equation (4.6) to have the same fixed points as equation (4.4) the weights have to be determined so that both functions have the same gradient at the fixed points. This is exactly the same approach taken in the derivation of the FOCUSS algorithm, which can therefore be seen as an IRLS algorithm.

Again, the weighting matrix $W(\mathbf{s})$ is a function of $\mathbf{s}$ and has to be re-evaluated in each iteration. The particular form of $W(\mathbf{s})$ has to be chosen such that the optimisation problems have the same fixed points. This is done by using $\mathbf{s}^T W(\mathbf{s})\mathbf{s} = f(\mathbf{s})$. This leads to exactly the same solution as the FOCUSS algorithm.

The FOCUSS and IRLS algorithms that use the L_p (quasi) norm or, equivalently, a generalised Gaussian prior, lead to an iterative calculation of the weighting matrix as $\mathbf{W}(\mathbf{s}) = diag(|s_1^{p-2}|, \cdots, |s_N^{p-2}|)$. Intuitively one would expect that in the limit, if we let p be zero, the algorithm would use a L_0 norm. This is however not true. In order to determine the exact form of the regularisation term, and therefore the marginalised prior used when p becomes zero, the general regularisation term minimisation problem has to be considered:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \frac{1}{2\sigma_\epsilon^2}(\mathbf{x} - \mathbf{A}\mathbf{s})^T(\mathbf{x} - \mathbf{A}\mathbf{s}) + \lambda_c f(\mathbf{s}),$$

the gradient of which is

$$\frac{1}{\sigma_\epsilon^2}(\mathbf{x} - \mathbf{A}\mathbf{s})\mathbf{A} + \lambda_c \frac{\partial}{\partial \mathbf{s}} f(\mathbf{s}).$$

The weighted least squares equivalents are:

$$\hat{s} = \arg\min_{s} \frac{1}{2\sigma_\epsilon^2}(\mathbf{x} - \mathbf{A}\mathbf{s})^T(\mathbf{x} - \mathbf{A}\mathbf{s}) + \mathbf{s}^T\mathbf{W}\mathbf{s},$$

with a gradient of

$$\frac{1}{\sigma_\epsilon^2}(\mathbf{x} - \mathbf{A}\mathbf{s})\mathbf{A} + 2\mathbf{s}^T\mathbf{W}.$$

With the weights obtained by setting $p = 0$ in the IRLS algorithm or the EM algorithm with a Jeffrey's hyper prior we can write

$$2\mathbf{s}^T\mathbf{W} = \lambda_c \frac{\partial}{\partial \mathbf{s}} f(\mathbf{s}) = 2[|s_1|^{-1}, |s_2|^{-1}, \cdots, |s_N|^{-1}].$$

By integration it can be found that for $\lambda_c = 1$ the regularisation term minimised in this case is

$$f(\mathbf{s}) = \sum_n \log |s_n|^2,$$

which is one of the regularisation terms proposed by Rao et al. in [71] and [112], which he termed Gaussian entropy [2]. This regularisation term goes

[2]The relation to Shannon-entropy however, is not clear. The term seems to stem from Donoho's paper [27] in which he called all sparsity enforcing measures entropy.

to $-\inf$ when any of the s goes to zero and it is therefore not possible to think of a global optimal solution for the optimisation problem. Furthermore, once a coefficient s becomes small enough, it is forced to zero very rapidly [20]. In practice, this behaviour is found to be of advantage if very sparse solutions are required, and it was found in practice, that the algorithm had a fast convergence. However, the results were clearly influenced by the starting conditions as there is no unique solution.

The EM algorithm leads to exactly the same algorithm as the IRLS approach when both algorithms assume a Laplacian prior distribution for $\mathbf{s}$. For $p = 0$, IRLS and FOCUSS algorithms use the logarithmic regularisation term, which can be seen to be the log prior of a parameter free, improper distribution of the form $\prod_1^N s_n^{-2}$. For other values of p the equivalence of the IRLS algorithm and the EM algorithm shows that the IRLS algorithm with a generalised Gaussian prior leads to the same maxima in the posterior as a scale mixture of Gaussians with a hyper prior of $\frac{1}{\sigma_{s_n}^2}$ of the form $\sigma_{s_n} p_\alpha(\alpha/2)$ where $p_\alpha(\alpha/2)$ is a symmetric alpha-stable distribution of $\frac{1}{\sigma_{s_n}^2}$. This suggests that the IRLS algorithm can also be interpreted as an EM algorithm for the priors of the generalised Gaussian family, however, an exact proof of this is not yet available. This also leads to the question of whether the generalised Gaussian family converges to $p(s) \propto s_n^{-2}$ in the limit as the generalised exponent p goes to zero.

Convergence is a key issue when dealing with iterative algorithms. The convergence of the FOCUSS algorithm has been proven in the noiseless case for both $p = 0$ and for $0 \neq p \leq 1$ in [110] and [112]. The previous discussion on the equivalence of the FOCUSS and the EM algorithms for certain priors and regularisation terms allows for the application of the convergence properties of the EM algorithm [90] to the FOCUSS algorithm in the noisy model. The rate of convergence of the FOCUSS algorithm in the noiseless case was also investigated in [110]. The rate of convergence for the noisy case is analysed in [20].

4.2.5 Parameter Estimation

The general optimisation problem to be solved can be written as:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \frac{1}{2\sigma_\epsilon^2}(\mathbf{x} - \mathbf{A}\mathbf{s})^T(\mathbf{x} - \mathbf{A}\mathbf{s}) + \lambda_c f(\mathbf{s}).$$

The ratio of the parameters σ_ϵ^2 and λ_c defines a trade-off between reconstruction accuracy and sparsity of the coefficients $\mathbf{s}$. In general, these parameters can be defined a priori, as was done in the case in which Jeffrey's hyper prior was used (which in effect defines $\lambda_c = 1$), or they can be estimated from the data as is done with the noise variance in the EM algorithm.

Different choices of prior formulations naturally lead to a different trade-off between reconstruction error and sparsity. Instead of using the Jeffrey's hyper prior as suggested above, it would also be possible to use the more general hyper prior $\sigma_{s_n}^{-2b}$ leading to the same algorithm as above, however, with differing values for λ_c.

Instead of fixing the parameter $\lambda = \lambda_c \sigma_\epsilon^2$ a priori, it is possible to estimate it. Engan [30] discusses three different approaches for estimating this parameter: a quality of fit criterion, a sparsity criterion and a modified L-curve criterion. In [68] a more heuristic method is given which enables a tuning of the trade-off between sparsity and noise. Here λ is calculated as:

$$\lambda = \lambda_{max}\left(1 - \frac{||\epsilon||}{||\mathbf{x}||}\right) \tag{4.7}$$

Here λ_{max} is a scaling of the λ parameter which can be adjusted depending on the problem at hand. In chapter 8 the results obtained using this λ parameter are compared to those obtained with the method proposed by Figueiredo, where an additional scaling parameter λ_c is used in Figueiredo's method for additional flexibility. Another method in which these parameters are estimated using maximum likelihood ideas is discussed in the next chapters.

4.2.6 Gauss Seidel Implementation

As was shown above, the FOCUSS algorithm can be interpreted as an EM algorithm for certain prior formulations. It is well known, that convergence of the EM algorithm is guaranteed even if the maximisation step is replaced by any method increasing the likelihood. It is therefore possible to replace the maximisation with respect to all coefficients $\mathbf{s}$ with a maximisation with respect to only a single coefficient or a set of coefficients. This can be done by using a Gauss Seidel iteration such that the inversion of the matrix can be avoided. However, more iterations are then needed in general. For structured dictionaries, such as unions of orthogonal dictionaries, the optimisation can be done with respect to a subset of the coefficients associated with a single orthogonal sub-dictionary. The optimisation can then be carried out very efficiently as no matrix inversion is required.

4.3 Subset Selection

Many engineering problems of interest suffer from high dimensionality. In the problems studied here, the length of the expected features can often be of the order of a few thousand and the number of features often in the hundreds. In the shift-invariant model this leads to a matrix $\mathbf{A}$ of substantial size, which means that the calculation of the maximum of the posterior $p(\mathbf{s}|\mathbf{A}, \mathbf{x})$ becomes prohibitively costly. This forbids a direct implementation of the above algorithms. Therefore, we propose the use of a subset selection step that offers a fast way to select a small subset of features depending on their correlation with the observation. After this selection, the optimisation routines mentioned in the previous section can be used by ignoring features not contained within the subset. With this approach, results can be obtained, even for problems of very high dimension.

Most of the coefficients s are zero with high probability and therefore most columns of $\mathbf{A}$ do not contribute to any one observation. In order to speed up the optimisation required to find the maximum of $p(\mathbf{s}|\mathbf{A}, \mathbf{x})$, we propose to exclude a large set of the columns of $\mathbf{A}$ from the optimisation. Information about which features to keep and which to exclude has to be taken from a particular observation $\mathbf{x}$. An additional requirement is that this selection process can be performed efficiently.

For extremely sparse approximations (as used in this thesis) we assume that each feature contributing to the observation has a high correlation with this observation, i.e. it is assumed that such signals are similar to their component features. The selection of a subset of features is therefore based on the correlation between the observation $\mathbf{x}$ and all columns of $\mathbf{A}$. Due to the structure in the matrix $\mathbf{A}$, this correlation can be evaluated efficiently using fast convolution. Based on this correlation it is possible to only select those features for which this correlation is high. However, an additional constraint has to be imposed. As smooth features shifted only slightly are similar to themselves, the same feature would be selected several times at adjacent locations. This can be avoided by constraining the selected subset to only include shifted versions of the same features if these are shifted by more than a certain distance, i.e. by selecting $\mathbf{a}_{kl}$ and $\mathbf{a}_{k\tilde{l}}$ only if $\frac{|l-\tilde{l}|}{L} > Q$ for some $Q < 1$.

The iterative selection procedure then selects the feature and shift with the highest correlation

$$\{k_i, l_i\} = \arg \max_{\{k,l\} \in \mathcal{K}_i \times \mathcal{L}_i} \langle \mathbf{a}_{kl}, \mathbf{x} \rangle,$$

where the product space of indices $\mathcal{K}_i \times \mathcal{L}_i$ is defined iteratively by removing subsets from the set of all features and shifts

$$\mathcal{K}_i \times \mathcal{L}_i = \mathcal{K} \times \mathcal{L} \backslash \bigcup_{\tilde{i}<i} k_{\tilde{i}} \times [l_{\tilde{i}} - QL; l_{\tilde{i}} + QL].$$

To better understand the assumptions made in this subset selection procedure we present the method from a statistical point of view. The posterior for $\mathbf{s}$ can be factored, using the index n instead of the indices k and l to denote the feature and the associated shift.

$$P(\mathbf{s}|\mathbf{A}, \mathbf{x}) = P(s_{n_1}|\mathbf{A}, \mathbf{x}) P(s_{n_2}|s_{n_1}, \mathbf{A}, \mathbf{x}) \dots$$

As a first approximation we only work with the MAP estimates for each distribution (i.e. to approximate the distributions with delta functions) and to further truncate the right hand side to a few terms, which are assumed to be non-zero. For the terms with non-zero s_n we assume a uniform prior for $P(s_n)$, and $P(\mathbf{x}|\mathbf{A}, s_n)$ is assumed to be Gaussian. These approximations lead to the posterior:

$$P(s_n|\mathbf{A}, \mathbf{x}) \sim \mathcal{N}(\mathbf{a}_n s_n, \sigma^2 \mathbf{I})$$

The problem now is to determine which coefficients to select to be non-zero. This is done iteratively. The first non-zero coefficient s_{n_1} is chosen by calculating:

$$n_1 = \arg\max_n P(s_n|\mathbf{A}, \mathbf{x}),$$

which is the index n, which maximises $\mathbf{x}^T \mathbf{a}_n$.

In order to approximate the other terms in the factorisation, an expression for $P(s_n|s_{1:\hat{n}-1}, \mathbf{A}, \mathbf{x})$ has to be found where we use the subscript notation $1 : \hat{n}$ to denote all variables with subscripts between 1 and $\hat{n}$. Here the notation $\hat{n}$ is used to distinguish the ordered indices $\hat{n}$ from the unordered indices n. Bayes' rule gives:

$$P(s_{\hat{n}}|\mathbf{A}, \mathbf{x}, s_{1:\hat{n}-1}) \propto P(\mathbf{x}|\mathbf{A}, s_{1:\hat{n}}) P(s_{\hat{n}}|s_{1:\hat{n}-1}).$$

The constraint on feature shifts can be interpreted in probabilistic terms as the use of the prior $P(s_{\hat{n}}|s_{1:\hat{n}-1}) = P(s_{\hat{n}}) U_{1:\hat{n}-1}$, where $P(s_{\hat{n}})$ is again a uniform distribution and $U_{1:\hat{n}-1}$ is a function which is zero for shifts around $l_{1:\hat{n}-1}$ but otherwise has a value normalising the distribution.[3] The main computational advantage in the subset selection procedure is the result of the selection of features close to the observation which has a statistical interpretation as an approximation of the probability $P(\mathbf{x}|\mathbf{A}, s_{1:\hat{n}})$ by $P(\mathbf{x}|\mathbf{A}, s_{\hat{n}})$. Note that for a Matching Pursuit algorithm, where each feature is selected to model the residual and not the original observation, the distribution $P(\mathbf{x}|\mathbf{A}, s_{1:\hat{n}})$ is Gaussian with a mean of $\sum a_{1:\hat{n}} s_{1:\hat{n}}$, whilst here a mean of $a_{\hat{n}} s_{\hat{n}}$ is used.

Selecting the index $\hat{n}$ can therefore be done in a similar fashion as above. The correlation of all features at those shifts which do not violate

[3]This interpretation of the constraint in terms of conditional priors is somewhat contrived, however it can be used to develop alternative methods in which other priors are specified such that close features are selected with a small but non-zero probability.

the constraint are again required. These correlations have already been calculated and do not have to be re-evaluated. In a Matching Pursuit algorithm the correlation would have to be recalculated at each step as it is determined from the residual.

The difference between Matching Pursuit and the method proposed here is that Matching Pursuit selects in each step a set of features that are as orthogonal as possible, whilst the proposed method selects a set of similar features. This choice seems to be more appropriate for harmonic musical mixtures as studied in this thesis, but may not have to be appropriate for other signals. The results presented in the later chapters of this thesis show the performance of this subset selection method. Similar experiments with an Orthogonal Matching Pursuit algorithm for subset selection did not produce satisfying results.

4.4 Implementation

The proposed algorithm can roughly be broken into three parts.

1. Selecting a subset of the features for each observation vector $\mathbf{x}$.
2. Finding the maximum of $p(\mathbf{s}|\mathbf{A}, \mathbf{x})$ within this subset.
3. Updating the features in matrix $\mathbf{A}$.

These steps are further explained below. The complete algorithm is shown in table 4.1.

In the algorithm described here, the length of the observation vector $\mathbf{x}$ can be chosen arbitrarily to be at least as long as the features $\mathbf{a}_k$. In the experiment reported later this vector was chosen to be twice the size of the feature length so that the matrix $\mathbf{A}$ contained $3 * K - 1$ shifted versions of each feature (where K is the feature length). The choice of the length of $\mathbf{x}$ must be a compromise between computational complexity and the problems caused by truncated features and the associated end-effects.

The features can be initialised with Gaussian noise but can also be pre-set to known functions such as Fourier bases. This can speed up convergence, but might also influence the outcome.

Table 4.1: Shift-invariant learning algorithm via EM.

```
     Input:
          User defined: signal $\{x_i\}$, the size of the subset $W$,
                        the percentage of maximal overlap $\mu$ and
                        the number and length of features $\mathbf{a}_k$.
     Output:
          $\mathbf{A}$.
1    $\{\mathbf{a}_k\}_{k\in\mathcal{K}}$=random,
         K is the set of all features
2    randomly select a data vector $\mathbf{x}$
3    calculate inner product between $\mathbf{x}$ and all shifted features
         $\{d_{k,l}\}_{k\in\mathcal{K},l\in\mathcal{L}} =< \mathbf{x}, \{\mathbf{a}_{k,l}\}_{k\in K,l\in L} >$
         $\mathcal{K}$ is the set of all features
         $\mathcal{L}$ is the set of all shifts.
4    for $\gamma = 1, \gamma < W$
         $[\tilde{\mathcal{K}}(\gamma), \tilde{\mathcal{L}}(\gamma)] = \arg\max_{k,l} \mathbf{d}_{k,l}$
         set $\{d_{k,l}\}_{k=\tilde{\mathcal{K}}(\gamma),l\in\hat{\mathcal{L}}} = 0$
             $\hat{\mathcal{L}}$ is the set of shifts close to the selected position.
5    for $r = 1, r < R$
         $\{s^{[r+1]}\}_{k\in\tilde{k},l\in\tilde{\mathcal{L}}}$
             $= EM(\{s^{[r]}\}_{k\in\tilde{k},l\in\tilde{L}}, \{\mathbf{a}_{k,l}\}_{k\in\tilde{\mathcal{K}},l\in\tilde{\mathcal{L}}})$
6    calculate gradient $\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}}$
         $\{\Delta\mathbf{a}_k\}_{k\in\tilde{K}} = \{\sum_{l\in\overline{\mathcal{L}}}(\mathbf{x} - \mathbf{a}_{k,l}s_{k,l})s_{k,l}\}_{k\in\mathcal{K},l\in\overline{\mathcal{L}}}$
         $\overline{L}$ is the set of all shifts for which features are not
         truncated
7    update $\{\mathbf{a}_{k,l}\}_{k\in\mathcal{K},l\in\mathcal{L}}$
         $\{\mathbf{a}_k\}^{[r+1]}_{k\in\tilde{\mathcal{K}}} = \{\mathbf{a}_k\}^{[r]}_{k\in\tilde{\mathcal{K}}} + \mu\{\Delta\mathbf{a}_k\}_{k\in\tilde{\mathcal{K}}}$
8    normalise $\{\mathbf{a}_k\}_{k\in\tilde{\mathcal{K}}}$
         $\{\mathbf{a}_k\}^{[r+1]}_{k\in\tilde{K}I} := \{\mathbf{a}_k\}^{[r+1]}_{k\in\tilde{\mathcal{K}}} / \|\{\mathbf{a}_k\}^{[r+1]}_{k\in\tilde{\mathcal{K}}}\|_2$
9    $\mu^{[r+1]} = \mu^{[r]}\nu; \nu < 1$
10   repeat from step 2 until convergence
```

The sparse coding model studied here has some indeterminacies. For example the value of the coefficients and the energy of the features can be scaled so that the model is still valid. To avoid problems with constant growth of the features re-normalisation has to be applied after each update. Here the L_2 norm of the features is arbitrarily normalised to 1. The model also has an ordering ambiguity, but as there is no natural order to the features, the found order is not relevant for the implementation.

The algorithm involves repeated calculation of $\mathbf{As}$ as well as $\mathbf{A}^T\mathbf{x}$. In the shift-invariant model these products are convolutions and can therefore be evaluated efficiently in the Fourier domain. However, due to the high sparsity of $\mathbf{s}$ a simple multiplication might be faster in some circumstances. Due to the shift-invariant structure, $\mathbf{A}$ does not have to be stored entirely; it is sufficient to store the individual features.

Conclusions

Approximations to the learning rule of the features $\mathbf{a}_k$ can be based on integral approximations around the MAP estimate of $p(\mathbf{s}|\mathbf{x}, \mathbf{A})$. An easy approximation based on a delta function can be used. This delta rule has an interpretation as a joint maximisation of the complete data likelihood in a missing data problem, which can justify its application.

For large problems, the derived learning rules cannot be used directly. Instead, we developed a subset selection step which can reduce the problem size. The reduced problem can then be solved using the learning rules derived. Experimental results and different applications of the developed algorithm are presented in chapters 7 and 8.

However, before studying the performance of the proposed method we develop other approximations to the learning rule. In the next two chapters we use Monte Carlo approximations. Chapter 5 deals with importance sampling Monte Carlo approximations of the learning rule, which can be much faster than the method developed here, so that the subset selection step is not required. In chapter 6 we study Gibbs sampling to draw samples from the posterior of $\mathbf{s}$. This method allows for an easy incorporation of additional constraints by the specification of more complex prior distributions.

Chapter 5

Importance Sampling Approximation[1]

The learning rule developed in chapter 3 used stochastic gradient descent steps for each data vector. As we use individual data vectors and not the entire set of available observations, the gradient is only on average the gradient of the complete data likelihood. This is however, sufficient for the stochastic gradient descent procedure [72] to find the maximum likelihood estimate. If we have a large amount of data we have to take many small gradient steps in this procedure. The learning rule developed in the previous chapter attempted to find a good approximation of the gradient of the likelihood of a single data vector. But as we have just stressed, this gradient is only a rough approximation of the gradient of interest and it seems wasteful to spend too much computation on an accurate approximation of this gradient.

Instead of finding a good approximation of the gradient we concentrate in this chapter on a fast method able to find a 'rough' approximation to this gradient. The parameters of interest in such an approximation are the variance and the bias. If the approximation is biased, then the stochastic gradient descent procedure only finds a biased estimate.

In this chapter we introduce an importance sampling approximation of the gradient. In order to develop such a method we first define a prior distribution that is a mixture of a Gaussian and a delta function. The delta function, which is centred at zero, forces coefficients to zero, whilst

[1]This chapter is based on work published in [12]

the Gaussian distribution models the non-zero coefficients.

In this model we have additional parameters, all of which can be estimated using maximum likelihood estimates. This is also done using stochastic gradient descent similar to the estimation of the dictionary. We therefore introduce the learning rules for these parameters in section 5.2. In the same section the particularities of the importance sampling method are introduced and the algorithm derived.

5.1 Model Formulation

The definition of sparsity used here assumes that many of the coefficients s are exactly zero. However, the prior probabilities used in the previous chapter had most of their probability mass close to zero but not at zero. In this and the next chapters we use different prior formulations that have a high probability mass at zero. Monte Carlo approximations can then be used to approximate the learning rule 3.1 introduced in subsection 3.2.1.

5.1.1 Prior Formulation

In this chapter we impose the following mixture prior in order to enforce sparsity of the coefficients $\mathbf{s}$:

$$p(\mathbf{s}|\mathbf{u}) = \prod_n p(s_n|u_n) = \prod_n (u_n \sqrt{\frac{\lambda_G}{2\pi}} e^{-\frac{\lambda_G}{2} s_n^2} + (1-u_n)\delta_0(s_n)), \tag{5.1}$$

where u_n is a binary indicator variable with discrete distribution:

$$p(u_n) = \frac{1}{1+e^{-\frac{\lambda_u}{2}}} e^{-\frac{\lambda_u}{2} u_n} \tag{5.2}$$

and $\delta_0(s_n)$ is the Dirac mass at zero. This prior is a mixture of a Gaussian distribution and the Dirac mass, therefore forcing many of the coefficients to be exactly zero with the hyper-prior regulating the sparsity of the distribution.

5.1.2 Dealing with Parameters

The parameters defining this model are $\theta = \{\mathbf{A}, \lambda_G, \lambda_u, \lambda_\epsilon\}$. These parameters can generally be dealt with in several ways: type one or two maximum likelihood, MAP estimation or marginalisation. Marginalisation is the proper Bayesian approach to deal with nuisance parameters; the MAP estimate would be the best possible estimate under zero-one error loss, whilst the posterior mean is the best estimate of a parameter of interest under a squared error loss. The main problem in this thesis is the approximation of integrals for marginalisation over nuisance parameters. Which parameters to estimate and which parameters to integrate out depends on the specific application and model. In this thesis we are primarily interested in marginalising over the coefficients $\mathbf{s}$ in the above model in order to calculate estimates of parameters of interest. The extension of the proposed methods to marginalisation over other parameters is possible by a straightforward extension of the ideas presented here and is not discussed further.

5.1.3 ML Learning of Model Parameters

Instead of adopting a fully Bayesian approach to the estimation of the parameters θ, i.e. instead of specifying prior distributions and calculating their joint posterior distribution or the maximum thereof, we again use a stochastic gradient descent algorithm to find the maximum likelihood estimate. In this model, the coefficients $\mathbf{s}$ and $\mathbf{u}$ are assumed to be nuisance parameters and are therefore integrated out of the data likelihood. The maximum likelihood estimate is then

$$\hat{\theta} = \arg\max_{\theta} \prod_i \int p(\mathbf{x}_i, \mathbf{s}_i, \mathbf{u}_i|\theta)\ d\{\mathbf{s}_i, \mathbf{u}_i\}.$$

We use the subscript i to denote the i^{th} observation vector and the associated coefficients, and I to denote the number of observations. This maximisation can again be solved using stochastic gradient optimisation by approximating the gradient w.r.t. all data with the gradient w.r.t. a single data vector $\mathbf{x}_i$. As discussed in chapter 2, we can write the gradient as:

$$\frac{\partial}{\partial\theta} \log p(\mathbf{x}|\theta) = \int p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \theta) \frac{\partial}{\partial\theta} \log p(\mathbf{x}, \mathbf{s}, \mathbf{u}|\theta)\ d\mathbf{s}\ d\mathbf{u}, \tag{5.3}$$

where from now on we drop the index i. Again, this expectation cannot be evaluated analytically in general and different approximations have been proposed in the literature [68, 80, 103], all of which require the calculation of the MAP estimate of $p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \theta)$. However, for many prior distributions the posterior over the coefficients is multi-modal and such estimates then only reflect a section of the distribution and might fail to account for most of the probability mass. Furthermore, such estimates are generally biased, so that convergence to the true maximum of the likelihood is not guaranteed.

During stochastic gradient learning of the parameters the algorithm randomly iterates through the available data, updating the parameters by a small amount in each iteration. This method therefore averages the gradient over several steps. This suggests the use of a less accurate approximation of the gradient in equation (5.3), which itself is already a rather poor approximation of the true gradient with respect to all available data. The stochastic gradient algorithm is then still able to converge to a maximum, given that the unbiasedness of the approximation is ensured and that the learning rate is decreased to zero [114].

Here we discuss a Monte Carlo approximation of the above integral using importance sampling [115]. This technique does not rely on MAP estimation and can therefore be implemented efficiently as shown below.

Importance sampling approximates an integral by a sum of weighted samples,

$$\int p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \theta) \frac{\partial}{\partial \theta} \log p(\mathbf{x}, \mathbf{s}, \mathbf{u}|\theta) \approx \sum_j^J w_j \frac{\partial}{\partial \theta} p(\mathbf{x}, \hat{\mathbf{s}}_j, \hat{\mathbf{u}}_j|\theta),$$

where $\hat{\mathbf{s}}_j$ and $\hat{\mathbf{u}}_j$ are samples drawn from a proposal distribution $q(\mathbf{s}, \mathbf{u})$ with the same support as $p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \theta)$. Here we use the subscript j to label the individual samples drawn. We further use J to denote the total number of samples. The weights are calculated as:

$$w_j = \frac{1}{J} \frac{p(\hat{\mathbf{s}}_j, \hat{\mathbf{u}}_j|\mathbf{x}, \theta)}{q(\hat{\mathbf{s}}_j, \hat{\mathbf{u}}_j)} = \frac{1}{J} \frac{p(\hat{\mathbf{s}}_j|\hat{\mathbf{u}}_j, \mathbf{x}, \theta) p(\hat{\mathbf{u}}_j|\mathbf{x}, \theta)}{q(\hat{\mathbf{s}}_j, \hat{\mathbf{u}}_j)}. \tag{5.4}$$

The use of the weights calculated with this formula gives an unbiased gradient estimate for the problem at hand. It can also be shown that the above Monte Carlo approximation converges for $J \to \infty$.

5.2 Algorithm

The dictionary learning algorithm is an iterative procedure repeating the three steps below until convergence.

1. Draw a data vector $\mathbf{x}_i$ at random from the available training data and draw a set of samples $\{\hat{\mathbf{s}}_j\}$ and $\{\hat{\mathbf{u}}_j\}$ from the data-dependent proposal distribution $p(\mathbf{s}|\hat{\mathbf{u}}, \mathbf{x}_i, \mathbf{A})\alpha(\mathbf{u}|\mathbf{x})$.
2. Calculate the weights w_j for each of the samples $\hat{\mathbf{s}}_j$ and $\hat{\mathbf{u}}_j$.
3. Update the parameters θ using a gradient step with a gradient approximation found by Monte Carlo integration using the weighted samples.

Each of the above steps and the required calculations are discussed below.

5.2.1 Proposal Distribution and Sampling

The mixture prior used enables us to draw samples $\hat{\mathbf{s}}$ conditionally on $\hat{\mathbf{u}}$ by setting $\hat{s}_n = 0$ if $\hat{u}_n = 0$. The non-zero coefficients $\hat{\mathbf{s}}_\emptyset$ are then Gaussian distributed with variance Λ^{-1} and mean

$$\Lambda\Lambda_{n,\emptyset}^{-1}\tilde{\mathbf{s}}_\emptyset,$$

where $\tilde{\mathbf{s}}_\emptyset$ is the least squares solution to the linear equation

$$\mathbf{x} = \mathbf{A}_\emptyset\mathbf{s}_\emptyset + \epsilon,$$

with

$$\Lambda = (\Lambda_{n,\emptyset} + \lambda_G\mathbf{I})$$

and

$$\Lambda_{n,\emptyset} = \lambda_\epsilon\mathbf{A}_\emptyset^T\mathbf{A}_\emptyset.$$

Here the subscript $\emptyset$ refers to a vector or matrix only including the elements associated with the non-zero coefficients u_n, e.g. $\mathbf{A}_\emptyset$ is a matrix with those columns of $\mathbf{A}$ which are multiplied by non-zero coefficients s_n. These calculations can be executed efficiently if only a few of the coefficients are non-zero. The distribution $p(\mathbf{u}|\mathbf{x}, \theta)$ cannot be sampled efficiently, so we resort to importance sampling. The variance of the approximation of the integral in equation (5.3) then depends not only on the number of samples used but also on the similarity between the proposal density and the density of interest. We therefore specify a proposal density that is proportional to the correlation between each feature and the data, hoping that this is a good first approximation of the true density.

$$\alpha(\mathbf{u}|\mathbf{x}) = \prod_n \alpha(u_n|\mathbf{x}), \tag{5.5}$$

where we use

$$\alpha(u_n = 1|\mathbf{x}) = p(u_n = 1) * f_n(\mathbf{x}),$$

with

$$f_n(\mathbf{x}) = 2 * \frac{< \mathbf{a}_n, \mathbf{x} >^{0.4}}{\max_{\hat{n}} < \mathbf{a}_{\hat{n}}, \mathbf{x} >},$$

where $\mathbf{a}_n$ is the n^{th} column of $\mathbf{A}$. The optimal non-linearity in $f_n(\mathbf{x})$ depends on the unknown distribution $p(\mathbf{u}|\mathbf{x}, \theta)$ and is therefore problem specific. The particular nonlinearity given above has been chosen empirically to give a small variance in the weights for the problem under study.

5.2.2 Calculating the Weights

Unfortunately $p(\hat{\mathbf{u}}_j|\mathbf{x}, \theta)$ in equation (5.4) can only be evaluated up to a normalising constant. Furthermore, evaluation of $p(\hat{\mathbf{u}}_j|\mathbf{x}, \theta)$ is computationally expensive. Equation (5.4) can, however, be replaced by:

$$\hat{w}_j = \frac{p(\mathbf{x}|\hat{\mathbf{s}}_j, \hat{\mathbf{u}}_j, \theta)p(\hat{\mathbf{s}}_j|\hat{\mathbf{u}}_j)p(\hat{\mathbf{u}}_j)}{p(\hat{\mathbf{s}}_j|\hat{\mathbf{u}}_j, \mathbf{x}, \theta)q(\hat{\mathbf{u}}_j)},$$

$$w_j = \frac{\hat{w}_j}{\sum_j \hat{w}_j}. \tag{5.6}$$

Unfortunately, normalising the weights introduces a bias in the gradient estimation. Nevertheless, $\sum_j \hat{w}_j$ converges to $p(\mathbf{x}|\theta)$ as $J \rightarrow \infty$ so that the gradient estimate is asymptotically (in the number of samples) unbiased.

5.2.3 Updating the Parameters

The parameters can be updated in each iteration using a gradient step. The approximations of the gradients are calculated using a Monte Carlo approximation using the weighted samples drawn with the importance sampling method. The update for the features $\mathbf{a}_k$ used in the shift-invariant sparse coding algorithm can be approximated as:

$$\{\mathbf{a}\}_{k\in\tilde{\mathcal{K}}}^{[r+1]} = \{\mathbf{a}\}_{k\in\tilde{\mathcal{K}}}^{[r]} + \nu\{\Delta\mathbf{a}_k\}_{k\in\tilde{\mathcal{K}}},$$

with

$$\{\Delta\mathbf{a}_k\}_{k\in\tilde{K}} = \{\sum_{i=1}^{I} w_i \sum_{l\in\overline{\mathcal{L}}} (\mathbf{x} - \mathbf{a}_{k,l}s_{k,l,j})s_{k,l,j}\}_{k\in\mathcal{K},l\in\overline{\mathcal{L}}}.$$

The inverse of the noise variance can be estimated using a gradient of the form:

$$\Delta\lambda_\epsilon = \sum_{j=1}^{J} w_j \left(\frac{M}{2\lambda_\epsilon} - \frac{1}{2}(\mathbf{x} - \mathbf{A}\mathbf{s}_j)^T(\mathbf{x} - \mathbf{A}\mathbf{s}_j)\right).$$

The inverse of the variance of the Gaussian in the mixture prior has a gradient approximation of:

$$\Delta\lambda_G = \sum_{j=1}^{J} w_j \left(\frac{\mathbf{u}_j^T\mathbf{u}_j}{2\lambda_G} - \frac{1}{2}\mathbf{s}_j^T\mathbf{s}_j\right)$$

and the parameter specifying the mixture hyper-prior has a gradient estimate of:

$$\Delta\lambda_u = \sum_{j=1}^{I} w_j \left(\frac{N}{2(1+e^{\frac{\lambda_u}{2}})} - \frac{1}{2}\mathbf{u}_j^T\mathbf{u}_j\right).$$

One issue has to be raised here; the issue of identifiability. With the model used here and in the next chapter, where a similar approach is taken, it is not clear, whether all the parameters for which update rules are given above can be identified uniquely. For example, two extreme signal models would be $\mathbf{x} = \epsilon$ and $\mathbf{x} = \mathbf{A}\mathbf{s}$. In the first case the noise variance parameter would be set to the variance of the observations and the hyper-parameter λ_u would be set so that the probability of setting any coefficient s_i to a non-zero value would be negligible. In the second case the noise variance would be set to zero with all the variation being explained by the variation in $\mathbf{s}$, which is possible when $\mathbf{A}$ spans the space of $\mathbf{x}$. However, whether the learning problem as stated above has a global maximum at parameters different from these extreme values is not immediately clear. This is an open problem and has not yet been investigated in full. Nevertheless, in the experiments reported in this thesis, we found that all parameters converged to some values different from the extreme points mentioned above. These points of convergence did depend on the starting values of the parameters, so that the learning problem seems to have multiple maxima. These issues are not fully investigated in this thesis as the main focus is on the convergence of the features, $\mathbf{a}_k$.

5.2.4 Implementation

An overview of the importance sampling method for shift-invariant feature learning is given in table 5.1.

5.3 Experimental Analysis

In this section we explore the performance of the importance sampling method when applied to the dictionary learning problem. The motivation for the introduction of the importance sampling method was to develop a method that is able to give a fast approximation of the gradient of the learning rule and that the possible increase in the variance of this estimate would have only a small effect over the many iterations required for the learning algorithm.

In order to evaluate the efficiency of the algorithm for different dictionary sizes, a measure of this efficiency is required. An estimate of the time required to calculate a gradient estimate with a certain accuracy, i.e. of approximating the gradient with a certain variance, cannot be used as a measure, as the idea behind the introduction of an importance sampling method is based on increasing this variance for each gradient estimate. Furthermore, the different prior formulations used with the different methods lead to different gradient estimates. It is therefore necessary to estimate the computation time required to calculate an estimate of the features to a certain accuracy. It was found that for the dictionary sizes

Table 5.1: Shift-invariant learning algorithm with the importance sampler.

```
   Input:
       User defined: signal $\{x_i\}$, number and length of $\mathbf{a}_k$,
                     $J, \zeta$ and $\nu$.
   Output:
       $\mathbf{A}$ and parameters $\lambda_u, \lambda_R$ and $\lambda_\epsilon$.
1  $\{\mathbf{a}_k\}_{k\in\mathcal{K}}$=random,
       $\mathcal{K}$ set of all feature indices
   random initialisation of $\lambda_u, \lambda_R$ and $\lambda_\epsilon$.
2  randomly select a data vector $\mathbf{x}$
3  for $j = 1, j < J$
           draw samples $u_{ij} \sim \alpha(u|\mathbf{x})$
           if $u_{ij} = 0$ draw samples $s_{ij} \sim \mathcal{N}$
           calculate weights $w_j$
4  Calculate the gradient for the dictionary elements $\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}}$
       $\{\Delta\mathbf{a}_k\}_{k\in K} = \{\sum_{j=1}^{J} w_j \sum_{l\in\mathcal{L}}(\mathbf{x} - \mathbf{a}_{k,l}s_{k,l,j})s_{k,l,j}\}_{k\in\mathcal{K},l\in\mathcal{L}}$
5  update $\{\mathbf{a}_{k,l}\}_{k\in\mathcal{K},l\in\mathcal{L}}$
       $\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r+1]} = (\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r]} + \nu^{[r]}\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}})/\|\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r]} + \nu^{[r]}\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}}\|_2$
6  calculate the gradients of the parameters:
       $\Delta\lambda_\epsilon = \sum_{j=1}^{I} w_j \left(\frac{M}{2\lambda_\epsilon} - \frac{1}{2}(\mathbf{x} - \mathbf{A}\mathbf{s}_j)^T(\mathbf{x} - \mathbf{A}\mathbf{s}_j)\right),$
       $\Delta\lambda_u = \sum_{j=1}^{I} w_j \left(\frac{N}{2(1+e^{\frac{\lambda_u}{2}})} - \frac{1}{2}\mathbf{u}_j^T\mathbf{u}_j\right)$
       $\Delta\lambda_G = \sum_{j=1}^{I} w_j \left(\frac{\mathbf{u}_i^T\mathbf{u}_j}{2\lambda_G} - \frac{1}{2}(\mathbf{s}_j)^T(\mathbf{s}_j)\right)$
7  update the parameters:
       $\lambda_\epsilon = \lambda_\epsilon + \nu^{[r]}\Delta\lambda_\epsilon$
       $\lambda_u = \lambda_u + \nu^{[r]}\Delta\lambda_u$
       $\lambda_G = \lambda_G + \nu^{[r]}\Delta\lambda_G$
8  $\nu^{[r+1]} = \nu^{[r]}\zeta; \zeta < 1$
9  repeat from step 2 until convergence
```

considered in this section, all algorithms found approximations of the features with comparable L_2 distance from the true features after the same number of iterations. In order to compare the speed of the methods we therefore only estimate the time required with the different methods to compute a single iteration.

This was done by estimating the time required to calculate the MAP estimate using 1) the EM algorithm [68] discussed in chapter 4; and 2) a standard gradient descent algorithm. These results are then compared to the estimated time required to generate 100 samples and calculate the associated weights using the importance sampling method proposed in this chapter. The test data was generated using a dictionary of five different functions (and all their shifts). The function length L was set to $c * 32$ with $c \in \{1, 2, 3, 4, 5\}$. The sizes of the dictionaries $\mathbf{A}$ were then 64×480, 128×960, 192×1440, 256×1920 and 320×2400, i.e. both M and N were increased linearly. We further used $\lambda_\epsilon = 100$ and $\lambda_G = 1$ and set the average number of non-zero coefficients to 1% so that the average number of non-zero coefficients was also increased linearly. In this experiment we assumed that all model parameters including the dictionary were known.

Figure 5.1 gives the computation time (using Matlab on a Macintosh G4 1.42 GHz dual processor machine) in seconds (averaged over 10 runs) for the three algorithms and for the different dictionary sizes. The diamonds in figure 5.1 show the performance of the importance sampling method, the crosses are the measurements from the EM algorithm and the circles are the measurements from the gradient descent algorithm. The inner panel in figure 5.1 shows the graph zoomed in on the y-axis to reveal the difference between the gradient descent and the importance sampling algorithms.

In these experiments we did not take advantage of the structure of the shift-invariant sparse coding formulation (i.e for the system matrix with all shifts, most operations can be calculated efficiently using convolutions). The results are therefore applicable to general dictionaries. It is important to realise that the performance of all three algorithms depends on different parameters. Both the computation time for the EM algorithm and for the gradient descent algorithm depend on the stopping rule employed. We stopped both algorithms when the change in the optimised function was below 0.0001. The EM algorithm has a computational complexity of $\mathrm{O}(M^3)$, the gradient algorithm of $\mathrm{O}(MN)$ while the proposed sampling algorithm only relies on the average number of non-zero components in the

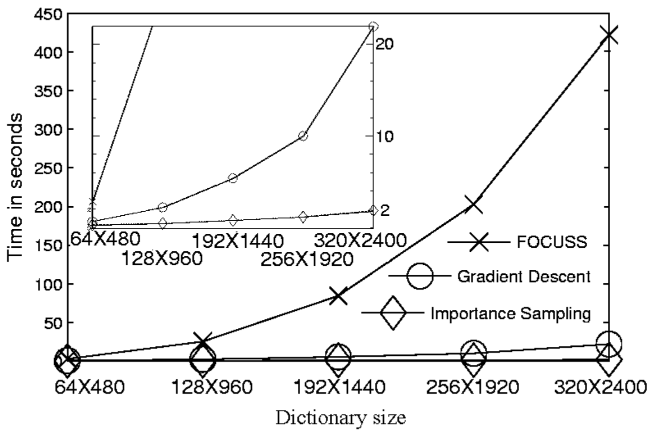

Figure 5.1: Computation time for the different algorithms for different dictionary sizes. Top left panel shows plot zoomed in on y-axis.

samples and is therefore dependent on the prior. For large problems, as investigated in the next chapters, the dominating computational burden is associated with sampling of $\mathbf{u}$ together with the search for the non-zero values.

Conclusions

The stochastic gradient descent learning rule used for sparse coding requires the repeated estimation of the gradient. This estimate is here only calculated with respect to a randomly selected single data vector. As this estimate is averaged over a large number of data-points, each gradient estimate has a large variance. The importance sampling algorithm in this section was developed as a fast method to estimate this gradient. This estimate has a larger variance than the previous one, however, the speed advantage allows for a much faster evaluation. For the problem studied here, the overall learning performance was equivalent between the algorithms for the same number of iterations. The speed advantage of the importance sampling method furthermore allows the use of more data-points and can therefore further reduce the overall variance. However, the used importance sampling method did not offer an unbiased estimate of the gradient. In chapter 7, in which we compare the importance sampler to the other methods developed in this thesis, we show that this bias becomes significant for larger dictionary sizes.

Chapter 6

Gibbs Sampling Approximation[1]

For many instruments such as the piano, we know that the excitations of the sound producing part of the instrument, e.g. the strings in the piano, are always in the same direction. In the piano, the hammer that hits the strings always moves the string in the same direction before the string oscillates on its own. Therefore, the excitation of the first half cycle of the recording of a piano note is also always in the same direction. For our model this means that for such instruments, the coefficients $\mathbf{s}$ always have the same sign. This additional knowledge can be incorporated into the model by specifying a non-negative prior distribution for these $\mathbf{s}$.

Such a prior formulation is introduced in this chapter. Again, a mixture distribution is used, where a delta function models the probability mass at zero. The non-zero coefficients are described by a modified Rayleigh distribution, which only has probability mass for all values greater than zero. To calculate approximations to the learning rule and to evaluate approximations to the gradient required for the estimation of other parameters, we introduce a Gibbs sampling method.

The Gibbs sampler offers potentially high performance, however, the high dimension of the distribution to be sampled from requires further thought. A modification of the subset selection procedure introduced previously is described, which keeps the asymptotic convergence of the Gibbs sampler and offers good performance in practice.

[1]The work presented here is based on [11]

6.1 Model Formulation

Again we consider the problem of estimating the learning rule:

$$\Delta a_{kp} = \frac{1}{\sigma_\epsilon^2} \left\langle \sum_m \epsilon_m s_{k,p-m} \right\rangle_{p(\mathbf{s}|\mathbf{A},\mathbf{x})}.$$

In this chapter we develop and analyse Markov chain Monte Carlo approximations of the integral in this learning rule. First we develop a Gibbs sampling strategy to draw samples from the posterior $p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \mathbf{A})$, where we again use a mixture prior. However, we introduce a novel distribution for the non-zero coefficients that differs from the one used in the previous chapter.

6.1.1 Sparse Coding with the Gibbs Sampler

In this chapter we again impose the following mixture prior given in equation (5.1) and equation (5.2) in order to enforce sparsity of the coefficients $\mathbf{s}$:

$$p(\mathbf{s}|\mathbf{u}) = \prod_n p(s_n|u_n) = \prod_n (u_n p(s) + (1 - u_n)\delta_0(s_n)),$$

where u_n is a binary indicator variable with discrete distribution:

$$p(u_n) = \frac{1}{1 + e^{-\frac{\lambda_u}{2}}} e^{-\frac{\lambda_u}{2} u_n}$$

and $\delta_0(s_n)$ is the Dirac mass at zero. Contrary to the model in the previous chapter, the distribution $p(s)$ is not necessarily assumed to be Gaussian. The Gibbs sampler introduced here can also be used if this distribution is uniform over some interval or has the distribution of the modified Rayleigh distribution introduced below.

In [43, 105] and [126] two Gibbs sampling algorithms [115] were proposed to solve the problem of learning an over-complete dictionary matrix $\mathbf{A}$ for sparse signal representations. Similar methods were previously suggested in [132, 25, 88, 52, 2]. These methods are based on a mixture prior similar to the one introduced above, but in [43, 105], the mixture was a mixture of Gaussians. A related approach is the method in [152, 153], which in addition modelled the relationships between the coefficients $\mathbf{s}$.

A connection between indicator variable methods as discussed here and model order selection is given in [45].

For the model discussed here, different implementations are possible in order to draw samples from the model. For a mixture of Gaussians it is possible to draw samples $p(u_n|u_{\hat{n}\neq n}, \mathbf{s}, \mathbf{x}, \theta)$ and $p(\mathbf{s}_n|s_{\hat{n}\neq n}\mathbf{u}, \mathbf{x}, \theta)$ [89], i.e. by standard Gibbs sampling from the conditional densities, where the subscript notation $\hat{n} \neq n$ refers to quantities with subscripts other than n. The problem with this method is that for mixtures of Gaussians in which each Gaussian has very different variances, the chain seldom switches states [43]. An extreme case would be the mixture of a Gaussian and a delta function used in the previous chapter, in which, whenever s_n is non-zero, the chain is not able to change the variable u_n, as such a change would have zero probability. In order to overcome this problem it is possible to sample from $p(u_n|u_{\hat{n}\neq n}, \mathbf{x}, \theta)$ [43, 105], i.e. by integrating out the coefficients $\mathbf{s}$. The distribution $p(u_n|u_{\hat{n}\neq n}, \mathbf{x}, \theta)$ is calculated as:

$$p(u_n|u_{\hat{n}\neq n}, \mathbf{x}, \theta) \propto p(u_n|u_{\hat{n}\neq n}) \int p(\mathbf{x}|\mathbf{s}, \mathbf{u}, \theta) p(\mathbf{s}|\mathbf{u}, \theta)\, d\mathbf{s}.$$

However, the evaluation of this distribution involves matrix inversion. This can be avoided by only integrating out a single coefficient s_n, i.e by sampling from

$$p(u_n|s_{\hat{n}\neq n}, u_{\hat{n}\neq n}, \mathbf{x}, \theta) \propto p(u_n|u_{\hat{n}\neq n}) \int p(\mathbf{x}|\mathbf{s}, \mathbf{u}, \theta) p(s_n|\mathbf{u}, \theta)\, ds_n. \quad (6.1)$$

After sampling of the indicator variable from either $p(u_n|u_{\hat{n}\neq n}, \mathbf{x}, \theta)$ or $p(u_n|s_{\hat{n}\neq n}, u_{\hat{n}\neq n}, \mathbf{x}, \theta)$ it is then easy to sample from $p(\mathbf{s}|\mathbf{u}, \mathbf{x}, \theta)$ or alternatively from $p(s_n \hat{n} \neq n, u_n, \mathbf{x}, \theta)$ as these distributions are either Gaussians or delta functions with parameters that are easily calculated.

An extension combining both methods can be developed by integrating out any set of coefficients s_n in each step. This would then lead to a block Gibbs sampler. The drawback, however, of both the first method and the proposed block method is the required evaluation of a full covariance matrix in the calculations. This problem does not arise in the second method, as all calculations can be done with univariate distributions. We therefore concentrate here on the second approach. The marginalisation method has another advantage as discussed in [115] as Rao-Blackwellisation. Marginalisation does reduce the variance of the chain and is therefore desirable.

In order to use this strategy, it is beneficial to choose a prior distribution which facilitates this integration. The mixture of Gaussian and delta function in the previous chapter as well as a mixture of Gaussians or a mixture of a delta and a uniform distribution can be used. The non-negative mixture prior developed in this chapter also allows this integration to be performed analytically.

If we integrate out a single coefficient, we set variable $u_n = 1$ with probability:

$$P(u_n = 1|s_{\hat{n}\neq n}, \mathbf{x}, \theta) = \frac{p(u_n = 1|s_{\hat{n}\neq n}, \mathbf{x}, \theta)}{\sum_{k=0}^{1} p(u_n = k|s_{\hat{n}\neq n}, \mathbf{x}, \theta)} = \frac{1}{1 + e^{-E_1}}, \tag{6.2}$$

where

$$E_1 = \log \frac{p(u_n = 1|s_{\hat{n}\neq n}, \mathbf{x}, \theta)}{p(u_n = 0|s_{\hat{n}\neq n}, \mathbf{x}, \theta)}. \tag{6.3}$$

So we only have to evaluate the logarithm of the ratio of the distributions such that the conditional distributions have to be known only up to a normalising term. The calculations for the other methods are similar, with the conditional distributions replaced accordingly.

6.1.2 Non-Negative Prior Formulation

The Bayesian paradigm allows for an easy incorporation of further constraints both on the parameters θ as well as on the coefficients $\mathbf{s}$. This is done by specifying a prior distribution that enforces these constraints. In this section we study one possible example of such an additional constraint; the positivity of the coefficients $\mathbf{s}$. To model the sparseness and independence assumptions on $\mathbf{s}$ we also use the factorial mixture prior for $p(\mathbf{s})$ of the form $\prod p(s_n|u_n)p(u_n)$, where u_n are binary indicator variables, $p(s_n|u_n = 0) = \delta_{s_n}(0)$, i.e. a mass at zero and $p(s_n|u_n = 1)$ is a positive distribution which is specified below. The factorial prior used reflects the prior belief that the coefficients $\mathbf{s}$ are independent a priori. This assumption can again be relaxed if certain prior dependencies can be assumed for a problem at hand. However, the derivation of the following algorithm

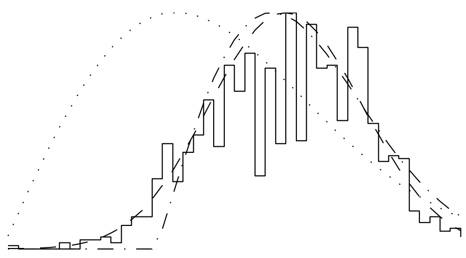

Figure 6.1: Histogram of MIDI note velocities (solid) versus the modified Rayleigh distribution (dashed). Also shown are an unshifted Rayleigh distribution (dotted) and a shifted Rayleigh distribution (dash dotted).

then becomes slightly more involved and the computational burden might increase.

The observation noise ϵ is again assumed to be i.i.d. Gaussian. (The extension to coloured noise is possible, however, many of the computational advantages of the algorithms discussed do not apply in this case.)

Positivity of the coefficients $\mathbf{s}$ can be enforced by restricting the prior distribution for the s_n to $\mathbb{R}^+$. Here we propose the use of a modified Rayleigh distribution. The use of this distribution is motivated by the application to piano music analysis that is studied in this thesis.

The physical mechanism in a piano always excites the piano strings in the same direction such that the first excursion of the observed waveform of a piano note is also always in the same direction. This means that the coefficients $\mathbf{s}$ always have the same sign. As the note prototypes $\mathbf{a}_k$ and the coefficients $\mathbf{s}$ can be inverted together without changing the reconstruction we can, without loss of generality, assume $\mathbf{s}$ to be non-negative. Furthermore, in most music performances notes are played at similar amplitudes - otherwise louder notes would overshadow quieter ones, and these would then be inaudible. These considerations lead us to propose the distribution for non-zero coefficients $\mathbf{s}$ described below.

This argument can be strengthened by comparing the modified Rayleigh distribution to the histogram of note amplitudes, which is done in figure 6.1. Here we show the histogram of note amplitude as recorded from the velocity value of a MIDI keyboard, i.e. an electronic keyboard which records the velocity with which keys are pressed during a musical performance. The histogram here shows the velocity values for the notes of a performance of Ludwig van Beethoven's Bagatelle No. 1 Opus 33. The dashed line in this figure is the graph of a modified Rayleigh distribution defined formally below. For comparison, we also show the standard Rayleigh distribution (which is also known as a square-root inverted Gamma distribution) with the dotted line and a shifted version of this Rayleigh distribution with the dash dotted line.

It is clear that the modified Rayleigh distribution fits the estimated distribution of the note activations better than the other two distributions. For other data such as biomedical time-series, other positive distributions for the non-zero coefficients might be more appropriate. For example, the modified Rayleigh distribution can be replaced by a zero mean Gaussian distribution restricted to positive values or by a uniform distribution over some positive interval. Both of these distributions can be used in the Gibbs sampler developed below. For these well known distributions the derivation of the required terms is relatively easy. We therefore concentrate on the presentation of the derivation of the algorithm for the more complicated modified Rayleigh distribution.

The Rayleigh distribution is given as:

$$p_R(s;\sigma_R^2) = \frac{1}{\sigma_R^2} s e^{-s^2/2\sigma_R^2}$$

for $s > 0$ and zero otherwise. This distribution is a special case of the inverted square-root gamma distribution. This distribution can be easily extended to allow for a shift parameter μ and is then:

$$p_R(s;\sigma_R^2) = \frac{1}{\sigma_R^2}(s-\mu)e^{-(s-\mu)^2/2\sigma_R^2}$$

for $s > \mu$ and zero otherwise. However, this distribution is zero for all values smaller than μ. In the problem studied here this is not desired. We therefore introduce a modification of the above distribution, which we call modified Rayleigh distribution in this thesis and define as:

$$p_{mR}(s;\mu\sigma_R^2) = \frac{1}{Z_{mR}}(s)e^{-(s-\mu)^2/2\sigma_{mR}^2}$$

for $s > 0$ and zero otherwise. Note that this distribution is nonzero for all positive values of s. An example of this distribution is shown in figure 6.1

(dashed line). The normalising constant for this distribution is:

$$Z_{mR} = \sigma_{mR} e^{-(\mu)^2/2\sigma_{mR}^2} + 0.5\mu\sqrt{2\pi\sigma_{mR}^2}(1 + \operatorname{erf}(\frac{\mu}{\sqrt{2\sigma_{mR}^2}})), \qquad (6.4)$$

where erf$(\cdot)$ is the error function.

6.2 Algorithm

6.2.1 The Gibbs Sampler with the Modified Rayleigh Distribution

It is interesting to note that the modified Rayleigh distribution is a conjugate prior for the Gaussian mean so that the posterior of the Gaussian mean is also a modified Rayleigh distribution. Therefore, the integral in equation (6.1) is still tractable analytically.

If we use $p(s_n) = p_{mR}(s; \mu_n, \lambda_R^{-1})$, the expression for E_1 in equation (6.2) becomes:

$$E_1 = -\frac{\lambda_{u_n}}{2} + \frac{\lambda_{E_n}}{2} b_n^2 + \ln \Phi,$$

where

$$\Phi = \frac{Z_E}{Z_p} \left[\frac{1}{\Psi_n} e^{-0.5\eta^2\Psi_n} + 0.5\eta\sqrt{\frac{2\pi}{\Psi_n}} \left(1 + \operatorname{erf}\left(\eta\sqrt{\frac{\Psi}{2}} \right) \right) \right]$$

with

$$Z_E = e^{-0.5(-\eta^2\Psi_n + b_n^2\lambda_{E_n} + \mu_n^2\lambda_R)}$$

and

$$Z_p = \lambda_R^{-1} e^{-\mu_n^2 0.5\lambda_R} + 0.5\mu_n\sqrt{2\pi\lambda_R^{-1}} \left(1 + \operatorname{erf}\left(\mu_n\sqrt{0.5\lambda_R} \right) \right).$$

The derivation of the above expressions is given in appendix A. η_n and $\frac{1}{\Psi_n}$ are the parameters of the posterior $p(s_n|s_{\hat{n}\neq n}, u_n = 1, \theta)$, which due to the conjugate prior is also of the modified Rayleigh form. The parameters are given analytically as:

$$\eta_n = \frac{\lambda_{E_n} b_n + \lambda_R \mu_n}{\lambda_{E_n} + \lambda_R}$$

and

$$\Psi_n = \lambda_{E_n} + \lambda_R$$

Here we have used the notation $\lambda_{E_n} = \|\mathbf{A}_n\|^2\lambda_\epsilon$ and $b_n = \frac{\mathbf{A}_n^T(\mathbf{I}-\mathbf{A}\S_{n=0})}{\|\mathbf{A}_n\|^2}$ where $\mathbf{A}_n$ is the nth column of the matrix $\mathbf{A}$ and λ_ϵ is the inverse of the variance of the likelihood.

To update the parameters we approximate the gradient of the marginal log likelihood using Monte Carlo integration. For a general parameter θ this gradient estimate is again:

$$\Delta\theta \propto \sum_j \frac{\partial}{\partial\theta} p(\mathbf{x}, \mathbf{s}_j, \mathbf{u}_j|\theta) p(\mathbf{s}_j, \mathbf{u}_j|\theta, \mathbf{x}).$$

Using the notation

$$\Delta_j\theta = \frac{\partial}{\partial\theta} p(\mathbf{x}, \mathbf{s}_j, \mathbf{u}_j|\theta) p(\mathbf{s}_j, \mathbf{u}_j|\theta, \mathbf{x})$$

and replacing θ by the parameters of interest we get:

$$\Delta_j\lambda_R = -0.5 \sum_{s_{jn}\neq 0} (s_{jn} - \mu)^2 - \frac{U}{c_1}(-0.5\mu\lambda_R^{-1}c_2 - c_3\lambda_R^{-2}),$$

where the sum is over the non-zero s_{jn}, U is the number of the non-zero s_{jn}, $c_1 = \mu c_2 + \lambda_R^{-1}c_3$, $c_2 = 0.5\sqrt{2\pi\lambda_R^{-1}}(1 + \mathrm{erf}(\mu\sqrt{0.5\lambda_R}))$ and $c_3 = e^{-0.5\lambda_R\mu^2}$,

$$\Delta_j\lambda_\epsilon = \lambda_\epsilon^{-1} - \frac{(\mathbf{x} - \mathbf{A}\mathbf{s}_j)^2}{\mu},$$

$$\Delta_j\lambda_u = \frac{1}{1 + e^{0.5\lambda_u}} - \frac{U}{N}$$

and

$$\Delta_j\mu = \sum \lambda_R(s_{jn} - \mu) - \frac{U}{c_1}c_3,$$

where the summation is again only over the non-zero s_{jn}.

6.2.2 Sampling from the Modified Rayleigh Distribution

In the above sampling scheme it is necessary to draw samples from the posterior distribution of s_n conditioned upon the other s. As the modified Rayleigh distribution is a conjugate prior for the Gaussian mean, this distribution is also of the modified Rayleigh form with the parameters given above. It is therefore necessary to implement a method that allows us to draw samples from this distribution for different parameters. Due to the non-standard form of this distribution a direct sampling scheme is not obvious. It is instructive to write the modified Rayleigh distribution in the form:

$$\frac{1}{Z_{mR}} s e^{-(s-\mu)^2/2\sigma_R^2} = \frac{1}{Z_{mR}} \left((s-\mu) e^{-(s-\mu)^2/2\sigma_R^2} + \mu e^{-(s-\mu)^2/2\sigma_R^2} \right). \quad (6.5)$$

For $s > \mu$ and $\mu > 0$, the modified Rayleigh distribution can be understood as a mixture distribution of a Gaussian and a shifted Rayleigh distribution. However, the modified Rayleigh distribution is defined for values greater than zero, while the shifted Rayleigh distribution is defined for values greater than μ, as it would be negative for values smaller than μ. We therefore propose a hybrid sampling strategy if $\mu > 0$, which first determines whether the value is greater or smaller than μ. We have

$$p(s > \mu) = p_1 = \frac{1}{Z_{mR}} (\sigma_{mR} + 0.5\mu\sqrt{2\pi\sigma_{mR}})$$

and

$$p(s < \mu) = p_2 = 1 - p_1.$$

With probability p_1, $s > \mu$ which means that we can sample from:

$$\begin{aligned} p(s|s > \mu) = \mu &+ \left[\frac{\sigma_{mR}}{Z_{mR}}\right] \sigma_{mR}^{-1}(s) e^{-0.5\sigma_{mR}^{-1}s^2} \\ &+ \left[0.5\frac{\mu}{Z_{mR}}\sqrt{2\pi\sigma_{mR}}\right] 2\sqrt{\frac{\sigma_{mR}^{-1}}{2\pi}} e^{-0.5\sigma_{mR}^{-1}s^2}, \end{aligned}$$

which is a mixture of a rectified Gaussian and a shifted Rayleigh distribution with mixing probabilities given in the square brackets. For $s < \mu$ we know that the distribution is bounded from above by

$$\frac{1}{Z_{mR}} \mu e^{-(s-\mu)^2/2\sigma_R}$$

as

$$\frac{1}{Z_{mR}} (s-\mu) e^{-(s-\mu)^2/2\sigma_R}$$

is negative. We can therefore use a simple rejection sampler to draw samples for $0 < s < \mu$ when $\mu > 0$.

If $\mu < 0$ we see from equation (6.5) that the second term becomes negative while the first term is positive for all $s > 0$. The distribution is then bounded from above by a shifted Rayleigh distribution which can be used for rejection sampling. This might not be a good strategy in general as it can lead to a very high rejection rate for certain parameter values. For our experiments, however, this was found to be of no great concern.

6.2.3 Random Subset Selection

The probabilistic model used in this chapter leads to a posterior for $\mathbf{u}$ that is multi-modal [2]. Furthermore, for many of the problems of interest here the dimension of this state space is very high. As the sampler has to be able to draw samples from often far apart modes associated with much of the probability mass and as the states between these modes have often very low probability, we generally require a large number of samples to be drawn.

In order to improve the Gibbs sampler performance different approaches could be adopted. We tried several methods, but most of these did not offer significant advantages. However, as these methods can be of interest for related applications, they have been included in appendix B. In order to significantly reduce the computational requirements we instead resort to subset selection. For example, the subset selection step introduced in chapter 4 could be used and we found this to work well in practice, however, the Gibbs sampler then does not have the chance to explore certain parts of the distribution.

An improvement on the deterministic subset selection procedure, which asymptotically explores the full probability space and therefore asymptotically draws samples from the correct distribution, is to use a random subset selection during each Gibbs cycle. The method that we found worked best for the problems under study used a combination of both approaches. We used the subset selection algorithm to select a fixed subset at the beginning of each Gibbs run. In each Gibbs cycle we then

[2]We use the term multi-modal here for discrete state spaces. In order to be able to talk about modes, we need to define a neighbourhood for each point. In the discrete state space used here, such a neighbourhood can be defined based on Levenshtein distance (more commonly known as edit distance). With such a definition, points that differ from any point by only a single indicator variable u_n constitute its neighbourhood. A mode is then a point with higher probability than all its neighbours.

added a random selection of further features to this initial set. In each Gibbs step we sample from $p(u_n|s_{\hat{n}\neq n}, u_{\hat{n}\neq n}, \mathbf{x}, \theta)$ and $p(s_n|s_{\hat{n}\neq n}, \mathbf{u}, \mathbf{x}, \theta)$. The order in which we sample from these distributions can be chosen at random and it is not necessary to cycle through all n before returning to any one coefficient. The sampler is still guaranteed to converge to the stationary distribution if we ensure that we sample from each coefficient with non-zero probability. The random subset selection method can be seen as a way to specify a random set of subscripts n. If we combine the random subset selection method with the fixed subset selection as proposed here, we use a fixed set of subscripts, say $\{n_1, n_2, \cdots, n_{W_1}\}$. In each cycle of the sampler we further select a random set of indices from the remaining set, i.e. $\{n_{W_1+1}, n_{W_1+2}, \cdots, n_{W_1+W_2}\}$. Each Gibbs cycle than samples from $p(u_{n_k}|s_{\hat{n}\neq n_k}, u_{\hat{n}\neq n_k}, \mathbf{x}, \theta)$ and $p(s_{n_k}|s_{\hat{n}\neq n_k}, \mathbf{u}, \mathbf{x}, \theta)$ where n_k is a random permutation of the indices n_1 to $n_{W_1+W_2}$. This method ensures that a fixed region of the probability space was explored by the sampler, but enables the sampler to asymptotically explore the complete space.

6.2.4 Convergence

A Markov chain requires several steps before the samples can be assumed to be generated from the stationary distribution and a major problem is to assess after how many samples the sampler has achieved this convergence. As there are no general numerical methods to assess convergence, convergence is normally monitored graphically by plotting estimates of interest versus sample number, however, this method has its problems [115] and we found that for the application studied here such plots were often ambiguous.

As we are only able to draw a very small number of samples for the applications of interest, it was found that the sampler often converges to a single local maximum of the distribution and that it is rarely capable of escaping this local mode. Nevertheless, similar learning algorithms to the ones proposed in [103] and [80] and discussed in chapter 4 use even cruder approximations based on local maxima. It is therefore likely that the Gibbs sampler offers better approximations to the distributions than these methods. This is shown empirically in chapter 7 in which the Gibbs sampler outperforms the other approaches even with a small burn in period of only 50 samples.

In order to improve convergence, an annealing method as discussed in [98] could be used. This method has similarities with the methods in [100, 141]. Instead of starting the chain using kernels with the stationary distribution of interest, sampling can be started from a more dispersed distribution. This flatter distribution can be derived by raising the distribution of interest to a power $T < 1$. During the first samples, which are discarded anyway, the target distribution is annealed to the distribution of interest by increasing T gradually to 1. We found this method improved performance, but due to the difficulty of assessing convergence it is hard to give a conclusive evaluation.

6.3 Implementation

As with the importance sampling approach in the previous chapter, the Gibbs sampling method can be used with dictionaries other than the shift-invariant dictionary. Again, the only change is the use of a different learning rule. An overview of the learning algorithm based on the proposed Gibbs sampling method is given in table 6.1.

Conclusions

For piano music we assume that note waveforms follow a similar time-domain waveform for different renditions of the same note. Due to the physical generation of the note, the first excursion of this waveform is assumed to be always in the same direction. This means that the coefficients $\mathbf{s}$ can be restricted to be non-negative. This reasoning, as well as the observed statistics of the strength with which each note is played in a typical piano performance, lead us to the introduction of the non-negative mixture prior in this section.

We introduced a Gibbs sampling method to approximate the learning based on this prior. As one of the mixture components in this prior is

Table 6.1: Shift-invariant learning algorithm with a non-negative prior.

```
     Input:
         User defined: signal $\{x_i\}$, number and length of $\mathbf{a}_k$,
             $J, \zeta$ and $\nu$.
     Output:
         $\mathbf{A}$ and parameters $\lambda_u, \lambda_R, \lambda_\epsilon$ and $\mu$.
1    $\{\mathbf{a}_k\}_{k\in\mathcal{K}}$=random,
         $\mathcal{K}$ set of all feature indices
     random initialisation of $\lambda_u, \lambda_R, \lambda_\epsilon$ and $\mu$
2    randomly select a data vector $\mathbf{x}$
3    calculate inner product between $\mathbf{x}$ and all shifted features
         $\{d_{k,l}\}_{k\in\mathcal{K},l\in\mathcal{L}} =< \mathbf{x}, \{\mathbf{a}_{k,l}\}_{k\in K,l\in L} >$
         $\mathcal{K}$ is the set of all features
         $\mathcal{L}$ is the set of all shifts.
4    draw samples $\mathbf{s}(j)$ using the Gibbs sampler. In each Gibbs cycle
     only use a randomly selected subset of features where the probability
     of using any feature depends on $\{d_{k,l}\}$
5    Calculate the gradient for the dictionary elements $\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}}$
         $\{\Delta\mathbf{a}_k\}_{k\in K} = \{\sum_{j=1}^{J}\sum_{l\in\mathcal{L}}(\mathbf{x} - \mathbf{a}_{k,l}s_{k,l,j})s_{k,l,j}\}_{k\in\mathcal{K},l\in\mathcal{L}}$
6    update $\{\mathbf{a}_{k,l}\}_{k\in\mathcal{K},l\in\mathcal{L}}$
         $\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r+1]} = (\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r]} + \nu^{[r]}\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}})/\|\{\mathbf{a}_k\}_{k\in\mathcal{K}}^{[r]} + \nu^{[r]}\{\Delta\mathbf{a}_k\}_{k\in\mathcal{K}}\|_2$
7    calculate the gradients of the parameters:
         $\Delta\lambda_\epsilon = \lambda_\epsilon^{-1} - \frac{(\mathbf{x}-\mathbf{As})^2}{\mu}$,
         $\Delta\lambda_u = \frac{1}{1+e^{0.5\lambda_u}} - \frac{U}{N}$
         $\Delta\mu = \sum\lambda_R(s_n - \mu) - \frac{U}{c_1}c_3$
         $\Delta\lambda_R = -0.5\sum_{s_n\neq 0}(s_n - \mu)^2 - \frac{U}{c_1}(-0.5\mu\lambda_R^{-1}c_2 - c_3\lambda_R^{-2})$
8    update the parameters:
         $\lambda_\epsilon = \lambda_\epsilon + \nu^{[r]}\Delta\lambda_\epsilon$
         $\lambda_u = \lambda_u + \nu^{[r]}\Delta\lambda_u$
         $\mu = \mu + \nu^{[r]}\Delta\mu$
         $\lambda_R = \lambda_R + \nu^{[r]}\Delta\lambda_R$
9    $\nu^{[r+1]} = \nu^{[r]}\zeta; \zeta < 1$
10   repeat from step 2 until convergence
```

a modified version of the Rayleigh distribution, we introduced a novel sampling strategy to draw samples from this distribution. The Gibbs sampler is relatively slow. To overcome this, we introduced a random subset selection step that only includes a small number of features in each Gibbs cycle. A comparison of the different methods introduced in this and the previous two chapters is presented in chapter 7.

Part III

Experimental Studies

Chapter 7

Comparative Study

In the previous three chapters we have developed three different approaches to the learning problem. The performance of these methods is evaluated in this chapter. To better assess the performance we used a simplified test signal that enabled us to analyse the influence of the choice of different parameters, assumptions and approximations on the results. This influence was evaluated in terms of the quality of the solution, the computational complexity and the convergence to local or global solutions. A better understanding of the behaviour of the algorithms developed in this thesis can then be used in the selection of a particular method for a particular application or problem. We also compare the performance of these algorithms to the Gibbs sampling method proposed in [126].

After introducing the simplified test signal in section 7.1, we analyse the learning performance of the methods in section 7.2 and compare the dictionary elements found with the different methods to the dictionary used to generate the test signal. Furthermore, we look at the learned model parameters and compare them to the estimates directly calculated from the true coefficients $\mathbf{s}$ used to generate the test signal.

In section 7.3 we analyse the representations $\mathbf{s}$ calculated with the different methods. To estimate the coefficients $\mathbf{s}$ we use the parameters and dictionary from which we generate the signal, as well as the learned parameters and dictionaries. These estimates are compared to the true sequence used to generate the signal. For the sampling based methods developed in this thesis we compare two different estimations of the representation, the MAP estimation of $\mathbf{s}$ and the estimated mean of $\mathbf{s}$.

In the final section of this chapter we compare the methods introduced in this thesis using a music analysis task. In this section we use the different algorithms to learn sets of piano notes from polyphonic piano recordings and concentrate on the differences and similarities in the learned features. A more detailed study of the musical structures found with one of the proposed methods is left for the subsequent chapters in this thesis.

7.1 Methodology

In this section we compare the performance of the EM method discussed in chapter 4, the importance sampler derived in chapter 5 and the Gibbs sampler with the non-negative prior developed in chapter 6. Furthermore, we compare these methods to the Gibbs sampler developed in [126], which is similar to the Gibbs sampler developed in chapter 6, but with a Gaussian and delta mixture prior.

In order to compare the different methods, we generated a test signal with known properties. To be able to draw conclusions from these experiments that are valid for the real world problems of interest, the test signal should have many of the properties of these signals. To simplify analysis the test signal should also have a smaller number of parameters to be estimated. Furthermore, the properties of such a signal should be controllable so that the influence of different signal properties can be studied in more detail. These requirements were met by using the recorded performance information of a real piano performance of Ludwig van Beethoven's Bagatelle No. 1 Opus 33 (see [6] for more information about the data). This information included the strength with which each note was played, the length of each note, the pitch of each note and the timing of each note. To generate the test signal, we restricted all pitches played to one octave (i.e. the note pitches were mapped onto a single octave) and reduced the time scale. The relative timing, as well as the strength of the notes, was preserved. We then generated the signal using notes with a fixed length of 128 samples. These notes were generated using an FM synthesis technique and are shown in figure 7.1. The signal therefore followed the model

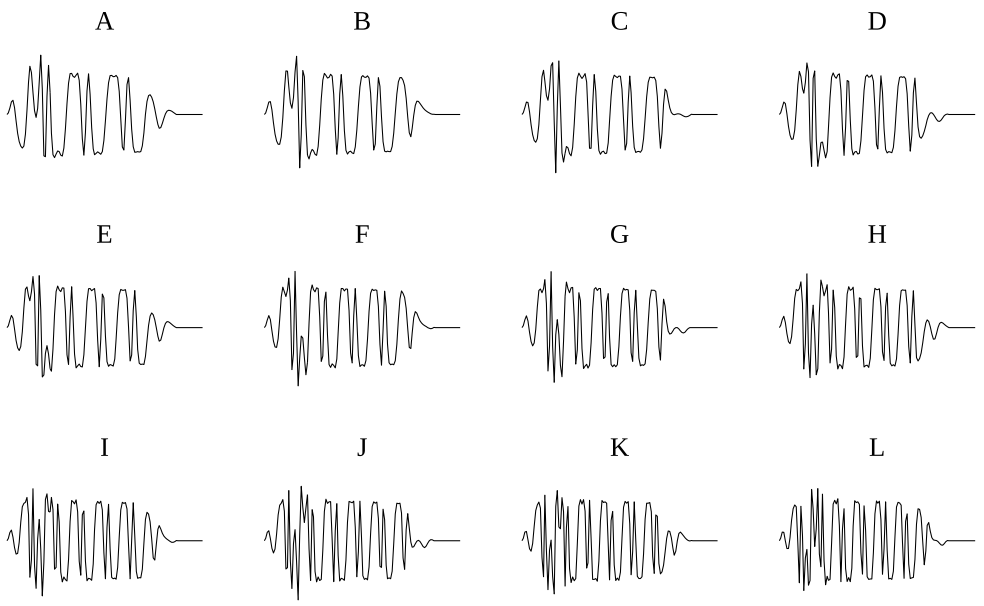

Figure 7.1: The twelve notes used to generate the test signal.

studied here in that it was generated from 12 waveforms at different locations and with different amplitudes. The location and the amplitudes of the waveforms had the same statistics as the original piano performance. However, we did not add noise to the model.

In all experiments we initialised the features in the matrix **A** with the same set of sinusoidal functions. The number of features to be learned was set to 12 with a length of 128 samples each. We then ran the experiments for 10 000 iterations, after which all methods had converged. The speed advantage of the importance sampling method allowed us to run this method for a further 1 000 000 iterations, however, no significant change in the results was observed after these additional iterations. As the importance sampling algorithm is much faster than the Gibbs sampler, all three Monte Carlo methods took roughly the same amount of time (ca. three days on a Macintosh G4 1.43 GHz computer).

For the EM method of chapter 4 we used 10 iterations for each EM step (this was found to be enough for the algorithm to converge sufficiently). In the subset selection step we set the maximally allowed overlap of one feature with a shifted version of itself to 50% and selected a maximum of 50 features for each data-block.

For the Monte Carlo methods we estimated the parameters with the maximum likelihood learning rules given in section 6.2 and subsection 5.2.3 respectively. Note that the learning rules for the Monte Carlo methods based on the Gaussian and delta mixture prior are similar and only differ in the use of the importance weights in the importance sampling estimate.

We used 100 samples in the importance sampler and the last ten samples of 50 for the Gibbs sampler. For both Gibbs samplers, the first sample in the chain was set to a vector of zeros.

It is important to stress that the small number of samples generated with the Markov chain sampling methods means that it cannot be assumed that the samples follow the exact distribution of interest. However, the computational burden is already very high with this small number of samples and an increase in the number of samples infeasible. Nevertheless, the results reported below show that the overall error introduced by the integral approximation was small, even with this small number of samples.

The importance sampling weight calculation introduces a bias, which, for the small number of samples used here, influences the results significantly. This bias decreases with the number of samples and to analyse this influence, we increased the number of samples used in a second experiment to 10 000. In this experiment we initialised the algorithm with the solution found in the previous experiment (which used 100 samples) and ran the method for 10 000 further iterations. With this increased number of samples the performance improved, however, the increase in the sample number also increased the computation time. The 10 000 iterations now took 5 days.

7.2 Learning Performance

7.2.1 Qualitative Evaluation

The 12 notes used to generate the test signal are shown in figure 7.1 and are also shown in figure 7.2 with dotted lines. In the same figure we show 10 of the features learned with the Gibbs sampler using the non-negativity constraint (solid lines). We assigned the features here depending on the

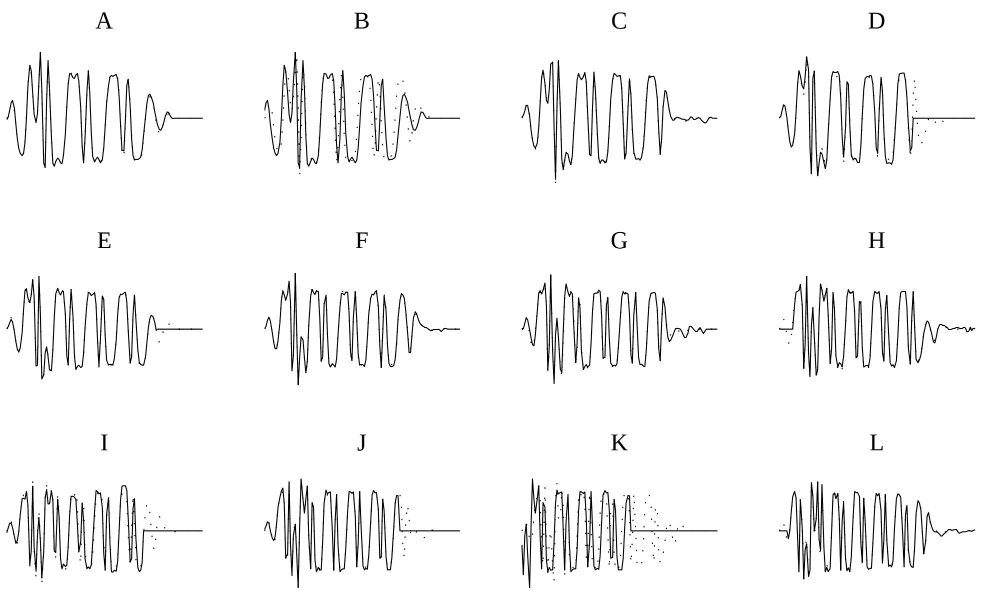

Figure 7.2: The twelve features learned with the Gibbs sampler and the non-negativity constraint. The dotted lines show the original notes.

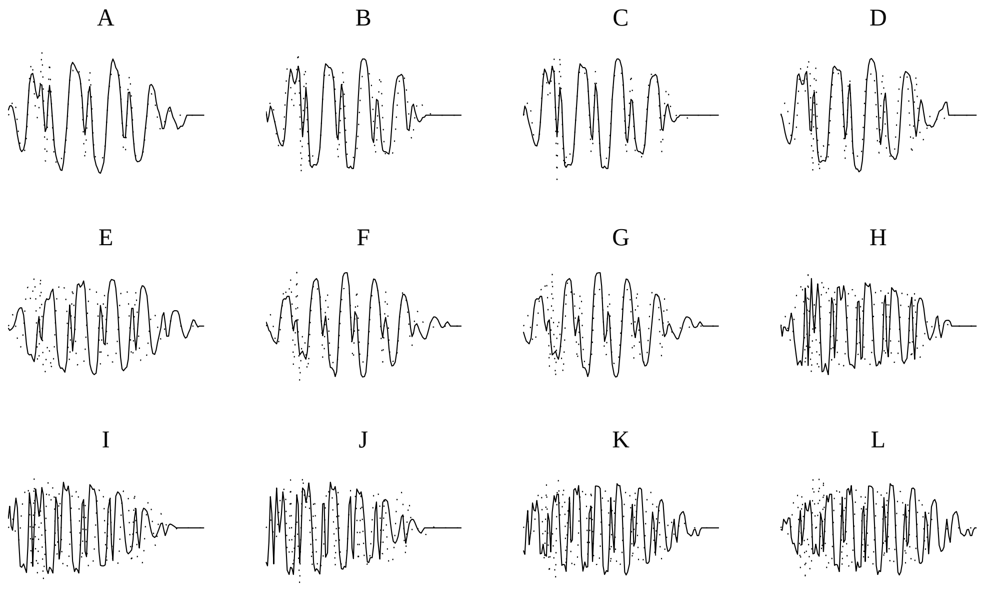

Figure 7.3: The twelve features learned with the importance sampler (b). The dotted lines show the original notes.

correlation between the learned features and the true features and took account of the different shifts of the learned features. As two of the original notes (B and K) did not have a high inner product with any of the learned

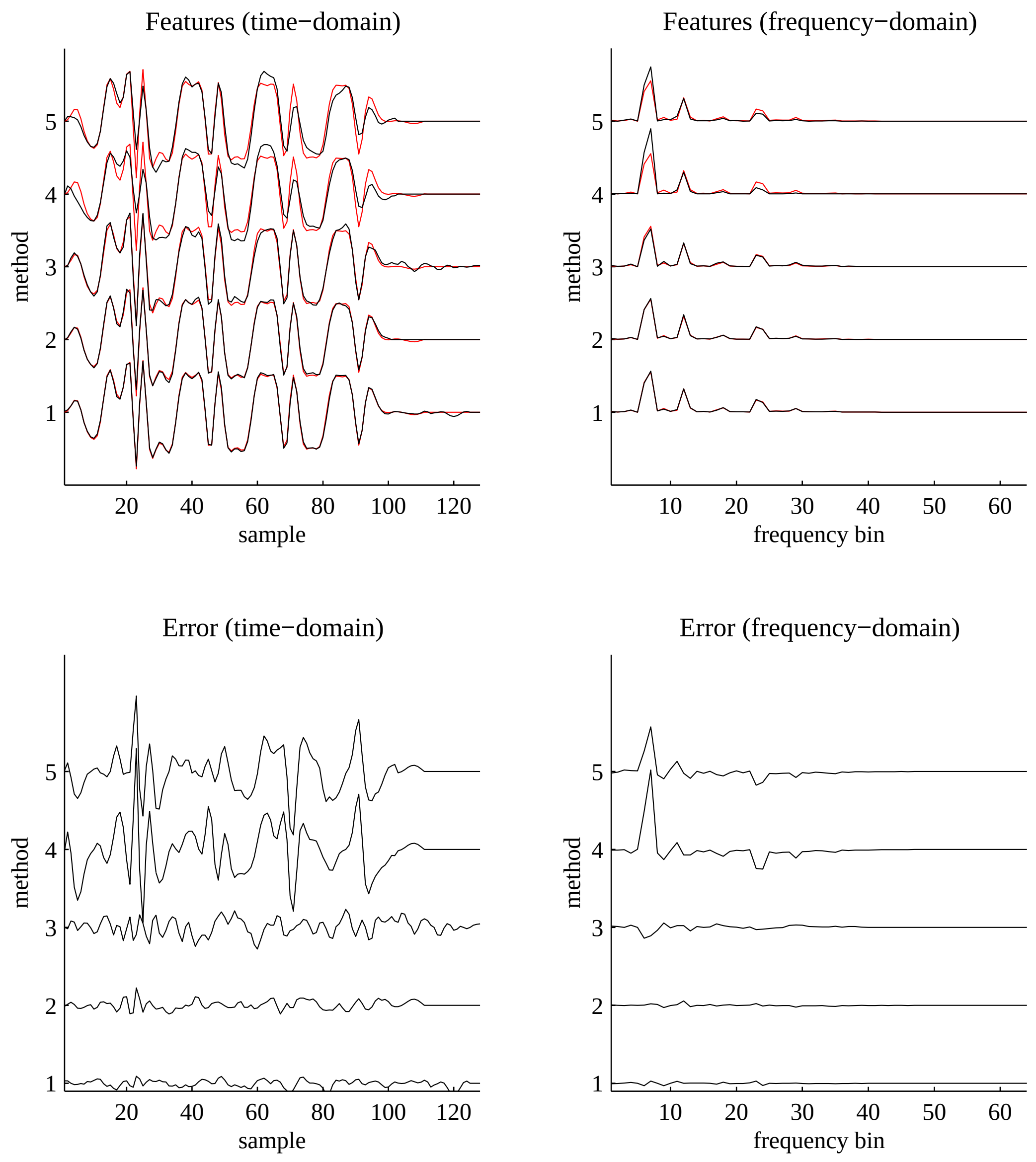

Figure 7.4: Learning performance comparison for note C. The features learned with the Gibbs sampler with the non-negativity constraint (1), the Gibbs sampler (2), the EM method (3), the importance sampler with 100 samples (4), and the importance sampler with 10 000 samples (5). The top left panel shows the features in the time domain with the true notes in red. The top right panel shows the power spectrum of the learned features, again with the true notes in red. The bottom left panel shows the difference between the learned and true notes in the time domain and the bottom right panel shows the difference of their power spectra.

features, we matched these up with the closest learned features, which are the features learned to model notes A and L. The similarity between most learned features and the true notes is evident.

Figure 7.3 compares the original notes to the features learned with the importance sampler using 100 samples. This time the true notes B and K are compared to the same learned features as notes C and L respectively. The features learned with the importance sampler are less similar to the true notes than the features learned with the Gibbs sampler using the non-negative prior. It is interesting to note that the features learned with the importance sampler have a smooth amplitude envelope. This seems to suggest that, during learning, the features were updated with notes at shifted positions where the shifts are multiples of the feature period as was discussed in section 3.3.1. This might be a result of the bias introduced in the normalisation of the weights as discussed below.

The difference in the learning performance can be clearly seen by looking at a single note. Figure 7.4 compares the time-domain and the power spectrum of note C to the closest features learned with all methods (The Gibbs sampler with the non-negative prior (1), the Gibbs sampler (2), the EM method (3) and the importance sampler with 100 (4) and 10 000 (5) samples). The top two panels show the features in the time-domain (left) and in the spectral-domain (right). The two panels at the bottom show the error between the learned and the true notes. It is again clear that the error is worse for the importance sampling method. We also see that the features learned with the importance sampling method have a stronger fundamental frequency but weaker upper harmonics. This is again an effect similar to the one discussed in section 3.3.1 and a result of the bias introduced in calculating the sample weights.

7.2.2 Quantitative Evaluation

A quantitative analysis of these results is given in table 7.1 where the inner products between the true notes and the closest learned features at the best shift are compared for all methods used. Here it is evident that the

Method	NN Gibbs	Gibbs	EM	Imp.100	Imp.10000
Note A	0.9980	0.9988	0.9962	0.9343	0.9779
Note B	0.7106	0.8241	0.7635	0.7693	0.7941
Note C	0.9986	0.9980	0.9980	0.9337	0.9280
Note D	0.9721	0.9879	0.9947	0.6786	0.6852
Note E	0.9958	0.9981	0.9971	0.9032	0.9265
Note F	0.9987	0.9974	0.9997	0.8509	0.8760
Note G	0.9957	0.9960	0.9980	0.6274	0.6149
Note H	0.9942	0.9960	0.9993	0.9757	0.9901
Note I	0.9102	0.4808	0.4891	0.5180	0.5201
Note J	0.9719	0.9966	0.9735	0.3453	0.3444
Note K	0.4836	0.5359	0.4901	0.5073	0.4859
Note L	0.9966	0.9904	0.9974	0.8995	0.9602
Total	11.0260	10.8010	10.6966	8.9433	9.1034

Table 7.1: Correlation between the learned features and the original notes.

Gibbs sampler with the non-negativity constraint offers the best performance. Interestingly, the EM method with a subset selection step offers nearly as good a performance as the Gibbs sampler with the Gaussian and delta mixture prior. This is somewhat surprising at first, as the MAP estimate used in this method gives a biased gradient estimate. However, the interpretation of this method as a joint estimation of the features and the coefficients $\mathbf{s}$ (as discussed in [57]) gives a statistical motivation for the use of this method. The fact that the Gibbs sampler with the non-negativity constraint outperforms these other methods can be assumed to be due to the prior distribution used, which more closely follows the actual distribution of the coefficients as shown in section 6.1. Interestingly the subset selection step does not seem to degrade the performance significantly.

The performance decrease for the importance sampling method is also clear from this table. The influence of the bias introduced by the required weight normalisation depends on the number of samples used. When the sample size is increased, the performance improves, nevertheless, even with 10 000 samples the importance sampler does not reach the performance of the other methods.

As is evident from table 7.1, not all notes were learned in the experiments. This seems to be due to local maxima in the marginal likelihood.

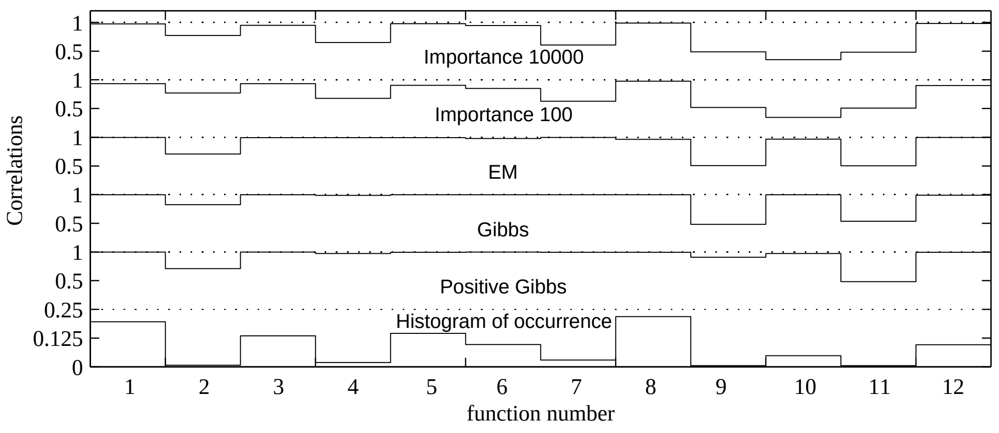

Figure 7.5: The correlation between the learned features and the original notes for the different methods and the histogram of the note occurrence in the training signal (bottom).

The notes which have no corresponding learned features are those which occur less frequently in the signal, as is clear from figure 7.5, where we compare the histogram of the occurrence of the notes in the signal to the correlation between the learned features and the original notes. It is evident that, independent from the method used, there exists a strong correlation between the note occurrence and the learning performance.

The overall results in table 7.1 are due to two distinct effects. On the one hand, the correlation between the true notes and the learned features varies between the different methods. The Gibbs sampler with the non-negativity constraint offered the best performance and the importance sampler offered the worst performance. On the other hand, the number of learned features also differed. The Gibbs sampler with the non-negativity constraint found all but two notes, whilst the importance sampling method again performs worst, with only six of the true notes having a correlation of more than 0.8 with any of the learned features. With all methods some notes have been learned more than once. With the EM method, the second note has been learned but is not very good. Running the algorithm for further 100 000 iterations did not improve the results and it appears that the non-convergence of some of the features is due to the convergence to local maxima in the likelihood function.

	λ_N	λ_R	λ_G	μ	λ_u
True	inf	21.1	1.54	0.73	14.25
Imp	185	n.a.	1.13	n.a.	15.07
NN	197.93	1.07	n.a.	0.99	13.44
Gibbs	436.46	n.a.	1.35	n.a.	14.82

Table 7.2: Estimation of the other model parameters compared for the different methods.

7.2.3 Parameter Estimation

In addition to the features $\mathbf{a}_k$, the proposed sampling methods calculate maximum likelihood estimates of some of the other parameters using stochastic gradient optimisation. These parameters include the noise variance, the prior parameters, λ_R or λ_G, the hyper prior parameter λ_u and, for the non-negative prior, μ. We used maximum likelihood methods to estimate these parameters from the true coefficient vector $\mathbf{s}$, which was used to generate the test signal. The true noise variance was not estimated as the signal was produced without added noise. Nevertheless, the algorithms developed in this thesis require the specification of a non-zero noise variance and in the experiments reported below the noise variance was set to 0.01 when the true parameters were used.

Table 7.2 compares the parameters estimated with the different approaches to the parameters estimated directly from the true coefficients $\mathbf{s}$. It is interesting to note that the Gibbs sampling method with the non-negative prior underestimated the prior parameters. There seem to be two reasons for this. The assumed noise term influenced the estimate, as the signal did not have added noise. Furthermore, it was found that the likelihood function did have several local maxima, so that the results found might correspond to such a local maximum. This was investigated by setting the parameters to the estimates found from the true parameters and running the algorithm for a further 10 000 iterations whilst keeping the dictionary fixed at the true values. In this experiment the λ_R value converged to 21, while the μ parameter converged to 0.45. The prior model that uses a mixture of a delta and a Gaussian did not suffer from these problems.

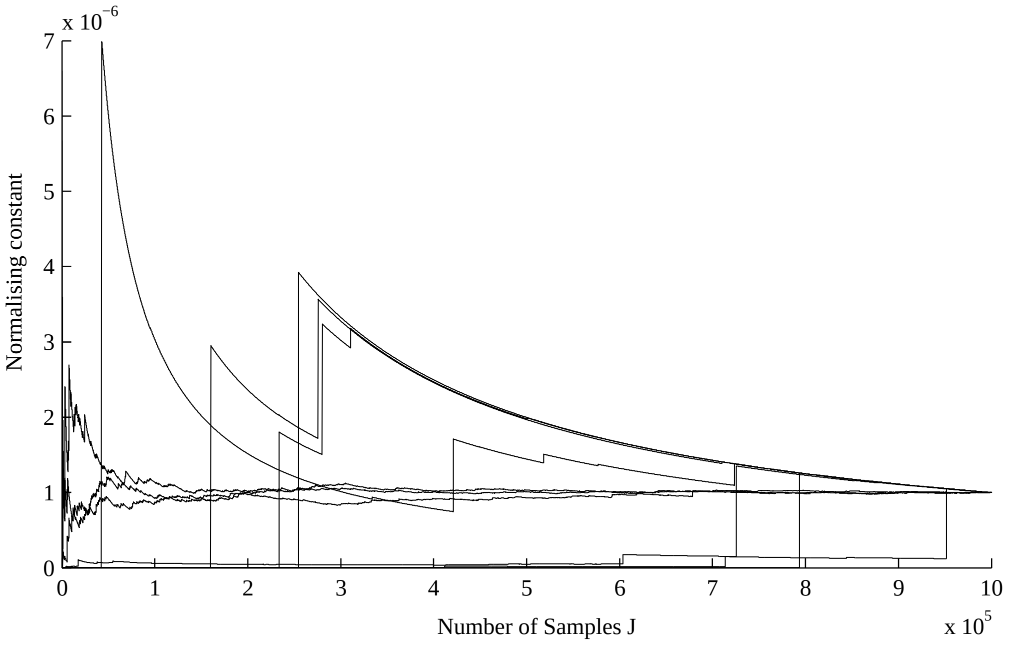

Figure 7.6: Convergence of the normalising constant estimate for the importance sampler plotted against the number of samples drawn. It is clear that at least 1 000 000 samples are required for the example used here to guarantee a good estimate.

7.2.4 Importance Sampler Convergence

To better understand the performance of the importance sampler and to see how many samples might be required for the importance sampler to significantly decrease the bias, we can monitor the convergence of the normalising constant $\sum_j \hat{w}_j$ in equation (5.6). Note that this quantity is an approximation of $\frac{1}{J}\sum_j \hat{w}_j \approx p(\mathbf{x}|\theta)$, i.e. the marginalised likelihood evaluated at the current data point and parameter values. In order to get an unbiased gradient estimate we need an accurate estimation of this quantity in the weight calculation of the importance sampler. To show the convergence of this estimate, we plot $\frac{1}{J}\sum_j \hat{w}_j$ against J (i.e. the number of samples used) in figure 7.6. We plot the estimate for ten different runs of the importance sampler. The jumps in the curves are due to samples drawn that fit the posterior well, leading to large weights. As most weights are very small the result is dominated by these few larger weights. For smaller dictionary sizes (as used in the comparison of chapter 5), the importance sampling method is a good and fast alternative. Unfortunately, for the problem of interest here, it can be seen that a good gradient estimate requires at least 1 000 000 samples, a prohibitively large number, which would make the method very slow, even when compared to the Gibbs sampling approach.

7.3 Representations

In this section we investigate the properties of the coefficients $\mathbf{s}$ found with the different algorithms and compare these representations to the coefficients $\mathbf{s}$ used to generate the data. We also analyse the signal approximation as measured by the L_2 norm of the reconstruction error.

The sampling methods enable us to calculate different estimates of $\mathbf{s}$. They can be used to calculate the sample mean but can also be used to estimate the MAP value by choosing the samples for which the posterior $p(\mathbf{u}|\mathbf{x}, \mathbf{A})$ is maximal. The coefficients $\mathbf{s}$ that maximise $p(\mathbf{s}|\mathbf{u}, \mathbf{x}, \mathbf{A})$ can then be found analytically. Another method of estimating the MAP with Markov chain methods is to use annealing techniques. However, we found that for the discrete state space of the indicator variables $\mathbf{u}$ used in the sampling strategies, an annealing method did not offer any significant advantages. Whether the sampler uses annealing or not, the sampler has to visit the discrete state at which the posterior reaches its maximum. The number of samples required for such an exploration with an annealing strategy, as well as the time required for the necessary slow annealing schedule, makes this method far too slow and we found that a direct calculation and comparison of the probability of all visited states performed better in practice when only a limited number of samples could be drawn.

Whether to use the mean or the MAP estimate depends on the application. The mean would be the optimal choice under a squared error utility, while the MAP would be optimal with a zero-one utility. However, in this thesis we have not specified utilities and will not do so now, but instead only mention the difficulty in doing so for the sparse coding problem. Our motivation for sparsity lead us to a prior formulation which forces many

	True	Learned
Imp MEAN	24.10	23.58
Imp MAP	25.41	24.97
NN MEAN	3.88	4.40
NN MAP	9.74	8.09
Gibbs MEAN	8.20	7.47
Gibbs MAP	10.12	9.73
EM	7.00	6.69

Table 7.3: Reconstruction error

coefficients to be zero. From a Bayesian point of view, the specification of such a prior is justified if we believe the generating process to follow such a distribution. However, the model used here only crudely models the physical generation of sound and such a prior assumption might then not be accurate for musical signals. On the other hand, we also enforced sparsity, as we believe that such a representation offers advantages for certain applications. This belief should, from a Bayesian perspective, be incorporated into the utility.

In general, a mean approximation leads to less sparse representations but better reconstructions under a squared error norm, while the MAP estimations can be expected to lead to sparser representations but worse reconstructions. This is shown experimentally below, where we compare the mean and MAP estimation for the methods developed here.

7.3.1 Comparing the Signal Reconstructions

We first analyse the reconstructions calculated by multiplying the estimated coefficients $\mathbf{s}$ with the dictionary. The L_2 distances of the reconstructed signals to the true signal are shown in table 7.3. The second column shows the reconstruction error found with the true parameters and dictionary and the third column shows the reconstruction error achieved with the learned model parameters and dictionary. The methods used are listed from top to bottom; the mean and MAP estimates found with the importance sampler (Imp), the Gibbs sampler with the non-negative prior (NN), the Gibbs sampler with the Gaussian and delta mixture prior (Gibbs), and the EM method. As expected, the reconstruction error is smaller if the mean is used instead of the MAP. Again the best performance is achieved with the Gibbs sampler with the non-negativity constraint followed by the EM method, the Gibbs sampler and the importance sampler. Interestingly, the performance is better with the learned dictionary and parameters, even though the learned dictionary does not include all of the features used to generate the signal. This shows that the model with the learned parameters better models the actual data. This difference in performance is particularly large for the two Gibbs sampling methods. It should be noted that all algorithms work for non-zero noise terms only. and so for the results based on the true parameters presented here, we have used a noise variance of 0.01. As the test signal did not have any noise added, the model parameters used did not exactly fit the data which explains why we obtained better results with the learned parameters.

7.3.2 Comparing the Signal Representations

The different representations are compared visually in figures 7.7 to 7.10. Here extracts of the estimated representations are shown in blue and are compared to the true coefficients $\mathbf{s}$, which are shown in red. For each method we show the representations estimated with the true (T) and the learned (L) parameters. For the sampling methods we also show the different estimation methods, mean (MEAN) and MAP. In these figures we have rectified the coefficients $\mathbf{s}$ (apart from the coefficients $\mathbf{s}$ found with the non-negative prior for which the coefficients $\mathbf{s}$ were already non-negative). We have also reduced the time resolution by a factor of 100 in order to produce clearer graphs.

For the coefficients $\mathbf{s}$ estimated with the learned parameters, it was difficult to relate all learned features to a different true note, as not all notes had been learned. However, the notes that had not been learned do not occur often in the true signal. Furthermore, some features were learned more than once leading to some notes being represented by more than one learned feature. This is, for example, evident in the representation calculated with the Gibbs sampler.

The coefficients $\mathbf{s}$ estimated with both Gibbs samplers, as well as the

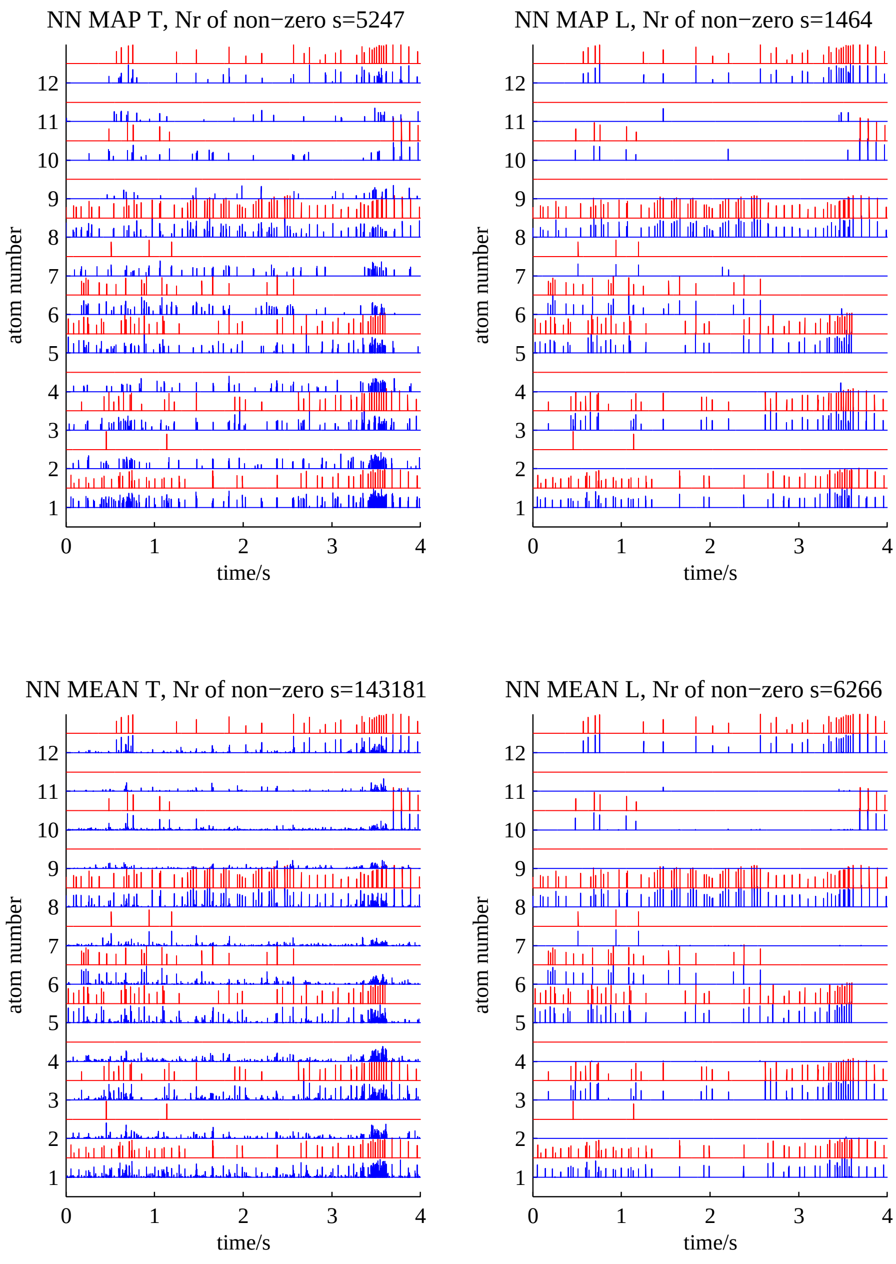

Figure 7.7: Comparison of the representations found with the Gibbs sampler with the non-negative prior (blue) to the true coefficients (red). The top left panel shows the MAP approximation found with the true parameters, the top right panel shows the MAP approximation found with the learned parameters, the lower left panel shows the mean approximation found with the true parameters and the lower right panel shows the mean approximation found with the learned parameters.

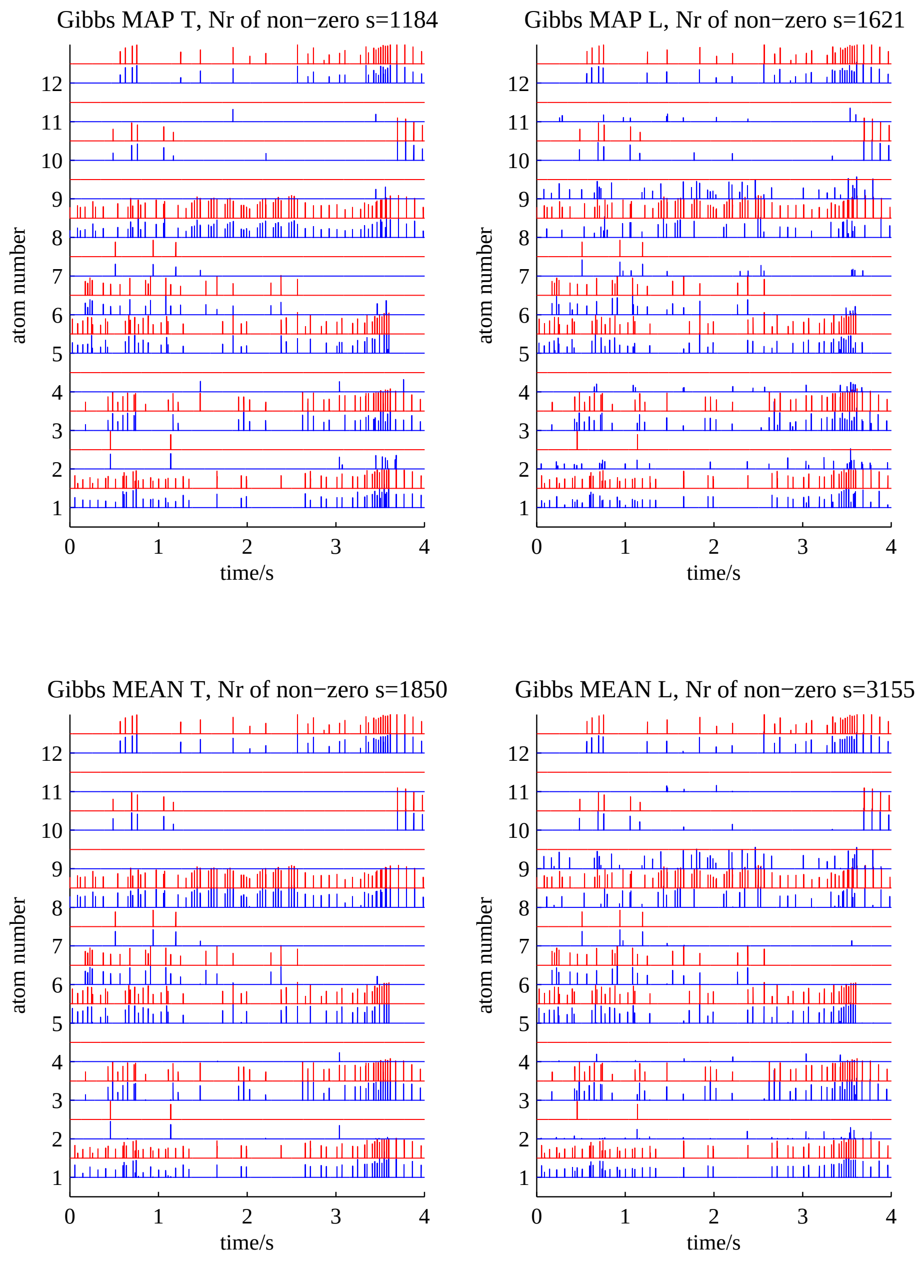

Figure 7.8: Comparison of the representations found with the Gibbs sampler (blue) to the true coefficients (red). The top left panel shows the MAP approximation found with the true parameters, the top right panel shows the MAP approximation found with the learned parameters, the lower left panel shows the mean approximation found with the true parameters and the lower right panel shows the mean approximation found with the learned parameters.

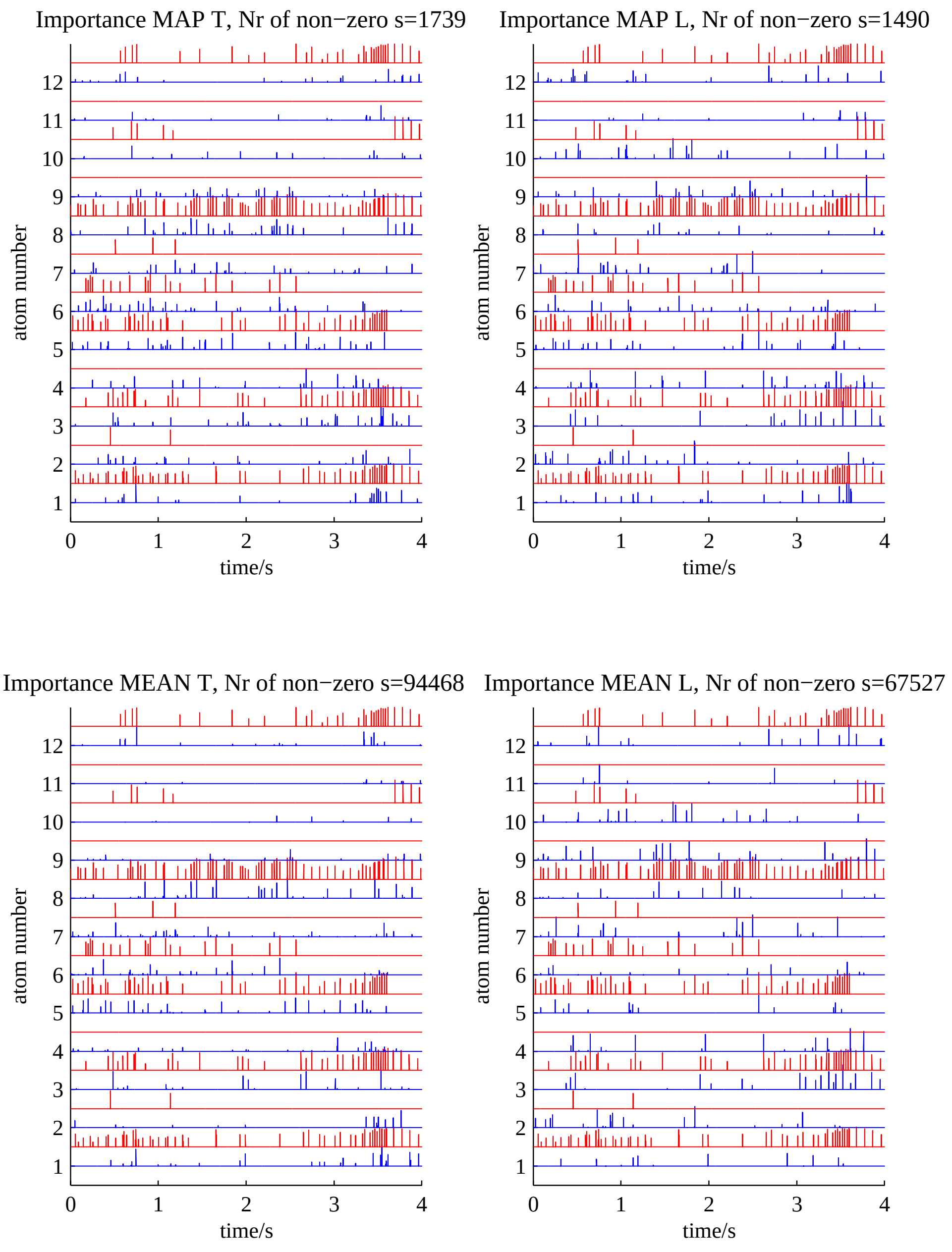

Figure 7.9: Comparison of the representations found with the importance sampler (blue) to the true coefficients (red). The top left panel shows the MAP approximation found with the true parameters, the top right panel shows the MAP approximation found with the learned parameters, the lower left panel shows the mean approximation found with the true parameters and the lower right panel shows the mean approximation found with the learned parameters.

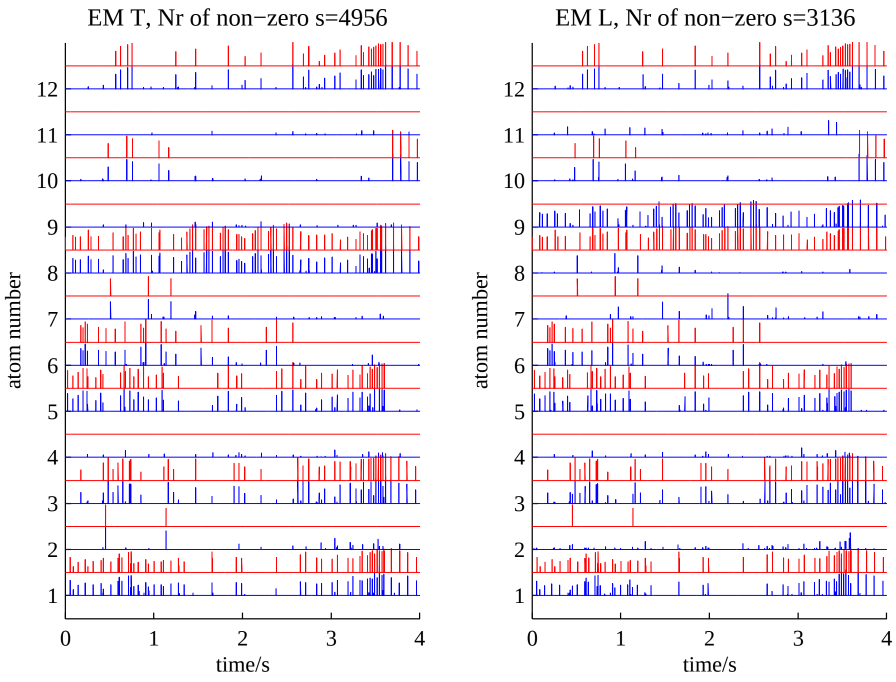

Figure 7.10: Comparison of the representations found with the EM method (blue) to the true coefficients (red). The left panel shows the approximation found with the true parameters and the right panel shows the approximation found with the learned parameters.

coefficients **s** estimated with the EM method, can be seen to closely mirror the underlying structure, while the importance sampling method does not offer a good representation.

MAP estimation via sampling can be seen as a form a random search, where the search is distributed depending on the posterior distributions. The Gibbs samplers draw samples from the correct distributions and the search frequently visits areas with high probability. The importance sampling method draws the samples from a different distribution and the searched areas are less likely to correspond to areas in the true distribution that have high probability, which explains the poor performance of the importance sampler in this task.

The number of non-zero atoms in the original signal was 1324. In table 7.4 the number of non-zero atoms is listed for each of the different methods. For mean estimates it is clear that the number of non-zero atoms

	True	Learned
Imp MEAN	94448	67283
Imp MAP	1165	1484
NN MEAN	143181	6266
NN MAP	5247	1464
Gibbs MEAN	1850	3155
Gibbs MAP	1184	1621
EM	4956	3136

Table 7.4: Number of atoms

is much larger than the number for MAP estimates.

7.4 Piano Note Extraction

In section 7.2 we have studied the learning performance using a simplified test signal. In this section we compare the different methods using a polyphonic piano recording and focus on the performance differences between the methods. A more thorough study of the applicability of shift-invariant sparse coding to music analysis is presented in the next chapters.

In this experiment we used a recording of Ludwig van Beethoven's Bagatelle No.33 Opus 1 as a test signal, which we resampled at 8000 Hz and summed to mono. We then used this signal as a training sequence for the three methods introduced in the previous chapters, utilising the subset selection method for the EM algorithm and the random subset selection method for the Gibbs sampler with the non-negativity constraint. The importance sampler was fast enough to be used without such a scheme, however, we only drew 100 samples in each iteration.

The features $\mathbf{a}_k$ learned with the different methods are shown in figures 7.11, 7.12 and 7.13 for the Gibbs sampler with the non-negativity constraint, the EM method and the importance sampler respectively. We show the features ordered by their approximated fundamental frequency. The time-domain representation of the features is shown on the left while the spectrum of the features is shown on the right. It is evident that the estimated features show clear harmonic structures, as is to be expected

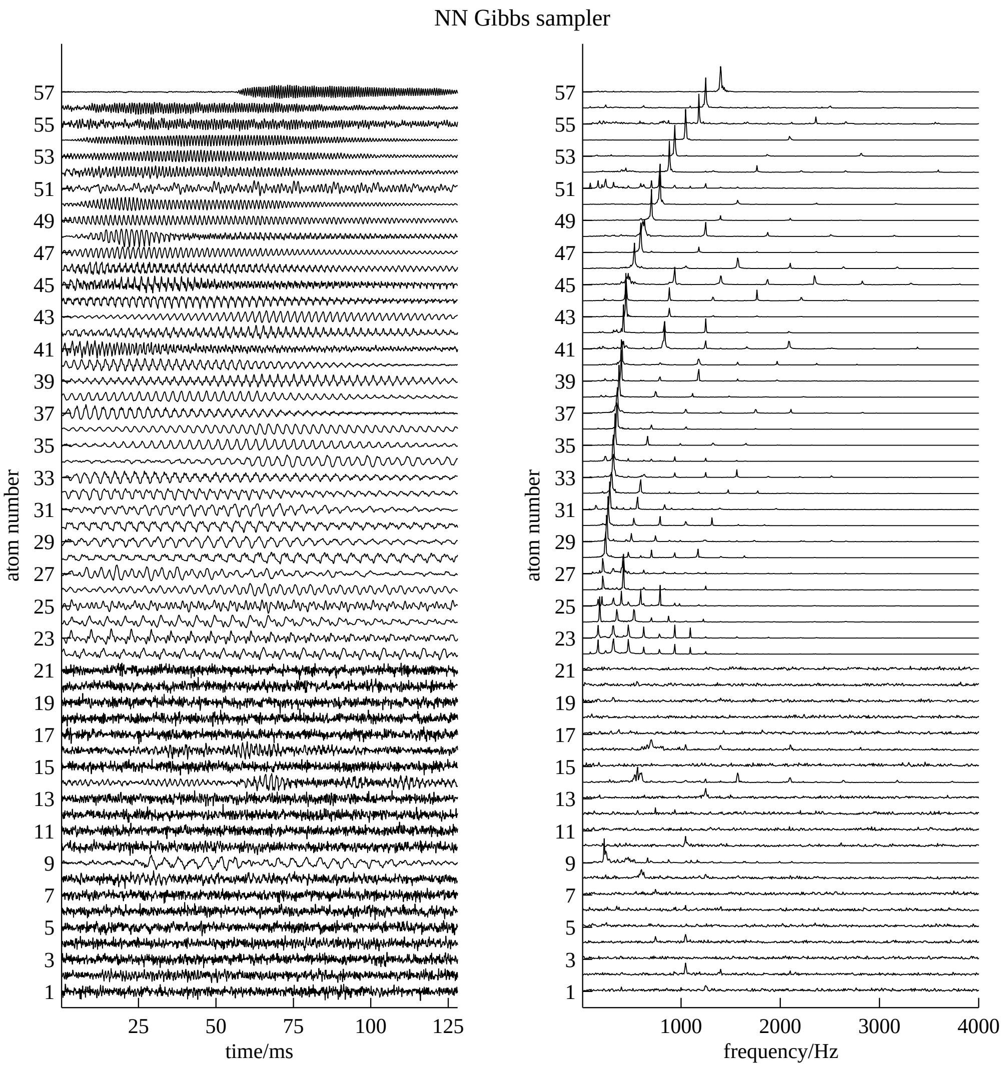

Figure 7.11: Features learned with the Gibbs sampler using the non-negative prior. Time domain representation on the left and spectral representation on the right.

from piano notes. However, the features learned with the importance sampler do have less clear harmonic structures than the results obtained with the other methods, confirming the results in section 7.2. It is interesting to note that with the importance sampling approach all features converged, however, many features have been learned repeatedly. With both other methods this phenomenon was less pronounced. The reason why the importance sampler learns features repeatedly seems to be due to the fact

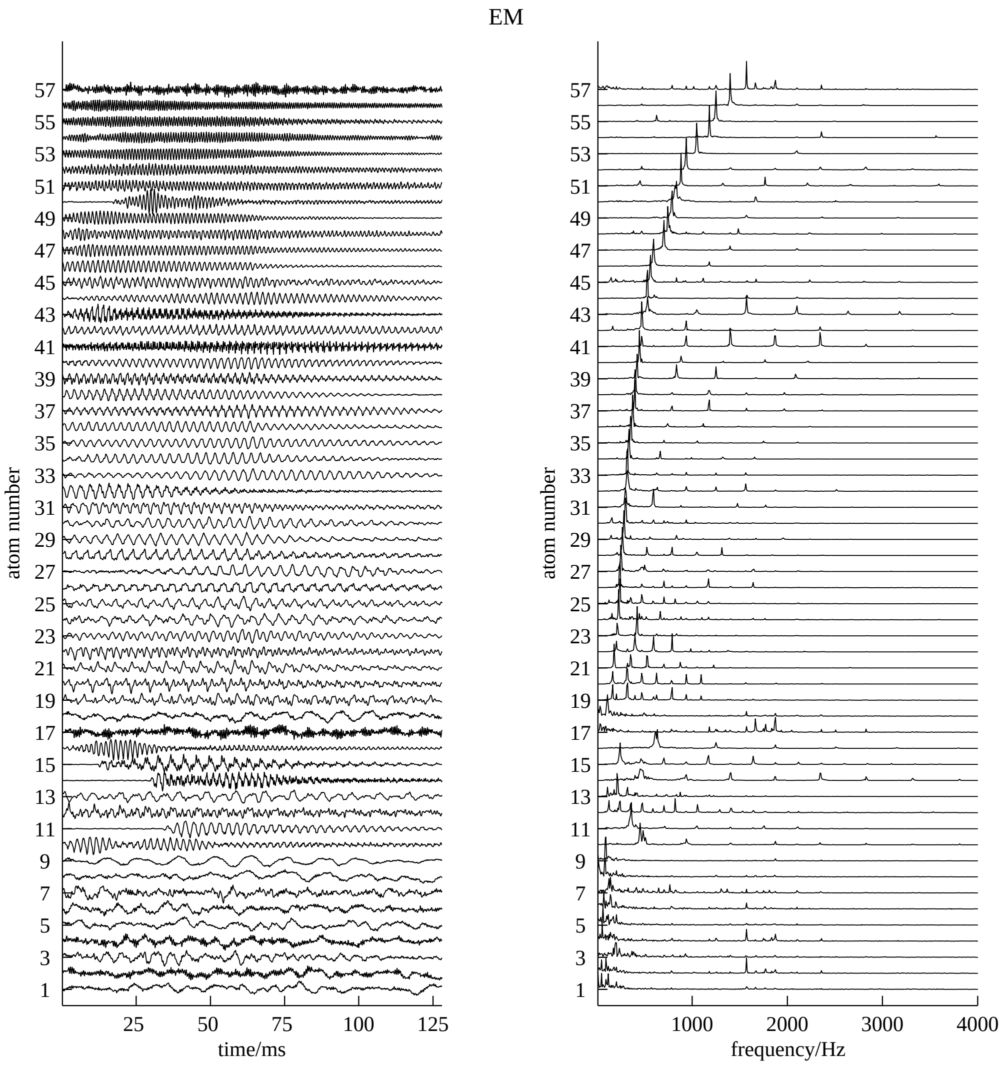

Figure 7.12: Features learned with the EM method. Time domain representation on the left and spectral representation on the right.

that the proposal distribution is more likely to draw different samples to model the same feature in the signal.

The main difference between the results obtained for the Gibbs sampler and the EM method are the number of harmonic features learned. With the EM method, more harmonic features emerged. This might be due to the fact that the random subset selection method selected fewer features in each iteration as compared to the subset selection method used for the EM algorithm. This was done in order to keep the computation time

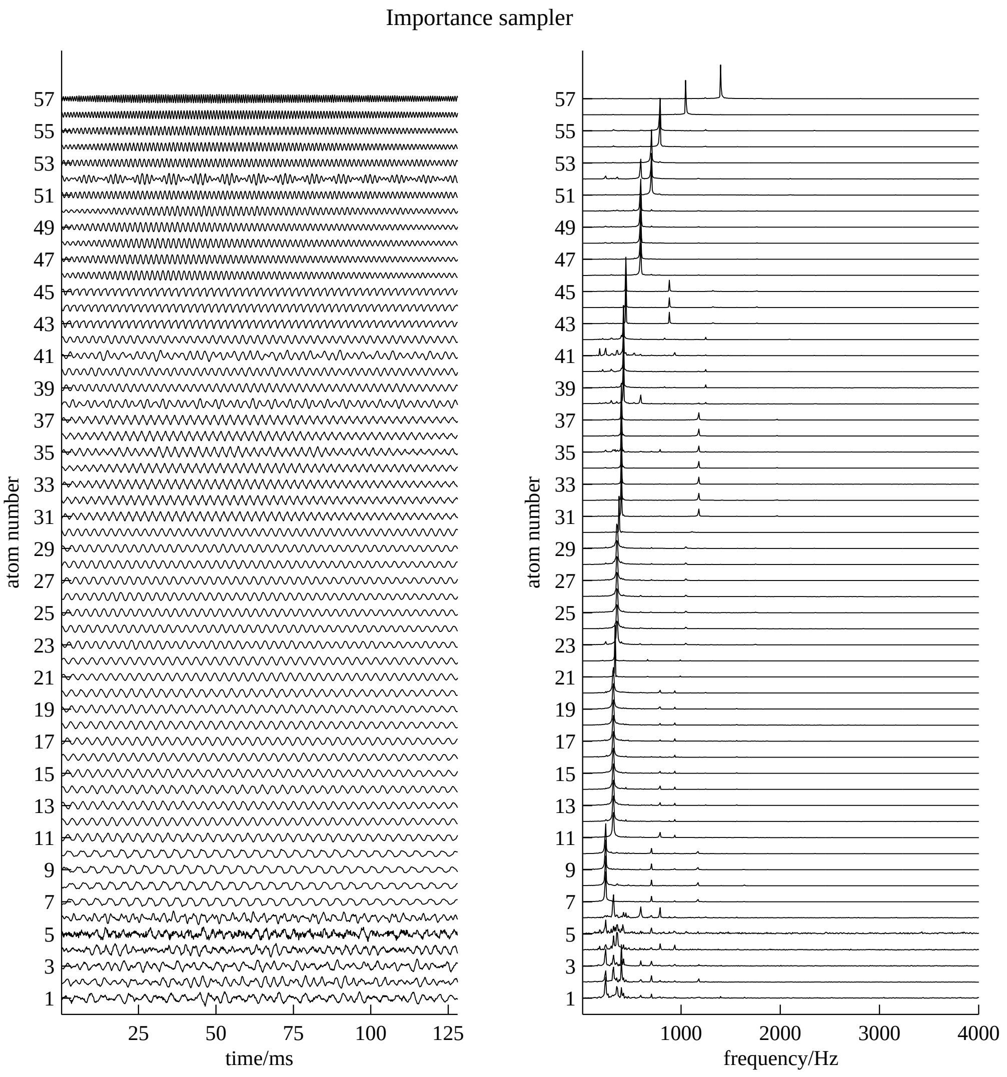

Figure 7.13: Features learned with the importance sampler. Time domain representation on the left and spectral representation on the right.

comparable, as the Gibbs sampler is generally slower than the EM method. However, the number of features with different fundamental frequencies was the same for the Gibbs sampling and the EM based methods. From the features learned with the Gibbs sampler, 34 of the 35 features with clear harmonic structures had different fundamental frequencies. With the EM method 34 features with different fundamental frequencies were also found.

The features learned with the importance sampling method are very

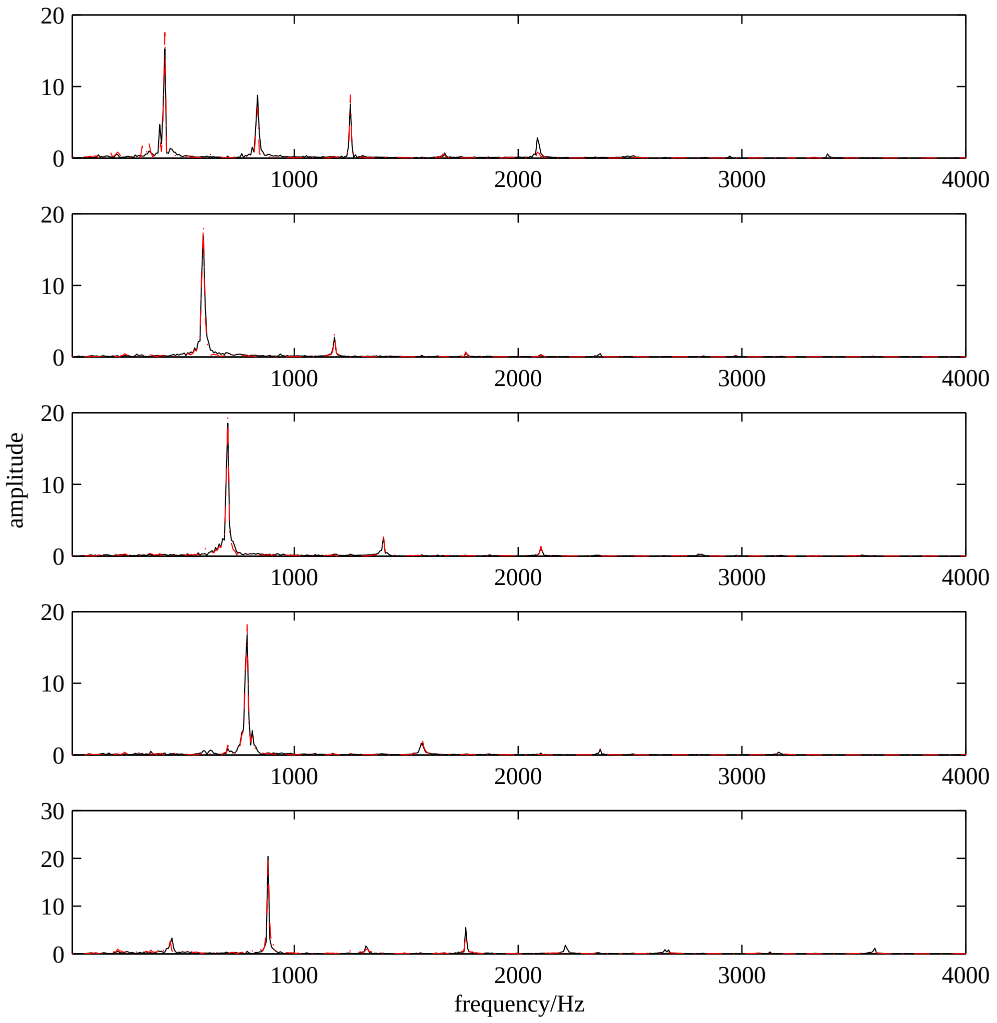

Figure 7.14: A comparison of the spectra of some of the features learned with the EM algorithm and the Gibbs sampler. The red dash-dotted line shows the spectra of the features learned using the EM method, while the solid black line shows the spectra of the features found with the Gibbs sampling method.

dissimilar to the features learned with the other methods, while many of the features found with the Gibbs sampler and the EM approaches are nearly identical. This similarity is shown in figure 7.14 where we compare the spectra of five of the features found with these two methods. Here we have overlaid the spectrum of the features learned with the EM algorithm (dash dotted red line) over the features learned with the Gibbs sampling method (black solid line). In this figure it is difficult to distinguish the red dash dotted line from the black solid line, which illustrates how similar the features learned with the different methods are.

Conclusions

We can distinguish two different performance criteria; the performance in learning of the parameters of the signal model, which include the dictionary elements, and the similarity of the found representation $\mathbf{s}$ to the coefficients used to generate the signal. In this chapter we have looked at these two aspects of the model and compared the proposed methods using a simplified test signal. It can be seen that the Gibbs sampler with the non-negative prior offers the best performance in terms of feature estimation and signal reconstruction. The method also found the representation which was closest to the true coefficients $\mathbf{s}$. The superiority in these respects can be mainly attributed to the prior distribution, which better models the true underlying signal structure and further offers strong constraints on the solution space. However, the Gibbs samplers are slower in general than the importance sampling method or the EM method with a subset selection step. When using the subset selection method for both approaches we found that, for the experiments reported in the last section of this chapter, the computation time for the Gibbs sampler was roughly twice as long as for the EM method. However, this depends on the size of the subsets, which affects the speed of the algorithms differently.

The observed decrease in performance, which due to the bias in the gradient estimate introduced by the importance sampler, shows the same artefacts discussed in section 3.4. Both a decrease in high frequency components (figure 7.4) and the emergence of an envelope (figure 7.3) have been observed. This shows that learning performance depends on the inference accuracy or bias of the gradient estimate.

The decrease in performance found when using the true parameters and dictionary was surprising. This decrease was independent of the particular method used. Even the EM method, which has no adjustable parameters apart from the dictionary $\mathbf{A}$, offered a slightly better performance with the learned dictionary when compared to the dictionary used to generate the training signal. This seems to be the effect of the algorithms used, which require the specification of a non-zero noise term that was not included in the signal generation.

In the next chapters we investigate the performance of the shift-invariant sparse coding model on several tasks in music analysis. The shift-invariant sparse coding model is applicable to a variety of tasks and we study the problems of source separation and polyphonic music transcription. Because of the good performance and speed of the EM method with the subset selection step we chose this method for the experiments in the remainder of this thesis.

Chapter 8

Emergence of Musical Structures[1]

Music is a highly structured signal. In most music, different harmonic and percussive sounds are superimposed and concatenated to create harmonic and rhythmic patterns. The model developed throughout this thesis can be seen as an approximation of this process. In particular the linear additive structure models signals such as music as a superposition of atomic elements. For certain instruments, such as the piano, these elements can be seen as individual notes or parts thereof. This is obviously a simplification, the time domain representation of a recorded and discretised piano note can vary both with respect to the short-time spectrum as well as with respect to its length. The proposed model can compensate for these variations to a certain extent by using several atoms to represent each note.

In the first section of this chapter we take a step back from the discussion of the previous chapters and take a closer look at piano notes and in particular the ability of a linear additive representation of such notes. The main finding is that over 88% of the variation in a set of fourteen renditions of a single piano note can be explained as the superposition of only two features. This suggests the applicability of the proposed model to the problem of modelling piano music.

In section 8.2 we tackle this problem using the shift-invariant sparse coding model. Here we extend on the experiments on piano note extraction from the previous chapter and analyse the learning performance in more detail. We also look at the emergence of score-like structures. If the

[1]Some of the results in this chapter also appear in [13]

features $\mathbf{a}_k$ represent individual notes or parts thereof, then the coefficients $\mathbf{s}$ encode information relating to the occurrence and strength of these notes in the performance. This relationship is further investigated in section 8.3, where we compare the original score of a piano recording with a transcription of this score obtained from the coefficients $\mathbf{s}$ estimated using the shift-invariant sparse coding model.

8.1 Applicability of the Shift-Invariant Sparse Coding Model to Piano Music

To gain a better understanding about the features found in polyphonic music recordings, it is beneficial to analyse the statistics of a single piano note. If the goal of a learning algorithm is to learn representations of individual notes, it has to be investigated if such representations are feasible in the time domain and, if so, of which form these representations are likely to be. The questions of interest are:

1. Is there a single time domain representation that contains the relevant features of a note in order to distinguish notes of different instrument types or even of different models of the same instrument, so that such a representation can be assigned to an individual note and instrument in a recording?

2. Does a single feature contain enough information for transcription, source separation or signal compression, i.e. can a signal be compressed using one general feature vector for each different note present?

3. What do such features look like and what information do they contain? Is, for example, the relationship of the phase of high frequency and low frequency components similar for different realisations of the same note?

4. How high is the dimension spanned by a note played several times? Can the series of notes be represented accurately enough with a single vector and what information gets lost in such a representation.

Here several realisations of a single note played on a piano under similar conditions are analysed and their properties studied. The notes analysed were taken from a commercial recording of Ludwig van Beethoven's Sonata for Piano No. 12, in A flat, Scherzo (Allegro molto) in which this particular note was played 14 times without any other overlapping notes. The note was extracted by cutting the recording just before the note onset and again just before the onset of the following note. This procedure gave a set of 14 notes of identical pitch, played at roughly the same loudness and of roughly the same length.

Principal component analysis (PCA) of the 14 piano notes was performed. The individual piano notes were normalised and time aligned to maximise the cross correlation between them before conducting PCA. The results are shown below. The top panel in figure 8.1 shows the ordered contribution each principal component makes to the variance observed in the 14 observations. It was found that two principal components account for 88% of the variance of the original notes. The other components are much less significant, with the third component accounting for about 4% of the variance. Note that only 13 principal components have been found, which means that the 14^{th} component had such a small contribution that it was smaller than the accuracy of the computation so that the 14 notes effectively span a 13 dimensional space. It is also clear that a two dimensional representation would account for 88% of the information and it seems that at least two components are necessary to represent a single note.

The time domain and spectral representations of one of the original piano notes is shown in the second row of figure 8.1. In the third and fourth row of figure 8.1 the time domain and spectral representations of the principal components related to the highest and second highest variance are shown respectively. The similarity of the spectrum of the original note to the principal components is evident. It can further be seen that the second principal component has much higher fifth and seventh harmonics than the first principal component.

The time domain and spectral representations of the weakest principal component are shown in the last row in figure 8.1. It is obvious that it

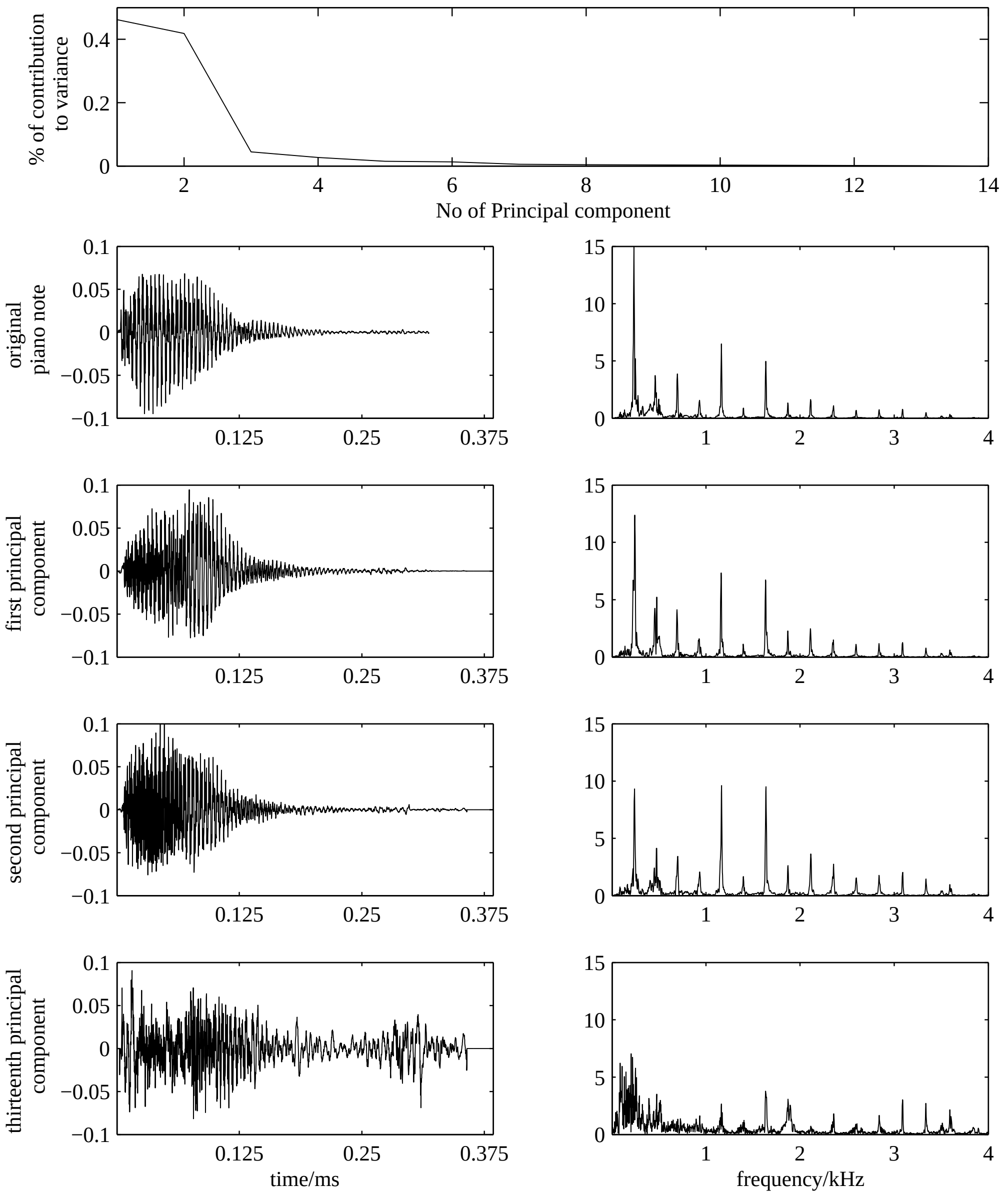

Figure 8.1: Principal component analysis of piano notes. Percentage that each principal component contributes to the variance of 14 piano notes (top), the time-domain (left) and spectral-domain (right) representations of one of the original piano notes (second row), the principal components with the largest eigenvalue (third row), the second largest eigenvalue (fourth row) and the smallest eigenvalue (fifth row).

contains much more noise but it still contains some harmonic structure. It is interesting to note that the higher harmonics are of comparable strength

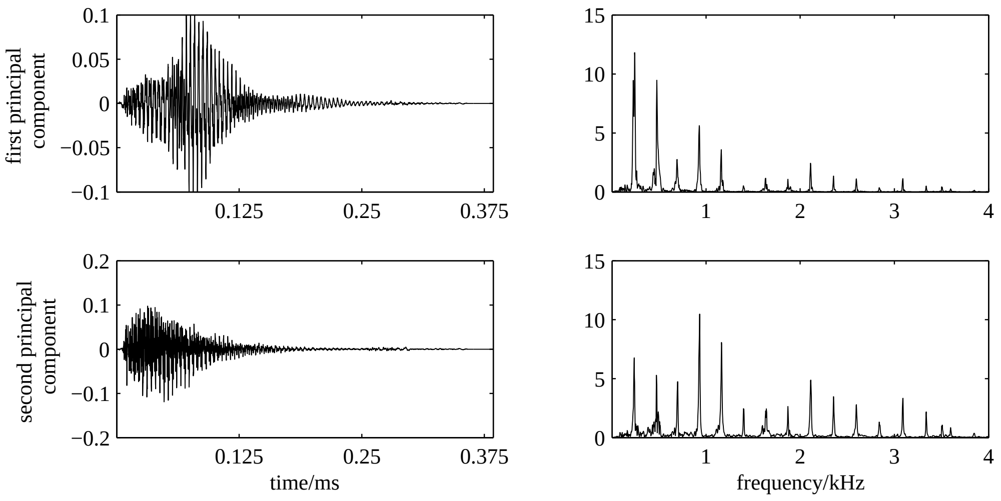

Figure 8.2: Time-domain (left) and spectral-domain (right) representations of the two strongest principal components of the left channel. The form of the envelope clearly suggests that the two principal components are necessary to represent the piano notes with different envelopes.

to the high harmonics in the other principal components, while the low harmonics are much weaker.

Here, as well as in the results presented in the next chapter, a stereo recording was summed to mono before analysing the signal. In order to see the effect of this summation and to investigate whether the second strongest principal component might be due to the signal reaching the different recording microphones with varying strengths and delays, the same experiment was conducted using only one of the stereo channels. The same observations were made as reported above. It is interesting to note that the two principal components that are responsible for most of the variation in the signal have different amplitude envelopes. One component models mostly the note onset, while the other component models mainly the latter part of the note. This is shown in figure 8.2. A similar observation, though not as pronounced, can be made for the strongest principal components shown in figure 8.1, which were found in the previous experiment. In both cases, the second strongest component is found to have a slightly higher high frequency content compared to the strongest component.

The above results were obtained from a set of notes, all of which were roughly of the same length. Obviously for notes of different lengths these results are not valid, however, the rest of this thesis demonstrates that notes of different lengths can be handled by the shift-invariant sparse coding model by concatenating features. This is shown in detail in chapter 9. It should also be mentioned that the piano notes used above where all of roughly the same loudness and the influence of changes in the loudness on a linear representation, could not be deduced from the above experiment. For the case of notes of similar lengths and with roughly similar amplitude we can give the following answers to some of the questions raised above.

1. It can be seen that a single component can represent the magnitude spectrum of a piano note quite accurately, but fails in representing the different time envelopes observed for different notes. For accurate reconstruction of notes it seems necessary to model a note with at least 2 features to cater for the different time envelopes. It can, however, be assumed that a single feature can capture much of the information and that two features can represent a piano note quite accurately. The question of whether features learned from different sources are significantly different in order to separate two sources was not investigated here, but experimental results are presented in chapter 10. These results show that at least for certain mixtures this assumption can be made.

2. In the piano example reported above it is evident that the pitch of a piano note can be described by a single feature. Whether this is still true for notes played on different pianos or for notes recorded in different acoustical environments is questionable. However, for the task of identifying individual notes played by a single piano, a set of features each describing an individual piano note might be sufficient. For blind source separation, features have to be found and grouped that relate to single sources and that offer good reconstruction of those sources. For piano signals, it was shown that at least 2 features are required for good reconstruction of individual notes. For high quality signal compression a single feature is therefore not enough.

However, if a MIDI representation of a musical signal is seen as a low quality compression of the original audio file, then such a compression can be achieved from transcription, which seems feasible with only a single feature.

3. It can be seen that the single time domain feature that represents the above piano notes relatively well, has a similar spectrum to the original piano note. It also has an envelope that is similar to the original time envelope of the notes in the sample space. It must again be mentioned that the above sample set was quite restricted in that it not only contained notes at the same velocity, but also notes of a similar length with only slight envelope deviations. It is therefore not surprising that a single time domain representation can be found with a similar time envelope. Such a simple representation is not possible for sounds of varying length or for sounds with different time envelopes.

4. It has been shown that for the example studied here, most of the variance of the notes is concentrated in a two dimensional subspace. However, the dimension of this subspace is likely to increase for more complex signals.

8.2 Learning Piano Notes

We have shown in section 8.1 that piano notes can be modelled using a linear additive time-domain model. In this section we use the shift-invariant sparse coding model to learn sparse approximations of piano music and investigate the estimated features. Here and in the rest of this chapter we use the EM algorithm developed in chapter 4.

To test the algorithm, a recording of Ludwig van Beethoven's Sonata for Piano No. 12, in A flat, Scherzo (Allegro molto) was used. The original stereo recording was summed to mono and resampled at 8 000 Hz. The number of possible features was set to 50, a feature length of 1024 samples was chosen, ν was set to 0.1 and the maximally allowed amount of overlap of one feature with a shifted version of itself was set to 50%. The EM

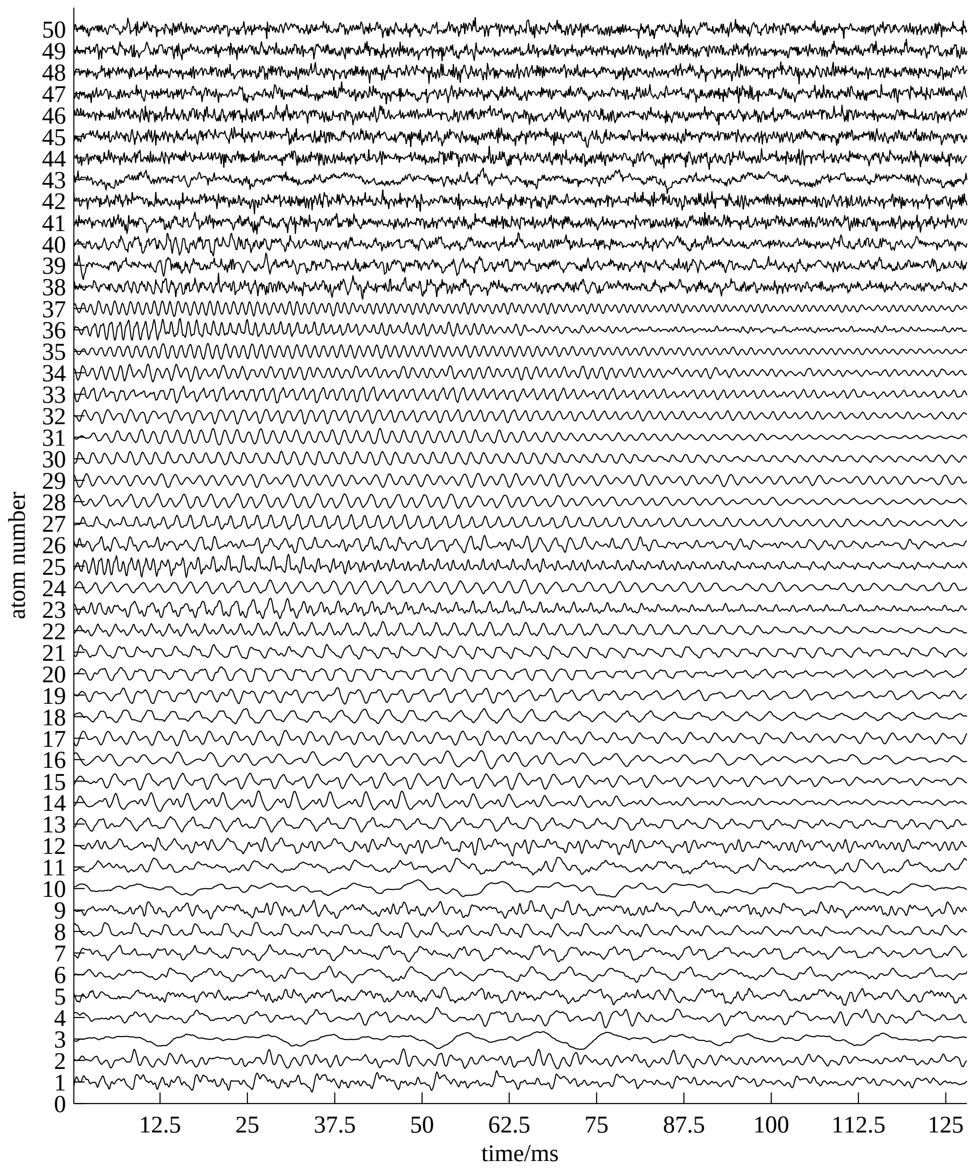

Figure 8.3: The 50 features learned from a recording of Beethoven's Sonata for Piano No. 12, in A flat, Scherzo (Allegro molto) shown in the time domain.

algorithm used a fixed number of 10 iterations and a sparsity measure of the form $\sum_n \log |s_n|$.

After 100 000 iterations, 12 of the features did not show any harmonic structure and were of a noisy nature. The other 38 features had a clear harmonic structure. Of these 38 features 35 had different fundamental frequencies whilst the other 3 features had a fundamental frequency equal

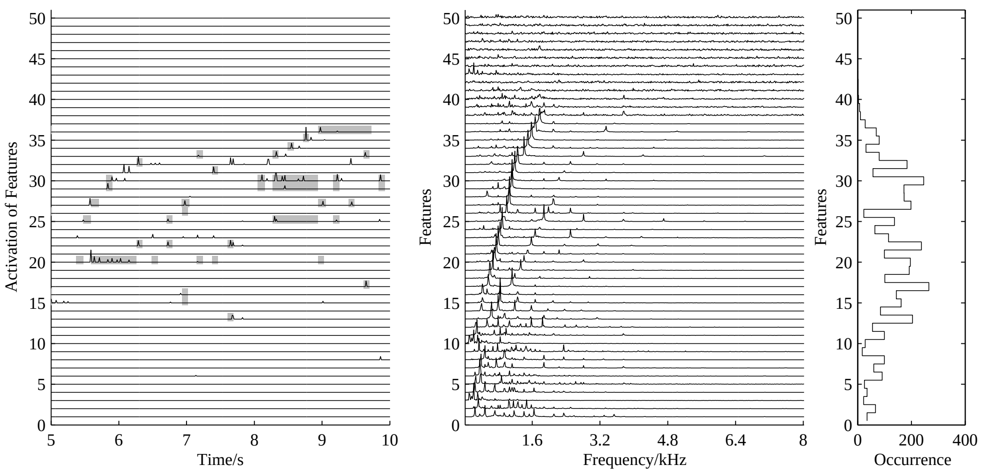

Figure 8.4: Extract of the rectified activation pattern of the features represented by spikes with grey blocks representing the notes in the original score (left), magnitude-spectrum of features (middle) and their number of occurrence in the decomposition (right).

to at least one other feature, however, these features differed in their harmonic structure. Further analysis of the features showed that the fundamental frequencies corresponded to the notes of the western equally tempered 12 tone scale spanning a range from C#2 to A5 with some notes missing. Most features were harmonic in that their spectrum had a harmonic series of peaks. The amplitude of these peaks varied with one harmonic often having a much higher amplitude than the others. It was noted that there were no harmonic series present with very low fundamental frequencies even though such notes were present in the analysed signal.

The time-domain representations of the learned features are shown in figure 8.3 while the middle panel of figure 8.4 shows the magnitude-spectrum of the features. The features have been ordered by their approximated fundamental frequencies. Features 38-50 could not be assigned to a certain frequency as they had no clear peaks in their spectrum. 4 of the features with clear spectral peaks did contain more than one harmonic series of peaks, i.e. they represented chord-like structures. However, it cannot be assumed that these features model piano chords in general, as the different notes in a chord are generally not phase locked for different renditions of the same chord. It seems more likely that these features have converged to local optima and are used to model different notes at different times.

As most of the features $\mathbf{a}_k$ can be assigned to individual notes, the coefficients $\mathbf{s}$ contain information about the occurrence of the notes in the piano recording. This can be seen in the left panel of figure 8.4, where we show an extract of the rectified coefficients $\mathbf{s}$ associated with each of the features. In grey we show the position and length of notes with the same pitch as they occur in the original score of the sonata. It can be seen that many of the occurrences of the features correspond to notes in the score. In the left panel of figure 8.4 we only show the notes of the original score for which a feature has been found. Some of the notes in the performance, however, do not have associated features and are therefore omitted. It is also clear that some of the notes in the score have no associated non-zero coefficients and that some non-zero coefficients do not correspond to notes in the score. Some of these errors seem to be due to a feature modelling a different note to the one assigned to it here. This can be seen in the left panel of figure 8.4 where non-zero coefficients in the activation of feature 23 correspond to notes that, in the assignment here, should have been modelled by feature 20.

It was noted that some features emerged with the same fundamental frequency but with different harmonic structure. An example for this are features 29 and 30 in figure 8.4. In the left panel it can be seen that these different features model different parts of a note and it was found that the note onset was often modelled by a feature with higher high frequency contents, while the latter part of the note was often modelled with a feature with less high frequency content. This is discussed in more detail in the next chapter.

In the right panel of figure 8.4 the number of occurrences of each feature in the decomposition is given. It is evident that a high number of features do not occur at all (7 features) and that other features only occur a few times (12 features occur fewer than 10 times each) in the entire training signal. The features that do not occur in this particular decomposition are those features shown on the top in figure 8.4. It can therefore be assumed that these features have not been updated during learning and so cannot represent salient features of the signal.

8.3 Quantitative Evaluation

In section 8.2 it has been shown that the shift invariant sparse coding algorithm is able to learn features from a polyphonic piano recording that represent individual notes or parts thereof. In the previous section we could also show that many of the coefficients $\mathbf{s}$ correspond to the occurrence of individual notes in the original score of the performance. This correspondence is explored further in this section, in which we present a numerical evaluation.

A recording of Ludwig van Beethoven's Bagatelle No.33 Opus 1 was used. In order to generate this recording we used the same MIDI information obtained from a real performance that we used in the previous chapter. However, this time this MIDI information was used to control the keys of a MIDI controlled acoustic grant piano. We therefore had a live recording of a real acoustic piano as well as the associated exact performance information. In order to learn the features an approach similar to the one used in the previous section was taken. The original stereo recording was summed to mono and resampled at 8 000 Hz. The number of possible features was set to 57 (as this was the known number of different notes played in this piece), a feature length of 1024 samples was chosen, ν was set to 0.1 and the maximally allowed amount of overlap of one feature with a shifted version of itself was set to 50%. The EM algorithm used a fixed number of 10 iterations and the $\sum_n \log |s_n|$ sparsity measure was used again unless noted otherwise. In the general optimisation problem $\arg\min \|\mathbf{x} - \mathbf{A}\mathbf{s}\|_2 + \lambda f(\mathbf{s})$, where $f(\mathbf{s})$ is a sparseness measure, we are left with the choice of λ. Different methods have been discussed in subsection 4.2.5. In the experiments in this section, λ was set to the estimated noise variance during learning of the features, however, different methods were used for the inference of the coefficients $\mathbf{s}$ once the features had converged. More details on this are given below. The features $\mathbf{a}_k$ were initialised with Gaussian noise.

After 100 000 iterations, 47 of the 57 learned features were found to have harmonic structures, whilst 10 features had not converged and remained in their original 'noisy' state. Of the 47 harmonic features, 10 features were found that represented the same note as at least one of the other features, so that 37 features with different fundamental frequencies were learned.

To evaluate the correspondence between the coefficients $\mathbf{s}$ and the original score numerically, a sparse approximation of the signal was calculated and the estimated coefficients $\mathbf{s}$ mapped into a score representation. It is clear from the example in the previous section that such a mapping is not trivial, as some features are used for more than one note. Furthermore, the features are of fixed length and notes are generally described by a concatenation of the features. The mapping used here associated a pitch to each feature depending on its estimated fundamental frequency. The occurrence of a non-zero coefficient $\mathbf{s}$ was then taken as the beginning of a note. Each note was assumed to be of the same length as a feature. However, if a non-zero coefficient followed another non-zero coefficient within the length of a feature, the note was assumed to start at the first coefficient in such a chain (which might have more than two non-zero coefficients in short succession) and end after the last non-zero coefficient in that chain.

To investigate the accuracy of such a transcription, the number of correctly identified notes was calculated by searching for a detected note in a window of 200ms centred at the start of each note in the original score. The percentage of detected notes is denoted by de.

Two types of error can then be found; firstly, false positive detections (denoted by fp), which are detected notes that are not occurring in the original score and secondly, false negative detections (denoted by fn), which are notes in the original score that are not detected. These numbers are expressed as a percentage of the total number of notes in the original score. The number of correctly identified notes or true positives is then related to fn as $true\ positives = 1 - fn$.

It is clear that the performance cannot be measured by the number of correct detections or the number of false positive or false negative errors alone. It would be possible to get one-hundred percent detection by just

	Noise variance λ_c=100	Noise variance L_1, λ_c=1 000	Eq.(4.7) $\lambda_{max} = 0.01$	Fixed λ λ=0.01	No EM step
de	0.5340	0.4824	0.5790	0.5553	0.6069
fn	0.4660	0.5176	0.4210	0.4447	0.3931
fp	0.0934	0.0885	0.1458	0.1032	0.3997
error	0.5594	0.6061	0.5667	0.5479	0.7928

Table 8.1: Comparison of transcription performance. Influence of different estimation methods for λ. For details see text.

assigning notes to all time locations. Obviously this would lead to a huge number of false positive errors. We calculate the total error as:

$$error = \frac{fn + fp}{N_o}$$

where N_o is the number of notes in the original recording and fn and fp are the number of false negative and false positive errors. This value can become greater than 1 as we have theoretically an unlimited number of false positive detections. Note that the above error term is linear in both the false positive and the false negative errors. Other suggested measures (for example [6]) are not linear in both error types, but give values in a range between 0 and 1, which does not seem necessary here. The problem of counting certain errors twice as would happen when a note is detected in, for example, the wrong octave, which leads to a false negative as well as a false positive error is not accounted for in this measure.

As the learned features did not represent all notes in the original score, only those notes for which a corresponding feature was found are used in the calculations. This was done in order to evaluate the performance of the approximation algorithm and not the feature estimation, which clearly requires improvement in order to learn representations of all notes present in the signal. For example, notes for which no feature has been learned could be modelled by pitch shifting features learned to represent notes with similar fundamental frequencies.

The labelling of features with individual pitches and MIDI note numbers was done by hand, however an automatic algorithm could be developed to find the pitch (i.e. the periodicity) of the individual features.

Several ways of calculating the λ parameter in the EM algorithm have been mentioned. Here different methods are compared, the first of which is the estimated noise variance as proposed by Figueiredo and discussed at the end of section 4.2. The values reported below are given for a range of different scaling parameters λ_c to investigate the optimality of the parameter. The scaling parameter λ_c was used to multiply the estimated noise variance in order to calculate λ ($\lambda = \lambda_c \hat{\sigma}^2$). The method described in subsection 4.2.5 proposed by Rao et al. and given in equation (4.7) is also used.

The best results obtained with each method are shown in table 8.1 together with the corresponding value for λ. The results calculated with the scaled noise variance are given in the first column and the results calculated by using equation (4.7) are given in the third column. In the second column the results obtained by using the EM algorithm with the L1 norm are shown again for different values for the scaling of the noise variance. In the fourth column, these results are compared to the results obtained by using a fixed λ value. In the last column of table 8.1 the transcription results are shown that were obtained by only using the subset selection step in the approximation together with a simple least squares minimisation in this set. The errors obtained in this case give an upper bound on the achievable false positive performance and a lower bound on the false negative error. The main limiting factor in the detection of notes is due to the subset selection step whilst the performance with respect to false positive notes can be attributed to the sparse coding in the selected subset.

Figure 8.5 shows a graphical representation of the performance for the four different approaches. Here the red line shows the percentage of correctly identified notes, the cyan line the total error, the blue line the percentage of false negatives and the green line shows the false positive detection. The abscissa gives the different scaling values used for each method (note that this axis is not linear).

From the figures and the tables in this section it is clear that the choice of method used to calculate λ has not had a strong impact on the results obtained. However, the choice of the free parameter (e.g. the prior scale

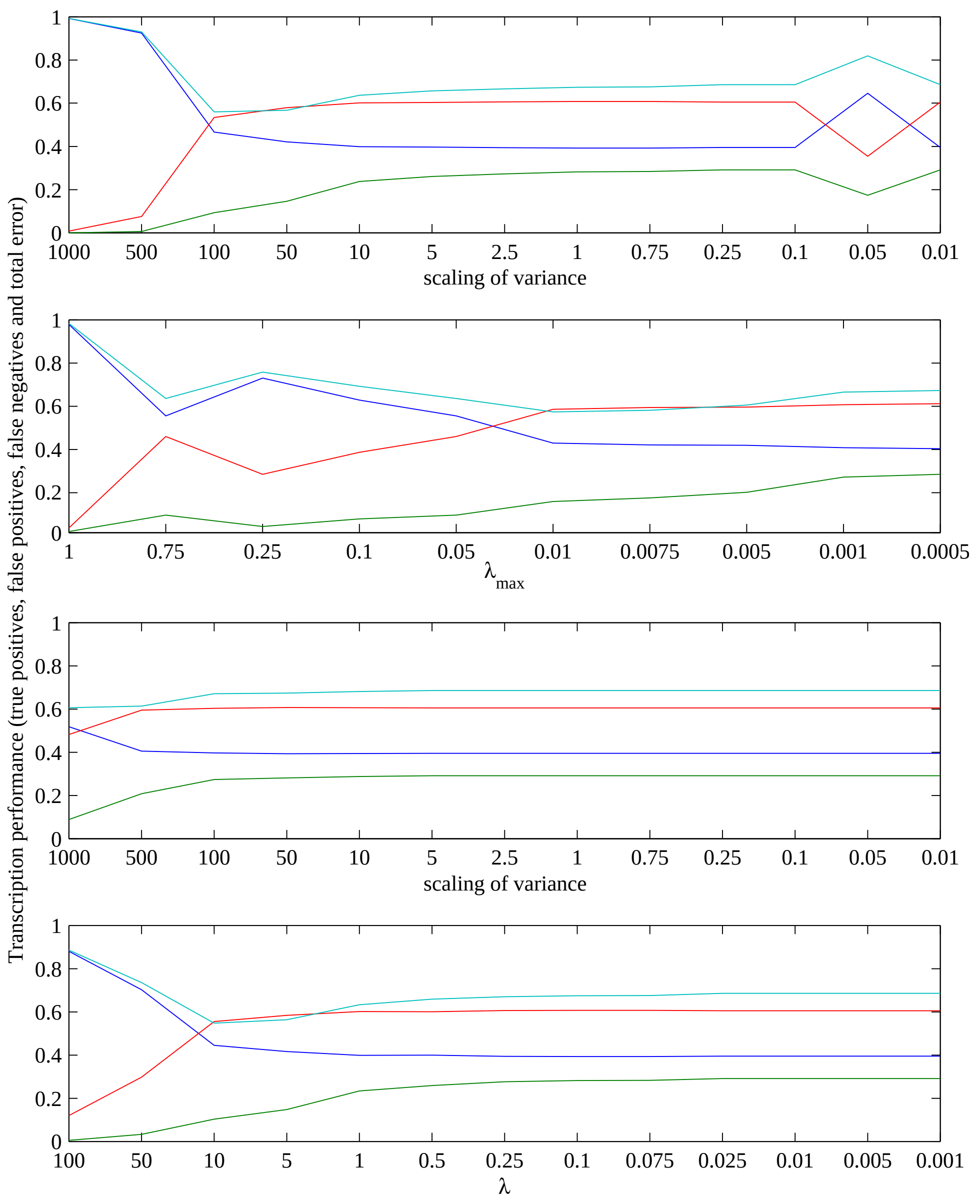

Figure 8.5: Transcription performance of the different approaches. Performance for the estimation of $\mathbf{s}$ based on different methods to estimate λ. The results for the method that uses the scaled estimated noise variance is given in the first row, the results for the method that uses equation (4.7) are given in the second row, the results for the method that uses the scaled estimated noise variance and a L_1 norm are given in the third row and the results for the method that uses a fixed λ are given in the fourth row. Correctly identified notes (red), total error (cyan), percentage of false negatives (blue) and percentage of false positive (green) are plotted vs. the free parameter in each of the approaches.

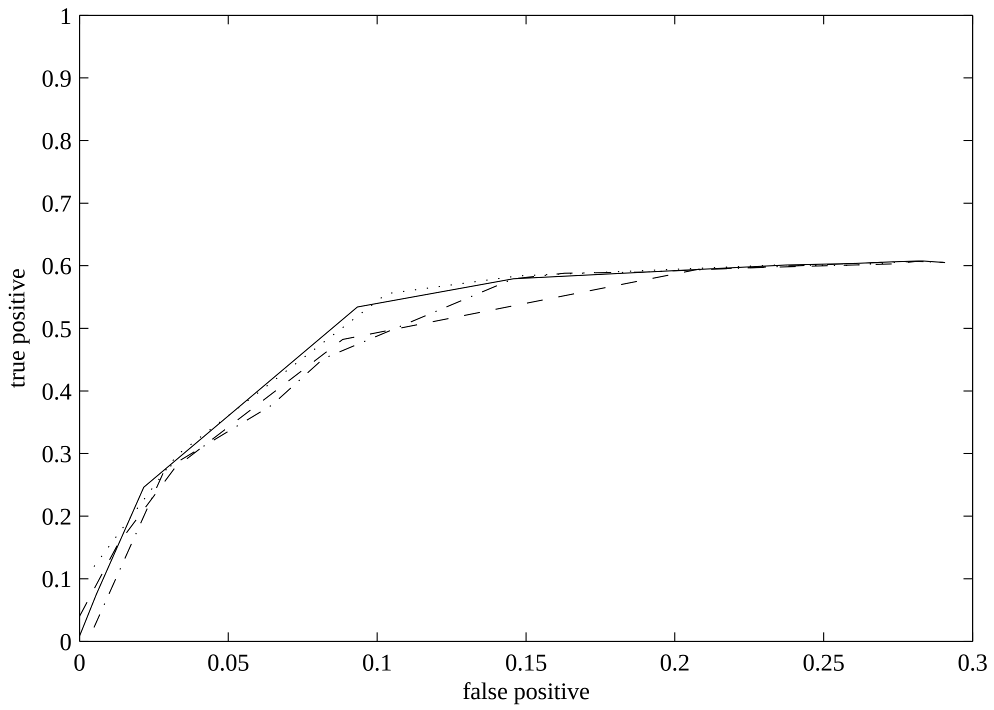

Figure 8.6: True positives vs. false positives. ROC style plot for the influence of the λ parameter on the transcription error. The solid line shows the results for the scaled variance, the dashed line shows the results for the scaled variance using the L_1 norm, the dash-dotted line shows the results when using equation (4.7) and the dotted line shows the results for fixed λ.

parameter λ_c or λ_{max} as discussed in subsection 4.2.5) in each method clearly has. It is also clear that the results do not differ greatly when using the L_1 norm instead of the $\sum_n \log |s_n|$ sparsity measure. The performance limit due to the subset selection step, which is the main restriction once an optimal value for the free parameter has been chosen, is also evident.

It was suggested that an optimal value for λ can be obtained by using a *L-curve* method (See for example [30]). This method is related to the common practise of comparing two error types using a receiver operation curve (ROC). As fp is unbounded, this error is plotted on the abscissa and $1 - fn$, or the true positives, is plotted on the ordinate. This plot can be seen in figure 8.6 where the results for the different methods of estimating λ are plotted. The solid line shows the results obtained using the scaled variance with the $\sum_n \log |s_n|$ sparsity measure, the dashed line shows the results for the scaled variance using the L_1 norm, the dash-dotted line shows the results when using equation (4.7) and the dotted line shows the results for fixed λ. The results for the EM algorithm with the $\sum_n \log |s_n|$ sparsity measure, either with fixed or estimated noise variance, seem to be slightly better compared to the other methods, however, this advantage is small.

Further insight can be gained when looking at the types of errors made. It has to be determined whether notes that have been detected incorrectly, i.e. false positive notes, have an octave relationship to undetected or detected notes. This type of error is likely to occur due to the high correlation of octave related notes. Other harmonic relationships such as fifths and fourths might also occur. We also investigated semitone relationships. This was done, as for the short feature length used here, the uncertainty principle might lead to features being unable to distinguish between notes with close fundamental frequencies, i.e. a note might be modelled with a feature representing a note one semitone in either direction, as the feature might fit the note relatively well due to its short time support. This effect could be seen in the previous chapter, where in the toy example, certain features were not learned. However, features learned to model notes a semitone either side did model the missing feature relatively well. Other errors that we investigated were double detection (i.e. detecting a single note twice in short succession) and detection of only one note when the same note is played twice in fast succession.

To investigate the occurrence of these types of errors a window of 200ms centred on the note onset was defined and the different types of errors counted in this window. The results obtained can be found in table 8.2. The error of not detecting a fast repetition of a note is normalised to the number of false negative errors whilst all other types of errors are normalised to the number of false positive errors. The results given here were obtained for the experiment as given in the first column of table 8.1.

It can be seen that the most common of the investigated errors are octave relationships between a false positive and a false negative error. Further investigation of the features that were used to describe notes in

Octave detected (2 octaves up/down)	7.02%
Fifth detected	4.39%
Fourth detected	2.63%
Semitone detected	4.39%
Thirds detected (Major or minor)	5.26%
Note detected twice	5.26%
Fast repetition of note not detected	0.35%

Table 8.2: Different sources of error encountered in the transcription.

different octaves showed that these features had a very strong second harmonic and very weak other harmonics. It was also observed that the number of octave-related errors started to rise faster when the number of false positive detections increased than compared with the other types of error.

It is clear that the coefficients **s** have captured much of the information about the score of the performance. However, the simple transcription scheme presented here is not the best solution to the problem of music transcription and much better results for piano music transcription have been obtained with more sophisticated approaches [146]. Nevertheless, it could be shown that the shift-invariant sparse coding algorithm can extract much of the information in a piano signal without prior specification of musical structures.

Conclusions

In this chapter we have used music as a testbed for the shift-invariant sparse coding algorithm. Music is a highly structured signal and it can be shown that the shift-invariant sparse coding algorithm is able to extract much of this structure without the use of prior musical knowledge. In music, such prior knowledge is often available, however, for other signals such information is often not given. Finding such structures is an important requirement for many tasks in signal processing such as coding, source separation and pattern analysis and the shift-invariant sparse coding model offers an important tool to discover such structures.

In this chapter it was shown that for piano music, the linear generative model is approximately valid and that the shift-invariant sparse coding model is able to extract much of the information in a musical signal, such as note waveforms and score representations. However, it is also clear that not all information can be captured with a linear mixture model using a restricted set of components.

Other models have been proposed to extract meaningful features and structures without the use of prior musical knowledge. In the next chapter we compare one such method to the shift-invariant sparse coding formulation. This method is based on a phase-blind spectral signal model and uses the positivity of the spectrum together with a sparseness measure.

Chapter 9

Comparison to Phase-Blind Methods[1]

A small number of papers exist that present results on the extraction of time-domain features [5, 76, 1] from audio signals. These results differ from the work in this thesis in that the sets of sound stimuli were much larger to the ones used here and, more crucially, the number of features was substantially smaller than the effective number used in the shift-invariant sparse coding model (it should be remembered that the standard sparse coding model has to learn each feature at all shifted positions). These experiments could therefore not be expected to produce features that correspond to certain 'sound objects' as found in this thesis.

The shift-invariant sparse coding model is not the only possible approach based on sparseness or similar constraints able to extract such features from audio signals. The requirement for shift-invariance has led to the use of phase-blind spectral methods, which are less sensitive to the location of features in the analysed observation block. Most previous approaches based on sparse coding or non-negative matrix factorisation [74] or derivatives of these methods, have therefore concentrated on the analysis of audio spectrograms [1, 129, 66, 130, 143].

In order to better assess the differences between the time-domain approaches as used in this thesis and these spectral methods, we compare the shift-invariant sparse coding model to a recently proposed phase-blind

[1]The results in this chapter were part of a collaborative journal paper[108].

spectral method. The phase-blind spectral method is based on a non-negative sparse coding model and was developed to learn note features from audio spectrograms. The details of this method can be found in [108], in which most of the results presented in this chapter have previously been published.

In section 9.1 we describe the signal used and state the parameters used for the two models. The results obtained are then discussed, in section 9.2 we compare the learned features and in section 9.3 we compare the found sparse representations. In section 9.4 we take a closer look at how individual notes are represented with the two methods.

9.1 Methodology

In order to compare the two different models we used the results calculated with the shift-invariant sparse coding method in section 8.3 of the previous chapter. The same signal (summed to mono and resampled to 8 000 Hz) of a recording of Ludwig van Beethoven's Bagatelle Opus 33 No.1 was also used to train the non-negative sparse coding method based on spectrogram decomposition. In order to learn comparable features, we used a frame-size of 1024 samples for the spectral method, which led to a feature length of 513 samples. The number of features in the spectral domain dictionary was set to 101 elements, as this is the number of notes in the western equal tempered scale between the lowest and the highest note in the recording plus three. This dictionary was initialised with 98 harmonic features in half-tone steps from the lowest note in the signal to the highest note in the signal. The three remaining features were initialised with 'flat' vectors.

The spectral domain method was run for 3 hours on a 1.3GHz Apple PowerBook G4 laptop while the shift-invariant sparse coding method converged after approximately 24 hours on a Apple PowerMac 1.42 dual processor G4. However, to guarantee convergence, the shift-invariant method was run for an additional 6 days without any significant changes to the dictionary elements.

9.2 Comparison of Dictionary Elements

Both methods learned features that display note-like structure. The 57 features learned using the time-domain method are given in figure 9.1. Inspection of the spectrum of these features reveales, that 47 of the learned features (1 to 47) have harmonic structures and are here shown ordered by their estimated fundamental frequency. The other ten features (48 to 57) cannot be assigned to a fundamental frequency. With some exceptions, the features have time-support over their entire length.

We show the spectra of these features in the left panel of figure 9.2. Again we see the harmonic structure of the features and it is also evident that some of the features have similar fundamental frequencies. These features and the contribution they make to the reconstruction of a single note are investigated below.

The features learned using the spectral-domain method are shown in the right panel of figure 9.2. The dictionary was initialised with single frequency features, so that the learned features did not have to be ordered. Again, the harmonic structure of these features is evident. We can also observe that some features have similar fundamental frequencies.

The main difference observed between the two methods is a small decrease in the strength of harmonics with high frequencies for the results obtained with the shift-invariant time-domain approach. This might be due to the higher variance of the phase in the higher harmonics of a piano note, which leads to a 'smearing out' of these higher harmonics. Another reason for this decrease could be due to the filtering effects mentioned in section 3.3.2, in which we predicted this behaviour for sampled signals with features at continuous locations.

We also observed that, with both methods, features emerged that could not be assigned to an individual pitch, i.e. that had more than one harmonic series of spectral peaks. These features occurred less frequently with the shift-invariant sparse coding method as notes in a chord would need to be phase locked.

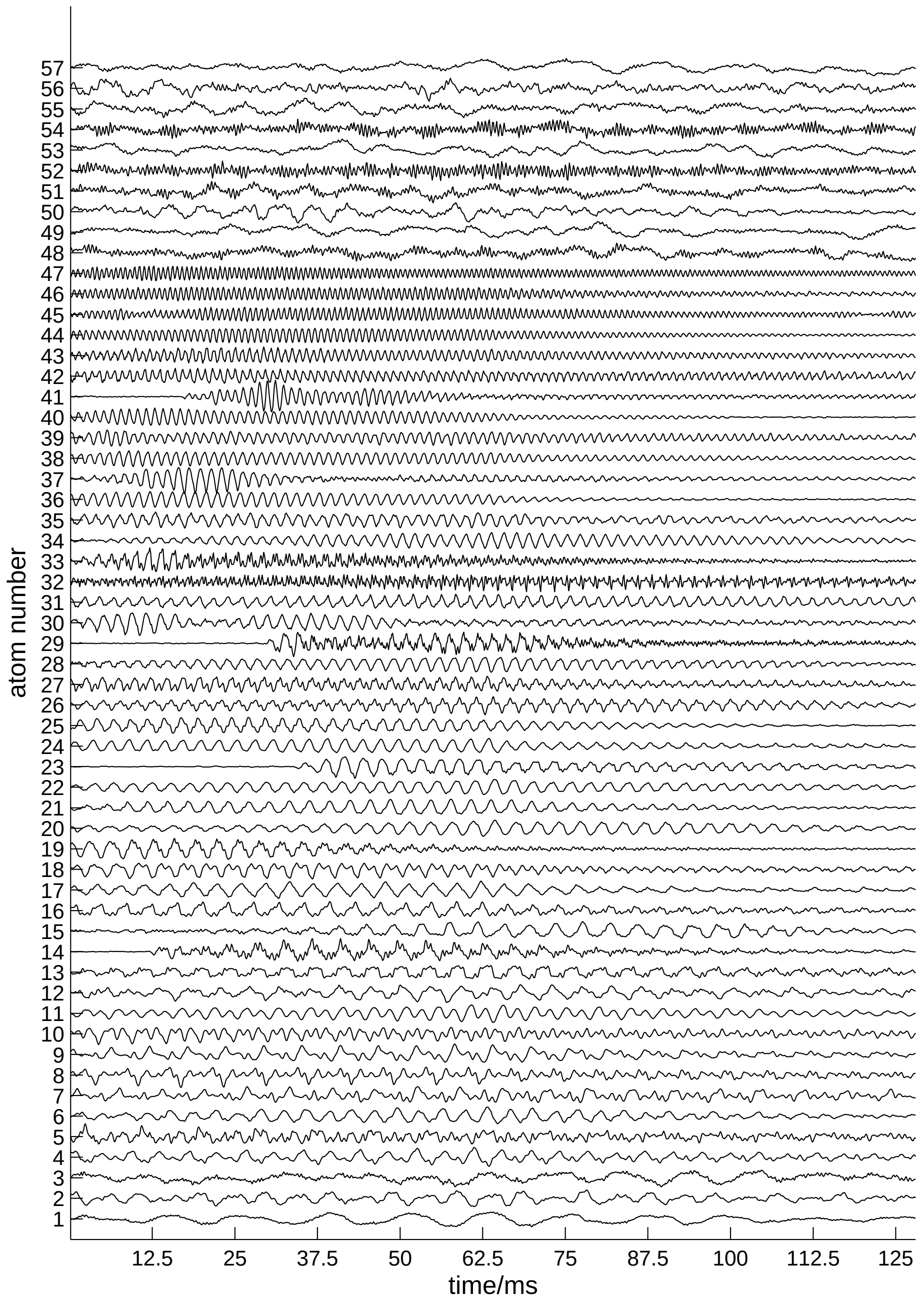

Figure 9.1: Time-domain waveforms of the dictionary learned using the time-domain method. The waveforms of each of the features are shown in each row, ordered by their estimated fundamental frequency. The top 10 features are those that could not be assigned to an individual fundamental frequency. Most features have a support of the full feature lengths, however some features (e.g. 13, 14, 22, 28 and 39) have a shorter time-support.

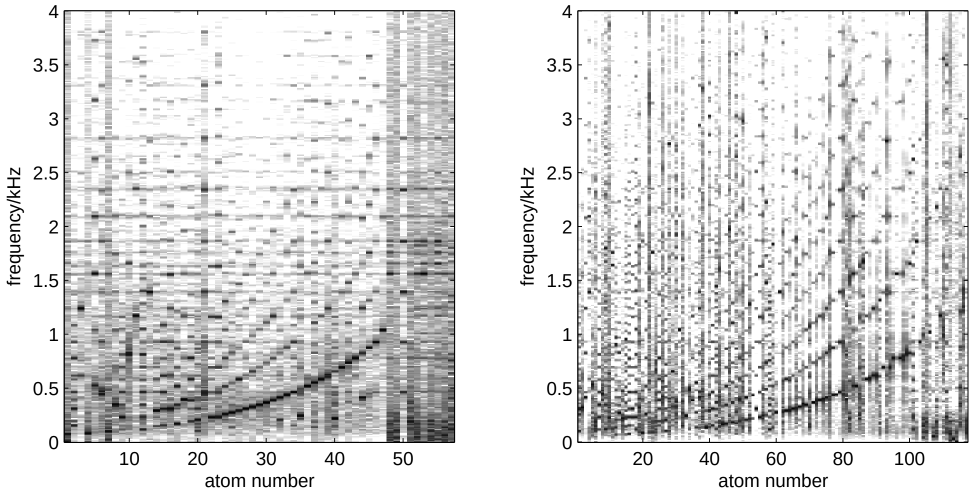

Figure 9.2: The log spectra of the dictionary learned using the time- (left) and spectral-domain (right) approach. The first 47 features learned using the shift-invariant sparse coding model have a clear harmonic structure and are ordered here by increasing fundamental frequency from left to right. The 10 features to the right in the left panel could not be assigned to any individual harmonic series. Spectra of the dictionary learned with the spectral-domain approach are shown in the right panel. As the dictionary elements were initialised with features with increasing frequency, reordering was not necessary. Most of these features clearly show the hormonic structure.

9.3 Comparison of Sparse Representations

Figures 9.3 shows the coefficients $\mathbf{s}$ found with the shift-invariant sparse coding approach. Here we have rectified the coefficients in order to clarify the presentation. We also omit the coefficients associated with non-harmonic features, as these did not significantly contribute to the representation.

A similar representation of the sparse coefficients found with the spectral method is shown in figure 9.4 and, for comparison, figure 9.5 shows the same part of the original Bagatelle score in a piano role notation. In figures 9.3 and 9.4 the coefficients have been ordered according to the estimated pitch of their associated features. The structures in these figures are clearly visible and many of the melodic lines, rhythmic patterns and chords in the score of the Bagatelle (figure 9.5) are evident in both sparse

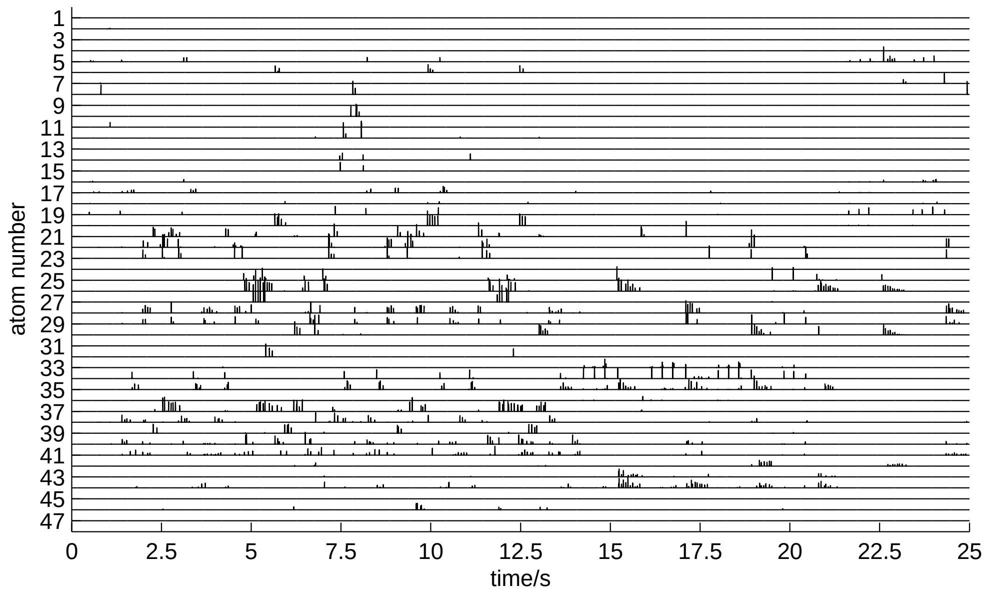

Figure 9.3: This figure shows the rectified activation of the different atoms found with the time-domain method. Again, the atoms are ordered by decreasing fundamental frequency.

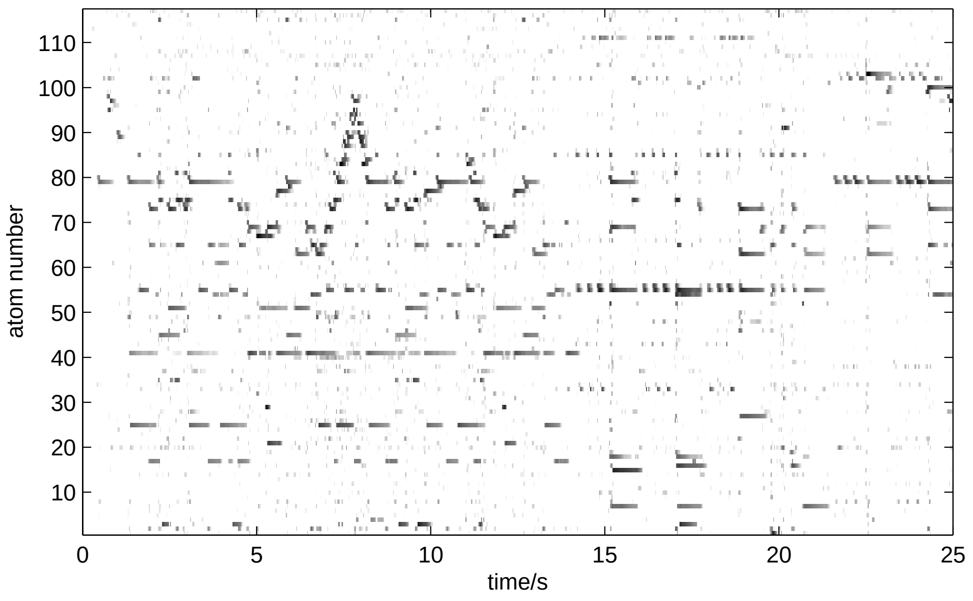

Figure 9.4: This figure shows the activation of the different atoms found with the spectral-domain method. Again the atoms are ordered by decreasing fundamental frequency.

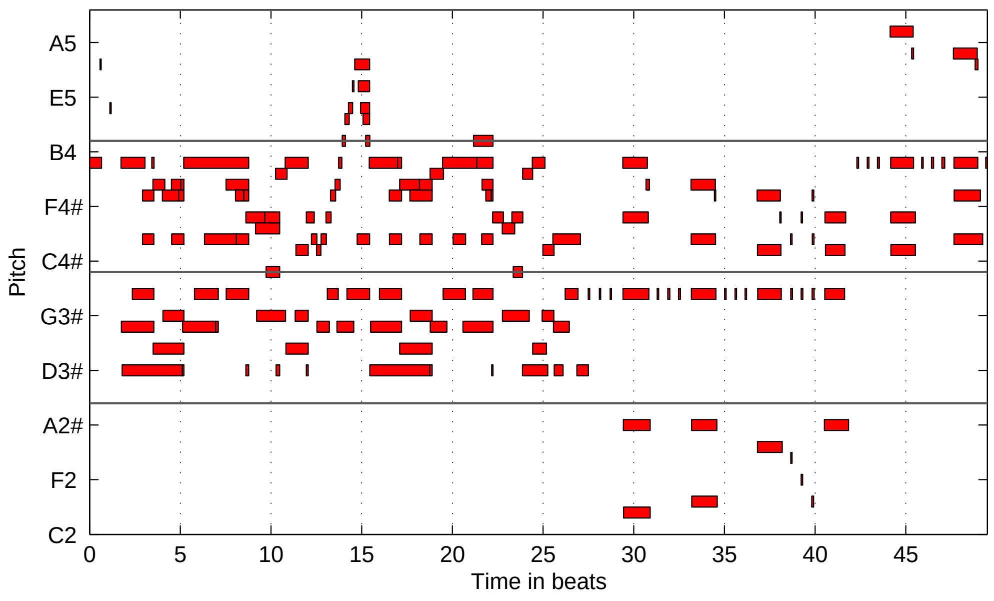

Figure 9.5: Representation of the notes from the Bagatelle Opus 33 No1. This 'piano-role' representation of the notes shows note pitch on the ordinate (higher notes on the top and lower ones on the bottom) and time along the abscissa. The bars represent the note activation.

representations (figures 9.3 and 9.4). However, as not all notes in the performance led to the emergence of associated features, some notes are not represented.

The main difference between these representations is that the shift-invariant sparse coding method leads to a representation in which clusters of spikes represent the occurrence of notes in the original score. The spectral method on the other hand leads to a representation with lower time resolution.

9.4 Representation of Notes by Multiple Components

As stated above, it was observed that several features learned using both approaches had the same fundamental frequency and therefore modelled a single note. We plot the original waveform of a piano note recorded with the same piano as the bagatelle in figure 9.6. The top panel shows the attack phase of the original note and the second panel shows the time domain waveform during the sustain of the note. In the third panel we

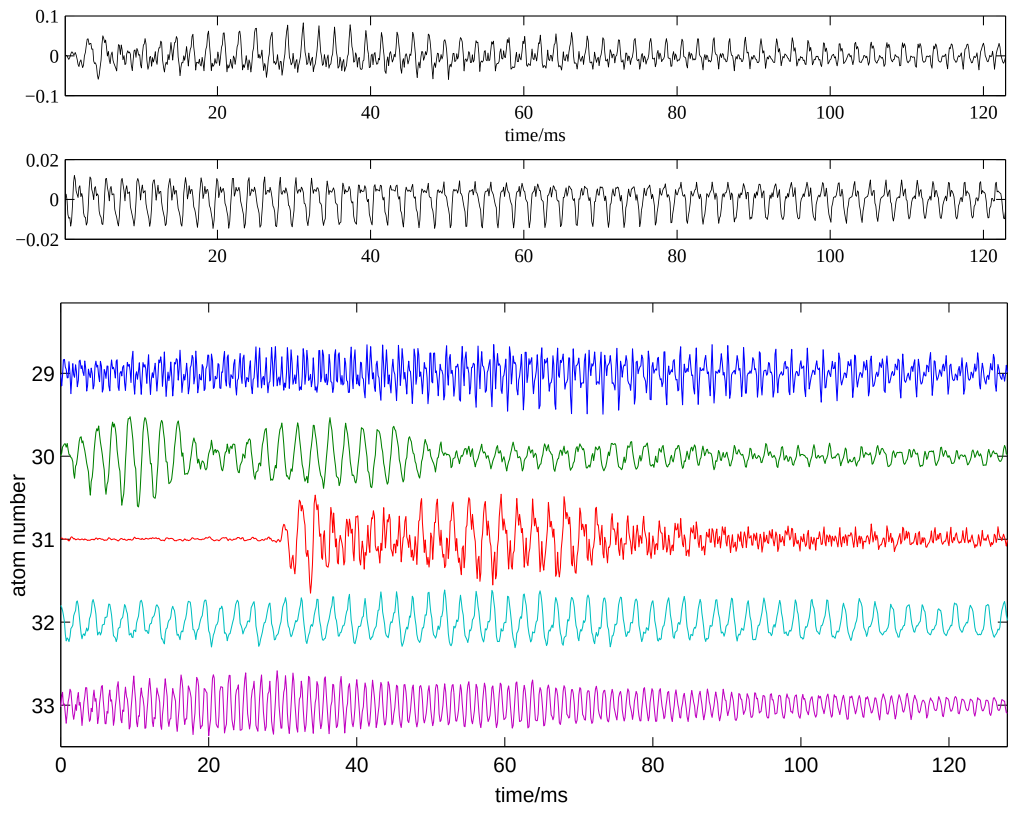

Figure 9.6: Comparison of the waveform of a single piano note with some waveforms of features learned using the time-domain method. The top two panels show the note onset and sustain of the same piano note respectively. This note was recorded in exactly the same way and using the same instrument as the training recording. The lowest panel shows five different features, all with the same fundamental frequency as the piano note in the first two panels. It is clear that the waveforms of the five features vary greatly, both with respect to the actual wave-shape as well as with respect to the overall time support and envelope.

show a set of five features learned using the shift-invariant sparse coding method, all of which have the same fundamental frequency as the note shown in the top two panels. In figure 9.7, we show the spectrum of these five features with the spectrum of the original piano note in the upper panel.

All five features differ significantly. The spectra of the features show a variation in the strengths of their individual harmonics, while the time-domain representation reveals their differing time-support and wave-shape.

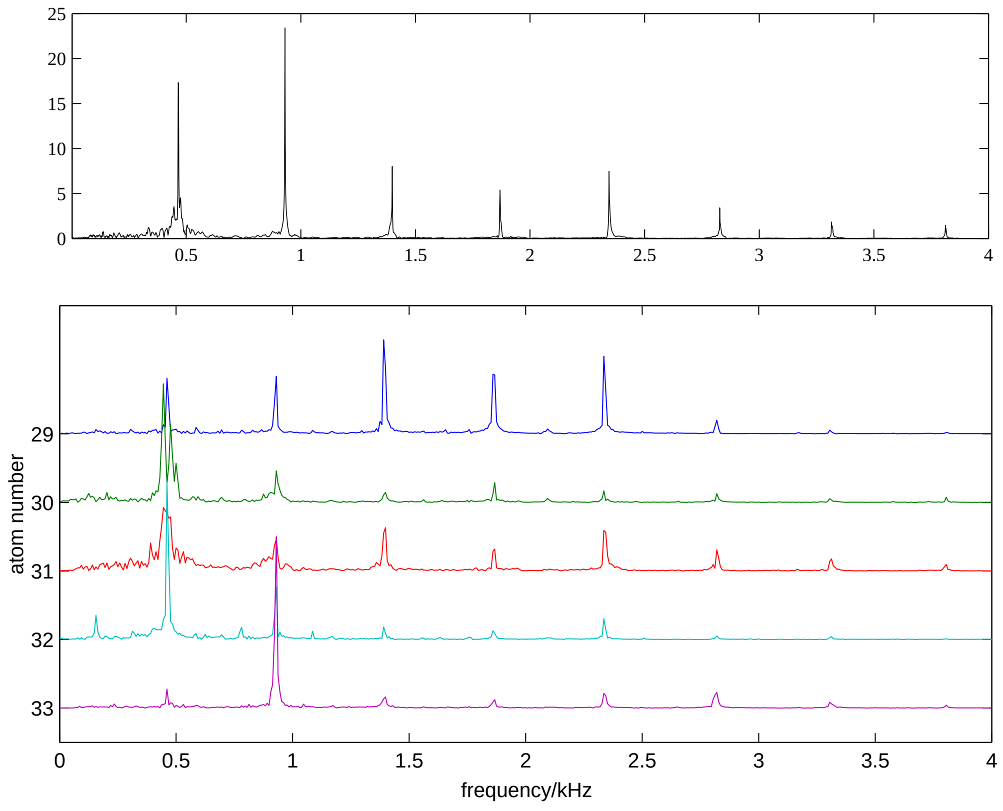

Figure 9.7: Comparison of the magnitude spectrum of a single piano note with the magnitude spectra of features learned using the time-domain method. The top panel shows the magnitude spectrum of the same piano note shown in figure 9.6. The lowest panel shows the magnitude spectra of the same features as shown in figure 9.6.

Figure 9.8 shows the coefficients **s** associated with these five features during a short time interval. In the piano score the note modelled by the five features is played twice, once at around 37.5 seconds and a second time at around 38.4 seconds. From figure 9.8 it can be seen that during the evolution of the note, different features from the above set become active. Feature 33 seems to model the note onset as it is only active at the beginning of the note. On both occasions this feature is followed by feature 29, which also only occurs during the note onset. Feature 30 is active only once during each note, but at different locations, while features 31 and 32 are modelling the sustain part of the note and are active

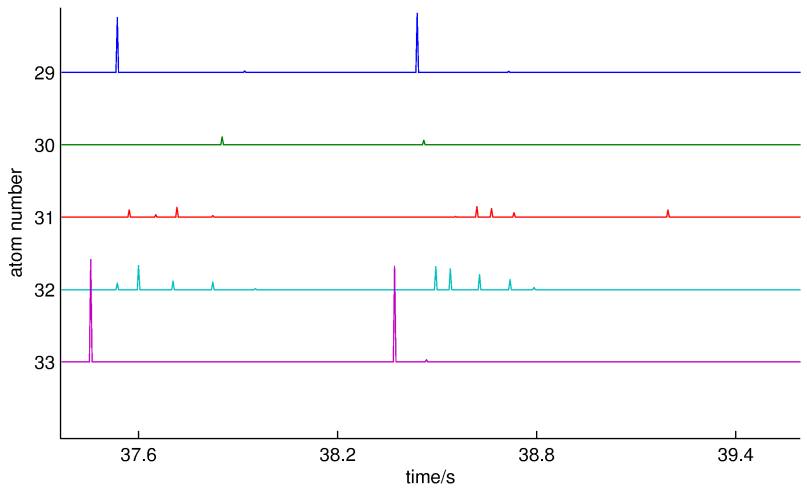

Figure 9.8: Activity of the five features shown in figures 9.6 and 9.7 over the duration of two notes. In this figure the activation (again rectified) of the five features is shown whilst modelling the note being played twice. This clearly shows how the different features are combined to reconstruct a single note. Features 29 and 33 are only active during note onset, feature 30 is only used once in each note while the other two features are used repeatedly with varying magnitude.

repeatedly throughout the note. From the spectra of the features it is clear that feature 29, which models the note onset, has much stronger higher harmonics. A similar observation can be made for feature 33, which has most of its energy in the second harmonic. This is in line with the well known fact that many musical sounds, such as piano notes, have more energy in their higher harmonics during note onset. During the evolution of the note, the energy in the higher harmonics decreases faster than the energy in lower harmonics. This is clearly reflected in the harmonic content of the features that model different parts of the note.

For the features extracted with the spectral method, features with equal fundamental frequency also emerged. The two features with the same fundamental frequency as the note used above are shown in figure 9.9, with the spectrum of the original note shown in the top panel. The different strengths of the high frequency harmonics is once again evident in the two features learned.

Figure 9.10 shows the activation of these features for the same two notes as in the example above. It is again clear that the features model

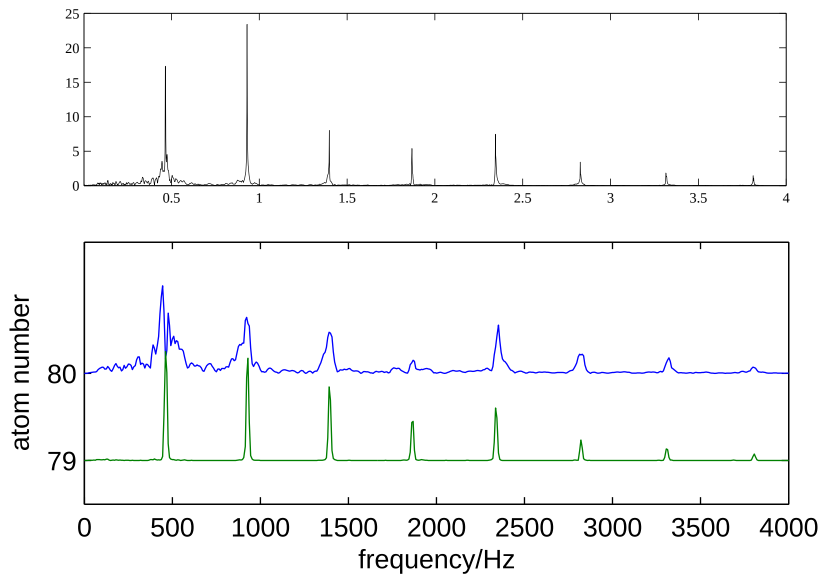

Figure 9.9: Comparison of the magnitude spectrum of two features learned using the spectral-domain method that are modelling a single note.

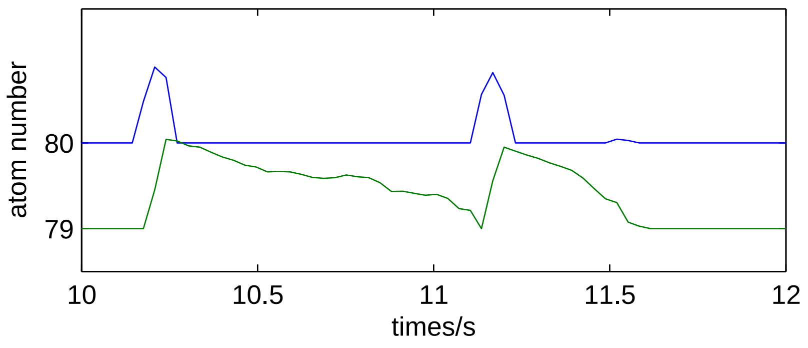

Figure 9.10: Activity of the two features shown in figure 9.9 over the duration of two notes. In this figure the activation of the two features is shown whilst modelling the note being played twice. This shows how the different features are combined to reconstruct a single note. Feature one is mainly active during the note onset while feature two is used repeatedly with varying magnitude through the steady state of both notes.

different parts of the note, with feature 80 modelling the note onset and feature 79 modelling the sustain part of the note. The spectra of these features also shows that the feature for the note onset has stronger high frequency harmonics.

Both models are not able to represent the varying spectrum of a piano note with a single feature. Nevertheless, only a small number of the notes in the composition were represented by multiple features. In the experiment with the bagatelle we found that 5 notes were represented by more than one feature when learned using the time-domain method, while 12 notes were represented by more than one note when learning used the spectral domain method. The notes with multiple representations were those notes that occurred very frequently in the recording.

The main difference between the two approaches with respect to the approximation is the higher over-completeness in the time-domain approach. It is clear from the previous figures that the time-domain approach offers sample accurate location of features, while the feature location for the spectral-domain approach is fixed to the window location used in the transform. This can be clearly seen in figures 9.8 and 9.10

Figure 9.4 reveals another artefact in the representation produced by the spectral method. As was shown above, during the onset of a note the spectrum typically has a much wider frequency support than during the sustain of the note. In the spectral method this often leads to the activation of several features that have a different fundamental frequency to the note to be modelled. This artefact has not been observed with the time-domain approach. However, this might be due to the much stronger sparsity measure used in the time-domain approximation. Nevertheless, spurious activity of features can also be observed with the shift-invariant sparse coding method, although these are not generally associated with note onsets.

Conclusions

Phase blind spectral methods offer an alternative approach to deal with the ambiguity of feature location in time-series and have therefore been used to model musical signals. In this chapter we have shown that these methods extract similar structures to those found with the shift-invariant sparse coding model. These structures include the representation of individual notes, often by more than one feature, and the emergence of score-like structures.

Several differences were, however, observed; the shift-invariant sparse coding model offers sample accurate timing, which cannot be achieved with the spectral method. We also observed that the features extracted by the spectral method had slightly stronger high frequency harmonics than the features found by the shift-invariant sparse coder, which seems to be due to the variation in the phase of these high frequency components in the piano signal. This observation also confirms the findings presented in chapter 3. Nevertheless, the shift-invariant sparse coding was less prone to finding features that corresponded to more than one harmonic series. The phase information learned also leads to a direct signal reconstruction, while for the phase blind method, the phase has to be estimated. However, these advantages came at the cost of greatly increased computational complexity.

Chapter 10

Single Channel Source Separation[1]

We can think of the separation of several source in terms of a weighted assignment of time-frequency points to each source. If different sources overlap in frequency, linear transforms such as the short-time Fourier transform cannot be used directly for this assignment. Sparse coding methods on the other hand offer such an assignment and can be used to learn source models with overlapping time-frequency support.

Previous approaches to single channel blind source separation reported in the literature either rely on prior knowledge of a source model for each source to be recovered (see for example [142] and [61]) or treat the extracted features as individual sources (see for example [130] and [143]). The models in [142, 130] and [143] are further based on phase-blind spectral models that recover the sources by Wiener filtering methods.

In this chapter we investigate the performance of the shift-invariant sparse coding algorithm for single channel blind source separation in the case where it cannot be assumed in general that individual notes have similar waveforms each time they occur. We nevertheless show that for more general musical signals, features can be extracted that can be assigned to individual sources in the mixture. This classification then leads to a reconstruction of a signal using only those features corresponding to a single source.

The main problem with this approach is the assignment of the individual features to each of the sources. In section 10.2 we use knowledge

[1]This chapter is taken to a large extent from [13].

of the source signals themselves in order find such a clustering. This enables us to derive an upper bound on source separation performance. In section 10.3 we then develop an unsupervised clustering approach based on features extracted from both the features $\mathbf{a}_k$ as well as their associated coefficients $\mathbf{s}$.

10.1 Methodology

For this experiment we recorded two separate signals; a vocal and a guitar track, which were mixed linearly and resampled to 8 000Hz. It is important to stress that these signals were musically related, i.e. both guitar and voice where performing the same musical piece in harmony and with the same tempo so that both sources had much structure in common. We used this single channel mixture as a training sequence for the algorithm. We learned 500 features of length 256 samples in a similar fashion to the experiments reported in the previous chapters. Of the 500 features 129 had converged after 500 000 iterations, while the remaining features had not been updated substantially. In this experiment all of the converged features had a clear harmonic structure. This can be seen in figure 10.1 where we show an extract of the coefficients $\mathbf{s}$ (left) associated with the learned features shown in the time domain (middle) and the spectral domain (right). Here we only show those features which could be clearly associated with a certain source using prior information (see below).

10.2 Oracle Clustering

In order to analyse the possible performance of the shift-invariant sparse coding method for blind source separation we first perform separation of the sources by assigning the learned features to each source based on knowledge of the actual sources themselves, i.e. we use a non-blind (oracle) method.

The oracle assignment of features to sources was done depending on the energy each feature contributed to the representation of the individual

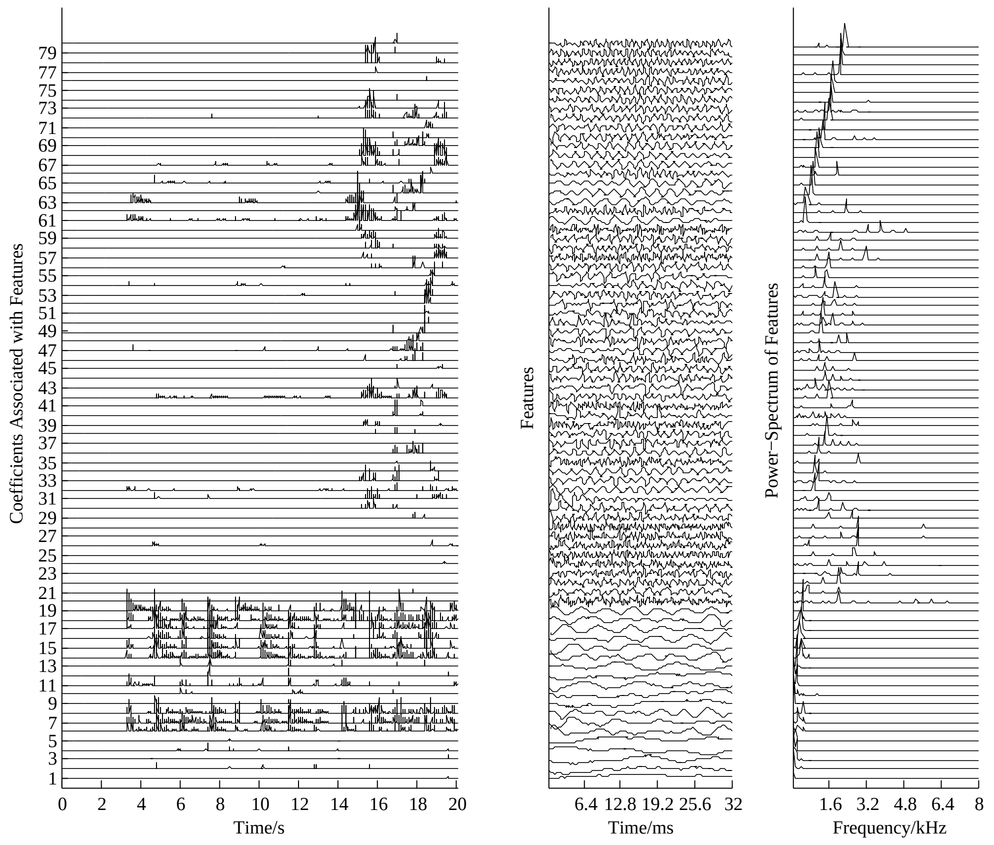

Figure 10.1: Decomposition coefficients for the first 20 seconds of the piece (left), the associated time domain features (centre) and their power-spectrum (right).

sources, which was determined as:

$$p_{k,vox} = \frac{\|s_{k,vox}\|}{\|s_{k,vox}\| + \|s_{k,guitar}\|},$$

with $s_{k,vox}$ ($s_{k,guitar}$) denoting the coefficients associated with feature k when analysing the original vocal (guitar) signal. Different clusters could then be built by assigning features to a source whenever $p_k > P$ for some P. The results below are given for different values of $P < 1$. Note that $P = 0$ corresponds to the case in which all features are assigned to both sources, $P = 0.5$ corresponds to the case where each feature is assigned to a single source and $P > 0.5$ corresponds to the case where some features are not assigned to any of the sources. For $P = 0.9$ we could assign 80 of the 129 features to a single source. These are the 80 features shown with

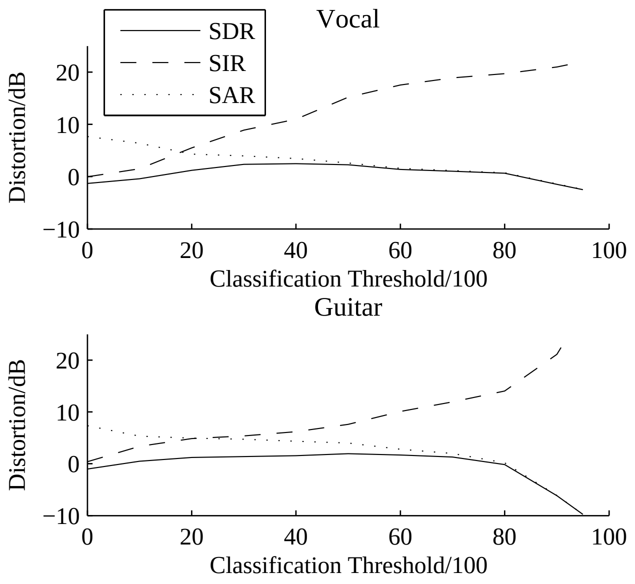

Figure 10.2: Distortion (in dB) for the separated sources. Vocal (top) and guitar (bottom) and their associated distortions; SDR (solid), SIR (dashed) and SAR (dotted).

their coefficients in figure 10.1.

After this clustering we used the coefficients **s** from the decomposition of the mixture to reconstruct the sources using only those features assigned to each individual source. The performance of this separation was then measured using the method proposed in [50]. This gives us a measure of the signal to inference ratio (SIR), i.e. the ratio of the true source to the interference of the other sources in the estimated source, as well as the signal to artefact ratio (SAR), i.e. a measure of the artefacts introduced by the method. We can also calculate the overall signal to distortion ration (SDR). For further details the reader may refer to [50].

The top panel of figure 10.2 shows the SDR (solid), the SIR (dashed) and the SAR (dotted) results for the vocal reconstruction while the lower panel gives the results for the guitar reconstruction. The SIR increases when fewer features are assigned to a source, whilst the overall SDR peaks at around 40% (vocal) and 50% (guitar) but is generally quite insensitive to the threshold. It is also clear that as fewer features are used in the reconstruction the SAR decreases as more artefacts are introduced. The

SIR (at $P = 0.9$) for the vocal reconstruction was 21dB while the SIR for the guitar reconstruction at this value was also 21dB. This means that the guitar track was suppressed by 21dB in the vocal reconstruction. However, this reduction in interference between the sources leads to the introduction of artefacts. For the SIR levels reported above the signal to artefact ratios were -1.4dB and -6.1dB respectively. It can also be seen that even the reconstruction of the signal with all features is not artefact free and the highest SAR is 7dB for this example.

10.3 Unsupervised Clustering

In real situations, the information used for clustering in the previous subsection is not available and other methods for assigning features to sources are required. In previous methods (e.g. [142]) the features and models of the sources were learned from training sequences. However, different recordings of even the exact same instrument might change the recorded waveforms if the microphone position is changed or the recording made in another acoustic environment so that such a prior assignment is not feasible. Instead it is necessary to cluster the features based only on the information available from the single mixture that was used in the feature learning procedure.

To facilitate clustering we exploit higher level dependencies not modelled in the shift-invariant sparse coding model. In particular, we exploit the residual dependencies found in the coefficients $\mathbf{s}$ as well as dependencies between the features $\mathbf{a}_k$.

The coefficients $\mathbf{s}$ have been modelled as independent and identically distributed variables. However, for real sources, observations are not independent from previous observations and the coefficients $\mathbf{s}$ are not independent over time.

In order to exploit temporal information in the coefficients $\mathbf{s}$ that has been ignored by the shift-invariant sparse coding algorithm, we estimate the probability of occurrence of a feature during a short time interval

$$p_t(\tilde{k}, i) = p(l \in [l_i, l_{i+1}) : s_{kl} \neq 0, k = \tilde{k})$$

by

$$p_t(\tilde{k}, i) \approx \frac{1}{\sum_i \sum_{l \in [l_i, l_{i+1})} |s_{\tilde{k}l}|} \sum_{l \in [l_i, l_{i+1})} |s_{\tilde{k}l}|.$$

The above histogram estimation does not only count the occurrence of the features but also takes their strength into account, which can be justified by assuming that a larger coefficient s is a sum of smaller 'quantum' coefficients. This feature can also be thought of as a smoothed and down-sampled version of the coefficients $\mathbf{s}$ and is based on the activation patterns of the coefficients s, which are assumed to be similar for features $\mathbf{a}_k$ associated with a single source. Other features, such as a histogram estimate of the distribution of the coefficients s associated with each feature $\mathbf{a}_k$ or features based on the autocorrelation of or the cross-correlation between the coefficients s associated with each feature $\mathbf{a}_k$ were found not to work well for unsupervised clustering.

Individual instruments often have fixed physical characteristics that shape the spectrum of the produced sounds in a characteristic manner. The features $\mathbf{a}_k$ associated with the same source can therefore be assumed to be similar. This similarity can be measured based on a spectral feature calculated by smoothing the power-spectrum of the features $\mathbf{a}_k$, which is done here by averaging the energy in the spectrum over a partitioning of the frequency range. In [76] the statistics of natural sounds have been shown to lead to efficient codes that have a wider frequency support at high frequencies. It was further argued in [76] that for speech, music and some natural sounds the average power spectrum is approximately $1/f$ so that in order for each frequency band to have equal average power, the width of the frequency bands has to increase linearly with frequency. This is reflected in the frequency-discrimination found in the human auditory system, which is known to roughly follow a logarithmic scale. Therefore, we use a logarithmic frequency-domain partitioning of each feature $\mathbf{a}_k$ and calculate the feature as:

$$p_f(\tilde{k}, i) \propto \sum_{l \in [2^i, 2^{i+1})} |\tilde{\mathbf{a}}_{\tilde{k}}(l)|^2$$

where $\tilde{\mathbf{a}}_{\tilde{k}}$ is the Fourier transform of feature $\mathbf{a}_{\tilde{k}}$. A linear partitioning is possible, however, for the experiments reported here, the results obtained were slightly worse than those obtained with the logarithmic partitioning.

Clustering of the features $\mathbf{a}_k$ can then be performed using standard clustering algorithms. Here we use the K-means algorithm. As a distance measure between the individual features we use the symmetric Kullback-Leiber divergence, which can be justified if we think of the features as histogram estimates of probability measures. The symmetric Kullback-Leiber divergence is

$$\begin{aligned} KL(p(k,i), p(\tilde{k},i)) &= 0.5 \sum_i p(k,i) \log \frac{p(k,i)}{p(\tilde{k},i)} \\ &+ 0.5 \sum_i p(\tilde{k},i) \log \frac{p(\tilde{k},i)}{p(k,i)}, \end{aligned}$$

where $p(k,i)$ and $p(\tilde{k},i)$ are the two features to be compared.

In addition to the features p_t and p_f we can also use a combination of these two features for clustering. The results obtained with these different features are shown in table 10.1. It is evident that for the example studied here, the feature p_t outperforms the feature p_f, a combination of both features, however, offers the best overall performance. We also show the results obtained with the oracle performance in the previous subsection for $P = 0.5$.

To show the trade-off between the SIR and the SAR, it is again possible to assign a feature to more than one source or to assign some features to no source at all. This can be done by introducing a margin (positive, to assign some features to no sources and negative, to assign some features to more than one source). The SIR, SAR and SDR values are given in

	p_f	p_t	$[p_t, p_f]$	Oracle P=0.5
SIR vocal	11.5	12.6	11.8	15.2
SIR guitar	4.7	9	9.9	7.6
SAR vocal	-0.2	3	3.2	2.6
SAR guitar	3.7	3.3	3	4.0
SDR vocal	-0.8	2.3	2.4	2.3
SDR guitar	0.4	1.8	1.8	1.9

Table 10.1: Comparison between the features for clustering.

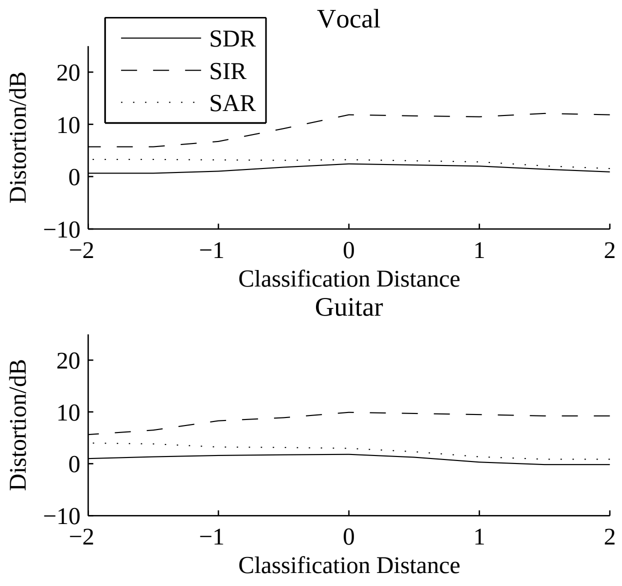

Figure 10.3: Distortion (in dB) for the blindly separated sources. Vocal (top) and guitar (bottom) and their associated distortions; SDR (solid), SIR (dashed) and SAR (dotted).

figure 10.3 for different margins. Here we show the results for clustering based on the combined features. The values obtained with a margin of zero are those shown in table 10.1. Again, the SDR is quite insensitive to the margin used, however, the change in SIR and the inverse change in SAR are less pronounced.

Conclusions

The human voice can produce a wide range of acoustic signals, which can vary significantly with respect to their energy envelope, their spectral characteristics as well as their phase spectra. Nevertheless, in this chapter we have shown that shift-invariant sparse coding can learn features from such a signal that capture salient structure in the singing voice. Furthermore, we have shown that features learned from a mixture of human singing voice and guitar can often be assigned to a single source. This approach can then be used for single channel blind source separation. The assignment of features to a source, must be done using an unsupervised technique and we have proposed a method based on features extracted from the coefficients $\mathbf{s}$ as well as the features $\mathbf{a}_k$. The blind source separation performance of this clustering method was found to be similar to the blind source separation performance achieved with the oracle method.

Chapter 11

Conclusions and Further Work

11.1 Conclusion

Signal processing methods find a huge number of applications in a variety of scientific and engineering disciplines. In many of these applications it is necessary to extract certain features and structures from observations. Often, these features and structures have to reflect certain aspects of the processes underlying an observed signal and should be able to offer further insight into these processes. Unfortunately, it is not always possible to specify such features and structures a priori and techniques able to discover such features from observations alone are therefore of immense value.

In this thesis we have investigated a sparse coding model that is able to discover salient signal structure. The method is based on a linear generative model and uses a shift-invariant structure in order to deal with salient features in time-series. The main theoretical concept involved in the work presented here is that of sparsity, i.e the assumption that salient features occur sparsely. Without any further domain specific prior knowledge it was shown that such a strong assumption can lead to the discovery of many of the underlying features and structures in a signal. As an interesting application domain, which has enabled a detailed analysis of the method, we have concentrated on musical signals. Theses signals are generated containing a large amount of structures, such as harmonics, notes, melodies, rhythms and chords and we have shown that the shift-invariant sparse coding model developed in this thesis can extract note-like features and find score-like representations from these signals.

In addition to sparsity, positivity is a strong constraint, significantly reducing the solution space. For many problems, even though the exact form of features is unknown, positivity can often be assumed for certain features. We have shown that the inclusion of this constraint further improves the performance of the developed method. Again, this was done using a musical example, assuming that notes in a piano recording have non-negative amplitudes.

In the introduction, the four main contributions made in this thesis were listed. Here we revisit these points and summarise the main advances and findings in these four areas.

- Subset selection:

  The subset selection step restricts the number of features used to model each observation block. The experiments reported here show that this restriction does not significantly restrict the algorithms ability to extract salient features from musical recordings and that this restriction was actually necessary in order to apply the shift-invariant sparse coding model to the problems studied here. The experiments in chapter 7 showed that the approximation of the learning rule based on a delta approximation of the posterior of $\mathbf{s}$ estimated using only the selected subset, offered nearly the same level of performance as the Gibbs sampling method that used the non-negative prior.

- Importance sampling for shift-invariant sparse coding:

  The importance sampling algorithm introduced in chapter 5 offers a fast method to approximate the learning rule. For small dictionary sizes this method was found to offer comparable learning performance to the other methods. However, for larger problems such as the applications to music studied here, the bias introduced by this method becomes significant and the results obtained were found to be significantly worse than those obtained by the other methods based on a subset selection step. This suggests that the subset selection step introduces less bias in the learning rule.

- Gibbs sampling with a novel positive prior for shift-invariant sparse coding:

For piano music a more accurate signal model can be specified by the introduction of the modified Rayleigh distribution as a novel conjugate prior for the Gaussian mean. Approximations of the learning rule can be calculated based on Markov chain Monte Carlo methods. However, these methods are computationally demanding. The random subset selection procedure introduced in chapter 6, which is based on the idea that the Markov chain is not able to explore the complete posterior in any reasonable time, forces the chain to concentrate on certain areas of the distribution that are likely to include much of the probability mass and was found to lead to good approximations with finite computational resources. In appendix B, other methods to increase the performance of the Gibbs sampler were proposed and studied. Unfortunately, most of these did not offer significant improvements.

- Application to Music:

  Music approximately follows the additive signal model used here and piano music in particular can be roughly modelled as a linear combination of note prototypes. This makes musical signals an ideal application domain in which to study shift-invariant sparse coding. In this thesis it has be shown that this method can extract a variety of structures from such signals and, to our knowledge, these results are the first results that show the emergence of such structures from real world time-series. Two main applications to music analysis have been studied in this thesis; the extraction of musically relevant features such as notes and score and blind separation of single channel mixtures.

  – Emergence of musical structure: Different musical structures emerged from the application of shift-invariant sparse coding to piano music. The features $\mathbf{a}_k$ converged to note-like atoms that could be assigned to individual piano notes. The coefficients $\mathbf{s}$ associated with these atoms contained information about the occurrence of notes in the recording. The similarity of the coefficients $\mathbf{s}$ to the score of the analysed music and the structure of the features $\mathbf{a}_k$ was shown in chapters 8 and 9.

- Blind source separation: In chapter 10 we have shown that the shift-invariant sparse coding model leads to representations in which individual features contain information that is primarily associated with a single source. The shift-invariant sparse coding model can therefore be used to separate different sources from a single mixture. In order to blindly separate different sources, an unsupervised clustering algorithm was introduced and it was shown that clustering of the features can be based on a combination of two features, one capturing spectral information of the features and the other capturing average occurrence of a feature during different intervals.

11.2 Open Problems and Further Work

Any PhD project like the one presented here throws up more new questions than it can possibly answer. In this section we discuss some of the unanswered problems related to the work presented here and discuss several possible directions for further study. In the following, we have grouped the problems to be addressed into three main categories, work extending the signal model, work improving the algorithms used for learning and inference and work on specific applications.

- Model:
  - It has been repeatedly reported in this thesis that not all of the features converged and that some features were not updated during learning. This problem seems to be inherent to the competitive learning procedure used, where once a set of features has been learned that models most observations relatively well, no other additional features are learned to model the residual. This might be overcome by increasing the update for features that occur less frequently. In our model we have assumed that all features occur with equal probability, i.e. we have used the

same prior formulation for all features. In the Monte Carlo algorithms, prior parameters were adapted and it is possible to adapt the probability of occurrence of the features individually. This probability can then be used in the updating step to give more weight to less frequently occurring features.

Another possible approach could be to use a second modelling step to model the residual. This would also overcome the restriction introduced with the subset selection step, which only uses features that model the observation well on their own but do not model residual structures well.

– In this thesis we have exploited the idea that meaningful features occur sparsely in observations. Another possible constraint, which might be used to extract meaningful features, would be to utilise time consistency, i.e. to use the assumption that features generally occur over certain time intervals. As discussed in chapter 3, this has previously been exploited in Slow Feature Analysis in [151] and similar ideas could be combined with sparseness and positivity constraints. With such a model, it might be possible to learn structure over larger time-scales.

– The additive linear model used here is rather restrictive and does not model all possible instruments well. For the piano example used in this thesis, this model worked relatively well, however, for more complex instruments, more complex models are required. In general, there is no reason why more complex instrument models should not be used instead of the note prototypes used here, although in this case the computational complexity would also increase. Nevertheless, a more complicated model can offer better performance in the tasks studied here and could be based on ideas similar to those in [142], in which a phase-blind spectral signal model was introduced that is more flexible than the one used in this thesis.

Computational savings could be made in a model by assuming more structure on these models than was assumed here. For example, by assuming that instrument sounds are periodic signals, models with fewer free parameters could be constructed. However, the aim of the work in this thesis was to show that musically relevant structures such as harmonic atoms could emerge from a simple model with no prior musical constraints.

- In this thesis it has become evident that much residual structure remains in the coefficients **s** that is not captured by the model. This was exploited in the section on blind source separation, in which this structure was used to cluster features into sources. How such structures can be modelled is not yet clear and different approaches have to be evaluated. One possible method would be to look at correlations and higher order relationships between the coefficients **s** of the same and of different features. However, it is difficult to estimate such correlations even if the coefficients are assumed to be a stationary process, which is clearly not the case. On the contrary, this non-stationarity is one of the main structures remaining in the signal.

- Further structures can be modelled and the simple linear model used here can be refined by taking expected structures into account. This can be done by developing more refined models of the coefficients **s** similar to those suggested in [142], however, these methods have to be adapted for the shift-invariant sparse coding model. Such methods could then lead to better inference and learning performance.

  Another model refinement could be the introduction of additional structure on the features $\mathbf{a}_k$. Such structure could include expected harmonic relationships or other knowledge of source structure. In this thesis the focus was on the emergence of such structures from simpler models, nevertheless, performance can be increased by including such structures in the model a priori.

- The location of features in the analysed time-series were assumed to be uniformly distributed a priori. In order to improve inference, additional information could be used to estimate these locations. Such information could come from onset detection algorithms often used in music analysis. Such algorithms could guide the shift-invariant sparse coding method and lead to faster implementations. Other information such as estimates of harmony or the key of the particular performance could also be used in a similar manner.

- Algorithm
  - Several methods have been analysed in this thesis in order to improve the efficiency of the Monte Carlo methods. One further possibility for improving the sampling strategies would be the inclusion of a gradient term in the proposal distribution that guides the Markov chain to areas of higher probability. However, the model used here has a high dimensional discrete state space in which gradient information is not available. Whether similar methods are applicable to such state spaces could be investigated. One possible approach might be to use Reversible Jump Markov chain sampling in which local information is used to guide the evolution of the chain as well as the jumps between different models.
  - The motivation for the introduction of the importance sampling method was to find a method which could be used to calculate a very rough but unbiased gradient estimate. It was found that the bias in the importance sampling algorithm was too large for the applications of interest. In order to improve this method it is therefore necessary to find methods to reduce this bias. How this could be achieved is, however, not yet clear. One possible approach might be to use an annealed Gibbs sampler (as introduced in the appendix) to calculate a proposal used in a subsequent rejection sampler. The Gibbs sampler would be used to achieve a proposal that is accepted with relatively high probability. Such a single sample, or a small number of such samples might then be used to estimate the gradient. Whether this strategy offers any computational advantages depends on

the required computational complexity of calculating the proposal and, more critically, on the achievable acceptance rate.

- Applications:
  - Previous applications of shift-invariant sparse coding have mainly been in image analysis, a problem domain in which the additive signal model is clearly not valid. Apart from the application to musical signals studied in this thesis, there are a large number of other signals that approximately follow the additive linear model. Possible application areas include biomedical signal analysis of (for example, EEG data as in [18] or fetal ECG data,) geological signal analysis and blind deconvolution in communication systems.
  - The results presented in this thesis on blind source separation of a single channel observation are only of a preliminary nature and a much more thorough evaluation must be undertaken. It could be seen that many of the learned features could be associated with a single source, however, some features were used for modelling of both sources. In order to increase the performance, the model has to be adapted to better model individual sources. The simple linear model used here is clearly restricted in this respect. Modelling of additional structures in the coefficients $\mathbf{s}$, as well as in the features $\mathbf{a}$ as suggested above seems a possible solution. In order to achieve a blind separation of single channel recordings, good generative source models are required, which have to be learned from the mixture itself. These models have to be flexible enough to model a wide range of different sources. Furthermore, a method to estimate the number of sources in the signal is required. For such an estimation, sparsity might be used by searching for the smallest number of sources able to describe an observation. The source models then need to be able to describe a single source, but not a mixture of sources.
  - In this thesis we have applied the shift invariant sparse coding model only to a single observation sequence. If more than one

observation is available, to each of which the different sources contribute with different delays and possibly with different amplitudes, then the shift-invariant sparse coding model can be used to estimate this delay and therefore the direction of arrival of the different signals. Two possible approaches could be taken; in the first, the features are shared between the different observations. This approach can be used if the delay between the observations can be approximated accurately with a multiple of the sampling time and if the waveforms observed are not otherwise changed by filtering effects. The delay is then encoded in the delay between the coefficients $\mathbf{s}$ for the same feature between the different observations.

The other approach would assume different features for each source and observation but use a common set of coefficients $\mathbf{s}$ for all observations. This approach can be used if features have slightly different waveforms at different observations and if the delays are of sub-sample length. The delay between the observations can then be estimated from the different phase of the features.

– One possible application area for sparse coding that has not been analysed in this thesis is that of signal coding and compression. A problem to be addressed is that the linear generative model used here leads to a relatively large reconstruction error. This is the result of the very strong sparsity constraint used in order to extract musically meaningful features. High quality coding would require much weaker sparsity constraints or an even higher number of features, both of which would lead to an increase in coding cost. This also leads to the question of the optimal size of the dictionary used for coding, as a larger dictionary leads to a higher number of bits required to specify the non-zero coefficients. Another important issue to be addressed for coding is the use of perceptual distortion measures and how such measures can be incorporated into sparse coding methods.

– Depending on the application, a different trade off between sparsity and reconstruction error is required. This relationship has not been investigated in detail in this thesis. From a Bayesian point of view, such information should be incorporated using utility functions. However, it is common practice to incorporate such information in parameter priors. For the model used in this thesis, this could be done by introducing prior distributions on the parameters governing this trade off. A more heuristic approach could be based on fixed parameters, set by experience gained from simulation studies. All of these approaches are necessarily application driven and might depend on related parameters to be optimised such as rate distortion in coding.

Bibliography

[1] S. Abdallah, "Towards music perception by redundancy reduction and unsupervised learning in probabilistic models," Ph.D. dissertation, King's College London, February 2003.

[2] C. Andrieu, P. Djuric, and A. Doucet, "Model selection by MCMC computation," *Signal Processing, Special Issue on MCMC for Signal Processing*, vol. 81, no. 1, pp. 19–37, 2001.

[3] H. B. Barlow, "Possible principles underlying the transformations of sensory messages," in *Sensory Communication*, W. A. Rosenblith, Ed. The MIT Press, 1961, pp. 217–234.

[4] M. S. Bartlett, *An Introduction to Stochastic Processes.* Cambridge University Press, 1960.

[5] A. J. Bell and T. J. Sejnowski, "Learning the higher-order structure of a natural sound," *Network: Computation in Neural Systems*, vol. 7, pp. 261–266, 1996.

[6] J. P. Bello, "Towards the automated analysis of simple polyphonic music: A knowledge-based approach," Ph.D. dissertation, Queen Mary, University of London, 2003.

[7] L. Benaroya, L. M. Donagh, F. Bimbot, and R. Gribonval, "Non-negative sparse representation for Wiener based source separation with a single sensor," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, 2003, pp. 613–616.

[8] E. Bingham and A. Hyvärinen, "A fast fixed-point algorithm for independent component analysis of complex-valued signals," *Int. J. of Neural Systems*, vol. 10, no. 1, pp. 1–8, 2000.

[9] T. Blumensath and M. Davies, "Unsupervised learning of sparse and shift-invariant decompositions of polyphonic music," in *Proc. of the*

IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, 2004, pp. V–497–500.

[10] ——, "On shift-invariant sparse coding," in *Proc. of the Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2004, pp. 1205–1212.

[11] ——, "Enforcing sparsity, shift-invariance and positivity in a Bayesian model of polyphonic music," in *Proc. of the IEEE Workshop on Statistical Signal Processing*, July 2005.

[12] ——, "A fast importance sampling algorithm for unsupervised learning of over-complete dictionaries," in *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing*, March 2005, pp. 213–216.

[13] ——, "Sparse and shift-invariant representations of music," *submitted for publication*, vol. 14, no. 1, pp. 50–57, Jan 2006.

[14] R. Bogacz, M. W. Brown, and C. Giraud-Carrier, "Emergence of movement-sensitive neurons' properties by learning a sparse code of natural moving images," in *Advances in Neural Information Processing Systems (NIPS)*, 1999, pp. 838–844.

[15] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal of Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[16] C. Chennubhotla and A. Jepson, "Sparse coding in practice," in *Proc. of the Second Int. Workshop on Statistical and Computational Theories of Vision*, 2001.

[17] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 713–718, 1992.

[18] A. d'Avella, P. Saltiel, and E. Bizzi, "Combinations of muscle synergies in the construction of a natural motor behavior," *Nature*, vol. 6, no. 3, pp. 300–308, March 2003.

[19] G. Davies, "Adaptive nonlinear approximations," Ph.D. dissertation, New York University, 1994.

[20] M. Davies, T. Blumensath, and L. Daudet, "Generalised IRLS," *in preparation.*

[21] M. Davies and L. Daudet, "Sparsifying subband decompositions," in *Proc. of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003, pp. 107–110.

[22] M. Davies and N. Mitianoudis, "A simple mixture model for sparse overcomplete ICA," *IEE Proc.-Vision, Image and Signal Processing*, vol. 151, no. 1, pp. 35–43, August 2004.

[23] L. De Lathauwer, "Signal processing based on multilinear algebra," Ph.D. dissertation, Faculty of Engineering, K.U.Leuven, Leuven, September 1997.

[24] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[25] J. Diebolt and C. P. Robert, "Estimation of finite mixture distributions through Bayesian sampling," *Journal of the Royal Statistical Society. Series B*, vol. 56, no. 2, pp. 363–375, 1994.

[26] O. Divorra Escoda, L. Granai, and P. Vandergheynst, "On the use of a priori information for sparse signal approximations," EPFL, Tech. Rep. 23/2004, November 2004.

[27] D. L. Donoho, "On minimum entropy segmentation," Dept. Statistics, Stanford University, Tech. Rep., 1994.

[28] D. L. Donoho and R. R. Coifman, *Wavelets and Statistics.* Berlin, Germany: Springer Verlag, 1995, ch. Translation-Invariant De-Noising.

[29] A. D'Souza, S. Vijayakumar, and S. Schaal, "Bayesian backfitting relevance vector machine," in *Proc. of the 21st Int. Conf. on Machine Learning*, 2004.

[30] K. Engan, "Frame based signal representation and compression," PhD Thesis, Stavanger University College, Norway, 2000.

[31] C. Fernandez and M. Steel, “On Bayesian inference under sampling from scale mixtures of normals,” Tilburg University, Center for Economic Research, Tech. Rep., 1996.

[32] C. Févotte and C. Doncarli, “Two contributions to blind source separation using time-frequency distributions,” *IEEE Signal Processing Letters*, vol. 11, no. 3, pp. 386–389, 2004.

[33] M. A. T. Figueiredo, “Adaptive sparseness using Jeffreys prior,” in *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, December 2001, pp. 697–704.

[34] ——, “Adaptive sparseness for supervised learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1150–1159, Sept. 2003.

[35] M. A. T. Figueiredo and A. K. Jain, “Bayesian learning of sparse classifiers,” in *Proc. of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR)*, Hawaii, December 2001, pp. 35–41 Vol. 1.

[36] M. A. T. Figueiredo and R. D. Nowak, “Wavelet-based image estimation: an empirical Bayes approach using Jeffrey’s non-informative prior,” *IEEE Transactions on Image Processing*, vol. 10, no. 9, pp. 1322–1331, Sept. 2001.

[37] P. Földiák, “Forming sparse representations by local anti-hebian learning.” *Biological Cybernetics*, vol. 64, no. 2, pp. 165–170, 1990.

[38] ——, “Learning invariance from transformation sequences,” *Neural Computation*, vol. 3, pp. 194–200, 1991.

[39] ——, “Models of sensory coding,” Ph.D. dissertation, Churchill College, Cambridge, 1991.

[40] B. J. Frey and N. Jojic, “Transformation-invariant clustering and the EM algorithm,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 1–17, Jan 2003.

[41] J.-J. Fuchs, “On sparse representations in arbitary redundant bases,” *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1341–1344, 2004.

[42] ——, “Recovery of exact sparse representations in the presence of bounded noise,” Institut de Recherche en Informatique et Systemes Aléatoires, Tech. Rep., 2004.

[43] J. Geweke, “Variable selection and model comparison in regression,” in *Bayesian Statistics 5.*, J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, Eds. Oxford University Press, 1996.

[44] M. Girolami, “A variational method for learning sparse and overcomplete representations,” *Neural Computation*, vol. 11, pp. 2517 – 2532, 2001.

[45] S. J. Godsill, “On the relationship between Markov chain Monte Carlo methods for model uncertainty,” *Journal of Computational and Graphical Statistics*, vol. 10, no. 2, pp. 230–248, 2001.

[46] P. J. Green, “Iterative reweighted least squares for maximum likelihood estimation, and some robust resistant alternatives,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 46, no. 2, pp. 149–192, 1984.

[47] ——, “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination,” *Biometrika*, vol. 82, no. 4, pp. 711–732, 1995.

[48] R. Gribonval, R. M. Figueras, and P. Vandergheynst, “A simple test to check the optimality of a sparse signal approximation,” Institut de Recherche en Informatique et Systemes Aléatoires, Tech. Rep. 1661, 2005.

[49] R. Gribonval and M. Nielsen, “Highly sparse representations are unique and independent of the sparseness measure,” Dept. of Mathematical Sciences, University of Aalborg, Denmark, Tech. Rep. R2003-16, 2003.

[50] R. Gribonval, L. Benaroya, E. Vincent, and C. Févotte, “Proposals for performance measurement in source separation,” Institut de Recherche et Coordination Acoustique/Musique, Tech. Rep. 1501, 2003.

[51] D. Grimes and R. Rao, "A bilinear model for sparse coding," in *Advances in Neural Information Processing Systems (NIPS)*, 2003, pp. 1287–1294.

[52] C. Han and B. P. Carlin, "Markov chain Monte Carlo methods for computing Bayes factors: A comparative review," *Journal of the American Statistical Association*, vol. 96, no. 455, pp. 1122–1134, 2001.

[53] G. F. Harpur, "Low entropy coding with unsupervised neural networks," Ph.D. dissertation, Queens College, Cambridge, Februar 1997.

[54] W. Harrison, L. Bamber, and S. Haykin, *Adaptive Filter Theory.* Prentice Hall, 2003.

[55] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *Journal of Machine Learning Research*, vol. 5, pp. 1457–1469, 2004.

[56] A. Hyvärinen and A. Hoyer, "Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces," *Neural Computation*, vol. 12, no. 7, pp. 1705–1720, 2000.

[57] A. Hyvärinen, "Independent component analysis in the presence of Gaussian noise by maximizing joint likelihood," *Neurocomputing*, vol. 22, pp. 49–67, 1998.

[58] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis.* John Wiley & Sons Inc, 2001.

[59] T. S. Jaakkola, "Variational methods for inference and estimation in graphical models," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 1997.

[60] S. Jaggi, W. Carl, S. Mallat, and A. Willsky, "High resolution pursuit for feature extraction," *Applied and Computational Harmonic Analysis*, vol. 5, no. 4, pp. 428–449, Oct 1995.

[61] G.-J. Jang and T.-W. Lee, “A probabilistic approach to single channel source separation,” in *Advances in Neural Information Processing Systems (NIPS)*, 2002, pp. 1173–1180.

[62] ——, “A maximum likelihood approach to single-channel source separation,” *Journal of Machine Learning Research*, vol. 4, pp. 1365–1392, 2003.

[63] S. M. Kay, *Fundamentals of statistical signal processing: Estimation Theory.* Prentice Hall, 1993.

[64] N. G. Kingsbury, “Complex wavelets for shift invariant analysis and filtering of signals,” *Journal of Applied and Computational Harmonic Analysis*, vol. 10, no. 3, pp. 234–253, May 2001.

[65] R. Kohn, M. Smith, and D. Chan, “Nonparametric regression using linear combinations of basis functions,” *Statistics and Computing*, vol. 11, pp. 301–322, 2001.

[66] K. P. Körding, P. König, and D. J. Klein, “Learning of sparse auditory receptive fields,” in *Proc. of the Int. Joint Conf. of Neural Networks*, vol. 2, 2002, pp. 1103–1108.

[67] D. Korže and D. Zazula, “Identifiability of the superimposed signals using third-order cumulants,” in *Proc. of the IEEE Signal Processing Workshop on Higher-Order Statistics*, July 1997, pp. 331–335.

[68] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, “Dictionary learning algorithms for sparse representation,” *Neural Computation*, vol. 15, no. 2, pp. 349–396, 2003.

[69] K. Kreutz-Delgado and B. D. Rao, “A general approach to sparse basis selection: Majorization, concavity, and affine scaling,” University of California, San Diego, Tech. Rep. UCSD-CIE-97-7-1, July 1997.

[70] ——, “Sparse basis selection, ICA and majorization: towards a unified perspective,” in *Procceedings of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 2, 1999, pp. 1081 –1084.

[71] K. Kreutzer-Delgado, B. D. Rao, and K. Engan, “Convex/ shure-convex (CSC) log-priors and sparse coding,” in *Proc. of the 6th*

Joint Symposium on Neural Computation, Pasadena, California, May 1999.

[72] H. J. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications.* Springer-Verlag New York, 2003.

[73] D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *Advances in Neural Information Processing Systems (NIPS)*, 2000, pp. 556–562.

[74] ——, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 21, pp. 788–791, Oct 1999.

[75] T.-W. Lee, M. Lewicki, M. Girolami, and T. Sejnowski, “Blind source separation of more sources than mixtures using overcomplete representations,” *IEEE Signal Processing Letters*, pp. 87–90, 1998.

[76] M. S. Lewicki, “Efficient coding of natural sounds,” *Nature Neuroscience*, vol. 5, no. 4, pp. 356–363, April 2002.

[77] ——, *Probabilistic Models of the Brain: Perception and Neural Function.* MIT Press, 2002, ch. Efficient Coding of Time-Varying Signals Using a Spiking Population Code, pp. 223–234.

[78] M. S. Lewicki and B. A. Olshausen, “A probabilistic framework for the adaptation and comparison of image codes,” *J. Opt. Soc. Am. A: Optics, Image Science, and Vision*, vol. 16, no. 7, pp. 1587–1601, 1999.

[79] M. S. Lewicki and T. J. Sejnowski, “Coding time-varying signals using sparse, shift-invariant representations,” in *Advances in Neural Information Processing Systems (NIPS)*, vol. 11, 1999, pp. 730–736.

[80] ——, “Learning overcomplete representations,” *Neural Computation*, no. 12, pp. 337–365, 2000.

[81] Y. Li, A. Cichocki, and S. Amari, “Analysis of sparse representation and blind source separation,” *Neural Computation*, vol. 16, pp. 1193–1234, 2004.

[82] ——, “Sparse component analysis for blind source separation with less sensors than sources,” in *Proc. of the Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2003, pp. 89–94.

[83] J. S. Liu, “Peskun’s theorem and a modified discrete-state Gibbs sampler,” *Biometrika*, vol. 83, no. 3, pp. 681–682, Sep. 1996.

[84] Y. Z. M. Zibulevsky, “Extraction of a single source from multichannel data using sparse decomposition,” *Neurocomputing*, vol. 49, pp. 163–173, 2001.

[85] D. J. C. MacKay, “Comparison of approximate methods for handling hyperparameters,” *Neural Computation*, vol. 11, no. 5, pp. 1035–1068, 1999.

[86] S. Mallat, *A Wavelet Tour of Signal Processing.* Academic Press, 1999.

[87] S. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[88] R. E. McCulloch and E. I. George, “Approaches for Bayesian variable selection,” *Statistica Sinica*, vol. 7, no. 2, pp. 339–374, 1997.

[89] ——, “Variable selection via Gibbs sampling,” *Journal of the American Statistical Association*, pp. 881–889., September 1993.

[90] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, ser. Wiley Series in Probability and Statistics: Applied Section. John Wiley & Sons Inc, 1996.

[91] A. Miller, *Subset selection in regression*, 2nd ed. Chapman and Hall, 2002.

[92] J. W. Miskin, “Ensemble learning for independent component analysis,” Ph.D. dissertation, Selwyn College, Cambridge, 2000.

[93] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing.* Addison Wesley, September 1999.

[94] P. Moulin and J. Liu, “Analysis of multiresolution image denoising schemes using generalized Gaussian and complexity priors,” *IEEE Transactions on Information Theory*, vol. 45, no. 3, pp. 909–919, April 1999.

[95] J. F. Murray and K. Kreutz-Delgado, “An improved FOCUSS-based learning algorithm for solving sparse linear inverse problems,” in *Conf. Record of the Thirty-Fifth Asilomar Conf. on Signals, Systems and Computers*, 2001, pp. 347–351.

[96] P. B. Nair, A. Choudhury, and A. J. Keane, “Some greedy algorithms for sparse nonlinear regression,” in *Proc. of the Eighteenth Int. Conf. on Machine Learning*, 2001, pp. 369 – 376.

[97] R. M. Neal, “Bayesian training of backpropagation networks by the hybrid Monte Carlo method,” Dept. of Computer Science, University of Toronto, Tech. Rep. CRG-TR-92-1, 1992.

[98] ——, “Probabilistic inference using Markov chain Monte Carlo methods,” Dept. of Statistics, University of Toronto, Tech. Rep. CRG-TR-93-1, 1993.

[99] ——, “Sampling from multimodal distributions using tempered transitions,” Dept. of Statistics, University of Toronto, Tech. Rep. 9421, 1994.

[100] ——, “Annealed importance sampling,” Dept. of Statistics, University of Toronto, Tech. Rep. 9805, 1998.

[101] J. A. Nelder and R. W. M. Wedderburn, “Generalized linear models,” *Journal of the Royal Statistical Society, Series A (General)*, vol. 135, no. 3, pp. 370–384, 1972.

[102] B. A. Olshausen, “Sparse coding of time-varying natural images,” in *Proc. of the Int. Conf. on Independent Component Analysis and Blind Source Separation*, 2000.

[103] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, vol. 13, no. 381, pp. 607–609, Jun 1996.

[104] ——, “Sparse coding with an overcomplete basis set: A strategy employed by V1?” *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[105] B. A. Olshausen and K. Millman, “Learning sparse codes with a mixture-of-gaussians prior,” in *Advances in Neural Information Processing Systems (NIPS)*, 2000, pp. 841–847.

[106] B. A. Olshausen, P. Sallee, and M. S. Lewicki, “Learning sparse image codes using a wavelet pyramid architecture,” in *Advances in Neural Information Processing Systems (NIPS)*, vol. 13, 2001, pp. 887–906.

[107] E. Plévin and D. Zazula, “Decomposition of surface EMG signals using non-linear LMS optimisation of higher-order cumulants,” in *Proc. of the 15th IEEE Symposium on Computer-Based Medical Systems*, 2002, pp. 149–154.

[108] M. Plumbley, S. Abdallah, T. Blumensath, and M. Davies, “Sparse representations of polyphonic music,” *to appear in ELSEVIR Signal Processing.*

[109] B. D. Rao, “Signal processing with the sparseness constraint,” *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1861–1864, May 1998.

[110] B. D. Rao and I. F. Gorodnitsky, “Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm,” *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 600–616, March 1997.

[111] B. D. Rao and K. Kreutz-Delgado, “Measures and algorithms for best basis selection,” *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1881–1884, May 1998.

[112] ——, “An affine scaling methodology for best basis selection,” *IEEE Transactions on Signal Processing*, vol. 47, no. 1, pp. 187–200, January 1999.

[113] F. Rieke, D. Warland, R. R. de Ruyter van Steveninck, and W. Bialeck, *Spikes, Exploring the Neural Code.* MIT Press, 1997.

[114] H. Robbins and S. Monro, “A stochastic approximation method,” *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[115] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, ser. Springer Texts in Statistics. Springer-Verlag, 1999.

[116] R. Rockafellar, *Convex Analysis.* Princeton University Press, 1970.

[117] E. T. Rolls and G. Deco, *Computational Neuroscience of Vision.* Oxford University Press, 2001.

[118] E. T. Rolls, M. C. M. Eliffe, and S. M. Stringer, “Invariant recognition of feature combinations in the visual system,” *Biological Cybernetics*, vol. 86, pp. 59–71, 2002.

[119] E. T. Rolls and T. Milward, “A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures,” *Neural Computation*, vol. 12, pp. 2547–2572, 2000.

[120] E. T. Rolls and N. Parga, “Transform invariant recognition by assosiation in a recurrent network,” *Neural Computation*, vol. 10, pp. 1507–1525, 1998.

[121] E. T. Rolls and S. M. Stringer, “Invariant object recognition in the visual system with error and temporal difference learning,” *Network: Computation in Neural Systems*, vol. 12, pp. 111–129, 2001.

[122] E. T. Rolls and G. Wallis, “Invariant object recognition in the visual system,” *Progress in Neurobiology*, vol. 51, pp. 167–194, 1997.

[123] D. Rowe, “A Bayesian approach to blind source separation,” *Journal of Interdisciplinary Mathematics*, vol. 5, no. 1, pp. 49–76, 2002.

[124] ——, “Bayesian source separation with jointly distributed mean and mixing coefficients via MCMC and ICM,” Division of Humanities and Social Sciences, California Institute of Technology, Pasadena, CA 91125, Social Science Working Paper 1119, 2001.

[125] ——, “Bayesian source separation of functional sources,” *Journal of Interdisciplinary Mathematics*, vol. 6, no. 2, pp. 129–138, 2003.

[126] P. Sallee and B. A. Olshausen, “Learning sparse multiscale image representations,” in *Advances in Neural Information Processing Systems (NIPS)*, 2003, pp. 1327–1334.

[127] K. Skretting, J. H. Husøy, and S. O. Aase, “A simple design of sparse signal representations using overlapping frames,” Senter for Informasjons og Kommunikasjonsteknologi, Tech. Rep. SIKTPR1/4, 2000.

[128] K. Skrettis, “Sparse signal representation using overlapping frames,” Ph.D. dissertation, Stavanger University College, Stavanger, 2002.

[129] P. Smaragdis, “Redundancy reduction for computational audition, a unifying approach,” Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 2001.

[130] ——, “Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs,” in *Proc. of the Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2004, pp. 494–499.

[131] E. Smith and M. S. Lewicki, “Efficient coding of time-relative structure using spikes,” *Neural Computation*, vol. 17, no. 1, pp. 19–45, Jan 2005.

[132] M. Smith and R. Kohn, “Nonparametric regression using Bayesian variable selection,” *Journal of Econometrics*, vol. 75, no. 2, pp. 317–343, 1996.

[133] D. J. Stracuzzi and P. E. Utgoff, “Randomized variable elimination,” *Journal of Machine Learning Research*, vol. 5, pp. 1331–1362, 2004.

[134] F. J. Theis, “A geometric algorithm for overcomplete linear ICA,” in *Proc. of Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2003, pp. 167–172.

[135] F. J. Theis and E. Lang, “Geometric overcomplete ICA,” in *Proc. 10-th European Symposium on Artificial Neural Networks (ESANN'02)*, 2002, pp. 217–222.

[136] F. J. Theis, C. Puntonet, and E. W. Lang, “A histogram-basd overcomplete ICA algorithm,” in *Proc. of the Int. Conf. on Independent*

Component Analysis and Blind Source Separation (ICA), 2003, pp. 1071–1076.

[137] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[138] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[139] ——, "Just relax: Convex programming methods for subset selection and sparse approximation," The University of Texas at Austin, ICES Report 04-04, February 2004.

[140] J. A. Tropp, A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss, "Improved sparse approximation over quasi-incoherent dictionaries," in *Proc. of the 2003 IEEE Int. Conf. on Image Processing*, Barcelona, September 2003.

[141] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated annealing : theory and applications.* Kluweiler Academic Publishers, 1988.

[142] E. Vincent and X. Rodet, "Underdetermined source separation with structured source priors," in *Proc. of the Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2004, pp. 327–334.

[143] T. Virtanen, "Separation of sound sources by convolutive sparse coding," in *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

[144] K. Waheed and F. M. Salem, "Algebraic overcomplete independent component analysis," in *Int. Conf. on Independent Component Analysis and Blind Source Separation (ICA)*, 2003, pp. 1077–1082.

[145] M. J. Wainwright, E. P. Simoncelli, and A. S. Willsky, "Random cascades on wavelet trees and their use in analysing and modelling natural images," *Applied and Computational Harmonic Analysis*, vol. 11, pp. 98–123, 2001.

[146] X. Wen and M. Sandler, “Transcribing piano recordings using signal novelty,” in *Proc. of the AES Conf.*, Barcelona, Spain, May 2005.

[147] H. Wersing, J. Eggert, and E. Körner, “Sparse coding with invariance constraints,” in *Proc. Int. Conf. Artificial Neural Networks ICANN*, 2003, pp. 385–392.

[148] H. Wersing and E. Körner, “Learning optimized features for hierarchical models of invariant recognition.” *Neural Computation*, vol. 15, no. 7, pp. 1559–1588, 2003.

[149] M. West, “On scale mixtures of normal distributions,” *Biometrika*, vol. 74, no. 3, pp. 646–648, Sep. 1987.

[150] D. Wipf and B. D. Rao, “Sparse Bayesian learning for basis selection,” *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[151] L. Wiskott and T. Sejnowski, “Slow feature analysis: Unsupervised learning of invariances.” *Neural Computation*, vol. 14, no. 4, pp. 715–770, 2002.

[152] P. J. Wolfe, S. J. Godsill, and W. J. Ng, “Bayesian variable selection and regularisation for time-frequency surface estimation,” *Journal of the Royal Statistical Society, Series B*, vol. 66, no. 4, pp. 575–589, 2004.

[153] P. J. Wolfe, S. J. Godsill, and M. Doerfler, “Audio signal modelling using Bayesian atomic decompositions,” in *110th Convention of the Audio Engineering Society.* Audio Engineering Society, 2002.

[154] D. Zazula and D. Korze, “Higher-order cumulants in system identification and signal decomposition,” in *Proc. of the 19th ÖAGM and 1st SDVR Workshop, V: Visual modes*, Wien; München: Oldenbourg, 1995.

[155] M. Zibulevsky and P. Bofill, “Underdetermined blind source separation using sparse representations,” *Signal Processing*, vol. 81, no. 11, pp. 2353–2362, 2001.

[156] ——, “Blind separation of more sources than mixtures using sparsity of their short-time fourier transform,” in *Proc. of the Int. Conf.*

on Independent Component Analysis and Blind Source Separation (ICA), June 2000, pp. 87–92.

[157] M. Zibulevsky, P. Kisilev, Y. Y. Zeevi, and B. Pearlmutter, "Blind source separation via multinode sparse representation," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 1049–1056.

[158] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[159] M. Zibulevsky, B. A. Pearlmutter, P. Bofill, and P. Kisilev, *Independent Component Analysis: Principles and Practice.* Cambridge, 2001, ch. Blind Source Separation by Sparse Decomposition.

Appendix A

Derivation of Gibbs Sampler Probability

From equation (6.3) we have:

$$\begin{aligned} E_1 &= \log \frac{p(u_n = 1) \int p(\mathbf{x}|\mathbf{s}, \mathbf{u}, \theta) p(s_n|u_n = 1)\ ds_n}{p(u_n = 0) \int p(\mathbf{x}|\mathbf{s}, \mathbf{u}, \theta) p(s_n|u_n = 0)\ ds_n} \\ &= \log \frac{e^{-0.5\lambda_u} \int e^{-0.5\lambda_\epsilon (\mathbf{x}-\mathbf{As})^T(\mathbf{x}-\mathbf{As})} \frac{1}{Z^p} s_n e^{-0.5\lambda_R (s_n - \mu_n)^2}\ ds_n}{\int e^{-0.5\lambda_\epsilon (\mathbf{x}-\mathbf{As})^T(\mathbf{x}-\mathbf{As})} \delta_0(s_n)\ ds_n} \end{aligned}$$

where Z_p is the normalising constant of the modified Rayleigh distribution given in equation (6.4). Using

$$b_n = (\mathbf{a}_n^T \mathbf{a}_n)^{-1} \mathbf{a}_n^T \mathbf{x},$$

$$\lambda_{E_n} = \lambda_\epsilon \mathbf{a}_n^T \mathbf{a}_n,$$

$$\eta_n = \frac{\lambda_{E_n} b_n + \lambda_R \mu_n}{\lambda_{E_n} + \lambda_R}$$

and

$$\Psi_n = \lambda_{E_n} + \lambda_R,$$

we can write this as

$$\begin{aligned} E_1 &= \log \frac{e^{-0.5\lambda_u} \int e^{-0.5\lambda_{E_n} (b_n - s_n)^T (b_n - s_n)} \frac{1}{Z^p} s_n e^{-0.5\lambda_R (s_n - \mu_n)^2}\ ds_n}{\int e^{-0.5\lambda_{E_n} (b_n - s_n)^T (b_n - s_n)} \delta_0(s_n)\ ds_n} \\ &= \log \frac{e^{-0.5\lambda_u} \frac{1}{Z^p} \int s_n e^{-0.5\Psi_n s_n^2 + \Psi_n \eta_n s_n - 0.5(\lambda_{E_n} {b_n}^2 + \lambda_R {\mu_n}^2)}\ ds_n}{e^{-0.5\lambda_{E_n} (b_n)^2}} \\ &= \log e^{0.5\lambda_{E_n} b_n^2} e^{-0.5\lambda_u} \frac{Z_E}{Z^p} \int s_n e^{-0.5\Psi_n (s_n - \eta_n)^2}\ ds_n, \end{aligned}$$

where in the last line we use

$$Z_E = e^{0.5(\Psi_n \eta^2 - \lambda_{E-n} b_n^2 - \lambda_R \mu_n^2)}.$$

The integral in the last line is the normalising constant in the modified Rayleigh distribution given in equation (6.4) so that the expression for E_1 in chapter 6 follows.

Appendix B

Gibbs Sampler Performance

In this appendix we discuss three different approaches developed in order to increase the performance of the Gibbs sampler for sparse linear models. These approaches are *metropolisation* discussed in section B.1, bridged transitions developed in section B.2 and problem specific proposal distributions introduced in B.3.

For the *Metropolisation* method each normal Gibbs step is replaced by a Metropolis-Hastings step to sample from the conditional distribution.[1] For discrete state spaces (such as the indicator variable in the mixture distribution) this method was suggested to improve performance, especially with sparse distributions [65, 83].

The bridged transition method is an extension of the method developed in [99] for Metropolis-Hastings sampler. Here we present the proofs for irreducibility and the convergence to the stationary distribution if the method is applied to the Gibbs sampler.

The problem-specific proposal introduced in section B.3 takes account of the structure of the problem and in particular the shift-invariant nature of the model under study. Using this structure we introduce a hybrid sampler that randomly replaces the Gibbs sampler with a Metropolis-Hastings step that tries to 'shift' individual features instead of switching them 'on' and 'off'.

The simulation study in section B.4 shows that all methods improve

[1]We use the term *Metropolised* Gibbs sampler to distinguish a sampler that uses a Metropolis-Hastings step to sample from the conditional distribution required in the Gibbs sampler from the standard Metropolis-Hastings algorithm that directly samples from the distribution of interest.

the ratio of accepted state changes to proposed changes. Unfortunately, for most methods, the additional computational cost leads to a decrease in performance when the computation time is taken into account. The only exception is the problem specific proposal, which was found to offer some advantages for the shift-invariant sparse coding model.

B.1 Improving Efficiency by Metropolisation

In this section we use ideas introduced in [65] to improve the efficiency of Gibbs sampling methods for sparse discrete distributions. For sparse discrete distributions, the standard Gibbs sampling approach used in this thesis has to evaluate the probability of a change for each of the coefficients u_n. For very sparse distributions as used here, these u_n are zero with high probability and set to one with a very small probability in each step of the Gibbs sampler. This leads to a high number of computations that have to be performed for each change in the sampler state. In [65] it was suggested that each Gibbs kernel could be replaced with a Metropolis-Hastings transition kernel to sample from the conditional probability. In [65] the prior distribution was used as a proposal distribution for the Metropolised Gibbs sampler. Such a method decreases the amount of computation required for each sample, but it also reduces the speed of the mixing of the chain. We therefore propose a data dependent proposal distribution of the same form as used in chapter 5. For each data point $\mathbf{x}$ the Euclidean distance between this point and all columns of $\mathbf{A}$ can be calculated. The proposal density can then be constructed as a function of these distances, giving a higher probability to include columns of $\mathbf{A}$ close to the data. We use the same proposal distribution as in equation (5.5), i.e.

$$\alpha(u_n = 1|\mathbf{x}) = p(u_n = 1) * f_n(\mathbf{x}), \tag{B.1}$$

with

$$f_n(\mathbf{x}) = 2 * \frac{< \mathbf{a}_n, \mathbf{x} >^{0.4}}{\max_{\hat{n}} < \mathbf{a}_{\hat{n}}, \mathbf{x} >},$$

where $< \cdot >$ is the inner product, $\mathbf{a}_n$ is the n^{th} column of $\mathbf{A}$ and the exponent of 0.4 is a variable chosen for good average performance in the experiments to be reported below. Note that, for the above formulation, we have the condition that $p(u_n = 1) < 0.5$ as can be assumed for many sparse problems. Obviously $f_n(\mathbf{x})$ can be replaced by any function chosen from prior beliefs or experience.

We assume that the distribution of interest generally has a higher probability of including coefficients and therefore columns of $\mathbf{A}$ for which these columns are highly related to the data. The proposal distribution introduced here has a higher probability of selecting such columns such that the acceptance probability is then generally higher, which could be shown experimentally in [65]. Why the replacement of an independent sampling method by a Markov chain with correlated draws can be of advantage for discrete distributions is not immediately obvious. However, in [83] it was shown theoretically that this is the case for certain chains. These results show that a Markov chain sampler can be an advantage if the transition kernel of this method has larger off-diagonal elements than those of the Gibbs sampler.

B.2 Bridged Transitions

The slow mixing of the sampler developed so far can be attributed to the fact that in each step only one coefficient s_n is changed. The probability of changing many s_n in series to arrive at a different mode of the distribution is very low. Instead of employing this Gibbs strategy it would be possible to use a Metropolis-Hastings algorithm that proposes a new coefficient $\mathbf{s}$ and associated indicator variables $\mathbf{u}$ that depend less on the previous coefficients. However, the proposal distribution must be chosen with care to minimise the rejection of this new sample. We develop a proposal distribution based on tempering ideas and the Gibbs sampler.

The proposal density samples a new state as follows. We use two Gibbs cycles and in each individual step during the first Gibbs cycle we vary the distribution we sample from to get more dispersed, i.e. we start with a Gibbs kernel sampling from the actual distribution of interest as in the normal Gibbs step but slowly change this distribution and finally draw the last few coefficients s_n and indicator variables u_n from the proposal distribution in equation (B.1). During the second Gibbs cycle we use the inverse order of conditional distributions from the first cycle and therefore return slowly to the distribution of interest. This method does not sample from the required distribution directly, but a Metropolis-Hastings acceptance step can be developed to ensure the correct stationary distribution. This method is an extension of the tempered transition kernel developed in [99] for the use with individual Gibbs kernels.

To formally define this algorithm we need some additional notation. We write the posterior distribution as $p_o(\mathbf{s}, \mathbf{u}) := p(\mathbf{s}, \mathbf{u}|\mathbf{x}, \theta)$. We further use the abbreviation $q_r(u^r_{n_r}) = \left(p(u^r_{n_r}|\mathbf{s}^r_{\hat{n}\neq n_r}, \mathbf{u}^r_{\hat{n}\neq n_r}, \mathbf{x}, \theta)\right)^{\tau_r} \left(p(u^r_{n_r})\right)^{1-\tau_r}$ and write $\rho_r(\mathbf{s}, \mathbf{u}) := p(s^r_{n_r}|s^{r-1}_{j\neq n_r}, \mathbf{u}, \mathbf{x}, \theta)q_r(u^r_{n_r})$ for the marginal tempered distributions (we again use $\mathbf{s}_{\hat{n}\neq 1}$ to denote the vector of coefficients $\mathbf{s}$ without the first element). Note that during tempering we only change the discrete distribution for u_n. We further use the notation $\{\hat{\mathbf{s}}^r, \hat{\mathbf{u}}^r\}$ to denote the sequence of samples drawn during the first cycle of Gibbs steps while we use the notation $\{\check{\mathbf{s}}^r, \check{\mathbf{u}}^r\}$ to denote the samples drawn during the second cycle. We use $\{\hat{\mathbf{s}}^0, \hat{\mathbf{u}}^0\}$ to denote the last sample drawn in the Markov chain, i.e. the current state, and $\{\check{\mathbf{s}}^0, \check{\mathbf{u}}^0\}$ to denote the new proposed sample. We also introduce the notation $\hat{K}^r_{n_r}$ and $\check{K}^r_{n_r}$ to denote the individual Metropolised Gibbs kernels drawing samples s_{n_r} and u_{n_r} from the conditional distributions $\rho_r(\mathbf{s}, \mathbf{u})$ and $\rho_{r-1}(\mathbf{s}, \mathbf{u})$ respectively[2]. We further use the abbreviations

$$\hat{c}^r = \frac{\rho_r(\hat{\mathbf{s}}^r \hat{\mathbf{u}}^r)}{\rho_r(\hat{\mathbf{s}}^{r-1} \hat{\mathbf{u}}^{r-1})}$$

and

$$\check{c}^r = \frac{\rho_r(\check{\mathbf{s}}^{r-1} \check{\mathbf{u}}^{r-1})}{\rho_r(\check{\mathbf{s}}^r \check{\mathbf{u}}^r)}.$$

Before formally defining the algorithm, it is instructive to consider the sequence of Gibbs steps required to propose a new sample. This is best

[2]Note that kernel $\hat{K}^r_{n_r}$ maps sample $\{\hat{\mathbf{s}}^{r-1}, \hat{\mathbf{u}}^{r-1}\}$ to $\{\hat{\mathbf{s}}^r, \hat{\mathbf{u}}^r\}$ while $\check{K}^r_{n_r}$ maps sample $\{\check{\mathbf{s}}^r, \check{\mathbf{u}}^r\}$ to $\{\check{\mathbf{s}}^{r-1}, \check{\mathbf{u}}^{r-1}\}$, i.e. the first kernel has a stationary distribution of $\rho_r(\mathbf{s}, \mathbf{u})$ while the second kernel has a stationary distribution of $\rho_{r-1}(\mathbf{s}, \mathbf{u})$. This is a notational convention introduced to preserve the symmetries in the derivation.

Algorithm 1 Tempered Transition Sampler

- For $r = 1$ to N and a random order of n_r: Draw sample $\hat{u}^r_{n_r}$ conditional on $\hat{s}^{r-1}_{n_r}$ and $\hat{s}^r_{n_r}$ conditional on $\hat{u}^r_{n_r}$ using a Gibbs kernel with stationary distribution $\rho_r(\mathbf{s}, \mathbf{u})$.
- For $r = N$ to 1, i.e. counting down: Draw sample $\check{u}^{r-1}_{n_r}$ from $\check{s}^r_{n_r}$ and $\check{s}^{r-1}_{n_r}$ from $\check{u}^{r-1}_{n_r}$ using a Gibbs kernel with stationary distribution $\rho_{r-1}(\mathbf{s}, \mathbf{u})$.
- Accept $\{\check{\mathbf{u}}^0, \check{\mathbf{s}}^0\}$ with probability

$$\alpha = \min\left\{1, \frac{p(\check{\mathbf{s}}^0, \check{\mathbf{u}}^0|\mathbf{x}, \theta)}{p(\hat{\mathbf{s}}^0, \hat{\mathbf{u}}^0|\mathbf{x}, \theta)} \frac{1}{\prod_{r=1}^N \hat{c}^r \prod_{r=1}^N \check{c}^r}\right\} \tag{B.2}$$

  or repeat $\{\hat{\mathbf{s}}^0, \hat{\mathbf{u}}^0\}$ otherwise.

done using the following representation.

$$\{\hat{\mathbf{s}}^0, \hat{\mathbf{u}}^0\} \xrightarrow{\hat{K}^1_{n_1}} \{\hat{\mathbf{s}}^1, \hat{\mathbf{u}}^1\} \xrightarrow{\hat{K}^2_{n_2}} \ldots \xrightarrow{\hat{K}^N_{n_N}} \{\hat{\mathbf{s}}^N, \hat{\mathbf{u}}^N\} =$$
$$\{\check{\mathbf{s}}^N, \check{\mathbf{u}}^N\} \xrightarrow{\check{K}^N_{n_N}} \{\check{\mathbf{s}}^{N-1}, \check{\mathbf{u}}^{N-1}\} \xrightarrow{\check{K}^{N-1}_{n_{N-1}}} \ldots \xrightarrow{\check{K}^1_{n_1}} \{\check{\mathbf{s}}^0, \check{\mathbf{u}}^0\}$$

Each Gibbs kernel $\hat{K}^r_{n_r}$ or $\check{K}^r_{n_r}$ only changes the n^{th}_r coefficient, i.e. the difference between two adjacent samples, say $\{\hat{\mathbf{s}}^r, \hat{\mathbf{u}}^r\}$ and $\{\hat{\mathbf{s}}^{r+1}, \hat{\mathbf{u}}^{r+1}\}$ is in the coefficients s_{n_r} and u_{n_r}.

The tempered transition sampler is formally defined in algorithm 1.

The ratios $\hat{c}^r$ and $\check{c}^r$ are evaluated in each step of the Gibbs sampler and, in order to calculate the acceptance probability in algorithm 1, these values do not have to be re-evaluated. Only the ratios of the conditional distribution $p(\mathbf{s}|\mathbf{u})$ and the conditionals $p_0(s_1|\mathbf{s}_{\hat{n}\neq 1}, \mathbf{u})$ need to be evaluated together with the product of these values and the ratio $p_o(\check{\mathbf{s}})/p_o(\hat{\mathbf{s}})$. Note that here we use two Metropolised Gibbs sampler cycles for each new sample. This is not mandatory and any sequence of Gibbs kernels can be used under the condition that, during the second part of the tempering procedure, the reversed order is used as in the first part. Furthermore, different strategies to select the temperature are possible. We found that a sigmoidal change of the tempering variable gave good results and used:

$$\hat{\tau}_r = \check{\tau}_r = \frac{1}{1 + e^{-(i-\frac{N}{2})\frac{10}{N}}}.$$

Whether the Markov chain produced by drawing samples as described in algorithm 1 satisfies the detailed balance condition and therefore leads to the required stationary distribution is not directly evident. We can, however, use a similar approach to the one in [99] to prove the following theorem.

Theorem B.2.1. *The Markov chain produced by drawing samples as in algorithm 1 has a stationary distribution $p_o(\mathbf{s},\mathbf{u})$.*

Proof. It is sufficient to show that

$$p_o(\hat{\mathbf{s}},\hat{\mathbf{u}})\prod_{r=1}^{N}\hat{K}^r_{N_r}\prod_{r=N}^{1}\check{K}^r_{N_r}\alpha(\{\mathbf{s},\mathbf{u}\}) = p_o(\check{\mathbf{s}},\check{\mathbf{u}})\prod_{r=1}^{N}\check{K}^r_{N_r}\prod_{r=N}^{1}\hat{K}^r_{N_r}\alpha(\{\mathbf{s}',\mathbf{u}'\}) \tag{B.3}$$

holds true for one sequence of samples $\{\mathbf{s},\mathbf{u}\}$. Here we use the notation $\{\mathbf{s},\mathbf{u}\}$ to denote the sequence of samples drawn from the proposed method, i.e. $\{\mathbf{s},\mathbf{u}\} = \hat{\mathbf{u}}^1,\hat{\mathbf{s}}^1,\cdots,\hat{\mathbf{s}}^N,\check{\mathbf{u}}^N,\cdots\check{\mathbf{s}}^1$ and $\{\mathbf{s}',\mathbf{u}'\}$ is the inverse of this sequence, i.e. $\{\mathbf{s}',\mathbf{u}'\} = \check{\mathbf{s}}^1,\check{\mathbf{u}}^1,\cdots,\check{\mathbf{s}}^N,\hat{\mathbf{u}}^N,\cdots\hat{\mathbf{u}}^1$. For a single Gibbs kernel we know that the reversibility condition [115]

$$\rho_r(\mathbf{s}^r,\mathbf{u}^r)\hat{K}^r_{n_r}(\mathbf{s}^r,\mathbf{u}^r,\mathbf{s}^{r+1},\mathbf{u}^{r+1}) = \check{K}^r_{n_r}(\mathbf{s}^{r+1},\mathbf{u}^{r+1},\mathbf{s}^r,\mathbf{u}^r)\rho_n(\mathbf{s}^{r+1},\mathbf{u}^{r+1}) \tag{B.4}$$

holds. Using this we write

$$\begin{aligned}
&\left[\prod_{r=1}^{N}\hat{K}^r_{n_r}(\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1},\hat{\mathbf{s}}^r,\hat{\mathbf{u}}^r)\right]\cdot\\
&\left[\prod_{r=1}^{N}\check{K}^r_{n_r}(\check{\mathbf{s}}^r,\check{\mathbf{u}}^r,\check{\mathbf{s}}^{r-1},\check{\mathbf{u}}^{r-1})\right]\\
=\;&\left[\prod_{r=1}^{N}\frac{\rho_r(\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1})}{\rho_r(\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1})}\hat{K}^r_{n_r}(\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1},\hat{\mathbf{s}}^r,\hat{\mathbf{u}}^r)\right]\cdot\\
&\left[\prod_{r=1}^{N}\frac{\rho_r(\check{\mathbf{s}}^r,\check{\mathbf{u}}^r)}{\rho_r(\check{\mathbf{s}}^r,\check{\mathbf{u}}^r)}\check{K}^r_{n_r}(\check{\mathbf{s}}^r,\check{\mathbf{u}}^r,\check{\mathbf{s}}^{r-1},\check{\mathbf{u}}^{r-1})\right]\\
=\;&\left[\prod_{r=1}^{N}\frac{\rho_r(\hat{\mathbf{s}}^r,\hat{\mathbf{u}}^r)}{\rho_r(\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1})}\check{K}^r_{n_r}(\hat{\mathbf{s}}^r,\hat{\mathbf{u}}^r,\hat{\mathbf{s}}^{r-1},\hat{\mathbf{u}}^{r-1})\right]\cdot\\
&\left[\prod_{r=N}^{1}\frac{\rho_r(\check{\mathbf{s}}^{r-1},\check{\mathbf{u}}^{r-1})}{\rho_r(\check{\mathbf{s}}^r,\check{\mathbf{u}}^r)}\hat{K}^r_{n_r}(\check{\mathbf{s}}^{r-1},\check{\mathbf{u}}^{r-1},\check{\mathbf{s}}^r,\check{\mathbf{u}}^r)\right]
\end{aligned}$$

$$= \left[\prod_{r=1}^{N} \check{K}^{r}_{n_r}(\hat{\mathbf{s}}^{r}, \hat{\mathbf{u}}^{r}, \hat{\mathbf{s}}^{r-1}, \hat{\mathbf{u}}^{r-1})\right] \left[\prod_{r=N}^{1} \hat{K}^{r}_{n_r}(\check{\mathbf{s}}^{r-1}, \check{\mathbf{u}}^{r-1}, \check{\mathbf{s}}^{r}, \check{\mathbf{u}}^{r})\right]$$
$$\prod_{r=1}^{N} \hat{c}^{r} \prod_{r=N}^{1} \check{c}^{r}$$

where the second equality follows from equation (B.4). Substituting these results into equation equation (B.3) we get

$$\frac{p(\hat{\mathbf{s}}^0, \hat{\mathbf{u}}^0|\mathbf{x}, \theta)}{p(\check{\mathbf{s}}^0, \check{\mathbf{u}}^0|\mathbf{x}, \theta)} \prod_{n=1}^{N} \hat{c}^{r} \prod_{n=N}^{1} \check{c}^{r} \alpha(\{\mathbf{s}, \mathbf{u}\}) = \alpha(\{\mathbf{s}', \mathbf{u}'\})$$

Note that $\hat{\mathbf{s}}^N = \check{\mathbf{s}}^N$ and $\hat{\mathbf{u}}^N = \check{\mathbf{u}}^N$. It can now be easily seen that the above equality holds for the α as defined in equation (B.2), completing the proof.

□

The other important property of a Markov chain is stated in the next theorem.

Theorem B.2.2. *The Markov chain produced by drawing samples as in algorithm 1 is irreducible.*

Proof. If the support of the distribution of each non-zero coefficient is independent of the other coefficients, then it remains to be shown that there is a path with non-zero probability from any point in the discrete space of the indicator variables to any other point in this space. As we randomly select the order of the Gibbs steps this leads to the condition that switching any indicator variable from any possible state has a non-zero probability. This can be guaranteed for the sampler in algorithm 1. □

B.3 Problem Specific Proposals

When using a Markov chain sampler for the sparse coding problem we need to ensure that the chain can easily jump from mode to mode. With the simple Gibbs strategy used in this thesis such moves are not very common as discussed in chapter 6. This is a general problem with many sparse multi-modal distributions and is not restricted to the model used here. However, the structure of the shift-invariant sparse coding model can be exploited. If we assume that the features in the shift-invariant sparse coding model have a harmonic structure, i.e they are nearly periodic over the length of the feature, then a feature is similar to itself when shifted by its period. Features at periodic shifts also offer a similar reduction in reconstruction error and have therefore a similar conditional probability. We introduce a second possible kernel, which can randomly replace the Gibbs kernel leading to a hybrid sampler [97].

For each indicator variable we propose a new vector of indicator variables $\mathbf{u}$ with $u_n = 0$ and $u_{\overrightarrow{n}} = 1$ where $\overrightarrow{n}$ is an indicator of a shifted version of the same feature, i.e. we set one indicator variable that has been one to zero and set an indicator variable that has previously been zero to one. If we ensure that the changed indicator variables are of the same feature but at different shifts, then we effectively 'move' a feature in the reconstruction. Different strategies can be employed to select the feature to be changed based on the first feature selected. The simplest strategy gives equal probability for all indicator variables to be changed. A more sophisticated method is to use a similar strategy as discussed for the Metropolised Gibbs sampler and to select an indicator variable with a probability proportional to the correlation of the feature at that shift with the signal. In this case the proposal distribution is:

$$q(u_n = 0, u_{\overrightarrow{n}} = 1) = q_n(u_{\overrightarrow{n}} = 1|\mathbf{x}),$$

where

$$\sum_{\overrightarrow{n} \in \mathcal{J}} q_n(u_{\overrightarrow{n}} = 1|\mathbf{x}) = 1,$$

with $\mathcal{J}$ being the set of all shifts of the current features u_n being zero. The Metropolis-Hastings acceptance probability for this move is then

$$\min\left\{1, \frac{p(u_{\overrightarrow{n}} = 1, u_n = 0)p(\mathbf{x}|\mathbf{s}_{\hat{n}\neq n,\overrightarrow{n}}, u_n = 0, u_{\overrightarrow{n}} = 1, \theta)q_n(u_n)}{p(u_n = 1, u_{\overrightarrow{n}} = 0)p(\mathbf{x}|\mathbf{s}_{\hat{n}\neq n,\overrightarrow{n}}, u_n = 1, u_{\overrightarrow{n}} = 0, \theta)q_{\overrightarrow{n}}(u_{\overrightarrow{n}})}\right\}.$$

Here we use $\mathbf{s}_{\hat{n}\neq n,\overrightarrow{n}}$ to denote the coefficients without the n^{th} and $\overrightarrow{n}^{th}$ elements.

This kernel ‘moves’ a feature, i.e it switches a feature ‘off’ and switches the same feature ‘on’ at a shifted position. This kernel does not lead to an irreducible Markov chain on its own as the number of non-zero coefficients does not change. However, if used in conjunction with any Gibbs sampling strategy then irreducibility follows from the irreducibility of the Gibbs sampler.

B.4 Performance Analysis

B.4.1 Measures of Efficiency

To compare different Markov chain sampling strategies, measures of the performance of different aspects of the chain need to be employed. Unfortunately, many aspects of Markov chains are hard to monitor and generic measures are often missing. A statistical measure of efficiency is introduced here, which can be used to monitor the mixing of a Markov chain and which gives an indication of the average performance of a Markov chain Monte Carlo method.

Efficiency is measured based on estimates of the computation time, as well as the statistic efficiency measured by the reduction in variance of an estimate. Computation time is measured for a particular implementation of the algorithm in the Matlab computational environment[3]. Statistical efficiency is measured in samples required to achieve a certain variance of an estimate. For the variance of a mean estimate $\overline{g(\{x\})}$ of functions of J i.i.d. samples x_j we have the textbook result of the form:

$$\sigma_0^2 = var(\overline{g(\{x\})}) = \frac{1}{\sqrt{J}} \sum_j var(g(x_j)),$$

whilst for correlated samples we have [4]:

$$var(\overline{g(\{x\})}) = \frac{1}{J} \lim_{I \to \inf} \sum_{r=-I}^{I} \left(1 - \frac{|r|}{I}\right) cov\{g(x_j), g(x_{j+r})\},$$

[3]We use such a measure instead of flops, as it also includes the time required for memory management and auxiliary processes. This measure is dependent on the computer, the programming language as well as the particular implementation. Nevertheless, it is an adequate measure to compare different strategies.

where *var* refers to the variance and *cov* denotes covariance. We estimate the above variance as in [65] using:

$$\hat{var}(\overline{g(\{x\})}) = \frac{\hat{\sigma}_0^2}{J}(1 + 2\sum_{r=1}^{I}(1 - \frac{r}{I})\hat{\rho}_r),$$

with the estimates:

$$\hat{\sigma}_0^2 = \frac{1}{J}\sum_{j=1}^{J} g(x_j)g(x_j)$$

and

$$\hat{\rho}_r = \frac{1}{J}\sum_{j=1}^{J-j} \frac{g(x_j)g(x_{j+r})}{\hat{\sigma}_0^2}$$

and the assumptions that the x_j are zero mean and that $\hat{\rho}_r$ is negligible for $j > I$. The term $2\sum_{r=1}^{I}(1 - \frac{r}{I})\hat{\rho}_r$ is the number of additional samples required from a sequence of dependent samples to achieve the same variance of the mean as a sequence of i.i.d samples. This term is known as the inefficiency factor or autocorrelation time. We can multiply this term by the time the algorithm takes to calculate each sample to get a measure of the computational efficiency of the proposed method.

B.4.2 Experimental Evaluation

We evaluated the efficiency of four different approaches: The standard Gibbs sampler without any modifications (Method 1), the Metropolised version of the Gibbs sampler (Method 2), the hybrid sampler (in which each sample is generated either from method 2 with probability of 0.75 or otherwise generated using the approach of subsection B.3) (Method 3) and the bridged transition method (Method 4). In these experiments we used a toy problem generated from five different features and all their shifts. The length of the features was 32 samples. We generated 20 independent chains of length 100 000 samples, with each chain using a different realisation of the observation vector **x**. All parameters were assumed to be known. For each approach the autocorrelation time was calculated using only the last 50 000 samples to ensure convergence to the stationary distribution. The results are shown in the top panel of figure B.1. The second panel shows the number of changes of the discrete state each second the sampler is

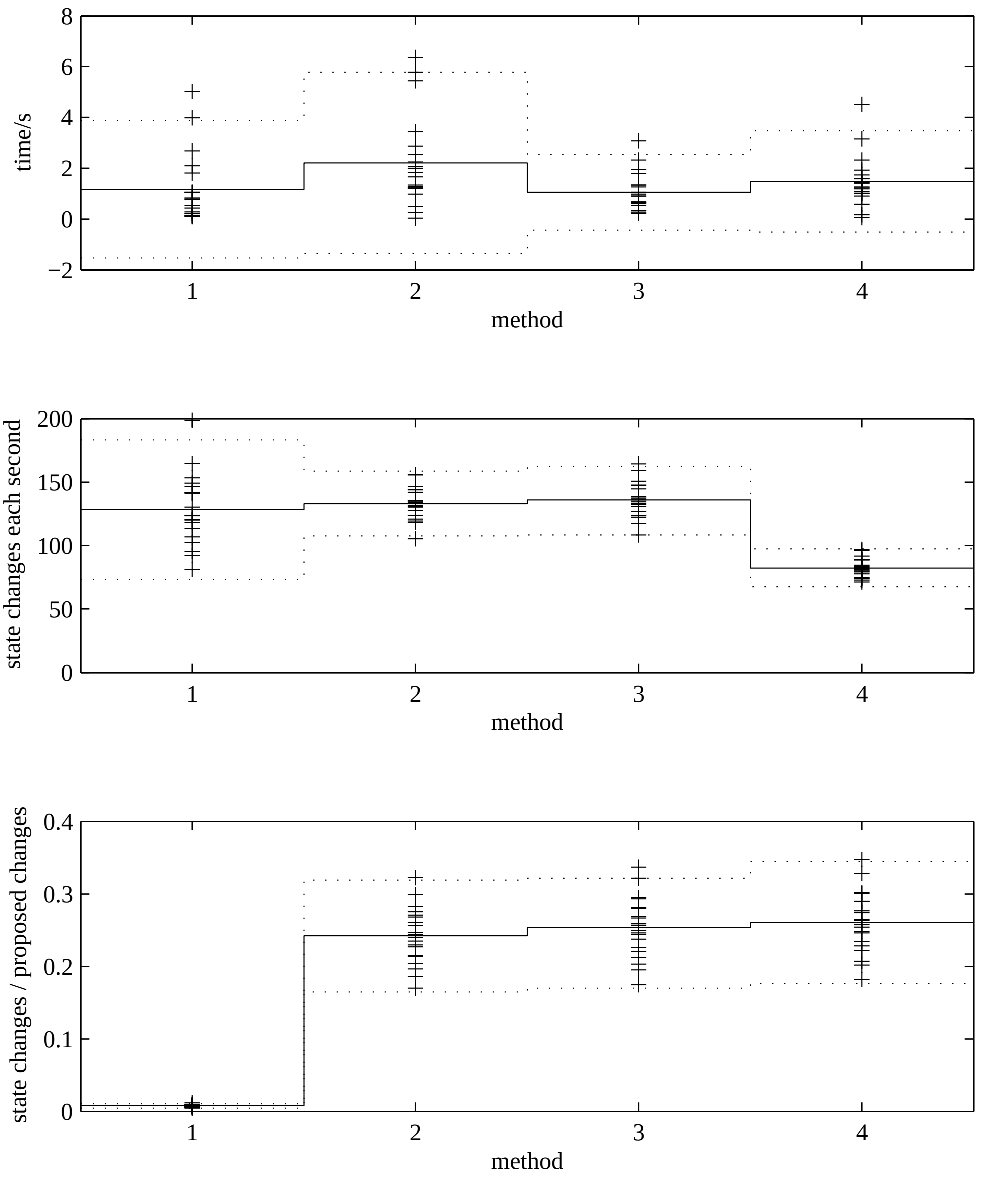

Figure B.1: Comparison of the efficiency of the algorithms proposed measured in seconds required to calculate the number of samples of the chain needed to reduce the variance of an estimate by the same amount as an i.i.d sample would (top). The number of state changes in each second is shown in the middle and the ratio of accepted changes to the number of proposed changes is shown on the bottom. The average (solid line) is shown with twice the standard deviation of the sample statistic (dashed line). The crosses are the sample points.

run. The last panel shows the ratio of accepted changes to the number of proposed changes.

It is clear that methods 2 and 4 perform worse than the standard Gibbs sampler. Method 3 seems to offer better performance than method 2, which means that the introduction of the problem specific proposal did improve performance. It can be seen that the variance over different data points is much larger than the improvements achieved.

The middle panel indicates that methods two and three both increase the number of state changes per second, however, this graph does not take the correlation between samples into account. Method four did not improve the number of state changes per second but did significantly reduce the correlation between samples. The poor performance of this method is therefore mainly due to the large increase in computations required to draw each sample. The last panel shows that the rate of accepted changes to proposed changes increases with each method, such that many more changes are accepted with method 4 than with method 1.